Member Management

**Low Level Design Document**

Contents

[1 Member Registration 3](#_Toc151368749)

[1.1 Introduction 3](#_Toc151368750)

[ Purpose Of the Document 3](#_Toc151368751)

[ Scope 3](#_Toc151368752)

[ System overview 3](#_Toc151368753)

[1.2 Understanding requirement 4](#_Toc151368754)

[1.3 User Interface design 4](#_Toc151368755)

[1.4 Database Design 5](#_Toc151368756)

[1.5 Component Design 5](#_Toc151368757)

[1.6 UML Design 5](#_Toc151368758)

[1.7 Testing Design 6](#_Toc151368759)

[1.7.1 Flow Diagram 6](#_Toc151368760)

# Introduction

* Purpose Of the Document

This document serves as a detailed guide for the development team to implement the low-level design of the Member Management Module. It outlines the specific components, data flow, algorithms, and interfaces required to realize the functionality of managing members within the application.

* Scope

The scope of this LLD document is focused on providing a detailed design for the Member Management module within the application. It encompasses the following aspects:

**Module Overview**

* Brief description of the module’s purpose and functionality
* List of key features or responsibilities of the module
* Dependencies on other modules or components

**User Input and Validation**

* + Registrant must provide a valid email id as input
  + Implementing validation logic for ensuring the correctness and completeness of user-provided email id.
  + Registrant also provide valid OTP same as system sends to his email account.
  + Validate OTP through a validation logic.

**Database Interaction:**

* + After validation member information must be stored in the member table .
  + OTP related informations must be stored in another table with a member Id.

**Error Handling and Feedback:**

* + Outlining strategies for handling errors during registration.
  + Members must be informed of successful or unsuccessful registration attempts.
* System overview

**Key component**

* + Registration component
  + OTP component

**Dependencies**

**Integration** **points**

* + Todo: Specify any external module with whom this system interacts
  1. Understanding requirement

**Functional requirements**

* + Member Registration
  + OTP verification

**Non-functional requirements**

* + Performance: Specify the expected response times for different operations.
  + Scalability: Define how the system will handle an increasing number of members.
  + Security: Outline security measures for protecting member data, including encryption and access controls.
  1. User Interface design

**Purpose**

The member registration interface is designed to register a member into the system and grant access to it. It includes two pages like registration page and OTP page.

**Components**

* + Email Id as input field
  + Verify Email button
  + Send OTP button
  + OTP as input field
  + Register button

**Layout**

* + Member registration UI: EmailId(TextBox), Verify Email(button)
  + OTP UI: SendOTP(button),OTP(TextBox)

**Styling**

* + TODO

**Error Handling**

* + For Wrong Email Id provided Textbox must be highlighted
  + For Already registered email Id Text box must be highlighted
  + After 3 attempts of Wrong OTP user redirect to Register page

**Performance consideration**

* + Caching strategies for registrant credentials (if applicable).
  1. Database Design

**Data Model**

**Member Table:**

Columns:

* + - MemberId(primary key)
    - EmailId(unique)

**OTP Table:**

Columns:

* + - EmailId(Fk)
    - OTP(Unique)
    - generated On
  1. Component Design
  2. UML Design

**Sequence Diagram**

**Registrant RegistrationManager DBManager EmailService**

**| | |**

**|-----------register-------------->| |**

**|-----------validateInputs------> | |**

**|<----------validationResult------|**

**|-----------checkMemberExists-🡪 ----------|**

**|<----------memberExists?------ |**

**| | | |**

**| |---storeMemberDetails-> |**

**| |<-----storeResult-------| |**

**| | | |**

**|<--registrationSuccess- ------- ----| | |**

**| | | |**

**|-----------------successEmailConfirmation------------------->|**

**Flow Diagram**

* 1. Testing Design
  + Todo