Independent University **Bangladesh (IUB)**

IUB Academic Repository

Internship Reports Summer 2023

2023-09

An Undergraduate Internship on Building A Remittance Service with FastAPI at GoZayaan

Zabir Khan, Tasfiat

Independent University, Bangladesh

https://ar.iub.edu.bd/handle/11348/663

Downloaded from IUB Academic Repository



An Undergraduate Internship on Building A Remittance Service with FastAPI at GoZayaan

By

Tasfiat Zabir Khan

Student ID: 1630545

Summer, 2023

Supervisor:

Mr. Md. Mahmudul Peyal

Research & Development Officer

Department of Computer Science & Engineering

Independent University, Bangladesh

September 21, 2023

Dissertation submitted in partial fulfillment for the degree of Bachelor of Science in Computer Science

Department of Computer Science & Engineering

Independent University, Bangladesh

Attestation

This is to certify that this report was completed by me, Tasfiat Zabir Khan (1630545), and submitted in partial fulfillment of the requirement for the degree of Computer Science and Engineering from Independent University, Bangladesh (IUB). It has been completed under the guidance of Mr. Md. Mahmudul Peyal (Supervisor), Research & Development Officer, IUB. I also certify that all of my work is original, which I have learned during my internship, including all the sources of information used in this project and report.



21/09/2023

Signature

Date

Tasfiat Zabir Khan

Name

Acknowledgement

I would like to thank the faculties of the Computer Science and Engineering department for delivering the core courses with such grace and detail, helping the students learn about the practical application of the courses. I would also like to thank my IUB supervisor, Mr. Md. Mahmudul Peyal, Research & Development Officer, Department of Computer Science and Engineering, Independent University, Bangladesh, who monitored and directed my work with his continuous guidance, instructions, and suggestions. I am also grateful to Mr. Tanvir Ahmed Palash, the Chief Technology Officer (CTO) at GoZayaan, Anwar-ul-Azim Bhuiya, our Senior Software Engineer (Backend), and the whole Backend team from the core of my heart for their support, guidance, constructive supervision, instructions, and advice, as well as for motivating me to be passionate about my work. I am lucky enough to be under their mentorship. I am also grateful to the Frontend and the App team of GoZayaan, for their support and cordiality.

Letter of Transmittal

Mr. Md. Mahmudul Peyal

Research & Development Officer

Department of Computer Science and Engineering

School of Engineering and Computer Science

Independent University, Bangladesh

Subject: Submission of Internship Report for the completion of Graduation.

Dear Sir,

I am submitting my Internship Report, which is a part of the Bachelor Program in Computer Science and Engineering curriculum. It has been a great opportunity to work under your active supervision. This report is on, “Building A Remittance Service with FastAPI at GoZayaan”. I have had the opportunity to work at GoZayaan for over 1 year, under the supervision of Anwar-ul-Azim Bhuiya, Senior Software Engineer (Backend) along with the guidance of Tanvir Ahmed Palash, Chief Technology Officer (CTO) at GoZayaan. I've gained both academic and practical experience through the internship. Along with inspiring me to learn more, it has allowed me to network with people of similar interests. With the skills I have acquired through my internship, I've attempted to be as comprehensive as I can in this report. I have adhered to the recommendations and provided adequate details for the required fields in order to write a well-organized internship report. I genuinely think that this report will help my internship program and this course achieve their goals. I would be grateful if you accepted this report and offered your constructive feedback. I hope you find this study interesting and informative.

Sincerely,

Tasfiat Zabir Khan

ID - 1630545

Department of Computer Science and Engineering

Independent University, Bangladesh

Evaluation Committee

Supervision Panel

Academic Supervisor	Industry Supervisor

Panel Members

Panel Member 1	Panel Member 2

Panel Members

Panel Member 3	Panel Member 4

Office Use

Program Coordinator	Head of the Department

Abstract

This report outlines the work of a 6 month internship at GoZayaan, a startup travel company in Dhaka, Bangladesh. GoZayaan is the leading travel and tourism company in Bangladesh known for its innovative solutions, commitment to exceptional travel experiences, and a wide range of offerings catering to diverse customer preferences.

I had the chance to learn a lot about the travel and tourism sectors during my academic internship at GoZayaan. Working with the GoZayaan team allowed me to put what I had learned in the classroom to use in actual situations, furthering the goal of the business to offer customers cutting-edge travel options.

'Hometown - Book flights & more' app is one of the products of GoZayaan, which provides hassle-free flight ticketing services to specifically Bangladeshi migrant workers in Singapore. And recently, GoZayaan has taken up a project to provide a seamless remittance service for migrant workers through the Hometown app.

When I applied at GoZayaan, I was recruited in September 2022 as a Junior Software Engineer in the Backend team, which used Django, a Python-based web development framework, to build backend systems and APIs. But we decided to use FastAPI, a Python-based web framework for building APIs, to build the APIs for the remittance service project. This project has enabled me to apply a lot of the knowledge I gained from my 4-year CSC bachelor program at Independent University, Bangladesh. My work as an academic intern started in April 2023 and focused only on building the backend APIs for the remittance project. This report covers the entire project, the research and development process that went into it, as well as what I learned while working for a company that specializes in providing travel solutions during my internship.

Keywords — GoZayaan, Travel Company, Remittance service, System Analysis, FastAPI

Contents

Attestation i

Acknowledgement ii

Letter of Transmittal iii

Evaluation Committee iv

Abstract v

1 Introduction 1

1.1 Overview/Background of the Work 1
1.2 Objectives 1
1.3 Scopes 2

2 Literature Review 3

2.1 Relationship with Undergraduate Studies 3
2.2 Related works 4

3 Project Management & Financing 5

3.1 Work Breakdown Structure 5
3.2 Process/Activity wise Time Distribution 6
3.3 Gantt Chart 6
3.4 Process/Activity wise Resource Allocation 6
3.5 Estimated Costing 8

4 Methodology 9

5 Body of the Project 10

5.1 Work Description 10
5.2 Requirement Analysis 12
5.3 System Analysis 18

5.3.1 Six Element Analysis 18
5.3.2 Feasibility Analysis 18
5.3.3 Problem Solution Analysis 19
5.3.4 Effect and Constraints Analysis 20

5.4 System Design 21
5.5 Implementation 23

6 Results & Analysis 25

6.1 Backend APIs 25
6.2 Mobile Application UI & Admin Portal UI. 28

7 Project as Engineering Problem Analysis 29

7.1 Sustainability of the Project/Work 29
7.2 Social and Environmental Effects and Analysis 30
7.3 Addressing Ethics and Ethical Issues 30

8 Lesson Learned 32

8.1 Problems Faced During this Period 32
8.1.1 Shifting to a new framework 32
8.1.2 Understanding third party API documentation 32

8.2 Solution of those Problems 32
8.2.1 Shifting to a new framework 32
8.2.2 Understanding third party API documentation 33

9 Future Work & Conclusion 34

9.1 Future Works 34
9.2 Conclusion 34
Bibliography 35

List of Figures

3.1 Work Breakdown Structure 5
3.2 Gantt Chart 6
4.1 Visualization of Scrum methodology 9
5.1 Rich Picture 12
5.2 UML Use Case Diagram 21
5.3 FastAPI and SQLAlchemy architecture 22
5.4 REST API architecture 22
5.5 API development with Swagger documentation 26
5.6 API development with Swagger documentation 26
5.7 Remittance service backend repo 25
5.8 App design in development (Figma). 25
6.1 Create remittance API swagger doc 27
6.2 Create remittance API swagger doc (success response) 27
6.3 Create remittance API swagger doc (failed response) 27
6.4 Currency rate API swagger doc 28
6.5 Currency rate API swagger doc (success & failed responses) 28
6.6 Mobile Application UI 29
6.7 Admin Portal UI for remittance feature 29

List of Tables

3.1 Resource Allocation Chart 7
3.2 Estimated costing 8
5.1 Customer Functional Requirements 13
5.2 Admin Functional Requirements 15
5.3 Six Element Analysis 18

Chapter 1

Introduction

1.1 Overview/Background of the Work

GoZayaan is a platform that mainly focuses on making traveling convenient for its users. It primarily allows users to search for, book, and ticket flights online, and also to search for and book hotels. Users can use the web view version or download the GoZayaan app from Google Play Store to use the services. 'Hometown - Book flights & more' is another product of GoZayaan that makes booking and ticketing flights to Bangladesh very convenient for Bangladeshi migrant workers particularly in Singapore. The company decided to provide a seamless remittance service for these migrant workers as well in March 2023. X is a licensed remittance service provider in Singapore, which also provides well documented APIs for third party partners like GoZayaan.

Being a Junior Software Engineer (Backend) who already knew Python, my main challenges were to first understand the requirements and simultaneously learn FastAPI (a Python-based web framework for building APIs), SQLAlchemy, Docker, then to read and understand the API documentation of X and integrate their APIs into our system, and finally to build our own APIs and test them. The frontend team and the app team would then use these APIs to create the user interface.

Users would be able to check currency conversion rates, initiate or update Know Your Customer (KYC), a term used in banking referring to a process that allows financial institutions to identify individuals they do business with, and finally initiate a remittance.

1.2 Objectives

Project objectives must be specific, measurable, must meet time, budget and organization requirements. The objectives of this project are as follows:

- Building APIs for both customers and administrators to check currency conversion rates.
- Building APIs for both customers and administrators to initiate or update KYC and check the status of KYC.
- Building APIs for both customers and administrators to initiate remittances.
- Building APIs for only administrators to create and/or update markups and policies.
- Building an API for users to upload files.
- Writing test cases for all APIs.

1.3 Scopes

The scope of my internship project at GoZayaan concerned the development of backend APIs for a remittance service targeted specifically at the Bangladeshi market. This project prompted an exciting opportunity to contribute to a cutting-edge financial technology (FinTech) solution that aimed to simplify the remittance process for Bangladeshi migrant workers in Singapore, addressing the unique needs of this user base.

The Hometown app has over 10,000 downloads on Play Store, meaning the remittance service already has a huge user base. These users would be able to seamlessly use the remittance services.

There are also plans to avail the remittance service to all Bangladeshi migrant workers in the world. So this project has a large scope.

Chapter 2

Literature Review

2.1 Relationship with Undergraduate Studies

Skills and concepts gained from the undergraduate courses have undoubtedly helped me in the development of the remittance service project during my internship. The following courses proved to be extremely useful and formed the basis for my software development journey.

- **CSC 203 - Data Structure:** This course proved to be invaluable as it taught about the different data types and structures like array, list, dictionary, tuple, set, etc. Understanding these structures and the underlying concepts has helped me understand what data types would need to be used and how to use them properly. One that I regularly used was the Python dictionary, which was used for creating requests and fetching data from responses mostly in JSON format. Python tuples and lists are also extremely useful and have been used a lot in this project.

- **CSC 305 + Lab - Object-Oriented Programming:** This course provided detailed insight into what classes and instances are in a programming context. One of the most important concepts learned in this course and also used in the project was inheritance. There were many cases where we created base classes that were later used in other classes following the inheritance concept. Thus, this course proved to be essential for my work.

- **CSC 401 + Lab - Database Management:** This course introduced Relational Database Management Systems (RDBMS), Structured Query Language (SQL), System Development Life Cycle (SDLC), Rich Picture, Entity Relationship Diagram (ERD), etc. RDBMS and SQL lectures proved to be invaluable as SQLAlchemy (a Python-based object relational mapper) and PostgreSQL (a relational database system) were used for this project.

- **CSC 405 - System Analysis and Design:** This course equipped me with the tools and techniques to design and analyze various information systems. Topics covered included understanding System Requirements, Data Flow diagrams, System Analysis, System Development Life Cycles (SDLC), Use Case Diagrams, Feasibility analysis, etc. Use case diagrams, feasibility analysis, and SDLC came in extremely handy during this internship project, as they allowed me to understand why the whole project took place and what the whole team was supposed to do.

- **CSC 453 - Software Engineering:** This course taught some of the best industry Software Engineering practices such as Gantt Charts and Process Flow Diagrams. More importantly, in-depth lectures of Functional Requirements and Non-functional requirements were essential for understanding the requirements of the remittance project for my internship from a backend point of view.

- **CSC 454 - Software Engineering Process Management:** This course played a vital role as it dove into Change Request Management, Version control, Configuration Management, and these lectures helped me deal with change management and version control for my project. I was instructed by my organizational supervisor to use GitHub for version control during the internship project.

- **CSC 455 - Web Application and Internet:** This course provided practical education about how web applications work, how frontend tools are used, and how backend systems should be designed. This allowed me to put into practice how the APIs of the remittance project during my internship should be designed and what their specific functions should be.

2.2 Related works

In 2022, the Foreign Exchange Policy Department (FEPD) of the Bangladesh Bank issued a circular allowing licensed MFS (Mobile Financial Service) providers like Bkash, Nagad, Upay, Rocket to send wage remittances to Bangladesh in cooperation with banks, digital wallet service providers, card schemes, and aggregators.

So, these MFS providers are already working with remittance services. But GoZayaan would be different.

Some of the banks in Bangladesh are working with providing remittance services, but not in such a seamless manner.

Chapter 3

Project Management & Financing

3.1 Work Breakdown Structure

Work Breakdown Structure (WBS) is a hierarchy based structure which breaks down a project into smaller nested sections. For the project, a WBS was produced so that team members have better coordination. In our WBS, we have used the top-down approach.

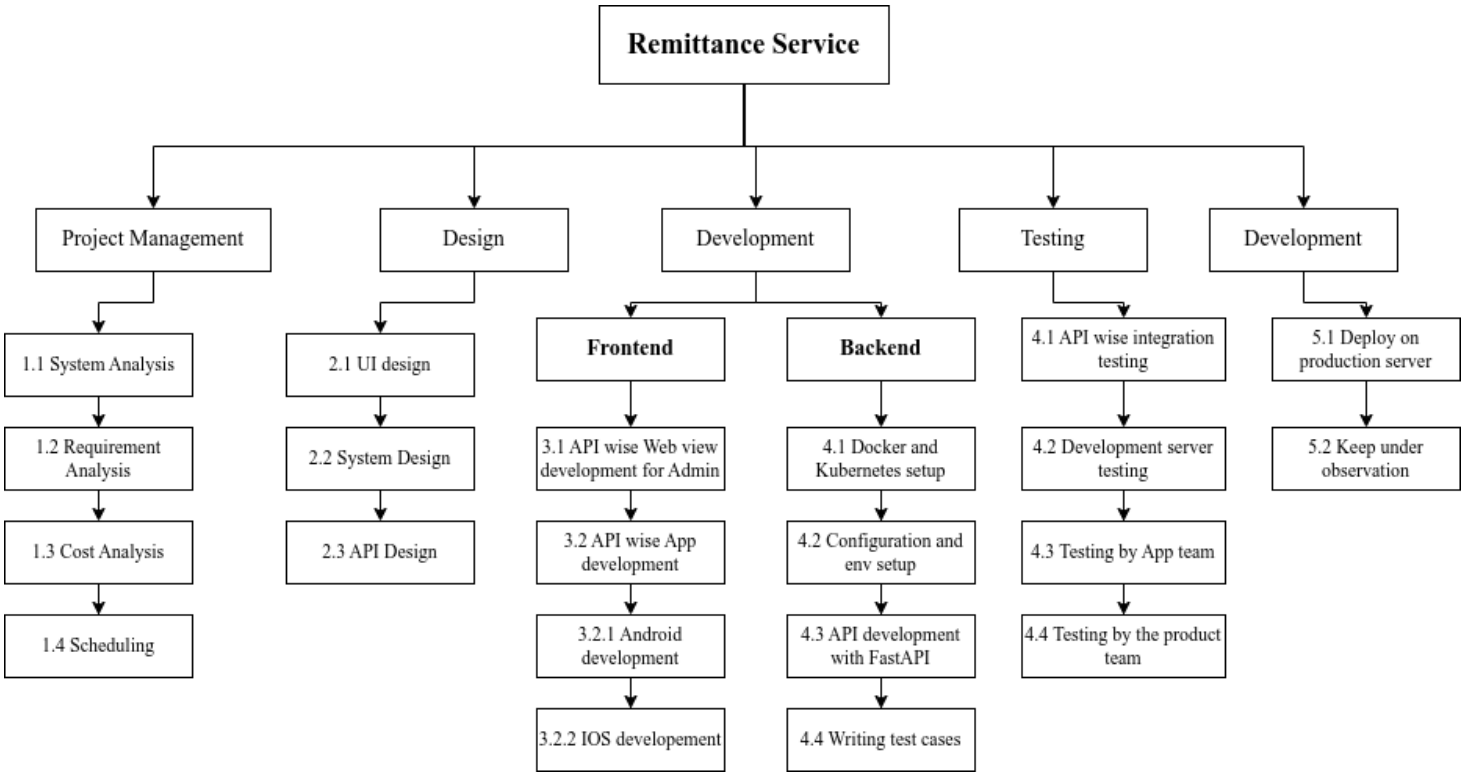


Figure 3.1: Work Breakdown Structure

3.2 Process/Activity wise Time Distribution

The estimated time required to complete the project process or task wise allocating that much time for each process or task is called process/activity wise time distribution. We also followed similar guidelines so that the team had a mental map of when each activity would need to be completed. This encouraged efficiency. There were some deadlines that needed extension due to the complications that arose from using a completely new backend framework - FastAPI for this project.

3.3 Gantt Chart

Below is the Gantt Chart that we created for our project. It was divided into 28 weeks of work and time distribution as shown in the figure:

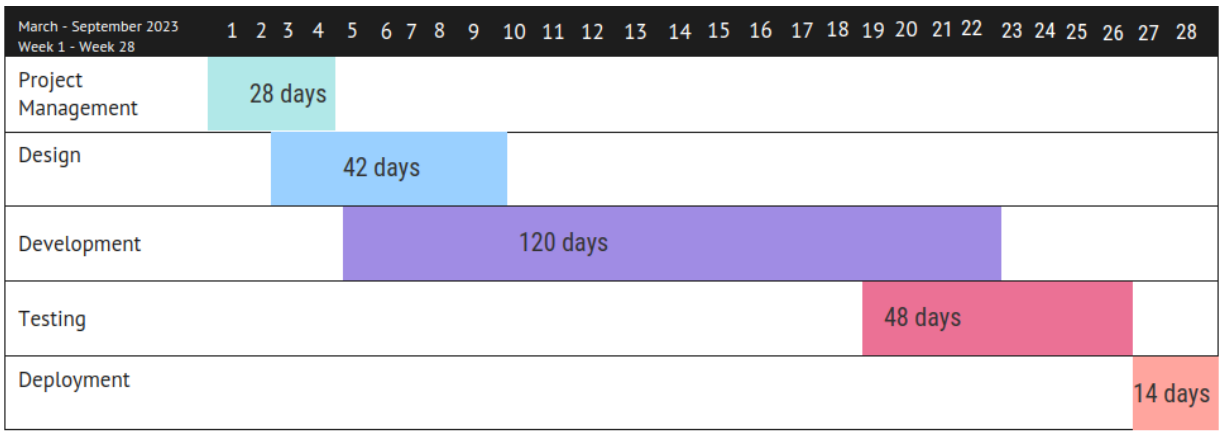


Figure 3.2: Gantt Chart

3.4 Process/Activity wise Resource Allocation

Resource allocation is the process of allocating all the required resources and assets efficiently in order to keep track of the project resources. Proper allocation of the resources is essential for successful completion of the project.

- **Project Management:** This is the period of the project where the product team, the CTO, and our senior software engineers had hours of discussion on how to design the project, how to approach the project, how to allocate the resources, the timeline of the project, the feasibility of the project, etc. This is the stage where all the important decisions were made.

- **Design:** The tech team had a designated person for UI design who would collaborate with the product team on a regular basis. The frontend and app teams worked closely to determine the UI design and work accordingly. This was where it was decided how both the user and admin UI would look and how the backend APIs should be designed to output the desired results.

- **Development/Coding:** At this stage, all the senior-level, mid-level, and junior-level developers were assigned multiple tasks on a weekly basis. Along with some coding, senior developers would manage version control, task assignment and management, code reviewing, and approving pull requests. The mid- and junior-level developers would simply focus on coding according to the requirements and testing.

- **Testing:** Testing took place throughout the development. The tech team did not have a designated testing team or person. We, the developers, were responsible for building and testing those APIs. We were also assigned to write test cases for the APIs we built. After the APIs were built, the frontend and app developers would also rigorously test the APIs. After the frontend work was done, the product team would be responsible for testing the final product and finding flaws.

- **Deployment:** Deployment required the intervention of our Devops engineer to make sure everything went according to plan. We used Docker to containerize and Google Cloud Platform (GCP) to deploy our application. Senior developers were also involved in this stage and would keep it under observation.

Table 3.1: Activity Wise Resource Allocation

Activity	Days	Work Percentage
Project Management	28	14.3%
Design (Parallel with coding)	42	21%
Development/Coding	120	61%
Testing	48	24.5%
Deployment	14	7%
Total	196	100%

3.5 Estimated Costing

The cost of the whole project was calculated by the CEO and the CTO. It includes the salary of all the employees in the tech team as the whole team focused on this specific project. This includes my salary as a junior software engineer as well. The cost of resources used was also taken into account. We did not need a new domain name or server as this project was part of an existing larger project - the Hometown app, and the remittance project would be a new product of the Hometown app. The estimated cost was Tk 20,90,000 (BDT) for the whole project.

Table 3.2: Estimated costing

Features	**Cost (BDT)**
Development	12,00,000
Design	1,20,000
GCP services	70,000
Other fixed costs	7,00,000
Total	20,90,000

Chapter 4

Methodology

There are many Software Development Life Cycle (SDLC) models being used in the industry, like Agile, Waterfall, Spiral, Iterative, Incremental, Rapid Application Development, Prototype, Extreme Programming, etc.

Scrum, an Agile project management system commonly used in software development, was used for the whole remittance project during my internship at GoZayaan. Early morning meetings were held at the start of every day, and work progress was measured and accounted for for every task.

Scrum allows teams to divide work into small tasks to be completed within time-constrained iterations, called sprints. Each sprint cannot last more than one month and commonly lasts two weeks. Our Chief Technology Officer was the Scrum Master and would review sprints in daily meetings.

Product management would hold meetings to discuss deliverables on a regular basis.

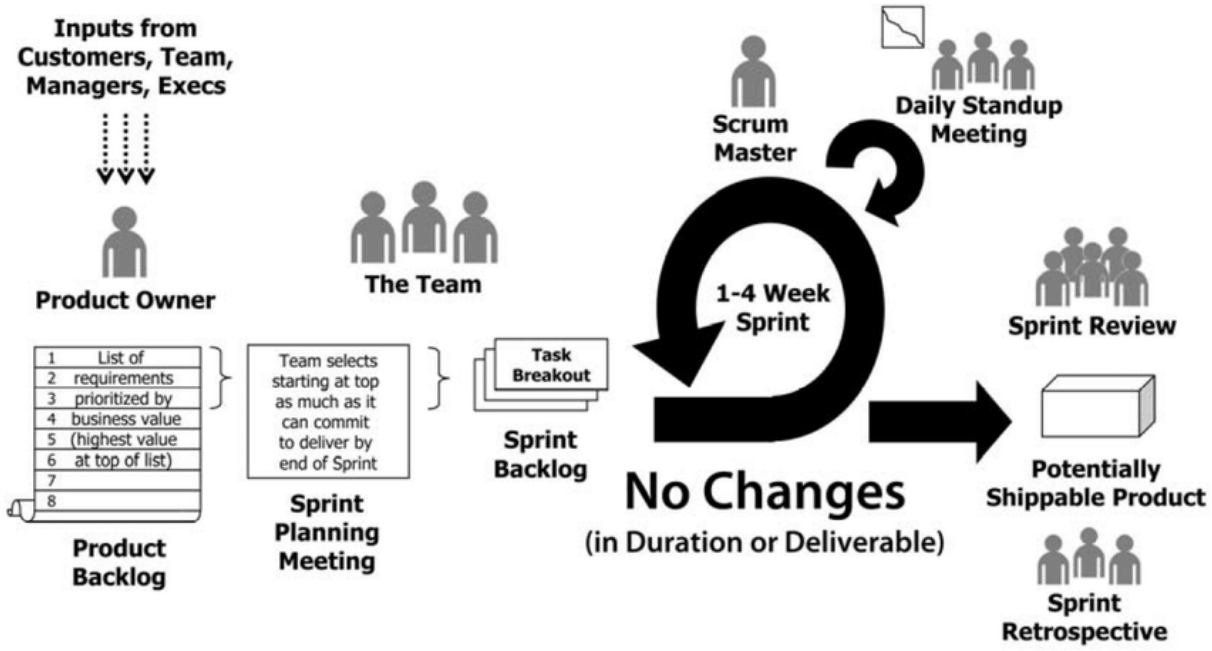


Figure 4.1: Visualization of Scrum methodology

Chapter 5

Body of the Project

5.1 Work Description

The Remittance service is a product featured within the Hometown app for migrant workers in Singapore. Users would be able to initiate a KYC (Know Your Customer), a term used in banking referring to a process allowing financial institutions to identify individuals who, in this case, would access the remittance service. Users would also be able to update their KYC data and upload pictures of their passport or any other identification document.

Users would be able to check currency conversion rates and initiate a remittance transaction for a particular rate. Users would then be able to select a bank from a list of banks or select a MFS (Mobile Financial Service) provider from a list of providers for collection purposes and provide details to create a contact. Users should also be able to select a contact from a list of previously created contacts, and finally complete the transaction and track the status of collection.

My role as a junior backend developer was to create both user and admin APIs for this system. The APIs that were built for the project are mentioned below.

User

- **Get or create users (GET):** This GET API is used to fetch the information of a certain user or create a user with the user's phone number. This API is mandatory for the user to be able to use other APIs. If the user is logged in to the Hometown app, then the frontend will use the token used in it to authenticate the user for the remittance service through this API. It creates a database session for that particular user.

Admin

- **List admin users (GET):** This API presents all admin user objects in a list with a limit and pagination applied.
- **Retrieve admin user (GET):** This API fetches a single admin user object using the ID of that user. The ID here is the primary key of the user and is stored in the database.
- Update admin user (PATCH): This API updates the status or information of an admin user.

User

- **Create contacts for users. (POST):** This API allows users to create contacts by providing receiver account details. It takes the user input, creates an instance of contact, and stores it in the database. Contacts are people to whom the money would be sent.
- **Retrieve a single contact (GET):** This allows the user to view all the details of a single contact.
- **Remove a contact (DELETE):** This allows the user to delete a contact of his choice. This also removes the contact from the database.
- **Update a contact (PATCH):** This allows a user to update the details of a contact.

Admin

- **List contacts for admin (GET):** This API lists all the contacts provided by the users and is only for admin use.
- **List contacts provided by a user. Admin API (GET):** This API lists all the contacts provided by a single user and is for admin use only.

User

- **List all banks for a user (GET):** This API lists all the banks that are allowed for collections.
- **List all branches of a bank for a user (GET):** This API lists all branches of a specific bank for collection.

Admin

- **List all banks for admin (GET):** This API lists all the banks that are allowed for collections.
- **Update details of a bank. (PATCH):** This API allows only admin users to update the details of a bank.
- **List branches of a bank (GET):** This API lists all branches of a specific bank for admin use
- **Update details of a branch. (PATCH):** This API allows only admin users to update the details of a bank branch.

User

- **Verify KYC for user (GET):** This GET API will communicate with X APIs and verify the KYC of a user.
- **Update KYC information for users (PATCH):** This API will allow users to update their KYC information. It will also communicate with X APIs.
- **Check KYC status of a user (GET):** This API will allow users to check whether or not they have been verified for KYC by X.
- **User file upload API (POST):** This API will be used by the users to upload their ID card images.

Admin

- **Verify KYC of user by admin (GET):** This GET API will be used by admins to communicate with X APIs and verify the KYC of a user.
- **Update KYC information of a user (PATCH):** This API will allow admins to to update the KYC information of a specific user.
- **Check KYC status of a user (GET):** This API will allow admins to check whether or not a user has been verified for KYC by X.
- **Update KYC status of a user (PATCH):** This API will allow admins to update the KYC status of a user.

Common

- **Get currency rates for both user and admin (GET):** This API shows the currency exchange rate between 2 currencies using X currency API.

User

- **Create remittance API for users (POST):** This API allows users to initiate a remittance. In the background it will also create a disbursement for collection. Users will be able select a contact from the ones they created and also select a collection gateway (MFS or banks).
- **Get all remittances for users (GET):** This API will be used by users to see all remittances created by them.

- **Retrieve a remittance for the user (GET):** This API will fetch a specific remittance for the user.

Admin

- **List all remittances for admin (GET):** This API will list all remittance objects for the admin.
- **Retrieve a specific remittance for admin (GET):** This will fetch a specific remittance instance for admin.
- **Update a remittance instance (PATCH):** This will allow admin to update a few specific fields in the remittance object for verification and validation.

Admin

- **List all collections (GET):** This API will list all the collections with or without applying gateway and/or status for admin use.
- **Create a collection (POST):** This will allow admins to create an instance of collection as collection requires admin verification and validation.
- **Validate a collection (PATCH):** This API will be used by admins in order to validate a specific collection

Webhook

- **Webhook API for third parties to use:** This API will be used by third parties like X to send webhooks.

5.2 Requirement Analysis

Rich Picture

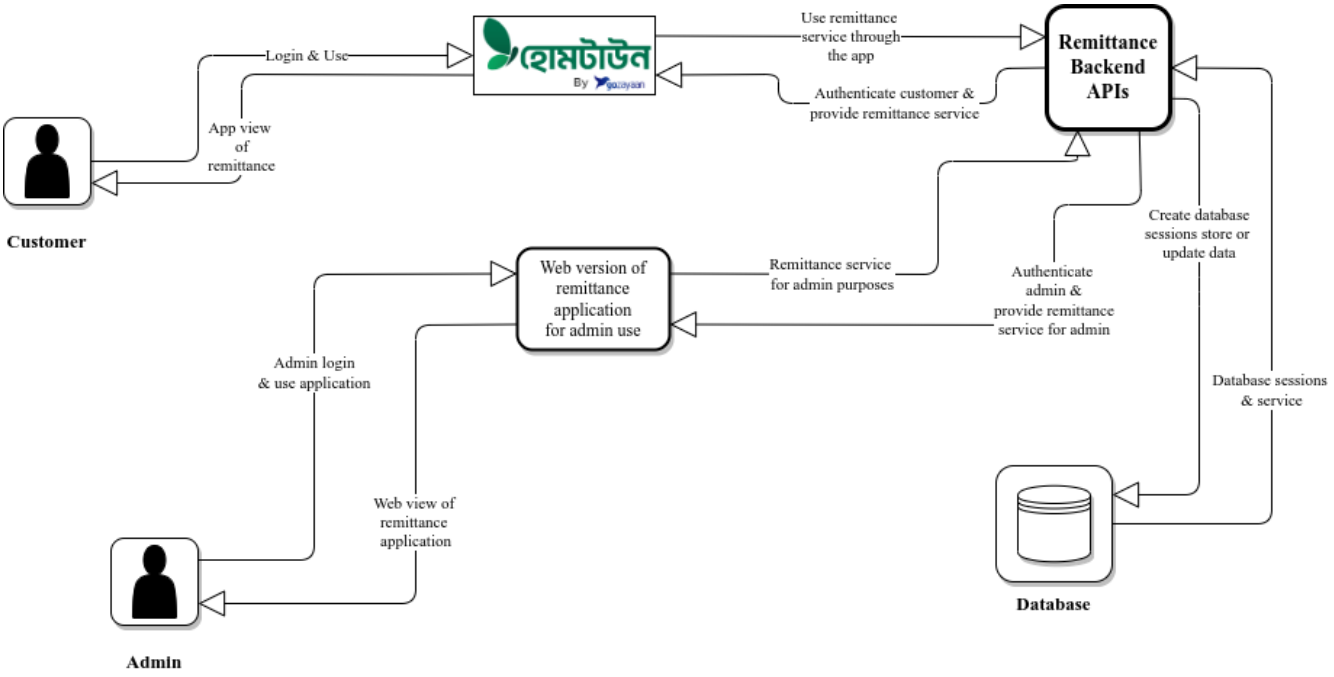


Figure 5.1: Rich Picture

Functional Requirements

Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. [1]

The following are the functional requirements for **customer** use cases:

Table 5.1: Customer Functional Requirements

Functions	**Input, output, process and conditions**
Create a contact	**Input**: Contact name, bank address or MFS wallet address. **Process**: Saves contact details. **Output**: Contact saved. **Precondition**: User must be logged in to the Hometown app and be a valid user. **Postcondition**: The contact should be saved in the database, multiple contacts can be created and be shown in a list if the user wants to select one.
Initiate KYC	**Input**: First name, last name, date of birth, phone, email, address, identification number, nationality, gender, occupation, ID card images. **Process**: Takes all the inputs and sends a KYC initiation request to X for the particular user with the provided inputs. X sends a success or error response. **Output**: KYC is initiated and is in process. **Precondition**: User must be logged in to the Hometown app and be a valid user. The input details have to be accurate. The user must know what KYC is. **Postcondition**: The input details must be stored in the database and the status of KYC should be updated upon receiving a webhook form X.
Update the KYC profile	**Input**: First name, last name, date of birth, phone, email, address, identification number, nationality, gender, occupation, ID card images (all are optional) **Process**: Updates only the fields provided by the user. **Output**: KYC profile updated. **Precondition**: KYC must already be initiated by that particular user and the status of the KYC must be in process. **Postcondition**: The updated KYC data should be saved in the database.

Check currency exchange rate	**Input**: From currency and to currency. The default for Hometown users would be SGD and BDT respectively **Process**: Send a currency exchange rate request to X and show the rate to the user. **Output**: The exchange rate. **Precondition**: User must be logged in to the Hometown app and be a valid user. **Postcondition**: If the user later proceeds with a particular rate, that rate should be saved in the database and be used for further use.
Initiate a remittance	**Input**: From currency, to currency, amount, user remarks, collection gateway, contact (receiver), source of funds. **Process**: Takes the inputs and sends a create remittance request to X, creates a disbursement and then initiates a collection upon receiving a successful webhook response. **Output**: Remittance creation is in process or failed. **Precondition**: User must be KYC verified. The currency rate may be chosen before by the user. **Postcondition**: All the details should be saved in the database. The status of the remittance should be updated upon receiving a successful webhook response. Disbursement should automatically be created at the time of remittance initiation.

The following are the functional requirements for **admin** use cases:

Table 5.1: Admin Functional Requirements

Functions	**Input, output, process and conditions**
Update customer policy	**Input**: policy id, daily attempts, monthly attempts, daily amount, monthly amount. **Process**: Take the inputs and update the policy tables corresponding to a customer. **Output**: Policy updated. **Precondition**: Must be an admin user to use this function. If no remittance was created by the customer then attempts and amounts would be saved as 0. **Postcondition**: The policy limits should be saved in the database and kept up to date.
Create banks and branches	**Input**: Bank name, bank address. **Process**: Save bank details. **Output**: Bank saved. **Precondition**: Must be an admin user to use this function. **Postcondition**: The bank details should be saved in the database and be shown in a list for customers.
Validate or update KYC of a customer	**Input**: First name, last name, date of birth, phone, email, address, identification number, nationality, gender, occupation, ID card images (all are optional). **Process**: Take the inputs and send requests to X API to update or validate the KYC of a customer. **Output**: KYC updated. **Precondition**: Must be an admin user to use this function. **Postcondition**: The updated KYC should be saved in the database. The KYC verification status should be changed to in-process.
List all collections with gateway and status filter parameters	**Input**: Gateway and/or status. Inputs can be blank as well. **Process**: Takes the parameters and applies a filtering on the dataset and shows only those data after applying filtering. **Output**: All the collections according to the filter parameters. **Precondition**: Must be an admin user to use this function. **Postcondition**: Should show 10 results per page.

Create a collection	**Input**: Remittance id, amount, gateway, source, remarks, issuer bank, account number, card number, card holder name. **Process**: Takes the inputs and creates a collection object for that specific remittance. **Output**: Collection created. Precondition: Must be an admin user to use this function. **Postcondition**: The collection should be saved in the database, multiple collections can be created and be shown in a list for the admin user.
Validate a collection	**Input**: Status, remarks, issuer bank, account number, card number, card holder name. **Process**: Takes the inputs and validates a specific collection. **Output**: Collection validated. **Precondition**: Must be an admin user to use this function. There must already be a collection and a corresponding remittance created in the database. **Postcondition**: The collection should be saved in the database. And the collection status should be updated accordingly.

Non-Functional Requirements

Non-functional requirements in software engineering refer to the characteristics of a software system that are not related to specific functionality or behavior. They describe how the system should perform, rather than what it should do.

- **Performance and Scalability:** The performance of our service is optimized as much as possible. The main purpose of using a completely new stack (FastAPI) for the remittance project was to make sure that the APIs that were built would work seamlessly. The system is also made scalable such that a large number of users can use the service without interruption.

- **Portability and Compatibility:** The Hometown app is already available on the market for both Android and iOS devices. So the remittance service is also available in both versions of the app. And the web view version is also available for admin use. So overall, the remittance service is well portable and compatible.

- **Reliability, Availability and Maintainability:** We always work to ensure that our code is divided into as many parts as we can and to eliminate any instances of redundant code. By doing so, we can modify it without causing the app to malfunction. We also have a single source of truth, which makes it simple to make changes that should affect various components without introducing unexpected errors.

- **Usability:** The UI in the Hometown app for the remittance service has been made to be as simple and user-friendly as possible, as the target customers are migrant workers living in foreign countries, specifically Singapore for now.

- **Security:** Since this is a financial service, the security of the system was given the most priority. We used JSON web tokenization methods to authenticate the customers and admins. The credentials used by users are saved securely in the database. All the APIs that were built require the specific user to be authenticated, so there is no chance for unauthorized users to be able to access the remittance service, not even partially.

5.3 System Analysis

5.3.1 Six Element Analysis

Table 5.3: Six Element Analysis

Process	System Roles - Human	System Roles - Hardware	System Roles - Software	System Roles - Database	Communication & Networks
Login	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
Create contact	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
List contacts	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
Check currency rate	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
Create a remittance transaction	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
Check collection Status	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
Check policy limits	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
Smartphone	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
Check KYC status	User	Smartphone	Android/IOS	Google Cloud Platform/PostgreSQL	WAN
List contacts for admin use	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN
Update KYC	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN
Validate KYC	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN
Validate remittance	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN
Create collection	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN
Update collection	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN
Create banks & branches	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN
Create gateways	Admin	Computer	Web Browser	Google Cloud Platform/PostgreSQL	WAN

5.3.2 Feasibility Analysis

Feasibility analysis is the measurement of a software product in terms of how beneficial it will be for the organization to develop the product from a feasible point of view. A feasibility study is conducted for a variety of reasons, including to determine whether a software project will be appropriate in terms of its development, implementation, and value to the company.

The following are the feasibility analysis of this project:

1. **Technical Feasibility:** In technical feasibility, all the resources, both hardware and software requirements, are analyzed for project development. This feasibility analysis will assess whether existing resources are enough or if other resources would be required to develop the product. For this project, we completely adopted a new framework called FastAPI. To use FastAPI, we also had to use SQLAlchemy and Pydantic. These were completely new tools to learn for the development team. Docker and Kubernetes were used for containerization. The hardware requirements did not change for this project.

2. **Operational Feasibility:** In operational feasibility, how easy it will be to operate the product and maintain it after deployment is analyzed. Along with this, other operational scopes, such as usability, are also analyzed in this feasibility study. Anyone who is accustomed to using the Hometown app to book flights can very easily navigate to this new feature and use it. Although it requires some technical know-how, it is comparatively easier for migrant workers to send remittances using this feature. Because people who are already familiar with sending remittances are expected to use the service, and they would have the technical knowledge.

3. **Economic Feasibility:** In economic feasibility, the costs and benefits of product development are analyzed. For this project, the economic feasibility study was carried out with the help of cost estimation and analysis. It was found that the benefits would be massive if the project was properly implemented, as one of the biggest sources of foreign currency income for Bangladesh is remittance. The project not only proved to be economically feasible but also economically beneficial for the organization.

4. **Market Feasibility:** The remittance service is a new feature inside the Hometown app. There is already a huge market for the app and a large user base. So the remittance feature will already have a large target customer base. The market feasibility study for this project proved to be favorable for its development.

5. **Legal Feasibility:** In legal feasibility analysis the project is analyzed from a legality point of view. To deal with the legal issues a license is required to provide remittance service to users. Since we do not have the license, we had to partner up with a company (X) that had the license. The legality of the project has been analyzed thoroughly.

5.3.3 Problem Solution Analysis

The main problems that we encountered were integrating third party APIs into our system and handling change requests. Whenever UI requirements changed during the development of the project, the backend APIs also needed to be updated. And this, in a lot of cases, caused delay in the delivery of the project. Although in SCRUM methodology requirement change is not allowed in the middle of a sprint, we had to accept the changes nevertheless because the organization demanded the changes.

Identification: Integrating third-party APIs requires a thorough understanding of their documentation. In some cases, documentation was poor, while in other cases, proper and detailed documentation was provided. One huge problem we faced was that there was no accurate expiration time for the login token that is used to communicate with the APIs of X. And the token would change for every call to the login API, and the previously generated token would become invalid. Due to this issue, we faced a lot of 'unauthorized' errors.

Solution: We had to solve it by thinking a little bit out of the box. We simply decided to store the token in our database on the first login call. Then for every other API call we used the token saved in the database. If login was called more than once from our system, it would still send the same token saved in the database for every call within a 24 hour period. A new token would be generated and saved in the database every 24 hours.

5.3.4 Effect and Constraints Analysis

Effect: Different mobile devices and their configuration make it difficult for developers as we have to consider all the devices in product development. Although Docker helps in this case, we needed to keep different OS in mind while developing the project. We needed to make sure that the codebase in the backend worked seamlessly on Android, IOS and any web browser. The backend APIs were designed in such a way that developers in all platforms could use the APIs and configure for different mobile devices.

Constraint: This was an in-house project without any external publisher. Only company resources were used in production without any assurance of money returns.

5.4 System Design

UML Diagrams

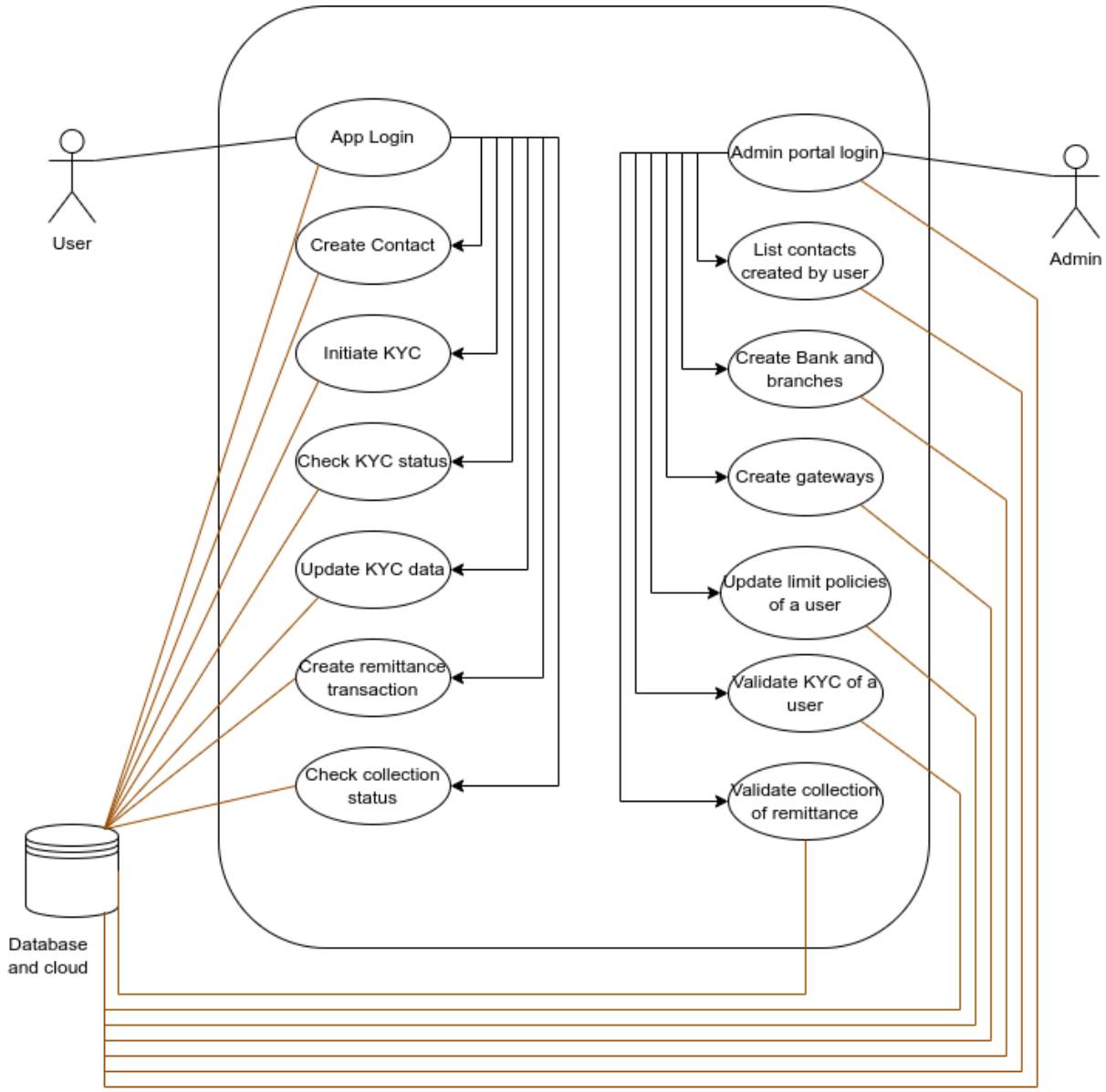


Figure 5.2: UML Use Case Diagram

Architecture

Below is a visual representation of the architecture of FastAPI, SQLAlchemy and Pydantic models:

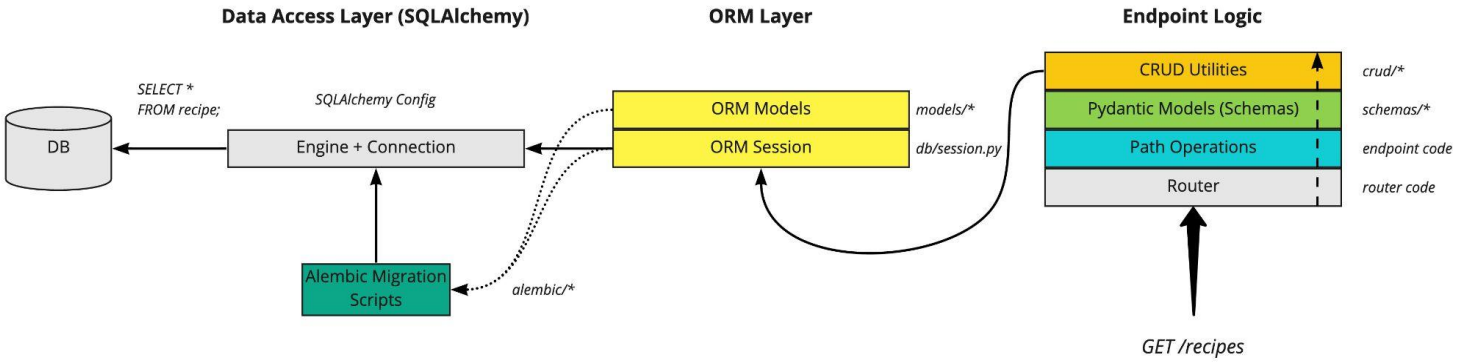


Figure 5.3: FastAPI and SQLAlchemy architecture

Below is a visual representation of the architecture of REST APIs:

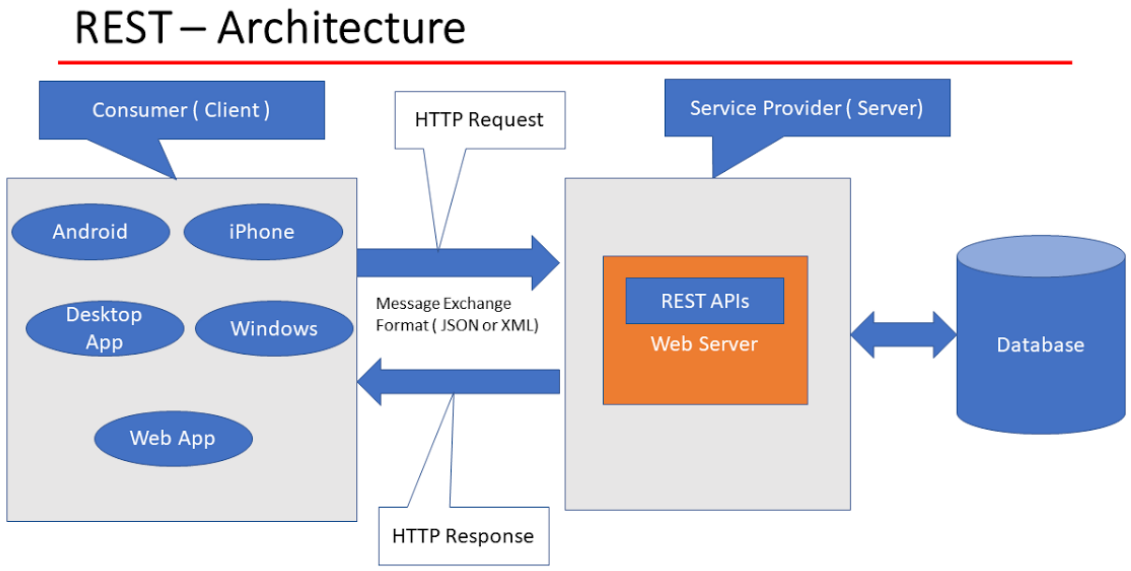


Figure 5.4: REST API architecture

5.5 Implementation

Development

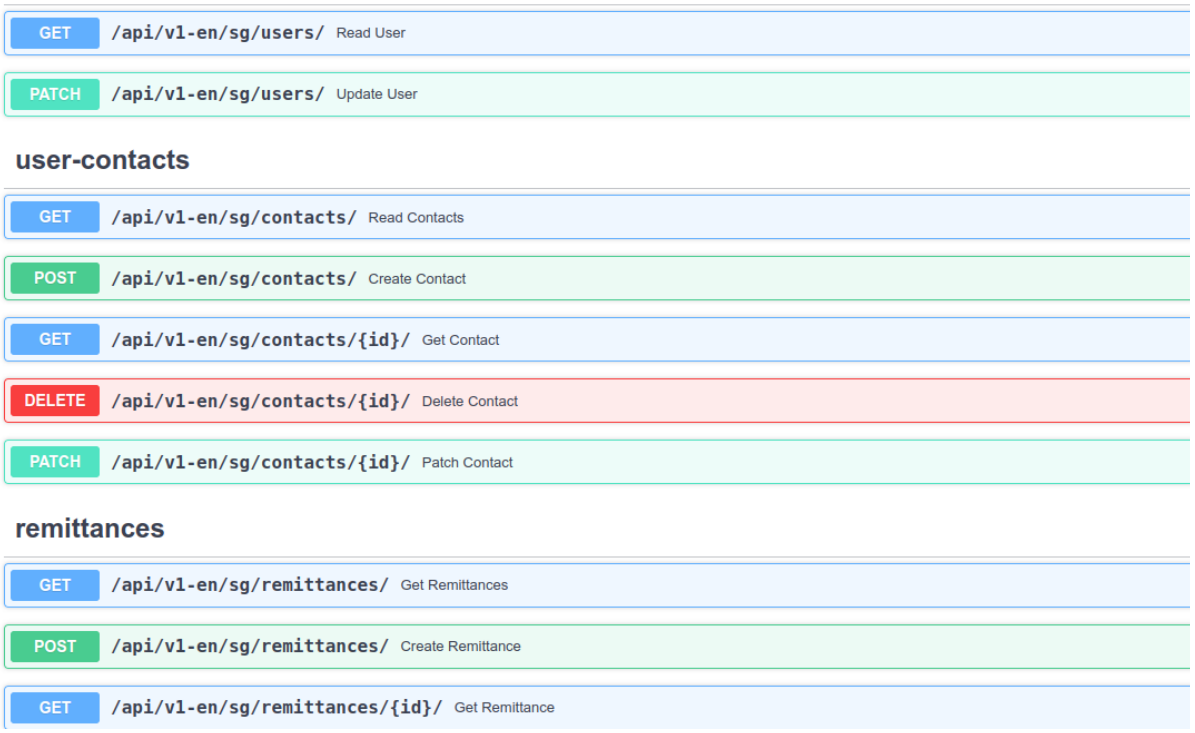


Figure 5.5: API development with Swagger documentation

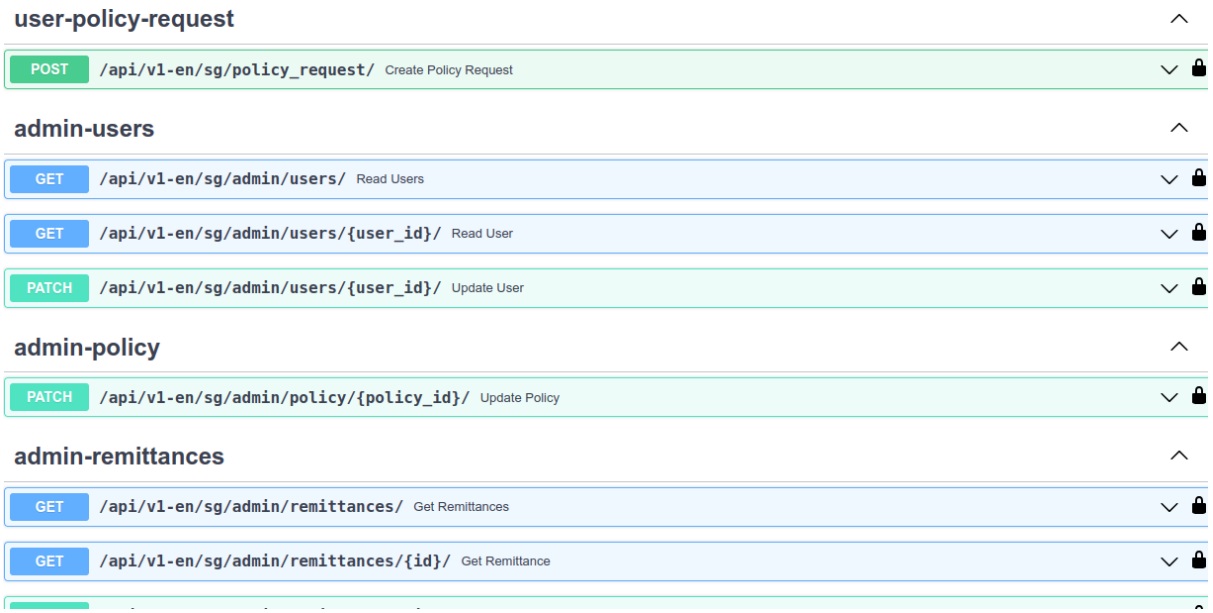


Figure 5.6: API development with Swagger documentation

Repo

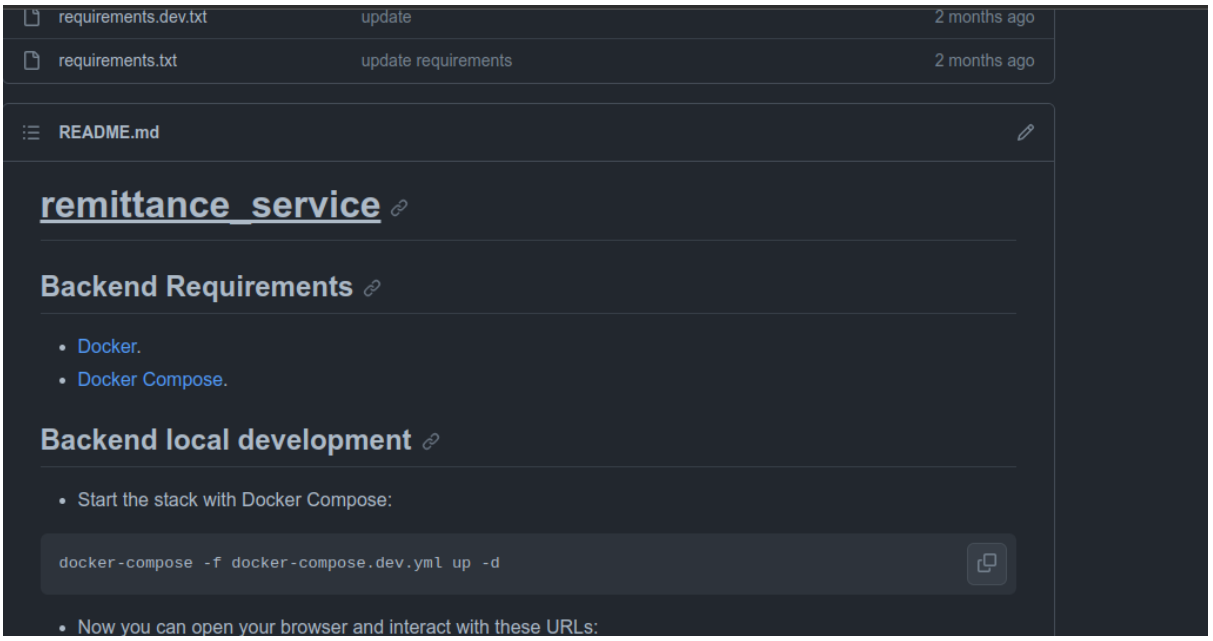


Figure 5.7: Remittance service backend repo

Design

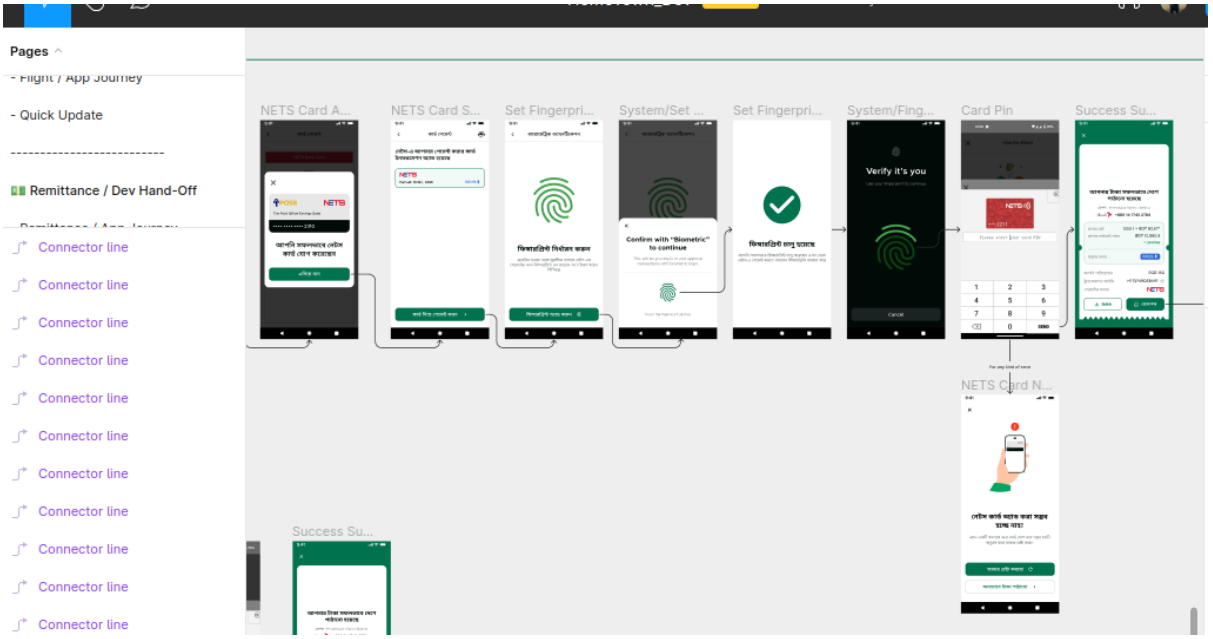


Figure 5.8: App design in development (Figma)

Chapter 6

Results & Analysis

The ‘Hometown - Book flights & more’ app has gained popularity among Bangladeshi migrant workers in Singapore since its launch. The remittance feature is still in its infancy. While in the frontend the remittance feature is just a part of the Hometown app, in the backend it is a completely separate project with a separate codebase and server. It required more work on the backend than it did on the frontend. As this internship report focuses mainly on the backend part, the results will also focus on the backend APIs.

One of the biggest perks of using FastAPI is the Swagger API documentation that comes with FastAPI.

6.1 Backend APIs

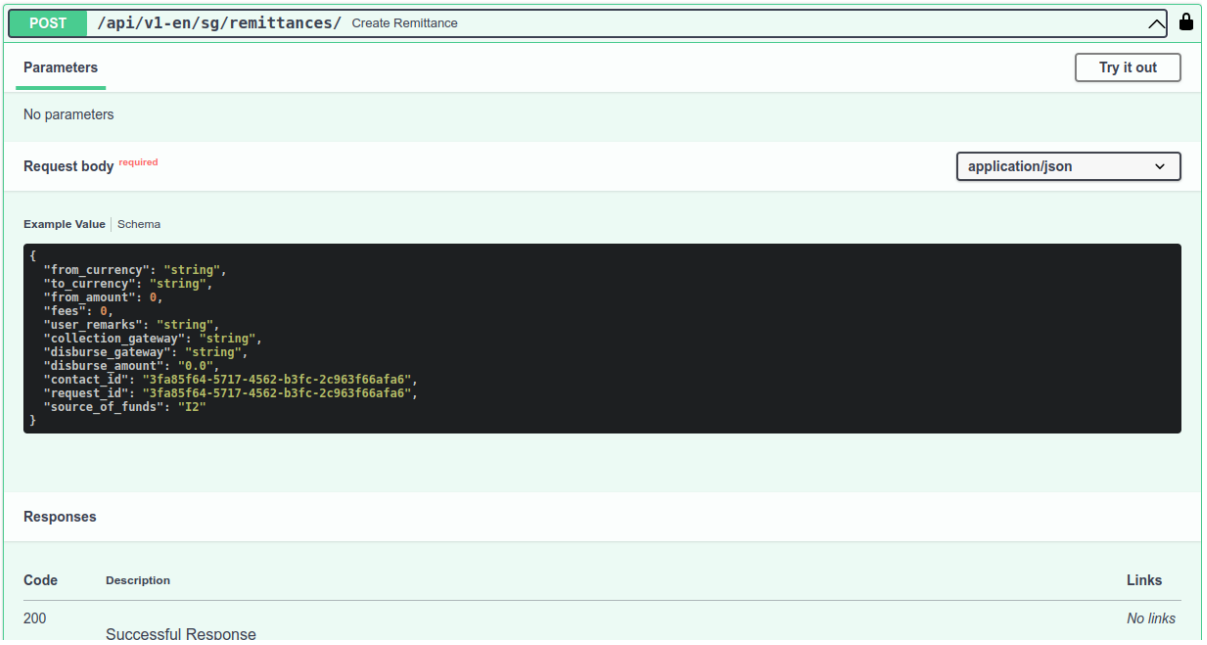


Figure 6.1: Create remittance API swagger doc



Figure 6.2: Create remittance API swagger doc (success response)

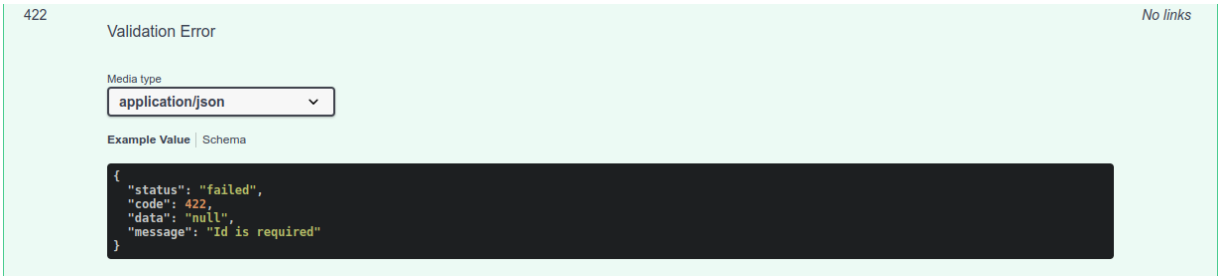


Figure 6.3: Create remittance API swagger doc (failed response)

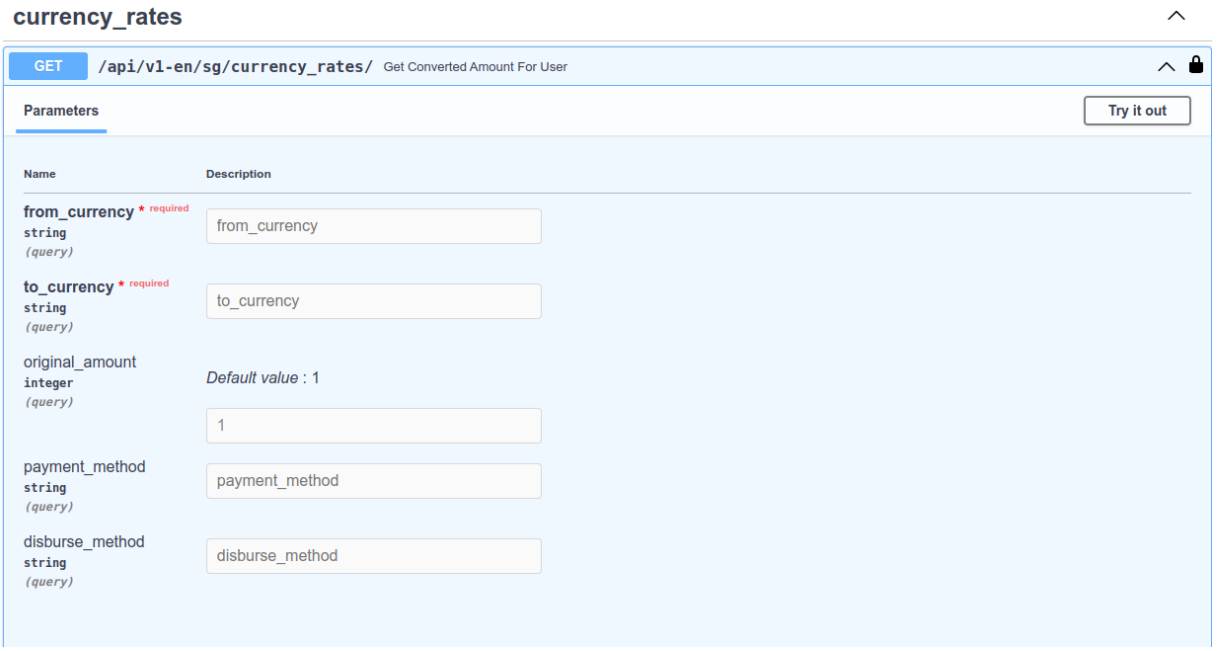


Figure 6.4: Currency rate API swagger doc

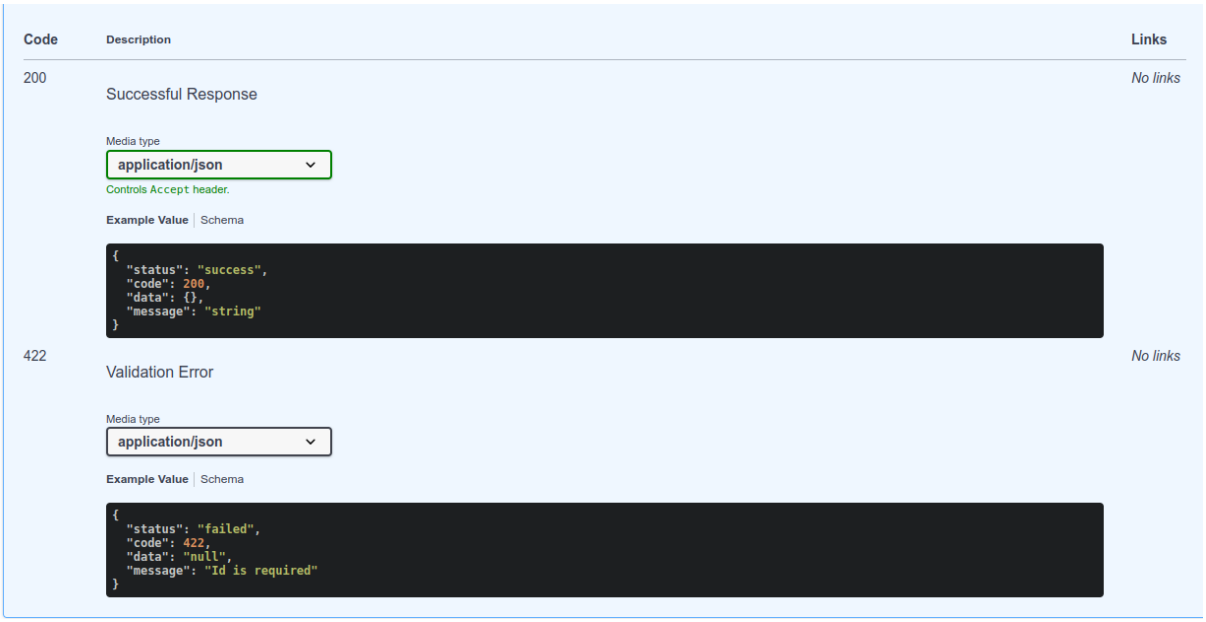


Figure 6.5: Currency rate API swagger doc (success & failed responses)

6.2 Mobile Application UI

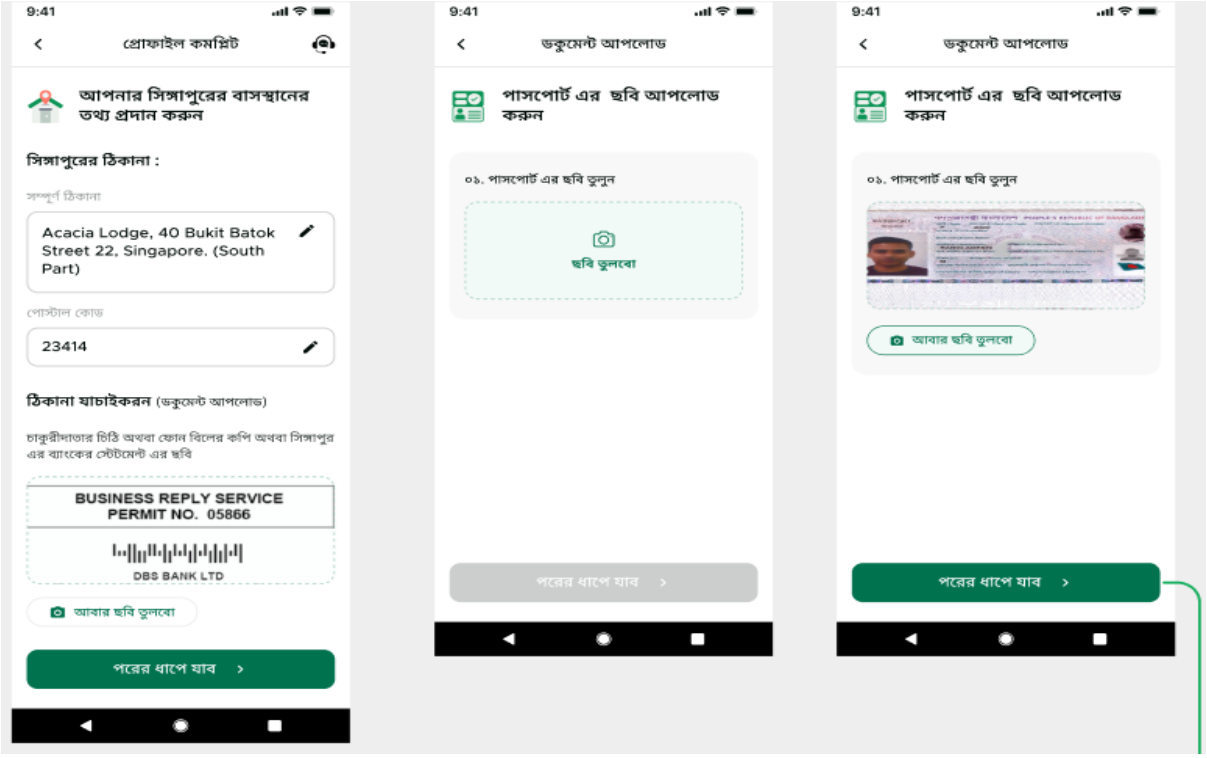


Figure 6.6: Mobile Application UI

6.3 Admin Portal UI

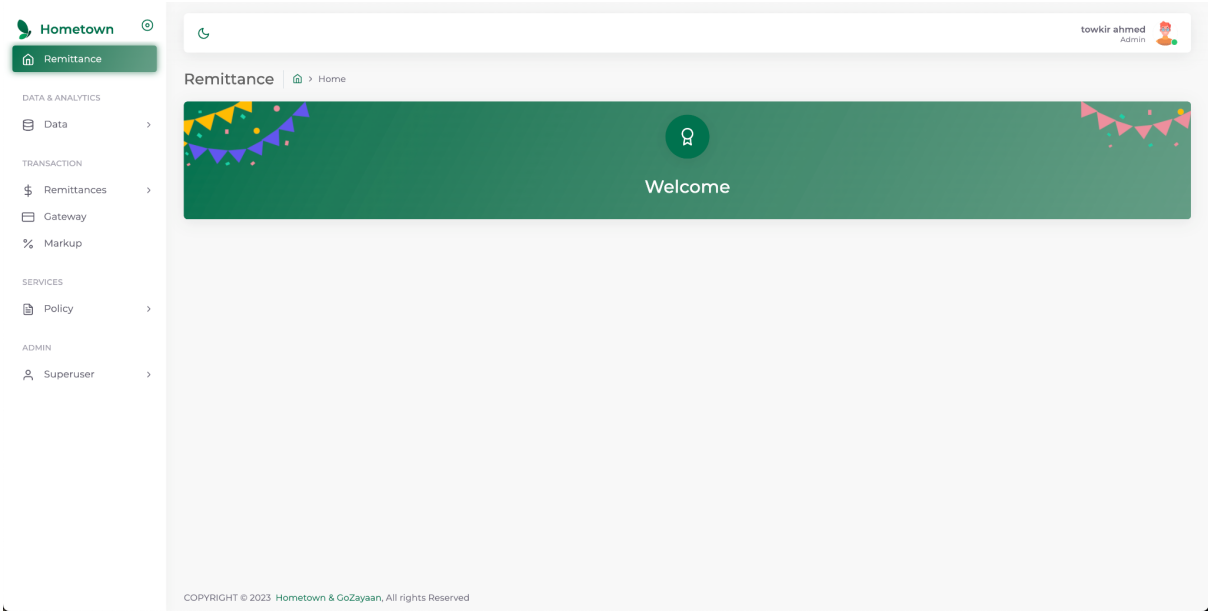


Figure 6.7: Admin Portal UI for remittance feature

Chapter 7

Project as Engineering Problem Analysis

7.1 Sustainability of the Project

Technical Sustainability

Scalability:
The underlying architecture of FastAPI assists in handling a large number of users. So, the remittance project would be scalable due to the use of FastAPI.

Maintainability:
The codebase for this project is clean and adheres to best practices. Even though there is no proper documentation, there is documentation in the codebase describing what APIs and functions do. The readability of the codebase makes it easy for beginners to work with it.

Upgradability:
The codebase can easily adapt to changes in technology, including updates to FastAPI or other dependencies, as we used Docker in the project. The project is designed to accommodate future enhancements and improvements.

Financial Sustainability

Cost-Efficiency:
The project is financially sustainable considering the ongoing operational costs, including server expenses, maintenance, and any third-party services. There is still some room for cost optimization without compromising performance or security.

Revenue Generation:
The project is destined to generate revenue as there is already an existing customer base for this service.

Security and Privacy

Data Security:
The JWT token authentication makes authentication really secure for the system and helps to prevent unauthorized access. Data related to user credentials and financial transactions are protected.

Privacy Compliance:
The project fully complies with data protection and privacy regulations and is fully committed to doing so in the future.

Community sustainability

The remittance project aims to make it really convenient for Bangladeshi migrant workers to be able to send remittances to their families. The target customer base is a particular niche.

7.2 Social and Environmental Effects and Analysis

Social Effect: The social effect of the remittance project could be huge, as this is a financial service targeted at a specific group of users. There are a lot of migrant workers living abroad. Although currently the targeted customer base is only those living in Singapore, our company aims to provide this service to all Bangladeshi migrant workers in the future. This will have a huge positive social impact on its users and Bangladesh as well.

Environmental Effects: There are no significant environmental effects to be considered.

7.3 Addressing Ethics and Ethical Issues

Data Privacy and Security

Informed Consent:
The app obtains explicit consent from users regarding the collection, processing, and storage of their data and also clearly communicates the reasons of data usage and potential sharing with third parties.

Security Best Practices:
The project has implemented robust encryption and security measures to safeguard user data against unauthorized access. The developers also regularly assess and update security protocols to address emerging threats.

Fair and Transparent Practices

Transaction Fees and Charges:
There are no hidden costs. Clear information about the financial aspects of the remittance service is always available and transparency regarding transaction fees and charges is always maintained.

Regulatory Compliance

Adherence to Laws:
We strictly adhere to the laws and regulations governing remittances and financial services. We also keep users informed about changes in regulations.

AML and KYC Compliance:
The remittance project has implemented stringent Anti-Money Laundering (AML) and Know Your Customer (KYC) procedures to prevent illicit financial activities.

Employee Ethics and Training

Continuous Training:
The company provides training for project personnel to maintain awareness of ethical issues. Also not every type of employee would be given access to the admin portal of the application.

Chapter 8

Lesson Learned

8.1 Problems Faced During this Period

The following are some of the problems I faced during my Internship:

8.1.1 Shifting to a new framework

Before working on the project, we were using Django, a Python-based web development framework. But for the remittance, we decided to use FastAPI, which was completely new to me. I also had to learn about SQLAlchemy and Pydantic, which was very challenging.

8.1.2 Understanding third party API documentation

Integrating the APIs of X into our system required a thorough understanding of their documentation. It took me days to understand their documentation properly and integrate them into our system.

8.1.3 Understanding Alembic migrations for SQLAlchemy

Alembic is a lightweight database migration tool designed specifically for SQLAlchemy. Understanding how it works is crucial for any development using SQLAlchemy.

8.2 Solution of those Problems

8.2.1 Shifting to a new framework

This required persistence. Learning a new framework from scratch is never easy, especially for an intern. But I was adamant about learning this technology for my project. The help from my supervisors proved to be invaluable.

8.2.2 Understanding third party API documentation

This required patience and persistence as well. I had to go through the documentation countless times to finally understand the flow and the functions of their APIs. Integrating any organization's API into another system requires a lot of reading.

8.1.3 Understanding Alembic migrations for SQLAlchemy

This required the intervention of my company supervisor, Anwar-ul-Azim, a lot of times before I could get comfortable with Alembic, which is still pretty much out of the depth of my understanding.

Chapter 9

Future Work & Conclusion

9.1 Future Works

The remittance project is currently targeted at Bangladeshi migrant workers in Singapore, but there are long term future plans to avail this service to Bangladeshi migrant workers all over the world. So, this project will be tested in terms of scalability in the future.

In the future, the remittance service will be separate from the Hometown app.

9.2 Conclusion

This internship experience has been an amazing experience as it taught me a lot about software engineering in general. I am thankful that I was lucky enough to work with such an amazing team and also work on such an exciting project. I am also thankful to my backend team members and the whole tech team of GoZayaan. It has been a wonderful experience.

Bibliography

[1] R. Malan, D. Bredemeyer, et al., “Functional requirements and use cases,” Bredemeyer Consulting, 2001.

An Undergraduate Internship on Building A Remittance Service with FastAPI at GoZayaan

By

Tasfiat Zabir Khan

Student ID: **1630545**

Summer, 2023

Consent from Supervisor

The student modified the internship final report as per the recommendations made by his/her academic supervisor and/or panel members during and/or before final viva, and the department can use this version for archiving as well as the OBE course material for CSE499.

This internship report is checked with Turnitin plagiarism checker, and the score is:

Turnitin Score (%) : 8%

(Signature of the Supervisor)

Mr. Md. Mahmudul Peyal
Research & Development Officer
Department of Computer Science & Engineering
Independent University, Bangladesh