Project Overview

Project Name: Construction Equipment Rental Management System (CERM)

Project Overview

The goal of this project is to develop a web-based portal for managing the rental of construction equipment and tools. The system will streamline operations, improve inventory management, and enhance customer service by providing a digital platform for tracking rentals, payments, and inventory.

Business Context

Currently, the business operates using traditional methods (pen and diary) for tracking rentals and inventory, which is inefficient and prone to errors. The new system aims to digitize these processes, allowing for better monitoring of outstanding payments, inventory levels, and customer interactions.

Target Users

1. **Admin:** Manages the overall system, including user roles, inventory, and financial reports.
2. **Staff:** Handles day-to-day operations, including processing rentals and payments.
3. **Customers:** Can view available equipment, make rental requests, and manage their accounts.

Objective

To develop a web portal that allows for the management of rental equipment, tracking of inventory, customer management, and financial transactions.

Key Features

1. User Management

- User Registration/Login (Admin, Staff, Customer)
- Role-based access control
- Profile management

2. Inventory Management

- Add/Edit/Delete Equipment

- Categorization of Equipment (e.g., Cantering Plates, Concrete Mixture Machines, etc.)
- Track current stock levels
- Track equipment on rent
- Upload images and specifications for each item

3. Rental Management

- Create Rental Agreements
- Calculate rental fees based on duration and quantity
- Manage advance deposits
- Track outstanding payments
- Generate invoices and receipts

4. Customer Management

- Add/Edit/Delete Customer Profiles
- Track rental history for each customer
- View outstanding payments and rental agreements

5. Reporting

- Generate reports on inventory status (available, rented, damaged)
- Financial reports (total income, outstanding payments)
- Customer reports (rental history, outstanding dues)

6. Notifications

- Email/SMS notifications for due payments
- Alerts for low stock levels

7. Admin Dashboard

- Overview of current stock, rented items, and outstanding payments
- Quick access to key functionalities

Technical Specifications

1. Technology Stack

- Frontend: HTML, CSS, JavaScript (React.js or Angular)
- Backend: Node.js or PHP (Laravel)
- Database: MySQL or PostgreSQL
- Hosting: Cloud-based (AWS, Azure, or DigitalOcean)

2. Security

- SSL encryption for data transmission
- Secure user authentication (JWT or OAuth)
- Regular backups of the database

3. Responsive Design

- Mobile-friendly interface
- Compatibility with major browsers

Project Structure

1. **User Interface (UI)**
   - Home Page
   - Login/Registration Page
   - Dashboard
   - Inventory Page
   - Rental Management Page
   - Customer Management Page
   - Reports Page
   - Notifications Page
2. **Backend Structure**
   - User Management Module
   - Inventory Management Module
   - Rental Management Module
   - Customer Management Module
   - Reporting Module
   - Notification Module

3. **Database Structure**
   - Users Table
     - UserID (Primary Key)
     - Username
     - Password (hashed)
     - Role (Admin, Staff, Customer)
     - Contact Information
   - Equipment Table
     - EquipmentID (Primary Key)
     - Name
     - Category
     - Size
     - Quantity
     - Rent per Day

    - Advance Deposit
    - Status (Available, Rented, Damaged)
- Rentals Table
    - RentalID (Primary Key)
    - UserID (Foreign Key)
    - EquipmentID (Foreign Key)
    - Start Date
    - End Date
    - Total Rent
    - Advance Deposit
    - Status (Active, Completed, Overdue)
- Payments Table
    - PaymentID (Primary Key)
    - RentalID (Foreign Key)
    - Amount
    - Payment Date
    - Status (Paid, Unpaid)
- Notifications Table
    - NotificationID (Primary Key)
    - UserID (Foreign Key)
    - Message
    - Date
    - Status (Read, Unread)

Implementation Plan

Phase 1: Requirement Gathering

- Discuss with stakeholders to finalize requirements.

Phase 2: Design

- Create wireframes and UI/UX designs.
- Design database schema.

Phase 3: Development

- Frontend and backend development.
- Integration of modules.

Phase 4: Testing

- Unit testing, integration testing, and user acceptance testing.

Phase 5: Deployment

- Deploy the application on a cloud server.
- Set up domain and SSL.

Phase 6: Training and Support

- Provide training to staff on how to use the system.
- Offer on-going support and maintenance.