SOFTWARE REQUIREMENT SPECIFICATION OF

SIMS PROJECT PAYMENT SYSTEM

**Table of Contents**

1.Introduction

1.1Purpose

1.2 Scope

1.3 Definitions, Acronyms, and Abbreviations

2.Overall Description

2.1 Application Perspective

2.2 Application Features

2.3 Operating Environment

3.Specific Requirements

3.1 Functional Requirements

3.1.1 User Authentication

3.1.2 Data Input and Management

3.1.3 Automated Calculations

3.1.4 Relationship Management

3.1.5 Report Generation

3.1.6 Adaptability and Intelligence

3.1.7 Dropdown Menus and Data Validation

3.1.8 Foreign Key Constraints

3.2 Non-Functional Requirements

3.2.1 User-Friendliness

3.2.2 Scalability

3.2.3 Security

3.2.4 Performance

4.External Interface Requirements

4.1 User Interfaces

4.2 Hardware Interfaces

4.3 Software Interfaces

4.4 Communication Interfaces

5.Database Design

5.1 Table Definitions and Structures

5.2 Relationships Between Tables

* 1. Data Integrity Constraints

5.4 Keys

6.Form Structure

6.1. Accessing Forms

6.2. Form Layout

6.3. Data Entry

6.4. Data Validation

6.5. Form Submission

6.6. Editing and Deleting Entries

6.7. Form Functionality

7.Design Layout

7.1 User Interface Design

7.1.1 Wireframes

7.1.2 Mock ups

7.2 Layouts and Report Generation

7.3 Form Layouts

8.Appendices

8.1 Glossary

8.2 Change Log

8.3 References

**1.Introduction**

**1.1 Purpose:**

This document outlines the requirements and specifications for the development of a user-friendly **Sim** **Project Payment Application**. The purpose of this application is to simplify the process of handling data by providing intuitive forms for data entry, thereby replacing manual entry traditionally done in Excel sheets. By transitioning to a digital platform with streamlined data input mechanisms, the application aims to enhance efficiency, accuracy, and accessibility in managing project-related financial transactions.

**1.2 Scope:**

This project involve the design, development, and implementation of a desktop-based payment application tailored to the specific needs of our organization. The application will feature user-friendly forms for entering and managing data related to customers, projects, invoices, and categories. These forms will replace the manual entry process typically performed in Excel sheets, offering a more efficient and organized approach to data management. The scope of the project includes modules for customer management, project tracking, invoice generation, and report generation, all aimed at simplifying the handling of data and improving overall productivity.

**1.3 Definitions, Acronyms, and Abbreviations:**

* UI: User Interface
* DBMS: Database Management System
* API: Application Programming Interface

**2.Overall Description**

**2.1 Application Perspective:**

The project payment application operates as an independent desktop tool aimed at simplifying project financial management. It provides user-friendly interfaces for data entry and report generation, facilitating efficient handling of project-related transactions.

**2.2 Application Features:**

The key features of the project payment application include:

* Forms for entering customer, project, invoice, and category data.
* Automated calculations for taxes, deductions, and invoice amounts.
* Report generation functionalities for project lists, invoice registers, and customer details.
* Secure user authentication.

**2.3 Operating Environment:**

The application is designed to run on standard desktop computers with major operating systems such as Windows, MAC OS, and Linux. It requires basic hardware specifications to ensure smooth operation, including sufficient processing power and memory.

**3. Specific Requirements**

**3.1 Functional Requirements**

**3.1.1 User Authentication:**

* Users must be able to register, log in, and manage their accounts securely.
* The authentication process should verify user credentials and grant access to authorized users only.

**3.1.2 Data Input and Management:**

* The application must provide forms for entering and managing data related to customers, projects, invoices, and categories.
* Users should be able to input, update, and delete data as necessary.

**3.1.3 Automated Calculations:**

* The system must perform automated calculations for delay and other financial metrics based on the input data.
* Calculations should be accurate and consistent across all transactions.

**3.1.4 Relationship Management:**

* The application should establish and maintain relationships between customer, project, and invoice data to ensure data integrity.
* Users should be able to view and navigate relationships between different data entities.

**3.1.5 Report Generation:**

* The system must have the ability to generate reports such as project lists, invoice registers, and customer details based on the entered data.
* Reports should be customizable and exportable in various formats for easy sharing and analysis.

**3.1.6 Adaptability:**

* The application should be adaptable to user preferences and evolving business requirements.

**3.1.7 Dropdown Menus and Data Validation:**

* Dropdown menus should be provided for selecting predefined options to ensure consistency and accuracy in data entry.
* Data validation mechanisms must be implemented to prevent errors and ensure data integrity.

**3.1.8 Foreign Key Constraints:**

* Foreign key constraints should be enforced to maintain referential integrity between related data entities such as projects and customers.
* Relationships between tables should be properly defined and enforced at the database level.

**3.2 Non-Functional Requirements**

**3.2.1 User-Friendliness:**

The application must have a user-friendly interface with clear navigation and instructions to facilitate ease of use for users of all levels.

**3.2.2 Scalability:**

The system should be designed to accommodate growth in data volume and user load without compromising performance or functionality.

**3.2.3 Security:**

Robust security measures must be implemented to protect user data, including encryption of sensitive information, secure authentication mechanisms, and role-based access control.

**3.2.4 Performance:**

The application should perform efficiently, with quick response times for data retrieval, calculations, and report generation, even under heavy usage.

**4. External Interface Requirements**

**4.1 User Interfaces:**

The user interface of the project payment application should be user-friendly, featuring the following components:

* Input Forms: User-friendly forms for entering data related to customers, projects, invoices, and categories.
* Navigation Menu: A clear and organized menu system for easy navigation between different sections of the application.
* Report Generator: A feature-rich interface for generating customizable reports based on the entered data.
* Authentication Module: Secure login and registration forms for user authentication and account management.

**4.2 Hardware Interfaces:**

The project payment application requires standard hardware components to operate effectively, including:

* Desktop or Laptop Computer: The application runs on desktop or laptop computers with standard specifications, including sufficient processing power and memory.
* Display Screen: A monitor or display screen is necessary for users to interact with the application's user interface.
* Input Devices: Users interact with the application using input devices such as a keyboard and mouse.

**4.3 Software Interfaces:**

The application will interact with the following software components:

**Database Management System (DBMS):**

* MySQL will serve as the backend database management system (DBMS) for storing and retrieving data efficiently.
* Java Database Connectivity (JDBC) API will be used to establish a connection between the Java application and the MySQL database, facilitating data retrieval, insertion, and modification operations.

**Java Development Environment:**

* Java programming language will be used for developing the application's frontend and backend logic.
* JDBC API will enable seamless integration between the Java application and the MySQL database, allowing for efficient data manipulation and retrieval.
* Integrated Development Environments (IDEs) such as IntelliJ IDEA or Eclipse will be utilized for writing, compiling, and debugging Java code, providing a streamlined development environment for the project.

**4.4 Communication Interfaces:**

Communication interfaces facilitate communication between different components of the application, including:

* Local Communication: Communication between the application's frontend and backend components within the same local environment.
* Remote Communication: Communication between the application and external systems or services, such as cloud-based storage or third-party APIs.
* Protocol: Communication protocols such as HTTP or TCP/IP may be used for data exchange between different components of the application.

**5. Database Design**

**5.1 Table Definitions and Structures:**

* **Category Table:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Input Format** | **Constrain** |
| Category ID | Number | 10 |
| Category name | Text | 50 |

Description: category\_id (Primary Key).

* **Customer Table:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Input Format** | **Constrain** |
| CUSTOMER ID | Text | 10 |
| Customer Name | Text | 10 |
| Category ID | Select | Drop Down |
| Address | Text | 50 |
| Country | Text | 10 |
| Type | Text | Customer, Partner, Agent |
| Contact details | Text | 50 |

Description: customer\_id (Primary Key: NOT NULL and UNIQUE). Customer\_id can be mapped with several project\_ids. If required category is not available then it should be added in category table then it will be appeared in the drop down in customer table.

**Project Table:**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Input Format** | **Constrain** |
| Project ID | Text | 20 |
| Description | Text | 50 |
| Customer ID | Text | 10 |
| End client detail | Text | 50 |
| End client country | Text | 10 |
| PO No | Text | 20 |
| PO Date | Date | YYYY-MM-DD |
| PO initial Value | Float | 10.2 |
| GST % | Float | 0 to 100 |
| PO Rev 1 value | Float | 10.2 |
| PO Rev 1 date | Date | YYYY-MM-DD |
| PO Rev 2 value | Float | 10.2 |
| PO Rev 2 date | Date | YYYY-MM-DD |
| PO Rev 3 value | Float | 10.2 |
| PO Rev 3 date | Date | YYYY-MM-DD |
| PO Rev 4 value | Float | 10.2 |
| PO Rev 4 date | Date | YYYY-MM-DD |
| Currency | Text | 5 |
| Payment terms | Text | 50 |
| Project Lead | Text | 20 |
| Expected completion date | Date | YYYY-MM-DD |
| Actual completion date | Date | YYYY-MM-DD |
| % completion | Float | 0 to 100 |

Description: project\_id (Primary Key). New project details can be entered for existing customer.

**Invoice Table:Attributes in Form**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Input Format** | **Constrain** |
| Project ID | Text | 20 |
| Invoive no | Text | 20 |
| Invoice date | Date | YYYY-MM-DD |
| Description | Fetched | 50 |
| Invoice type | Select | Milestone/Expense |
| Inv amount | Float | 12.2 |
| Inv amount in INR | Float | 12.2 |
| GST amount | Float | 12.2 |
| TDS dedected | Float | 12.2 |
| Retention amount | Float | 12.2 |
| Amount received | Float | 12.2 |
| Amount received in INR | Float | 12.2 |
| Received date | Date | YYYY-MM-DD |
| FIRC details | Text | 50 |

Description: invoice\_number (Primary Key). Invoice can be raised for existing projects only.The above attributes is to be entered in form. Some of them are drop down and fetch automatically based on the available details. With this the remaining attributes of the actual table will be calculate.If project\_id doesn,t exist in project table then invoice cannot be raised so it displays that projected is not available.

**Actual table**

|  |  |  |
| --- | --- | --- |
| **Attributes** | **Data type** | **Constrain** |
| Customer ID | String | 20 |
| Project ID | String | 20 |
| Po no | String | 20 |
| Invoive no | String | 20 |
| Invoice date | Date | YYYY-MM-DD |
| Description | String | 50 |
| Invoice type | Select | Milestone/Expense |
| Inv amount in Euro | Float | 12.2 |
| Inv amount in USD | Float | 12.2 |
| Inv amount in INR | Float | 12.2 |
| GST amount | Float | 12.2 |
| TDS dedected | Float | 12.2 |
| Retention amount | Float | 12.2 |
| Amount received | Float | 12.2 |
| Amount received in INR | Float | 12.2 |
| Received date | Date | Select |
| FIRC details | String | 50 |
| Delay | Integer | Calculated |

Description: Based on project\_id entered, customer\_id,description,po no will be fetched. Delay=received date-Invoice date. The payment entries are based on currency which should be identified through project\_id.

**5.2 Relationships Between Tables:**

* The Customer Table is related to the Project Table through the customer\_id foreign key.
* The Project Table is related to the Invoice Table through the project\_id foreign key.
* The Invoice Table is related to the Customer Table and Project Table through the customer\_id and project\_id foreign keys respectively.
* The Customer Table is related to the Category Table through the category\_id foreign key.

**5.3 Data Integrity Constraints:**

* Foreign key constraints are enforced to maintain referential integrity between related tables.
* Data validation constraints ensure that only valid data is entered into the database, preventing data inconsistencies and errors.

**5.4 Keys:**

* Primary keys are defined for each table to uniquely identify records.
* Foreign keys establish relationships between tables and ensure data integrity.

**6.Form Structure:**

The form structure plays a crucial role in the usability and effectiveness of the project payment system application. This provides detailed instructions on how to navigate and utilize the various forms within the application.

**6.1. Accessing Forms:**

* To access forms for data entry and management, navigate to the corresponding section from the main menu.
* For example, to enter customer data, click on the "Add Customers" tab in the customer session, which will lead you to the customer form.

**6.2. Form Layout:**

* Each form follows a consistent layout with clearly labelled input fields and buttons.
* Input fields are organized logically, with related fields grouped together.
* Dropdown menus are provided for selecting predefined options where applicable.
* Required fields are marked with an asterisk (\*) to indicate mandatory entry.

**6.3. Data Entry:**

* To enter data into the form, click on the respective input field and type in the required information.
* Use dropdown menus to select predefined options for fields such as country, type, and category.
* Ensure accuracy and completeness of data entry to avoid errors in calculations and reports.

**6.4. Data Validation:**

* Data validation mechanisms are in place to ensure that entered data meets specified criteria.
* If invalid data is entered, error messages will be displayed, prompting users to correct the entries.
* Pay attention to error messages and make necessary corrections before submitting the form.

**6.5. Form Submission:**

* Once all required information is entered and validated, click on the "Submit" or "Save" button to save the data.
* A confirmation message will be displayed upon successful submission, indicating that the data has been saved.

**6.6. Editing and Deleting Entries:**

* To edit or delete existing entries, navigate to the respective section and locate the entry you wish to modify.
* Click on the "Edit" or "Delete" button next to the entry to make changes or remove it from the database.

**6.7. Form Functionality:**

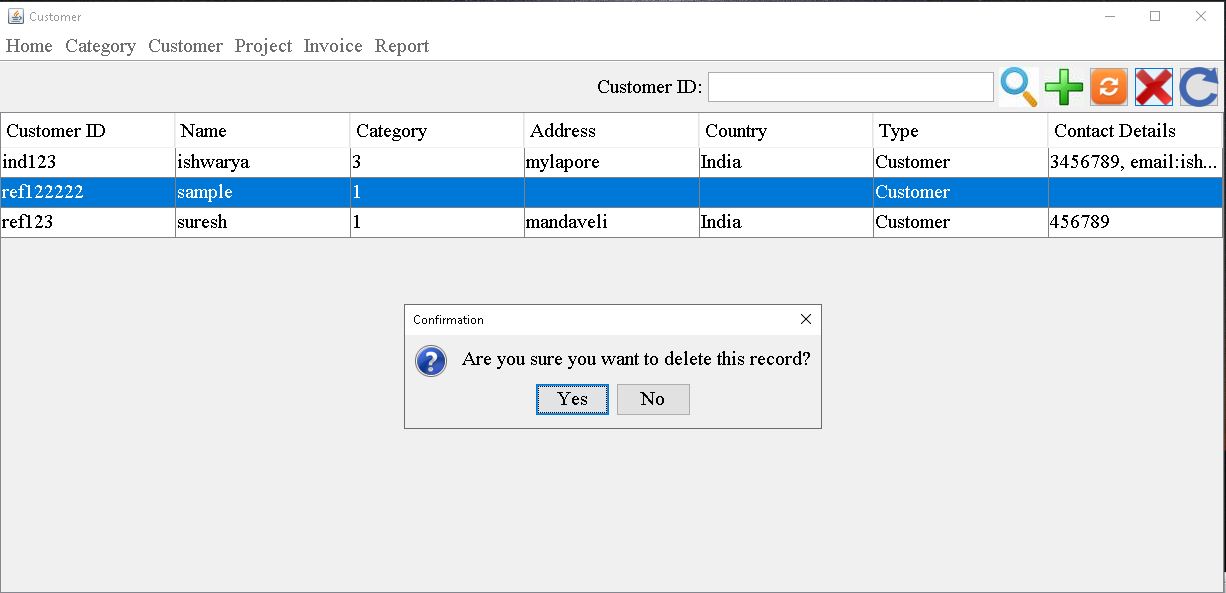
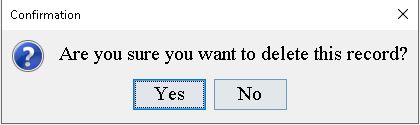
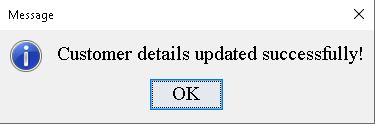
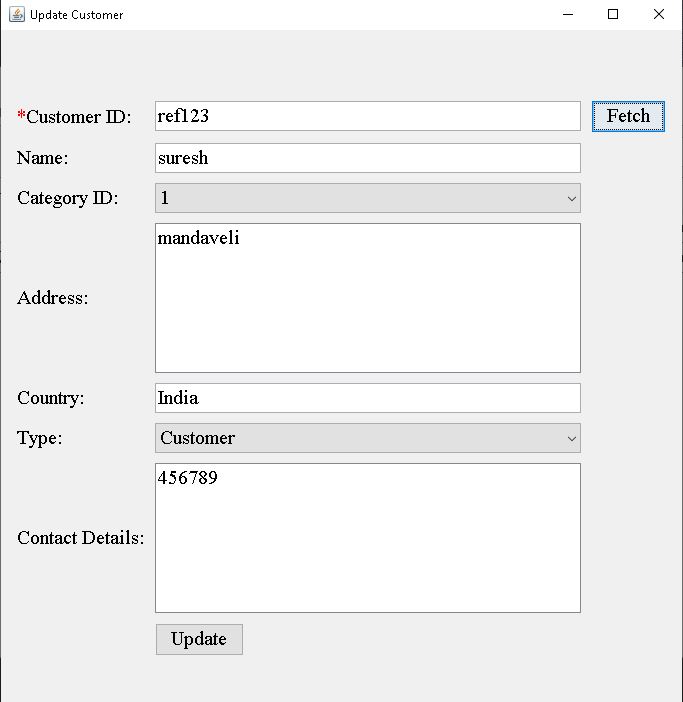
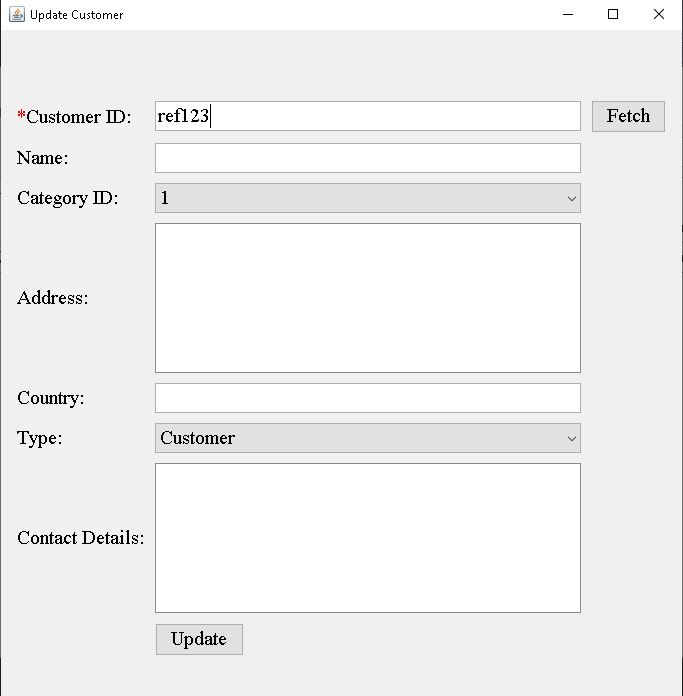
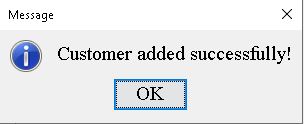
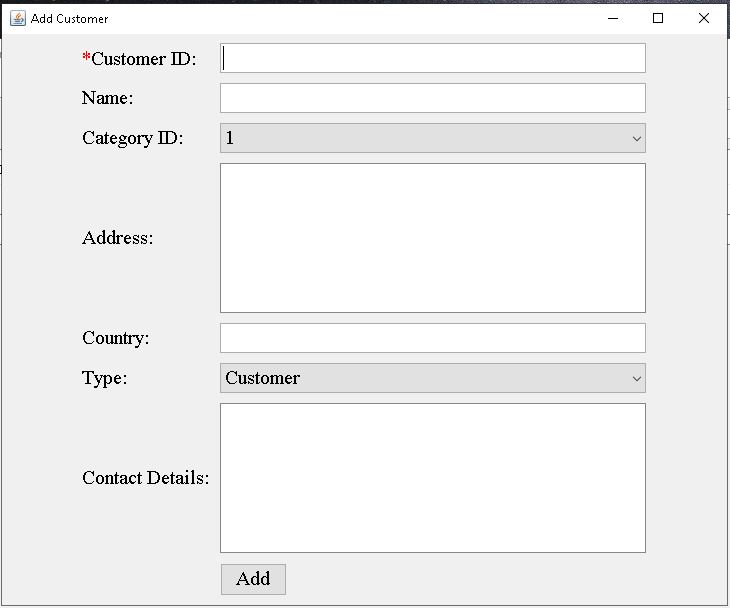
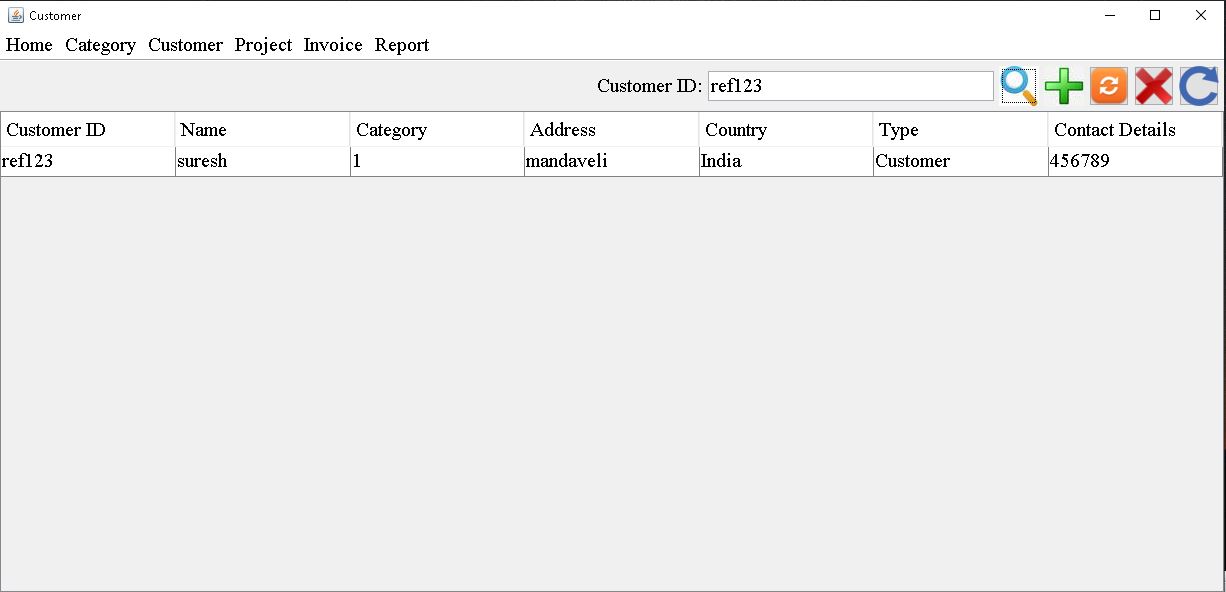
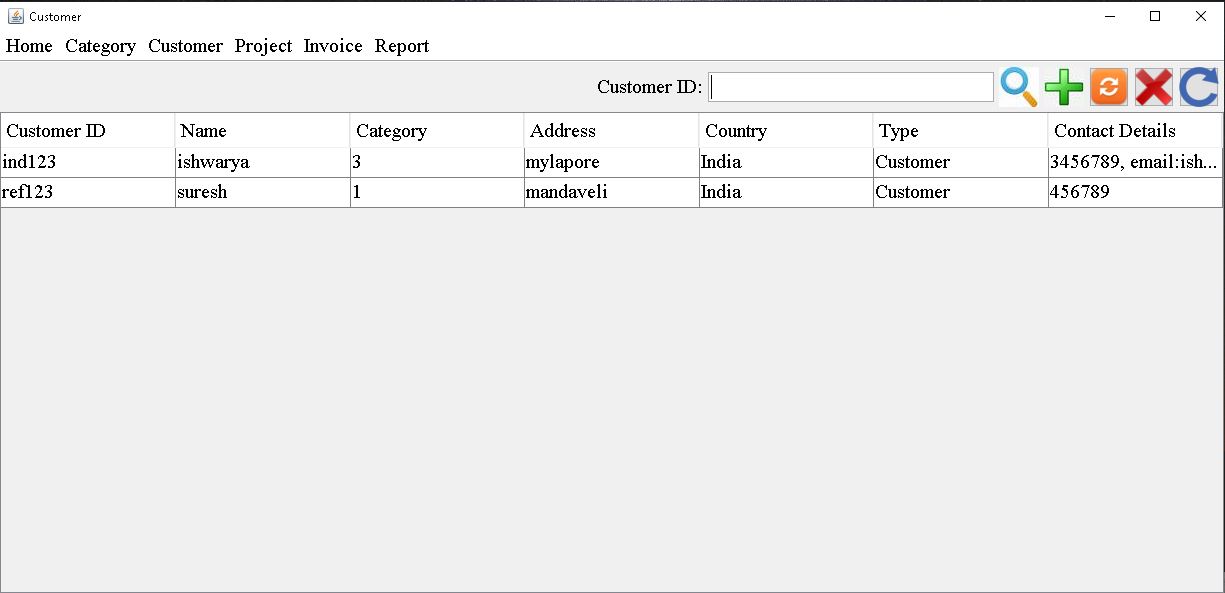
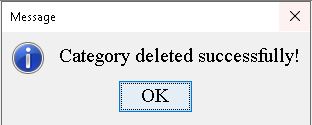
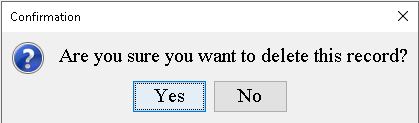
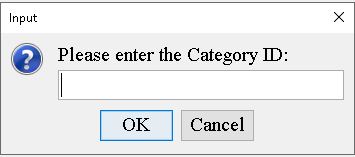
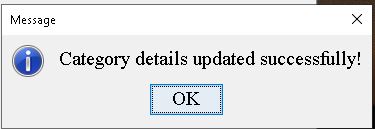
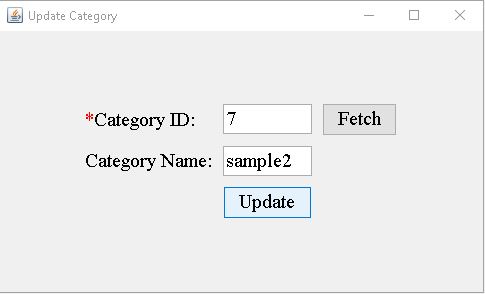
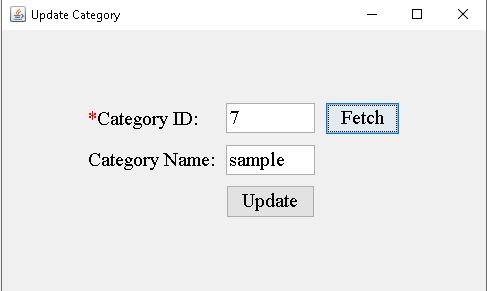
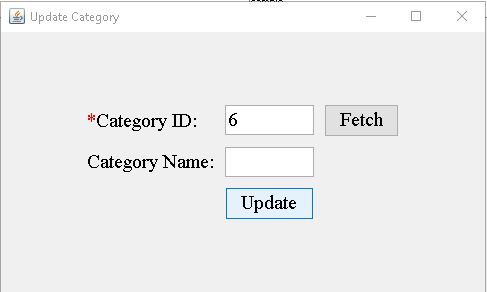
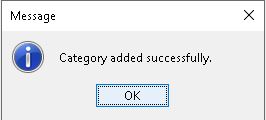
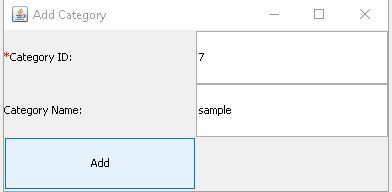
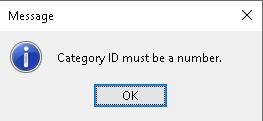
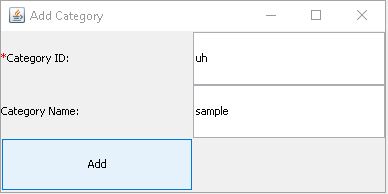
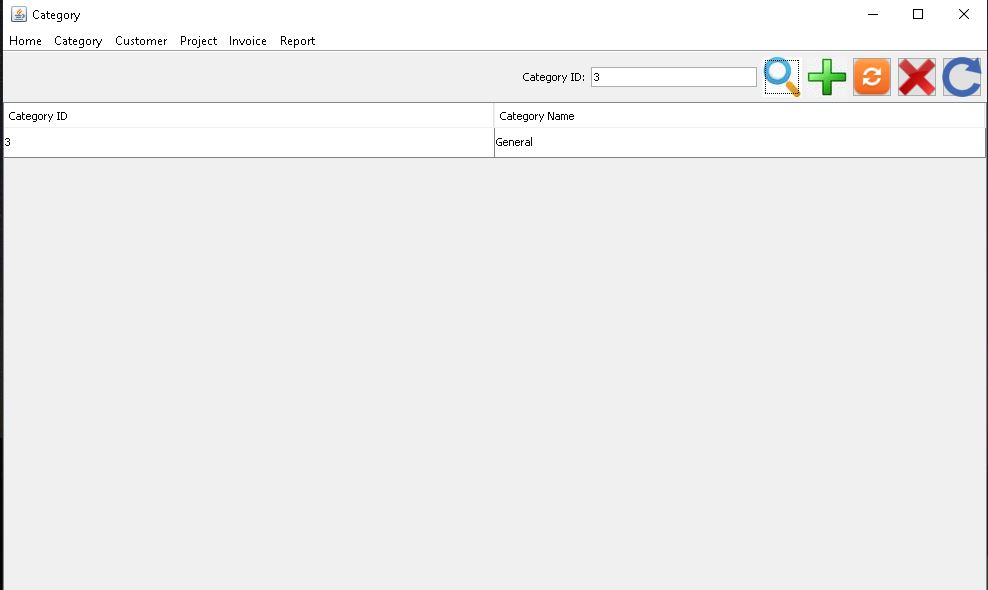
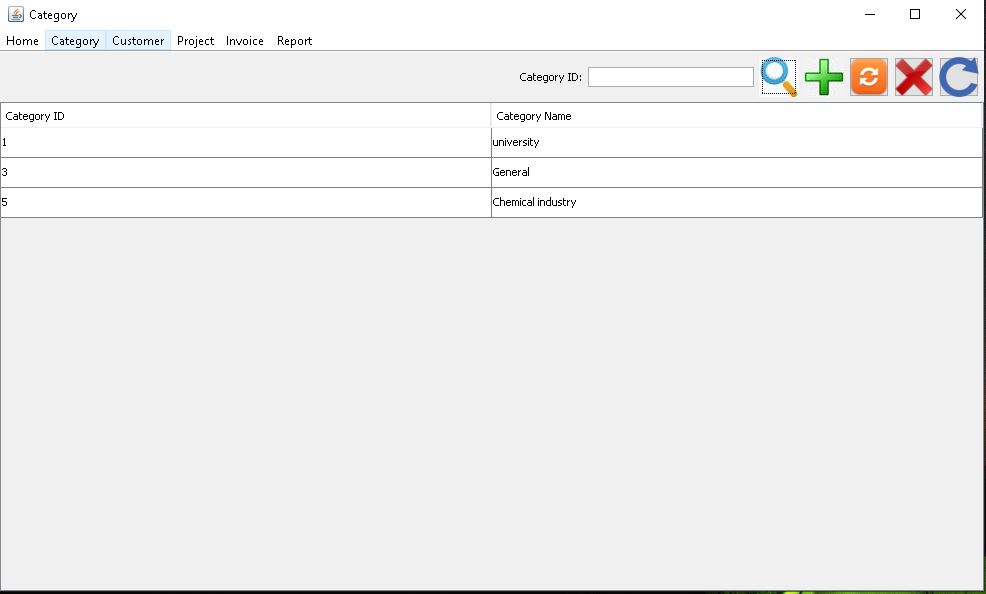
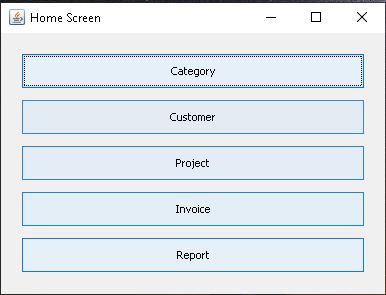
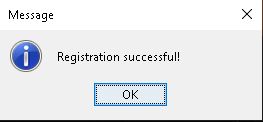
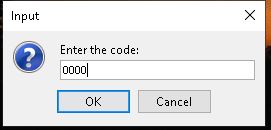
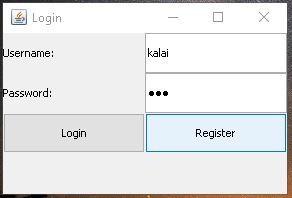
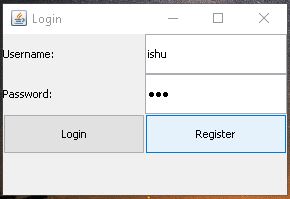
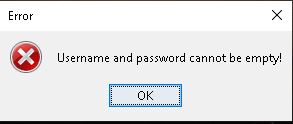
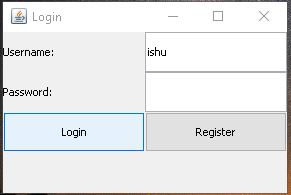
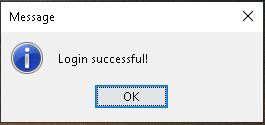
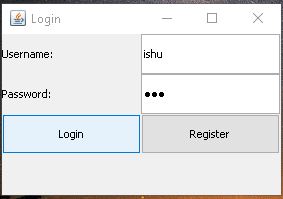
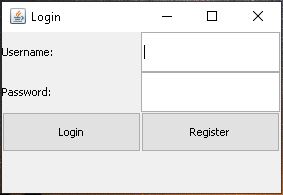
* Each form may include additional functionality such as:
* Automatic calculations: Fields that are automatically populated based on the entered data.
* Dynamic field updates: Changes in one field may trigger updates in related fields.
* Data filtering and sorting: Options to filter and sort data for easier navigation and analysis.

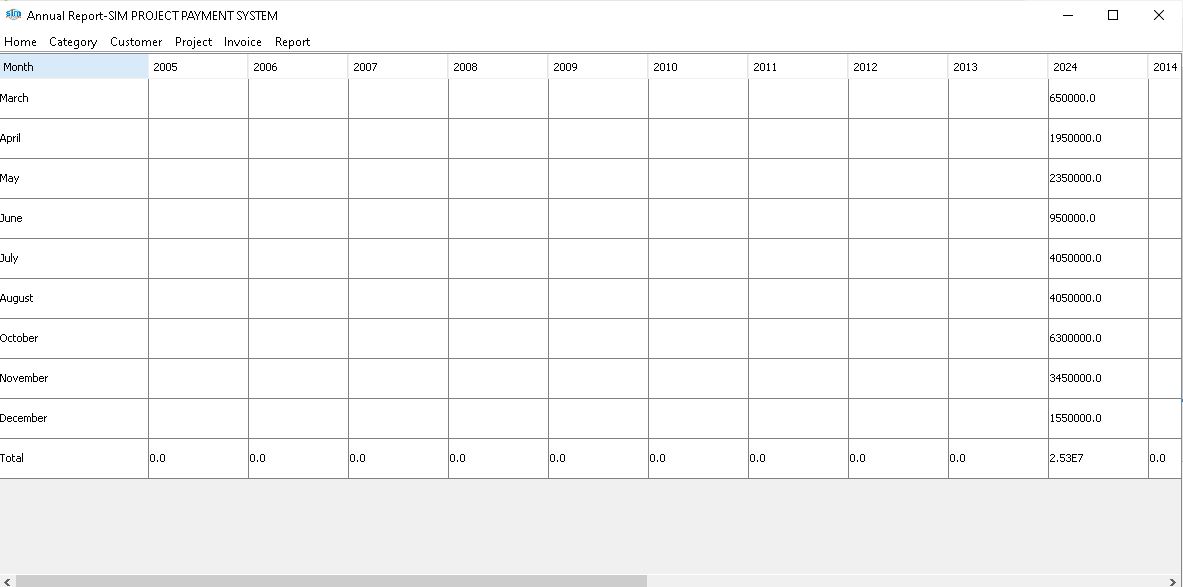
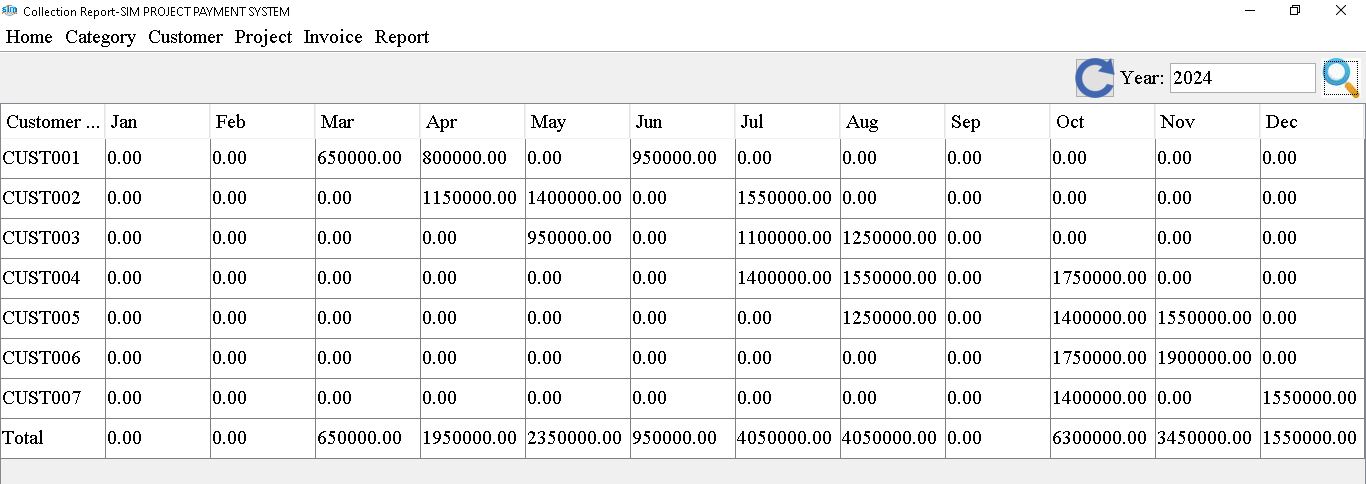
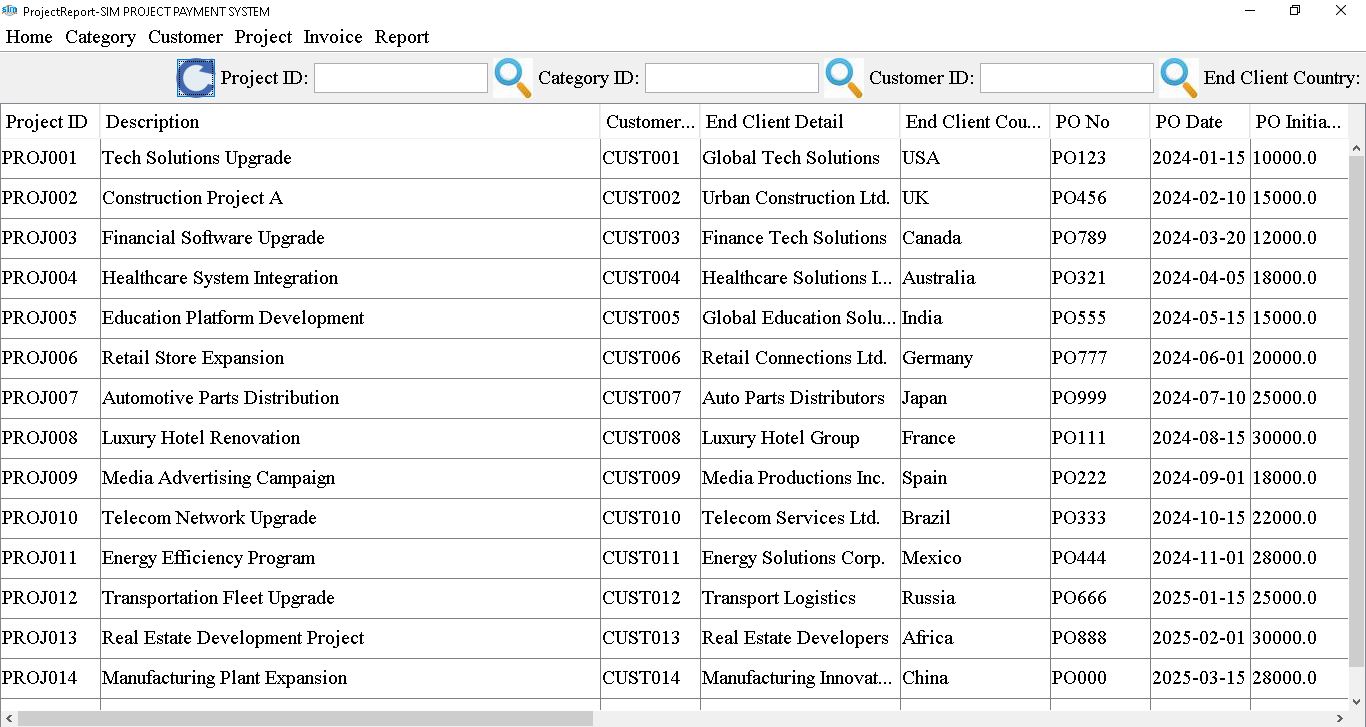
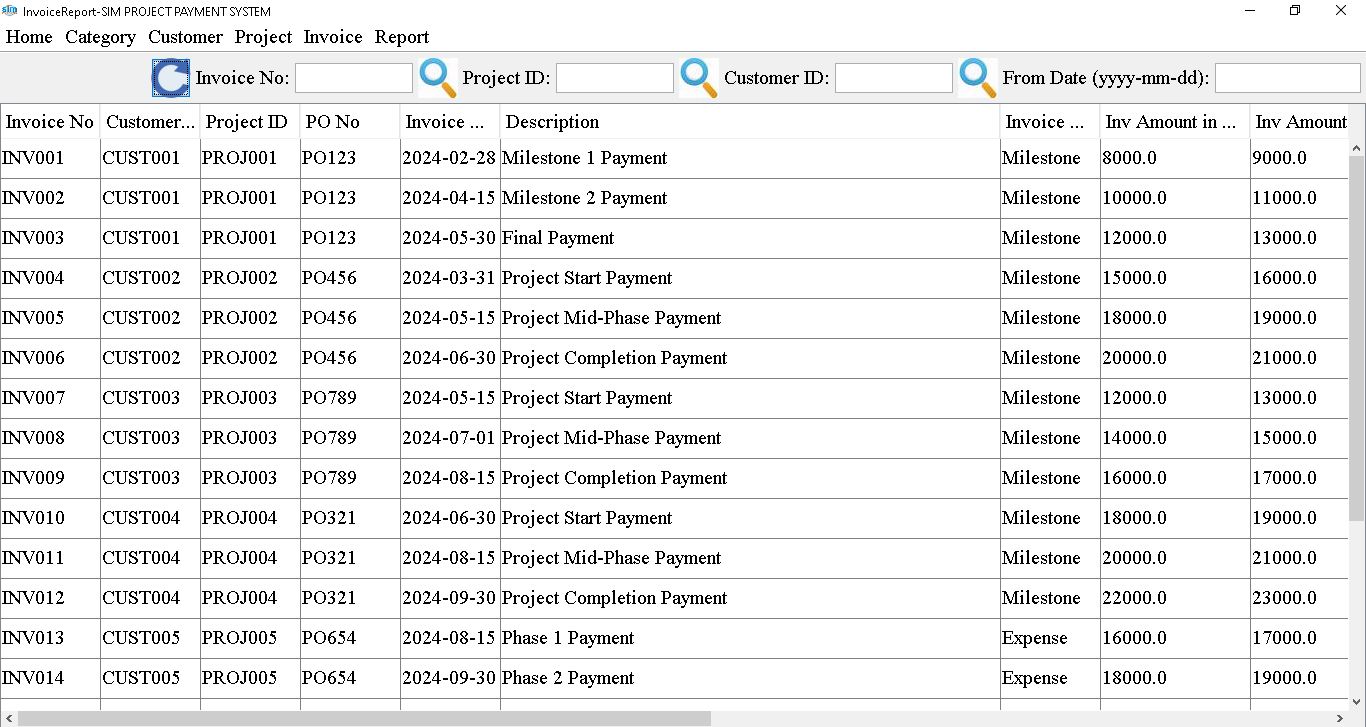
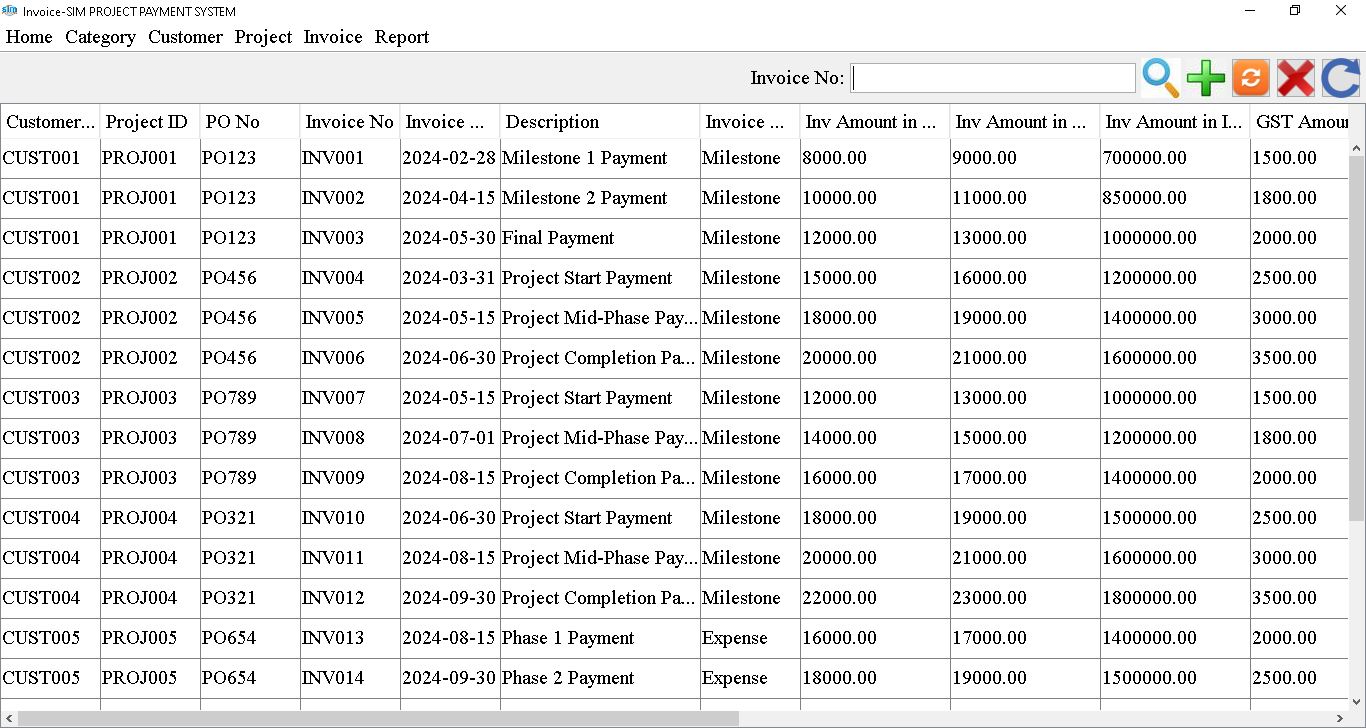
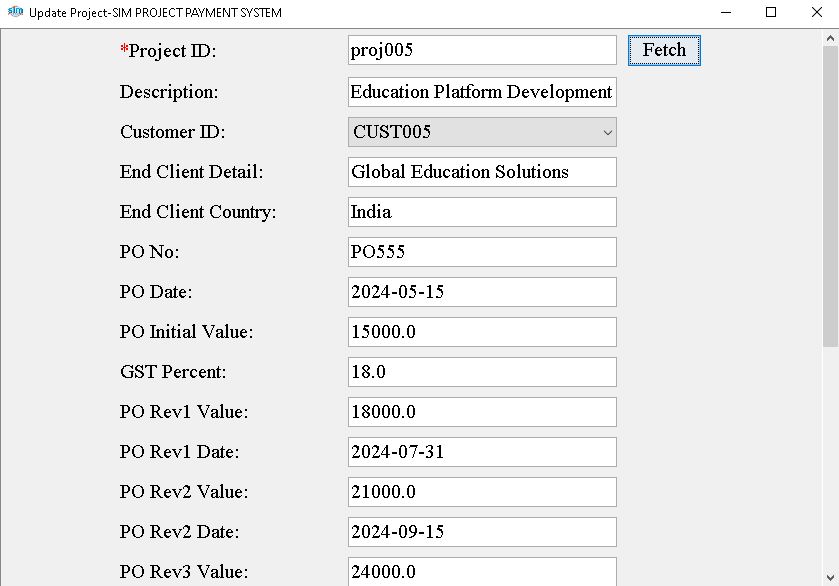
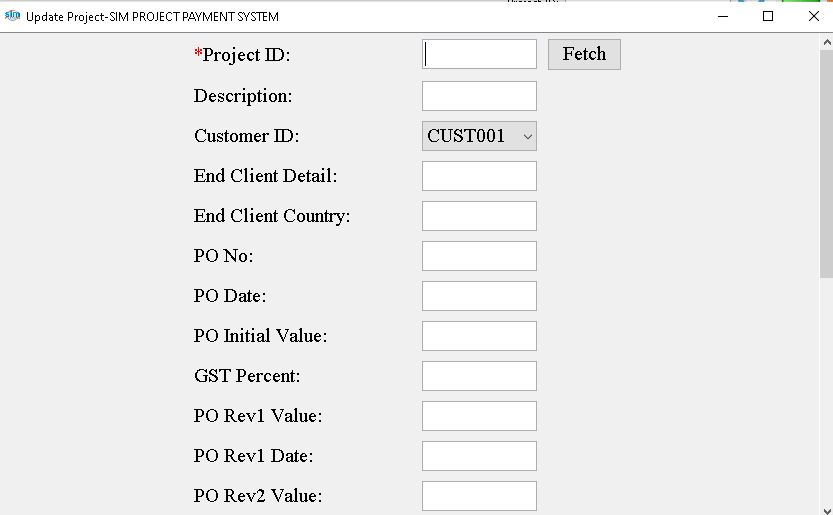
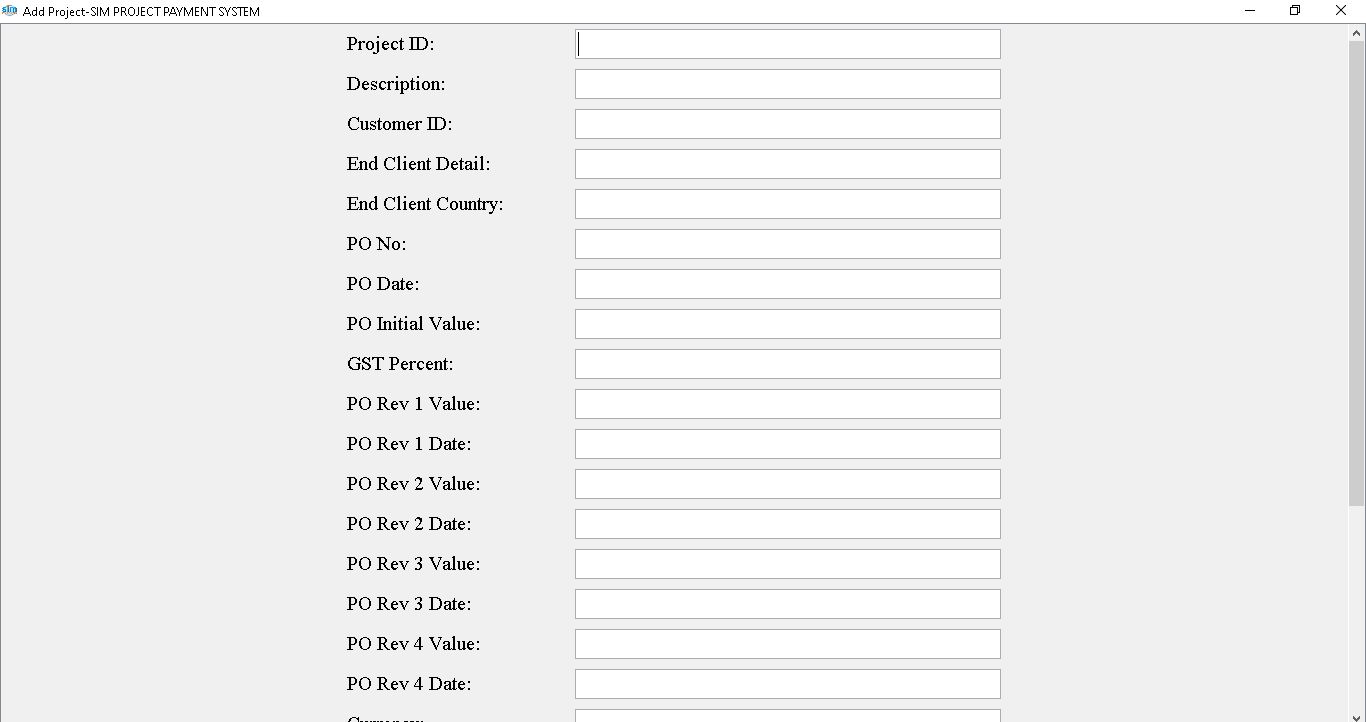
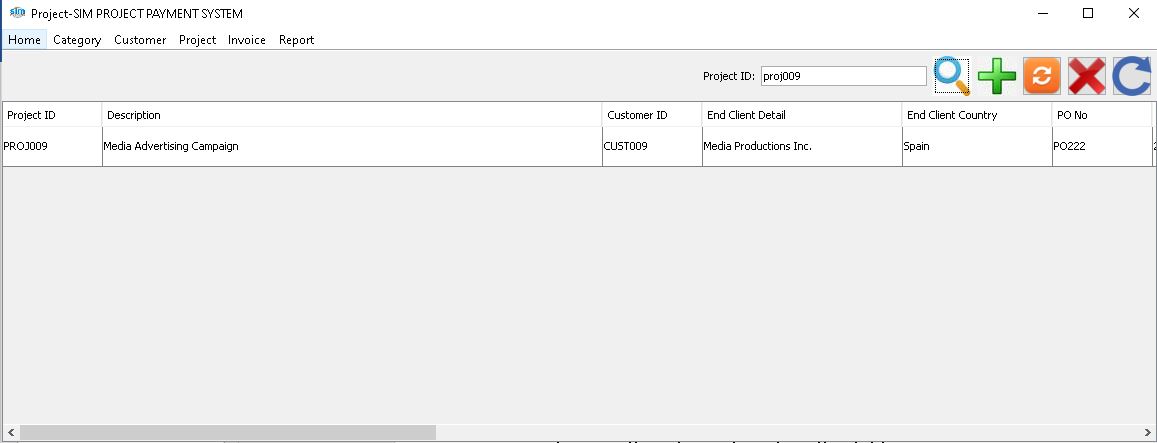
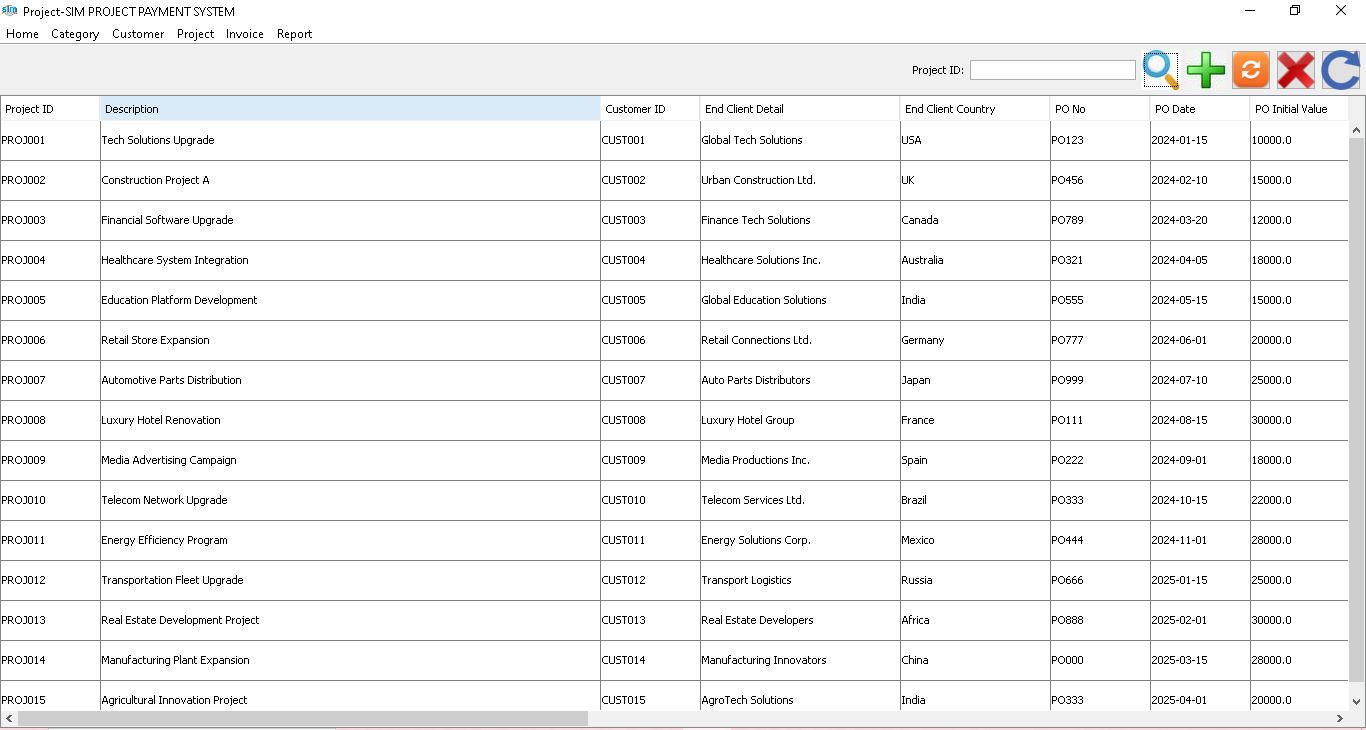
**8. Design Layout**

**7.1 User Interface Design:**

**7.1.1 Wireframes:**

**7.1.2 Mock ups:**

****

****

**9. Appendices**

**9.1 Glossary:**

* Customer ID: Unique identifier for each customer in the system.
* Project ID: Unique identifier for each project in the system.
* Invoice Number: Unique identifier for each invoice in the system.
* GST: Goods and Services Tax, a tax levied on the supply of goods and services.
* TDS: Tax Deducted at Source, a tax deducted from income at the time of payment.
* Foreign Key: A column or set of columns in a database table that references the primary key of another table.
* Data Validation: The process of ensuring that data entered into a database meets specified criteria for accuracy and completeness.
* Wireframes: Basic visual representations of a user interface layout.
* Mock ups: High-fidelity visual representations of a user interface design.
* Dropdown Menus: A graphical control element used to select one item from a list of options.

**9.2 Change Log:**

- Version 1.0: Initial release of the project payment system document.

**9.3 References:**

- No references available for this project as it is a standalone application developed internally by the organization.