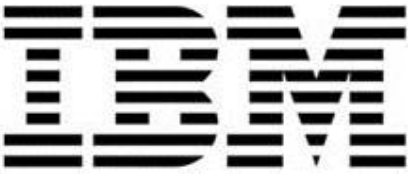

COLLEGE CODE : 9623

COLLEGE NAME : Amrita College Of Engineering And Technology

DEPARTMENT : Computer Science and Engineering

STUDENT NM ID - 2A81889A479217DBC564038EFF6EDBDB

ROLL NO : 962323104110

DATE : 25-09-2025

Completed the project named as

Phase_03_ TECHNOLOGY

PROJECT NAME : Login Authentication System

SUBMITTED BY,

NAME : Sweatha M L

MOBILE NO : 93457 68379

Project Setup

Login Authentication System — MVP Implementation

The project setup phase involves laying the groundwork for developing a secure and reliable Login Authentication System. This includes defining the scope of the MVP to focus on core functionalities such as user registration, login, and logout. The setup ensures the system can effectively validate user credentials and manage user sessions securely. Establishing this foundation will support future enhancements like password recovery, multi-factor authentication, and role-based access control.

Key activities in this phase include:

- Defining project objectives and core features
- Setting up development environment and tools
- Planning the system architecture for security and scalability
- Preparing database schemas and user data management strategies
- Establishing protocols for session handling and authentication

This structured setup phase is critical to ensure a smooth and secure development process for the authentication system.

Core Features Implementation

The MVP phase of the Login Authentication System focuses on implementing key features that ensure users can securely access the application. These core features lay the foundation for a safe and functional authentication experience:

- **User Registration:**
  New users can create accounts by providing necessary information such as username, email, and password. Input validations ensure data integrity by preventing invalid or malicious data entry, such as weak passwords or duplicate accounts.
- **User Login:**
  Registered users can authenticate themselves by entering their credentials. Passwords are never stored in plain text; instead, secure hashing algorithms (e.g., bcrypt) are used to compare the entered password with the stored hash, protecting user data from breaches.
- **Session Management:**

After successful login, user sessions are maintained either through token-based methods like JSON Web Tokens (JWT) or traditional server-side sessions. This allows users to remain authenticated across pages and requests securely.

- **Logout:**
  Users can safely terminate their sessions to prevent unauthorized access, especially on shared or public devices. Proper session invalidation ensures that no lingering authentication tokens or sessions remain active after logout.

These essential features ensure a smooth and secure authentication flow, making the system reliable for everyday use.

Data Storage

Proper data storage is fundamental to maintaining security and usability within the Login Authentication System:

- **Database Storage:**
  User data, including credentials, is securely stored in a database such as MySQL, PostgreSQL, or MongoDB. The schema is designed to efficiently manage user information while ensuring scalability for future growth. Passwords are hashed before storage to protect against data breaches and unauthorized access.
- **Local State Management:**
  On the frontend, temporary user data (such as form inputs during registration and login) is managed using local state mechanisms. This allows immediate feedback to the user and helps maintain form data during interactions, improving the overall user experience.
- **Security Measures:**
  Encryption and access control protocols are implemented to safeguard sensitive data both at rest (in the database) and in transit (during communication). Best practices such as HTTPS, secure cookies, and input sanitization prevent common security vulnerabilities like man-in-the-middle attacks or SQL injections.

By combining secure database practices with responsive local state handling, the system ensures both data integrity and user-friendly interactions.

Testing Core Features

Testing is vital to validate the correctness, security, and usability of the authentication system before deployment:

- **Unit Testing:**

Individual components such as password hashing functions, input validation routines, and authentication logic are tested in isolation. This helps catch errors early and ensures each function performs as expected under various conditions.

- **Integration Testing:**
  The end-to-end workflow, including user registration, login, session management, and logout, is tested to verify that all components interact correctly. This ensures the system behaves reliably in real-world scenarios.

UI Testing:
Forms and user interfaces are tested for proper validation feedback, error messaging, and responsiveness. This helps provide clear guidance to users and enhances overall user satisfaction by preventing confusion during authentication

Regular testing throughout the development lifecycle helps identify and resolve bugs, ensuring a robust and secure authentication system.

Version Control

Effective version control practices are crucial for managing the project source code and supporting collaborative development:

- **Git:**
  Git will be used to track changes in the codebase, allowing developers to manage different versions of the project efficiently. This facilitates code review, branching for feature development, and easy rollback if issues arise.
- **GitHub Repository:**
  Hosting the project on GitHub provides a centralized platform for collaboration. It enables issue tracking, pull requests, and continuous integration workflows to improve code quality and team coordination.
- **Best Practices:**
  Developers will commit code frequently with clear and descriptive messages. Branching strategies (such as feature branches and main/master branches) will be employed to organize work and reduce conflicts. This structured workflow ensures smooth development, easier debugging, and better documentation of the project's evolution.

Together, these version control strategies promote transparency, accountability, and efficient teamwork throughout the project.