

Connect Bridge Management Studio A web-based application used to manage and test Connect Bridge servers

MASTER PROJECT

Vítor Miguel Costa Baptista
MASTER IN COMPUTER ENGINEERING



UNIVERSIDADE da MADEIRA

A Nossa Universidade

www.uma.pt

October | 2018

Connect Bridge Management Studio

A web-based application used to manage
and test Connect Bridge servers

MASTER PROJECT

Vítor Miguel Costa Baptista

MASTER IN COMPUTER ENGINEERING

SUPERVISOR

Filipe Magno de Gouveia Quintal

CO-SUPERVISOR

Jefferson Kenji Takahashi

Agradecimentos

A realização deste projeto de mestrado contou com o apoio de diversas pessoas e entidades às quais não poderia deixar de agradecer.

Agradeço a todos os docentes que me acompanharam ao longo desta caminhada, por todo o conhecimento que me transmitiram. Em particular, agradeço ao Professor Doutor Filipe Quintal pela orientação deste projeto, pela sua disponibilidade, paciência, amizade e por todas as opiniões e conselhos que transmitiu.

Agradeço à empresa Connecting Software pela oportunidade que me deu de desenvolver este projeto e por todo o suporte que me deu ao longo destes últimos meses. Agradeço de uma forma especial ao meu supervisor Jefferson Takahashi, por todo o apoio técnico que me deu.

Agradeço ao Fábio e à Filipa que comigo partilharam esta experiência de trabalhar junto da Connecting Software. Ao longo do último ano, passamos ótimos momentos que ficarão sempre na nossa memória.

Agradeço de forma especial ao Fábio, que me acompanhou ao longo de todo o meu percurso académico. A ele agradeço a sua paciência, presença e amizade. Sem ele tudo teria sido mais complicado.

Finalmente, agradeço aos meus pais por todo o seu apoio, incentivo e amizade e por sempre terem feito o máximo que podiam para me dar uma formação de qualidade, pois sem eles com certeza não estaria neste momento a terminar este Mestrado.

Abstract

Connecting Software is a software development company focused on the development of integration solutions. For this, the company created a platform named Connect Bridge and some products based on this platform which provide an out-of-the-box solution for the common integration issues faced by companies.

The Connect Bridge (CB) is an integration platform mainly based on a server (usually named CB Server) that uses a plugin architecture to allow its users to connect to a variety of systems (e.g., Microsoft Exchange, SharePoint or Dynamics CRM). Each plugin (usually called as a connector), allows the platform to access a different target system.

Connect Bridge platform is managed by two different applications, the Administration Tool and Query Analyzer. These two tools are old-fashioned and only compatible with Windows operating systems, which limits the number of customers that can use this platform.

This project aims to solve these problems by creating a Single Page Web Application called Connect Bridge Management Studio which was developed using Angular and ASP.NET Core.

At the end of this project, Connecting Software has a ready to deploy application that is capable of replacing Query Analyzer and Administration Tool in all their core features. The evaluation of this project was made comparing the existing tools with the CB Management Studio. Also, an evaluation with users was performed using the think-aloud technique and SUS test which presented satisfactory results.

The present document describes the process followed to achieve this solution, focusing on the most critical aspects of its development and the faced challenges.

Keywords

Desktop application replacement

Single Page Application

Cross-platform application

Connect Bridge Platform

Angular

ASP.NET Core

Resumo

A Connecting Software é uma empresa de desenvolvimento de software focada no desenvolvimento de soluções de integração. De modo a atingir este objetivo, a empresa criou uma plataforma chamada Connect Bridge e alguns produtos baseados na mesma que fornecem uma solução *out-of-the-box* para os problemas de integração mais recorrentes em empresas.

O Connect Bridge (CB) é uma plataforma de integração que consiste principalmente em um servidor (geralmente chamado CB Server) que usa uma arquitetura de plugins para permitir que os seus utilizadores se conectem a uma variedade de sistemas (por exemplo, Microsoft Exchange, Microsoft SharePoint ou Dynamics CRM). Cada plugin (geralmente chamado de conector) permite que a plataforma aceda a um sistema diferente.

A plataforma Connect Bridge é gerida através de duas aplicações distintas, denominadas por Administration Tool e Query Analyzer. Estas duas ferramentas são antigas e compatíveis apenas com os sistemas operativos Windows, o que limitava o número de clientes que podiam usar essa plataforma para desenvolver as suas soluções de integração.

Este projeto visa solucionar esses problemas criando uma aplicação Web chamada Connect Bridge Management Studio. Esta é uma *Single Page Application* desenvolvida usando as *frameworks* Angular e ASP.NET Core.

No final deste projeto, a Connecting Software dispõe de uma aplicação pronta a ser lançada capaz de substituir o Query Analyzer e o Administration Tool. A avaliação desta solução foi efetuada a partir da comparação destas ferramentas com o CB Management Studio. Para além disso, efetuou-se uma avaliação com utilizadores utilizando a técnica *think-aloud* onde se obtiveram resultados satisfatórios.

O presente documento descreve o processo seguido para alcançar esta solução, focando-se nos aspetos mais importantes do seu desenvolvimento e desafios enfrentados.

Palavras-chave

Substituição de aplicações desktop

Single Page Application

Aplicações multiplataforma

Connect Bridge Platform

Angular

ASP.NET Core

Table of Contents

Agradecimentos	i
Abstract	iii
Keywords.....	v
Resumo.....	vii
Palavras-chave	ix
Table of Contents.....	xi
List of Tables	xv
List of Figures.....	xvii
List of Appendices.....	xxi
List of Abbreviations and Acronyms.....	xxiii
1 Introduction.....	1
1.1 Context.....	1
1.1.1 <i>Connect Bridge</i>	1
1.1.2 <i>Connect Bridge Server Administration Tool</i>	2
1.1.3 <i>Connect Bridge Query Analyzer</i>	3
1.2 The problem	5
1.3 Goals	5
1.4 Dissertation Structure	6
2 State of the Art	7
2.1 Development of Desktop Cross-Platform Applications.....	8
2.1.1 <i>Mono</i>	8
2.1.2 <i>.NET Core</i>	9
2.1.3 <i>Web Technologies</i>	10
2.2 Development of Database Management Software.....	19
2.3 Software Integration Technologies.....	21
3 System Design.....	23
3.1 Methodology.....	23
3.2 Requirements Gathering.....	24
3.2.1 <i>Use Cases</i>	24

3.2.2	<i>Requirements List</i>	26
3.3	Prototyping.....	32
3.4	Architecture.....	36
3.4.1	<i>Broker Architecture</i>	36
3.4.2	<i>Client-Server Architecture</i>	39
4	Development	41
4.1	Project Planning.....	41
4.2	Development Environment.....	43
4.3	Versioning.....	43
4.4	Back-end.....	44
4.4.1	<i>Project Structure</i>	45
4.5	Front-end.....	48
4.5.1	<i>Project Structure</i>	48
4.5.2	<i>Usage of CSS Pre-processor</i>	51
4.6	Challenges.....	51
4.6.1	<i>Back-end</i>	51
4.6.2	<i>Replication of the Desktop look and feel</i>	52
4.7	Setup Installer.....	55
4.8	Documentation.....	56
5	Evaluation	59
5.1	Comparison between CB Management Studio and its predecessors.....	59
5.1.1	<i>Administration Tool vs. CB Management Studio</i>	59
5.1.2	<i>Query Analyzer vs. CB Management Studio</i>	65
5.2	Evaluation with Users.....	70
5.2.1	<i>Results of Thinking Aloud Technique</i>	73
5.2.2	<i>Results of SUS</i>	84
5.3	Evaluation from Connecting Software Management Team.....	86
5.4	Evaluation Analysis.....	86
6	Conclusions	89
6.1	Problem.....	89
6.2	Research.....	89
6.3	Analysis.....	90
6.4	Development.....	91
6.5	Contributions.....	91

6.6	Evaluation	92
6.7	Future Work	92
6.8	Lessons Learned	93
7	References	95
8	Appendices	99
	Appendix A - Connect Bridge Management Studio Medium-Low Fidelity Prototypes	101
	Appendix B - Think-Aloud Protocol	111
	Appendix C - Survey used after think-aloud	113

List of Tables

Table 2.1 - Comparison between approaches used with Mono for the development of cross-platform applications	9
Table 5.1 - Responses to the question “How easy was to perform the Task 1?”	73
Table 5.2 - Responses to the question “How easy was to perform the Task 2?”	75
Table 5.3 - Responses to the question “How easy was to perform the Task 3?”	77
Table 5.4 - Responses to the question “How easy was to perform the Task 4?”	77
Table 5.5 - Responses to the question “How easy was to perform the Task 5?”	80
Table 5.6 - Responses to the question “How easy was to perform the Task 6?”	81
Table 5.7 - Responses to the question “How easy was to perform the Task 7?”	82
Table 5.8 - Responses to the question “How easy was to perform the Task 8?”	83
Table 5.9 - Responses to the question “How easy was to perform the Task 9?”	84
Table 5.10 - Results of SUS Evaluation	85

List of Figures

Figure 1.1 - Connect Bridge Platform Concept.....	2
Figure 1.2 - Connect Bridge Server Administration Tool	3
Figure 1.3 - Connect Bridge Query Analyzer.....	4
Figure 2.1 - Visual Studio Code (based on Electron)	16
Figure 2.2 - WhatsApp Desktop (based on NW.js)	17
Figure 2.3 - Treeview on phpMyAdmin	20
Figure 2.4 - Treeview on Navicat.....	20
Figure 2.5 - Treeview on SQL Management Studio.....	20
Figure 2.6 - Tabs on Navicat.....	20
Figure 2.7 - Tabs on SQL Management Studio.....	20
Figure 2.8 - Architecture of an application that uses Connect Bridge.....	22
Figure 3.1 - Use Cases for Developer User.....	25
Figure 3.2 - Use Cases for Administrator User	26
Figure 3.3 - Medium-low-fidelity prototype of Users' view	33
Figure 3.4 - First version of the high-fidelity prototype of Users' view.....	34
Figure 3.5 - Existing User Interface of SaaS platform	34
Figure 3.6 - First version of the high-fidelity prototype of Users' view	35
Figure 3.7 - Final version of Users' view.....	36
Figure 3.8 - Initial architecture to support multiple CB Servers.....	37
Figure 3.9 - Architecture considering CB Management Studio as a hybrid desktop application	38
Figure 3.10 - CB Management Studio using Client-Server Architecture	39
Figure 4.1 - Excerpt of the backlog for the first release of CB Management Studio.....	43
Figure 4.2 - Sequence diagram of the action "Update CB User" on the back-end of CB Management Studio	46
Figure 4.3 - Workflow of the repositories that interact with CB Server	47
Figure 4.4 - Sequence Diagram (simplified) for the Use Case "Add CB User"	50
Figure 4.5 - Generated Swagger Documentation Page.....	56
Figure 4.6 - Example of a page generated by Compodocs	57
Figure 4.7 - Connect Bridge Management Studio User Manual	58
Figure 5.1 - Navigation Tree on Administration Tool.....	60
Figure 5.2 - Navigation Bar on CB Management Studio	60
Figure 5.3 - Management of CB Users on Administration Tool (List of CB Users).....	61

Figure 5.4- Management of CB Users on CB Management Studio (List of CB Users)	61
Figure 5.5- Management of CB Users on Administration Tool (Edit CB User).....	62
Figure 5.6- Management of CB Users on CB Management Studio (Edit CB User)	62
Figure 5.7 - List of Licenses on Administration Tool	63
Figure 5.8 - List of Licenses on CB Management Studio.....	63
Figure 5.9 - Logs Configuration on Administration Tool	64
Figure 5.10 - Logs Configuration on CB Management Studio	64
Figure 5.11 - Visualization of logs on CB Management Studio	65
Figure 5.12 - Connection Browser on Query Analyzer	65
Figure 5.13 - Connection Browser on CB Management Studio	66
Figure 5.14 - Connection String generated by Query Analyzer.....	67
Figure 5.15 - Connection Strings generated by CB Management Studio	67
Figure 5.16 - Parameterized Query on Query Analyzer.....	68
Figure 5.17 - Parameterized Query on CB Management Studio	68
Figure 5.18 - Auto generation of queries on Query Analyzer.....	68
Figure 5.19 - Auto generation of queries in CB Management Studio	69
Figure 5.20 - Code suggestions on CB Management Studio query editor	69
Figure 5.21 - Auto-completion of code on CB Management Studio query editor	70
Figure 5.22 - Common error performed by the participants in Task 2, using Administration Tool	74
Figure 5.23 - Inconsistent date presented on Licenses Page	74
Figure 5.24 - Common error performed by the participants in Task 3, using Administration Tool	76
Figure 5.25 - Misleading button on Connectors Page	76
Figure 5.26 - Dual List used to associate accounts with users on CB Administration Tool	78
Figure 5.27 - Dual List used to associate accounts with users on CB Management Studio	79
Figure 5.28 - Add file as a parameter of a query on CB Management Studio.....	82
Figure 5.29 - Add file as a parameter of a query in Query Analyzer	82
Figure 5.30 - Normalization of SUS evaluation [62]	85
Figure 5.31 - Connection Browser before and after User's Evaluation.....	87
Figure 5.32 - New step of add/edit account used to directly associated users and accounts	87

Figure 5.33 - Dual list used to associate user and accounts on Users edit/add window before
and after user's evaluation 88

List of Appendices

Appendix A - Connect Bridge Management Studio Medium-Low Fidelity Prototypes ...	101
Appendix B - Think-Aloud Protocol	111
Appendix C - Survey used after think-aloud	113

List of Abbreviations and Acronyms

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
AT	Administration Tool
CB	Connect Bridge
CEO	Chief Executive Officer
CRM	Customer Relationship Management
CRUD	Create, Read, Update and Delete
CSS	Cascading Style Sheets
DOM	Document Object Model
ECMA	European Computer Manufacturers Association
ERP	Enterprise Resource Planning
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
IT	Information Technologies
JDBC	Java Database Connectivity
JS	JavaScript
JSON	JavaScript Object Notation
MVC	Model-View-Controller
MVP	Minimum Valuable Product
ODBC	Open Database Connectivity
OOP	Object-Oriented Programming
OS	Operating System
QA	Query Analyzer
REST	Representational State Transfer
SPA	Single Page Application
SQL	Structured Query Language
SSL	Secure Socket Layer
SUS	System Usability Scale
TS	TypeScript

UWP	Universal Windows Platform
UX	User Experience
WPF	Windows Presentation Foundation
XSS	Cross-site scripting

1

INTRODUCTION

In this chapter, the project is introduced, starting by presenting the company where this work was developed. After this, the products that are directly related to this project are presented. Then, a small reflection about the existing problems on those applications is made. Finally, the structure of this document is presented.

1.1 Context

Connecting Software is a software development company focused on the development of solutions for the integration between different business services like e-mail, Enterprise Resource Planning (ERP), Customer Relationship Management (CRM) systems or documents storage services. For this, the company created a platform named Connect Bridge and several products based on this platform that provide out-of-the-box solutions for some of the more common integrations requested in the market.

1.1.1 Connect Bridge

The Connect Bridge (CB) is an integration platform mainly based on a server (usually named as CB Server) which allows users to connect to several systems like Microsoft Exchange, Microsoft SharePoint or Dynamics CRM. This server uses a plugin architecture, where each plugin (usually called as a connector) allows the platform to receive and transmit data to a specific target system, using the official Application Programming Interface (API) provided by those target systems.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

The connection of client applications used to develop integration solutions and a CB Server can be done recurring to Webservices, Open Database Connectivity (ODBC) or Java Database Connectivity (JDBC) drivers and afterwards the integration can be controlled using CB Structured Query Language (SQL). With this, we can access, for example, the list of contacts of a user in Microsoft Exchange just doing a simple query like “SELECT * FROM Contacts”. The existence of these drivers combined with the SQL grants the possibility to use any programming language to develop these client applications.

The Figure 1.1 schematizes the concept explained above.

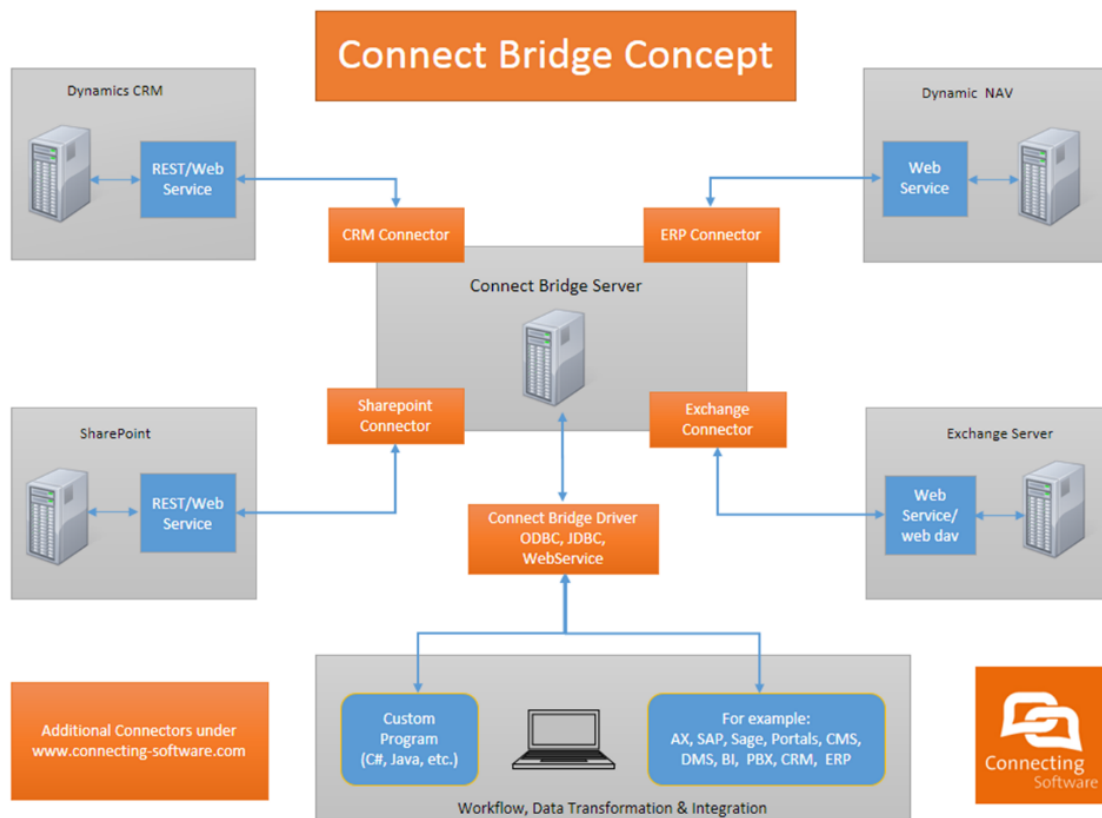


Figure 1.1 - Connect Bridge Platform Concept

1.1.2 Connect Bridge Server Administration Tool

The Connect Bridge Server Administration Tool (AT) is the official tool used to manage and configure a CB Server installed on a local PC or a remote server. In the context of a software development team that uses CB to create an integration solution, this tool is used by the professionals responsible for the management of the projects or system administrators.

The tool provides a Graphical User Interface (GUI) to help the users on these processes. Each action made using this interface is reflected on the server using simple query statements and specific System Stored Procedures.

The communication between AT and CB Server is made using ODBC Driver and the internal CB Administration Connector of the server. This connector is responsible for communicating with the internal database of the system, providing all the mechanisms to interact with it.

Nowadays, the main functionalities provided by Administration Tool are:

- Request and apply licenses for CB Server and connectors
- Activation/deactivation of connectors
- Users and member groups management
- Target systems account management
- Diagnostics related settings

The figure bellow presents the GUI of Connect Bridge Server Administration Tool.

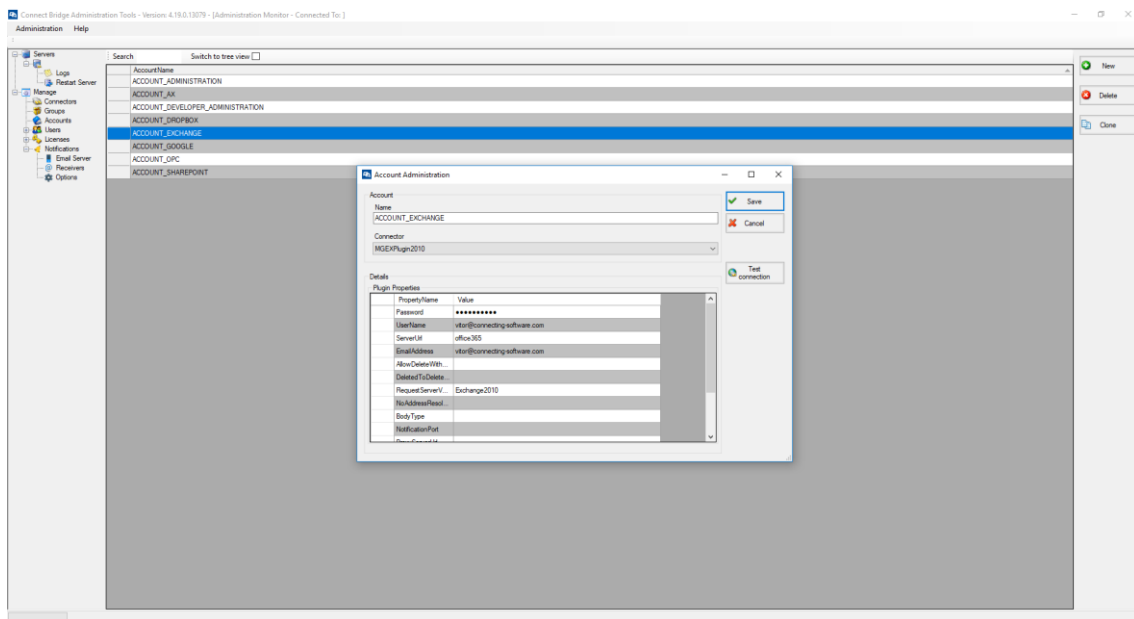


Figure 1.2 - Connect Bridge Server Administration Tool

1.1.3 Connect Bridge Query Analyzer

The Connect Bridge Query Analyzer (QA) is a tool used by the developers responsible for the development of integration solutions. The purpose of this tool is to test SQL statements against all target systems connected to the CB server via CB Connectors before using them in the custom code created during the development of an integration solution.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

For this, QA uses ODBC driver to connect to CB Server and the correspondent connector, executes the SQL statement or Stored Procedure, waits for the response from the target system and displays it.

The core features of this tool are:

- Creation of connections to CB Server using the accounts¹ configured in the AT;
- List the accessible tables of a target system;
- List the table columns and their metadata;
- Automatic generation of scripts for Create, Read, Update and Delete (CRUD) operations on a table;
- Possibility to define and execute an SQL command and evaluate its result;

The figure bellow presents the GUI of Connect Bridge Query Analyzer.

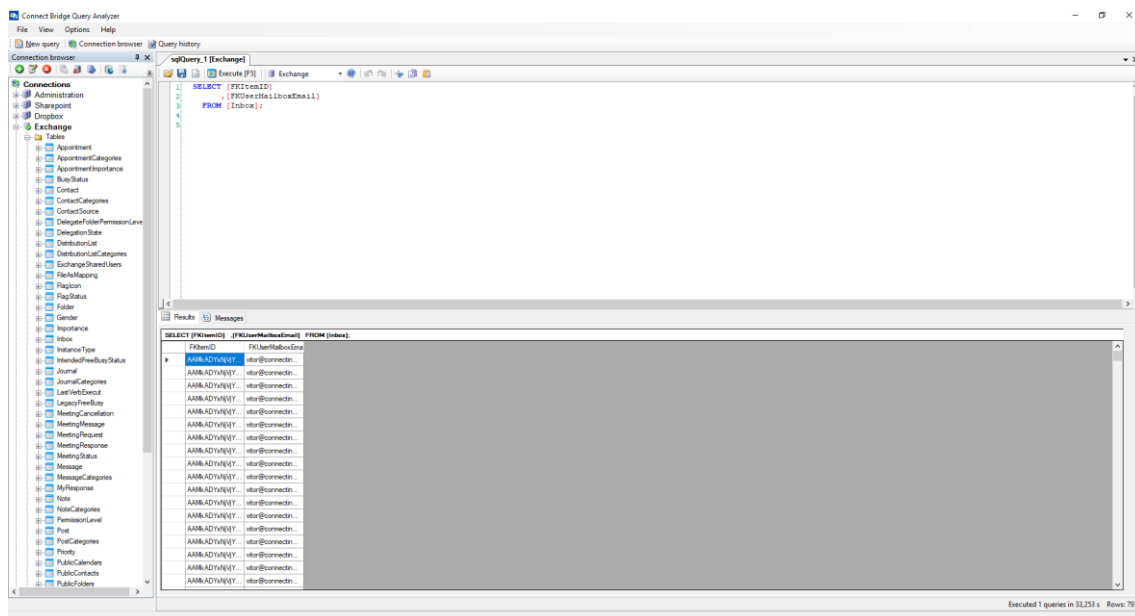


Figure 1.3 - Connect Bridge Query Analyzer

¹ In the context of Connect Bridge, an account is the definition of all the necessary parameters used to establish a connection to a target system. For example, to access a Microsoft Exchange server, it is needed to create a CB account which holds informations like e-mail and password.

1.2 The problem

Nowadays, AT and QA fulfil their purpose, although, they are technologically outdated. These tools were developed as Windows Forms Applications which inheritably created two drawbacks: the creation of old-fashioned desktop applications and the usage of Windows Operating Systems. This last characteristic limits the market of Connecting Software, since it requires that the developers use Windows during the development of integration solutions via Connect Bridge Platform. Also, this technology is quite old and was already replaced by Windows Presentation Foundation (WPF) stack and more recently by Universal Windows Platform (UWP) [1].

Regarding usability, these two tools have another drawback, due to the way they were architected. These tools are usually used together, and their primary goal is to help the users to configure and test Connect Bridge. Having these two tools separated enforces the user to change between the two applications several times which is not ideal. The idea of integrating/joining different systems that are frequently used together is becoming widespread. One example of that is Microsoft Word that is always being improved with functionalities that before were executed on other applications, like the recent feature that allows the user to research about what he/she is writing without leaving the application [2].

In order to solve these problems, it is proposed the creation of a modern Web-based platform for the management and testing of CB Servers, called Connect Bridge Management Studio. This new application would join the existing tools and would have the possibility to be extended in the future with other tools that might be useful for this kind of management and development, creating an Integrated Development Environment (IDE) for Connect Bridge Platform.

1.3 Goals

At the end of this project, it is expected to have the following outcomes:

- A modern and easily extensible web-based platform built under a single page web application that will substitute the existing AT and QA tools;
- Documentation for the application;
- User testing results and feedback of the application;

1.4 Dissertation Structure

This dissertation is structured in the following chapters:

- Chapter 1: Introduction - Introduces the project, its scope, problems, and goals;
- Chapter 2: State of the Art – Overview about similar works, study about web technologies that could be used in this project;
- Chapter 3: System Design – Presents the process, requirements, and architecture of the developed solution;
- Chapter 4: Development – Presents an overview of the software development process done during this project;
- Chapter 5: Evaluation – Analysis of the developed project;
- Chapter 6: Conclusions – Presents the conclusions of this project and presents some of the future works that can be done to improve or add new features in CB Management Studio;

2

STATE OF THE ART

In this chapter an overview about the recent developments in three different fields that are related with this project (the development of desktop cross-platform applications, database management softwares and integration solutions) is presented.

This section will start by defining cross-platform development for desktop applications.

Three different basis for this development are analysed (Mono, .NET Core and Web Technologies).

Specifically, in Web Technologies, the use of Single Page Applications is discussed as an alternative to traditional desktop applications and an analysis of frameworks that promote the development of this paradigm is made. Also, it is presented two platforms that allow the creation of native cross-platform applications (NW.js and Electron) using Web Technologies.

Afterwards, an analysis of the GUI of 3 different Database Management Systems applications is presented to understand the similarities and functionalities that should be part of CB Management Studio.

Finally, an overview of the development of integration solutions is presented, showing how the Connect Bridge Platform becomes a differentiator when compared with the existing alternatives.

2.1 Development of Desktop Cross-Platform Applications

Software development is always subject to changes. The change of platform is only one of the possibilities and one of the most difficult for developers. Changing the operating system (OS) usually makes an application that is designed for a single OS to fail [3].

Cross-platform or multiplatform desktop development is the commonly used term for the development of applications compatible with multiple desktop operating systems such as Windows, Mac, and Linux [4].

This topic is not new, therefore multiple alternatives for the development of this kind of applications were created. In the following subsections, alternatives for this task will be presented.

2.1.1 Mono

Mono is an open-source platform based on .NET Framework, which allows the development of cross-platform applications using C# [5]. In order to use this platform for the development of desktop applications with a GUI, it is necessary to use a toolkit which is the most significant constraint when developing cross-platform applications, since not all toolkits work on all systems and many of them do not look and feel appropriately in all environments. There are some approaches to solve this problem which are resumed in the following table [6]–[8]:

Table 2.1 - Comparison between approaches used with Mono for the development of cross-platform applications

	Description	Advantages	Disadvantages
Traditional Approach	For each OS create a unique view and use specific toolkit for the OS (GTK for Linux, Cocoa for MacOS, WPF for Windows)	<ul style="list-style-type: none"> Looks native All the native resources are available 	<ul style="list-style-type: none"> Need to create a different code for each system
XWT/ Eco.Forms	Uses the same API for all the systems. Uses native widgets used depending on the running system	<ul style="list-style-type: none"> Same API for all platforms Look native 	<ul style="list-style-type: none"> Poor documentation
GTK#	Uses the same API for all the systems but always uses the same widgets (native to Linux)	<ul style="list-style-type: none"> Stable Same API for all platforms 	<ul style="list-style-type: none"> Only look native on Linux Incomplete documentation

2.1.2 .NET Core

.NET Core is a platform developed and maintained by Microsoft and .NET community that supports the development of cross-platform applications using C#, Visual Basic or F# [9].

The development of this new platform started in 2014, due to the requests received to make ASP.NET compatible with Linux, and with the need of creating a new platform compatible with Windows Nano, a new and smaller version of Windows for servers. The first public and stable version, .NET Core 1.0 was released in 2016 [10].

Although this framework looks promising for the development of desktop cross-platform applications, there is no existing stable and robust toolkit for developing a GUI with it.

2.1.3 Web Technologies

Web applications start to become an alternative to desktop applications with the emergence of Asynchronous JavaScript and XML (AJAX), introduced by J. Garret [11]. This technology allowed the creation of web pages that look and perform more like desktop applications [12]. One of the first examples of this was Gmail, that presented a web-based e-mail client when this kind of applications were typical desktop ones (e.g., Microsoft Outlook, Thunderbird) [12].

Nowadays, web applications, in general, offer some advantages when comparing with desktop ones since they are independent of many factors like operating system, virtual machines (like Java Virtual Machine), browser's plugins and do not require the installation of additional software on the user's computer [13].

AJAX revolutionized the web development, eliminating the need for continuously reloading the entire web page on each user interaction. This allows the improvement of the user experience, since it reduces the waiting times, turning website more responsive and with a look and feel similar to desktop applications. This changed the way web applications are developed, originating the so-called Single Page Applications (SPA) [13].

2.1.3.1 Single Page Applications

As referred, the most significant difference and characteristic of an SPA when compared with traditional web pages is the reduced amount of page refreshes, due to the usage of AJAX. Although, this approach also has its disadvantages [14]:

- The application takes more time to load since the client frameworks used to develop these kind of applications require the download of larger JavaScript files;
- Requires JavaScript (JS) to be enabled on the client's browser;
- Is less secure, if not protected, cross-site scripting (XSS) can be used to inject client-side scripts into the web application;
- If some memory leak occurs using JavaScript, even powerful systems can be affected by this performance issue;

Single Page Applications are by default very interactive, which require a significant amount of JavaScript code. Several JS frameworks were created to provide a clean and well-

architected codebase for such tasks. Angular² and React³, are currently the more popular ones according to the website HotFrameworks⁴. This website evaluates the popularity of frameworks based on the number of stars it has on GitHub and the number of questions placed on Stack Overflow related with it [15]. Other client-side JavaScript frameworks include Vue.js⁵, Meteor⁶ or Ember.js⁷.

Considering their popularity and apparent adequacy for this project, the next subsections will present in more detail the React and Angular frameworks.

2.1.3.1.1 *React*

React is not always considered a framework, since when we consider the Model-View-Controller (MVC) pattern, it only focuses on the View module [16].

The development of this project started in 2011 by Facebook. Facebook's development team aim was to make news feed updates to occur simultaneously with updates on the chat. The idea behind the project was to combine XHP (the Facebook markup syntax) with the JS coordinate system. Afterwards, in 2013, React was released as an open-source project and, since then, it became trendy to program user interfaces [17].

Advantages of React [16]–[20]:

- React uses the concept of Virtual Document Object Model (DOM) that controls the existing differences with the Real DOM. This allow only the change of what effectively needs to be changed, avoiding expensive DOM operations and reducing the time needed to update the GUI, improving efficiency;
- React promotes reusability since all components are isolated and reusable;
- The project is open-source and is continuously being developed and improved by many developers and is becoming more and more popular;
- It is effortless to create GUI test cases due to virtual DOM;
- It is the lightest weight framework among the most widely used ones;

² <https://angular.io/>

³ <https://reactjs.org/>

⁴ <https://hotframeworks.com/>

⁵ <https://vuejs.org/>

⁶ <https://www.meteor.com/>

⁷ <https://www.emberjs.com/>

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

Disadvantages of React [16]–[20]:

- React is continually changing and requires the developer to relearn and frequently change the work done;
- The fast development of the library created a lack of proper and actual documentation;
- Since it is not a truly framework, it lacks some functionalities present in many frameworks such as routing mechanism, encapsulation of AJAX capabilities and data layers;
- Difficult to learn for beginners in web development;

2.1.3.1.2 Angular

Angular is a front-end framework created and maintained by Google. The first version of it, called AngularJS, was released in 2009 and solved many problems that JavaScript had before the release of ECMAScript2015⁸. With this release, AngularJS started to have some inadequacies and Google decided to completely rewrite the framework, releasing Angular 2 (now simply referred as Angular) in 2016 [21].

Angular has several built-in features that allow developers to create fully functional applications without using other dependencies. In this new version of the framework, Google decided to implement it using TypeScript instead of using standard JavaScript (TypeScript will be discussed in more detail in the section o) [22].

Advantages of Angular [17], [23]:

- Great performance, due to dependency injection support and Angular Universal that allows the application to be rendered on a server;
- Support for directives (custom HyperText Markup Language (HTML) tag attributes that add/change the behaviour of the tag) and functionalities directly in the HTML;
- Built-in support for unit and end-to-end testing;

⁸ ECMAScript is a standardized language that serves as base for JavaScript

- Angular CLI helps the creation of an application, creating an initial boilerplate⁹ for the project, and for each component, service, directive or pipe created afterwards;
- Component-based architecture increases reusability, readability, and maintainability;
- Google granted long-term support for the project;

Disadvantages of Angular [17], [23]:

- Despite having a big community using and helping on the development of Angular, this community is divided between AngularJS and Angular. Many developers using AngularJS do not consider changing to Angular since they are incompatible;
- Angular is complex, the component-based architecture also presents a disadvantage, since by each component the developer may need to create up to five files, inject dependencies and declare the component lifecycle interfaces;
- The framework has a steep learning curve since new developers familiar to JavaScript need to learn and cover many new topics like modules, dependency injection, components, data binding, services, RxJS (a reactive programming library that is fundamental for Angular), TypeScript and others;
- Angular manipulates directly the DOM which makes it slower and inefficient when compared with other libraries like React;

Since Angular uses TypeScript as its main language, the next section will briefly introduce this technology.

⁹ Standard skeleton/tempalte for a project or part of it, which usually is repated many times with little or no change.

2.1.3.1.3 TypeScript

TypeScript (TS) is a superset of JavaScript that can be transpiled¹⁰ to plain JavaScript that adds to it static typing and class-based object-oriented programming. This new programming language was released in 2012 by Microsoft with the intent to make JavaScript more scalable [25], [26].

Advantages of TypeScript [27], [28]:

- Static typing reduces the number of mistakes, improves code analysis and allows the IDE to provide better hints, assistance or refactoring suggestions;
- Class-based object-oriented programming, with inheritance or interfaces helps the usage by developers used to languages like C# or Java;
- All code that works in JavaScript also works in TS, and it is effortless to learn and to pass from JavaScript to this superset;
- Many features from TS are becoming part of ECMA specification;
- Many tools support TS, providing code analysis and auto-completion;
- Big community and documentation help developers in the usage of TypeScript;

Disadvantages of TypeScript [28]–[30]:

- To get maximum advantage of TS, developers need to create strongly-typed code; this can bring a big effort to convert existing JavaScript code to this;
- TS removes some of the characteristic flexibility of JavaScript;
- Can be complicated for a developer that is not familiar with Object-Oriented Programming (OOP) languages;
- Requires extra steps for transpile the code to JS;
- Fewer examples and support when compared to normal JS;

An example of a code snippet created using TypeScript and its equivalent transpiled JavaScript code is presented bellow:

¹⁰ Transpilation, also known as source-to-source compilation, is a type of compilation that takes a source code written in one programming language as input and generates an equivalent source code in another language as output [24].

TypeScript

```

interface IAnimal {
  name: string;
  sound: string;
  sayHello(): string;
}

abstract class Animal implements IAnimal {
  name: string;
  sound: string;
  constructor(name: string) {
    this.name = name;
  }
  sayHello(): string {
    return this.sound + "! I'm " + this.name + "!";
  }
}

class Duck extends Animal {
  name: string;
  sound: string;
  constructor(name: string) {
    super(name);
    this.sound = "Quack Quack";
  }
}

function greatOnHtmlBody(animal: IAnimal): void {
  let par = document.createElement("p");
  let text = document.createTextNode(animal.sayHello());
  par.appendChild(text);
  document.body.appendChild(par);
}

let duck = new Duck("Donald Duck");

greatOnHtmlBody(duck);

```

JavaScript

```

var __extends = (this && this.__extends) || (function () {
  var extendStatics = Object.setPrototypeOf ||
    ({ __proto__: [] } instanceof Array && function (d, b) { d.__proto__ = b; }) ||
    function (d, b) { for (var p in b) if (b.hasOwnProperty(p)) d[p] = b[p]; };
  return function (d, b) {
    extendStatics(d, b);
    function __() { this.constructor = d; }
    d.prototype = b === null ? Object.create(b) : (__proto__ = b.prototype, new __());
  };
})();

var Animal = /** @class */ (function () {
  function Animal(name) {
    this.name = name;
  }
  Animal.prototype.sayHello = function () {
    return this.sound + "! I'm " + this.name + "!";
  };
  return Animal;
})();

var Duck = /** @class */ (function (_super) {
  __extends(Duck, _super);
  function Duck(name) {
    var _this = _super.call(this, name) || this;
    _this.sound = "Quack Quack";
    return _this;
  }
  return Duck;
})(Animal);

function greatOnHtmlBody(animal) {
  var par = document.createElement("p");
  var text = document.createTextNode(animal.sayHello());
  par.appendChild(text);
  document.body.appendChild(par);
}

var duck = new Duck("Donald Duck");

greatOnHtmlBody(duck);

```

2.1.3.2 Hybrid Desktop Applications

Although SPA became a viable alternative to the desktop application, they did not replace them completely. This occurred for the following reasons [31]:

- There is a lack of vertical applications¹¹ based on web technologies;
- Some applications require a high level of data security, and this cannot be compromised when exposing data on the Internet;
- Some applications need to be running continuously, even without internet connectivity;
- Operating systems are still locally installed;

This led to the development of new technologies that allowed the creation of cross-platform desktop applications based on Web, also referred as hybrid applications. The most used ones are NW.js and Electron. There are already several applications developed using these platforms, Visual Studio Code (based on Electron) and WhatsApp Desktop App (based on NW.js) are examples of this new trend (see Figure 2.1 Figure 2.2).

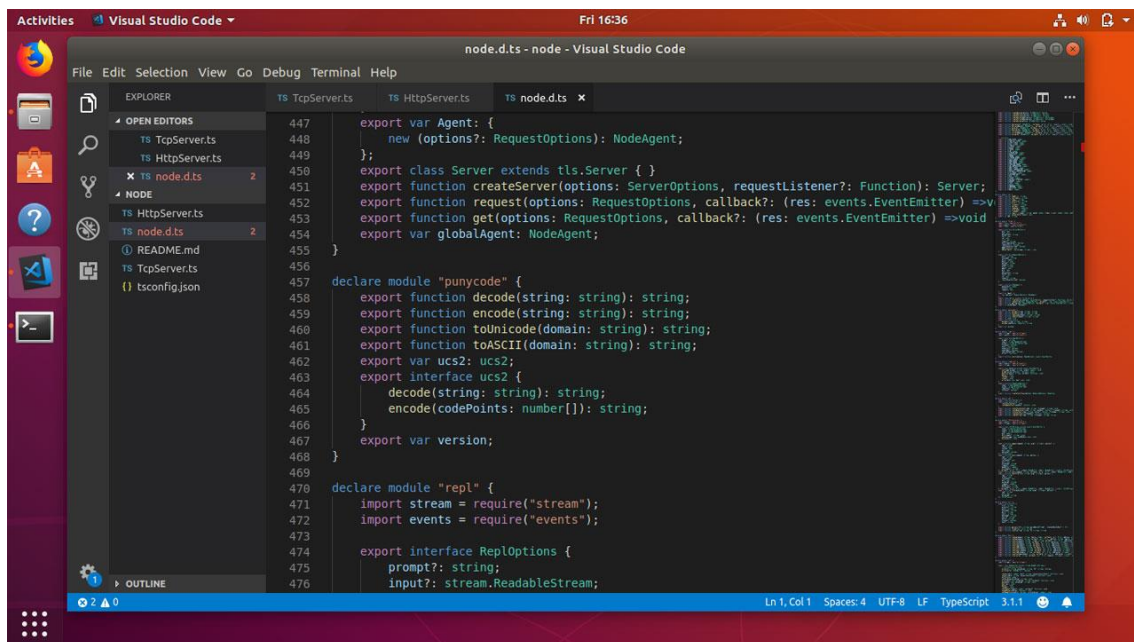


Figure 2.1 - Visual Studio Code (based on Electron)

¹¹ Vertical application is an application designed and built according with the user's requirements. Usually, it is very customized and adapted for the target customer [32].

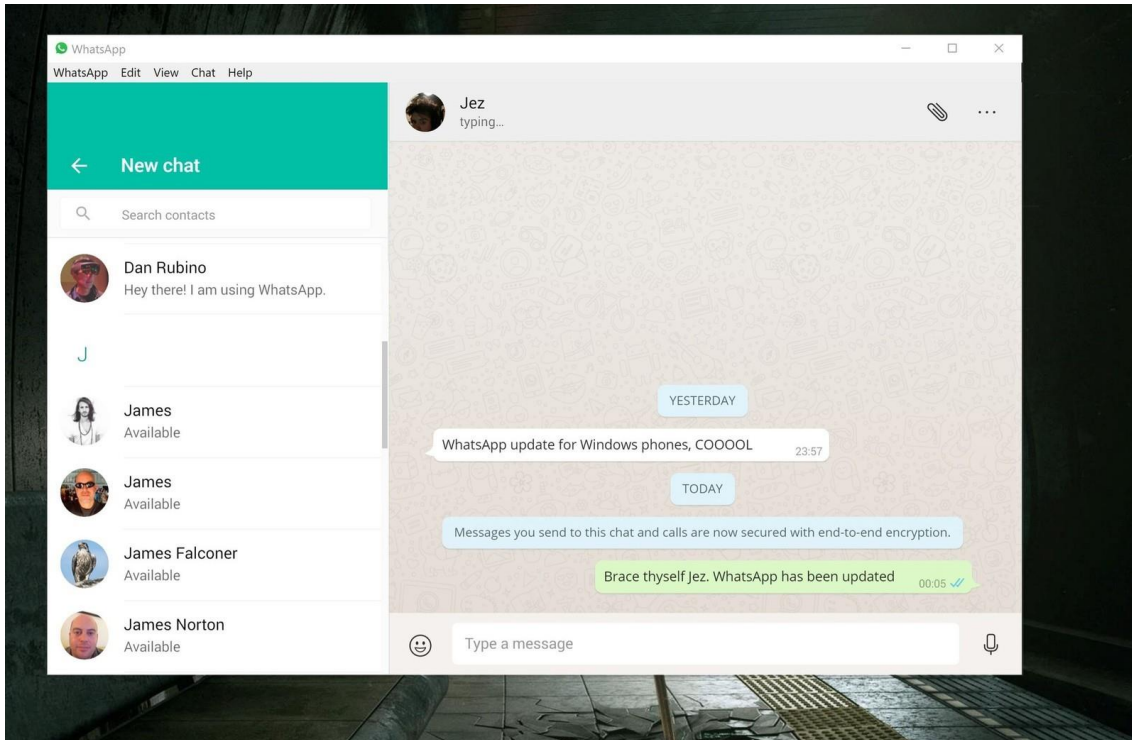


Figure 2.2 - WhatsApp Desktop (based on NW.js)

Both platforms adopt the same concept: use of Chromium for the rendering of the application, combined with Node.js to easily access the operating system resources. Their main difference consists in the way they execute the applications. More information about these two technologies will be presented below.

2.1.3.2.1 NW.js

NW.js is a platform created by Intel in 2011.

An application built using NW.js is more “browser-oriented”. A standard NW.js program loads an HTML file that is able to access the Node.js context. Then, it is possible to use Node.js to interact with the operating system using NW.js APIs [33].

Advantages of NW.js [34]:

- There is a significant number of demo applications and video games that use NW.js;
- It has a great community support;
- Is more mature than Electron;

Disadvantages of NW.js [34]:

- Requires more effort and extra modules to build applications when compared with other solutions;

2.1.3.2.2 Electron

Electron is a project developed by GitHub in 2013, built to be the base of their text editor (Atom).

Electron is more Node.js-oriented than NW.js. While NW.js loads directly an HTML file, Electron starts by creating a Node.js runtime which opens a window and loads, inside it, a web page [33].

It uses two different kinds of processes, the renderer, and the main process. The Main Process is responsible for creating and controlling the lifecycle of the app and is also in charge of communicating with native operating system APIs. The Renderer Process loads web pages to display a graphical user interface. Electron also includes a mechanism to facilitate communication between processes in order to allow the renderer process to communicate with the main process [4].

Advantages of Electron [34], [35]:

- Relies on web standards;
- Allows developers to focus on the core functionality of the application;
- Provides various functionalities for desktop applications like auto-update, crash reports or installer;
- Is in active development (there are updates every month);
- Has good documentation and many examples;

Disadvantages of Electron [34], [35]:

- Is not as mature as NW.js;
- No built-in MVC is provided;
- Users have report memory issues;

From the alternatives presented (Mono and Web Technologies) we decided to follow the second approach, specifically by creating a Single Page Application using the Angular framework. We decided to not encapsulate this application in an hybrid application using NW.js or Electron since it was against the company's CEO plan to create a pure web-based application. For the same reason, and also by advice from the company's supervisor the option of using Mono was also discarded.

2.2 Development of Database Management Software

After analysing the technologies used to develop cross-platform solutions, it is important to analyse applications that are used to manage database systems. Due to the nature of Connect Bridge, target systems are converted into SQL elements (i.e., tables, stored procedures, functions and views), which are later used to simplify the interaction with the system, by simply executing SQL operations over it. Therefore, it is relevant to analyze how commercial systems behave and which types of interaction they provide to execute database management tasks.

In this section, an analysis of software used to manage, and query databases will be presented. This evaluation is very important since it will be later used to refine requirements and help with the implementation of features that are a part of the Connect Bridge Management Studio .

The evaluation presented in this section is based in the 10 Usability Heuristics for User Interface Design by Nielsen. During the evaluation, we focused in two of the aspects that according with the Connecting Software mentor were the most relevant for this application, these were the “Consistency and standards” and the “Recognition rather than recall”, these heuristics are the most relevant, since they can simplify the transition between applications used to manage databases and the CB Management Studio [36].

Based on this, and considering that Microsoft SQL Management Studio, Navicat, and phpMyAdmin are representative tools for database management, an analysis of their GUI was made in order to identify common patterns on them.

The major similarity in all these applications is the placement of a treeview on the left side presenting all the servers that the user can manage. These nodes can be expanded showing extra nodes like databases, schemas, tables or views (see Figure 2.3, Figure 2.4, Figure 2.5).

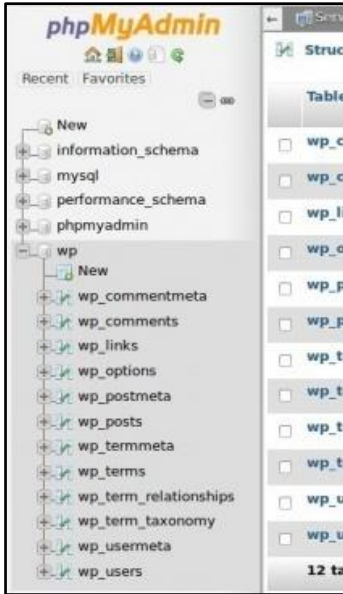


Figure 2.3 - Treeview on phpMyAdmin

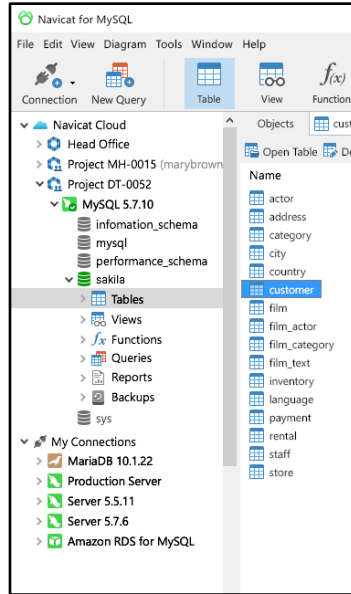


Figure 2.4 - Treeview on Navicat

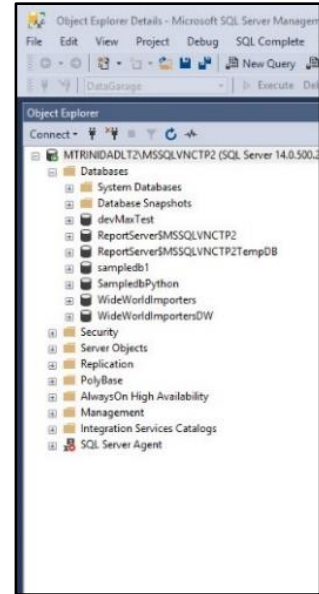


Figure 2.5 - Treeview on SQL Management Studio

Another common aspect is the usage of tabs to organize the different views of the application. Both Navicat and SQL Management Studio have this built-in, allowing the users to have multiple views open and to easily switch between them (see Figure 2.6, Figure 2.7). phpMyAdmin does not have this implemented, although, since it is a web application that runs in browsers and considering that nowadays all modern browsers have built-in tabs systems, it is also possible to open several tabs with different views, which enhances the efficiency of the navigation activities.

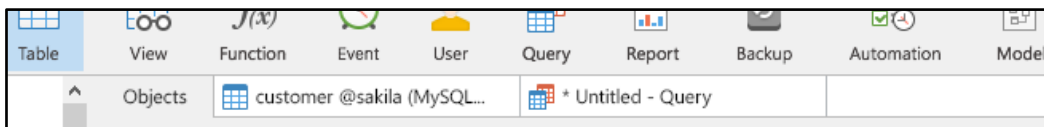


Figure 2.6 - Tabs on Navicat

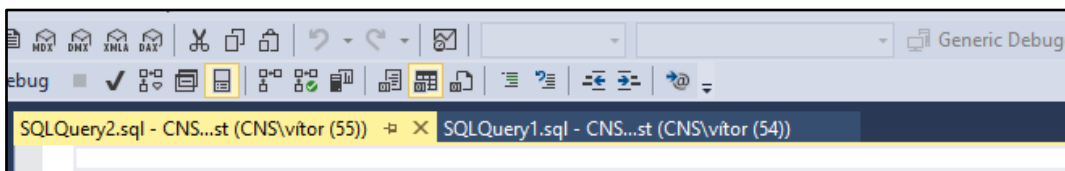


Figure 2.7 - Tabs on SQL Management Studio

Regarding tools organization, the three applications differ a bit.

Navicat groups the different actions in a menu bar, placing the most common views in a ribbon, thus, in each tab, it is presented a couple of other actions depending on the view the tab is presenting.

Microsoft SQL Management Studio uses a more old-fashioned design, organizing the actions in a menu bar and in some toolbars on the top, in this case, many buttons might be disabled because they are not related with the view presented on that moment, increasing the visual load of the application unnecessarily.

phpMyAdmin follows a similar approach to Navicat, it has a buttons' group in the top that allows the users to navigate to other pages, and on those pages, other actions are presented.

2.3 Software Integration Technologies

Now that we have covered the characteristics of database management applications, we will make an overview of how the integration of different systems is done. The evaluation of this topic will help to have a better understanding of this market and what is the role of the Connect Bridge Platform on it.

Software integration or system integration refers to the process of combining different subsystems in order to easily and quickly share data between them [37].

Usually, companies use many different applications with different complexity levels. Sometimes to increase efficiency and answer to challenges posed by a world in constant change, it is needed to integrate these complex systems. Due to this, in the last years, data integration played a significant role, with many organizations spending millions of dollars on the development of solutions for this problem [38]. Usually these solutions follow one of the following approaches: Custom Coding or Workflow Engines.

In the first approach, the integration is made manually using custom code developed specifically for a specific integration problem. This is the traditional approach to this problem, and demands a significant effort from developers, since it requires in-depth knowledge about each one of the systems that are being integrated. There is a number of factors that can delay the delivery of custom solutions; they arise not only from the system complexity but also from a set of other challenges like different programming languages, complex database structures or inadequate documentation. Furthermore, these solutions require high support and maintenance, because every time a new update is released, the integration code needs to be changed [38].

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

The second approach consists of applications with drag and drop interfaces that allow the creation of data workflows. This approach is easier and less error-prone, although it limits integration actions, due to its dependency on already developed modules. It is possible to find many products in this area like Nintex, Pentaho, Actian, IBM InfoSphere, Microsoft SQL Server Integration Services, SAP® BusinessObjects™ Integration, among others. [38].

Connect Bridge is a facilitator for the development of integration solutions. CB is a developer's tool that helps with the creation of integration code without the concern of knowing the target systems databases, APIs, or their versions. CB acts as an abstraction layer that allows developers to interact with these systems using SQL. The simplified architecture of an integration program that uses CB is as follows:

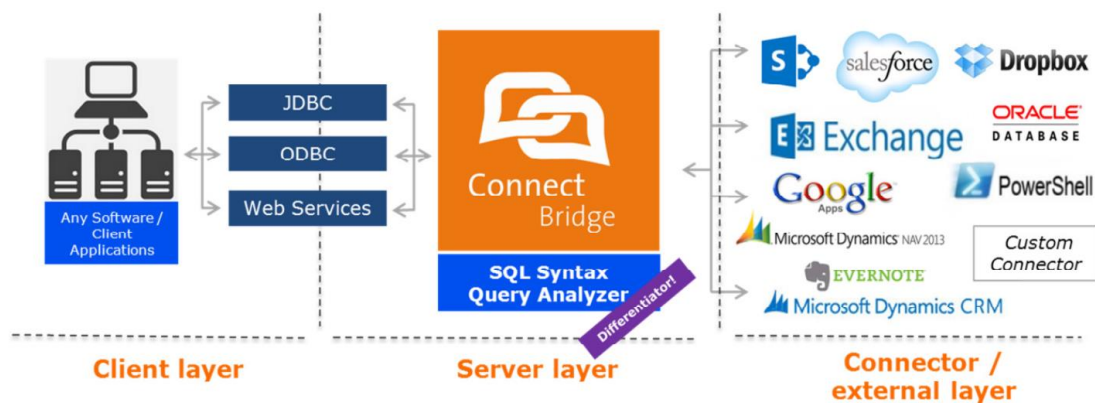


Figure 2.8 – Architecture of an application that uses Connect Bridge

On the Client layer, the developer establishes a connection to the server layer using one of the available drivers (JDBC, ODBC or Web Services) and creates all the integration logic needed, requesting/uploading data from/to the target systems using SQL queries to the CB Server.

On the Server layer, the SQL is interpreted and forwarded to the target system connector which converts the SQL command into an API call that is sent to the target system.

As a complement to this architecture, two other tools are available for the integrator. The developer can test his queries with the Query Analyzer where he/she writes the SQL commands and verifies its results. To configure the connection between CB Server and the target system, the developer uses the Administration Tool application. This dissertation proposes the combination of these two tools in a unique web-based application.

3

SYSTEM DESIGN

In this chapter, the process, architecture and implementation decisions will be presented.

3.1 Methodology

For the development of this project, none of the standard development cycles was strictly followed. Instead, an iterative agile-based approach was used.

Since the beginning of the project, Connecting Software gave excellent support for it, allowing the possibility of having weekly meetings with the company's supervisor for this project.

The development of this project followed the following line:

1. Requirements Gathering – Read and understand internal documents about the company's vision for this project, discussions with company's supervisor about it (see section 3.2).
2. Research – Research about possible technologies to be used in the project (see Chapter 2).
3. Prototyping – Creation of GUI interfaces prototypes and feedback collection from company members (see section 3.3).
4. Development – Iterative process following an Agile methodology. All the development was followed by the company's supervisor, which performed the role of Product Owner (person responsible to define and prioritize the list of

Connect Bridge Management Studio:

A web-based application used to manage and test Connect Bridge servers

features, bugs and improvements to be addressed) according to the chosen development method (see Chapter 4).

5. Evaluation – Comparison between the developed solution and the previous tools and User Evaluation using the Think Aloud technique (see Chapter 5).

3.2 Requirements Gathering

Connect Bridge Management Studio is an old ambition of Connecting Software. Because of this, there was already old internal documents that described the goals and features of this tool. Despite this, the requirement gathering process was not straightforward since several of the features described in those documents were unrealistic or not relevant anymore.

In order to better understand which requirements were still valid, several discussions between the company's supervisor and the developer responsible for this project were made. As a result of that, a use cases diagram and a more formal list of requirements were produced.

3.2.1 Use Cases

Use Cases, as defined by Object Modelling Group, are used to capture the requirements of systems [39].

A Use Case represents an action that an Actor can perform in a system being specified.

An Actor specifies a role played by a user or any other system that interacts with the system being specified.

During the meetings, it was clear that the system would have two types of users: Administrator and Developer.

The Administrator is the user that is mainly responsible for the management of the Connect Bridge server, therefore, an example of a Use Case for him is to “Manage CB Users”, or “Manage CB Accounts”. Mainly this actor will perform all the tasks that were previously executed on the CB Administration Tool.

The Developer is the user that uses Connect Bridge Platform to develop an integration solution, therefore he is only concerned on testing the different connectors through the execution of multiple SQL statements. This will help him understanding how Connect Bridge Platform can be useful for his integration solution. This scenario is

represented by the Use Case “Query Target Systems”. In general, this actor represents a normal user of the existing tool CB Query Analyzer.

Since an Administrator can perform all the Use Cases of a Developer, we consider that the Actor Administrator extends the Actor Developer.

The two diagrams below specify the use cases for this two Actors.

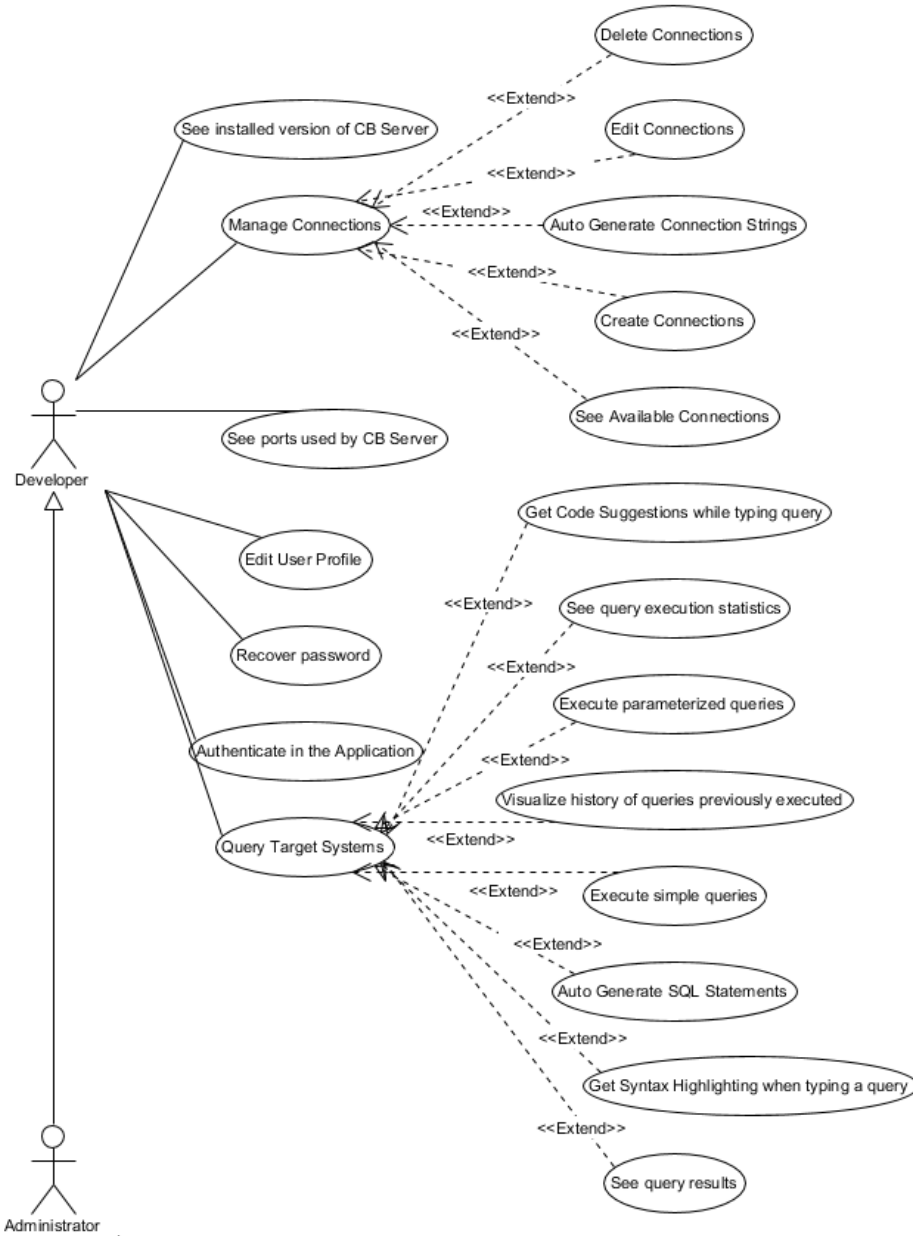


Figure 3.1 - Use Cases for Developer User

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

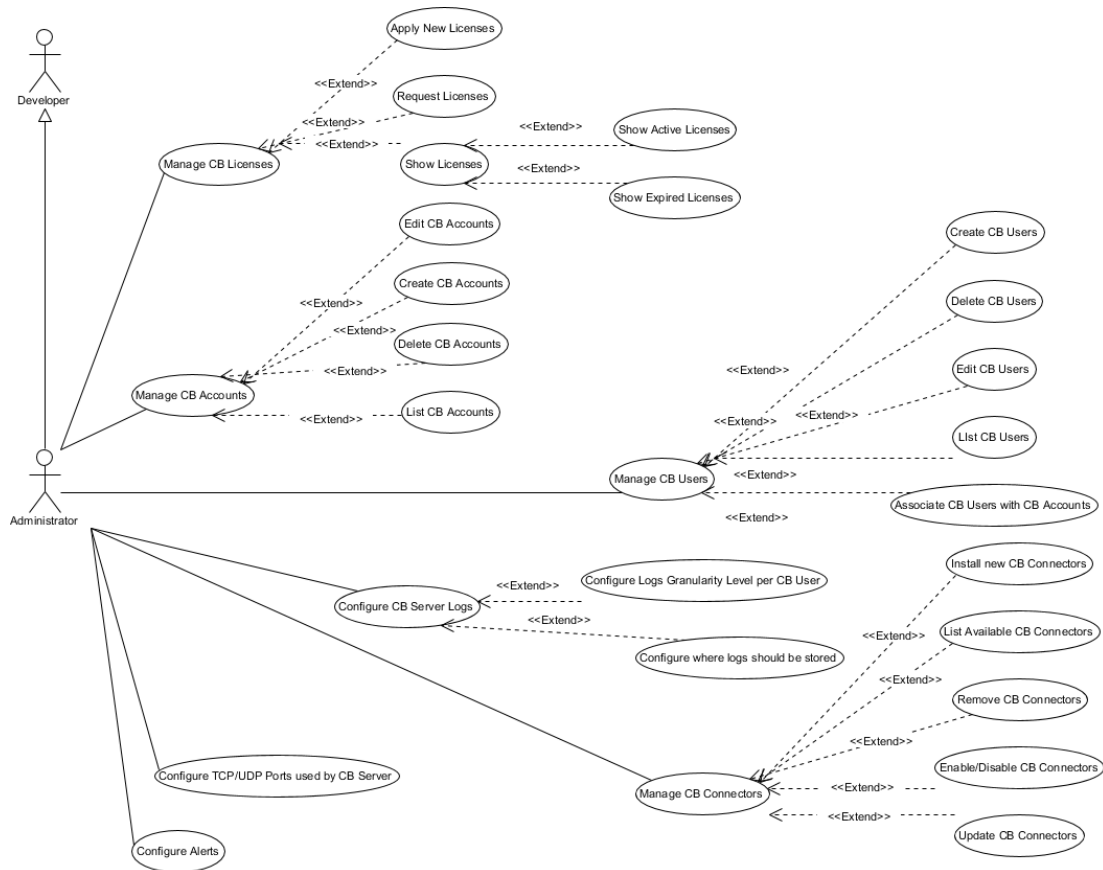


Figure 3.2 - Use Cases for Administrator User

3.2.2 Requirements List

In this section we will present a formal list of requirements. The majority of these requirements were extracted from old documents that described the goals of this project and were complemented with the use cases presented above and with an analysis of the features available on the existing tools that we aim to replace.

3.2.2.1 Business Requirement

According with the Product Owner: *“This project aims to create a modern, flexible, extendable cross-platform desktop application for Connect Bridge Software which will replace the existing client tools CB Administration Tool, CB Query Analyzer and CB Service Controller.*

The ultimate business goal is to provide a better user experience across all major platforms for Connect Bridge customers and partners which will attract new prospects.

By unifying all mentioned client tools and adopting the concept of an IDE (Integrated Development Environment), Connecting Software pretends to create a DevOps¹² tool, allowing the Development (Query Analyzer) and Operations (Administration Tool) users to work more efficiently and effectively.”

3.2.2.2 Generic System Requirements

This section presents a group of requirements that are transversal to all the application.

3.2.2.2.1 Authentication

G.Authentication.1. The system must allow users to login into the application using simple authentication (username and password).

G.Authentication.1.1. When entering the application, a form with textboxes for username and password shall be presented.

G.Authentication.2. The system shall allow the users to recover its password when forgotten.

G.Authentication.2.1. The recovery process shall be processed after the user correctly answers three security questions predefined by him.

3.2.2.2.2 User interface

G.UserInterface.1. The system must have a modern look and feel appropriate for desktop and laptop computers.

G.UserInterface.2. The system shall have the capability for the user to personalize the interface according to their preferences.

3.2.2.2.3 Multiple users

G.MultipleUsers.1. The system shall support multiple users connected in parallel.

G.MultipleUsers.1.1. Each connected user should not interfere with the other users logged in at the same time.

¹² “DevOps is a set of practices that automates the processes between software development and IT teams, in order that they can build, test, and release software faster and more reliably” [40].

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

3.2.2.2.4 Roles and Permissions

G.RolesPermissions.1. The system must support at least two roles: Administrator and Developer.

3.2.2.2.5 Technologies

G.Technologies.1. The system must be platform independent.

G.Technologies.1.1. The application must run on the following desktop operating systems : Windows, macOS, and Linux.

G.Technologies.2. The system must use web-based technologies.

3.2.2.2.6 Multiple CB Servers

G.MultipleServers.1. The application shall be able to manage and query multiple CB Servers.

3.2.2.3 Management Module Requirements

This group of requirements defines all the requested features related to the management of CB Servers.

3.2.2.3.1 CB Server Logs Configuration

M.Logs.1. The system shall allow the Administrator to configure the logs generated by CB Server

M.Logs.1.1. The system shall allow the Administrator to configure where the logs of CB Server will be stored (Windows Event Viewer or Text File).

M.Logs.1.1.1. The system shall allow the Administrator to select any of these options at the same time.

M.Logs.1.2. The system shall allow the Administrator to configure which level of granularity (Error, Info, Warning, Fatal or Debug) should be stored.

M.Logs.1.2.1. The system shall allow the Administrator to select any of these options at the same time.

M.Logs.2. The system shall allow the Administrator to export the generated logs to a zip file.

M.Logs.3. The system shall allow users to browse logs

M.Logs.3.1. The logs must be presented in a way more structured than simple text

M.Logs.3.1.1. Each log line should have a different colour according to its granularity.

M.Logs.3.2. The logs shall be filtered by granularity level.

M.Logs.4. The system shall allow users to clear the logs.

3.2.2.3.2 CB Server Alerts Configuration

M.Alerts.1. The system shall allow the configuration and creation of alerts (to inform the users about important information (e.g., license expiration)).

M.Alerts.1.1. The generated alerts could be in the form of push-up notifications and e-mails

3.2.2.3.3 CB Server Users Management

M.Users.1. The system must allow the Administrator to manage CB Server users.

M.Users.1.1. The system shall allow the Administrator to create new users on a CB Server.

M.Users.1.2. The system shall allow the Administrator to modify users on a CB Server.

M.Users.1.3. The system shall allow the Administrator to remove users on a CB Server.

M.Users.2. The system must allow the Administrator to list all the CB users created on a CB Server. CB Server Accounts Management

M.Accounts.1. The system must allow the Administrator to manage CB Server accounts.

M.Accounts.1.1. The system shall allow the Administrator to create new Accounts on a CB Server.

M.Accounts.1.2. The system shall allow the Administrator to modify Accounts on a CB Server.

M.Accounts.1.3. The system shall allow the Administrator to remove Accounts on a CB Server.

M.Accounts.1.4. The system shall allow the Administrator to associate CB Accounts with CB Users on a CB Server.

M.Accounts.2. The system must give the Administrator the possibility to test the connection of the configured accounts with its correspondent target system.

M.Accounts.3. The system must allow the Administrator to list all the accounts created on a CB Server.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

3.2.2.3.4 CB Server Connectors Management

M.Connectors.1. The system must allow the Administrator to manage CB Server connectors.

M.Connectors.1.1. The system shall allow the Administrator to install new CB Server Connectors.

M.Connectors.1.2. The system shall allow the Administrator to upgrade CB Server Connectors.

M.Connectors.1.3. The system shall allow the Administrator to activate/deactivate CB Server Connectors.

M.Connectors.2. The system must allow the Administrator to list all the connectors installed on a CB Server, including its version and installation path.

3.2.2.3.5 CB Server Licenses Administration

M.Licenses.1. The system shall be able to generate notifications to inform Administrator when the licenses are close to expire.

M.Licenses.2. The system must allow the Administrator to list all the licenses associated with CB Server.

M.Licenses.2.1. The system shall allow the visualization of active and expired licenses.

M.Licenses.3. The system must allow the Administrator to request new licenses.

M.Licenses.4. The system must allow the Administrator to apply new licenses.

3.2.2.3.6 CB Server Version and Updates

M.VersionUpdates.1. The system shall be able to list the currently installed version of CB Server, connectors and drivers.

M.VersionUpdates.2. The system shall generate notification every time a new version of CB Server/Connector/Driver is available.

M.VersionUpdates.2.1. The system shall inform how the user can apply those updates.

3.2.2.3.7 TCP/UDP Ports Configuration of CB Server

M.Ports.1. The system shall present information about the TCP/UDP ports that are currently used by the CB Server.

M.Ports.2. The system shall allow Administrator to change TCP/UDP ports used by CB Server.

3.2.2.4 Querying Module Requirements

The following requirements define all the functionalities that should be included regarding the querying system used to test CB Servers.

3.2.2.4.1 *Connections with the Target Systems*

Q.Connections.1. Based on the current logged in user, the system must automatically show all the CB Server accounts the user has permission to access.

Q.Connections.2. The system shall be able to automatically generate connection strings for all the available drivers.

Q.Connections.2.1. The system shall be able to show to the users the automatically generated connection strings when requested.

Q.Connections.3. The system shall allow the possibility to configure Extended Properties¹³.

Q.Connections.3.1. The system shall generate connection strings according to the configured Extended Properties.

Q.Connections.3.2. The system shall start querying the target systems considering configured Extended Properties.

3.2.2.4.2 *Query Editor*

Q.Editor.1. The editor used to write queries shall provide syntax highlighting for all the supported keywords.

Q.Editor.2. The editor shall have the functionality of smart autocompletion.

Q.Editor.2.1. The editor shall provide autocompletion for SQL keywords.

Q.Editor.2.2. The editor shall provide autocompletion for tables names.

Q.Editor.2.3. The editor shall provide autocompletion for columns names.

Q.Editor.2.4. The editor shall provide autocompletion for procedures names.

3.2.2.4.3 *Query Results*

Q.Results.1. The system must be able to open multiple query windows and run queries in parallel

Q.Results.2. After the execution of a query on the query view, the system shall show the results of that query in the form of a table.

¹³ Extended properties are properties that are defined on a connection string that allow the users to override the CB Account properties previously defined.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

Q.Results.3. SQL statements separated by a semicolon and new line shall return multiple result tables.

Q.Results.4. If any SQL statements are selected on the editor, only that one shall be executed.

Q.Results.5. After the execution of a query on the query view, the system shall allow the users to access statistics related to the query.

Q.Results.5.1. The statistics presented shall include the execution time, number of returned/affected rows and error messages (if occurred).

3.2.2.4.4 Query History

Q.History.1. The system must log the queries that are executed by the user.

Q.History.2. The system must allow the user to visualize the queries executed in the past and the statistics associated with them.

Q.History.2.1. The statistics shown shall be the same specified on Q.Results.5.1

3.2.2.4.5 Auto-generation of Queries

Q.AutoGeneration.1. The system shall be able to generate template query statements for each table/procedure automatically.

3.2.2.4.6 Parameterized Queries

Q.ParameterizedQuery.1. The system shall have the possibility to execute parameterized queries.

Q.ParameterizedQuery.2. When the parameter data type is an array of bytes, the system shall allow the user to select an existing file on his computer.

3.3 Prototyping

After collecting the requirements presented above, medium-low fidelity mock-ups of the user interface were created using the tool Balsamiq¹⁴. This was an iterative process where the company's supervisor reviewed all the mock-ups. An example of the mock-ups created is presented in Figure 3.3, (see the Appendix A to visualize the other mock-ups designed).

¹⁴ <https://balsamiq.com/>

After approving all the mock-ups, the Product Owner prioritized them. This prioritization was defined considering their importance to achieve a Minimum Valuable Product (MVP) and estimated difficulty to implement.

Therefore, the views were prioritized in the following order:

1. Views that would be used to replace Administration Tool;
2. Views that would be used to replace Query Analyzer;
3. Nice-to-have views (views that were not essential to achieve an MVP);

From this point, the development of high fidelity prototypes was started, using the front-end technology stack used (Angular 5 Framework).

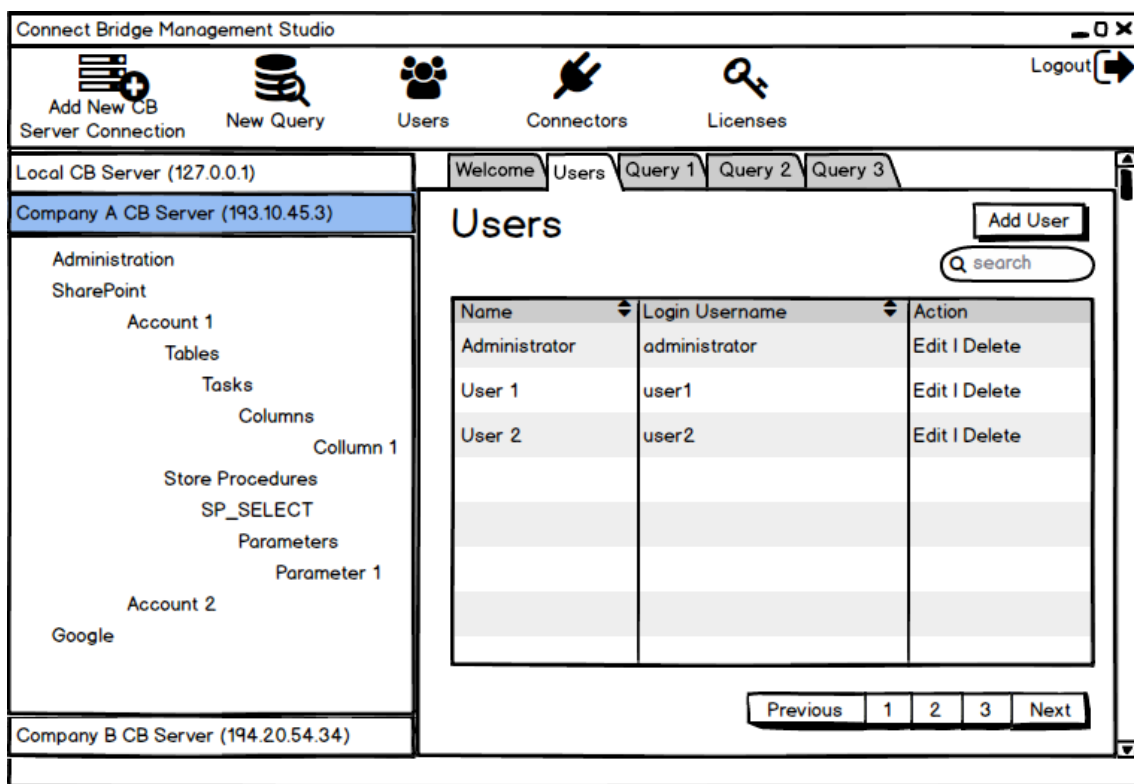


Figure 3.3 – Medium-low-fidelity prototype of Users’ view

During this process, a meeting with a multidisciplinary team from the company, including colleagues from Management, Marketing, Customer Support and Development allowed us to receive inputs about the work completed so far. This meeting was suggested by the company’s Chief Executive Officer (CEO) and was immediately accepted since it allowed to get feedback and other perspectives about the design, and the User Experience (UX) associated to this product.

Connect Bridge Management Studio:

A web-based application used to manage and test Connect Bridge servers

In this meeting, the medium-low fidelity prototypes were presented as also some of the already implemented views in Angular. An example of a view already implemented as high-fidelity prototype, which was reviewed during this meeting, is presented in Figure 3.4, which implements the low fidelity prototype presented before (see Figure 3.3).

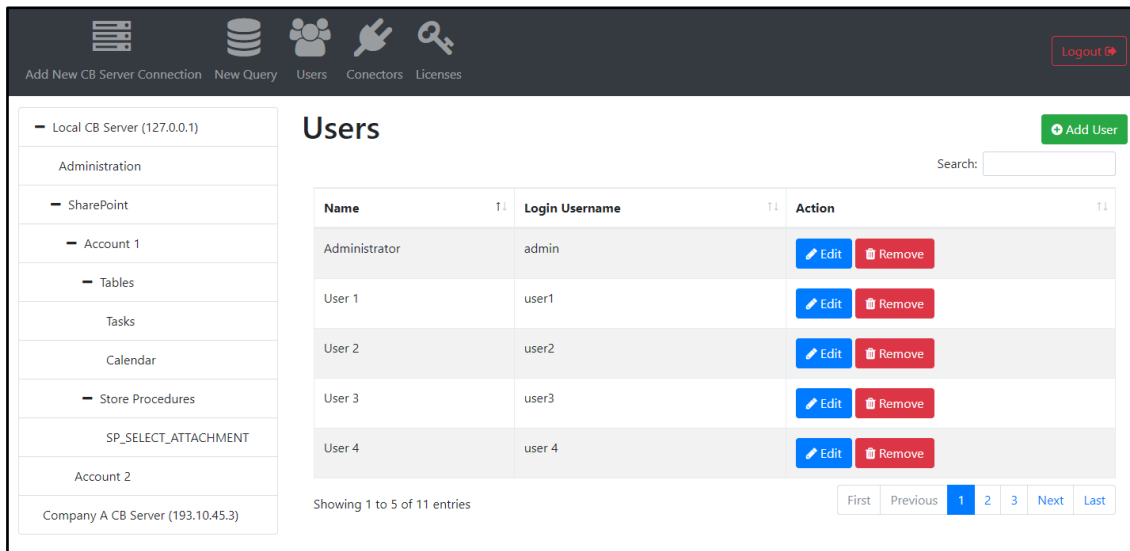


Figure 3.4 – First version of the high-fidelity prototype of Users' view

The overall feedback was positive, although, the CEO showed concerns about the design consistency between the company products. It was argued that the company already has a product that uses a web interface (see Figure 3.5, for an example view of that product) and that we should use at least the same colour pallet and base template to preserve consistency.

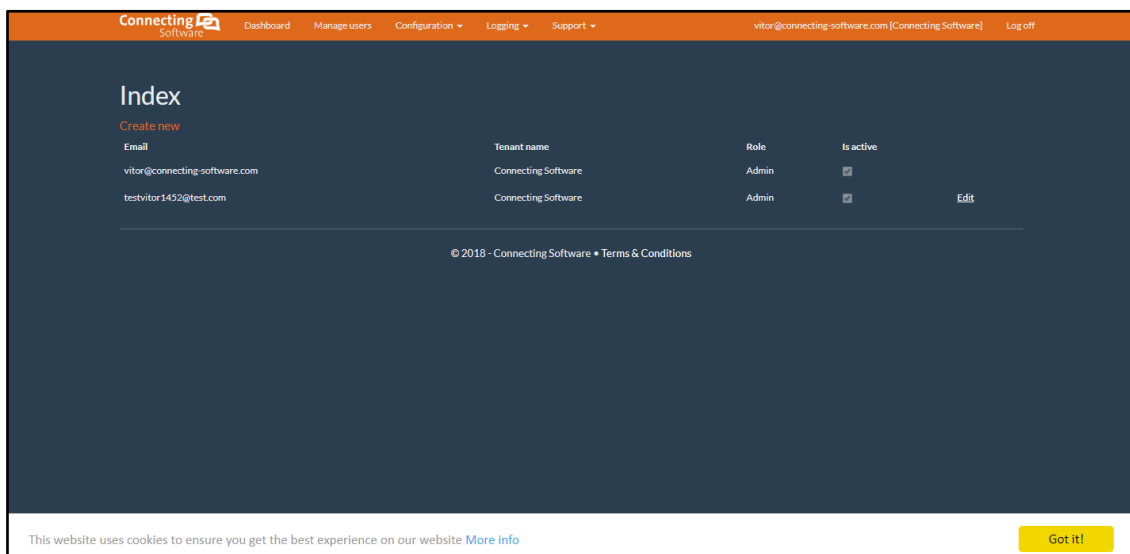


Figure 3.5 - Existing User Interface of SaaS platform

Based on this feedback, a new version of the high-fidelity prototype was made, in which the same template and colours of SaaS¹⁵ platform were applied. Finished with this, a new meeting with the same people was scheduled. Figure 3.6 shows one of the views presented at this meeting.

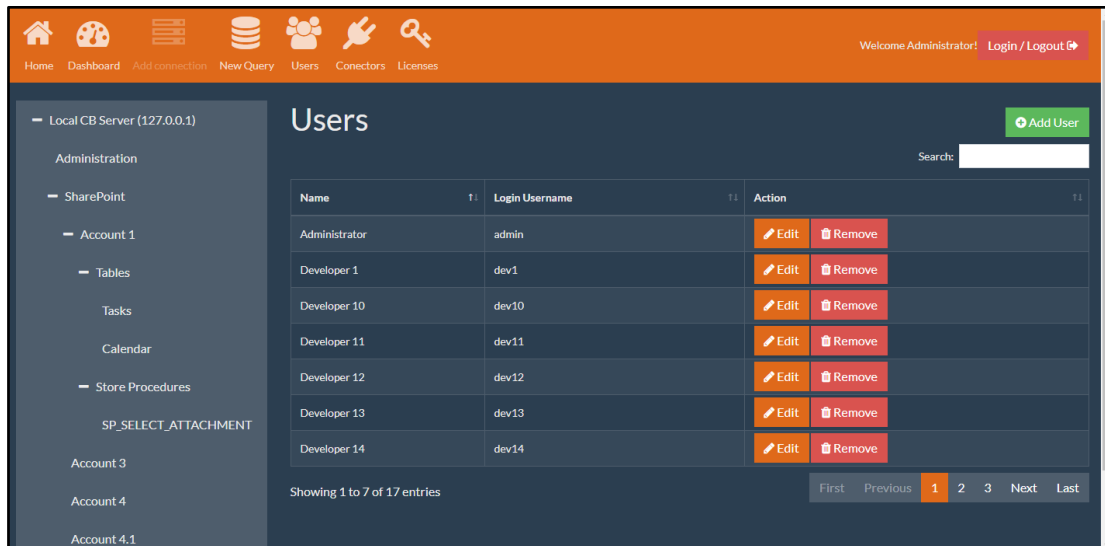


Figure 3.6 - First version of the high-fidelity prototype of Users' view

In this meeting, the CEO argued that some aspects needed to be refined and that some views had too many colours. Although, considering that this was a problem related with the template used by Connecting's Software SaaS Platform, it was decided to execute refinements and continue the development. It was also decided that in the future, and already outside of the scope of this thesis, a new template would be prepared by the marketing team.

Figure 3.7 shows the final version of the Users view, achieved at the end of this project.

¹⁵ SaaS - Software as a Service – In the context of this document is used to reference a cloud service offered by Connecting Software

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

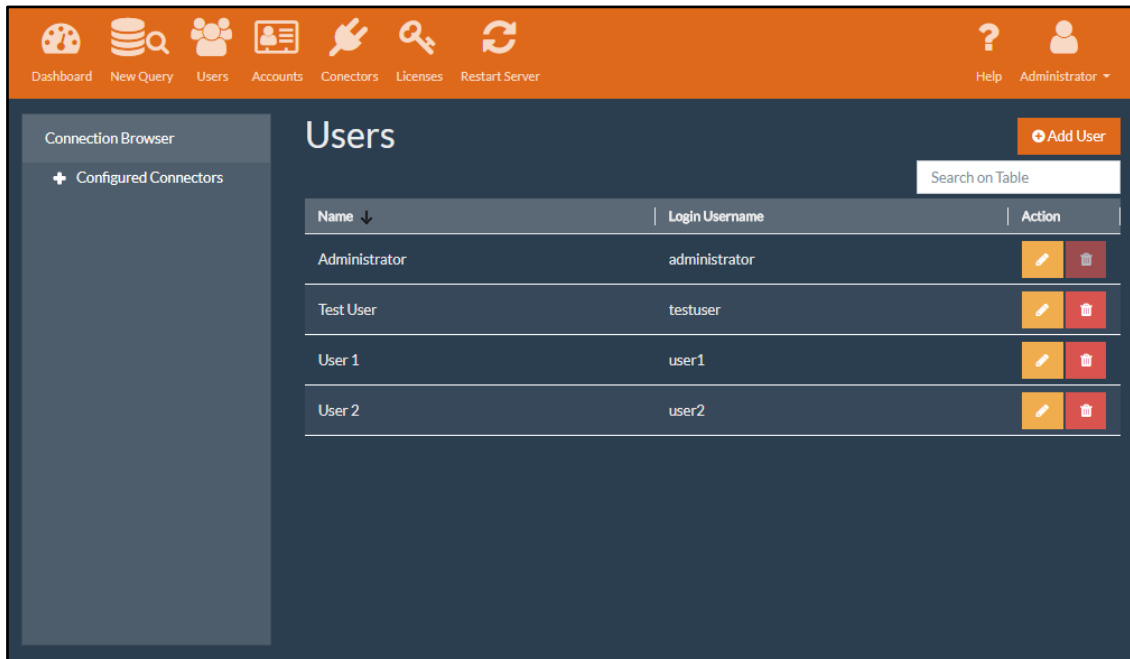


Figure 3.7 - Final version of Users' view

3.4 Architecture

After the design and approval of all the views that would be implemented we started by defining architectural aspects. This section presents the different possible architectures idealized for CB Management Studio. All the architectures presented were designed considering the previously collected requirements, idealized during the initial stage of the project and reviewed by the Product Owner and the company's CEO.

3.4.1 Broker Architecture

Considering the requirements G.MultipleServers.1, G.Technologies.1 and G.Technologies.2 and the research during the beginning of this project (see Chapter 2.1), two similar architectures were proposed.

The first architecture consists of a single server that would host the back-end of CB Management Studio and its database. This server would provide a web interface as GUI and would act as a broker in the sense that would be connected to several CB Servers to manage and query them. Since the system shall be cross-platform, the communications between CB Management Studio Server and each CB Server should be done using HTTP (using CB Webservices driver) considering that CB ODBC and JDBC drivers currently only work in Windows-based systems.

The Web application shall follow the pattern of a Single Page Application as presented on 2.1.3.1, for this reason, the communications between this application and its back-end should be made through HTTP (mainly using asynchronous requests).

The Figure 3.8 schematizes this architecture.

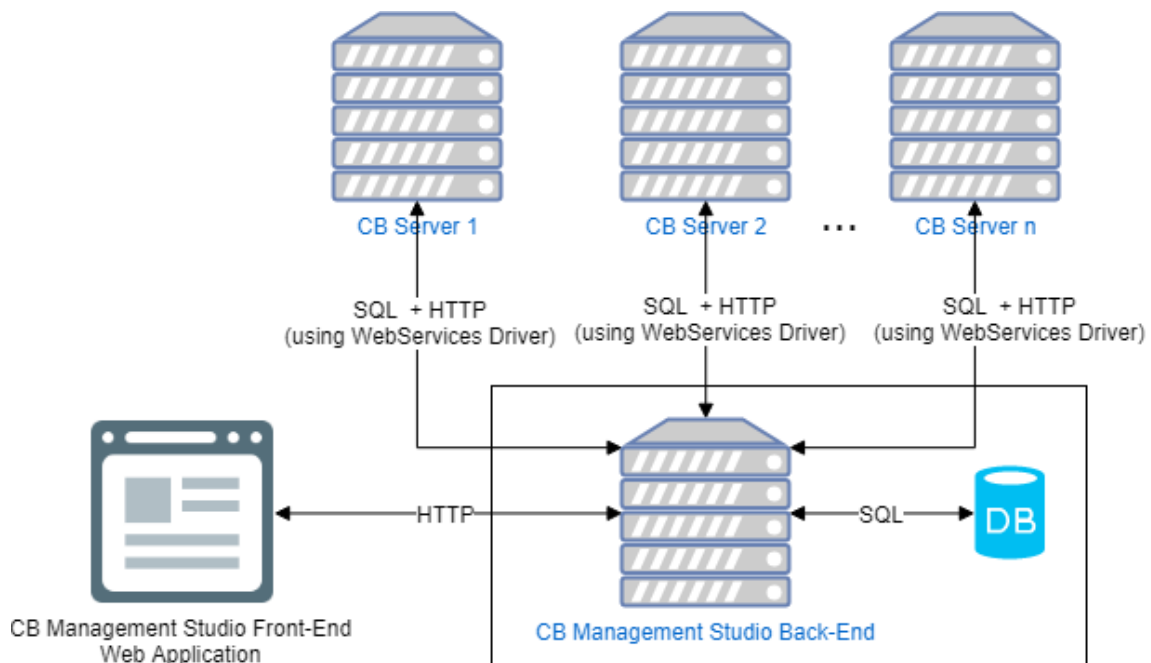


Figure 3.8 - Initial architecture to support multiple CB Servers

Considering the new technologies like Electron or NW.js, already presented on Chapter 2.1.3.2, another architecture was also developed and discussed with Connecting Software.

This architecture is very similar to the previous one but considers making CB Management Studio a native Desktop application, where the front-end web application and its back-end are packed together in a single Electron application.

Figure 3.9 presents a schema that considers this architecture.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

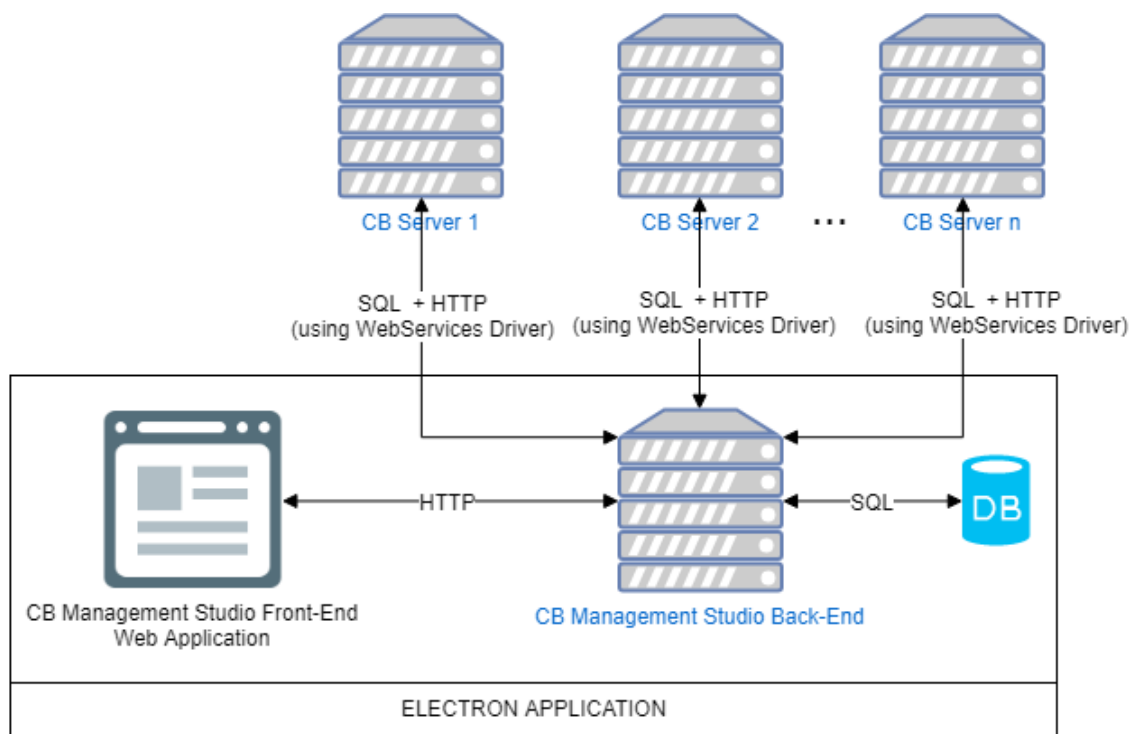


Figure 3.9 – Architecture considering CB Management Studio as a hybrid desktop application

Despite both architectures fit the purpose of this project, Connecting Software's CEO refused the idea presented on the last architecture, considering that, despite continuing being a cross-platform solution it would require future users to install and configure a new application in their computers which is something he wanted to avoid with this project.

The first architecture, although simple, has several pitfalls that worried the company's supervisor. The major one was the fact of having an intermediate server. Is not clear which would be the best place to physically install this server (examples might be Azure, other cloud systems or on Connecting Software's premises). Also it would increase the complexity of the solution since the application would need to be prepared to handle the multiple servers using only one GUI and URL which would represent a big challenge both in terms of front-end and back-end.

Because of this, it was agreed that a simpler architecture should be designed, which will be explained in the following section. Although, despite being simpler, this architecture still uses the core ideas discussed above.

3.4.2 Client-Server Architecture

To reduce the complexity of the architectures presented above and to ensure that at the end of this thesis, the artefact produced corresponds to a Minimum Valuable Product (MVP), an architecture following the traditional Client-Server approach was recommended by the Product Owner. In this solution, CB Management Studio back-end will be hosted on the same machine as CB Server, and the users would use their browser to access CB Management Studio front-end connecting to this machine in a predefined TCP port (see Figure 3.10). The communications between CB Management Studio Back-End and the CB Server should be done using the same technologies described above (CB Web Services Driver) for the reasons already presented.

At first look, this architecture may suggest that the requirement G.MultipleServers.1 would not be satisfied, although, since all modern browsers support tab views, if the users decide to manage or query different servers, they can do it recurring to different tabs of the browser, each one targeting a different server.

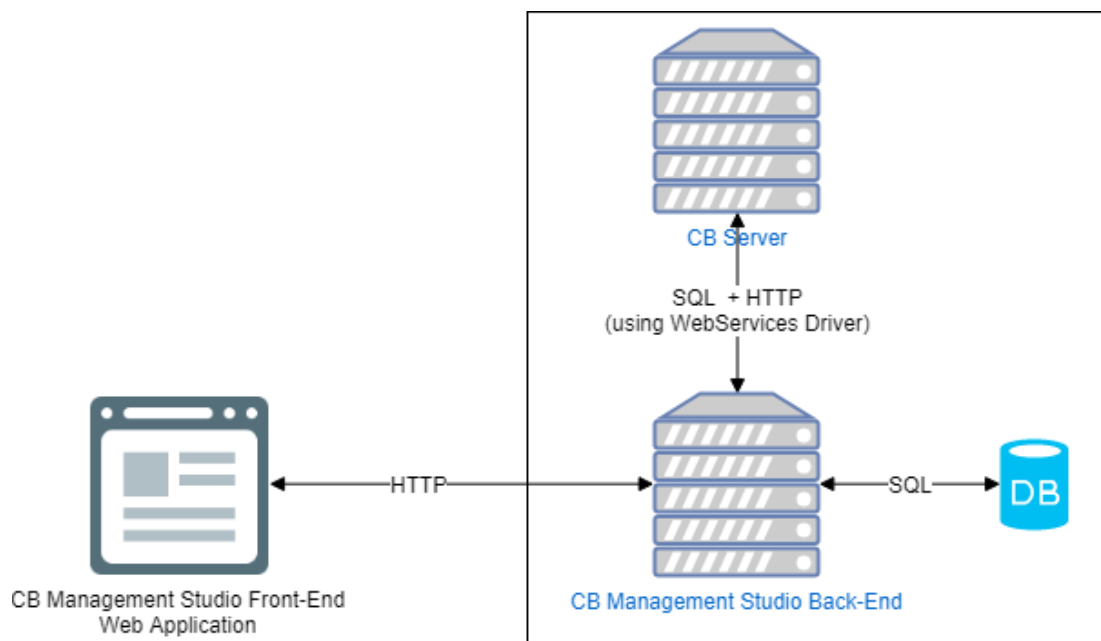


Figure 3.10 – CB Management Studio using Client-Server Architecture

This was the final architecture presented, and despite it being simpler and easier to implement, it fulfills all the requirements, which resulted in the Connecting Software approval. Consequently, all the development was made considering this architecture.

4

DEVELOPMENT

In this chapter, details about the development of this solution will be presented. This chapter starts by exposing the planning of the development stage, followed by a description of the most critical concepts, decisions, and the difficulties faced during the development of the presented solution.

4.1 Project Planning

Connecting Software uses Team Foundation Server (TFS)¹⁶ as control versioning and project management system.

The company tries to follow an agile methodology based on Scrum¹⁷ and TFS is configured to help on that management, this way, the development stage of this project was managed using this tool.

Scrum was created by Ken Schwaber and Jeff Sutherland and, according to them, it is “A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value” [41].

This framework describes a process that, in a resume, consists of the following steps [42]:

¹⁶ <https://visualstudio.microsoft.com/tfs/>

¹⁷ <https://www.scrum.org/>

Connect Bridge Management Studio:

A web-based application used to manage and test Connect Bridge servers

1. A document called as product backlog is created by a person named as a product owner which consists of a prioritized list of customer wishes/requirements (named as User Stories).
2. At the beginning of an iteration, also called as a sprint, the development team takes a set of the User Stories with higher priority and decides how they will be implemented, forming the sprint backlog.
3. A sprint is a short period (usually two to four weeks) that is established to implement all the User Stories present on the sprint backlog.
4. During the sprint, the team should have daily short meetings to discuss their progress and problems. Also, a person named as Scrum Master helps the team to keep focused on achieving the goals defined to the sprint.
5. At the end of the sprint, the work done should result on a new version of the product ready to show to a customer or stakeholder and the team should make a review and retrospective over the sprint.
6. The process is repeated until all User Stories are fulfilled.

Scrum is aimed to be used by teams around 3 to 9 members (excluding the Product Owner and Scrum Master), although, despite that this project only had one developer, many of the concepts of Scrum were applied to it:

1. The company's supervisor performed the roles of Product Owner and Scrum Master. He created a backlog with User Stories, prioritized them and followed all the development through weekly meetings.
2. The development was split into iterations, following the iterations that Connecting Software uses to develop Connect Bridge Platform.
3. For each iteration, the developer took the User Stories with higher priority from the product backlog and split them into small tasks, estimating them.
4. At the end of each iteration, the developed solution was tested by the company's supervisor.

Figure 4.1 shows an excerpt of the product backlog that was configured on TFS for the first release of CB Management Studio.

ID	Work Item...	Title	Assigned To	State	Stack Ran...
10524	User Story	EPIC - CBManagementStudio - Release 1.0	Vitor Baptista	Active	
10738	User Story	login with my user and password	Vitor Baptista	Closed	1
10756	Task	Implement HTML views for login	Vitor Baptista	Closed	
11260	Task	Implement Authentication and Authorization using JWT	Vitor Baptista	Closed	
10837	User Story	List Licenses	Vitor Baptista	Closed	1.1
10855	Task	Create view to list licenses	Vitor Baptista	Closed	
11467	Task	Retrieve Licenses Data from CB Server	Vitor Baptista	Closed	
11471	Task	Present licenses retrieved from API in the view	Vitor Baptista	Closed	
10747	User Story	create, modify, disable and delete users in the Connect Bridge Plat...	Vitor Baptista	Closed	1.2
11512	Task	Implement Creation of Users	Vitor Baptista	Closed	
10821	Task	Create HTML view to add users	Vitor Baptista	Closed	
11513	Task	Implement update of users	Vitor Baptista	Closed	
11514	Task	Implement Delete users	Vitor Baptista	Closed	
11515	Task	List users	Vitor Baptista	Closed	
10773	Task	Create HTML view for list users	Vitor Baptista	Closed	
11433	Task	Retrieve Users data from CB Server	Vitor Baptista	Closed	
11442	Task	Populate the HTML view with the users from the API	Vitor Baptista	Closed	
11771	Task	Add validations to avoid mistakes while adding, updating or del...	Vitor Baptista	Closed	
11815	Task	Add Server side validations for users management	Vitor Baptista	Closed	
10750	User Story	list all the connectors that are enabled and see their versions	Vitor Baptista	Closed	1.3
10830	Task	Create view to list connectors	Vitor Baptista	Closed	
11428	Task	Retrieve list of connectors from CB Server and show them in th...	Vitor Baptista	Closed	
11516	Task	Present the version of the connectors	Vitor Baptista	Closed	

Figure 4.1 - Excerpt of the backlog for the first release of CB Management Studio

4.2 Development Environment

For the back-end, Visual Studio Enterprise 2017 was used as a development environment. Visual Studio is the official IDE for ASP.NET Core, with all the necessary tools to develop these kinds of applications.

For the front-end, Visual Studio Code was used. This tool is very lightweight and combined with some of the available extensions on its marketplace turns it in a powerful text editor to code Angular applications, providing snippets, syntax highlighting, code suggestions and syntax checking, among others.

4.3 Versioning

Control versioning of code allows keeping track of all modifications that are done on the code, allowing to easily rollback the changes if something wrong happens.

As referred to in section 4.1, Connecting Software uses Team Foundation Server (TFS) as control versioning. Due to this all the written code was version controlled and stored on it. The usage of Visual Studio Enterprise 2017 and Visual Studio Code helped with this

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

versioning, since these two tools allow direct integration with TFS, providing a GUI for performing versioning actions like Check-in (send a version for TFS Source Control system), navigate over the source code stored on TFS, perform roll-backs, create and merge branches, among others.

4.4 Back-end

Since Connecting Software uses mainly C# for their development, ASP.NET was the natural choice for developing a web application. Most of the company's developers are already familiar with this framework and selecting it to create CB Management Studio, would result in an application easier to maintain by other company's elements in the future.

Currently, ASP.NET has two main versions, ASP.NET and ASP.NET Core. The first one uses .NET Framework and targets only Windows-based systems and the second one uses .NET Core that targets Windows, Mac, and Linux-based systems.

Considering that CB Server currently only works on Windows and that we decide to go for the architecture presented in section 3.4.2, where CB Management Studio Back-end and the CB Server are installed on the same machine, we initially thought on using ASP.NET. Although, considering the plans of turning CB Server compatible with Linux-based systems, and the suggestions from Microsoft to start using .NET Core for new applications, we opted by using ASP.NET Core (v.2.0). This decision grants that, in the future, if Connecting Software decides to port CB Server and all its connectors to .NET Core, CB Management Studio will not have to change.

The back-end of CB Management Studio requires the storage of information like users, connections or logs. Due to this, a database in SQLite was created. SQLite was chosen as Database Management System (DBMS) considering that we would not need a very complex database. Moreover, this DBMS, does not require any additional software to work which decreases overloads or future configuration issues.

The interaction between ASP.NET Core and SQLite database was achieved using Entity Framework Core (v.2.0), an object-relational mapper (O/RM), that enables the usage of .NET objects to interact with the database.

4.4.1 Project Structure

The back-end project tried to follow the architectural guidelines recommended by Microsoft and the concept of Clean Architecture. Based on this, the project is divided in three major parts: Core, Web API and Infrastructure.

4.4.1.1 Core

The Core is responsible for defining all the entities that are relevant for the application, additionally, it also implements all the business logic needed to fulfil the requirements presented in section 3.2.

In order to implement the business logic, for each entity, there is a class named Service which handles all the logic related with that entity. Since some of the actions defined on the services also impact other entities, a set of Events were defined and are raised when any of those action occurs. For each event, exists a class called EventHandler that holds the behaviour needed to handle those events.

The following diagram shows an example of this procedure. The designed solution has two entities to represent the concept of User: the CB User and the App User. A CB User is a user registered on a CB Server, while the App User represents a user of CB Management Studio. In practical terms, they represent the same person, although this distinction needs to exist because CB Management Studio needs to associate extra data with the CB User, for example a record of queries executed or connections configured. Since they represent the same person, it is needed to ensure that every time a CB User is updated, then the corresponding App User is also updated. Due to this, after CbUserService updates a CB User in a CB Server, it publishes an event that is handled by CbUserEditedEventHandler, which, based on the notification received, updates the App User record on CB Management Studio Database. The sequence diagram presented bellow schematizes this process.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

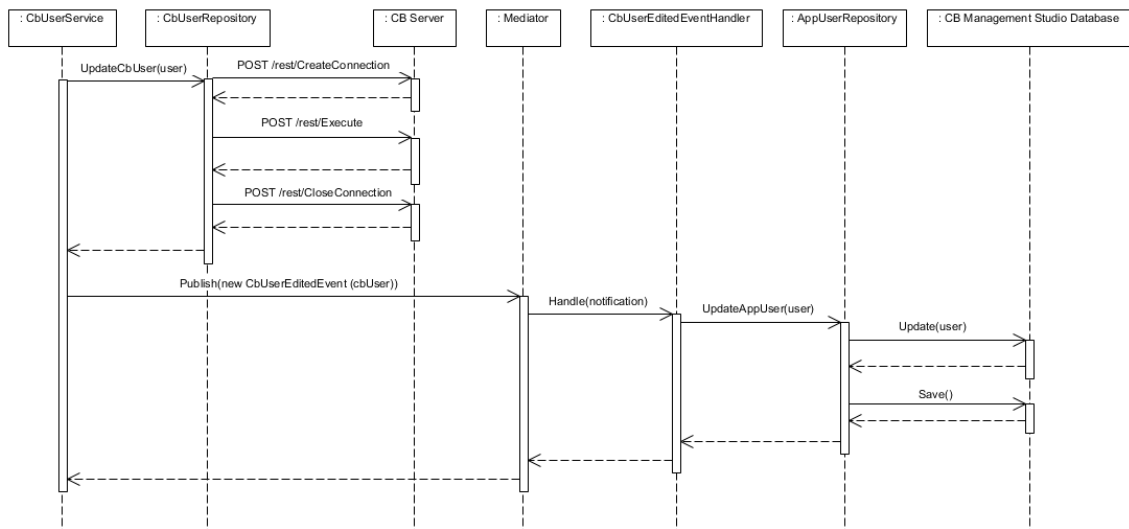


Figure 4.2 - Sequence diagram of the action "Update CB User" on the back-end of CB Management Studio

The Core is not responsible for interacting with the data sources, although it defines a set of interfaces with all the methods needed to access or modify the data on those data sources. These interfaces enforce the usage of the Repository and Specification Patterns.

4.4.1.1.1 *Repository and Specification Patterns*

The Repository Pattern provides an abstraction over the data, providing a set of methods for performing the CRUD operations. The usage of a Repository removes the responsibility of interacting directly with a database (and its concerns like creating connections, commands or readers) from the classes that invoke the Repository methods. Also, the existence of a Repository allows these classes to be unaware of the data's source, meaning that if the source needs to be changed, only the repository classes will have to be modified [43].

The Specification Pattern tries to solve the problem of creating complex queries like sorting, joins or paging. Without applying the Specification Pattern, a Repository is limited to methods that get all the data or get a specific data (usually filtered by Id). Specification Pattern allows Repositories to have an extra method that receives an object (usually called as Specification) that defines extra conditions for those selections [44].

4.4.1.2 Infrastructure

The Infrastructure is the part of the project responsible for interacting with the CB Management Studio database and with the CB Server.

The communication with the CB Management Studio database was made using Entity Framework Core which was also used to create and manage the migrations of this database.

The interaction with a CB Server tries to be as similar as possible with the interaction with a standard relational database management system. Although, it is not possible to use the Entity Framework for it, which means that all the communications are done using SQL statements directly. For that, the CB Webservices Driver is being used. This driver exposes a Representational State Transfer (REST) API that allows the user to send CB SQL statements to a CB Server using HTTPS requests. For each request, CB Webservices Driver returns a response in a JavaScript Object Notation (JSON) format that can easily be adapted to a C# DataTable and then to objects that are instances of the Entity classes defined on the Core part. Figure 4.3 schematizes this workflow using as an example the execution of the query `SELECT * FROM Users` used to get all the CB users that are registered on a CB Server.

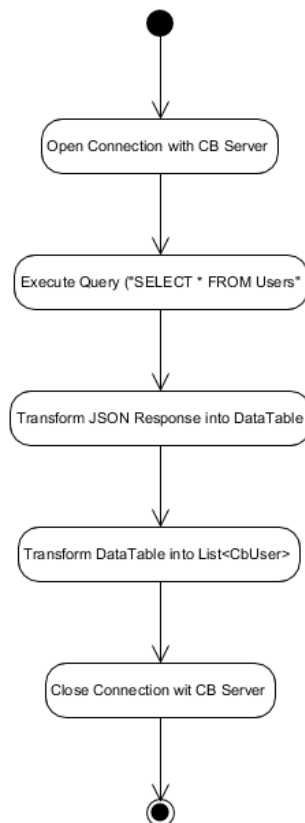


Figure 4.3 - Workflow of the repositories that interact with CB Server

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

4.4.1.3 Web API

The Web API is the back-end interface that allows the front-end to communicate with the rest of the back-end. The API was designed following the REST conventions and essentially consists in two main set of classes: ApiModels and Controllers. The ApiModels, define the structure of the requests and responses made/sent from/to Angular project. The Controllers define the routes of the Web API and are responsible to parse the ApiModels into Core entities, invoke the Services defined on the Core and convert their response into ApiModels.

4.5 Front-end

As already analysed in section 2.1.3.1, nowadays, Single Page Applications tend to be the solution for Web Applications that aim to replace desktop applications. Considering that CB Management Studio fits on this kind of applications we decided to use Angular, a JavaScript framework presented on section 2.1.3.1.2 since we believe this framework was the best fit to create an application with the desired characteristics.

4.5.1 Project Structure

The front-end was developed using Angular 5 following its best practices. Therefore, the Angular project is organized into Modules, where each module contains Components, Pipes, Directives, and Services.

4.5.1.1 Components

A Component can be considered as a part of a view. It is formed by an HTML file with all the markup necessary for that component, a Cascading Style Sheets (CSS) file (or in this case SCSS, since we are using Sass) with the styling rules for that component, and a TypeScript class with all the logic needed.

4.5.1.2 Pipes

Pipes are responsible for transforming the data they received as input in the desired output. There are already many pipes defined by Angular, for example, the pipe “date” transforms a JavaScript Date object into a human-readable date. Although, it is possible to define our own pipes if we want to define any particular data transformation.

4.5.1.3 Directives

Directives are extension over the existing HTML attributes which can be used to change the appearance or behaviour of an element.

4.5.1.4 Services

If the component needs any data from the back-end, we used what Angular calls Services, which mainly consists on TypeScript classes with a set of methods that asynchronously calls the Web API defined in the back-end to receive that data. These services are then passed through the components using the Dependency Injection mechanism provided by Angular.

4.5.1.5 Modules

A Module is a container that groups all the Components, Pipes, Directives and Services that are necessary to fulfil a feature. Usually, it is created one Module per view or feature which can then be lazily loaded, granting that they are only loaded when the user tries to access that view, allowing better memory management. This lazy loading feature is built-in in Angular but needs to be configured before using it.

The figure below shows a sequence diagram to clarify the communication between each element of the front and the back-end using, as an example, the action “Add CB User”.

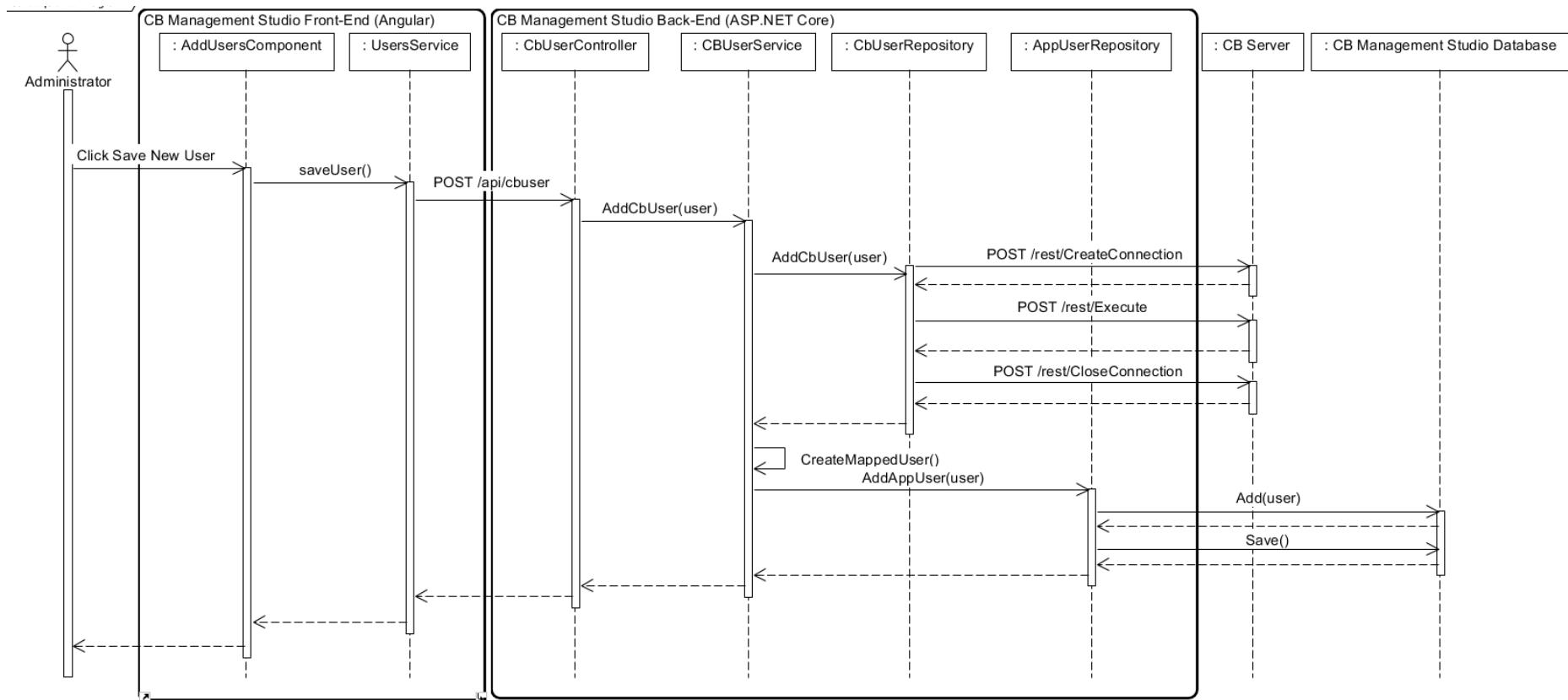


Figure 4.4 - Sequence Diagram (simplified) for the Use Case “Add CB User”

4.5.2 Usage of CSS Pre-processor

In large applications, CSS can be repetitive and difficult to maintain. A CSS pre-processor can be viewed as a scripting language used to extend CSS and solve this maintainability problem. For example, it is common when creating CSS to define a colour that is used in several elements of a Website. Without using a CSS pre-processor, if the developer decides to change that colour, he/she needs to go over all CSS rules and modify the colour in every rule that uses it. With a CSS pre-processor, the developer has the possibility of using variables which allows storing values like colours that are then replicated in different rules. Then if a change in the colour is needed, that change only needs to be done in a single place.

Nowadays there are two main CSS pre-processors: LESS and Syntactically Awesome StyleSheets (Sass).

For this project, Sass is being used. Sass was chosen instead of LESS since it is becoming more popular and the template used by the company (which is being reused) uses Sass.

Since the features included by Sass are not part of pure CSS, the browsers are not capable of interpreting them. Because of this, it is necessary to use Sass processors which are responsible for transforming the written .scss files into proper .css files.

By default, an Angular project uses standard CSS, although the framework already contains a Sass processor which makes the usage of Sass in an Angular project a matter of configuration.

4.6 Challenges

The sections above presented an overview of the development process and technologies used for the development of CB Management Studio. During this development process, a set of difficulties and challenges were faced which are described in this section.

4.6.1 Back-end

The primary role of CB management Studio back-end is to make a bridge between CB Server and the front-end.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

For some features, the information received/sent from the front-end to CB Server is simple and do not require any extra processing, which questions the necessity of having this back-end module. However, there are features that require data processing which does not make sense to be in the front-end. Also, there is a need to maintain a database that includes relevant information like mappings between CB Management Studio users and CB Server users, connections or history of executed queries which justify the existence of this back-end.

This database revealed to be a big challenge, especially regarding consistency.

CB Management Studio database consists mainly on a database mapped with the CB Server database, which means that when a CB User is created on a CB Server, a user must be created on CB Management Studio database and the same for the update and delete operations. Also, when a CB User receives access to a CB Account, a Connection needs to be added to the CB Management Studio database. The management of this kind of mappings is not trivial, and an error can be fatal for the correct behaviour of the application. For example, in the scenario described above, if the Connection is not created when the CB User receives access to an Account, that CB User will be impeded to use that Account to query a target system on CB Management Studio, despite having access to it.

As a developer aware of the importance of having clean, reusable and maintainable code combined with the inexperience with ASP.NET, it was a challenge to make sure that the created code followed the good practices and conventions for ASP.NET applications.

4.6.2 Replication of the Desktop look and feel

The development of a large Angular application is not a simple task. Angular is still in active development, and new third-party modules are continuously being released. Also, Angular by itself does not provide any ready-to-use components for common GUI elements, so everything needs to be created from scratch or obtained from third-party modules. This means that when compared with Windows Forms applications, it is difficult to find stable modules that perform some of the features we aim to implement.

The next sections will describe some of the most significant challenges we faced when replicating traditional Desktop features into Angular.

4.6.2.1 Present Tabular Data

Windows Forms application comes with a built-in component named DataGridView. This component is used to show tabular data like the results of a query. This component

is very stable and fulfils all the needs for presenting this kind of data including features like sorting the results presented or changing the size of the columns presented. Also it has an excellent performance when presenting large result sets.

HTML also has built-in the tag `<table>` designed exactly to create grids, although without any extra JavaScript these tables are very simple not allowing any of the features previously mentioned. Because of this, early in the development process, an Angular module to handle grid views has seen as essential.

In the initial stage, a module called `angular-datatables` [45] was selected. This seemed to be a good option since it was based on the project `DataTables` [46], a project based on jQuery started in 2007 that had all the features we were searching. Although, after some time we ended up understanding that this was not a good option, the Angular version of this jQuery library was reducing the performance of the application. Analyzing its source, we figured out that this library still uses jQuery for their core operations, which many times resulted in conflicts with Angular's internal mechanisms to manipulate the DOM.

After searching for alternatives, the project `ngx-datatable` [47] was found. It was a project developed entirely in Angular without any jQuery dependency. This combined with all the features it supports made us try it the project. Everything seemed to function well, and this library was used every time a table was needed. However, during the development of the query window and after testing it with complex queries which return more than 5000 rows and 20 columns, the browser got unresponsive and sometimes even crashed, forcing us to search for alternatives.

One possible solution for this problem would be the utilization of a pagination mechanism, although there are CB Connectors that do not support this mechanism and would force us to create our own pagination mechanism on the CB Management Studio back-end project. Therefore, we decided to search for other front-end alternatives.

It was decided to create our own grid module, but quickly we found that even a simple HTML table with this amount of data would not be easily rendered by the browser, especially if it is in a page with a complex DOM (i.e., a DOM with many elements). Based on this, we kept searching, but now, focusing our attention on rendering performance.

As a result of all this process, we found a library that fulfils many of our needs, `ag-grid` [48]. This library is written entirely in pure JavaScript (containing versions for the main JavaScript frameworks, including Angular) and provides outstanding performance when presenting large datasets. Additionally, `ag-grid` includes a broad set of additional features, which includes the ones we were looking. It was necessary to refactor all the views

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

that contained tables to apply this new module, although it granted a much more performant solution when presenting tabular data.

However, ag-grid presents some limitations. To increase the rendering performance of the tables presented in the query window, it was necessary to simplify the DOM of this page, this was achieved by reducing the number of elements of it.

Moreover, ag-grid contains premium features which are included in a pack named ag-grid Enterprise[49]. One of those features was the copy, which was an essential feature for the tables presented in this application. This feature allows the copy of values from the table cells into the clipboard. Therefore, and due to high prices associated to the premium version of ag-grid, it was decided that in the future the source code for the free version of ag-grid would have to be modified in order to support it.

4.6.2.2 Tree View

When developing using Windows Forms it is easy to create an element that shows hierarchical data in a tree format; the developer only needs to use the TreeView control that is built-in at Windows Forms library.

For Angular, there are not many alternatives. The modules angular-tree-component [50] and ng2-tree [51] were evaluated and tested. The decision fell back over ng2-tree since its API was easier to understand and modify. Some features and bugs in the library had to be added and solved, these were related with node expansion when double clicking and with improvements on the position of context menus. In order to do this, the source code of this module was taken from GitHub and was modified, in particular, an event handler for double clicking on a node was created and changes on the CSS related with the context menu were performed.

4.6.2.3 Context Menus

Context Menus are very common on desktop applications, although on Web they are not so common mainly because the browser itself already contains its context menu. Nevertheless, it is possible, using JavaScript, to prevent the Context Menu of the browser to be presented and instead, present an element similar to a context menu that has the same purpose.

The main difficulty creating context menus in this kind of applications appears when a developer needs to add a context menu to a third-party component that does not support this type of interaction, forcing the developers to access the source code and modify the

component to support the context menu. Since this is not very common on Web, most of them do not support it. In this project, this process was followed in the Tabs Component.

As occurred with the module used to create the Connections Tree View, in order to add a context menu to the Tabs Component, we started by getting the source code of it from GitHub. Then, a new Angular Component, with the same look and feel of a context menu was created which was afterward added to the Tab Component. Particularly in the component, it was necessary to create an event handler for the mouse left click that shows the Context Menu created instead of the browser's default menu.

4.6.2.4 Keyboard Shortcuts

Like Context Menus, Keyboard Shortcuts are also a regular presence on Desktop Applications but not on Web. This was another not so common feature that was implemented on this project. The implementation of a shortcut mechanism was not too difficult since JavaScript (and Angular) offer support for keyboard events. The primary challenge consisted in granting that a keyboard shortcut event is not triggered when that shortcut does not make sense (for example, the shortcut used to navigate to Accounts window, should be ignored if the dialog to Edit Users is being shown). This was solved analysing the DOM to understand if there was any dialog being shown.

4.7 Setup Installer

After finishing the development of CB Management Studio, we started preparing the build of the first release. The Administration Tool and Query Analyzer are installed together with the Connect Bridge Platform using the same setup installer. Considering that CB Management Studio will replace these two tools, the existing setup installer was modified to install CB Management Studio instead of Administration Tool and Query Analyzer. In order to accomplish that, it was necessary to perform a set of different tasks. This included the configuration of ASP.NET and Angular projects to be built in Release/Production modes, which optimizes and minifies all the files; the mapping of the created files in the corresponding installation directory; the configuration of CB Management Studio to run as a Windows Service; and the generation of a self-signed Secure Socket Layer (SSL) certificate for HTTPS communications with CB Management Studio.

For the development of the setup installer Connecting Software uses a software called Advanced Installer [52] which turns the creation of installers a simple process. This

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

software provides a set of features like the definition of product details and the structure of files that will be created on the machine that uses the installer, the configuration of the setup dialogs or the configuration of Windows Services. Also it can be extended with custom actions coded in several languages including C#.

4.8 Documentation

To document the API that feeds the Angular application, the library Swashbuckle [53] was added to the ASP.NET Core project which automatically creates a Swagger¹⁸ based documentation of the API using the comments written on the Web API project. Figure 4.5 shows the page that is generated by swagger with all the available routes.

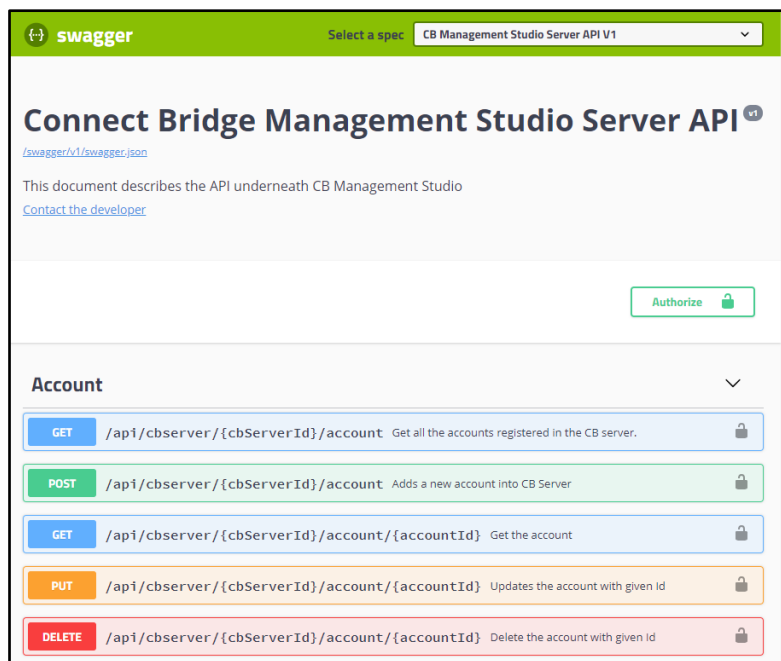


Figure 4.5 - Generated Swagger Documentation Page

To document the Angular application code, a similar approach was used, in this case using the library Compodocs¹⁹, that automatically generates HTML files describing all the modules, components, directives and classes used by the Angular application based on the

¹⁸ Open source specification for RESTful APIs (recently renamed to OpenAPI Specification). Includes tools to design, build and document RESTful APIs (see more at <https://swagger.io/>)

¹⁹ <https://compodoc.app/>

comments written on the code. Figure 4.6 shows one page that was generated by this mechanism.

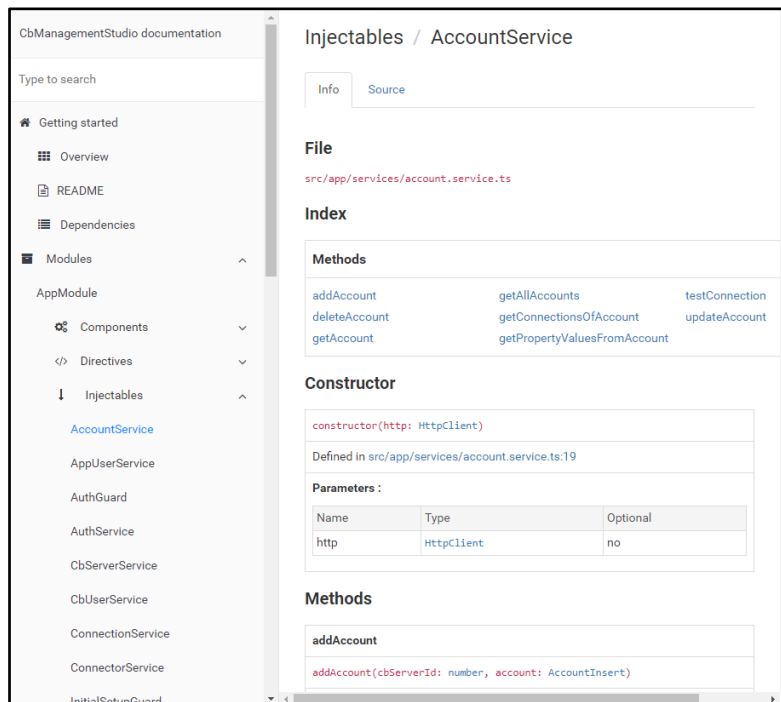


Figure 4.6 - Example of a page generated by Compodocs

During the development, some reports were made describing the progress made, and in the final phase of the project, a user manual was written explaining how to use CB Management Studio (see Figure 4.7). All these documents were written using Microsoft Word and followed a template defined by Connecting Software to ensure consistency in all its documentation.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

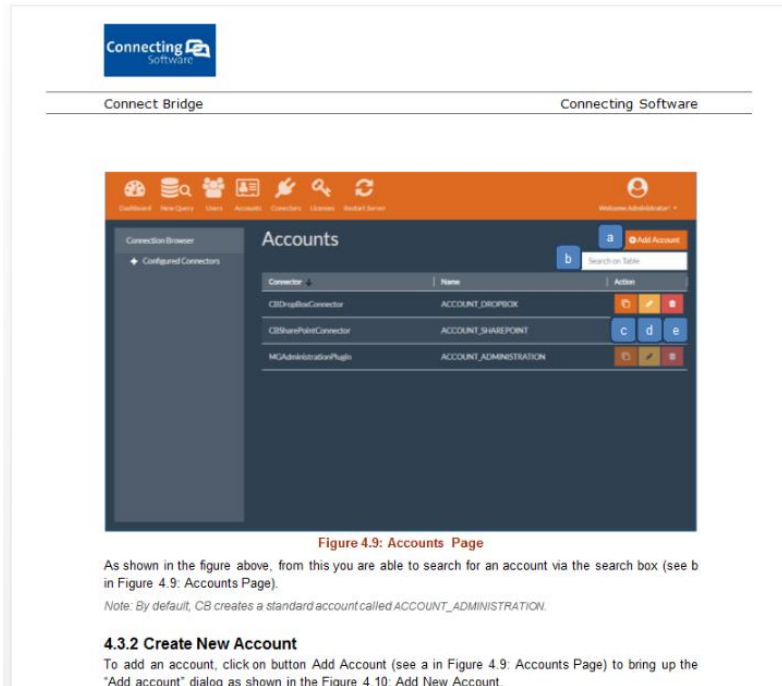


Figure 4.7 - Connect Bridge Management Studio User Manual

5

EVALUATION

In this chapter, it is presented a critic evaluation of the developed solution. For this evaluation, a comparison between CB Management Studio and its predecessors is exposed followed by the results of a user evaluation that was done using the Think Aloud technique [55].

5.1 Comparison between CB Management Studio and its predecessors

CB Management Studio in its current stage is already capable of replacing the Administration Tool and Query Analyzer in their core features. In the next sections, a comparison between these tools and CB Management Studio is presented.

5.1.1 Administration Tool vs. CB Management Studio

The following sections will compare the Administration Tool and CB Management Studio.

5.1.1.1 Navigation

In Administration Tool, the navigation between the different manageable entities (Users, Accounts, Groups, Connectors, Licenses) was made recurring to a Treeview on the left side of the window (see Figure 5.1). In CB Management Studio this was replaced for a navigation bar on the top (see Figure 5.2). This decision was inspired mainly by Navicat

Connect Bridge Management Studio:

A web-based application used to manage and test Connect Bridge servers

(see section 2.2) and on modern web dashboards, leaving the left side available for the treeview used to show the available connections to the target systems on the Connection Browser (see section 5.1.2.1).

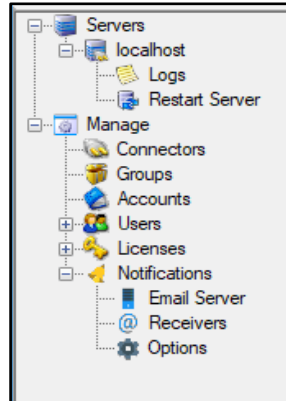


Figure 5.1 - Navigation Tree on Administration Tool

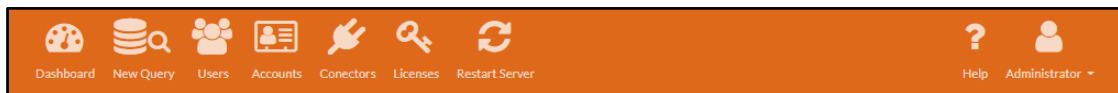


Figure 5.2 - Navigation Bar on CB Management Studio

5.1.1.2 Management of CB Entities (Users, Groups, Accounts, Connectors, Licenses)

The management of CB entities (Users, Groups, Accounts, Connectors, Licenses) on CB Management Studio remains very similar to the Administration Tool. On the old application, for each of these entities, a Grid is shown on the central part of the window with all the registered entries for that entity. The available actions over that entities are available through buttons on the right side of the window (see Figure 5.3).

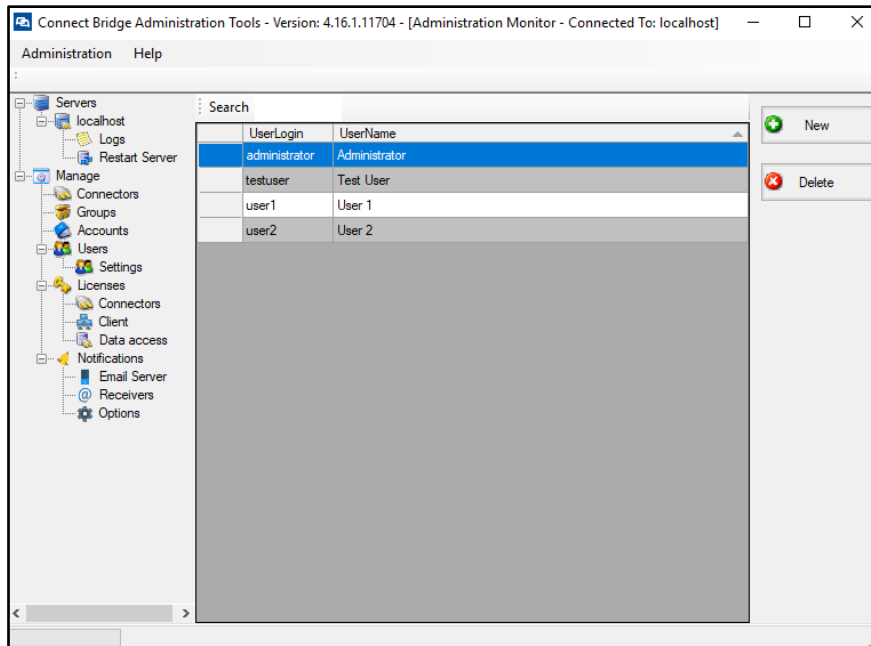


Figure 5.3 - Management of CB Users on Administration Tool (List of CB Users)

CB Management Studio presents a similar approach, presenting the entries using a table with the difference of having a column named “Action” that contains the available actions for each entry. For example, for the management of CB users on CB Management Studio, the registered users are listed on a table similar to the one used by Administration Tool with the difference that now the available actions (Edit and Delete) can be accomplished recurring to a button that is present on each row of the table (see Figure 5.3).

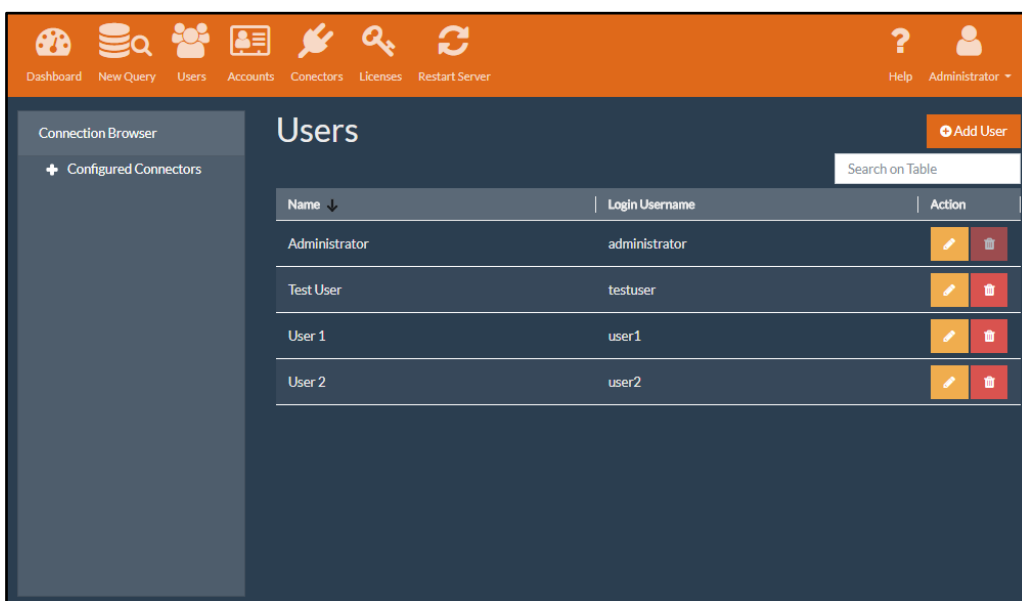


Figure 5.4- Management of CB Users on CB Management Studio (List of CB Users)

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

The dialog used to add/edit users is also very similar, although, it is important to highlight a difference. In Administration Tool it was possible to associate a User to a Group (Groups was an association between multiple users and multiple accounts). In CB Management Studio the concept of Group was removed, and now the Users are directly associated with Accounts (see Figure 5.5 and Figure 5.6).

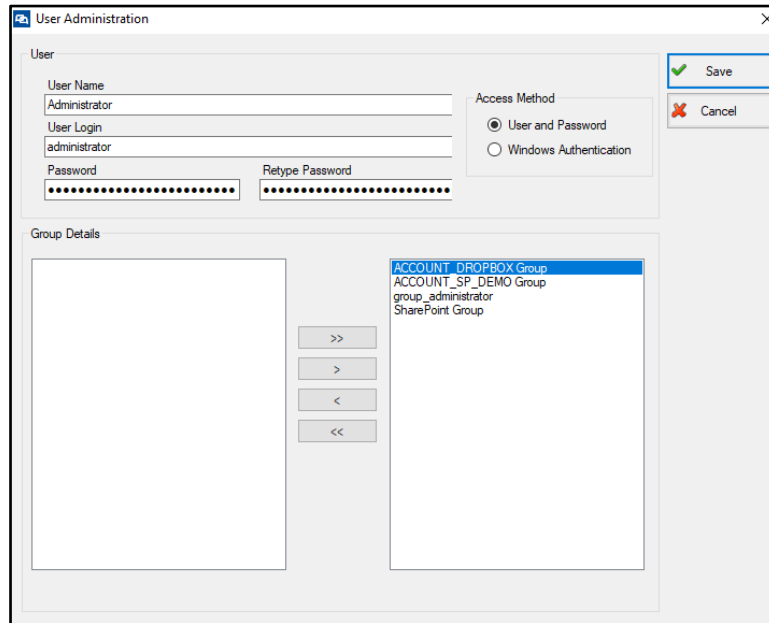


Figure 5.5- Management of CB Users on Administration Tool (Edit CB User)

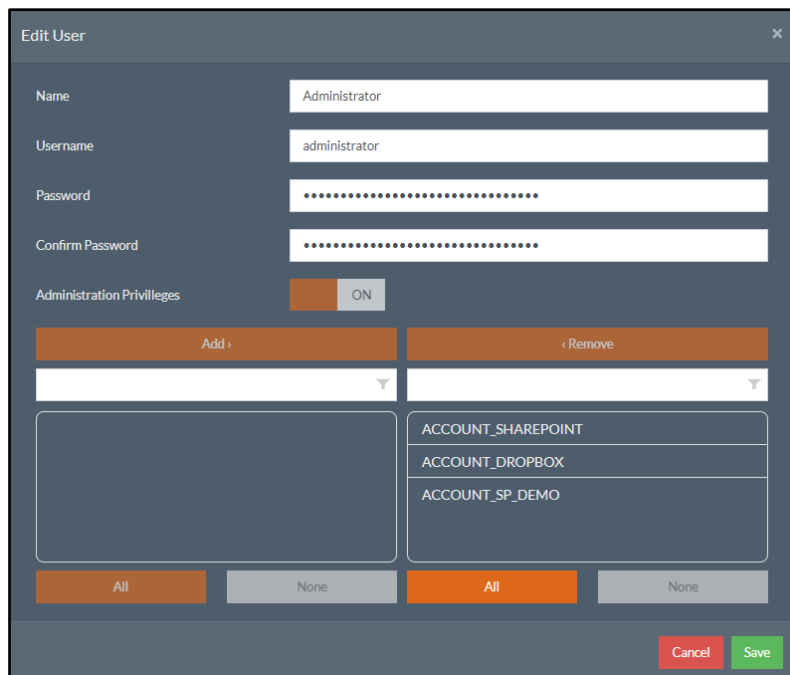


Figure 5.6- Management of CB Users on CB Management Studio (Edit CB User)

Other difference available on CB Management Studio is the list of Licenses. Now it is possible to check the licenses that are active and expired (see Figure 5.8), while on Administration Tool was only possible to check the active Licenses (see Figure 5.7).

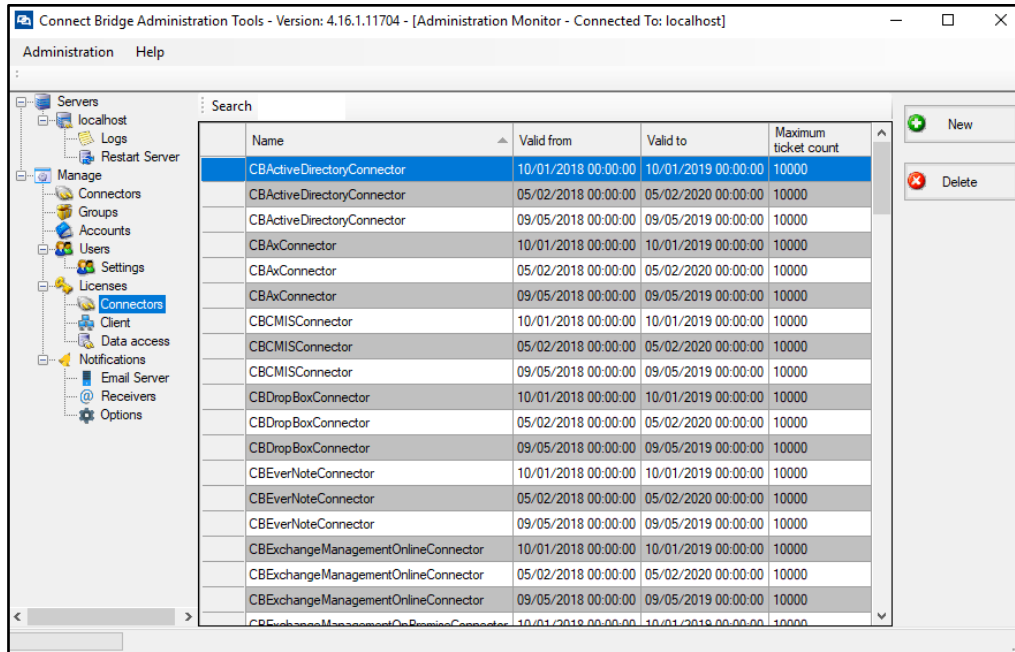


Figure 5.7 - List of Licenses on Administration Tool

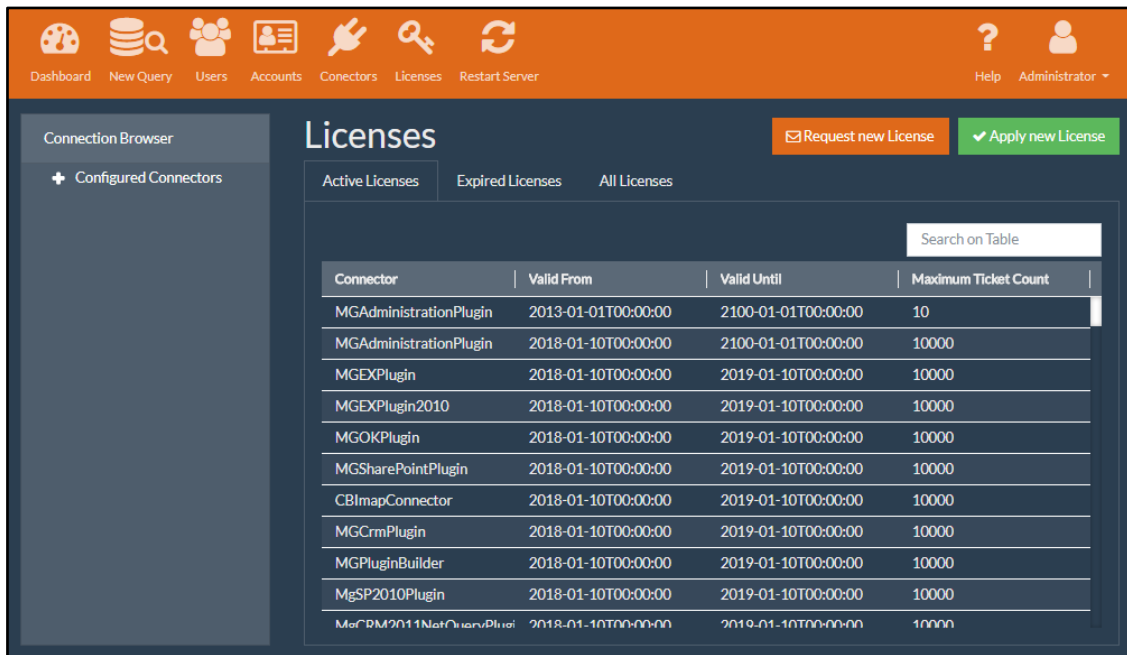


Figure 5.8 - List of Licenses on CB Management Studio

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

5.1.1.3 Management of Logs

In the Administration Tool, it was also possible to manage logs, defining the granularity of those, but was not possible to define that per user (see Figure 5.9). In CB Management Studio that possibility was added (see Figure 5.10). It is also possible to use CB Management Studio to see the logs without having to access to CB Server filesystem to check the logs files or the CB Service Controller. The logs that are being presented can also be exported into a text file (see Figure 5.11).

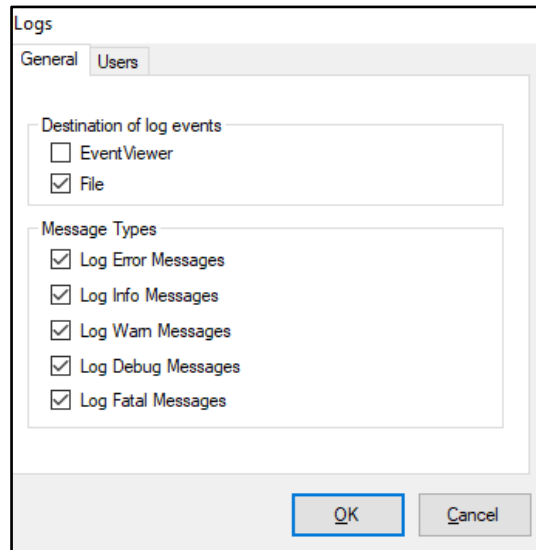


Figure 5.9 - Logs Configuration on Administration Tool

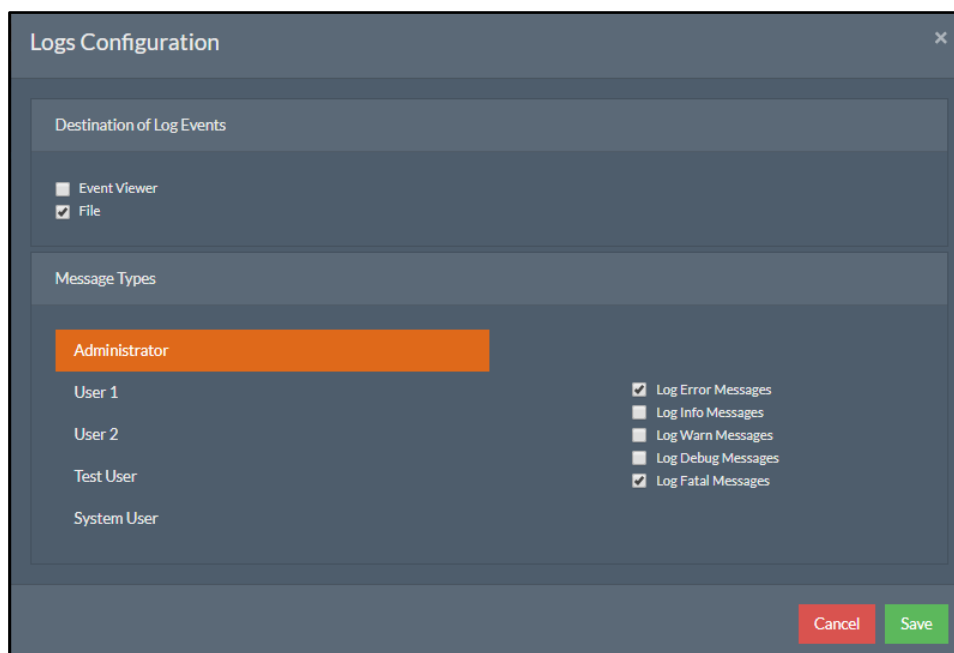


Figure 5.10 - Logs Configuration on CB Management Studio

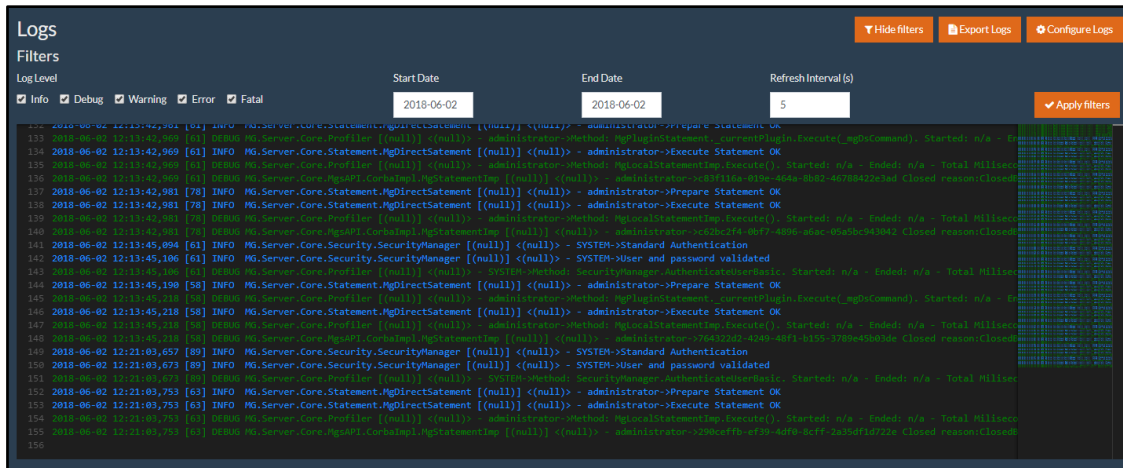


Figure 5.11 – Visualization of logs on CB Management Studio

5.1.2 Query Analyzer vs. CB Management Studio

The following sections will compare Query Analyzer and CB Management Studio.

5.1.2.1 Connection Browser

The connections with the target systems, on Query Analyzer are represented in a treeview called Connection Browser (see Figure 5.12). All the connections are created manually by the users. This means that for each CB Account previously configured on Administration Tool, it was required to configure a connection for that account on Query Analyzer, repeating this process for all the users that had access for that account.

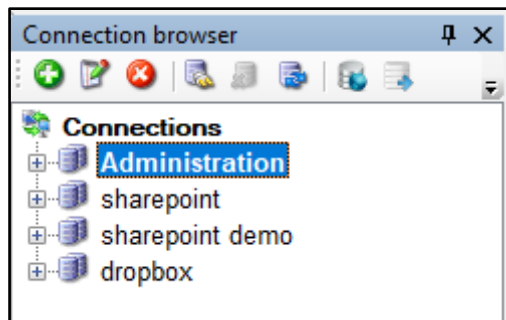


Figure 5.12 - Connection Browser on Query Analyzer

CB Management Studio removed this step. Now, when a user logs into the application, the Connection Browser is automatically filled with a default connection for each account he has access.

Also, the organization of the treeview was changed. Before, all the connections were presented without any organization. Now, to help users quickly identify which

Connect Bridge Management Studio:

A web-based application used to manage and test Connect Bridge servers

connections are available for each connector and account, the treeview follows the organization Connector-Account-Connection (see Figure 5.13).

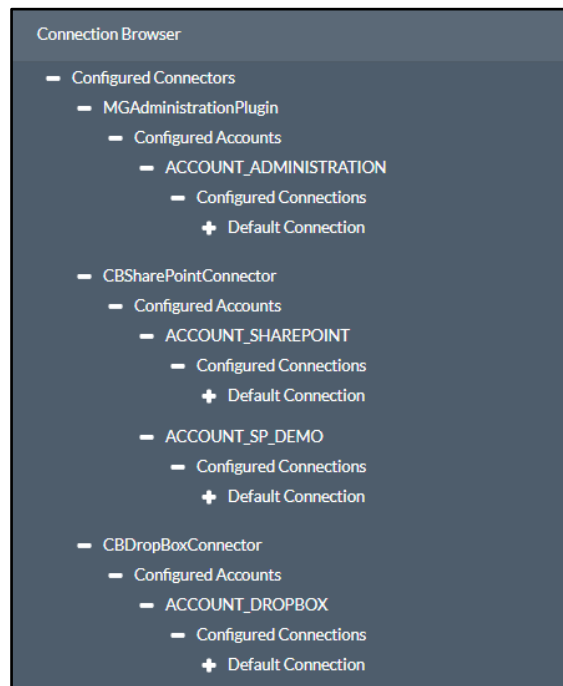


Figure 5.13 - Connection Browser on CB Management Studio

A Connection String is a description of the parameters necessary to establish a connection with a target system using the drivers available for CB Server. Therefore, it is mandatory for the users to include a Connection String on their code to develop a custom integration solution using Connect Bridge.

Each connection shown on the Connection Browser has a corresponding Connection String. With Query Analyser it is possible to extract Connection Strings for the ODBC driver, but it is not possible to access any Connection String for the other drives (i.e., JDBC driver and Web Service driver) (see Figure 5.14). CB Management Studio resolves this issue as can be seen in Figure 5.15.

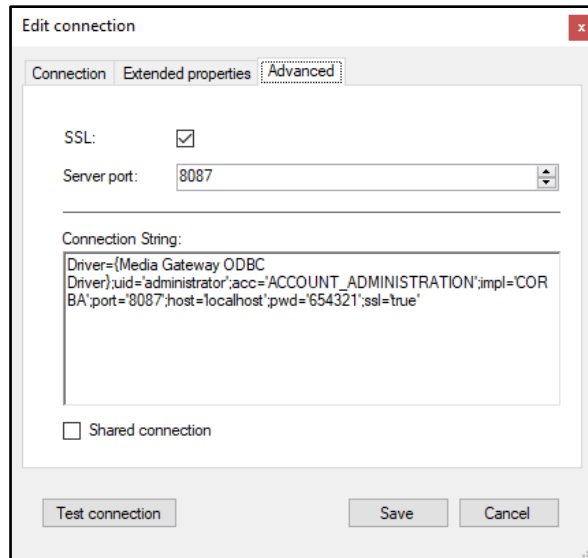


Figure 5.14 - Connection String generated by Query Analyzer

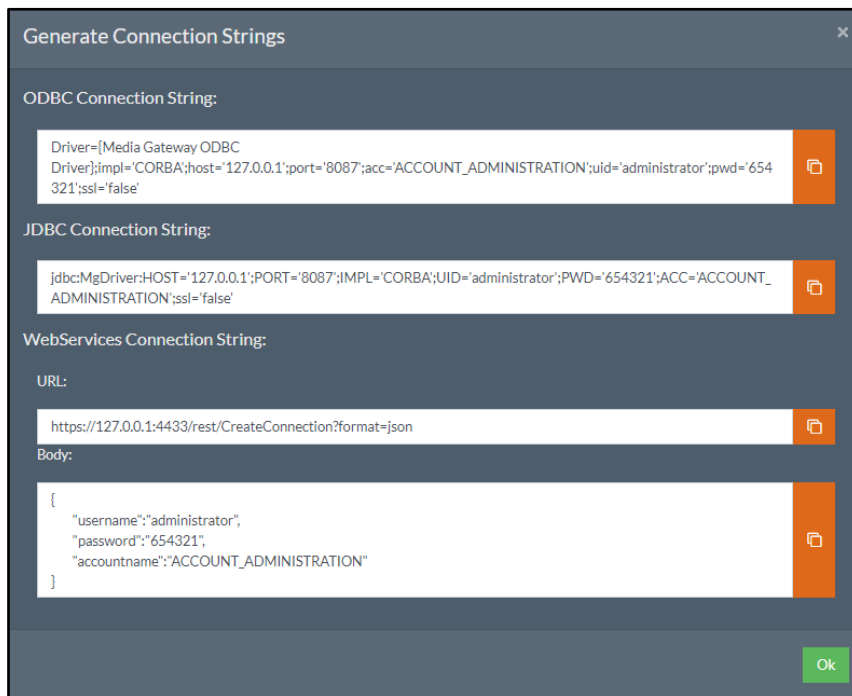


Figure 5.15 - Connection Strings generated by CB Management Studio

Query Analyzer allowed users to create additional connections using the same account, using the Extended Properties. This feature enables the override of the properties defined in an account. This feature continues being supported by CB Management Studio.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

5.1.2.2 Execution of Queries

The execution of queries on CB Management Studio maintains all the features available on Query Analyzer, including the support of parameterized queries (see Figure 5.16 and Figure 5.17) and the auto-generation of queries (see Figure 5.18 and Figure 5.19).

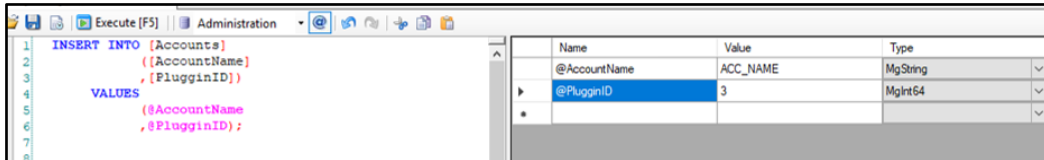


Figure 5.16 - Parameterized Query on Query Analyzer

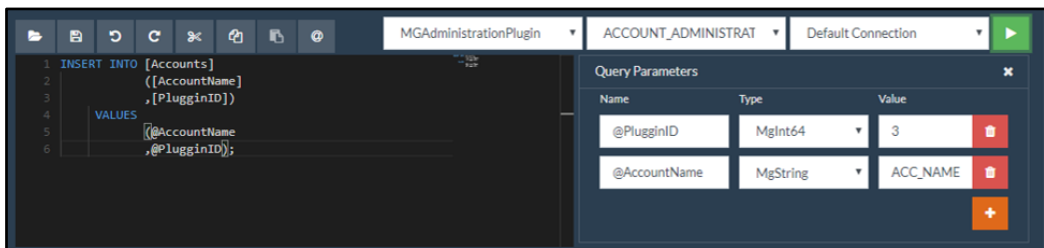


Figure 5.17 - Parameterized Query on CB Management Studio

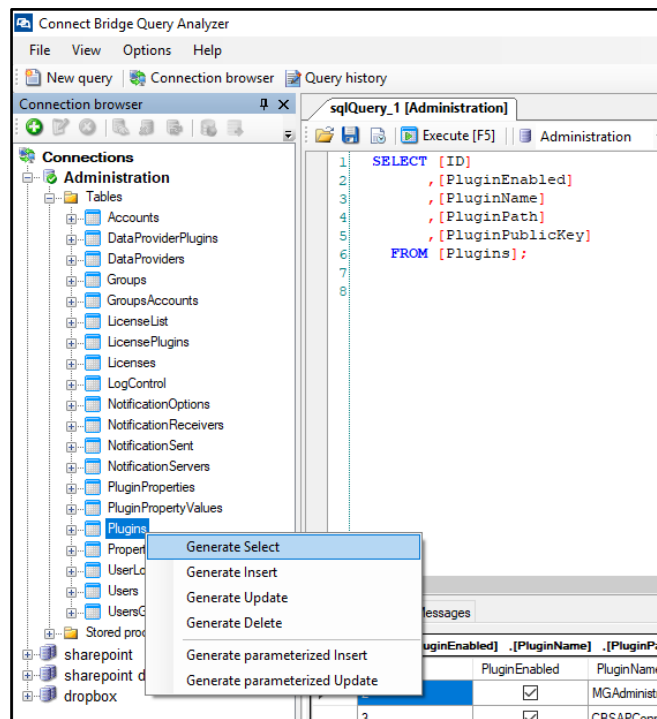


Figure 5.18 - Auto generation of queries on Query Analyzer

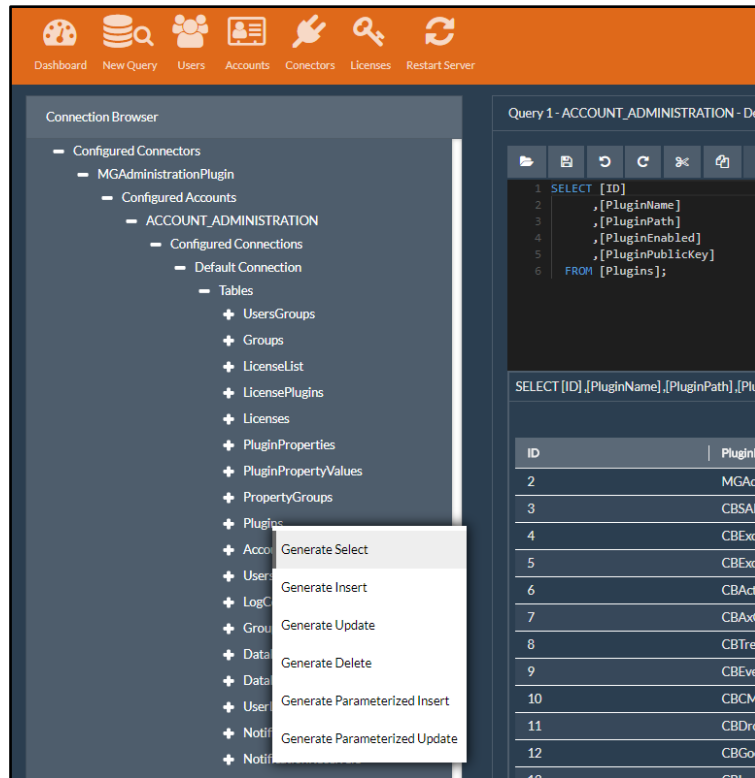


Figure 5.19 - Auto generation of queries in CB Management Studio

The text editor used by CB Management Studio is Monaco, the same editor used by Visual Studio Code [56], [57]. The usage of this editor allowed the implementation of a powerful text editor that includes the possibility to develop syntax highlighting, code suggestions, and completion mechanisms (for this project, and for now, only for SQL keywords and functions were developed) as can be seen in Figure 5.20 and Figure 5.21.

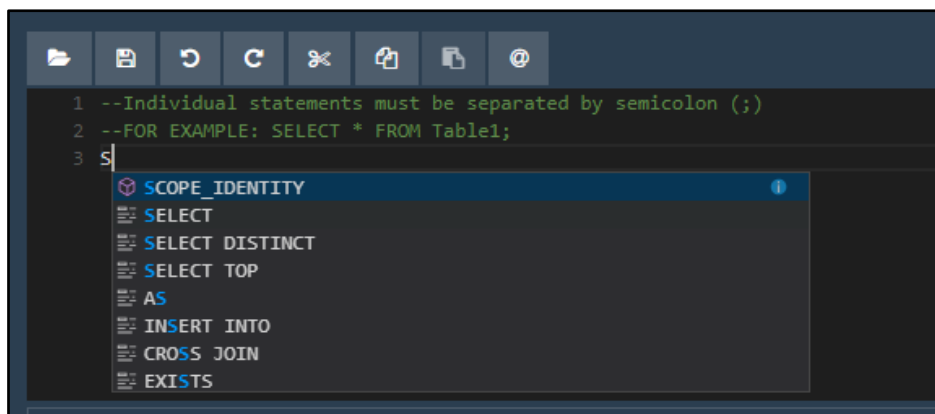
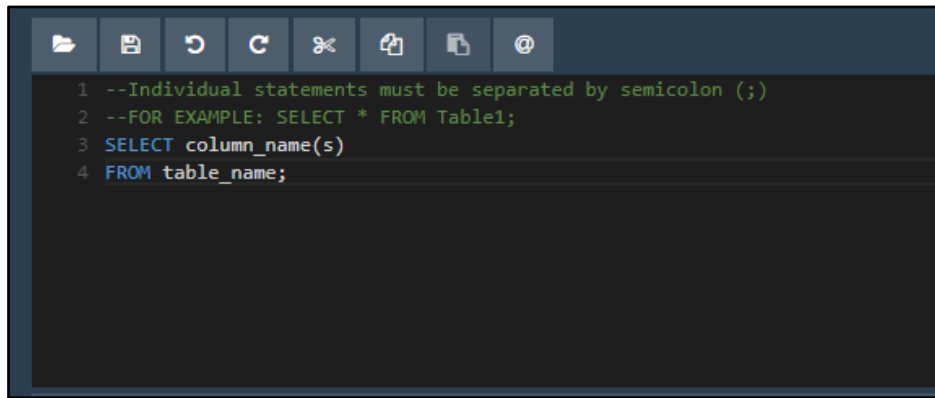


Figure 5.20 - Code suggestions on CB Management Studio query editor

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers



```
1 --Individual statements must be separated by semicolon (;)
2 --FOR EXAMPLE: SELECT * FROM Table1;
3 SELECT column_name(s)
4 FROM table_name;
```

Figure 5.21 - Auto-completion of code on CB Management Studio query editor

5.2 Evaluation with Users

After finishing the development and the formal comparison presented in the previous section, it was also important to evaluate the usability of the front-end, for this purpose an evaluation with users was necessary. The usability was not the most significant focus of this project, although, improving it would be a pleasant enhancement. To perform this evaluation, we used a methodology called “Think-aloud”.

Think aloud is a usability testing that allows receiving feedback of an interface while the user is interacting with it. In this technique, the users are requested to verbalize their thoughts while performing a set of predetermined tasks, allowing us to understand their difficulties when performing these tasks and what caused these difficulties [58], [59].

The think-aloud experiment was conducted with 4 participants. All of them were elements of the company and were familiar with the usage of the old tools. The selection of this users was preferred over people that never used Connect Bridge since they are already familiar with all the Connect Bridge concepts necessary for the correct usage of the CB Management Studio and would allow getting feedback about how this new system is compared with the old ones. This also grants that the users can focus entirely on the user interface and not in details about Connect Bridge Platform itself.

The users were requested to perform a total of 9 tasks that are representative of the major features of CB Management Studio. These tasks were selected to evaluate the biggest changes introduced by this new tool and, for the views that remained similar to the old tools, it intended to evaluate if there were any improvements that could be done. The tasks are ordered by difficulty level in order to increase the confidence of the user while advancing in the test. Each task needs to be performed initially on Administration Tool + Query Analyzer and then on the new CB Management Studio. The protocol used for the

Think-Aloud session can be consulted on Appendix B - Think-Aloud Protocol. The following list describes the tasks performed by the participants when applying this technique:

1. Login into the application
 - In this task, the participants had to login into the application using the credentials provided;
 - It aimed to test the receptiveness of the users to the new login page of CB Management Studio;
2. Check the active licenses on CB Server
 - In this task, the participants had to visualize the licenses that are active on a CB Server;
 - It aimed to test how if having the licenses divided by groups (Active, Expired, All) on CB Management Studio helps when performing this task;
3. Disable/Enable the Dropbox connector on CB Server
 - In this task, the participants had to enable or disable a connector on CB Server;
 - It aimed to test if the addition of a button to perform this task directly on the connectors' table was a right decision, and to test if the icon used for this button was the most appropriated;
4. Create a new Account using the Exchange connector (ACCOUNT_EX)
 - In this task, the participants had to create a new CB Account using the configuration provided on the protocol;
 - The dialog window used on CB Management Studio to perform this task is a copy of the existing in Administration Tool, then the goal of this task was to understand if the participants had any suggestion to improve it;
5. Allow the user Administrator to access the ACCOUNT_EX
 - In this task, the participants had to permit the user Administrator to access the account created during the previous task;
 - It aimed to test the most significant conceptual change on the administration of CB Servers, the removal of the Groups used to associate users and accounts. With this task, we are evaluating if the participants agree with this change and if placing this association configuration in the Users edit/add dialog was the appropriated decision;

Connect Bridge Management Studio:

A web-based application used to manage and test Connect Bridge servers

6. Create a new user with Administration Privileges and with access to the ACCOUNT_EX
 - In this task, the participants had to create a new CB User with Administration Privileges and with access to the ACCOUNT_EX;
 - As happened with the Task 4, this task aimed to understand if the participants had any suggestion to improve it;
7. Execute a SELECT on the table Inbox using the ACCOUNT_EX
 - In this task, the participants had to execute a SELECT statement on a table of the ACCOUNT_EX;
 - This task required different actions from the participants, while on Query Analyzer the participant must initially create a connection using this account. On CB Management Studio this account is automatically created by default when the association between the user and the account is done;
 - This task aimed to evaluate the reaction of the participants about not having to create this connection and also the way the results are presented;
8. Execute the Stored Procedure SP_UPLOAD_MESSAGE_MSG
 - In this task, the participants had to execute the Stored Procedure SP_UPLOAD_MESSAGE_MSG on the ACCOUNT_EX;
 - To execute this stored procedure, the participants must pass an e-mail file as a parameter. The process to add this file as a parameter is similar in both tools. Although, CB Management Studio shows a button to add this file, while on Query Analyzer this is done recurring to a double-click on a textbox;
 - This task aimed to evaluate the participants' reaction about having this button to add a file as a parameter of a statement;
9. Get the connection string of a connection that uses the ACCOUNT_EX
 - In this task, the participants had to obtain a connection string for a connection of ACCOUNT_EX;
 - This task aimed to evaluate other significant change in CB Management Studio, the organization of the connection browser which was already discussed in section 5.1.2.1;

After the execution of all the tasks, the users are asked to fill out a questionnaire which is divided into two sections (see Appendix C- Survey used after think-aloud).

The first one consists of an evaluation of the difficulty faced when performing each of the tasks realized during the Think Aloud session, this consists on a set of 9 questions, one for each task, where the user must classify the execution of the task in each tool using a 5-Likert-scale with the options: 1-Very difficult, 2-Difficult, 3-Moderate, 4-Easy, 5-Very Easy.

The second section consists of an overall evaluation of Connect Bridge Management Studio, for that, we are using the test called System Usability Scale (SUS) [60]. This consists on a test with ten questions, each one with a 5-Likert-scale with the options: 1-Strongly disagree, 2-Disagree, 3-Neutral, 4-Agree, 5-Strongly Agree. This test was selected since it is a consistent and verified tool for measuring the usability of a system, with references in over 1300 articles and publications [61]. The survey finishes with a question that aims to collect which was, in general, the interface the participant preferred, and an open question for the participants that might want to write additional comments about CB Management Studio (see Appendix C).

5.2.1 Results of Thinking Aloud Technique

All the participants were able to complete the tasks that were proposed in both systems. We will analyse individually each task combining the feedback received during the think-aloud process with the survey used after applying this technique.

5.2.1.1 TASK 1: Login to the application.

This task was straight-forward for all the participants. Anyone of them weaved any comments about this particular task. In the survey, the 3 participants considered the task Very Easy, and 1 considered the task as Easy in both systems. None of the participants changed their opinion when comparing the answer given for the Administration Tool and CB Management Studio (see Table 5.1).

Table 5.1 - Responses to the question “How easy was to perform the Task 1?”

	Administration Tool	CB Management Studio
P1	Easy	Easy
P2	Very Easy	Very Easy
P3	Very Easy	Very Easy
P4	Very Easy	Very Easy

Connect Bridge Management Studio:

A web-based application used to manage and test Connect Bridge servers

The results obtained were expected. Both Interfaces are very similar regarding interaction and consist of a simple login form that all the users are already familiar.

5.2.1.2 TASK 2: Check the active licenses on CB Server

In the Administration Tool, all the participants initially click on the Licenses node of the tree view instead of expanding it and click on its children node called Connectors (see Figure 5.22).

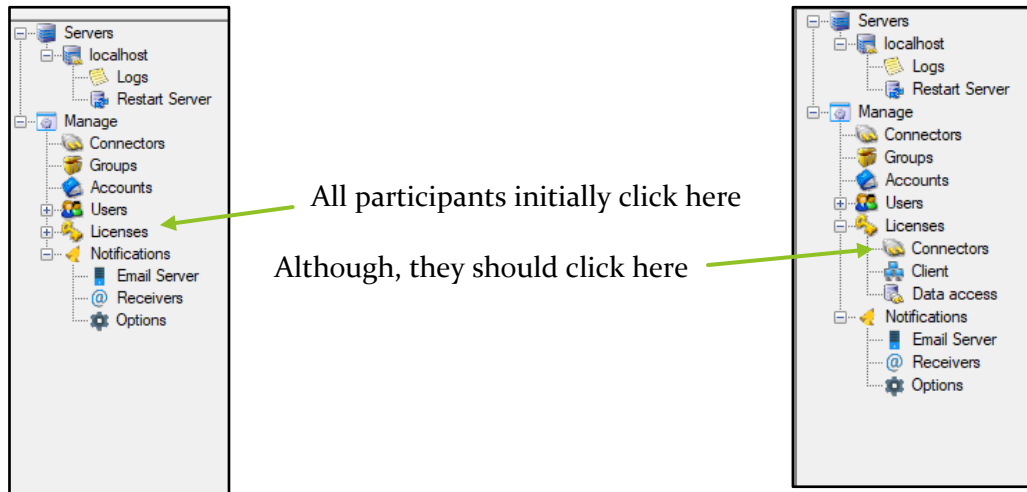


Figure 5.22 - Common error performed by the participants in Task 2, using Administration Tool

In CB Management Studio, all participants went directly to the page that shows the active licenses. The overall reaction was that this task was more straightforward in the new application. For example, participant P2 said “*I think it is better now because you can immediately see the active licenses*” and participant P4 referred “*Much easier*”. Participant P1 founded a consistency problem: “*The Dates should not be presented like this, it should be presented like CB shows when returning queries results. I mean 2010-01-01 00:00:00.000, and not with a T separating the date from the time*”. The Figure 5.23 shows the aspect referred by P1.

Active Licenses	Expired Licenses	All Licenses	
Search on Table			
Connector	Valid From	Valid Until	Maximum Ticket Count
MGAdministrationPlugin	2018-05-14T00:00:00	2100-01-01T00:00:00	10000

Figure 5.23 – Inconsistent date presented on Licenses Page

Analysing the results of the survey, it was possible to see that one participant (P2) considered this task as Easy on Administration Tool and Very Easy in CB Management Studio, while all the others give the same evaluation on both systems (see Table 5.1).

Table 5.2 - Responses to the question “How easy was to perform the Task 2?”

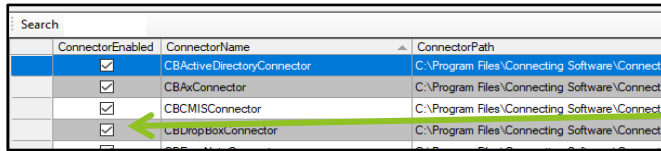
	Administration Tool	CB Management Studio
P1	Easy	Easy
P2	Easy	Very Easy
P3	Very Easy	Very Easy
P4	Easy	Easy

The error that all participants made on Administration Tool was already detected, when analysing the usability problems of the tool, because of this, the design of the Licenses View of CB Management Studio was changed to show the active licenses immediately when clicking over licenses.

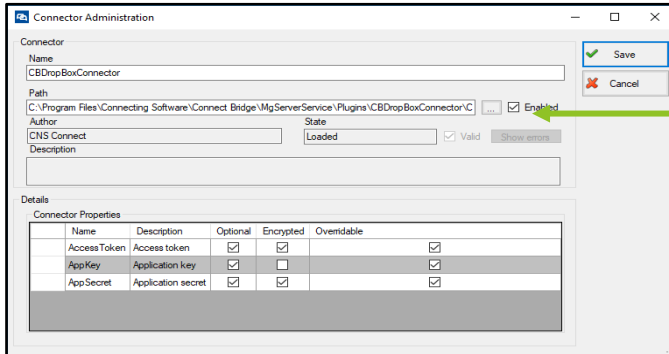
5.2.1.3 TASK 3: Disable the Dropbox connector on CB Server

In the Administration Tool, participants P3 and P4 expected that they could change the state of the connector from Enable to Disabled by unchecking the checkbox presented on the column ConnectorEnabled on Connectors View. Although, to perform this task the users need to open the edit connector dialog and uncheck the Enabled Checkbox (see Figure 5.24). About this, after realizing the correct way, participant P3 said: *“Really?!? This does not make sense”*.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers



Participants try to disable the connector unchecking this



Although, they should double-click on the connector and uncheck this

Figure 5.24 - Common error performed by the participants in Task 3, using Administration Tool

The participants easily performed the same task on CB Management Studio; participant P2 said: *“This way is much faster”*. Participant P3 said that the button used to disable/enable connectors is not very intuitive, then he suggested to remove the column Enabled and replace this button with a toggle button, giving the sense of On/Off. From Participant P4, it was understood that the icon used on the button could be misleading and he suggested to use an icon representing On and Off. The Figure 5.25 shows the aspect of the interface that was criticised by both participants.



Figure 5.25 - Misleading button on Connectors Page

From the survey, we checked that participant P3 evaluated the task as Moderate in Administration Tool and as Easy on CB Management Studio and that P4 considered the task as Moderate on the old application and Very Easy on the new application. Participant P1 considered the task as Moderate in both systems and participant P2 considered the task as Very Easy on both tools (see Table 5.3).

Table 5.3 - Responses to the question “How easy was to perform the Task 3?”

	Administration Tool	CB Management Studio
P1	Moderate	Moderate
P2	Very Easy	Very Easy
P3	Moderate	Easy
P4	Moderate	Very Easy

The fact of having the possibility to change the state of the connector was included on the table of Connectors View, exactly to simplify this task. Although, the feedback received from P3 and P4, showed that the button used maybe is not the perfect solution, in fact, this was also a topic that the Product Owner was aware before, since for him, the icon was also confusing.

5.2.1.4 TASK 4: Create a new Account using the Exchange connector, named ACCOUNT_EX.

All the participants completed the task without leaving any comments. Analysing the questionnaire, we can see that only participant P1 considered that performing this task on CB Management Studio was worst (Moderate) than in the Administration Tool (Easy). All the other participants considered similar difficulty (see Table 5.4).

Table 5.4 - Responses to the question “How easy was to perform the Task 4?”

	Administration Tool	CB Management Studio
P1	Easy	Moderate
P2	Very Easy	Very Easy
P3	Very Easy	Very Easy
P4	Easy	Easy

The response of P1 was surprising, considering that the way to perform this task in both systems is very similar. One thing that might justify his opinion is the time needed to load the connector properties, which noticeably takes longer on CB Management Studio.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

5.2.1.5 TASK 5: Allow the user Administrator to access the ACCOUNT_EX

In order to perform this task on the Administration Tool, the participants need to add the ACCOUNT_EX to the Group “group_administrator”, or create a new group containing that account and assign the user Administrator to that group.

Participant P1 immediately went to the Users, opened the user Administrator and then realised that this change needs to be done on Groups. Then he went to that view and correctly concluded the task.

Participant P2 initially went to Users but then realised that this task needs to be done on Groups and correctly perform the task without leaving any comments.

Participant P3 initially went to the Accounts and tried to get a way to associate the user Administration with the account ACCOUNT_EX. Then he went to Users and only after that he went to Groups. In Groups, he said that he could not easily understand which side of the dual list contains the accounts that a group does not have access (see Figure 5.26). When finishing the task, he referred: “Associate users and accounts through groups do not say anything to me... I only know this because I did this a couple of times before...”.

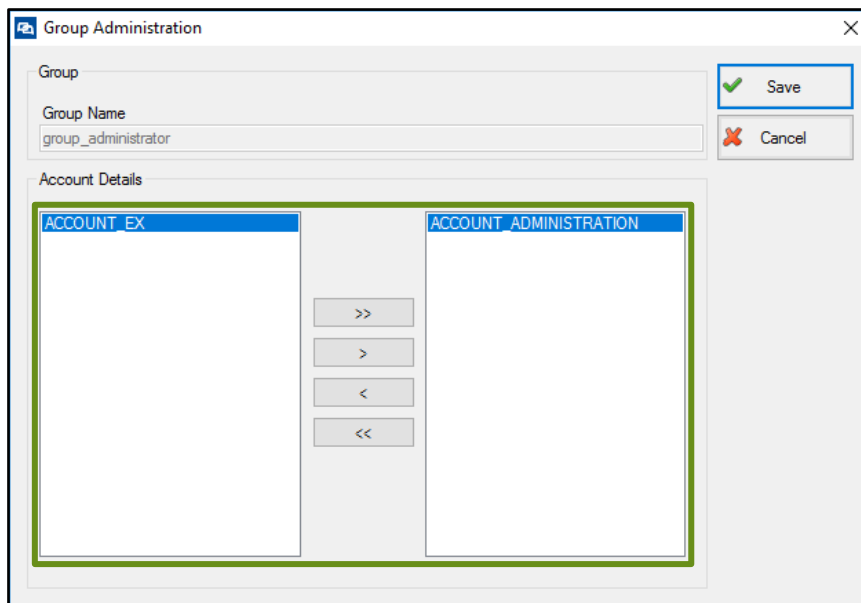


Figure 5.26 – Dual List used to associate accounts with users on CB Administration Tool

Participant P4 also went to the Accounts view in the beginning, but then he remembered that this action is done through Groups and correctly associated this account with the group_administrator.

When performing the task on CB Management Studio, participant P₁ primarily clicked on Users; then he thought that this association should be made through the Accounts page but then returned to Users and correctly performed the task.

Participant P₂ concluded the task without any mistake; he also referred: *“I like the possibility of using the mouse to drag the accounts instead of clicking on the buttons to move them between the lists”*.

Participant P₃ correctly performed the task but pointed that the dual list used by CB Management Studio has the same problem as the ones existing on Administration Tool, it is not clear what they do and what they represent (see Figure 5.27).

Participant P₄ initially clicked on Users but didn't press on Edit button of user Administrator, instead, he started looking to find the Groups option, as exist in the Administration Tool. Then he went to the Accounts page and, after realizing that this was not the right page, he went again to Users and did the requested change using the edit option of user Administrator. The participant criticised the dual list control (see Figure 5.27) to select which accounts the user has access. He said that it is confusing to understand how he can move items between the lists. He referred: *“The buttons Add and Remove should be on the bottom or in the middle because you initially select the accounts and then click on Add or Remove”*.

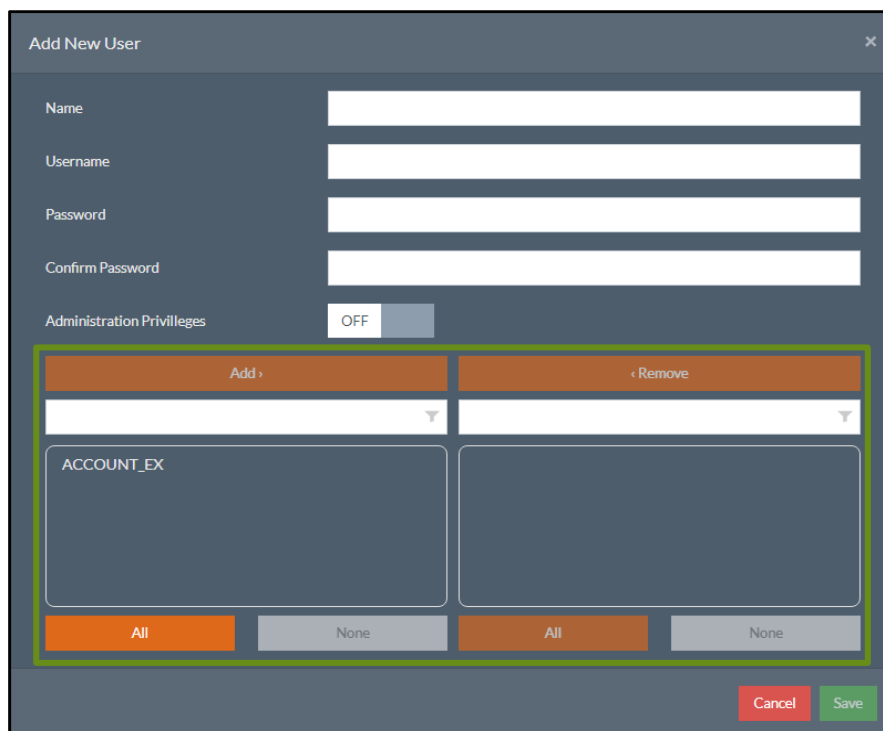


Figure 5.27 – Dual List used to associate accounts with users on CB Management Studio

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

From the questionnaire, we can understand that participants P1 and P4 considered that performing this task was harder to perform in CB Management Studio. Although, participant P3 had the opposite opinion and gave better classification to CB Management Studio. Only participant P2 considered the task as Very Easy in both systems (see Table 5.5).

Table 5.5 - Responses to the question “How easy was to perform the Task 5?”

	Administration Tool	CB Management Studio
P1	Easy	Moderate
P2	Very Easy	Very Easy
P3	Very Difficult	Moderate
P4	Easy	Moderate

This task evaluated the most significant change introduced by CB Management Studio regarding the management of CB Servers since it involved the removal of the concept of Groups. Analysing the behaviour of the participants when performing this task on Administration Tool it is clear that the users do not associate Groups as the association between Users and Accounts, which was expected. For this reason, we removed this concept, and we add the possibility to directly make this association using the users' edit/add form. We thought that this would be the best approach, although it is visible that for some users it would make more sense to have it on the Accounts Page which means that in the future it might make sense also to allow the users to create these associations from the Account Edit/Add form.

5.2.1.6 TASK 6: Create a new user with Administration Privileges and with access to the ACCOUNT_EX

All the participants correctly performed the task on both systems, although, when performing it on Administration Tool, the participants looked undecided if associating the new user to the group_administrator they were granting administration privileges and access to the ACCOUNT_EX.

Analysing the responses of the questionnaire, we can see that participants P1 and P3, considered that this task has the same difficulty level in both systems (Easy for P1 and Very

Easy for P₃) while P₂ and P₄ considered that CB Management Studio was easier (see Table 5.6).

Table 5.6 - Responses to the question “How easy was to perform the Task 6?”

	Administration Tool	CB Management Studio
P ₁	Easy	Easy
P ₂	Easy	Very Easy
P ₃	Very Easy	Very Easy
P ₄	Moderate	Easy

The results for this task show a tendency that CB Management Studio was easier to use. We believe that the fact of having the option to directly associate the users and accounts is the main contributor to this feeling.

5.2.1.7 TASK 7 Using the user Administrator, execute a SELECT on the table Inbox using the ACCOUNT_EX

All participants correctly started by configuring a connection in Query Analyzer and used the tree view in order to find the table Inbox and automatically generate the SELECT query.

In CB Management Studio, all participants started by clicking on the button “New Query”, then participants P₁ and P₃ selected the connection using the select boxes near the query editor. Participant P₃ wrote the query manually, while all the others tried to use the treeview to find the table Inbox and automatically generate the SELECT. Participant P₂ started searching for a way to add a connection.

It was visible that participants were not feeling confident about using this new tree. Participant P₁ said: *“This tree is strange, I cannot understand. Why do I have accounts and connections?”*. Participant P₄ showed satisfaction when he realized that he does not need to create a connection (*“There is no need to create anything? So great!”*).

From the questionnaire, we saw that participant P₁ considered the task harder in CB Management Studio, while P₄ considered the opposite (see Table 5.7).

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

Table 5.7 - Responses to the question “How easy was to perform the Task 7?”

	Query Analyzer	CB Management Studio
P1	Easy	Moderate
P2	Very Easy	Very Easy
P3	Very Easy	Very Easy
P4	Moderate	Very Easy

This task was used to analyse the users’ interaction when writing/executing a query. It also aimed to get impressions about the new Connection Browser. In general, we realized that this new tree view is a bit confusing. Although, we believe that adding some representative icons to it, combined with a better notion about its concept will turn its usage simpler.

5.2.1.8 TASK 8: Execute the Stored Procedure SP_UPLOAD_MESSAGE_MSG using the ACCOUNT_EX

All the participants performed the tasks without much effort both in Query Analyzer and CB Management Studio.

One difference of performing this task in Query Analyzer is that to add a file as a ByteArray, the users need to double click on the Value field (see Figure 5.29), while on CB Management Studio a button is shown to open the file explorer dialog (see Figure 5.28).

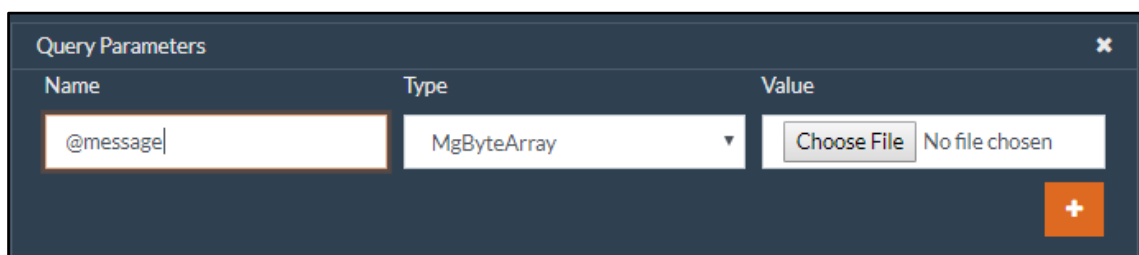


Figure 5.28 – Add file as a parameter of a query on CB Management Studio

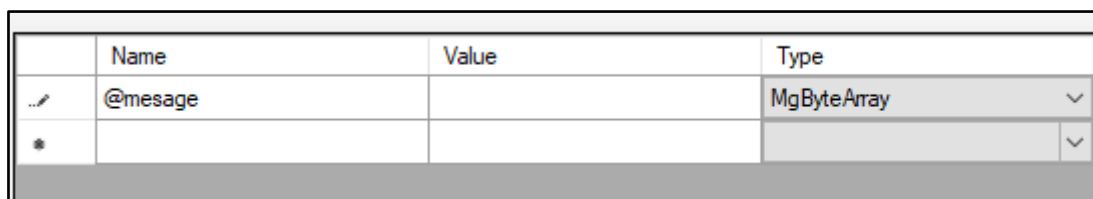


Figure 5.29 – Add file as a parameter of a query in Query Analyzer

When navigating the tree view of CB Management Studio, participant P1 mentioned that it might be useful to have a search option on it, to quickly find the stored procedure. Also, when filling the parameters data, he referred that having the button to open the file explorer dialog, when the parameter is a ByteArray was a right decision. About the query editor, he suggested: *“When you have a parameter on the query, it should be highlighted with a different colour.”*

Participant P2 also referred that would be good to have a search option on the tree view and also praised the button for opening the file explorer dialog, he said: *“This is more intuitive than double clicking like we do in Query Analyzer”*.

From the questionnaire results, we can see that all the users felt that performing this task is easier or with the same difficulty level of Query Analyzer (see Table 5.8).

Table 5.8 - Responses to the question “How easy was to perform the Task 8?”

	Query Analyzer	CB Management Studio
P1	Easy	Easy
P2	Very Easy	Very Easy
P3	Difficult	Easy
P4	Moderate	Very Easy

This task revealed to be easier to perform on CB Management Studio. We believe that the better results consist mainly on the fact of having a button that opens the file explorer dialog to select a file as a parameter, while to perform this on Query Analyzer, the user must “guess” that when double-clicking on the field it would open the file explorer dialog.

5.2.1.9 TASK 9: Using the user Administrator, get the connection string of a connection that uses the ACCOUNT_EX

All participants performed this task on Query Analyzer without showing any difficulty. Although, the same did not happen with CB Management Studio.

Participant P1 focused his attention on the tree view and tried to find the node that could have the option to generate a connection string, after some tries he correctly went to the node that represents the connection and selected the option “Generate Connection String”.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

Participants P₂ and P₃ had the same behaviour as P₁, although before looking into the tree view, P₂ start by looking for this option on the page Accounts while P₃ started by looking on the query page. About this task, P₂ referred: *“This was hard... I knew that it should be somewhere on the tree with right clicking... But, well it makes sense to be on the Connection and not in anywhere else”* and P₃ mentioned: *“I am not sure if this is the best place to have this option, although I do not see any other good option.”*

Participant P₄ start navigating on the tree but instead trying to right-click on all the nodes he initially right clicked on the node that represents the ACCOUNT_EX; then he did the same for the node that represents the connection for ACCOUNT_EX and choosed the option “Edit Connection”. Aware that this was not the correct option, he right clicked again on this node and selected the correct option: “Generate Connection String”.

The responses from the survey used to evaluate this task evidence that, in general, it was harder to perform it on CB Management Studio than in Query Analyzer (see Table 5.9).

Table 5.9 - Responses to the question “How easy was to perform the Task 9?”

	Query Analyzer	CB Management Studio
P ₁	Easy	Difficult
P ₂	Very Easy	Moderate
P ₃	Difficult	Moderate
P ₄	Very Easy	Moderate

This task revealed to be the hardest one, where in general we got the worst results when comparing with Query Analyzer. We believe that this happened due to a misunderstanding of the organization of the tree view and because it is a “hidden” feature, only available when right-clicking over a node that represents a connection. We must rethink if this is the best place to include this option since it will be a feature used very often by the developers that use Connect Bridge.

5.2.2 Results of SUS

As referred above, we used SUS to have a quantitative evaluation for CB Management Studio, using a very widespread test. In order to extract a result from the SUS test, its authors created a formula that consists of the following procedure [60]:

- Sum the score contributions from each item.

- For items 1, 3, 5, 7 and 9, the score contribution is the scale position minus 1.
- For items 2,4,6,8 and 10, the contribution is 5 minus the scale position.
- Multiply the sum of the scores by 2.5 to obtain the overall value of SU

Table 5.10 shows the results obtained using the SUS Evaluation.

Table 5.10 - Results of SUS Evaluation

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	SUS score
P1	3	3	3	1	4	2	4	3	2	2	62,5
P2	5	1	5	1	4	1	5	1	3	1	92,5
P3	3	1	4	4	4	1	5	1	4	1	80
P4	5	1	5	1	5	1	5	1	5	1	100
Average score											83,75

The authors of SUS did not present any strategy to interpret the SUS score, for this reason, J. Sauro realized a study over a set of 500 different SUS evaluations and concluded that the average classification is 68. He considered that the best way to evaluate a SUS score is by using a normalization like the one presented below [62].

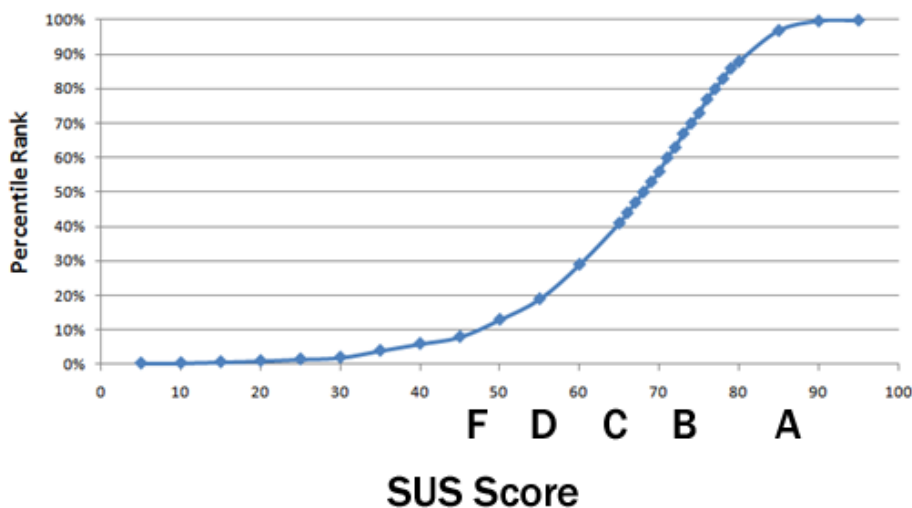


Figure 5.30 - Normalization of SUS evaluation [62]

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

Based on this, CB Management Studio is evaluated with an A Grade, which is an excellent classification. The combination of this result and the answers to the last question of the questionnaire, where all the participants revealed that they prefer CB Management Studio than the existing tools, give us confidence that this application should be well received by the Connecting Software customers.

5.3 Evaluation from Connecting Software Management Team

After finishing the development of CB Management Studio, a meeting with Connecting Software Management Team was realized to present the finalized product and collect feedback about it. The management team is formed by the company's CEO, the Project Managers of all projects in development, the Solution Architect and the Business Development Head. During this presentation, all the features of the application were demonstrated, while doubts and comments were raised by the attendants. The overall feedback was positive, and some of the concerns presented by them were already identified during the Think Aloud process. The satisfaction over the developed solution made the company's supervisor envision a future project that would consist on an extension of the CB Management Studio, to manage and test, not only Connect Bridge Platform, but also all the other products available in the company.

Receiving this feedback at the end of the project was very pleasing and gave us the sense that we fulfilled the expectations of the company.

5.4 Evaluation Analysis

All the feedback collected during this process will be very important for the next releases of CB Management Studio and all the negative aspects were added to the product backlog in order to be addressed on future iterations. In fact, at the time of this writing, some of the aspects referred were already solved.

For example, one of the most criticised aspects was the tree view, which for the users was difficult to understand and navigate. In order to solve that, we removed some levels of it to make it simpler, and we also added icons on each node, which we believe will help users to understand what each node represents. Figure 5.31 shows the difference between the treeview tested and the new treeview.

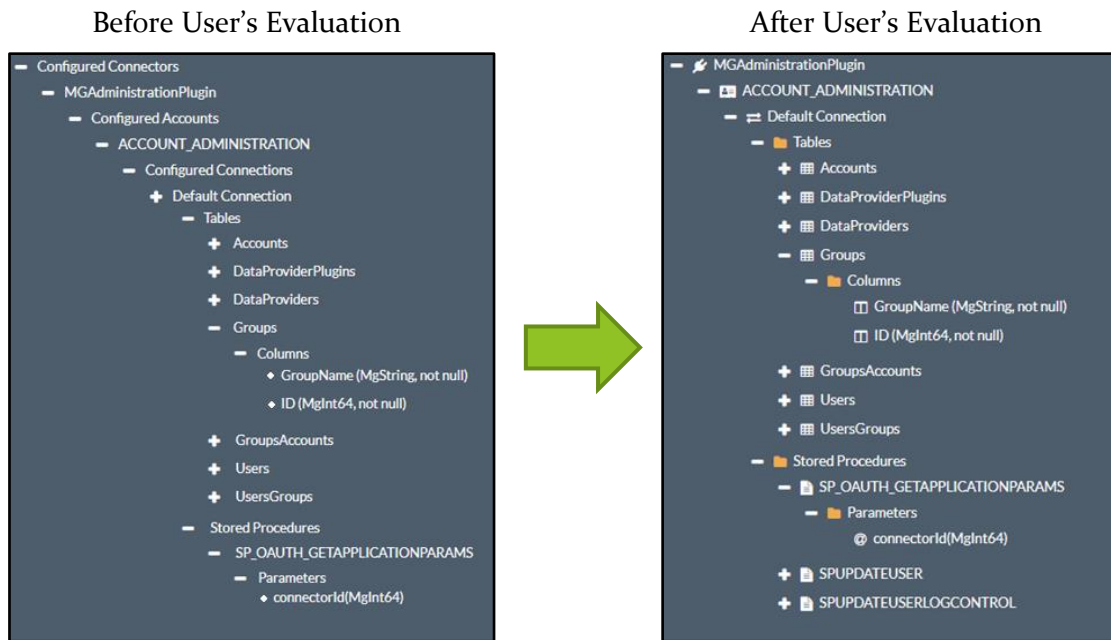


Figure 5.31 - Connection Browser before and after User's Evaluation

Other aspect already modified is the method used to associate users and accounts. Since some users went initially to the accounts page, we decided to create a redundancy allowing the Administrator user to associate the users and accounts directly when creating/editing an account. Figure 5.32 presents the new step that is now available to make this association. This new step presents a dual list very similar with the existing one on the User's add/edit form which was also improved in order to easily understand what each list specifies (see Figure 5.33).

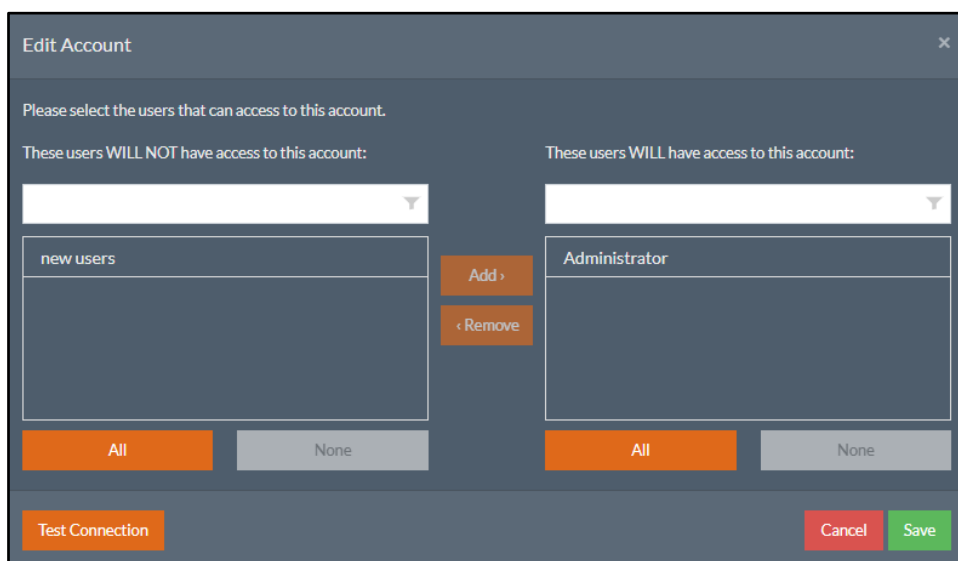


Figure 5.32 - New step of add/edit account used to directly associated users and accounts

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

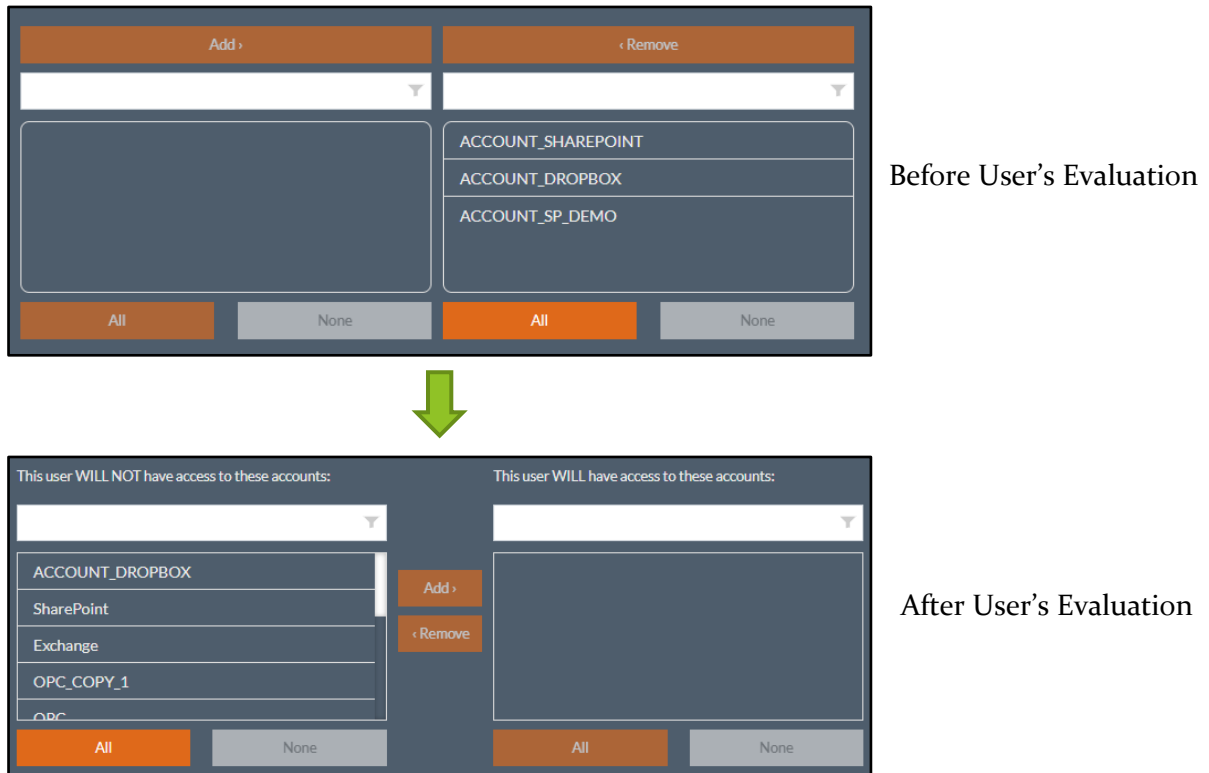


Figure 5.33 - Dual list used to associate user and accounts on Users edit/add window before and after user's evaluation

All these refinements and others are continuously being made in order to get the best user experience.

It is expected to have all these improvements released together with the version 4.19 of Connect Bridge Platform, which is predicted for the beginning of October 2018. This release will be the first official release containing CB Management Studio available for all current and future customers. Before that, it is planned that all the developers of Conencting Software will start using CB Management Studio, as also some partners and friendly customers that will provide more feedback about it.

6

CONCLUSIONS

This chapter describes the conclusions about this project, revisiting the work presented in the chapters above and presenting a critic overview of all the process.

6.1 Problem

Before the execution of this project, the tasks of managing and testing Connect Bridge Servers were performed recurring to existing tools made by Connecting Software named as Connect Bridge Server Administration Tool and Connect Bridge Query Analyzer, which were old and only compatible with Windows operating systems.

This company was conscious of this problem and had a plan to create a new solution to combine these tools in a single cross-platform application using Web Technologies, which would be called as Connect Bridge Management Studio.

The aim of this master's project was exactly to create this new tool which is, at the moment, already capable to replace both Administration Tool and Query Analyzer.

6.2 Research

The initial research made for this project concluded that during the last years, Web Technologies are an excellent opportunity to create solutions that are cross-platform and can replace existing desktop applications. This occurs mainly due to the emergence of AJAX and the Single Page Application since they allow to replicate many of the look-and-feel existing on these kinds of applications.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

The analysis of existing software used to manage databases allowed the identification of patterns that were followed on CB Management Studio, following the usability heuristics “Consistency and standards” and the “Recognition rather than recall”,

Finally the research about the market of integrations solutions allowed the understanding of how Connecting Software and Connect Bridge Platform contributes for this market and allowed us to understand the his user’s target are usually technical people with solid IT background.

Since we decided to go through the approach of using Web Technologies to replace existing Desktop applications, and considering that many of the look and feel of that applications should be retained, it would be useful if we investigated about JavaScript libraries that focus specifically on this aspect like PhosphorJS²⁰.

6.3 Analysis

Connect Bridge Management Studio was an old ambition of Connecting Software and, due to this, there was already a set of documents describing its goals and requirements. Although these documents were old and did not reflect the currents needs for this tool, it was possible to use them. Based on some of the features and requirements specified in these old documents and through meetings with the Product Owner it was possible to design Use Cases and extract an up-to-date list of requirements for this project.

Based on the requirements, three different architectures were designed and discussed with the company members. It was decided that a simpler architecture would grant the existence of a Minimum Valuable Product at the end of this project.

The definition of these requirements and architecture in the initial stage of this project, combined with the constant support and feedback received from the company allowed us to focus entirely on the development without being afraid that we were doing anything that was against the expectations of the company.

Although, it would have been useful if we initially have done an evaluation with users using the existing tools. This evaluation would probably provide some extra requirements in terms of usability and would have avoided some mistakes that were performed when replicating features and interactions methods used by Administration Tool and Query Analyzer on CB Management Studio.

²⁰ <https://phosphorjs.github.io/>

6.4 Development

The development of this project consisted of the creation of a back-end server using ASP.NET Core and a front-end web Single Page Application using Angular. During this stage, many challenges arise, from these, we emphasise the necessity of getting familiar with all these technologies, which were new for us and replicate as maximum as possible the look and feel of a desktop application.

ASP.NET Core is a very solid framework and counts with a good documentation and community that actively participates in blogs and forums, then it is very easy to get started with it. Also the fact of using C# which is a strictly typed, object-oriented language makes it natural to write code using the best practices and design patterns. One of the biggest problems developing in ASP.NET Core is that its predecessor is called only as ASP.NET which sometimes makes the research a bit harder since usually we found more results for ASP.NET than for ASP.NET Core.

Angular suffers the same problem as ASP.NET Core, it is much more easy to find information about AngularJS than to Angular. Despite that, Angular is also very stable and already counts with many developers supporting it with extra modules and libraries. We believe that Angular was a good choice and we are sure that we would not be able to complete this project if we decided to not use any JS framework to support it.

One lack of this project is the unexistence of tests. This forces the manual testing of all the application everytime a change is made. With tests, both on the back-end as also on the GUI, we could create test scripts that would avoid this constant manual testing which in long-term would result in more time to develop new features and in the increasing of confidence on the witten code.

6.5 Contributions

As a result of this project, currently Connecting Software owns an application ready to deploy that is capable of replacing Query Analyzer and Administration Tool in all their main features.

Also, this new application is cross-platform allowing Connecting Software to target companies whose developers and Information Technologies (IT) administrator do not use Windows-based systems on their daily working routine.

From the academic point of view, this project can be used as one additional example of a Web application that is entirely capable of replacing existing desktop applications.

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

Many of the challenges faced when replicating desktop look-and-feel on a web application will be frequent when creating new projects like this. Then, we tried to provide some ideas about how to solve them, which usually consists on accessing and modifying the source code of libraries, although we advice to always be aware if the author allow modifications on its code since open-source projects does not implicitly means that everyone can use it or modify it freely. Also, we presented an evaluation process which evidences how these solutions are received by the users which we believe that could be replicated in other projects with the same characteristics.

6.6 Evaluation

The evaluation performed concluded that the goals of this project were fulfilled. When comparing the new CB Management Studio with Administration Tool and Query Analyzer, we can see that all the main features were replicated and can now be performed using the new application.

Also, from the Think Aloud process, we collected some feedback that increased our confidence in the proposed solution. The generality of comments received were satisfactory and the SUS evaluation concluded that we have an excellent application regarding usability. Of course, some failures were detected by the participants that will need to be addressed in future, like the organization of the tree view or the attribution of the associations between users and accounts.

At the end of the project, a final evaluation meeting was performed combining all the management team of Connecting Software, from this, the feedback collected was pleasing since the final result fulfilled their expectations.

6.7 Future Work

However, this project can still be improved, and, as future work, it is expected to add new and valuable features that will turn CB Management Studio into a completely new tool sufficiently appealing to convince the old costumers to use it instead of the old tools. Features that might be useful to introduce are:

- Make the interface entirely customizable, allowing to change the position of GUI elements using a docking system;
- Introduce support to a light theme;

- Introduce support to users create their own library of queries instead of saving all the queries in different text files;
- Embed the documentation of the connectors into the application with a library of sample queries for the most common actions;
- Add support to configuration and reception of alerts about useful information.
- Show real-time information about the resources used by Connect Bridge Server like RAM, CPU, I/O operations, and others.

There are also technical improvements that should be performed in the application, including:

- Support copy of values presented on the tables;
- Improve code suggestions to suggest the names of tables, stored procedures, and columns;
- Add support to lazy loading modules in Angular;
- Upgrade ASP.NET Core 2.0 to 2.1 and Angular 5 to Angular 6;
- Use state management mechanism like Redux²¹ to help to maintain the state and data consistency in Angular;
- Creation and configuration of automated tests.

6.8 Lessons Learned

This first experience as a software engineer in a professional environment was very pleasing and allowed me to apply many of the concepts learned during all my academic course at the University of Madeira.

This project was very challenging and enriching, allowing me to learn some new trending technologies like Angular and ASP.NET Core, two frameworks that were entirely new to me. The support received by Connecting Software, particularly from my supervisor was essential for the execution of this project as well as all the support received by Prof. Filipe Quintal.

After concluding this project, I believe I'm more prepared to face the new challenges that the labour market will present me.

²¹ <https://redux.js.org/>

7

REFERENCES

- [1] T. Anderson, 'Which .NET framework for Windows: UWP, WPF or Windows Forms?', *Tim Anderson's IT Writing*. [Online]. Available: <https://www.itwriting.com/blog/10182-which-net-framework-for-windows-uwp-wpf-or-windows-forms.html>. [Accessed: 08-Apr-2018]
- [2] Kirk Koenigsbauer, 'New to Office 365 in July—new intelligent services Researcher and Editor in Word and more', *Microsoft 365*. 26-Jul-2016 [Online]. Available: <https://blogs.office.com/en-us/2016/07/26/the-evolution-of-office-apps-new-intelligent-services-such-as-researcher-and-editor-in-word-and-outlook-focused-inbox-as-well-as-continued-powerpoint-innovation-with-zoom/>. [Accessed: 21-Sep-2017]
- [3] J. Bishop and N. Horspool, 'Cross-Platform Development: Software that Lasts', *Computer*, vol. 39, no. 10, pp. 26–35, Oct. 2006.
- [4] A. Alkhars and W. Mahmoud, *Cross-Platform Desktop Development (JavaFX vs. Electron)*. 2017 [Online]. Available: <http://www.diva-portal.org/smash/record.jsf?pid=diva2:1081105>. [Accessed: 21-Sep-2017]
- [5] Mono Project, 'About Mono | Mono'. [Online]. Available: <http://www.mono-project.com/docs/about-mono/>. [Accessed: 26-Sep-2017]
- [6] M. James, 'Cross-Platform Desktop UIs with C#', *Mike Codes .NET*. 16-Nov-2014 [Online]. Available: <https://mikecodes.net/2014/11/16/cross-platform-desktop-uis-with-c/>. [Accessed: 26-Sep-2017]
- [7] L. Li, 'The Story About .NET Cross Platform UI Frameworks', *The Half-Blood Programmer*, 04-Jul-2017. [Online]. Available: <https://blog.lexstudio.com/the-story-about-net-cross-platform-ui-frameworks-dd4a9433doea>. [Accessed: 26-Sep-2017]
- [8] Mono Project, 'GUI Toolkits | Mono'. [Online]. Available: <http://www.mono-project.com/docs/gui/>. [Accessed: 26-Sep-2017]
- [9] R. Lander, P. Carter, and K. Cieślak, '.NET Core Guide'. [Online]. Available: <https://docs.microsoft.com/en-us/dotnet/core/>. [Accessed: 29-Oct-2017]
- [10] R. Lander, 'Announcing .NET Core 1.0'. [Online]. Available: <https://blogs.msdn.microsoft.com/dotnet/2016/06/27/announcing-net-core-1-0/>. [Accessed: 29-Oct-2017]

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

- [11] J. J. Garrett, 'Ajax: A New Approach to Web Applications | Adaptive Path'. [Online]. Available: <http://adaptivepath.org/ideas/ajax-new-approach-web-applications/>. [Accessed: 22-Sep-2017]
- [12] L. D. Paulson, 'Building rich web applications with Ajax', *Computer*, vol. 38, no. 10, pp. 14-17, Oct. 2005.
- [13] J. S. Zepeda and S. V. Chapa, 'From Desktop Applications Towards Ajax Web Applications', in *2007 4th International Conference on Electrical and Electronics Engineering*, 2007, pp. 193-196.
- [14] Neoteric, 'Single-page application vs. multiple-page application', *Medium*. 02-Dec-2016 [Online]. Available: <https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58>. [Accessed: 29-Oct-2017]
- [15] HotFrameworks, 'Web framework rankings | HotFrameworks'. [Online]. Available: <https://hotframeworks.com/languages/javascript>. [Accessed: 29-Oct-2017]
- [16] A. Ray, 'ReactJS For Beginners', *AndrewRay.me*, 21-Sep-2014. [Online]. Available: <http://blog.andrewray.me/reactjs-for-stupid-people/>. [Accessed: 29-Oct-2017]
- [17] AltexSoft, 'The Good and the Bad of ReactJS and React Native', *AltexSoft*. [Online]. Available: <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/>. [Accessed: 30-Oct-2017]
- [18] Pro-Tek, 'ADVANTAGES & DISADVANTAGES OF REACT.JS', *Pro-Tek Blog*. 01-Mar-2016 [Online]. Available: <http://www.pro-tekconsulting.com/blog/advantages-disadvantages-of-react-js/>. [Accessed: 05-Nov-2017]
- [19] K&C Team, 'Angular 4 vs React - what to choose in 2017', *K&C Blog*. [Online]. Available: <https://kruschecompany.com/blog/post/angular-vc-react>. [Accessed: 05-Nov-2017]
- [20] A. Alexseyenko, 'Angular 2 vs React. What to chose in 2017?', *Blog - TechMagic*, 20-Sep-2017. [Online]. Available: <http://blog.techmagic.co/angular-2-vs-react-what-to-chose-in-2017/>. [Accessed: 05-Nov-2017]
- [21] Code School, 'Single-page Applications', *Code School*. [Online]. Available: <https://www.codeschool.com/beginners-guide-to-web-development/single-page-applications>. [Accessed: 29-Oct-2017]
- [22] E. Korotya, '5 Best JavaScript Frameworks in 2017', *Hacker Noon*, 19-Jan-2017. [Online]. Available: <https://hackernoon.com/5-best-javascript-frameworks-in-2017-7a63b3870282>. [Accessed: 17-Nov-2017]
- [23] A. Fiori, 'Angular 2 pros and cons', *AndreaFiori.net*. [Online]. Available: <http://www.andreafori.net/posts/angular-2-pros-and-cons>. [Accessed: 17-Nov-2017]
- [24] compilers.net, 'compilers.net > paedia > compiler'. [Online]. Available: <http://www.compilers.net/paedia/compiler/index.htm>. [Accessed: 26-Dec-2017]
- [25] M. J. Foley, 'Microsoft takes the wraps off TypeScript, a superset of JavaScript', *ZDNet*. [Online]. Available: <http://www.zdnet.com/article/microsoft-takes-the-wraps-off-typescript-a-superset-of-javascript/>. [Accessed: 17-Nov-2017]
- [26] Microsoft, 'TypeScript - JavaScript that scales.' [Online]. Available: <https://www.typescriptlang.org/>. [Accessed: 17-Nov-2017]
- [27] Aux, 'TypeScript: pros and cons', *Aux*. 19-Oct-2016 [Online]. Available: <https://medium.com/@auxx/typescript-pros-and-cons-873529634099>. [Accessed: 17-Nov-2017]
- [28] Agriya, 'AngularJS is No More - Pros and Cons of TypeScript and Angular 2?', *Agriya Blog*. [Online]. Available: <https://www.agriya.com/blog/angularjs-is-no-more-the-future-of-typescript-and-angular-2/>. [Accessed: 17-Nov-2017]
- [29] DataArt, 'What is TypeScript? Pros and Cons', *Designmodo*, 03-Dec-2014. [Online]. Available: <https://designmodo.com/typescript/>. [Accessed: 17-Nov-2017]

- [30] P. Desjardins, 'Pros and Cons of Using TypeScript', *Patrick Desjardins' Blog*. [Online]. Available: <http://patrickdesjardins.com/blog/pros-and-cons-of-using-typescript>. [Accessed: 17-Nov-2017]
- [31] A. Benoit, *NW.js essentials: build native desktop applications for Windows, Mac OS, or Linux using the latest web technologies*. 2015 [Online]. Available: <http://proquest.safaribooksonline.com/?fpi=9781785280863>. [Accessed: 26-Sep-2017]
- [32] Techopedia, 'What is a Vertical Application? - Definition from Techopedia', *Techopedia.com*. [Online]. Available: <https://www.techopedia.com/definition/9909/vertical-application>. [Accessed: 26-Sep-2017]
- [33] J.-P. Côté, 'NW.js & Electron Compared (2016 Edition)', *TangibleJS*, 26-Mar-2016. [Online]. Available: <http://tangiblejs.com/posts/nw-js-and-electron-compared-2016-edition>. [Accessed: 23-Oct-2017]
- [34] Ashutosh KS, 'Frameworks & Tools to Develop Cross-Platform Desktop Apps – Best of, *HKDC*'. [Online]. Available: <http://www.hongkiat.com/blog/frameworks-tools-build-cross-platform-desktop-apps/>. [Accessed: 23-Sep-2017]
- [35] Slant, 'Slant - NW.js - What are the best tools for building cross-platform desktop apps with web technologies?', *Slant*. [Online]. Available: <https://www.slant.co/topics/3520/~tools-for-building-cross-platform-desktop-apps-with-web-technologies>. [Accessed: 23-Oct-2017]
- [36] J. Nielsen, '10 Heuristics for User Interface Design: Article by Jakob Nielsen', *Nielsen Norman Group*. [Online]. Available: <https://www.nngroup.com/articles/ten-usability-heuristics/>. [Accessed: 27-Dec-2017]
- [37] M. Rouse, 'What is integration? - Definition from WhatIs.com', *SearchCRM*. [Online]. Available: <http://searchcrm.techtarget.com/definition/integration>. [Accessed: 27-Dec-2017]
- [38] GCI Management LLC, 'Software and Data Integration'. 2014.
- [39] Object Management Group, 'Unified Modeling Language, v2.5.1'. Dec-2017.
- [40] Atlassian, 'What is DevOps?', *Atlassian*. [Online]. Available: <https://www.atlassian.com/devops>. [Accessed: 18-Jul-2018]
- [41] K. Schwaber and J. Sutherland, 'Scrum Guide', *Scrum Guides*. [Online]. Available: <http://www.scrumguides.org/scrum-guide.html>. [Accessed: 16-Jun-2018]
- [42] Scrum Alliance, 'Scrum Alliance - Learn About Scrum'. [Online]. Available: <https://www.scrumalliance.org/learn-about-scrum>. [Accessed: 16-Jun-2018]
- [43] DevIQ, 'Repository Pattern | DevIQ'. [Online]. Available: <http://deviq.com/repository-pattern/>. [Accessed: 03-Jun-2018]
- [44] DevIQ, 'Specification Pattern | DevIQ'. [Online]. Available: <http://deviq.com/specification-pattern/>. [Accessed: 03-Jun-2018]
- [45] L. LIN, 'Angular DataTables'. [Online]. Available: <https://l-lin.github.io/angular-datatables/#/welcome>. [Accessed: 31-May-2018]
- [46] SpryMedia Ltd., 'DataTables | Table plug-in for jQuery'. [Online]. Available: <https://datatables.net/>. [Accessed: 31-May-2018]
- [47] Swimlane, 'Introduction - ngx-datatable'. [Online]. Available: <https://swimlane.gitbook.io/ngx-datatable>. [Accessed: 31-May-2018]
- [48] ag-Grid, 'ag-Grid: Datagrid packed with features that your users need with the performance you expect.' [Online]. Available: <https://www.ag-grid.com/>. [Accessed: 31-May-2018]
- [49] ag-Grid, 'ag-Grid'. [Online]. Available: <https://www.ag-grid.com/license-pricing.php>. [Accessed: 12-Jun-2018]

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

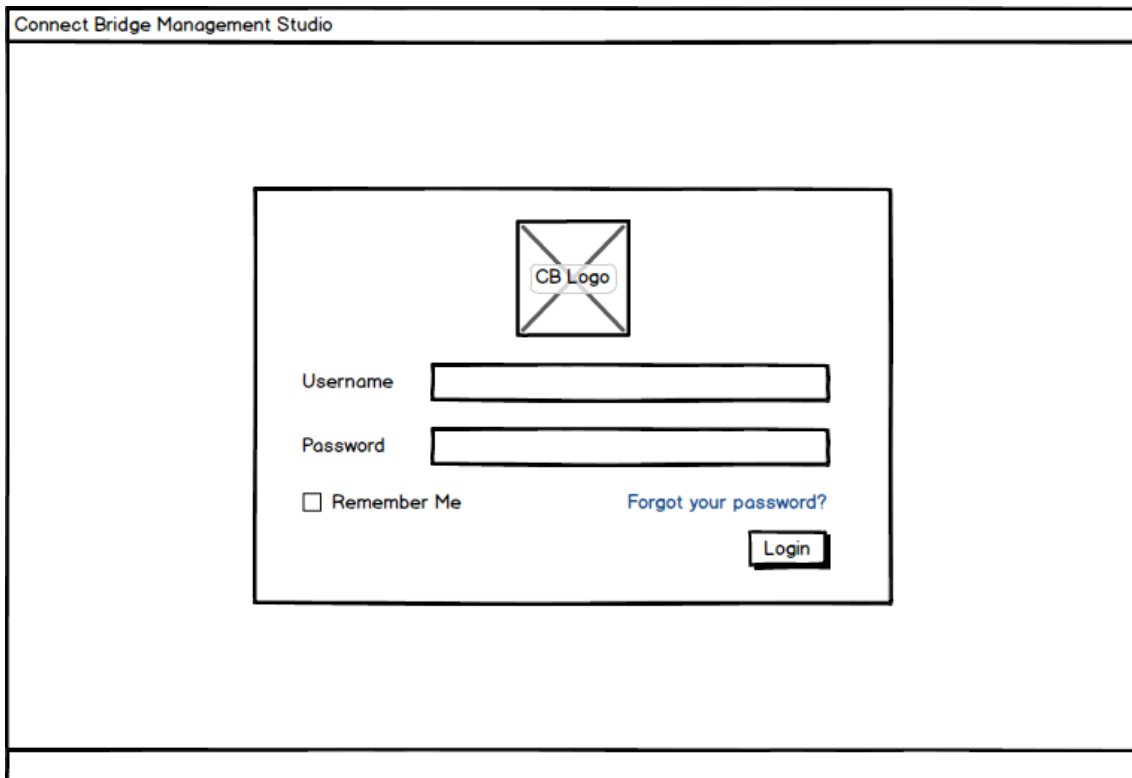
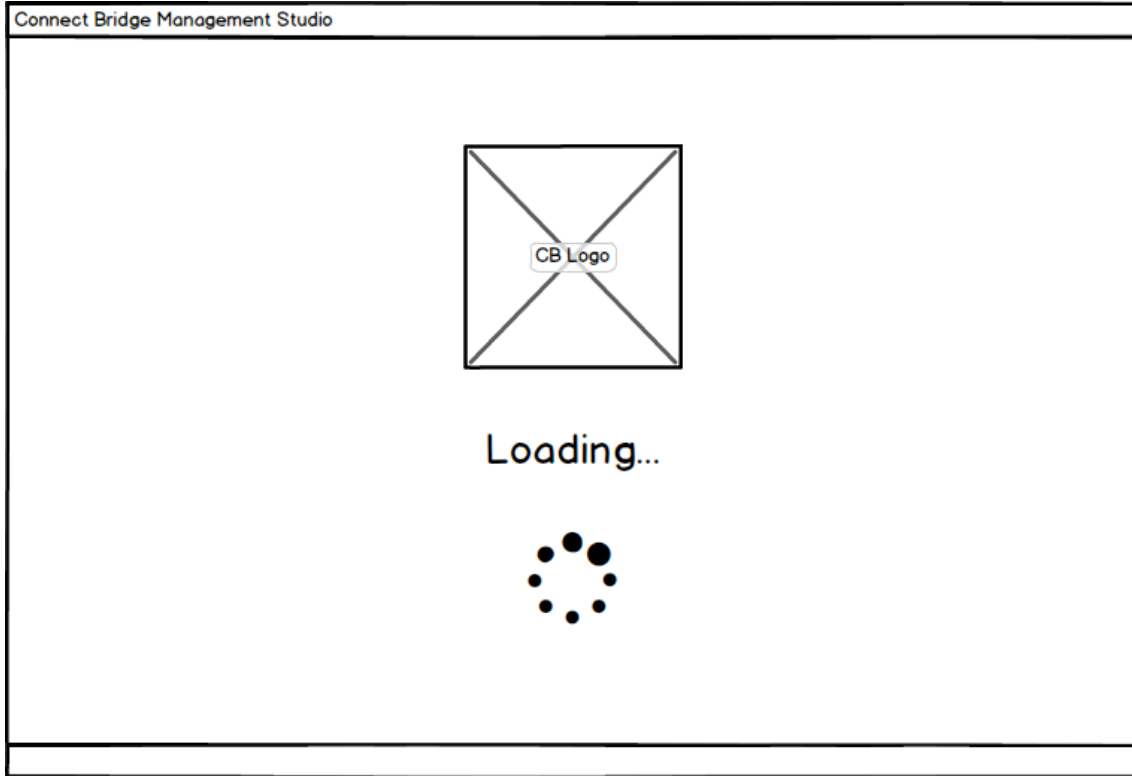
- [50] 500 Tech, *angular-tree-component: A simple yet powerful tree component for Angular (2-6)*. 500Tech, 2018 [Online]. Available: <https://github.com/500tech/angular-tree-component>. [Accessed: 31-May-2018]
- [51] Valor Software, *ng2-tree: Angular tree component*. Valor Software, 2018 [Online]. Available: <https://github.com/valor-software/ng2-tree>. [Accessed: 31-May-2018]
- [52] Advanced Installer, 'Free Windows Installer - MSI Installer Tool'. [Online]. Available: <https://www.advancedinstaller.com/>. [Accessed: 12-Jun-2018]
- [53] R. Morris, *Swashbuckle.AspNetCore: Swagger tools for documenting APIs built on ASP.NET Core*. 2018 [Online]. Available: <https://github.com/domaindrivendev/Swashbuckle.AspNetCore>. [Accessed: 06-Jul-2018]
- [54] SmartBear Software, 'The Best APIs are Built with Swagger Tools | Swagger'. [Online]. Available: <https://swagger.io/>. [Accessed: 06-Jul-2018]
- [55] Clayton Lewis, 'Using the "Thinking-aloud" Method in Cognitive Interface Design', 1982 [Online]. Available: [https://domino.research.ibm.com/library/cyberdig.nsf/papers/2513E349E05372CC852574EC0051EEA4/\\$File/RC9265.pdf](https://domino.research.ibm.com/library/cyberdig.nsf/papers/2513E349E05372CC852574EC0051EEA4/$File/RC9265.pdf). [Accessed: 27-Jun-2018]
- [56] Microsoft, 'Visual Studio Code - Code Editing. Redefined'. [Online]. Available: <http://code.visualstudio.com/>. [Accessed: 12-Jun-2018]
- [57] Microsoft, 'Monaco Editor'. [Online]. Available: <https://microsoft.github.io/monaco-editor/>. [Accessed: 12-Jun-2018]
- [58] P. Bhattacharya and J. Keup, 'A Case Study of the Effects of a Web Interface Redesign Based on Usability Guidelines', 2011.
- [59] J. Nielsen, 'Thinking Aloud: The #1 Usability Tool', *Nielsen Norman Group*. [Online]. Available: <https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/>. [Accessed: 02-Jun-2018]
- [60] J. Brooke, 'SUS - A quick and dirty usability scale', p. 8.
- [61] A. S. for P. Affairs, 'System Usability Scale (SUS)', 06-Sep-2013. [Online]. Available: </how-to-and-tools/methods/system-usability-scale.html>. [Accessed: 18-Jun-2018]
- [62] J. Sauro, 'MeasuringU: Measuring Usability with the System Usability Scale (SUS)'. [Online]. Available: <https://measuringu.com/sus/>. [Accessed: 26-Jun-2018]

8

APPENDICES


Appendix A - Connect Bridge Management Studio Medium-Low Fidelity Prototypes ...	101
Appendix B - Think-Aloud Protocol	111
Appendix C - Survey used after think-aloud	113

Appendix A - Connect Bridge Management Studio Medium-Low Fidelity Prototypes



Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

Connect Bridge Management Studio




Password Expired

Please enter a new password for your account

Password

Confrim Password

Connect Bridge Management Studio



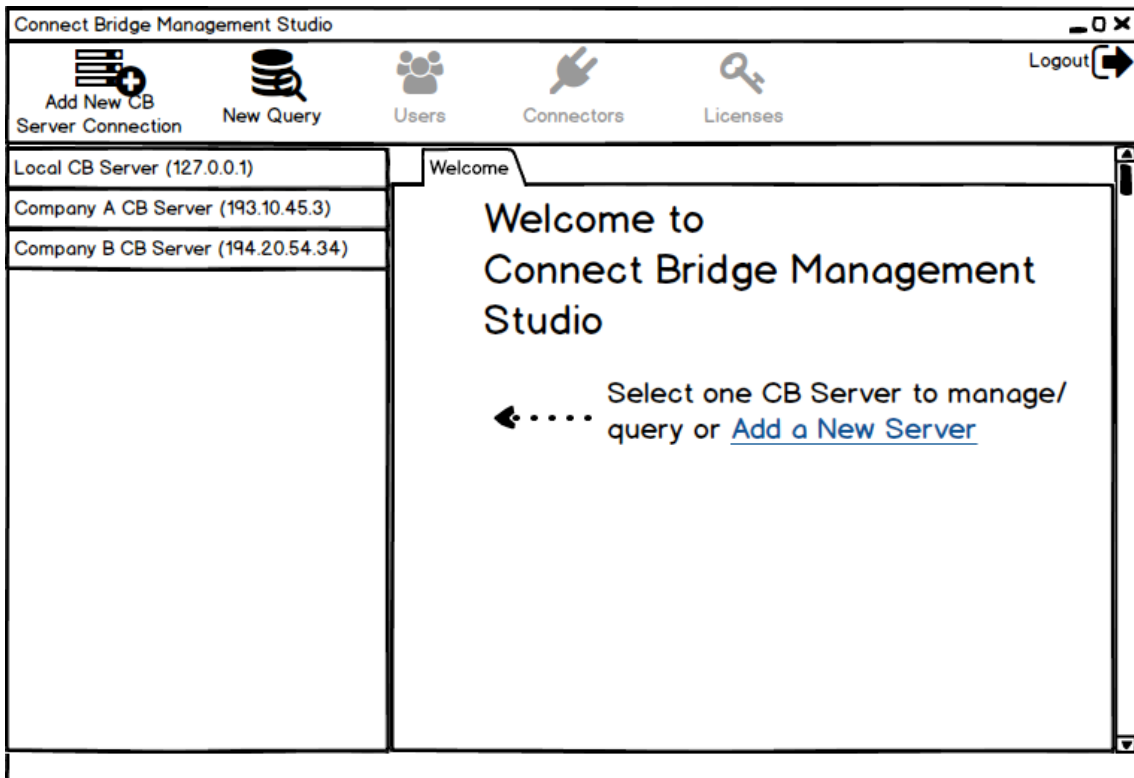
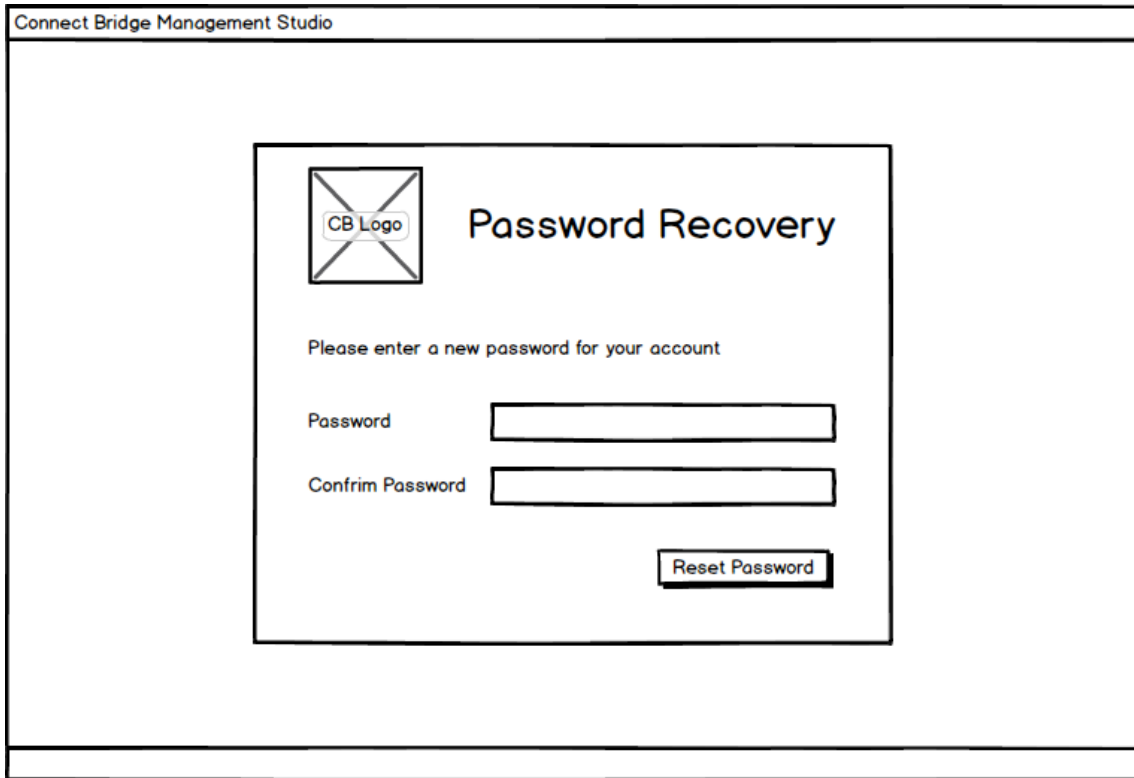
Password Expired

Please answer the following security questions

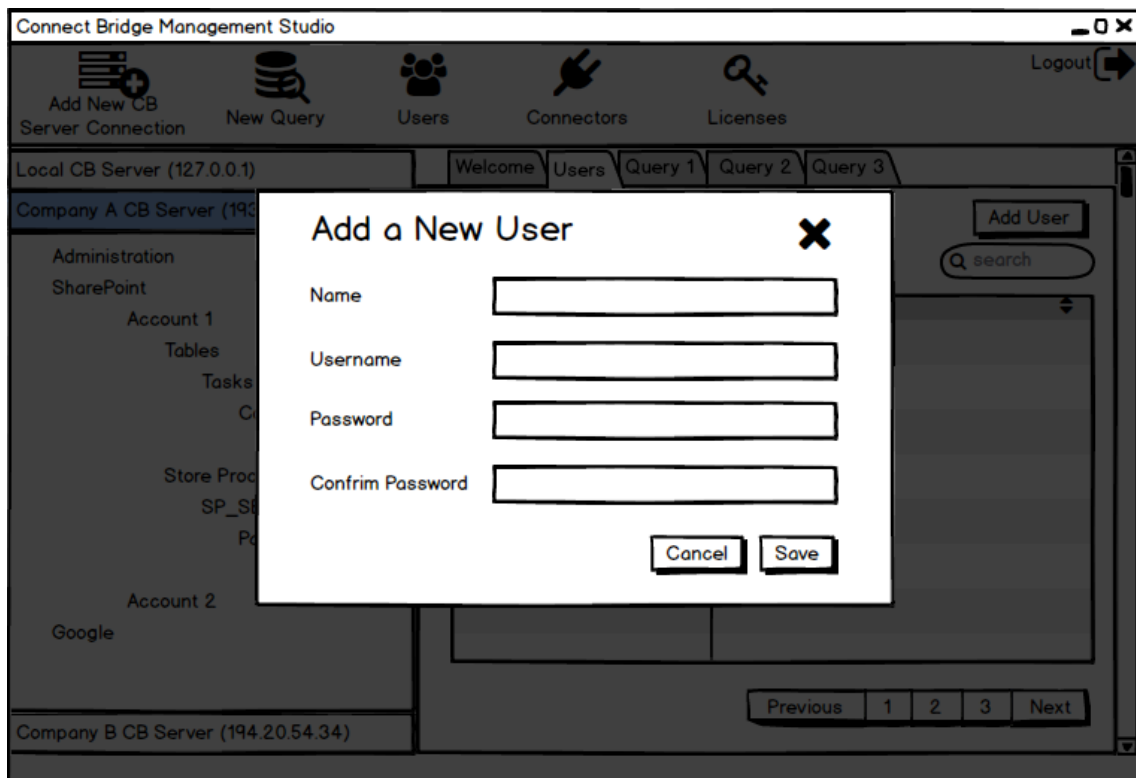
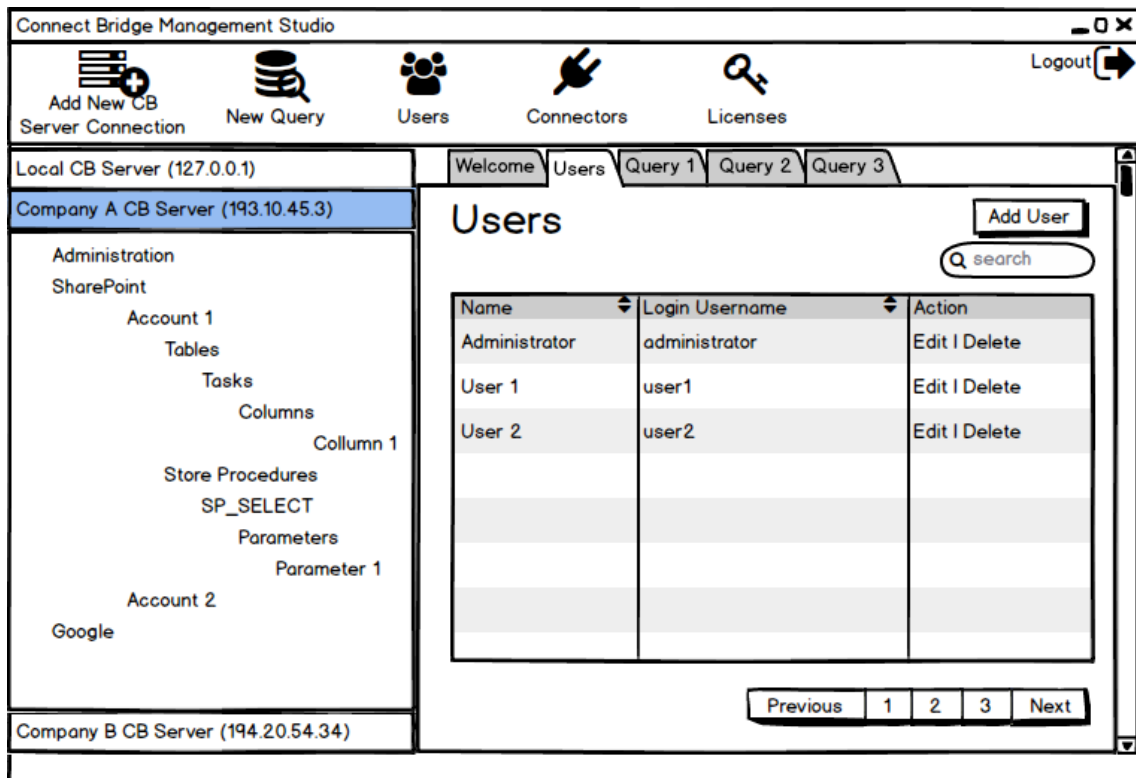
Question 1?

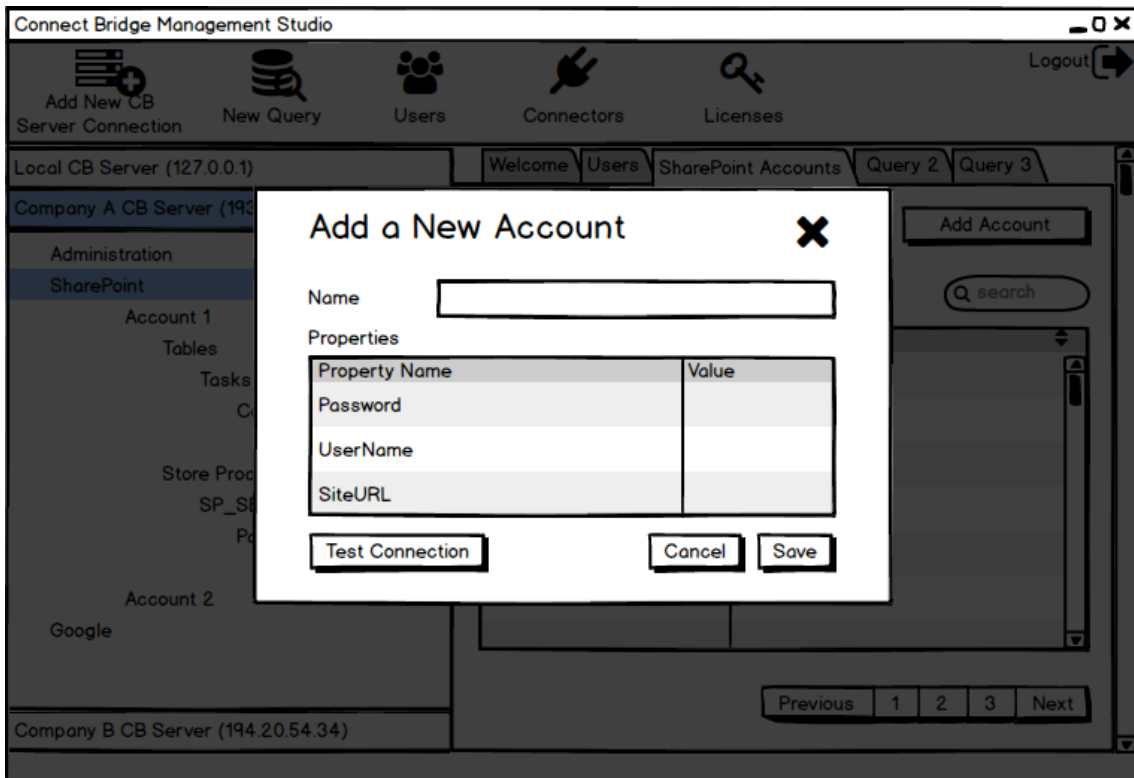
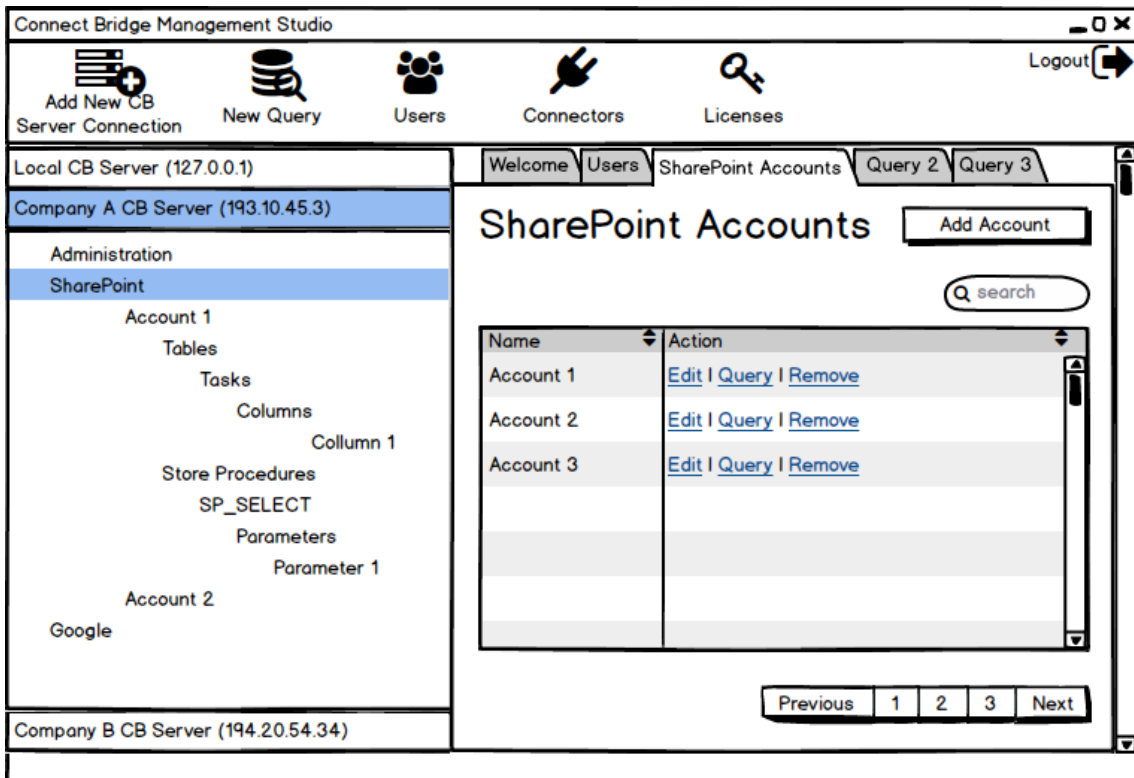
Question 2?

Question 3?

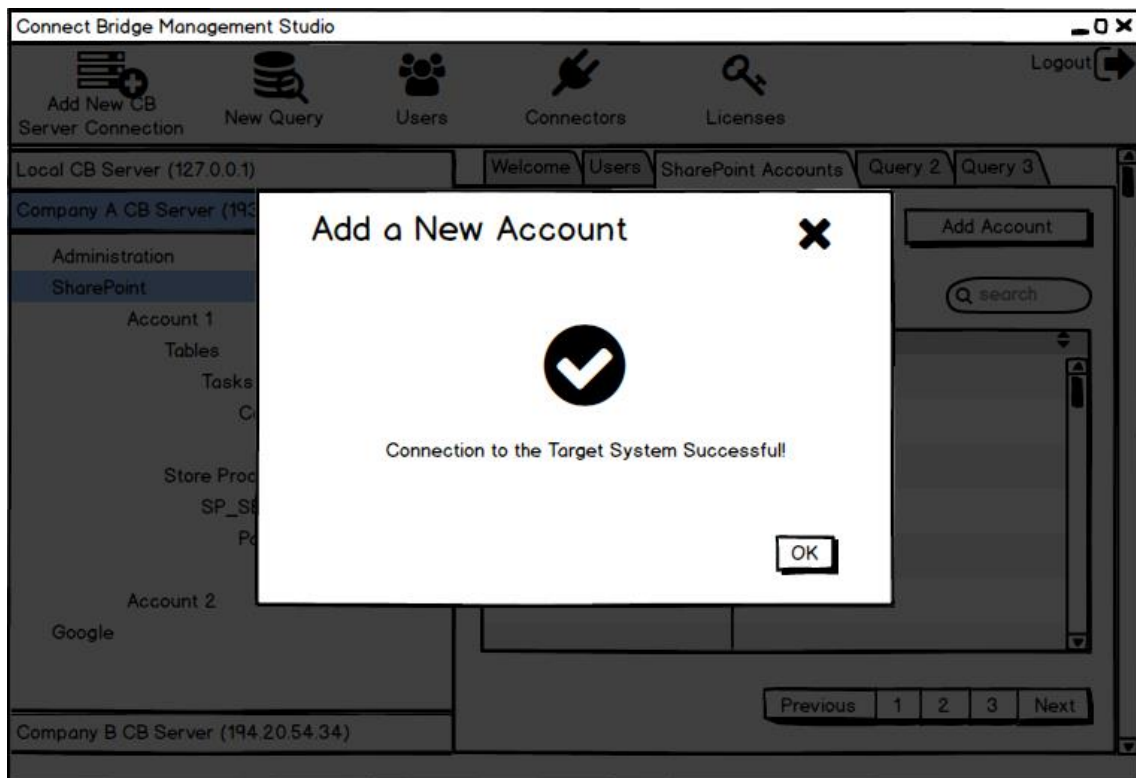
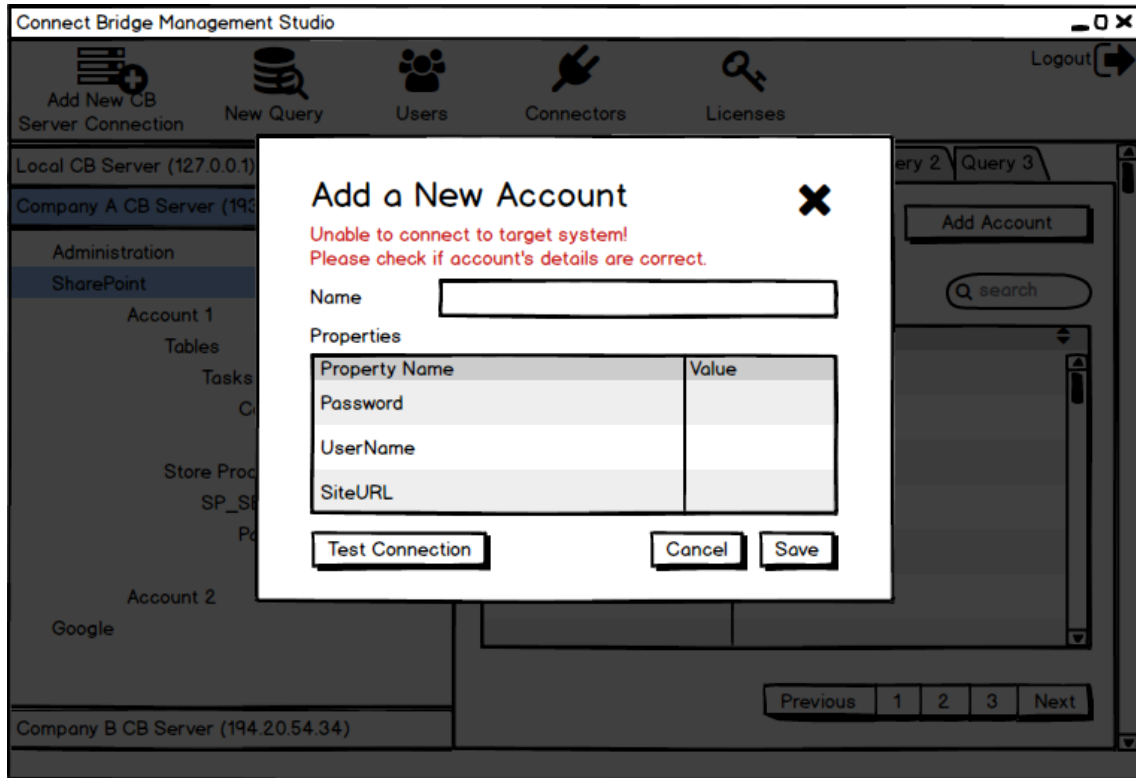


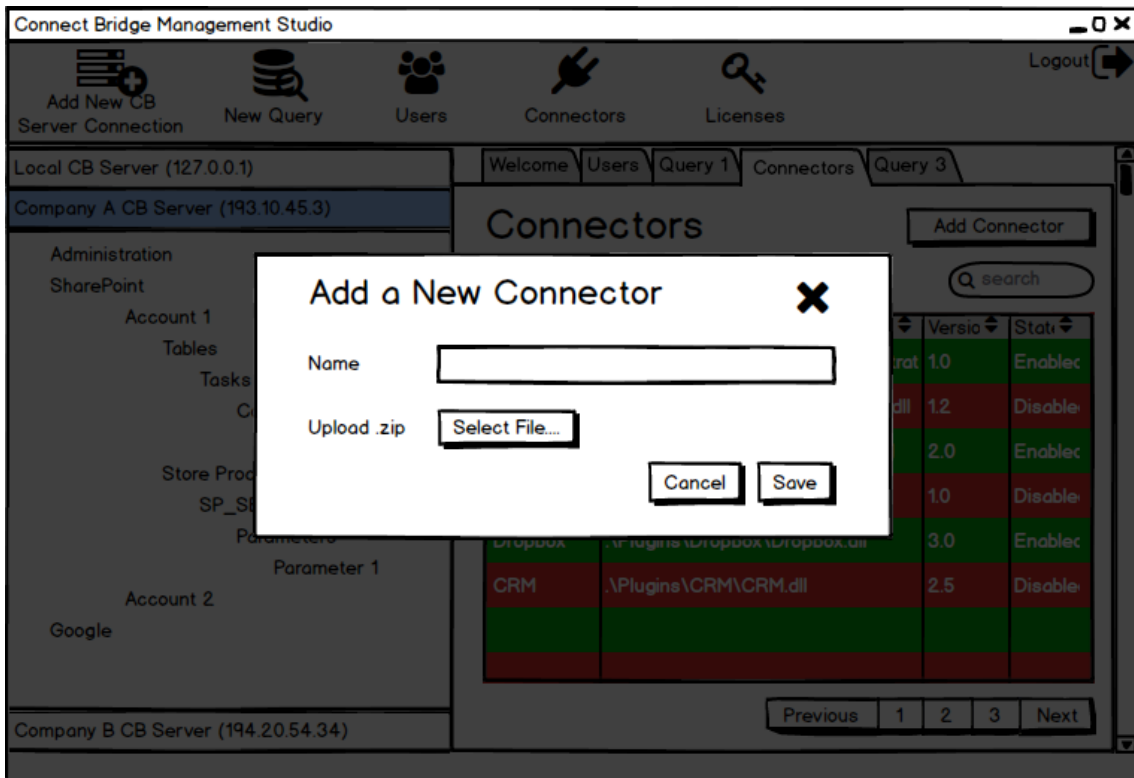
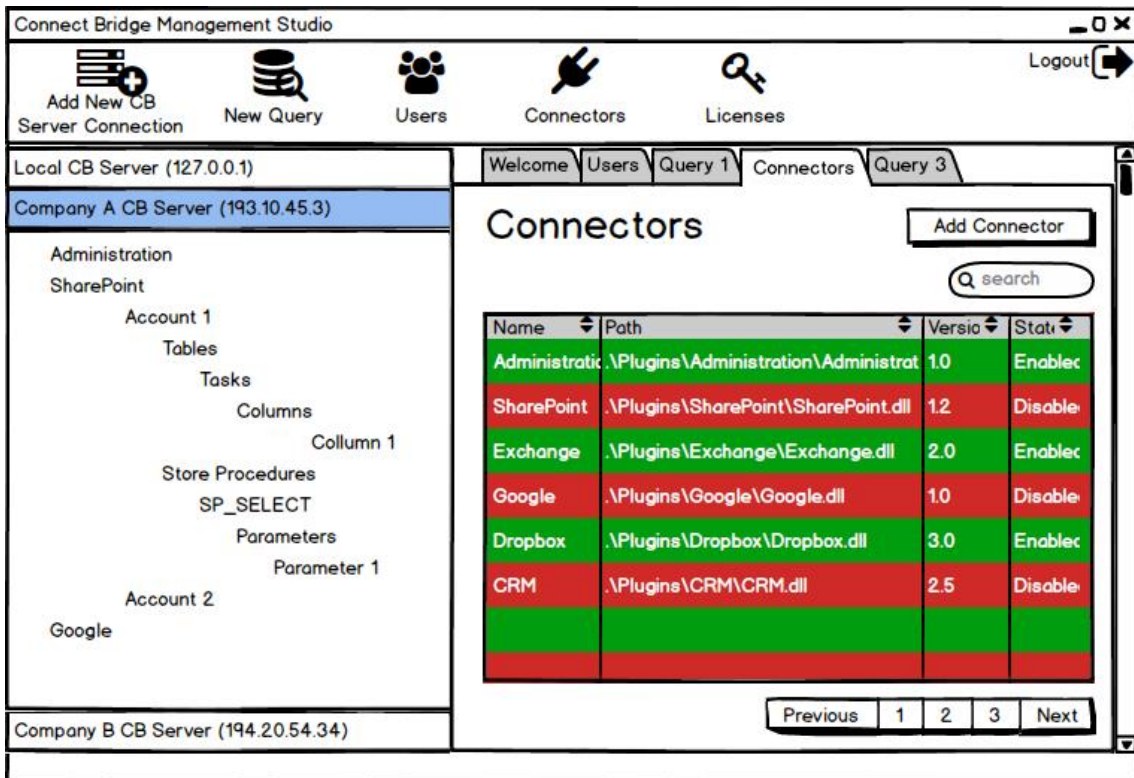
Connect Bridge Management Studio:
 A web-based application used to manage and test Connect Bridge servers



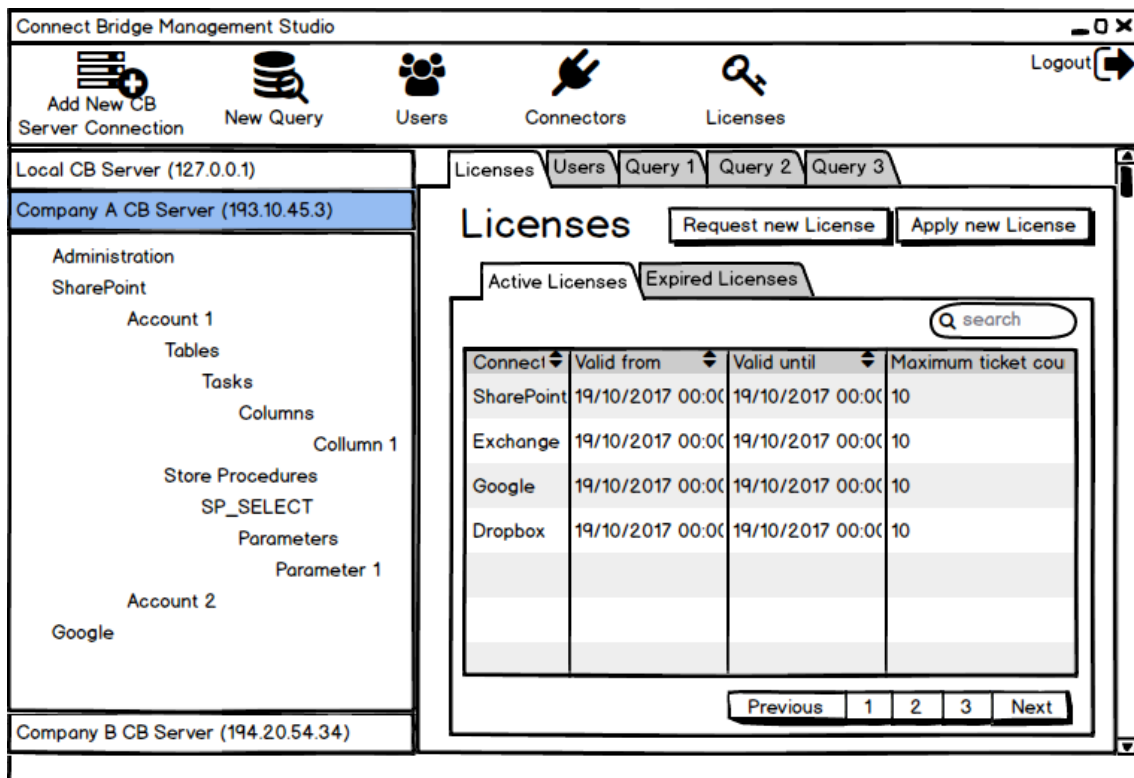
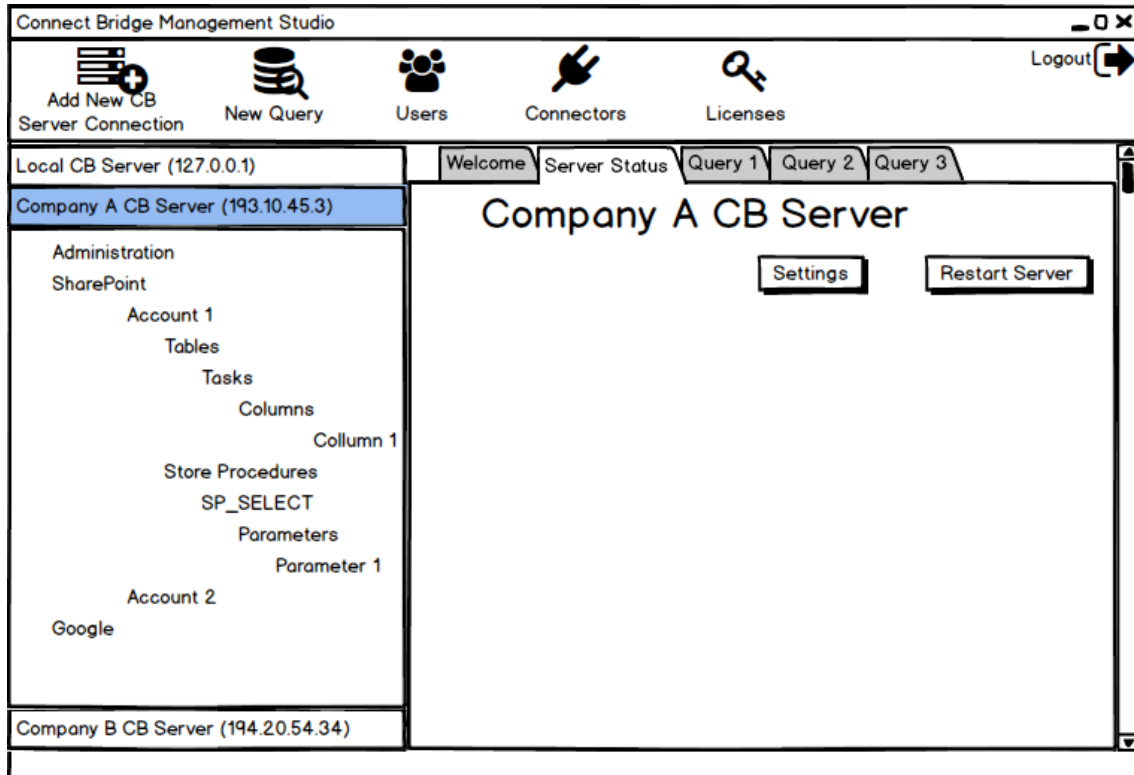


Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers





Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers



Local CB Server (127.0.0.1)

Company A CB Server (193.10.45.3)

- Administration
- SharePoint
- Account 1
 - Tables
 - Tasks
 - Columns
 - Column 1
 - Store Procedures
 - SP_SELECT
 - Parameters
 - Parameter 1
- Account 2
- Google

Company B CB Server (194.20.54.34)

Execute Query

Server: [v] Connector: [v] Account: [v] Run Query Stop

Query Results

Name (job title)	Age	Nickname	Employee
Giacomo Guizzoni Founder & CEO	40	Peldi	<input type="radio"/>
Marco Botton Tuttofare	38		<input checked="" type="checkbox"/>

Previous 1 2 3 Next

Local CB Server (127.0.0.1)

Company A CB Server (193.10.45.3)

- Administration
- SharePoint
- Account 1
 - Tables
 - Tasks
 - Columns
 - Column 1
 - Store Procedures
 - SP_SELECT
 - Parameters
 - Parameter 1
 - Account 2
 - Google

Company B CB Server (194.20.54.34)

SP_SELECT_...

Create Query

Description

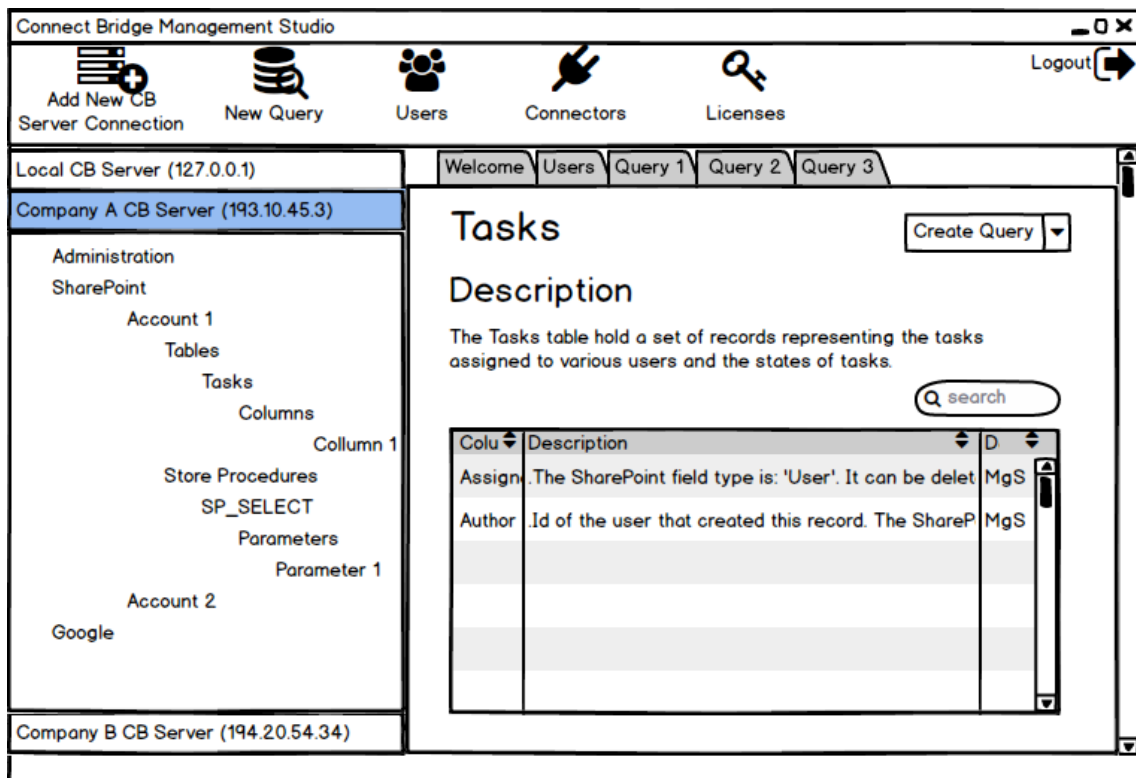
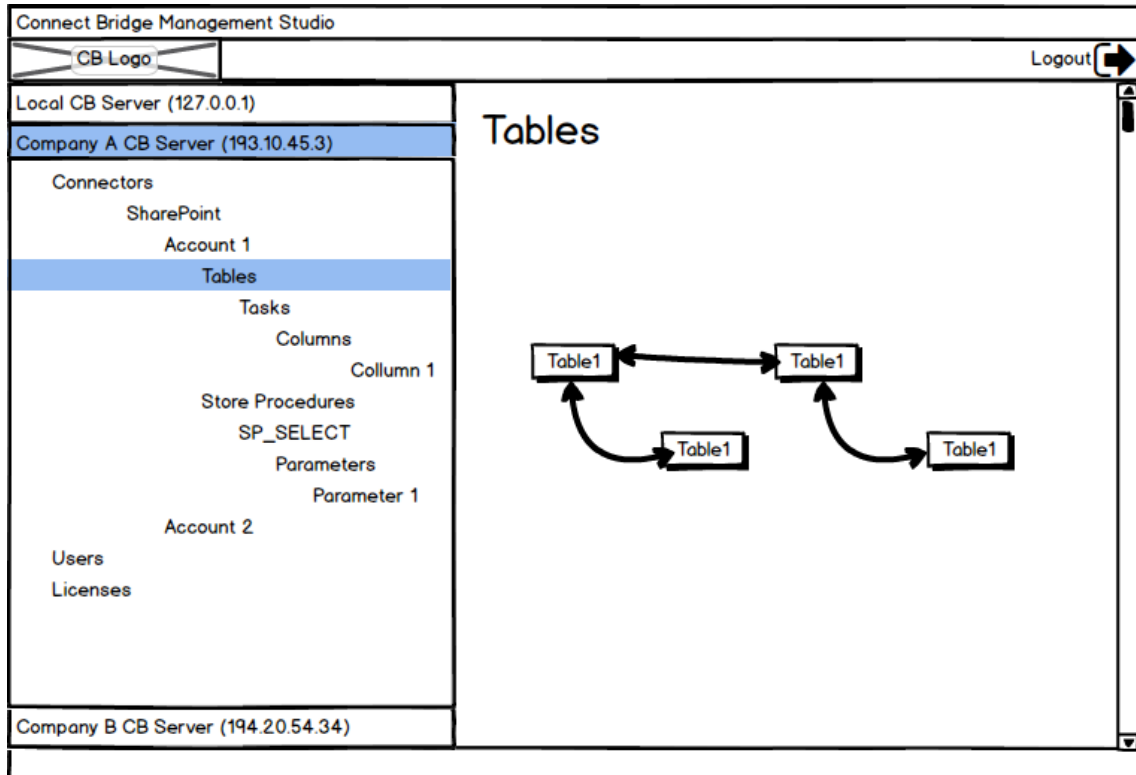
Lists the available attachments for an entity.

Input Parameters

search

Param	Description	Data
@tableName	Table name in which entity with @entityId is found	MgStr
@entityId	Id of the entity to search the attachments for.	MgInt

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers



Appendix B - Think-Aloud Protocol

Connect Bridge Management Studio Evaluation

Please read the following tasks out loud and then perform the task. As you are working, please tell us what you are thinking and feel free to express your concerns. This survey intends to evaluate the interface and is not intended to test you or your skills.

TASK 1

Login into the application using credentials:

- Username: Administrator
- Password: 123456

TASK 2

Check the active licenses on CB Server

TASK 3

Disable/Enable the Dropbox connector on CB Server

TASK 4

Create a new Account using the Exchange connector (MGEXPlugin2010) with the following information:

- Account Name: ACCOUNT_EX_YOUR_NAME
- UserName: Your company's e-mail.
- Password: Your Office365 Password.
- ServerUrl: office365
- EmailAddress: Your company's e-mail.
- RequestServerVersion: Exchange2010SP1

TASK 5

Allow the user Administrator to access the ACCOUNT_EX_YOUR_NAME.

TASK 6

Create a new user with your name with Administration Privileges and with access to the ACCOUNT_EX_YOUR_NAME

TASK 7

Execute a SELECT on the table Inbox using the ACCOUNT_EX_YOUR_NAME.

TASK 8

Execute the Stored Procedure SP_UPLOAD_MESSAGE_MSG with the following parameters:

- @InBuffer = The file "Test email" that is on Desktop

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

TASK 9

Get the connection string of a connection that uses the `ACCOUNT_EX_YOUR_NAME`

Appendix C - Survey used after think-aloud

Connect Bridge Management Studio

The following survey aims to evaluate your experience using Connect Bridge Management Studio.

1. How easy was to perform the Task 1?

Log in into the application
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Administration Tool	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

2. How easy was to perform the Task 2?

Check the active licenses on CB Server
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Administration Tool	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

3. How easy was to perform the Task 3?

Disable/Enable the Dropbox connector on CB Server
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Administration Tool	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

4. How easy was to perform the Task 4?

Create a new Account using the Exchange connector
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Administration Tool	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

5. How easy was to perform the Task 5?

Allow the user Administrator to access this account.
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Administration Tool	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Connect Bridge Management Studio:
A web-based application used to manage and test Connect Bridge servers

6. How easy was to perform the Task 6?

Create a new user with your name
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Administration Tool	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

7. How easy was to perform the Task 7?

Execute a SELECT on the table Inbox using the ACCOUNT_EX_YOUR_NAME.
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Query Analyzer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

8. How easy was to perform the Task 8?

Execute the Stored Procedure SP_UPLOAD_MESSAGE_MSG
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Query Analyzer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

9. How easy was to perform the Task 9?

Get the connection string of a connection that uses the ACCOUNT_EX_YOUR_NAME
Mark only one oval per row.

	Very Dificult	Difficult	Moderate	Easy	Very Easy
Query Analyzer	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

10. Based on your recent experience with Connect Bridge Management Studio, please indicate whether you agree or disagree with the following statements

Mark only one oval per row.

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
I think that I would like to use Connect Bridge Management Studio frequently	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found Connect Bridge Management Studio unnecessarily complex	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I thought Connect Bridge Management Studio was easy to use	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I think that I would need the support of a technical person to be able to use Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found the various functions in Connect Bridge Management Studio were well integrated	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I thought there was too much inconsistency in Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I would imagine that most people would learn to use Connect Bridge Management Studio very quickly	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I found Connect Bridge Management Studio very cumbersome to use	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I felt very confident using Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I needed to learn a lot of things before I could get going with Connect Bridge Management Studio	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

11. Which interface did you like the most?

Mark only one oval.

- Query Analyzer + Administration Tool
- Connect Bridge Management Studio

12. Please use this space to leave any comments about the new Connect Bridge Management Studio
