

**Mentor Appointment Scheduling Application**

**Software Design Document**

– Ho Chi Minh, May 2022 –

**Table of Contents**

[I. Overview](#_heading=h.gjdgxs) **3**

[1. Code Packages/Namespaces](#_heading=h.30j0zll) 3

[1.1 Back-end Code Packages](#_heading=h.o1jjkkxx1xtq) 3

[2. Database Schema](#_heading=h.1fob9te) 5

[II. Code Designs](#_heading=h.3znysh7) **6**

[1. <Feature/Function Name1>](#_heading=h.2et92p0) 6

[a. Class Diagram](#_heading=h.tyjcwt) 6

[b. Class Specifications](#_heading=h.3dy6vkm) 7

[XYZ Class](#_heading=h.ab5sxe9j80kc) 7

[ABC Class](#_heading=h.m008ovp6n2fp) 7

[c. Sequence Diagram(s)](#_heading=h.1t3h5sf) 7

[d. Database queries](#_heading=h.4d34og8) 7

[2. <Feature/Function Name2>](#_heading=h.2s8eyo1) 8

[III. Database Tables](#_heading=h.17dp8vu) **8**

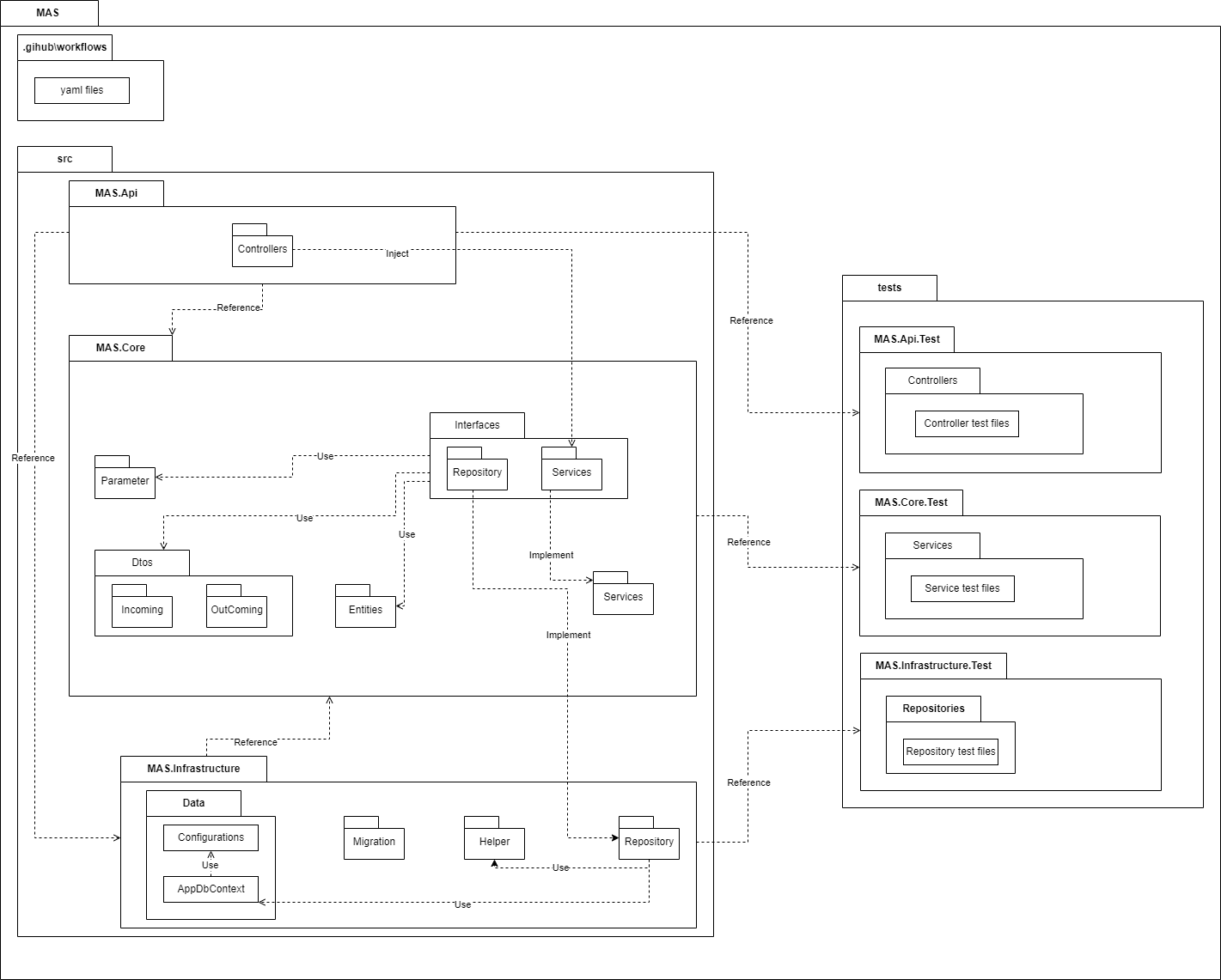
[1. <Table name 1>](#_heading=h.3rdcrjn) 8

[2. <Table name 2…>](#_heading=h.26in1rg) 8

# I. Overview

## 1. Code Packages/Namespaces

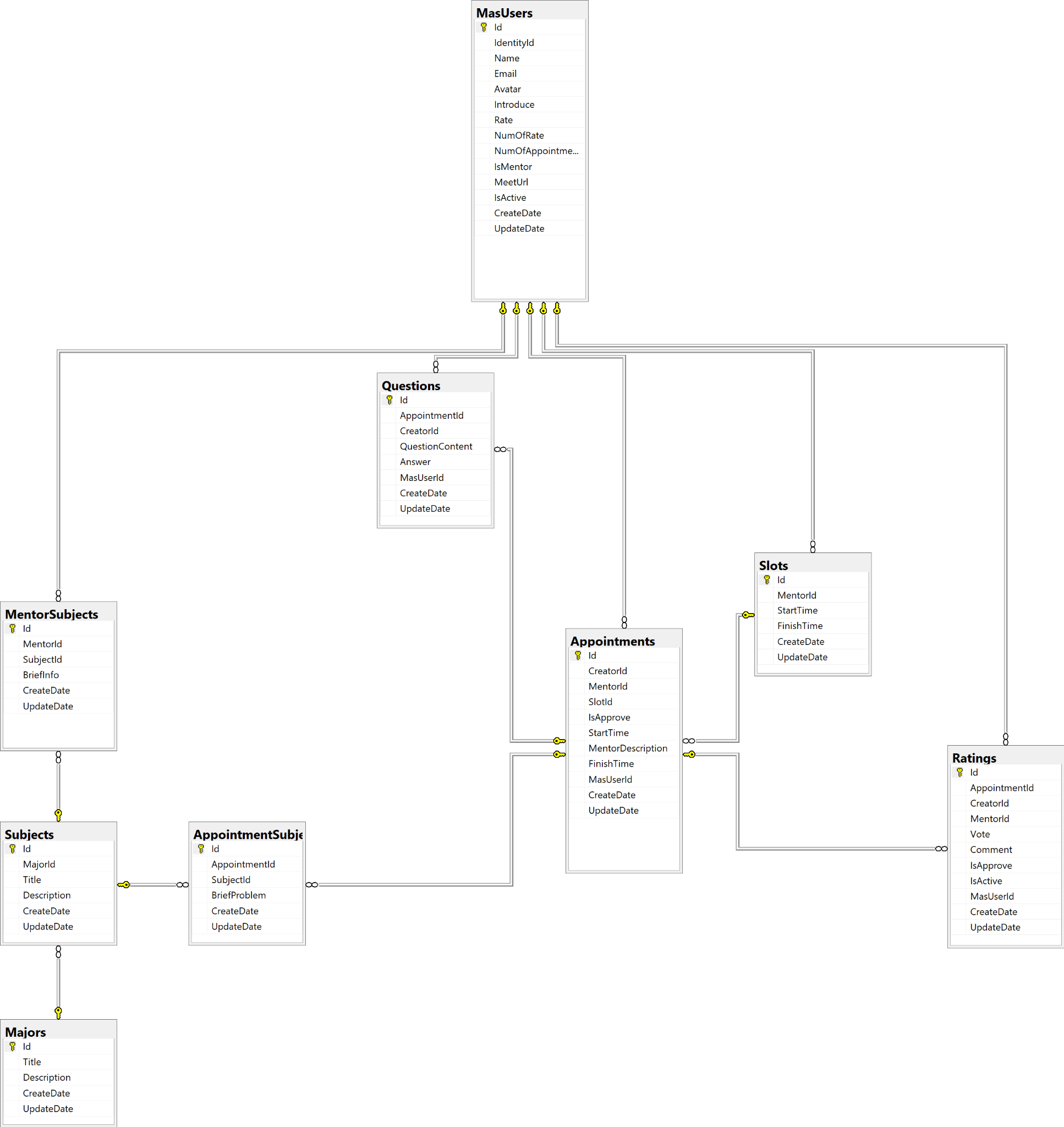
### 1.1 Back-end Code Packages

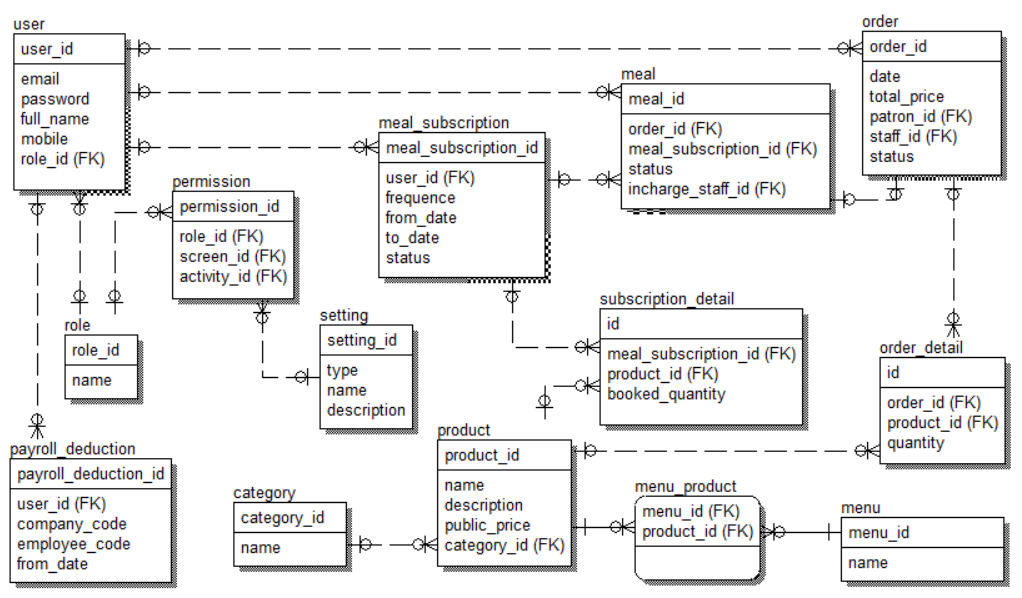


***Package descriptions & package class naming conventions***

| **No** | **Package** | **Description** |
| --- | --- | --- |
| 01 | .github\workflows | Contain config files (yaml files) used to work with GitHub Actions for CI/CD source code. |
| 02 | src | Contain main source code of project API, include: MAS.Api, MAS.Core and MAS.Infrastructure. |
| 03 | MAS.API | Contains source code of the .NET 5 API, it is a Web API project. It has two main packages: Helpers and Controllers. |
| 04 | Helpers | Contains config files for the API such as: service injection (declare the dependency injection in to Dependency Inject Container of Asp.Net Core), config for swagger UI to easy testing api on Web browser, auto Seed Data file use to seed data to Database if it is not exists, constant file for constant value of API. |
| 05 | Controllers | Contains controller files responsible for handling HTTP requests and returning responses with values. |
| 06 | MAS.Core | Contains source code of the Core Business of API, It is a Classlib project, including some main packages such as: Parameters, Interfaces (contains Repository interfaces and Service interfaces), Data Transfer Object (Dtos), Services (Implement level). |
| 07 | Entities | Contains all model classes for mapping with database tables. |
| 08 | Parameters | Contains all Parameters class files, it is used for query routes in API. |
| 09 | Exceptions | Contains files used for logging in Product Environment (Deploy in Azure). |
| 10 | Dtos | Contains Data transfer object classes used for data binding input object (Incoming package) and response mapping output object (Outcoming package). |
| 11 | Interfaces | Contains Interface classes of Repositories (which are working with Entity Framework Core and Database SQL Server to get data), and Services (which will be use by the API, they are same to the Repository Interfaces but they call and use the Repository Interfaces). |
| 12 | Services | Contains the Implement classes of the Interfaces/Services, It calls the Repository, catches the error from the input parameters from users, then passes these input parameters through the repository (Dependency Injection). |
| 13 | MAS.Infrastructure | Contains main files used for Entity Framework Core Object Relational Mapping with Database and the Repository (Implementation from the Repository Interfaces in Core layer). |
| 14 | Data | This package contains the AppDbContext class, where all entities or model classes use mapping with the table in Database. |
| 15 | Entities | Contains all model classes for mapping with database tables. |
| 16 | Migrations | Contains migrations file, history design and modified database. |
| 17 | Helpers | Contains files help populate new model objects. |
| 18 | Repository | Contains files implementation of the Repository Interface in Core layer. |
| 19 | tests | Contains test projects and test files. |

## 2. Database Schema





***Table descriptions & package class naming conventions are as below***

| **No** | **Table** | **Description** |
| --- | --- | --- |
| *01* | *<Table name>* | *<Description of the table>*  *- Primary keys: <<list of primary key fields>>*  *- Foreign keys: <<list of foreign key fields>>* |
| *02* |  |  |

# II. Code Designs

## 1. <Feature/Function Name1>

*[Provide the detailed design for the function <Feature/Function Name1>. It include Class Diagram, Class Specifications, and Sequence Diagram(s)]*

### a. Class Diagram

*[This part presents the class diagram for the relevant feature]*



### b. Class Specifications

*[Provide the description for each class and the methods in each class, following the table format as below]*

#### XYZ Class

*[Provide the detailed description for the class methods]*

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *<method name>* | *<Description of the method, including the inputs, outputs & internal method processing>* |
|  |  |  |

#### ABC Class

***Class Methods***

*[Provide the detailed description for the class methods]*

| **No** | **Method** | **Description** |
| --- | --- | --- |
| *01* | *<method name>* | *<Description of the method, including the inputs, outputs & internal method processing>* |
|  |  |  |

### c. Sequence Diagram(s)

*[Provide the sequence diagram(s) for the feature, see the sample below]*



### d. Database queries

*[Provide the detailed SQL (select, insert, update...) which are used in implementing the function/screen]*

## 2. <Feature/Function Name2>

…

# III. Database Tables

## 1. <Table name 1>

*[Give some lines about the table here>>*

*[Table fields, in the form of table format as below]*

| **#** | **Field name** | **Type** | **Size** | **Unique** | **Not Null** | **PK/FK** | **Notes** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Field name1 |  |  |  |  |  |  |
| 2 | Field name2 |  |  |  |  |  |  |

## 2. <Table name 2…>