Aplicació web per realitzar operacions SQL de forma gràfica

Marc Alvarez Ricart

Resum– Normalment quan s'ha de realitzar una consulta a una Base de dades ens trobem davant d'un full en blanc sobre el qual hem d'escriure la consulta. De manera que una persona que està aprenent el llenguatge SQL ha de recordar quina estructura té cada tipus de consulta per poder programar-la. Per tant, el projecte consisteix en dissenyar una aplicació web que permeti als usuaris realitzar consultes SQL sobre la base de dades que desitgin mitjançant una interfície gràfica creada utilitzant elements web. Amb aquesta aplicació web es proporciona una estructura sobre la qual anar construint la consulta, és a dir, l'usuari ja no es troba davant d'un full en blanc. És per aquesta raó que l'aplicació està orientada a persones que s'estàn iniciant en el món de la programació de consultes a Bases de Dades.

Paraules clau– Base de dades, aplicació, consulta, SQL, vista, MVC.

Abstract– Usually when we have a query to a database we face a blank page on which we write the query. So a person who is learning the SQL language must remember the structure of each type of query to program it. Therefore, the project involves designing a web application that allows users to do SQL queries on the database they wish using a graphic interface with Web elements. This application provides a web structure on which to build the query. So the user is no longer in front of a blank page. This is the reason why the application is being aimed at people who are starting to study Database queries.

Keywords– Database, application, SQL, query, view, MVC.

1 Introducció

Aquest projecte ha sorgit a partir d'una necessitat que vaig patir quan vaig començar a aprendre el llenguatge SQL, ja que em resultava complicat recordar els operadors que havia d'utilitzar en cada tipus de consulta i recordar l'estructura de les consultes. De manera que a l'hora de triar una proposta per a realitzar el TFG vaig veure l'oportunitat de cobrir aquesta necessitat, ja que igual que jo la vaig tenir en el seu moment, és molt probable que hi hagi més persones que tinguin aquesta necessitat. Per això l'aplicació va dirigida a l'àmbit de l'educació, als estudiants que s'estan iniciant en el món de les Bases de Dades i que comencen a aprendre el llenguatge de programació SQL i necessiten comprovar si les operacions que realitzen estan ben programades. Tot i que l'aplicació també pot ser d'utilitat per altres persones que no necessàriament siguin estudiants però que necessiten realitzar alguna operació en una base de dades.

El projecte consisteix en una aplicació web que permeti realitzar operacions SQL de manera gràfica, és a dir, sense haver de programar l'operació. L'aplicació consisteix en una interfície web en la qual el primer pas que ha de fer un usuari és resgistrar-se i crear un compte amb el que haurà d'iniciar sessió. Posteriorment l'usuari ha de carregar un *script* per crear la base de dades sobre la qual vol realitzar les consultes mitjançant una funcionalitat d'aquesta aplicació. Quan la base de dades sobre la que es volen fer les consultes ja està disponible, quan l'usuari vulgui realitzar una operació apareix un desplegable on apareixen totes les bases de dades que té disponibles i n'ha de triar una.

Cal dir que una consulta SQL pot arribar a tenir moltes opcions diferents, per tant, en el motor de consultes de l'aplicació és on resideix la dificultat d'aquest projecte. També cal tenir en compte les vulnerabilitats de seguretat que pot tenir una aplicació web i implementar les funcions necessàries perquè l'aplicació sigui segura.

• marc.alvarezr@gmail.com
• Enginyeria del Software
• Treball tutoritzat per: Oriol Ramos Terrades (Departament de Ciències de la Computació)
• Curs 2016/17

El desenvolupament de l'aplicació ha d'utilitzar l'arquitectura Model Vista Controlador, ja que es tracta d'una aplicacó que interacciona molt amb la base de dades.

2 Objectius

A continuació es mostren els objectius referents al desenvolupament de l'aplicació:

1. Desenvolupar una aplicació web per a ajudar als estudiants que s'estiguin iniciant en el llenguatge SQL i també per a persones que vulguin realitzar operacions SQL i no saben com programar-les.
2. Conèixer i saber desenvolupar una aplicació web amb l'arquitectura Model Vista Controlador.
3. Conèixer i implementar mesures de seguretat en aplicacions web per evitar injeccions de codi entre d'altres problemes de seguretat.
4. Conèixer i implementar l'arquitectura REST en l'aplicació web.

En la Taula 1 podem veure una classificació dels objectius segons la seva prioritat, de manera que els objectius crítics són els objectius essencials i els objectius secundaris els menys prioritaris.

TAULA 1: Classificació Dels Objectius

Objectiu	Crítics	Prioritaris	Secundaris
1	X		
2		X	
3		X	
4			X

3 Metodologia

S'ha utilitzat la idea de metodologia àgil i s'ha adaptat al nostre cas particular, ja que es tracta d'un projecte en el qual es treballa individualment i jo he d'assumir tots els rols. En una metodologia àgil el projecte es divideix en blocs de tasques a fer en un temps determinat. Com podem veure en la Fig. 1, aquestes iteracions de temps poden ser de 2, 3 o fins a 4 setmanes i s'ha d'assignar el volum de tasques adients per a poder assolir-les dins del temps establert i a més a més que la qualitat de la feina sigui bona.

En cada iteració es realitza una entrega del producte final (a mesura que van passant les iteracions el producte final es va millorant de forma incremental).

Activitats a realitzar per treballar amb una metodologia àgil:

- Captura de requisits: s'ha de realitzar una llista amb requisits que ha de tenir el projecte. Dins d'aquesta llista s'han de prioritzar els requisits.
- Planificació de la iteració: en cada iteració s'han de seleccionar i acordar els requisits que ens comprometem a assolir per la iteració següent. S'ha de tenir en compte la càrrega de treball que suposa cada requisit.

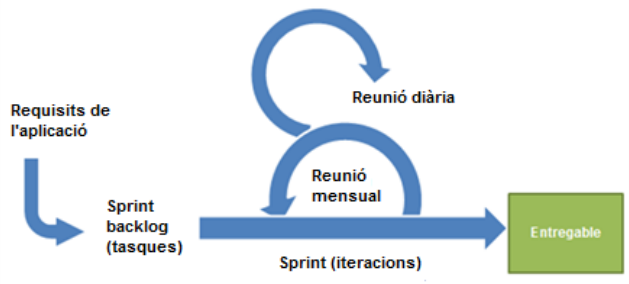

Fig. 1: Esquema de metodologia àgil [1]

- Execució de la iteració: durant aquesta fase es tracta de realitzar una reflexió diària sobre el treball realitzat, els problemes que s'han trobat i el treball que tenim pendent, i seguidament organitzar-nos per ser el més productius possibles.
- Finalització de la iteració: en aquesta etapa s'ha de presentar al client (en el meu cas al tutor del TFG) el treball realitzat. S'ha de reflexionar sobre com millorar la manera de treballar i mirar si s'han assolit els objectius que s'havien plantejat en la reunió anterior. En aquesta mateixa reunió es torna a fer una planificació de la següent iteració.

Abans de realitzar l'entrega final també és necessari realitzar una etapa de test del software, per comprovar que el software funciona com s'espera i està lliure d'errors [2].

Cal afegir que durant l'etapa de desenvolupament cal mantenir un control de versions. S'han d'anar realitzant *commits* i d'aquesta manera sempre podem tornar a alguna versió anterior del codi.

4 Riscs

A continuació es mostra un seguit de possibles riscs que poden aparèixer durant l'implementació i posterior manteniment de l'aplicació. Alguns dels riscs que s'esmenten a continuació són riscs comuns en el desenvolupament d'aplicacions web, per tant, cal fer una distinció entre riscs comuns i riscs específics d'aquesta aplicació.

Alguns dels riscs comuns que poden existir en el desenvolupament d'una aplicació web són els següents:

1. No fer un bon ús del software de control de versions. Tenir un bon control de versions és essencial a l'hora de desenvolupar codi, ja que permet tenir còpies de les diferents versions del software que es guarden al llarg del temps i recuperar-les si s'escau.
2. Que la fase de proves s'endarrereixi massa. Ja que com el llenguatge SQL té moltes condicions i moltes maneres diferents de fer les consultes, és molt probable que no es puguin realitzar suficients proves per assegurar un software completament lliure d'errors.
3. Cross-Site Scripting, que és un tipus d'atac que permet la introducció de codi JavaScript per evitar mesures de control de l'aplicació, també permet l'obtenció de variables que es passen per la URL [3].

4. Vulnerabilitats d'injecció de codi: com el projecte és una aplicació que opera sobre bases de dades, aquest és un risc que cal tenir molt en compte, ja que l'injecció de codi SQL s'usa per introduir instruccions de manera il·lícita i alterar el contingut de les consultes.

5. Mala configuració de la seguretat. Si hi ha forats de seguretat, qualsevol usuari amb males intencions pot fer caure el servei o esborrar i modificar dades.

6. Falta de controls d'accés per funció: protecció davant l'accés a funcions referenciades per la URL.

7. Usar components amb vulnerabilitats conegudes: si s'usa algun framework s'ha d'assegurar que es disposa de la versió adequada de les llibreries.

8. Mostrar dades sensibles (com per exemple el correu de l'usuari o la contrasenya).

   Els riscs específics que pot tenir el desenvolupament d'aquesta aplicació en particular són:

9. No complir amb algun dels objectius crítics. Si no es compleix l'objectiu crític el projecte fracassa, per tant, s'ha de donar prioritat a aquest objectiu sobre la resta.

10. No acabar el projecte dins del termini establert. Com s'ha dit, pot ser que la fase de proves s'allargui tant que endarrereixi la data de termini.

11. Que l'implementació de l'arquitectura REST endarrereixi l' etapa de codificació. Com és una arquitectura que mai he implementat, requereix un temps d'investigació i implementació força elevat.

En la Taula 2 podem veure una classificació dels riscs segons la probabilitat que tenen d'aparèixer i segons l'importància d'aquests, de manera que un risc amb severitat crítica tindrà un impacte molt major que un amb severitat marginal.

TAULA 2: CLASSIFICACIÓ DELS RISCS

Probabilitat Severitat	Poc Probable	Probable	Molt Probable
Marginal			7
Mitja	5	11	2
Crítica	1, 3, 4, 8, 10	6, 9	

5 REQUERIMENTS

Diferenciarem entre els requeriments funcionals i els no funcionals.

5.1 Requeriments funcionals

Els requeriments funcionals són requeriments que afecten al comportament del sistema:

- Cal que la pàgina principal compti amb un *login* per tal de poder iniciar sessió, o en cas de no tenir un compte poder crear-ne un.
- Cal que un cop l'usuari ha estat autenticat, es mostri la pàgina principal amb totes les opcions disponibles: carregar una base de dades, les diverses operacions que es poden realitzar o accedir a operacions freqüents que han estat guardades.
- Poder carregar la base de dades sobre la qual es realitzaràn les consultes. D'aquesta manera, cada usuari pot crear la base de dades que vol utilitzar, per exemple en el cas dels estudiants de la Universitat Autònoma se'ns proporciona una base de dades que hem d'utilitzar per realizar les consultes.
- Cal que si l'usuari no ha carregat cap base de dades sobre la qual treballar surti un missatge d'advertència informant que ha de carregar una base de dades si vol realitzar alguna operació, ja que haver carregat com a mínim una base de dades és indispensable per poder realitzar consultes.
- Quan l'usuari vol realitzar una operació, l'aplicació ha de permetre la visualització de les taules i els camps de la base de dades prèviament seleccionada de forma amigable per poder seleccionar els que utilitzarem, posteriorment, per a cada tipus de consulta ha d'oferir l'estructura que té i ha de permetre introduir totes les condicions necessàries de la consulta mitjançant elements web.
- Quan l'aplicació mostra el resultat, també ha d'oferir l'opció de guardar la consulta per poder-la executar sense haver de repetir tot el procés.
- Poder realitzar diferents operacions SQL (agregacions, diferències i divisions).

5.2 Requeriments no funcionals

Els requeriments no funcionals són aquells requeriments que no afecten al comportament del sistema:

- Totes les dades sensibles, com les contrassenyes, han de ser xifrades abans de ser guardades en la base de dades, d'aquesta manera l'aplicació és més segura.
- Cal que l'aplicació sigui robusta: des d'evitar errors de codificació fins evitar problemes com la injecció de codi. La injecció de codi pot causar molts problemes a una aplicació web, es tracta d'introduir codi SQL en algun camp de text per tal d'accedir a la base de dades de l'aplicació i corrompre dades o realitzar altres accions.
- Cal que l'aplicació sigui escalable, és a dir, que s'ha de desenvolupar l'aplicació de manera que en un futur es puguin afegir funcionalitats i pugui suportar un nombre d'usuaris creixent.
- El disseny de l'aplicació ha de ser intuïtiu i amigable, l'usuari no ha de tenir cap inconvenient en poder realitzar les funcionalitats que desitja, és a dir, la usabilitat de l'aplicació ha de permetre a l'usuari saber en tot moment quina acció ha de realitzar per assolir el seu propòsit.

- Com tota aplicació que tacta amb dades de caràcter personal, ha de complir amb la legalitat vigent. Així doncs s'ha de complir amb tots els requeriments vigents de la LOPD i d'altres lleis sobre l'ús i tractament de dades.

6 Planificació

A continuació es llisten les diferents fases en què es divideix el projecte i les activitats que s'han de realitzar en cadascuna d'aquestes. Les tasques que s'han realitzat es poden classificar en tasques de disseny i estructura de l'aplicació i tasques del motor de consultes a més a més de classificar-les en fases.

En cada fase del projecte s'ha reservat cert temps per realitzar l'informe inicial i els informes de seguiment del projecte.

6.1 Tasques de disseny i estructura de l'aplicació

En aquesta categoria hi ha les tasques de les dues primeres fases i la primera meitat de les tasques de la tercera fase, són les tasques de disseny i desenvolupament de l'interfície i l'arquitectura de l'aplicació.

Fase 1 (setmanes 1 a 4): en aquesta primera fase s'ha fet una captura de requisits i s'ha realitzat una planificació del projecte per tal de tenir clars els objectius, els riscs, i tot el necessari per poder organitzar millor les tasques que s'han de realitzar en cada fase del projecte.

Fase 2 (setmanes 5 a 9): la segona fase ha consistit en primer lloc, en realitzar el disseny de l'aplicació, és a dir, decidir quins apartats conté l'aplicació i com distribuïrlos en la pàgina web. Posteriorment s'ha configurat la base de dades i s'han creat les taules necessàries per poder desenvolupar el projecte.

El següent pas ha estat desenvolupar l'estructura de l'aplicació web, que s'ha desenvolupat mitjançant l'arquitectura Model Vista Controlador. Aquesta tasca ha consistit en crear una pàgina inicial i les vistes que necessitarem per començar a desenvolupar el projecte.

Posteriorment s'ha desenvolupat el registre d'usuaris i l'inici de sessió, i per acabar, s'ha desenvolupat la funcionalitat de carregar un *script* a la base de dades.

Fase 3 (setmanes 10 a 15): les tasques d'aquesta fase han estat l'implementació de mesures de seguretat web i la funcionalitat de guardar consultes freqüents.

6.2 Tasques del motor de consultes

En la fase 3 també s'ha desenvolupat una tasca de desenvolupament del motor de consultes de l'aplicació. Aquesta tasca ha estat el desenvolupament de la funcionalitat per poder realitzar agregacions.

Fase 4 (setmanes 16 a 19): en la fase 4 s'han implementat les funcionalitats per poder realitzar diferències i divisions, primer s'ha desenvolupat la diferència perquè per realitzar les divisions, al tenir una estructura similar a la de la diferència, s'ha pogut aprofitar part del codi.

Durant aquesta fase també s'ha de realitzar un test del software desenvolupat.

Fase 5 (setmanes 20 a 21): en aquesta última fase s'ha realitzat l'article i la presentació d'aquest projecte.

En la Fig. 2 podem veure una imatge de la planificació durant aquests mesos.

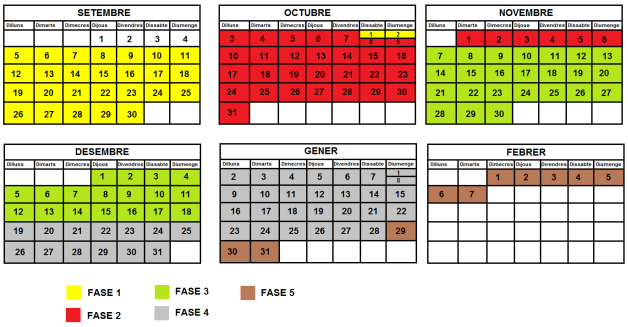

Fig. 2: Calendari de la planificació

7 Estat de l'art

En aquest apartat s'explica l'estat de l'art, és a dir, quines tecnologies hi ha actualment que permeten realitzar un projecte com el que s'ha plantejat. També es llisten les tecnologies que he utilitzat i les raons per les quals les he escollit.

En referència a la realització d'operacions SQL de manera que no s'hagin de programar, a continuació es mostren algunes opcions que ens permeten realitzar algunes operacions.

Amb PHPMyAdmin ja es poden realitzar certes operacions SQL de manera gràfica, com per exemple insercions, actualitzacions i eliminació de files, columnes i taules de la base de dades. També ens permet realitzar *joins* [4].

Amb SQL Developer també és possible realitzar algunes funcionalitats mitjançant el Query Builder sense haver de programar-les, com per exemple *select*s amb *joins* o amb producte cartesià i amb la possibilitat de fer order by [5].
Les tecnologies que he utilitzat pel desenvolupament de l'aplicació són les que s'esmenten a continuació:

Els llenguatges de programació amb els quals s'ha desenvolupat el projecte són diversos llenguatges web, com PHP, JavaScript, JQuery, AJAX i HTML5.

Els recursos que es requereixen per dur a terme aquesta aplicació són bàsicament de software (exceptuant un ordinador). Els recursos de software necessaris són els que s'esmenten a continuació:

Netbeans 8.0.2 com a editor de codi, ja que és de codi lliure, fàcil d'utilitzar, ofereix la possibilitat de tabular el codi automàticament, ofereix l'opció d'autocompletar el codi i té incorporada l'eina de control de versions Subversion [6].

També hi ha la possibilitat d'usar eclipse, ja que també és gratis i fàcil d'usar, però m'he decantat per Netbeans perquè ja l'havia utilitzat i em sembla un entorn més senzill i en el qual puc desenvolupar codi més ràpidament.

Subversion per al control de versions. He escollit aquesta eina perquè Netbeans la incorpora i és gratuïta [7]. La veritat és que GitHub és una eina molt bona per al control de versions i ja l'havia usat anteriorment, però sempre he volgut provar una eina com Subversion. La diferència principal entre aquestes dues eines és que GitHub és un sistema

de control de versions distribuït i tenim una còpia local del repositori, mentre que Subversion és un sistema de control de versions centralitzat i amb el qual no tenim un còpia local del repositori.

L'arquitectura Model Vista Controlador ens permet dividir el desenvolupament en tres parts. La vista és on es desenvolupa la part que veurà l'usuari, és a dir, l'aspecte que tindrà l' interfície. El model és on es desenvolupen les funcions d'accés a les dades i també d'actualització d'aquestes. Les dades s'envien del model a la vista perquè siguin mostrades per pantalla. Per últim, el controlador fa d'intermediari entre la vista i el model. A més a més, aquesta arquitectura pot ajudar a desenvolupar l'aplicació de manera que sigui escalable. En la Fig. 3 podem veure un esquema del comportament d'aquesta arquitectura.

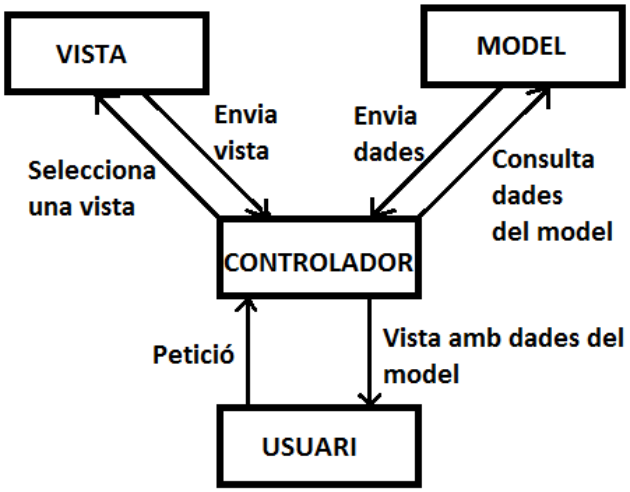

Fig. 3: Esquema de comportament MVC

El framework CodeIgniter, que ajuda a seguir una codificació Model Vista Controlador (MVC) i permet codificar les vistes de forma responsive, és a dir, que el contingut s'adapta al tamany de la pantalla [8]. Un altre framework que s'ha contemplat és Laravel, però he escollit Codeigniter perquè és un framework més lleuger i fàcil d'aprendre, per tant, crec que és més adient tractant-se d'un projecte amb una data d'entrega ajustada.

El framework bootstrap que incorpora certs estils CSS per poder fer l'aplicació més amigable als usuaris [9]. Es tracta d'un framework que no requereix cap instal·lació ni configuració i que permet desenvolupar l'aplicacoó de manera responsive, és a dir, adaptable al tamany de la pantalla del dispositiu des del qual es visualitza.

WampServer per proporcionar els serveis necessaris perquè l'aplicació funcioni correctament (com PHP, MySQL i Apache), l'he triat perquè té una interfície intuïtiva.

Com a SGBD s'utilitza PHPMyAdmin, ja que és una eina de codi lliure, és més intuïtiva que altres eines com PostgreSQL i a més a més cobreix totes les necessitats del projecte. Cal afegir que al ser un projecte que requereix certa immediatesa, PHPMyAdmin ens interessa més perquè no cal instal·lar i configurar aquest SGBD, per tant ens estalviem un temps molt valuós que podem aprofitar per a desenvolupar el software.

8 DISSENY

8.1 Arquitectura física

Com podem observar en la Fig. 4, l'arquitectura consta d'un o més clients que realitzen peticions, un servidor web, que distribueix les pàgines d'informació que sol·liciten els clients i una connexió de xarxa que permet la comunicació entre els clients i el servidor web mitjançant el protocol HTTP.

També és necessari un servidor de base de dades que proporcioni les dades al servidor web perquè les mostri al client. Aquest servidor de base de dades té una base de dades on es guarda la informació dels usuaris, les consultes que guarda cada usuari i les bases de dades que carrega cada usuari. El servidor de base de dades també conté les bases de dades que els usuaris volen utilitzar, és a dir, les bases de dades que carreguen els usuaris mitjançant la funcionalitat de carregar un *script* [10].

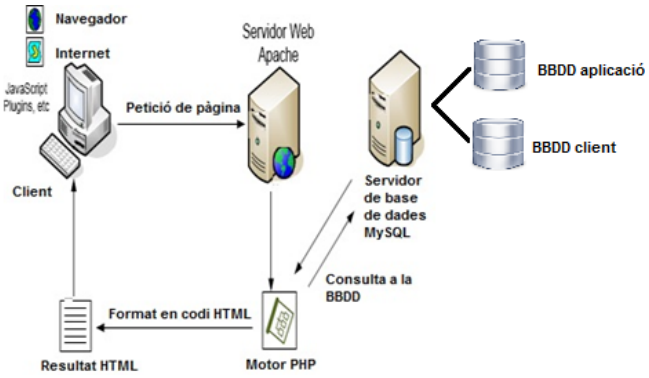

Fig. 4: Arquitectura física

8.2 Diagrama d'estats de l'aplicació

En aquest diagrama es tracta d'explicar, de forma visual, els diferents estats pels quals pot passar l'usuari i les diferents accions que pot realitzar en cadascun d'aquests estats. En el diagrama que podem veure en la Fig. 10 de l'apèndix A1 es plasmen les accions crítiques amb les que compta l'aplicació, és a dir, per simplificar el diagrama no hi apareixen les vistes de informació general sobre l'aplicació o la d'informació de contacte, ja que s'hi pot accedir des de qualsevol estat i no són accions rellevants del projecte.

Els passos que ha de seguir un usuari per utilitzar l'aplicació són els que s'esmenten a continuació:

- Registrar-se i iniciar sessió.
- Carregar una base de dades mitjançant un *script*.
- Seleccionar un tipus d'operació (agregació, diferència o divisió).
- Seleccionar una base de dades del llistat de bases de dades que té disponibles.
- L'aplicació mostra totes les taules de la base de dades que ha seleccioant l'usuari i els camps que contenen. L'uauri ha de seleccionar les taules i els camps que utilitzarà en la consulta.

- L'aplicació proporciona l'estructura de la consulta. L'usuari ha de seleccionar els camps del *select* de la consulta, introduir les condicions de la consulta i tot el necessari per obtenir el resultat desitjat.
- L'aplicació mostra la consulta per pantalla i permet a l'usuari realitzar operacions d'agregació sobre els camps del *select*.
- Per últim l'aplicació mostra el resultat de la consulta i dóna l'opció de guardar la consulta a l'usuari.
- L'usuari pot accedir a l'apartat de consultes freqüents per executar qualsevol de les consultes que ha guardat sense haver de repetir tot el procés anterior.

8.3 Disseny de la base de dades

El primer que cal tenir clar és que hi ha dues bases de dades, una pròpia de l'aplicació per tal de gestionar els usuaris i les operacions SQL generades per cada usuari, i l'altre que serà la que cada usuari carregarà mitjançant l'aplicació i sobre la que realitzarà les operacions SQL.

El disseny de la base de dades d'aquesta aplicació només requereix tres entitats, una d'usuaris per poder gestionar els usuaris i les sessions, una segona entitat de consultes per poder guardar les consultes freqüents que vulgui guardar cada usuari i una tercera entitat per emmagatzemar els noms de les bases de dades que ha carregat cada usuari.

En la Fig. 5 es mostra el disseny ER de la base de dades de la aplicació:

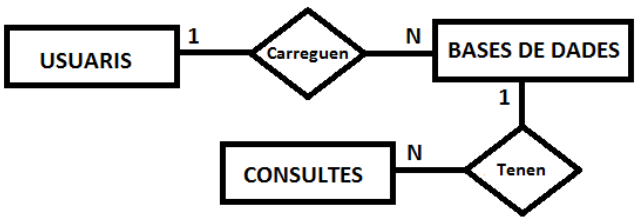

Fig. 5: Disseny ER

9 DESENVOLUPAMENT

Distingim entre el desenvolupament de l'interfície de l'aplicació i el desenvolupament del motor de consultes.

9.1 Desenvolupament del disseny i estructura de l'aplicació

El primer que s'ha fet ha estat instal·lar i configurar el software necessari i crear i configurar la base de dades.

El següent pas ha estat implementar la aplicació mitjançant l'arquitectura Model Vista Controlador: en la carpeta de vistes hi ha els arxius php que contenen el codi HTML del que es mostrarà per pantalla, en la carpeta de models hi ha els arxius php que contenen funcions que operen amb la base de dades i en la carpeta de controladors és on es realitza l'interacció entre els models i les vistes. L'avantatge de treballar amb el framework Codeigniter és que l'estructura de directoris del projecte ja està configurada per treballar amb l'arquitectura MVC.

La primera vista que s'ha implementat ha estat la vista de la pàgina principal, des de la qual es poden realitzar les següents accions: registrar-se, iniciar sessió, consultar informació sobre l'aplicació, consultar l'apartat de contacte, carregar una base de dades, anar a la funcionalitat de realitzar operacions SQL o anar a l'apartat de consultes freqüents.

La següent vista que ha estat implementada ha estat la vista del registre. Si és el primer cop que l'usuari utilitza l'aplicació, s'ha de crear un perfil. L'usuari ha d'introduir les dades que es requereixen i s'han implementat unes regles de validació del formulari de registre, de manera que si les dades no són correctes no s'introdueixen a la base de dades i a més a més ens apareix un text dient que el formulari no és correcte.

Després s'ha implementat el *login*. Des d'aquesta vista també podem accedir a la vista per a poder registrar-nos en cas que no tinguem un compte creat. Posteriorment es modifiquen les vistes de manera que quan un usuari inicia sessió, apareix un missatge de benvinguda i un *link* per poder tancar sessió.

En aquestes vistes que contenen formularis s'ha implementat una mesura de seguretat contra injecció de codi que evita certs caràcters utilitzats en aquestes pràctiques.

Posteriorment s'ha realitzat la funcionalitat de poder carregar un *script* a la base de dades sobre la qual es realitzaran les operacions. Aquesta funcionalitat consisteix en un formulari que permet introduïr el nom de la base de dades que es vol crear i adjuntar un *script* SQL que conté la creació de les taules de la base de dades. Primer es crea la base de dades amb el nom que ha introduït l'usuari i posteriorment s'executa l'arxiu que ha seleccionat l'usuari i que conté la creació de les taules de la base de dades. Si l'arxiu s'executa correctament es mostra un missatge informant de l'èxit, en cas contrari es mostra un missatge d'error.

9.2 Desenvolupament del motor de consultes

Per acabar, s'han implementat les funcionalitats principals de l'aplicació, és a dir, les agregacions, diferències i divisions.

Per a realitzar aquestes funcionalitats, primer s'ha de programar la vista, on hi ha el codi HTML que ens permet implementar un formulari per captar totes les dades necessàries per a generar la consulta i funcions jQuery que permeten donar dinamisme a l'aplicació. Seguidament s'ha de programar el controlador, que organitzarà les dades en estructures òptimes per a enviar-les al model, aquesta estructura de dades es tracta d'un *array* on les claus són el tipus de dada, com per exemple "operador aritmètic" i el valor és el valor que ha introduït l'usuari en el formulari de la vista. De manera que per cada taula de restriccions i per cada subconsulta es construeix un *array* d'aquest tipus.

El següent pas ha estat programar les funcions del model, que són les funcions que generen la consulta segons les dades que ha introduït l'usuari i segons el tipus de consulta que ha triat. Aquestes funcions són cridades des del controlador per obtenir els resultats de la consulta i mostrar-los en una vista.

Tant en la vista de les agregacions com en la de diferències i divisions, les condicions de les restriccions es mostren en una taula dinàmica, sobre la qual es poden afe-

gir i eliminar files dinàmicament. D'altra banda, si en el *dropdown* de l'operador seleccionem l'opció "subconsulta", s'afegeix una fila a la taula per afegir les condicions de la subconsulta, i si volem afegir més condicions a la subconsulta també tenim un botó per a poder afegir files a la subconsulta.

Una altra implementació que s'ha realitzat en aquestes vistes és que en els desplegables que permeten seleccionar els camps només apareixen els camps que corresponen a la taula que hem seleccionat.

També hi ha una validació d'errors que comprova que els camps obligatoris no estiguin buits, de manera que si ens deixem l'operador ens sortirà un error dient l'operador que ens hem oblidat de posar i on es troba (taula i fila) com podem veure en la Fig. 6.

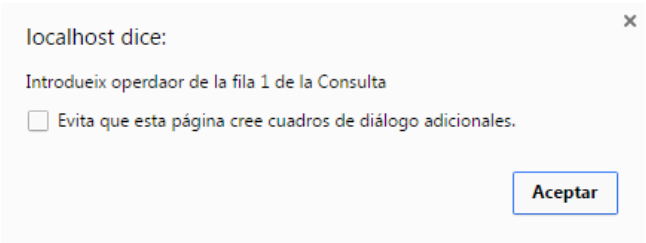

Fig. 6: Missatge d'error

Amb aquestes últimes dues implementacions es minimitzen els errors que pugui cometre l'usuari.

Les consultes SQL permeten realitzar operacions d'agregació sobre els camps del *select* de la consulta, aquesta funcionalitat també s'ha implementat amb una taula dinàmica. El que s'ha fet és una vista a part per poder realitzar aquestes agregacions, de manera que abans de mostrar el resultat, l'usuari veu el codi que ha generat la seva consulta i si vol realitzar operacions d'agregació disposa d'una taula dinàmica per fer-ho. En el model hi ha una funció que s'encarrega de fer el *group by* necessari per realitzar aquesta agregació i executa la consulta per posteriorment mostrar el resultat en la vista de resultats.

10 EXPOSICIÓ DE RESULTATS

En aquesta secció s'exposen els resultats obtinguts en les tasques del projecte, i també els inconvenients que he trobat a l'hora de desenvolupar algunes funcionalitats. Primer s'exposen els resultats de les tasques relacionades amb el desenvolupament de l'interfície i posteriorment les relacionades amb el motor de consultes de l'aplicació.

La desviació principal d'aquest projecte ha estat l'implementació de l'arquitectura REST, en aquesta funcionalitat s'han invertit hores i no acaba de funcionar, així que s'ha passat a la següent tasca per no quedar-nos endarrerits en l'implementació de les funcionalitats principals.

L'altra desviació en la planificació ha estat no poder realitzar una bateria de test cobrint tots els casos possibles, el motiu d'aquesta desviació ha estat subestimar el temps que requereix implementar les divisions. Per tant, s'han realitzat proves amb algunes consultes, però no s'ha pogut realitzar un test exhaustiu del software.

En la tasca de disseny de la base de dades he trobat alguns inconvenients:

Un dels problemes que m'he trobat desenvolupant aquesta funcionalitat és que l'aplicació necessita accedir a dues bases de dades diferents segons la funcionalitat que utilitza l'usuari. De manera que s'ha hagut de tocar l'arxiu de configuració de bases de dades del projecte per fer-ho possible.

Un altre problema que he trobat ha estat el tipus de cotejament de les bases de dades que carreguen els usuaris, és a dir, la base de dades que he prés com a exemple contenia dades amb accents i amb "ç", i això ha donat problemes a l'hora d'operar amb aquests camps.

A continuació es mostren els resultats de les funcionalitats principals. En les agregacions s'ha de procedir de la següent manera:

Si fem clic sobre "OPERACIONS SQL" i seleccionem l'opció "AGREGACIÓ" ens apareix una vista en la qual s'ofereix l'opció de carregar una base de dades si encara no tenim una base de dades sobre la qual treballar. Si l'usuari ja ha carregat alguna base de dades es mostra un desplegable amb totes les bases de dades que ha carregat l'usuari i aquest en pot seleccionar una.

En la següent vista ens apareixen totes les taules que conté la base de dades que ha seleccionat l'usuari amb els seus camps corresponents i l'usuari ha de seleccionar totes les taules i els camps que es veuran involucrats en la consulta.

En la vista que apareix es mostra l'estructura de la consulta, s'ha de començar seleccionant els camps del *select* i les condicions de les restriccions.

En la Fig. 7 es pot apreciar l'aspecte que tenen les taules dinàmiques.

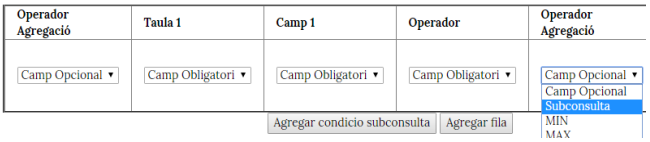

Fig. 7: Taula dinàmica

En aquest cas, en comptes d'existir una segona o una tercera subconsulta, tenim un operador *having* que s'ha implementat de la mateixa manera que les restriccions, és a dir, amb una taula dinàmica. Quan seleccionem l'operador de subconsulta s'afegeix una fila per introduir les condicions de la subconsulta. Hi ha l'opció de fer una agrupació sobre el *having*. Com en tots els tipus de consultes, també hi ha l'opció de odernar els resultats mitjançant l'operador *order by* i seleccionar si volem que s'ordenin els camps de manera ascendent o descendent.

Per guardar una consulta s'han de seguir aquests passos:

Quan ens mostra el resultat de la consulta, podem guardar la consulta donant-li un nom.

Si accedim a l'apartat "CONSULTES FREQÜENTS" podem seleccionar qualsevol consulta que hem guardat i executar-la sense haver de repetir tot el procès anterior.

Després d'implementar les agregacions i la funcionalitat de guardar les consultes, s'han implementat les diferències:

Quan s'han seleccionat les taules i els camps involucrats en la consulta i s'ha confirmat, apareix una vista en la qual s'han de seleccionar els camps i taules del *select*, les condicions de les restriccions de la consulta, l'operador sobre el

qual es vol aplicar la diferència, el *select* de la segona taula i les condicions de les restriccions de la segona taula.

Com veiem en la Fig. 8, el següent pas per a realitzar la diferència és seleccionar el camp sobre el qual volem aplicar la diferència, seleccionar l'operador i seleccionar el camp del *select* de la segona taula. En aquesta imatge també es pot apreciar que la manera de seleccionar les taules i els camps és mitjançant un *checkbox*.

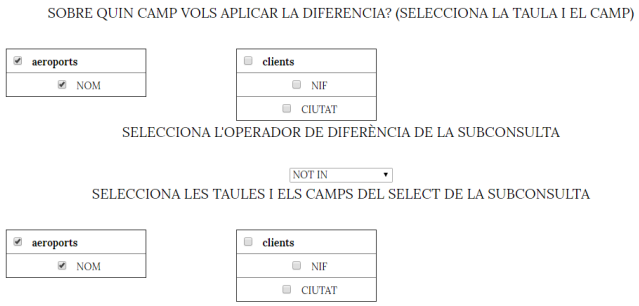

Fig. 8: Seleccionar camp diferència i operador

Ara hem de posar les condicions de les restriccions de la segona taula de la diferència.

Quan enviem la consulta, si aquesta no té resultat, ens mostra un missatge informant que la consulta no té resultat. Si la consulta retorna un error mySQL, es mostra la consulta i l'error mySQL. En cas contrari, si tot va bé, ens mostra la consulta i ens permet realitzar operacions d'agregació sobre el *select*. Com veiem en la Fig. 9, es pot fer una operació d'agregació sobre numero d'aeroports.

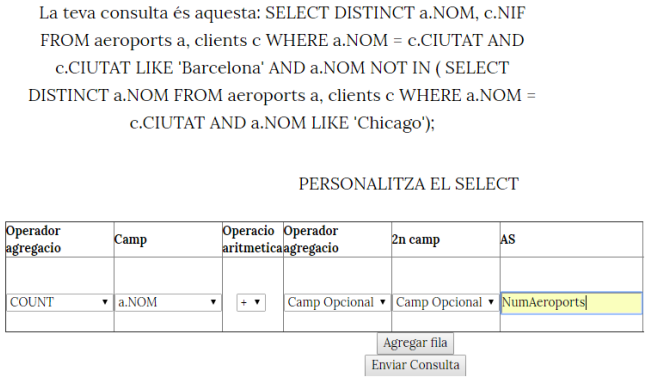

Fig. 9: Agregació del select

En la vista de resultats apareix el camp que hem posat en la operació d'agregació de la Fig. 9 en la vista de resultats. En la vista de resultats també apareix l'opció de guardar la consulta, quan fem clic sobre el botó ens surt un missatge comunicant que s'ha guardat correctament.

La darrera tasca del motor de consultes que s'ha implementat han estat les divisions:

Hi ha moltes formes de realitzar divisions en SQL, de manera que he escollit la forma més senzilla, ja que hi ha una manera d'implementar les divisions que és similar a l'implementació de les diferències i s'ha pogut aprofitar gran part del codi de la tasca anterior. De fet, per realitzar les divisions s'ha de fer el mateix que per realitzar una diferència, però ara tindrem dues subconsultes.

Els passos per realitzar una divisió són els mateixos que s'han vist en la diferència, però en aquest cas s'ha de realitzar dues vegades el procés de seleccionar el camp sobre el qual volem aplicar la divisió, l'operador i el camp del *select* de la subconsulta.

Com en la tasca anterior, es mostra la vista de personalització del *select*. és opcional personalitzar-lo, simplement s'ofereix aquesta opció per si l'usuari necessita realitzar alguna operació d'agregació sobre el *select*.

Si s'envia la consulta ens mostra el resultat de la mateixa manera que hem vist en el cas de la diferència.

11 Futures implementacions

De cara al futur es poden implementar algunes millores en aquesta aplicació.

Crec que la primera millora que s'hauria d'implementar és aconseguir que l'aplicació pugui ser usada per persones que no han programat mai en SQL, ja que l'aplicació que s'ha desenvolupat requereix que els usuaris tinguin certs coneixements de programació per introduïr correctament els *joins* i les restriccions de les consultes.

Una altra millora pot ser implementar l'arquitectura REST. Aquesta arquitectura permet que el servidor no hagi de recordar cap estat per comunicar-se amb els clients, per tant, a l'hora de fer l'aplicació escalable és un element molt interessant.

Realitzar una fase testing i posteriorment llençar una versió beta perquè els usuaris reportin tots els errors que no s'han pogut trobar en la etapa de proves, ja que hi ha tantes variables i tantes combinacions possibles que és molt difícil realitzar proves amb totes les combinacions possibles.

Amb el desenvolupament actual, si dos usuaris carreguen una base de dades amb el mateix nom podria haver conflictes, de manera que s'hauria de millorar aquesta funcionalitat.

Millorar els estils CSS de l'aplicació permet millorar l'aparença de l'aplicació perquè sigui més atractiva per als usuaris. En aquest projecte m'he centrat en la funcionalitat de l'aplicació.

Una millora que no pot faltar és implementar més tipus d'operacions. En el món de la programació SQL hi ha molts tipus d'operacions i en futures implementacions es pot ampliar el numero de tipus d'operacions que es poden realitzar amb aquesta apicació.

Per últim crec que cal millorar la usabilitat. Tot i que els elements dinàmics i el control d'errors fan que la usabilitat sigui bona sempre es pot millorar.

12 Conclusió

Desenvolupant aquest projecte m'he adonat de l'importància que hi ha en la fase de planificació, ja que fent una bona feina en aquesta fase després és més probable que s'assoleixin els objectius proposats.

Cal dir que crec que he encertat en les tecnologies i arquitectures que s'han utilitzat per desenvolupar l'aplicació, ja que la majoria no requerien d'instal·lació i configuració per poder utilitzar-les i això m'ha estalviat temps.

Durant tot el procés s'han seguit les etapes de l'Enginyeria del Software, excepte la de proves, que hauria d'haver estat més extensa però com ja s'ha explicat s'ha hagut d'escurçar. Això també m'ha remarcat l'importància d'aplicar aquestes etapes a l'hora de desenvolupar software.

Crec que gràcies al control d'errors a l'hora d'omplir les condicions de les restriccions i que l'aplicació contingui elements dinàmics com afegir i eliminar files o que només es mostrin els camps de la taula que hem seleccionat en els *dropdowns*, minimitza els errors que pugui cometre l'usuari i fa que l'aplicació sigui més amigable.

La veritat és que l'usuari que utilitzi l'aplicació haurà de tenir coneixements de SQL, ja que ha d'introduir les condicions de les restriccions per a realitzar els *joins*.

Tot i que per a usuaris no experts pot ajudar tenir l'estructura de la consulta i visualitzar els camps de les taules, crec que per a usuaris experts és molt més ràpid i pràctic programar directament la consulta.

També m'agradaria mencionar que aquest projecte es podria ampliar molt, però que per raons de limitació de temps s'ha hagut d'acotar a certes funcionalitats i deixar la resta per a futures implementacions.

Agraïments

Vull agrair a Oriol Ramos Terrades els seus consells i l'interès mostrat durant el desenvolupament d'aquest treball.

Referències

[1] Mprende. (1 de Abril de 2015). Agile, Scrum y Lean Startups: Metodologías de desarrollo ... - Mprende. Recuperat el 1 de Octubre de 2016, de http://mprende.co/gesti%C3%B3n/agile-scrum-y-lean-startups-metodolog%C3%ADas-de-desarrollo-%C3%A1gil-de-productos

[2] DesarrolloWeb.com. (30 de Septiembre de 2016). Metodologías ágiles para el Desarrollo de Software. Recuperat el 1 de Octubre de 2016, de http://www.desarrolloweb.com/manuales/metodologias-agil-desarrollo-software.HTML

[3] SlideShare. (20 de Noviembre de 2015). Los 10 principales riesgos en aplicaciones web #CPMX5. Recuperat el 29 de Septiembre de 2016, de es.slideshare.net/SemanticWebBuilder/los-10-principales-riesgos-en-aplicaciones-web-cpmx5

[4] Contributors, p. (1 de Octubre de 2016). phpMyAdmin. Recuperat el 1 de Octubre de 2016, de https://www.phpmyadmin.net/

[5] Oracle. (24 de Septiembre de 2016). SQL Developer: Building Queries Visually. Recuperat el 29 de Septiembre de 2016, de http://www.oracle.com/technetwork/issue-archive/2008/08-mar/o28SQL-100636.HTML

[6] Oracle. (11 de Septiembre de 2016). NetBeans IDE - Oracle. Recuperat el 29 de Septiembre de 2016, de http://www.oracle.com/technetwork/developer-tools/netbeans/overview/index.HTML

[7] sun.com, t. @. (1 de Octubre de 2016). Recuperat el 1 de Ovtubre de 2016, de Using Subversion Support in NetBeans IDE: https://netbeans.org/kb/docs/ide/subversion.HTML

[8] Johannes Gamperl (Germany), A. P. (30 de Septiembre de 2016). CodeIgniter Web Framework. Recuperat el 30 de Septiembre de 2016, de https://www.codeigniter.com/

[9] LibrosWeb. (6 de Mayo de 2016). Bootstrap 3, el manual oficial - LibrosWeb. Recuperat el 29 de Septiembre de 2016, de https://librosweb.es/libro/bootstrap-3/

[10] Web, P. (4 de 11 de 2013). Programacion Web. Recuperat el 5 de Noviembre de 2016, de https://programacionwebisc.wordpress.com/2-1-arquitectura-de-las-aplicaciones-web/

Apèndix

A.1 Disseny

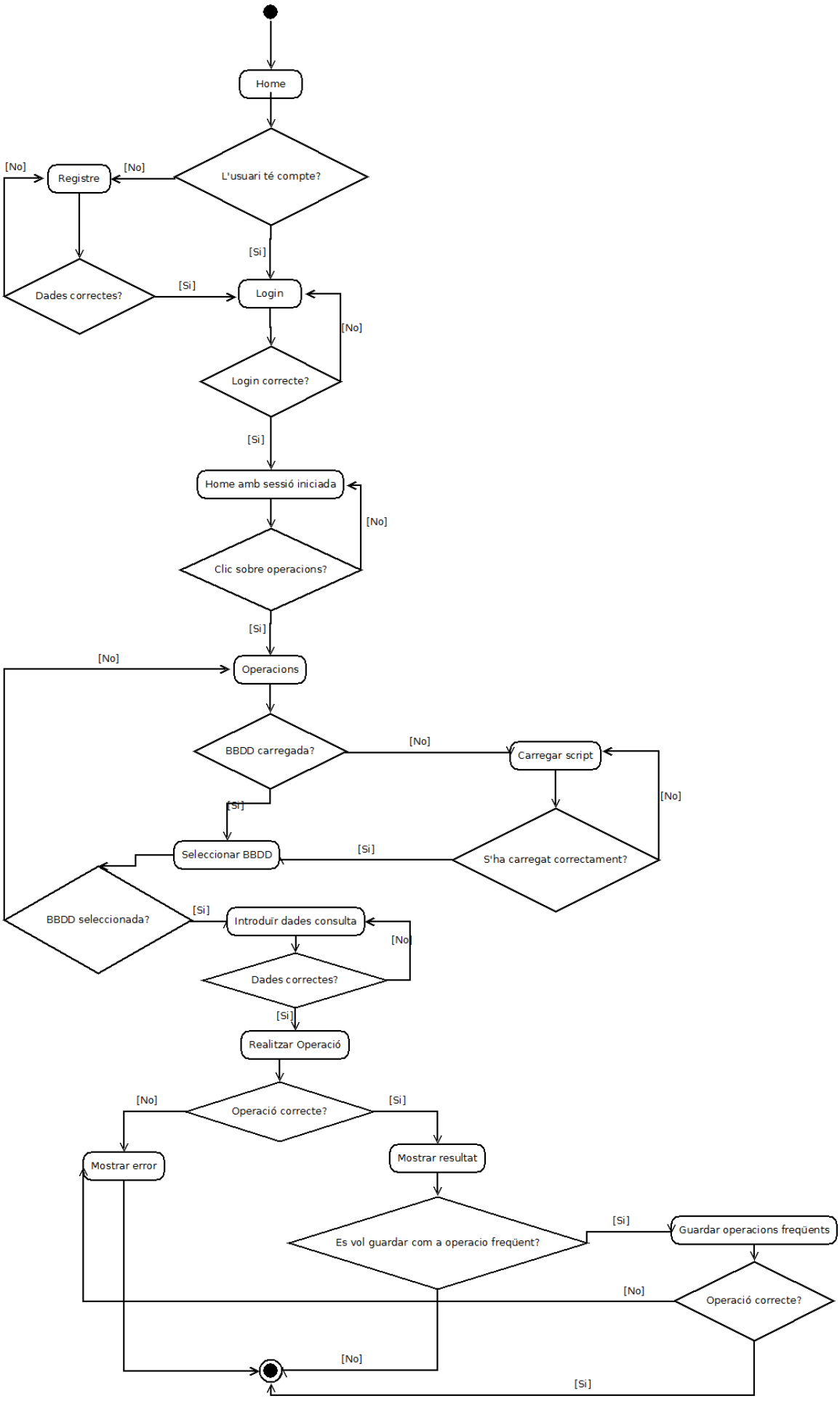

Fig. 10: Diagrama d'estats de l'aplicació