KATHMANDU UNIVERSITY

Department of Computer Science and Engineering
Dhulikhel,Kavre

A Project Report
on
"Distributed Library"

[Code No. : COMP 207]
(For partial fulfillment of 2nd Year/ 4th Semester in Computer Science)

Submitted by
Rabin Bhandari(Roll No. 6)
Susil Raj Neupane(Roll No. 39)
Aayush Pokharel(Roll No. 43)
Bijen Rayamajhi(Roll No. 48)

Submitted to
Project Cordinator
Pranita Karki
Dept of Computer Science and Engineering

Submission Date: 6th May, 2022

Bonafide Certificate

This project work on "Distributed Library" is the bonafide work of

"

Aayush Pokharel(Roll No. 43)
Susil Raj Neupane(Roll No. 39)
Rabin Bhandari(Roll No. 6)
Bijen Rayamajhi(Roll No. 48)

"

who carried out the project work under my supervision.

Project Supervisior

_____

Name: Amrit Dahal

Academic Designation: Lecturer

Dept. of Computer Science and Engineering

1 Acknowledgement

We wish to express our sincere thanks to the Department of Computer Science and Engineering for including the COMP 207 project into our curriculum. We would like to express our heartfelt gratitude to our project coordinator Ma'am Pranita Karki and our supervisor Mr. Amrit Dahal for their regular guidance and encouragement throughout the project. Taking this opportunity, we would like to thank all those individuals who directly or indirectly helped us in making this project successful, one be it by encouraging us throughout the project or else through their valuable suggestions which we have tried our best to assimilate within our work.

2 Abstract

The project 'Decentralized Library' proposal is drafted to meet the prerequisites to partially fulfill the COMP 207 course offered by the Department of Computer Science and Engineering at Kathmandu University. This project is designed to overcome the lack of large public Libraries in Nepal and the economic burden it puts on students as well as other people who have to buy too many books due to lack of proper book lending facilities. The proposed App will act as a search engine platform which will catalog books near the users and facilitate exchange of books between its users. It will include displaying a curated selection of books available for exchange as per user-selected genres along with a list of books the users are searching for. There would be a chat feature to facilitate the exchange along with an embed map and GPS for ease of setting exchange spots.

We, the involved project members, have decided to create a WebApp that uses REST API framework in accomplishing this project by using Python and TypeScript as the main programming language. We will be using the Django(Python) framework for backend and Angular(TypeScript) framework for frontend.

Keywords: TypeScript, WebApp, frontend, backend, GPS

3 List of Symbols

4 Abbreviation

- **CSS : Cascading Style Sheets**
- **HTML : Hyperlink Text Markup Language**
- **SQL : Structured Query Language**
- **CLI : Command Line interface**
- **UI : User Interface**
- **SCSS : Syntactically Awesome Styles Sheets**
- **API : Application Programming Interface**
- **GUI : Grpahical User Interface**
- **JWT : Java Web Token**
- **ER : Entity-Relation**
- **BLOB : Binary Large Object**
- **vw : Wiew Width**
- **REST : Representational State Transfer**

Contents

List of Figures

5 CHAPTER 1: INTRODUCTION

5.1 Background

Even in today's time the majority of people still prefer to read paper books. For this they buy books, read them and afterwards it's left gathering dust in the shelves. Buying books is a costly endeavor when a devoted reader goes through a double digit number of books in a span of a few months. This leads to both an economic burden as well as a space problem.
Borrowing from the Library could easily solve this problem but sadly Libraries are an infrastructure that is in poor quality in the context of Nepal. Mostly all the Libraries are either a personal one or an institutional one. With institutional Library being mostly of Schools and University where course books are the focus and personal Libraries are personal collections of an individual. It is very hard to borrow books and people need to either buy books or read online pirated transcripts.
Therefore our proposed app comes into play as a platform to lend and borrow books. With a peer to peer exchange facility, the physical need to house books is circumvented and an active local community of book readers could potentially arise.

5.2 Objectives

The main objective of this project is to make a virtual decentralized library where people can exchange books with each other and actually deploy it on the internet.

Besides, the other objectives are listed below:

- To circumvent the physical need to store books in a central location.
- To have a robust website with proper validators.
- To provide a book cataloguing service.
- To lighten the economic burden of buying books as well as provide an alternative to book hoarding.
- To promote a book reading culture.

5.3 Motivation and Significance

Our project is particularly inspired by the lack of fictional books in Kathmandu University Central Campus Library. Furthermore this is a situation that is prevalent in other educational institutions as well. As students the economic burden of buying new books every time is a heavy burden and the lack of any platform to properly share books is a situation that we the team members have personally experienced.

Furthermore the site Goodreads; a books cataloging website; is a primary point of inspiration in tackling this project. A platform that works on a local level to provide an exchange catalog of locally available books is the main draw for the team to work in this project.

6 CHAPTER 2: RELATED WORKS

There are similar sites that provide book cataloguing,lending or buying service. Some of such sites that are in use are:

Some of such sites that are in use are:

6.1 GoodReads

Figure 1: GoodReads

One of the primary experiences for this project, Goodreads is an American website which provides book cataloging features and is one of the most widely used services among book readers to keep track of books that they have read, are looking to read and new upcoming books.

6.2 Scribd

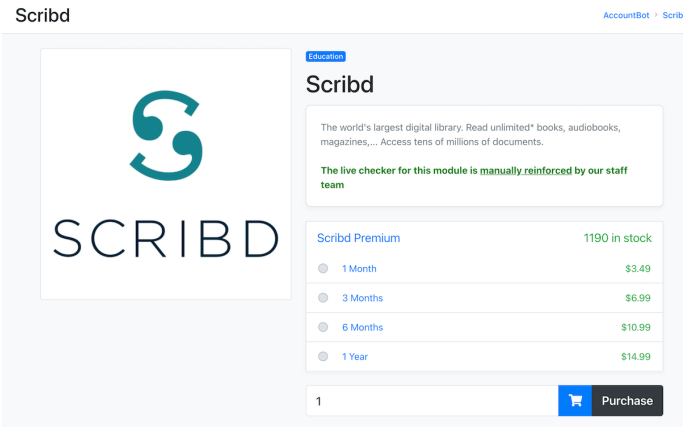

Figure 2: Scribd

An e-book and audio book subscription service, Scribd has been referred to as the ”Netflix of Books”. For a small subscription charge, Scribd grants access to a large repository of e-books and audio books.

7 CHAPTER 3: DESIGN AND IMPLEMENTATION

7.1 Architectural Design

7.1.1 Flowchart

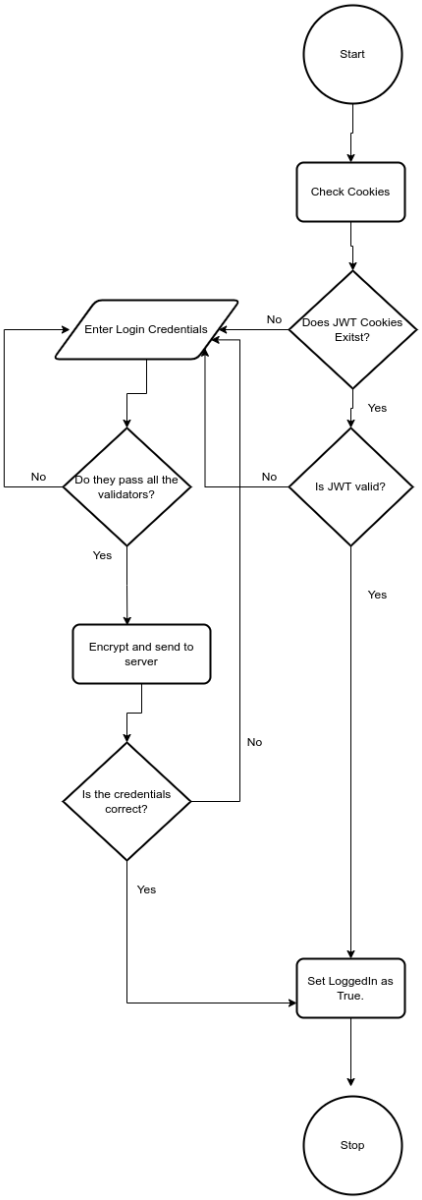

Figure 3: Log In Flowchart

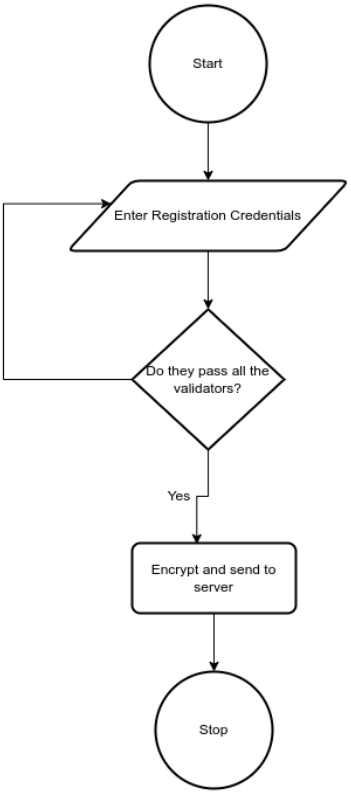

Figure 4: Register Flowchart

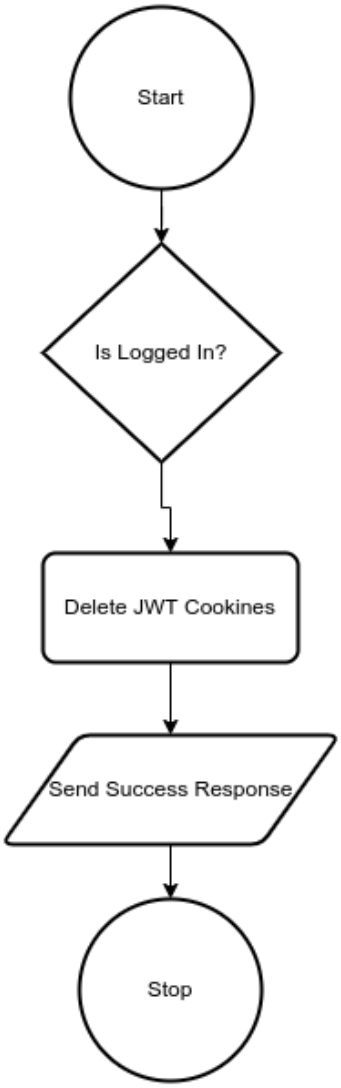

Figure 5: Logout Flowchart

7.2 System Requirement Specification

7.2.1 Software Requirements:

- **Front End Tools**
  - The UI is created using TypeScript,CSS and HTML.
  - Development Tools : Angular framework
- **Back End Tools**
  - The backend is created using Python.
  - SQL or No-SQL database will be used for storing data.
  - Development Tools : MySQL, Django

7.2.2 Hardware Requirements:

- Compatibility: (Development) Compatible with all PCs able to run python and Angular-CLI.
- Compatibility: (Deployment) Compatible with all servers capable of python hosting.
- Compatibility: (Usage) Compatible with all PCs able to run a modern Internet Browser.

Python : A dynamically typed multi paradigm general purpose programming language.

SQL : Structured Query Language, an standardized programming used to maintain Relational databases as well as perform operations on the data in them.

Django Framework : An open source web framework written in python that follows model-templates- view architecture.

Angular CLI : The Command line interface of popular frontend web development framework Angular which is maintained by Google.

8 CHAPTER 4: DISCUSSION AND ACHIEVEMENT

8.1 Methodology

The whole process of developing the software has been divided into following aspects:

- Research and study
- UI Design and prototyping
- Core programming
- Program Testing
- Documentation

8.1.1 Research and Study

This project being the first of the kind we have ever done, we decide to choose a subject matter that we clearly know and have experience with i.e. Database focused application .

We have interacted with the various software/sites mentioned above in chapter 2 on a semi-regular basis and have experienced its strength and pitfall. We are trying to provide a grass root service that we hope will take wide spread adoption.

8.1.2 UI design and Prototyping

The initial designs are sketched on A4 papers by our members and shared on group chat for peer review and then we use figma to design test the look of our UI before adding any frontend code to make it in CSS or SCSS.

8.1.3 Core Programming

The most amount of project time was spent in this stage. We are using the Python language to write our backend program. The Angular framework provides the necessary tools to create the GUI. Django (WhiteNoise plugin) is being used to serve the static web pages to the web and Django Rest API provides get, put and update APIs to manipulate data for LogIn/Register (Using JWT) or to update the various other data models.

8.1.3.1 Issues Encountered

- ***Team Members catching COVID-19***
  After the initial design phase of the the project was complete. Half the members of the development caught COVID-19 and work was halted. This went on for some time and the issue was compounded by other academic responsibilities due to which there was periods of inactivity during development phase.
- ***Adapting the frontend code with framework***
  Due to part of the team members only having learned vanilla HTML and CSS, it was confusing to adapt the code written by multiple parties to comply with single page application code. one prominent example was of using anchor tag for navigation with href and Routerlink.

- ***Routing issue***
  Due to Django and Angular both providing web page routing,initially it was confusing to separate the frontend routing and backend routing. The web pages' view routing were done in angular along with error handling whereas API endpoint routing was done in Django.

- ***Managing Django Migrations***
  We initially made some IR models and went to code the application as per the design schema but managing multi-valued attributes and foreign key constraints was difficult and there was a lot of deviation to the initial design. Furthermore, due to these deviations, the data needed to be entered multiple times in the database and we had to rollback the migrations and redo them. This causes issues like accidental deletion of wrong migrations and a deadlock in generation of new migrations. This had us repeat this procedure.

- ***Image Storage***
  Initially we had thought to store images as BLOB objects after they were standardized in resolution through manipulation via the Python Image Library in the database but the Django framework calls for the files to be stored in the directory and it was difficult to store them. Idea to introduce MongoDB for image file storage was introduced but due to time and code complexity of working multiple databases, it was passed over and images were simply stored in a directory.

- ***Geo-location Features***
  Initially our application was supposed to have Geo-location ability via Open Streets API but due to our limited understanding of converting the Geo=location coordinates to get a radius and the query getting more and more complex with each API. We decided to forgo the Geo-location.

- ***Brainstorming features to be added***
  Initially the project was on different trajectory and due to amount of time invested, team members were locked in a certain direction. The team was a newly formed team and it took some time for internal communication between members to be smooth.

- ***Database Loading and Exporting***
  MySQL database wasn't being loaded via a CSV file and was saying that local files from only certain directory could be loaded or exported to. The problem was that the default specified directory was within root directory and superuser privilege was necessary to even access the directory further more we could change the directory as it required various system setting for MySQL service to be edited as per stack over flow which we didn't understand.

- ***Difficulty in getting domain from mercantile host master***
  Despite filling forms properly multiple times to get a domain, mercantile host master always denied the request claiming insufficient documents. Thus domain and hosting had to be bought from Tech Himalaya.

8.1.4 Program Testing

Unit testing was done as soon as we completed the code of a single widget to check its functioning properly. This was repeated multiple times during the development period to create a robust system and once the core programming is completed, alpha testing was done to find different types of bugs or ill optimised issues.

8.1.4.1 Bugs found

- The CSS absolute units (vw and vw) were making the layout of the have horizontal scroller even when it wasn't necessary.
- Chat API were not loading the JSON properly.
- The View was not updating when the database was updated and was needing server restart.

8.1.4.2 Bugs debugged

- Relative units percentage was chosen to make the layout.
- Single class based API for chat was broken down and expanded to make 3 different classes, one for inital loading,one to update the gui and one to send messages.
- It was an error caused due to the function only being called on intialization. It's function call was changed to call everytime there was a update via obserables.

8.2 Features

We wanted to make this a deployable project so instead of a large project, we tried to make the code robust with a lot of validators and proper work.

8.2.1 FrontEnd

- Single Page application with Angular routing for a smooth transition between views without reloading the web page.
- Encryption of the Login/Register Information on front end to deny packet sniffing threats.
- Catalogue as per genres.
- Catalogue as per newly added items.
- Catalogue as per author.
- Chat box for peer to peer interaction with other book owners registered in the site.
- Search bar according to genres, newly added items and author.

8.2.2 Backend

- Actual Deployment
- JWT based authentication
- MySQL connection via Django Library
- Class based Restful API.

9 CHAPTER 5: CONCLUSION AND RECOMMENDATION

With the constant supervision of the supervisor and hard work of team members, the project is finally complete. After this project we believe our team is ready to tackle more sophisticated projects in the future.

9.1 Limitations

The program has the following limitations:

- ***No Geo-location***
  The project was initially designed to show recommendations based on user location but due to the Geo-location queries' complexity, it was changed to display all the available content.
- ***Lack of channels in chat***
  We tried to actually have channels in chat so two users could connect with each other and then their messages would get saved in the database but we found it incompatible to use channels with REST APIs so the whole chat was remade using REST APIs for backend logic and storage where as Angular consumable was used to update the UI.

9.2 Further Enhancements

The following action could be undertaken to improve the program.

- ***Making a proper Geo-location service application***
  None of the team members had any experience with using the Geo-location API and the complex queries which required trigonometric functions to work. Therefore, as we would actually like to launch this service, we would like to spend more time on learning about latitudes and longitudes and properly add the functionality.
- ***Making the app single language***
  Due to the confusing stacks of programming languages, we wish to either shift to an expression express server or make it a server side Django application.
- ***Making a Mobile App*** If possible, in next semester we would like to explore mobile apps and convert this application to either work as a Hybrid app or a web view app.

10 References

Goodreads. Goodreads. (2022). Retrieved from https://www.goodreads.com/.

Subscribe to the world's largest digital library. Scribd. (2022). Retrieved from https://www.scribd.com/.

The web framework for perfectionists with deadlines — Django. Djangoproject.com. (2022). Retrieved from https://www.djangoproject.com/.

Angular. Angular.io. (2022). Retrieved from https://angular.io/cli.

Typescriptlang.org. (2022). Retrieved from https://www.typescriptlang.org/.

MySQL. Mysql.com. (2022). Retrieved from https://www.mysql.com/.

11 Appendix

Task Summary	Weeks					
	1	2	3	4	5	6
Planning						
Design						
Coding						
Testing						
Documentation						

Figure 6: Gantt Chart

Index

Task Completed