Appointment Scheduling System

1.0 Problem statement

The Appointment Scheduling System (ASS) application helps to automate the process of managing the activities of Member Group Coaching like Registration for sessions, Cancellation, Rescheduling etc.

The following section will cover aspects related to Appointment Scheduling System.

a) Appointment Registration
b) Appointment Cancellation
c) Appointment Rescheduling
d) Appointment Notifications/Remainder

Scope of the System

The scope of the system is explained through its modules as follows

- Appointment Login and Registration – will be used by members to register the existing schedules (sessions) and save them into the system. The system stores the details of the appointment registration in the system along with the member's details.

  A member should be able to login with a User Id and Password that exists in the database. On clicking logout, the session should be invalidated, and the login page must be displayed

  Capture the details like Name, Username, Password, Email Address, Contact No, DOB, along with appointment registration details.

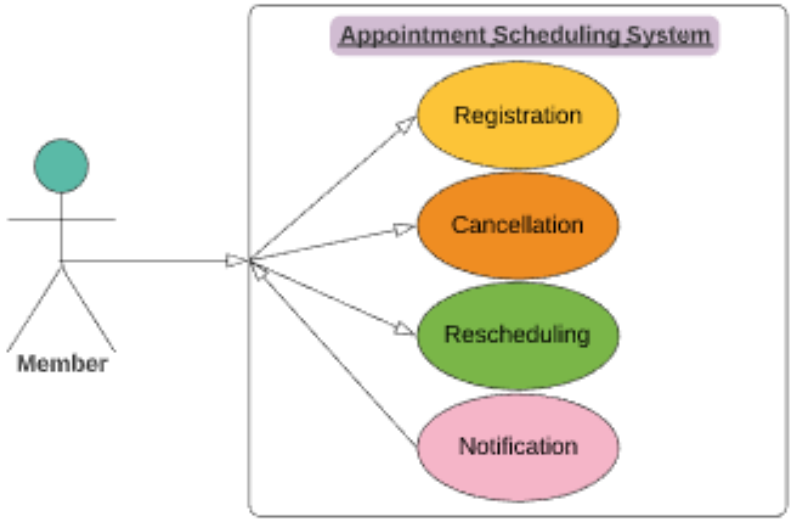
- Appointment Cancellation - will be used by members to cancel already booked schedules and save them into the system. The system stores the details of the appointment cancellation in the system along with the member's details.

  Capture the details like Name, Username, Password, Email Address, Contact No, DOB, along with appointment cancellation details.

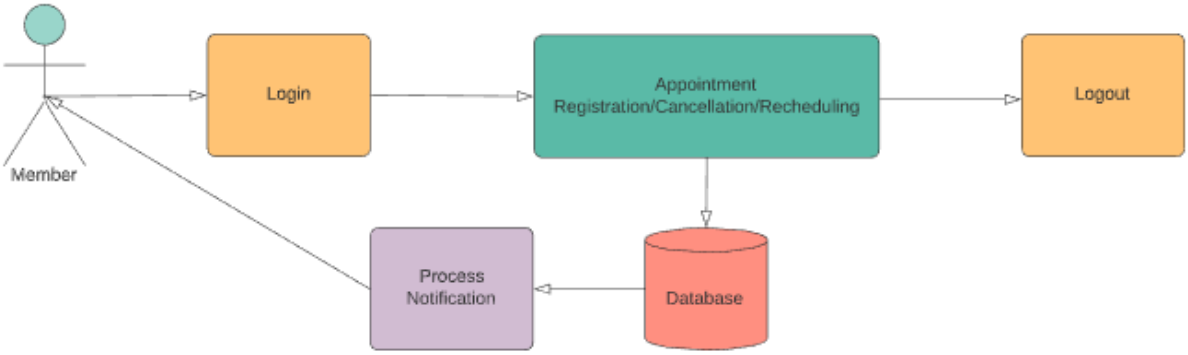
- Appointment Rescheduling - will be used by members to reschedule the already booked schedules and save them into the system. The system stores the details of the appointment rescheduling in the system along with the member's details.

- Appointment Notification/Remainder - will be used when any of the Appointment Registration, Cancellation and Rescheduling occurs in the Appointment system. The system notifies the member with the appropriate details.

2.0 Use Case Diagram



Flow Diagram



3.0 Project Development Guidelines

The project to be developed based on the below design considerations

Backend Development	• Use Rest APIs (Springboot/ASP.Net Core WebAPI to develop the services • Use Java/C# latest features

	• Use ORM with database • Use Swagger to invoke APIs • Implement API Versioning • Implement security to allow/disallow CRUD operations • Message input/output format should be in JSON (Read the values from the property/input files, wherever applicable). Input/output format can be designed as per the discretion of the participant. • Any error message or exception should be logged and should be user-readable (not technical) • Database connections and web service URLs should be configurable • Implement Unit Test Project for testing the API • Follow Coding Standards
Frontend Development	• Use Angular/React to develop the UI • Implement Forms, databinding, validations • Implement Routing and navigations • Use JavaScript to enhance functionalities • Implement External and Custom JavaScript files • Implement Typescript for Functions, Operators. • Any error message or exception should be logged and should be user-readable (and not technical) • Follow coding standards • Follow Standard project structure

4.0 Good to have implementation features

- Generate a SonarQube report and fix the required vulnerability
- Use the Moq framework as applicable
- Create a Docker image for the frontend and backend of the application
- Implement OAuth Security
- Implement Logging
- Implement design patterns
- Use JWT for authentication in SpringBoot/WebApi. A Token must be generated using JWT. Tokens must expire after a definite time interval, and authorization must be handled accordingly based on token expiry
- Deploy the docker image in AWS EC2 or Azure VM
- Build the application using the AWS/Azure CI/CD pipeline. Trigger a CI/CD pipeline when code is checked-in to GIT. The check-in process should trigger unit tests with mocked dependencies
- Use AWS RDS or Azure SQL DB to store the data