WEB TECHNOLOGIES

MARYAM TAHIR

FA22-BSE-023

ASSIGNMENT 01

TABLE OF CONTENTS

Question:

To begin this journey of building a working software web application, you are required to prepare a document that describes the project idea and includes the application requirements along with a high-level application development plan, technologies selected and architecture to use as your first assignment. The semester project must follow the software development process, and it will undergo through different phases from requirement gathering to the deployment. So, the outcome of each phase will be evaluated against the course assignments respectively as agreed by all the students in a mutual discussion for how to evaluate the assignments.

Project Proposal: Tenant Management System

1. Introduction

Managing rental properties, tenant payments, and communication between landlords and tenants can be overwhelming. Traditional methods like manual record-keeping and informal communication often result in missed payments and delays in addressing tenant concerns. The **Tenant Management System** (TMS) is a web-based platform designed to streamline this process by providing a centralized system where property owners can manage their rental properties and tenants can handle rent payments and concerns.

2. Problem Statement

Managing rental properties and payments can be tedious and time-consuming for landlords, especially when dealing with multiple tenants. Tenants often struggle to track rent due dates and payments. The absence of a centralized system results in disorganized communication and missed payments. The Tenant Management System will provide an online platform to facilitate transparent and efficient communication between property owners and tenants, improving the management of rent payments, property concerns, and overall interactions.

3. Assumptions and Claims

Assumptions:

1. Users are familiar with basic online portals.
2. Both tenants and landlords have internet access and prefer online communication.
3. Tenants will receive notifications about rent due dates.
4. The system will allow monthly rent payments, with flexibility for other options like quarterly payments.
5. The system will ensure data security and privacy.
6. Payment transactions will be processed through external gateways.

Claims:

1. The system will enhance communication between tenants and landlords.
2. Landlords will have better control over properties and payments.
3. Tenants will be able to track payments and get reminders, reducing missed payments.
4. Both tenants and landlords will experience increased transparency in managing rent records.
5. Automated reminders will improve on-time rent payments.

4. Application Requirements

Functional Requirements:

1. **User Registration & Authentication:**
   - Account creation for both tenants and landlords.
   - Secure login with role-based access.
2. **Property Management:**
   - Landlords can add properties and manage tenants.
   - Track payment histories and rent dues.
3. **Payment Tracking:**
   - Tenants can view payment history and receive notifications.
   - Landlords can generate payment reports.
4. **Issue Management:**
   - Tenants can submit repair requests.
   - Landlords can track and resolve issues.
5. **Notification System:**
   - Automated email notifications for rent dues and issues.

Non-Functional Requirements:

1. **Security:** Secure storage of data and role-based authentication.
2. **Performance:** Efficient handling of concurrent users.
3. **Usability:** Intuitive user interface for all users.
4. **Scalability:** Ability to add more properties and users as needed.

---

5. High-Level Application Development Plan

Phase 1: Requirement Gathering

- Meet with stakeholders to gather functional and non-functional requirements.

Phase 2: System Design

- Design database schema and wireframes for the user interface.

Phase 3: Backend Development

- Develop the backend using **PHP** and **Laravel** framework.
- Set up the database to store user, property, and payment data.

Phase 4: Frontend Development

- Build the user interface using **React.js**.
- Implement role-based access control.

Phase 5: Testing

- Test for security, usability, and performance.

Phase 6: Deployment

- Deploy to a cloud platform like **AWS** or **Heroku** with CI/CD pipelines.

6. Technologies Selected

- **Frontend:** React.js for the user interface.
- **Backend:** PHP with Laravel framework for handling API requests.
- **Database:** MySQL for relational data storage.
- **Authentication:** Role-based authentication using Laravel's built-in methods.
- **Payment Gateway:** Stripe or PayPal for payment tracking.
- **Deployment:** AWS or Heroku for hosting.

7. System Architecture

The Tenant Management System will follow a **client-server architecture**, with a React.js frontend and a PHP/Laravel backend. A REST API will connect the two, handling requests and responses. MySQL will store tenant data, property information, and payment history. The system will integrate with external payment gateways and use Laravel's authentication for secure user management.

7. Conclusion

The Tenant Management System will provide a streamlined, user-friendly platform that improves communication, rent payment tracking, and overall property management for both landlords and tenants. By automating key tasks and centralizing records, the system will enhance transparency and efficiency in managing rental properties.