Builder Portfolio Management System

A. Project Overview

- **Purpose:**
  - Manage construction projects efficiently
  - Assign clients, managers to projects
  - Track project timelines, budgets, and expenses
  - Maintain project-related documents
  - Provide role-based access for Builders, Project Managers, and Clients
- **Technologies Used:**
  - **Java (Core and JDBC):** For implementing business logic and database interaction
  - **PostgreSQL:** For persistent storage of users, projects, and documents
  - **JUnit :** For unit testing service layer with DAO mocking
  - **Maven:** For dependency management and build automation

B. User Roles and Responsibilities

1. Builder

- Register on the system and log in
- Add new projects with name, description, and client assignment
- Assign a Project Manager to a project
- View all projects they have created
- Access project details, timelines, budgets, and documents
- Ensure proper project management by coordinating with Project Managers

2. Project Manager

- Log in to the system
- Update project status (e.g., Upcoming, Ongoing, Completed)
- Update project timelines with start and end dates

- Update project budgets and expenses
- Upload project-related documents
- Update project progress percentage
- View all projects assigned to them
- Track budget utilization and project timeline visually

3. Client

- Log in to the system
- View all projects assigned to them
- Access detailed project information
- Track project budget vs. actual expenses
- Monitor project timelines and progress
- View all project-related documents

4. System/Technical Responsibilities

- **Authentication Module:**
  - Validate registration inputs (email, role)
  - Authenticate login credentials
- **Project Management Module:**
  - Handle all CRUD operations for projects
  - Assign managers and link clients
  - Maintain accurate budget, timeline, and progress data
- **Document Management Module:**
  - Store and retrieve project documents
  - Ensure documents are linked to the correct project
- **Exception Handling:**
  - Handle invalid users, projects, and invalid inputs gracefully

C. Core Logic

1. Authentication Module (AuthenticationServiceImpl)

- **Responsibilities:**
  - User registration (Builder)
  - Login validation
- **Registration Flow:**
  - Check if the provided email already exists via UserDAO
  - If email does not exist:
    - Create User
    - Save User via UserDAO
    - Return generated user ID
- **Login Flow:**
  - Retrieve user by email using UserDAO
  - Validate password and role
  - If valid, return user ID

2. Project Management Module (ProjectServiceImpl)

- **Responsibilities:**
  - Handle project operations for Builders, Managers, and Clients
- **Add Project:**
  - Builder provides project name, description, and client ID
  - Validate client existence via UserDAO
  - Create Project object
  - Save project using ProjectDAO
  - Return project ID
- **Assign Manager:**
  - Builder selects project and manager
  - Update manager_id in project via ProjectDAO

- **Track Budget:**
    - Retrieve project by ID
    - Generate visual bar for budget vs. expenses
    - Display total budget, expenses, remaining amount
- **View Timeline:**
    - Retrieve project progress, start date, and end date
    - Generate visual progress bar based on completion percentage
    - Display timeline with visual representation
- **Upload Documents:**
    - Manager provides file name and path
    - Create ProjectDocument object
    - Save document using DocumentDAO
- **Update Project Attributes:**
    - Status, budget, expenses, timeline, progress can be updated
    - DAO updates the database
    - Returns success/failure
- **View Projects & Project Details:**
    - Retrieve projects associated with a user using ProjectDAO
    - Retrieve detailed information for a project by ID
- **View Project Documents:**
    - Retrieve list of documents associated with a project using DocumentDAO

3. Data Access Layer (DAO)

- **UserDAO:**
    - Save new users
    - Retrieve users by email
    - Validate user existence
- **ProjectDAO:**
    - Add new projects

    - Assign managers
    - Update status, budget, expenses, timeline, progress
    - Retrieve projects by user ID or project ID
- **DocumentDAO:**
    - Save project documents
    - Retrieve documents by project ID

4. Exception Handling

- **InvalidUserException:** Thrown if user ID is invalid or does not exist
- **ProjectNotFoundException:** Thrown if project ID is invalid or does not exist
- **InvalidEmailException:** Invalid email format exception for registration validation

5. User Workflows

- **Builder:**
    - Register or log in
    - Add new projects
    - Add clients and project managers
    - Assign project managers
    - View project details
- **Project Manager:**
    - Log in
    - Update project status, budget, expenses, timeline, progress
    - Upload project documents
    - View assigned projects and project details
- **Client:**
    - Log in
    - View projects assigned to them
    - Track project budget and timeline
    - Access project documents

GitHub Link: https://github.com/Rithika-Mamilla/builder-portfolio-management-system

Instructions to Run the Builder Portfolio Management System

1. Prerequisites

- Java Development Kit (JDK) 11 or above installed
- PostgreSQL installed and running
- Maven installed for project build and dependency management

2. Database Setup

1. Open MySQL and create a database: builder_db
2. Run the SQL script provided to create the required tables:
   - users
   - projects
   - documents
3. Ensure the database connection details in DBUtil.java match your local MySQL configuration:
   - URL, username, password

3. Project Setup

1. Clone the GitHub repository
2. Open the project in an IDE (like IntelliJ IDEA or Eclipse)
3. Ensure Maven dependencies are downloaded
4. Compile the project to generate .class files

4. Running the Application

1. Run the Main class
2. Follow the console-based menu:
   - Register as Builder, Client, or Project Manager
   - Login with registered credentials
   - Access Builder, Manager, or Client functionalities