VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JNANA SANGAMA", BELAGAVI-59001

A
Mini Project Report
On

BANKING MANAGEMENT SYSTEM

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT IN 5TH SEMESTER,

DATABASE MANAGEMENT SYSTEM LAB(15CSL58)
IN

COMPUTER SCIENCE AND ENGINEERING

By

PIYUSH RANJAN [1JB16CS102]

UNDER THE GUIDANCE OF

Mr.DEEPAK B.L
Asst.Prof.
Dept. of CSE,SJBIT

Mr.LOCHAN GOWDA
Asst. Prof.
Dept. of CSE, SJBIT

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

S.J.B INSTITUTE OF TECHNOLOGY

B G S HEALTH AND EDUCATION CITY
Kengeri-Bengaluru-560060
2018-2019

|| Jai Sri Gurudev ||
Sri Adichunchanagiri Shikshana Trust ®

SJB INSTITUTE OF TECHNOLOGY

BGS Health & Education City, Kengeri, Bengaluru – 560 060

Department of Computer Science & Engineering

CERTIFICATE

Certified that the Mini project work entitled "**BANKING MANAGEMENT SYSTEM**" carried out by **PIYUSH RANJAN** bearing USN [**1JB16CS102]** is bonafide student of **S J B Institute of Technology** in partial fulfilment of mini project of 5th semester, DATABASE APPLICATION LAB(15CSL58) in **COMPUTER SCIENCE** as prescribed by **VISVESVARAYA TECHNOLOGICAL UNIVERSITY,BELAGAVI** during the academic year **2018-2019 .** It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The Mini project report has been approved as it satisfies the academic requirements in respect of Mini Project prescribed for the said degree.

Mr.Deepak B.L **Asst. Professor** **Dept. Of CSE**	**Mr.Lochan Gowda** **Asst. Professor** **Dept. of CSE**	**Dr. Krishna A.N** **Professor & Head of** **Dept. Of CSE,SJBIT**

EXTERNAL EXAM

Name of Examiners	**Signature with Date**
1. _____	_____
2. _____	_____

ACKNOWLEDGEMENT

I would like to express my gratitude to His Divine Soul **Jagadguru Padmabhushan Sri Sri Sri Dr. Balagangadharanatha MahaSwamiji** and His Holiness **Jagadguru Sri Sri Sri Dr. Nirmalanandanatha MahaSwamiji** for providing me an opportunity to pursue my academics in this esteemed institution.

I would like to express my profound thanks to **Reverend Sri Sri Sri Dr.Prakashnath Swamiji**, Managing Director, SJB Institute of Technology, for his continuous support in providing amenities to carry out this mini project in this admired institution.

I express my gratitude to **Dr. Puttaraju**, Principal, SJB Institute of Technology, for providing an excellent facilities and academic ambience; which have helped me in satisfactory completion of mini project work.

I extend our sincere thanks to **Dr. Krishna A N**, Head of the Department, Computer Science and Engineering; for providing an invaluable support throughout the period of mini project work.

I wish to express heartfelt gratitude to our **guides, Mr. Deepak B L and Mr.Lochan Gowda** for their valuable guidance, suggestions and cheerful encouragement during the entire period of this work.

Finally, we take this opportunity to extend our earnest gratitude and respect to our parents, Teaching & Non-teaching staffs of the department, the library staff and all our friends, who have directly or indirectly supported us during the period of this mini project work.

Regards,
Piyush Ranjan

ABSTRACT

The Banking Management System is used by the banks for maintaining a person's account in the bank.In this project I have tried to show the working of a banking system and cover the basic functionalities of a banking management system.

The database will store the information of the customer list with their profiles.It is capable of managing the banking operations of each customer in the bank like the withdrawals,deposits,trasfers,viewing balance,etc.Each customer data will be entered into the database and a unique id will be created which is their account number,which will help in faster retrieval of records.Each record can be updated ,deleted and retrieved whenever needed.

This system is based on relevant technologies and has been developed to carry out the processes easily and quickly,which is not possible manually.The aim is to minimize man power and increase the maintainability of data for quick and safe access to increase the efficiency involved in banking.

TABLE OF CONTENTS

List of Figures

Chapter-1

Introduction

1.1 Overview of the project

In present times, all major economic transactions have started taking place digitally. The major trends of modern digital transactions is substantiated by use of database management. Thesedatabases can be accessed by anyone with specific rights, and perform certain actions on it. Thedata update is done almost automatically and is much faster.Users can, in present days can access their accounts directly without going to a bank, makingtransfers, transactions and accessing cash directly without standing in long queues as wasprevalent earlier using ATM machines.On employee-side the data is much more organized, and accessing and performing actions onuser accounts is easier for them. Due to this the bank has better work efficiency and customerexperience improves as well.

Banking Management System is a platform that allows the bank to open the accounts of customers and add their details. It has simple UI (user interface) web design. Since,there is a requirement to maintain all the customer details , and to achieve that there is a need to use a system which will store and process all this data. The customer should register i.e. he should have his account in the bank in order to access the bank database website.

After registration the customer will be provided by his/her own account feed in database with a unique account no. and pin. The website is simple to use and it is also user-friendly. The aim is to minimize man power and increase the maintainability of data for quick and safe access to increase the efficiency involved in bankingThe database '**banking** 'consists of six tables with its specific attributes and the six tables are:-

alogin

1. **balance**
2. **contact**
3. **transfer**
4. **regform**
5. **logs**

The database consists of two views, which are virtual table derived its data from one or more than one table columns stored in the database.They are:

1. **clogin**
2. **cust_list**

1.2 Theory and Concepts

- **PHP**

HP (recursive acronym for *PHP: Hypertext Preprocessor*) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

Instead of lots of commands to output HTML (as seen in C or Perl), PHP pages contain HTML with embedded code that does "something" (in this case, output "Hi, I'm a PHP script!"). The PHP code is enclosed in special start and end processing instructions <?php and ?> that allow you to jump into and out of "PHP mode."

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.

The best things in using PHP are that it is extremely simple for a newcomer, but offers many advanced features for a professional programmer. Don't be afraid reading the long list of PHP's features. You can jump in, in a short time, and start writing simple scripts in a few hours.

- **SQL**

SQL (Structured QueryLanguage) originally called SEQUEL (Structured English Query Language) was designed and implemented at IBM Research as the interface for an experimental relational database system. SQL is now the standard language for commercial relational DBMSs. SQL is a comprehensive database language. It has statements for data definitions , queries and updates. Hence it is both a DDL and a DML.SQL is a language to request data from a database ,to add, update,or remove data within a database ,or to manipulate the metadata of the database.In addition , it has facilities for defining views on database, for specifying security and authorization, for defining integrity constraints, and for specifying transaction controls.

- **DBMS**

A **database-management system** (**DBMS**) is a computer-software application that interacts with end-users, other applications, and the database itself to capture and analyze data. A general-purpose DBMS allows the definition, creation, querying, update, and administration of database. A database is not generally portable across different DBMSs, but different DBMSs can Inter-operate by using standards such as SQL and ODBC or JDBC to allow a single application to work with more than one DBMS.

Chapter-2

Literature Survey

2.1 Traditional File Systems

In the early days of computing, data management and storage was a very new concept for organizations. The traditional approach to data handling offered a lot of the convenience of the manual approach to business processes (e.g. handwritten invoices & account statements, etc.) as well as the benefits of storing data electronically.

The traditional approach usually consisted of custom built data processes and computer information systems tailored for a specific business function. An accounting department would have their own information system tailored to their needs, where the sales department would have an entirely separate system for their needs.

Initially, these separate systems were very simple to set up as they mostly mirrored the business process that departments had been doing for years but allowed them to do things faster with less work. However, once the systems were in use for so long, they became very difficult for individual departments to manage and rely on their data because there was no reliable system in place to enforce data standards or management.

Separate information systems for each business function also led to conflicts of interest within the company. Departments felt a great deal of ownership for the data that they collected, processed, and managed which caused many issues among company-wide collaboration and data sharing. This separation of data also led to unnecessary redundancy and a high rate of unreliable and inconsistent data.

2.2 Pros and Cons of the Traditional Approach

Pros:

- Simple
    - Matched existing business processes and functions
    - Companies were not as interested in funding complicated information systems
- Initially low-cost
    - Early computing was not viewed as beneficial for large funding
    - Systems were designed to be cheap in order to save on cost

Cons:

- Separated ownership
    - Business functions had a high sense of data ownership
    - Departments unwilling to share data for fear of minimizing their superiority
- Unmanaged redundancy
    - Multiple instances of the same data appeared throughout various files, systems, and databases
    - Information updated in one place was not replicated to the other locations
    - Disk space was very expensive, and redundancy had a big impact on storage
- Data inconsistency
    - Redundant data stored in various locations was usually never stored the same way
    - Formatting was not centrally managed
- Lack of data sharing
    - Same data stored in multiple locations
    - Caused unnecessary doubling of efforts for processing and managing data
- High costs in the long run
    - Hiring data processors for each department was very expensive, and each position was typically working on the same thing just for a different area
    - Doubling of work as well as excessive maintenance costs

2.3 Downfall of Traditional Management System

Conceived in a relatively centralized era when software was deployed in static environments, legacy database architectures fail to support an increasingly mobile world where applications are accessed anytime, anywhere. Today software users want consistent improvements in usability and expect SaaS vendors to deliver new features and functionalities needed to achieve their business objectives.

However, legacy database technologies fall short in serving the needs of today's distributed and cloud environments for the following reasons:

- Inadequate failover capabilities
- Latency issues

- Insufficient provisions during peak demands
- Lack of high availability at all times
- Increasing operational costs
- Inability to meet the demands of global markets

For all of these reasons, traditional databases are unable to deliver results in a rapidly growing environment where the workload is geographically distributed across heterogeneous datacentres. Upgrading to a more distributed data model is costly and complicated and your DBAs can't just sit back and give up on this situation. Hence, due to these various reasons, the downfall of the traditional system was inevitable.

2.4 Introduction to the Database Management System

A database management system (DBMS) refers to the technology for creating and managing databases. Basically, a DBMS is a software tool to organize (create, retrieve, update and manage) data in a database.

The main aim of a DBMS is to supply a way to store and retrieve database information that is both convenient and efficient. By data, we mean known facts that can be recorded and that have embedded meaning. Normally people use software such as DBASE IV or V, Microsoft ACCESS, or EXCEL to store data in the form of database. A datum is a unit of data. Meaningful data combines to form information. Hence, information is interpreted data – data provided with semantics. MS ACCESS is one of the most common examples of database management software.

Database systems are meant to handle large collection of information. Management of data involves both defining structures for storage of information and providing mechanisms that can do the manipulation of those stored information. Moreover, the database system must ensure the safety of the information stored, despite system crashes or attempts at unauthorized access.

2.5 Indicative areas for the use of a DBMS

- Airlines: reservations, schedules etc.
- Telecom: calls made, customer details, network usage etc.
- Universities: registration, results, grades etc.
- Sales: products, purchases, customers etc.
- Banking: all transactions etc

2.6 Advantages of a DBMS

A Database Management System has many advantages over the traditional file system used in the earlier days, such as:

- Data independence: Application programs should be as free or independent as possible from details of data representation and storage. DBMS can supply an abstract view of the data for insulating application code from such facts.
- Efficient data access: DBMS utilizes a mixture of sophisticated concepts and techniques for storing and retrieving data competently and this feature becomes important in cases where the data is stored on external storage devices.
- Data integrity and security: If data is accessed through the DBMS, the DBMS can enforce integrity constraints on the data.
- Data administration: When several users share the data, integrating the administration of data can offer major improvements. Experienced professionals understand the nature of the data being managed and can be responsible for organizing the data representation to reduce redundancy and make the data to retrieve efficiently.

2.7 Components of a DBMS

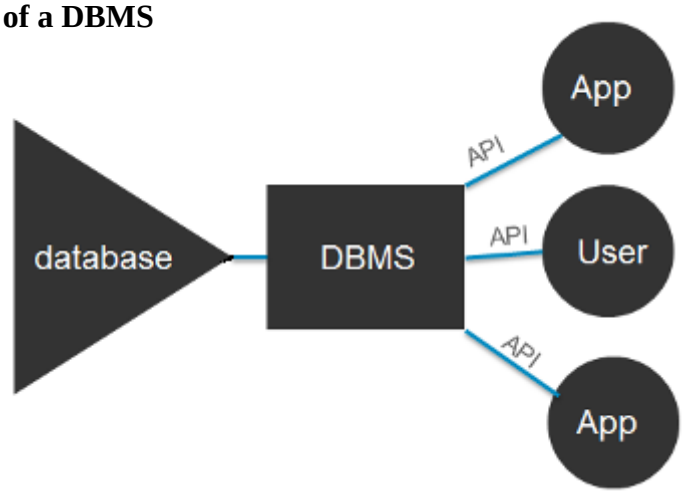


Fig 2.1 Components of a DBMS

DBMS

- **Users**: Users may be of any kind, such as database administrators, system developers or database users.
- **Database application**: Database application may be Departmental, Personal, Organizational and /or Internal.
- **DBMS**: Software that allows users to create and manipulate database access.
- **Database**: Collection of logical data as a single unit.

Chapter-3

System Requirements and Design

3.1 Software Requirements Specification

Minimum Hardware Requirements PROCESSOR: Intel PENTIUM IV or above

- **RAM** : 2GB or more
- **GPU** : Intel HD Graphics
- **Peripherals** : Standard PS/2 or USB Keyboard , Standard PS/2 or USB Wheel/Optical Mouse

3.2 Software Requirements

Technologies used:

- **Front end :** HTML, PHP, CSS,JAVA SCRIPT
- **Back end/Database :** MYSQL
- **Web Server:** Apache(on XAMPP7)

3.3 Softwares used

- **Text Editor**: Sublime Text
- **Server**: Apache (on XAMPP 7)
- **Operating System**: Windows 10
- **Database Support**: MySQL 5.7

3.4 Software Tools Used

We have two phases in our project:

1. **Front end**
2. **Back end**

3.4.1 Front End

- **HTML5 : HTML5** is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current major version of the HTML standard.It was published in October 2014 by the World Wide Web Consortium(W3C) to improve the language with support for the latest multimedia, while keeping it both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc. HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

  HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-platform mobile applications, because it includes features designed with low-powered devices in mind. Many new syntactic features are included. To natively include and handle multimedia and graphical content, the new <video>, <audio> and <canvas>elements were added, and support for scalable vector graphics (SVG) content and MathML for mathematical formulas. To enrich the semantic content of documents, new page structure elements such as <main>, <section>, <article>, <header>, <footer>, <aside>, <nav> and <figure>, are added. New attributes are introduced, some elements and attributes have been removed, and others such as <a>, <cite> and <menu> have been changed, redefined or standardized.

  The APIs and Document Object Model (DOM) are now fundamental parts of the HTML5 specificationand HTML5 also better defines the processing for any invalid documents

- **PHP : PHP** is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team.PHP originally stood for *Personal Home Page*, but it now stands for the recursive acronym *PHP: Hypertext Preprocessor*.PHP code may be embedded into HTML or HTML5 markup, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server software combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

- **CSS : Cascading Style Sheets** (**CSS**) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

  CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts.This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

- **JAVA SCRIPT:**

  JavaScript often abbreviated as **JS**, is a high-level, dynamic, weakly typed, prototype-based, multi-paradigm, and interpreted programming language. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production. It is used to make webpages interactive and provide online programs, including video games. The majority of websites employ it, and all modern web browsers support it without the need for plug-ins by means of a built-in JavaScript engine. Each of the many JavaScript engines represent a different implementation of JavaScript, all based on the ECMAScript specification, with some engines not supporting the spec fully, and with many engines supporting additional features beyond ECMA.

  As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative (including object-oriented and prototype-based) programming styles. It has an API for working with text, arrays, dates, regular expressions, and basic manipulation of the DOM, but the language itself does not include any I/O, such as networking, storage, or graphics facilities, relying for these upon the host environment in which it is embedded.

  Initially only implemented client-side in web browsers, JavaScript engines are now embedded in many other types of host software, including server-side in web servers and databases, and in non-web programs such as word processors and PDF software, and in runtime environments that make JavaScript available for writing mobile and desktop applications, including desktop widgets.

3.4.2 Back End

- **MYSQL: MySQL** is an open-source relational database management system(RDBMS).Its name is a combination of "My", the name of co-founder Michael Widenius's daughter,and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.For proprietary use, several paid editions are available, and offer additional functionality.

  MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer.MySQL works on many system platforms.It including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64 . A port of MySQL to OpenVMS also exists

- **XAMPP : XAMPP** is a free and open source cross-platform web server solution stack package Apache Friends, consisting mainly of the Apache HTTP Server, Maria DB database, and interpreters for scripts written in the PHP and Perl programming languages. XAMPP stands for Cross-Platform (X), Apache , MariaDB (M), PHP (P) and Perl (P). It is a simple, lightweight Apache distribution that makes it extremely easy for developers to create a local web server for testing and deployment purposes. Everything needed to set up a web server – server application (Apache), database (Maria DB), and scripting language (PHP) – is included in an extractable file. XAMPP is also cross-platform, which means it works equally well on Linux, Mac and Windows. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server extremely easy as well.

3.5 System Design

The purpose of design phase is to develop a clean understanding of what the developer wants people to gain from his/her project. As the developer works on the project, the test for every design decision should be

"Does this features fulfill the ultimate purpose of the project ? "

A purpose statement affects design process by explaining what the developer what's the project to do, rather then describing the project itself. The design document will verify that the current design meets all of the explicit requirement contained in the system model as well as the implicit requirement described by the customer.

Structure of design document

System architecture section has:

- **System Architecture Design**– The detailed diagram of the system, server and client
- **Data Design**– The Data Design includes an ERD as well as Database design.

3.5.1 System Architecture

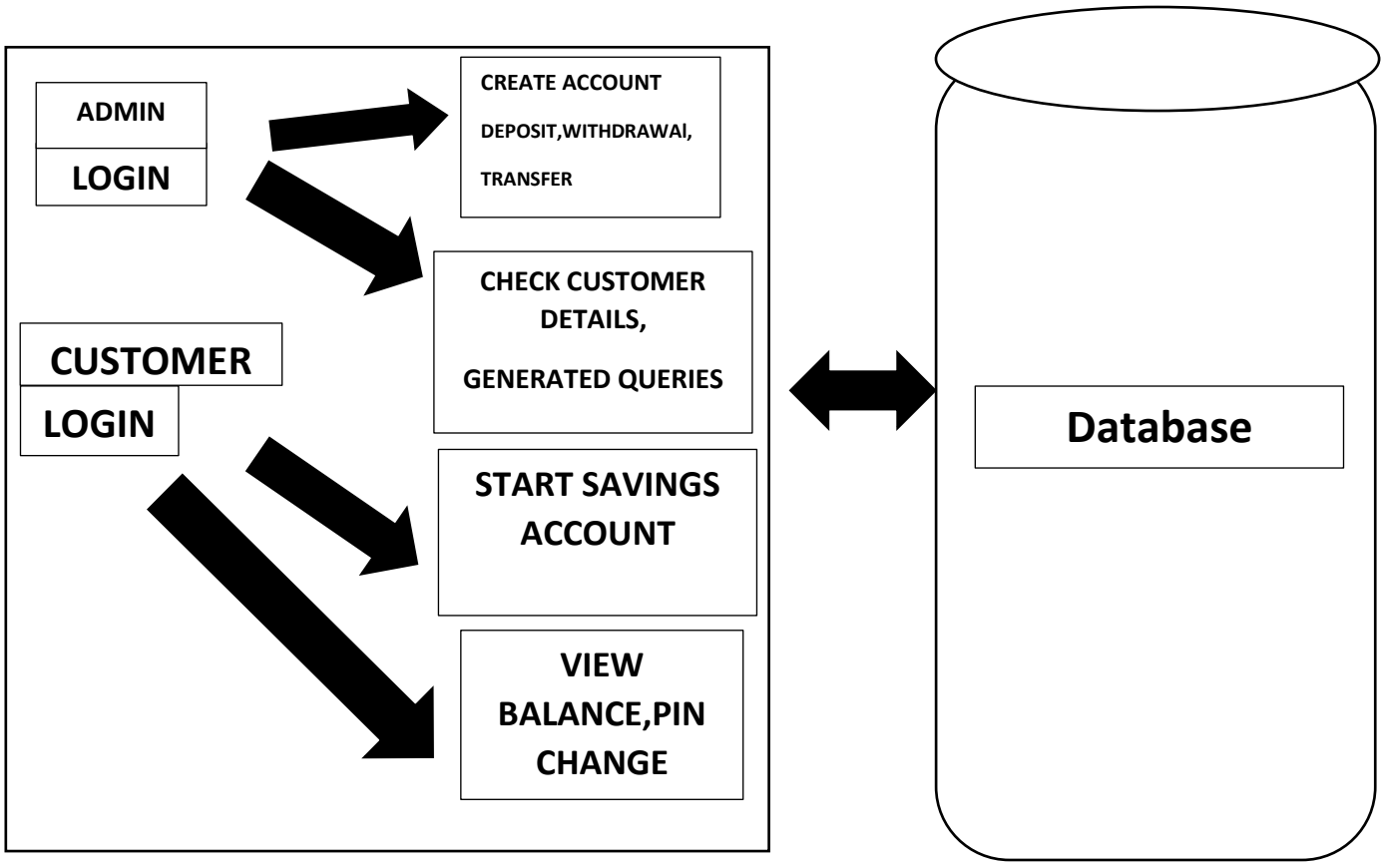


Figure 3.5.1 System Architecture of Banking Management System

3.5.2 Data Design:

- **Entity Relationship Diagram:**

This relationship diagram shows how the tables in the database are connected to each other and how the controls flows from one table to another when some action triggered by the user. It also shows the constraints on the database such as primary key constraints, foreign key constraints and procedures and trigger. Entity Relationship Diagram also called as ER Diagram.

When documenting a system or process, looking at the system in multiple ways increases the understanding of that system. ERD diagrams are commonly used in conjunction with a data flow diagram to display the contents of a data store. They help us to visualize how data is connected in a general way, and are particularly useful for constructing a relational database.

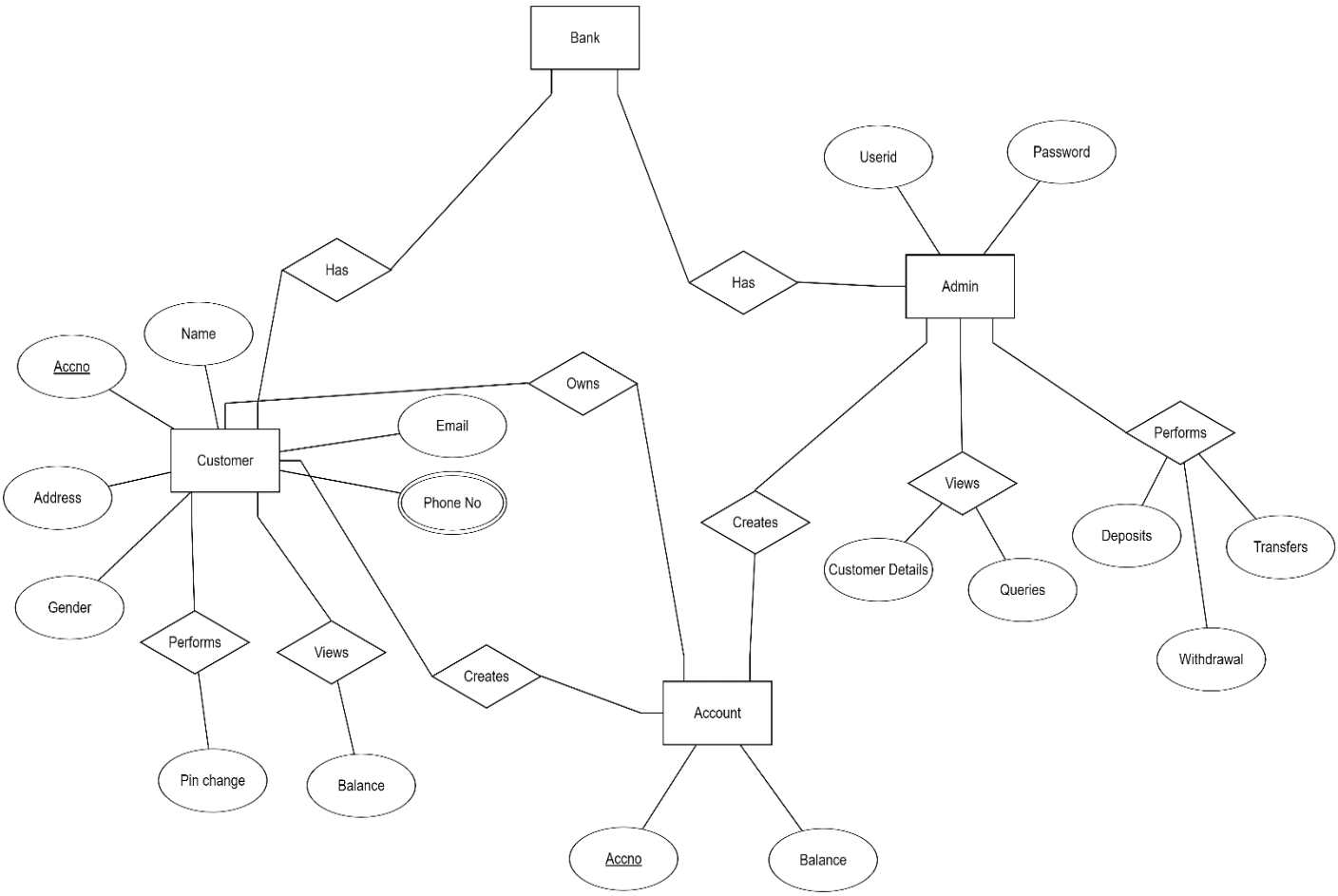


Figure 3.5.2.1: ER Diagram for Banking Management System

- **Schema Diagram:**

The Schema Diagram gives us the information about the attributes in the table of the database and how the given tables are related to each other.

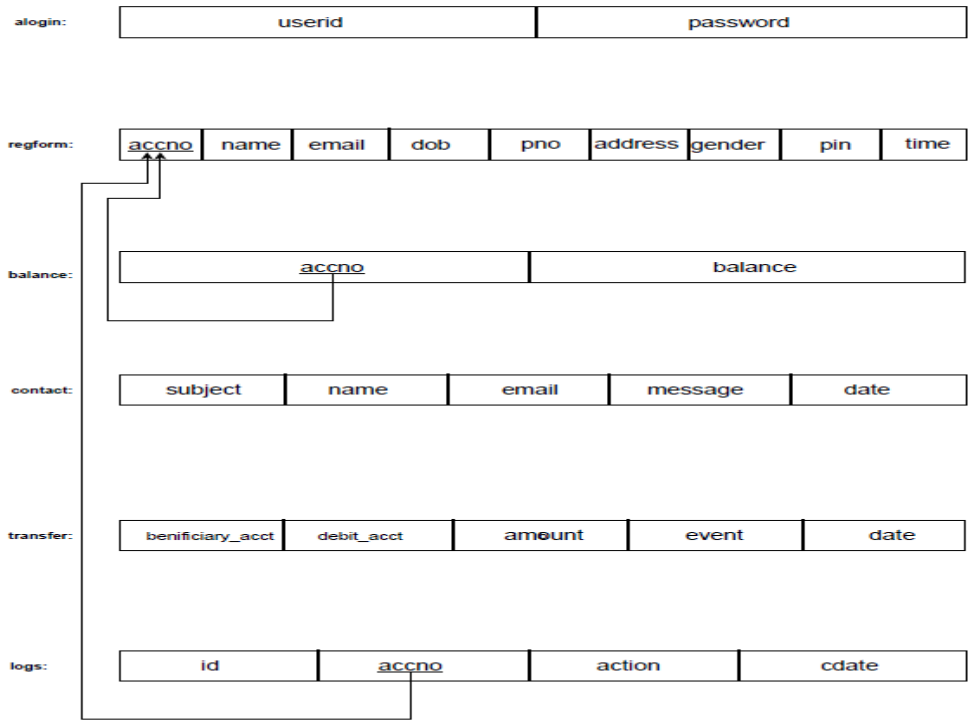


Figure 3.5.2.2: Schema Diagram for Banking Management System

3.5.3 Creating a database:

Type http://localhost/phpmyadmin/ in the web browser.

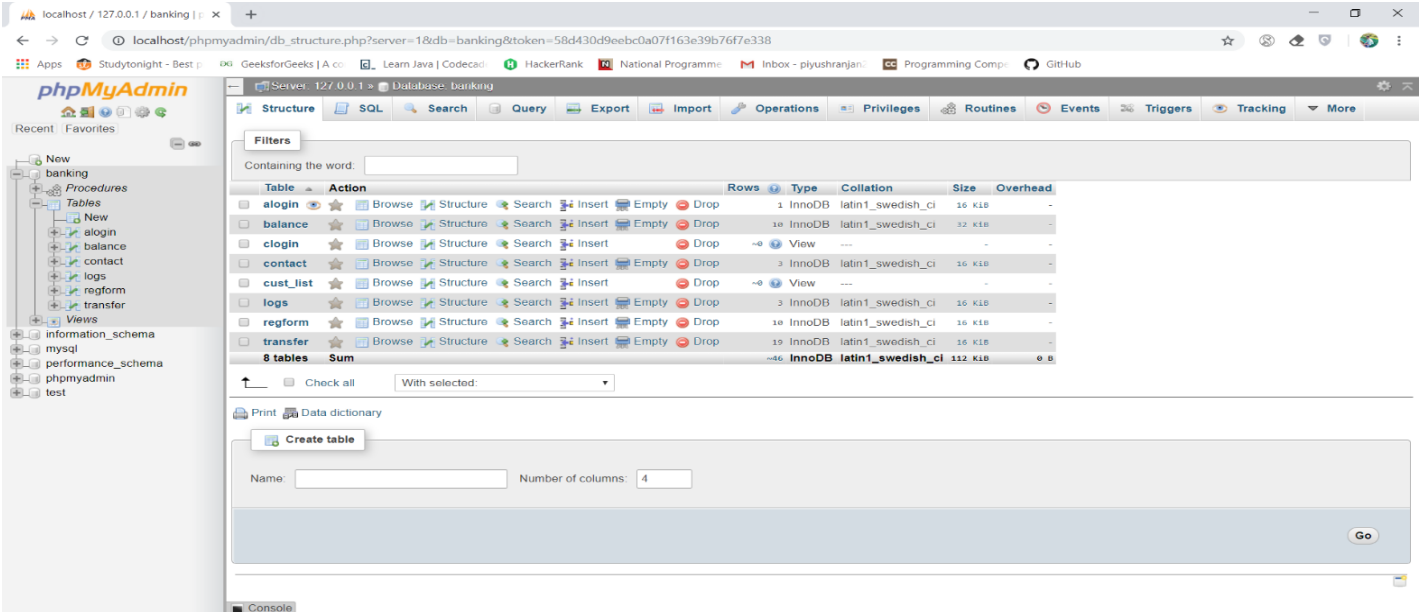


Figure 3.5.3: Database Description

The above picture shows how exactly the xampp phpMyAdmin page looks like. All the SQL commands can be executed here.

3.5.4. Table Design:

- **alogin table**

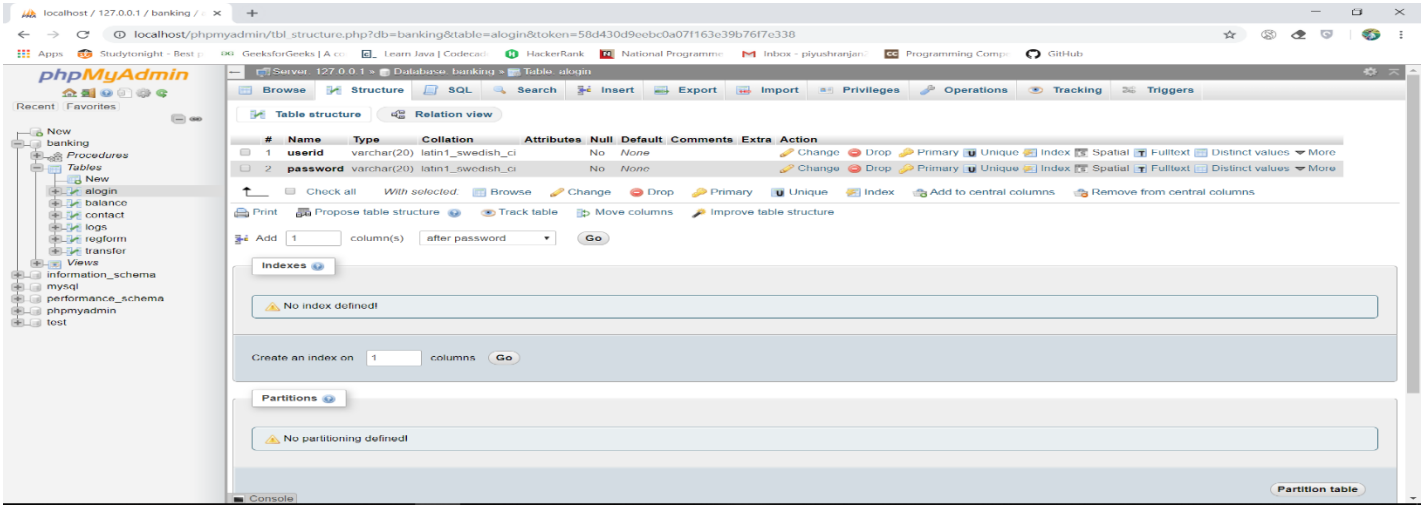


Figure 3.5.4.1: alogin Table Description

- **regform table**

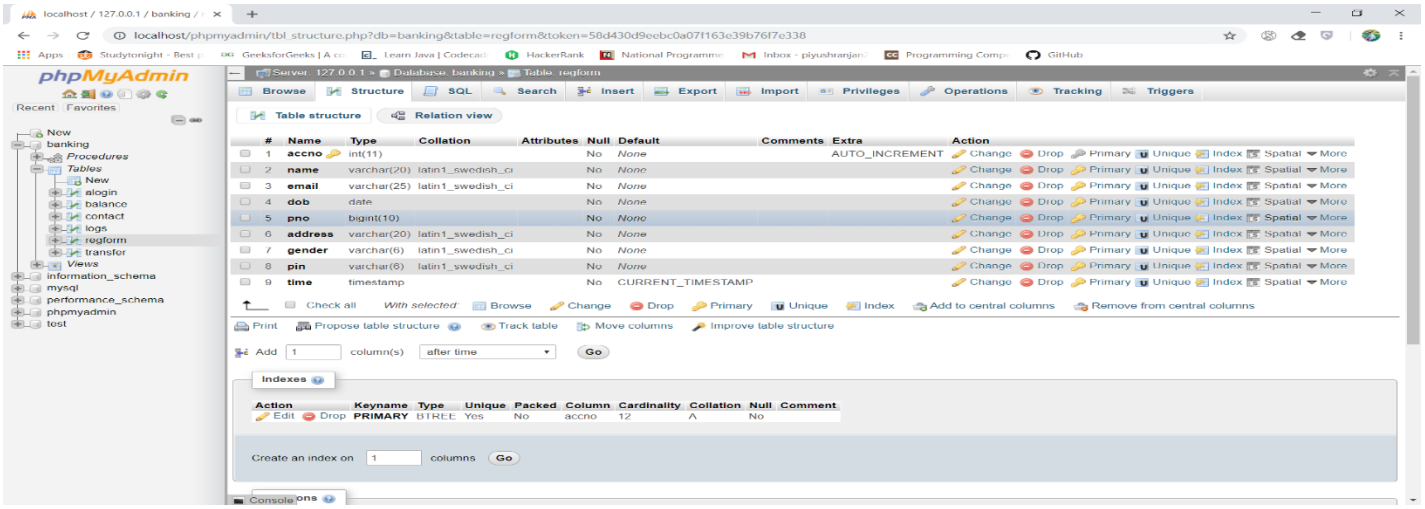


Figure 3.5.4.2: regform Table Description

- **balance table**

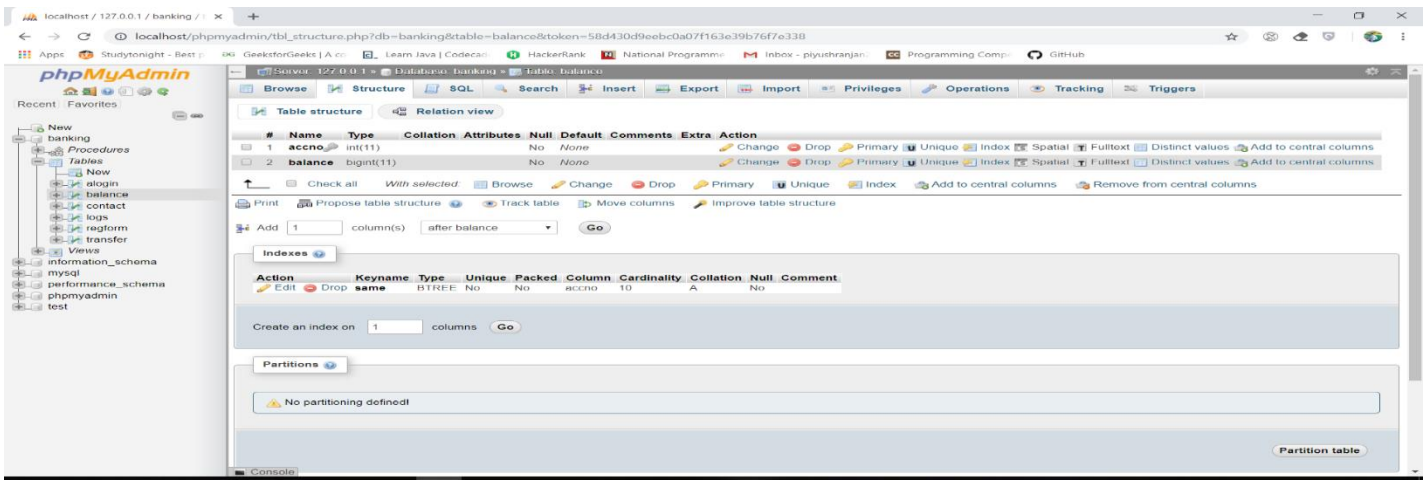


Figure 3.5.4.3: balance Table Description

- **contact table**

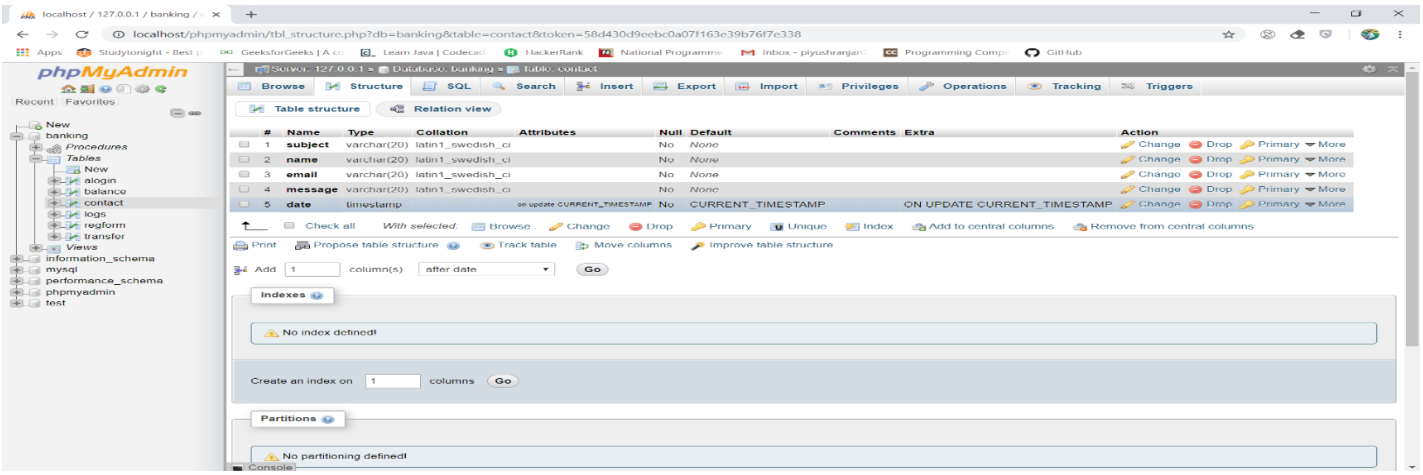


Figure 3.5.4.4: contact table Description

- **logs table**

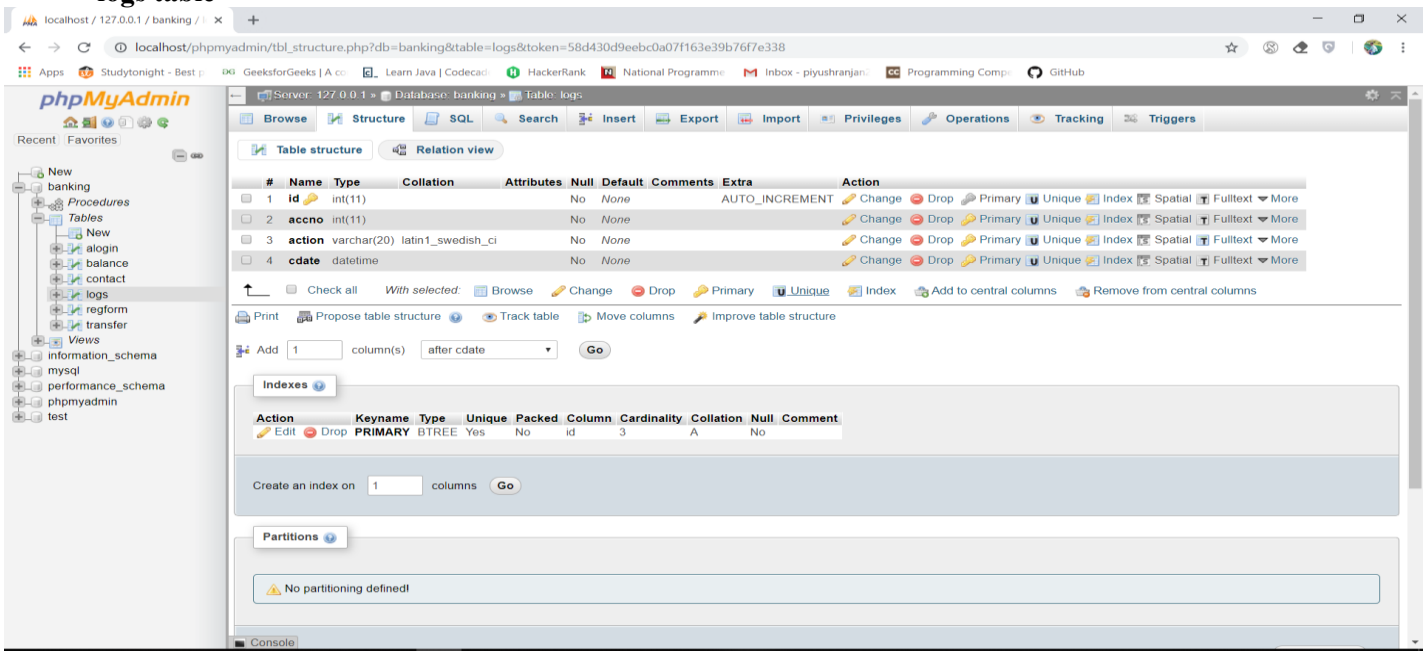


Figure 3.5.4.5: logs Table Description

- **transfer table**

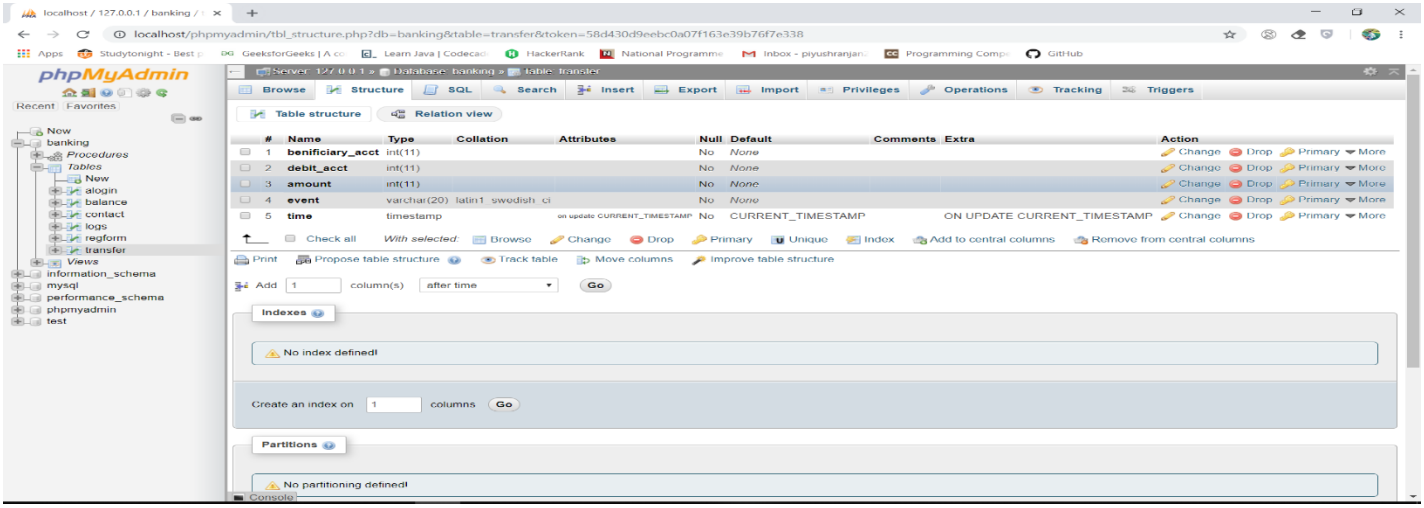


Figure 3.5.4.6:transfer Table Description

Chapter 4

Implementation

4.1 Component Modules :

Module 1 : Home Page

The Home page has five buttons, which will redirect to the required pages.

They are:

1. Home
2. Services
3. About
4. Contact
5. Admin Login

The home page also have a window for Customer Login asking for account number and associated pin. **T**he webpage**(index.php)** is linked with a php server page namely **serverc.php** which connects the database from back-end to the front-end using above mentioned member functions.

1) index.php:

- Contains links to different redirect to different pages.
- Contains html code for the user interface used in login page.
- Consists of definition of customer login page objects.

2) Customer Login window:

- Contains form based user interface for logging in into a customer's account..
- Asks for the account number of the customer and pin
- On successful verification redirects to the Customer panel **(custindex.php)** else displays some error message.

3) Open Savings Account(Self Signup)
reg1.php

- Contains form based user interface for new customer registration.

- Contains html code for the user interface used in new customer registration page.
- Contains linked php page **serverreg.php** to connect database in back-end to the front-end.
- Consists of button namely **Home** which redirects to **index.php**.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- Consists of member functions defined in php.
- Action for "**Get started**" button defined.
- Displays a message with the **generated account number** of the customer on successful registration.

Module 2 : Admin Panel

The webpage **Admin Login (alog.php)** is opened with a window admin login asking for userid and password of the admin.When the admin is logged in the website. This admin panel page is displayed with marquee image and with buttons which hold the following links:

1. Home : It redirects to homepage(index.php)

2. Logout : It redirects to homepage(index2.php) with a message ,"Successfully Logged Out".

The webpage**(alog.php)** is linked with a php server page namely **server.php** which connects the database from back-end to the front-end using below mentioned member functions.This page has the following functionalities under admin panel and hold the following links:

1. Customer Registration : It redirects to the webpage reg.php
2. Account Deposit : It redirects to the webpage dep.php
3. Account Withdrawal : It redirects to the webpage wdl.php
4. Transfer Balance : It redirects to the webpage trf.php
5. Customers List : It redirects to the webpage clist.php
6. Generated Queries : It redirects to the webpage Query.php

1) reg.php

- Contains links to redirect to different pages.
- Contains form based user interface for new customer registration.
- Contains html code for the user interface used in new customer registration page.
- Contains linked php page **serverreg.php** to connect database in back-end to the front-end.
- Consists of buttons namely **Home** and **Logout** which redirects to **index.php** and **index2.php** respectively.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- Consists of member functions defined in php.
- Action for "**Get started**" button defined.
- Displays a message with the **generated account number** of the customer on successful registration.
- The admin asks for the desired pin from the customer during registration and gives the same after successful registration.The Customer can use this pin for Login using his account number for viewing his account balance and can change his pin.

2) dep.php

- Contains links to redirect to different pages.
- Contains form based user interface for funding a customer's account.
- Contains html code for the user interface used in Account Deposit page.
- Contains linked php page **serverdep.php** to connect database in back-end to the front-end.
- Consists of buttons namely **Home** and **Logout** which redirects to **index.php** and **index2.php** respectively.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- Consists of member functions defined in php.
- Action for "**UPDATE**" button defined
- Displays a message **"Deposit Successful"** on successful funding.

3) wdl.php

- Contains links to redirect to different pages.
- Contains form based user interface for withdrawing balance from a customer's account.
- Contains html code for the user interface used in Account Withdrawal page.
- Contains linked php page **serverdep.php** to connect database in back-end to the front-end.
- Consists of buttons namely **Home** and **Logout** which redirects to **index.php** and **index2.php** respectively.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- Consists of member functions defined in php.
- Action for "**UPDATE**" button defined
- Displays a message **"Withdrawal Successful"** on successful withdrawal else **insufficient balance**
  when the amount to be withdrawn is greater than the existing balance.

4) trf.php

- Contains links to redirect to different pages.
- Contains form based user interface for transfering balance to a customer's account.
- Contains html code for the user interface used in Intra Accounts Transfer page.
- Contains linked php page **servertrf.php** to connect database in back-end to the front-end.
- Consists of buttons namely **Home** and **Logout** which redirects to **index.php** and **index2.php** respectively.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- sConsists of member functions defined in php.
- Action for "**TRANSFER**" button defined
- Displays a message **"Transfer Successful"** on successful transfer of the balance else **insufficient balance** when the amount to be transfered is greater than the existing balance.

5) clist.php

- Contains links to redirect to different pages.
- Contains table based user interface for displaying the list of customers.
- Contains html code for the user interface used in Customers List page.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- Consists of member functions defined in php.
- Consists of buttons namely **Home** and **Logout** which redirects to **index.php** and **index2.php** respectively

6) Query.php

- Contains links to redirect to different pages.
- Contains table based user interface for displaying the customer generated queries.
- Contains html code for the user interface used in Queries List page under contac.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- Consists of member functions defined in php.
- Consists of buttons namely **Home** and **Logout** which redirects to **index.php** and **index2.php** respectively.

* Each webpage has the following:

- Contains member functions defined in php code
  The member functions used are:
  - Session_start (): This function is used to start a session and store session variables.
  - Mysql_connect (): This function is used to connect to the database server, it takes 4 inputs 'localhost','user','pass','dbname'
  - Isset (): This checks is the variable contains a value or not.
  - Mysql_query (): this method is used to execute a query, it takes two parameters 'query' and 'connection'.
  - Mysql_fetch_array(): this method is used to fetch the result set of the executed query.

Each webpage is linked with a php page namely server page which connects the database from back-end to the front-end using above member functions.

- Contains buttons definition and functions.
- Contains input slots definitions

Module 3 : Customer Panel

The customer login window is there in the homepage(**index.php**) asking for account no and pin of the customer.When the customer is logged in the website. This customer panel page(**custindex.php**) is displayed with marquee image and with buttons which hold the following links:

1. Home : It redirects to homepage(index.php)
2. Logout : It redirects to homepage(index2.php) with a message ,"Successfully Logged Out".
3. Contact : It redirect to webpage(contact.php)

This page has the the html code for the following functionalities under customer panel and holds the following windows:

1. Change pin
2. View balance

1) Change Pin window:

- Contains form based user interface for changing pin of a customer's account..
- Asks for the account number of the customer again to verify the account number.
- Asks for the old pin of that accunt to verify the pin.
- Asks for new desired pin and retype the pin.
- Displays a message **"Pin Successfully changed"** on successful verification of each fields and pin change else displays some error message.

2) View Balance window:

- Contains user interface for viewing balance of a customer's account.
- Asks for the account number of the customer again to verify the account number..
- Displays a message **"Your account balance is:_____"** showing account balance on successful verification of account number else displays some error message.

The webpage**(custindex.php)** is linked with a php server page namely **servercu.php** which connects the database from back-end to the front-end using above mentioned member functions.

Module 4 : Services window

#Services

- Redirects to the services panel on the homepage**(index.php)'**
- Contains user interface for displaying the services provided by the bank.

Module 5: About window

about.php

- **About** button on the homepage **(index.php)** redirects to the webpage **(about.php).**
- Contains user interface for displaying the page containing information about the bank.
- **"Read More "** button to display more information.
- Actions for the **"Read More"** button defined.
- Consists of buttons namely **Home ,Contact** and **Admin Login** which redirects to webpages **index.php,contact.php** and **alog.php** respectively
- Actions for the buttons **Home ,Contact** and **Admin Login** are defined.

Module 6 :Contact window

contact.php

- **Contact** button on the homepage **(index.php)** redirects to the webpage **(contact.php).**
- Contains form based user interface for generating queries by a customer which gets stored in the back-end and can be accessed by the admin under admin panel.
- Contains html code for the user interface used in Contact page.
- Consists of POST and SESSION variables to get the input from the form and from the previous pages respectively.
- Consists of buttons namely **Home ,Contact** and **Admin Login** which redirects to webpages **index.php,contact.php** and **alog.php** respectively
- Actions for the buttons **Home ,Contact**,**Admin Login and Submit** are defined.

4.2 Table Creation:

Table 1: alogin

Creating table **alogin** :

CREATE TABLE **alogin** (userid varchar(20),password varchar(20));

The **alogin** table consists of the following attributes shown in the figure:

userid	password

Table 2: regform

Creating table **regform**:

CREATE TABLE r**egform** (accno int primary key,name varchar(11),email varchar(25),dob date,pno bigint(10),address varchar(20),gender varchar(6),pin varchar(6),time timestamp);

ENGINE=InnoDB AUTO_INCREMENT= 546787550 DEFAULT CHARSET=latin1

The **regform** table consists of the following attributes shown in the figure:

accno	name	email	dob	pno	address	gender	pin	time

Table 3: balance

Creating table **balance**:

CREATE TABLE **balance** (accno references regform(accno) on delete cascade,balance integer);

The **balance** consists of the following attributes shown in the figure:

accno	balance

Table 4: contact

Creating table **contact**:

CREATE TABLE **contact** (subject varchar(20),name varchar(20),email varchar(20),message varchar(20),date timestamp);

The **contact** table consists of the following attributes shown in the figure:

subject	name	email	message	date

Table 5: transfer

Creating table **contact**:

CREATE TABLE contact (beneficiary_acct int,debit_acct int,amount int,event varchar(20),time timestamp);

The **transfer** table consists of the following attributes shown in the figure:

beneficiary_acct	debit_acct	amount	event	date

Table 6: logs

Creating table **logs**:

CREATE TABLE **logs** (id integer primary key,accno references regform(accno) on delete cascade,action varchar(20),cdate datetime);

The **logs** table consists of the following attributes shown in the figure:

id	accno	action	cdate

ENGINE=InnoDB AUTO_INCREMENT= 1 DEFAULT CHARSET=latin1

- Creating view **clogin**:

```
CREATE VIEW clogin AS
SELECT accno,pin
FROM regform;
```

- Creating view **cust_list**:

```
CREATE VIEW cust_list AS
SELECT a.accno,a.name,a.email,a.dob,a.gender,a.time,b.balance
FROM regform a,balance b

Where a.accno=b.accno;
```

4.3 Inserting values to the tables:

Inserting values into the table **alogin**:

INSERT INTO alogin(userid,password) VALUES('&userid','&password');

Inserting values into the table **regform**:

INSERT INTO regform (name,email,dob,pno,address,gender,pin)
VALUES('&username','&email','&dob','&pno','&address','&gender','&pin');

Inserting values into the table **balance**:

CREATE TABLE **balance** (accno ,balance) VALUES('&accno',0);

Inserting values into the table **contact**:

INSERT INTO contact (subject,name,email,message) VALUES('&subject','&name','&email','&message');

Inserting values into the table **transfer**:

INSERT INTO transfer(benificiary_acct,debit_acct,amount,event) VALUES
('&accno1','&accno2','&depos', 'Transfer');

Inserting values into the table **logs**:

CREATE TABLE **logs** (id,accno,action,cdate) VALUES ('&id','&accno', 'Inserted','&cdate');

4.4 Stored procedure:

A **procedure** (often called a stored procedure) is a subroutine like a subprogram in a regular computing language, stored in database. There are many useful applications of SQL procedures within a database or database application architecture. SQL procedures can be used to create simple scripts for quickly querying transforming, updating data, generating basic reports, improve application performance, modularizing applications, and improve overall database design, and database security.

The procedure used in this project is called **getCustomers**, which retrieves the information of the customer based on the accno given by the user.

Code:

```
CREATE DEFINER=`root`@`localhost` PROCEDURE ` getCustomers`( IN `account_no` INT)
NOT DETERMINISTIC NO SQL SQL SECURITY
DEFINER select account_no,name from regform where account_no=accno;
DROP PROCEDURE `getCustomers`; CREATE DEFINER=`root`@`localhost` PROCEDURE
`getCustomers`(IN `account_no` INT, OUT `cust_name` VARCHAR(20)) NOT DETERMINISTIC NO
SQL SQL SECURITY
```

DEFINER select name INTO cust_name from regform where account_no=accno;

4.5 Trigger:

Triggers are stored programs, which are automatically executed or fired when some event occurs. Triggers are written to be executed in response to any of the following events. A database manipulation (DML) statement (DELETE, INSERT, or UPDATE). A database definition (DDL) statement (CREATE, ALTER, or DROP).

The trigger named **insert_Log** used in this project gives the logs for new account number and date&time of new customer registration.

The code of trigger is given below:

Code:

```
CREATE TRIGGER `insertLog`;
CREATE DEFINER=`root`@`localhost` TRIGGER `insertLog` AFTER INSERT ON
`regform` FOR EACH ROW insert into logs value(null,NEW.accno,'inserted',NOW());
```

4.6 Code Snippets:

4.6.1 Connection to database:

The database is connected to the front end html using php, the code for database connection is shown below.

```
        $dbhost = "localhost";
        $dbuser = "root";
        $dbpass = "";
        $dbname=”banking”;
   // Create connection
        $conn = mysql_connection($dbhost, $dbuser, $dbpass,$dbname);
   // Check connection
         if(! $conn )
        {
                die('Could not connect: ' . mysql_error());
        }
```

4.6.2 Home page

```
<?php include('servercu.php');?>
<!DOCTYPE html>
<html lang="en">
<head>
<title>Home</title>
        <link href="css/bars.css" rel="stylesheet" type="text/css" />
        <link rel="stylesheet" href="css/bootstrap.min.css" type="text/css" media="all" />
        <link rel="stylesheet" href="css/style.css" type="text/css" media="all" />
        <link rel="stylesheet" href="css/font-awesome.css" />
        <script type="text/javascript" src="js/jquery-2.1.4.min.js"></script>
        <script type="text/javascript" src="js/bootstrap.min.js"></script>

<linkhref="//fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,700i,800,800i&amp;sub
set=cyrillic,cyrillic-ext,greek,greek-ext,latin-ext,vietnamese" rel="stylesheet">
<link href="//fonts.googleapis.com/css?family=Ropa+Sans:400,400i&amp;subset=latin-ext" rel="stylesheet">
<script type="text/javascript">

jQuery(document).ready(function($) {
        $(".scroll").click(function(event){
        event.preventDefault();
        $('html,body').animate({scrollTop:$(this.hash).offset().top},1000);
  });
})
</head>
<body>
<div class="top-bar">
<nav class="navbar navbar-default">
```

```
<div class="container-fluid">
<div class="navbar-header">
<button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#myNavbar">
<span class="sr-only">Toggle navigation</span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</button>
</div>
<div class="collapse navbar-collapse" id="myNavbar">
<ul class="nav navbar-nav navbar-right">
<li><a href="http://localhost/BankingMS/index.php">home</a></li>
<li><a href="#services" class="scroll">services</a></li>
<li><a href="http://localhost/BankingMS/about.php">about</a></li>
<li><a href="http://localhost/BankingMS/contact.php">contact</a></li>
<li><a href="http://localhost/BankingMS/alog.php">Admin login</a></li>
</ul>
</div>
</div>
</nav>
</div>
<div class="logo">
<a href="index.php"><!--<i class="fa fa-inr" aria-hidden="true"></i>-->Corporate <span>bank</span></a>
</div>
<div class="banner-main jarallax">
<div class="container">
<div class="banner-inner">
<div class="col-md-5 banner-left">
<form action="index.php" method="post">
<h3>Customer Login</h3>
<input type="text" name="accno" placeholder="Account No" required=""/>
<input type="text" name="pin" placeholder="PIN" required=""/>
<div class="submit">
<input type="submit" name="open" value="get started">
</div>
</form>
</div>
<div class="logo">
<a href="reg1.php"><!--<i class="fa fa-inr" aria-hidden="true"></i>-->__Start Savings<span> Account__</span></a>
</div>
<div class="col-md-7  banner-right">
<h1>Corporate Vision</h1>
<center><h4>"Emerge as a Model for Inclusive Growth and Innovative Banking Services"</h4></center>
<h1>Corporate Mission</h1>
<div class="banner-right-text">
<div class="main-icon">
<i class="fa fa-share" aria-hidden="true"></i>
</div>
<p>To expand our reach to meet the financial needs of people</p>
<div class="clearfix"></div>
</div>
<div class="banner-right-text">
<div class="main-icon">
<i class="fa fa-share" aria-hidden="true"></i>
</div>
<p>To provide full range of banking services with innovative products</p>
```

```
<div class="clearfix"></div>
</div>
<div class="banner-right-text">
<div class="main-icon">
<i class="fa fa-share" aria-hidden="true"></i>
</div>
<p>To continue to adopt modern technology for superior banking experience</p>
<div class="clearfix"></div>
</div>
<div class="banner-right-text">
<div class="main-icon">
<i class="fa fa-share" aria-hidden="true"></i>
</div>
<p>To create a rewarding environment for  all stakeholders</p>
<div class="clearfix"></div>
</div>
<div class="banner-right-text">
<div class="main-icon">
<i class="fa fa-share" aria-hidden="true"></i>
</div>
<p>To continue as a model organisation for transparent and ethical practices</p>
<div class="clearfix"></div>
</div>
</div>
<div class="clearfix"></div>
</div>
<div class="clearfix"></div>
</div>
</div>
<section class="copyright">
<div class="agileits_copyright text-center">
<p>© 2018 Corporate Bank. All rights reserved</p>
</div>
</section>

<script src="js/jarallax.js"></script>
<script src="js/SmoothScroll.min.js"></script>
<script type="text/javascript">
                    $('.jarallax').jarallax({
                            speed: 0.5,
                            imgWidth: 1366,
                            imgHeight: 768
                    })
</script>
<script type="text/javascript" src="js/move-top.js"></script>
<script type="text/javascript" src="js/easing.js"></script>
<script type="text/javascript">
                    $(document).ready(function() {
                    $().UItoTop({ easingType: 'easeOutQuart' });
});
</script>
<script src="js/bars.js"></script>
</body>
</html>
```

Chapter 5

Results

SNAPSHOTS

Module 1 : Home Page

index.php: Customer Login Window

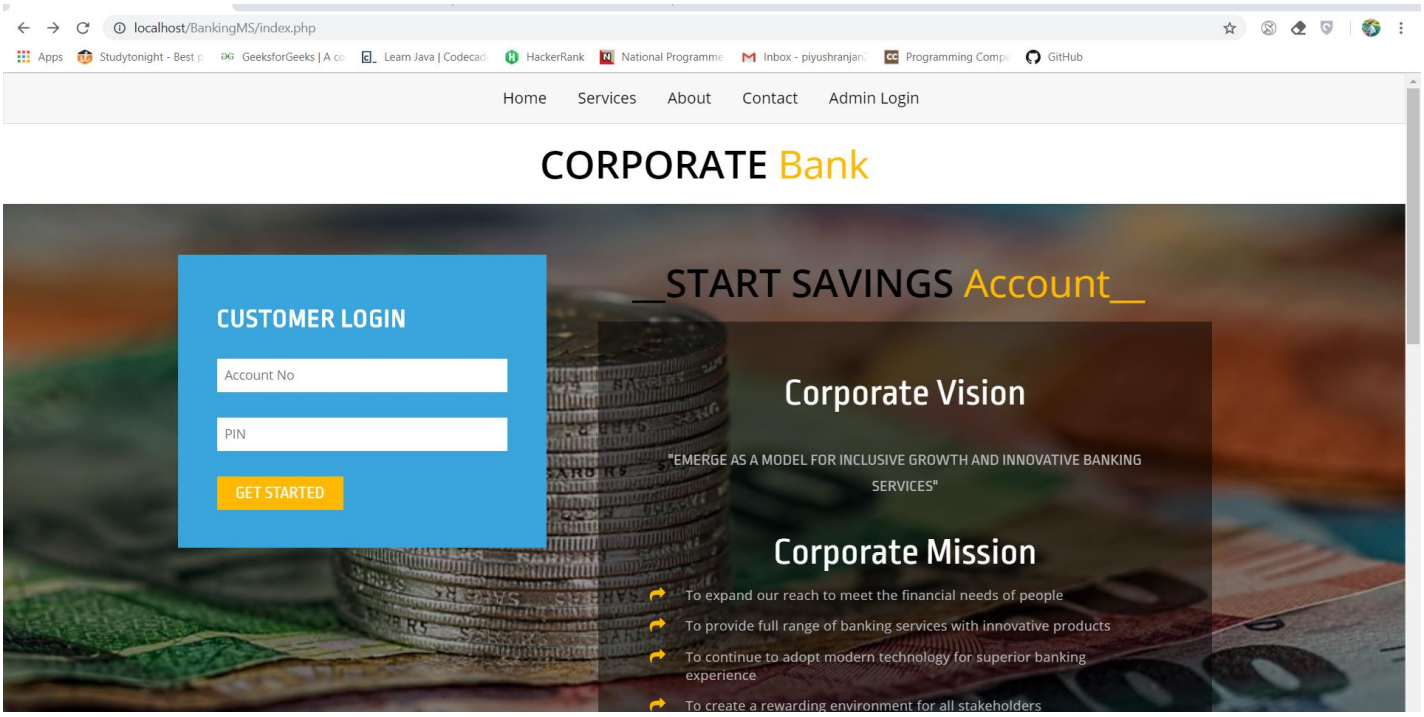


Figure 5.1.1: Customer Login Window

Open Savings Account(Self Signup) reg1.php

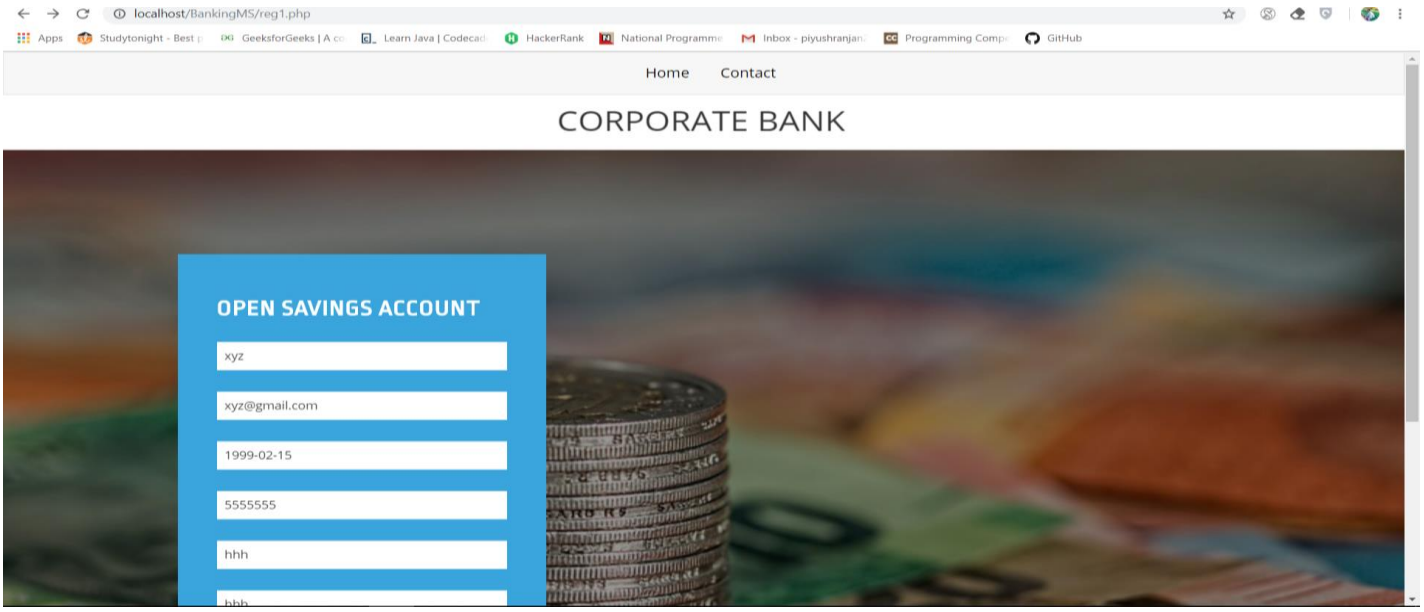


Figure 5.1.2.: Signup Window

Module 2 : Admin Panel

- Admin Login Page

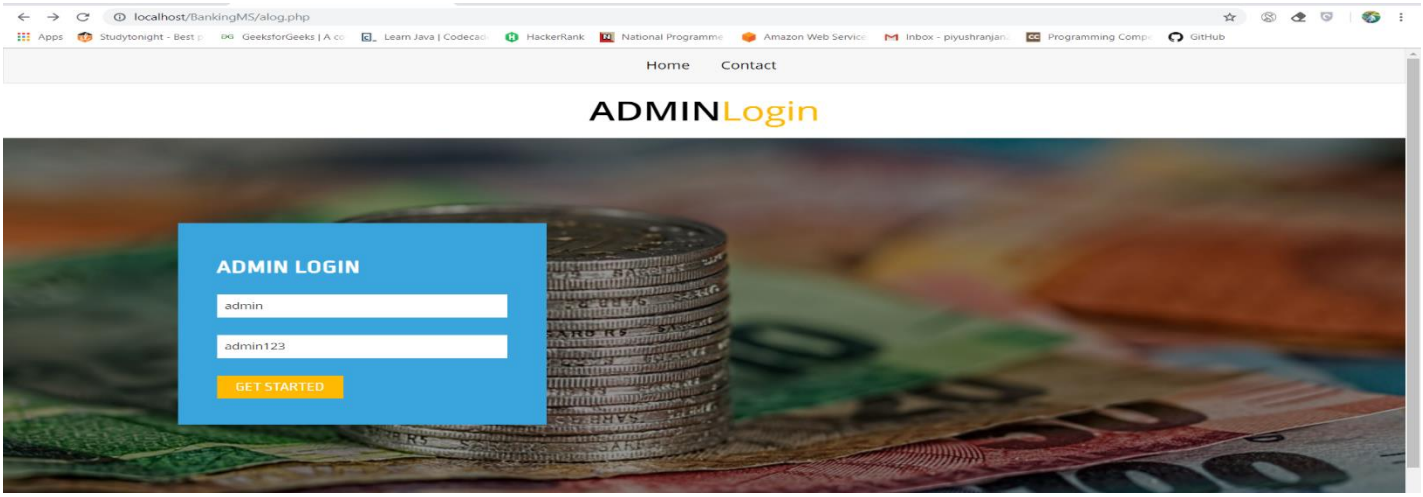


Figure 5.2.1: Admin Login Panel

- Admin Panel Page

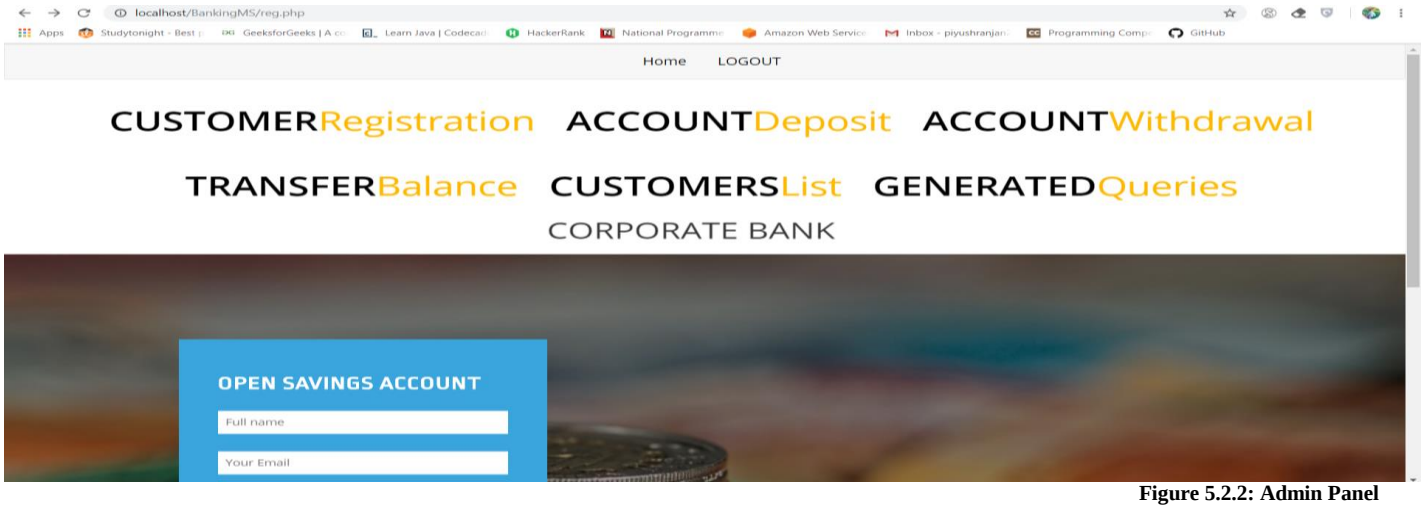


Figure 5.2.2: Admin Panel

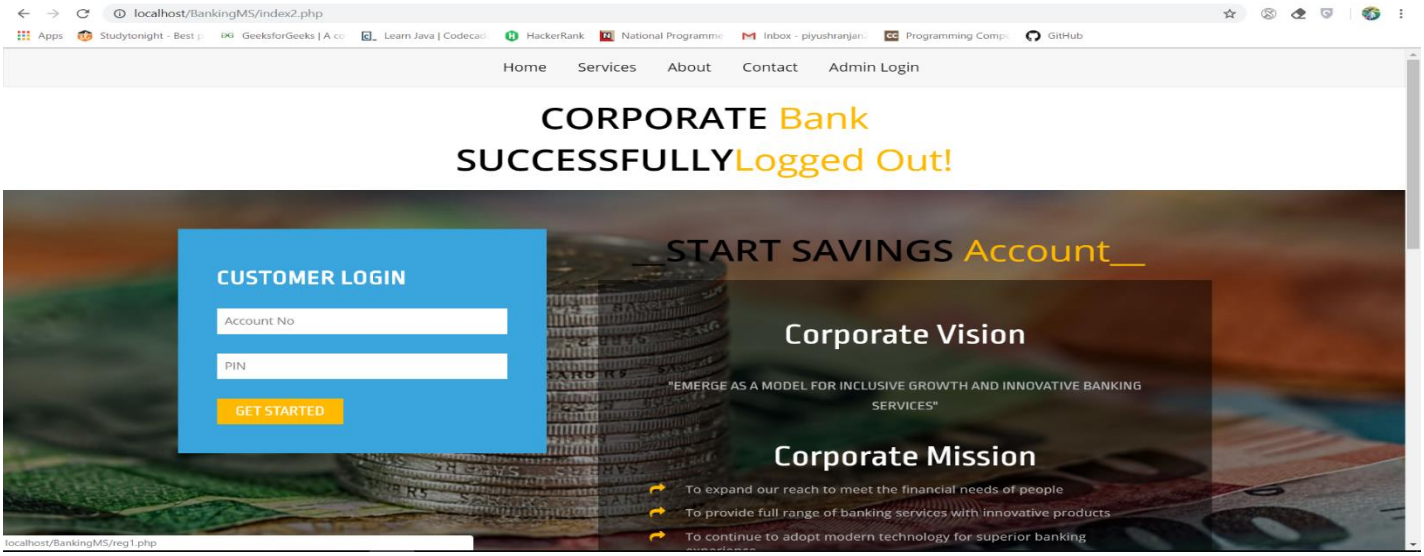


Figure 5.2.3: Logout From Admin panel

1) reg.php

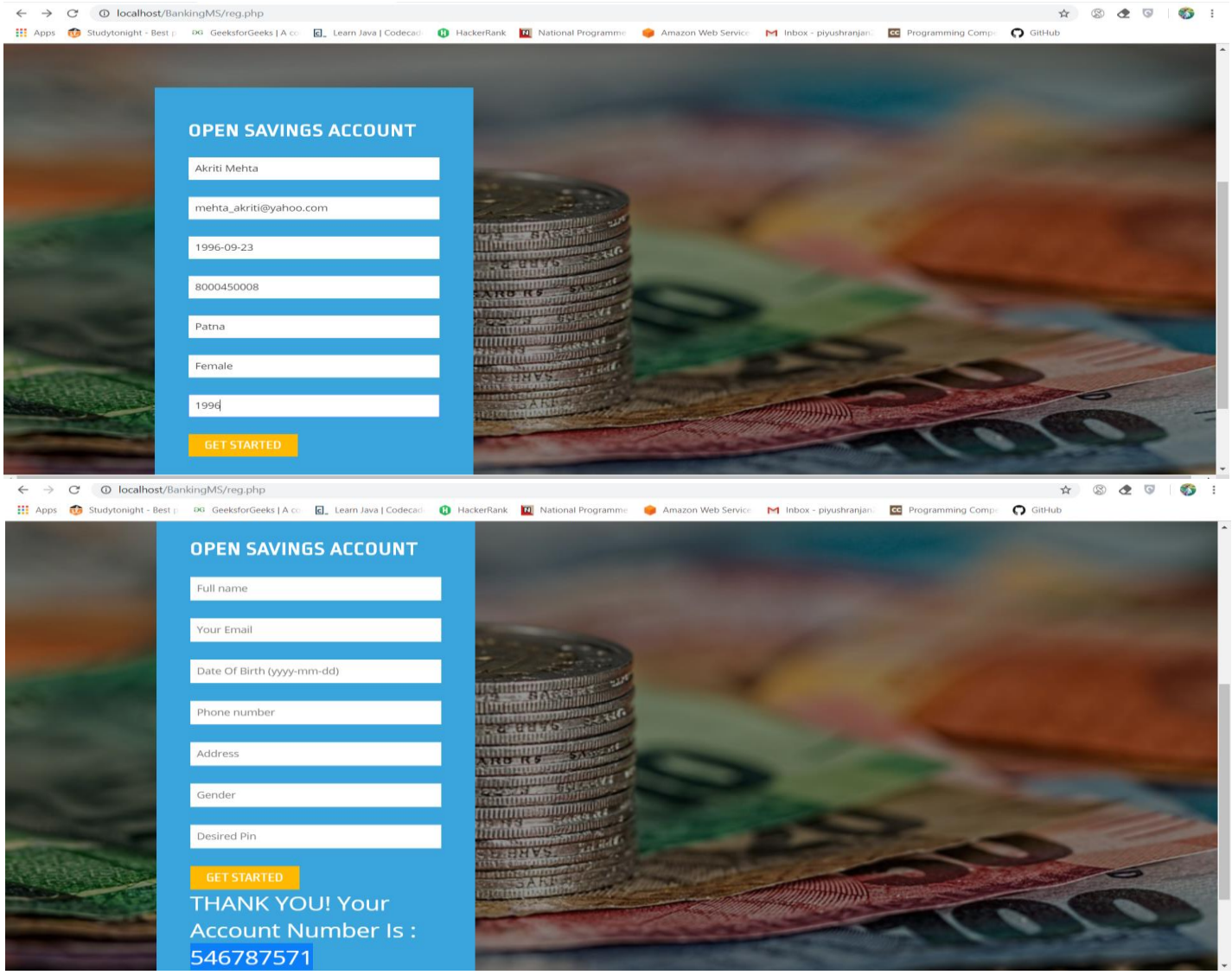


Fig 5.2.4: Customer Registration by admin

2) dep.php

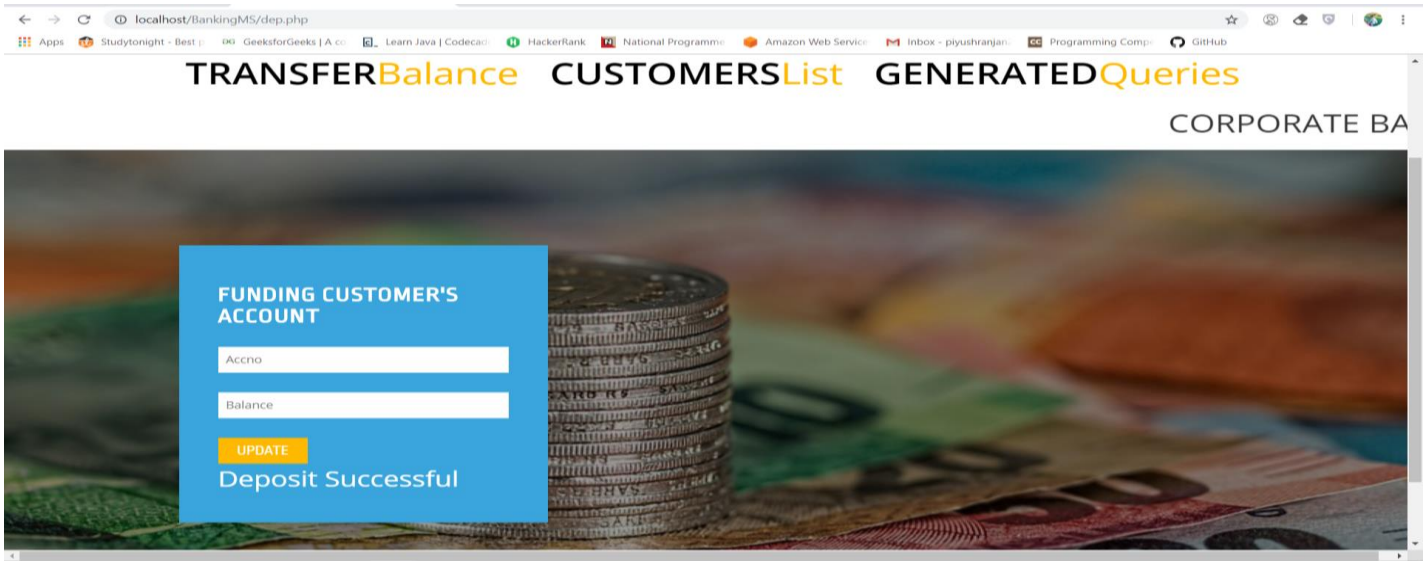


Fig 5.2.5:Account deposit by admin

3) wdl.php

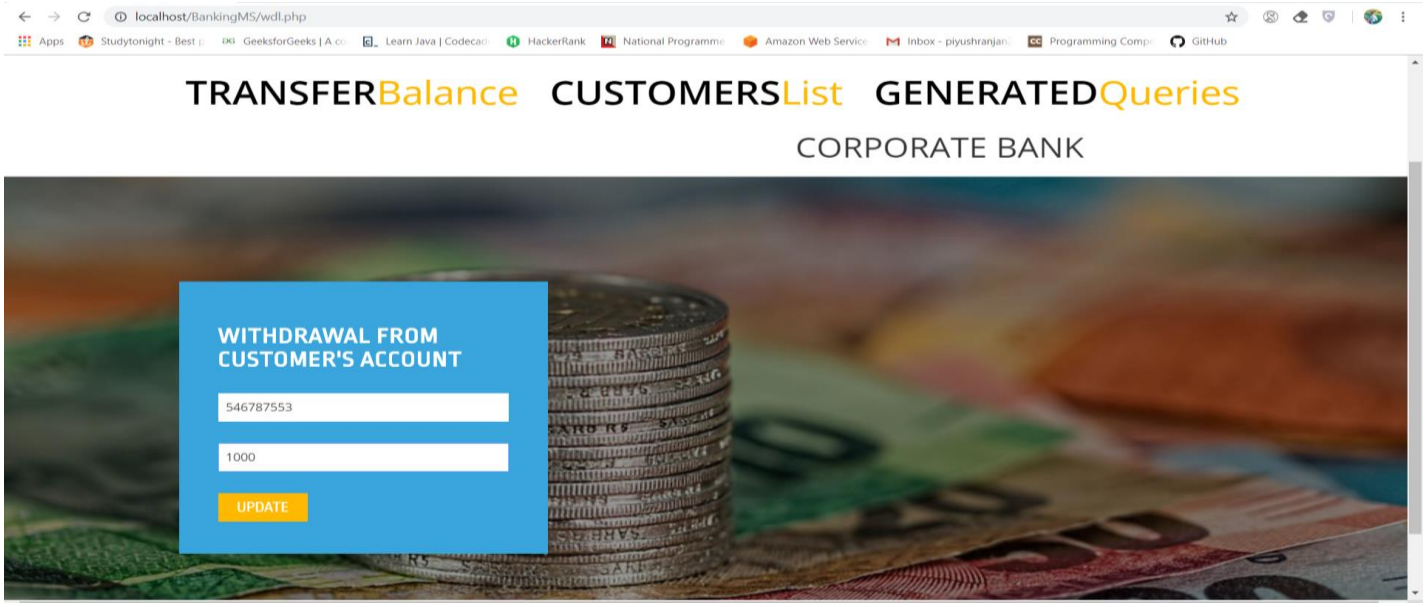
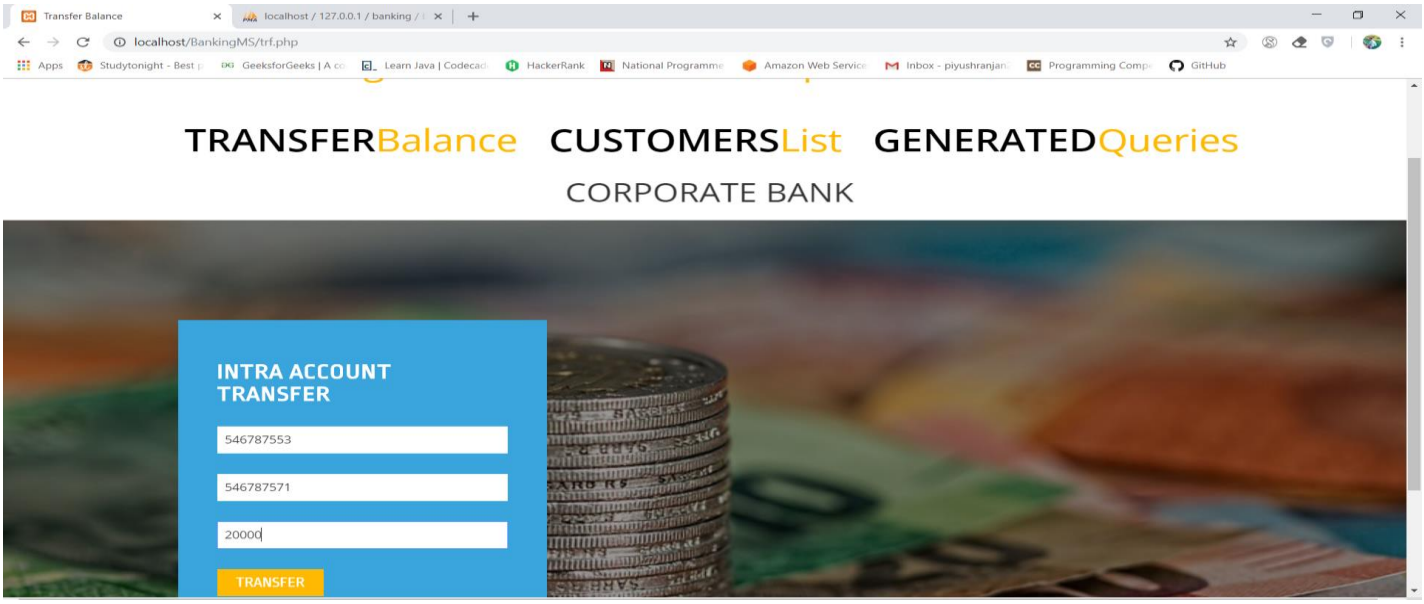


Fig 5.2.6:Account Withdrawal by admin

4) trf.php



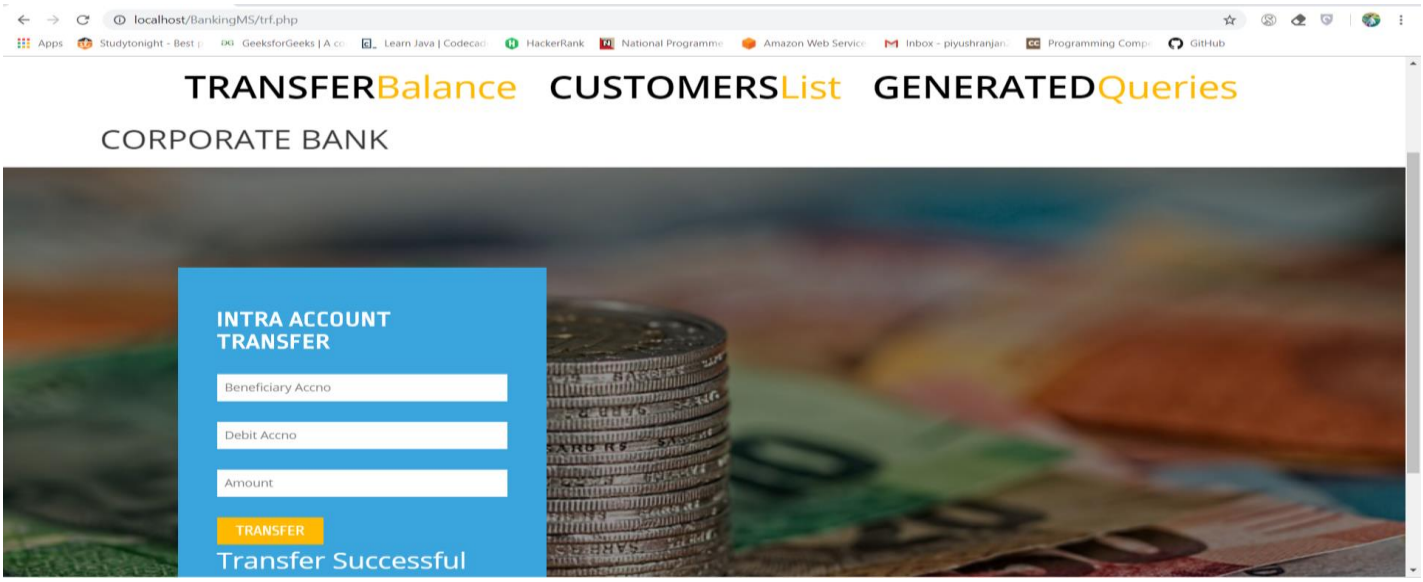


Fig 5.2.7:Account Balance Transfer by admin

5) clist.php

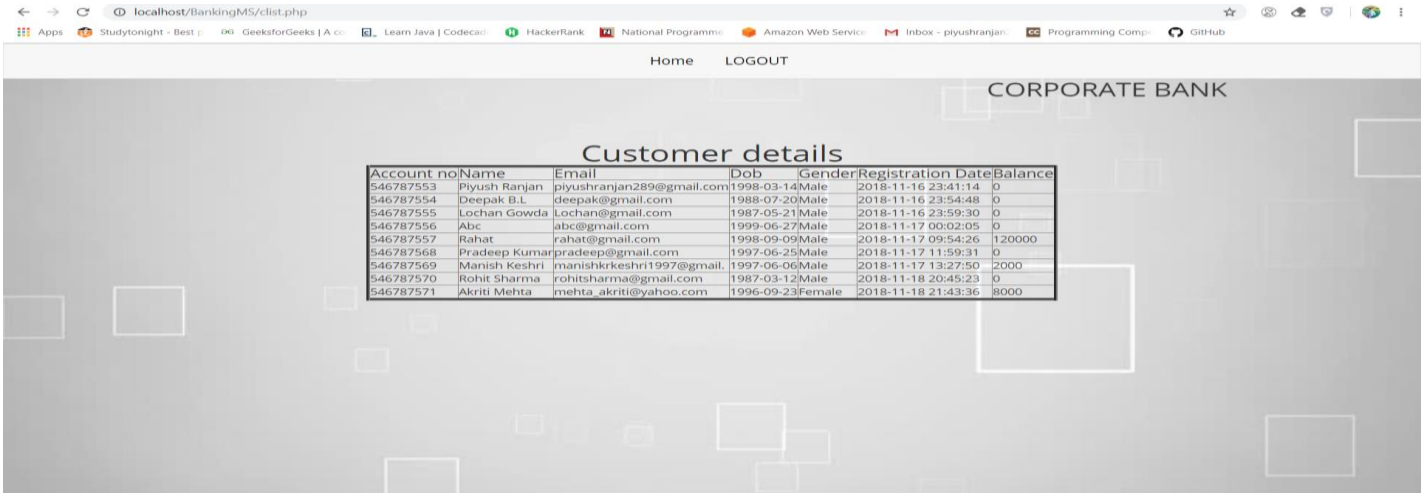


Fig 5.2.8:Displaying customer details from database

6) Query.php

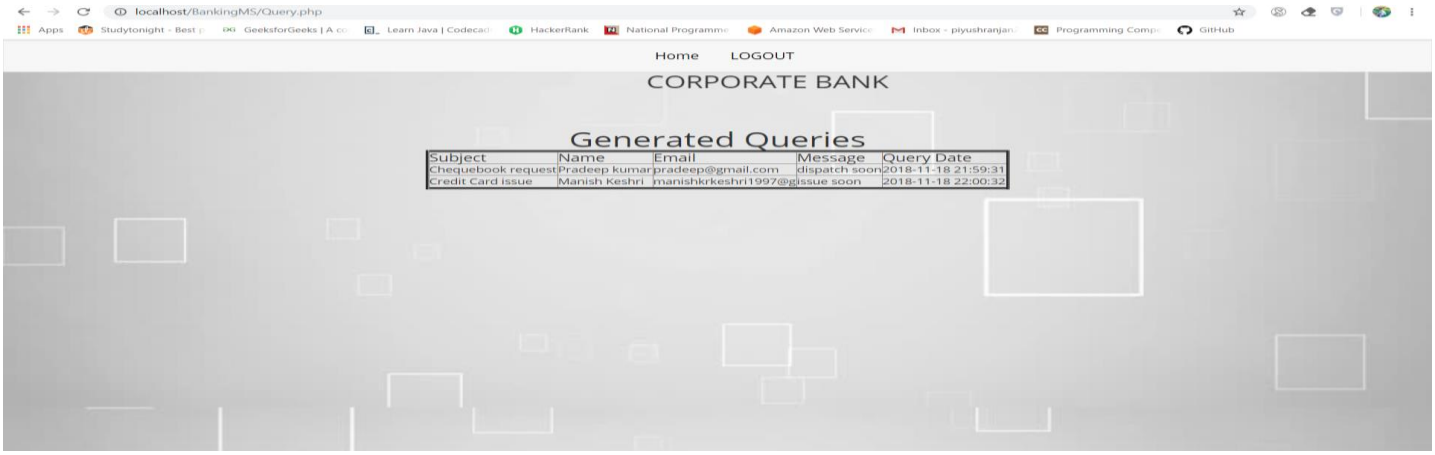
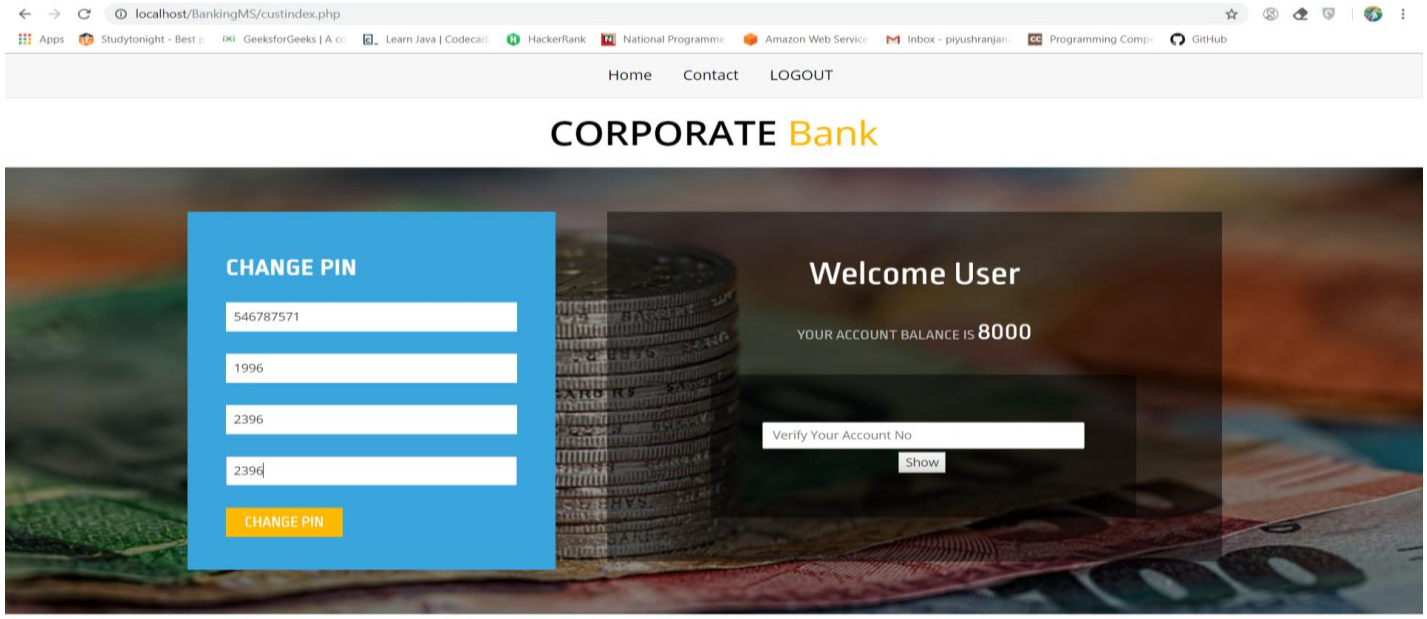


Fig 5.2.9:Displaying customer generated queries from database

Module 3 : Customer Panel

1) Change Pin window:



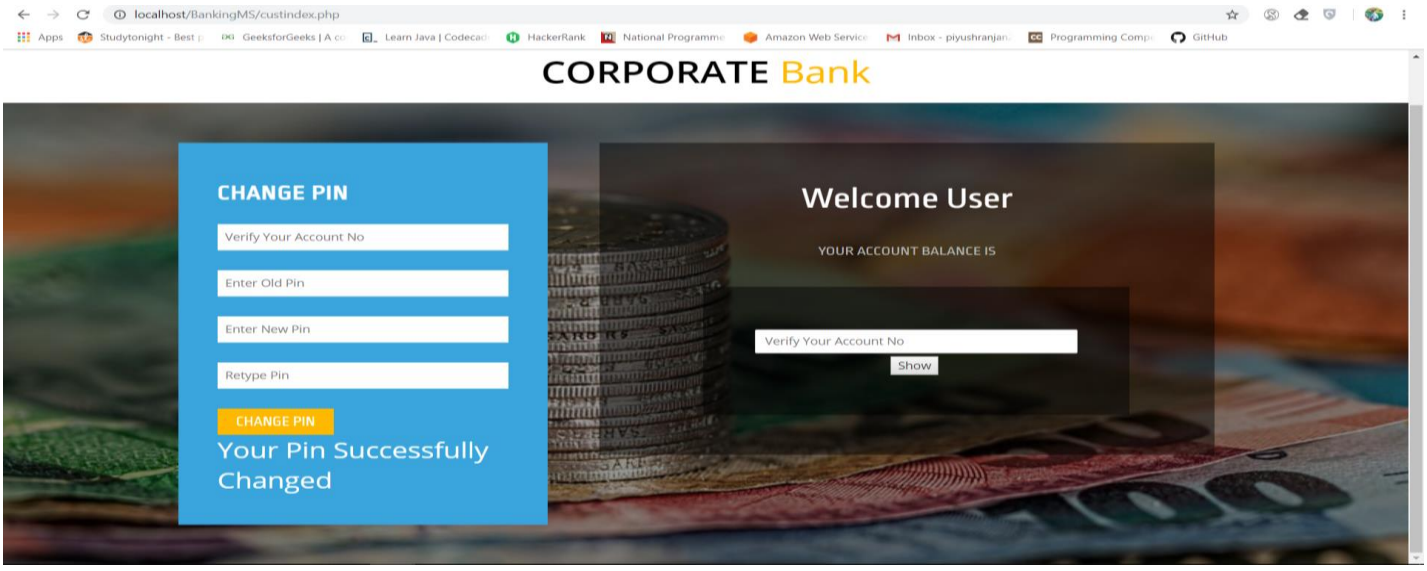


Fig 5.3.1:Change Pin window under customer panel

2) View Balance window

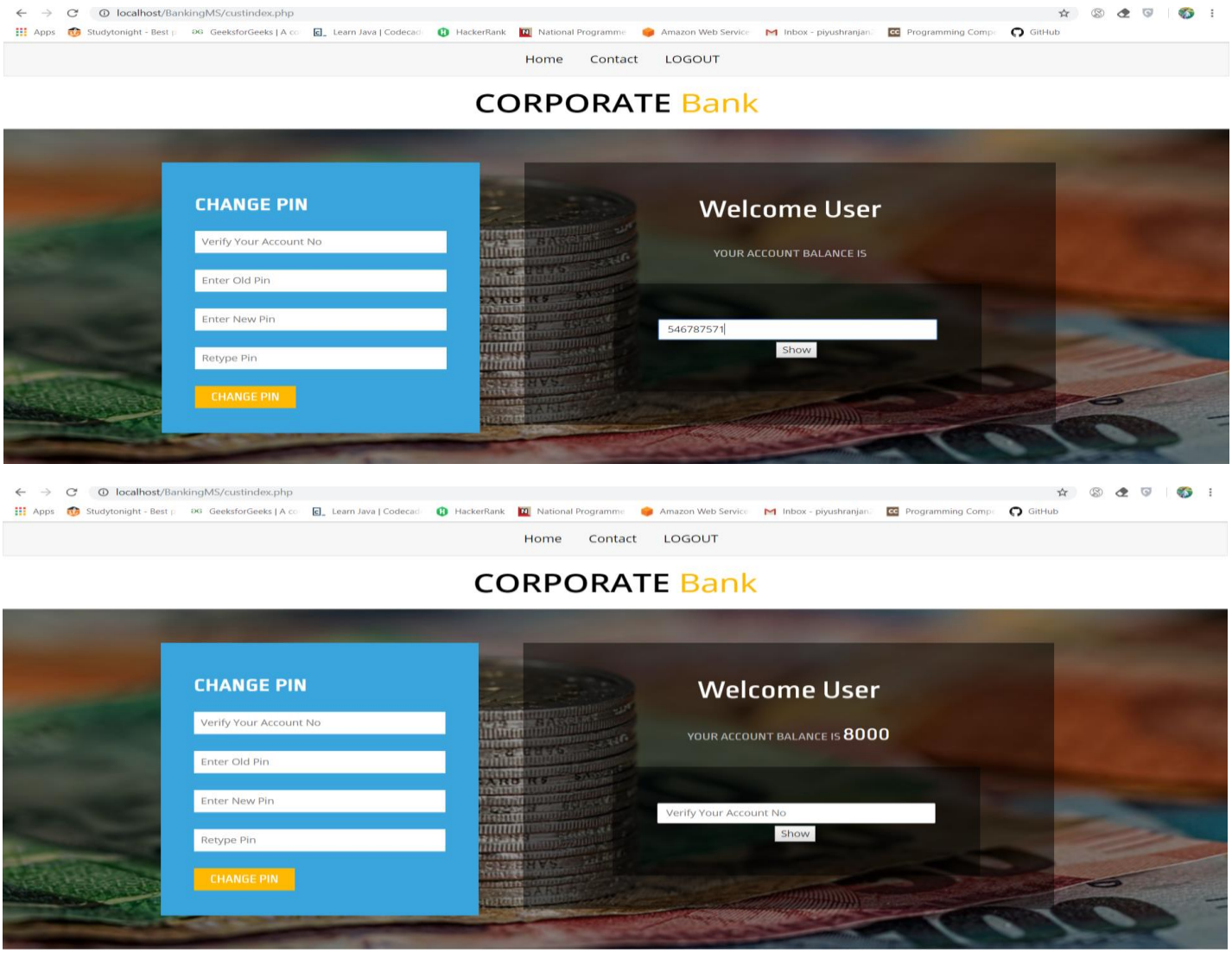


Fig 5.3.2:View balance window under customer panel

Module 4 : Services window

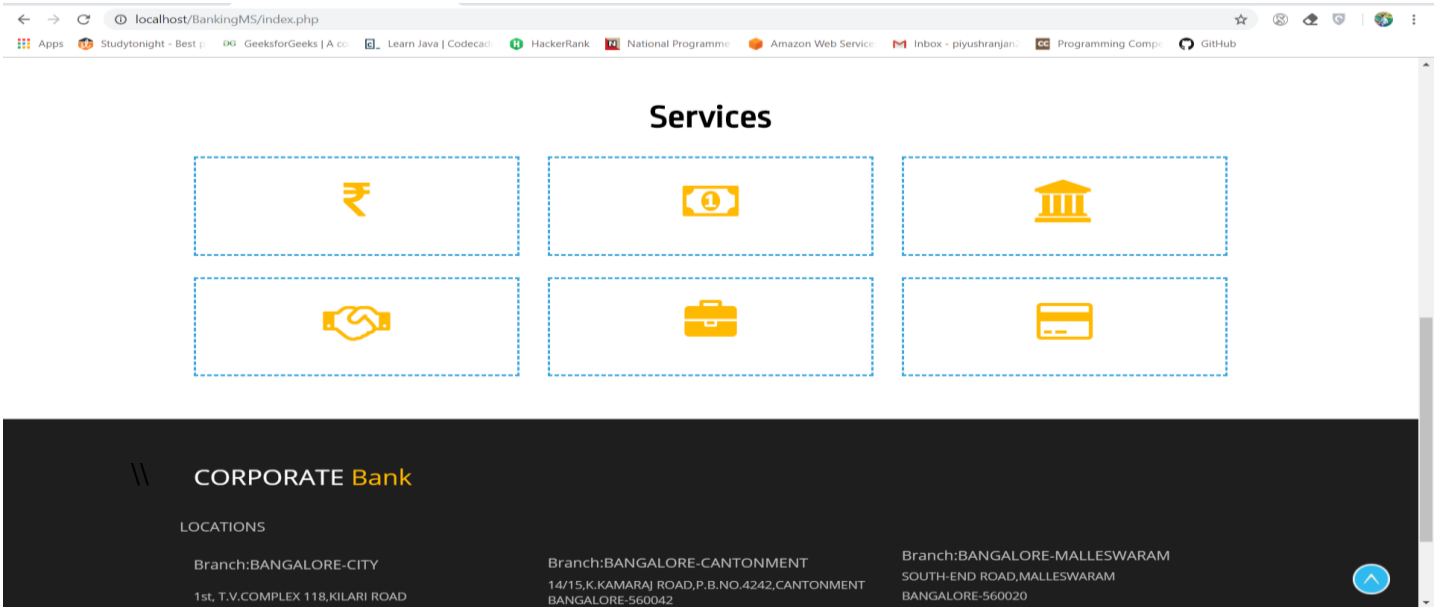


Fig 5.4 Services window

Module 5: About window

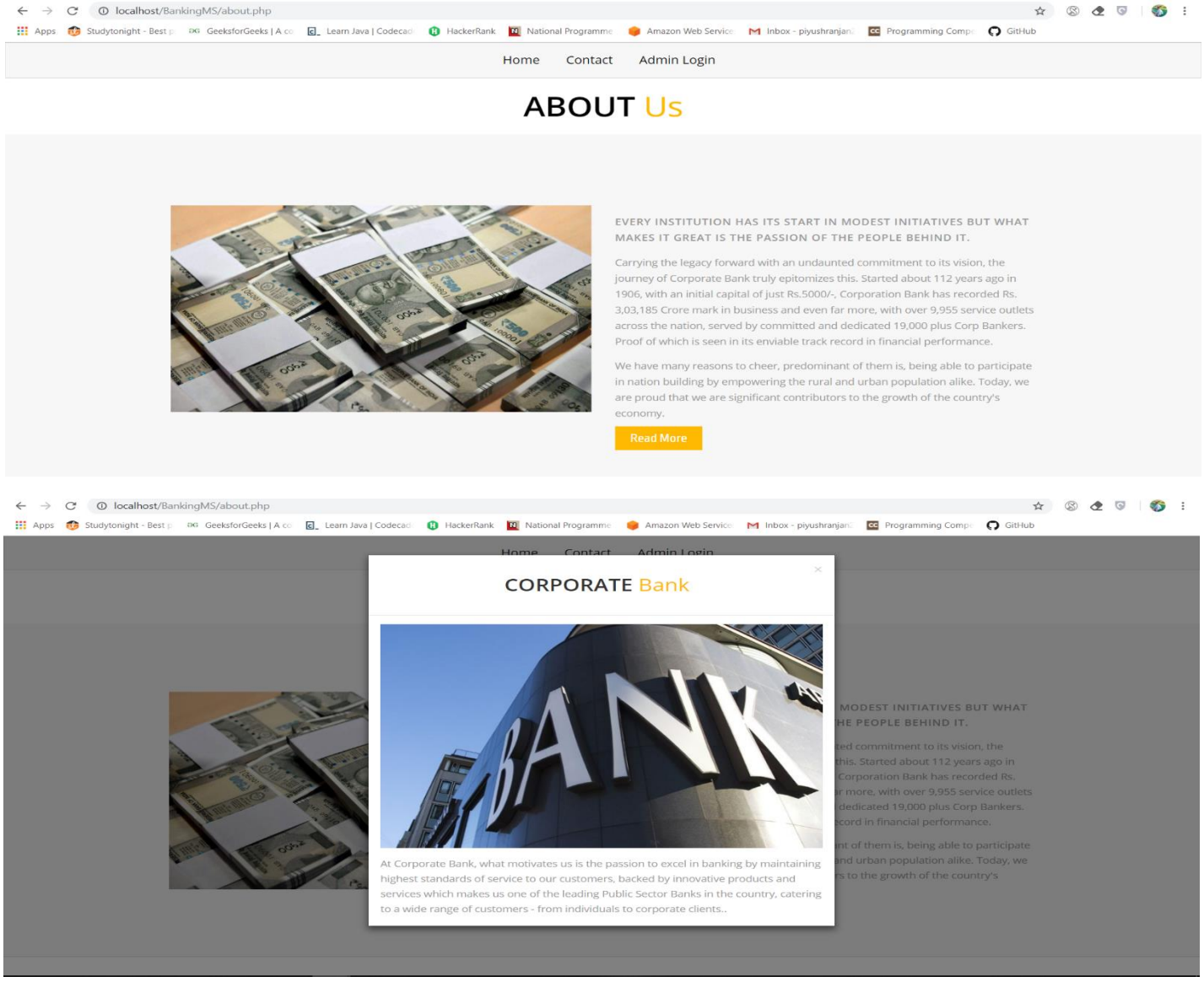


Fig 5.5:About window

Module 6 :Contact window

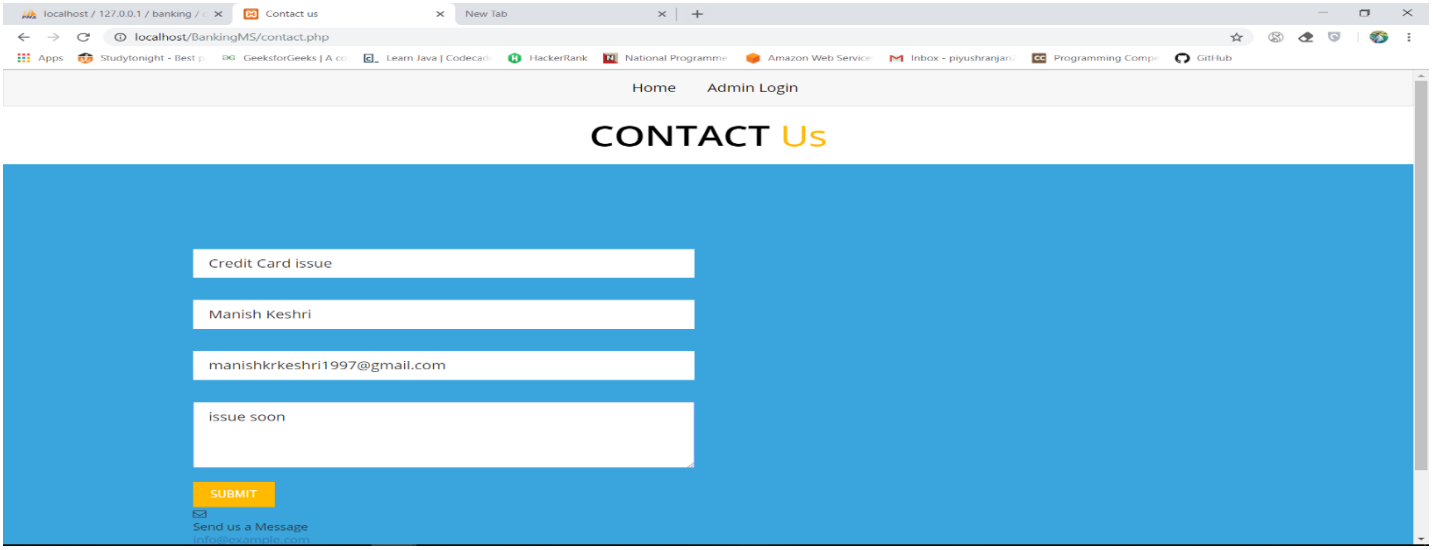


Fig 5.6:Contact Window

Chapter 6

Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.

Two Types of Testing are:

1.Unit testing

2. Integration testing

Functionality	Action	Expected Result	Actual Result	Test Result
Accepting login credentials for customer login	Login is clicked	Should enter customer panel page	Entering into customer panel Page	Pass
Creating a new savings account,if he/she is a new customer	Start Savings account button is clicked	Should enter into new registration page	Entering into new registration page	Pass
Accepting user input for new customer registration	Get started is clicked	Account number should be generated and displayed after successful registration	Displaying generated account number	Pass
Accepting user input for for changing pin under customer panel	Change pin button is clicked	Pin should be changed successfully and message should be displayed	Pin changed successfully and message dispalyed	Pass
Accepting user input to view balance	View balance button is clicked	Should display the customer's account balance correctly	Customer's account balance displayed correctly	Pass

Logout	Logout is clicked	Should successfully logout and display login page	Successfully redirected to Login page	Pass
Accepting login credentials for admin login	Login is clicked	Should enter Admin Panel Page	Successfully redirected to admin panel page	Pass
To generate the Customers list under admin panel	Customer List button is clicked	Customers List should generate	Customers List generated	Pass
Accepting user input for Funding customer's account under admin panel	Update balance button is clicked	Balance should be added and message should be displayed	Balance added and message displayed	Pass
Accepting user input for withdrawing from customer's account under admin panel	Update balance button is clicked	Balance should be withdrawn and message should be displayed	Balance withdrawn and message displayed	Pass
Accepting user input for transfering balance from customer's account under admin panel	Transfer balance button is clicked	Balance should be transferred and message should be displayed	Balance transferred and message displayed	Pass
To view customer generated queries under Login Panel	Generated Queries button is clicked	Should display the customer generated Queries	Customer generated Queries Displayed	Pass
Logout	Logout is clicked	Should successfully logout and display Home page	Successfully redirected to Home page	Pass

Chapter 7

Conclusion and Future Enhancements

- **Conclusion**

Our project is only a humble venture to satisfy the needs to manage their project work. Several user friendly coding also have adopted. This package shall prove to be a powerful package in satisfying all the requirements of the bank. The objective of a software planning is to provide a framework that enables the manager to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

The website Banking Management System provides a organized system to add customers, account deposits,account withdrawals,account transfers,etc. This reduces the time taken for the bank staff to add customers and perform the banking operations manually.

- **Future Enhancements**

1. Improving the user interface.
2. Should include features and operations in detail, including screen Layouts.
3. The project shall include a master-salve database structure to reduce overload on databases on regular basis on different servers.

References

- Fundamentals of Database systems, RamezElmasri and Shamkant B.Navathe, 7th edition, 2017, Pearson.
- Database Management System, Ramakrishnan, Gehrke, 3rd edition, 2017, McGraw Hill.
- Silberschatz Korth and Sudarshan, Database Sysetm concepts, 6th edition, McGraw Hill, 2013.
- http://www.tutorialspoint.com/mysql/
- httpd.apache.org/docs/2.0/misc/tutorials.html
- W3Schools
  http://www.w3schools.com/html/html_intro.asp
  http://www.w3schools.com/php/php_forms.asp
  http://www.w3schools.com/css/css_background.asp
- Stack OverFlow
  http://stackoverflow.com
- My sql Tutorial
  www.mysqltutorial.org

- The Bootstrap Website (for design) – https://getbootstrap.com/
- PHP Official Documentation – http://php.net/docs.php
- The MySQL Documentation – https://dev.mysql.com/doc
- Stack Exchange – https://stackexchange.com/
- Wikipedia – https://www.wikipedia.org