CS 5200
Database Management Systems

Project Report

Logitt - your savings Companion

Group Name : SainiDhPantMu
Group Members : Mumuksha Pant , Dhruv Saini

ABSTRACT

Do you ever see a cute black dress and think *'Oh, its cute, I'll buy it'* only to regret your decision afterwards? Well, *impulse buying* and *unconscious spending* can really be your worst enemy and soon you could develop a habit of mindless spending, even before you realize it & when you probably do, it could be a little too late ..

Logitt is a one stop solution which will help you in analyzing your spending patterns by keeping a track of daily spends which would in turn help in saving money and stopping the wasteful expenditure.

Logitt is an Expense Management System where you can log your daily expenses in your preferred currency and manage personal finances easily.

TABLE OF CONTENTS

Chapter 1: INTRODUCTION 04
1.1 Overview 04
1.2 Problem Statement 04
1.3 Glossary 04

Chapter 2: REQUIREMENTS SPECIFICATION 06
2.1 Overall Description 06
2.2 Specific Requirements 06
2.2.1 Technical Specifications 06
2.2.2 Hardware Requirements 07
2.2.3 Technology 07
2.3 Intended Audience 07

Chapter 3: DETAILED DESIGN 08
3.1 System Design 08
3.2 Conceptual Design .....
3.3 Logical Design 09
3.4 User Flow 10
3.4.1 User 10
3.4.2 Administrator 12

Chapter 4: IMPLEMENTATIONS 13
4.1 Stored Procedures 13
4.2 Triggers 13
4.3 Functions 13

Chapter 5: SNAPSHOTS 14

Chapter 6: CONCLUSION 21

Chapter 7: FUTURE ENHANCEMENTS 22

Chapter 8: LESSONS LEARNT 23

Chapter 9: REFERENCES 24

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Logitt is an ultimate stop solution for analyzing spending patterns, helping users save money and prevent wasteful expenditure. As an Expense Management System , Logitt enables users to effortlessly log their daily expenses in their preferred currency and manage personal finances .

With a user-friendly dashboard and filters and ability to track expenses for different categories like Shopping, Groceries, Utilities, etc *Logitt* enables its users to stay on top of their spendings and make more informed financial decisions.

1.2 PROBLEM STATEMENT

Impulse buying can be defined as buying a product or a service without careful consideration or planning. Rather than rational thinking, this type of buying is based on emotions and urge.
While it may have a temporary sense of happiness, it could have negative impact on the consumer's finances. One of the primary concerns with impulse buying is that it could lead to overspending and exceeding budget limits which is why it is considered problematic.

This is where *Logitt* comes into play by empowering users to not only log but track their expenditures based on different categories and make more informed financial decisions. Not only this, No matter in which currency the user spends, *Logitt* will covert it into the User's domestic (preferred) currency and reflect the same in his wallet.

1.3 GLOSSARY

Term	Definition	Actions in Logitt
New User	"New user" is a person who has not been registered to the system. In order for a new user to access the dashboard and start using the *Logitt* portal, he will have to Sign Up where his details are required. All the fields are mandatory.	Signup functionality *A new(unregistered) user will not be able to use the Logitt portal.*

Registered User	A “registered user” is a person who has signed up to use *Logitt* portal. With a valid username, the registered user can enter into his dashboard to view Expenses, Deposits and Feedbacks .	Log an Expense Log a Deposit Delete an Expense Delete a Deposit View *Logitt* History Split an Expense Add Feedback Add Query Update User Information Delete User
Administrator	An “admin” short for administrator is a person who has the responsibility to manage the *Logitt* portal.	View all Queries and Feedbacks Delete any queries / feedbacks. Delete Users from the system. Manage Currencies, ie , add a new currency, update an existing currency or delete a currency Manage Categories , ie, add a new category, update an existing category or delete a category.

CHAPTER 2

REQUIREMENTS SPECIFICATION

2.1 OVERALL DESCRIPTION

The *Logitt* Portal is a web based application which is built using Mysql and Java primarily where in we aimed to achieve the solution for our problem statement . Our solution is aimed at creating a one stop ultimate product for the consumers to log their expenses and achieve their financial goals. Our solution offers a seamless experience for the users to manage their finances. We have tried to create our UI as interactive as possible.

2.2 SPECIFIC REQUIREMENTS

Specific Requirements for *Logitt* Are Mentioned as Below -

2.2.1 TECHNICAL SPECIFICATIONS

		Version
IDE	IntelliJ Idea CE	2022.3.1
	Visual Studio Code	1.77.3 (Universal)
Database Support	mysql Workbench(Community)	8.0.29 build 1751076
	mysql-connector-j	8.0.32
Java	jdk	1.7
Web Browser	Google Chrome	112.0.5615.49 (Official Build) (arm64)
Operating System	macOS	13.0(22A380)
	Windows	10
Frameworks	Springboot	
	ReactJS	Nodejs 16

Technical Specifications

2.2.2 HARDWARE REQUIREMENTS

Keyboard , Mouse
Monitor : 13.6-inch (2560 × 1664) Liquid Display
Hard disk : 3 GB+
RAM : 4 GB+
Processor i5

2.2.3 TECHNOLOGY

SQL
Sql is a structured query language used for database creation and manipulation. SQL is used to communicate with the database. It is used for storing, retrieving and manipulating the data stored in the relational database.

Java
Java is a multi-platform, object-oriented programming language which can be used as a platform in itself. It is fast, secure, robust and reliable. Java is designed to be compatible with different operating environments.

JDBC
JDBC stands for Java Database Connectivity. It is a Java API used to connect Java Program to the database. JDBC helps to write Java applications that manage connecting to database, sending queries and updating the database. It can access any tabular data.

Spring boot
Spring boot is an open source Java based framework which is used to create micro service. It is built on Spring Framework and provides a faster and easier way to develop, build and configure web applications and micro services.

SwaggerUI
SwaggerUI helps to visualize the APIs with all the logic complexity encapsulated

NodeJS
It is an Open Source run time environment used for executing Java Scripts

2.3 INTENDED AUDIENCE

- Typical users such as students, who want to budget their monthly finances using *Logitt*
- Impulse buyers who want to control their overspending and make more informed financial decisions

CHAPTER 3

DETAILED DESIGN

3.1. SYSTEM DESIGN

The web app is built using the java application and ReactJs framework where the user can perform creation, updation, deletion and retrieval of expenses and deposits. These CRUD operations are managed using the SQL Stored procedure that are called using JDBC.
In Spring boot , we are using MVC (model view controller) architecture to organize our web application into distinct layers and each layer has a specific responsibility.

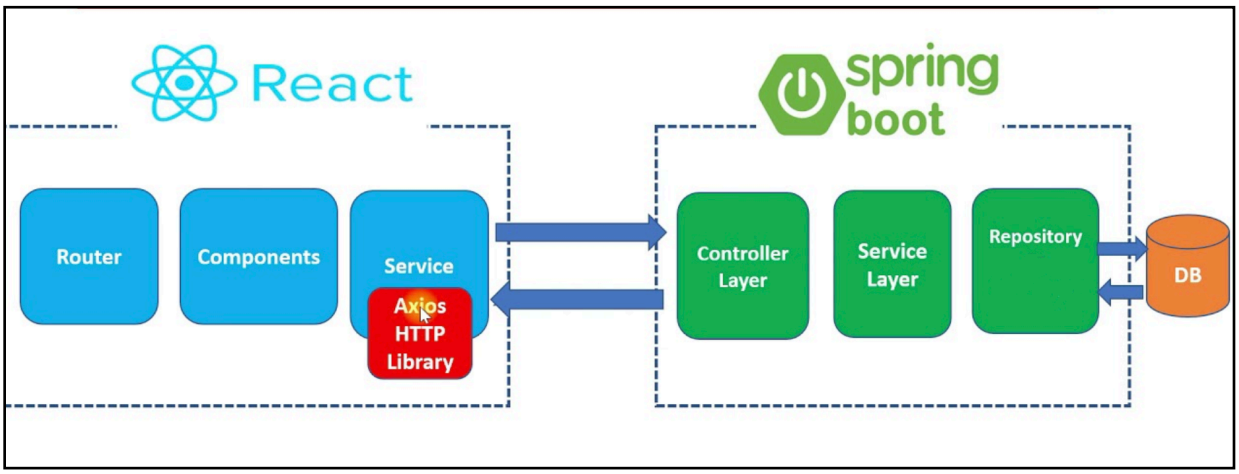


Fig 3.1 - System Architecture

I. **Model :** It is used for defining the structure of the data which will be used by our application

II. **Controller :** It is responsible for handling incoming requests and returning the responses. In our application we are using rest controllers.

III. **Service :** It is used for encapsulating the logic of the application.

IV. **Repository:** It is used for interacting with the data stored in our mysql workbench database . It is implemented using JDBC.
V.

3.2 ENTITY RELATIONAL DIAGRAM (CONCEPTUAL DESIGN)

There are 7 tables in our system, namely :

I. user
II. deposit

III. expense
IV. feedbacks
V. currency
VI. categories
VII.logit_history

This application supports two roles : ***Admin*** and a ***Registered User*** of the system.

Every Registered user can log multiple expenses and deposits .

The main user of this portal will be the registered 'user' which will have a unique username, firstname, lastname, dob, preferred currency(domestic), phone number, email and gender.

Using the **'is_admin'** flag, a user can be made as the admin of the system and an admin will have different privileges .

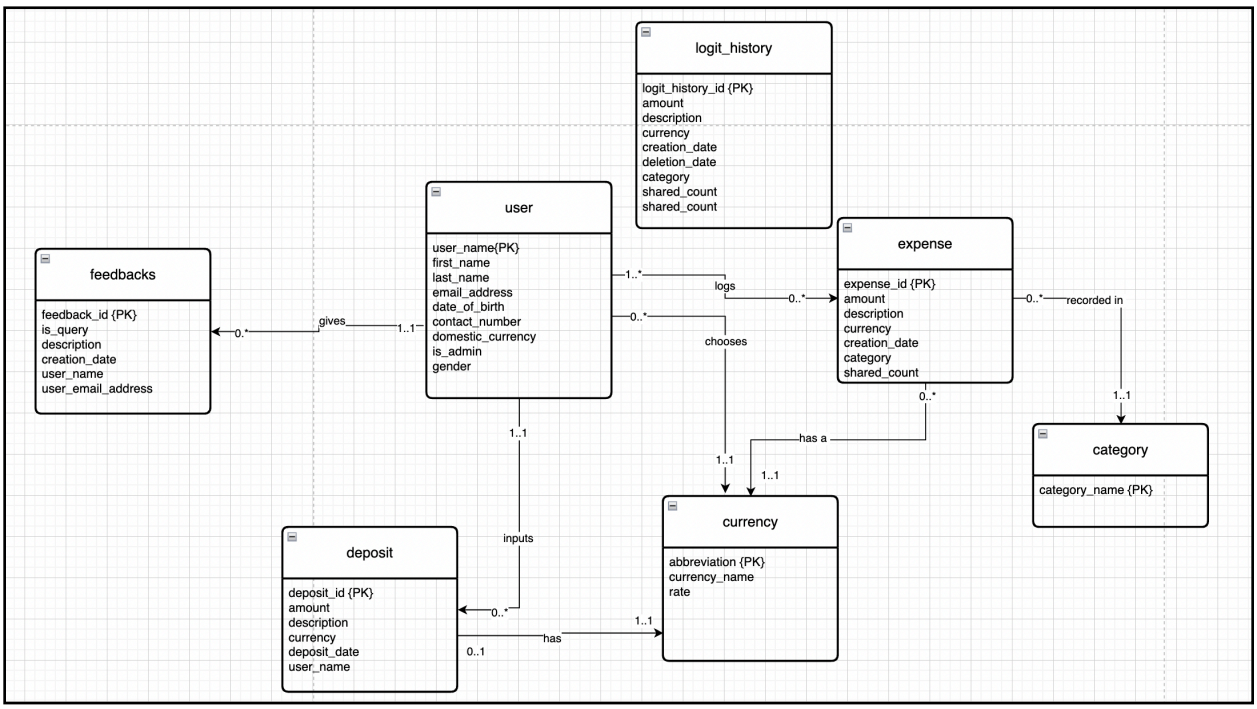


Fig 3.2 - Conceptual Design - Entity Relational Diagram

3.3 LOGICAL DESIGN

Logical design for *Logitt* using Reverse Engineering of the data is shown below. The logical design is the process of transforming the conceptual design (ERD) of the application into a schema for data modeling.

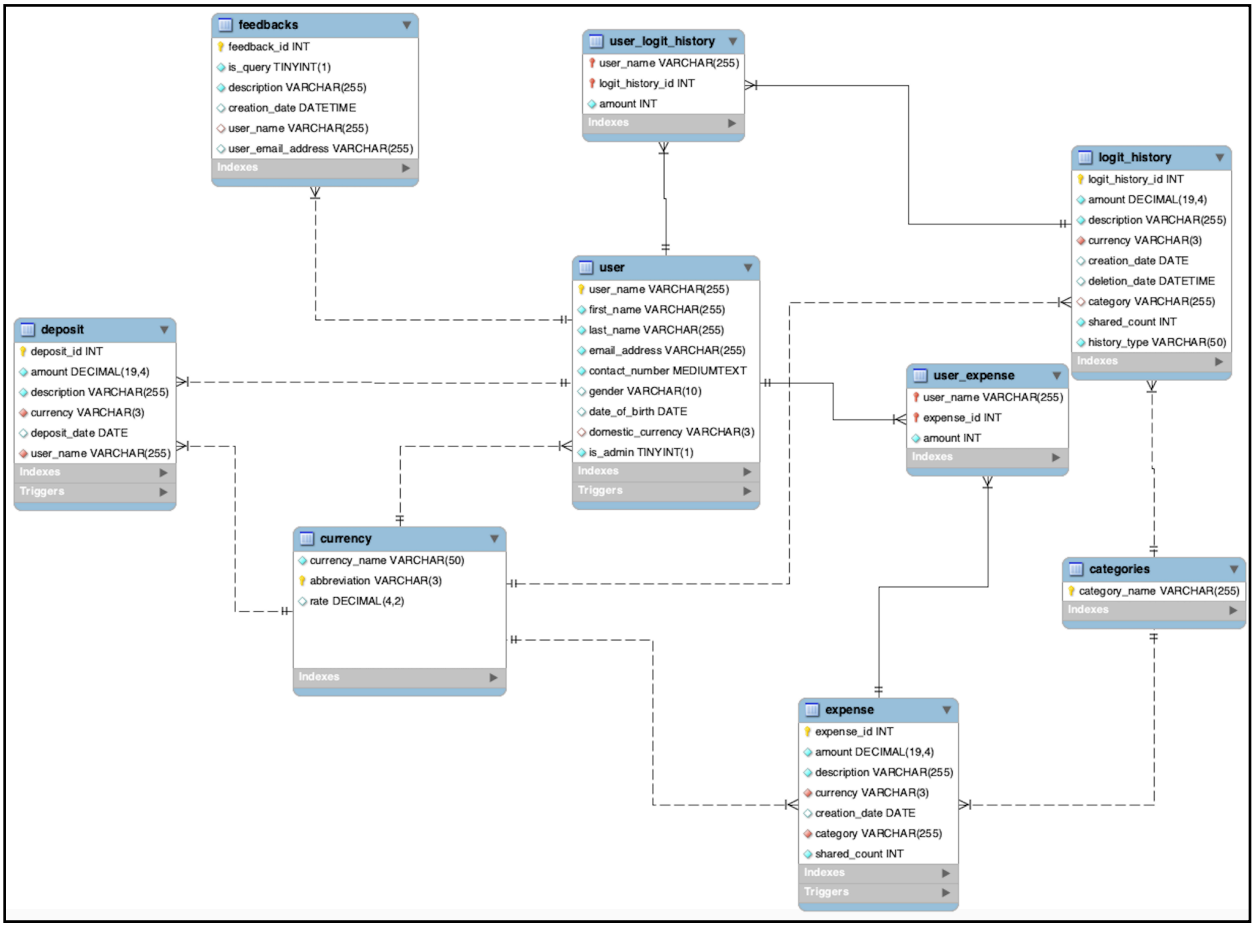


Fig 3.3 - Enhanced Entity Relational Diagram of Logitt

3.4 USER FLOW

3.4.1 User

Here is an activity diagram which represents the user's action in our system .
The user can add expenses, deposits and add in feedbacks or queries . An added expense can also be equally split among different users .

- A registered user can log in to the *Logitt* portal by clicking on **"Sign in"** and entering a valid username.

- A new user can create on **"Sign Up"** to create a new account. On clicking the sign up button user is redirected to a screen where they are required to enter their Username(unique), First Name, Last Name, Date of Birth , Email Address, choose a Gender from the dropdown menu, Contact Number, and a preferred currency from the dropdown menu. Clicking on "Create User" will create a user in the *Logitt* system.

- On successful login, the user is directed to the **dashboard which displays various options to the user** - "Add Expense," "Add Deposit," "Add Feedback," and "Update User."

- To log an expense, the user clicks on the **"Add Expense"** button which directs them to a screen where they can input the amount, select a category from the dropdown list, add a description, select a

currency from a dropdown, choose an expense date and click on the "Submit" button to log the expense into the system.

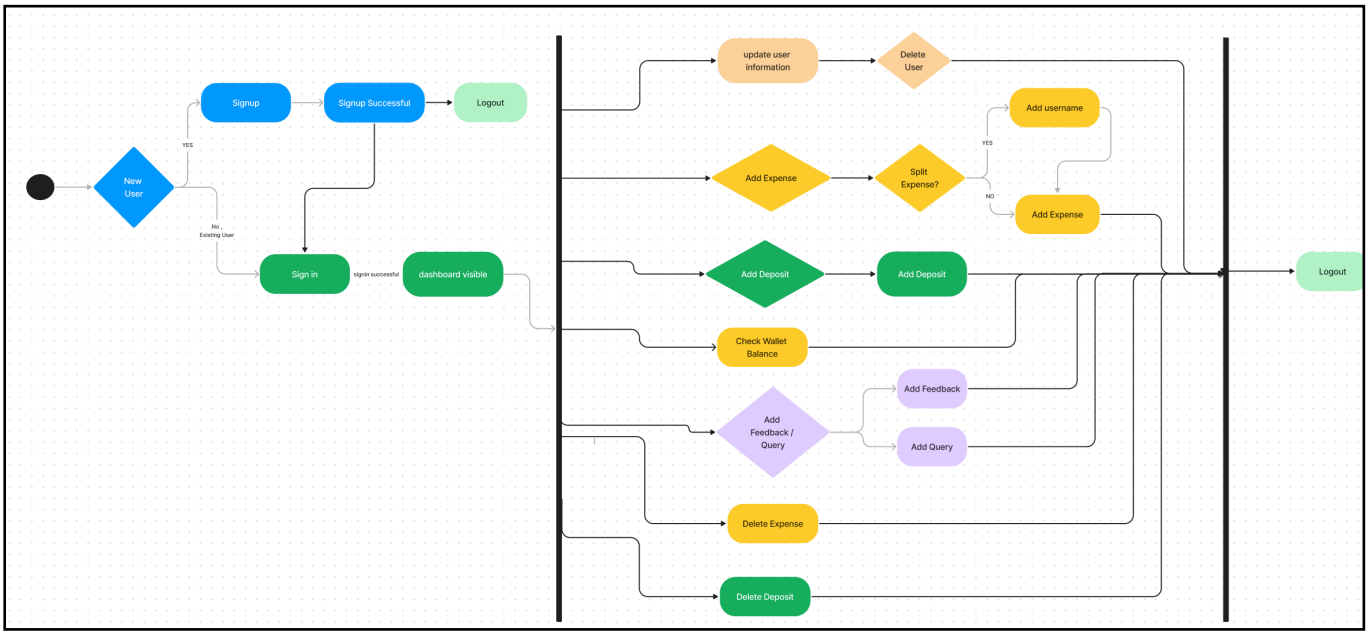
- If the user wants to split an expense among friends or family , they can do so by adding a valid username(s) in the "**Split expense ? Add usernames: ”** textbox, and the expense will be equally shared among the selected users.

- To add a deposit, the user clicks on the **"Add Deposit"** button which directs them to a screen where they can input the amount, select a currency from a dropdown menu, add a deposit date (optional), add a description, and click on the "Add Deposit" button to submit. *Logitt* system will then add the funds to the user's account.

- The user can also check their **wallet balance and the Wallet Balance will automatically convert the expenses and deposits to the user’s domestic currency, no matter in which currency they are logged with.**

- If the user wants to provide feedback for the system or their experience, they can select the "**Add Feedback"** option from the dashboard. By default, the "Is this a query" field is set to "NO." If the user is asking a query, they can select "Yes" from the dropdown menu, enter the description, and click on the "Submit" button to add their feedback or query to the system.
  Any feedback added will be displayed to all the users on their dashboards, and a query will only be visible to the admin.
  Also, the user can update his/her information using the update User option on the user dashboard. The same update modal can be used to delete the user as well.

Clicking on “**Logout”** will log the user out of the *Logitt* portal.



3.4.2 Administrator

Here is an activity diagram which represents the admin's action in our system .
The Admin has the responsibility to manage the *Logitt* portal. An admin can :

Manage Categories: An admin can add new categories, update the existing ones, and delete the categories .

Manage Currencies: An admin can add new currencies, update existing currency values, and remove outdated ones.

Manage Users: An admin can update his own user information. Also, the admin can delete the users that are registered with the system.

Manage Feedback and Queries: An admin can delete feedbacks and queries that are no longer relevant or are inappropriate.

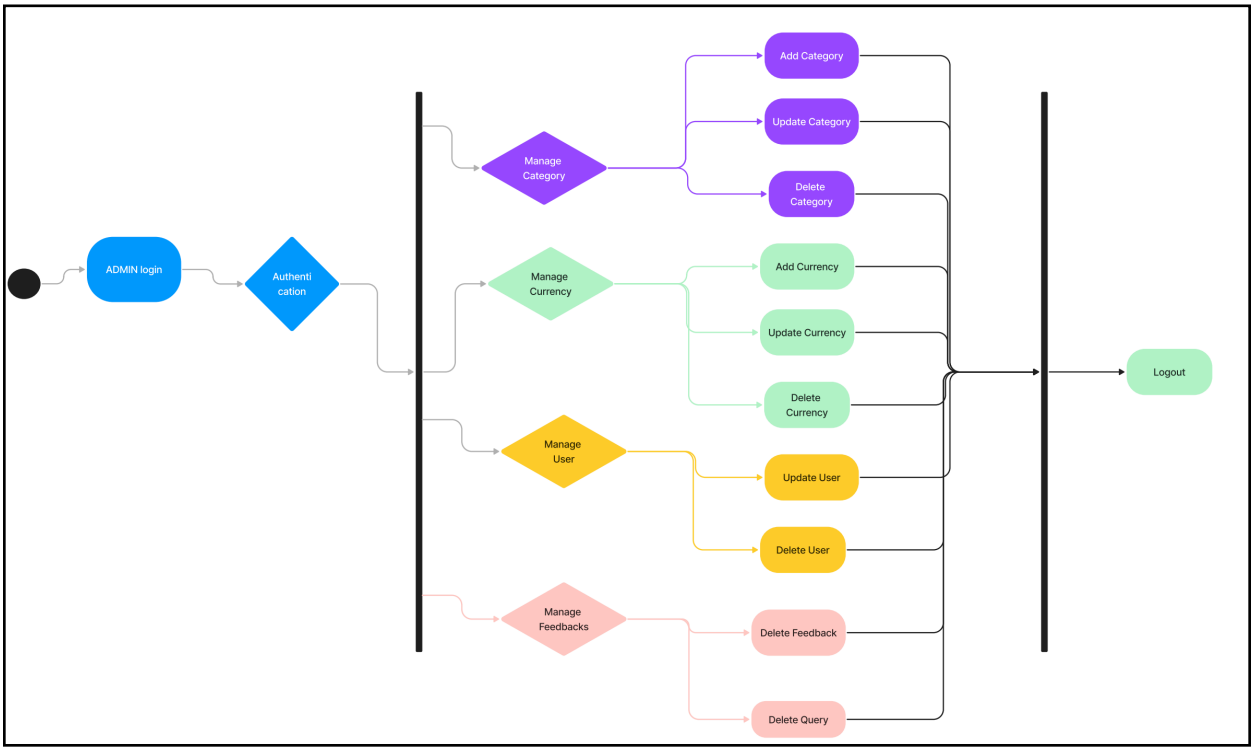


Fig 3.5 - Administrator Activity Diagram

CHAPTER 4

IMPLEMENTATIONS

4.1 STORED PROCEDURES

1. **create_category :** to create a category
2. **create_currency :** to create a currency
3. **create_user:** to create a new user
4. **calculate_user_expense :** to calculate the wallet balance
5. **delete_category:** to delete a category
6. **delete_currency:** to delete a currency
7. **delete_deposit:** to delete a deposit
8. **delete_expense:** to delete an expense
9. **delete_feedback:** to delete a feedback
10. **delete_user:** to delete a user
11. **find_expense_by_username :** to find the expense when a username is entered
12. **find_feedbacks:** list all feedbacks
13. **find_history_by_username:** find the deleted history basis the username
14. **find_queries:** list all queries
15. **find_users_with_feedback :** find users who have entered the feedbacks
16. **getUserDetails:** to get the details of all the users existing in the system
17. **insert_deposit:** to insert a deposit
18. **insert_expense:** to insert an expense
19. **update_category:** to update any category
20. **update_currency:** to update any currency
21. **update_user:** to update the user information details

4.2 TRIGGERS

1. tr_deposit_delete : Trigger on Deposit table - Before user deletes any deposit, it is logged in the logit_history table

2.tr_expense_delete: Trigger on Expense table - Before user deletes any expense, it is logged in the logit_history table

3.delete_user_expenses: Trigger on user table - Before user deletes any expense, it is logged in the logit_history table

4.3 FUNCTIONS

1. **convertUSDtoDomesticCurr :** used in Stored Procedure “calculate_user_expense” to calculate the wallet balance. This function is converting total wallet balance of user in USD to user’s domestic currency.

CHAPTER 5

SNAPSHOTS

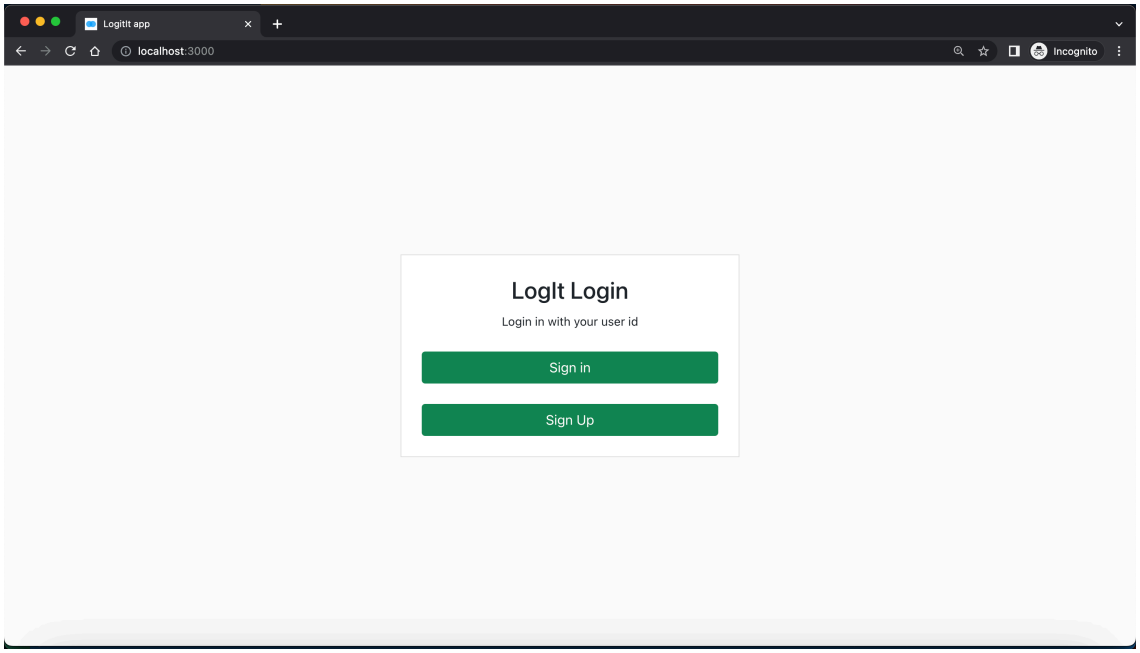


Fig 5.6 - Logitt Main Screen

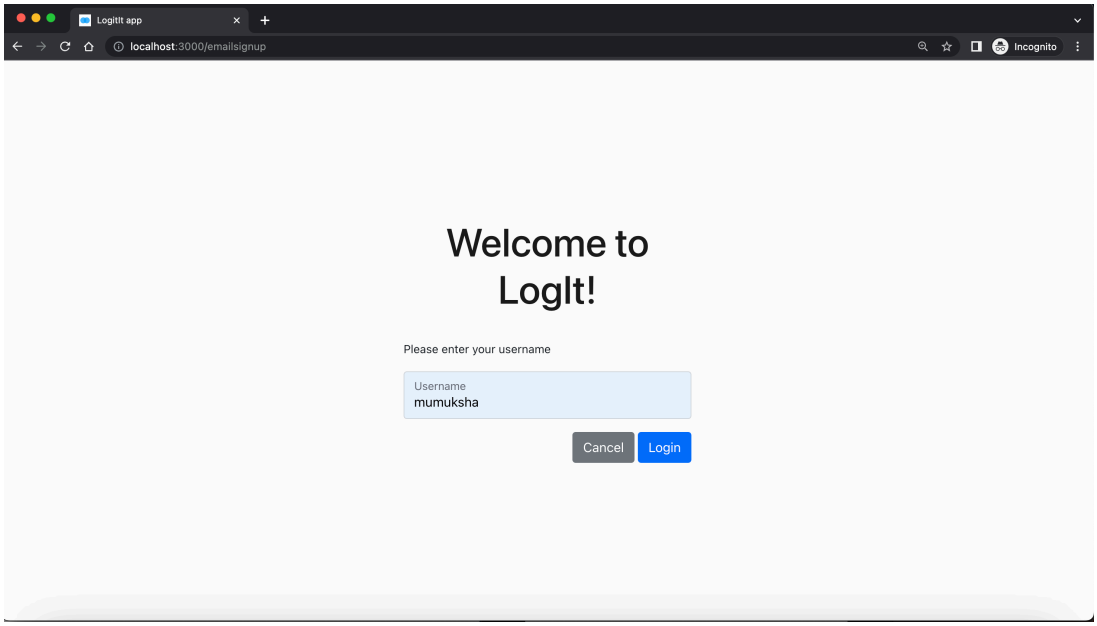


Fig 5.7 - Login Screen

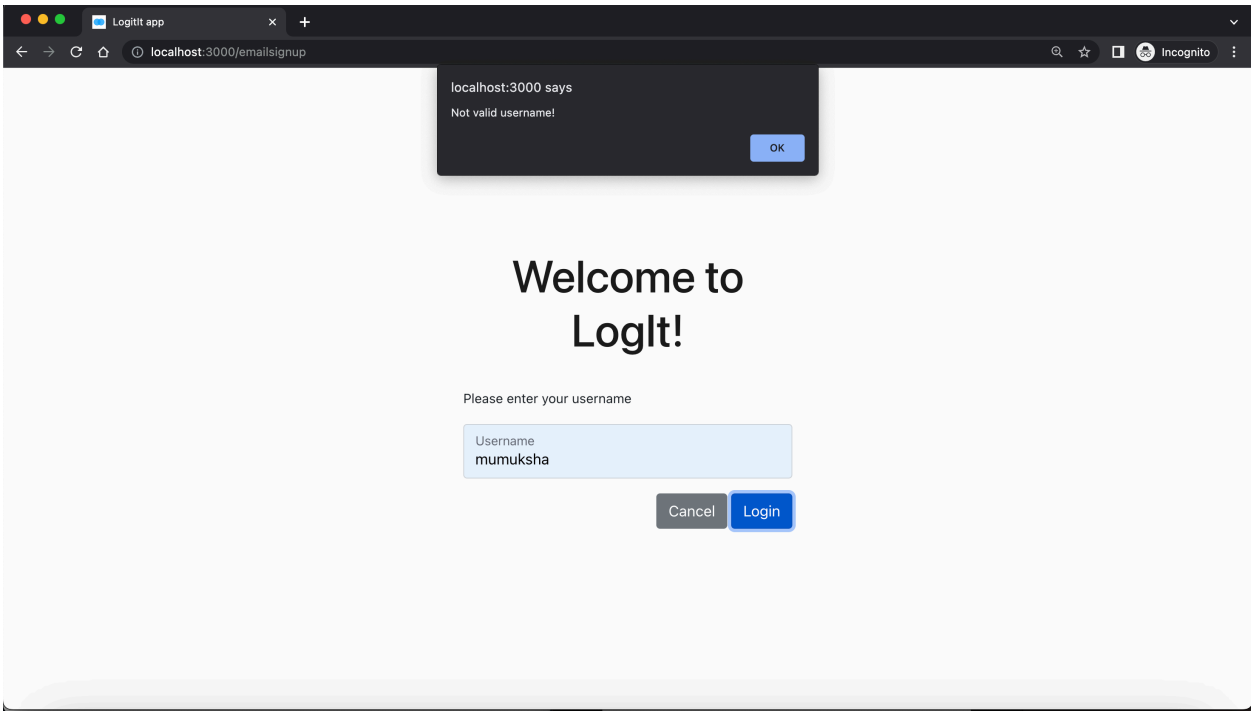


Fig 5.8 - Invalid username alert

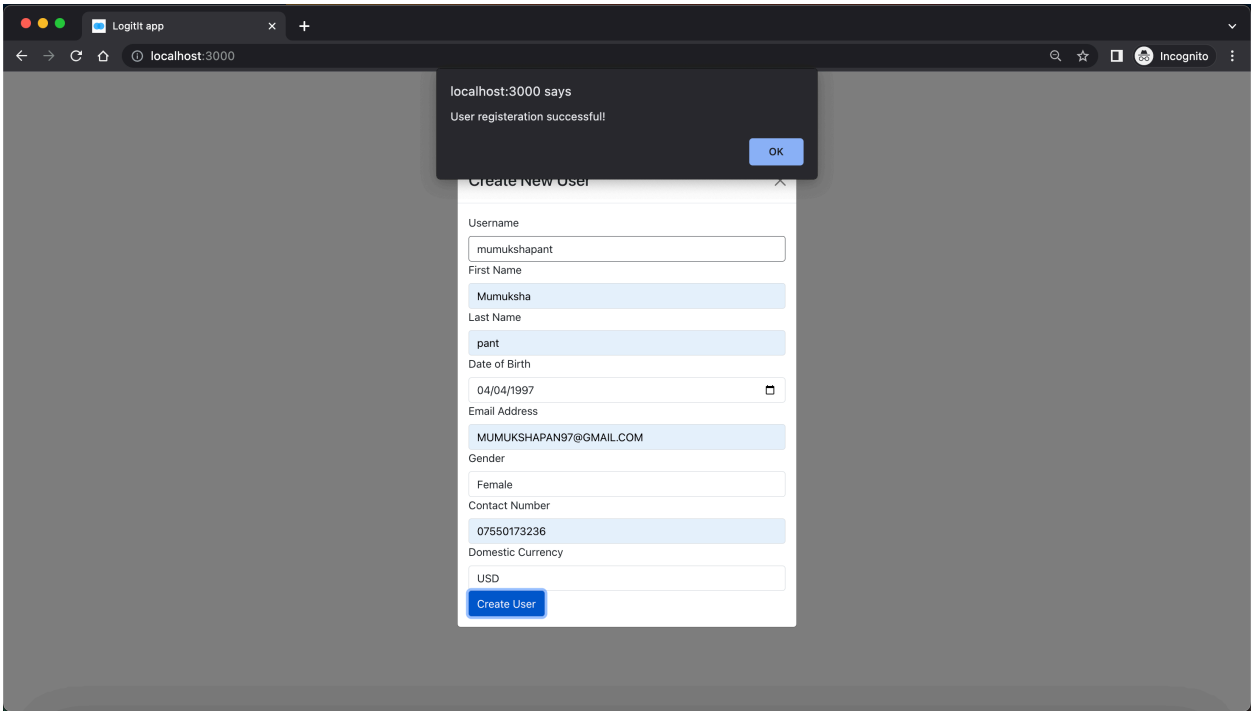
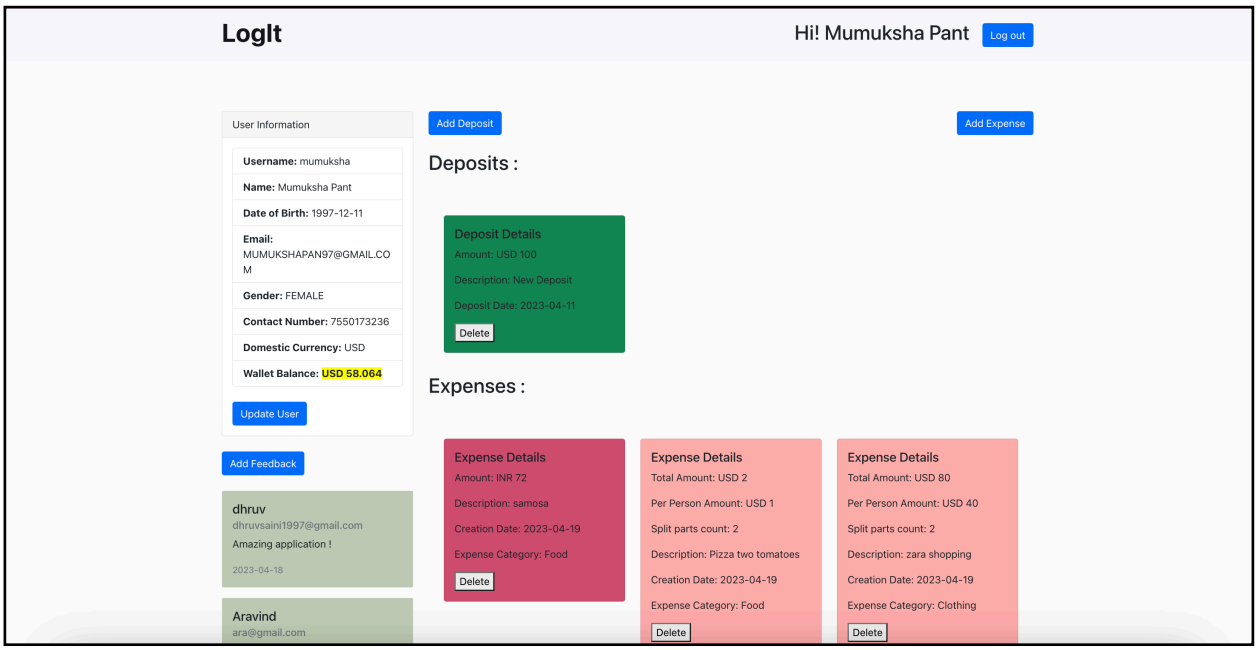


Fig 5.9 - Signup Screen - User registration successful



5.10 User Dashboard

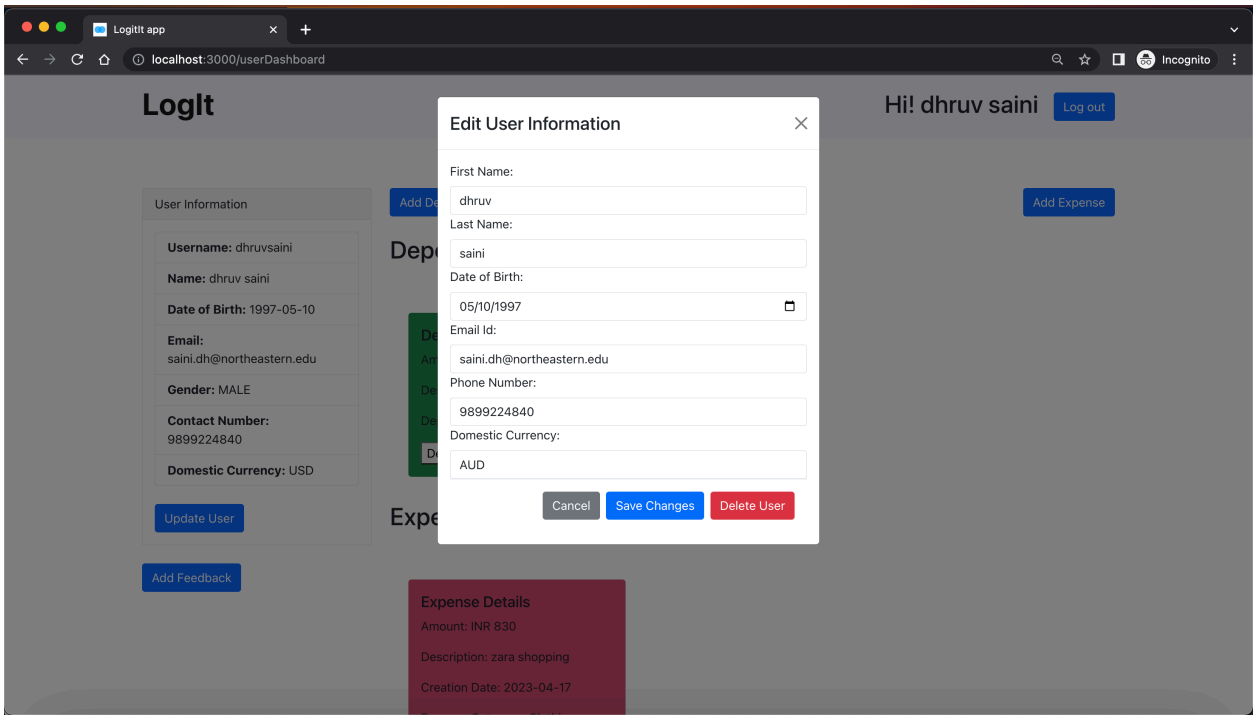


Fig 5.11 - Edit User Information - User Dashboard

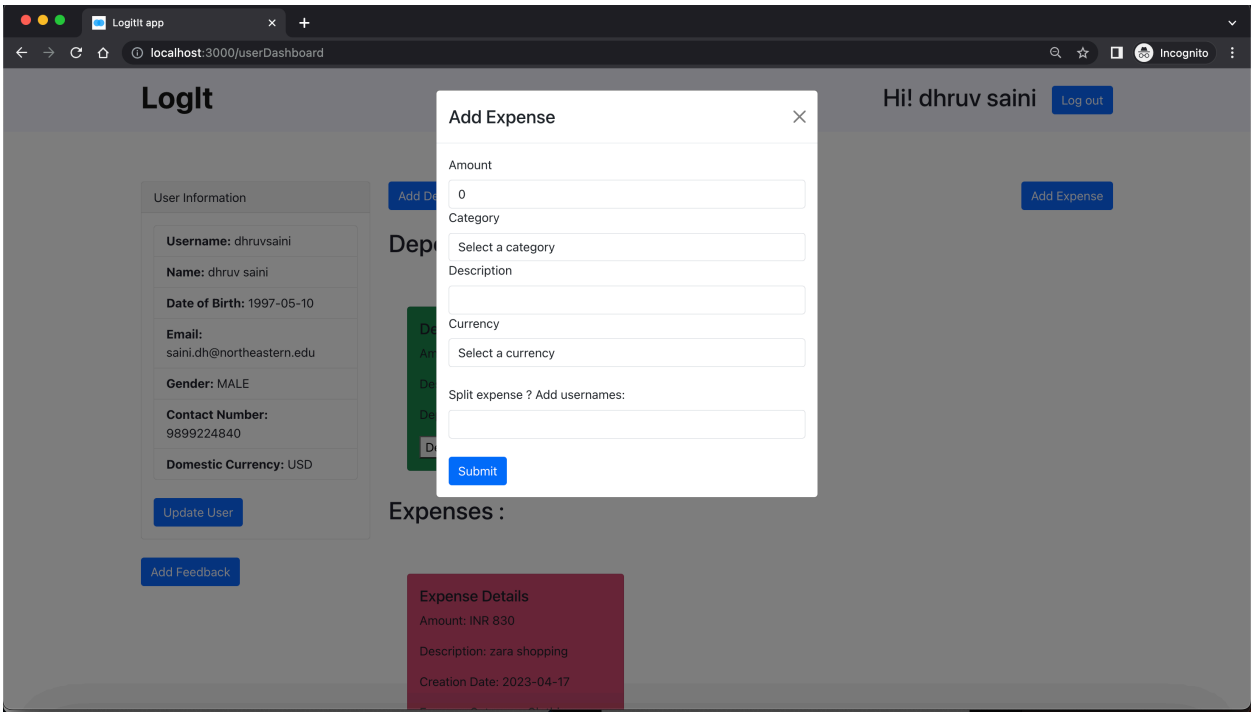


Fig 5.12 - Add Expense Screen - User Dashboard

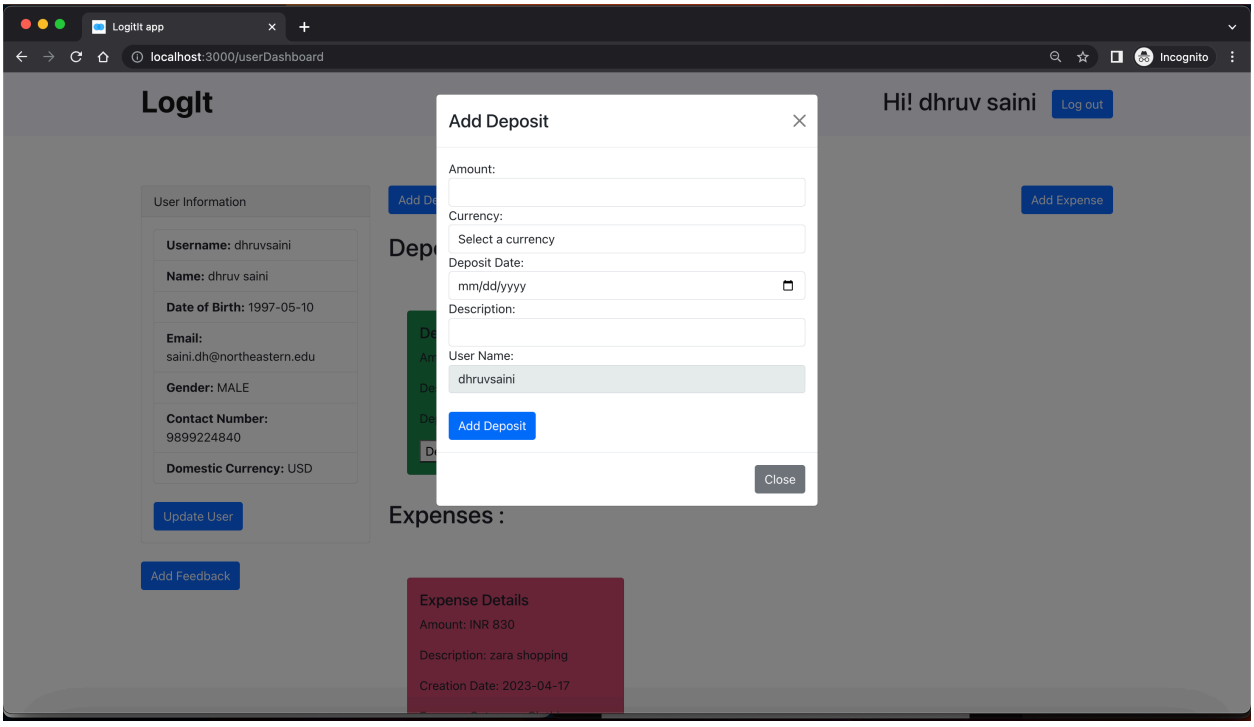


Fig 5.13 - Add Deposit Screen - User Dashboard

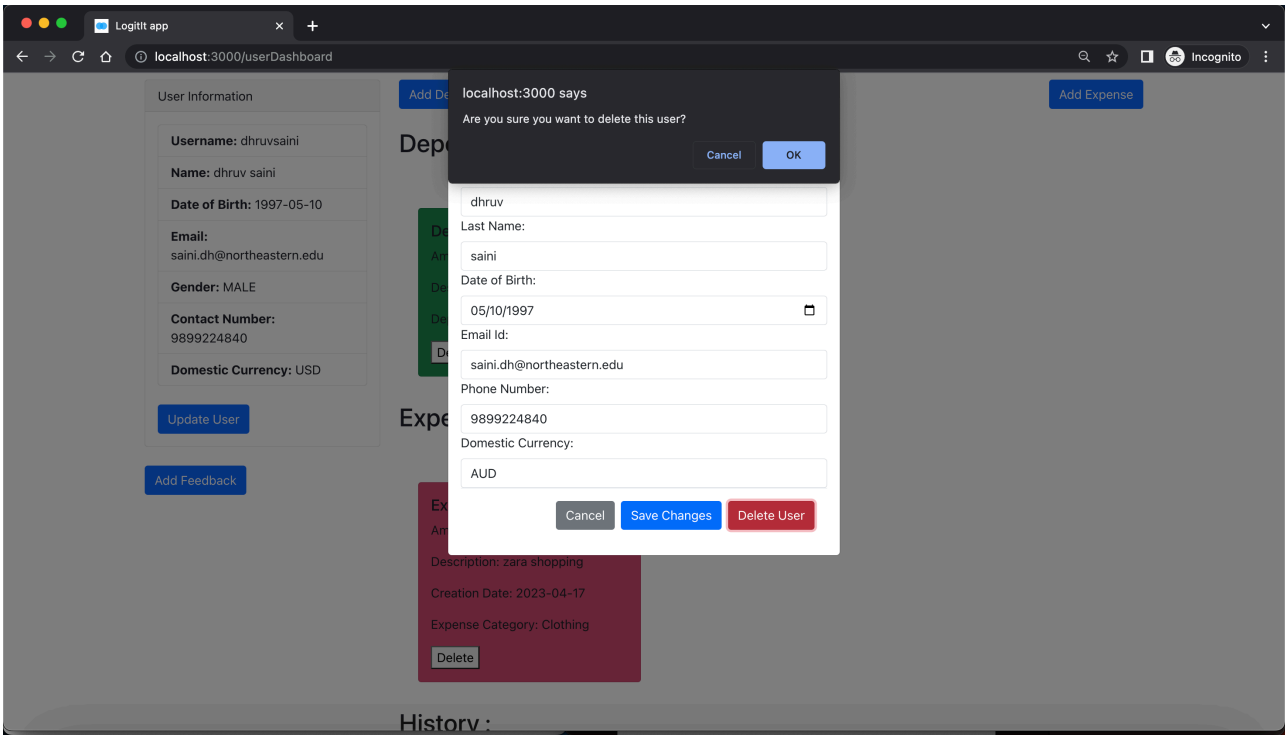


Fig 5.14 - Deleting Self User - User Dashboard

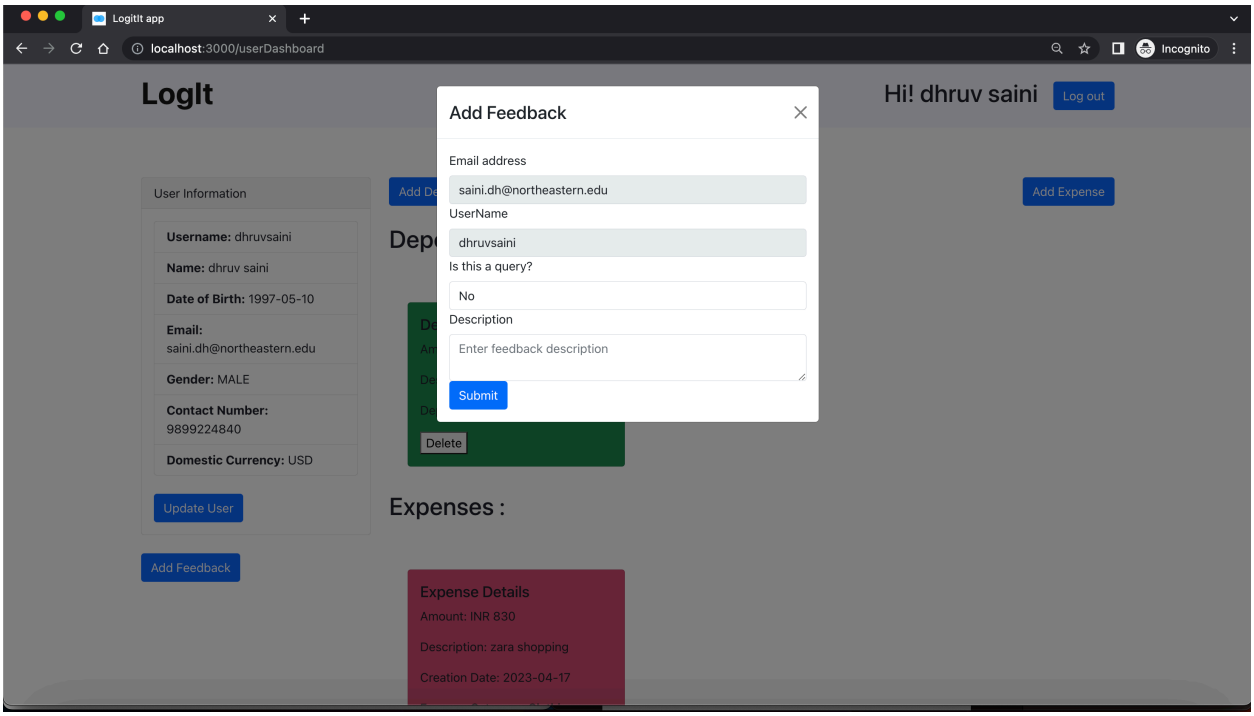


Fig 5.15 - Adding a Feedback / Query - User Dashboard

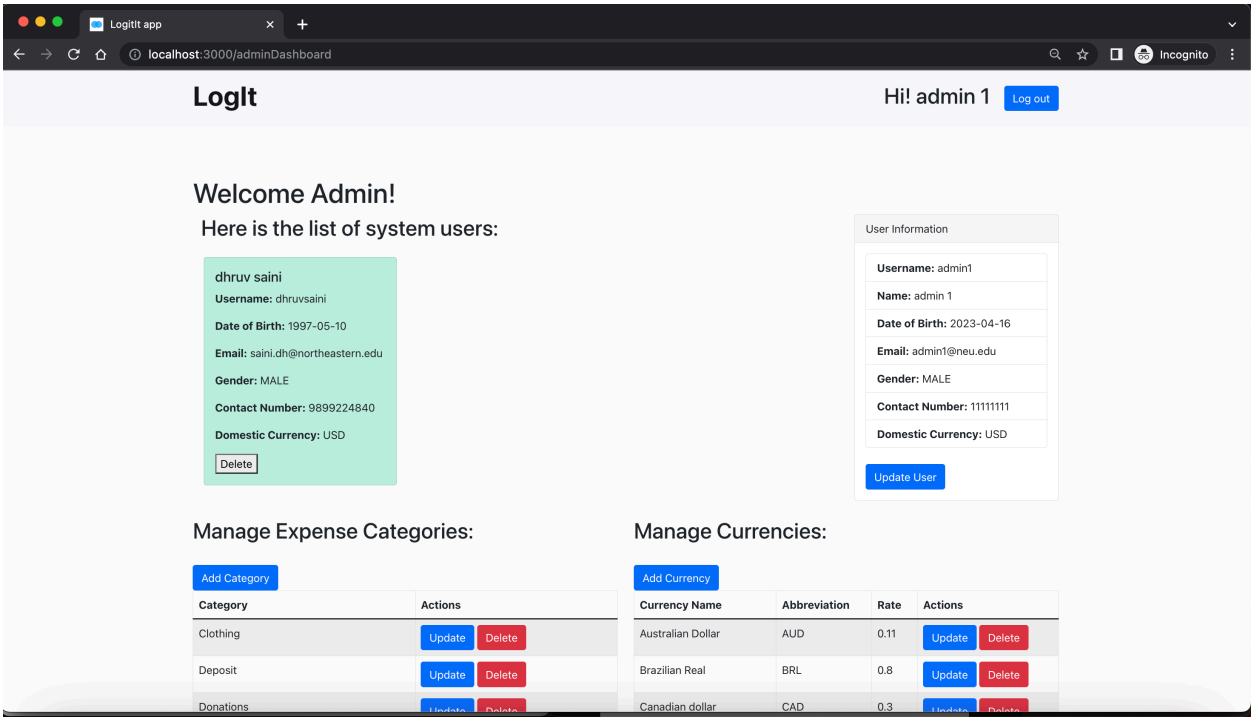


Fig 5.16 - Admin Dashboard

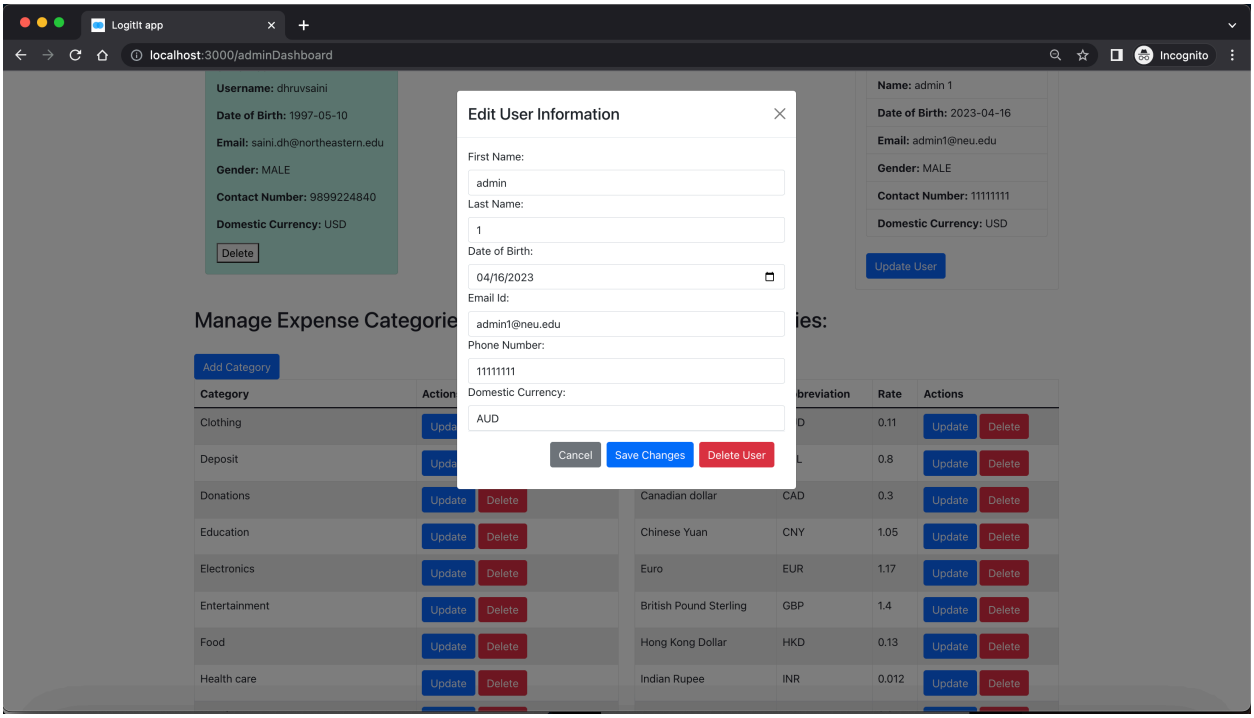


Fig 5.17 - Edit User Information - Admin Dashboard

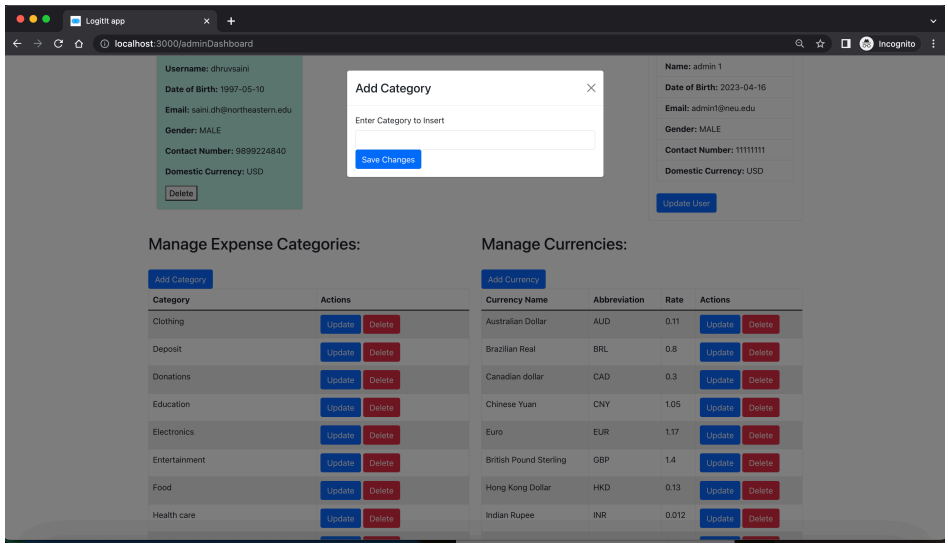


Fig 5.18 - Add Category - Admin Dashboard

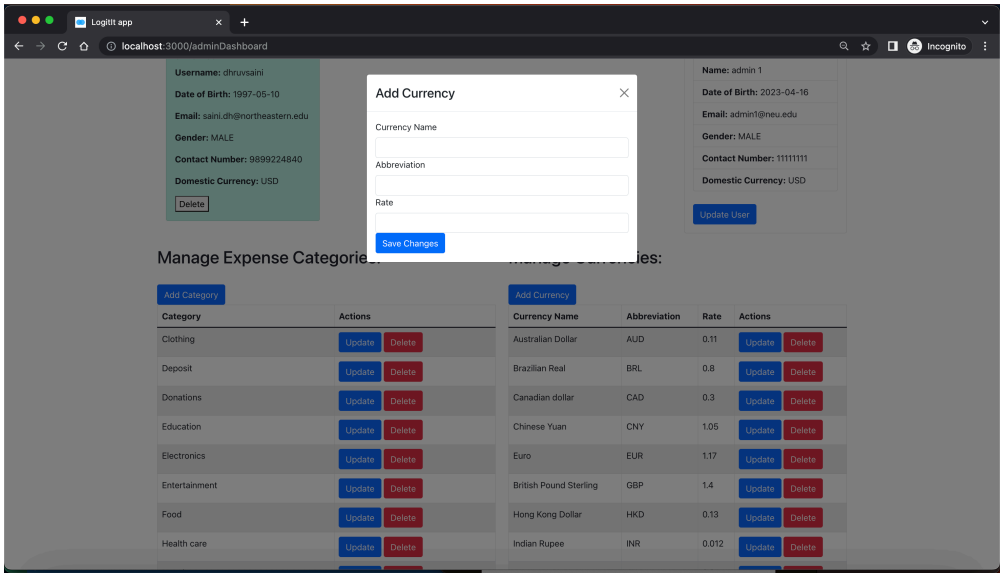


Fig 5.19 - Add Currency - Admin Dashboard

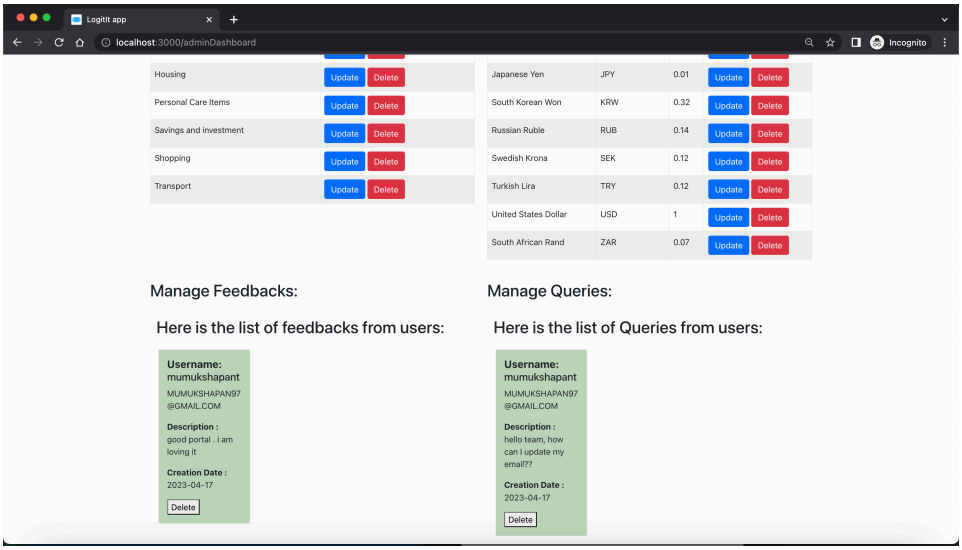


Fig 5.20 - Deleting Feedbacks / Queries - Admin Dashboard

CHAPTER 6

CONCLUSION

Logitt is a one stop expense management system designed to help users achieve their financial goals with ease. By categorizing expenses into different categories, *Logitt* empowers users to differentiate between their "needs" and "wants", providing them with insights into their spending patterns. Whether you're looking to save money or invest your money for the future, *Logitt* is the perfect tool to help you manage your finances.

CHAPTER 7

FUTURE ENHANCEMENTS

Future enhancements and upgrades to this project will include the following :

- Better User Interface with more enhanced capabilities .
- Visualization in the form of pie charts and bar graphs for the users to see a visual representation of how their expenses are distributed across different categories
- Splitting expenses among different users with different ratios in expense amount.
- Make the application compatible for mobile window as well.

CHAPTER 8

LESSONS LEARNT

Planning and Time Management :

Having to manage this project with other coursework ,assignments and exams was difficult initially. But with deadlines coming closer, we drafted a planner and assigned our time to this project . We set up our deadlines and most importantly created a routine for our work. This made sure our deliverables did not suffer.

Technical Expertise:

Starting to code from scratch was a daunting task, we had to go through several resources including as YouTube videos and various blogs to understand how to start with an MVC structure in Java.

While working on this project, we gathered technical expertise on using Spring boot , Stored Procedures and React JS . It was challenging to create API using JDBC at first, but as we gained momentum we started enjoying the process.

Data Modeling :

Data Modeling and creating a Conceptual design entirely by ourselves was challenging.

Nevertheless, we enjoyed the process and were able to overcome the challenge and produce a functional project.

CHAPTER 9

REFERENCES

[1] https://www.studocu.com/in/document/visvesvaraya-technological-university/database-management-systems/2dbms-project-report-template/10965678

[2] https://www.javatpoint.com/spring-boot-jdbc

[3] https://javainspires.blogspot.com/2018/05/spring-boot-mysql-jdbc-template-exmaple.html

[4]

[5] https://react.dev/learn
[6] https://github.com/gkamal/jdbc-template-one-to-many/tree/master/src/main/java/com/tenxperts/spring/jdbc
[7] https://www.javatpoint.com/spring-boot-tutorial