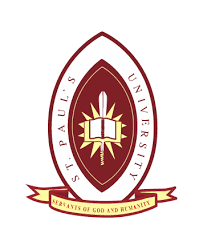
**Health Center Solution – Phase 2**

**Group Name: Group C.**

**Date:11th November, 2024**

**Health Center Solution - Phase 2**

****

**St. Paul’s University**

**Limuru Campus**

**Group Members**

**Date of Submission: 11th November, 2024**

**1. Project Definition**

**a. Project Description**

The Health Center Solution project is designed to provide a digital solution for managing patient data, appointment scheduling, prescription management, and related functionalities for a local health center. Utilizing a microservices architecture, the system will facilitate streamlined operations, improving efficiency and reducing administrative load. Key components include:

Patient registration and record management

Appointment booking

Prescription and medication management

Secure data storage and retrieval

Reporting and analytics features

**b. Link to the GitHub Repository**

* The GitHub repository for version control and project tracking can be accessed here:
* **<https://github.com/Seba-Tech-Company/software-engineering-2.git>**
  + This repository includes the project’s version history, team contributions, and documentation updates.

**2. Software Design Considerations**

**a. UI Design (Including Wireframes)**

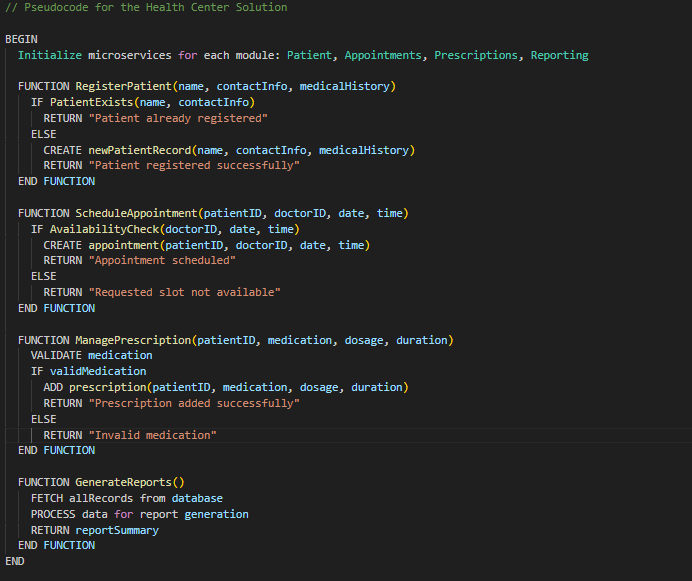
* **User Interface (UI) Overview**: The UI design prioritizes user-friendly and accessible interfaces for both health center staff and patients. Key UI features include easy navigation, clear labeling, and responsive layouts to ensure usability across devices.
* **Wireframes**:\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*???????????///
  + Attach wireframe images or include links if stored in the repository.
  + Wireframes outline primary screens, such as the patient dashboard, appointment scheduler, and prescription pages.

**b. Design Patterns**

* **Design Patterns Used**:
  + This project utilizes the following design patterns:
    1. **Model-View-Controller (MVC)**: Separates data management, user interface, and application logic, enhancing maintainability and facilitating clear data flow.
    2. **Microservices Architecture**: Allows for modular services (e.g., patient services, appointment services), improving scalability and flexibility.
    3. **Repository Pattern**: Helps in data management, abstracting data access for simplified and flexible storage handling.
* **Advantages of Reusing Design Patterns**:
  + By reusing established design patterns, the project benefits from reduced development time and improved software maintenance. Adopting well-known patterns increases consistency, allowing easier debugging and future scalability.

**3. Software Implementation**

**a. System Pseudocode**



This pseudocode is designed to outline key operations in each module, with a focus on simplicity and readability. It serves as the framework for more detailed code development in each microservice.

**4. Software Test Plan**

**a. Overview of Testing Levels**

* **Unit Testing**:
  + Each microservice component will be tested individually to ensure that isolated functions (e.g., patient registration, appointment scheduling) perform as expected.
  + Tools: [Specify tools used, such as JUnit, Mocha, etc.]
* **Integration Testing**:
  + Tests will ensure that different microservices (e.g., Patient and Appointment services) interact correctly.
  + Focus will be on API calls, data sharing, and service communication.
* **System Testing**:
  + Conduct end-to-end testing across the entire system to ensure that it meets the functional and performance requirements.
  + Test cases will simulate real-world usage, including registration, scheduling, and data access workflows.
* **User Acceptance Testing (UAT)**:
  + This final level involves healthcare staff reviewing the application for usability, accuracy, and alignment with operational workflows.
  + Feedback will be gathered to make last adjustments before deployment.

**b. Sample Test Cases**

* **Patient Registration**:
  + *Objective*: Verify that a new patient can register with valid inputs.
  + *Expected Result*: Registration successful, and patient data is stored.
* **Appointment Scheduling**:
  + *Objective*: Confirm that a patient can book an appointment only when a doctor is available.
  + *Expected Result*: Appointment confirmation message or an error if the time slot is unavailable.
* **Prescription Management**:
  + ***Objective***: Ensure valid prescriptions can be entered, and invalid data is rejected.
  + ***Expected Result***: Prescription successfully added or error message displayed for invalid medication.

**c. Test Results Documentation**

Test results will be documented in the project repository to track outcomes, issues, and resolutions.

Each test case will include the test objective, execution date, tester’s name, expected vs. actual results, and follow-up actions if needed.