**Software Documentation for the Contributory Provident Fund (CPF)**

**Product Name: Contributory Provident Fund**

Release Version:

Testing Started:

Testing Completed:

Technical Lead:

Product Owner: BBA

Primary Tester:

Developer:

Types of Test Performed:

**Browser Tested:**

1. Chrome

2. Mozilla Firefox

**Device Tested:**

1. Desktop

2. Laptop

3. Tab

4. Smartphone

**Instance Tested:**

1. Development

2. Testing

3. Production

4. UAT

**Table of Contents**

[Revision History 3](#__RefHeading___Toc2357_1497391501)

[Design Specification 4](#__RefHeading___Toc2359_1497391501)

[Abstract 5](#__RefHeading___Toc2361_1497391501)

[Chapter-1: Introduction 6](#__RefHeading___Toc2365_1497391501)

[1.1 Review of the CPF System 6](#__RefHeading___Toc2367_1497391501)

[1.2 Motivation 6](#__RefHeading___Toc2369_1497391501)

[1.3 Aims and Objective 6](#__RefHeading___Toc2371_1497391501)

[1.4 Solution Methodology 6](#__RefHeading___Toc2373_1497391501)

[Chapter - 2. Background Study 7](#__RefHeading___Toc2375_1497391501)

[2.1 Contributory Provident Fund 7](#__RefHeading___Toc2377_1497391501)

[2.2 Why should we use this software system ? 7](#__RefHeading___Toc2379_1497391501)

[2.3 User Experience 7](#__RefHeading___Toc2381_1497391501)

[2.4.4 Benefits 7](#__RefHeading___Toc2389_1497391501)

[2.4.5 Report & Analysis 7](#__RefHeading___Toc2391_1497391501)

[2.5 CPF main modules 8](#__RefHeading___Toc2393_1497391501)

[Chapter – 3: Software Requirement Analysis 9](#__RefHeading___Toc2395_1497391501)

[3.1 Introduction 9](#__RefHeading___Toc2397_1497391501)

[3.2 Problem Analysis 9](#__RefHeading___Toc25949_827625388)

[3.3 Requirement 9](#__RefHeading___Toc2399_1497391501)

[3.4 Software Requirement Specifications 9](#__RefHeading___Toc2401_1497391501)

[3.4.1 SRS Scopes 9](#__RefHeading___Toc2403_1497391501)

[3.4.2 Functional Requirements 9](#__RefHeading___Toc2405_1497391501)

[3.4.3 Non-functional Requirement 9](#__RefHeading___Toc2407_1497391501)

[3.5 Interface Requirement 10](#__RefHeading___Toc2409_1497391501)

[3.5.1 User Interface 10](#__RefHeading___Toc2411_1497391501)

[3.5.2 Hardware Interface 10](#__RefHeading___Toc2413_1497391501)

[3.5.3 Software interface 11](#__RefHeading___Toc2415_1497391501)

[3.5.4 Communication Interface 11](#__RefHeading___Toc2417_1497391501)

[3.6 SRS table 11](#__RefHeading___Toc2419_1497391501)

[3.6 Report Analysis 11](#__RefHeading___Toc2421_1497391501)

[Chapter – 4: System Design 13](#__RefHeading___Toc2423_1497391501)

[4.1 Algorithm Design 13](#__RefHeading___Toc2634_1971331298)

[4.2 Process Flow Diagram 13](#__RefHeading___Toc2431_1497391501)

[4.3 Data flow Diagram 14](#__RefHeading___Toc2634_19713312982)

[Chapter – 5: Testing Documentation 15](#__RefHeading___Toc2481_1497391501)

[5.1 Introduction 15](#__RefHeading___Toc2483_1497391501)

[5.2 Scopes 15](#__RefHeading___Toc2485_1497391501)

[5.3 Quality Objective 15](#__RefHeading___Toc2487_1497391501)

[5.4 Testing Methodology 15](#__RefHeading___Toc2489_1497391501)

[5.4.1 Model 16](#__RefHeading___Toc2491_1497391501)

[5.4.2 Strategies 16](#__RefHeading___Toc2493_1497391501)

[5.4.3 Levels 16](#__RefHeading___Toc2495_1497391501)

[5.4.4 Test Deliverables 16](#__RefHeading___Toc2511_1497391501)

[5.4.5 Tools 17](#__RefHeading___Toc2513_1497391501)

[Test Cases with Result 17](#__RefHeading___Toc8692_1469782388)

[Test Scenario 1: Dashboard 17](#__RefHeading___Toc8694_1469782388)

[Test Scenario 2: Opening Balance 17](#__RefHeading___Toc8696_1469782388)

[Test Scenario 3: Sub-Opening Balance 20](#__RefHeading___Toc8698_1469782388)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Design Specification

**Overview:**

CPF stands on Central Provident Fund that is the vital part of Finance & Account in Management Control System. In the account section, CPF enrolls in the back-end and front-end management system. Server side and client side communication make the system user-friendly.

**The Back-end part:**

1. Node.js

2. Database → Oracle

3. MS Server Version 2.0

**The Front-end part:**

1. React.js

2. Bootstrap

3. Domain → Friendly

# Abstract

The Contributory Provident Fund is an employment-based savings scheme with the help of employees contributing a mandated amount to the fund for their benefits. Provident fund is a fund through which a company provide employees with payments when they leave their place of employment. Our aim is to design a software system that makes this provident fund system user-friendly to the whole BBA community.

# Chapter-1: Introduction

In this chapter we will focus on the status of the current management system and some of the goals for the progress of this system. That is why we will continue to move towards the future by overviewing some current systems. In the following sections, these terms have been discussed in details.

https://www.fmassociatesbd.com/provident-fund-laws-and-audit-in-bangladesh

## 1.1 Review of the CPF System

As per the provident fund laws in Bangladesh, which can be understood upon reading the Labour Act 2006 in conjunction with the Labour Rules 2015, it is noteworthy that besides what the employees contribute to the provident fund in Bangladesh, employers are also required to contribute a matching amount to the fund. However, provident fund must not be confused with the pension fund.

As per the provident fund laws in Bangladesh, a Board of Trustee must be formed by a company which shall administer the provident fund. The law requires that, a Board of Trustee must be comprised of equal number of representatives of the employer and employees (section 264(5) of the Labour Act).

## 1.2 Motivation

Contributory Provident fund system makes the employee & company or any other organization easier to understand their funding system. Thus it helps administrator to create a system that can manages the opening & sub-opening balance.

## 1.3 Aims and Objective

Our aims is to achieve a good transport system that has ability to fulfill our requirements.

* Platform Independent
* Auto Synchronizing
* Friendly Environment
* Domain and Hosting
* Generate Each section Report
* Multi-functionality

## 1.4 Solution Methodology

1. Design opening balance section and create & manage section operate this module.
2. Design sub-opening balance section and create & manage section operate this module.

# Chapter - 2. Background Study

## 2.1 Contributory Provident Fund

Contributory Provident Fund means a fund in which subscriptions or deposits of any class or classes of employees are received and held on their individual accounts, and includes any contributions and any interest or increment accruing on such subscriptions, deposits or contributions under the rules of the Fund.

In relation to a company, means a person liable to contribute to the assets of the company in the event of its being wound up, and includes the holder of fully paid shares in the company and, prior to the final determination of the persons who are contributories, includes any person alleged to be a contributory;

## 2.2 Why should we use this software system ?

The purpose of this software is to define the functionality and specifications of the design of a web application for Managing Employees and their payroll. The expected audiences of this software system are the developers and the admin of the web application. Now with the help of this system the admin has the information on his finger tips and can easily prepare a good record based on their requirements. Finally, we can say that this system will not only automate the process but save the valuable time of the manager or the admin, which can be well utilized buy his institute. This will be an additional advantage and management of power based on their free time from his normal duty.

## 2.3 User Experience

This software system is built keeping in mind that it is to be used by only one user that is the admin.

### 2.4.4 Benefits

* Save time - Manually it takes a lot of time while storing the lease information
* Resources - except this software system, the gathering of lease information will be

difficult as there need a lot of man power resources in rural and urban areas.

* Reusable - the system can be reused in the similar types of management.
* It is cost effective as the user control the web application himself and does not go for

professional service.

* Validating procedures and checks restrict user from making mistakes.
* The software is easy to use and is user friendly so no expertise is required.
* The calculations are automated so no chance of error.

### 2.4.5 Report & Analysis

The software system has been developed to overcome the problems faced in the practicing of manual system. This software is built to eliminate and in some cases reduce the hardships faced by the existing system. Moreover this system is designed for particular need of the company to carry out its operations in a smooth and effective manner. This web application is reduced as much as possible to avoid errors while entering data. It also provides error message while entering invalid data. It is user-friendly as no formal knowledge is required to use the system. Human resource challenges are faced by every organization which has to be overcome by the organization. Every organization has different employee and payroll management needs. Therefore I have design exclusive Employee and payroll Management System that are adapted to the organization’s Managerial Requirements.

## 

## 2.5 CPF main modules

1. Dashboard
2. Opening Balance

i) Create

ii) Manage

1. Sub-opening Balance

i) Create

ii) Manage

# Chapter – 3: Software Requirement Analysis

## 3.1 Introduction

A software requirement is a field within software engineering that deals with establishing

the needs of stakeholders that are to be solved by software. The IEEE Standard Glossary of

Software Engineering Terminology defines requirements.

## 3.2 Problem Analysis

The process of manually calculating salary causes some problems in the management. Problems faced by several section today are:

1. A large number of company employees who constantly growing raises wage calculation process becomes longer and not effective.
2. The possibility of human error in the calculation of salaries
3. To search for data takes a long time because data is not stored properly.

## 3.3 Requirement

The Contributory Provident Fund Management System Requirements provides high-level of functional requirements.

## 3.4 Software Requirement Specifications

* Documentation (user and administration)
* Operational behavior
* Technical Support

### 3.4.1 SRS Scopes

The CPF system is part of the Finance and Accounts management system. Activities carried out from the accounts system are processes of data from the BBA management of Accounts & finance under the management control system .

### 3.4.2 Functional Requirements

Every section has a specific functionality that’s are given below:

* Login Functionality
* Opening & sub-opening balance availablity
* Creation & management for each balance functionality

### 3.4.3 Non-functional Requirement

For an specific criteria, there should be applied non-functional requirements which are

shortly described below:

**Reliability**

The operational interface can’t fail within a period of time.

**Availability**

The system is available or not while a user wants access the information.

**Security**

All user can login to the system with secured procedure. Each user has a unique

identification system. Admin can store user’s login information.

**Maintainability**

All source code and related document should be controlled under a version of the

system.

**Portability**

As the software run into several platform, the platform independence should be ensure.

Different OS can access the software easily.

**Usability**

The Software should be easy to use and given a positive experience by the

user’s.

## 3.5 Interface Requirement

Computer system including software and hardware can turn the system on by interfacing

them with human thought.

### 3.5.1 User Interface

1. Login Screen
2. Home Screen
3. Balance & sub-opening balance information
4. create & manage sections

### 3.5.2 Hardware Interface

* Server Configuration
* CPU/Processor
* Windows with apache preloaded
* Client Configuration

### 3.5.3 Software interface

* OS = Windows, Linux
* Language = PHP, JS
* DB = Oracle

### 3.5.4 Communication Interface

These interfaces include E-mail, Web Browser, Server Network and use http or ftp for

security purposes.

## 3.6 SRS table

It describes necessary behavior of the software system and shows in the table

|  |  |  |
| --- | --- | --- |
| S.No. | Requirement Name | Requirement Classified |
| 1. | Dashboard |  |
| 2. | Opening Balance |  |
| 3. | create |  |
| 4. | manage |  |
| 5. | Sub-opening balance |  |
| 6. | create |  |
| 7. | manage |  |

## 3.6 Report Analysis

Independent to each other. At the end of the phase, the document of the requirement will be provided.

**Priority Checklist**

* Level-1**:** Must Be Performed
* Level-2: Must be perform for next level
* Level -3: Not Mandatory

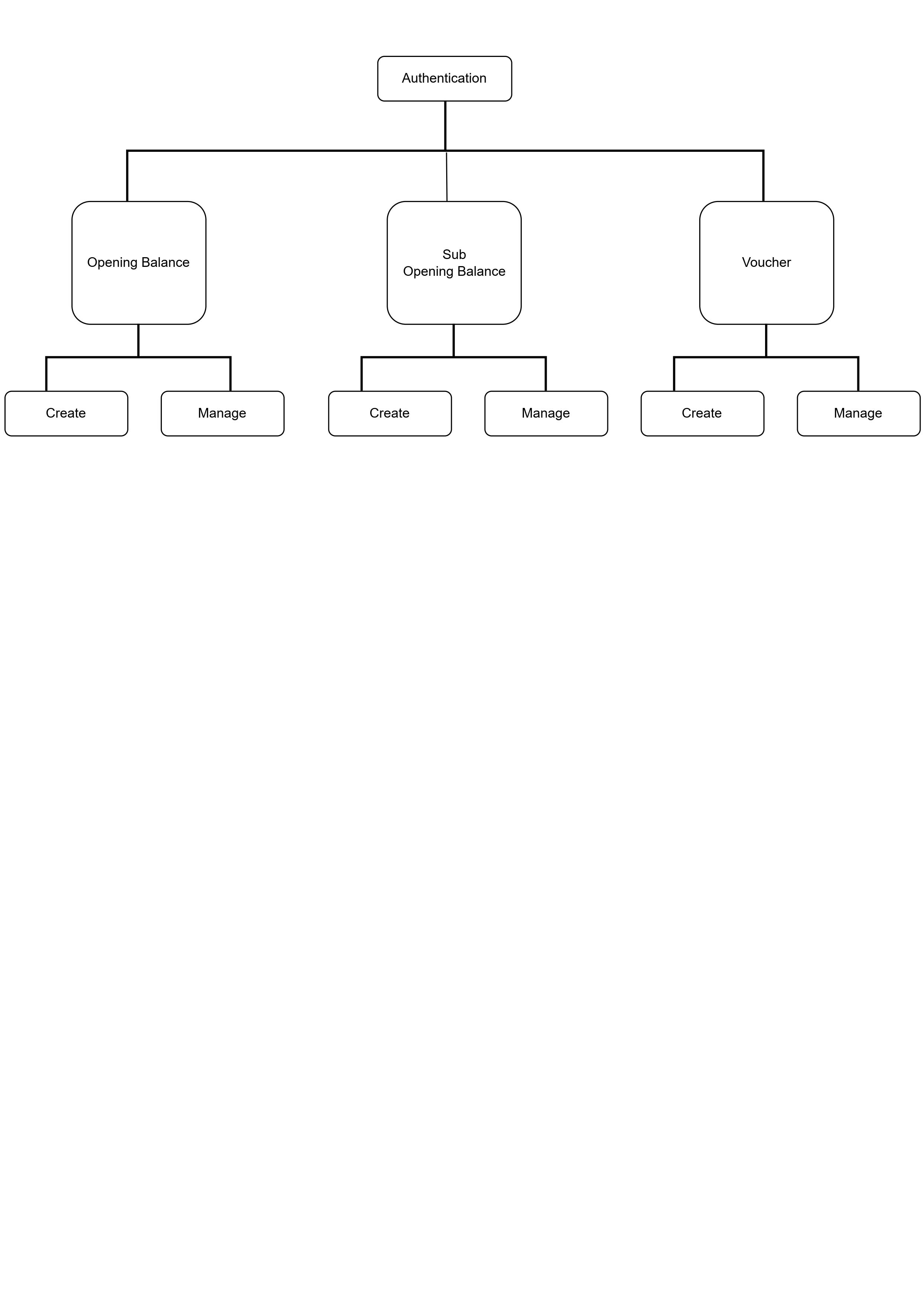
|  |  |  |  |
| --- | --- | --- | --- |
| SI.No. | Requirement Name | Functional/Non-  functional | Priority |
| 1. | **Dashboard** | Functional | 2 |
| 2. | **Opening Balance** | Functional | 2 |
| 3. | create | Functional | 1 |
| 4. | manage | Functional | 1 |
| 5. | **Sub-opening balance** | Functional | 2 |
| 6. | create | Functional | 1 |
| 7. | manage | Functional | 1 |
| 8. | Reliability | Non-functional | 1 |
| 9. | Availability | Non-functional | 1 |
| 10. | Security | Non-functional | 1 |
| 11. | Maintainability | Non-functional | 1 |
| 12. | Portability | Non-functional | 1 |
| 13. | Usability | Non-functional | 1 |
| 14. | Database | Functional | 1 |

# Chapter – 4: System Design

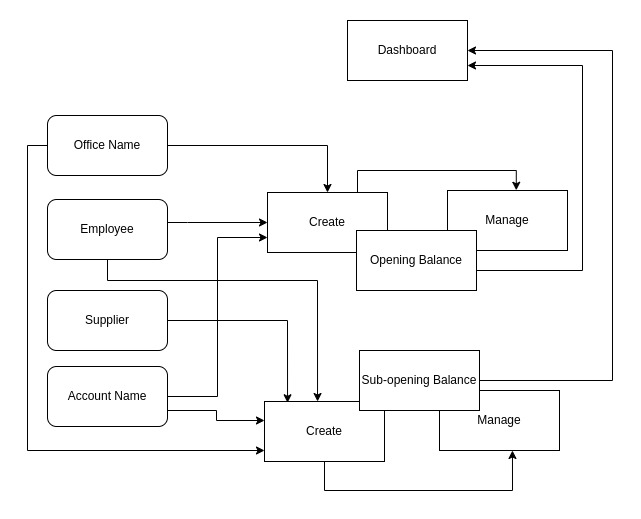
## 4.1 Algorithm Design

|  |  |
| --- | --- |
| STEP – 1: START  STEP – 2: CHECK Authentication  STEP – 3: CALL Fun(X)  STEP – 4: if(X = ‘Opening Balance’)  SELECT Y  if(Y == ‘create’)  ADD openingBalance  if(Y == ‘manage’)  MANAGE openingBalancelist | if(X = ‘Sub Opening Balance’)  if(Y == ‘create’)  ADD subopeningBalance  if(Y == ‘manage’)  MANAGE sopeningBalancel  STEP – 5: SHOW dashboard  STEP – 6: END |

## 4.2 Process Flow Diagram



## 4.3 Data flow Diagram



# Chapter – 5: Testing Documentation

## 5.1 Introduction

The Test Plan is designed to prescribe the scope, approach, resources, and schedule of all testing activities of the transport Management System of BBA. The plan identify the items to be tested, the features to be tested, the types of testing to be performed, the personnel responsible for testing, the resources and schedule required to complete testing, and the risks associated with the plan.

## 5.2 Scopes

**1. In scope:**

**For Admin**

|  |  |  |
| --- | --- | --- |
| Name | Applicable Roles | Description |
| Dashboard | admin | Display dashboard |
| Opening balance – create & manage | admin | Create a CPF opening balance and manage them through management section |
| Sub-opening Balance – create & manage | admin | Create a CPF sub-opening and manage them through management section |

**2. Out scope**

* User Interface (UI)
* Hardware
* System security and performance

## 5.3 Quality Objective

The test objectives are to verify the Functionality of the Central Provident Fund System, the project should focus on testing the transportation system such as vehicle requisition, vehicle information, add new vehicle information to guarantee all these operation can work normally in real business environment.

## 5.4 Testing Methodology

### 5.4.1 Model

* **Waterfall model** – a dynamic approach where each step depends on the previous one, where the developers follow some sequence of steps progressively downwards forwards to achieve the goal. It is like a waterfall approach containing several phases.

### 5.4.2 Strategies

• Tester has access to the full view of code.

• It is known as transparent or glass box testing

• Avoiding errors and wasting time at an early stage

• White box testing contains various tests -

1. Basic Path testing
2. Loop testing
3. Condition testing
4. Memory perspective testing
5. Performance testing

### 5.4.3 Levels

* Integration testing
* System testing
* Cross Browser Compatibility testing

User Acceptance Test (UAT)

* API testing

### 5.4.4 Test Deliverables

(a) Before Testing

* Test plans document
* Test case document

(b) After Testing

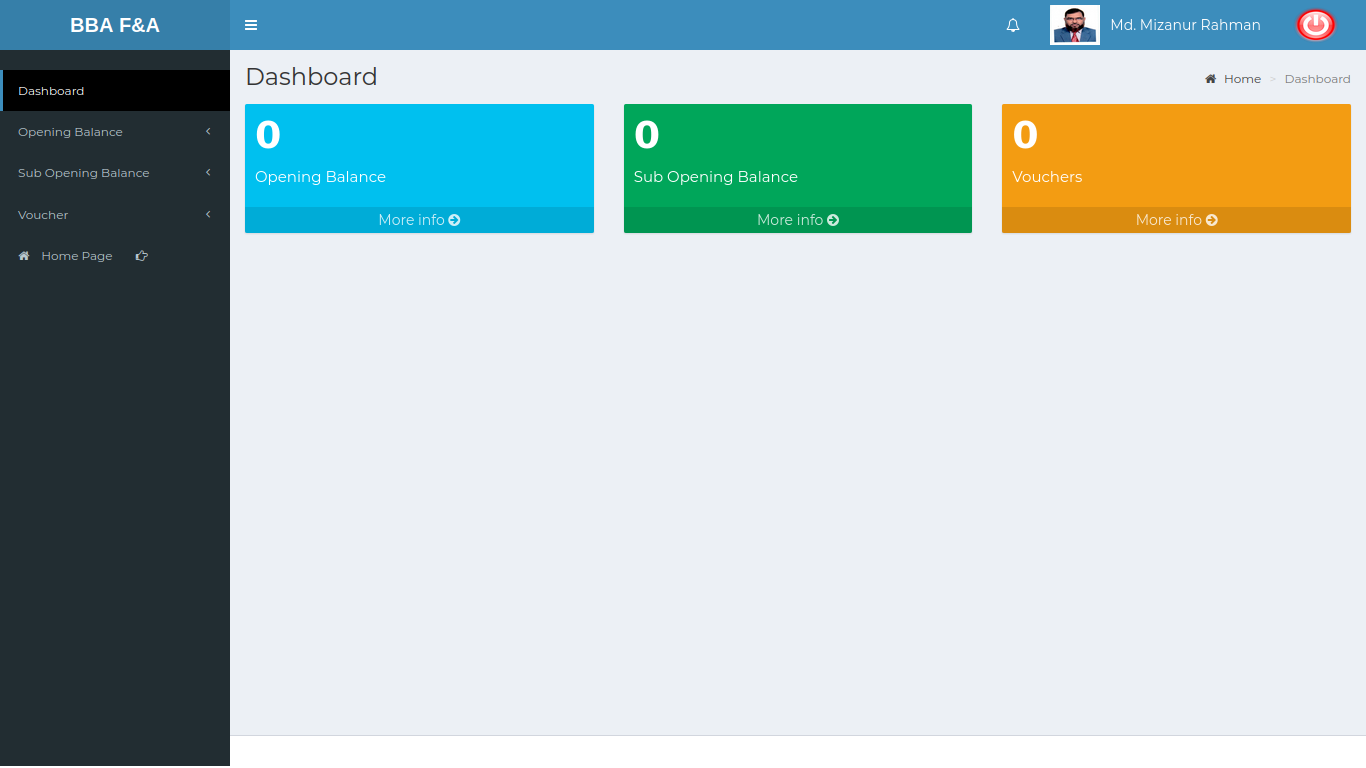
* Test Results/reports­
* Defect Report
* Installation/ Test procedures guidelines

### 5.4.5 Tools

|  |  |  |
| --- | --- | --- |
| No. | Resources | Description |
| 1. | Server | Oracle Database Server |
| 2. | Network | LAN line. |
| 3. | Computer | PC and other OS configuration |

# Test Cases with Result

## Test Scenario 1: Dashboard



Positive Test Case:

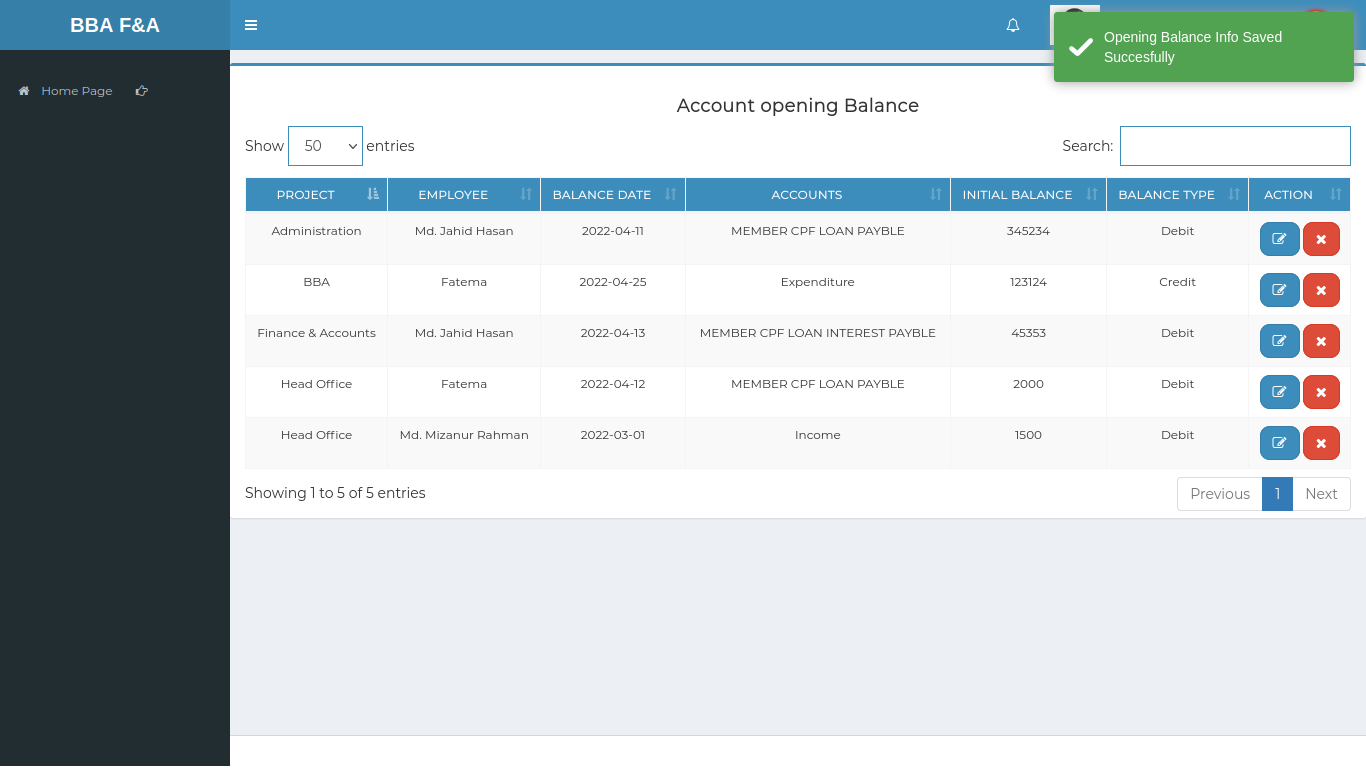
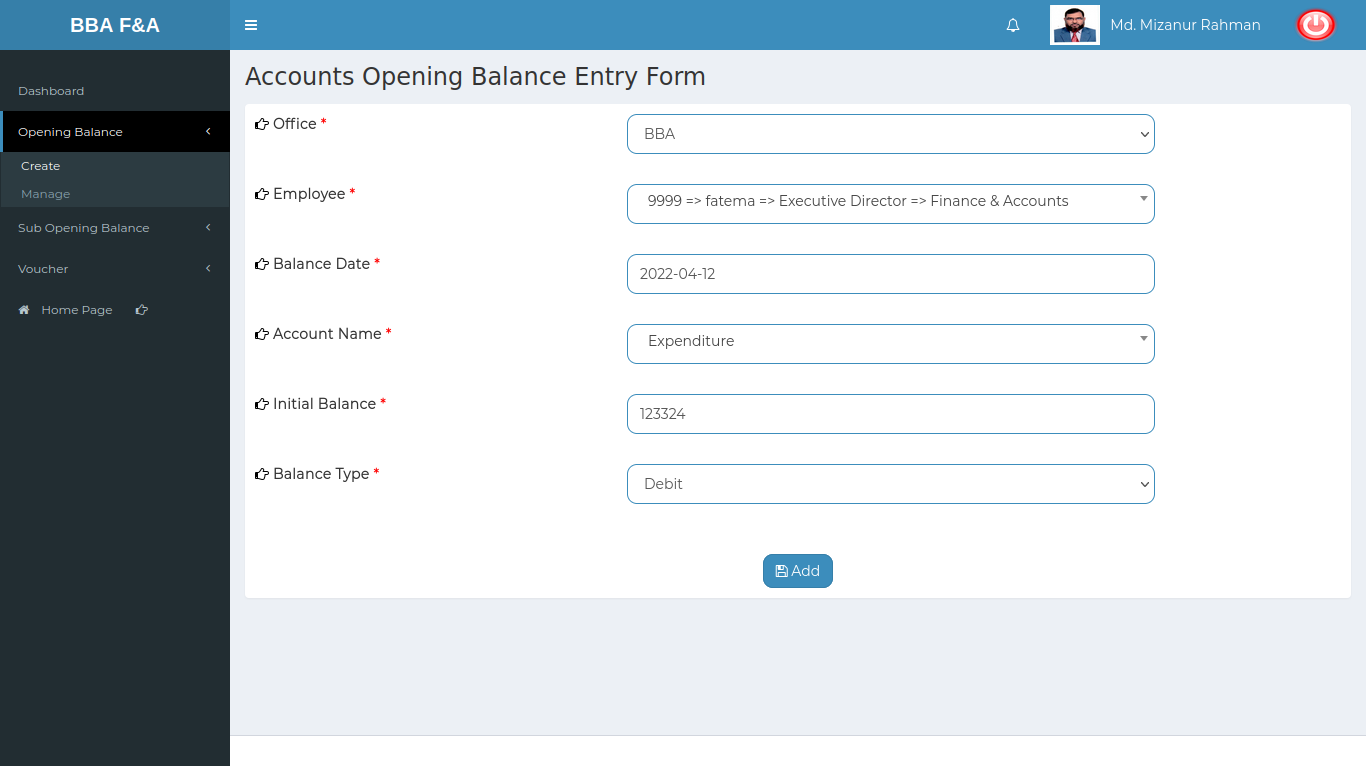
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Opening Balance |  |  |  |  |
| 2 | Sub-Opening Balance |  |  |  |  |
| 3 | Vouchers |  |  |  |  |

Negative Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Opening Balance |  |  |  |  |
| 2 | Sub-Opening Balance |  |  |  |  |
| 3 | Vouchers |  |  |  |  |

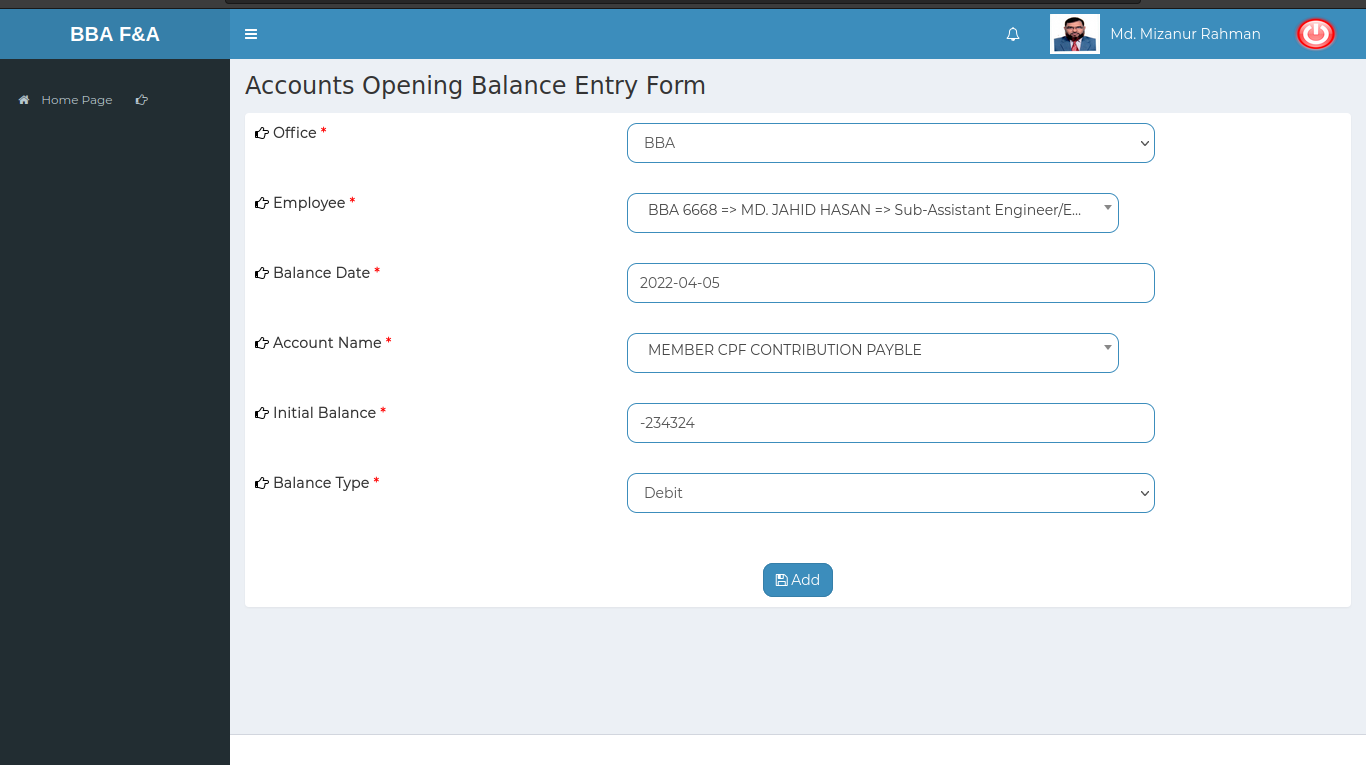
## Test Scenario 2: Opening Balance

1. Create



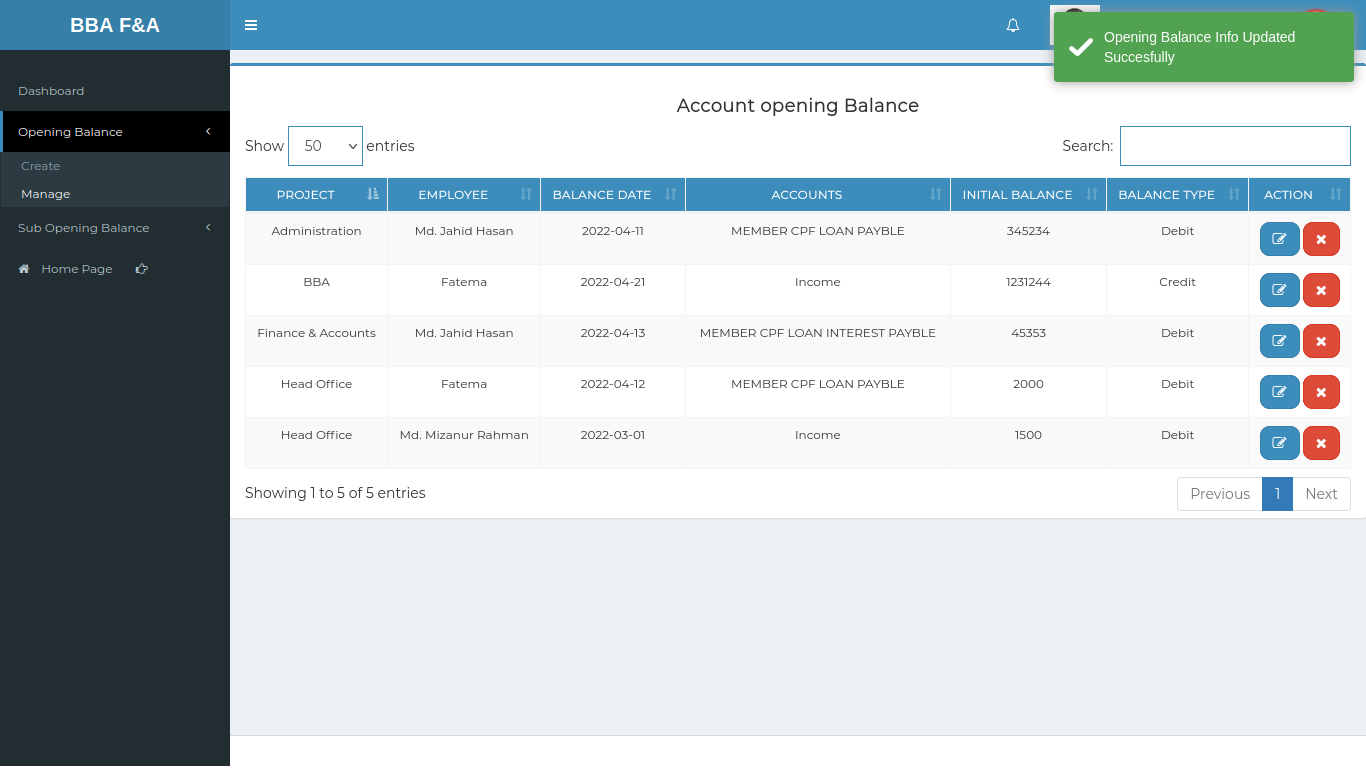
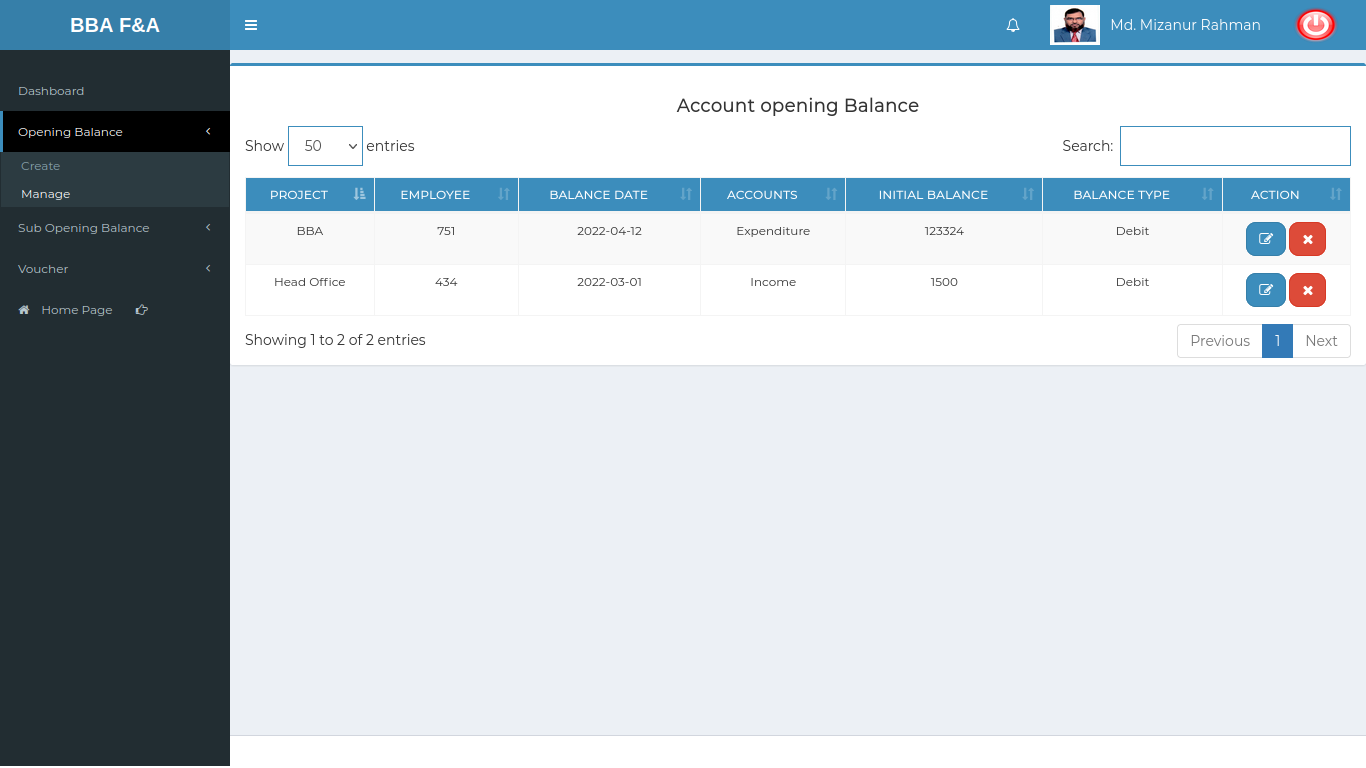
Positive Test Case:

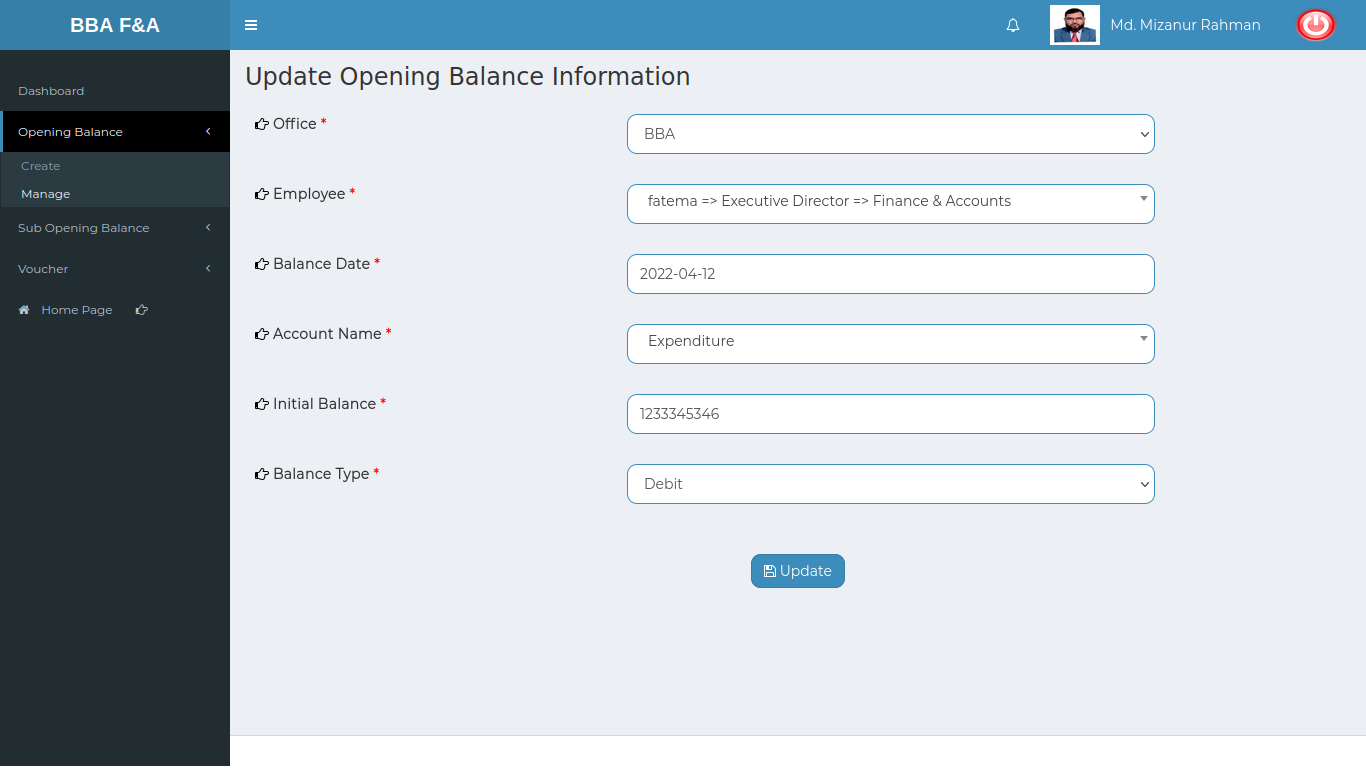
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Accounts Opening Balance Entry | 1. Select office Name  2. Select Employee Name  3. Select Balance Date  4. Select Account Name  5. Enter Initial Balance  6. Select Balance Type  7. Click add | Data saved successfully | Show in the list | P |

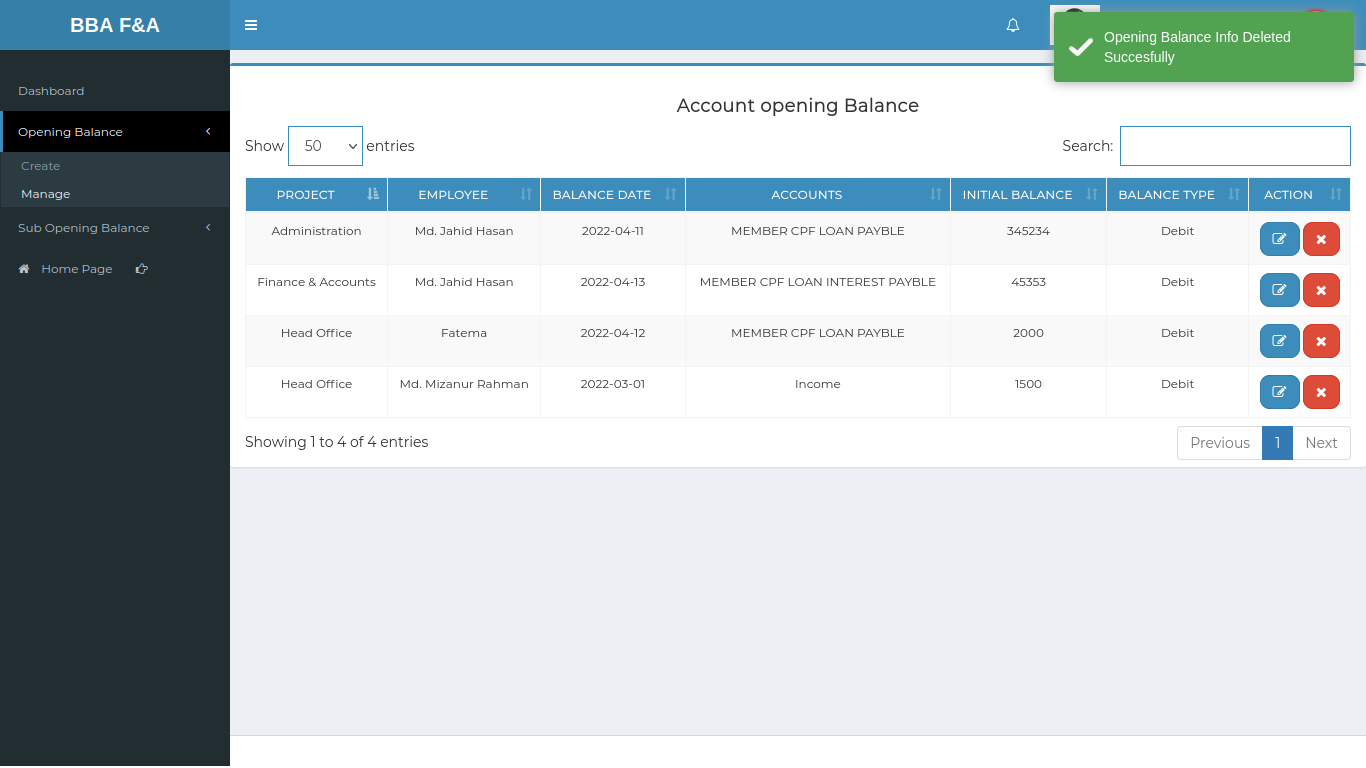
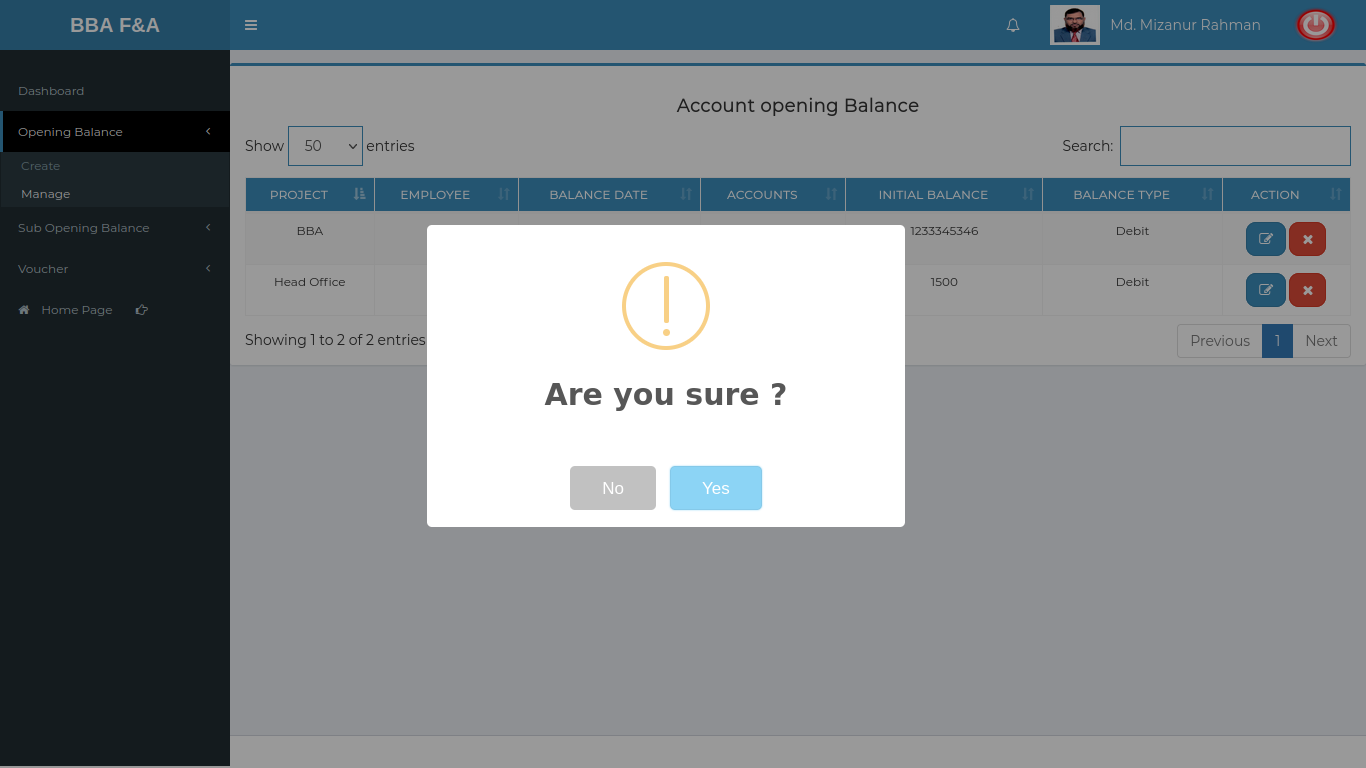
Negative Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Accounts Opening Balance Entry | 1. Office name, Employee, balance date, balance type, accounts should be selected  2. initial balance should be entered  3. Add button should store the information | No missing information while accessing any other information from other database | No error occurred while adding entry | P |

2. Manage







Positive Test Case:

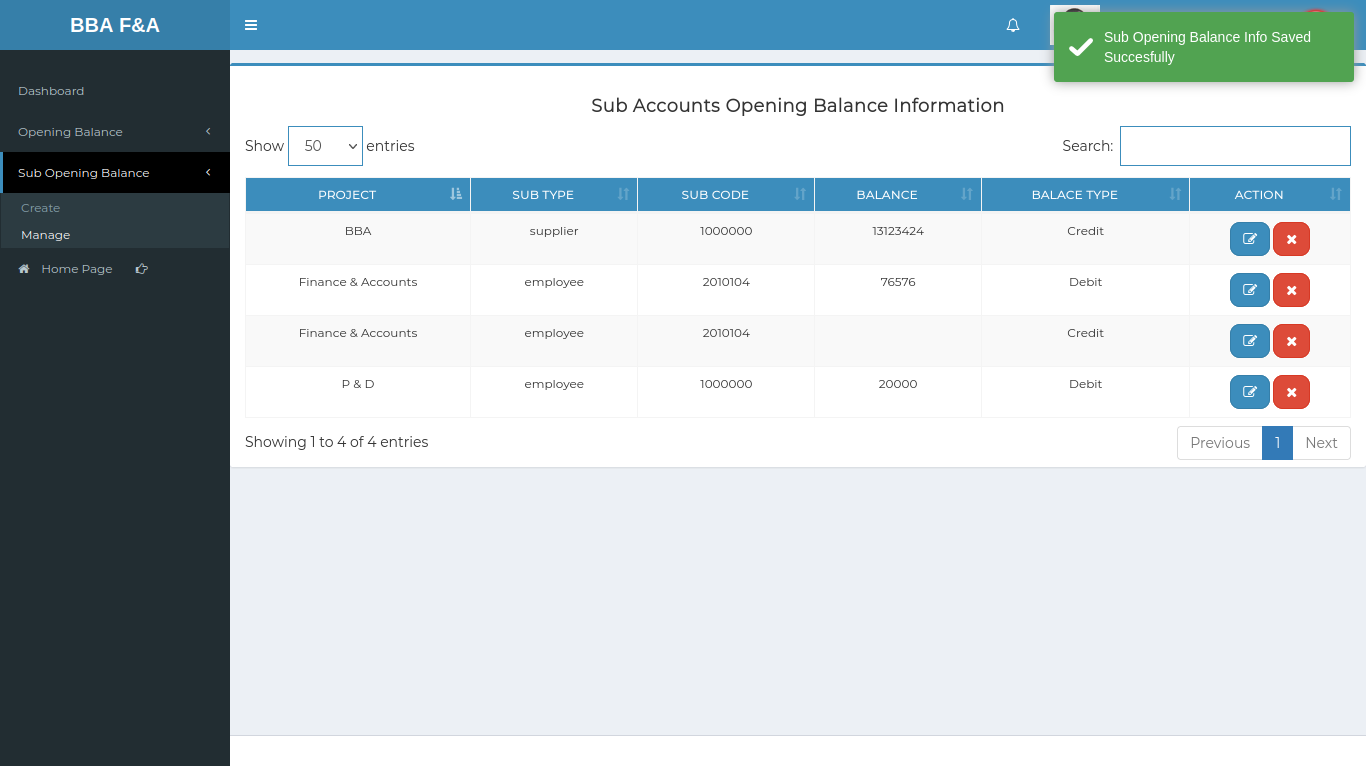
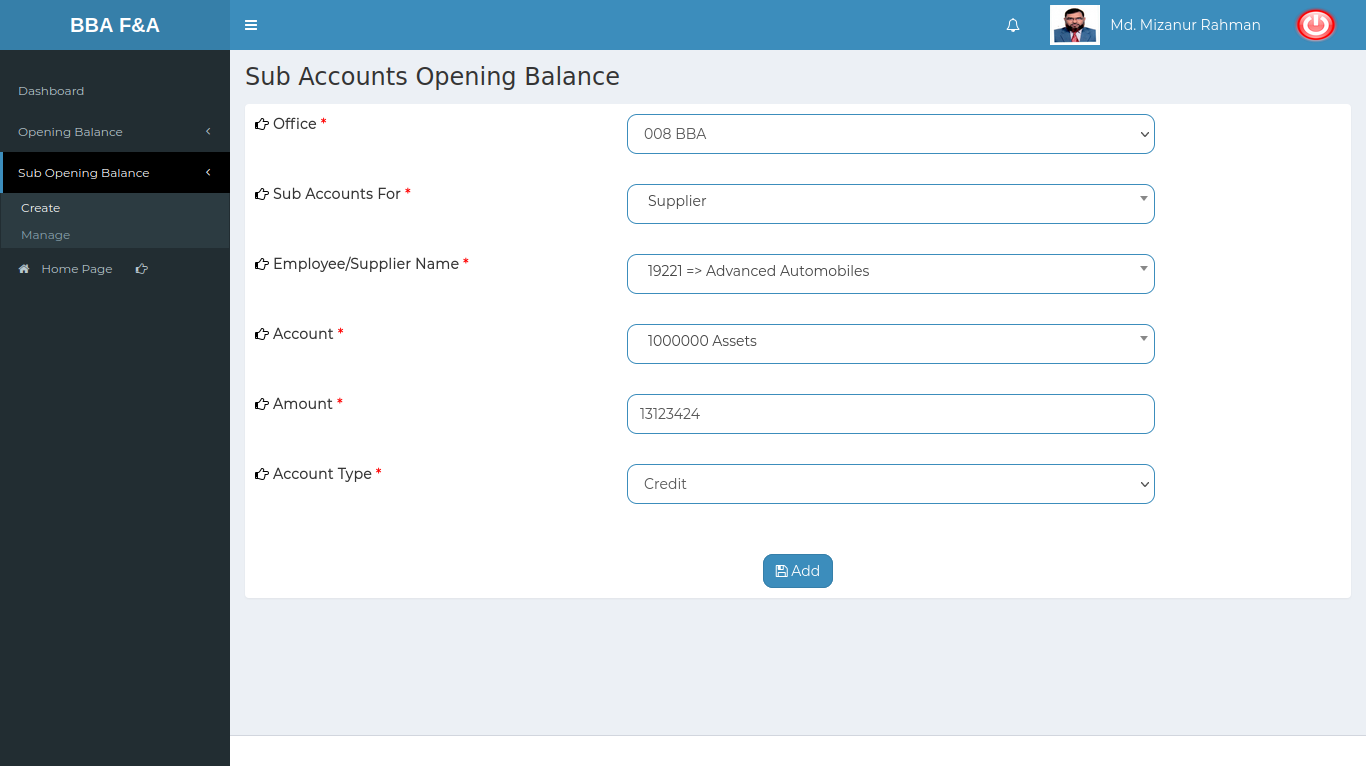
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Account Opening Balance List | 1. Display all the account opening balance Information  2. Type an employee id to find an employee from list | All information show in the list | No information is missing | P |
| 2 | Update | 1. Select those data which need to be updated  2. click update | Update the existing information | Updated and show in the list | P |
| 3 | Delete | Press delete for deleting the account opening balance information | Deleting a specific information | Delete the information from the list |  |

Negative Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Account Opening Balance List | Action Button and search bar Should work fine | Listed button works fine & no information is missing | No error is occurred | P |
| 2 | Update | Update button should update the existing information | Existing information mount & update successfully | No error while updating | P |
| 3 | Delete | Delete the information from the list and update the database | Successfully deleted by delete button | Delete the information & Database updated | P |

## Test Scenario 3: Sub-Opening Balance

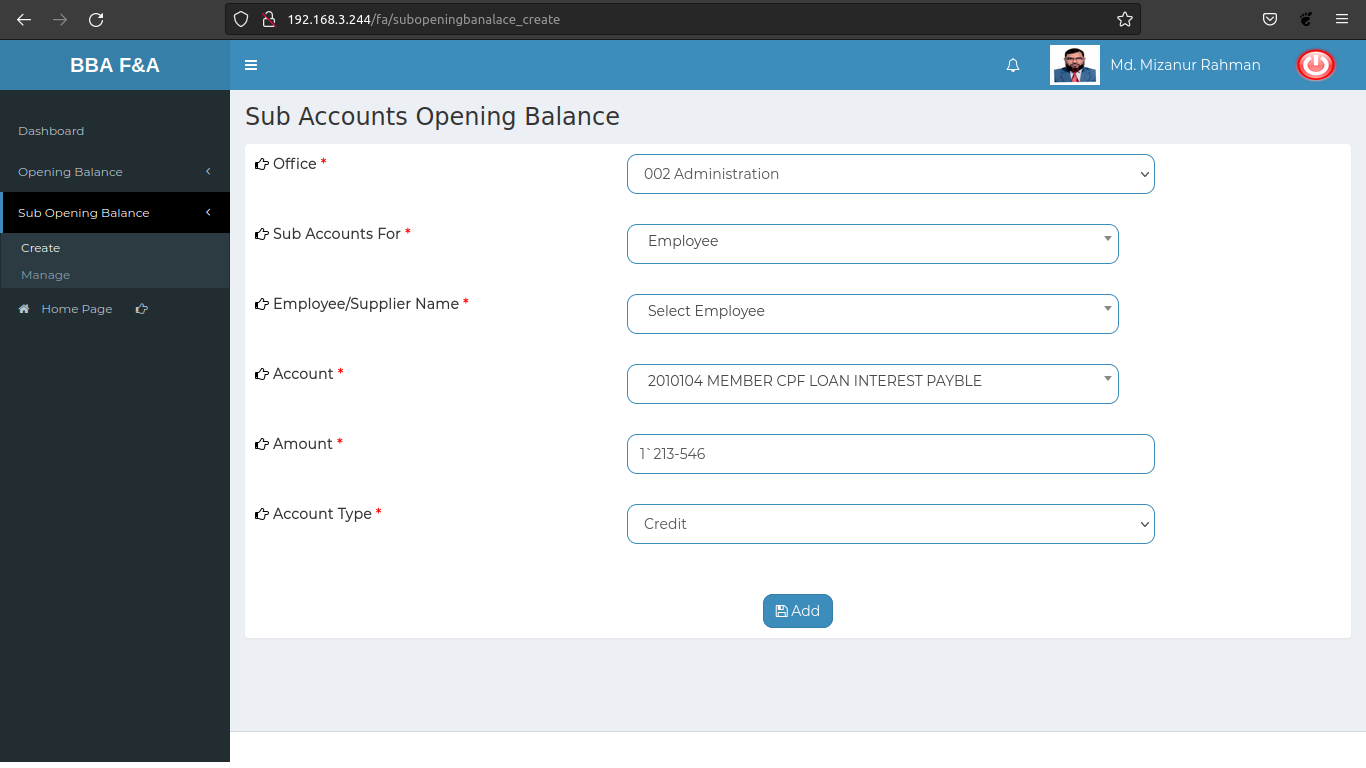
1. Create



Positive Test Case:

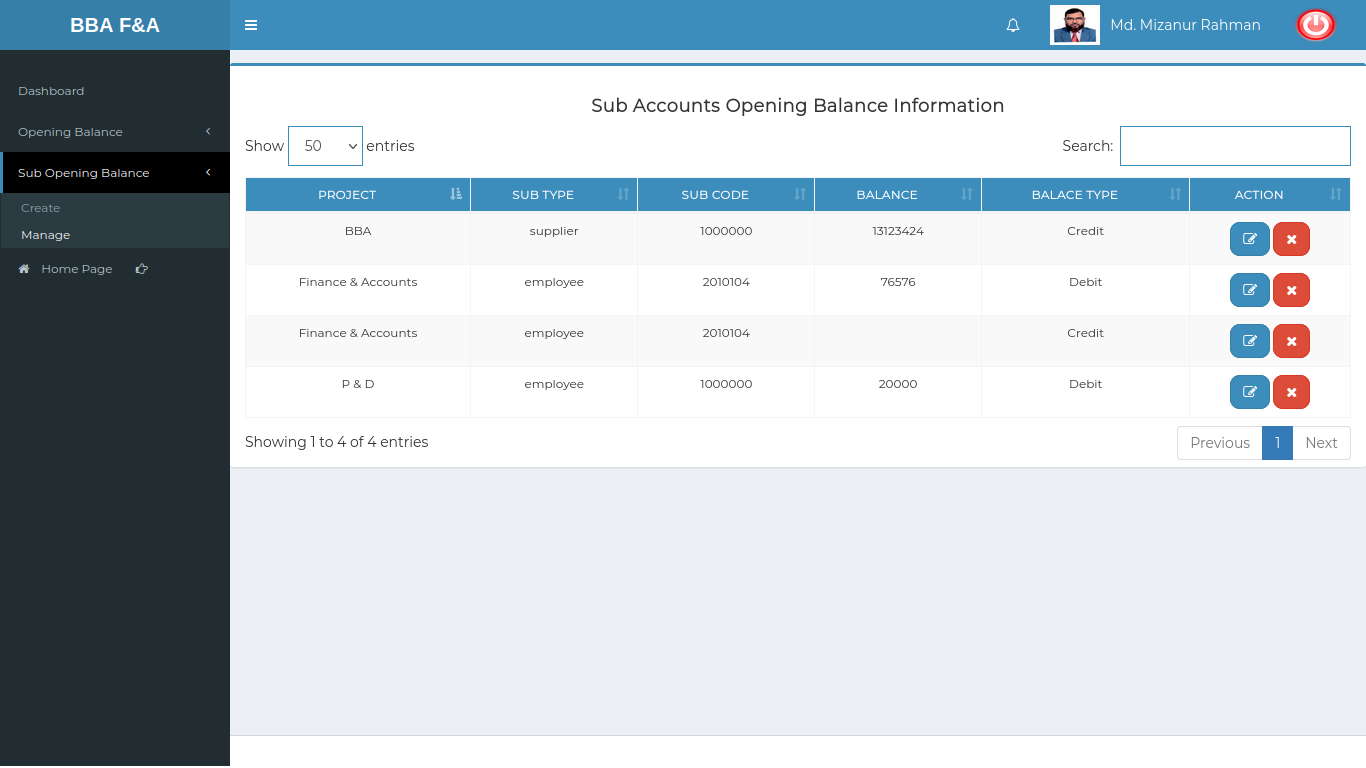
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Sub-account opening Balance | 1. Select office name  2. Select sub accounts for whether it is employee or supplier  3. Select employee or supplier name  4. Select account name  5. Enter amount  6. select account type  7. Click add | Data saved successfully | Show in the list | P |

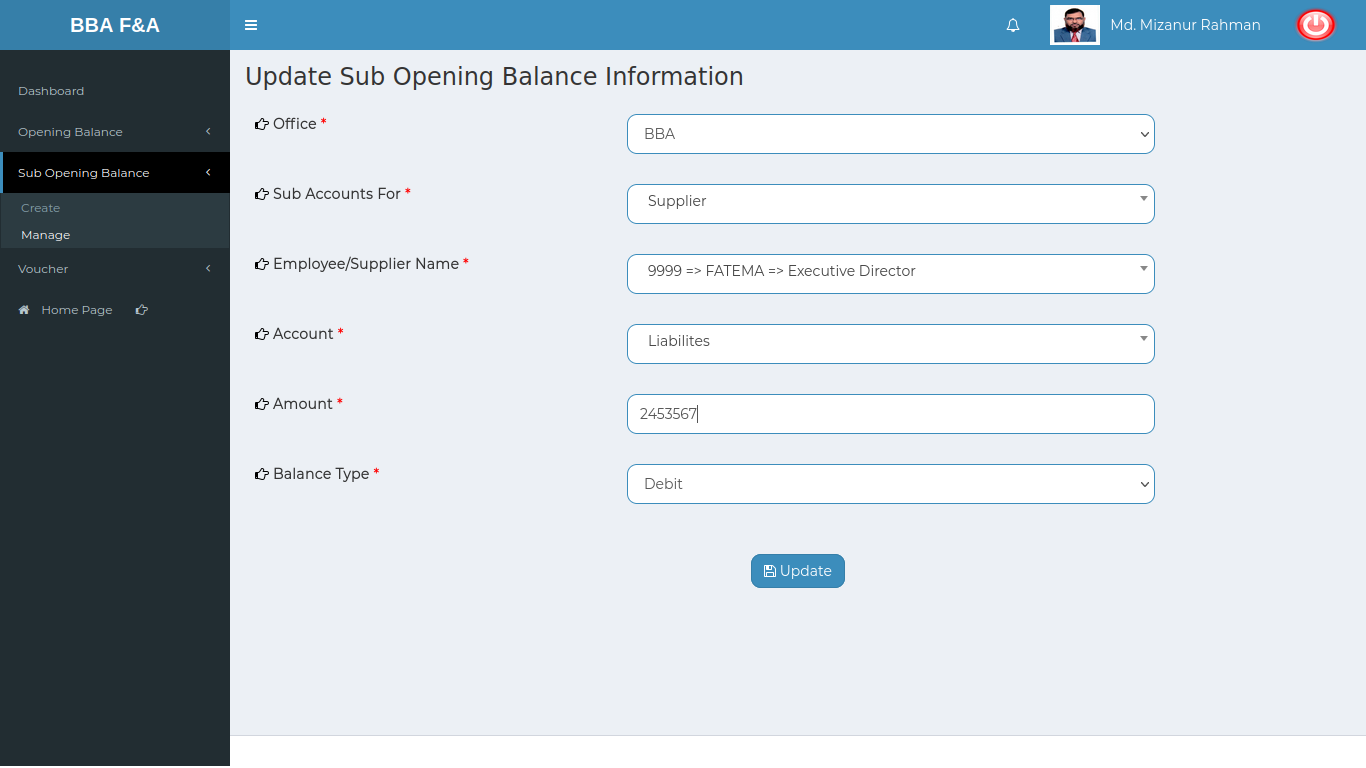
Negative Test Case:



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Sub-account opening Balance | 1. Office name, sub-accounts for, account name, account type should be selected  2. amount should be entered  3. Add button should store the information | No missing information while accessing any other information from other database | No error occurred while adding entry | P |

2. Manage





Positive Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Sub-account opening Balance List | 1. Display all the sub-account opening balance Information  2. Type an employee id to find an employee from list | All information show in the list | No information is missing | P |
| 2 | Update Sub-account opening Balance List | 1. Select those data which need to be updated  2. click update | Update the existing information | Updated and show in the list |  |
| 3 | Delete | Press delete for deleting the account opening balance information | Deleting a specific information | Delete the information from the list |  |

Negative Test Case:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Description | Expected Result | Actual Result | Pass/ Fail |
| 1 | Sub-account opening Balance List | Action Button and search bar Should work fine | Listed button works fine & no information is missing | No error is occurred | P |
| 2 | Update Sub-account opening Balance List | Update button should update the existing information | Existing information mount & update successfully | No error while updating |  |
| 3 | Delete | Delete the information from the list and update the database | Successfully deleted by delete button | Delete the information & Database updated |  |