**MINISTRY OF EDUCATION AND TRAINING**

|  |
| --- |
| **FPT UNIVERSITY** |
| **Capstone Project Document** |
| **Olives health care system** |
|  |
| |  |  | | --- | --- | | **TrendyPlus Team** | | | **Group Member** | * Nguyễn Duy Linh (SE03150) * Bùi Đức Trọng * Trần Ánh Dương (SE02797) * Phạm Đức Thắng * Phạm Minh Tuấn * Nguyễn Văn Hưng | | **Supervisor** | Lecturer : | | **Capstone Project code** | Olives |   Hanoi, 24/4/2015 |

**Contents**

[Acknowledgements 5](#_Toc459337012)

[Definitions and Acronyms 6](#_Toc459337013)

[CHAPTER 4: SOFTWARE DESIGN 7](#_Toc459337014)

[1. Purpose 7](#_Toc459337015)

[2. Architecture Overview 7](#_Toc459337016)

[2.1. System Architecture 7](#_Toc459337017)

[2.2. System Architecture Explanation 7](#_Toc459337018)

[3. Architecture Design 7](#_Toc459337019)

[3.1. Olives Administration sub-system 7](#_Toc459337020)

[i. API 7](#_Toc459337021)

[3.2. Olives sub-system 7](#_Toc459337022)

[ii. API layer 7](#_Toc459337023)

[4. Database Design 7](#_Toc459337024)

[4.1. ERD 7](#_Toc459337025)

[4.2. Database dictionary 7](#_Toc459337026)

[4.2.1. Person 7](#_Toc459337027)

[4.2.2. Patient 8](#_Toc459337028)

[4.2.3. Doctor 8](#_Toc459337029)

[4.2.4. Place 9](#_Toc459337030)

[4.2.5. Specialty 9](#_Toc459337031)

[4.2.6. Diary 9](#_Toc459337032)

[4.2.7. Rating 9](#_Toc459337033)

[4.2.8. Relationship 10](#_Toc459337034)

[4.2.9. RelationshipRequest 10](#_Toc459337035)

[4.2.10. Allergy 10](#_Toc459337036)

[4.2.11. Addiction 11](#_Toc459337037)

[4.2.12. BloodPressure 11](#_Toc459337038)

[4.2.13. Heartbeat 12](#_Toc459337039)

[4.2.14. BloodSugar 12](#_Toc459337040)

[4.2.15. MedicalRecord 12](#_Toc459337041)

[4.2.16. Prescription 13](#_Toc459337042)

[4.2.17. PrescriptionImage 13](#_Toc459337043)

[4.2.18. MedicalNote 14](#_Toc459337044)

[4.2.19. MedicalCategory 14](#_Toc459337045)

[4.2.20. MedicalImage 15](#_Toc459337046)

[4.2.21. MedicalRecord 15](#_Toc459337047)

[4.2.22. Notification 15](#_Toc459337048)

[4.2.23. RealtimeConnection 16](#_Toc459337049)

[4.2.24. Message 16](#_Toc459337050)

[4.2.25. AccountToken 17](#_Toc459337051)

[4.2.26. Appointment 17](#_Toc459337052)

[5. References 19](#_Toc459337053)

[6. Detail design 19](#_Toc459337054)

[6.1. General sequence diagram 19](#_Toc459337055)

[6.1.1 Olives back-end general sequence diagrams 19](#_Toc459337056)

[6.1.1.1. Find record 19](#_Toc459337057)

[6.1.1.2. Initialize record 19](#_Toc459337058)

[6.1.1.3. Edit record 19](#_Toc459337059)

[6.1.1.4. Delete record 19](#_Toc459337060)

[6.1.1.5. Filter record 19](#_Toc459337061)

[6.1.2. Olives front-end general sequence diagrams (Patient application) 19](#_Toc459337062)

[6.1.2.1. Non-medical records data sequence 19](#_Toc459337063)

[6.1.2.2. Medical records data sequence 19](#_Toc459337064)

[6.1.3. Example of ordinary sequence diagrams 20](#_Toc459337065)

[6.1.3.1. Initialize blood pressure note 20](#_Toc459337066)

[6.1.3.3. Initialize appointment 20](#_Toc459337067)

[6.1.4. Detail design & use case 20](#_Toc459337068)

[6.1.4.1. Olives Administration sub-system 20](#_Toc459337069)

[6.1.4.1.1. Login 20](#_Toc459337070)

[a. Class diagram 20](#_Toc459337071)

[b. Sequence diagram 21](#_Toc459337072)

[c. Class definition 22](#_Toc459337073)

[**[Class]ApiParentController** 22](#_Toc459337074)

[[Method]RetrieveValidationErrors() 22](#_Toc459337075)

[**[Class]AdminController** 22](#_Toc459337076)

[[Attribute] 22](#_Toc459337077)

[[Method]LoginAsync() 23](#_Toc459337078)

[Method: EditAdminProfileAsync() 23](#_Toc459337079)

[**[Class]RepositoryAccountExtended** 23](#_Toc459337080)

[Attribute 23](#_Toc459337081)

[Method: FilterDoctorsAsync() 24](#_Toc459337082)

[Method: FilterPatientsAsync() 24](#_Toc459337083)

[Method: SummarizePersonRoleAsync() 24](#_Toc459337084)

[6.1.4.1.1. Logout 25](#_Toc459337085)

[Create medical categories 25](#_Toc459337086)

[MedicalCategoryController class 26](#_Toc459337087)

[Attribute 26](#_Toc459337088)

[Method: InitializeMedicalCategory() 26](#_Toc459337089)

[Method: ModifyMedicalCategory() 27](#_Toc459337090)

[Method: FilterCategories() 27](#_Toc459337091)

[Edit medical categories 28](#_Toc459337092)

[Filter medical categories 28](#_Toc459337093)

[Activate account 28](#_Toc459337094)

[Get information of a specific doctor 29](#_Toc459337095)

[DoctorController class 30](#_Toc459337096)

[Attribute 30](#_Toc459337097)

[Method: Get() 31](#_Toc459337098)

[Method: Filter() 31](#_Toc459337099)

[Filter doctors list 31](#_Toc459337100)

[Filter patients list 32](#_Toc459337101)

[Statistic account by role 33](#_Toc459337102)

[PersonController class 34](#_Toc459337103)

[Attribute 34](#_Toc459337104)

[Method: Status() 34](#_Toc459337105)

[Create place 35](#_Toc459337106)

[PlaceController class 35](#_Toc459337107)

[Attribute 36](#_Toc459337108)

[Method: FindPlace() 36](#_Toc459337109)

[Method: InitializePlace() 36](#_Toc459337110)

[Method: ModifyPlace() 36](#_Toc459337111)

[Method: FilterPlace() 37](#_Toc459337112)

[Edit place 37](#_Toc459337113)

# Acknowledgements

*The success and final outcome of this project required a lot of guidance and assistance from many people and wish to express our deepest gratitude to them. Without them, this project would have been done with much more obstacles than it was.*

*First, we want to say thanks to Mr. Nguyen Tat Trung for his continuously sharing and motivating throughout the project. Mr. Nguyen Tat Trung has always been very responsive in providing necessary information, and without his generous support project would lack in accurate information on current developments. Under the instructions of Mr. Nguyen, the project team has always felt comfortable and confident. Apart from all the technologies and methodologies, Mr. Nguyen shared with us the philosophy of keeping the right attitude towards problems.*

*Second, we would like to say thanks to the instructors at FPT University for all the classes. Without the knowledge that university teaches us through each class, semester, we cannot finish this project.*

*Third, we truly appreciate the random fellows on the internet who has discussed with us. Every our tiny problem was answered clearly.*

*Finally, we want to thank our family for their help, inspiring during this project.*

# Definitions and Acronyms

|  |  |  |
| --- | --- | --- |
| **Acronym** | **Definition** | **Note** |
| SRS | Software Requirement Specification |  |
| API | Application Programming Interface |  |
| HTML | Hypertext Markup Language |  |
| CSS | Cascading Style Sheets |  |
| IDE | Integrated Development Environment |  |
| DBMS | Database Management System |  |
| UML | Unified Modeling Language |  |
| PM | Project Manager |  |
| UC | Use Case |  |
| N/A | Not Available |  |
| ORM | Object Relational Mapping |  |
| GUI | Graphical User Interface |  |
| HTTP | Hypertext Transfer Protocol |  |
| PHP | Hypertext Preprocessor |  |
| REST | Representational State Transfer |  |
| RAM | Random Access Memory |  |
| URL | Uniform Resource Locator |  |

# CHAPTER 4: SOFTWARE DESIGN

## Purpose

This chapter is to give the developer team a guidance of what the system’s architecture is, and how they should be implemented. This chapter consists of:

* Architecture overview
* Component diagram
* Detailed design
* Detailed description of components
* Database design

## Architecture Overview

### 2.1. System Architecture

<TODO: Import system data architecture>

**

*Figure 4-1: Olives System Architecture*

### 2.2. System Architecture Explanation

The Olives System is divided into 2 sub-systems hosted on Azure cloud services, use one SQL in common:

* Olives system :
  + This sub-system is for ordinary users in the system: guests, doctors and patients. It includes 2 mobile applications and 1 web application and 1 API service.
    - Mobile applications for users who are doctors and patients.
      * These 2 applications are design for 2 specific roles: doctors and owners. They're built by using Objective C and run on iOS devices. These applications help users to be more flexible in using system. Messages and Notifications are sent instantly to bring Real-Time experience.
    - One web application for users of both role.
      * An application which has been built on AngularJS. It helps users interact and consume system services. It can be connected any time, any places by using browsers. Notifications are sent instantly to bring Real-Time experience.
    - API service:
      * A service which is built on ASP.NET Web API 2 framework and hosted on Azure cloud service. It takes responsibility in taking requests come from user applications, connecting to database and processing the received data and respond processed result to user applications.
* Olives administration system:
  + This sub-system is for administrators to manage system. It includes one web application and one API service application.
    - Web application: an application which helps administrators to manage users in system, to track their statuses, to activate/deactivate ordinary users …
    - API Service: A service which is built on ASP.NET Web API 2 framework and hosted on Azure cloud service. It takes responsibility in taking requests come from user applications, connecting to database and processing the received data and respond processed result to user applications.

## Architecture Design

### 3.1. Olives Administration sub-system

#### API

**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Namespace/Directory** | **Description** | **Convention** |
| 1 | Controllers | Contains all controller source files which receive requests come from clients, processs data and return results back to requesters. | xxxController |
| 2 | Interfaces | Contains all interfaces definition files which are inherited by another classes, such as repositories, services, ... | **Ixxx** |
| 3 | Attributes | Contains all validation attributes source file which are used for data validation purposes. | xxxAttribute |
| 4 | Enumerations | Contains enumerations which are used in service. |  |
| 5 | Repositories | Contains classes which is used for access database to process data and return data to controller. Controllers don’t access database directly. | **Repositoryxxx** |
| 7 | Scripts | Contains javascript files which are used in Views. |  |
| 8 | ViewModels | Contains classes which define data structure sent from clients and vice versa. | xxxViewModel |

*Table 4-1: Namespace Explanation*

### 3.2. Olives sub-system

#### API layer

*Figure 4-13: TrendyPlus Layer Design*



|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Namespace/Directory** | **Description** | **Convention** |
| 1 | Controllers | Contains all controller source files which receive requests come from clients, processs data and return results back to requesters. | xxxController |
| 2 | Interfaces | Contains all interfaces definition files which are inherited by another classes, such as repositories, services, ... | **Ixxx** |
| 3 | Attributes | Contains all validation attributes source file which are used for data validation purposes. | xxxAttribute |
| 4 | Enumerations | Contains enumerations which are used in service. |  |
| 5 | Repositories | Contains classes which is used for access database to process data and return data to controller. Controllers don’t access database directly. | **Repositoryxxx** |
| 7 | Scripts | Contains javascript files which are used in Views. |  |
| 8 | ViewModels | Contains classes which define data structure sent from clients and vice versa. | xxxViewModel |
| 9 | Views | Contains csthml files which are used for being displayed on client browsers. |  |

*Table 4-2: Namespace Explanation*

## Database Design

### 4.1. ERD

<TODO: Import database diagram>

### 4.2. Database dictionary

#### 4.2.1. Person

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about users : Admin, Patients, Doctors. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | The id of user | No |
| 2 | Email | VARCHAR(64) |  | Email which is used for account registration. This can be understood as account name | No |
| 3 | Password | VARCHAR(16) |  | Password of user | No |
| 4 | FirstName | NVARCHAR(32) |  | User first name | No |
| 5 | LastName | NVARCHAR(32) |  | User last name | No |
| 6 | FullName | NVARCHAR(96) |  | Full name of user, this is the combination of First Name and Last Name. It is created for quick access. | No |
| 7 | Birthday | FLOAT |  | Time when user was born. Its format is UTC. | No |
| 8 | Phone | VARCHAR(15) |  | User phone number | Yes |
| 9 | Gender | TINYINT |  | User gender | No |
| 10 | Role | TINYINT |  | Role in the system: Admin, patient or doctor | No |
| 11 | Created | FLOAT |  | Time when account was signed up | No |
| 12 | LastModified | FLOAT |  | Time when the account information was lastly modified | Yes |
| 13 | Status | TINYINT |  | Status of account : pending, disabled or active | No |
| 14 | Address | NVARCHAR(128) |  | Address where user lives or works | Yes |
| 15 | PhotoUrl | NVARCHAR(MAX) |  | Relative path of user photo, this is used for render hyperlink. | Yes |
| 16 | PhotoPhysicPath | NVARCHAR(MAX) |  | The physical path where photo is stored, this is used for file access. | Yes |

#### 4.2.2. Patient

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about patients | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK/FK | The id of patient. This is the same as person’s id | No |
| 2 | Height | FLOAT |  | Height of user | Yes |
| 3 | Weight | FLOAT |  | User weight | Yes |

#### 4.2.3. Doctor

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about doctors | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK/FK | The id of doctor. This is the same as person’s id | No |
| 2 | Rank | FLOAT |  | Doctor rank, this indicates how good the doctor is | Yes |
| 3 | SpecialtyId | INT | FK | Id of doctor specialty | No |
| 4 | Voters | INT |  | Number of people who voted for this doctor | Yes |
| 5 | PlaceId | INT | FK | Id of place where doctor lives or works | No |
| 6 | ProfileUrl | NVARCHAR(MAX) |  | Url of doctor profile file. | Yes |
| 7 | ProfilePhysicalPath | NVARCHAR(MAX) |  | Storage location on the server where real file is stored. | Yes |

#### 4.2.4. Place

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about doctors | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK/FK | The id of doctor. This is the same as person’s id | No |
| 2 | City | FLOAT |  | City name. | No |
| 3 | Country | INT | FK | Country name | No |

#### 4.2.5. Specialty

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about medical specialty. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id which is used for identifying specialties. | No |
| 2 | Name | NVARCHAR(32) |  | Specialty name | No |

#### 4.2.6. Diary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about medical diary. Diary is a personal of doctor about his/her one specific patient. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Diary unique index. | No |
| 2 | Owner | INT | FK | Id of doctor who owns the diary. | No |
| 3 | Target | INT | FK | Id of patient who diary notes about. | No |
| 4 | Time | FLOAT |  | Time when the diary was noted. | No |
| 5 | Note | NVARCHAR(512) |  | Diary note. | No |
| 6 | Created | FLOAT |  | Time when the diary was created. | No |
| 7 | LastModified | FLOAT |  | Time when the diary was lastly modified. | No |

#### 4.2.7. Rating

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about rating: who made the rating, who was rated, … | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Maker | INT | PK/FK | Id of patient who made the rating, with Target, they combine to one primary key. | No |
| 2 | Target | INT | PK/FK | Id of doctor who was rated, with Maker, they combine to one primary key. | No |
| 3 | Value | INT |  | Rating point. | No |
| 4 | Comment | FLOAT |  | Comment of patient when rating was made. | No |
| 5 | Created | FLOAT |  | When the rating was created. | No |

#### 4.2.8. Relationship

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores general information about relationships between patients and doctors. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of relationship. | No |
| 2 | Source | INT | FK | Id of patient who started the relationship. | No |
| 3 | Target | INT | FK | Id of doctor who accepted the relationship. | No |
| 4 | Created | FLOAT |  | When the relationship was created. | No |

#### 4.2.9. RelationshipRequest

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores relationship requests information between patients and doctors. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of relationship. | No |
| 2 | Source | INT | FK | Id of patient who started the relationship request. | No |
| 3 | Target | INT | FK | Id of doctor who accepted the relationship request. | No |
| 4 | Content | NVARCHAR(128) |  | Message sent from patient to doctor about this request. | No |
| 5 | Created | FLOAT |  | When the relationship was sent. | Yes |

#### 4.2.10. Allergy

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of allergies which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of allergy note. | No |
| 2 | Owner | INT | FK | Id of patient who owns the allergy note. | No |
| 3 | Name | INT | FK | Name of allergy. | No |
| 4 | Cause | NVARCHAR(128) |  | Which patient is allergic with. | No |
| 5 | Note | FLOAT |  | Note information of allergy. | Yes |
| 6 | Created | FLOAT |  | When the note was created. | No |
| 7 | LastModified | FLOAT |  | When the note was lastly modified. | Yes |

#### 4.2.11. Addiction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of addictions which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of addiction note. | No |
| 2 | Owner | INT | FK | Id of patient who owns the addiction note. | No |
| 3 | Cause | NVARCHAR(128) |  | Which patient is addictive with. | No |
| 4 | Note | FLOAT |  | Note information of addiction. | Yes |
| 5 | Created | FLOAT |  | When the note was created. | No |
| 6 | LastModified | FLOAT |  | When the note was lastly modified. | Yes |

#### 4.2.12. BloodPressure

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of blood pressure measurements which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of blood pressure note. | No |
| 2 | Owner | INT | FK | Id of patient who owns the note. | No |
| 3 | Systolic | INT |  | Max pressure of blood. | No |
| 4 | Diastolic | INT |  | Min pressure of blood. | No |
| 5 | Time | FLOAT |  | When the measurement was done. | No |
| 6 | Note | FLOAT |  | Note information of addiction. | Yes |
| 7 | Created | FLOAT |  | When the note was created. | No |
| 8 | LastModified | FLOAT |  | When the note was lastly modified. | Yes |

#### 4.2.13. Heartbeat

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of heartbeat measurements which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of blood pressure note. | No |
| 2 | Owner | INT | FK | Id of patient who owns the note. | No |
| 3 | Rate | FLOAT |  | Heart rate at the time measurement was done. | No |
| 4 | Time | FLOAT |  | When the measurement was done. | No |
| 5 | Note | FLOAT |  | Note information of addiction. | Yes |
| 6 | Created | FLOAT |  | When the note was created. | No |
| 7 | LastModified | FLOAT |  | When the note was lastly modified. | Yes |

#### 4.2.14. BloodSugar

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of blood sugar measurements which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of blood sugar note. | No |
| 2 | Owner | INT | FK | Id of patient who owns the note. | No |
| 3 | Value | FLOAT |  | Blood sugar level at the time measurement was done. | No |
| 4 | Time | FLOAT |  | When the measurement was done. | No |
| 5 | Note | FLOAT |  | Note information of addiction. | Yes |
| 6 | Created | FLOAT |  | When the note was created. | No |
| 7 | LastModified | FLOAT |  | When the note was lastly modified. | Yes |

#### 4.2.15. MedicalRecord

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of medical records which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of medical record. | No |
| 2 | Owner | INT | FK | Who owned the record. | No |
| 3 | Creator | INT | FK | Who created medical record. | No |
| 4 | Category | INT |  | Category which medical record belongs to. | No |
| 5 | Name | NVARCHAR(64) |  | Name of medical record. | Yes |
| 6 | Info | NVARCHAR(MAX) |  | Key-value of necessary information.I.e : “Eye”:”good” | Yes |
| 7 | Time | FLOAT |  | When the record was about. | Yes |
| 8 | Created | FLOAT |  | When the record was created. | No |
| 9 | LastModified | FLOAT |  | When the note was lastly modified. | Yes |

#### 4.2.16. Prescription

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of prescriptions which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of prescription. | No |
| 2 | Owner | INT | FK | Who owns the prescription. | No |
| 3 | Creator | INT | FK | Who created prescription. | No |
| 4 | MedicalRecordId | INT | FK | Id of medical record which prescription belongs to. | No |
| 5 | From | FLOAT |  | When the prescription should be available. | No |
| 6 | To | FLOAT |  | When the prescription should be expired | No |
| 7 | Name | NVARCHAR(32) |  | Name of prescription | No |
| 8 | Medicine | NVARCHAR(MAX) |  | Stores serialized information of medicine. Refer to API documentation for more detail (api/medical/prescription) | Yes |
| 9 | Note | NVARCHAR(128) |  | Note of prescription. | Yes |
| 10 | Created | FLOAT |  | When the prescription was created. | No |
| 11 | LastModified | FLOAT |  | When the prescription was lastly modified. | Yes |

#### 4.2.17. PrescriptionImage

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of prescription images which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of prescription image. | No |
| 2 | Owner | INT | FK | Who owns the prescription image. | No |
| 3 | Creator | INT | FK | Who created prescription image. | No |
| 4 | MedicalRecordId | INT | FK | Id of medical record which prescription image belongs to. | No |
| 5 | PrescriptionId | INT | FK | Id of prescription which prescription image belongs to. | No |
| 6 | Image | NVARCHAR(32) |  | Name of image. | No |
| 7 | FullPath | NVARCHAR(MAX) |  | Physical path where image is stored on server. | No |
| 8 | Created | FLOAT |  | When the prescription image was created. | No |
| 9 | LastModified | FLOAT |  | When the prescription image was lastly modified. | Yes |

#### 4.2.18. MedicalNote

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of medical notes which belong to patients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of prescription image. | No |
| 2 | Owner | INT | FK | Who owns the prescription image. | No |
| 3 | Creator | INT | FK | Who created prescription image. | No |
| 4 | MedicalRecordId | INT | FK | Id of medical record which note belongs to. | No |
| 5 | Note | NVARCHAR(128) |  | Note of medical record. | No |
| 6 | Time | INT |  | Time when the note was about. | No |
| 7 | Created | FLOAT |  | When the note was created. | No |
| 8 | LastModified | FLOAT |  | When the note was lastly modified. | Yes |

#### 4.2.19. MedicalCategory

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of medical category which medical records belong to. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of prescription image. | No |
| 2 | Name | NVARCHAR(32) |  | Name of medical category. | No |
| 3 | Created | FLOAT |  | When the category was created. | No |
| 4 | LastModified | FLOAT |  | When the category was lastly modified. | Yes |

#### 4.2.20. MedicalImage

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of medical image which are contained in medical records. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of .medical image. | No |
| 2 | MedicalRecordId | INT | FK | Id of medical record which medical image belongs to. | No |
| 3 | Owner | INT | FK | Id of person who owns the medical image. | No |
| 4 | Creator | INT | FK | Id of person who created medical image. | No |
| 5 | Image | NVARCHAR(32) |  | Name of image. | No |
| 6 | FullPath | NVARCHAR(MAX) |  | Physical path on server where image is stored. | No |
| 7 | Created | FLOAT |  | When medical image was created. | No |

#### 4.2.21. MedicalRecord

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of medical image which are contained in medical records. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of of experiment note. | No |
| 2 | MedicalRecordId | INT | FK | Id of medical record which medical record belongs to. | No |
| 3 | Owner | INT | FK | Id of person who owns the experiment note. | No |
| 4 | Creator | INT | FK | Id of person who created experiment note. | No |
| 5 | Name | INT |  | Name of experiment note. | No |
| 6 | Info | INT |  | Stores serialized information of experiment measurement. | No |
| 7 | Time | INT |  | Time when the experiment note is about. | No |
| 8 | Created | FLOAT |  | When medical image was created. | No |
| 9 | LastModified | FLOAT |  | When medical image was lastly modified. | No |

#### 4.2.22. Notification

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of notification which broadcasted from service. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of of notification. | No |
| 2 | Type | INT |  | Type of notification : create/edit/remove/… | No |
| 3 | Topic | INT |  | Topic of notification : Medical record/ Prescription/… | No |
| 4 | Container | INT |  | Id of the container (Medical Record ID, Prescription ID, …). | No |
| 5 | ContainerType | INT |  | Type of container (Medical Record, Prescription, …). | No |
| 6 | Broadcaster | INT |  | Who causes the notification broadcasted. | No |
| 7 | Recipient | INT |  | Who should receive the notification. | No |
| 8 | Record | FLOAT |  | Id of record which is targeted. | No |
| 9 | Message | FLOAT |  | Message from service. | No |
| 10 | Created | FLOAT |  | When the notification is created. | No |
| 11 | IsSeen | FLOAT |  | Whether recipient has seen the notification or not. | No |

#### 4.2.23. RealtimeConnection

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of connections made to service, the connections information will be used by service to broadcast realtime notification to clients. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of of realtime connection. | No |
| 2 | Owner | INT |  | Id of person who connects to service. | No |
| 3 | ConnectionId | INT |  | Connection index which generated by service. | No |
| 4 | Created | INT |  | Time when the note was created. | No |

#### 4.2.24. Message

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of message which are broadcasted on server. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK | Id of of message. | No |
| 2 | Broadcaster | INT | FK | Id of person who broadcasted the message. | No |
| 3 | Recipient | INT | FK | Id of person who receives the message. | No |
| 4 | Content | NVARCHAR(512) |  | Message content. | No |
| 5 | Created | FLOAT |  | Time when the message was broadcasted. | No |
| 6 | IsSeen | BIT |  | Whether message has been seen or not. | No |

#### 4.2.25. AccountToken

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Stores information of tokens which are used for account activation or account password modification. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Owner | INT | PK/FK | Id of patient who has the code. | No |
| 2 | Code | NVARCHAR(36) |  | Code for account activation/account password modification. | No |
| 3 | Type | INT | PK/FK | Id of person who receives the message. | No |
| 4 | Expired | DATETIME |  | Time when the account token is expired. | No |

#### 4.2.26. Appointment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Appointment information which is broadcasted from patients to doctors and vice versa. | | | | | |
| **No.** | **Field** | **Type** | **Key** | **Description** | **Null** |
| 1 | Id | INT | PK/FK | Appointment index. | No |
| 2 | Maker | INT | FK | Id of person who made the appointment request. | No |
| 3 | Dater | INT | FK | Id of person who received the appointment request. | No |
| 4 | From | FLOAT |  | Time when the appointment should be started. | No |
| 5 | To | FLOAT |  | Time when the appointment should be emded. | No |
| 6 | Note | NVARCHAR(128) |  | Note of appointment | No |
| 7 | Created | FLOAT |  | Time when the appointment was created. | No |
| 8 | LastModified | FLOAT |  | Time when the appointment was lastly modified. | No |
| 9 | LastModifiedNote | NVARCHAR(128) |  | Note of the time appointment was lastly modified. | Yes |
| 10 | Status | TINYINT |  | Status of appointment | No |

## References

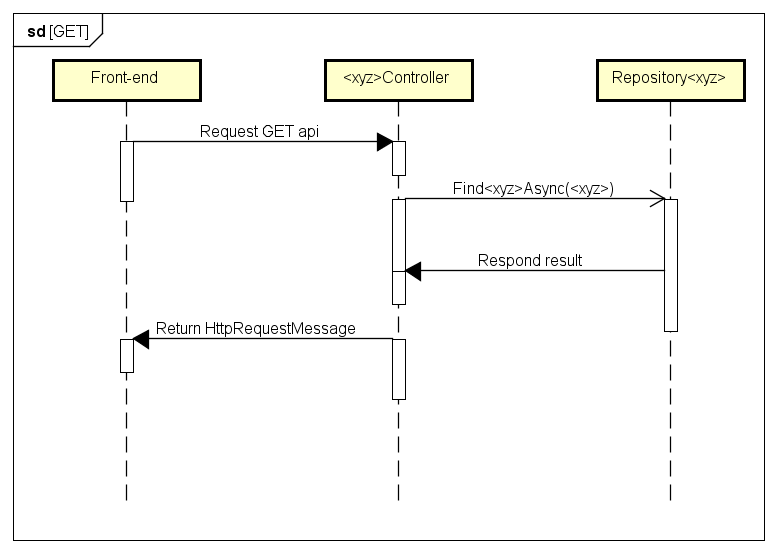
## Detail design

### 6.1. General sequence diagram

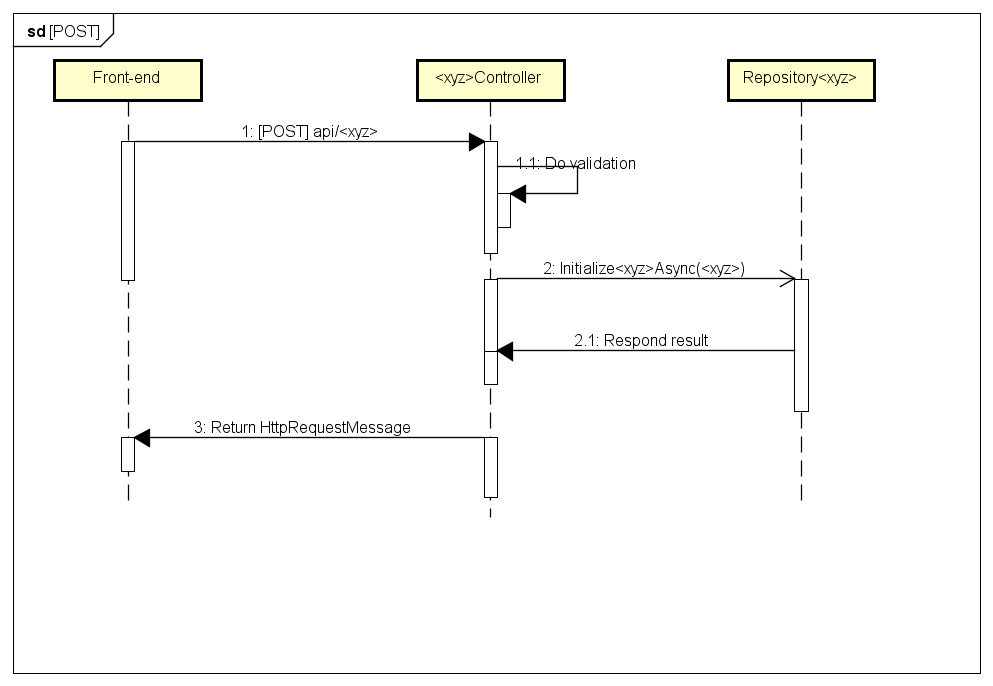
Olives back-end services and front-end applications all have some general sequence diagram which will be used by Components Assemblers in development stage. As in a specific function, sequence diagram is not defined, that function should follows the general sequence diagram.

#### 6.1.1 Olives back-end general sequence diagrams

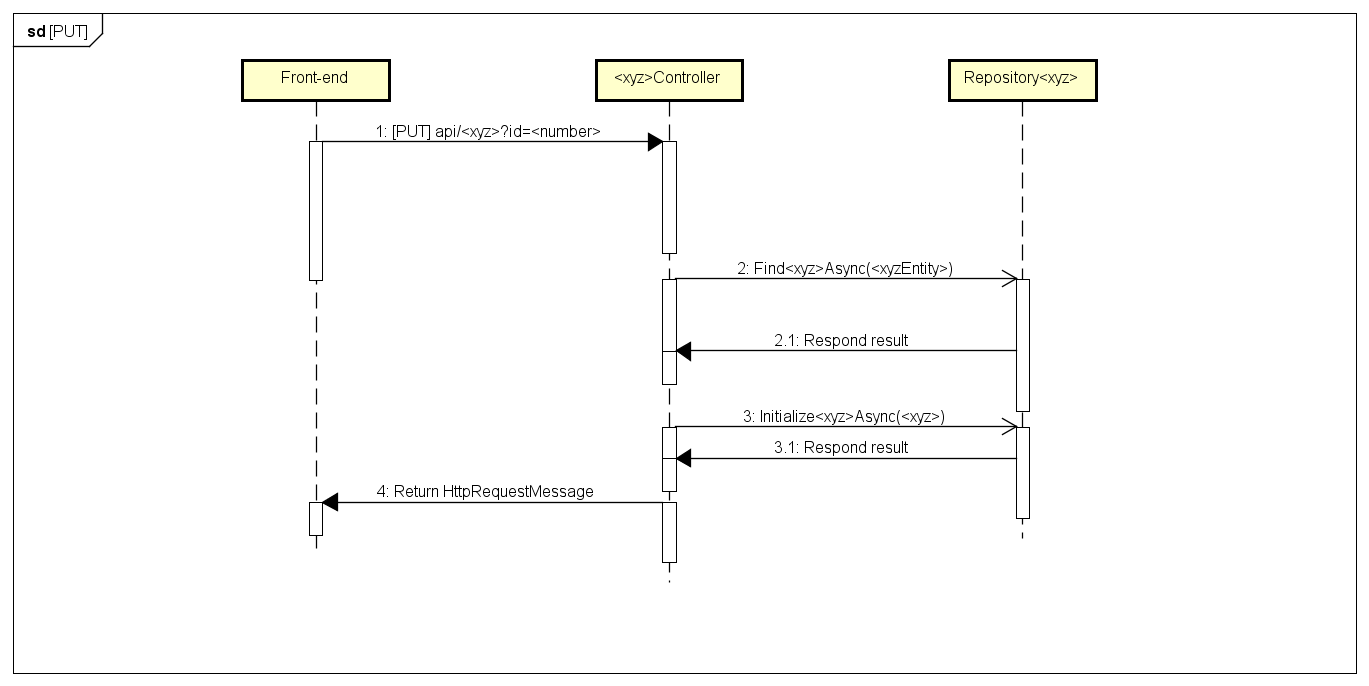
##### 6.1.1.1. Find record



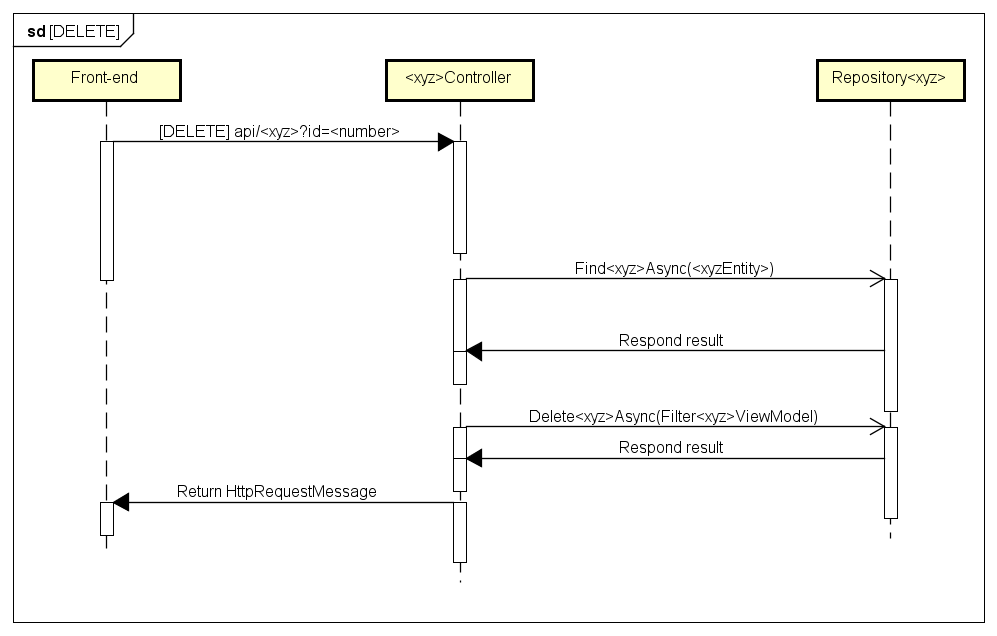
##### 6.1.1.2. Initialize record



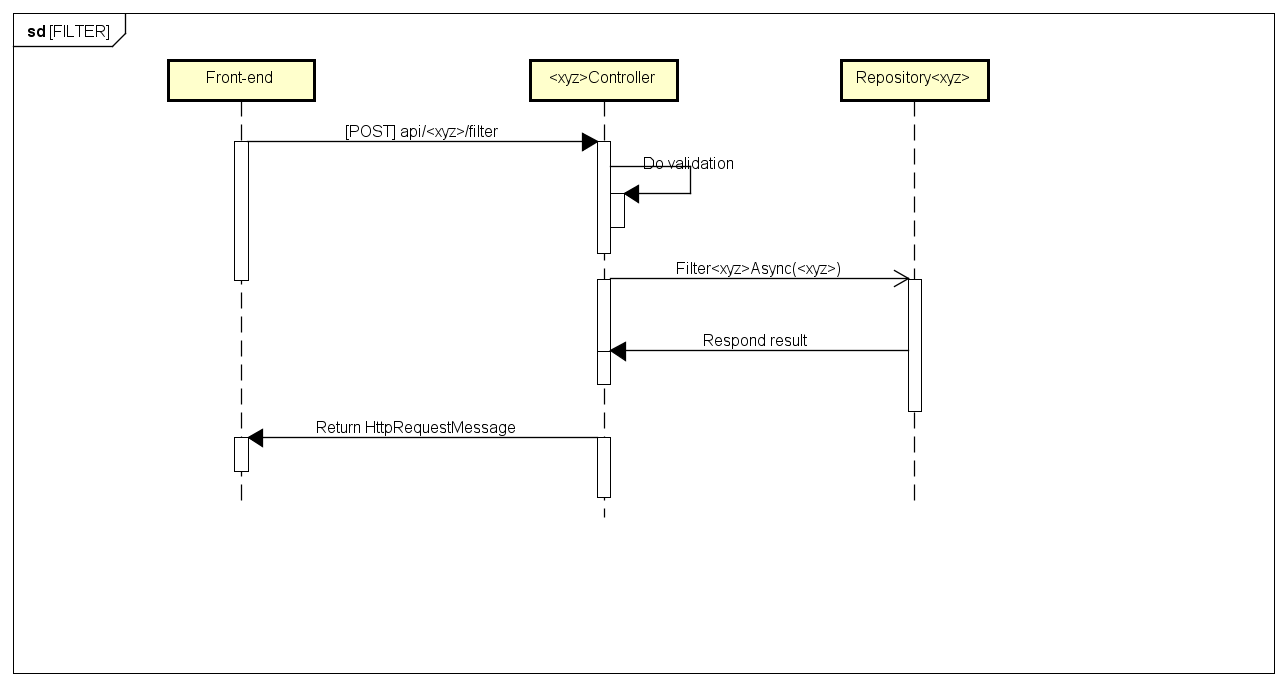
##### 6.1.1.3. Edit record



##### 6.1.1.4. Delete record



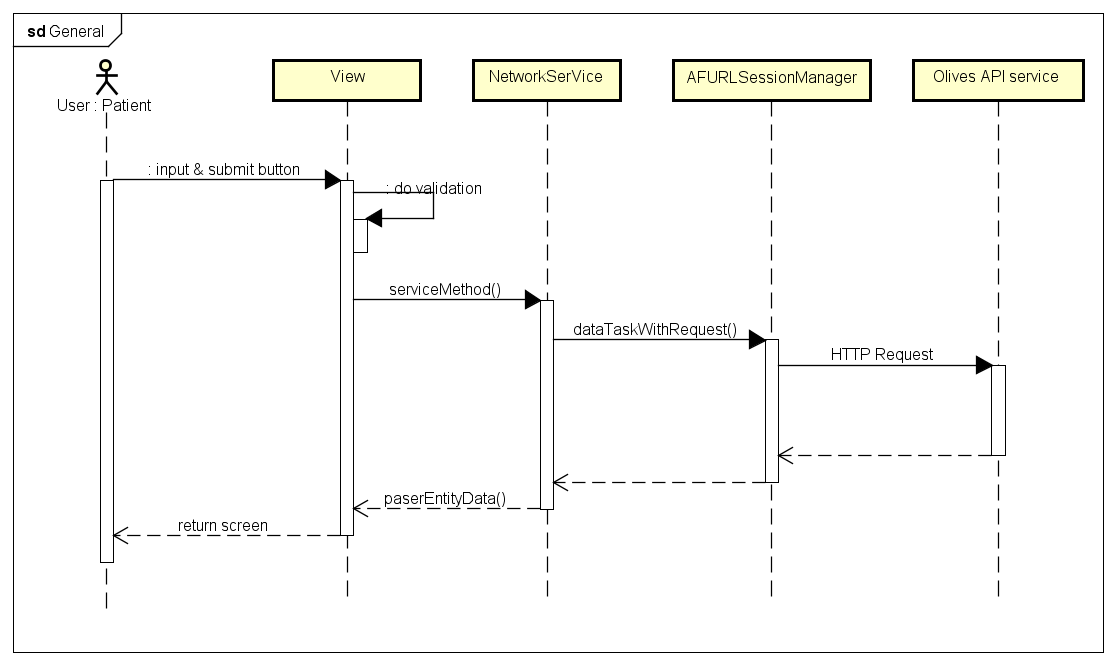
##### 6.1.1.5. Filter record



#### 6.1.2. Olives front-end general sequence diagrams (Patient application)

##### 6.1.2.1. Non-medical records data sequence

This type of sequence is for data which doesn’t belong to these types: Medical Record, Experiment Note, Medical Note, Medical Category, Medical Image, Prescription, Prescription Image.

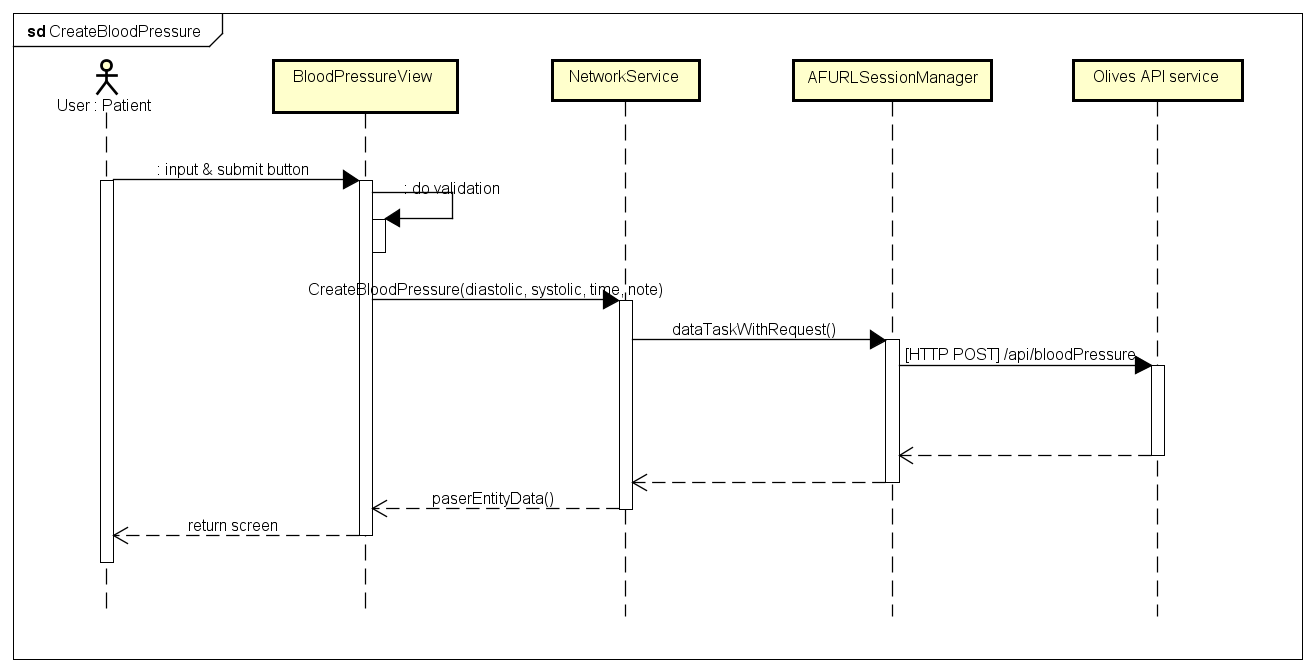


##### 6.1.2.2. Medical records data sequence

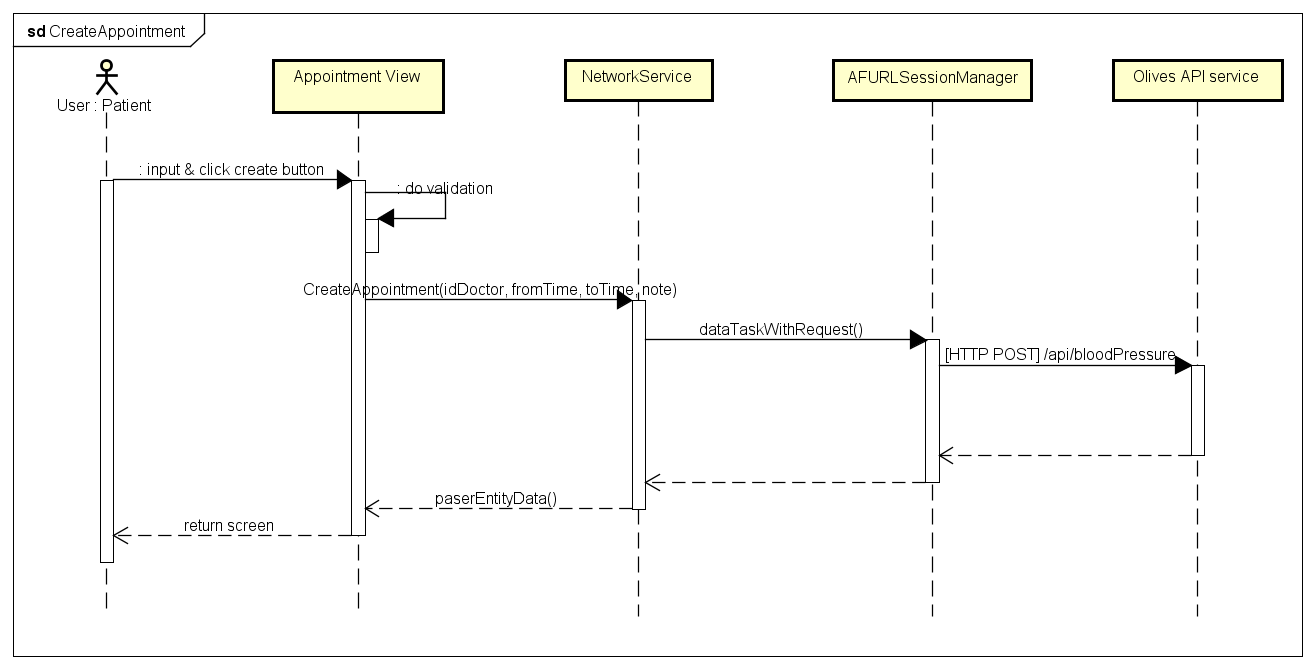
This type of sequence is for data which belongs to those excluded data types above.

#### 6.1.3. Example of ordinary sequence diagrams

##### 6.1.3.1. Initialize blood pressure note



##### 6.1.3.3. Initialize appointment

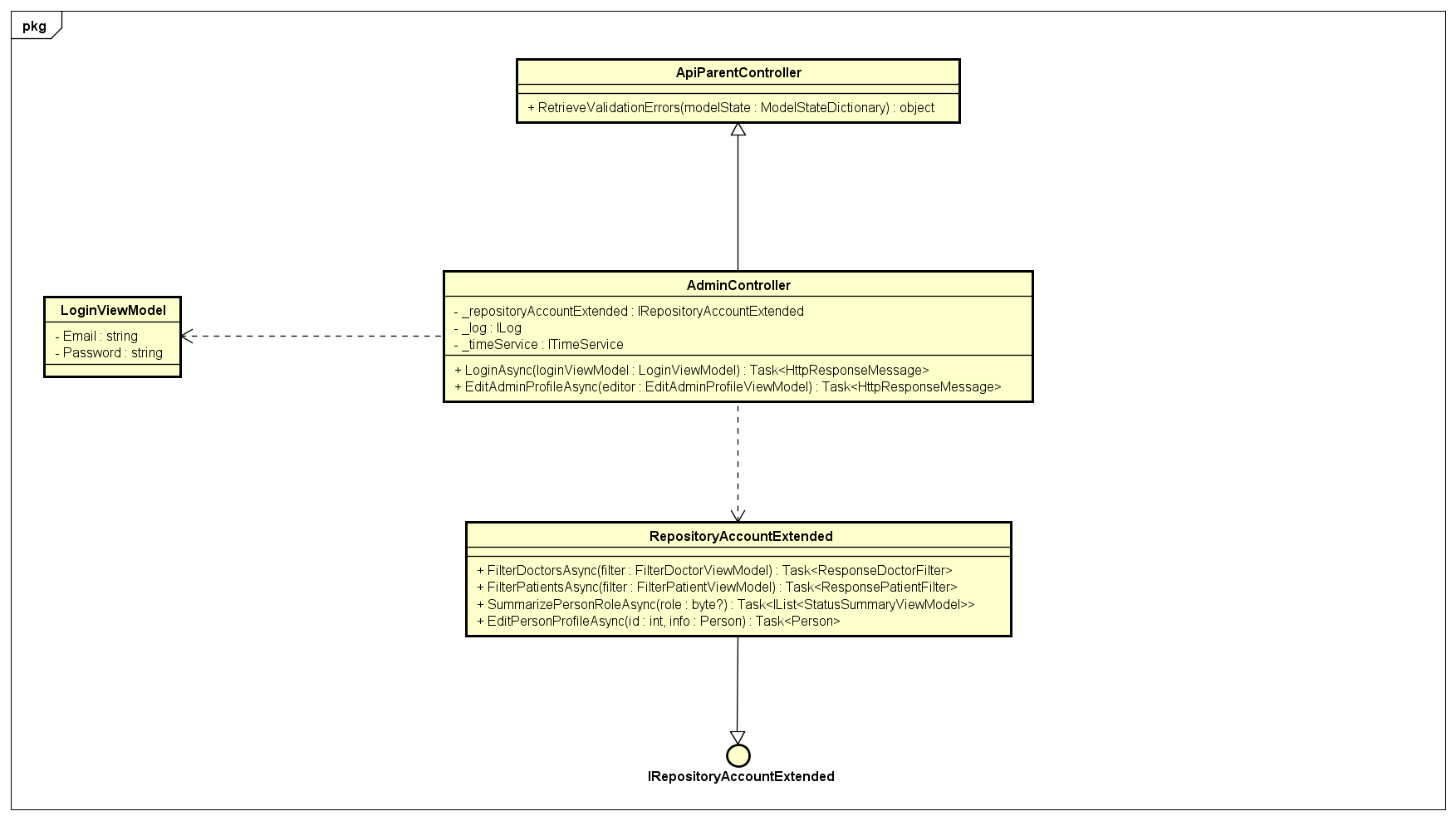


### 6.2. Detail design & use case

#### 6.2.1. Olives Administration sub-system

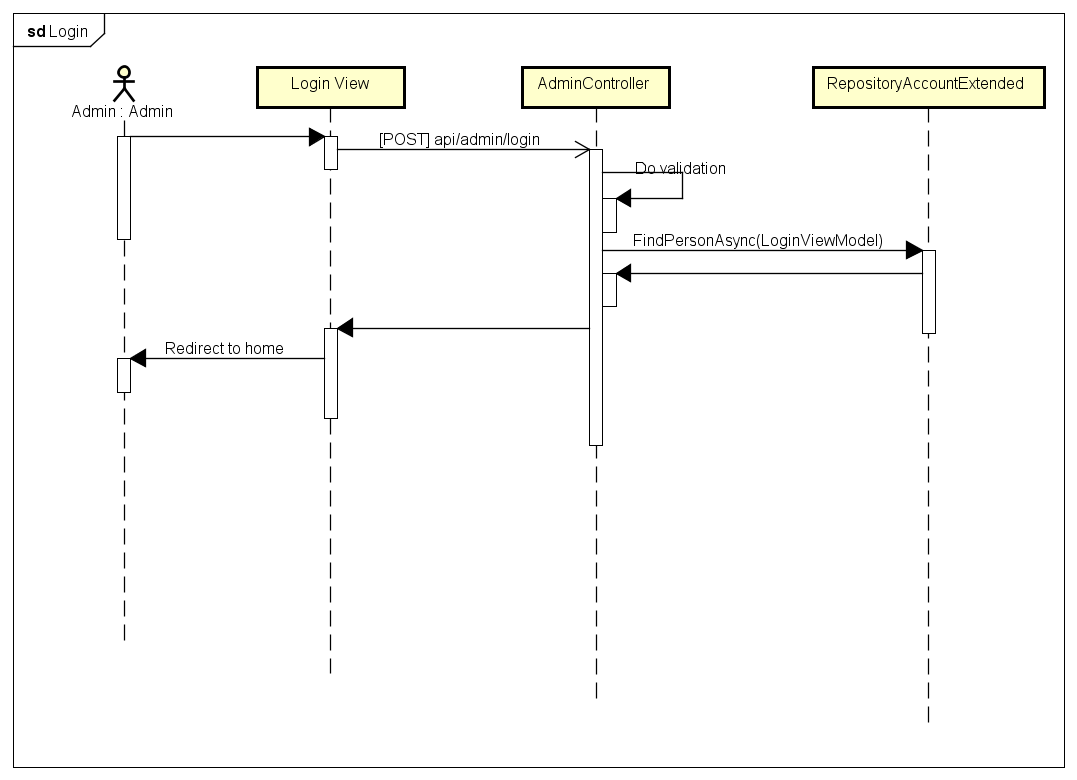
##### 6.2.1.1. Login

###### a. Class diagram



Class diagram of Admin Login.

###### b. Sequence diagram



###### c. Class definition

ApiParentController class

*[Method]RetrieveValidationErrors()*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the request parameter model and do the validation and construct a list of validation errors as available which should be seen by clients. | | | |
| Return | **Type** | | | **Description** |
| object | | | Validation error response. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | modelState | ModelStateDictionary | ModelState of MVC/Web API which contains request parameters. |

AdminController class

*Attributes*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Type | Description |
| 1 | \_repositoryAccountExtended | var | Instance of interface using to call methods in IRepositoryAccountExtended. |
| 2 | \_log | var | Instance of interface using to call methods in ILog. |
| 3 | \_timeService | var | Instance of interface using to call methods in ITimeService. |

*[Method]LoginAsync()*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the parameters come from clients, do validation and log user into system as the information is valid. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Response message which is thrown back to front-end to handle the processed result. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | loginViewModel | LoginViewModel | A view model contains Email and Password which are used for logging into system. |

*[Method]EditAdminProfileAsync()*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the parameters come from clients, do validation and edit the profile of admin as the information is valid. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Response message which is thrown back to front-end to handle the processed result. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | editor | EditAdminProfileViewModel | A view model contains the information which is used for changing information in profile. |

RepositoryAccountExtended class

*Attributes*

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Type | Description |
| 1 | \_dataContext | var | Instance of interface using to call methods in IOliveDataContext. |

*[Method]FilterDoctorAsync()*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the request parameters, filter and return a list of condition matched doctors. | | | |
| Return | **Type** | | | **Description** |
| ResponseDoctorFilter | | | Contains a filtered results list and a total record counter. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | filter | FilterDoctorViewModel | Model contains the conditions which are used for filtering. |

*[Method]FilterPatientsAsync()*

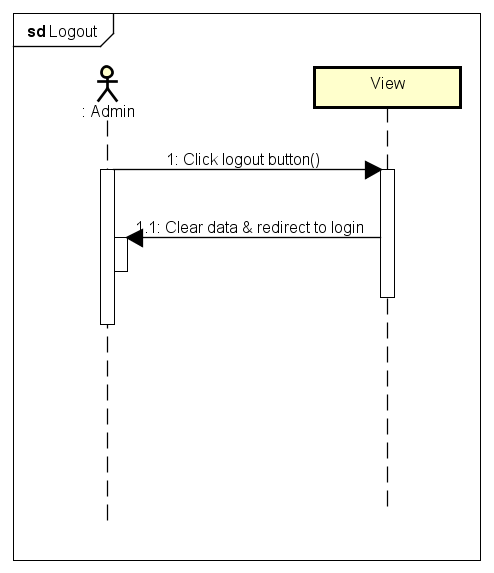
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the request parameters, filter and return a list of condition matched patients. | | | |
| Return | **Type** | | | **Description** |
| ResponsePatientFilter | | | Contains a filtered results list and a total record counter. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | filter | FilterPatientViewModel | Model contains the conditions which are used for filtering. |

*[Method]SummarizePersonRoleAsync()*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the request parameters, filter and return a list of condition matched people with role statistic. | | | |
| Return | **Type** | | | **Description** |
| IList<StatusSummaryViewModel> | | | Contains a filtered results list and a total record counter. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | role | Byte? | Role which is used for determining the filter. |

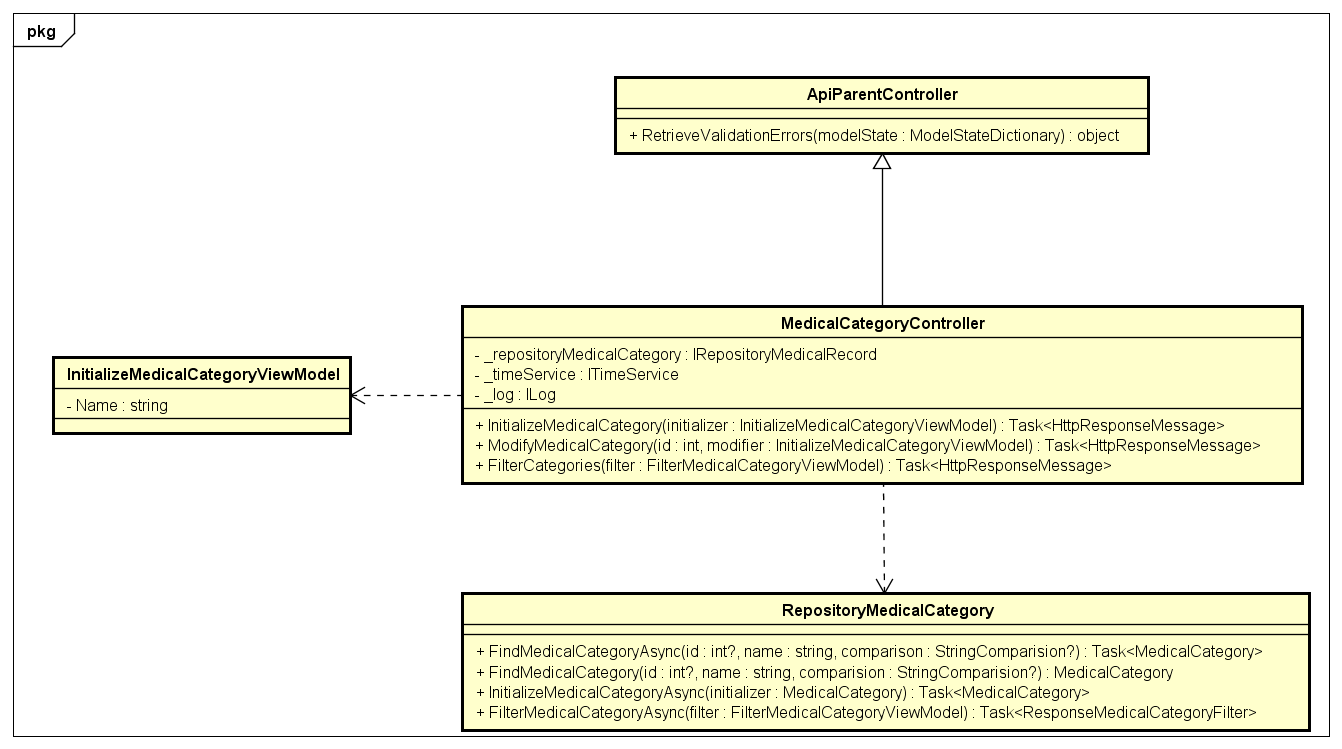
##### 6.2.1.2. Logout

1. Sequence diagram



##### 6.2.1.3. Create medical category

1. Class diagram



1. Class definition

MedicalCategoryController class

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Type | Description |
| 1 | \_repositoryMedicalCategory | var | Instance of interface using to call methods in IRepositoryMedicalCategory. |
| 2 | \_timeService | var | Instance of interface using to call methods in ITimeService. |
| 3 | \_log | var | Instance of interface using to call methods in IRepositoryMedicalCategory. |

Method: InitializeMedicalCategory()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the request parameters, validate and initialize a medical category as the information is valid. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | initializer | InitializeMedicalCategoryViewModel | Model contains the information of medical record which will be used for initializing a medical category. |

Method: ModifyMedicalCategory()

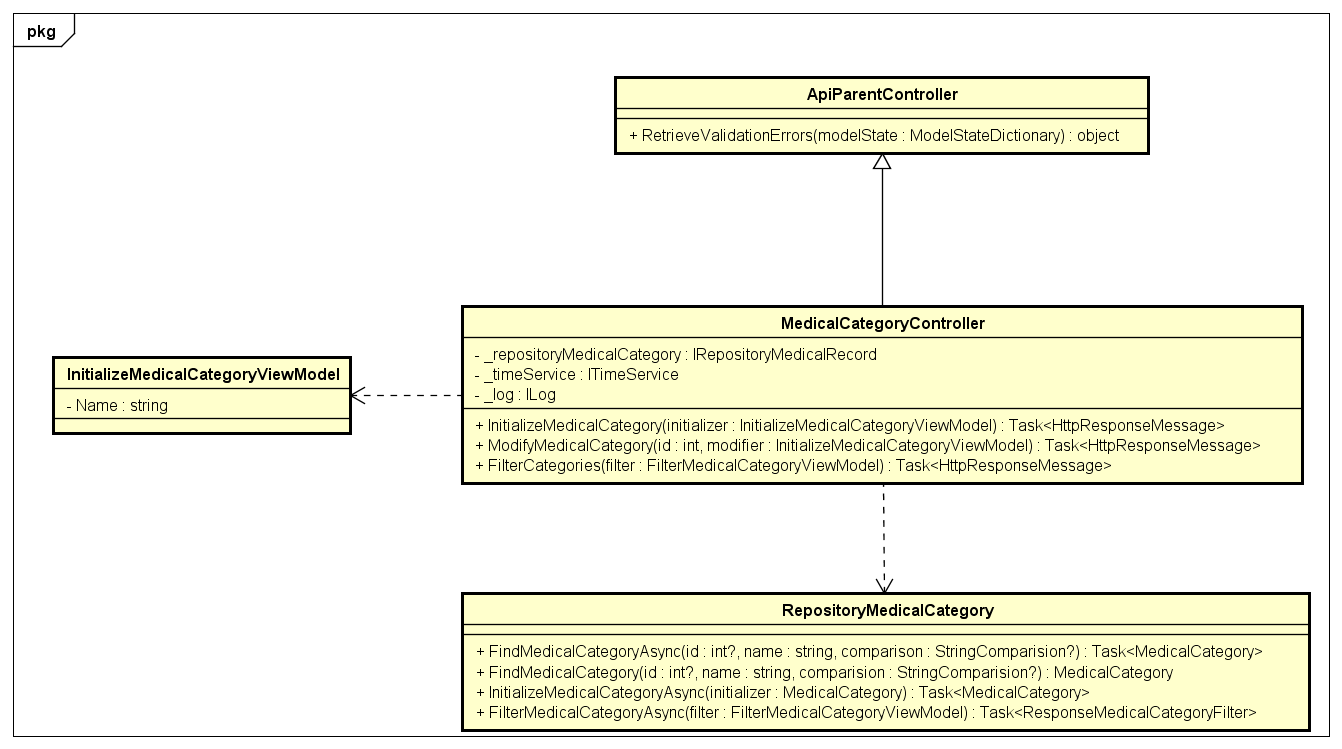
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the request parameters, validate and modify a medical category as the information is valid | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | id | int | Id of medical category which will be targeted to be edited. |
| 2 | modifier | InitializeMedicalCategoryViewModel | Model contains the information of medical record which will be used for modifying a medical category. |

Method: FilterCategories()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the request parameters, validate and filter and return the list of condition matched medical categories. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 2 | filter | FilterMedicalCategoryViewModel | Model contains the information which are used for medical categories filtering. |

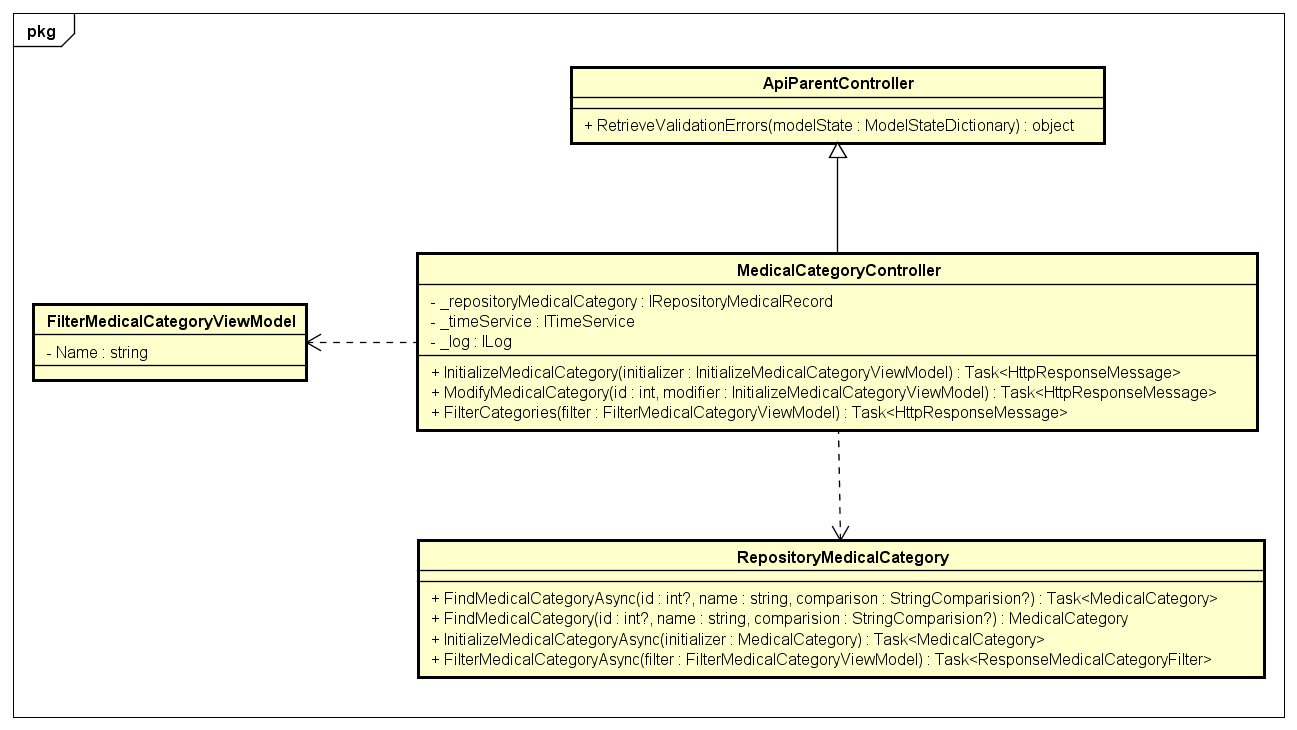
##### 6.2.1.4. Edit medical category

1. Class diagram



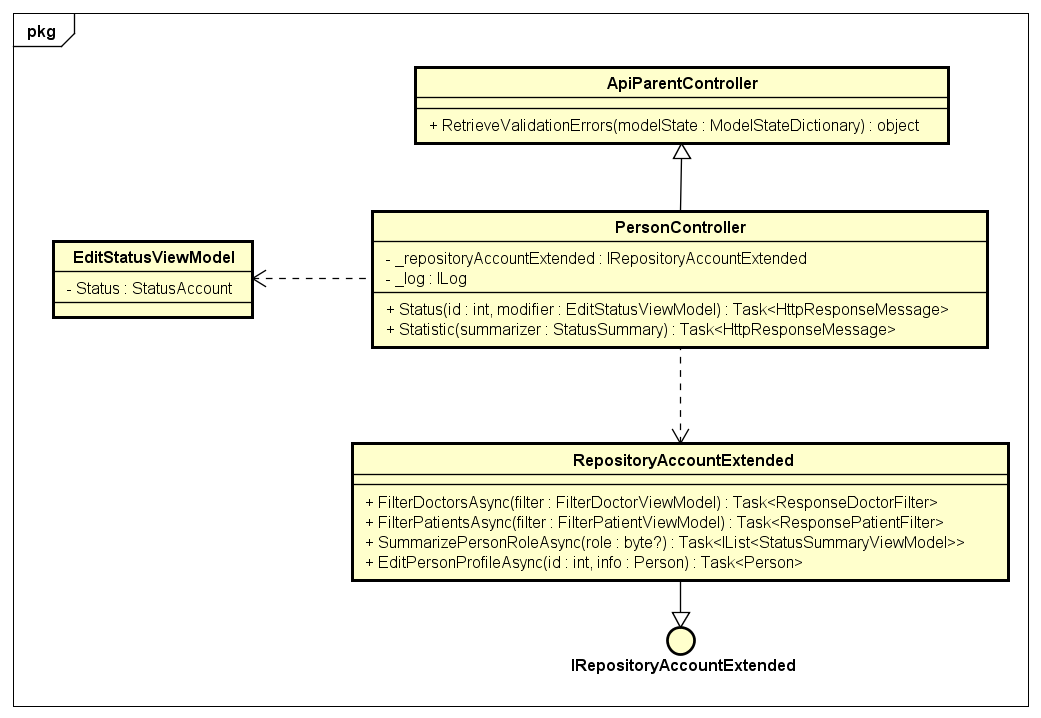
##### 6.2.1.5. Filter medical category

###### a. Class diagram

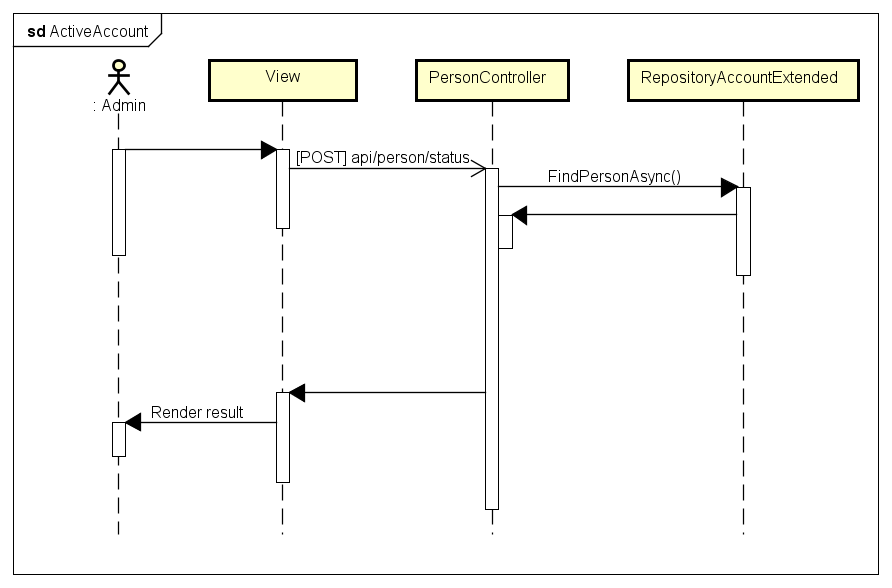


##### 6.2.1.6. Activate account

1. Class diagram

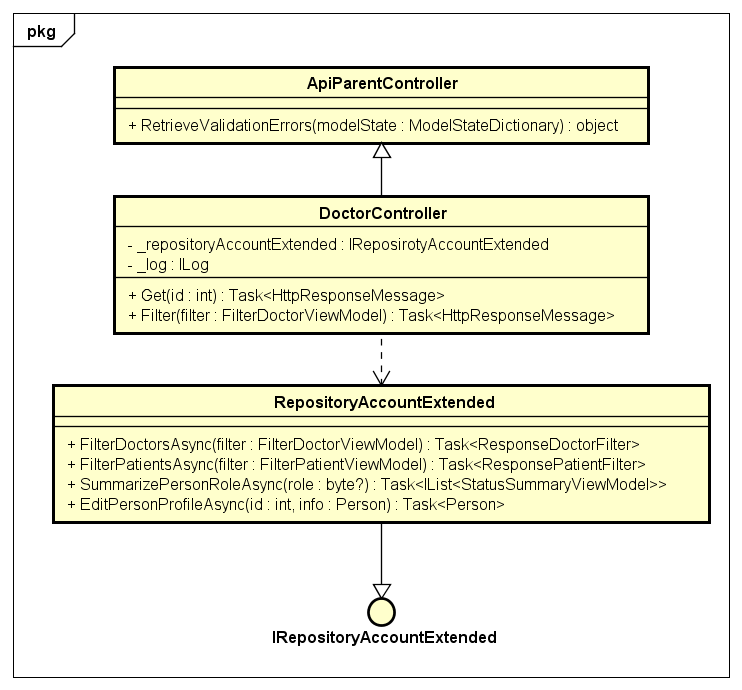


1. Sequence diagram



##### 6.2.1.7. Find doctor

1. Class diagram



1. Class definition

DoctorController class

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Type | Description |
| 1 | \_repositoryMedicalCategory | var | Instance of interface using to call methods in IRepositoryMedicalCategory. |
| 2 | \_timeService | var | Instance of interface using to call methods in ITimeService. |
| 3 | \_log | var | Instance of interface using to call methods in IRepositoryMedicalCategory. |

Method: Get()

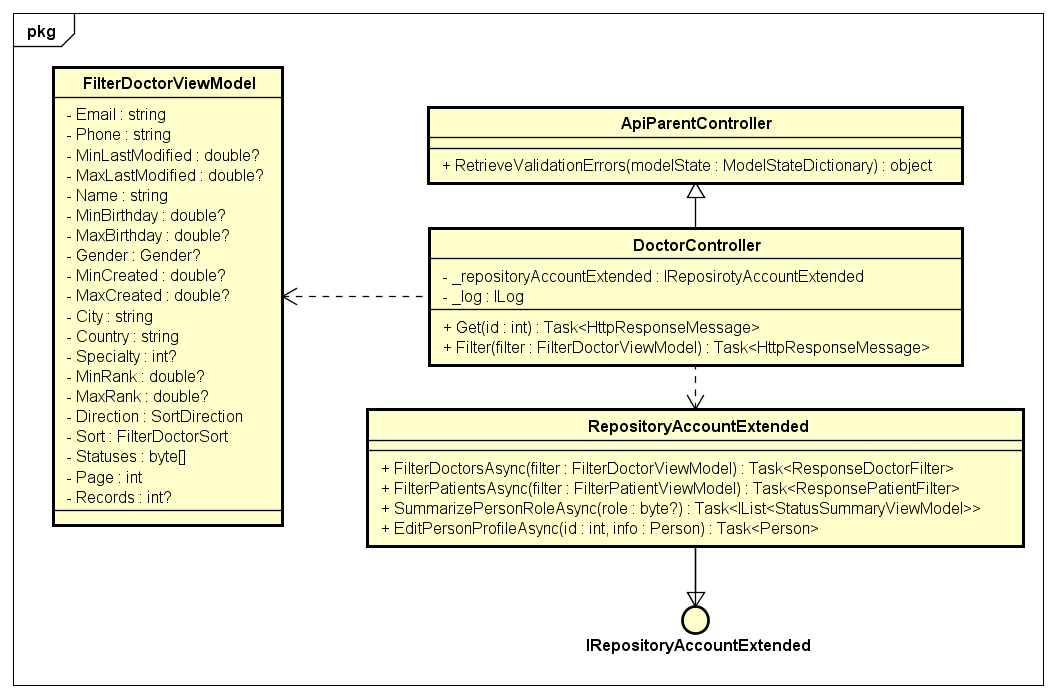
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the id sent from client and find the doctor by using id. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | id | int | Id of doctor. |

Method: Filter()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the id sent from client and find the doctor by using id. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | filter | FilterDoctorViewModel | Model contains the information for filter. |

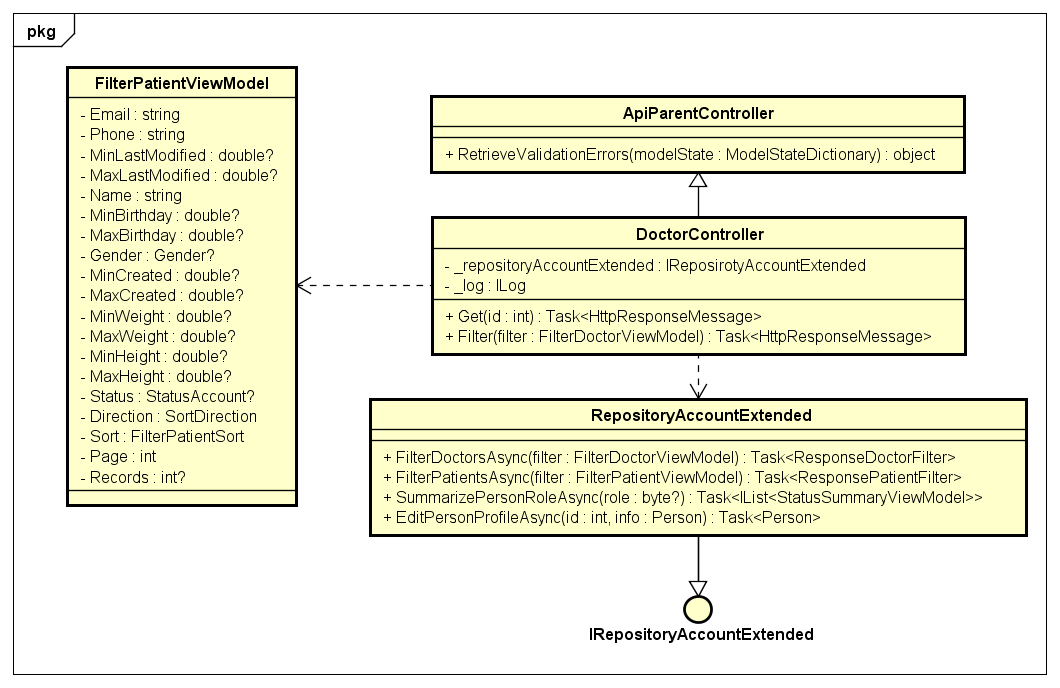
##### 6.2.1.8. Filter doctors

1. Class diagram



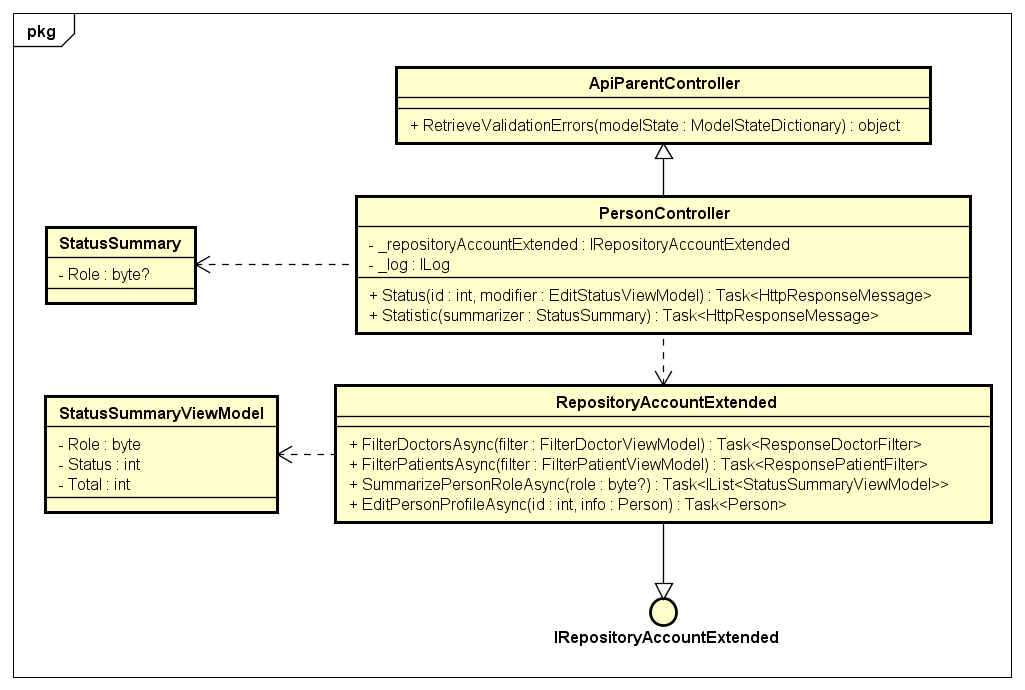
##### 6.2.1.9. Find patients

1. Class diagram

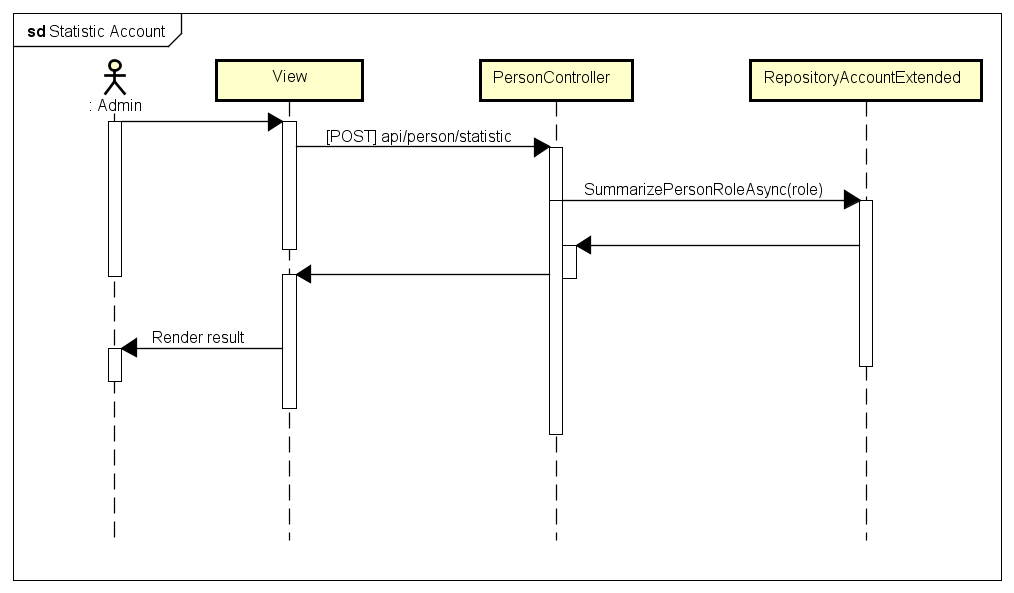


##### 6.2.1.10. Statistic accounts by role

1. Class diagram



1. Sequence diagram



1. Class diagram

PersonController class

Attribute

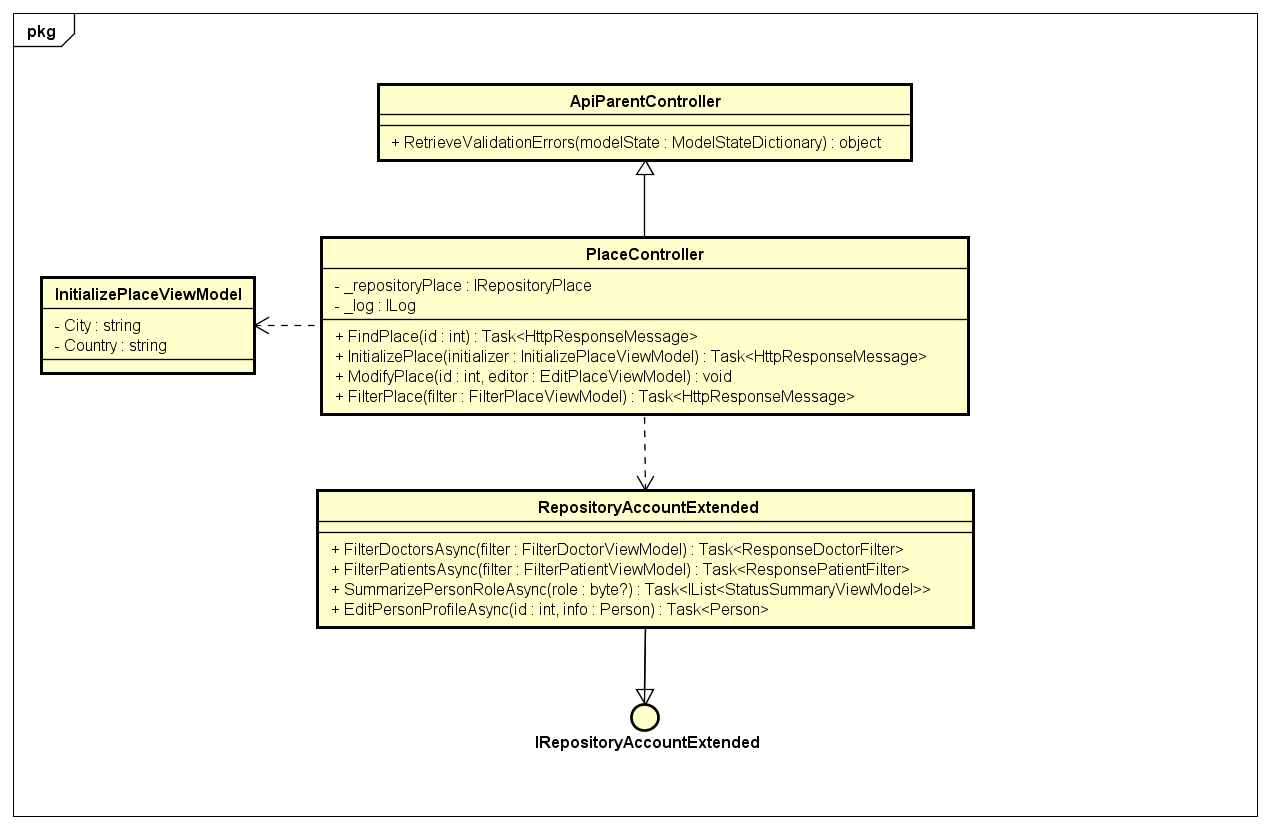
|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Type | Description |
| 1 | \_repositoryMedicalCategory | var | Instance of interface using to call methods in IRepositoryMedicalCategory. |
| 2 | \_timeService | var | Instance of interface using to call methods in ITimeService. |
| 3 | \_log | var | Instance of interface using to call methods in IRepositoryMedicalCategory. |

Method: Status()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the id sent from client and find the person by using id. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | id | int | Id of doctor. |
| 2 | modifier | EditStatusViewModel | Model contains information which is used for modifying person status. |

##### 6.2.1.11. Initialize place

1. Class diagram



1. Class definition

PlaceController class

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name | Type | Description |
| 1 | \_repositoryPlace | IRepositoryPlace | Instance of interface using to call methods in IRepositoryPlace. |
| 2 | \_log | ILog | Instance of interface using to call methods in ILog. |

Method: FindPlace()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the id sent from client and find the place by using id. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | id | int | Id of place. |

Method: InitializePlace()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the information requested from client and create a place in database as the informations is available. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | initializer | InitializePlaceViewModel | Model which contains place information for being initialized. |

Method: ModifyPlace()

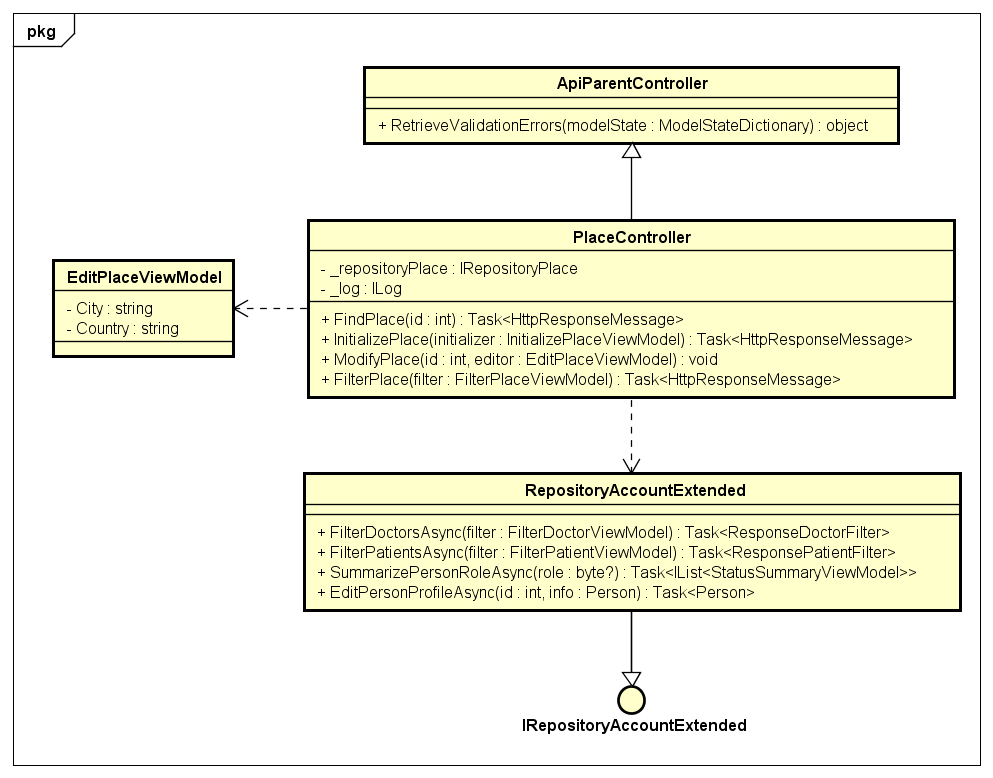
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the information requested from client, do validation and modify place information as the parameters are valid. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 1 | id | int | Id of place which needs modifying |
| 2 | initializer | EditPlaceViewModel | Collection of information which is used for replacing the old ones. |

Method: FilterPlace()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Purpose | Take the information requested from client, do validation filter places list as parameters are valid. | | | |
| Return | **Type** | | | **Description** |
| HttpResponseMessage | | | Contains the information which will be thrown back to the clients. |
| Parameter | **No** | **Name** | **Type** | **Description** |
| 2 | filter | FilterPlaceViewModel | Class which contains conditions to filter places. |

##### 6.2.1.12. Edit place

1. Class diagram



#### 6.2.2. Olives sub-system