

8-2023

TRANSACTION MANAGEMENT SYSYEM FOR A PUBLISHER

HASSAIN SHAREEF MOHAMMED JR
California State University - San Bernardino

Follow this and additional works at: <https://scholarworks.lib.csusb.edu/etd>



Part of the [Data Storage Systems Commons](#)

Recommended Citation

MOHAMMED, HASSAIN SHAREEF JR, "TRANSACTION MANAGEMENT SYSYEM FOR A PUBLISHER"
(2023). *Electronic Theses, Projects, and Dissertations*. 1779.
<https://scholarworks.lib.csusb.edu/etd/1779>

This Project is brought to you for free and open access by the Office of Graduate Studies at CSUSB ScholarWorks. It has been accepted for inclusion in Electronic Theses, Projects, and Dissertations by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.

TRANSACTION MANAGEMENT SYSTEM FOR A PUBLISHER

A Project
Presented to the
Faculty of
California State University,
San Bernardino

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Hassain Shareef Mohammed

August 2023

TRANSACTION MANAGEMENT SYSTEM FOR A PUBLISHER

A Project
Presented to the
Faculty of
California State University,
San Bernardino

by
Hassain Shareef Mohammed

August 2023

Approved by:

Dr. Ronald Salloum, Advisor, Computer Science and Engineering

Dr. Khalil Dajani, Committee Member

Ernesto Gomez, Committee Member

© 2023 Hassain Shareef Mohammed

ABSTRACT

Managing the day-to-day operations of a publishing house isn't easy. For example, it's hard to keep track of stocks, manage vendor orders, and maintain transaction records. Our project titled Transaction Management System aims to solve these problems by offering a single tool that helps standardize and digitize publishing operations. It assists in automating and improving the efficiency of the various processes involved in a publishing house.

The Transaction Management System is a full-stack web application that can be accessed through an internet browser. The user experience and interface is simple and easy to use so that users can find information quickly and complete tasks easily. The system gets rid of the need for handwritten records and papers, which can take time and lead to mistakes. With this, producers can focus primarily on creating great content and let the system handle the management tasks.

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF FIGURES	vii
CHAPTER ONE: INTRODUCTION	
Background	1
Purpose	2
Scope	3
CHAPTER TWO: PROJECT MANAGEMENT PLAN	
Project Organization	4
Lifecycle Model	4
Risk Analysis	5
Hardware Requirements	6
Deliverables and Schedule	6
Monitoring, Reporting and Control Mechanisms	7
Impact of Project on Individuals and Organizations	8
CHAPTER THREE: REQUIREMENT SPECIFICATION	
Stakeholders of the System	9
Use Case Diagram	9
Non-Functional Requirements	12
CHAPTER FOUR: SYSTEM DESIGN	
Model View Controller Architecture	14
Architectural Model	15

Model	15
View	15
Controller	16
Technologies Used	16
Rationale for Architecture	17
Graphical User Interface	18
Login Page	18
Dashboard	19
Publications	21
Publications Sections	24
Orders	26
Users	28
My Profile	30
Backend	30
Entities	31
Controllers	32
Services	34
Class Diagram	37
Sequence Diagram	38

CHAPTER FIVE: TEST PLAN

Requirements / Specifications Based System	
Level Test Cases	41
User Login Functionality	41

Publication Creation Functionality	43
Publication Creation Section Functionality	44
Publication Editing Functionality	46
Publication Section Editing Functionality	47
View All Publications Functionality	48
Order Creation Functionality	48
User Role Management Functionality	49
Techniques Used for Test Generation	50
Boundary Value Analysis	50
Equivalent Partitioning	51
State Transitioning Testing	51
Use Case Testing	52
REFERENCES	53

LIST OF FIGURES

Figure 1. Use Case Diagram	10
Figure 2. Login Page	19
Figure 3. Dashboard	21
Figure 4. List of Publications	22
Figure 5. Create Publication Form	23
Figure 6. Publication Created	23
Figure 7. Publication Section	24
Figure 8. Create Publication Section Form	25
Figure 9. Publication Section Created	26
Figure 10. List of Orders	26
Figure 11. Publication Id Verification	27
Figure 12. Order Created Successfully.....	28
Figure 13. List of Users	29
Figure 14. Create User Form	29
Figure 15. Update Profile Form	30
Figure 16. Order Class	32
Figure 17. Order Controller Class	34
Figure 18. Order Service Class	36
Figure 19. Class Diagram	37
Figure 20. Sequence Diagram.....	38
Figure 21. Enter Correct Login Credentials	41

Figure 22: Test User Able to Login	42
Figure 23. Enter Invalid Credentials	42
Figure 24. Create Publication with Required Fields.....	43
Figure 25. Create Publication Section Successfully	44
Figure 26. Publication Section Form Validations	44
Figure 27. Create Publication Section Form	45
Figure 28. Publication Section Created Successfully	45
Figure 29. Publication Section Form Validation	46
Figure 30. Publications List	47
Figure 31. Update Publication Section	47
Figure 32. Create Order Form	48
Figure 33. Order Created Successfully	48
Figure 34. List of Orders	49
Figure 35. User Creation with Proper Data	49
Figure 36. List of Users	50
Figure 37. Create Publication with Large Input	50
Figure 38. Create Publication Section with Various Types	51
Figure 39. Editing Publication that is Already Created	52
Figure 40. Creating Order with Zero Price	53

CHAPTER ONE

INTRODUCTION

Background

In recent years, the publishing business has changed a lot because digital technologies have grown so quickly. With the rise of e-books, digital media, and online platforms for sales and distribution, publishers are finding it harder to keep track of their stock, fill orders from distributors, and keep correct records of transactions. Because of these problems, we need a new way to manage transactions that can streamline processes, make them more efficient, and make sure data is safe.

Publishers used to do all of their daily tasks by hand, like keeping track of their stock, filling orders, and making financial deals. These processes often took a long time and were easy to mess up, which made them inefficient and led to mistakes in the records of transactions. Also, they couldn't keep up with the exponential rise in the number of transactions and the complexity of those transactions that the digital change caused.

Enterprise resource planning (ERP) systems, accounting software, and e-commerce platforms are some of the software options that have been suggested to solve these problems. But these choices have a few problems that make them unsuitable for the needs of printing houses. For example, ERP systems are often expensive and hard to change to meet the specific needs of a printing company.

On the other hand, accounting software is mostly used for managing money, and it might not have the features you need for managing goods and fulfilling orders. E-commerce sites, on the other hand, focus on online sales and might not be able to handle the complicated transactions that come with publishing.

Purpose

The goal of the Transaction Management System is to provide a centralized and automatic way for a publishing house to run its day-to-day business. By doing this, the system hopes to solve the problems authors have when trying to keep track of their stock, fill orders, and keep accurate records of their transactions.

One of the main goals of the project is to standardize and digitize the printing process, which will cut down on the need for handbooks and records that are hard to keep track of and prone to mistakes. The system does this by giving real-time information about deals and handling tasks like order processing and managing supplies. By keeping track of sales, product levels, and other important success factors with this data, publishers may be able to make better choices and improve their business.

One of the project's main goals is to make printing processes more productive and efficient. By easing processes and improving routines, the system cuts down on the time and work needed to finish deals. So, producers

might focus more on making content and less on running the day-to-day business of their company.

The design of the Transaction Management System must also think about how to keep data safe. The system protects private data, like transactions and banking information, with cutting-edge security techniques. By doing this, producers can work with more trust and lower the risk of data breaches and other security problems.

Because it was made with latest web development tools, the system is very flexible and easy to change. It can be changed to meet the needs of different printing houses based on their size, scope, and business plan. Because the user experience is friendly and easy to use, users can quickly find information and finish deals.

Scope

The goal of the system is to provide a single place to manage deals, automate processes, improve speed, and keep data safe. The method is flexible and can be changed to fit the needs of different printing houses. Overall, the project's scope includes the whole process of transaction management, from product management to financial transactions, and it aims to give authors a complete answer for the digital age.

CHAPTER TWO

PROJECT MANAGEMENT PLAN

Project Organization

Proper project management is important if you want to finish a job on time and well. The project will go through steps like research, planning, execution, testing, and launch. To keep the job on track, a detailed project plan will be made with tasks, dates, and major steps. A list of what needs to be done will also be included. This list will include the project plan, requirements definition, design papers, source code, and test results. A version control system will also be used to keep track of source code and changes as they happen during development. Last but not least, the project boss will get frequent progress reports to make sure the project is going along on time and that any problems are dealt with as soon as they come up.

Lifecycle Model

Agile is an iterative method that puts a lot of emphasis on how team members and partners work together, are flexible, and change. It involves breaking the project up into smaller, easier-to-manage pieces called sprints. Each sprint produces a working product that can be looked at and changed based on what people say. This process made sure that the project was built in a

way that met their needs and made it easier for partners to talk to each other more often and effectively. Also, by handling possible problems early on in the project's growth, the Agile method made it easy to respond quickly to changes in needs and helped lower risks. Agile is built on two main ideas: continuous release and integration. This means that instead of waiting until the end of the project to test and combine all the parts, the software is made and tried in steps throughout the project. This makes feedback faster, problem-solving easier, and resource management more effective.

Risk Analysis

Risk analysis was an important part of the project management plan. It includes finding possible risks that could stop the project from being successful, figuring out how likely they are and how bad they could be, and coming up with ways to lower them. Since the project involved making software, some of the risks that were brought up were technical, such as software bugs, hardware problems, and problems with platforms not working together. Risks in project management were also pointed out, such as timeline delays, scope creep, and a lack of support from partners.

The risk management approach also included a backup plan for the biggest risks. As an example, I knew that unanticipated tech problems could cause schedule delays, so I made a backup plan that included reducing the size of the project if that was needed to meet the goal. Another backup plan was to add more

resources to the project if it became clear that the team was falling behind schedule.

Hardware Requirements

- Processor: Intel Core i5 or higher
- RAM: 8GB or higher
- Storage: At least 256GB SSD
- Network: Ethernet or Wi-Fi for internet connectivity
- Display: A 15-inch monitor with at least 1920x1080 resolution

Deliverables and Schedule

For project monitoring and timely completion, the current transaction management system project's deliverables and timeline are essential. The following main deliverables for the project have been identified by the project team.

- A detailed system requirements specification document
- A comprehensive design documents
- A fully functional and tested transaction management system
- A user manual for the system
- A final project reports

I also created a project timeline with significant checkpoints and due dates for each product. The timetable is broken down into numerous stages, such as

gathering requirements, designing, developing, testing, and deploying. Each phase has its own timeframe and set of activities, and there are frequent checkpoints and reviews to make sure the project is moving along as planned. They made the decision to adopt project management software that offers real-time task tracking and status updates in order to properly monitor progress and manage the timetable.

Monitoring, Reporting, and Controlling Mechanisms

Monitoring, reporting, and control are important parts of every project, including the project for the transaction management system. These checks make sure that the project is going along as planned and that any problems are found and fixed quickly. To do this, the success of the project must be tracked and reported on a regular basis in reference to the schedule, price, and scope. Systems that regulate make sure that any changes from the plan are fixed with corrective measures. Microsoft Project and other tools for handling projects can be used to keep an eye on processes, make reports, and handle them. Also, holding frequent project status meetings and giving partners information on the project's progress are good ways to let everyone know how things are going and make sure everyone knows about any problems that come up. By putting these processes in place, we can make sure that the project stays on track and that any changes from the original plan are dealt with quickly and effectively.

Impact of the Project on Individuals and Organizations

The effort to develop a transaction management system will have a big influence on both people and businesses. The technology attempts to speed up transaction processing while improving the precision and effectiveness of financial transactions. This technology will benefit people and companies by enabling them to concentrate on other crucial duties, boosting productivity and profitability. Apart from reducing potential mistakes during transaction processing, the system will also provide accurate and trustworthy financial records.

Also, the adoption of this system will open up employment prospects for those who will be in charge of maintaining and supporting it. The initiative will benefit society as a whole by increasing financial transparency and enabling government authorities to efficiently monitor financial activity. The adoption of the transaction management system will, all things considered, benefit people, businesses, and society as a whole.

CHAPTER THREE

REQUIREMENTS SPECIFICATION

Stakeholders of the System

The stakeholders of the system are individuals, groups, or organizations who are directly or indirectly affected by the system or who have an interest in the system. In this project, the stakeholders are:

- Authors or content creators who may use the system to create and publish their work.
- Editors or reviewers who may use the system to review and edit content.
- Distributors or publishers who may use the system to distribute or sell content to customers.
- Administrators or moderators who may use the system to manage user accounts, content moderation, or other administrative tasks

Use Case Diagram

Here is the use case diagram for the Transaction Management System. This describes the high-level use cases for all the stakeholders involved.

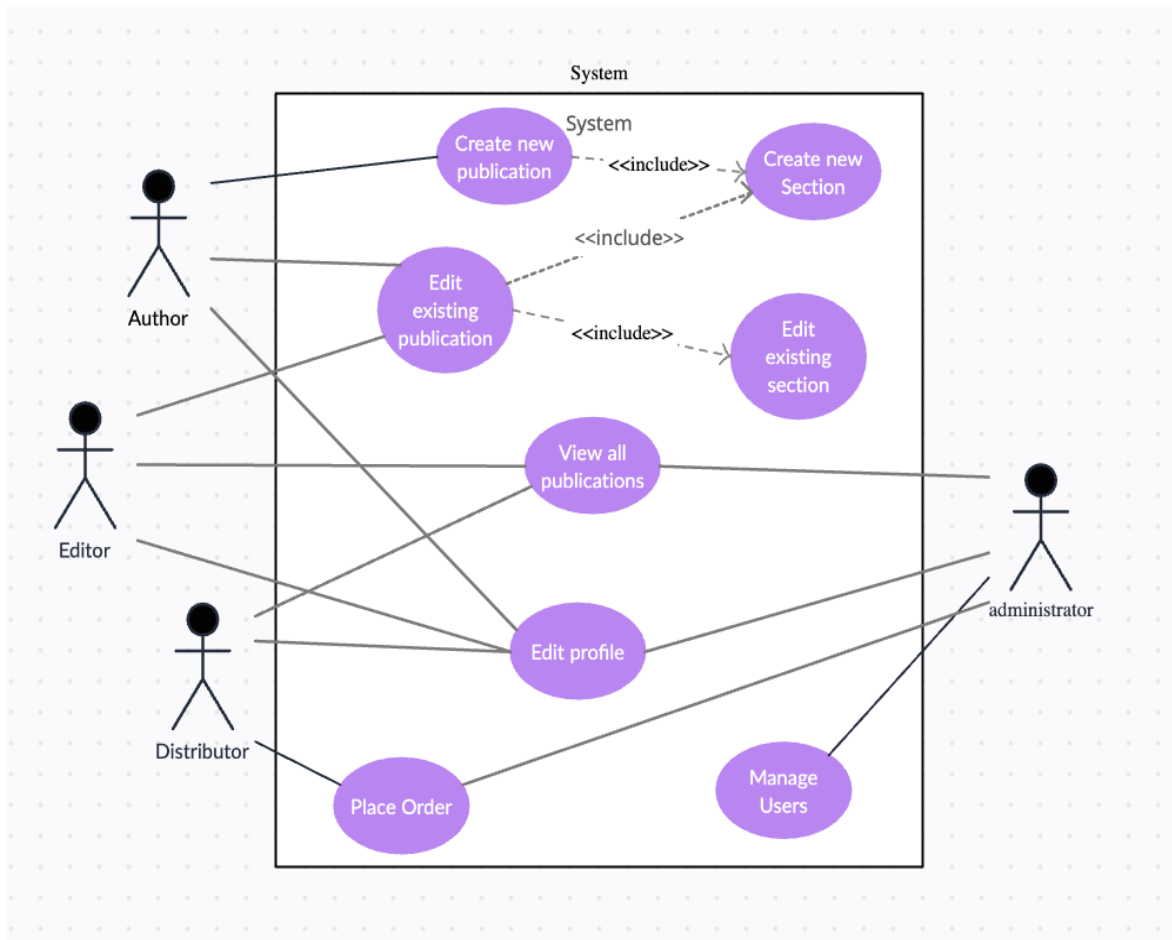


Figure 1. Use Case Diagram

- Authors: Authors can create and manage their publications. They can create new sections under the publication and also edit them. An author can also edit his profile information.
- Editors: An editor can edit the publication created by an author. The editor can view all the sections of a publication, create new and edit already existing sections of a publication.

- Distributor: A distributor can view all the publications present in the system and also place an order to publish any of them.
- Admin: The administrator of the system is responsible for managing all the users present in the system. An administrator can also perform all the operations done by the other roles.

Description of each use case are given below

Create a New Publication: A publication can be anything from a book to a magazine. A new publication can be added to the database by an author. Title, sub-title, category, subjects, and price make up a publication. On starting a new publication, an author enters all of this information.

Edit Existing Publication: An existing publication can be edited by an author or an editor. The interface via which the publication's information is loaded and an author or editor may make changes and persist them are displayed when we click on the edit sign for a publication. Updating a magazine also include adding new sections or altering existing ones.

Create a New Section: A publication is made up of multiple sections. New parts can be added to a publication by the author. An author must input information such as the section type, title, and content when creating a new section.

Edit Existing Section: An existing section may be edited by the author or editor. The interface where the information for the currently loaded section is

displayed when we click on the edit sign for a section. The content can now be edited and saved by the author or editor.

View All Publications: We have the opportunity to view every publication currently in the system from the dashboard. This is a module that is available to all system stakeholders. We provide every publication written by every author in this interface.

Place Order: A distributor has the ability to place orders for any publication they wish to print and distribute. While placing an order, the total cost of the order is calculated and shown to the user.

Manage Users: The admin has the ability to manage all the users present in the system. He can view and edit all the details and role information of all users.

Non-Functional Requirements

- Performance: The system should respond to user requests within two seconds.
- Security: The system needs to guarantee secure transactions, including user authentication and the encryption of private data.
- Scalability: The system needs to be able to accommodate a growing volume of users and transactions without experiencing performance issues.

- Reliability: There should be a maximum of one hour of monthly downtime for maintenance, with the system being online 99% of the time.
- Usability: There should be obvious navigation and only a few steps needed to execute a transaction on the user interface.
- Compatibility: The system should work with a variety of web browsers, operating systems, and other applications that the company uses.
- Maintainability: The system should be simple to keep up-to-date and bug-free, with clear documentation and a modular architecture.
- Data integrity: The system must make sure that every transaction is correctly captured and preserved, and that the necessary backup and recovery procedures are in place.

CHAPTER FOUR

SYSTEM DESIGN

Model View Controller Architecture

The MVC architecture is a program design structure that splits an application into three key parts: the Model, the View, and the Controller. The Controller handles what the user types and how the Model and the View talk to each other. The View is in charge of showing the data to the user in an easy-to-understand way. The Model shows the facts and business strategy of the program.

The Model would be a representation of the database and the business logic used to handle deals, stocks, and orders. This includes doing calculations, making sure all the data is the same, and adding, changing, and removing records from the database. The View would be in charge of showing the user information like transaction records, order progress, and product amounts in a way that is easy to understand. The Controller would then handle user input, such as adding new deals, changing orders, and keeping an eye on the number of items in stock, and would talk to the Model and View to do what needed to be done.

In this use case, the MVC design works well for more than one reason. First of all, it makes it very clear what each part of the program wants. This makes it easier to keep and grow each part on its own, which could cut down on

mistakes and save time. It also makes development more customizable and flexible because each part can be improved or changed separately from the others.

Second, the MVC design easily splits server-side code from client-side code, which makes it a good fit for web apps. Most of the time, HTML, CSS, and JavaScript are used to build the View on the client side, while the Model and Controller are built on the server side. Because client-side code can handle interactions and changes without having to restart the whole page, user experiences are faster and more efficient.

Architectural Model

The architectural model for the project is given below.

Model

- Publication data model contains data related to books, including book ID, title, author, genre, price, and quantity.
- Order data model contains data related to orders, including transaction ID, date, book ID, quantity, and total cost.
- User data model contains data related to users, including user ID, name, email, and password.

View

- Publications view: displays the list of books in the inventory, with options to add, edit, or remove books.

- Orders view: displays the list of orders, with options to filter by date or book ID.
- User view: displays the login and registration forms for users.

Controller

- Publications controller: handles user requests related to the inventory, such as adding, editing, or removing books from the inventory.
- Orders controller: handles user requests related to orders, such as creating a new transaction or filtering transactions by date or book ID.
- User controller: handles user requests related to authentication and registration, such as validating login credentials or creating a new user account.

Technologies Used

The Transaction Management System is a full-stack web application, built using a combination of different technologies to create a cohesive system. A Java Spring Boot server, which offers the back-end functionality needed for the system to work, is at the center of the application.

For creating web applications in Java, Spring Boot is a strong and popular framework. It offers a variety of features and tools, including as built-in support for RESTful web services and database access, for developing scalable, reliable, and secure applications. We were able to swiftly construct a back-end system

that is capable of handling the intricate business logic needed by the Transaction Management System by using Spring Boot.

We used ReactJS, a very popular JavaScript framework for creating user interfaces, on the front end. ReactJS is renowned for its adaptability, modularity, and simplicity of usage, and it enables us to quickly and effectively construct responsive and understandable user interfaces. Since the Transaction Management System is a single-page application (SPA) thanks to the use of ReactJS, the user experience is slicker and more streamlined as the user is not required to reload the full page while browsing between different portions of the application.

The Transaction Management System also uses a MySQL database to manage and store the transaction data that is necessary for the system to function. The popular relational database management system MySQL is ideal for web applications. It is extremely scalable and adaptable and has strong data management features, including support for difficult queries and transactions.

The website which included frontend, backend and database are all hosted on Google Cloud Platform. The website can be accessed through a public URL which is hosted on the cloud.

Rationale for Architectural

The Model-View-Controller (MVC) architectural style is a proven approach for developing software systems with complex user interfaces. The MVC

architecture was chosen for the Transaction Management System project for publishing firms because it can clearly separate the concerns of display logic, application logic, and data administration. Whereas the View oversees display and user interface, the Model represents the system's data and business logic. The Controller serves as a middleman, taking user input, calling the proper Model activities, and changing the View as necessary. For complex projects like the Transaction Management System, such a separation of concerns produces a highly modular and manageable codebase.

Graphical User Interface

Login Page - Authentication

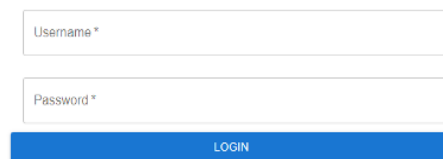
The login page serves as the entrance to the Transaction Management System, where users must input their login information to gain access. An authentication request is sent to the server when the user enters their login and password. A warning alerting the user that they supplied the wrong information is displayed if the credentials are found to be invalid.

The server creates a token containing the user's details, such as their role, and a session is created for the user at the backend if the credentials are legitimate. The token is then given back to the user and used for later requests to the server that require authentication.

As soon as the user's authentication is successful, they are taken to the dashboard interface, where they can access all the system's features and

capabilities. The Transaction Management System's login page is an important part overall since it ensures system security and gives users a seamless experience.

Transaction Management System for a publisher



The image shows a login form with two text input fields. The first field is labeled 'Username*' and the second is labeled 'Password*'. Below these fields is a blue rectangular button with the text 'LOGIN' in white capital letters.

Figure 2: Login Page

Dashboard

The Transaction Management System's dashboard serves as its main user interface and gives users access to all the application's modules. The user is directed to the dashboard after successfully logging in, where they can access various modules depending on their role. With a clear and straightforward layout, the dashboard is made to be intuitive and user-friendly.

Role-based access control, which guarantees that each user can only access the modules that are pertinent to their role, is one of the dashboard's primary features. For instance, an author can browse their published works in the All-Publications module, while an editor can manage and amend the publications in the same module. Similar to this, an admin can access the All-Orders module to manage and track orders while a distributor may access it to submit orders.

Users can use a variety of analytical tools through the dashboard to better understand their operations and make data-driven decisions. Our application's analytics page offers a dashboard view of different system utilization information. The number of publications that are now in the system, the number of users who have registered, and the number of orders that distributors have placed are all displayed on this page. By running a query against the appropriate tables in the MySQL database, the information for these metrics can be found.

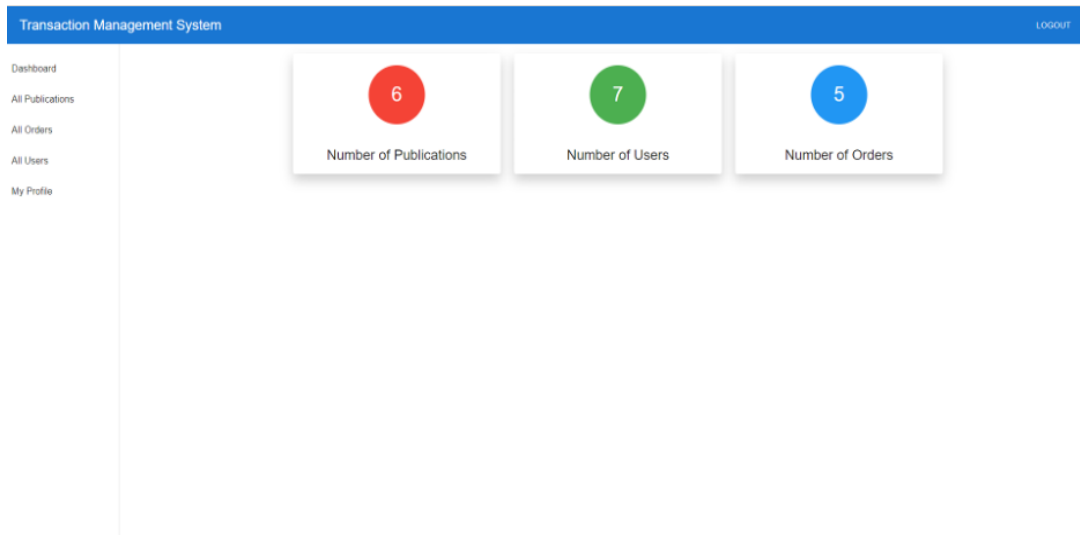


Figure 3. Dashboard

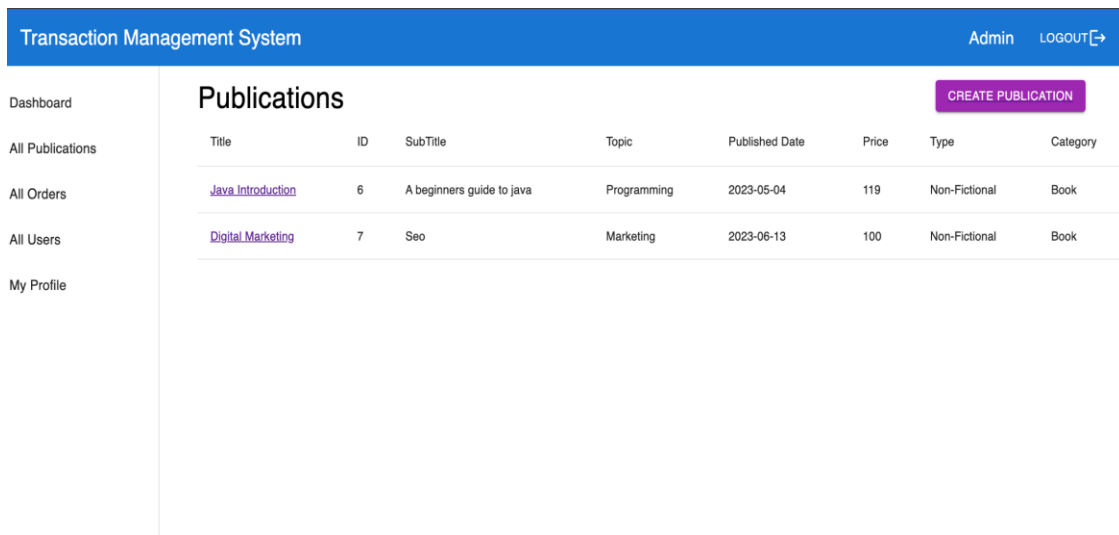
Publications

The Transaction Management System's publications module offers authors a platform to design and control their own publications. By entering the title and type of publication, the module enables authors to establish a new publication. A new publication may be produced, and the author may then begin to add sections to it. Each section has a title and a description, both of which are editable by the author. The author must also provide crucial details about the publication while producing it, such as the genre it belongs to, the category to which it belongs, and the price that the author desires to offer for the book. Also, the author can tag the article with specific keywords.

In the publication's module, I've incorporated form validations at several points. For instance, the author must choose the type of publication and offer a

title when creating a new publication. The author cannot submit the form unless all mandatory fields have been filled in.

I introduced the feature of changing sections of a publication. Even after the publication has been released, the author is always free to amend any section's content. As a result, authors can improve their writing and make sure their materials are always current.



The screenshot shows a web application interface for a Transaction Management System. The top navigation bar is blue and contains the text "Transaction Management System" on the left, "Admin" in the center, and "LOGOUT" with an external link icon on the right. A sidebar on the left lists navigation options: "Dashboard", "All Publications", "All Orders", "All Users", and "My Profile". The main content area is titled "Publications" and features a purple "CREATE PUBLICATION" button in the top right corner. Below the title is a table with the following data:

Title	ID	SubTitle	Topic	Published Date	Price	Type	Category
Java Introduction	6	A beginners guide to java	Programming	2023-05-04	119	Non-Fictional	Book
Digital Marketing	7	Seo	Marketing	2023-06-13	100	Non-Fictional	Book

Figure 4. List of Publications

The user can click on the "Create Publication" button in order to create a new publication. The user is prompted to enter information about the publication such as title, sub-title, category, publication type etc.

The screenshot shows a web application interface for creating a publication. At the top, there is a blue header with the text 'Transaction Management System' on the left and 'Admin LOGOUT' on the right. A vertical sidebar on the left contains navigation links: 'Dashboard', 'All Publications', 'All Orders', 'All Users', and 'My Profile'. The main content area is titled 'Create Publication' and contains the following form fields: 'Title' with the value 'Web Design', 'Sub Title' with 'Software Engineering', 'Category' (a dropdown menu showing 'Book'), 'Publication Type' (a dropdown menu showing 'Fictional'), 'Topic / Tags' with 'web design', and 'Price' with '250'. Below the form are two buttons: a blue 'CREATE' button and a purple 'BACK' button.

Figure 5. Create Publication Form

This screenshot shows the same 'Create Publication' form as in Figure 5, but after a successful submission. A green notification banner at the top of the form area displays a checkmark icon and the text 'Publication Created' with a close button (X) on the right. The form fields are now empty, and the 'CREATE' and 'BACK' buttons remain at the bottom.

Figure 6. Publication Created

The user can also view the details about the publications by clicking on the link provided in the list view. We are then taken to the page where we see more details about the publication such as the list of sections it has and with functionality to create / edit them.

Publication Sections

An author can create, update, and manage sections of a publication using our application's Publication Section module. Each section in a magazine can have a different section type, a different section title, and a different section's actual content.

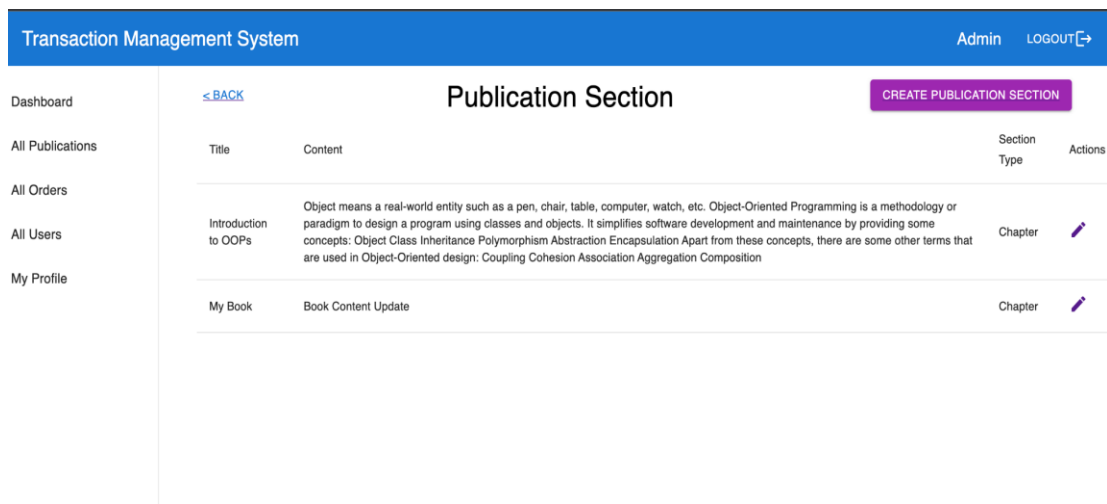


Figure 7. Publication Section

By specifying the section type, title, and content when creating a new publication, an author can also create sections within that publication. The parts can then be modified by the author as necessary, such as by changing the title or the content. The editor of the publication, in addition to the author, has access to control the parts. If necessary, the editor can modify already-existing parts, add new ones, or eliminate existing ones.

To guarantee that the section information provided by the author or editor is accurate and satisfies the necessary requirements, the Publishing Section module also includes a number of form validations. This entails making sure the section content isn't too lengthy and that the section title isn't empty.

The screenshot shows a web application interface for a Transaction Management System. At the top, there is a blue header bar with the text "Transaction Management System" on the left and "Admin LOGOUT" on the right. On the left side, there is a vertical navigation menu with the following items: "Dashboard", "All Publications", "All Orders", "All Users", and "My Profile". The main content area is titled "Create Publication Section". It contains three input fields: "Section Type*" with a dropdown menu showing "Chapter", "Title*" with the text "Introduction to OOPS", and "Content*" with a large text area containing a paragraph about Object-Oriented Programming. The paragraph text is: "Object means a real-world entity such as a pen, chair, table, computer, watch, etc. Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects. It simplifies software development and maintenance by providing some concepts: Object Class Inheritance Polymorphism Abstraction Encapsulation Apart from these concepts, there are some other terms that are used in Object-Oriented design: Coupling Cohesion Association Aggregation Composition". At the bottom of the form is a blue "SUBMIT" button.

Figure 8. Create Publication Section Form

Transaction Management System Admin [LOGOUT](#)

Dashboard
All Publications
All Orders
All Users
My Profile

Create Publication Section

✔ Publication section created successfully ✕

Section Type *

Title *

Content *

[SUBMIT](#)

Figure 9. Publication Section Created

Orders

Distributors can easily place orders for printing papers using the application's sales feature. Distributors must provide basic information to place an order.

Transaction Management System Admin [LOGOUT](#)

Dashboard
All Publications
All Orders
All Users
My Profile

Orders [CREATE ORDER](#)

ID	Date & Time	Publication Titles	Order Value	Ordered By
1	Thursday, May 4, 2023 at 12:15 PM	15 x Java Introduction	1785\$	Testing 2
2	Tuesday, June 13, 2023 at 1:35 PM	12 x Digital Marketing	1200\$	Roze Akter

Figure 10. List of Orders

The distributor must first enter the publication ID. This information comes from the publication table in the database and makes sure that the right release comes first. Next, the distributor has to put in how many copies they need. Using the prices in the release table and this information, the total cost of the sale is then calculated.

Distributors can use this feature to place new orders and manage the orders they already have.

The screenshot shows a web application interface for a Transaction Management System. The header is blue with the text 'Transaction Management System' on the left and 'Admin' and 'Logout' with an arrow on the right. A sidebar on the left contains links: 'Dashboard', 'All Publications', 'All Orders', 'All Users', and 'My Profile'. The main content area is titled 'Create Order'. At the top of this area is a green success message: 'Found publication, title: Java Introduction' with a checkmark icon and a close 'x' button. Below this are three input fields: 'Distributor ID*' containing '17', 'Publication ID*' containing '6', and 'Quantity*' containing '12'. Under the quantity field, it says 'Quantity: 10 - 100'. Below that, the total cost is displayed as 'TOTAL: 1428\$'. At the bottom center of the form is a large blue button labeled 'ORDER'.

Figure 11. Publication Id Verification

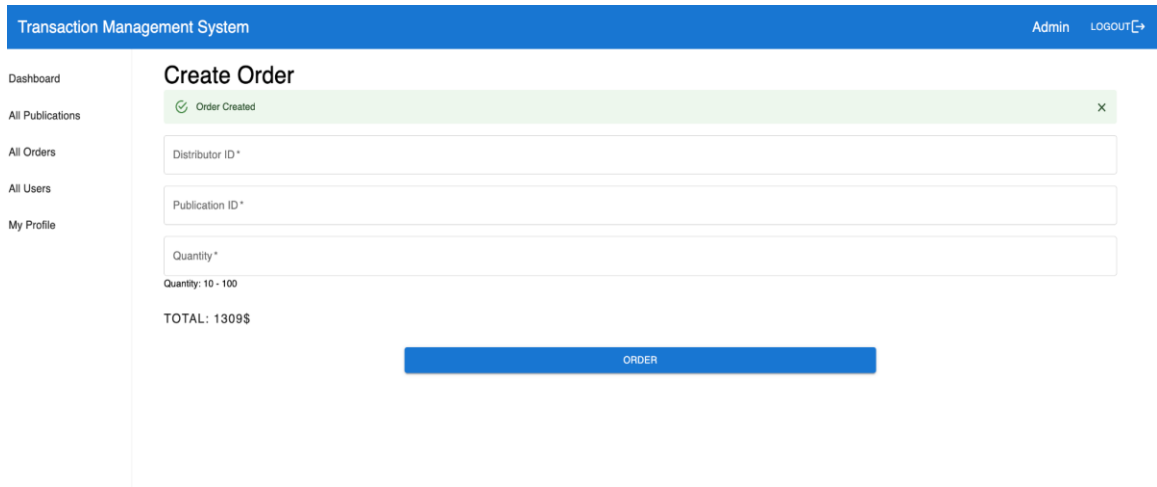


Figure 12. Order Created successfully

Users

The Users module is an important part of the system that can only be viewed by the administrator. It gives a full list of all the people who have signed up for the app, along with information about each one. When adding a new user, the admin needs to know the person's username, first and last names, password, and role.

Transaction Management System						Admin	LOGOUT
Dashboard All Publications All Orders All Users My Profile	Users						
							CREATE USER
	ID	Name	Username	Mobile	Role		
	15	Hassain Talwar	hassain	9193498336	Admin		
	16	Testing 1	test1	9347518503	Editor		
	17	Testing 2	test2	7729061297	Distributor		
	18	Srinivasa Tigarru	manoj	1234567890	Editor		
19	Talha Mohammed	talha	8686915558	Author			
20	Roze Akter	dipi	8121165397	Distributor			

Figure 13. List of Users

The users' part also lets you change or delete a user's information. For example, a supervisor can get rid of a user who is no longer logged in or change a user's job from writer to author. This feature gives you full control over who can use the system, making sure that only allowed users can get in.

Transaction Management System		Admin	LOGOUT
Dashboard All Publications All Orders All Users My Profile	Create User		
	Username		
	Mobile		
	First Name	Last Name	
	Password	Confirm Password	
	Role		
	<div style="text-align: center;"> CREATE BACK </div>		

Figure 14. Create User Form

My Profile

In the My Profile module, users can see and change their own personal information. All users of the system can use this feature. Users can get to this area by clicking the My Profile button.

The screenshot shows a web application interface for a 'Transaction Management System'. At the top, there is a blue header bar with the system name on the left and 'Admin' and 'LOGOUT' with an arrow icon on the right. On the left side, there is a vertical navigation menu with the following items: 'Dashboard', 'All Publications', 'All Orders', 'All Users', and 'My Profile'. The 'My Profile' item is highlighted. The main content area is titled 'My Profile' and contains four input fields arranged in a 2x2 grid. The top-left field is labeled 'First Name' and contains the text 'Hassain'. The top-right field is labeled 'Last Name' and contains the text 'Talwar'. The bottom-left field is labeled 'User Number' and contains the text '9193498336'. The bottom-right field is labeled 'Role' and contains the text 'Admin'.

Figure 15. Update Profile Form

Back End

The design of our system is based on a client-server setup. ReactJS is used for the client, and Java Spring Boot is used for the server. MySQL is the database.

The client and server parts of the program can talk to each other using RESTful API calls. The API calls are handled by the server, which is also in charge of getting data from the database, change it, or store it.

We used Spring Security to make JWT authorization so that clients and servers can talk safely. This makes sure that only authorized people can access

the application. The design of our system is based on best practices in the industry, with a focus on being flexible, easy to manage, and able to scale.

Entities

Entities are Java objects that represent the data that will be kept in a database. Usually, they are mapped to database tables and their corresponding rows. I have the following entities created in the system:

- Order
- OrderItem
- Publication
- PublicationCategory
- PublicationSection
- PublicationSectionType
- PublicationType

The below figure shows us the snapshot of an entity.

```
J Order.java ×
src > main > java > com > example > publishermanagement > entities > J Order.java > ...
25
26 @Data
27 @Entity
28 @Table(name = "orders")
29 public class Order {
30
31     @Id
32     @GeneratedValue(strategy = GenerationType.IDENTITY)
33     private Long id;
34
35     @ManyToOne
36     @JoinColumn(name = "receiver_id", nullable = false)
37     private User receiver;
38
39     @Column(nullable = false)
40     private Double amount = 0.0;
41
42     @Enumerated(EnumType.STRING)
43     private OrderStatus status=OrderStatus.PROCESSING;
44
45     @OneToMany(mappedBy = "order", cascade = CascadeType.ALL)
46     private List<OrderItem> items = new ArrayList<>();
47
48     @CreationTimestamp
49     private Date createdAt;
50
51     @UpdateTimestamp
52     private Date updatedAt;
53 }
54 |
```

Figure 16. Order Class

Controllers

Controllers are in charge of processing HTTP requests and sending HTTP responses back to the client. Controllers define methods that correspond to certain HTTP requests and are annotated with the `@Controller` or `@RestController` annotation. I have created the following controllers.

- OrderController
- PublicationCategoryController
- PublicationController
- PublicationSectionController
- PublicationSectionTypeController
- PublicationTypeController
- UserController
- UserRoleController

The following figure shows the snapshot of a controller:

```
J OrderController.java 5 x
src > main > java > com > example > publishermanagement > controllers > api > J OrderController.java > {} com.example.publish
29 @RestController
30 @RequestMapping("/orders")
31 @Slf4j
32
33 public class OrderController {
34
35     @Autowired
36     private OrderService orderService;
37
38     @GetMapping()
39     public ResponseEntity<List<OrderDTO>> fetch() {
40         log.info(msg:"In get orders");
41         List<OrderDTO> orders = orderService.fetch();
42         log.info(format:"Get order", orders);
43         return new ResponseEntity<List<OrderDTO>>(orders, HttpStatus.OK);
44     }
45
46     @GetMapping("/{id}")
47     public ResponseEntity<OrderDTO> getById(@PathVariable Long id) {
48         OrderDTO orderDTO = orderService.findById(id);
49         return new ResponseEntity<OrderDTO>(orderDTO, HttpStatus.OK);
50     }
51
52     @PostMapping()
53     public ResponseEntity<SuccessResponse> create(@Valid @RequestBody OrderRequest orderRequest) {
54         orderService.create(orderRequest);
55         SuccessResponse response = new SuccessResponse();
56         response.setMessage(Constants.ORDER_CREATED_SUCCESSFULLY);
57         return new ResponseEntity<SuccessResponse>(response, HttpStatus.CREATED);
58     }
59 }
60
```

Figure 17. Order Controller Class

I have implemented an authentication service using the Authentication Controller through which I have created the login API. After the user signs in, I create the tokens using the JWT package.

Services

Services are classes that house the application's business logic. They offer a layer of abstraction between the data access layer and the controllers,

which manage incoming requests (which interacts with the database or other external systems). The following are the services present in the system

- OrderService
- PublicationCategoryService
- PublicationSectionService
- PublicationSectionTypeService
- PublicationService
- PublicationTypeService
- UserRoleService
- UserService

The following figure shows the snapshot of a Service.

```
J OrderService.java 2 X
src > main > java > com > example > publishermanagement > services > J OrderService.java > ...
29 @Service
30 @Slf4j
31 public class OrderService {
32
33     @Autowired
34     private UserRepository userRepository;
35
36     @Autowired
37     private PublicationRepository publicationRepository;
38
39
40     @Autowired
41     private OrderRepository orderRepository;
42
43 > public List<OrderDTO> fetch() {-
50
51 > public OrderDTO findById(Long id) {-
63
64
65 > public void create(OrderRequest orderRequest) {-
85
86     private List<OrderItem> getOrderItems(List<OrderItemRequest> itemsRequest, Order order) {
87         List<OrderItem> items = new ArrayList<>();
88         for (OrderItemRequest itemRequest : itemsRequest) {
89
90             OrderItem item=new OrderItem();
91             Optional<Publication> publication = publicationRepository.findById(itemRequest.getPublicationId());
92
93             if (!publication.isPresent()) {
94                 throw new EntityNotFoundException(
95                     Constants.PUBLICATION_NOT_FOUND + " with ID " + itemRequest.getPublicationId());
96             }
97
98             item.setPublication(publication.get());
```

Figure 18. Order Service Class

Class Diagram

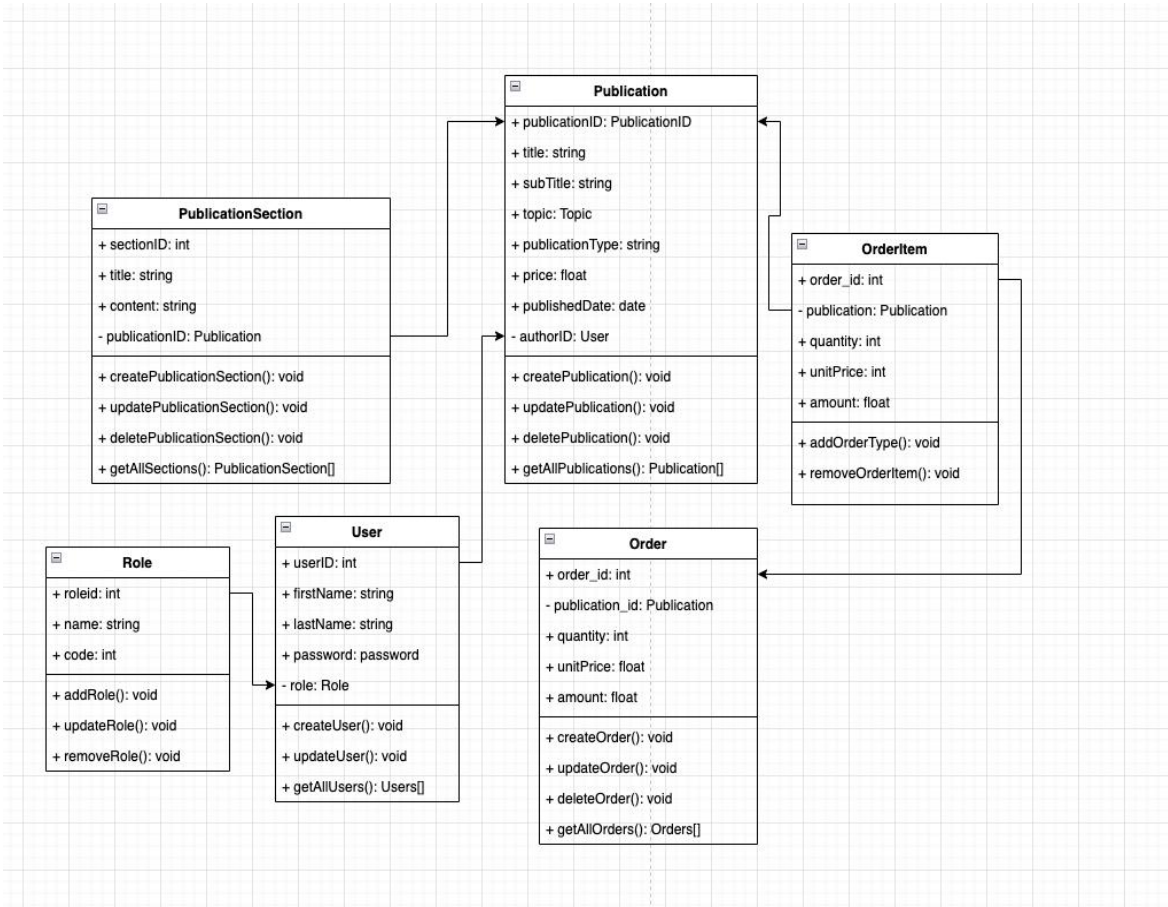


Figure 19. Class Diagram

Sequence Diagram

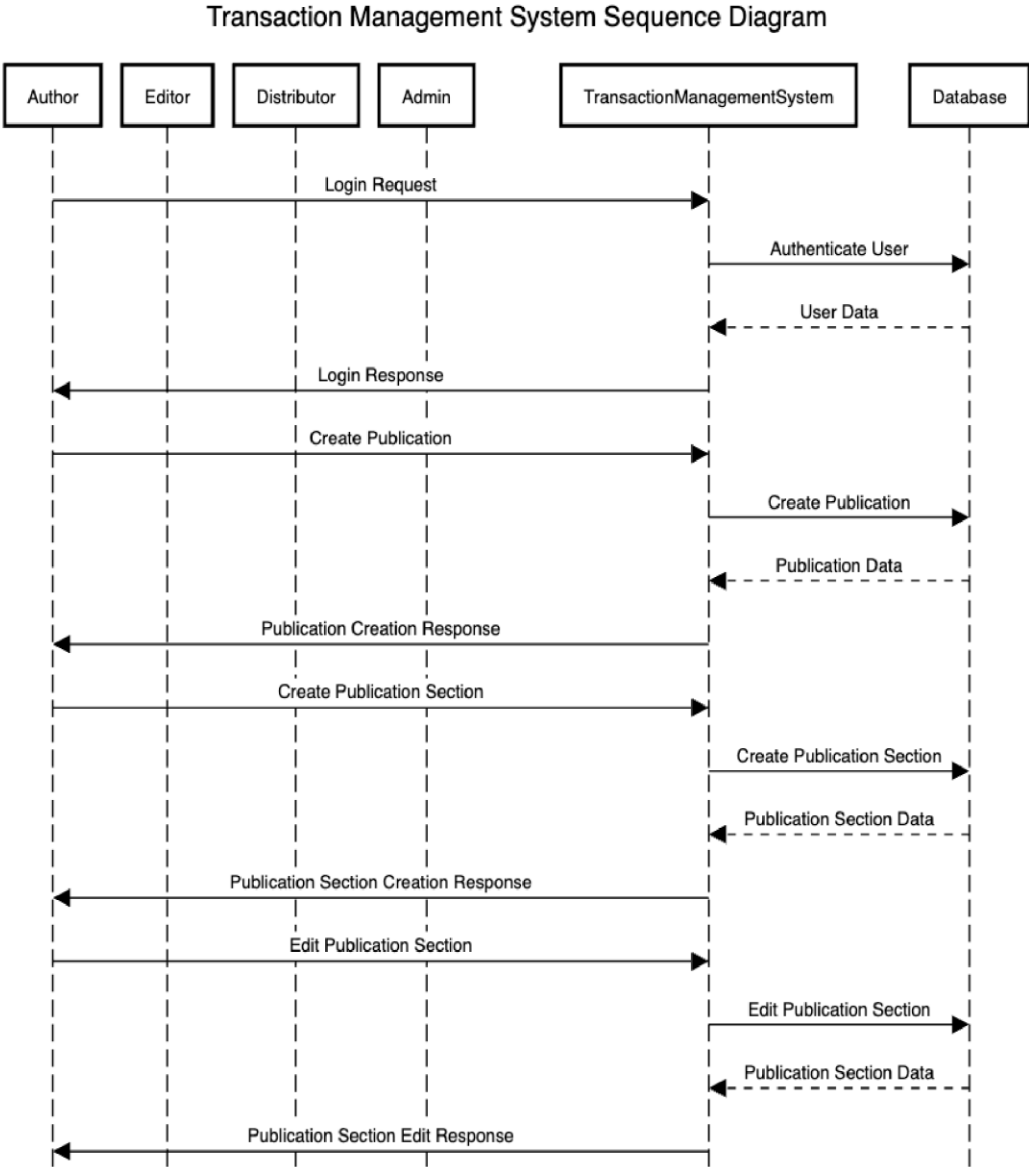


Figure 20. Sequence Diagram Part One



Figure 20. Sequence Diagram Part Two

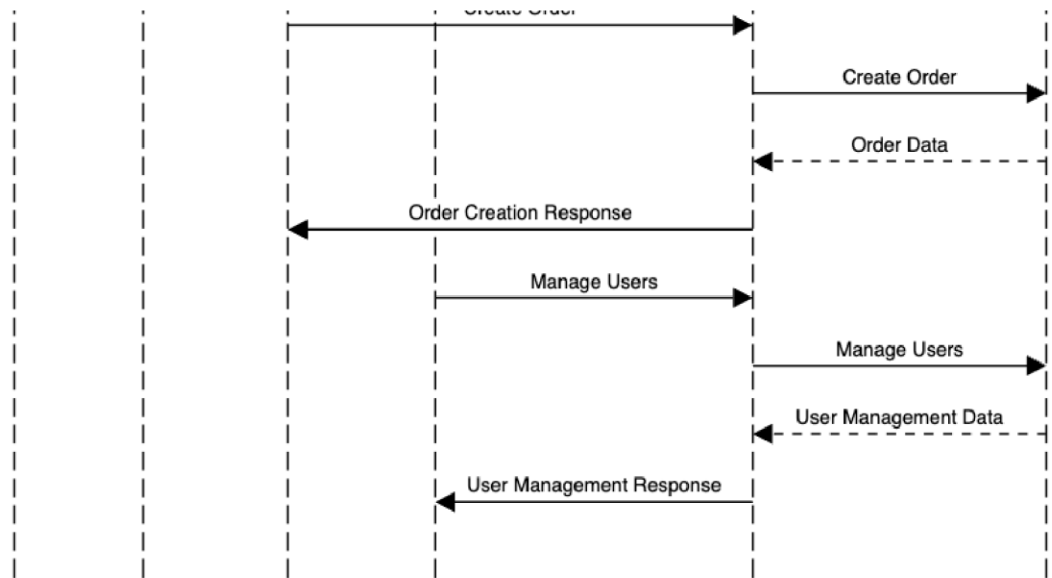


Figure 20. Sequence Diagram Part Three

The workflow for creating publications and processing orders is depicted in the sequence diagram along with the interactions that take place between the various users and the Transaction Management System. The system authenticates the user's credentials once the user logs in at the top of the figure. After then, the Editor can alter publications and their parts while the Author is free to create them. The Distributor has access to every publication and may place orders for any of them. The admin has the ability to generate orders as well as manage users and user roles. The sequence diagram primarily focuses on the many activities that users can do within the system and the accompanying interactions with system components to complete these tasks.

CHAPTER FIVE

TEST PLAN

Requirements/specifications-based system-level test cases

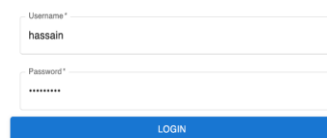
Some of the Requirements/specifications-based system-level test cases

are:

User Login Functionality

Test Case: Enter the correct credentials and check that the user is successfully logged in.

Transaction Management System for a publisher



Username*
hassain

Password*

LOGIN

Figure 21. Enter Correct Login Credentials

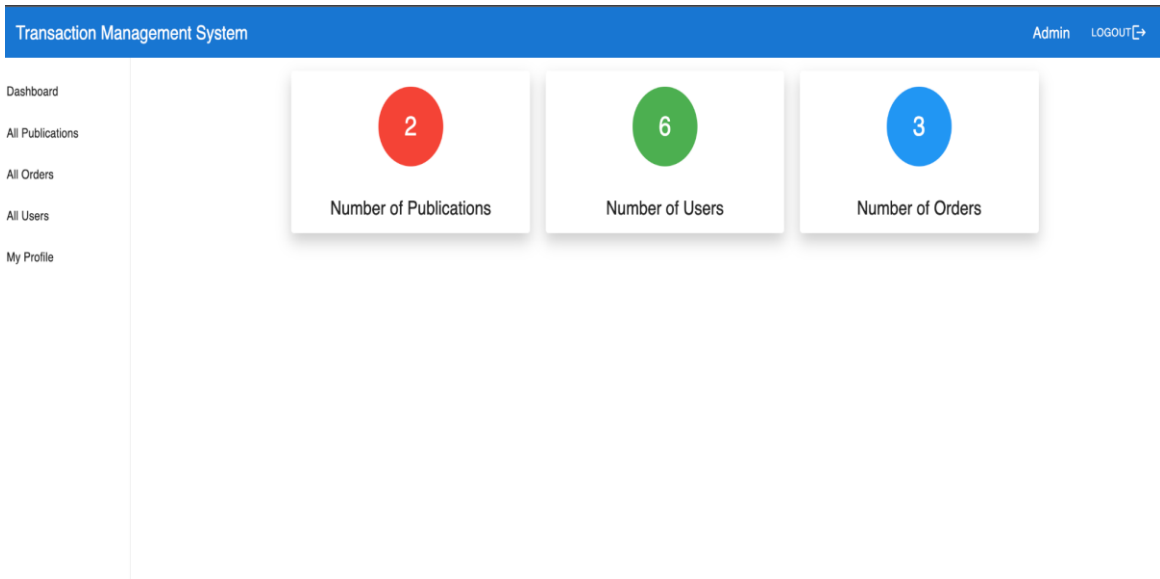


Figure 22: Test User Able to Login

Test Case: Enter incorrect credentials and check that the user is not able to log in.

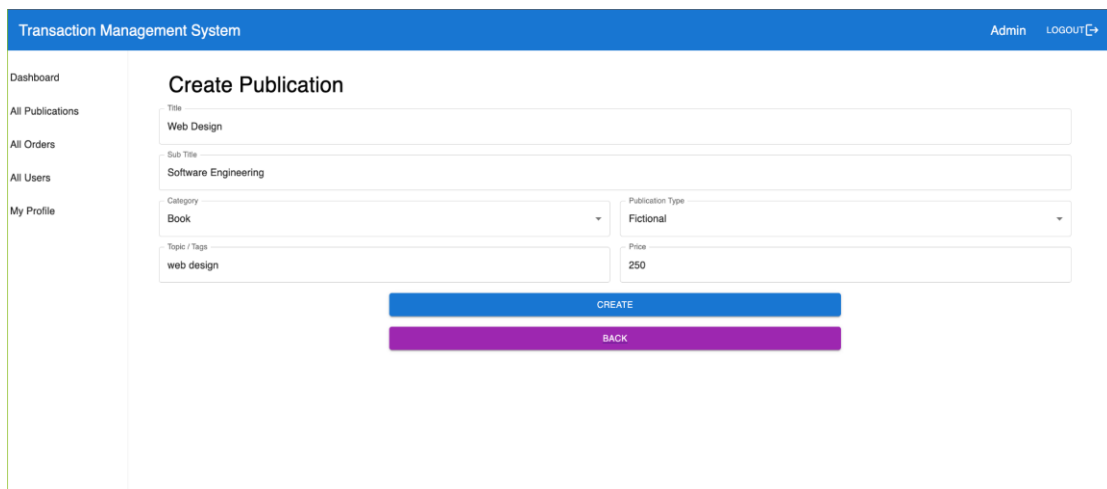
Transaction Management System for a publisher

The screenshot shows a login form for a publisher. It consists of two input fields: 'Username*' with the value 'hassain' and 'Password*' with masked characters. Below the password field, there is a red error message that reads 'Invalid Login Credentials'. At the bottom of the form is a blue button labeled 'LOGIN'.

Figure 23. Enter Invalid Credentials

Publication Creation Functionality:

Test Case: Create a publication with all required fields and verify that it is successfully created.

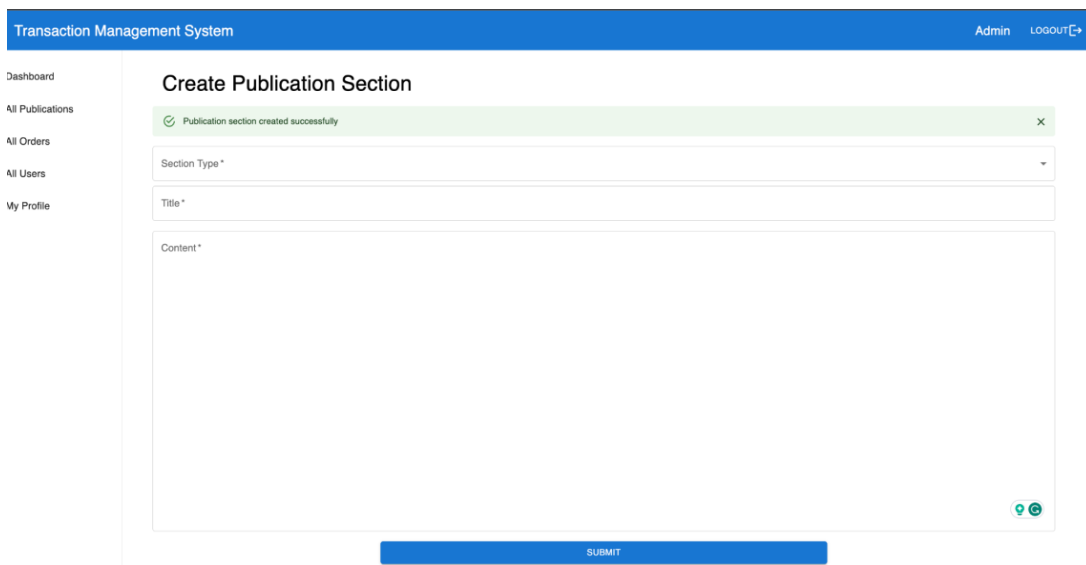


The screenshot shows the 'Create Publication' form in the Transaction Management System. The form includes the following fields and values:

- Title: Web Design
- Sub Title: Software Engineering
- Category: Book
- Publication Type: Fictional
- Topic / Tags: web design
- Price: 250

At the bottom of the form, there are two buttons: a blue 'CREATE' button and a purple 'BACK' button.

Figure 24. Create Publication with Required Fields



The screenshot shows the 'Create Publication Section' form in the Transaction Management System. A green success message is displayed at the top: 'Publication section created successfully'. The form includes the following fields:

- Section Type *
- Title *
- Content *

At the bottom of the form, there is a blue 'SUBMIT' button.

Figure 25. Created Publication Section Successfully

Test Case: Create a publication with missing required fields and verify that an error message is displayed.

The screenshot shows the 'Create Publication' form in the Transaction Management System. The form is titled 'Create Publication' and has a red error message at the top: 'All fields are required.' The form contains the following fields:

- Title: Web Design
- Sub Title: Software Engineering
- Category: Book
- Publication Type: Fictional
- Topic / Tags: web design
- Price: (empty)

At the bottom of the form, there are two buttons: a blue 'CREATE' button and a purple 'BACK' button.

Figure 26. Publication Section Form Validations

Publication Section Creation Functionality

Test Case: Create a publication section with all required fields and verify that it is successfully created.

Transaction Management System Admin [LOGOUT](#)

Dashboard
All Publications
All Orders
All Users
My Profile

Create Publication Section

Section Type *
Chapter

Title *
Classes

Content *
Classes are important part of Java Object Oriented Programming.

[SUBMIT](#)

Figure 27. Create Publication Section Form

Transaction Management System Admin [LOGOUT](#)

Dashboard
All Publications
All Orders
All Users
My Profile

Create Publication Section

✔ Publication section created successfully

Section Type *
Title *
Content *

[SUBMIT](#)

Figure 28. Publication Section Created Successfully

Test Case: Create a publication section with missing required fields and verify that an error message is displayed.

The screenshot shows a web application interface for a Transaction Management System. The main heading is "Create Publication Section". The form contains the following elements:

- Section Type***: A dropdown menu with "Chapter" selected.
- Title***: An empty text input field.
- Classes**: A text input field with "Classes" entered.
- Content***: A large empty text area.

At the bottom of the form, a blue button labeled "SUBMIT" is visible. A validation error message, "Please fill out this field.", is displayed in a small box above the button, pointing to the empty Content field.

Figure 29. Publication Section Form Validation

Publication Editing Functionality

Test Case: Edit a publication and verify that the changes are successfully saved.

Title	ID	Author	Topic	Published Date	Price	Type	Category
Java OOPs Concepts	1	Ramesh F. Java	programming	2021-04-10	200	Non-Fictional	Book

Figure 30. Publications List

Test Case: Attempt to edit a publication without appropriate permissions and verify that an error message is displayed.

Publication Section Editing Functionality

Test Case: Edit a publication section and verify that the changes are successfully saved.

Figure 31. Update Publication Section

Test Case: Attempt to edit a publication section without appropriate permissions and verify that an error message is displayed.

View All Publication's Functionality

Test Case: Verify that all publications are displayed when the "View all publications" button is clicked.

Order Creation Functionality

Test Case: Create an order with all required fields and verify that it is successfully created.



The screenshot shows the 'Create Order' form in the Transaction Management System. The form has a blue header bar with the system name and a user profile icon. On the left, there is a navigation menu with links to 'Dashboard', 'All Publications', 'All Orders', 'All Users', and 'My Profile'. The main content area is titled 'Create Order' and contains three input fields: 'Publisher ID' with the value '2', 'Publication ID' with the value '1', and 'Quantity' with the value '20'. Below these fields, the total amount is shown as 'TOTAL: 40005'. At the bottom of the form, there is a prominent blue button labeled 'ORDER'.

Figure 32. Create Order Form



The screenshot shows the 'Create Order' form after a successful order creation. The form has a blue header bar with the system name and a user profile icon. On the left, there is a navigation menu with links to 'Dashboard', 'All Publications', 'All Orders', 'All Users', and 'My Profile'. The main content area is titled 'Create Order' and features a green notification banner at the top that says 'Order Created'. Below the notification, there are three empty input fields: 'Distributor ID', 'Publication ID', and 'Quantity'. The total amount is now 'TOTAL: 40008'. At the bottom of the form, there is a prominent blue button labeled 'ORDER'.

Figure 33. Order Created successfully

ID	Date & Time	Publication Title	Order Value	Ordered By
1	Thursday, April 10, 2020 at 11:30 PM	2019 Java OCP® Concepts	4999	Suresh Thakur

Figure 34. List of Orders

Test Case: Create an order with missing required fields and verify that an error message is displayed.

User Role Management Functionality

Test Case: Add a new user and verify that it is successfully added.

Figure 35. User Creation with Proper Data

ID	Name	User Number	Role
1	Admin	1744200	Admin
2	Customer	123456789	Customer

Figure 36. List of Users

Techniques Used for Test Generation

Boundary Value Analysis (BVA):

This method entails putting the system to the test with values that are just within and outside of the input range, as well as values that are at the top and lower limits of the range. For instance, examining how the system responds when a publication with the most parts possible is created.

Transaction Management System LOGOUT

Dashboard
My Publications
My Profile

Create Publication

Title: Masters Of Computer Science International Students Struggles To Success In Life In California Sate University San Bernardino Campus After Masters Degree

Sub Title: Struggle is a sign that you are trying, that you have set on a path to a goal. The second positive thing about struggle is that it teaches us a lot in our real life

Category: Book | Publication Type: Lifestyle

Topic / Tags: A good life can give you happiness and positive thinking to live a good vibe | Price: 200

CREATE
BACK

Figure 37. Create Publication with Large Input

Equivalence Partitioning (EP):

By categorizing the input domain into equivalence classes, this approach entails choosing representative values for testing from each class. As an illustration, evaluate the system's behavior while producing a publication with various section kinds.



Figure 38. Create Publication Section with Various Types

State Transition Testing:

This method entails observing how the system behaves when it switches between various states. For instance, evaluating how the system responds when an editor modifies a piece of a publication section that has already been created.

Transaction Management System LOGOUT

Dashboard
My Publications
My Profile

Edit Publication Section

Section Type *
Chapter

Title *
Data Base Management System

Content *
A database management system (or DBMS) is essentially nothing more than a computerized data-keeping system. Users of the system are given facilities to perform several kinds of operations on such a system for either manipulation of the data in the database or the management of the database structure itself.

Figure 39. Editing Publication that is Already Created

Use Case Testing:

With this method, the system's behavior is evaluated against predetermined use cases or scenarios. As an illustration, evaluate how the system responds when a distributor makes an order for a book whose price is 0.

Transaction Management System LOGOUT

Dashboard
All Publications
All Orders
My Profile

Create Order

Found publication, title: System Design In Computer Science X

Distributor ID *
2

Publication ID *
4

Quantity *
5

TOTAL: 0\$

[ORDER](#)

Figure 40. Creating Order with Zero Price

REFERENCES

- Oracle, Java Documentation. <https://docs.oracle.com/en/java/>
- Entity-Relationship model, Wikipedia.
https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model
- Introduction to publishing title management software, Consonance.
<https://www.consonance.app/blog/introduction-to-publishing-title-management-software/>
- Management System, Wikipedia.
https://en.wikipedia.org/wiki/Management_system
- "FitSM Part 0: Overview and vocabulary". Itemo. 2015-04-01. Archived from the original on 2015-08-31. Retrieved 2015-07-24
- "ISO 9000:2015 Quality management systems — Fundamentals and vocabulary". iso.org. International Organization for Standardization. 2020-10-12. Retrieved 2020-10-12
- Gosling, James; Joy, Bill; Steele, Guy; Bracha, Gilad. "The Java Language Specification, 2nd Edition". Archived from the original on August 5, 2011. Retrieved February 8, 2008
- "What is Java Swing? - Definition from Techopedia". Techopedia Inc. Retrieved 2018-11-03