WIL Documentation

Group 19

Introduction to Project

- **Project Overview**: Our project aims to develop a comprehensive system that integrates a web application and a mobile application for efficient management of [specific project purpose].
- **Team Members and Roles**:

  Saneliso Lehlohla (Team Lead): Fullstack/Documentation (ST10031913)

  Liam Cole Abraham (Vice Lead): Fullstack/Documentation (ST10144656)

  Hamzah Ballim: Backend/Documentation (ST10084302)

  Seth Rajkumar: Frontend/Documentation (ST10138969)

  Saieshen Teal Naidoo: Documentation (ST10393289)

Technology Stack

- **Web Development**:

  Languages: HTML, CSS, JavaScript

  Framework: ASP.NET Core (MVC architecture)

  IDE: Visual Studio 2022

- **Mobile Development**:

  Languages: Kotlin

  Framework: Giraffe on Android Studio

  Database: Firebase, SSMS Software

- **APIs**: Firebase Authentication API, PayPal API, Google Maps Platform, Disqus API

Website Development

- **Frontend**:

  HTML/CSS: Used for structuring and styling the web pages.

  JavaScript: Enables interactive functionalities.

  Bootstrap: Used for responsive design.

- **Backend**:

  ASP.NET Core (MVC): Provides the structure and development framework.

  C#: Core language for server-side logic.

- **Database Integration**:

  Firebase: Real-time database for dynamic content.

  SSMS: For managing SQL databases.

App Development

- **Development Environment**:

  Android Studio: Main IDE for app development.

  Giraffe Framework: Provides necessary libraries and tools.
- **Programming Language**: Kotlin
- **Database**: Firebase for real-time data management.
- **APIs**: Integration with various APIs for enhanced functionalities.

API Integration

- **Firebase Authentication API**: For user authentication and authorization.
- **PayPal API**: Enables secure payment processing.
- **Google Maps Platform**: Provides location-based services.
- **Disqus API**: Adds commenting features to the application.

Non-Functional Requirements

- **Performance**:
- The system should handle up to 10,000 concurrent users with a response time of less than 2 seconds.
- Database queries should return results within 500 milliseconds.
- **Scalability**:
- The application should be scalable to accommodate future growth in user base and data volume.
- Horizontal scaling of servers should be supported.
- **Reliability**:
- The system should have an uptime of 99.9%.
- Regular backups and disaster recovery mechanisms should be in place.

Non-Functional Requirements

- **Usability**:
- The user interface should be intuitive and easy to navigate.
- Accessibility standards should be met to ensure usability for people with disabilities.
- **Security**:
- Data encryption should be used for sensitive information.
- Secure authentication mechanisms should be implemented.
- **Maintainability**:
- Code should be well-documented and follow coding standards.
- The system should support easy updates and bug fixes.

Project Risks

- **Data Loss**: Risk of losing critical data; mitigated by regular backups.
- **Integration Challenges**: Compatibility issues between different technologies; addressed by thorough testing.
- **Downtime**: Potential service interruptions; minimized by using reliable hosting services.
- **Security Vulnerabilities**: Risk of data breaches; managed by implementing robust security measures.

Assumptions, Constraints, and Dependencies

- **Assumptions**:

Users will have access to the internet and compatible devices.

All team members have the necessary skills and tools to complete their tasks.

The APIs and third-party services used will remain available and maintain backward compatibility.

- **Constraints**:

Budget limitations may affect the scope and features of the project.

The project must be completed within the allocated timeline.

The system must comply with relevant data protection and privacy regulations.

- **Dependencies**:

Successful integration with Firebase, PayPal, Google Maps, and Disqus APIs.

Availability of necessary hardware and software resources.

Dependence on third-party services and their reliability.

Coordination with stakeholders and timely feedback for iterative development.

Ethical and Piracy Concerns

- **User Privacy**: Ensuring user data is handled responsibly and securely.
- **Financial Transparency**: Clear communication of financial transactions and costs.
- **Resource Use**: Ethical use of third-party resources and APIs.
- **Accessibility**: Ensuring the application is accessible to all users, including those with disabilities.
- **Plagiarism**: Original content creation and proper attribution of external resources.

Definition of Done

- **Code Completion**: All features implemented and code written.
- **Testing**: Comprehensive testing to ensure functionality and performance.
- **Peer Review**: Code and features reviewed and approved by team members.
- **Deployment**: Application successfully deployed and accessible to users.

Definition of Ready

- **Independent Development**: Tasks can be completed independently without dependencies.
- **Acceptance Criteria**: Clear criteria defined for each task.
- **Value**: Each task adds value to the overall project.
- **Testability**: Tasks are designed to be easily testable.

Work Agreement and Deliverables

- **Agreement Details**: Outline of team agreements on working hours, communication, and collaboration.
- **Project Deliverables**:
- **Registration System**: User registration and management.
- **Patient Profile Management**: Managing patient data and history.
- **Appointment Scheduling**: Scheduling and managing appointments.
- **Treatment Plan Management**: Creating and tracking treatment plans

Roadmap and Timeline

- **Project Phases**:

  Phase 1: Requirements gathering and initial design.

  Phase 2: Development of core functionalities.

  Phase 3: Integration and testing.

  Phase 4: Deployment and user training.

- **Timeline**: Detailed timeline with milestones and deadlines for each phase.

Devops

- Github
- CircleCI, (CI/CD Pipeline)