

SMART BANKING SYSTEM

A MINI PROJECT REPORT

Submitted by

SHUBHANKAR SINGH(RA2011031010039)
NAMAN MITHAWANI(RA2011031010054)
SANJU DAS (RA2011031010063)

Under the guidance of
Dr. S. Thenmalar
(Associate Professor, SRM Institute
of Science and Technology)

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING
With specialization in Information Technology



SCHOOL OF COMPUTING
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR - 603203

APRIL 2023



SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

COLLEGE OF ENGINEERING & TECHNOLOGY
SRM INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this project report **“Smart Banking System”** is the bonafide work of **“Shubhankar Singh (RA2011031010039), Naman Mithawani (RA2011031010054) and Sanju Das (RA2011031010063)”** of III Year/VI Sem B.tech(CSE-IT) who carried out the mini project work under my supervision for the course 18CSC303J- Database Management systems in SRM Institute of Science and Technology during the academic year 2022-2023(Even sem).



S. Thenmalar
SIGNATURE

Dr. S. Thenmalar
Associate Professor
Networking and Communications
SRM Institute of Science Technology

ABSTRACT

An smart banking system is a piece of software that enables users to conduct financial transactions and manage their accounts online. Customers can use the system to securely and conveniently access their bank accounts, check account information, and carry out a variety of tasks such as transfers, bill payments, and money management. A login module, a transactions module, a money management module, and a reporting module are just a few of the components that make up the online banking system. Customers may securely access their bank accounts using the login module, and they can complete a variety of operations, including transfers and bill payments, using the transactions module. The reporting feature offers account information and transaction history, while the money management section gives consumers tools for creating budgets and managing their funds. Customers can manage their funds and conduct financial transactions using the online banking system, which offers a convenient and secure platform. The Online Banking System improves the overall functioning of the bank and enhances the customer experience with its user-friendly interface, real-time updates, and secure transactions.

TABLE OF CONTENTS

Chapter No.	Title	Page No.
	ABSTRACT	iii
	TABLE OF CONTENTS	iv
	LIST OF FIGURES	v
	ABBREVIATIONS	vi
1	INTRODUCTION	1
1.1	Introduction	1
1.2	Problem statement	2
1.3	Objectives	3
1.4	Scope and applications	4
1.5	General and Unique Services in the database application	5
1.6	Software Requirements Specification	7
2	LITERATURE SURVEY	9
2.1	Existing system	9
2.2	Comparison of Existing vs Proposed system	10
3	SYSTEM ARCHITECTURE AND DESIGN	11
3.1	Architecture Diagram	11
3.1.1	Front end (UI) design	12
3.1.2	Back end (Database) design	19
3.2	ER Diagram and Use case Diagram	20
4	Modules and Functionalities	24
4.1	List of Modules	24
4.2	Connectivity used for Database access	24
4.2.1	MySQL	24
5	CODING AND TESTING	26
6	RESULTS AND DISCUSSIONS	35
6.1	Result	35
6.2	Discussion	35
7	CONCLUSION AND FUTURE ENHANCEMENT	37
	REFERENCES	39

LIST OF FIGURES

Figure No.	Figure Name	Page No.
3.1	Architecture Diagram for Online Banking System	12
3.2	Home page	12
3.3	Admin Login Page	13
3.4	Staff Login Page	13
3.5	Admin Dashboard Page	14
3.6	Staff Dashboard Page	14
3.7	Client Signup Page	15
3.8	Client Login Page	16
3.9	Client Dashboard Page	16
3.10	BackEnd Connectivity	19
3.11	ER Diagram	20
3.12	Use Case Diagram	22

ABBREVIATIONS

ER	Entity Relationship
CSS	Cascading Style Sheet
HTML	HyperText Markup Language
PHP	Hypertext Preprocessor
DBMS	Database Management System
SQL	Structured Query Language
XAMPP	Cross Platform Apache MySQL PHP Perl
UI	User Interface
ER	Entity Relationship

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

In recent years, online banking has become an integral part of our daily lives. With the advancement in technology, banking has become more convenient and accessible than ever before. An online banking system provides customers with the ability to conduct banking transactions over the internet from anywhere and at any time.

The primary objective of this mini-project is to design and implement an online banking system using the principles of Database Management System (DBMS). The system will be developed using a client-server architecture, where the user will interact with the application through a web-based interface, while the server will handle the processing and storage of user data.

The project will be developed using various software tools such as HTML, CSS, PHP, MySQL, and JavaScript. The web-based interface will provide the user with the ability to perform various banking transactions such as checking account balances, transferring funds, and making bill payments. The system will also have features for user authentication and authorization, as well as security measures to protect user data.

The development of an online banking system using DBMS will not only provide convenience and accessibility to customers but also improve the efficiency of banking operations. By eliminating the need for manual processing of transactions, the system will reduce the risk of errors and increase the speed of processing.

Overall, the project will provide a practical demonstration of the use of DBMS principles in the development of an online banking system. The project will also equip students with the necessary skills to develop similar applications in the future.

1.2 PROBLEM STATEMENT

PROBLEM DEFINITION:

Despite the widespread adoption of online banking, many traditional banks still rely on manual processing of transactions. This results in delays, errors, and higher operating costs. Additionally, customers often face issues such as long wait times and limited banking hours, which can be inconvenient for those with busy schedules.

The project will also focus on addressing the security concerns associated with online banking, such as data breaches and unauthorized access. The system will incorporate various security features such as user authentication and authorization, data encryption, and multi-factor authentication to ensure the confidentiality and integrity of customer data

The project will also focus on addressing the security concerns associated with online banking, such as data breaches and unauthorized access. The system will incorporate various security features such as user authentication and authorization, data encryption, and multi-factor authentication to ensure the confidentiality and integrity of customer data.

PROBLEM SOLUTION:

The proposed solution is to develop an online banking system using DBMS principles that will provide customers with a more convenient and efficient way to conduct banking transactions. The system will offer 24/7 accessibility and faster processing times, reducing the need for manual intervention and human error.

To address security concerns associated with online banking, the system will incorporate various security features such as user authentication and authorization, data encryption, and multi-factor authentication to ensure the confidentiality and integrity of customer data.

By developing a reliable and secure online banking system, the project aims to promote the adoption of online banking and help traditional banks become more efficient in their operations. The system will provide customers with a more seamless and convenient banking experience, ultimately leading to higher customer satisfaction.

1.3 OBJECTIVES

The main objectives of the proposed mini-project on developing an online banking system using DBMS are as follows:

1. To provide customers with a more convenient and accessible way to conduct banking transactions from anywhere and at any time
2. To reduce the processing time of transactions and minimize the risk of errors by eliminating the need for manual processing.
3. To incorporate various security measures such as user authentication and authorization, data encryption, and multi-factor authentication to ensure the confidentiality and integrity of customer data.
4. To develop a user-friendly interface that is easy to navigate and allows customers to perform banking transactions with ease.
5. To provide customers with a wide range of banking services, including account management, fund transfers, bill payments, and transaction history tracking.
6. To promote the adoption of online banking and help traditional banks become more efficient and cost-effective in their operations.
7. Data Management: Provide a centralized platform for banks to manage customer information, account data, and transaction history.
8. Reporting: Generate real-time reports on banking activities, account information, and customer transactions.
9. Compliance: Ensure compliance with regulatory requirements and industry standards for security and data protection.

Overall, the project aims to provide a practical demonstration of the use of DBMS principles in the development of an online banking system. The project will also help bridge the gap between traditional banking methods and modern technology and promote the adoption of online banking.

1.4 SCOPE AND APPLICATIONS

The scope of the proposed mini-project on developing an online banking system using DBMS principles is vast and has significant potential for implementation in the banking industry. The project aims to provide a reliable and efficient platform for conducting banking transactions online, which is becoming increasingly important in today's fast-paced world.

The online banking system developed using DBMS principles can be used by banks of all sizes, ranging from small local banks to large international financial institutions. The system can also be customized to suit the specific needs of each bank and their customers.

The system can provide customers with a range of banking services, including account management, fund transfers, bill payments, and transaction history tracking. Customers can also access their account information from anywhere and at any time, which provides greater convenience and accessibility.

The online banking system developed using DBMS principles can help banks become more efficient and cost-effective in their operations by reducing the need for manual processing and minimizing the risk of errors. The system can also help banks expand their customer base by offering a more seamless and convenient banking experience.

The project has applications beyond the banking industry as well. The principles and techniques used in developing an online banking system using DBMS can be applied to other industries that require web-based applications, such as e-commerce, healthcare, and government services.

Overall, the project has significant scope and applications in the banking industry and beyond. It has the potential to revolutionize the way banking transactions are conducted and help bridge the gap between traditional banking methods and modern technology.

1.5 GENERAL AND UNIQUE SERVICES IN THE DATABASE

LIST OF GENERAL SERVICES:

- 1. Account management:** Customers can use the online banking system to open new accounts, update their personal information such as address and contact details, and manage their existing accounts. They can check their account balances, view transaction histories, and monitor their account activity.
- 2. Fund transfers:** The online banking system allows customers to transfer funds between their accounts, or to other accounts within the same bank or other banks. Customers can schedule recurring transfers and set up one-time transfers. They can also view the status of their transfer requests.
- 3. Transaction history:** The online banking system allows customers to view their transaction history, including deposits, withdrawals, and transfers. Customers can filter transactions by date range, transaction type, and account.
- 4. Account statements:** Customers can view and download account statements, including monthly statements and annual statements. The online banking system can store statements for multiple accounts, and customers can view their statements in PDF or CSV format.
- 5. Alerts and notifications:** Customers can set up alerts and notifications for transactions, account balance updates, and bill payments. The online banking system can send alerts via email or SMS, and customers can choose the type of alerts they want to receive.
- 6. Mobile banking:** Customers can access the online banking system through a mobile application, allowing them to conduct banking transactions on-the-go. The mobile application can provide all the functionality of the online banking system in a mobile-friendly format.
- 7. Loan applications:** Customers can apply for loans online, including personal loans, home loans, and auto loans. The online banking system can provide customers with information about available loan products, interest rates, and eligibility criteria. Customers can also track the status of their loan applications and receive alerts when their applications are approved or rejected.

LIST OF UNIQUE SERVICES:

1. **24/7 Availability:** Unlike traditional banking services that have limited hours of operation, online banking systems are available 24/7. This means customers can access their accounts and conduct transactions at any time of the day or night, from anywhere in the world.
2. **Convenience:** Online banking systems provide a high level of convenience to customers. They can access their accounts from their home or office, without having to visit a physical bank branch. Customers can conduct transactions quickly and easily, without the need to wait in long lines.
3. **Real-time updates:** Online banking systems provide real-time updates on account balances and transaction activity. Customers can view their account balances and transaction history instantly, as soon as the transactions are processed.
4. **Enhanced Security:** Online banking systems have advanced security features, such as multi-factor authentication, encryption, and fraud detection. These features protect customer accounts and transactions from unauthorized access and fraud.
5. **Personalization:** Online banking systems can provide personalized banking services to customers. For example, they can offer customized financial advice based on a customer's financial goals, spending patterns, and investment portfolio.
6. **Automatic alerts:** Online banking systems can send automatic alerts to customers for various events, such as low account balances, large transactions, and bill payment due dates. These alerts help customers to stay on top of their finances and avoid overdraft fees or missed payments.
7. **Easy money management:** Online banking systems can provide tools for easy money management, such as budget tracking, expense categorization, and financial goal setting. These tools can help customers to better manage their finances and achieve their financial goals.
8. **Integration with other financial services:** Online banking systems can be integrated with other financial services, such as investment accounts, credit cards, and loan accounts. This allows customers to access all of their financial information in one place, making it easier to manage their finances.

1.6 SOFTWARE REQUIREMENT SPECIFICATION

1. HTML:

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and applications. Each page contains a series of connections to other pages called hyperlinks. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are delineated by tags, written using angle brackets. Tags such as `` and `<input/>` introduce content into the page directly. Others such as `<p>....</p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look. Without HTML, a browser would not know how to display text as elements or load images or other elements. HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance.

2. CSS:

CSS (Cascading Style Sheets) is a style sheet language used for describing the presentation of a document written in HTML (Hypertext Markup Language). It is used to style and layout web pages, including those that interact with a database management system (DBMS). Here are some ways CSS can be used in building web applications that interact with a DBMS:

Styling User Interface Components: CSS can be used to style various user interface components in web applications that interact with a DBMS, such as forms, tables, and charts. By using CSS, developers can customize the look and feel of these components to match the branding and visual design of the application.

3. Responsive Design:

CSS can be used to create responsive designs that adapt to different screen sizes and device types. This is especially important in web applications that interact with a DBMS, as users may need to access the application from different devices with varying screen sizes.

4. Php:

PHP is an acronym for "PHP: Hypertext Preprocessor". It is a widely-used, open source scripting language. Its scripts are executed on the server. It is free to download and use. It can generate dynamic page content. It can create, open, read, write, delete, and close files on the server. It can collect form data. It can send and receive cookies. It can add, delete, modify data in your database. It can be used to control user-access. It can encrypt data.

5. MySQL:

Databases are the essential data repository for all software applications. For example, whenever someone conducts a web search, logs in to an account, or completes a transaction, a database system is storing the information so it can be accessed in the future.

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structure is organized into physical files optimized for speed. The logical data model, with objects such as data tables, views, rows, and columns, offers a flexible programming environment.

The "SQL" part of "MySQL" stands for "Structured Query Language." SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

6. Javascript:

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

The existing system contains of the following features:

- 1. Digital banking platforms:** Most banks offer online banking services through digital banking platforms that allow customers to manage their accounts, conduct transactions, and access other banking services through a website or mobile app.
- 2. Account management:** Online banking systems allow customers to manage their accounts, including opening new accounts, updating personal information, and managing existing accounts. Customers can check their account balances, view transaction histories, and monitor their account activity.
- 3. Fund transfers:** Online banking systems allow customers to transfer funds between their accounts, or to other accounts within the same bank or other banks. Customers can schedule recurring transfers and set up one-time transfers. They can also view the status of their transfer requests.
- 4. Transaction history:** Online banking systems allow customers to view their transaction history, including deposits, withdrawals, and transfers. Customers can filter transactions by date range, transaction type, and account.
- 5. Account statements:** Online banking systems allow customers to view and download account statements, including monthly statements and annual statements. The online banking system can store statements for multiple accounts, and customers can view their statements in PDF or CSV format.
- 6. Customer service:** Online banking systems can provide customers with access to customer service support through online chat, email, and phone support. Customers can get assistance with their banking needs, including technical issues, account management, and transactions

Overall, the existing system of online banking offers a range of services to customers, providing them with convenient access to banking services from anywhere, at any time. However, there is always room for improvement in terms of user experience, security, and integration with other financial services.

2.2 COMPARISON OF EXISTING VS PROPOSED SYSTEM

To compare the existing system and proposed system for a online banking system, we need to consider the differences in features, functionality, and efficiency. Here are some potential differences that could exist between the two systems:

Existing System:

1. Basic user interface, with limited features
2. Single-factor authentication, with a password and username
3. Limited transaction management features, such as one-time transfers and limited bill payments
4. No personal financial management tools or mobile app
5. Limited integration with other financial services
6. Limited customer service support, such as email or phone support
7. Basic reporting features, such as transaction history
8. No artificial intelligence or personalized financial advice

Proposed System:

1. User-friendly interface, with advanced features
2. Multi-factor authentication, with advanced security features
3. Enhanced transaction management features, such as the ability to schedule recurring transfers and pay bills online
4. Mobile app for easy access to accounts and transactions on-the-go
5. Integration with other financial services, such as investment accounts and credit cards
6. Access to customer service support through online chat, email, and phone support
7. Advanced reporting features, such as customized reports on account activity, transaction history, and investment performance

Overall, the proposed system offers a more comprehensive and user-friendly online banking experience, with advanced security features, enhanced transaction management, personal financial management tools, mobile app access, and integration with other financial services. The proposed system also offers better customer service support, advanced reporting features, and personalized financial advice through artificial intelligence.

CHAPTER 3

ARCHITECTURE AND DESIGN SYSTEM

3.1 ARCHITECTURE DIAGRAM

The architecture diagram consists of the following components:

- 1. User Interface:** This is the front-end of the system that customers interact with. It provides a graphical user interface for accessing their accounts, managing their transactions, and conducting financial operations.
- 2. Application Server:** The application server is the middle-tier of the system that processes and manages the customer's requests. It handles the business logic, data processing, and other operations related to online banking services.
- 3. Database Server:** The database server is the back-end of the system that stores and manages customer data. It stores account information, transaction history, and other data related to online banking services.
- 4. Security:** Security is an essential component of any online banking system. It includes measures such as firewalls, encryption, and multi-factor authentication to protect customer data and prevent unauthorized access.
- 5. Payment Gateway:** The payment gateway is responsible for processing payments, such as bill payments, loan payments, and fund transfers. It handles the payment processing, including payment authorization, settlement, and reconciliation..
- 6. Mobile Banking:** Many online banking systems also provide mobile apps that allow customers to access their accounts and conduct transactions on-the-go.

Overall, the architecture of an online banking system consists of several components that work together to provide customers with a comprehensive and secure online banking experience.

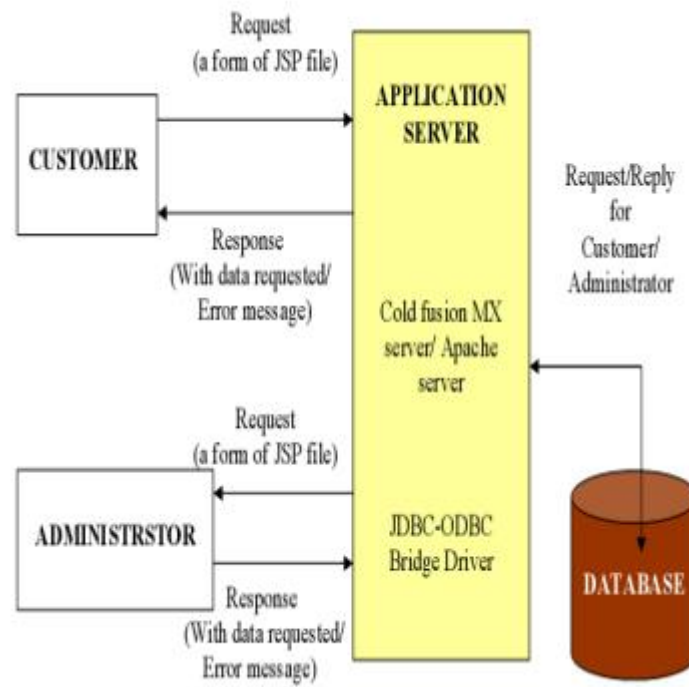


Fig 3.1 Architecture Diagram

3.1.1 FRONT END (UI) DESIGN

HOME PAGE

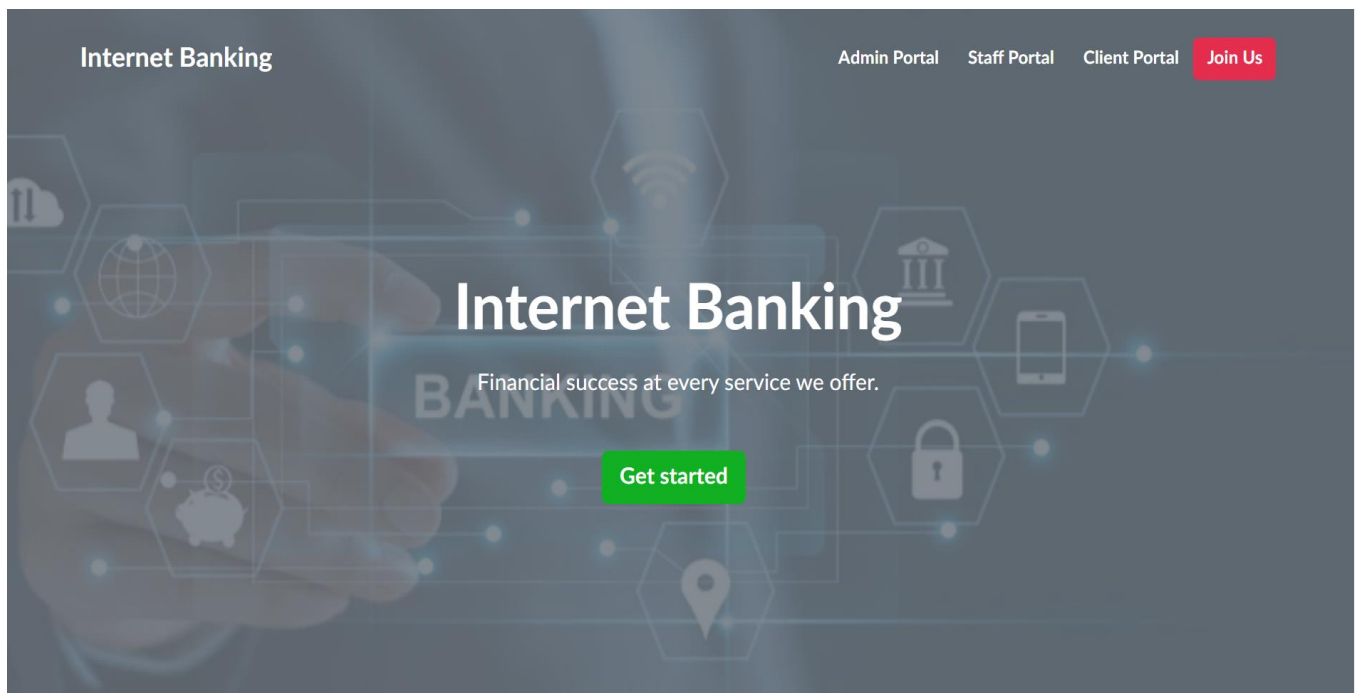
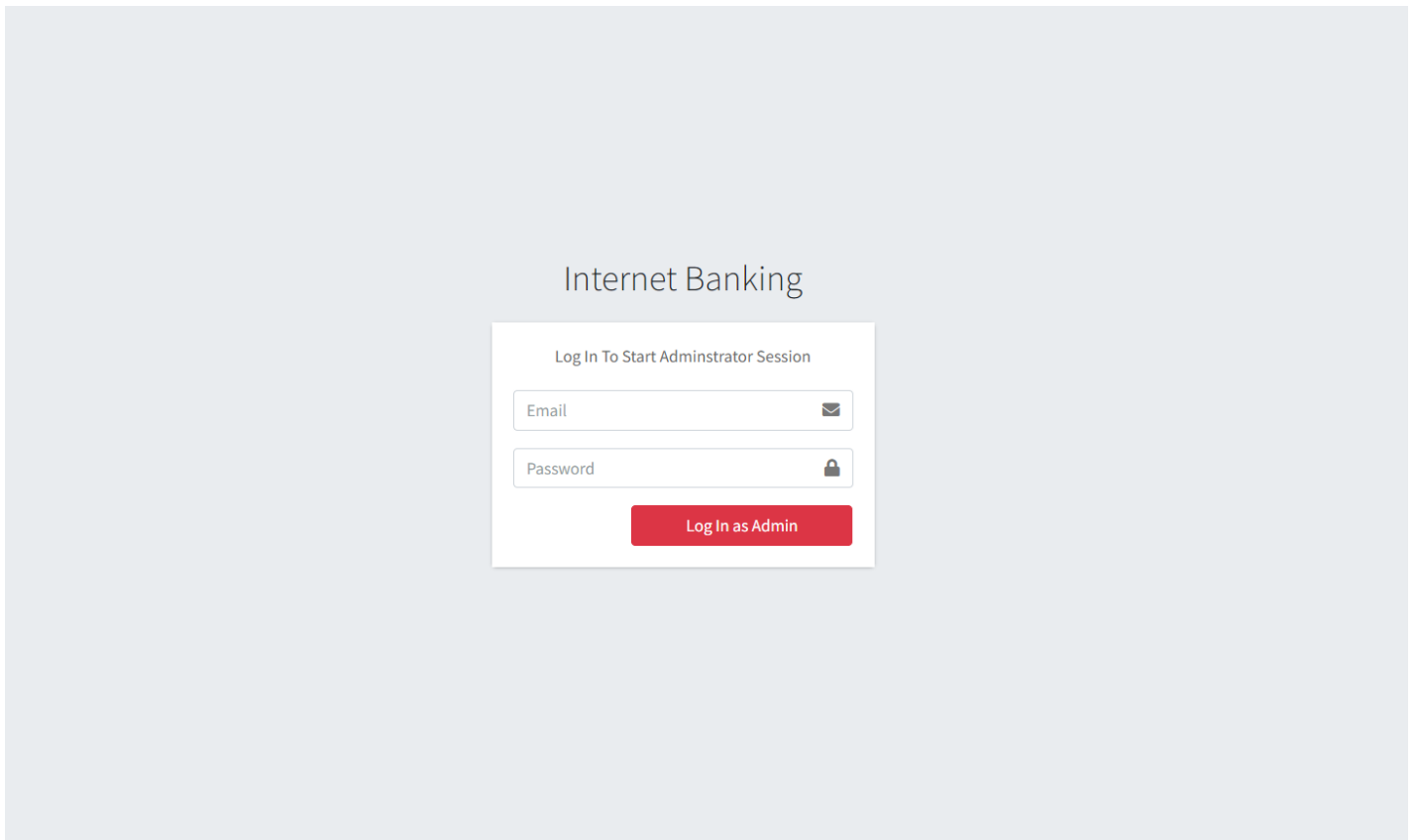


Fig 3.2 Home Page

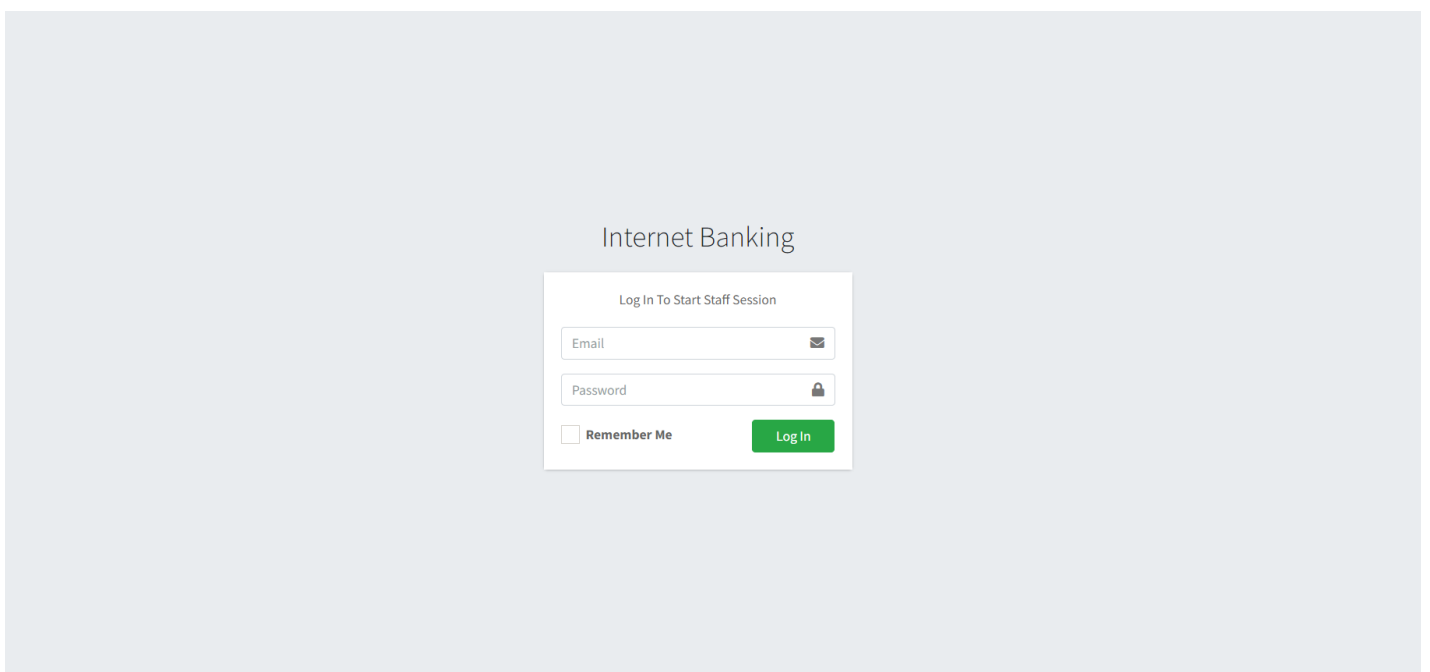
ADMIN LOGIN PAGE



The screenshot shows the Admin Login Page of an Internet Banking system. The page has a light gray background. At the top center, the text "Internet Banking" is displayed. Below it, a white login box is centered. Inside the box, the heading "Log In To Start Administrator Session" is at the top. There are two input fields: "Email" with an envelope icon and "Password" with a lock icon. Below these fields is a red button labeled "Log In as Admin".

Fig 3.3 Admin Login Page

STAFF LOGIN PAGE



The screenshot shows the Staff Login Page of an Internet Banking system. The page has a light gray background. At the top center, the text "Internet Banking" is displayed. Below it, a white login box is centered. Inside the box, the heading "Log In To Start Staff Session" is at the top. There are two input fields: "Email" with an envelope icon and "Password" with a lock icon. Below these fields, there is a checkbox labeled "Remember Me" and a green button labeled "Log In".

Fig 3.4 Staff Login Page

ADMIN DASHBOARD PAGE

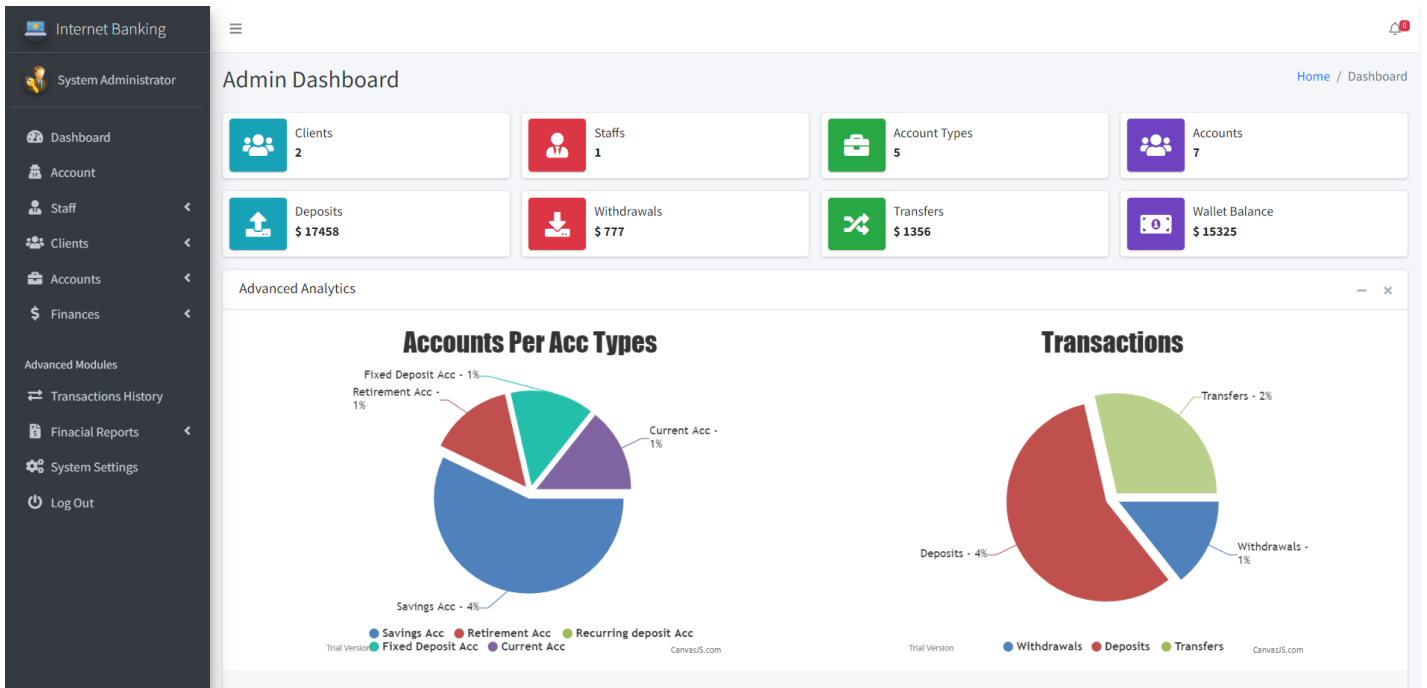


Fig 3.5 Admin Dashboard

STAFF DASHBOARD PAGE

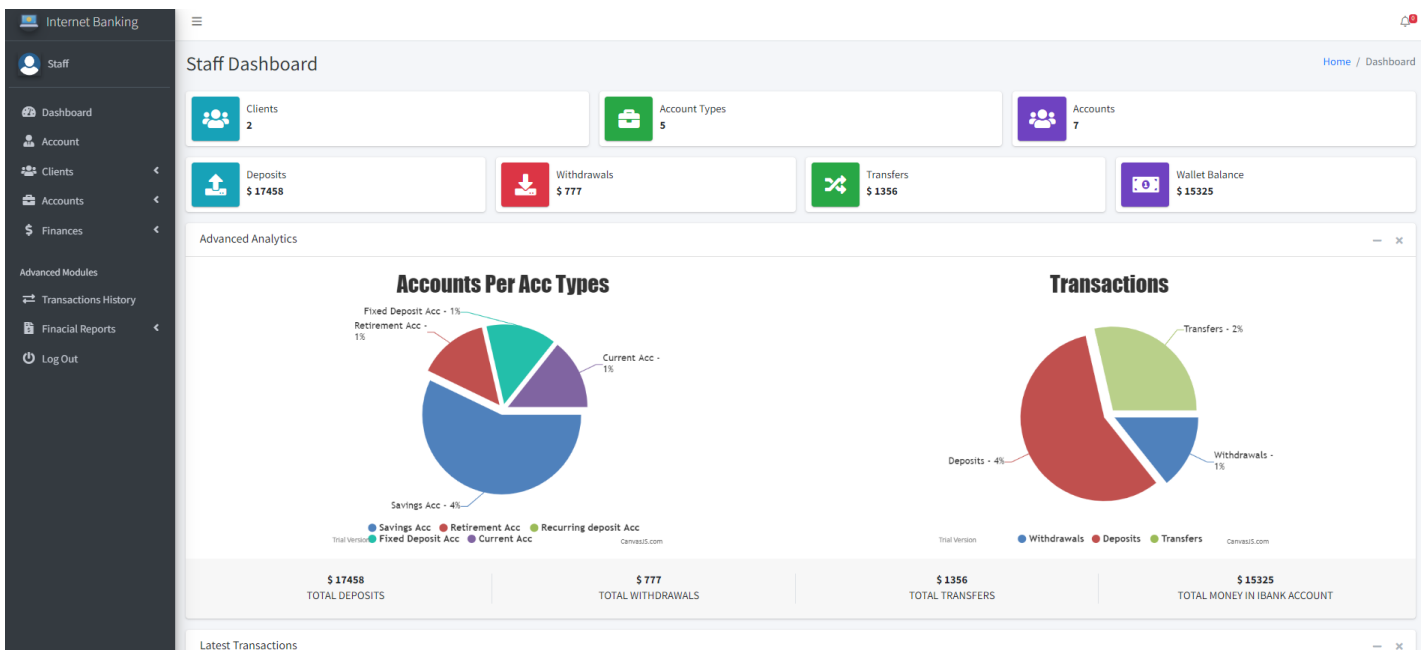




Fig 3.6 Staff Dashboard


CLIENT SIGNUP PAGE


Internet Banking - Sign Up


Sign Up To Use Our IBanking System


Client Full Name 

National ID Number 

Client Phone Number 

Client Address 

Client Address 

Password 

[Login](#)

[Sign Up](#)

Fig 3.7 Client Signup Page

CLIENT LOGIN PAGE

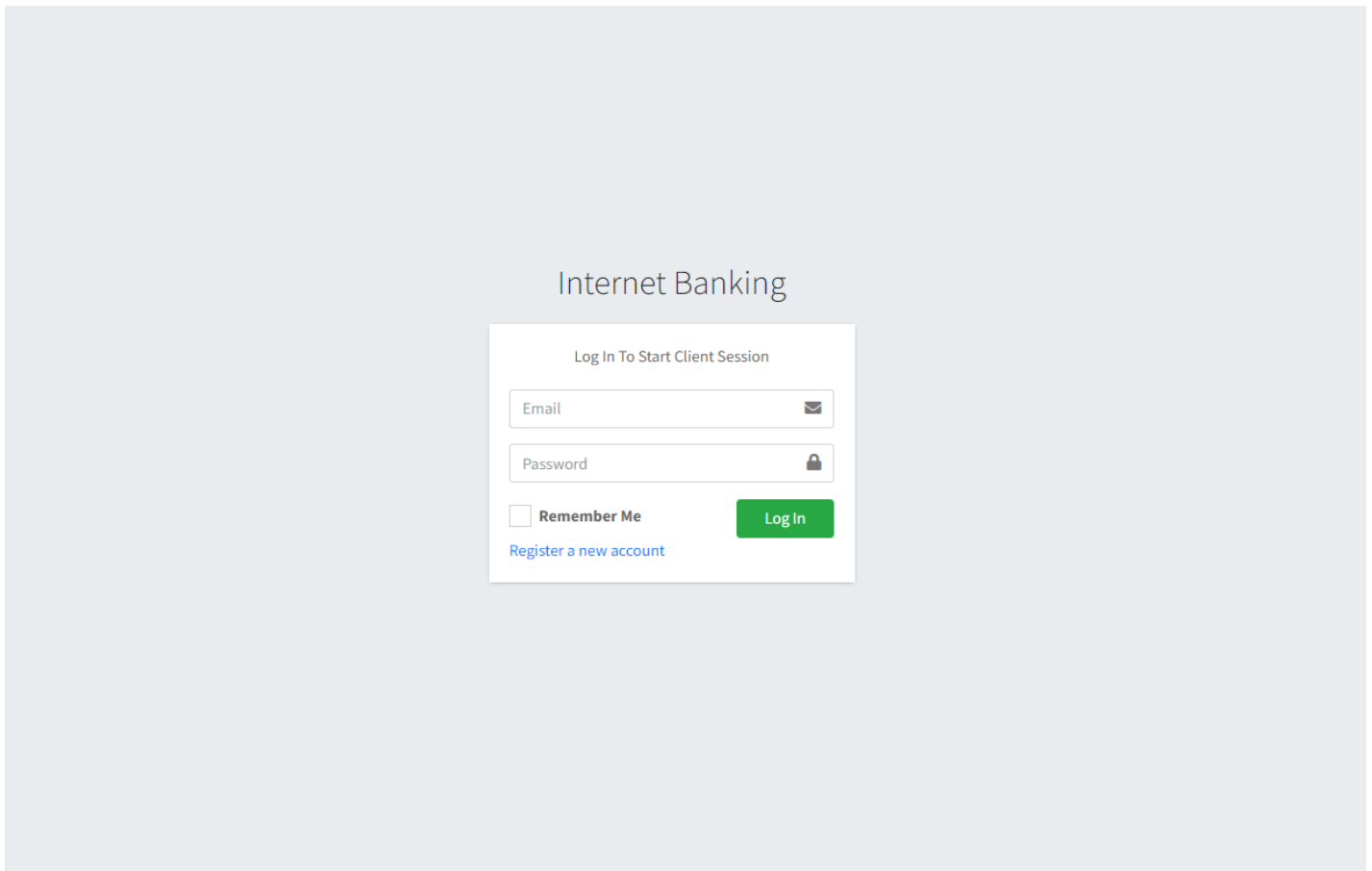


Fig 3.8 Client Login Page

CLIENT DASHBOARD PAGE

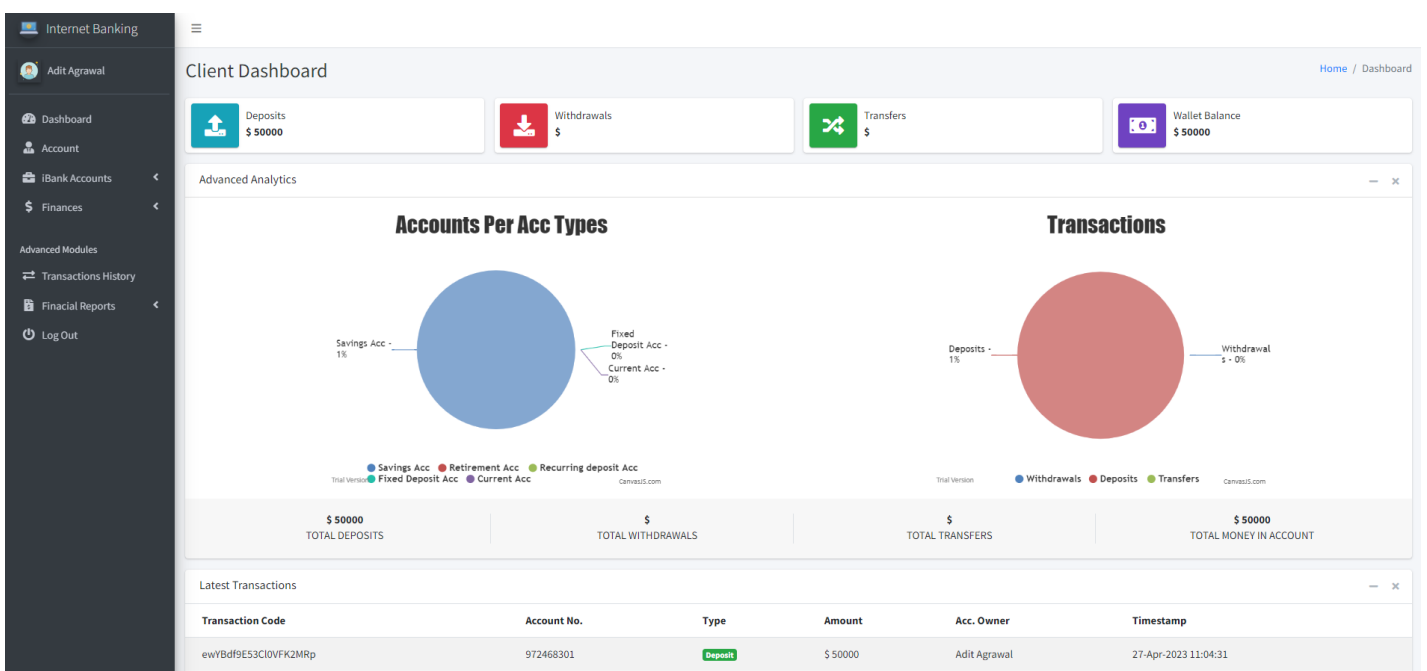


Fig 3.9 Client Dashboard

SOFTWARE USED

1. HTML:

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and applications. Each page contains a series of connections to other pages called hyperlinks. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are delineated by tags, written using angle brackets. Tags such as and <input/> introduce content into the page directly. Others such as <p>....</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look. Without HTML, a browser would not know how to display text as elements or load images or other elements. HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance.

2. CSS:

CSS (Cascading Style Sheets) is a style sheet language used for describing the presentation of a document written in HTML (Hypertext Markup Language). It is used to style and layout web pages, including those that interact with a database management system (DBMS). Here are some ways CSS can be used in building web applications that interact with a DBMS:

Styling User Interface Components: CSS can be used to style various user interface components in web applications that interact with a DBMS, such as forms, tables, and charts. By using CSS, developers can customize the look and feel of these components to match the branding and visual design of the application.

3. Responsive Design:

CSS can be used to create responsive designs that adapt to different screen sizes and device types. This is especially important in web applications that interact with a DBMS, as users may need to access the application from different devices with varying screen sizes.

4. Php:

PHP is an acronym for "PHP: Hypertext Preprocessor". It is a widely-used, open source scripting language. Its scripts are executed on the server. It is free to download and use. It can generate dynamic page content. It can create, open, read, write, delete, and close files on the server. It can collect form data. It can send and receive cookies. It can add, delete, modify data in your database. It can be used to control user-access. It can encrypt data.

5. MySQL:

Databases are the essential data repository for all software applications. For example, whenever someone conducts a web search, logs in to an account, or completes a transaction, a database system is storing the information so it can be accessed in the future.

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structure is organized into physical files optimized for speed. The logical data model, with objects such as data tables, views, rows, and columns, offers a flexible programming environment.

The "SQL" part of "MySQL" stands for "Structured Query Language." SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

6. Javascript:

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers

Showing rows 0 - 6 (7 total. Query took 0.0004 seconds.)

```
SELECT * FROM `ib_bankaccounts`
```

Number of rows: 25 Filter rows: Search this table Sort by key: None

	account_id	acc_name	account_number	acc_type	acc_rates	acc_status	acc_amount	client_id	client_name	client_national_id	client_phone	client_number	client_email
<input type="checkbox"/> Edit Copy Delete	13	Christine Moore	421873905	Current account	20	Active	0	4	Christine Moore	478545445812	7785452210	iBank-CLIENT-9501	christine@mail.com
<input type="checkbox"/> Edit Copy Delete	14	Harry M Den	357146928	Savings	20	Active	0	5	Harry Den	100014001000	7412560000	iBank-CLIENT-7014	harryden@mail.com
<input type="checkbox"/> Edit Copy Delete	15	Amanda Stiefel	287359614	Savings	20	Active	0	8	Amanda Stiefel	4780000001	7850000014	iBank-CLIENT-0423	amanda@mail.com
<input type="checkbox"/> Edit Copy Delete	16	Johnnie Reyes	705239816	Retirement	10	Active	0	6	Johnnie J. Reyes	147455554	7412545454	iBank-CLIENT-1698	reyes@mail.com
<input type="checkbox"/> Edit Copy Delete	17	Liam M. Moore	719360482	Savings	20	Active	0	9	Liam Moore	170014695	7014569696	iBank-CLIENT-4716	liamoore@mail.com
<input type="checkbox"/> Edit Copy Delete	18	Johnny M. Doen	724310586	Fixed Deposit Account	40	Active	0	3	John Doe	36756481	9897890089	iBank-CLIENT-8127	johndoe@gmail.com
<input type="checkbox"/> Edit Copy Delete	19	mysaving	972468301	Savings	20	Active	0	10	Adit Agrawal	27423737	9977443900	iBank-CLIENT-9243	aa@smit.edu.in

Check all With selected: Edit Copy Delete Export

Fig 3.10 PHP Database

The following image represents the backend connectivity to MySQL database wherein the user's information is stored for future reference. From the track gateway page the Backend gets connected where we use MySQL and PHP to store the track history and all details about the users.

1. Storing customer information:

MySQL can be used to store customer information, such as name, sender's address, and receiver's address. This information can be encrypted and secured using MySQL security features to protect customer data from unauthorized access.

2. Recording transactions details:

MySQL can be used to record transaction details such as checking whether the transaction is successful or not. This data can be used for auditing and reporting purposes.

3. PHP:

PHP is used for the backend and can connect to the database very easily. This is because PHP is designed with specific capabilities to facilitate these connections in an efficient way. It's no surprise the PHP is part of the widely popular LAMP stack.

4. Xampp-SQL Server--Apache Server:

XAMPP is used to symbolize the classification of solutions for different technologies. It provides a base for testing of projects based on different technologies through a personal server. This collection of software contains a web server named Apache, a database management system named MariaDB and scripting/ programming languages such as PHP and Perl. X denotes Crossplatform, which means that it can work on different platforms such as Windows, Linux, and macOS.

3.2 ER DIAGRAM AND USE CASE DIAGRAM

ER DIAGRAM

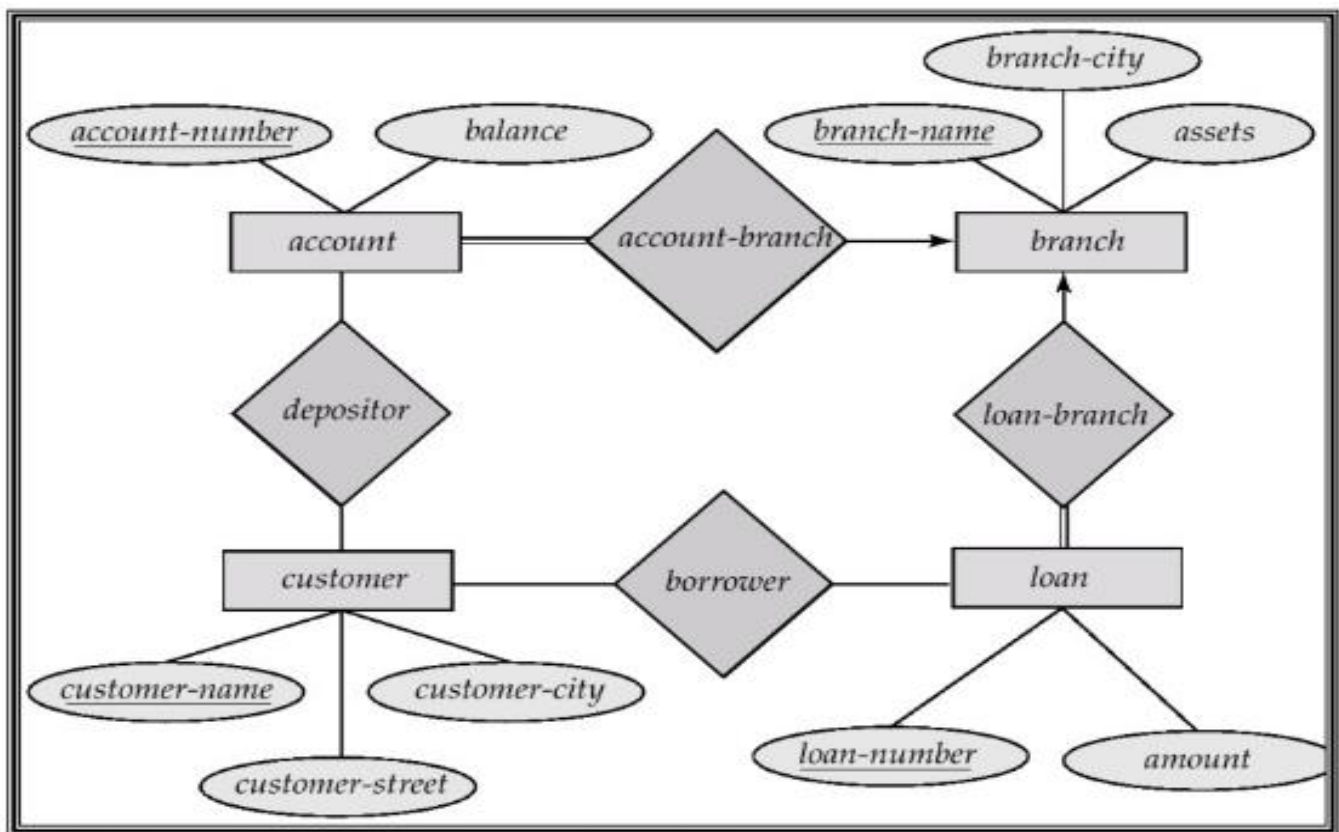


Fig 3.11 ER Diagram

In the ER diagram each bank has attributes like Name, code and address and has functions i.e. it can have many branches which further have branch id's, branch name and address.

An Entity Relationship (ER) diagram for an online banking system is a visual representation of the relationships between the various entities in the system. Entities can include customers, accounts, transactions, and other relevant components.

An entity-relationship (ER) diagram for an online banking system typically includes several entities, including Customer, Account, Transaction, and Bank. Here is a brief explanation of each entity and its relationships:

1. **Customer Entity:** This entity represents the customers who have accounts with the bank. It includes attributes such as Customer ID, Name, Address, Phone, and Email.
2. **Account Entity:** This entity represents the different types of accounts offered by the bank, such as Checking, Savings, or Investment accounts. It includes attributes such as Account Number, Account Type, Balance, and Interest Rate.
3. **Transaction Entity:** This entity represents the different types of transactions that can be performed by customers, such as deposits, withdrawals, transfers, or payments. It includes attributes such as Transaction ID, Amount, Date, and Type.
4. **Bank Entity:** This entity represents the bank itself, including its name, address, and other details.

The ER diagram typically shows the relationships between these entities, which are as follows:

- One customer can have many accounts, but each account belongs to only one customer (a one-to-many relationship).
- Each account can have many transactions, but each transaction belongs to only one account (a one-to-many relationship).
- Each transaction is associated with one account, and each account can have many transactions (a many-to-one relationship).
- The bank entity is not directly related to the other entities but may be linked to the customer entity through a separate relationship (e.g., a customer may have accounts at multiple branches of the bank).

Overall, the ER diagram provides a visual representation of the relationships between different entities in the online banking system, helping to ensure that the system is designed and implemented in a logical and efficient manner.

USE CASE DIAGRAM

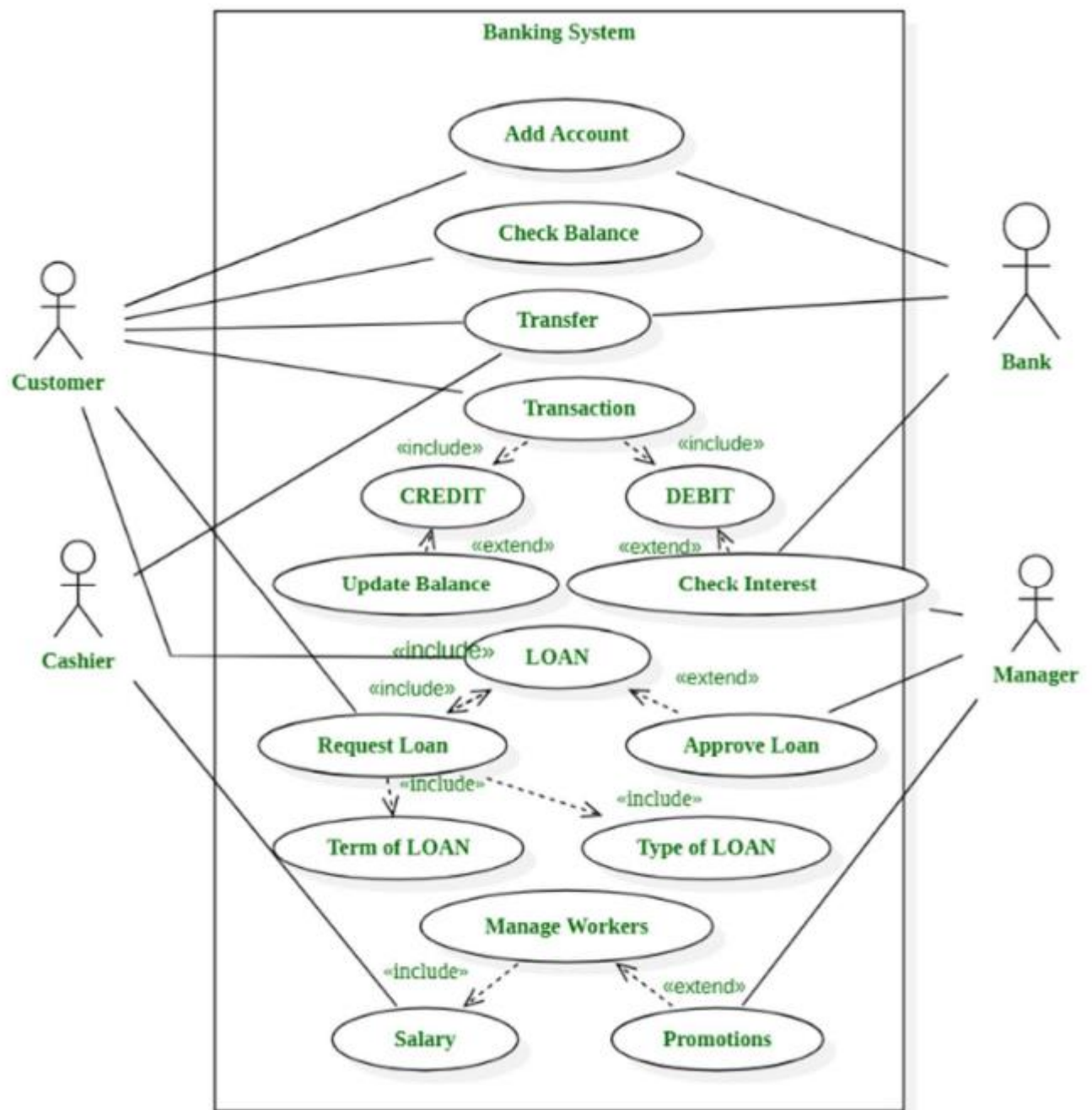


Fig 3.12 UseCase Diagram

A use case diagram is a visual representation of the different actions that can be performed by various actors in a system. In the context of an online banking system, the use case diagram outlines the different actions that customers, bank staff, and administrators can perform.

The diagram shows the various use cases that are available to the different actors in the system. The customer, bank staff, and administrator are the three main actors in the system, and each of them can perform different actions.

The customer actor can perform various actions, such as logging in, viewing account balance, transferring funds, and making payments. They can also update their profile information, view transaction history, and access help and support.

The bank staff actor can perform a range of actions, including accessing customer information, viewing account details, approving or rejecting loan requests, and managing customer accounts. They can also generate reports, manage system settings, and perform other administrative tasks.

The administrator actor has access to more advanced features, such as managing system settings, viewing logs and reports, and managing user accounts. They can also monitor system performance and security, and ensure that the system is running smoothly.

Some of the use cases shown in the diagram include:

- 1. Log in:** This use case allows the customer, bank staff, and administrator to log in to the system using their credentials.
- 2. Transfer funds:** This use case allows the customer to transfer funds between their own accounts or to other accounts.
- 3. Make payments:** This use case enables the customer to make payments to other individuals or businesses. **View transaction history:** This use case allows the customer to view their transaction history, which provides them with a detailed record of their financial activity.
- 4. Approve or reject loan requests:** This use case enables the bank staff to review loan requests and either approve or reject them based on the customer's financial position.
- 5. Generate reports:** This use case allows the bank staff to generate reports on customer activity, account balances, and other system data.
- 6. Manage user accounts:** This use case enables the administrator to manage user accounts, such as creating new accounts, modifying existing accounts, and deleting accounts.

Overall, the use case diagram provides a clear overview of the different actions that can be performed by the actors in the online banking system. It serves as a useful tool for understanding the functionality of the system and for designing and testing the system.

CHAPTER 4

MODULES AND FUNCTIONALITIES

4.1 LIST OF MODULES

- **User Registration and Authentication:** This module includes user registration, login, and password recovery. It also includes multi-factor authentication features such as SMS or email verification to enhance security.
- **Account Management:** This module allows users to manage their accounts, including viewing their account balance, transaction history, and statements. Users can also update their personal information, such as their address or phone number.
- **Fund Transfers:** This module enables users to transfer funds between their own accounts, to other users' accounts within the same bank, or to external accounts in other banks. Users can also schedule recurring transfers or set up standing orders.
- **Deposit and Withdrawals:** This module enables users to deposit or withdraw money from their accounts. They can also set up automatic deposit or withdrawal instructions.
- **Customer Support:** This module includes customer support features such as online chat, email support, or phone support. It also includes FAQs and knowledge base articles.
- **Security:** This module includes security features such as firewalls, encryption, and fraud prevention. It also includes monitoring tools to detect suspicious activity and alert the appropriate authorities.
- **Mobile Banking:** This module includes mobile apps for iOS and Android devices. Users can access their accounts, make transactions, and receive alerts on-the-go.
- Overall, these modules work together to provide a comprehensive online banking system that allows users to manage their finances securely and efficiently.

4.2 CONNECTIVITY USED FOR DATABASE ACCESS

4.2.1. MySQL

MySQL is a highly popular database management system that can power projects of all sizes. Its ability to handle huge volumes of data without breaking a sweat is one of its biggest selling points. Connect MySQL with PHP code, you can make use of one of three methodologies.

1. Create MySQL Database at the Localhost
 - 1.1. Create Database
 - 1.2. Create a Folder in htdocs
 - 1.3. Create Database Connection File In PHP
 - 1.4. Create new php file to check your database connection
 - 1.5. Run it
2. Create MySQL Database at Cloudways Server
 - 2.1. Create Database Connection
 - 2.2. MySQLi Procedural Query
 - 2.3. Connect MySQL Database with PHP Using PDO
 - 2.4. Check Connection
3. Remote MySQL
4. Top MySQL Management tools
 - 4.1. MySQL Workbench
 - 4.2. Navicat For MySQL
 - 4.3. MySQL Yog
 - 4.4. Cloudways MySQL Database Manager

CHAPTER 5

CODING AND TESTING

Admin Login

```
pages_index.php X
C: > Users > soura > OneDrive > Desktop > InternetBanking-PHP > admin > pages_index.php
1  <?php
2  session_start();
3  include('conf/config.php'); //get configuration file
4  if (isset($_POST['login'])) {
5      $email = $_POST['email'];
6      $password = sha1(md5($_POST['password'])); //double encrypt to increase security
7      $stmt = $mysqli->prepare("SELECT email, password, admin_id FROM ib_admin WHERE email=? AND password=?"); //sql to log in user
8      $stmt->bind_param('ss', $email, $password); //bind fetched parameters
9      $stmt->execute(); //execute bind
10     $stmt->bind_result($email, $password, $admin_id); //bind result
11     $rs = $stmt->fetch();
12     $_SESSION['admin_id'] = $admin_id; //assign session to admin id
13     //$_uiip=$_SERVER['REMOTE_ADDR'];
14     //$_ldate=date('d/m/Y h:i:s', time());
15     if ($rs) { //if its successfull
16         header("location:pages_dashboard.php");
17     } else {
18         #echo "<script>alert('Access Denied Please Check Your Credentials');</script>";
19         $err = "Access Denied Please Check Your Credentials";
20     }
21 }
22
23 /* Persist System Settings On Brand */
24 $ret = "SELECT * FROM `ib_SystemSettings` ";
25 $stmt = $mysqli->prepare($ret);
26 $stmt->execute(); //ok
27 $res = $stmt->get_result();
28 while ($auth = $res->fetch_object()) {
29     ?>
30     <!-- Log on to codeastro.com for more projects! -->
31     <!DOCTYPE html>
32     <html>
33     <meta http-equiv="content-type" content="text/html; charset=utf-8" />
34     <?php include("dist/_partials/head.php"); ?>
35
36     <body class="hold-transition login-page">
37         <div class="login-box">
38             <div class="login-logo">
39                 <p><?php echo $auth->sys_name; ?></p>
40             </div>
41             <!-- /.login-logo -->
42             <div class="card">
43                 <div class="card-body login-card-body">
44                     <p class="login-box-msg">Log In To Start Administrator Session</p>
45                 </div>
46             </div>
47         </div>
48     </body>
49 </html>
50 <!-- Log on to codeastro.com for more projects! -->
51 <!-- Doctype html -->
52 <html>
53 <meta http-equiv="content-type" content="text/html; charset=utf-8" />
54 <?php include("dist/_partials/head.php"); ?>
55
56 <body class="hold-transition login-page">
57     <div class="login-box">
58         <div class="login-logo">
59             <p><?php echo $auth->sys_name; ?></p>
60         </div>
61         <!-- /.login-logo -->
62         <div class="card">
63             <div class="card-body login-card-body">
64                 <p class="login-box-msg">Log In To Start Administrator Session</p>
65             </div>
66         </div>
67     </div>
68 </body>
69 </html>
70 <!-- Log on to codeastro.com for more projects! -->
```



```

45
46 <form method="post">
47   <div class="input-group mb-3">
48     <input type="email" name="email" class="form-control" placeholder="Email">
49     <div class="input-group-append">
50       <div class="input-group-text">
51         <span class="fas fa-envelope"></span>
52       </div>
53     </div>
54   </div>
55   <div class="input-group mb-3">
56     <input type="password" name="password" class="form-control" placeholder="Password">
57     <div class="input-group-append">
58       <div class="input-group-text">
59         <span class="fas fa-lock"></span>
60       </div>
61     </div>
62   </div>
63   <div class="row">
64     <div class="col-4">
65       <div class="icheck-primary">
66         <!-- <input type="checkbox" id="remember">
67         <label for="remember">
68           Remember Me
69         </label> -->
70       </div>
71     </div>
72     <!-- /.col -->
73     <div class="col-8">
74       <button type="submit" name="login" class="btn btn-danger btn-block">Log In as Admin</button>
75     </div>
76     <!-- /.col -->
77   </div>
78 </form>
79
80
81 <!-- /.social-auth-links -->
82
83 <!-- <p class="mb-1">
84   <a href="pages_reset_pwd.php">I forgot my password</a>
85 </p> -->
86 <!--
87 Uncomment this line to allow account creations for admins
88
89 <p class="mb-0">
90   <a href="pages_signup.php" class="text-center">Register a new membership</a>
91 </p>
92 -->
93 </div>

```

Client Registration Page

C: > Users > soura > OneDrive > Desktop > InternetBanking-PHP > client > pages_dashboard.php

```
1  <?php
2  session_start();
3  include('conf/config.php');
4  include('conf/checklogin.php');
5  check_login();
6  $client_id = $_SESSION['client_id'];
7
8  /*
9      get all dashboard analytics
10     and numeric values from distinct
11     tables
12     */
13
14     //return total number of ibank clients
15     $result = "SELECT count(*) FROM ib_clients";
16     $stmt = $mysqli->prepare($result);
17     $stmt->execute();
18     $stmt->bind_result($iBClients);
19     $stmt->fetch();
20     $stmt->close();
21
22     //return total number of iBank Staffs
23     $result = "SELECT count(*) FROM ib_staff";
24     $stmt = $mysqli->prepare($result);
25     $stmt->execute();
26     $stmt->bind_result($iBStaffs);
27     $stmt->fetch();
28     $stmt->close();
29
30     //return total number of iBank Account Types
31     $result = "SELECT count(*) FROM ib_Acc_types";
32     $stmt = $mysqli->prepare($result);
33     $stmt->execute();
34     $stmt->bind_result($iB_AccsType);
35     $stmt->fetch();
36     $stmt->close();
37
38     //return total number of iBank Accounts
39     $result = "SELECT count(*) FROM ib_bankAccounts";
40     $stmt = $mysqli->prepare($result);
41     $stmt->execute();
42     $stmt->bind_result($iB_Accs);
43     $stmt->fetch();
44     $stmt->close();
```

```

46 //return total number of iBank Deposits
47 $client_id = $_SESSION['client_id'];
48 $result = "SELECT SUM(transaction_amt) FROM iB_Transactions WHERE client_id = ? AND tr_type = 'Deposit' ";
49 $stmt = $mysqli->prepare($result);
50 $stmt->bind_param('i', $client_id);
51 $stmt->execute();
52 $stmt->bind_result($iB_deposits);
53 $stmt->fetch();
54 $stmt->close();
55
56 //return total number of iBank Withdrawals
57 $client_id = $_SESSION['client_id'];
58 $result = "SELECT SUM(transaction_amt) FROM iB_Transactions WHERE client_id = ? AND tr_type = 'Withdrawal' ";
59 $stmt = $mysqli->prepare($result);
60 $stmt->bind_param('i', $client_id);
61 $stmt->execute();
62 $stmt->bind_result($iB_withdrawal);
63 $stmt->fetch();
64 $stmt->close();
65
66
67
68 //return total number of iBank Transfers
69 $client_id = $_SESSION['client_id'];
70 $result = "SELECT SUM(transaction_amt) FROM iB_Transactions WHERE client_id = ? AND tr_type = 'Transfer' ";
71 $stmt = $mysqli->prepare($result);
72 $stmt->bind_param('i', $client_id);
73 $stmt->execute();
74 $stmt->bind_result($iB_Transfers);
75 $stmt->fetch();
76 $stmt->close();
77
78 //return total number of iBank initial cash->balances
79 $client_id = $_SESSION['client_id'];
80 $result = "SELECT SUM(transaction_amt) FROM iB_Transactions WHERE client_id = ?";
81 $stmt = $mysqli->prepare($result);
82 $stmt->bind_param('i', $client_id);
83 $stmt->execute();
84 $stmt->bind_result($acc_amt);
85 $stmt->fetch();
86 $stmt->close();
87 //Get the remaining money in the accounts
88 $TotalBalInAccount = ($iB_deposits) - (($iB_withdrawal) + ($iB_Transfers));
89
90

```

```

90
91 //ibank money in the wallet
92 $client_id = $_SESSION['client_id'];
93 $result = "SELECT SUM(transaction_amt) FROM iB_Transactions WHERE client_id = ?";
94 $stmt = $mysqli->prepare($result);
95 $stmt->bind_param('i', $client_id);
96 $stmt->execute();
97 $stmt->bind_result($new_amt);
98 $stmt->fetch();
99 $stmt->close();
100 //Withdrawal Computations
101
102 ?>
103
104 <!DOCTYPE html>
105 <html lang="en">
106 <meta http-equiv="content-type" content="text/html; charset=utf-8" />
107 <?php include("dist/_partials/head.php"); ?>
108 <!-- Log on to codeastro.com for more projects! -->
109 <body class="hold-transition sidebar-mini layout-fixed layout-navbar-fixed">
110
111     <div class="wrapper">
112         <!-- Navbar -->
113         <?php include("dist/_partials/nav.php"); ?>
114         <!-- /.navbar -->
115
116         <!-- Main Sidebar Container -->
117         <?php include("dist/_partials/sidebar.php"); ?>
118
119         <!-- Content Wrapper. Contains page content -->
120         <div class="content-wrapper">
121             <!-- Content Header (Page header) -->
122             <div class="content-header">
123                 <div class="container-fluid">
124                     <div class="row mb-2">
125                         <div class="col-sm-6">
126                             <h1 class="m-0 text-dark">Client Dashboard</h1>
127                         </div><!-- /.col -->
128                         <div class="col-sm-6">
129                             <ol class="breadcrumb float-sm-right">
130                                 <li class="breadcrumb-item"><a href="#">Home</a></li>
131                                 <li class="breadcrumb-item active">Dashboard</li>
132                             </ol>
133                         </div><!-- /.col -->
134                     </div><!-- /.row -->
135                 </div><!-- /.container-fluid -->
136             </div>
137             <!-- /.content-header -->

```

Dashboard page

```
pages_deposits.php X
C: > Users > soura > OneDrive > Desktop > InternetBanking-PHP > client > pages_deposits.php
1  <?php
2  session_start();
3  include('conf/config.php');
4  include('conf/checklogin.php');
5  check_login();
6  $client_id = $_SESSION['client_id'];
7  ?>
8  <!-- Log on to codeastro.com for more projects! -->
9  <!DOCTYPE html>
10 <html>
11 <meta http-equiv="content-type" content="text/html; charset=utf-8" />
12 <?php include("dist/_partials/head.php"); ?>
13
14 <body class="hold-transition sidebar-mini layout-fixed layout-navbar-fixed">
15 <div class="wrapper">
16     <!-- Navbar -->
17     <?php include("dist/_partials/nav.php"); ?>
18     <!-- /.navbar -->
19
20     <!-- Main Sidebar Container -->
21     <?php include("dist/_partials/sidebar.php"); ?>
22
23     <!-- Content Wrapper. Contains page content -->
24     <div class="content-wrapper">
25         <!-- Content Header (Page header) -->
26         <section class="content-header">
27             <div class="container-fluid">
28                 <div class="row mb-2">
29                     <div class="col-sm-6">
30                         <h1>Deposits</h1>
31                     </div>
32                     <div class="col-sm-6">
33                         <ol class="breadcrumb float-sm-right">
34                             <li class="breadcrumb-item"><a href="pages_dashboard.php">Dashboard</a></li>
35                             <li class="breadcrumb-item"><a href="pages_deposits">iBank Finances</a></li>
36                             <li class="breadcrumb-item active">Deposits</li>
37                         </ol>
38                     </div>
39                 </div>
40             </div><!-- /.container-fluid -->
41         </section>
42     </div>
```

```

42 |
43 | <!-- Main content -->
44 | <section class="content">
45 |   <div class="row">
46 |     <div class="col-12">
47 |       <div class="card">
48 |         <div class="card-header">
49 |           <h3 class="card-title">Select on any account to deposit money</h3>
50 |         </div>
51 |         <div class="card-body">
52 |           <table id="example1" class="table table-bordered table-hover table-striped">
53 |             <thead>
54 |               <tr>
55 |                 <th>#</th>
56 |                 <th>Name</th>
57 |                 <th>Account No.</th>
58 |                 <th>Rate</th>
59 |                 <th>Acc. Type</th>
60 |                 <th>Acc. Owner</th>
61 |                 <th>Action</th>
62 |               </tr>
63 |             </thead>
64 |             <tbody>
65 |               <?php
66 |                 //fetch all IB Accs
67 |                 $client_id = $_SESSION['client_id'];
68 |                 $ret = "SELECT * FROM ib_bankAccounts WHERE client_id = ?";
69 |                 $stmt = $mysqli->prepare($ret);
70 |                 $stmt->bind_param('i', $client_id);
71 |                 $stmt->execute(); //ok
72 |                 $res = $stmt->get_result();
73 |                 $cnt = 1;
74 |                 while ($row = $res->fetch_object()) {
75 |                   //Print Timestamp to DD-MM-YYYY : H-H-S
76 |                   $dateOpened = $row->created_at;
77 |
78 |                 }
79 |
80 |               <tr>
81 |                 <td><?php echo $cnt; ?></td>
82 |                 <td><?php echo $row->acc_name; ?></td>
83 |                 <td><?php echo $row->account_number; ?></td>
84 |                 <td><?php echo $row->acc_rates; ?>%</td>
85 |                 <td><?php echo $row->acc_type; ?></td>
86 |                 <td><?php echo $row->client_name; ?></td>
87 |                 <td>
88 |                   <a class="btn btn-success btn-sm" href="pages_deposit_money.php?account_id=<?php echo $row->account_id; ?>&account_number=<?php echo $row->account_number; ?>&client_id=<?php echo $row->client_id; ?>">

```

Home Page

```

index.php X
C: > Users > soura > OneDrive > Desktop > InternetBanking-PHP > index.php
1  <?php
2  include("admin/conf/config.php");
3  /* Persisit System Settings On Brand */
4  $ret = "SELECT * FROM `ib_SystemSettings` ";
5  $stmt = $mysqli->prepare($ret);
6  $stmt->execute(); //ok
7  $res = $stmt->get_result();
8  while ($sys = $res->fetch_object()) {
9  }
10
11 <!DOCTYPE html>
12 <html lang="en">
13 <meta http-equiv="content-type" content="text/html; charset=utf-8" />
14
15 <head>
16   <meta charset="utf-8">
17   <meta http-equiv="X-UA-Compatible" content="IE=edge">
18   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
19   <meta name="description" content="">
20   <meta name="author" content="">
21   <title><?php echo $sys->sys_name; ?> - <?php echo $sys->sys_tagline; ?></title>
22   <link href="dist/css/robust.css" rel="stylesheet">
23 </head>
24
25 <body>
26
27   <nav class="navbar navbar-lg navbar-expand-lg navbar-transparent navbar-dark navbar-absolute w-100">
28     <div class="container">
29       <a class="navbar-brand" href="index.php"><?php echo $sys->sys_name; ?></a>
30       <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-label="Toggle navigation">
31         <span class="navbar-toggler-icon"></span>
32       </button>
33
34       <div class="collapse navbar-collapse" id="navbarCollapse">
35         <ul class="navbar-nav ml-auto">
36           <li class="nav-item active">
37             <a class="nav-link" target="_blank" href="admin/pages_index.php">Admin Portal</a>
38           </li>
39           <li class="nav-item active">
40             <a class="nav-link" target="_blank" href="staff/pages_staff_index.php">Staff Portal</a>
41           </li>
42           <li class="nav-item active">
43             <a class="nav-link" target="_blank" href="client/pages_client_index.php">Client Portal</a>
44           </li>
45         </ul>
46       </div>
47     </div>
48   </nav>

```

```

48     </nav>
49
50     <div class="intro py-5 py-lg-9 position-relative text-white">
51         <div class="bg-overlay-gray">
52             
53         </div>
54         <div class="intro-content py-6 text-center">
55             <div class="container">
56                 <div class="row align-items-center">
57                     <div class="col-12 col-sm-10 col-md-8 col-lg-6 mx-auto text-center">
58                         <h1 class="my-3 display-4 d-none d-lg-inline-block"><?php echo $sys->sys_name; ?></h1>
59                         <p class="lead mb-3">
60                             <?php echo $sys->sys_tagline; ?>
61                         </p>
62                         <br>
63                         <a class="btn btn-success btn-lg mr-lg-2 my-1" target="_blank" href="client/pages_client_signup.php" role="button">Get started</a>
64                     </div>
65                 </div>
66             </div>
67         </div>
68     </div>
69     <script src="dist/js/bundle.js"></script>
70 </body>
71
72 </html>
73 <?php
74 } ?>

```

```

?>

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
    <style>
        body {
            background-image: url('images/10.jpg');
            background-repeat: no-repeat;
            background-size: cover;
        }
    </style>
</head>

<body>
    <h1 align='center' style="margin: 15px; color:seagreen;font-weight: bold;font-family:'Times New Roman', Times, serif">TYPHOON COURIER SERVICE</h1>
    <hr />
    <p align='center' style="font-weight: bold;color:orange;font-family:'Times New Roman', Times, serif">The Fastest Courier Service Ever</p>
    <div>
        <h5><a href="admin/adminlogin.php" style="float: right; margin-right:40px; color:blue; margin-top:0px">AdminLogin</a></h5>
    </div>
    <div class="container" style="margin-top: 60px; width:50%;">
        <div class="row">
            <div class="col-md-12">
                <h2 style="color: #273c75;">Login</h2>
                <p style="color:#e84118;">Please Fill Your !!</p>
                <!-- <?php echo $error; ?> -->
                <form action="" method="post">
                    <div class="form-group">
                        <label>Email Address</label>

```

```

<div>
  <h5><a href="admin/adminlogin.php" style="float: right; margin-right:40px; color:blue; margin-top:0px">AdminLogin</a></h5>
</div>
<div class="container" style="margin-top: 60px; width:50%;">
  <div class="row">
    <div class="col-md-12">
      <h2 style="color: #273c75;">Login</h2>
      <p style="color:#e84118;">Please Fill Your !!</p>
      <!-- <?php echo $error; ?> -->
      <form action="" method="post">
        <div class="form-group">
          <label>Email Address</label>
          <input type="email" name="email" class="form-control" placeholder="Enter username/emailId" required />
        </div>
        <div class="form-group">
          <label>Password</label>
          <input type="password" name="password" class="form-control" placeholder="Enter your password" required>
        </div>
        <div class="form-group">
          <input type="submit" name="submit" class="btn btn-primary" value="signIn" />
          <button onclick="window.location.href='resetpswd.php'" class="btn btn-danger" style="cursor:pointer">Reset Password</button>
        </div>
        <p style="color: #e84118;">Don't have an account?>>> <a href="register.php">Register here</a>.</p>
      </form>
    </div>
  </div>
</div>
</body>

</html>

```


CHAPTER 6

RESULTS AND DISCUSSIONS

6.1 Results

- Increased efficiency and convenience for bank customers: The online banking system can enable customers to access their accounts and perform transactions quickly and easily, without having to visit a physical branch.
- Improved security: By implementing multi-factor authentication, encryption, and other security measures, the online banking system can enhance the security of customer information and prevent fraud and unauthorized access.
- Reduced operational costs: With fewer customers visiting physical branches, the bank can reduce its operational costs related to staffing and maintenance.
- Improved customer satisfaction and loyalty: A well-designed and user-friendly online banking system can enhance customer satisfaction and loyalty, leading to increased retention and word-of-mouth referrals.
- Valuable insights into customer behavior: By tracking customer transactions and interactions with the online banking system, the bank can gain valuable insights into customer behavior and preferences, which can inform future product and service offerings.

6.2 DISCUSSIONS

At the beginning of the project we had set out certain objectives which ought to be accomplished by the end of the system creation, at the end we were able to achieve most of the objectives set out

Overview of the project :

The online banking system mini project is focused on the development and implementation of a web-based platform that allows bank customers to access their accounts and perform transactions over the internet. The objective of the project is to provide customers with a more convenient and efficient banking experience while also improving security and reducing operational costs for the bank. The project includes designing and implementing a database to store customer and transaction information, developing a user-friendly web interface for customers to interact with the system, and implementing security measures to protect customer information and prevent fraud.

Developed a user friendly interface :

- **Consistency:** The user interface should be consistent across all pages and functions, with a clear and recognizable design style, color scheme, and typography.
- **Simplicity:** The interface should be simple and intuitive, with a clear hierarchy of information and a minimal number of steps required to perform tasks.
- **Navigation:** Navigation should be easy and straightforward, with a clear and accessible menu structure and easily recognizable icons or buttons.
- **Responsiveness:** The interface should be responsive to different screen sizes and devices, adapting to different resolutions and input methods.

Developed a back-end system only accessible to the administrator

This system incorporates a database by the name multi_login. The database stores information of the users in the user table, and the changes that the admin makes at any given time.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

In conclusion, the development of an online banking system provides a range of benefits for both customers and banks. Customers are able to perform transactions more conveniently and efficiently, while banks can reduce operational costs and gain valuable insights into customer behavior and preferences. The proposed system includes a user-friendly web interface, a secure database, and a range of features to meet the needs of both customers and banks. The system is designed to be scalable, adaptable, and responsive to changing user needs and technological advancements.

While there may be challenges and obstacles along the way, the development of an online banking system is an important step towards providing a more efficient, convenient, and secure banking experience for customers. By utilizing modern technologies and design principles, banks can stay competitive and attract and retain customers in a rapidly changing financial landscape.

7.2 FUTURE ENHANCEMENT

- **Mobile App:** Develop a mobile app to make it even more convenient for customers to access their accounts on-the-go.
- **AI-powered chatbots:** Implement chatbots to provide 24/7 customer support, answer frequently asked questions, and handle routine transactions.
- **Advanced fraud detection:** Integrate advanced machine learning algorithms to detect and prevent fraudulent activities in real-time.
- **Personalized recommendations:** Utilize customer data to provide personalized recommendations for financial products and services.
- **Virtual assistants:** Develop virtual assistants to provide personalized financial advice, track spending, and manage savings.
- **Biometric authentication:** Implement biometric authentication methods such as facial recognition or fingerprint scanning for added security.
- **Voice-activated banking:** Allow customers to use voice commands to perform transactions, check account balances, and make payments.

- **Integration with other financial institutions:** Enable customers to view and manage accounts from multiple financial institutions in one place.
- **Blockchain technology:** Explore the use of blockchain technology to provide secure, decentralized, and transparent transactions.

These are just a few potential future enhancements to an online banking system. The specific enhancements will depend on the needs and priorities of the bank and its customers, as well as the availability of technology and resources.

REFERENCES

1. "Online Banking System" by Jignesh Vaghela and Harshida Patel: This research paper outlines the design and implementation of an online banking system, including modules such as user registration, account management, and fund transfers
2. "Building a Better Online Banking Experience" by Forrester Research: This report provides insights into how banks can improve their online banking systems to provide a better customer experience. It includes case studies and best practices from leading banks.
3. "Secure Online Banking" by the National Institute of Standards and Technology: This publication provides guidelines for secure online banking, including authentication, access control, and data protection.
4. "Mobile Banking: Security and Privacy Issues" by the Federal Reserve Bank of Boston: This report discusses the security and privacy risks associated with mobile banking and provides recommendations for mitigating those risks.
5. Online documentation and resources provided by leading banking software providers, such as Fiserv, Temenos, and FIS. These resources can include best practices, case studies, and technical documentation for building online banking systems
6. "Online Banking System: A Comprehensive Review." International Journal of Advanced Computer Science and Applications, vol. 9, no. 5, 2018.
7. "Secure Online Banking System: An Overview." International Journal of Scientific & Engineering Research, vol. 5, no. 11, 2014.

8. "Design and Implementation of a Secure Online Banking System." *International Journal of Emerging Trends & Technology in Computer Science*, vol. 3, no. 1, 2014.
9. "Online Banking System: A Comparative Analysis." *International Journal of Computer Science and Mobile Computing*, vol. 7, no. 5, 2018.
10. "Evaluating the Usability of Online Banking Systems: A Case Study." *Journal of Computer Science and Technology*, vol. 34, no. 1, 2019.
11. "Online Banking System: Security and Risk Management." *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 10, 2013.
12. "Design and Implementation of a Web-based Online Banking System." *International Journal of Computer Science Issues*, vol. 9, no. 3, 2012.
13. "A Framework for Evaluating Online Banking Websites." *Journal of Internet Banking and Commerce*, vol. 22, no. 3, 2017.
14. "Adoption of Mobile Banking: A Literature Review." *International Journal of Business and Management*, vol. 12, no. 5, 2017.
15. "Online Banking: Factors Influencing Customer Adoption in the UK." *Journal of Retailing and Consumer Services*, vol. 20, 2013.
16. "Factors Influencing the Adoption of Online Banking in Indonesia." *Journal of Business and Retail Management Research*, vol. 7, no. 3, 2013.
17. "The Impact of E-Service Quality on Online Banking Usage." *Journal of Service Theory and Practice*, vol. 25, no. 1, 2015..