**Software Documentation for the Account and Budget Management System**

**Product Name: Account and Management System**

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 4](#__RefHeading___Toc2357_1497391501)

[Design Specification 5](#__RefHeading___Toc2359_1497391501)

[Abstract 6](#__RefHeading___Toc2361_1497391501)

[Chapter-1: Introduction 7](#__RefHeading___Toc2365_1497391501)

[1.1 Review of the CPF System 7](#__RefHeading___Toc2367_1497391501)

[1.2 Motivation 7](#__RefHeading___Toc2369_1497391501)

[1.3 Aims and Objective 7](#__RefHeading___Toc2371_1497391501)

[1.4 Solution Methodology 7](#__RefHeading___Toc2373_1497391501)

[Chapter - 2. Background Study 8](#__RefHeading___Toc2375_1497391501)

[2.1 Central Provident Fund 8](#__RefHeading___Toc2377_1497391501)

[2.2 Why should we use this software system ? 8](#__RefHeading___Toc2379_1497391501)

[2.3 User Experience 8](#__RefHeading___Toc2381_1497391501)

[2.4.4 Benefits 8](#__RefHeading___Toc2389_1497391501)

[2.4.5 Role of the CPF 8](#__RefHeading___Toc25943_827625388)

[2.4.6 Impact of the CPF 8](#__RefHeading___Toc25945_827625388)

[2.4.7 Report & Analysis 9](#__RefHeading___Toc2391_1497391501)

[2.5 MIS main modules 9](#__RefHeading___Toc2393_1497391501)

[Chapter – 3: Software Requirement Analysis 10](#__RefHeading___Toc2395_1497391501)

[3.1 Introduction 10](#__RefHeading___Toc2397_1497391501)

[3.2 Problem Analysis 10](#__RefHeading___Toc25949_827625388)

[3.3 Requirement 10](#__RefHeading___Toc2399_1497391501)

[3.4 Software Requirement Specifications 10](#__RefHeading___Toc2401_1497391501)

[3.4.1 SRS Scopes 10](#__RefHeading___Toc2403_1497391501)

[3.4.2 Functional Requirements 11](#__RefHeading___Toc2405_1497391501)

[3.4.3 Non-functional Requirement 11](#__RefHeading___Toc2407_1497391501)

[3.5 Interface Requirement 12](#__RefHeading___Toc2409_1497391501)

[3.5.1 User Interface 12](#__RefHeading___Toc2411_1497391501)

[3.5.2 Hardware Interface 12](#__RefHeading___Toc2413_1497391501)

[3.5.3 Software interface 13](#__RefHeading___Toc2415_1497391501)

[3.5.4 Communication Interface 13](#__RefHeading___Toc2417_1497391501)

[3.6 SRS table 13](#__RefHeading___Toc2419_1497391501)

[3.6 Report Analysis 13](#__RefHeading___Toc2421_1497391501)

[Chapter – 4: System Design 15](#__RefHeading___Toc2423_1497391501)

[4.1 Algorithm Design 15](#__RefHeading___Toc2634_1971331298)

[4.1 Process Flow Diagram 15](#__RefHeading___Toc2431_1497391501)

[4.1 Data flow Diagram 15](#__RefHeading___Toc2634_19713312982)

[Chapter – 5: Testing Documentation 16](#__RefHeading___Toc2481_1497391501)

[5.1 Introduction 16](#__RefHeading___Toc2483_1497391501)

[5.2 Scopes 16](#__RefHeading___Toc2485_1497391501)

[5.3 Quality Objective 16](#__RefHeading___Toc2487_1497391501)

[5.4 Testing Methodology 17](#__RefHeading___Toc2489_1497391501)

[5.4.1 Model 17](#__RefHeading___Toc2491_1497391501)

[5.4.2 Strategies 17](#__RefHeading___Toc2493_1497391501)

[5.4.3 Levels 17](#__RefHeading___Toc2495_1497391501)

[5.4.4 Test Deliverables 17](#__RefHeading___Toc2511_1497391501)

[5.4.5 Tools 18](#__RefHeading___Toc2513_1497391501)

# Revision History

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

# Design Specification

**Overview:**

This­ Account and Budget management System is one o of the major part of Management Control System. Following sever side and client side communication make the system user-friendly. For the user’s view, back-end and front-end section controls the user interface and their functionality.

**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

A budget is an estimation of revenue and expenses over a specified future period of time and is usually compiled and re-evaluated on a periodic basis. Budgets can be made for a person, a group of people, a business, a government, or just about anything else that makes and spends money. In a system , budget can manage all the information related to the account corresponding employee and their relevant information. In this software system our aim is to build a software system that performs all the account information, balance information, budget and voucher information. All the operation can be done in a single module named as Account & Budget information.

# 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.

Ref: <https://opentextbc.ca/principlesofaccountingv2openstax/chapter/describe-how-and-why-managers-use-budgets/>

## 1.1 Review of the Account & Budget management System

The control of expenditure is usually the most important aspect for any viable undertaking as institutions are always under pressure to achieve their targets within limited resources. Budgetary control allows for monitoring of proper use of available funds at a reasonable frequency. It enables periodic statements of accounts to be prepared comparing budgets and actual results. Budgets are a core feature of control systems. The Organization will prepare budget that translates work plans into projections of expenditure. These will be prepared on an annual basis and updated regularly during the year.

## 1.2 Motivation

Implementation of a company’s strategic plan often begins by determining management’s basic expectations about future economic, competitive, and technological conditions, and their effects on anticipated goals, both long-term and short-term. Many firms at this stage conduct a situational analysis that involves examining their strengths and weaknesses and the external opportunities available and the threats that they might face from competitor.

There are many advantages to budgeting, including:

* Communication
  + Budgeting is a formal method to communicate a company’s plans to its internal stakeholders, such as executives, department managers, and others who have an interest in—or responsibility for—monitoring the company’s performance.
  + Budgeting requires managers to plan for both revenues and expenses.
* Planning
  + Preparing a budget requires managers to consider and evaluate
    - The assumptions used to prepare the budget.
    - Long-term financial goals.
    - Short-term financial goals.
    - The company’s position in the market.
    - How each department supports the strategic plan.
  + Preparing a budget requires departments to work together to
    - Determine realizable sales goals.
    - Compute the manufacturing or other requirements necessary to meet the sales goals.
    - Solve bottlenecks that are predicted by the budget.
    - Allocate resources so they can be used effectively to meet the sales and manufacturing goals.
    - Compare forecasted or flexible budgets with actual results.
* Evaluation
  + When compared to actual results, budgets are early alerts and they forecast:
    - Cash flows for various levels of production.
    - When loans may be required or when loans may be reduced.
  + Budgets show which areas, departments, units, and so forth, are profitable or meet their appropriate goals. Similarly, they also show which components are unprofitable or do not reach their anticipated goals.
  + Budgets set defined benchmarks that may be used for evaluating company and management performance, including raises and bonuses, as well as negative consequences, such as firing.

## 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
* Account code, bank, office, service availability
* opening bank account
* Budget & Voucher Management

## 1.4 Solution Methodology

* Design Settings → Account & GoB code, Office, bank account, Supplier/Consultant/Contractor, service
* Design Opening Balance & sub-opening balance accounting
* Design Specific budget and voucher information

# Chapter - 2. Background Study

Ref: <http://acmedigitek.com/budget_financial_Accounting>

<https://www.treasury.tas.gov.au/Documents/BMS_User_Manual_2002-11-07.pdf>

[https://mof.portal.gov.bd/sites/default/files/files/mof.portal.gov.bd/page/8b31f768\_35db\_46b0\_bf7d\_cb6c4754a011/BACS%20Manual27.4.17.pdf](https://mof.portal.gov.bd/sites/default/files/files/mof.portal.gov.bd/page/8b31f768_35db_46b0_bf7d_cb6c4754a011/BACS Manual27.4.17.pdf)

<https://courses.lumenlearning.com/managacct/chapter/introduction-to-budgeting-and-budgeting-processes/>

## 2.1 Budget

Financial management is a process which brings together planning, budgeting, accounting, financial reporting, internal control, auditing, procurement, disbursement and the physical performance with the aim of managing the resources properly and achieving the organization’s objectives. Thus a reliable budget & financial accounting system is a critical ingredient of the organization’s success. Timely and relevant financial information provides a basis for better decisions and efficient service delivery. The objective of the proposed solution includes :

* To ensure that budget is used only for their intended purposes in an efficient way;
* To ensure that funds are properly managed and flow smoothly, adequately, regularly and predictably in order to meet the objectives of the organization;
* To enable the preparation of accurate and timely financial reports;

Time and money are scarce resources to all individuals and organizations; the efficient and effective use of these resources requires planning. Planning alone, however, is insufficient. Control is also necessary to ensure that plans actually are carried out. A budget is a tool that managers use to plan and control the use of scarce resources. A budget is a plan showing the company’s objectives and how management intends to acquire and use resources to attain those objectives

## 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.

* It saves time as it speeds up every aspect of the employee database management and

payroll process with a range of automated features.

* 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 Role of the Account & Budget Information

The accounting system allows gathering, processing and organizing of accounting data in order to generate useful financial information. The accounting system reflects the Organizational needs and is designed to provide the financial information required by all stakeholders. The accounting system is designed to :

* Provide information compatible with the agreed accounting standards;
* Chart of Accounts.
* Provide adequate documentation and audit trails.
* Bank reconciliation.
* Financial Accounting Vouchers.
* Financial Statements & Report Generation.
* Maintain data and financial integrity.

### 2.4.6 Impact of the Account and Budget Information

The Budget Management System has b een designed to facilitate the flow of

data between agencies, Treasury, and other stakeholders for the purposes of:

* monitoring and reporting on the current year Budget;
* maintaining the forward estimates for future Budget periods; and
* preparing annual Budget docum ents and any other docum entation requiring Budget information.

### 2.4.7 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 account and budget management needs. Therefore I have design exclusive Employee and payroll Management System that are adapted to the organization’s Managerial Requirements.

## 

## 2.5 Budget main modules

The following features are the main things of account and budget management system.

1. Dashboard
2. Settings  
   i) Chart Of account Code

ii) Chart of GoB Code

iii) Bank Account

iv) Office

v) Supplier/Consultant/Contractor

vi) Service

1. Opening Balance
2. Sub-opening Balance
3. Budgets  
   i) Entry

ii) Manage

iii) Expenses

iv) Bills

v) Report

1. Voucher

i) Entry

ii) Manage

iii) Report

iv) Old Records

## 2.6 Budget Management Highlights

* Supports a base budget, current budget, and infinite working budgets per fiscal year
* Budgets can be generated from existing historical data in the system
* Budgets can be updated by values or percentages within individual periods, or on overall top-line values spread to corresponding periods
* Supports accessibility to historical prior year budgets
* Multiple budgets can be created to perform what-if analysis (best case, worst case, most likely, etc.)
* Reporting can be performed on actual vs. budget across any combination of one or more of the base, current, and working budgets at a detail or summary-level
* Increases or decreases overheads by entering a percentage for specific accounts or for groups of accounts
* Supports online display of actuals to budget
* Reports can compare multiple years’ actual and budgeted amounts

Ref: https://www.tgiltd.com/web-erp-system/erp-accounting-software/budget-management-software

# 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 Transport 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 budget system is part of the human resource management system. Activities carried out from the payroll system are processes of data from the BBA management of human resources.

### 3.4.2 Functional Requirements

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

* Login Functionality
* Report – download a copy of corresponding section

### 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

### 3.5.2 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. | Settings |  |
| 2. | Chart of account code |  |
| 3. | Chart of GoB code |  |
| 4. | Bank Account |  |
| 5. | Office |  |
| 6. | Supp/Cons/Contractor |  |
| 7. | Service |  |
| 8. | Opening Balance |  |
| 9. | Sub-opening Balance |  |
| 10. | Budget |  |
| 11. | Entry |  |
| 12. | Manage |  |
| 13. | Expenses |  |
| 14. | Bills |  |
| 15. | Report |  |
| 16. | Voucher |  |
| 17. | Manage |  |
| 18. | Report |  |
| 19. | Old records |  |

## 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

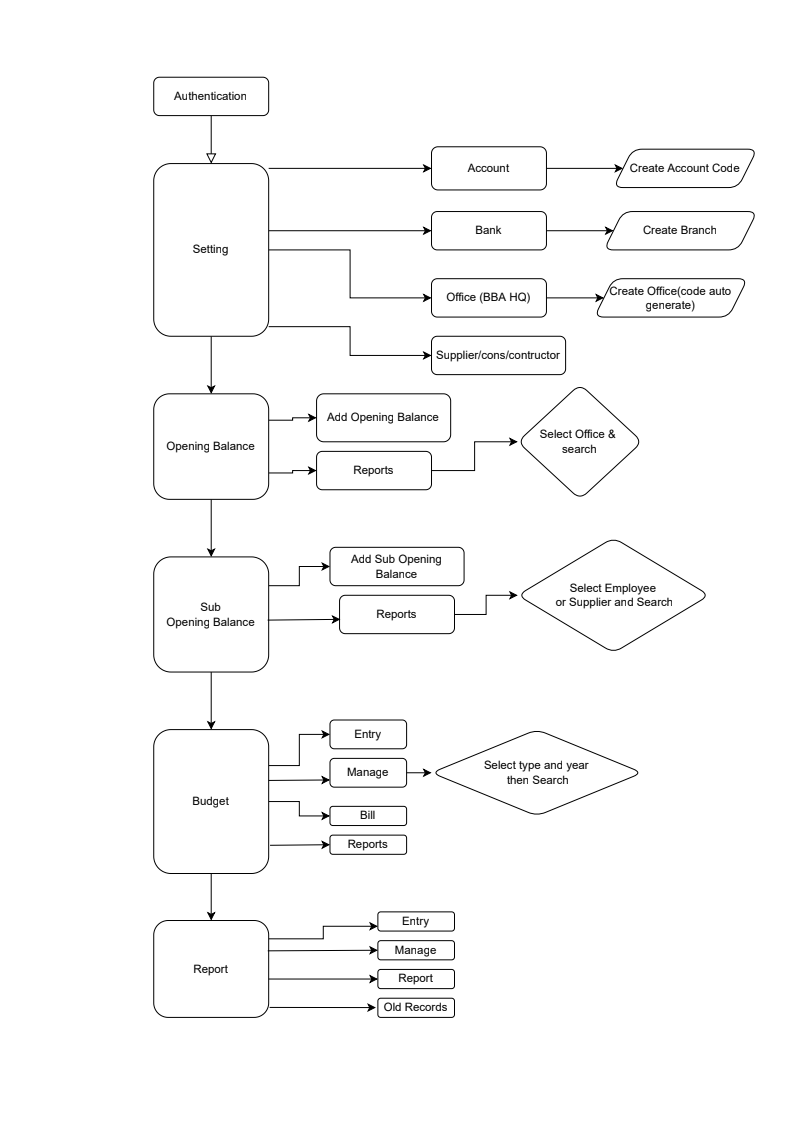
|  |  |  |  |
| --- | --- | --- | --- |
| SI.No. | Requirement Name | Functional/Non-  functional | Priority |
| 1. | **Dashboard** | Non-functional | 3 |
| 2. | **Settings** | Non-functional | 2 |
| 3. | Chart of account code | Functional | 1 |
| 4. | Chart of GoB code | Functional | 1 |
| 5. | Bank Account | Functional | 1 |
| 6. | Office | Functional | 1 |
| 7. | Supp/Cons/Contractor | Functional | 1 |
| 8. | Service | Functional | 1 |
| 9. | **Opening Balance** | Functional | 1 |
| 10. | **Sub-opening Balance** | Functional | 1 |
| 11. | **Budget** | Non-Functional | 2 |
| 12. | Entry | Functional | 1 |
| 13. | Manage | Functional | 1 |
| 14. | Expenses | Functional | 1 |
| 15. | Bills | Functional | 1 |
| 16. | Report | Functional | 1 |
| 17. | **Voucher** | Non-Functional | 1 |
| 18. | Manage | Functional | 1 |
| 19. | Report | Functional | 1 |
| 20. | Old records | Functional | 1 |
| 21. | Reliability | Non-functional | 1 |
| 22 | Availability | Non-functional | 1 |
| 23 | Security | Non-functional | 1 |
| 24 | Maintainability | Non-functional | 1 |
| 25 | Portability | Non-functional | 1 |
| 26 | Usability | Non-functional | 1 |
| 27 | 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 = ‘Settings’)  SELECT Y  if(Y = ‘ChartOfAccountCode’)  ADD accountCode  SHOW chartOfAccounts  if(Y = ‘ChartOfGoBCode’)  ADD GoBCode  SHOW GoBCodes  if(Y = ‘BankAccount’)  ADD bank  SHOW accountBanks  if(Y = ‘Office’)  ADD office  SHOW officeInfo  if(Y = ‘Supplier OR Consultant  OR Contractor’)  SELECT supplierType  ADD supplierType  SHOW suppliertypes  ADD Supplier OR Consultant  OR Contractor  SHOW sccList  if(Y = ‘Service’)  ADD service  SHOW serviceInfo | if(X = ‘opening-balance’)  ADD openingBalance  SHOW accountOpBalanceInfo  GENERATE report  if(X = ‘subopening-balance’)  ADD subopeningBalance  SHOW subOpBalanceInfo  GENERATE report  if(X = ‘budgets’)  SELECT Y  if(Y = ‘Entry’)  ADD budget  if(Y = ‘Manage’)  SHOW budgetInfo  if(Y = ‘Bills’)  SHOW budgetbillInfo  if(Y = ‘Report’)  GENERATE report  if(X = ‘voucher’)  SELECT Y  if(Y = ‘Entry’)  ADD entry  if(Y = ‘Manage’)  SHOW voucherInfo  if(Y = ‘Report’)  GENERATE report  if(Y = ‘Old-Records’)  SHOW oldvoucherInfo  STEP – 5: SHOW dashboard  STEP – 6: END |

## 4.2 Process Flow Diagram



## 4.3 Data flow Diagram

//Incomplete software

# 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 |
| Settings | Admin | Operating Account code, GoB code, Bank Account, Office name, Supplier or consultant, contractor name, Service information |
| Opening balance | Admin | Select an office to open a bank account |
| Sub-opening balance | Admin | Sub-account for supplier or an employee |
| Budget | Admin | Budget entry, managing budget such as expenses and bill info as well as their report |
| Voucher | Admin | Making a voucher , managing these voucher, reporting as well as show their old records information |

**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 Contributory 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*

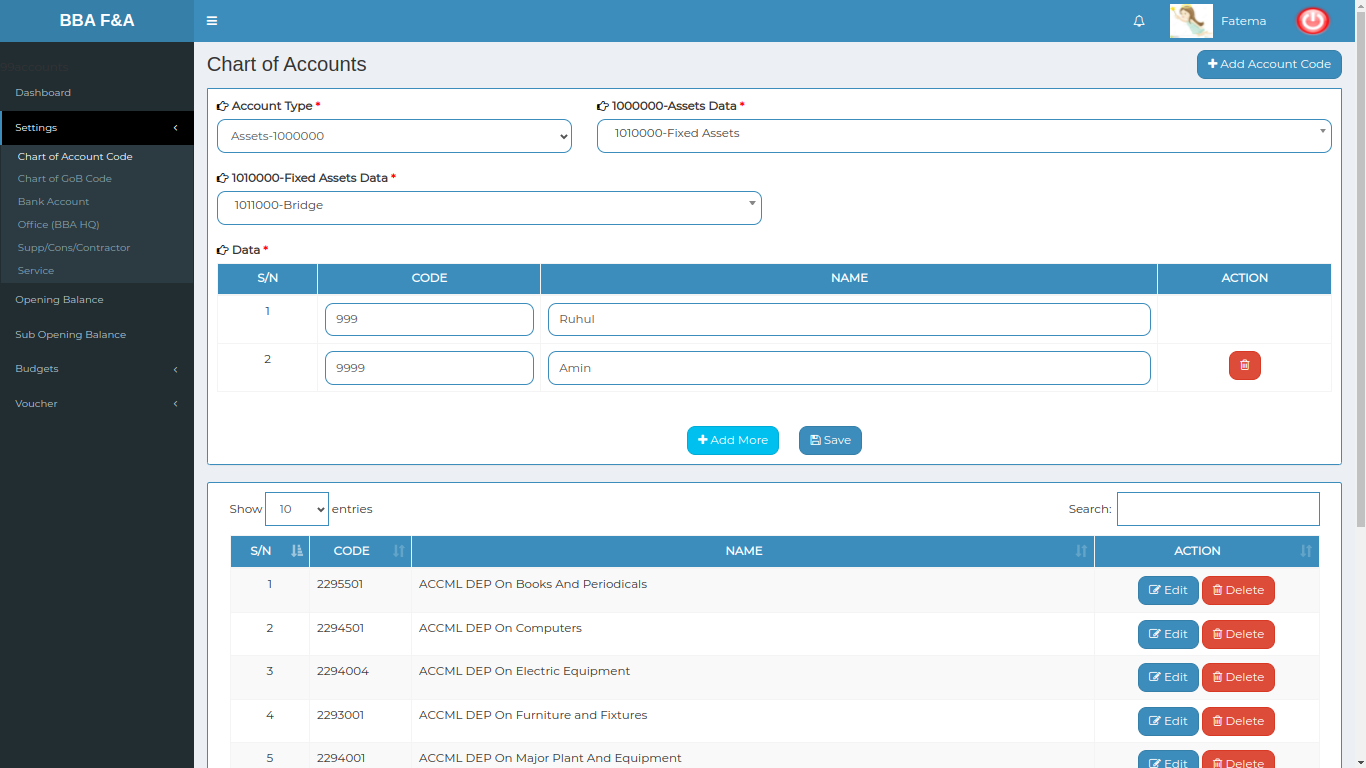
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Opening Balance | Click to see all opening balance | Visualized as expected | No error | P |
| 2 | Sub-Opening Balance | Click to see all sub-opening balance | Visualized as expected | No error | P |
| 3 | Vouchers | Click to see all Voucher information | Visualized as expected | No error | P |
| 4 | Budget | Click to see all Budget information | Visualized as expected | No error | P |
| 5 | FDR | Click to see all FDR information | Visualized as expected | No error | P |

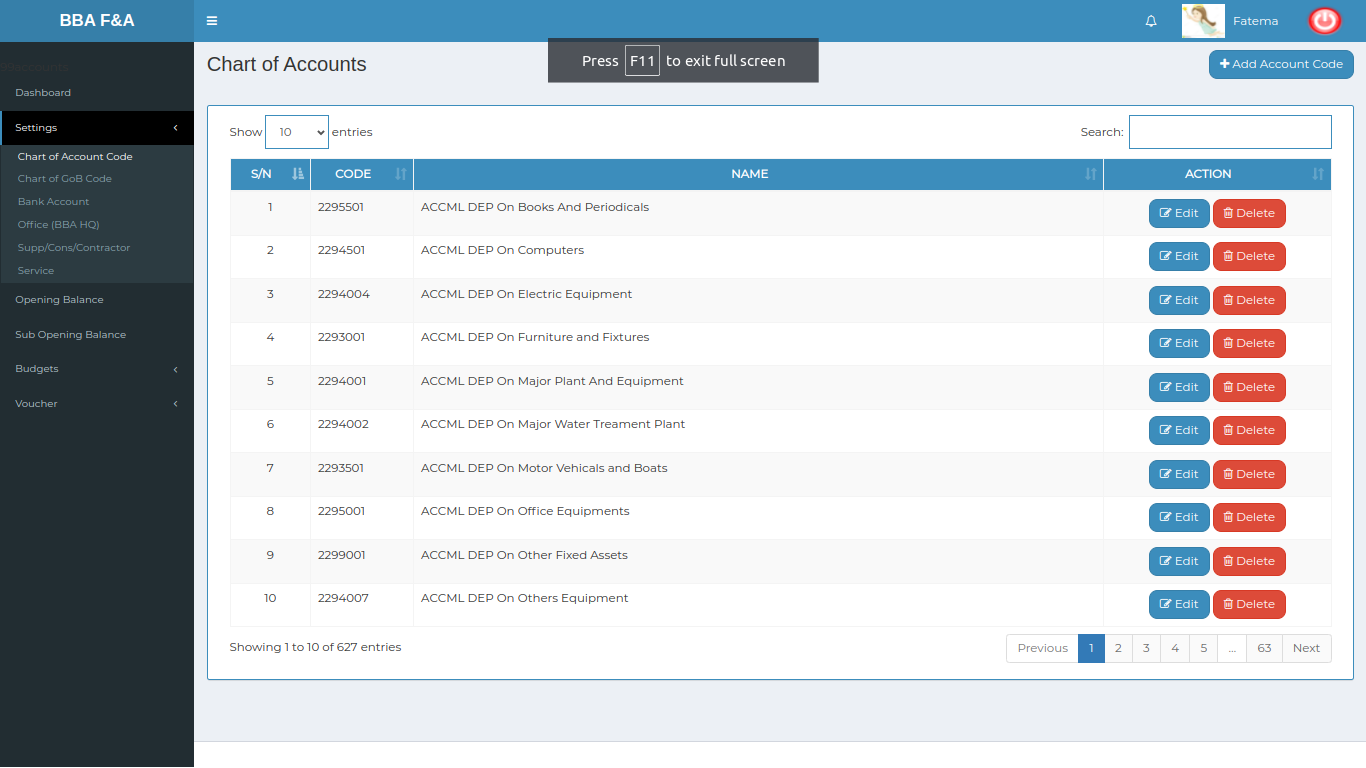
*Negative Test Case*

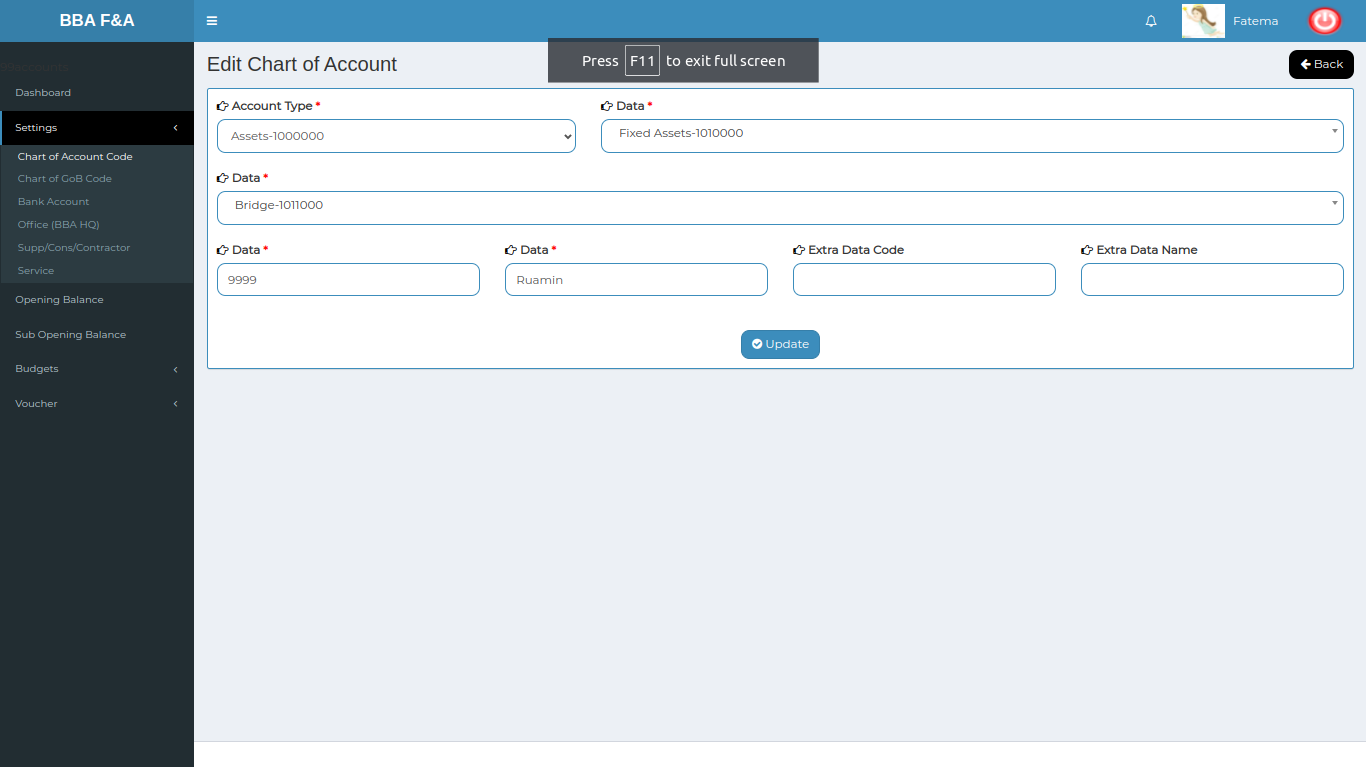
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Opening Balance | No information is missing while displaying respected information | Perfectly Operated | No error is occurred | P |
| 2 | Sub-Opening Balance | No information is missing while displaying respected information | Perfectly Operated | No error is occurred | P |
| 3 | Vouchers | No information is missing while displaying respected information | Perfectly Operated | No error is occurred | P |
| 4 | Budget | No information is missing while displaying respected information | Perfectly Operated | No error is occurred | P |
| 5 | FDR | No information is missing while displaying respected information | Perfectly Operated | No error is occurred | P |

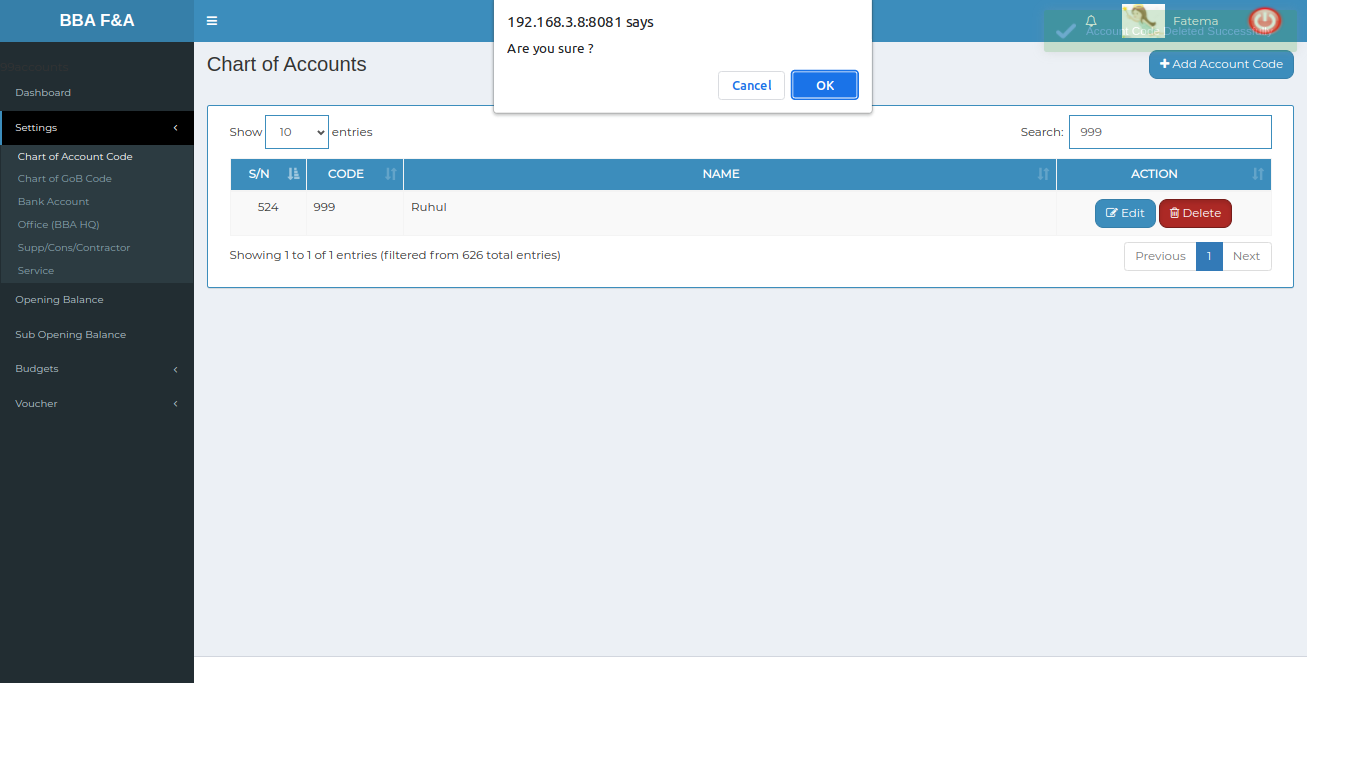
**Test Scenario 2: Settings**

1. Chart of Account Code









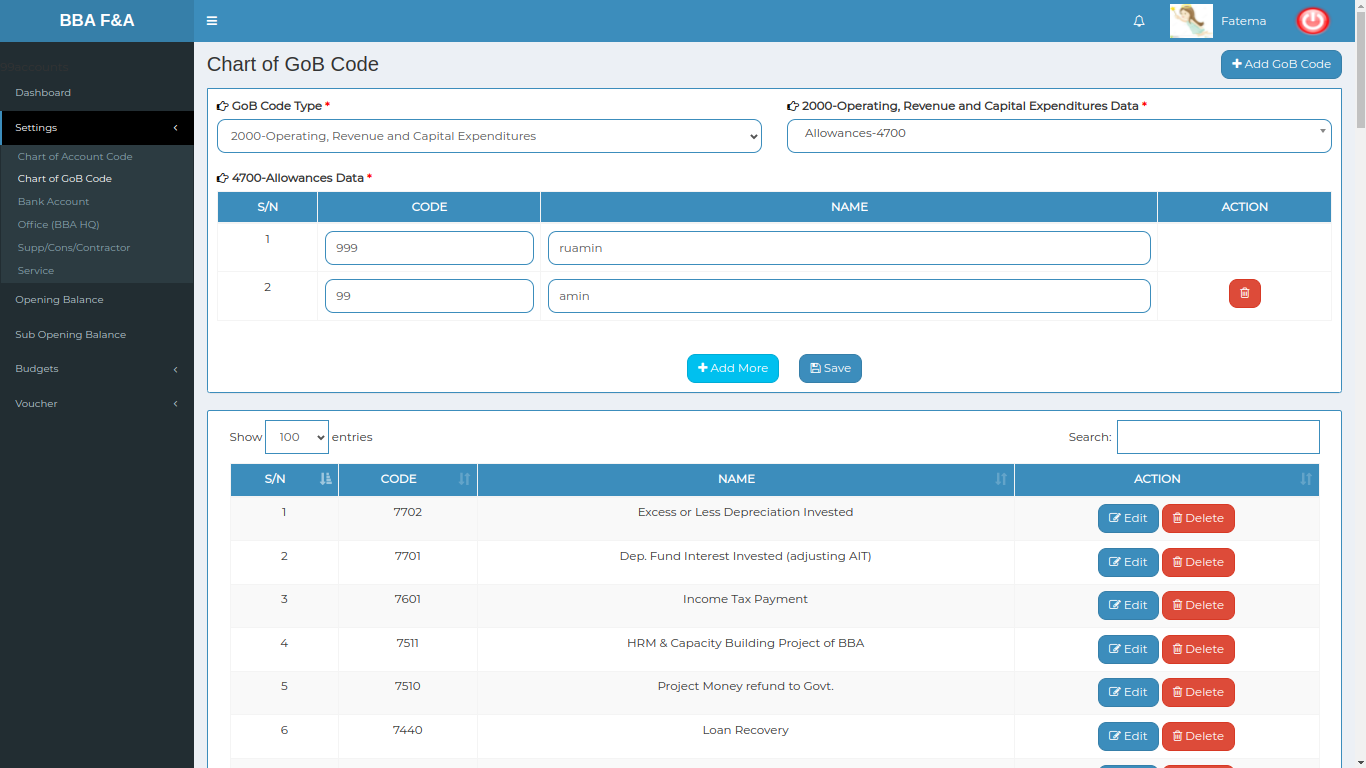
*Positive test case*

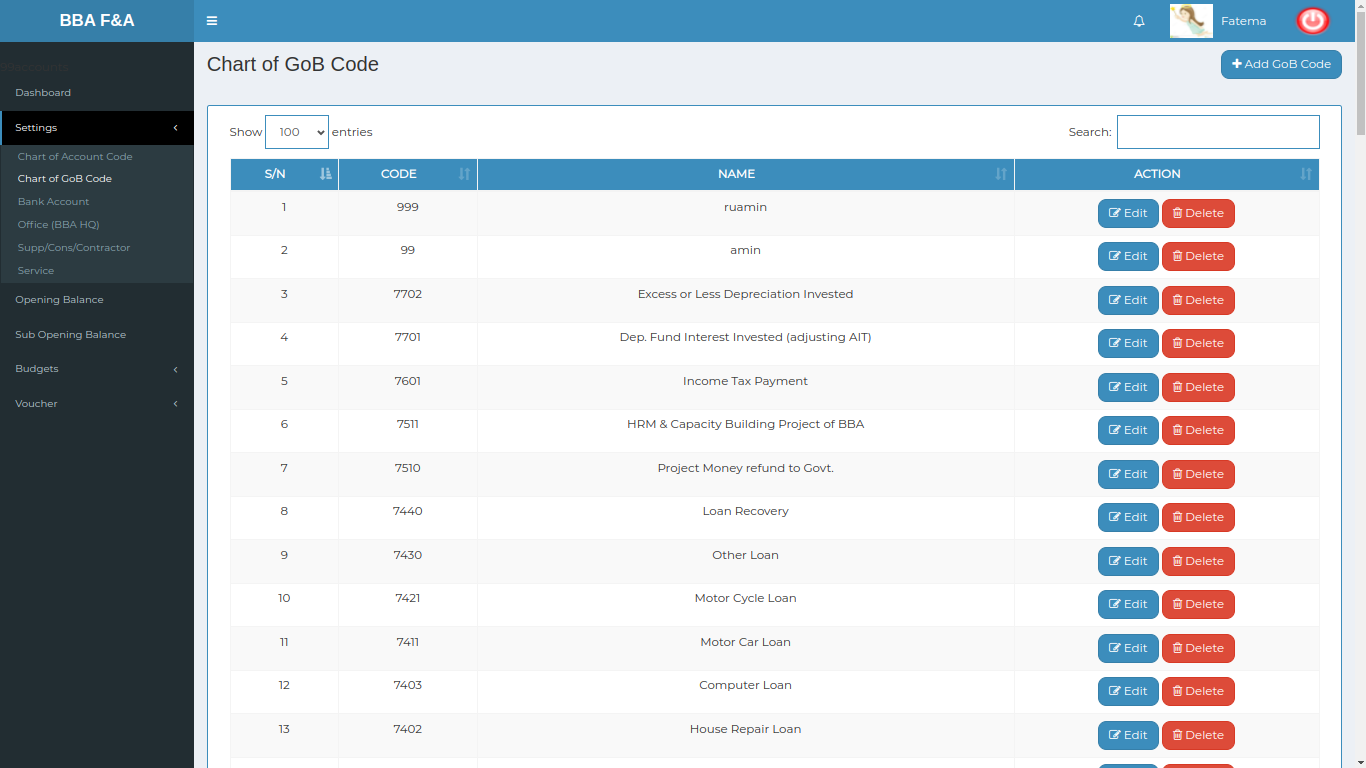
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Account Code | 1. Select Account Type  2. Select respective account type related data  3. Select data form the step 2 selective area  4. After passing steps 1,2,3 Enter data code & name  5. Click save or add more for add another Account type data serially | Data Should be saved | Information should be Displayed in list | pass |
| 2 | Chart of accounts | 1. Displaying all the account code  2. Type in the search bar to find a account code easily | All information are displayed | No information is missing | P |
| 3 | Update | 1. Select an specific item which need to be updated  2. click update | Updated database | No error while updating | P |
| 4 | Delete | Click delete and press yes to delete an item from the list | Updated database | No error while deleting | P |

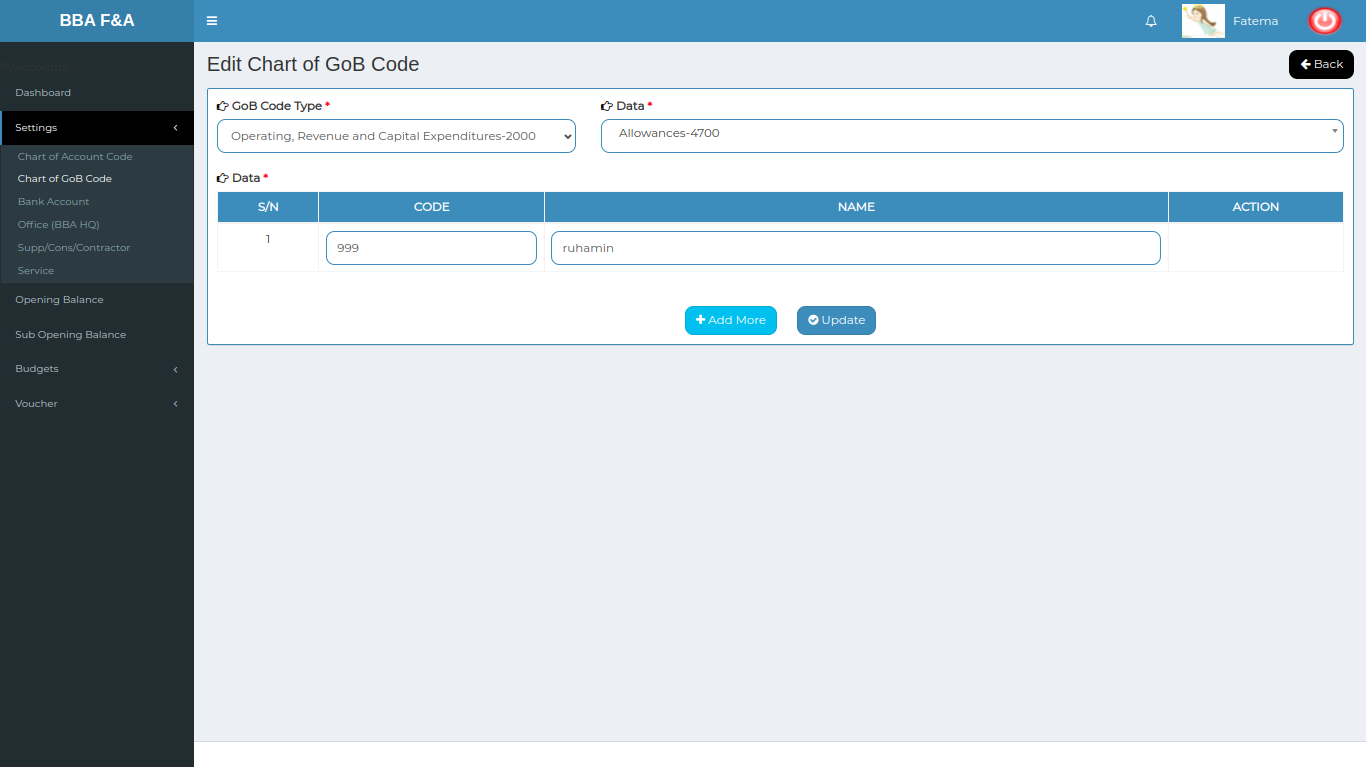
*Negative Test Case*

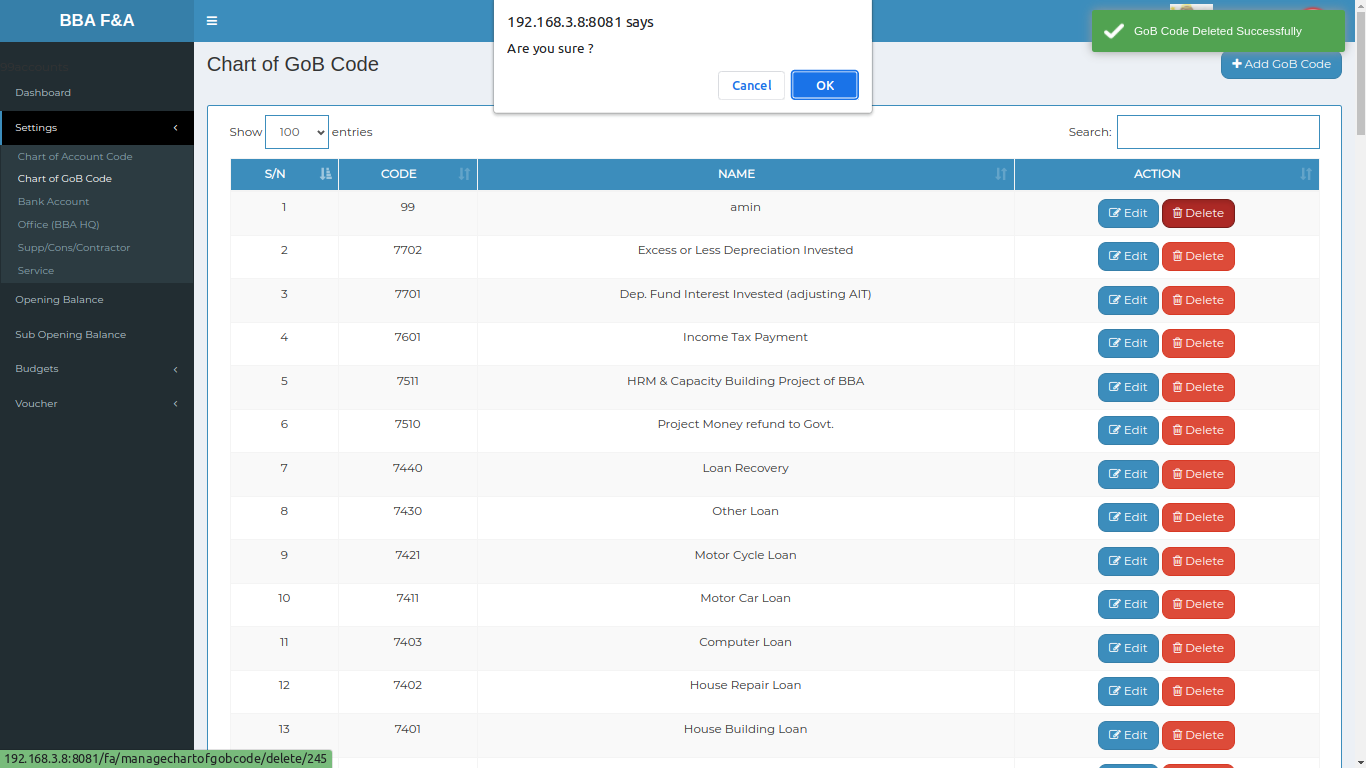
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Account Code | 1. Account data should be selected one by one  2. Entered information should have a valid information | Data checked | No error | P |
| 2 | Chart of accounts | 1. In the account charts, action button should work fine  2. Search bar should work fine | No missing information | No error occurred while displaying | P |
|  | Update | 1. Existing data should be editable  2. Update button should update the existing information | Updated existing information | Perfectly executed | P |
|  | Delete | Delete button should delete the account code information | Updated existing information | Perfectly executed | P |

2. Chart of GoB code









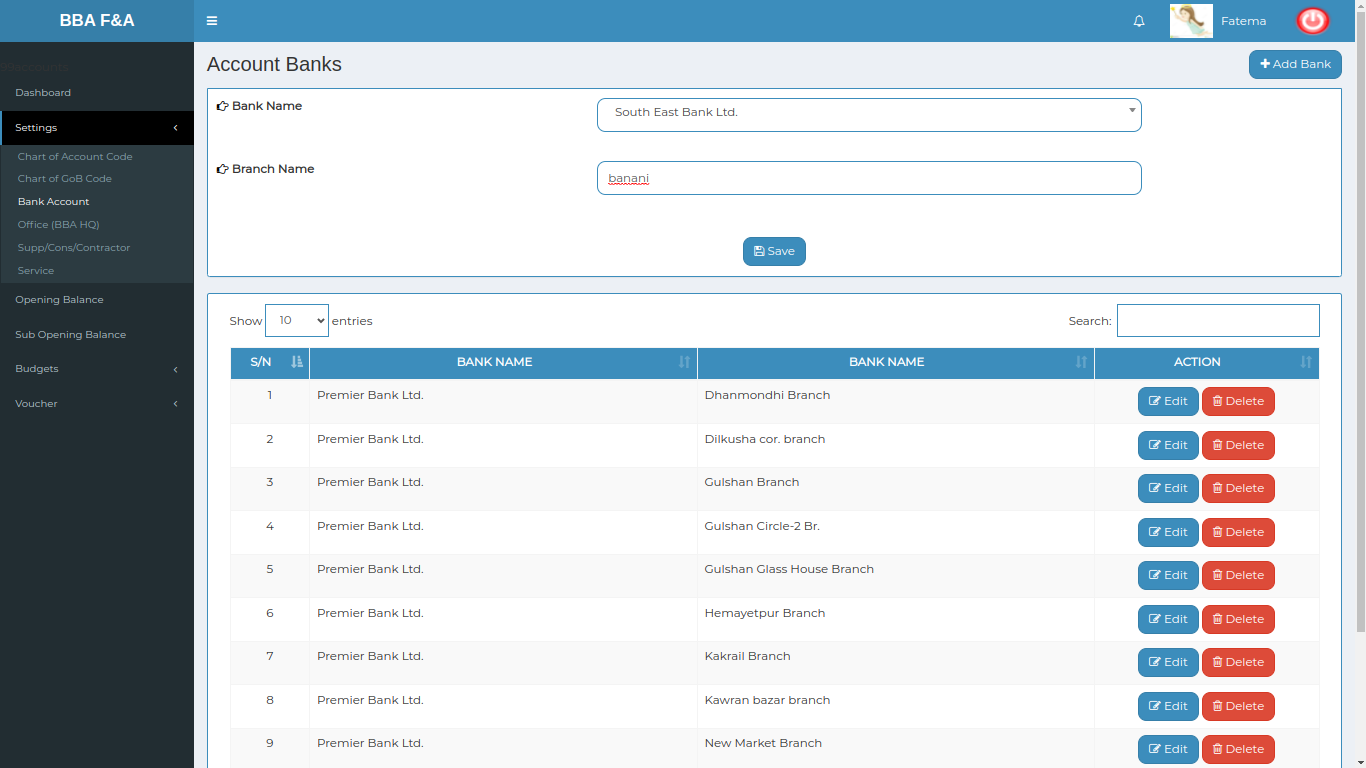
*Positive test case*

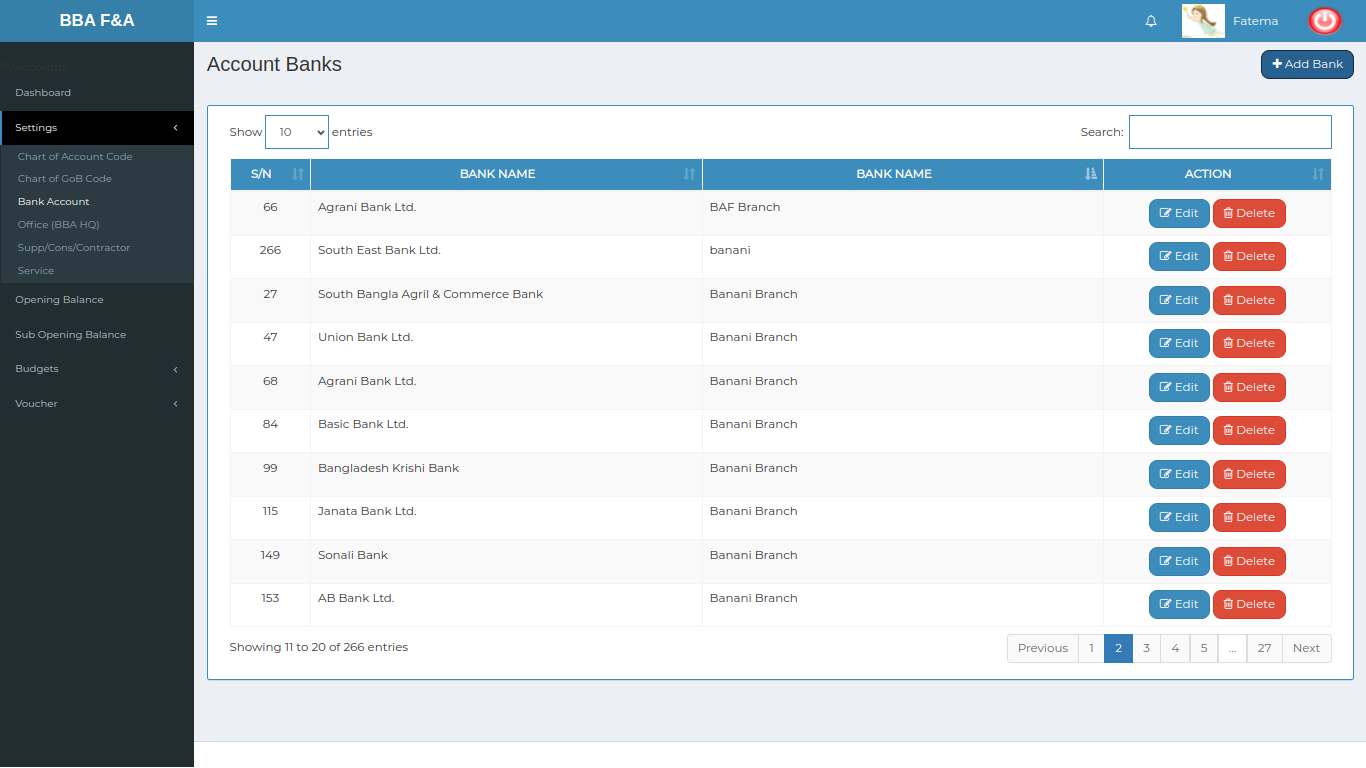
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add GoB Code | 1. Select GoB code type & operating and revenue incomes data  2. Enter data code & data name  3. Click save GoB information & add more for adding another GoB information | Data Should be saved | Information should be Displayed in list | pass |
| 2 | Chart of GoB code | 1. Show all the GoB code  2. Click search and type a GoB code to find that account information | All information are displayed | No information is missing | P |
| 3 | Update | 1. Select an specific item which need to be updated  2. click update | Updated database | No error while updating | P |
| 4 | Delete | Click delete and press yes to delete an item from the list | Updated database | No error while deleting | P |

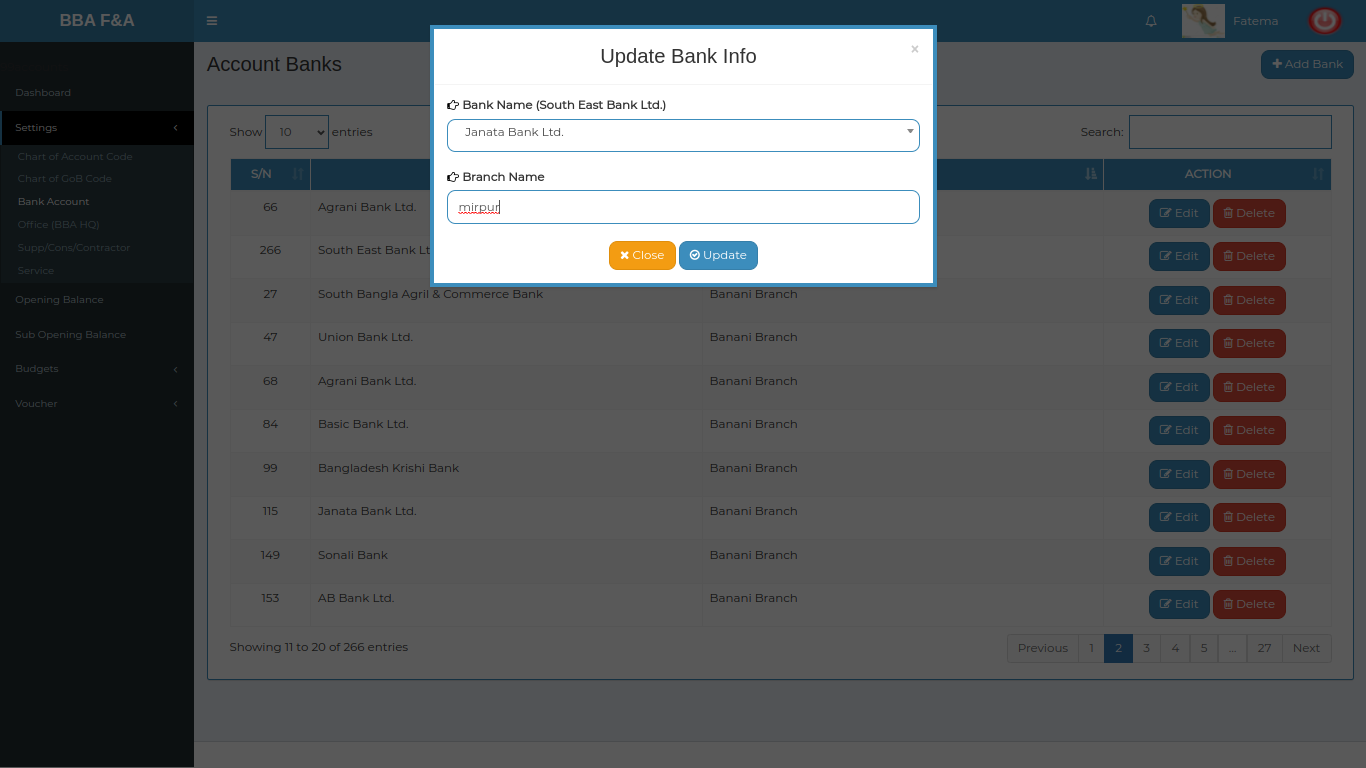
*Negative Test Case*

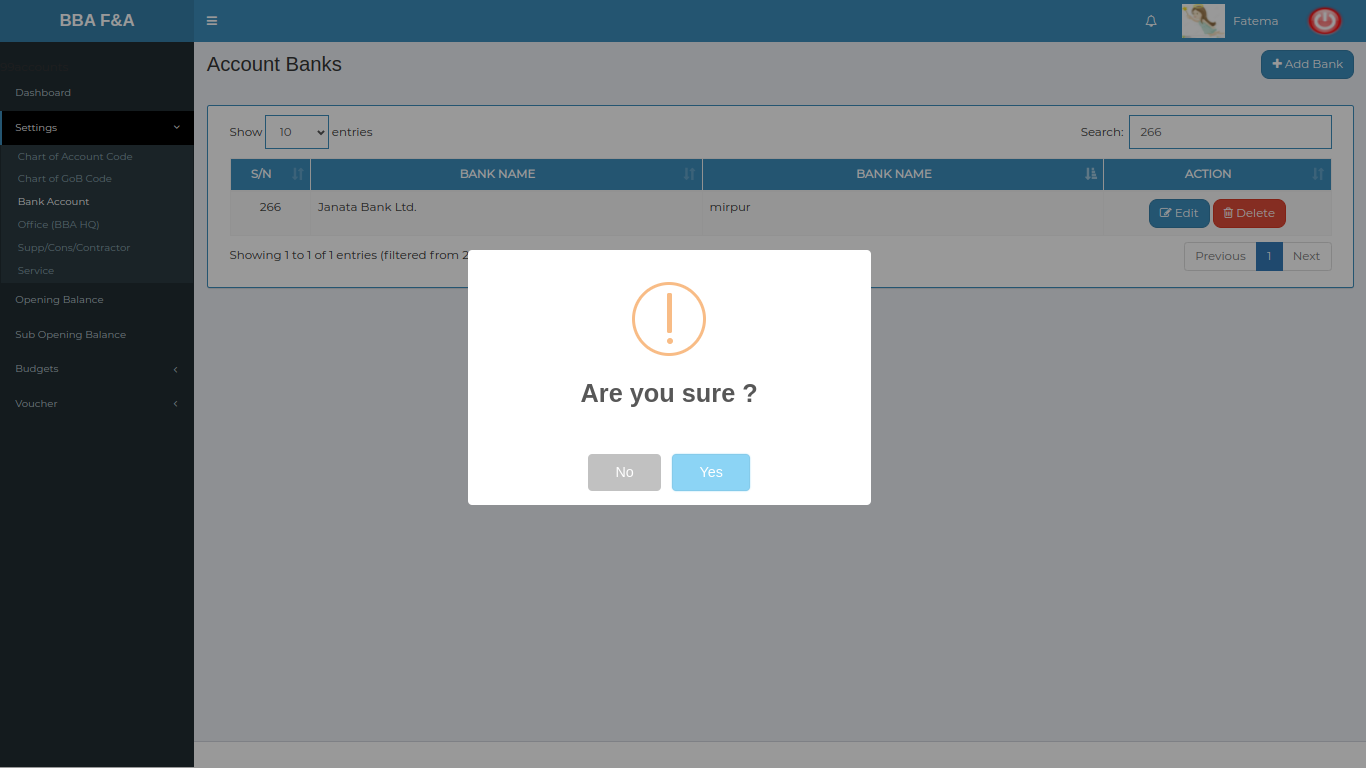
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add GoB Code | 1. GoB data should be selected  2. Entered data should be valid | Data checked | No error | P |
| 2 | Chart of GoB code | In the GoB code chart, Display all the GoB code chart list | No missing information | No error occurred while displaying | P |
| 3 | Update | 1. Existing data should be editable  2. Update button should update the existing information | Updated existing information | Perfectly executed | P |
| 4 | Delete | Delete button should delete the account code information | Updated existing information | Perfectly executed | P |

3. Bank Account









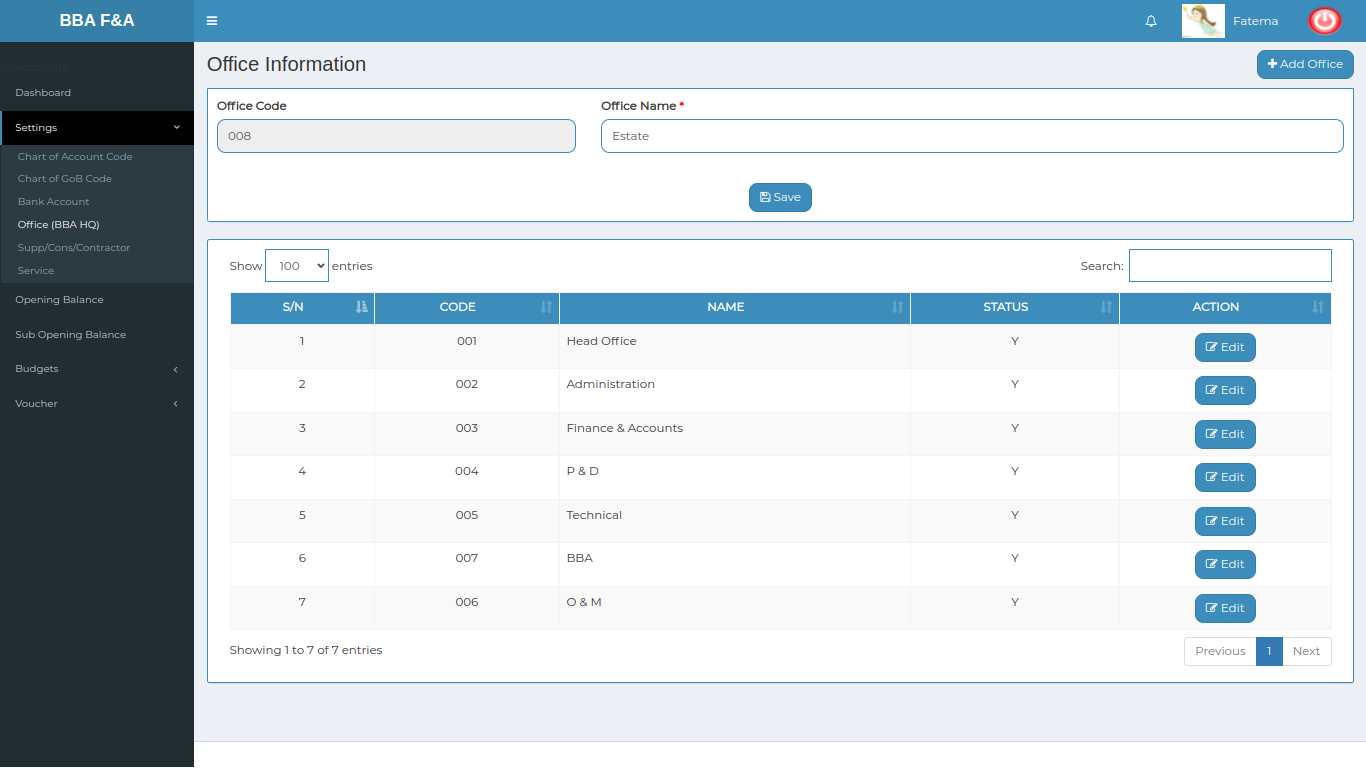
*Positive test case*

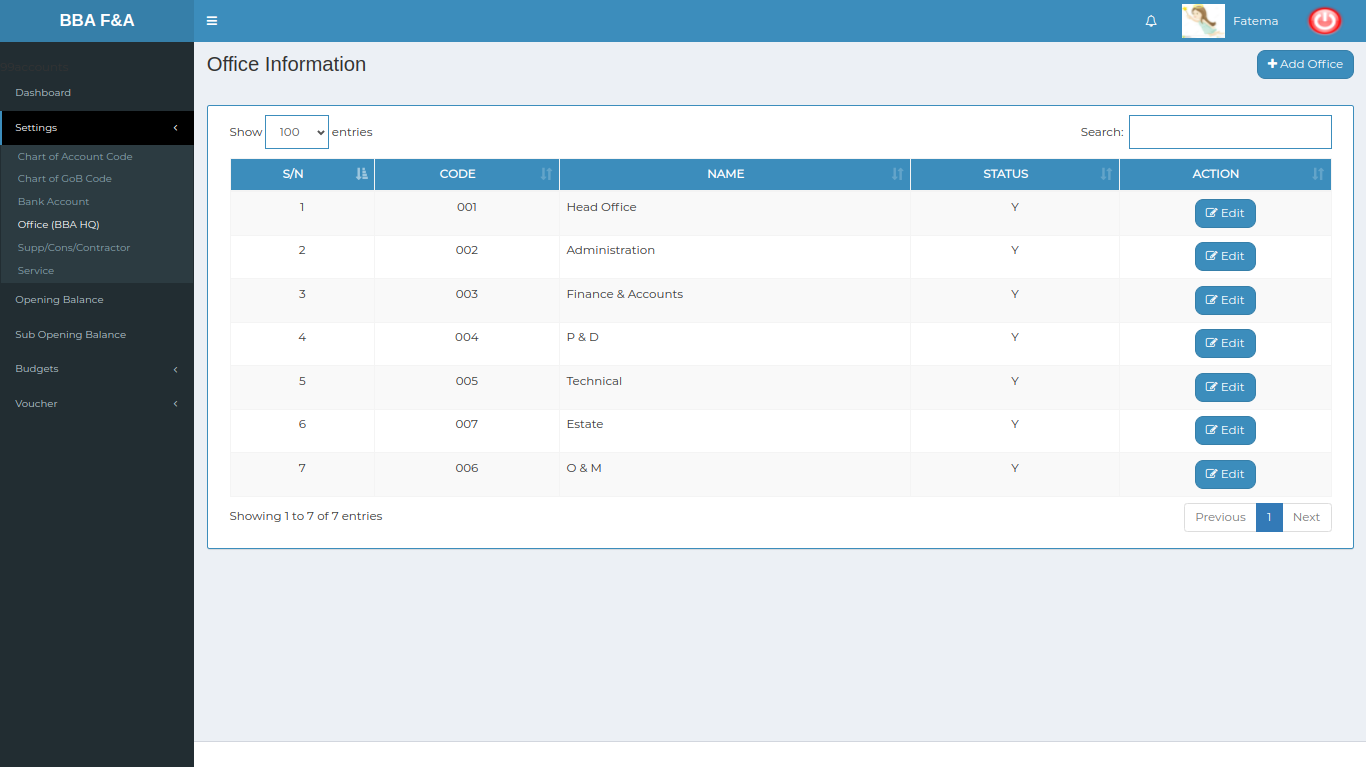
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Bank | 1. Enter new bank name or select a bank name  2. Enter branch name  3. click save | Data Should be saved | Information should be Displayed in list | pass |
| 2 | Bank Accounts | 1. Show all the bank account in a list  2. Click search and type a bank name to find that account information | All information are displayed | No information is missing | P |
| 3 | Update | 1. Select an specific item which need to be updated  2. click update | Updated database | No error while updating | P |
| 4 | Delete | Click delete and press yes to delete an item from the list | Updated database | No error while deleting | P |

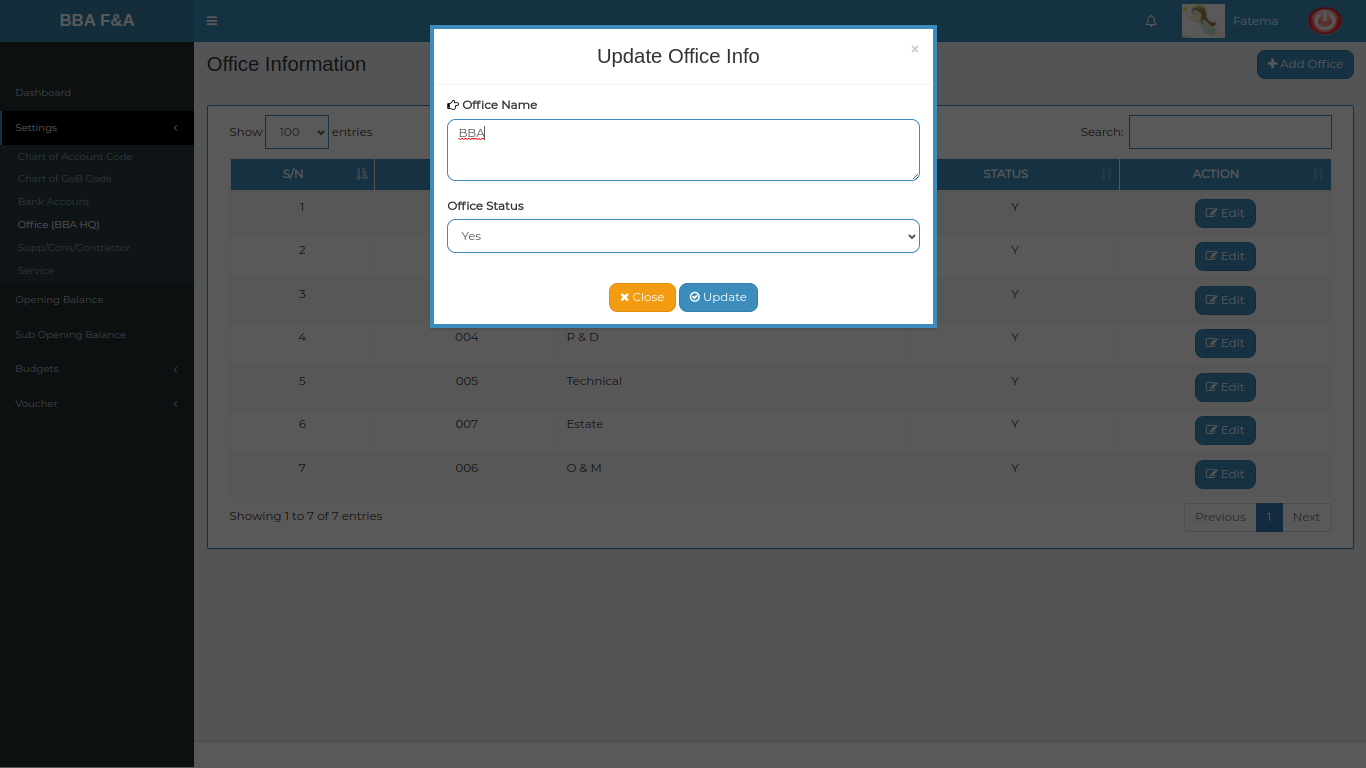
*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Bank | Bank name should be selected and branch name have to be entered | Data checked | No error | P |
| 2 | Bank Accounts | Action button should work fine | No missing information | No error occurred while displaying | P |
| 3 | Update | 1. Existing data should be editable  2. Update button should update the existing information | Updated existing information | Perfectly executed | P |
| 4 | Delete | Delete button should delete the account code information | Updated existing information | Perfectly executed | P |

4. Office(BBA/HQ)







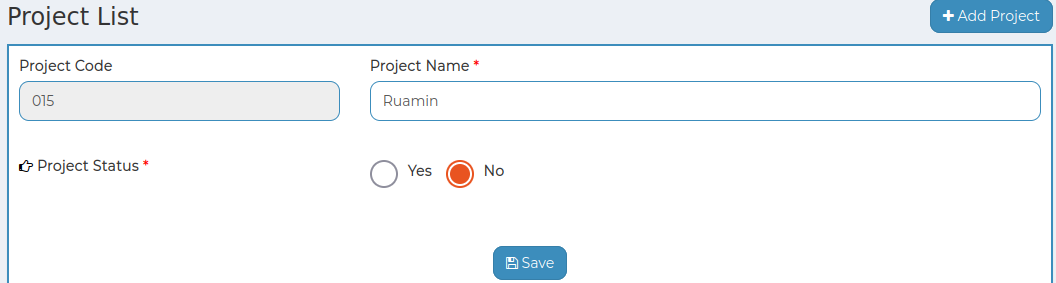
*Positive test case*

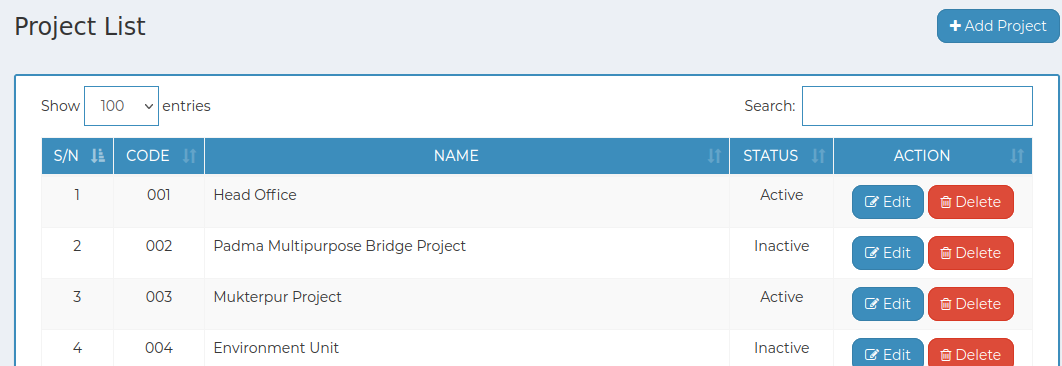
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Office | 1. Enter office name  2. Click save | Data Should be saved | Information should be Displayed in list | pass |
| 2 | Office Information | 1. Click search and type a office name to find that account information  2. Show all the office information | All information are displayed | No information is missing | P |
| 3 | Update | 1. Select an specific item which need to be updated  2. click update | Updated database | No error while updating | P |

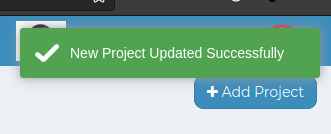
*Negative Test Case*

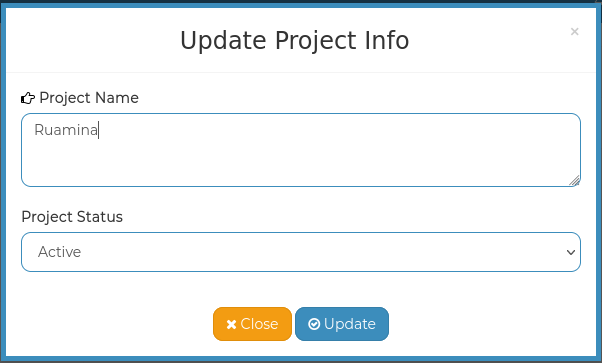
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Office | 1. Office code have to be generated  2. Office name should be entered and save button should work | Data checked | No error | P |
| 2 | Office Information | Action button should work fine | No missing information | No error occurred while displaying | P |
| 3 | Update | 1. Existing data should be editable  2. Update button should update the existing information | Updated existing information | Perfectly executed | P |

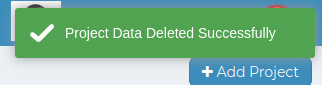
5. Projects











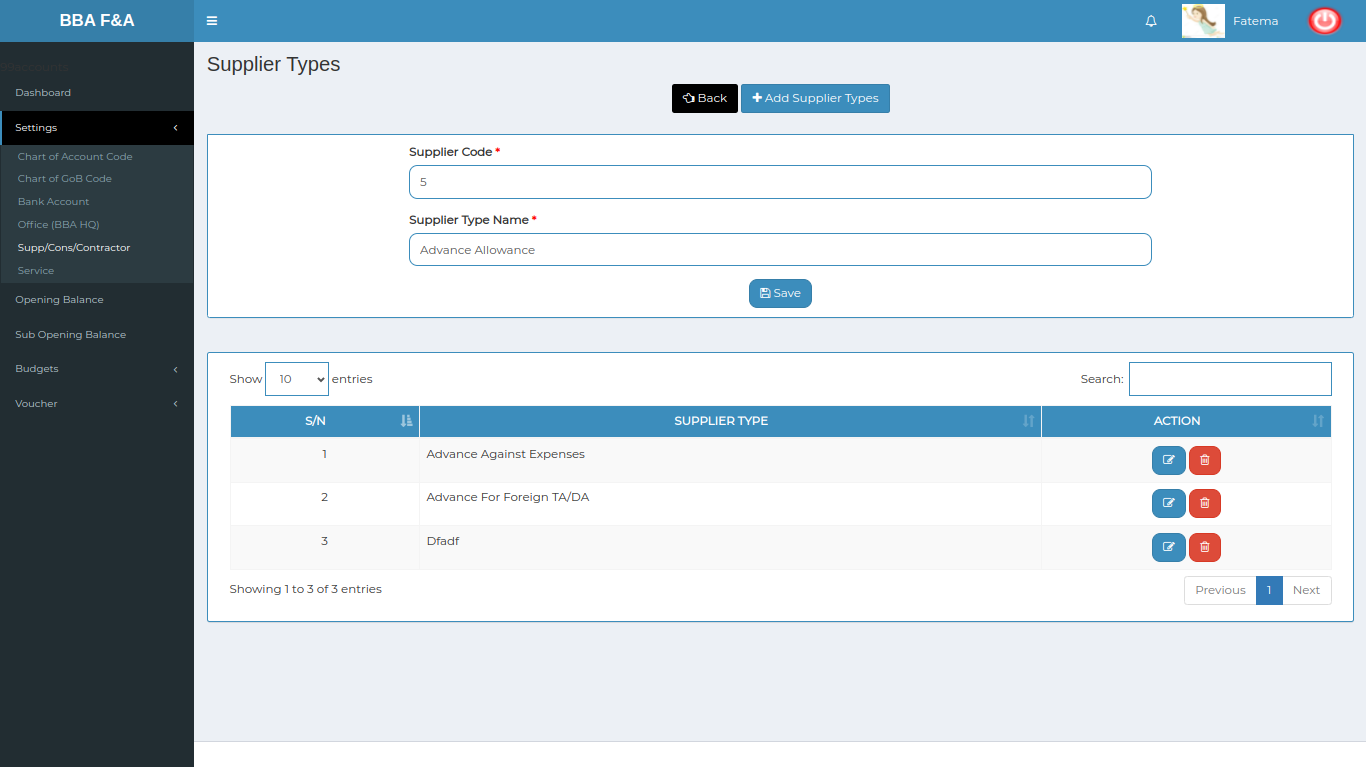
*Positive test case*

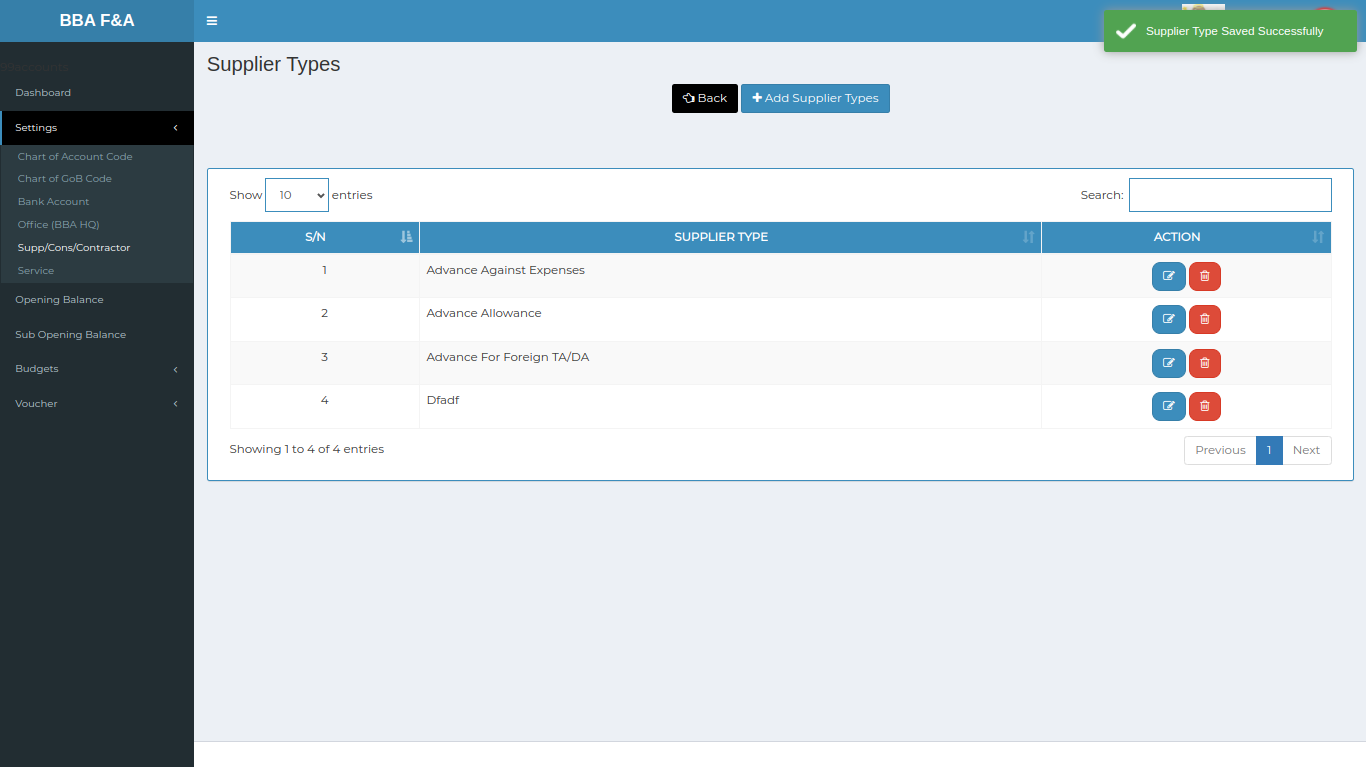
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Project | 1. Enter project name  2. Click save | Data saved successfully | No error while saving | P |
| 2 | Project list | 1. Click search and type a project name to find that account information  2. Show all the project list | All information are displayed | No information is missing | P |
| 3 | Update & Delete | 1. Select an specific item which need to be updated  2. click update | Updated database | No error while updating or deleting any information | P |

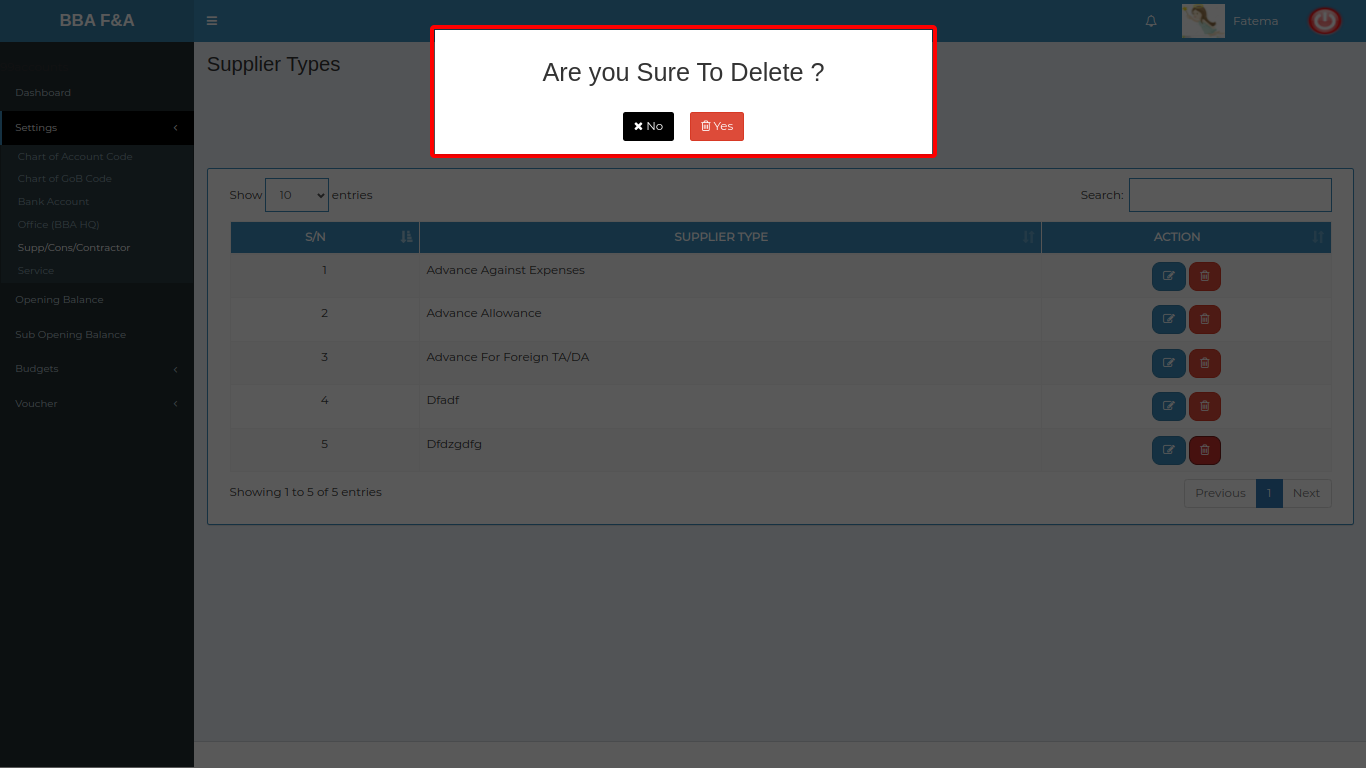
*Negative Test Case*

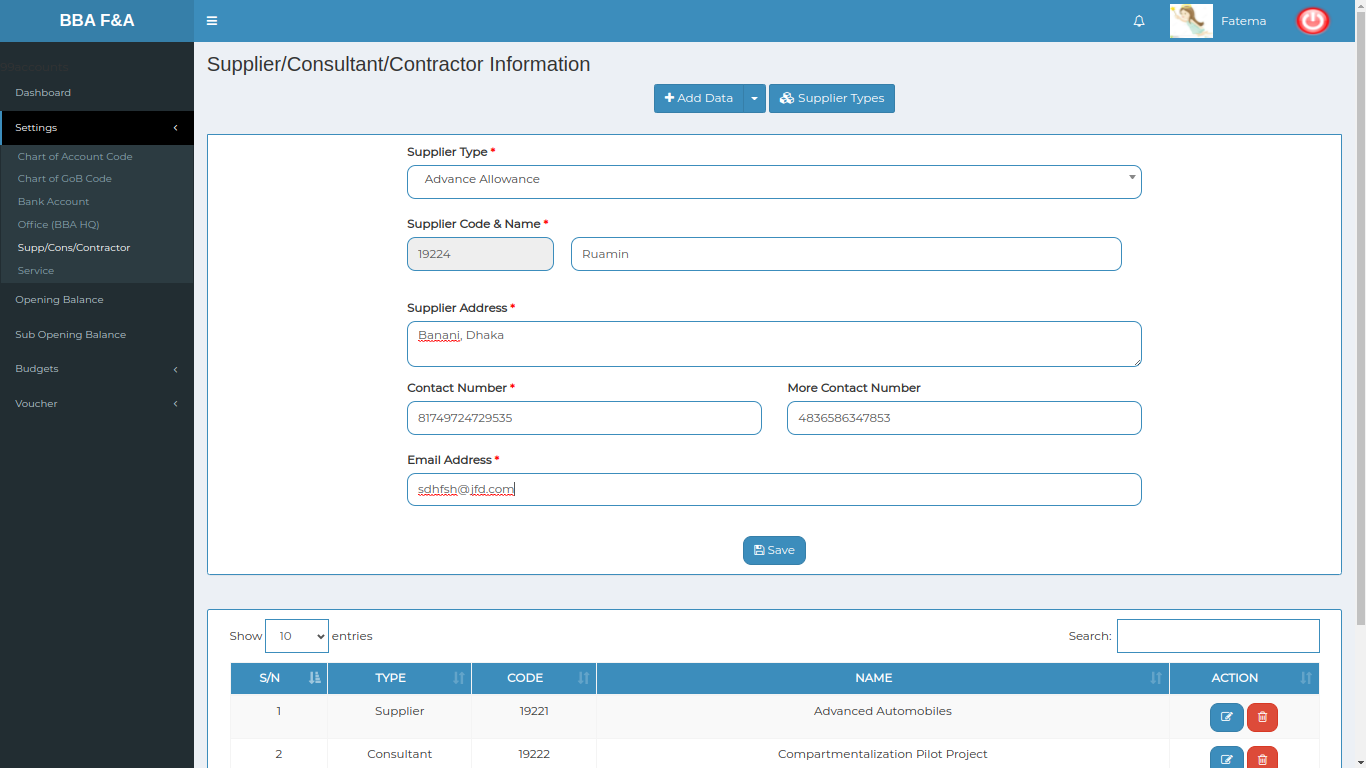
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Project | 1. project code have to be generated  2. project name should be entered and save button should work | Data checked | No error | P |
| 2 | Project list | Action button should work fine | No missing information | No error occurred while displaying | P |
| 3 | Update & Delete | 1. Existing data should be editable  2. Update button should update the existing information | Updated existing information & delete selected information | Perfectly executed the operation | P |

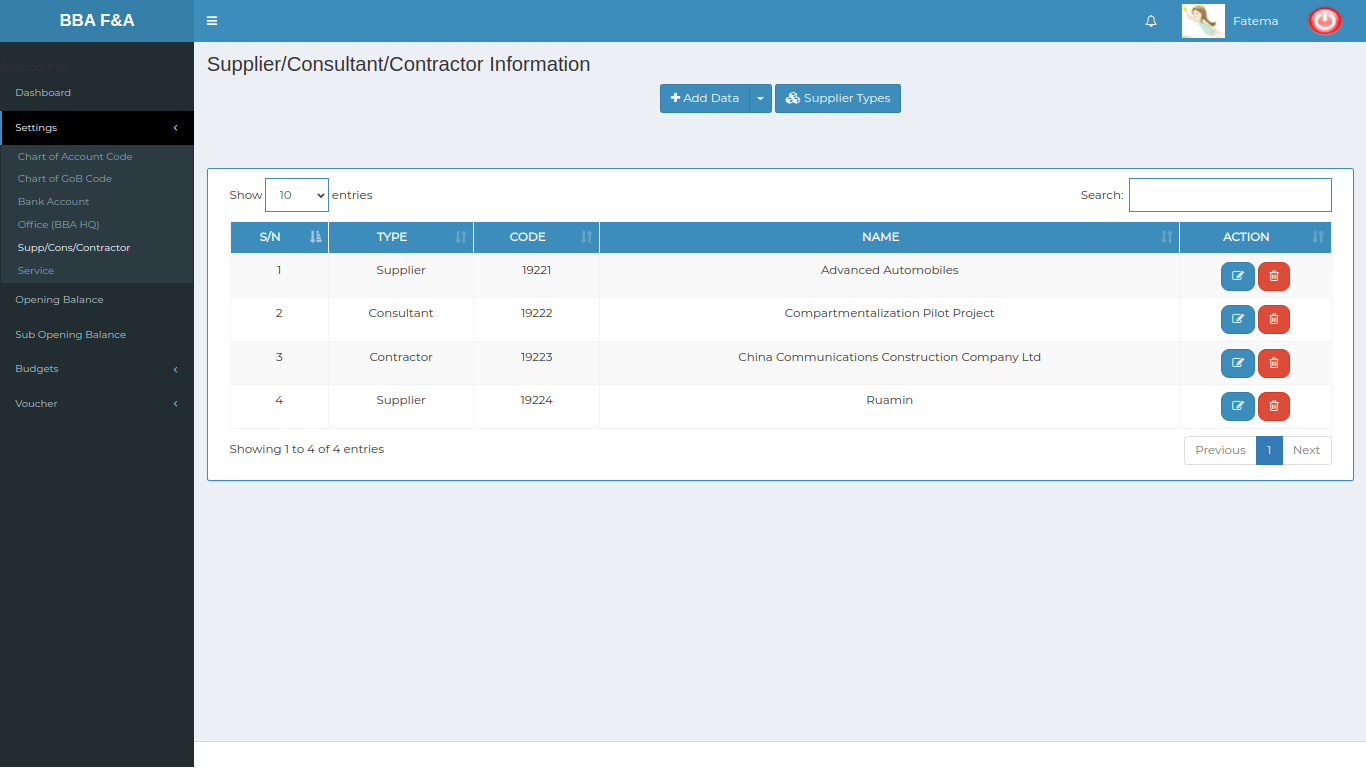
6. Supplier/Consultant/Contractor

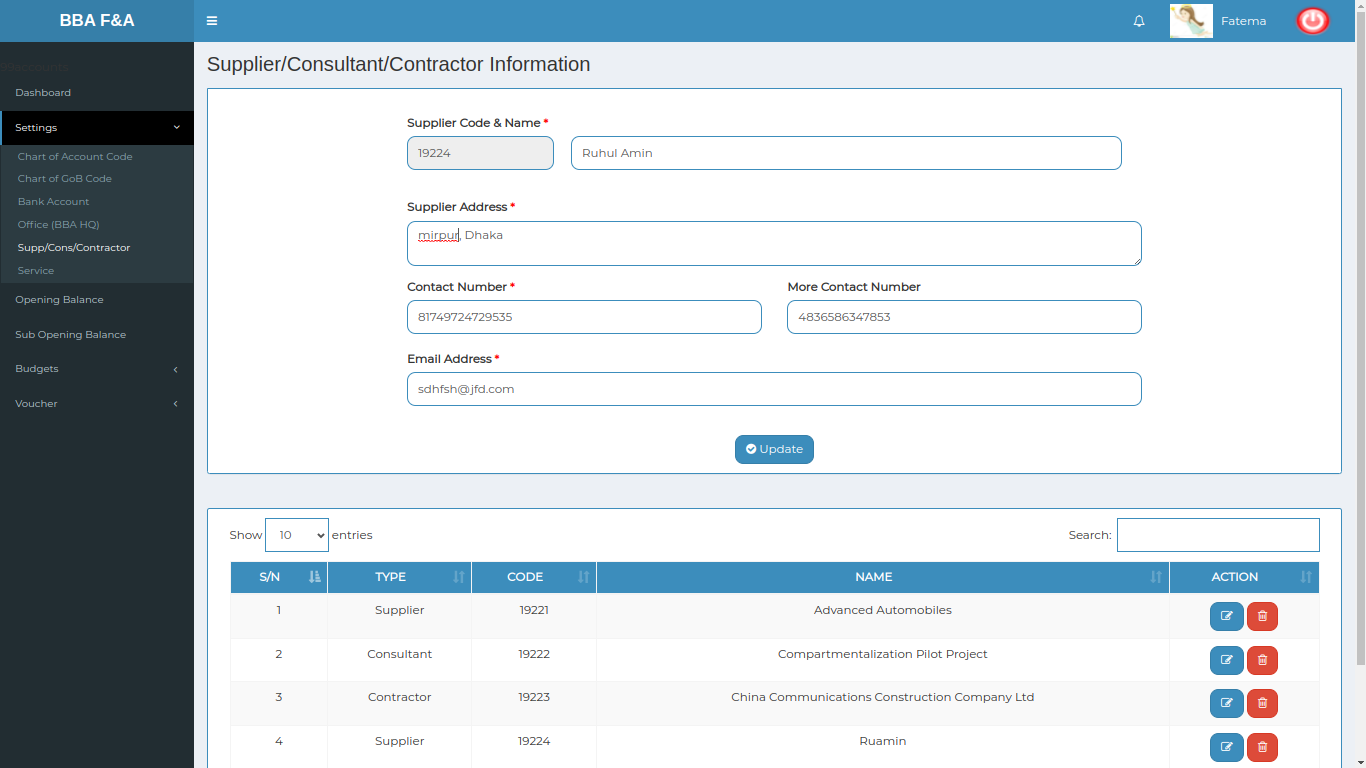


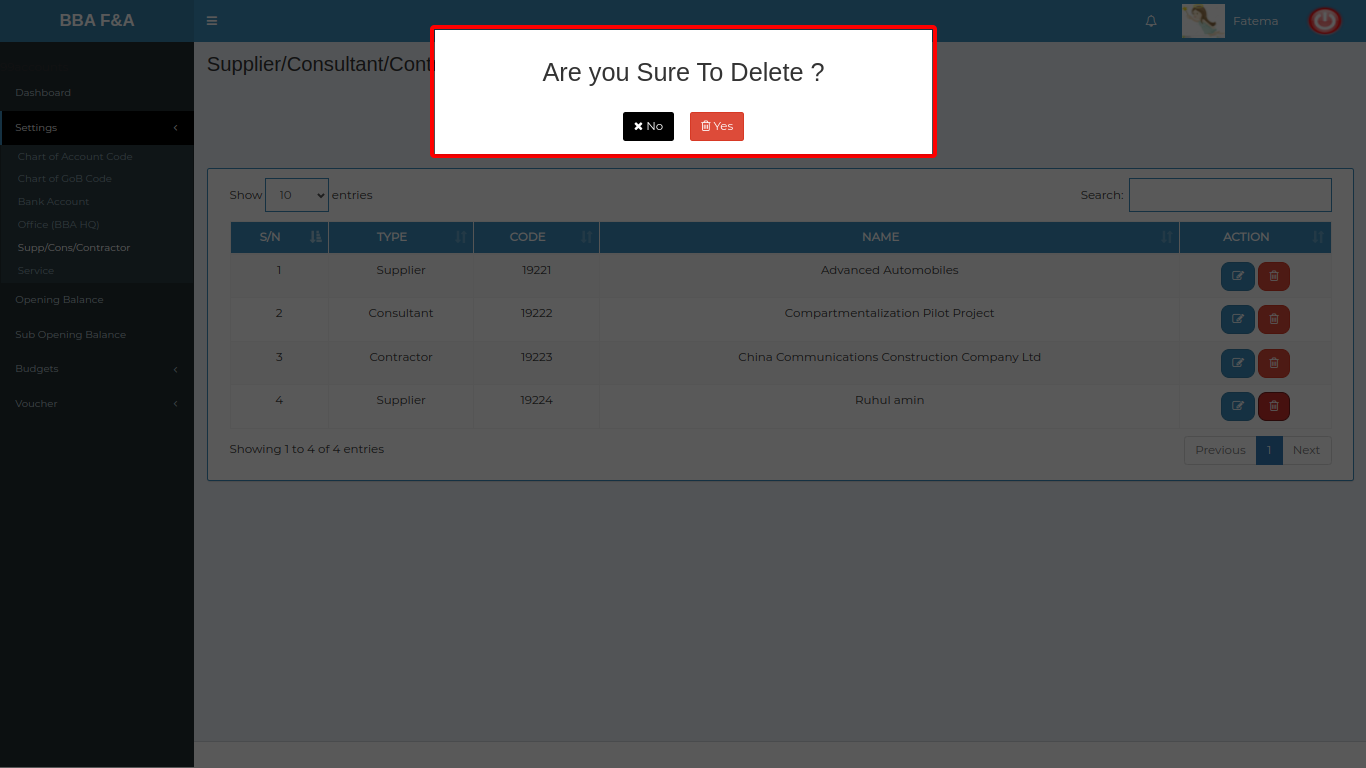












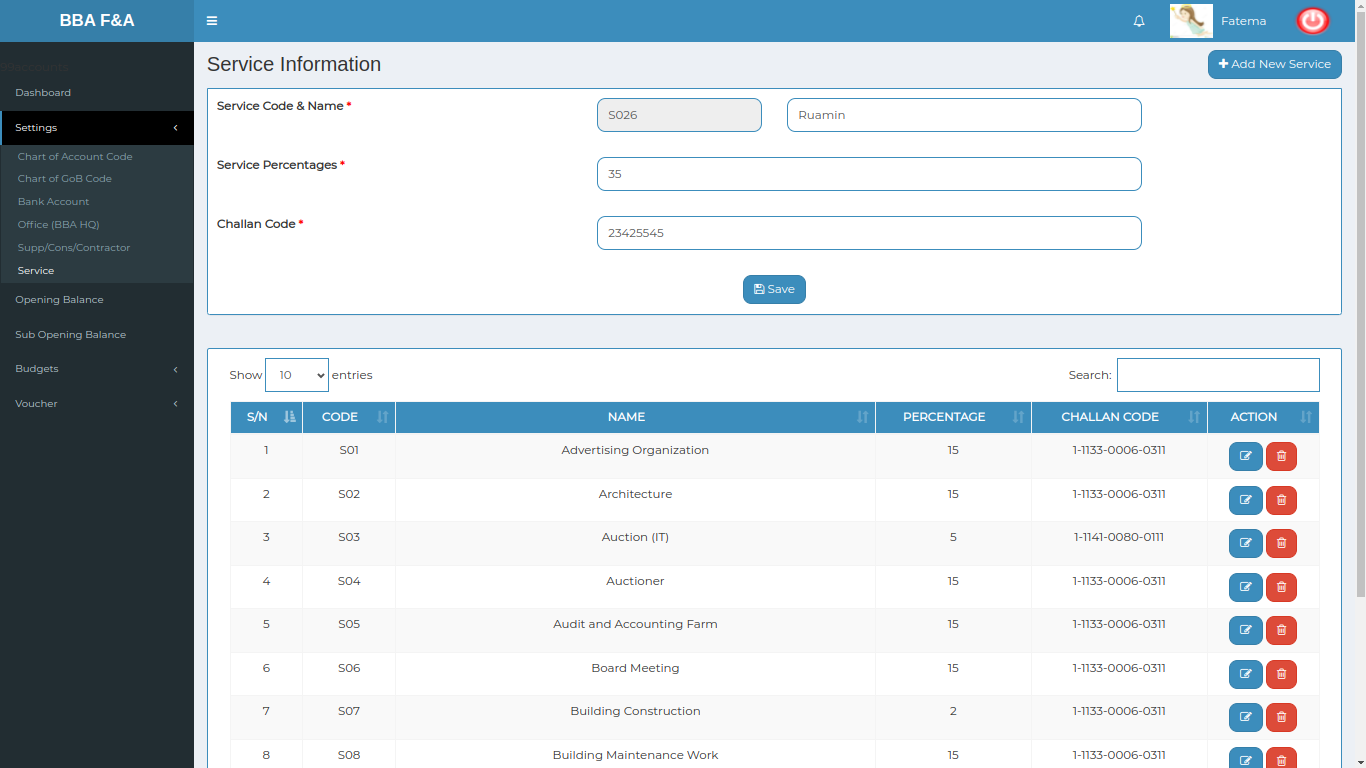
*Positive test case*

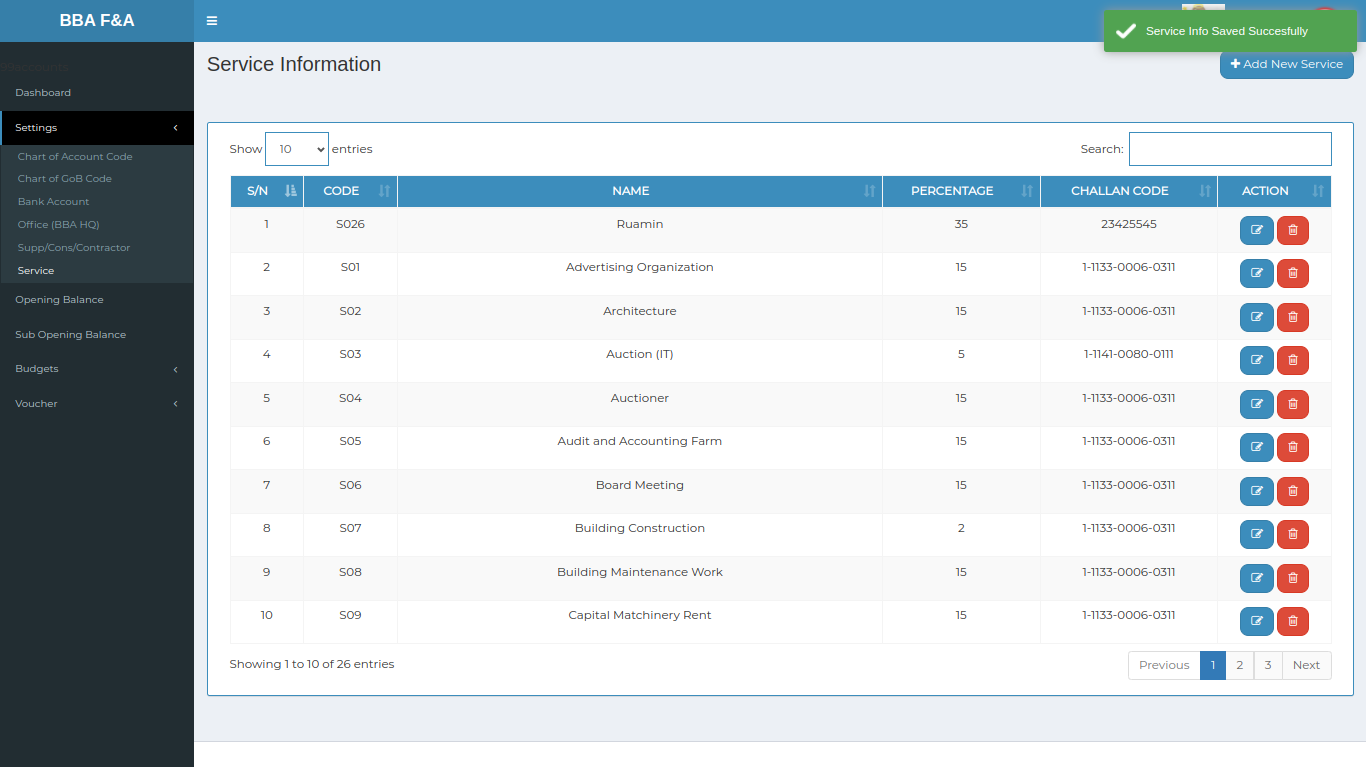
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Supplier types | 1. Enter supplier code & Supplier Type name  2. Click save | Data saved successfully | No error while saving | P |
| 2 | Supplier Types | 1. Display all the supplier type information  2. Type a Supplier type in search bar to find that information  2. Update button can edit the existing information  3. Delete button can delete current information | All information are displayed | No information is missing | P |
|  | Add Data | 1. Select data Type & Select supplier type  2. Enter supplier name, contact number, email address  3. Click save | Data saved successfully | No error while saving | P |
|  | Supplier/Consultant/Contractor information | 1. Display all the information  2. type a supplier type in search bar to find specific information | All information are displayed | No information is missing | P |
|  | update | 1. Select an specific item which need to be updated  2. click update | Updated database | No error while updating | P |
|  | delete | Click delete and press yes to delete an item from the list | Updated database | No error while deleting any information | P |

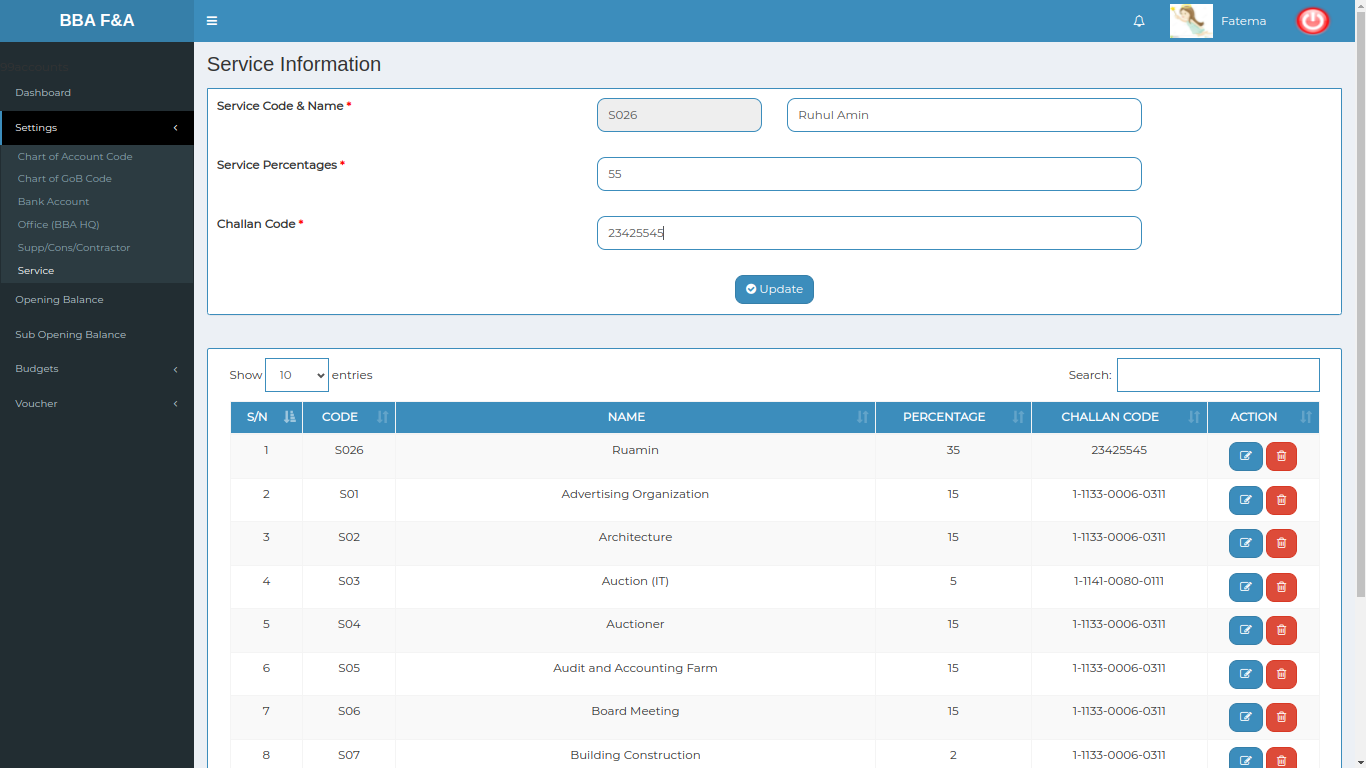
*Negative Test Case*

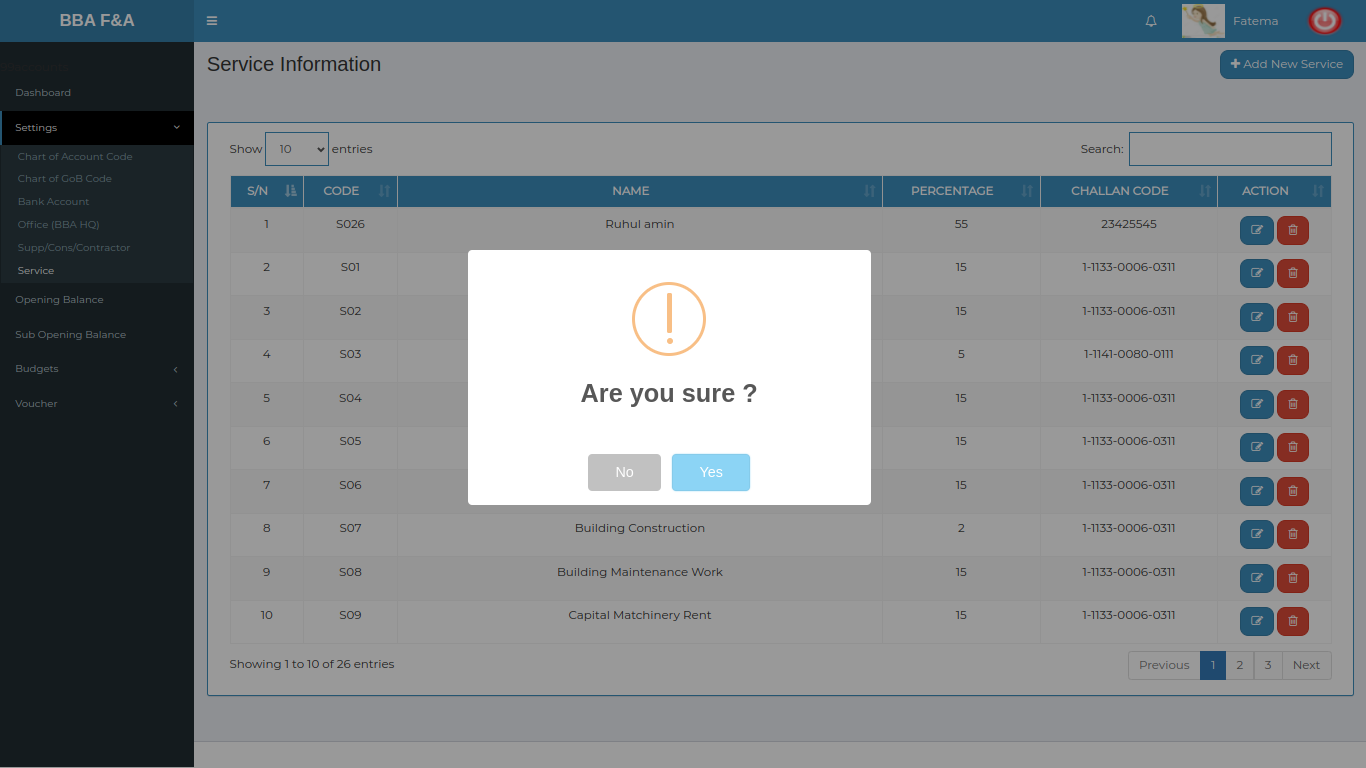
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Supplier types | 1. Enter supplier code & suppler type name  2. click save | Data checked | No error | P |
| 2 | Supplier Types | 1. Display all the information  2. type a supplier type in search bar to find specific information  3. Update | No missing information | No error occurred while displaying | P |
|  | Add Data | 1. Data type & supplier type should be selected  2. Address should be valid  3. Contact number & email address should be verified  4. save button should work fine | Data checked | No error | P |
|  | Supplier/Consultant/Contractor information | 1. Display all the information  2. type a supplier type in search bar to find specific information | No missing information | No error occurred while displaying | P |
|  | update | 1. Existing data should be editable  2. Update button should update the existing information | Updated existing information | Perfectly executed | P |
|  | delete | Delete button should delete the account code information | Updated existing information | Perfectly executed | P |

6. Service









*Positive test case*

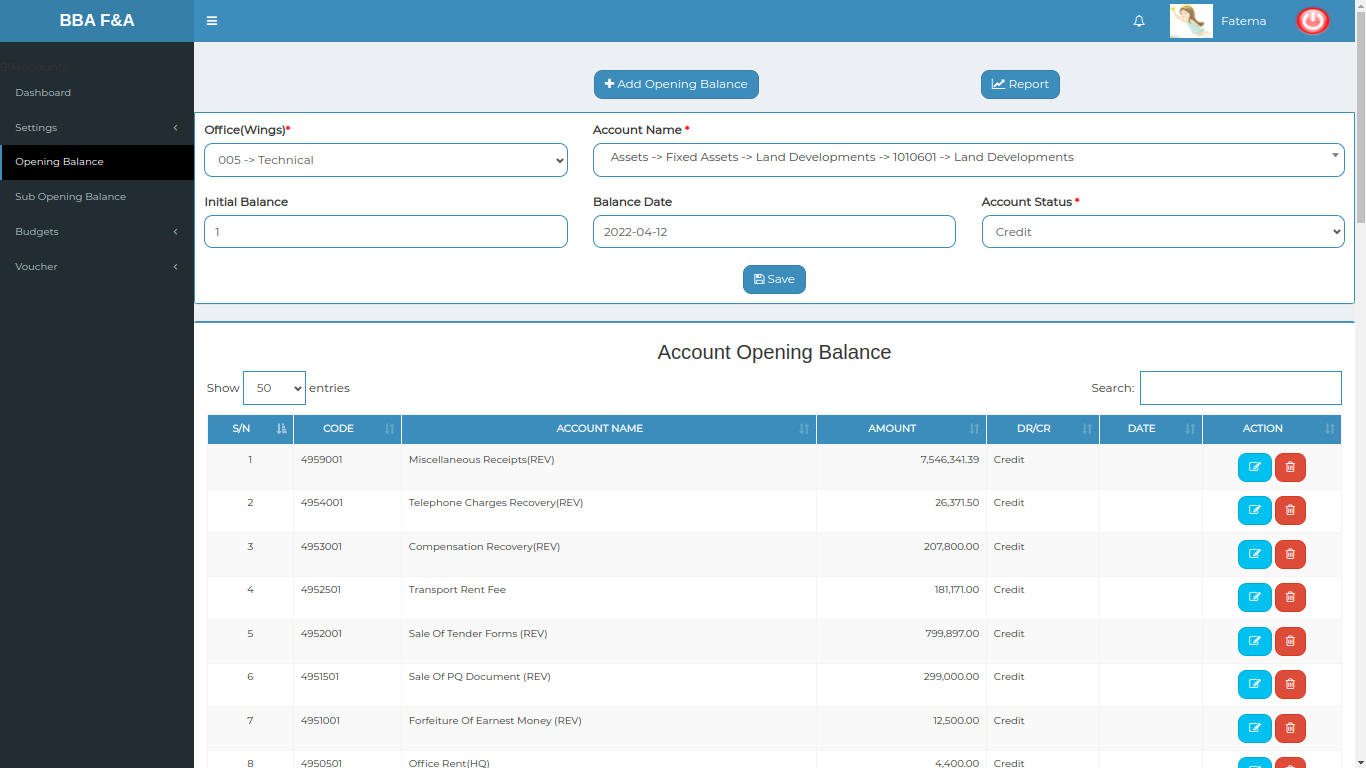
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add new service | 1. Enter service name, percentage & challan code  2. Click save | Data Should be saved | Information should be Displayed in list | pass |
| 2 | Service information | 1. Display all the information  2. type a supplier type in search bar to find specific information | All information are displayed | No information is missing | P |
| 3 | Update | 1. Select an specific item which need to be updated  2. click update | Updated database | No error while updating | P |
| 4 | Delete | Click delete and press yes to delete an item from the list | Updated database | No error while deleting | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add new service | 1. service name, percentage & challan code should be entered  2. save button should work fine | Data checked | No error | P |
| 2 | Service information | Action button should work fine | No missing information | No error occurred while displaying | P |
| 3 | Update | 1. Existing data should be editable  2. Update button should update the existing information | Updated existing information | Perfectly executed | P |
| 4 | Delete | Delete button should delete the account code information | Updated existing information | Perfectly executed | P |

**Test Scenario 3: Opening Balance**

1. Add Opening Balance



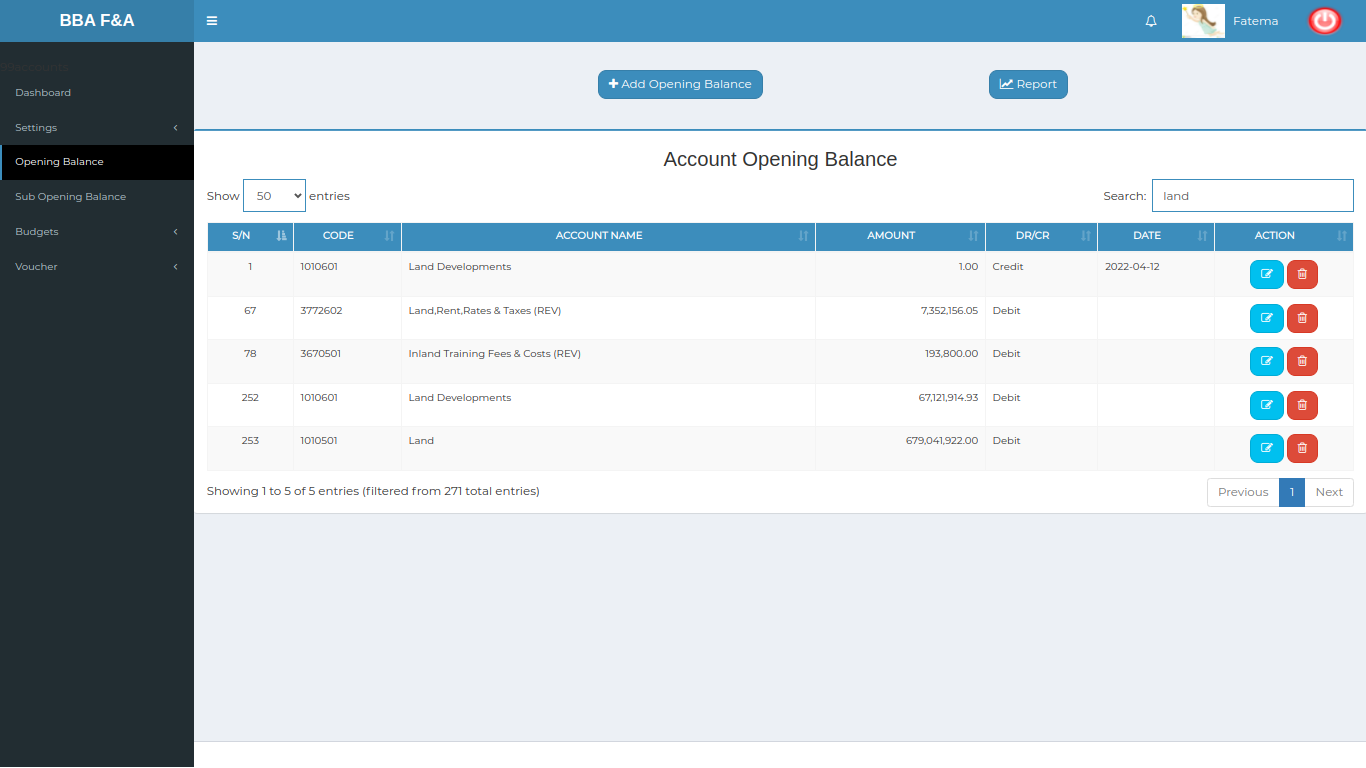
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Opening Balance | 1. Select office name & account name  2. Enter initial balance & balance date  3. Select account status  4. click save | Data Should be saved | Information should be Displayed in list | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Opening Balance | Display all the information & can search to find information | Data checked | No error | P |

2. Account Opening Balance Information



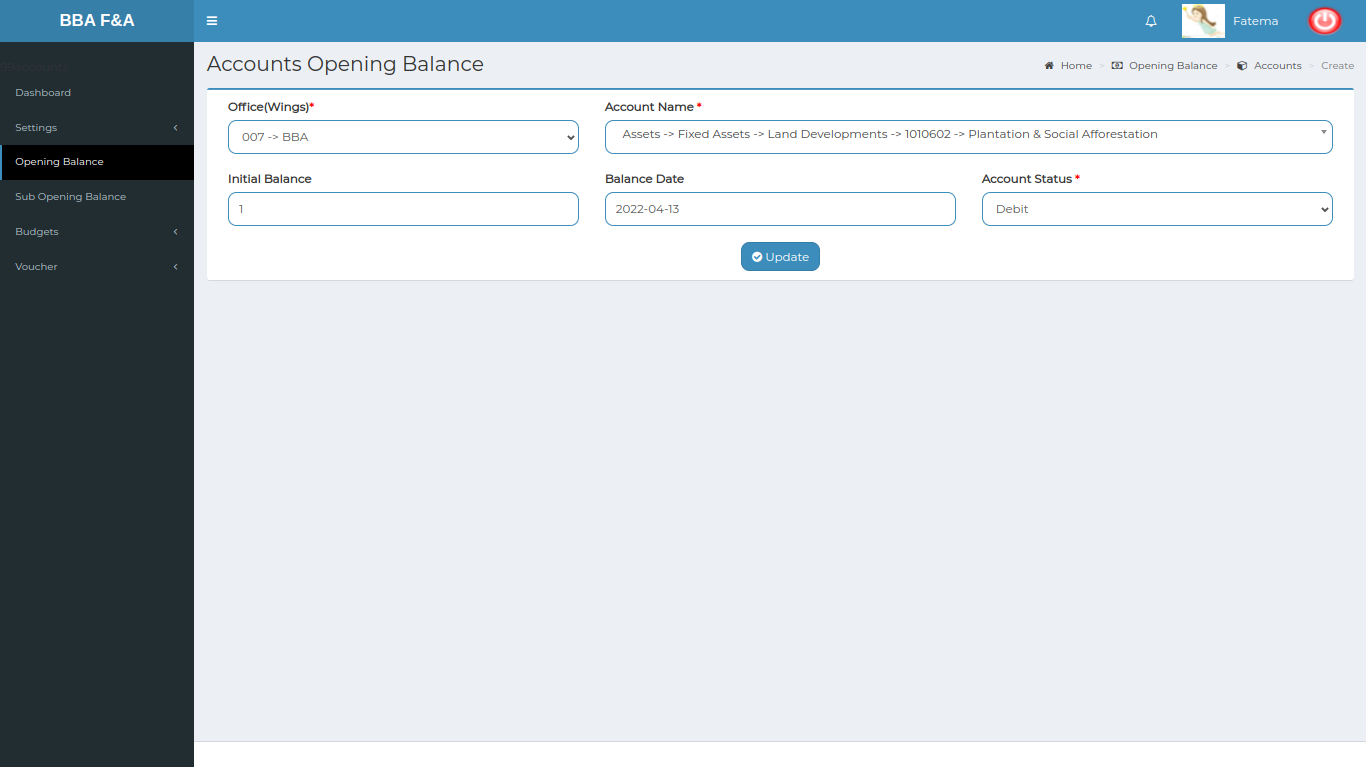
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Account Opening Balance List | 1. Display all the information  2. Search an item | All information are displayed | No information is missing | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Account Opening Balance List | Action button should work fine | No missing information | No error occurred while displaying | P |

3. Update Opening Balance



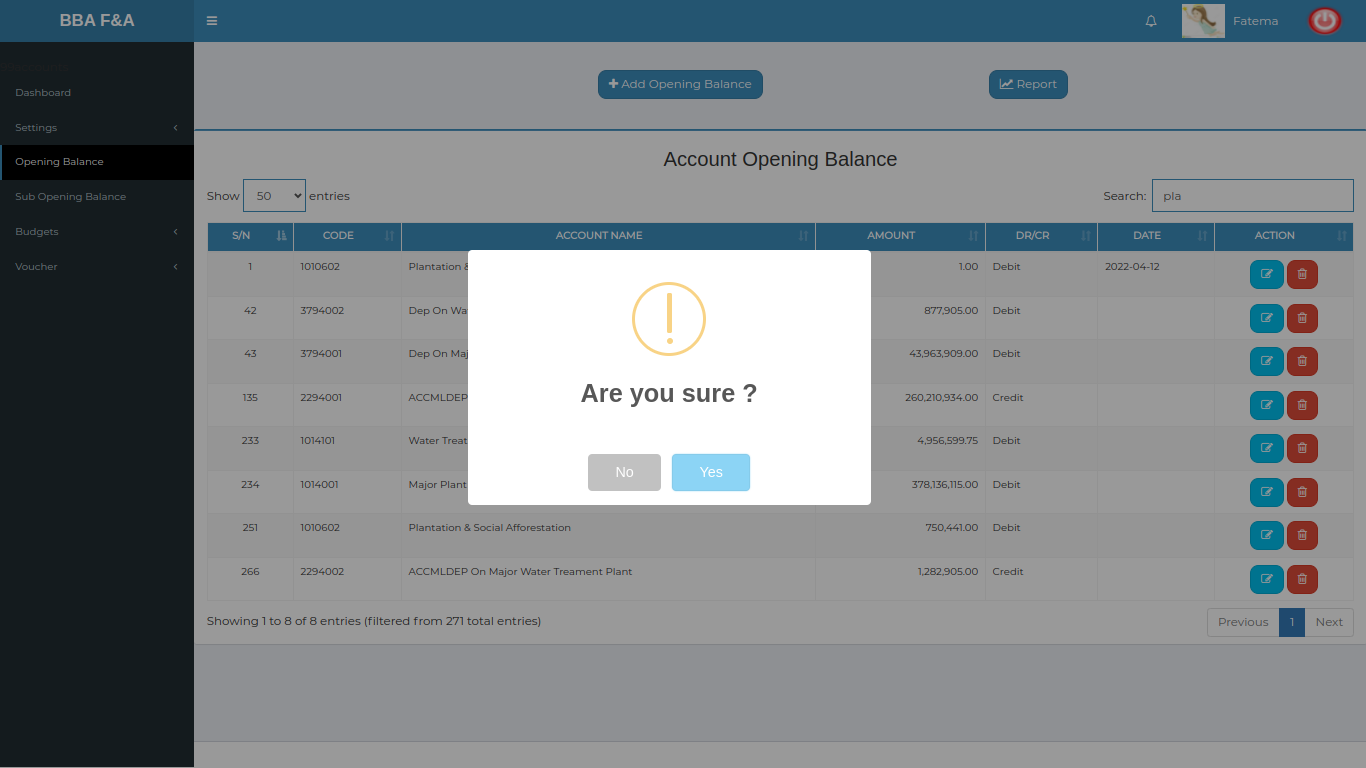
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Update | Update existing data by selecting neccessary information | Updated database | No error while updating | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Update | Update button should update information | Updated existing information | Perfectly executed | P |

4. Delete Opening Balance



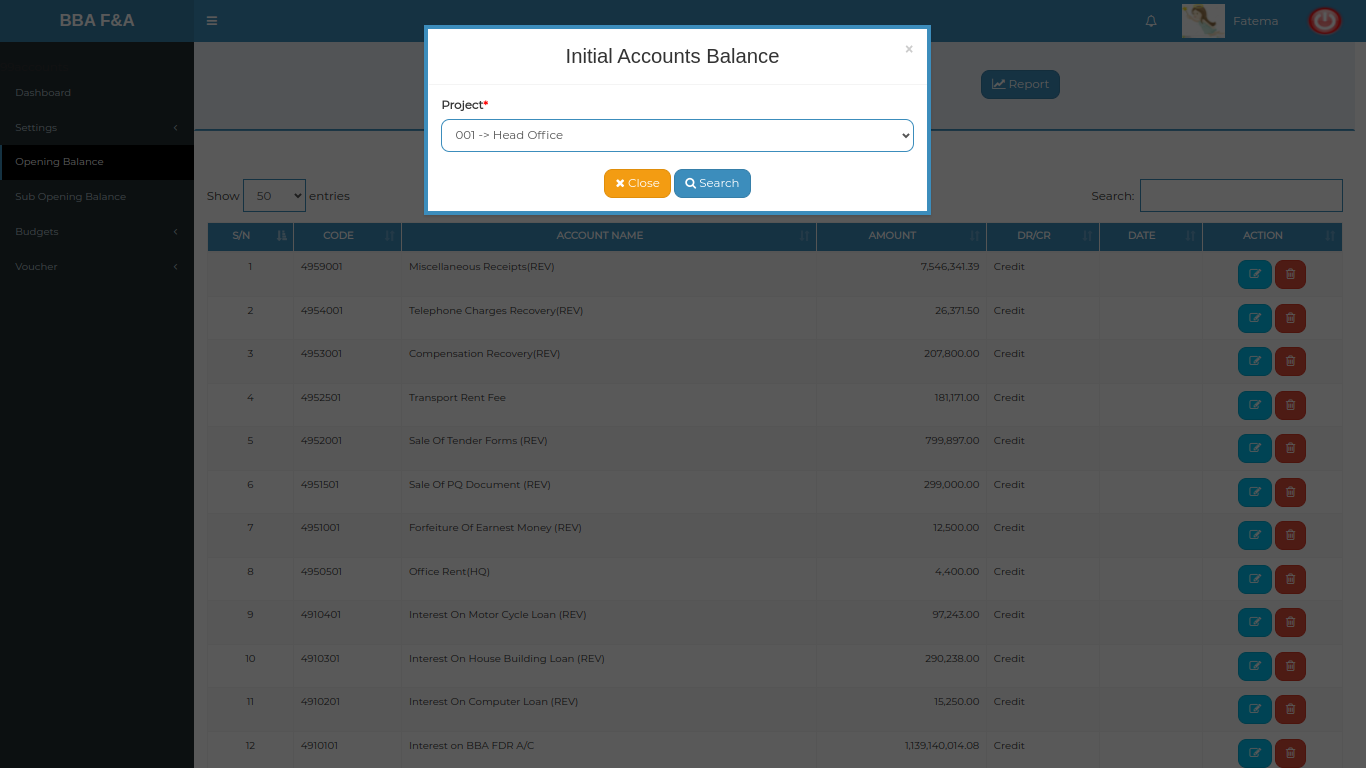
*Positive test case*

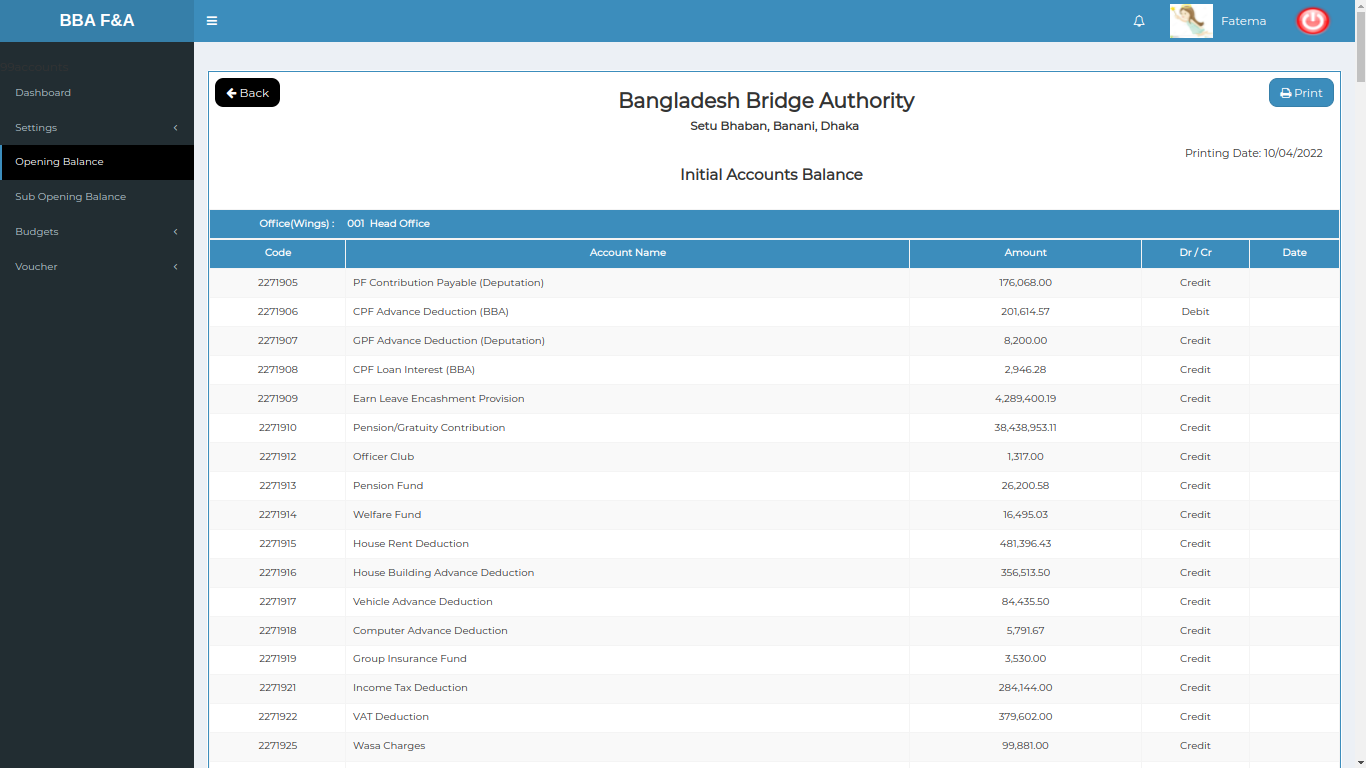
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Delete | Delete the information by pressing delete button | Updated database | No error while deleting | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Delete | Button should work fine | Updated existing information | Perfectly executed | P |

5. Report Opening Balance





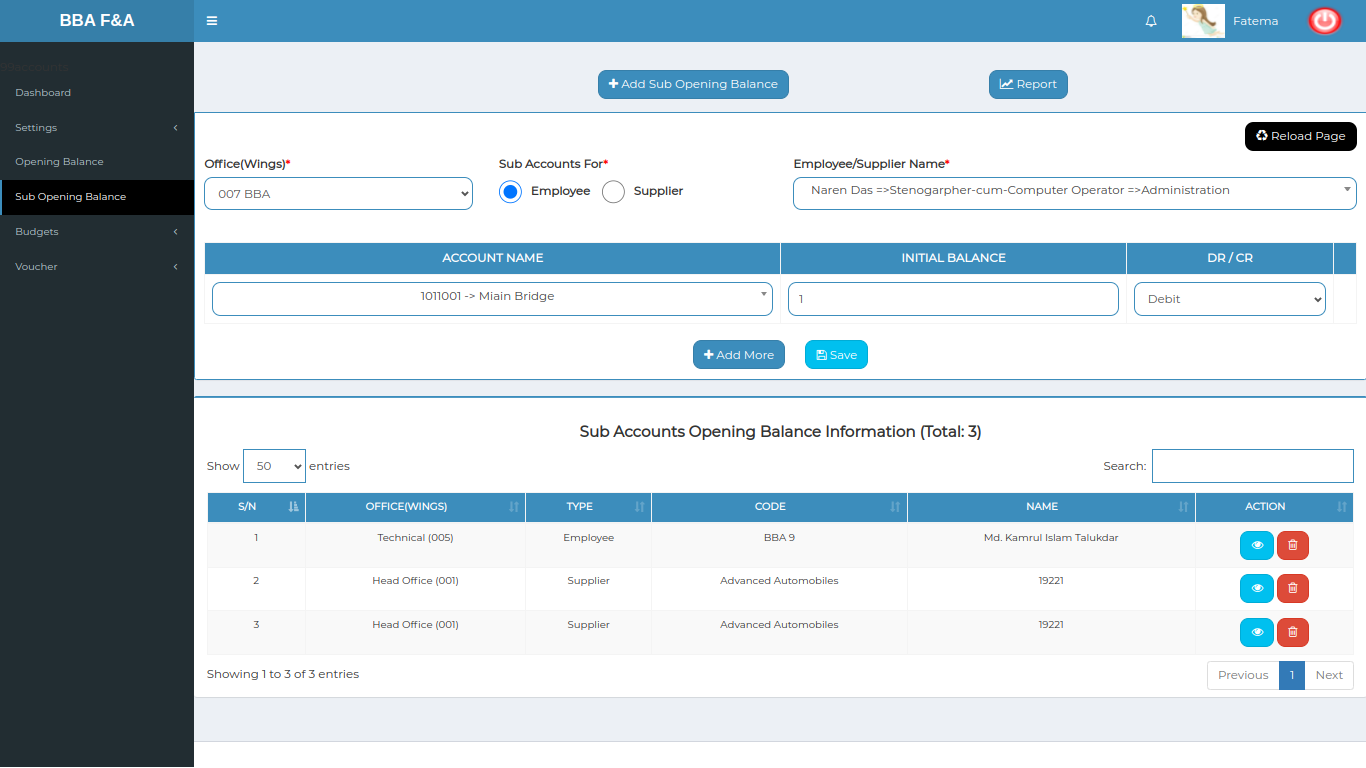
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Report | 1. Select report type  2. A report is displayed | Displayed | No error while generating | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Report | 1. Report type should be selected  2. report can be generated in document format | Generated | No data missing | P |

**Test Scenario 4: Sub-Opening Balance**



1. Add Sub-Opening Balance

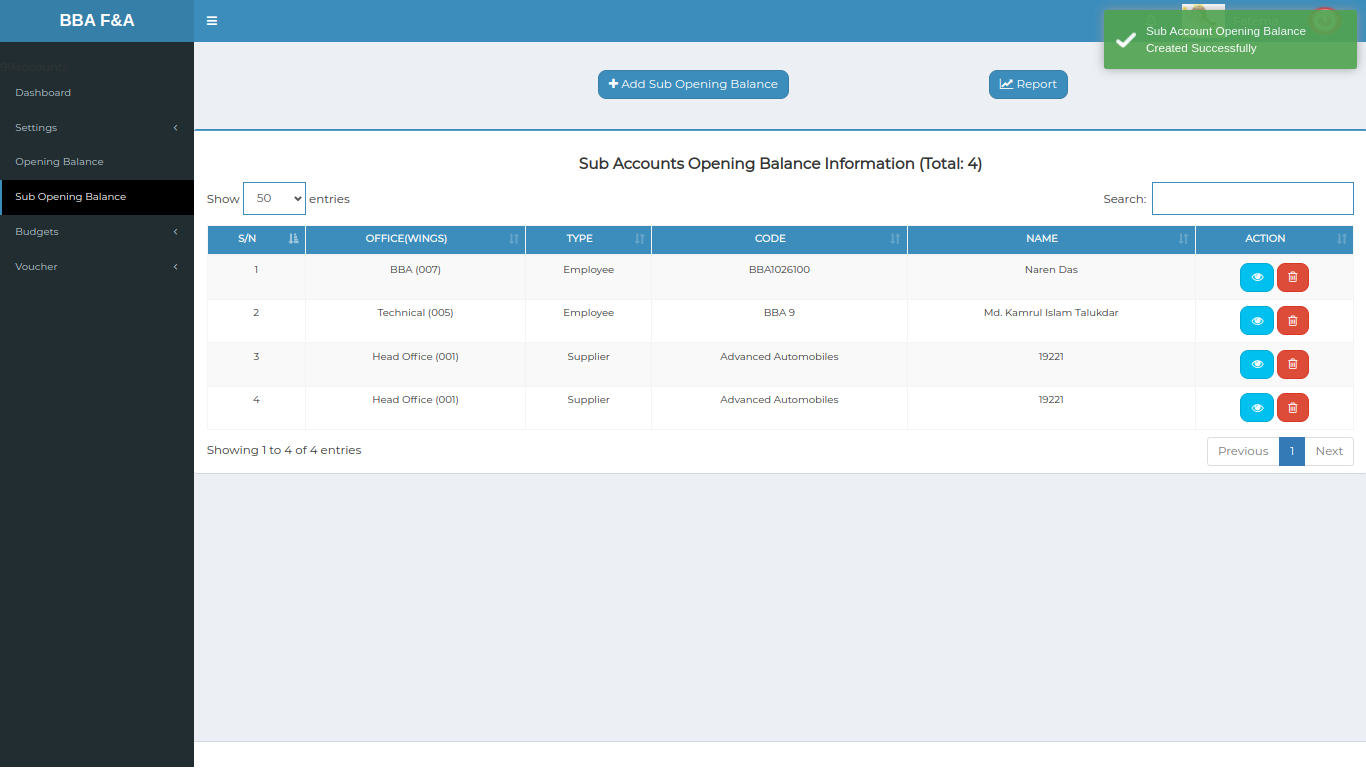
*Positive test case*

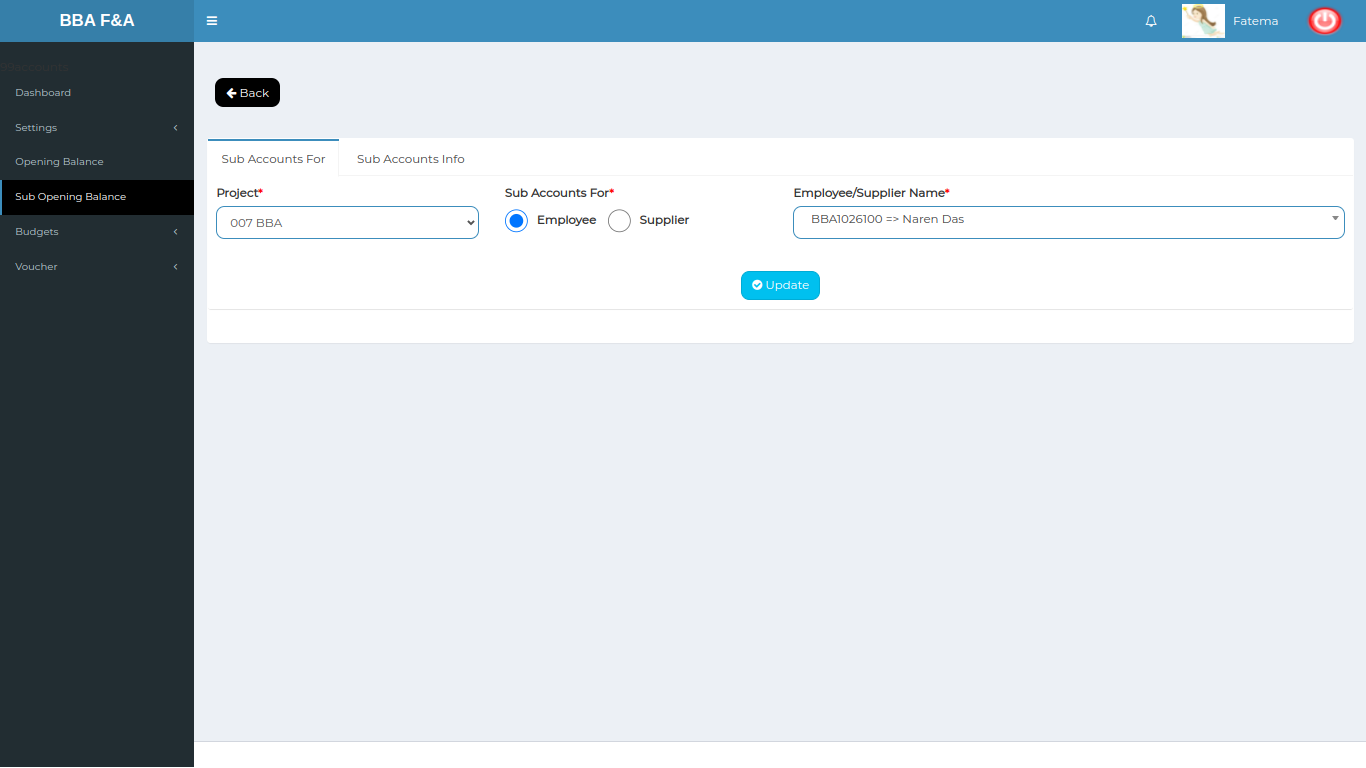
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add Balance | 1. Select office name  2. choose sub-accounts  3. Select employee or supplier name  4. Enter account name , initial balance & select account status  5. click add more or click save | All information are displayed | No information is missing | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add balance | 1. Necessary data should be selected & entered  2. save button should work fine | No missing information | No error occurred while displaying | P |

2. View & Update Sub-Opening Balance





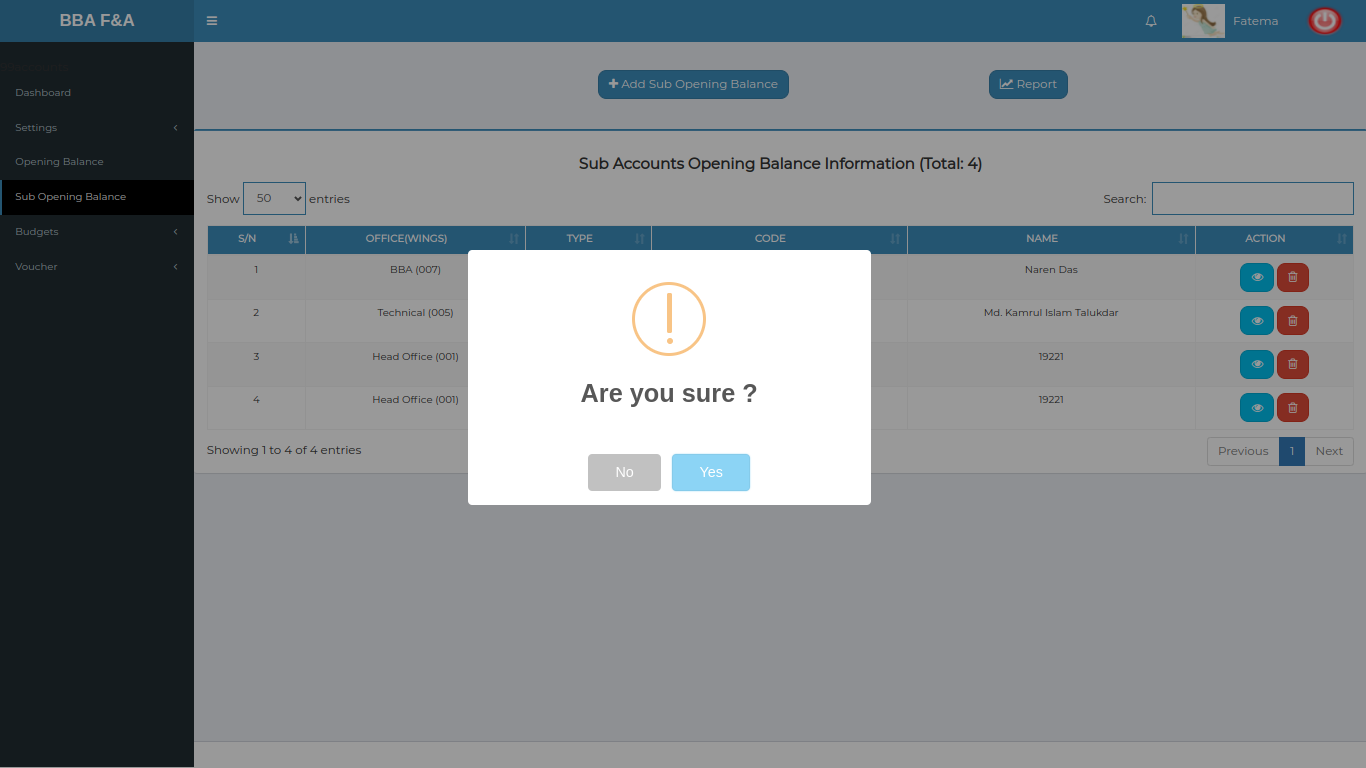
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Sub-opening account list | 1. Display all the information  2. search bar to find specific information | All information are displayed | No information is missing | P |
| 2 | Update | 1. select information & update them  3. click update to save data | Updated database | No error while updating | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Sub-opening account list | Action buttons should work fine | No missing information | No error occurred while displaying | P |
| 2 | Update | Update button should update the following database | Updated existing information | Perfectly executed | P |

3. Delete Sub-Opening Balance



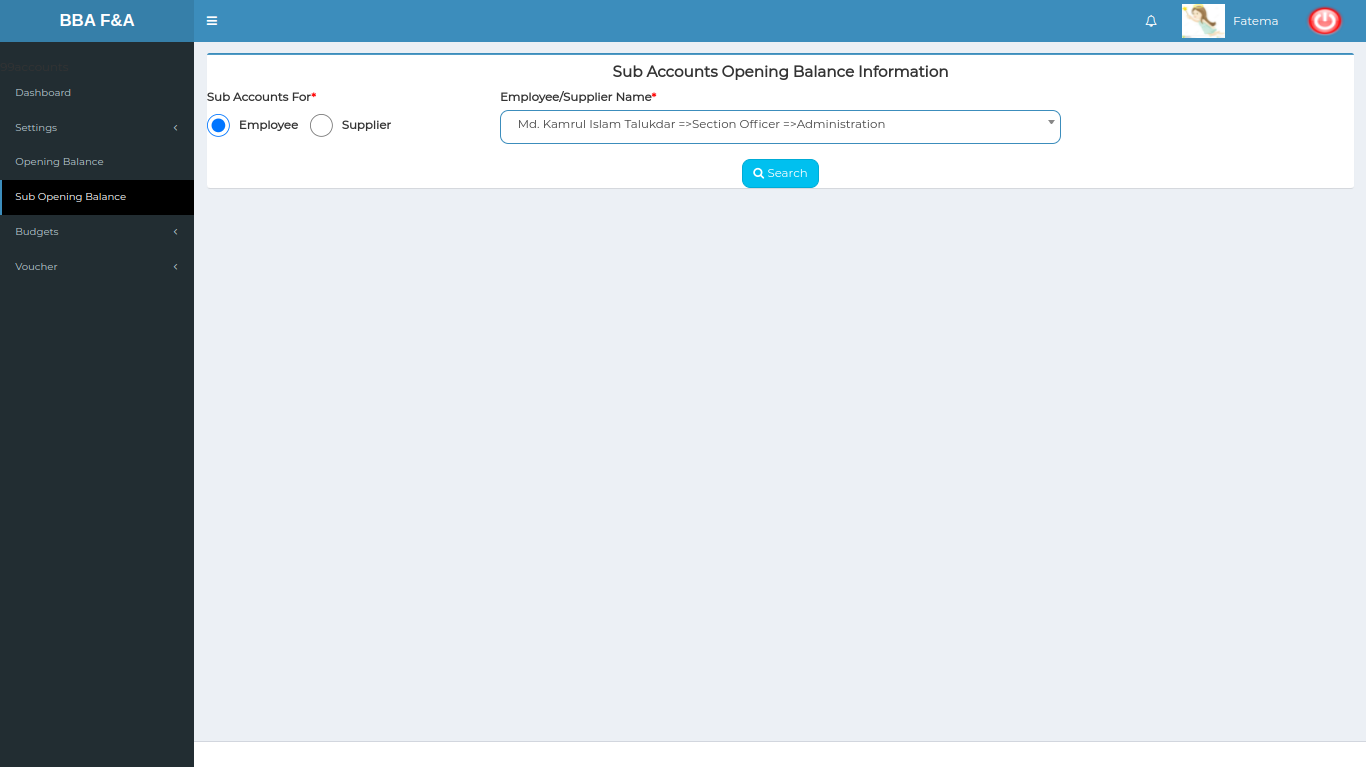
*Positive test case*

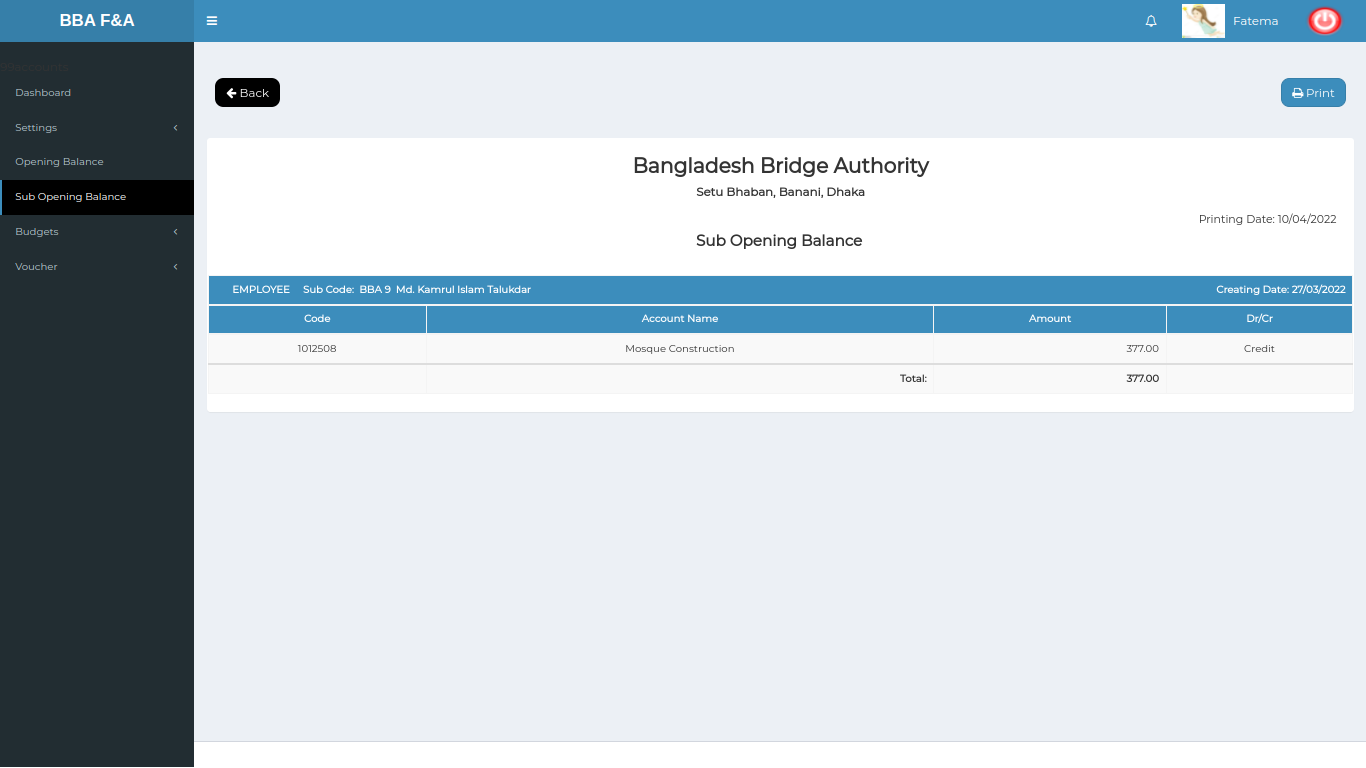
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Delete information | Delete the sub-opening account information | Updated database | No error while deleting | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Delete information | Delete button should work fine | Updated existing information | Perfectly executed | P |

4. Report Sub-Opening Balance





*Positive test case*

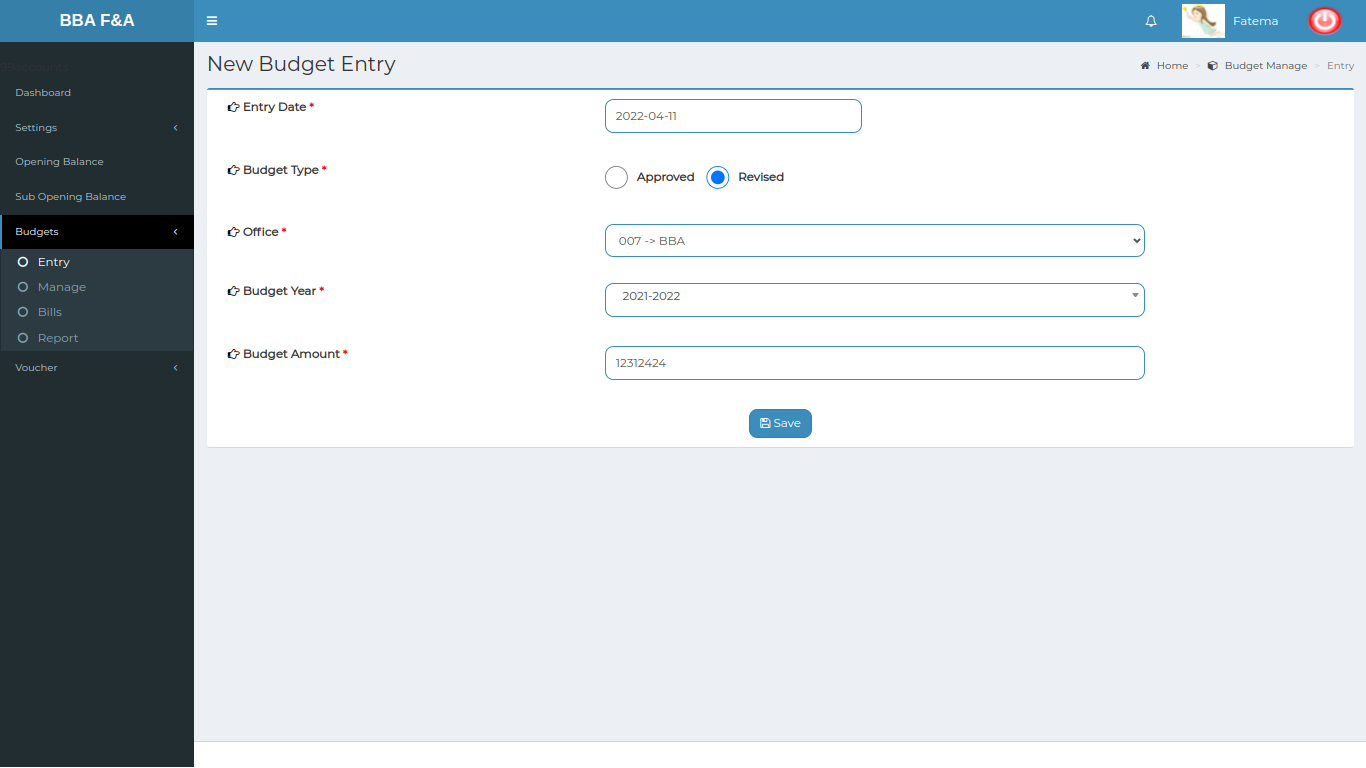
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Select information | 1. Choose sub-account for employee or supplier  2. click search | Searched successfully | No error | P |
| 2 | Report | Show sub opening balance | Displayed | No error while generating | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Select information | 1. sub account should be selected  2. search button should work fine | Perfectly executed | No error | P |
| 2 | Report | Report should be generated in document format | Generated | Displayed | P |

**Test Scenario 5: Budget**

**1. Budget Entry**

****

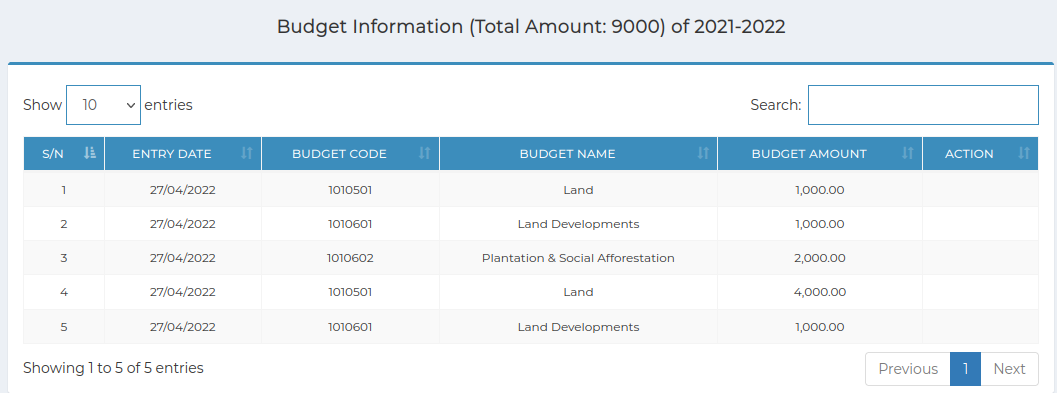
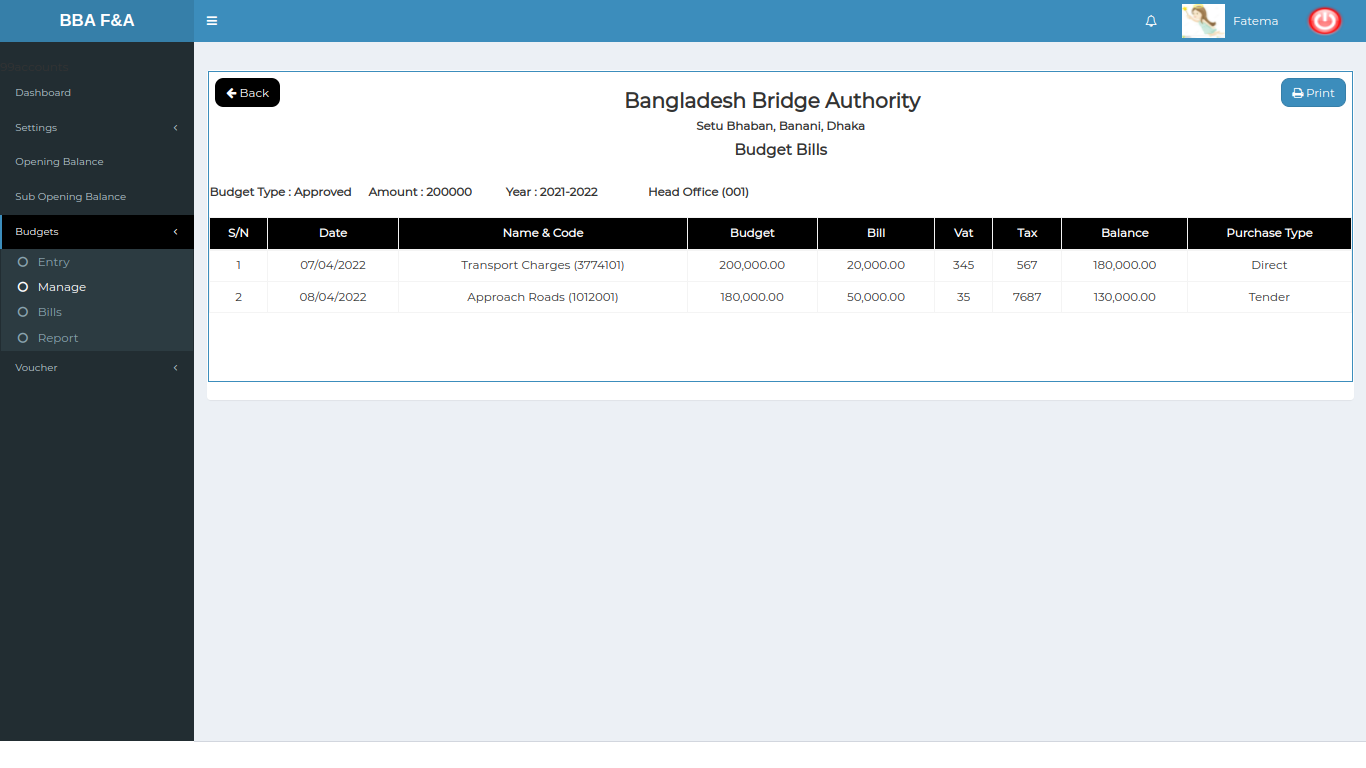
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | New Budget Entry | 1. Select date & choose budget type & budget sector  3. enter budget amount  4. click add to list  5. click delete to delete current information  6. click save to save all the data | Data saved successfully | No error occurred while adding information | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | New Budget Entry | 1. Data should be selected  2. save button should work fine | No irrelevant data is accepted | Perfectly added | P |

**2. View budgets**

****

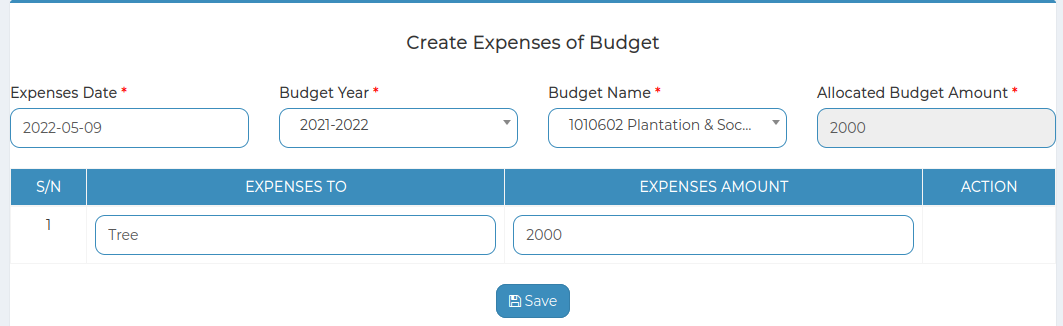
*Positive test case*

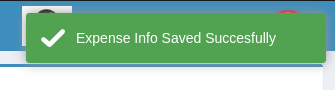
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Budget Information | 1. Display all the information  2. select budget type & year to find specific budget information | Display all the information | No data is missing | P |
| 2 | View Budget Information | Show all budget information in that year | Show the specific budget information | No information is missing | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Budget Information | The budget information should be displayed | Displayed | No error | P |
| 2 | View Budget Information | Information should be displayed for that year | Displayed | No error | P |

**3. Expense Entry**

****

****

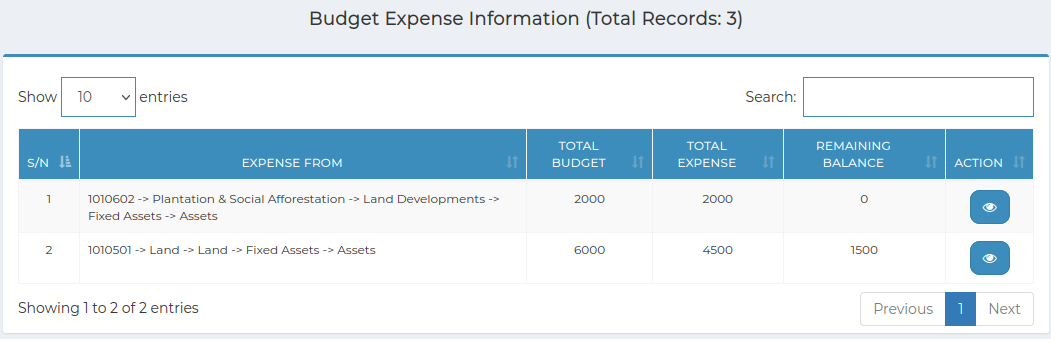
*Positive test case*

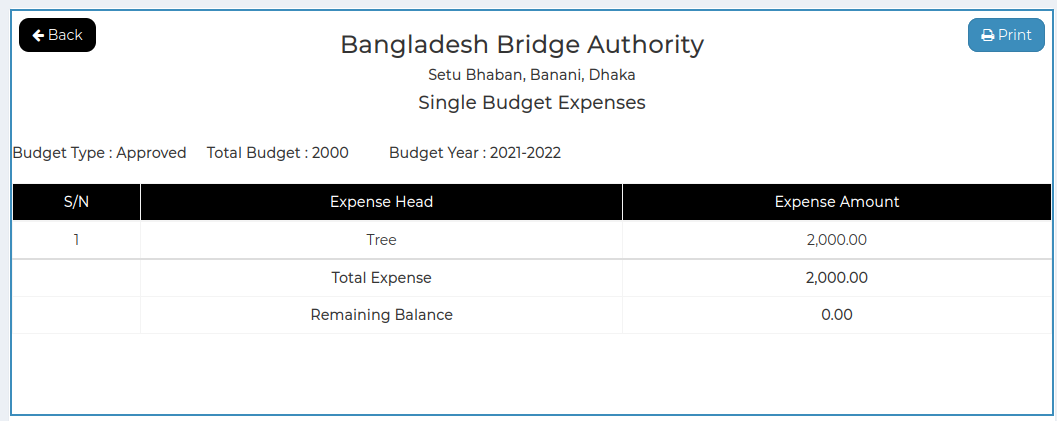
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Create Expenses of Budget | 1. Select expense date, budget year & type  2. Enter expenses to and amount  3. Click save | Data saved successfully | Successfully added in the expense list | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Create Expenses of Budget | Date year amount should be valid | No irrelevant data is inserted | No error | P |

**4. View expenses**





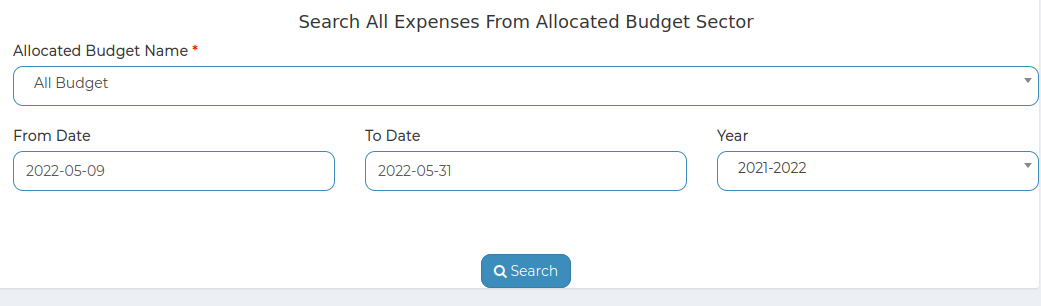
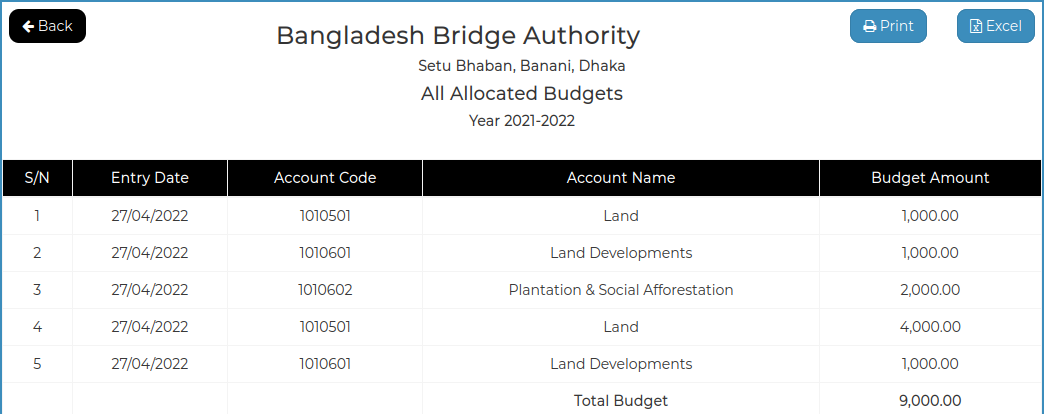
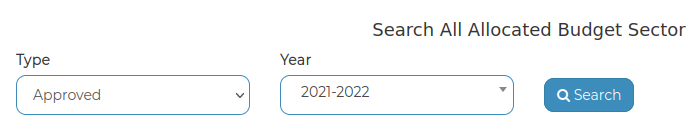
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Budget Expense Information | Display budget expense information | Displayed | No error is occurred | P |
| 2 | View | Click to view specific budget information | Show Individual details information | No error | P |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Budget Expense Information | Everything is working fine | Displayed in respective area of information | No error | P |
| 2 | View | Data is generated to display | Displayed | Works fine | P |

**5. Report**

****

**

*Positive test case*

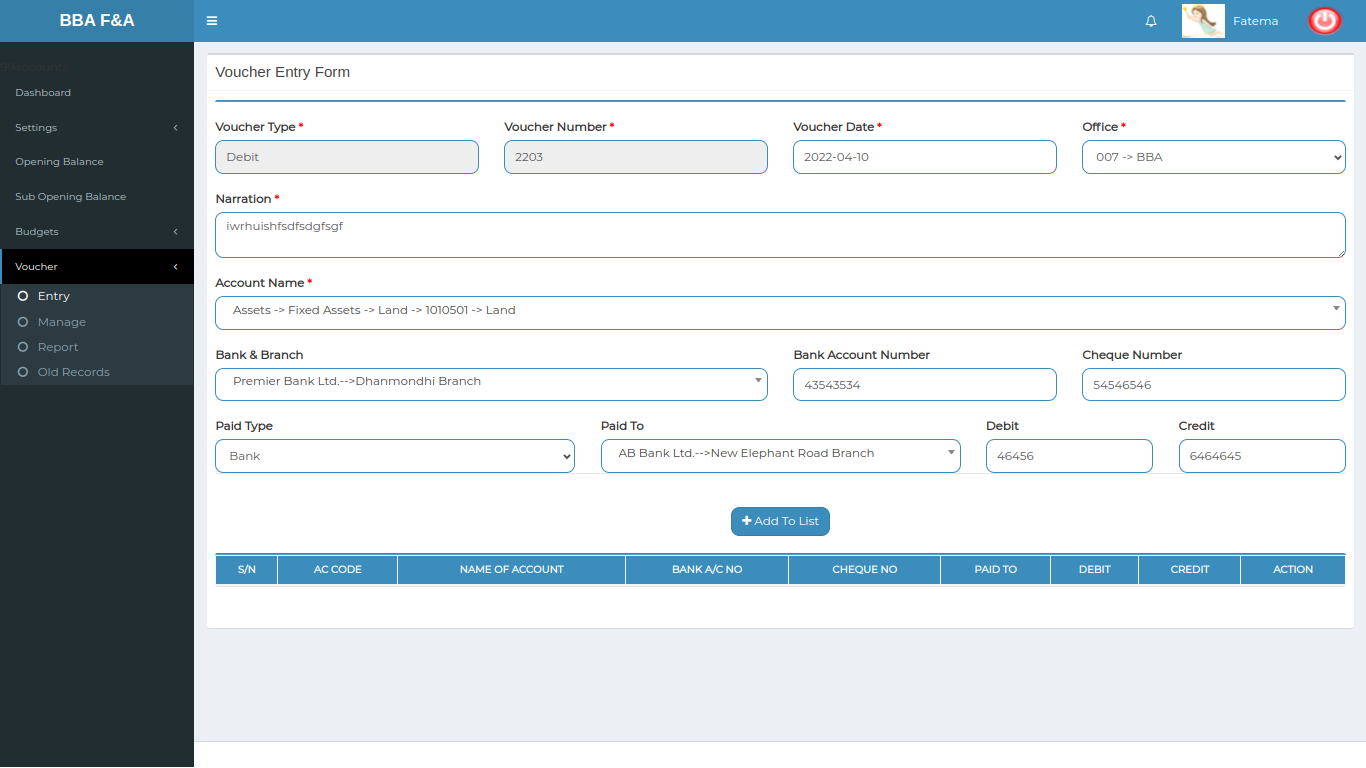
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search Budget Report | 1. Select all the information  2. click search |  |  |  |
| 2 | Report | Show the information |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search Budget Report | All the information should be selected |  |  |  |
| 2 | Report | Report should be generated in document format |  |  |  |

**Test Scenario 6: Voucher**

**1. Entry**



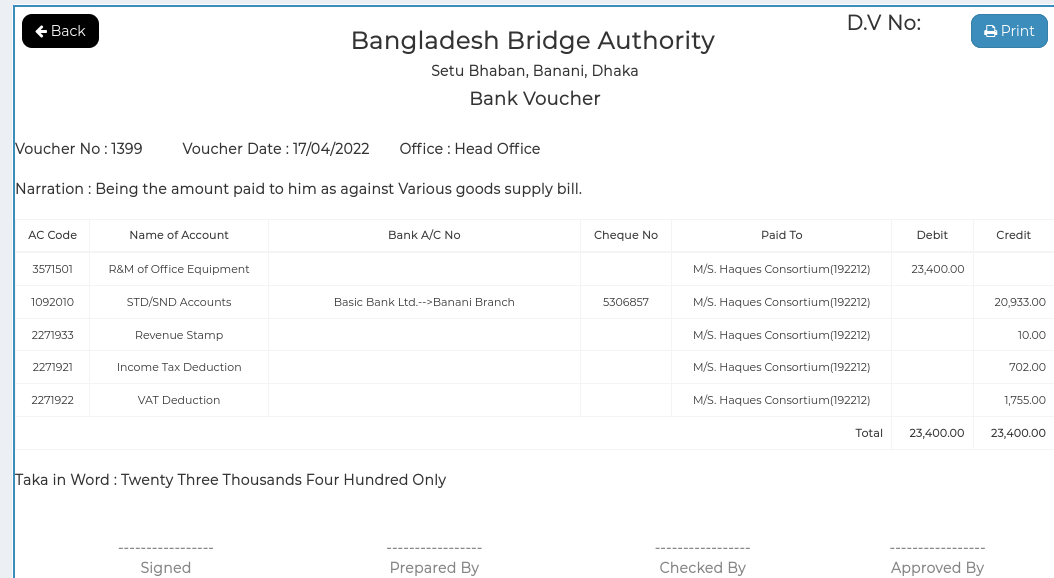
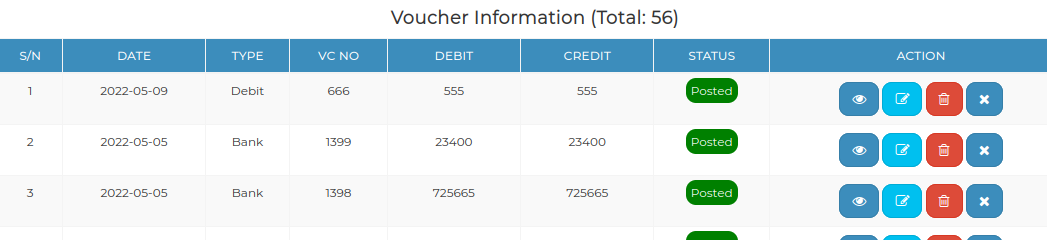
*Positive test case*

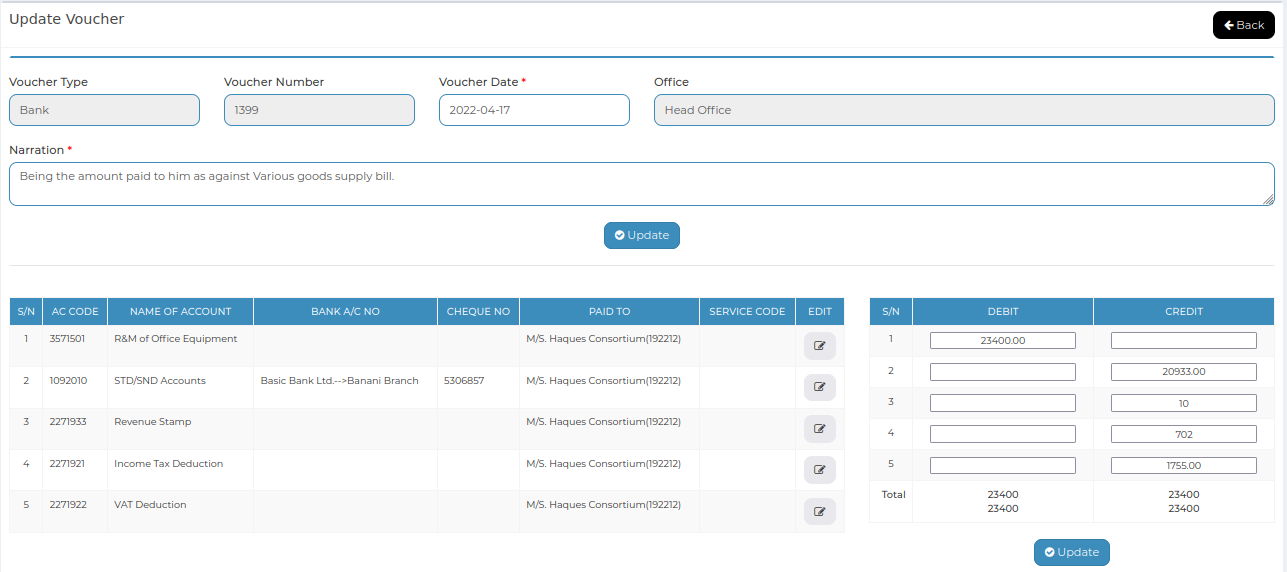
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add new voucher | 1. Select voucher type, voucher date, office name  2. describe narration  3. Select account name, bank & branch name  4. Enter Bank account number, check number  5. select paid type, paid to  6. enter debit or credit amount  7. press add to list |  |  |  |
| 2 | Voucher List |  |  |  |  |

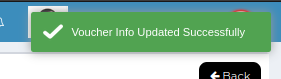
*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Add new voucher | 1. Data should be selected & entered  2. List should be generated |  |  |  |
| 2 | Voucher List |  |  |  |  |

**2. Manage**







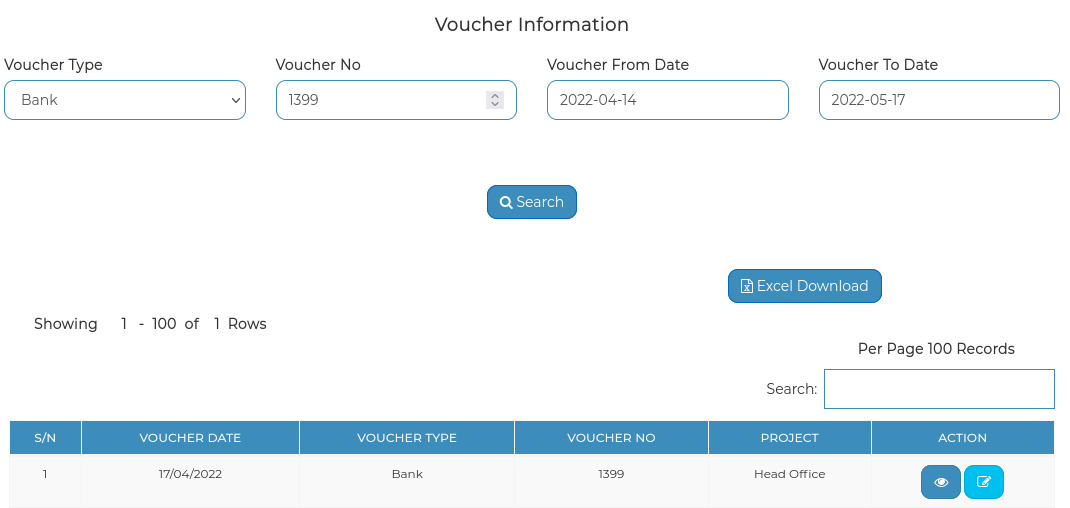
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Voucher Information | 1. Display all voucher information  2. type a voucher name in search bar to find that information |  |  |  |
| 2 | View Details | Click view button to see details & generate a document |  |  |  |
| 3 | Update, delete & post status | Update voucher information like date & narration |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Voucher Information | Action button should work fine |  |  |  |
| 2 | View Details | Report should be generated in document format |  |  |  |
| 3 | Update, delete & post status | Update button should update the data |  |  |  |

**3. Search data**

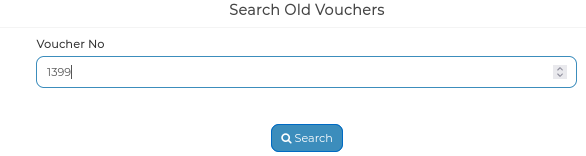


*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search Voucher Report | 1. Select voucher type  2. Enter voucher number  3. Select voucher from date & to date  4. click search |  |  |  |
| 2 | Report | Report is displayed |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search Voucher Report | Data should be selected & search button should work |  |  |  |
| 2 | Report | Report should be generated in document format |  |  |  |

**4. Old Records**

*Positive test case*

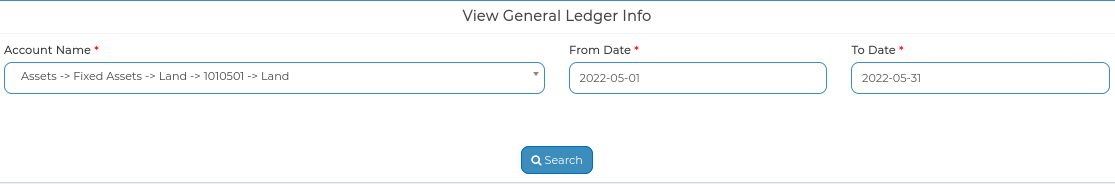
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search Voucher Number | 1. Enter voucher number  2. click search |  |  |  |
| 2 | Old Voucher information | 1. Display Old voucher information list  2. view button use to see details |  |  |  |

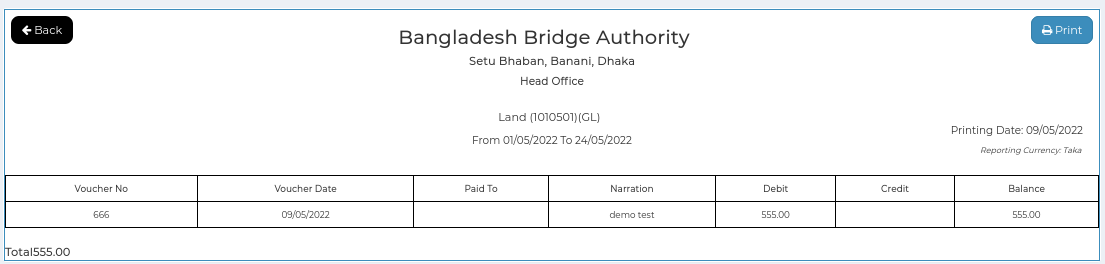
*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search Voucher Number | Voucher number have to be exist |  |  |  |
| 2 | Old Voucher information | View button should work fine |  |  |  |

**Test Scenario 7: Ledgers**

**1. General Ledger**





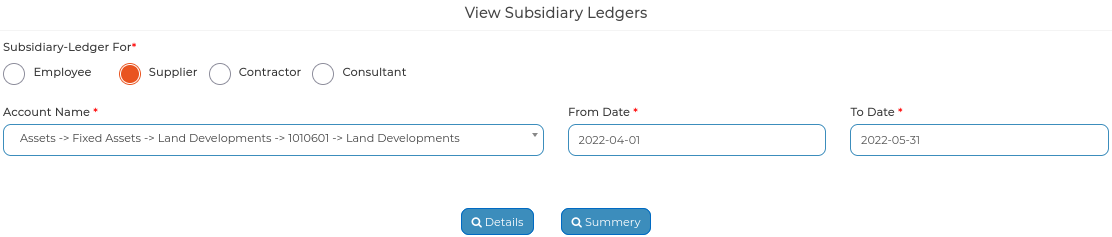
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search |  |  |  |  |
| 2 | Ledger Report |  |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search |  |  |  |  |
| 2 | Ledger Report |  |  |  |  |

**2. Subsidiary Ledger**







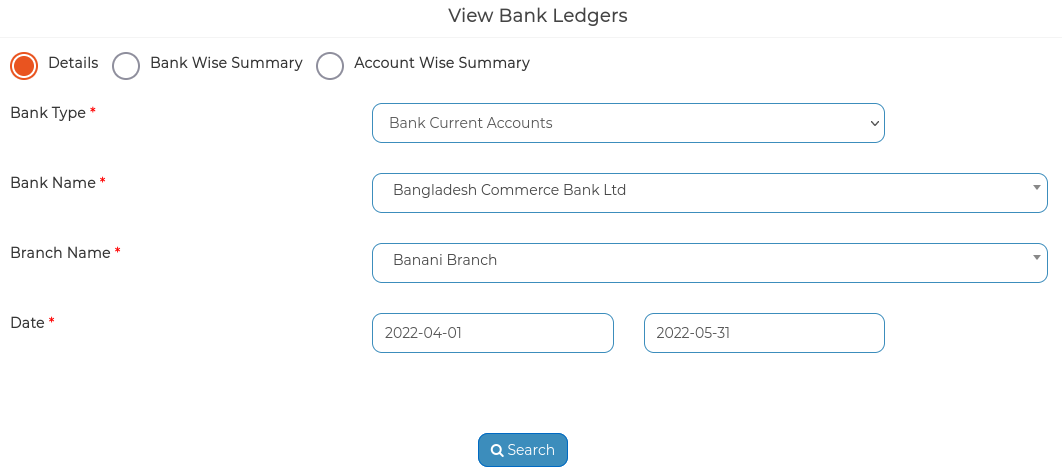
*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search |  |  |  |  |
| 2 | Details |  |  |  |  |
| 3 | Summary |  |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Search |  |  |  |  |
| 2 | Details |  |  |  |  |
| 3 | Summary |  |  |  |  |

**3. Bank Ledger**



*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Details |  |  |  |  |
| 2 | Bank wise Summary |  |  |  |  |
| 3 | Account wise Summary |  |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 | Details |  |  |  |  |
| 2 | Bank wise Summary |  |  |  |  |
| 3 | Account wise Summary |  |  |  |  |

**Test Scenario 7: Bank Reconcillation**

**1. View statement**

*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |

**2. Generate**

*Positive test case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |

*Negative Test Case*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Case ID | Title | Description | Expected Result | Actual Result | Pass/Fail |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |