



Assessment Submission Form

Student Number (If this is group work, please include the student numbers of all group participants)	GH1039032 Yatik Bhadreshbhai Anghan GH1037834 Ayush Jashvantbhai Pandav
Assessment Title	Bank Management System
Module Code	M604A
Module Title	Advanced Programming
Module Tutor	Dr. Mazhar Hameed
Date Submitted	27-March-2025
Github	https://github.com/yatikanghan/Banking-Management-Application
Youtube	https://youtu.be/YypJbHAWmcQ

Declaration of Authorship

I declare that all material in this assessment is my own work except where there is clear acknowledgement and appropriate reference to the work of others.

I fully understand that the unacknowledged inclusion of another person's writings or ideas or works in this work may be considered plagiarism and that, should a formal investigation process confirms the allegation, I would be subject to the penalties associated with plagiarism, as per GISMA Business School, University of Applied Sciences' regulations for academic misconduct.

Signed...Yatik.....Ayush..... Date ...27/03/2025.....

1. ABSTRACT	1
2. INTRODUCTION & MOTIVATION	2
3. SYSTEM REQUIREMENTS	3
3.1 HARDWARE REQUIREMENTS.....	3
3.2 SOFTWARE REQUIREMENTS.....	4
4. SYSTEM ARCHITECTURE.....	6
4.1 HIGH-LEVEL SYSTEM ARCHITECTURE.....	6
4.2 KEY COMPONENTS OF THE SYSTEM	7
5. DATABASE DESIGN	8
5.1 DATABASE OVERVIEW	8
5.2 ER DIAGRAM.....	9
6. BACKEND DEVELOPMENT.....	11
6.1 OVERVIEW OF BACKEND ARCHITECTURE.....	11
6.2 TECHNOLOGY	11
6.3 API DEVELOPMENT	11
7. FRONTEND DEVELOPMENT	16
7.1 TECHNOLOGY	16
7.2 KEY UI FEATURES.....	16
7.3 SAMPLE CODE(LOGIN PAGE).....	16
	17
8. SECURITY MEASURES.....	17
8.1 AUTHENTICATION & AUTHORIZATION.....	18
8.2 ROLE-BASED ACCESS CONTROL.....	18
8.3 PROTECTION AGAINST COMMON THREATS	18
9. CHALLENGES AND SOLUTIONS.....	19
9.1 BACKEND DEVELOP CHALLENGES.....	20
9.2 DATABASE CHALLENGES.....	20
9.3 FRONTEND & INTEGRATION CHALLENGES	21
10. RESULT & DEMONSTRATION	22
10.1 FUNCTIONAL OUTCOMES	23
10.2 SCREENSHOTS & DEMONSTRATION.....	23
11. CONCLUSION & FUTURE	44
11.1 CONCLUSION.....	44
11.2 FUTURE ENHANCEMENTS	44
12. REFERENCES	46

1. Abstract

This project showcases a BANK MANAGEMENT SYSTEM build with Spring boot for back end logics and HTML and CSS for the front end visualization. The system gives a initial banking functionality like account creation, transaction processing, balance enquiry, customer support and user authentication. It also ensures that secure and efficient data management through an SQL database.

This project is designed with a RESTful API architecture and system gives a facilities to seamless communication between front end and back end with a smooth user interface. This project follow a modular, scalable and secure system with using Springboot's API. The data base is designed using SQL and ensuring data integrity and consistency.

This report provides a detail insight into the system architecture, database design, backend implementation, security measures and testing methods used in this project. In extras, we will give a demonstration of this project and discusses the challenges that we haved face during development, solutions implemented and potential enhancements. The banking management system serves as a foundational model for modern applications. With the back end and user centric design.

2. Introduction & Motivation

In digital world, Banking management system plays a vital role in financial transactions, personal financing management, and business operations. Although, many old banking systems still face some challenges like inefficiency, security and less access to the most of features. To over come this issues, we have developed a BANK MANAGEMENT SYSTEM using Spring boot for backend logic and HTML,CSS for frontend.

The real reason behind developing this project stems from the need for a secure, efficient and user friendly system that gives smooth transactions, account management and customer interactions. This project provides a structured, databased-given to handling operation, confirming that users can do initial task such as account creation, transaction and balance enquiry. Customer also can deposit money as a fixed deposit and gain interest on it. These are essential task which can user access and use seamlessly.

By using Spring boot application for backend operation, the management system ensures scalability, hardiness and high performance, during this use of the HTML and CSS for frontend gives a simple yet effective user interface. This project aligns with modern requirement and serves as a foundational model for developing more advanced system in the nearest future.

In more additional, this project is as an opportunity to give a boost to our knowledge of JAVA development, database connectivity, and as well as backend frameworks. During this applying read-world programming. It also fulfiil the requirements of our M604 Advanced Programming course assessment, demonstrats ability to design and implement a web app using technology.

3. System Requirements

The development and implementation of the Bank Management System requires both components of hardware and software to work smoothly and functionally, securely and scalability. This section outlines the essential hardware specifications, software tools, development environment need for implementations.

3.1 Hardware Requirements

To run Bank Management System easily, the following hardware are required:

3.1.1 Processor

Minimum requirements: Intel core i5(10 Gen or later) / AMD Ryzen 5.

Recommended : Intel Core i7 / AMD Ryzen 7.

3.1.2 Memory(RAM)

Minimum requirements: 8GB (Enough for basic development).

Recommended : 16GB or more (Improved Performance).

3.1.3 Storage

Minimum requirements: 20 GB free disk space.

Recommended : SSD with atleast 256GB for faster Read/Write.

3.1.4 OS(Operating System)

- Windows 10/11(Recommend for easy development with tools like IntelliJ IDEA)
- macOS(For those who use MacBook)
- Linux(Ubuntu 20.04 or later)

3.2 Software Requirements

The development of the BANK MANAGEMENT SYSTEM involves technologies to ensure a robust, scalable and secure banking system.

3.2.1 Backend Development

The backend is built using Spring Boot, a used java framework for building Restful API.

- Programming Language: Java (JDK 17 or later)
- Framework: Spring boot(Spring MVC, Spring data JPA)
- Build tool: Maven

3.2.2 Database Management

A relational database management system is used to store banking transactions, details and account records securely.

- Database Engine: MySQL
- Database Connector: MySQL Connect for JAVA.
- Schema Management: Hibernate ORM

3.2.3 Frontend Development

The designed with HTML and CSS to create a simple user friendly interface.

- HTML5
- CSS3

3.2.4 Development Environment and Tools

- IDE : IntelliJ IDEA
- Version Control : Git & GitHub
- API testing tool: Postman
- Database Management tool: MySQL Workbench

4. System Architecture

The project has three-tier architecture which are application's fundamental structure. The three layer include:

1. Presentation Layer(Frontend)
2. Business Layer(Backend)
3. Data Layer(Database)

4.1 High-Level System Architecture

4.1.1 Presentation Layer(Frontend)

- Develop using HTML and CSS to provide a user interface.
- Display account details, transactions and customer support.
- Communicate with the backend via RESTful APIs.

4.1.2 Business Logic Layer(Backend)

- Develop use Spring Boot to handle APIs requests and Logic behind it.
- Implement CRUD operations For managing bank system and data.

4.1.3 Data Layer(Database)

- A RDBMS for storing structured banking data.
- Ensures data integrity with compliance.
- Uses Spring data JPA (Hibernate) to interact with data and database.

4.2 Key Components of the System

4.2.1 API Endpoints

- The backend exposes several RESTful API endpoints to operate banking operations.

HTTP Method	Endpoints	Functionality
GET	/users/{id}	Fetch user detail
POST	/account/create	Create a account
POST	/transaction/deposit	Deposit money
POST	/transaction/withdraw	Withdraw money
GET	/transaction/history	Get trans. history

4.2.2 Security Measures

- User authentication: Implemented using JWT for secure API access.
- Role-based access control: Different user roles like admin, customer have specific access.
- Data encryption : Uses hashing techniques for store sensitive data.

5. Database Design

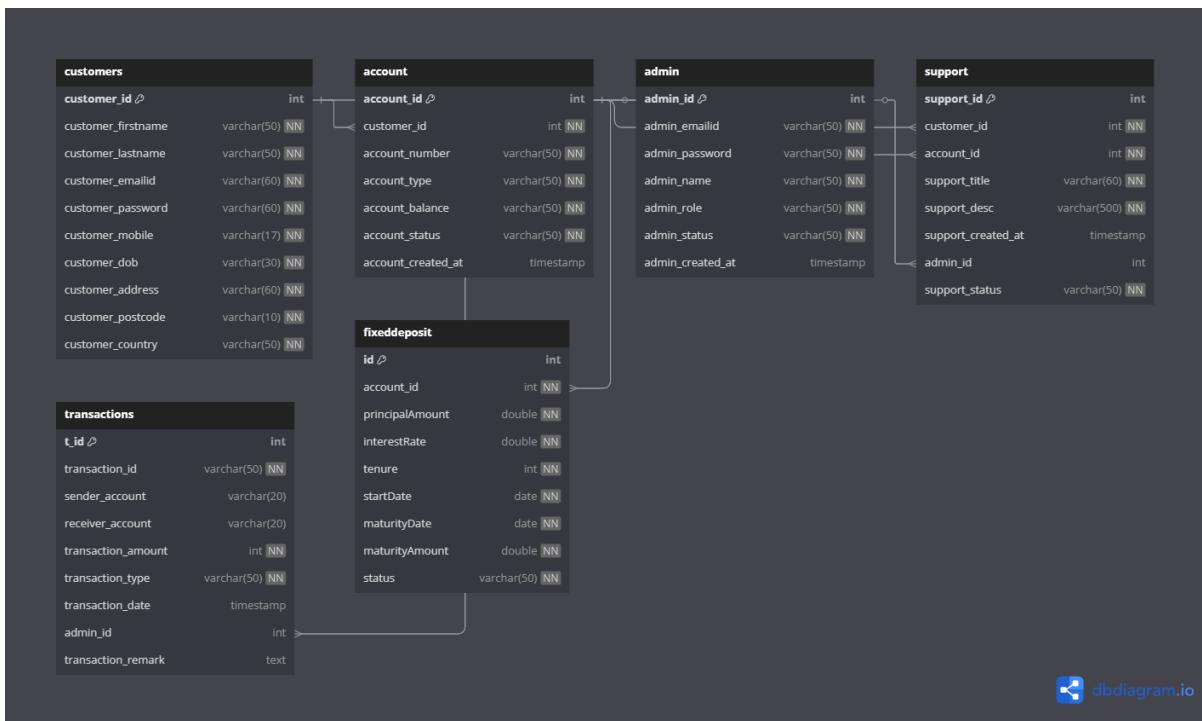
Bank Management System also relies on a well-structured relational database(RDBMS) to store and manage the customer's details, accounts, transactions, and many other operating systems. The system is designed using MySQL with proper constraints and efficient query processing.

5.1 Database Overview

In the database below down tables are essential tables for banking management system.

- Customers table: Store customer details and credential.
- Accounts table: Manages customer accounts, balance sheet and status.
- Admin table: Keep data of administrator credentials and role.
- Support table: Handle customer support requests.
- Transaction table: Log all financial transactions between accounts.
- Fixed deposit table: Manage fixed deposit accounts and interest rates and other details.

5.2 ER Diagram



5.2.1 Customer and Accounts(One to Many)

- One User can have multiple bank accounts and each account belongs to only one customer.
- Customer_id in the accounts table is a foreign key reference from customer table.

5.2.2 Admin and Transaction(One to Many)

- Admin can authorize multiple transactions, each transaction is associated with only one admin.
- Admin_id in the transaction table is a foreign key refer from admin table.

5.2.3 Accounts and Transactions(One to Many)

- Accounts can have multiple transactions and each transaction links with specific account.
- Sender_account and receiver_account in the transaction table store account_number from accounts table.

5.2.4 Customer and support(One to Many)

- A customer can raise multiple ticket for support, each request linked with single customer.
- Customer_id in support table referering from customers table.

5.2.5 Admin and support(One to Many)

- An admin can handle multiple tickets, each ticket is assign to one admin.
- Admin_id in support table is a foreign key from admin table.

5.2.6 Accounts and Fixed deposit(One to Many)

- An account can have multiple fixed deposits, each deposit is linked with one account.
- Account_id in fixed deposit table is a foreign key from the account table.

6. Backend Development

The backend development of the system is developed using Spring Boot, a java framework that simply backend development by provide RESTful APIs, database interactions. This section details the fundamentally functionalities, API design, security implementation and database integration in backend system.

6.1 Overview of backend architecture

The backend architecture is followed the key components includes:

1. Controller Layer: Handles requests and API endpoints.
2. Service Layer: Contain logical and interact with database.
3. Repository Layer: Manages database using Spring data JPA.
4. Database Layer: MySQL for secure data storage.

6.2 Technology

The backend is developed using this following technologies:

- Programming Language: JAVA(JDK 17 or later)
- Framework: Spring Boot
- Database: MySQL
- Build tool: Maven
- Security: Spring Security with JWT authentication

6.3 API Development

The backend exposes RESTful API to interact with the frontend. Below down are the API endpoints which are use for bank management system.

6.3.1 Customer Login API

HTTP Methods	Endpoint	Function
GET	/login	Customer Login Page
GET	/register	Customer Register Page
POST	/registeraccount	Customer Register new Account
POST	/customerlogin	Customer Logged In
GET	/customerclosedpage	If account closed and still try to login customer

6.3.2 Customer General API

HTTP Methods	Endpoint	Function
GET	/customerdashboard	Customer Dashboard
GET	/profile	Customer Profile
POST	/customerprofilesave	Customer profile save

6.3.3 Customer Transfer API

HTTP Methods	Endpoint	Function
GET	/customertrasfer	Customer Transfer Page
POST	/customertransfermoney	Customer Transfer Money

6.3.4 Customer Support API

HTTP Methods	Endpoint	Function
GET	/customersupport	Customer support
GET	/customeraddsupport	Customer Add Support Form
POST	/customeraddsupportinsert	Customer Add Support
GET	/customersupport/{id}	Customer Support detail

6.3.5 Admin Login API

HTTP Methods	Endpoint	Function
GET	/adminlogin	Admin Login Page
POST	/adminlogin	Admin Loged In
GET	/admindashboard	Admin Dashboard
GET	/adminlogout	Admin Logout

6.3.6 Admin API

HTTP Methods	Endpoint	Function
GET	/adminprofile	Admin Profile
POST	/adminprofilesave	Admin Update Data

6.3.7 Admin Management API

HTTP Methods	Endpoint	Function
POST	/admindetailsave	Admin Detail save
POST	/adminnewadminsave	Add New Admin
GET	/adminnewaccount	Admin List Admin
GET	/adminnewaccount/{id}	Admin Detail Update Form

6.3.8 Transaction API

HTTP Methods	Endpoint	Function
GET	/admindebit	Deposit Money Page.
POST	/admindebitamount	Deposit Amount
GET	/adminwithdraw	Withdraw Money Page
POST	/adminwithdrawamount	Withdraw Amount
GET	/admintransfer	Transfer Money Page
POST	/admintrasfermoney	Admin Transfer Amount

6.3.9 Fixed Deposit API

HTTP Methods	Endpoint	Function
GET	/customerfixeddeposit	Customer Fixed Deposit
GET	/adminfixdeposit	Admin Fixed Deposit
GET	/adminfixdeposit/{id}	Admin Fixed Deposit detail
POST	/adminfdclosed	Admin Closed Fixed Deposit

6.3.10 Admin Login API

HTTP Methods	Endpoint	Function
POST	/adminlogin	Admin Loged In
GET	/deactiveadmin	If Admin is deactive show you are deactive
GET	/accessdeniedpage	IF Admin has not permission so show access denied page

6.3.11 Admin Manage Accounts API

HTTP Methods	Endpoint	Function
POST	/adminacconfirm	Admin account detail save
POST	/adminnewacconfirm	Admin new account verify
GET	/adminmanageaccount	Admin Manage Account
GET	/adminmanageaccount/{id}	Admin Manage account detail

6.3.12 Admin Todo List API

HTTP Methods	Endpoint	Function
GET	/admintodolist	Admin Todo List show
GET	/addtask	Admin Todo List Form
POST	/addtodobtn	Admin Todo Task Add
GET	/deletetodo/{id}	Admin Todo Task Delete
GET	/updatetodo/{id}	Admin Update Task Form
POST	/updatetodobtn	Admin Update Task Update

7. Frontend Development

The frontend of this project is developed with normal HTML and CSS and JavaScript to give a dynamic, simple, clean and user friendly interface of web application to take a seamless data exchange and real time updates.

7.1 Technology

- HTML5 for structure for web pages.
- CSS3 for styles and design.
- JavaScript for handles user interactions and API requests.

7.2 Key UI features

- User Authentication: Customer and admin can login securely.
- Dashboard: Display account details, balance, transaction history.
- Transaction handling: User can deposit, withdraw and transfer.
- Support Request: Customer can raise supports tickets.

7.3 Sample Code(Login page)

Here is the sample code for login page in html.

```
<html lang="en">
<body oncontextmenu="return false">
<section>
  <div class="loginpage">
    <div class="container-fluid">
      <div class="loginbox">
        <h3>Administration</h3>
        <h3>Login</h3><br>
        <form action="/adminlogin" method="post">
          <div class="form-group">
            <label for="adminemailid">Email address</label>
            <input type="email" class="form-control" id="adminemailid" name="adminemailid" placeholder="Email Address" required>
          </div>
          <div class="form-group">
            <label for="adminpassword">Password</label>
            <input type="password" class="form-control" id="adminpassword" name="adminpassword" placeholder="Password" required>
          </div>
          <br>
          <div class="loginbtn">
            <p th:text="${error}"></p><br>
            <button type="submit">Login</button>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>
</div>
</section>
</body>
</html>
```

8. Security Measures

Security play a vital role in any kind of application due to sensitive

nature of financial data, user credentials and transactions. This project follows a multi layer security approach to make sure data confidentiality and protect against unauthorized access or cyber threats..

8.1 Authentication & Authorization

8.1.1 User Authentication with JWT

The system uses JWT for authentication. When a user login in the system generates a JWT token, which sent in API requests to verify identity.

How JWT Works:

1. User login using email and password.
2. System validate credentials and generates JWT token.
3. The token is sent in authorization headers with API.
4. The backend validates the token before granting access.

8.2 Role-Based Access control

To restrict unauthorized access, the system uses Role-Based Access Control.

User Role	Permission
Customer	View account, transaction and raise requests
Admin	Manage accounts, approve transaction
Support	Handle customer queries, assist with disputes

The backend uses Spring Security to enforce roled-based access.

8.3 Protection against common threats

The system implements multiple security against common cyber threats:

Threat	Mitigation Strategy
--------	---------------------

SQL Injection	Uses prepared statement to prevent queries
Cross-Site Scripting (XSS)	Sanitize user inputs and prevents script injections
Cross-Site Request Forgery (CSRF)	CSRF tokens to validate user requests
Brute Force Attacks	Implement login attempt limits and CAPTCHA verification

9. Challenges and Solutions

During the development of the system, several technical and implementation challenges were identified. The challenges are across backend development, database management, security and frontend integration. Each issue was analyzed with a solution to work smoothly.

9.1 Backend Develop challenges

Challenge 1: Handling Complex Business Logic

Issue :

- During development of transactions, account management, and fixed deposit required handling complex business rules.
- Make sure transactions integrity was tough.

Solution :

- Used Spring Boot Service Layer to separate business logic from main controller.
- Implemented Spring Transaction to rollback database changes if an operation failed.

Challenge 2: Manage API performance

Issue :

- System need to handle large number of transaction easily.
- Querying large datasets caused lack of performance

Solution :

- Used pagination and sort in Spring Data JPA to optimization.

9.2 Database challenges

Challenge 3: Make data consistency and integrity

Issue :

- Banking operates require strict consistency to prevent issues.

Solution :

- Used ACID- compliant transactions with JPA to maintain.

- Applied foreign key constraints and referential integrity in SQL.

Challenge 4: Optimizing queries for faster performance

Issue :

- Queries on large tables were slow due to increasing data.

Solution :

- Used proper indexing on frequently queried columns.

9.3 Frontend & Integration Challenges

Challenge 5: Handle API communication errors

Issue :

- If the backend was down, frontend display errors.

Solution :

- Implemented error handle and load indicators in javascript.

Challenge 6: Ensure Frontend Responsiveness

Issue :

- The initial frontend design was not mobile friendly and lacked appeal.

Solution :

- Used CSS flex and bootstrap for responsiveness.

10. Result & Demonstration

This section gives an overview of outcome of Bank Management System, showcase its feature, performance and real-world usability. The system tested to confirm secure transactions, account management and seamless user interactions. Below down are the screenshots and demonstrations the application.

10.1 Functional Outcomes

The system meets the initial requirements of a scalable banking application.

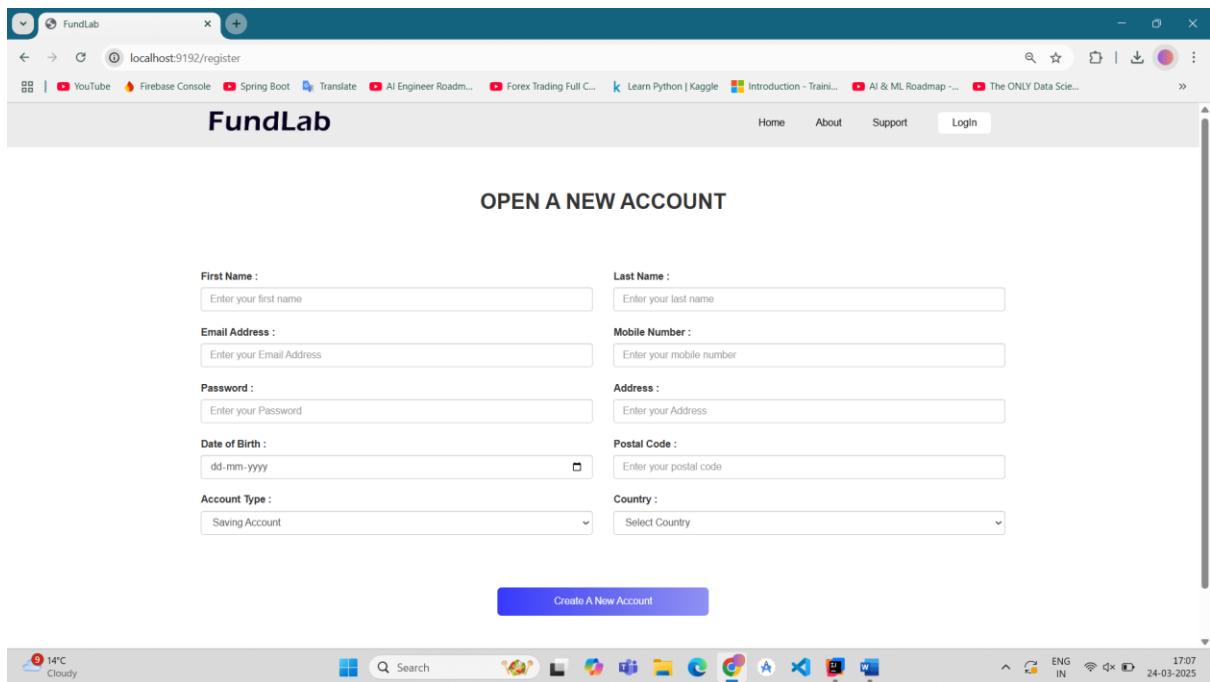
Feature	Outcome
User Authentication	Customers & Admin can securely login with JWT
Account Management	User can create account, view balance and many more
Transaction Processing	Deposit, withdraw & transfer are executed with data integrity
Fixed Deposit	User can create fixed deposit account
Security Measures	Password encryption, API, role-based access control implemented
Admin & Support System	Admin can approve transaction, manage user, and respond to support requests

10.2 Screenshots & Demonstration

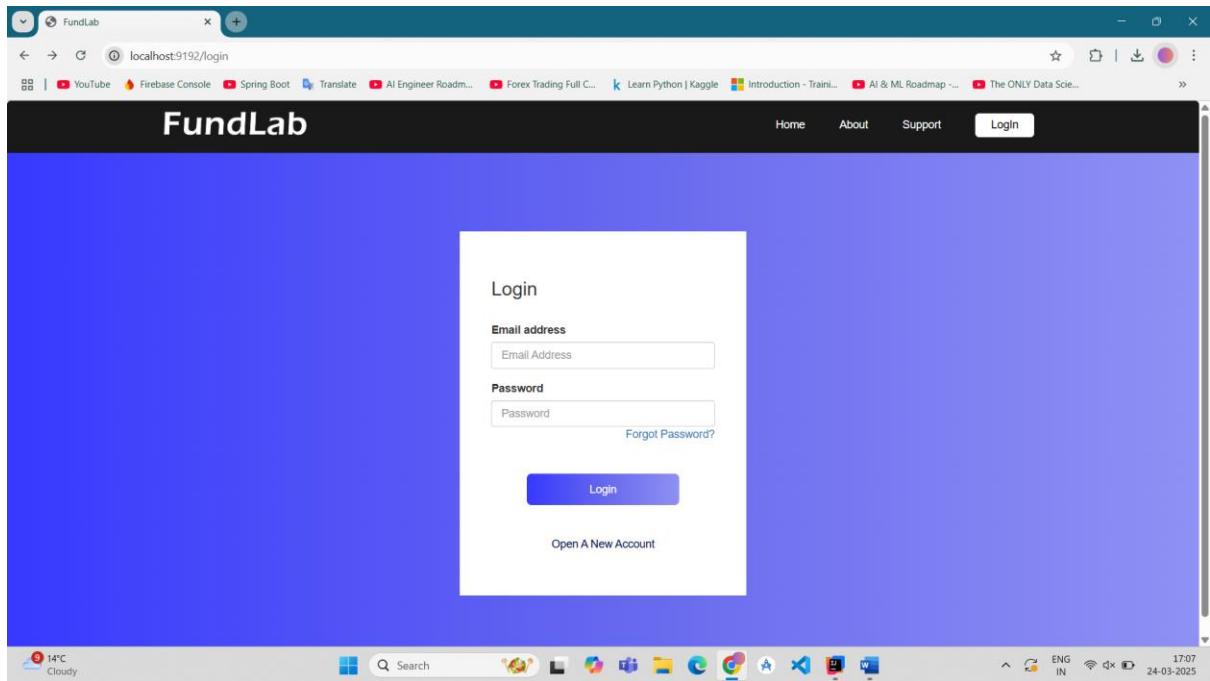
10.2.1 Customer Panel

- Customers can register & login in securely.

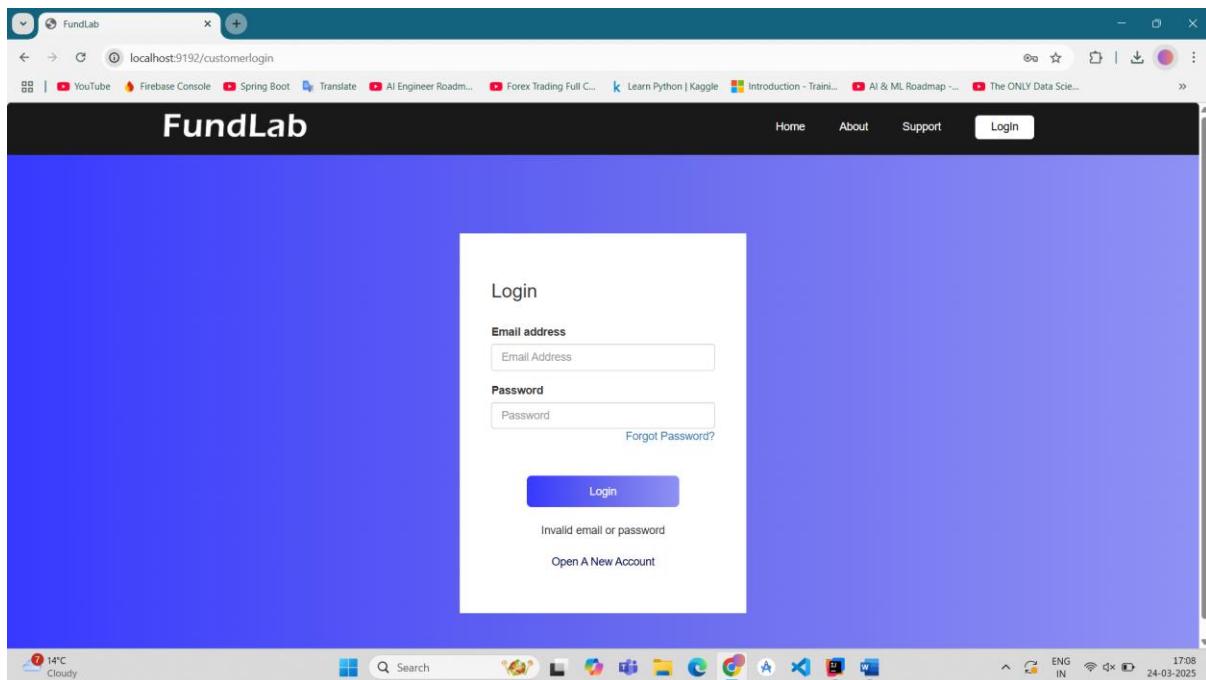
Screen shot : Registration Page



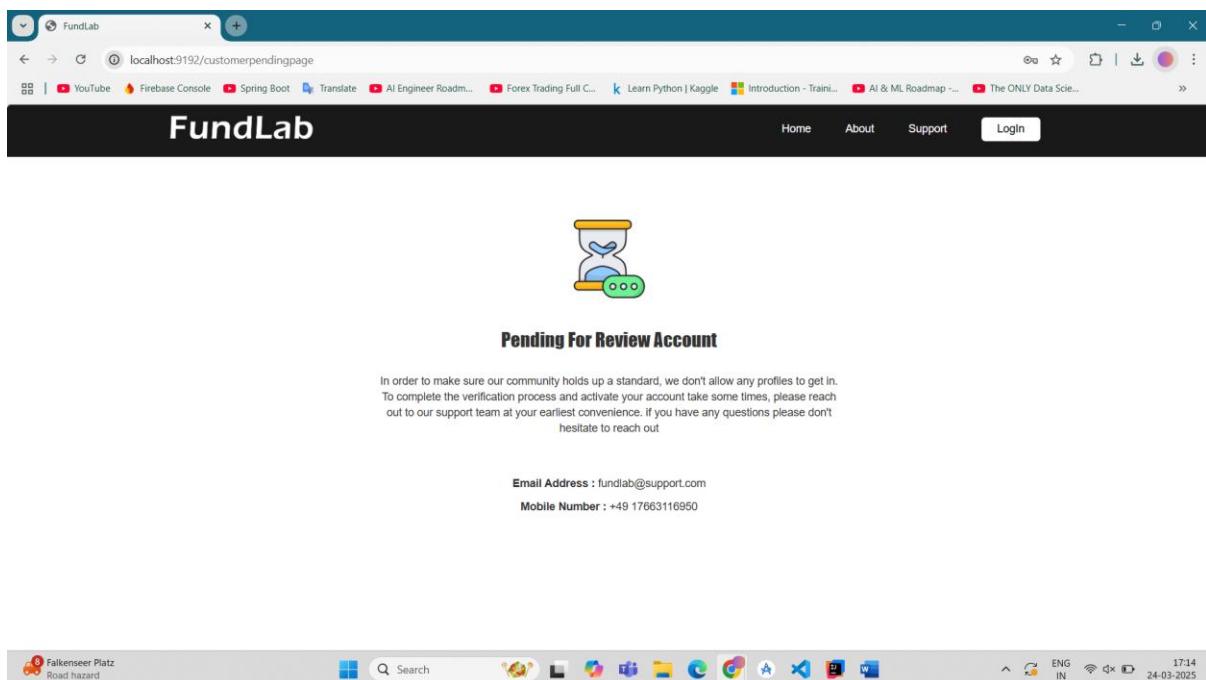
Customers Registration Page



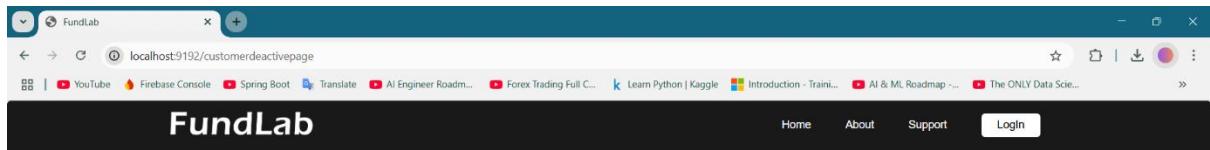
Customer Login Page



If Customer Login Failed



If Customer Logged in but not verified



Your account has been Deactivated

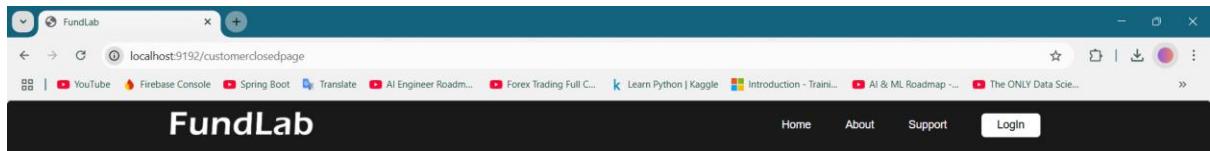
If you having some problem, contact our support team.
We have huge amount of customer request so you will receive response in 2 - 3 working days
out to our support team at your earliest convenience. If you have any questions please don't
hesitate to reach out.

Email Address : fundlab@support.com

Mobile Number : +49 17663116950



Customer Logged In but account has been deactivated



Your account has been closed

If you having some problem, contact our support team.
We have huge amount of customer request so you will receive response in 2 - 3 working days
out to our support team at your earliest convenience. If you have any questions please don't
hesitate to reach out.

Email Address : fundlab@support.com

Mobile Number : +49 17663116950



Customer Logged In but account has been closed

Welcome, Yatik Anghan

Transaction ID	Sender	Receiver	Amount	Date	Type	Remark
1741888146554	1000000001	NA	50	2025-03-13 18:49:06	Withdraw	cash
1741888156961	NA	1000000001	100	2025-03-13 18:49:16	Deposit	cash
1741888202288	1000000001	1000000002	50	2025-03-13 18:50:02	Transfer	charges
1741888222524	1000000002	1000000001	60	2025-03-13 18:50:22	Transfer	Transfer
1741888373907	NA	1000000001	150	2025-03-13 18:52:53	Deposit	cut

Customer Home Page

Transfer Money

Receiver Account Number:

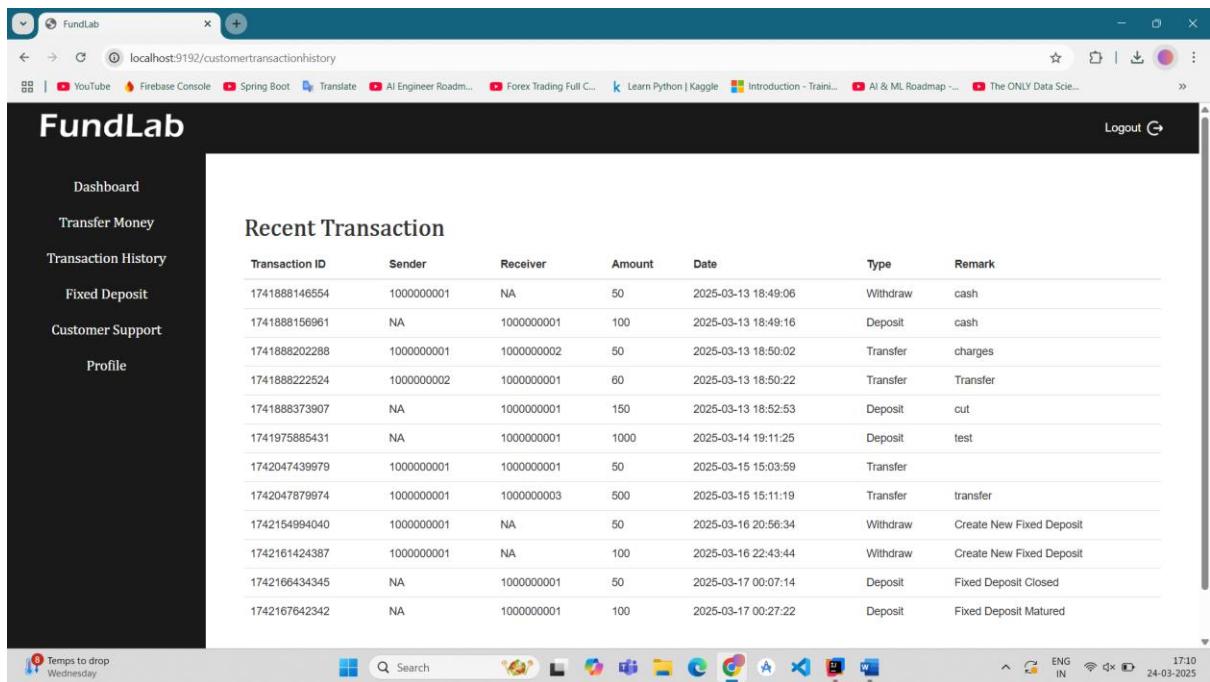
Re-enter Account Number:

Transfer Amount:

Remark:

Transfer Now

Customer Transfer Money Page

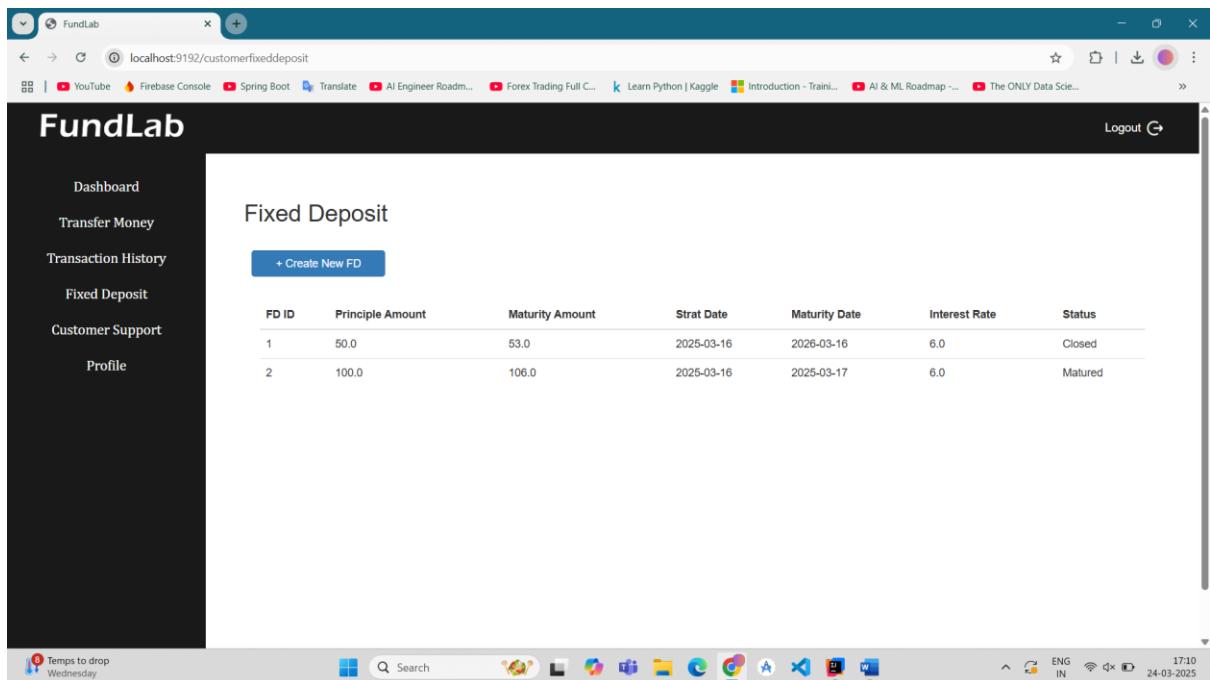


The screenshot shows a web browser window for 'FundLab' at localhost:9192/customertransactionhistory. The left sidebar has a dark theme with white text and icons. It includes links for Dashboard, Transfer Money, Transaction History, Fixed Deposit, Customer Support, and Profile. The main content area is titled 'Recent Transaction' and displays a table of transaction history. The table has columns: Transaction ID, Sender, Receiver, Amount, Date, Type, and Remark. The transactions listed are:

Transaction ID	Sender	Receiver	Amount	Date	Type	Remark
1741888146554	1000000001	NA	50	2025-03-13 18:49:06	Withdraw	cash
1741888156961	NA	1000000001	100	2025-03-13 18:49:16	Deposit	cash
1741888202288	1000000001	1000000002	50	2025-03-13 18:50:02	Transfer	charges
1741888222524	1000000002	1000000001	60	2025-03-13 18:50:22	Transfer	Transfer
1741888373907	NA	1000000001	150	2025-03-13 18:52:53	Deposit	cut
1741975885431	NA	1000000001	1000	2025-03-14 19:11:25	Deposit	test
1742047439979	1000000001	1000000001	50	2025-03-15 15:03:59	Transfer	
1742047879974	1000000001	1000000003	500	2025-03-15 15:11:19	Transfer	transfer
1742154994040	1000000001	NA	50	2025-03-16 20:56:34	Withdraw	Create New Fixed Deposit
1742161424387	1000000001	NA	100	2025-03-16 22:43:44	Withdraw	Create New Fixed Deposit
1742166434345	NA	1000000001	50	2025-03-17 00:07:14	Deposit	Fixed Deposit Closed
1742167642342	NA	1000000001	100	2025-03-17 00:27:22	Deposit	Fixed Deposit Matured

The status bar at the bottom shows system information: Temp to drop Wednesday, ENG IN, 17:10, 24-03-2025.

Customer Recent Transaction Page

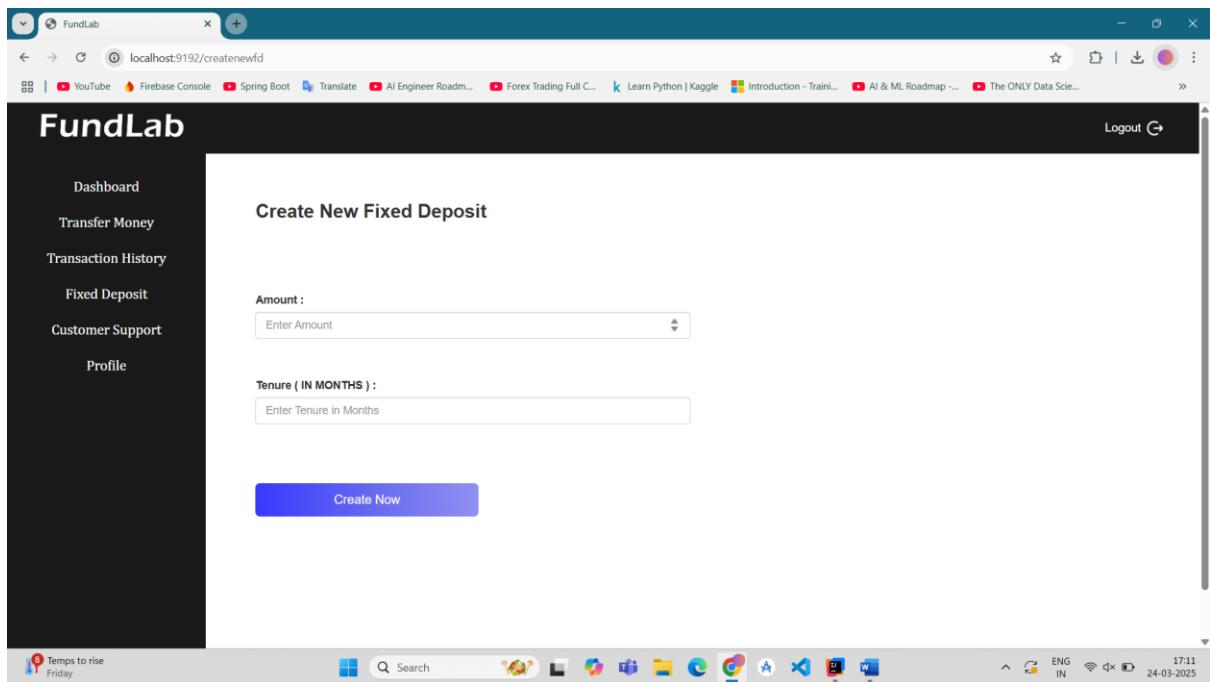


The screenshot shows a web browser window for 'FundLab' at localhost:9192/customerfixeddeposit. The left sidebar has a dark theme with white text and icons. It includes links for Dashboard, Transfer Money, Transaction History, Fixed Deposit, Customer Support, and Profile. The main content area is titled 'Fixed Deposit' and displays a table of fixed deposit details. The table has columns: FD ID, Principle Amount, Maturity Amount, Strat Date, Maturity Date, Interest Rate, and Status. The deposits listed are:

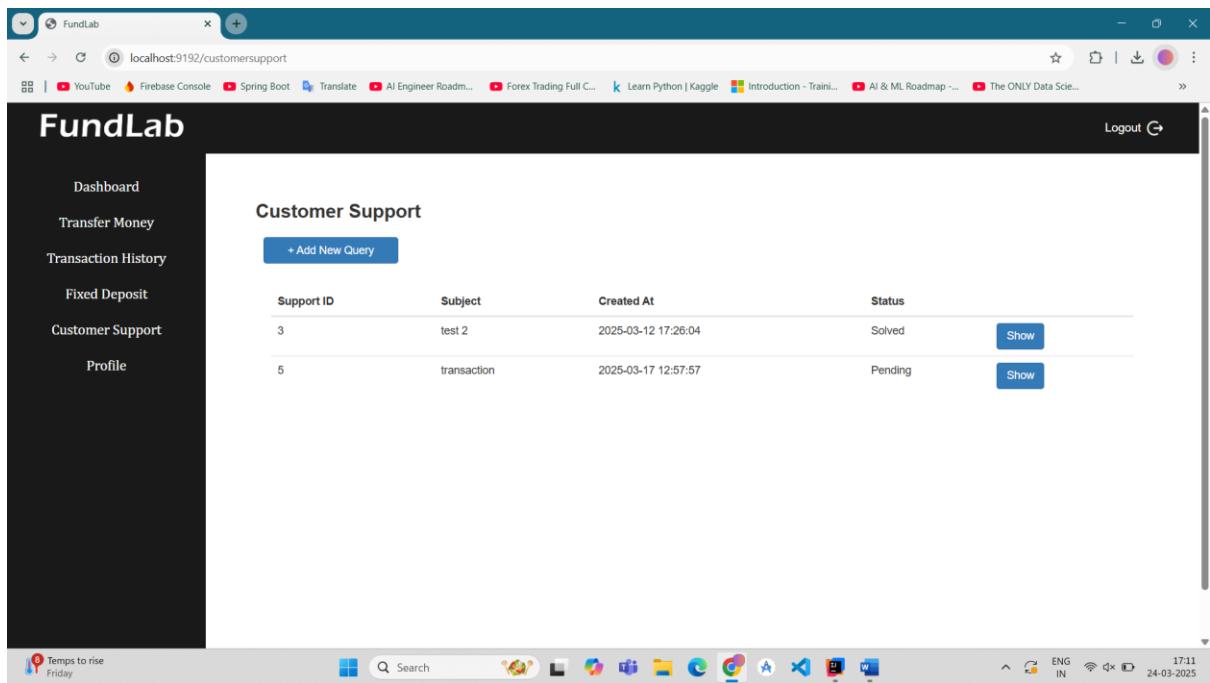
FD ID	Principle Amount	Maturity Amount	Strat Date	Maturity Date	Interest Rate	Status
1	50.0	53.0	2025-03-16	2026-03-16	6.0	Closed
2	100.0	106.0	2025-03-16	2025-03-17	6.0	Matured

The status bar at the bottom shows system information: Temp to drop Wednesday, ENG IN, 17:10, 24-03-2025.

Customer Fixed Deposit Page



Customer New FD Form



Customer Support Page

FundLab

Customer Support

Created At: 2025-03-12 17:26:04

status: Solved

Subject: test 2

Description: test 2

Customer Support Detail

FundLab

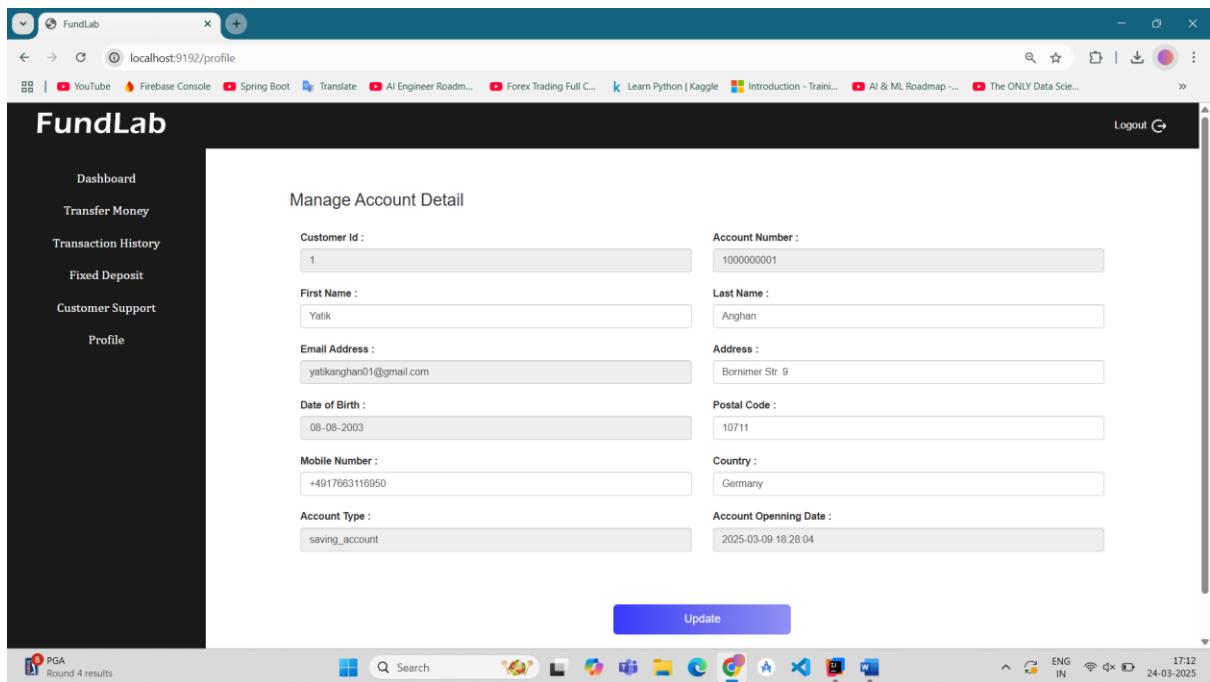
Customer Support

Subject: Enter your subject

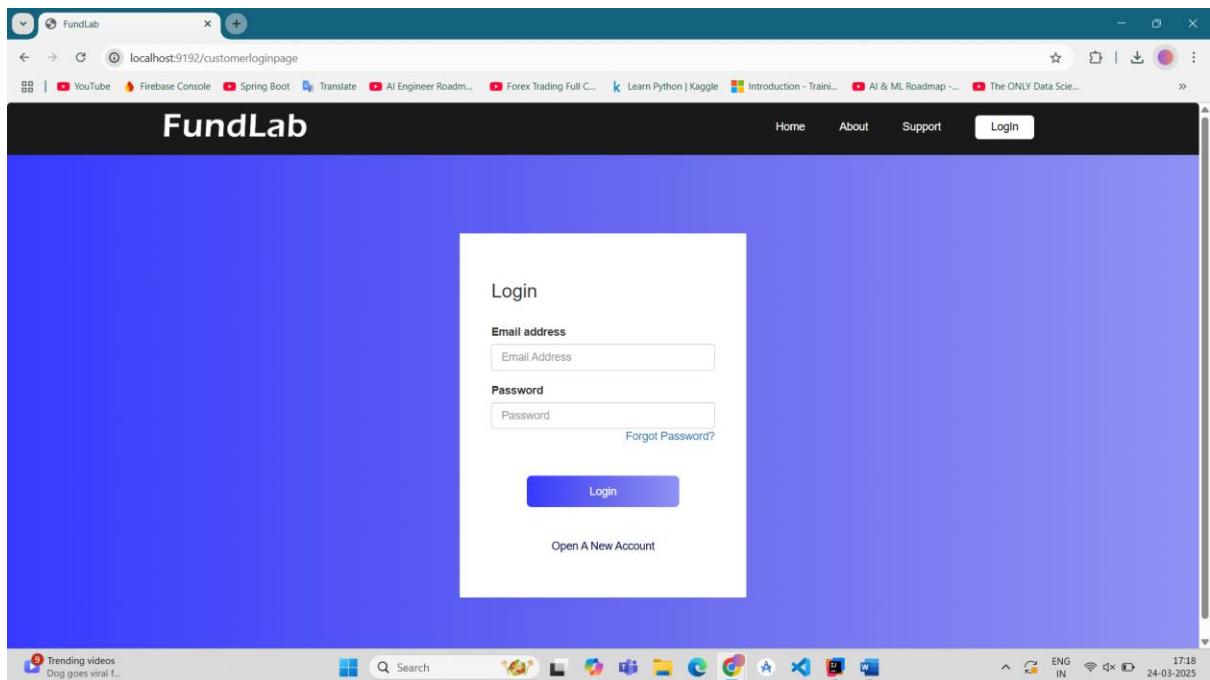
Description:

Submit

Customer Support Add

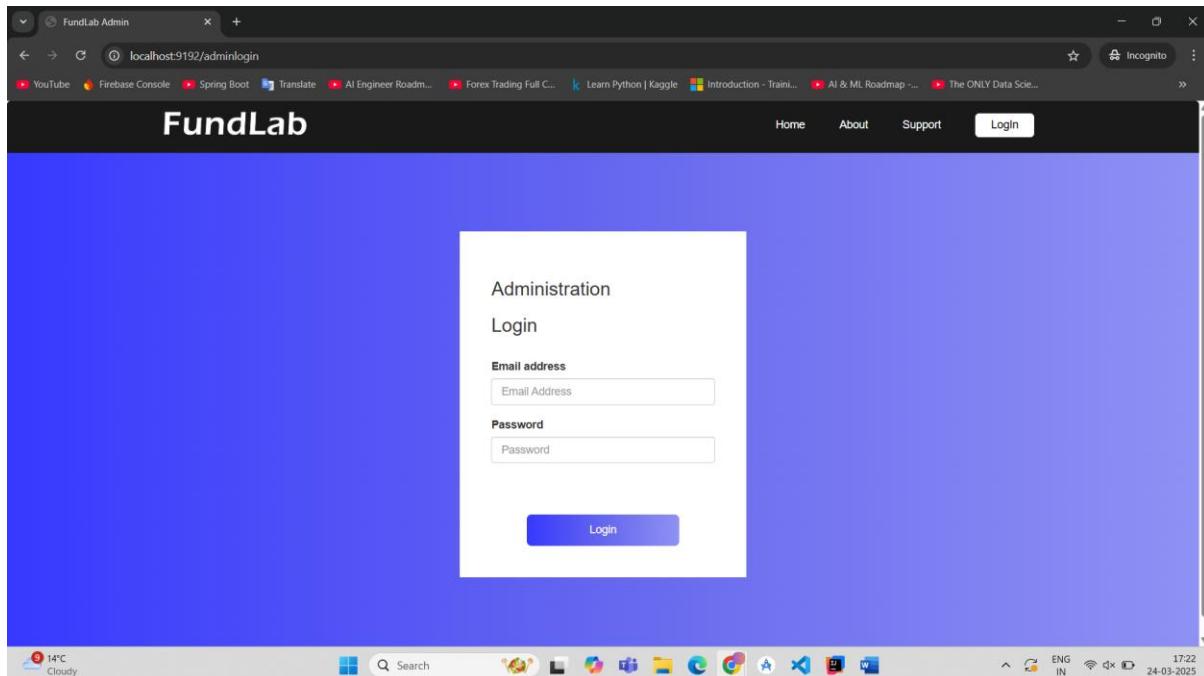


Customer Profile Page

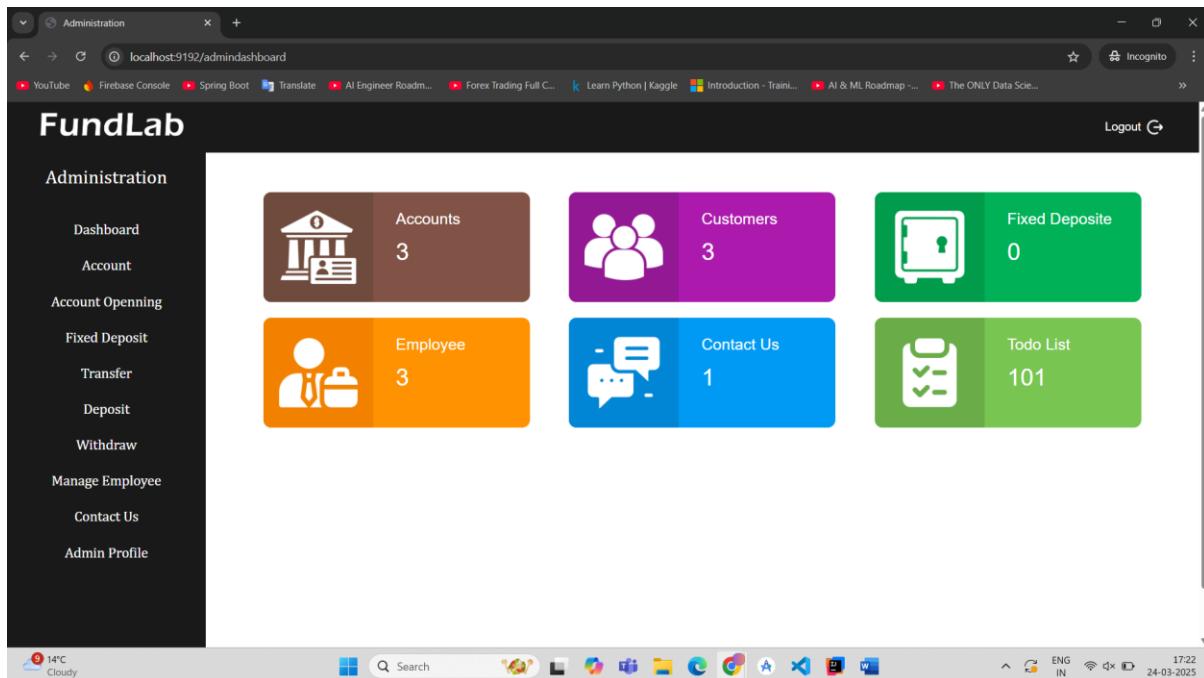


Customer Logout

10.2.4 Administrative Panel



Admin Login Page



Admin Dashboard Page

The screenshot shows a web application interface titled "FundLab". On the left, a dark sidebar menu lists various administrative functions: Dashboard, Account, Account Opening, Fixed Deposit, Transfer, Deposit, Withdraw, Manage Employee, Contact Us, and Admin Profile. The main content area is titled "Manage Accounts" and displays a table with three rows of account information. Each row includes columns for Account Number, First Name, Last Name, Email Address, Mobile Number, and Action (with a green "Update" button). The table data is as follows:

Account Number	First Name	Last Name	Email Address	Mobile Number	Action
1000000001	Yatik	Anghan	yatikanghan01@gmail.com	+917663116950	<button>Update</button>
1000000002	Bansi	Sonani	bansisonani01@gmail.com	+917663117172	<button>Update</button>
1000000003	Bhavin	Asodariya	bhavin01@gmail.com	+917663115487	<button>Update</button>

Admin Manage Account Page

The screenshot shows the "Manage Account Detail" page. The left sidebar remains the same as the previous page. The main content area is titled "Manage Account Detail" and contains a form with various input fields. The fields and their values are:

Customer Id :	1	Last Name :	Anghan
First Name :	Yatik	Mobile Number :	+917663116950
Email Address :	yatikanghan01@gmail.com	Address :	Bornimer Str. 9
Password :	Postal Code :	10711
Date of Birth :	08-08-2003	Country :	Germany

Below this, there is a section titled "Account Detail" with fields for "Account ID :" and "Account Number :".

Admin Manage Account Detail Page

Date of Birth : 08-08-2003

Country : Germany

Account Detail

Account ID :	Account Number :
1	1000000001
Customer ID :	Account Type :
1	Saving Account
Balance :	Account Status :
681.0	Active
Account Opening Date : 2025-03-09 18:28:04	

Update

Admin Customer Account Details

FundLab

New Account Openings

Customer ID	First Name	Last Name	Email Address	Mobile Number	Action
3	Bhavin	Asodariya	bhavin01@gmail.com	+91 17663115487	Action

Admin New Account pending for verifications

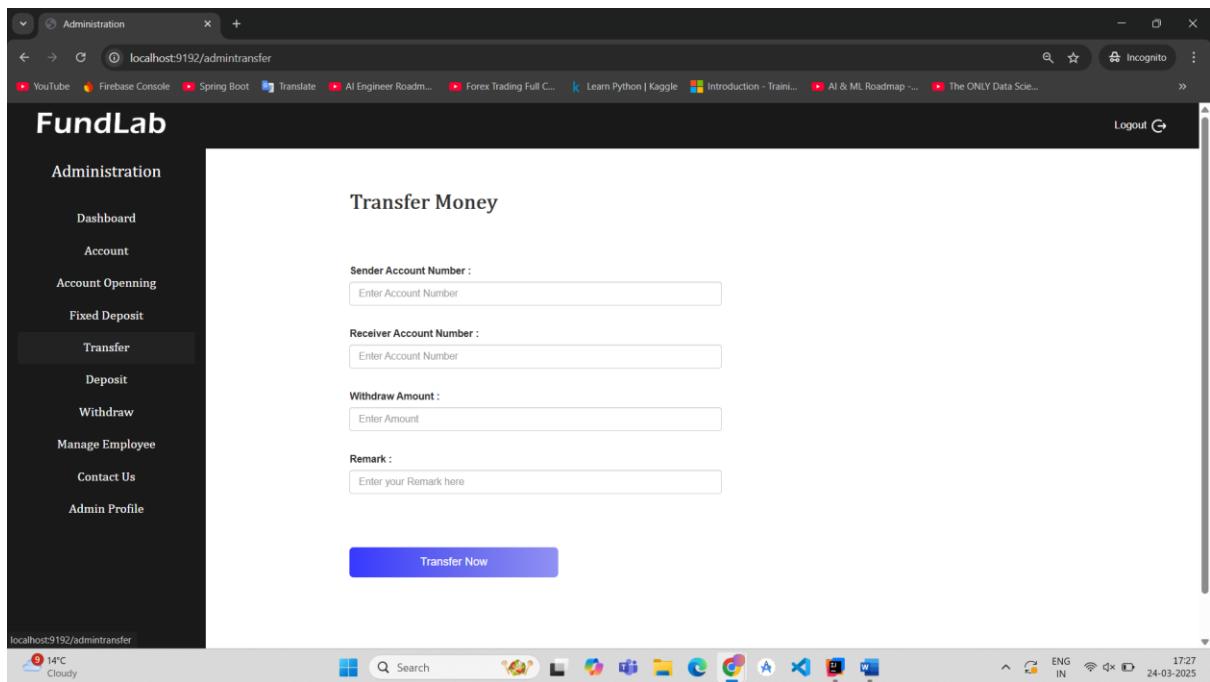
The screenshot shows a web browser window titled "FundLab" at the URL "localhost:9192/customerfixeddeposit". The left sidebar contains navigation links: Dashboard, Transfer Money, Transaction History, Fixed Deposit (selected), Customer Support, and Profile. The main content area is titled "Fixed Deposit" and features a table with three rows of deposit information. A blue button labeled "+ Create New FD" is located above the table.

FD ID	Principle Amount	Maturity Amount	Strat Date	Maturity Date	Interest Rate	Status
1	50.0	53.0	2025-03-16	2026-03-16	6.0	Closed
2	100.0	106.0	2025-03-16	2025-03-17	6.0	Matured
3	50.0	53.0	2025-03-24	2026-03-24	6.0	Active

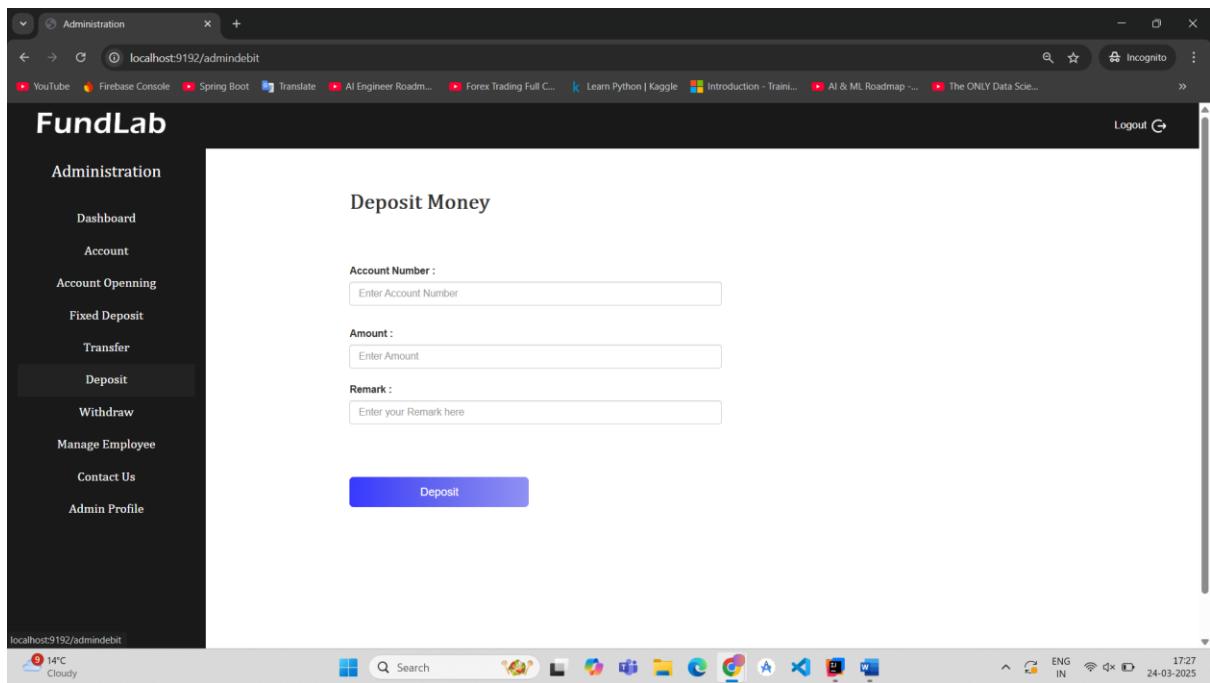
Admin Fixed Deposit Page

The screenshot shows a web browser window titled "FundLab" at the URL "localhost:9192/adminfixdeposit/3". The left sidebar contains navigation links: Dashboard, Account, Account Opening, Fixed Deposit (selected), Transfer, Deposit, Withdraw, Manage Employee, Contact Us, and Admin Profile. The main content area is titled "Fixed Deposit" and displays a form for closing a deposit. The form fields include: FD ID (3), Start Date (2025-03-24), Account Number (1000000001), Maturity Date (2026-03-24), Principle Amount (50.0), Maturity Amount (53.0), Interest Rate (6.0), Status (Active), and Tenure (12). A blue button labeled "Closed Fixed Deposit" is at the bottom of the form.

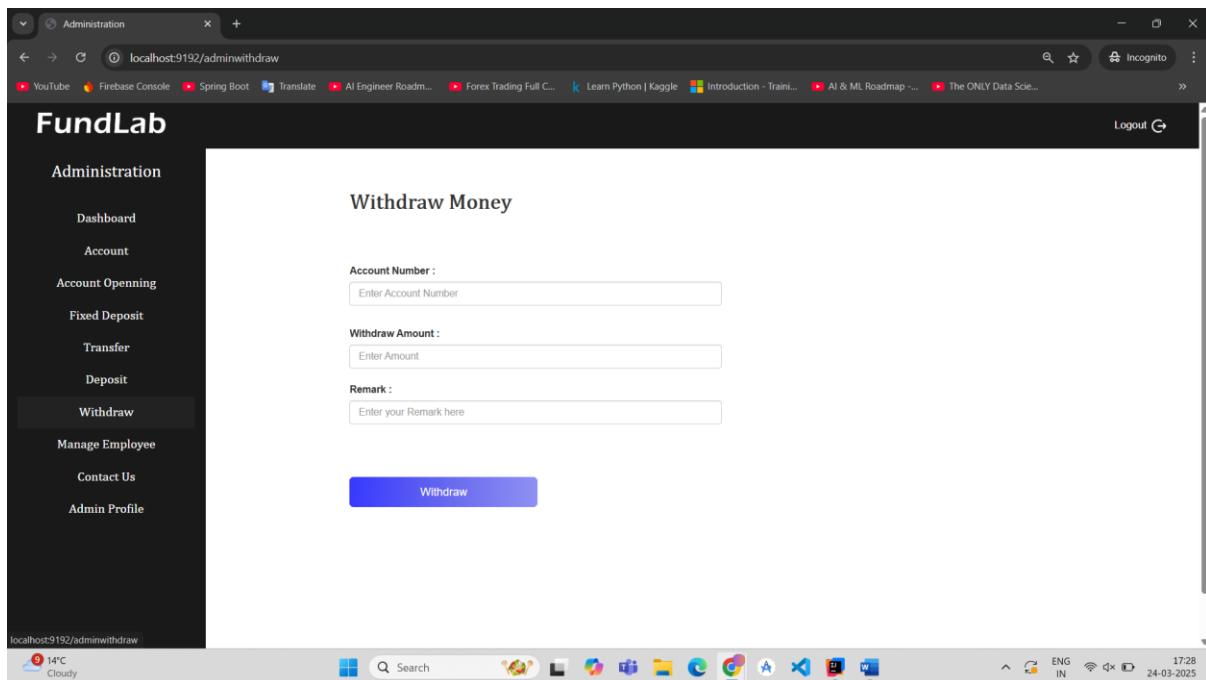
Admin Fixed Deposit Detail



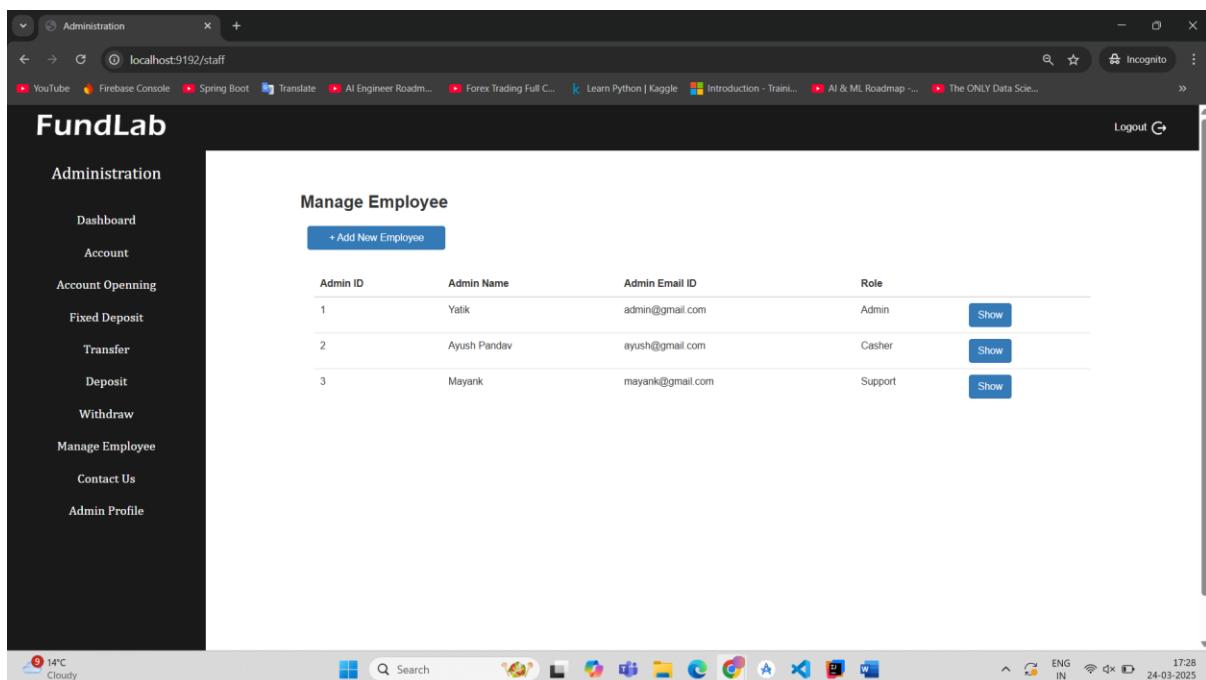
Admin Transfer Money Page



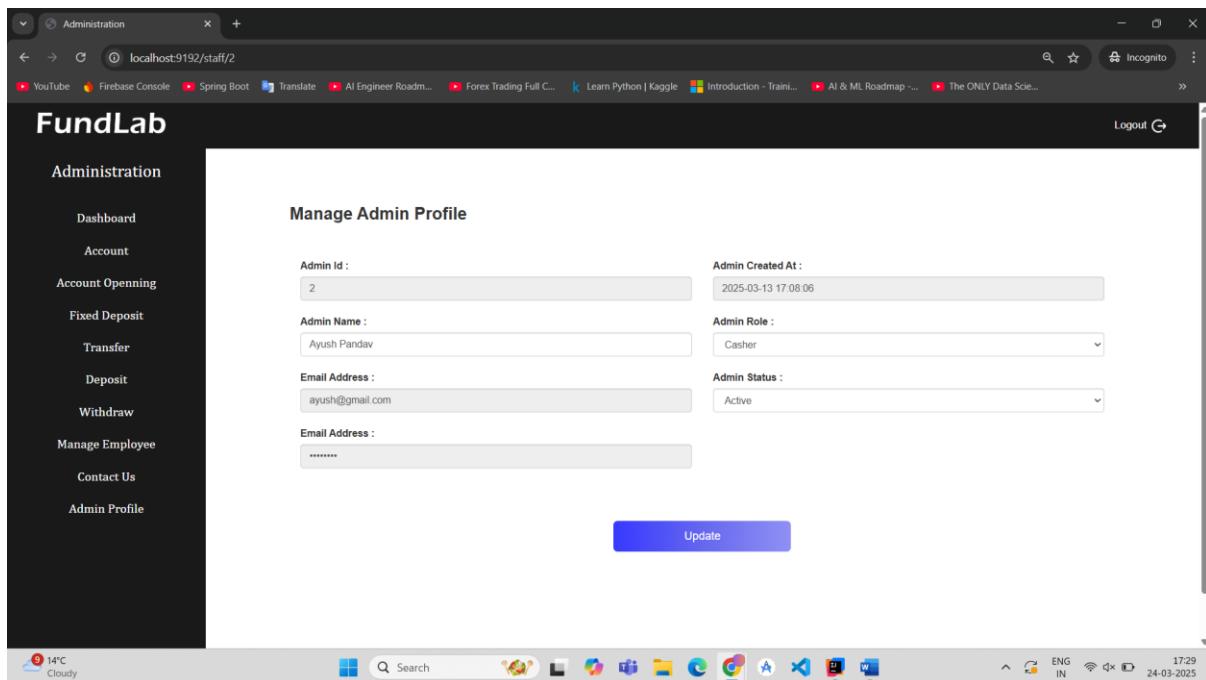
Admin Deposit Money Page



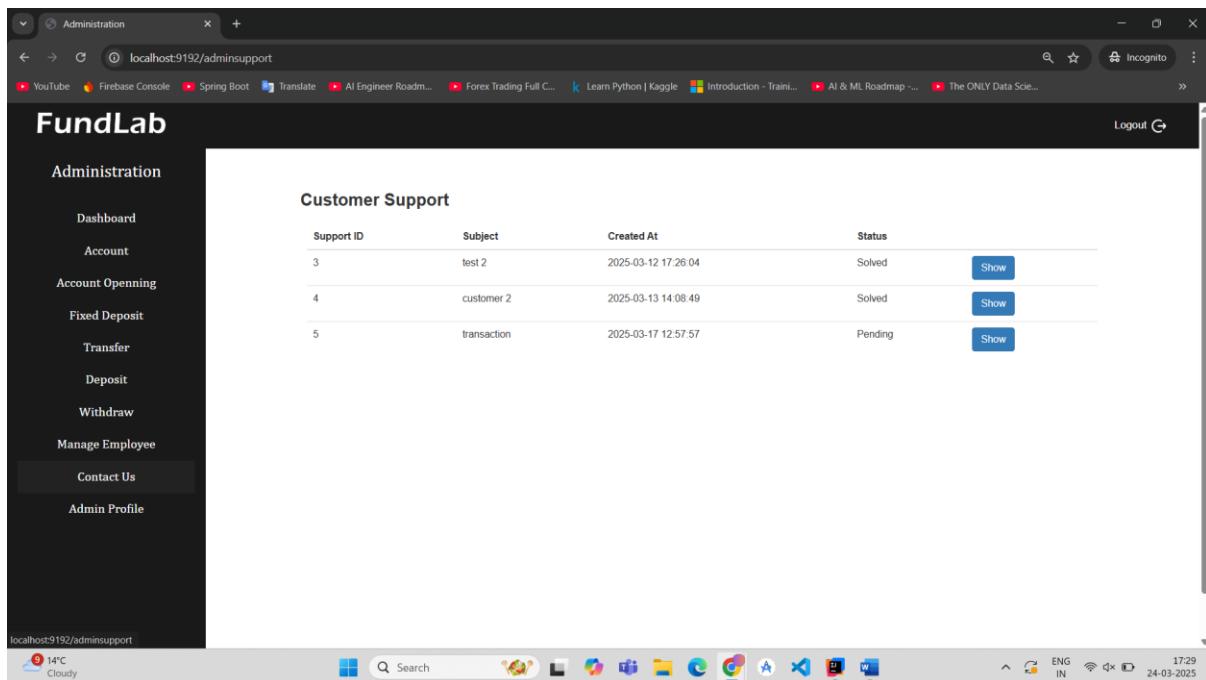
Admin Withdraw Money Page



Admin Manage Employee Page



Admin Magane Employee Detail Page



Admin Customer Support Page

Customer Details

Customer ID :	1	Email Address :	yatikanghan01@gmail.com	Address :	Bornimer Str. 9
First Name :	Yatik	Mobile Number :	+9171653116950	Postal Code :	10711
Last Name :	Anghan	Date of Birth :	2003-08-08	Country :	Germany
Account ID :	1	Account Number :	1000000001	Account Opening :	2025-03-09 18:20:04

Support Details

Support ID :	5	Support Status :	Pending
Support Title :	transaction	Admin ID :	0
Support Description :	this is test query	Support Created At :	2025-03-17 12:57:57

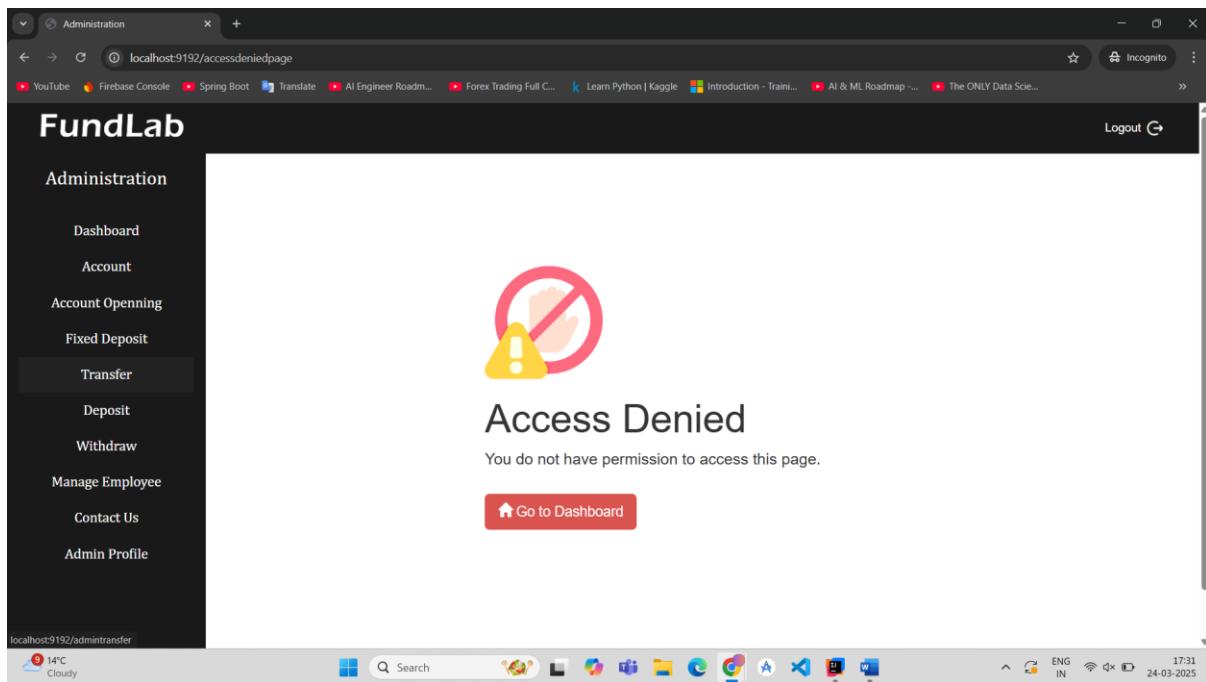
Buttons: Delete Query, Resolve Query

Admin Customer Support Detail Page

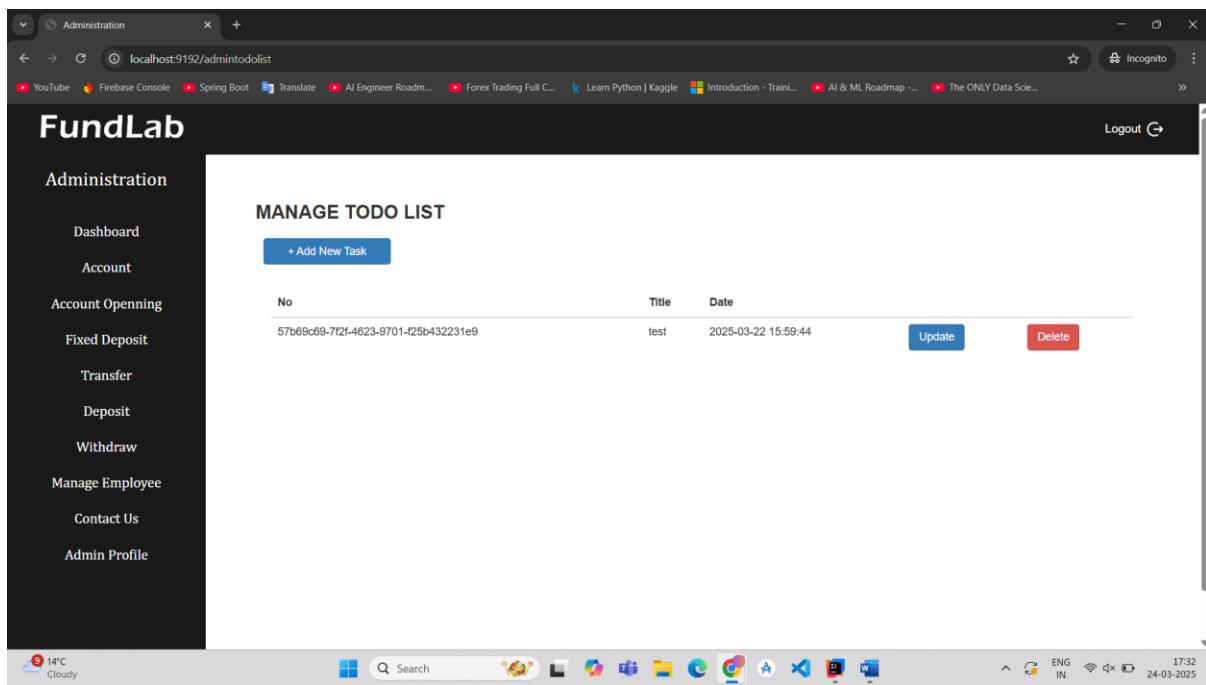
Manage Admin Profile

Admin Id :	1	Admin Created At :	2025-03-08 18:02:20
Admin Name :	Yatik	Admin Role :	Admin
Email Address :	admin@gmail.com	Admin Status :	Active

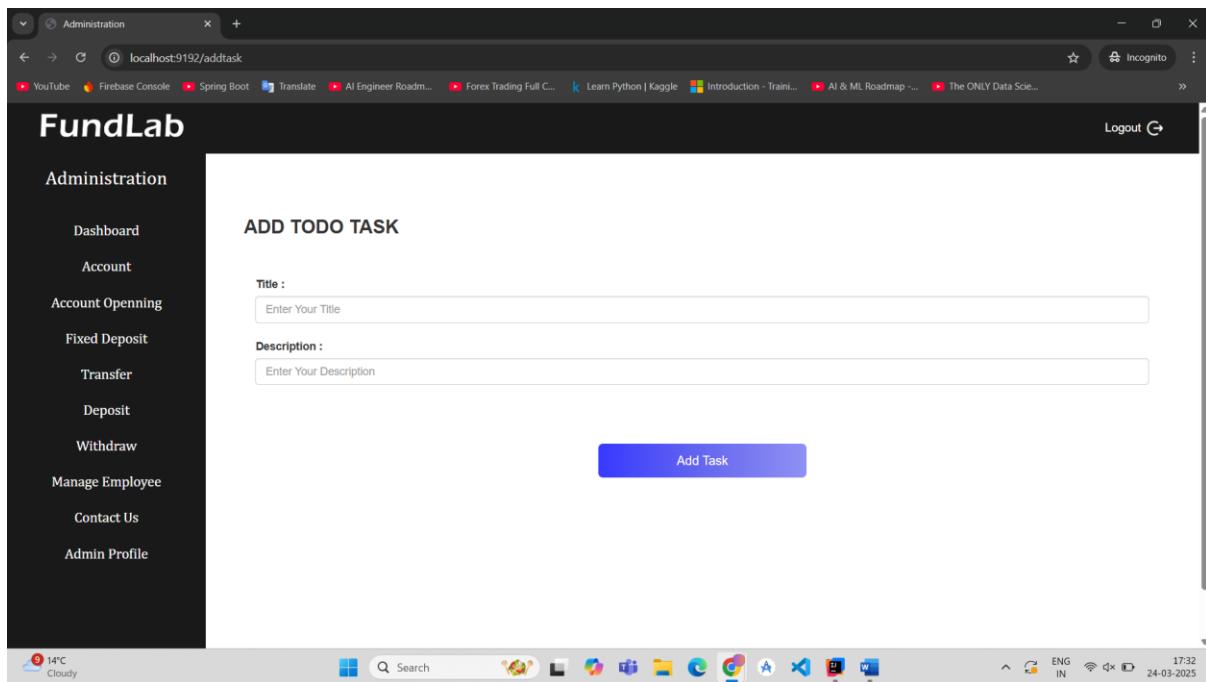
Admin Profile Page



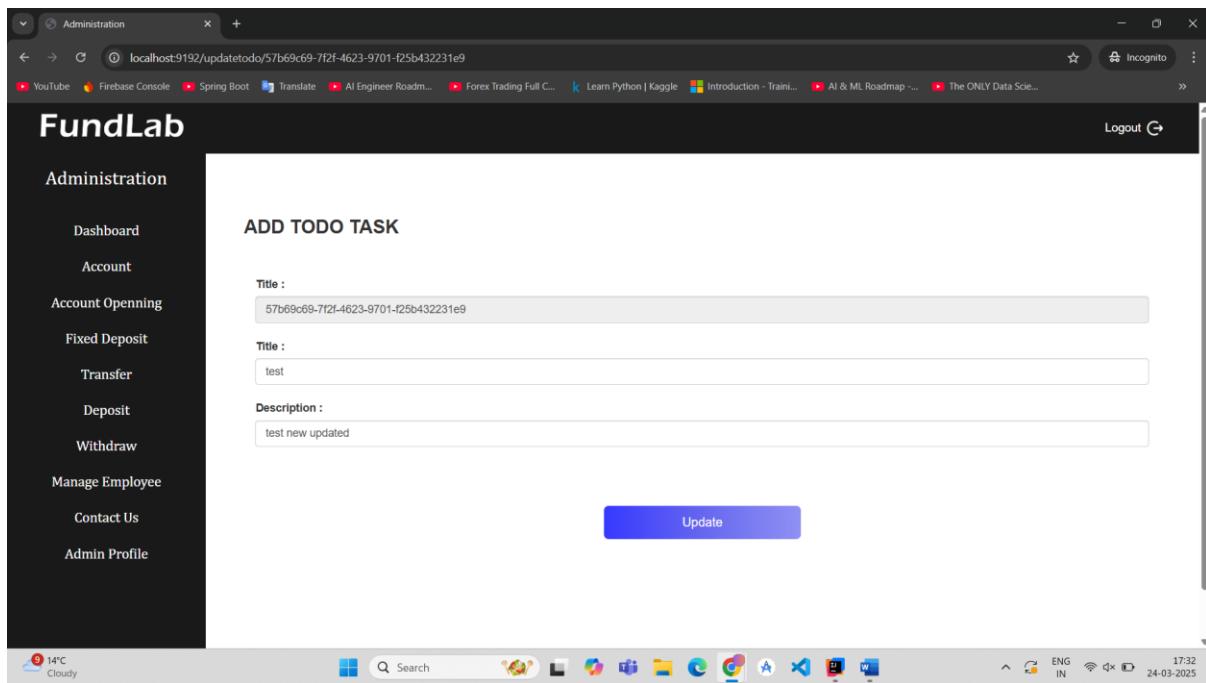
If employee is not access for some feature so that shows access denied page



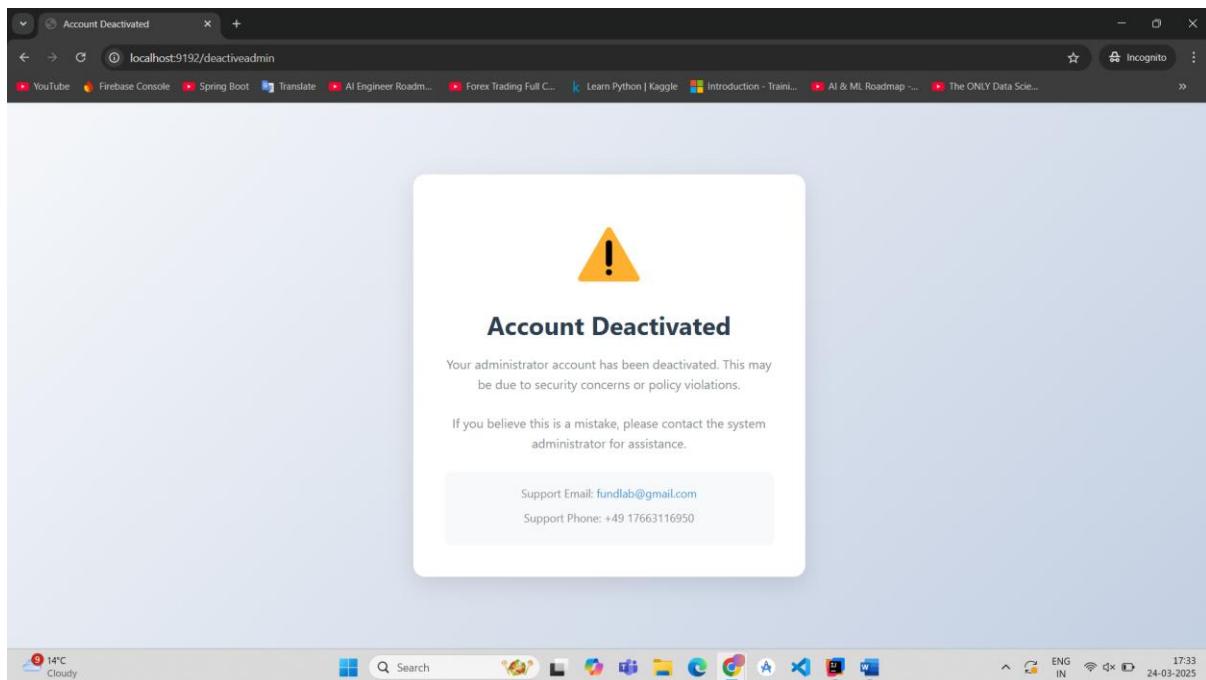
Admin Todo Page



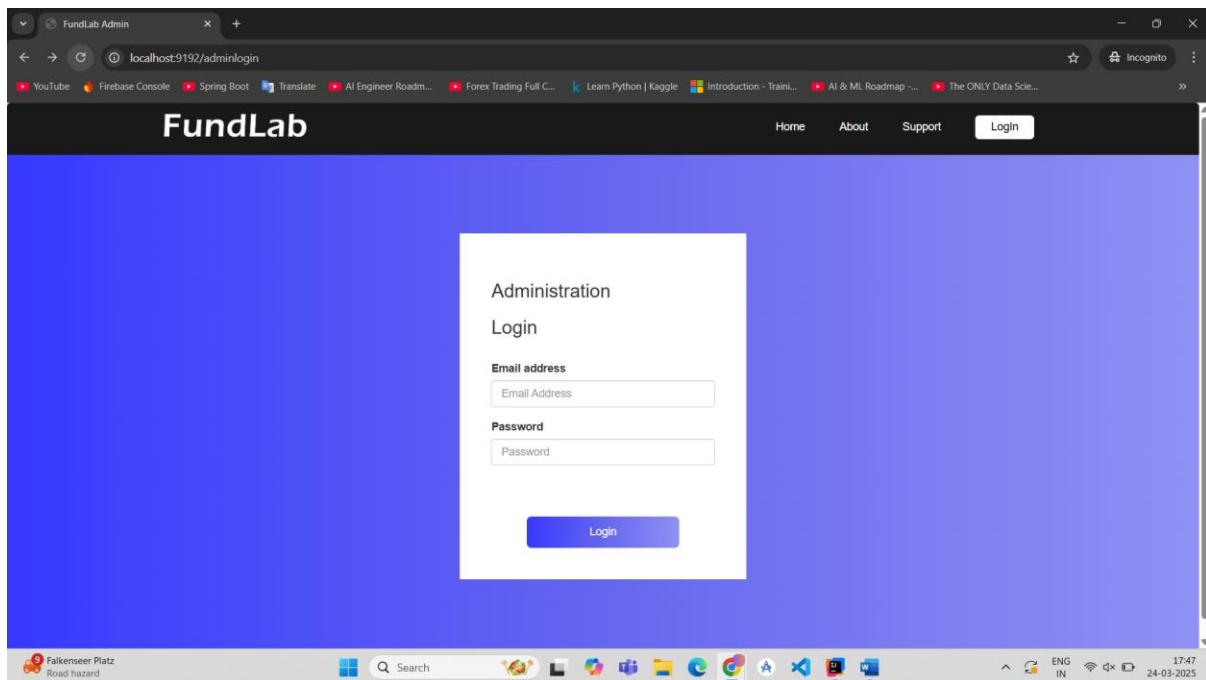
Admin Todo Add Page



Admin Todo Task Detail Page



Employee is Deactivated



Admin Logout

10.2.4 Secure & Error Handling

- System prevents unauthorized access and alerts users on failed login attempts.

Screen shot : Security alert for unauthorized login.

10.2.5 Customer Support System

- User can raise support ticket, and admins can respond to system.

Screen shot : Support Ticket Submission

The screenshot shows the FundLab customer support interface. On the left, there's a dark sidebar with navigation links: Dashboard, Transfer Money, Transaction History, Fixed Deposit, Customer Support (which is currently selected), and Profile. The main content area has a white background. At the top, it says "Customer Support". Below that, there are two input fields: "Subject :" with a placeholder "Enter your subject" and "Description :" with a large text area for input. At the bottom right of the form area is a blue "Submit" button.

11. Conclusion & Future

11.1 Conclusion

The Bank Management System was successfully developed as a web application using Spring Boot for backend development and HTML,CSS and JavaScript for frontend. The system easily handles user authentication, accounts, transactions, fixed deposit, support gives a smooth banking experience.

Through the development, several challenges were faced such as data consistency, security, API performance but in the last project implemented with a secure interaction for customers and administrators both.

The system was tested and got successful outcomes in performance, making it reliable banking solution for financial institutions.

11.2 Future Enhancements

1: Email and SMS notifications

- **Enhancements :** Send automatic email and sms alerts for transactions.
- **Benefits :** Improve user awareness and security without complex technology.

2: Transaction History and Filtering/Sorting

- **Enhancements :** Allow users to filter and sort transactions history by date, type or amount.
- **Benefits :** Make it more easier for users to track transactions without searching.

3: Profile Update Feature

- **Enhancements :** Allow users to update their profile details directly from dashboard.
- **Benefits :** No need to raise ticket to changes and reduces admin works.

4: Downloaded Bank Statements

- **Enhancements :** Allow users to download accounts statements in pdf format.
- **Benefits :** Help users to keep record for personal financial management.

5: Improved error Messages and help section

- **Enhancements :** Display more user-friendly error messages.
- **Benefits :** Helps users understand why an error occurred and how to fix it.

12. References

1. Baeldung : Spring Security Tutorial: Authentication

<https://www.baeldung.com/>

2. Oracle : Java SE Documentation

<https://docs.oracle.com/en/java/>

3. Spring Framework : Spring boot Documentation

<https://docs.spring.io/spring-boot/docs/current/reference/html/>

4. MySQL Documentation : MySQL 8.0 reference Manual

<https://dev.mysql.com/doc/>