# Group06

**ToeicExam**
**Software Architecture Document**

**Version <1.2>**

**Revision History**

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 22/06/2022 | 1.0 | Introduction, LogicalView, Use Case Model, View, Model | Huynh Ngoc Son<br>Doan Huu Loc<br>Nguyen Hoang Luan |
| 26/06/2022 | 1.1 | Controller, Service | Cao Nhat Duc<br>Nguyen Dang Manh Tu |
| 01/07/2022 | 1.2 | Fix View, Controller, Service, Model | Nguyen Hoang Luan<br>Huynh Ngoc Son<br>Cao Nhat Duc<br>Doan Huu Loc<br>Nguyen Dang Manh Tu |

**Table of Contents**

**1. Introduction ..... 5**

1.1 Purpose ..... 5

1.2 Scope ..... 5

1.3 Definitions, Acronyms and Abbreviations ..... 5

1.4 References ..... 5

1.5 Overview ..... 6

**2. Architectural Goals and Constraints ..... 6**

**3. Use-Case Model ..... 7**

**4. Logical View ..... 8**

4.1 Component: HomeView ..... 10

4.2 Component: ProfileView ..... 10

4.3 Component: MiniTestView ..... 11

4.4 Component: FullTestView ..... 11

4.5 Component: DiscussionForumView ..... 12

4.6 Component: BlogView ..... 12

4.7 Component: AccountView ..... 12

4.8 Component: ErrorView ..... 13

4.9 Component: ManageAccountView ..... 13

4.10 Component: ManageBlogView ..... 13

4.11 Component: ManageExamView ..... 13

4.12 Component: UserView ..... 14

4.13 Component: GuestView ..... 14

4.14 Component: AdminView ..... 15

4.15 Component: HomeController ..... 15

4.16 Component: FullTestController ..... 16

4.16 Component: MiniTestController ..... 16

4.17 Component: BlogController ..... 17

4.18 Component: DiscussionController ..... 17

4.19 Component: ProfileController ..... 18

4.20 Component: ErrorController ..... 18

4.21 Component: AccountController ..... 18

4.22 Component: UserController ..... 19

4.23 Component: GuestController ..... 19

4.24 Component: AdminController ..... 19

4.25 Component: AccountService ..... 20

4.26 Component: ExamService ..... 21
4.27 Component: SupportService ..... 21
4.28 Component: ListeningModel ..... 22
4.29 Component: ReadingModel ..... 22
4.30 Component: BlogModel ..... 22
4.31 Component: AccountModel ..... 23
4.32 Component: DicussionModel ..... 23
4.33 Component: DictionaryModel ..... 24
**5. Deployment ..... 24**
**6. Implementation View ..... 24**

**Software Architecture Document**

# 1. Introduction

## 1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

## 1.2 Scope

This document presents the technical architecture of the TOEIC exam system. In this document, we focus on the options offered for the TOEIC exam system. Following, readers will find information about the frameworks, tools, and technologies used by the TOEIC exam system.

## 1.3 Definitions, Acronyms and Abbreviations

| Abbreviations and Acronyms | |
| --- | --- |
| **Abbreviation** | **Meaning** |
| API | Application Programming Interface |
| HTTP | HyperText Transfer Protocol |
| MVC | Model View Controller |
| URL | Uniform Resource Locator |
| OOP | Object Oriented Programming |

## 1.4 References


- https://etendering.ted.europa.eu/document/document-file-download.html?docFileId=4901
- https://tsh.io/blog/how-to-document-your-architecture/
- https://blog.ndepend.com/software-architecture-document-dont-need/
- https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-package-diagram/
- https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-

class-diagram/
- https://stackoverflow.com/questions/5702391/mvcs-model-view-controller-service
- https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

### 1.5 Overview

The report will present a detailed analysis of the architecture of the Online TOEIC Examination system. Further sections cover the architectural representation of the project including architectural representation, architectural goals, and constraints, and use case realizations. The later sections cover the detailed specific details of the 4 main views ( logical view, deployment view, and implementation view ) of the system.

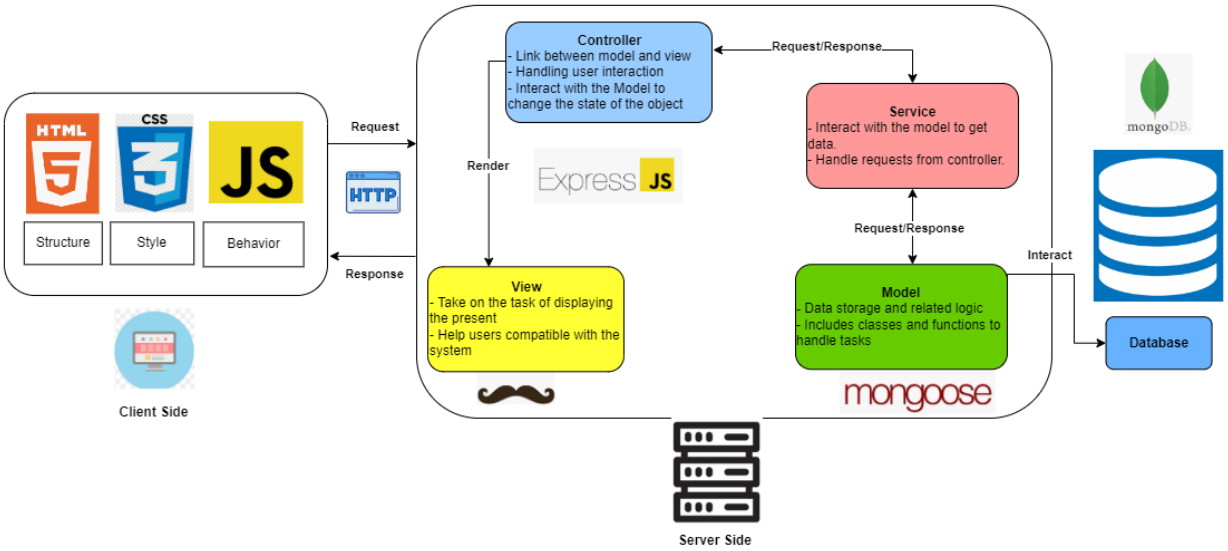
## 2. Architectural Goals and Constraints

There are some key requirements and system constraints that have a significant bearing on the architecture. There are:

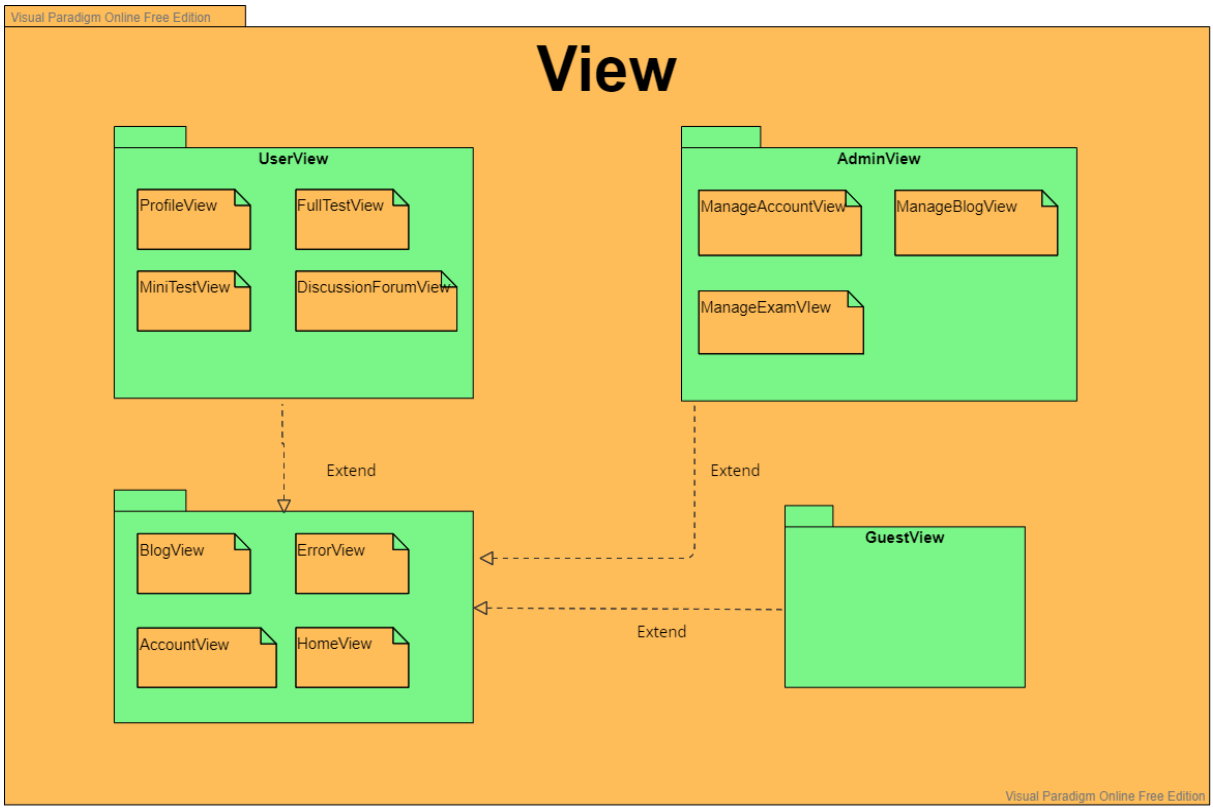
- The system works 24/7.
- The system will work on most operating systems.
- The system will work on most devices
- Friendly interface, easy to use for many ages
- User information is kept confidential
- The current site provides most of the content for display. An interface for this system must be able to handle heavy traffic.
- All functions must be available through commercially available web browsers.

## 3. Use-Case Model

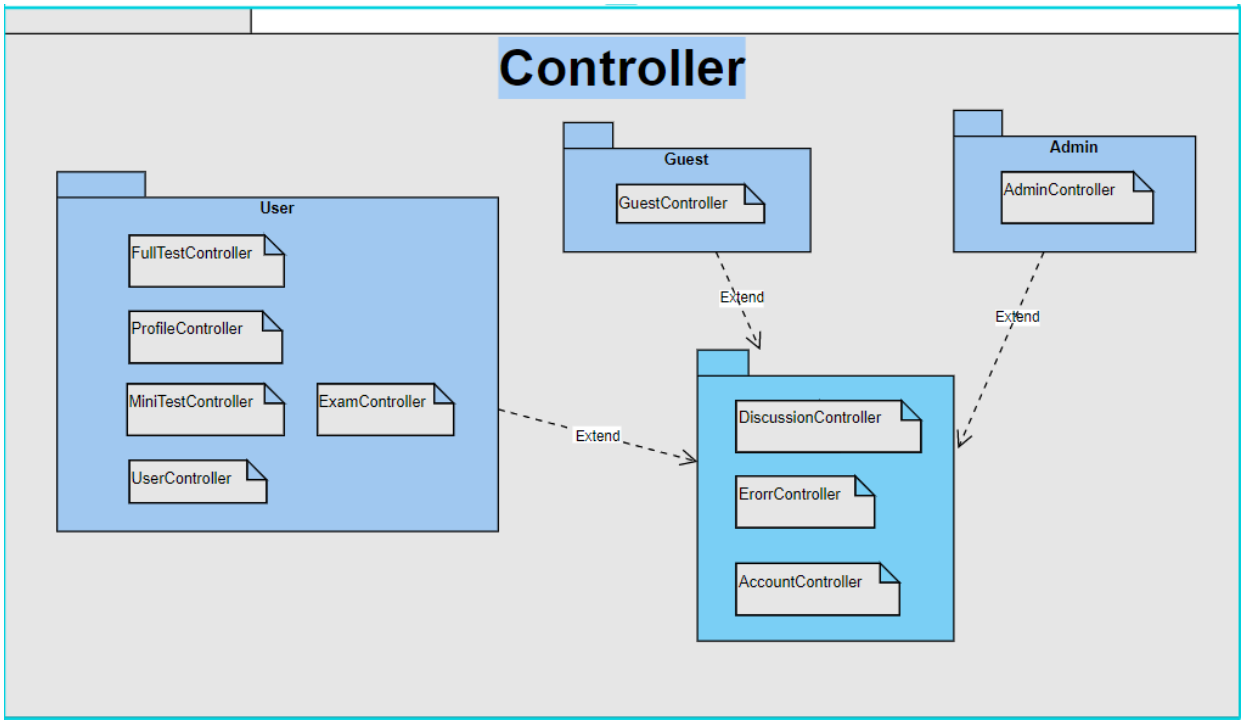
## 4. Logical View



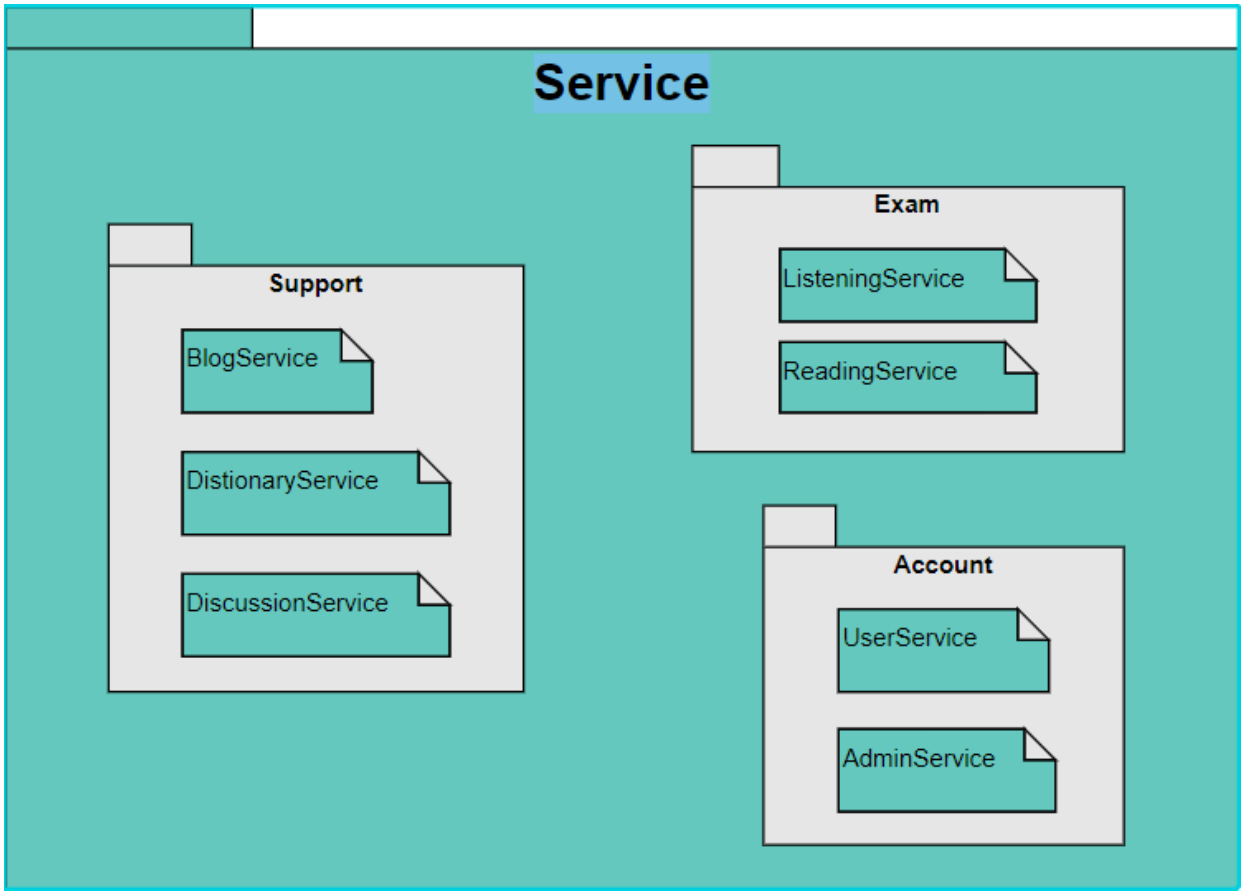

Picture 4.1:Logical View Overview of ToeicExam



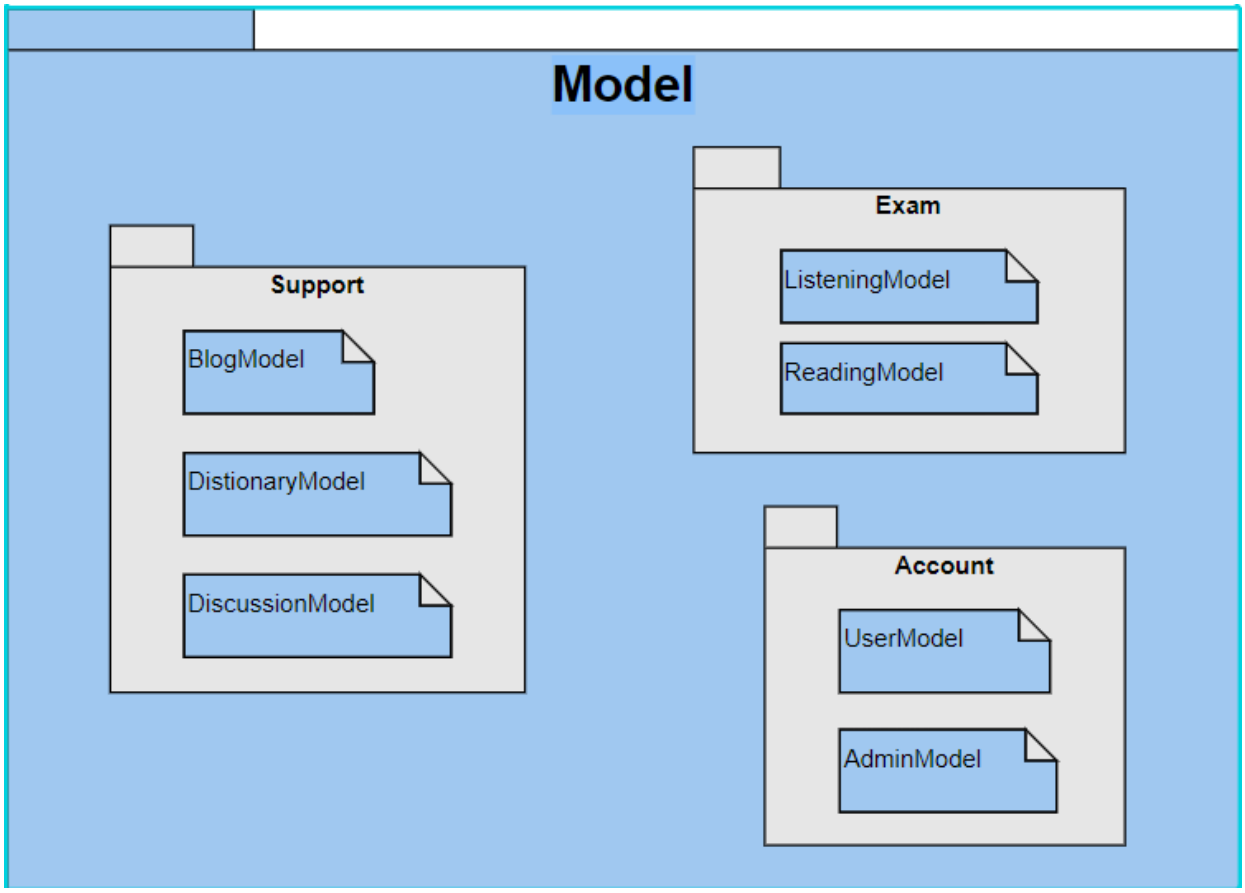

Picture 4.2 *Package Diagram of View*




4.3: *Package Diagram of Controller*

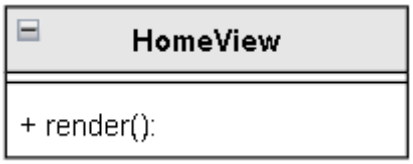



4.4 *Package Diagram of Service*




4.5 *Package Diagram of Model*

### 4.1 Component: HomeView

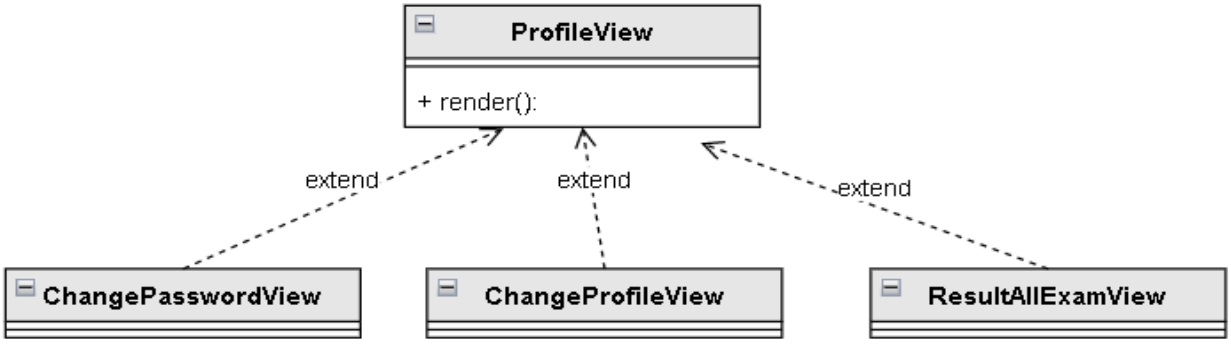



*** HomeView**

Description: HomeView is the interface when users visit the homepage of the web.

- render(): the function returns the HTML string to the Client to display the interface.

### 4.2 Component: ProfileView

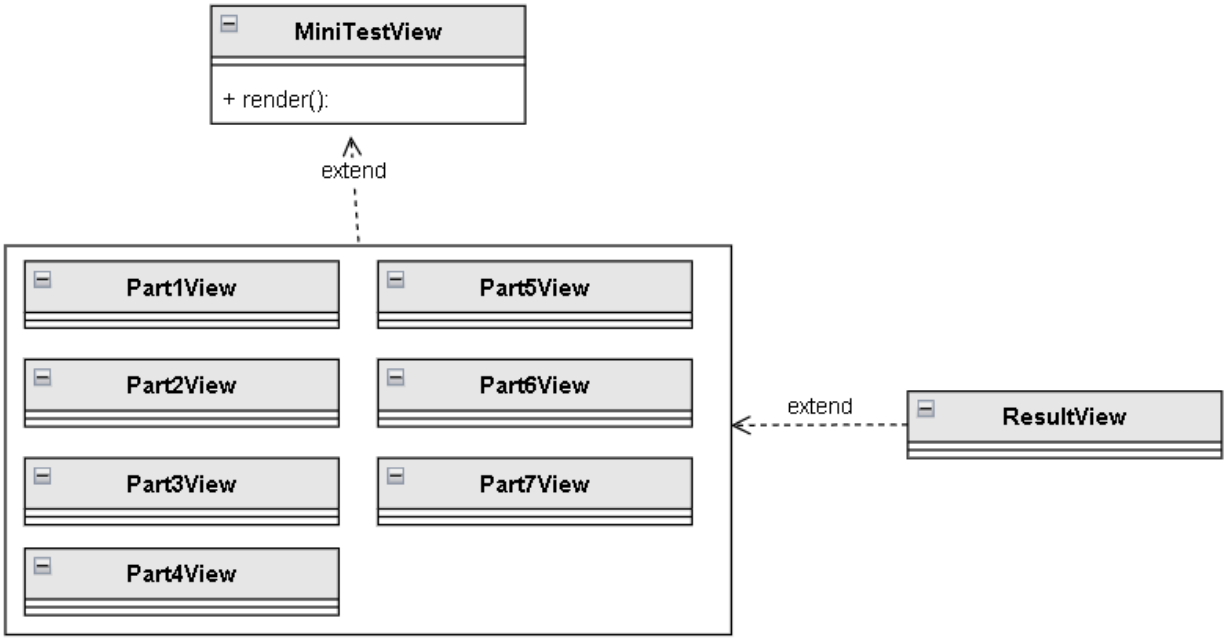


*** ProfileView**

Description: ProfileView is an interface that displays user personal information (account name, password, ...) and can change personal information, password and view test results of that user. ProfileView has 4 interfaces: ProfileView, ChangePasswordView, ChangeProfileView, ResultAllExamView

- render(): the function returns the HTML string to the Client to display the interface.

### 4.3 Component: MiniTestView

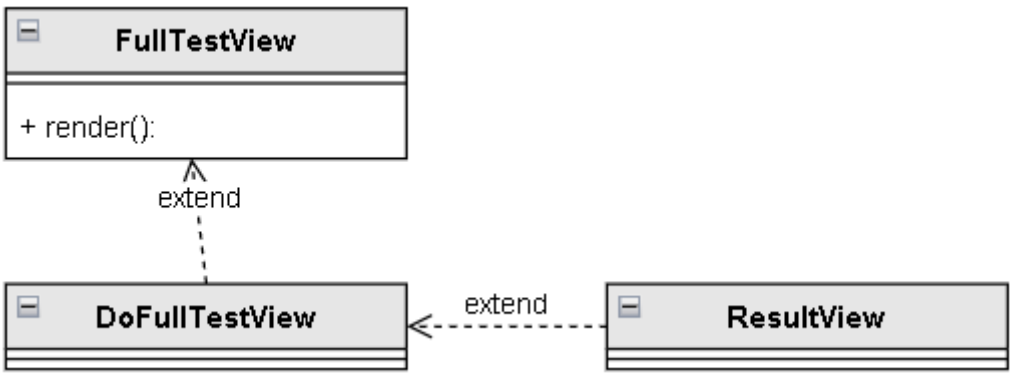



*** MiniTestView**

Description: MiniTestView is the interface that displays the exam questions and parts to take the test, the user selects the test part and the exam question to enter the Part?view interface to perform the test and after completing the test, displays the ResultView interface to view the results. test. MiniTestView has 9 interfaces: MiniTestView,, Part1View -> Part7View, ResultView

- render(): the function returns the HTML string to the Client to display the interface.

### 4.4 Component: FullTestView

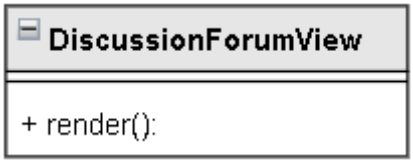


*** FullTestView**

Description: FullTestView is the interface that displays the exam questions, the user selects the test to enter the DoFullTestView interface to perform the test and after completing the test, the ResultView interface is displayed to view the test results.

- render(): the function returns the HTML string to the Client to display the interface.

### 4.5 Component: DiscussionForumView




*** DiscussionForumView**

Description: DiscussionForumView is a forum display interface for users to discuss, ask questions they have or answer questions with each other.

- render(): the function returns the HTML string to the Client to display the interface.

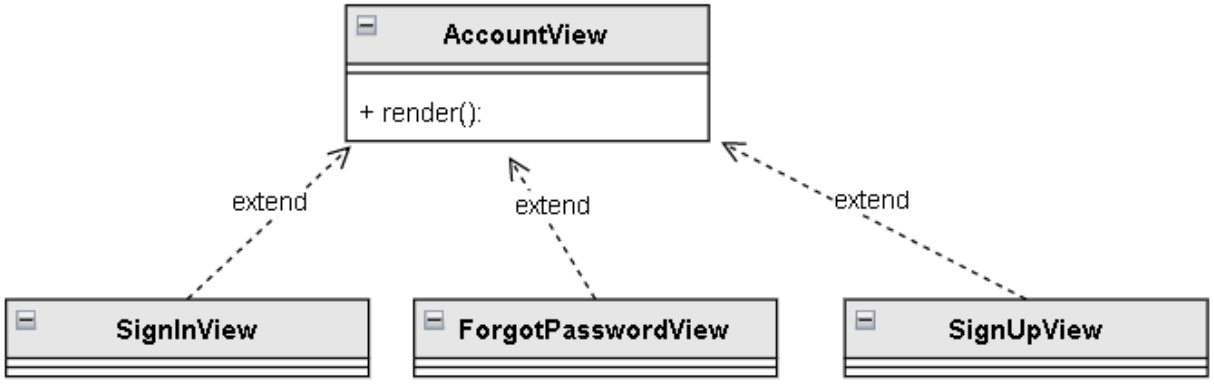
### 4.6 Component: BlogView

*** BlogView**

Description: BlogView is an interface that displays articles so that users can learn more knowledge through that article.

- render(): the function returns the HTML string to the Client to display the interface .

### 4.7 Component: AccountView



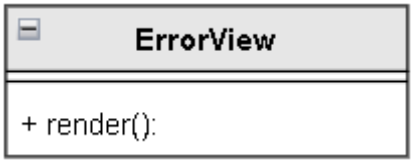

*** AccountView**

Description: AccountView is the interface for users to log in, register or retrieve forgotten passwords (Only Guest).

AccountView has 3 interfaces SignInView, SignUpView, ForgotPassWordView.

- render(): the function returns the HTML string to the Client to display the interface.

### 4.8 Component: ErrorView




*** ErrorView**

Description: ErrorView is an interface that displays an error when the page cannot be accessed.

- render(): the function returns the HTML string to the Client to display the interface.

### 4.9 Component: ManageAccountView

*** ManageAccountView**

Description: ManageAccountView is a web interface to manage User Accounts.

- render(): the function returns the HTML string to the Client to display the interface.

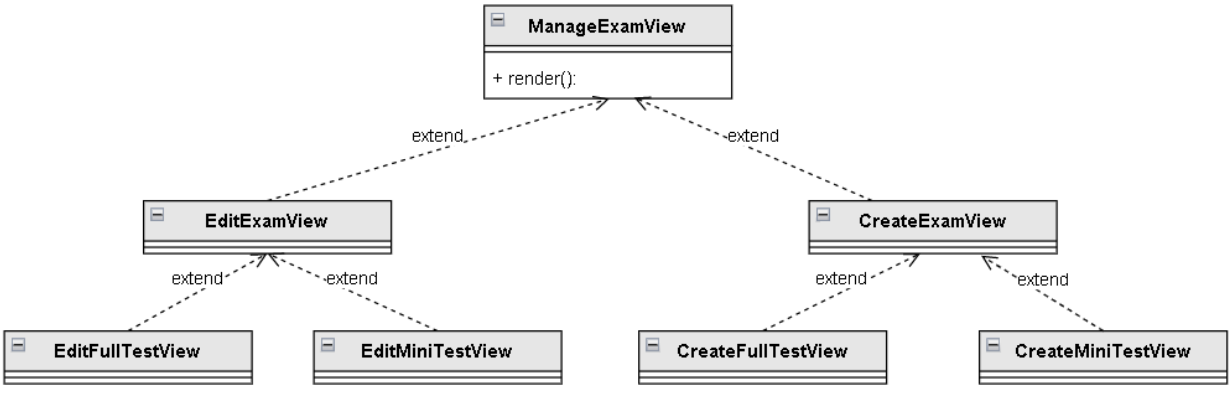
### 4.10 Component: ManageBlogView

*** ManageBlogView**

Description: ManageBlogView is a web interface for managing blogs.

- render(): The function returns an HTML string to the Client to display the interface.

### 4.11 Component: ManageExamView

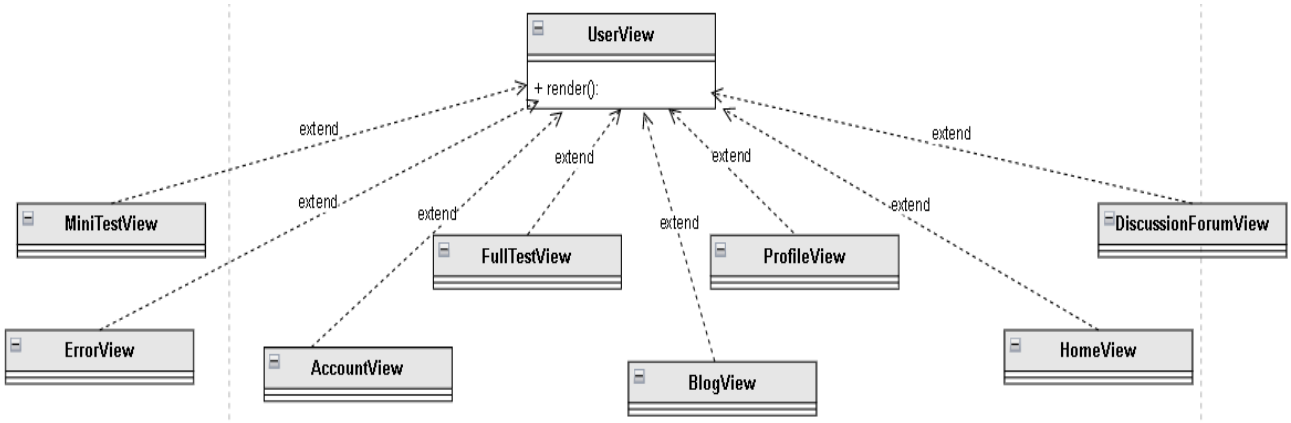



*** ManageExamView**

Description: ManageExamView is the interface to manage (create or edit) the tests created by the admin.

- render(): The function returns an HTML string to the Client to display the interface.

### 4.12 Component: UserView

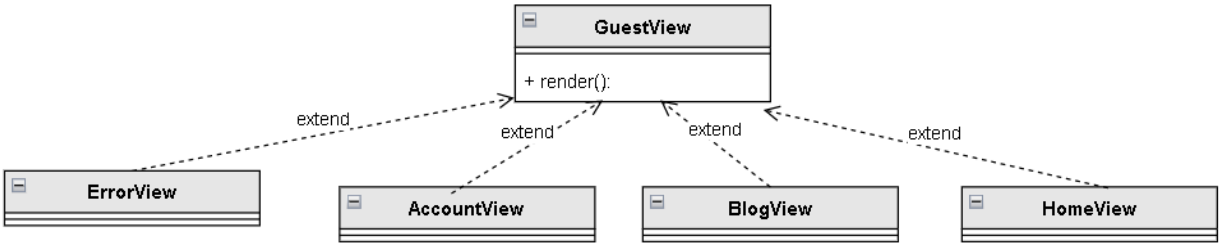



*** ManageExamView**

Description: UserView is a display interface so that users can choose to do parts such as Mini test, Full test, edit user information or discuss (including 5 small parts: MiniTestView, FullTestView, DiscussionForumView, ProfileView).

- render(): The function returns an HTML string to the Client to display the interface.

### 4.13 Component: GuestView

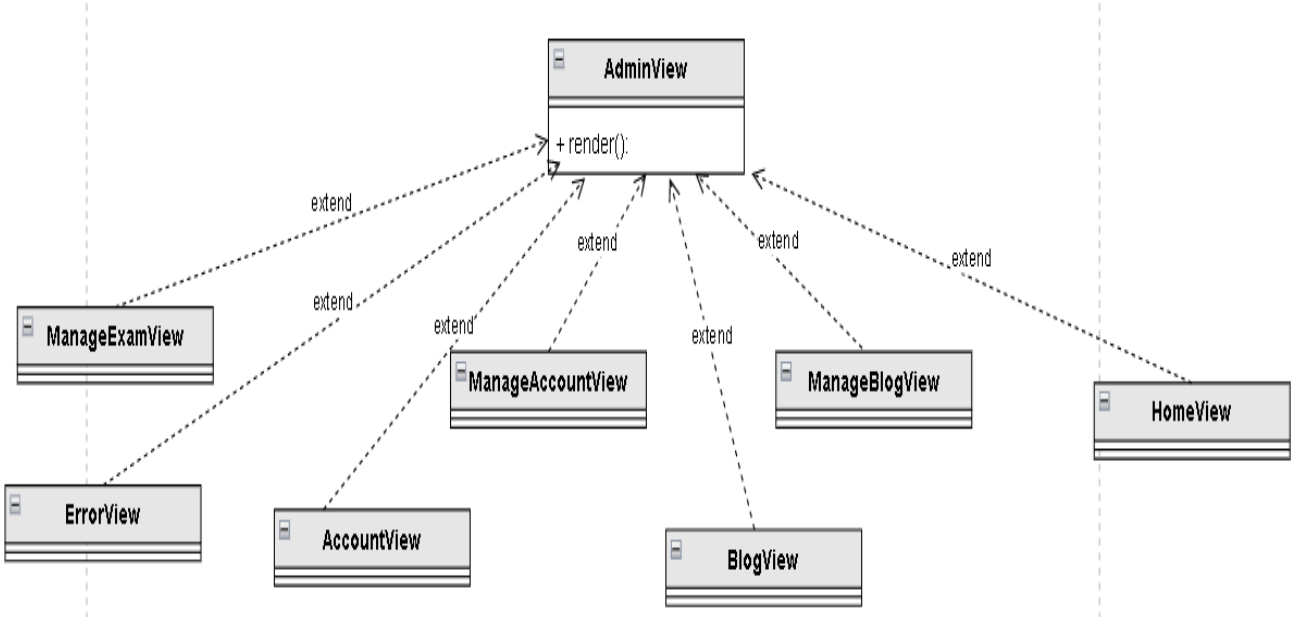



*** GuestView**

Description: GuestView is the display interface for users to read articles in BlogView.

- render(): The function returns an HTML string to the Client to display the interface.

### 4.14 Component: AdminView

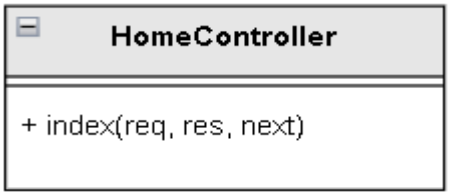



*** AdminView**

Description: AdminView is a display interface for administrators to manage users and main website functions (including: ManageExamView, ManageBlogView, ManageAccountView)

- render(): The function returns an HTML string to the Client to display the interface.

### 4.15 Component: HomeController

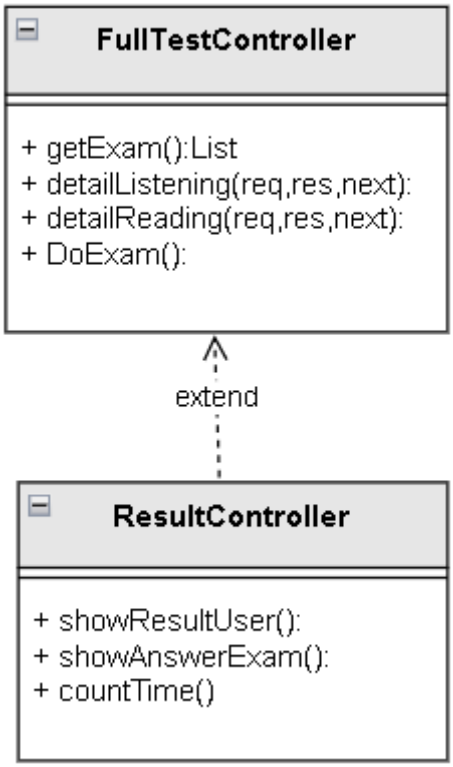



*** HomeController**

Description: Handle navigation to the site's home page

- index(): Perform navigation tasks.

### 4.16 Component: FullTestController




*** FullTestController**

Description: Process requests to view the list of exam questions and take the test

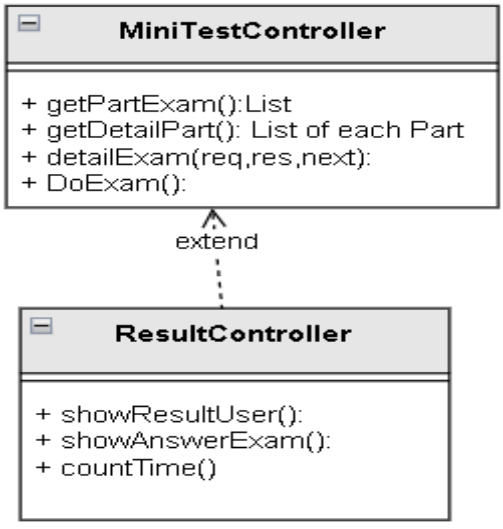
- getExam(): Returns the list of questions in the full test
- detailListening(): Show details of the questions of the listening test
- detailReading(): Show details of the questions of the reading test

*** ResultController**

Description: Process and display results and answers on the screen

- showResultUser(): Returns the display point to the user
- showAnswerExam(): Displays the list of answers for the test
- countTime(): Count the time

### 4.16 Component: MiniTestController



*** MiniTestController**

Description: Processing requests to see the list of sections and detailed exam questions in parts

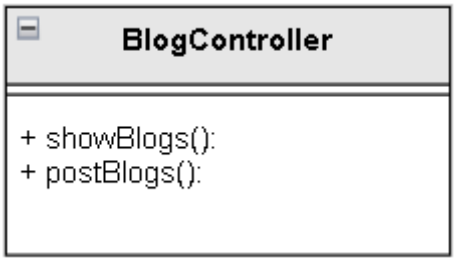
- getPartExam(): Returns a list of part in the MiniTest
- getDetaiPart() Returns a list of exam questions for each part
- detailExam(): Show details of the questions in each exam question of each respective part
- DoExam(): Function to take the test

*** ResultController**

Description: Process and display results and answers on the screen

- showResultUser(): Returns the display point to the user
- showAnswerExam(): Displays the list of answers for the test
- countTime(): Count the time

### 4.17 Component: BlogController

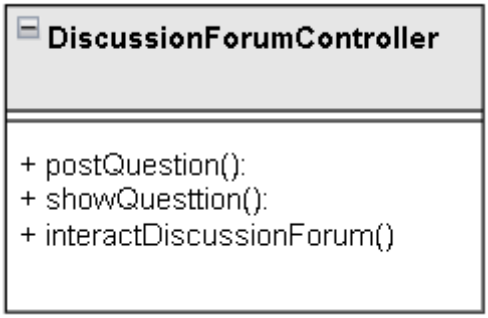



*** BlogController**

Description: Perform the task of viewing the Blog list and posting Blog

- showBlog(): Show details of blog posts
- postBlogs(): Implement blog posts

### 4.18 Component: DiscussionController

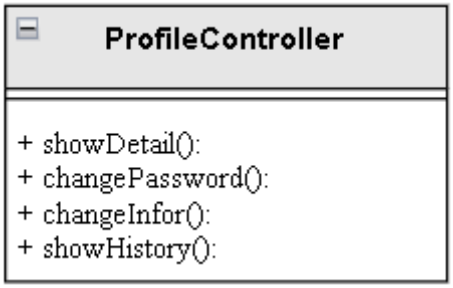



*** DiscussionForumController**

Description: Handle posting and display of questions and interactions in the discussion section

- postQuestion(): Perform posting of questions
- showQuestion(): Show the questions
- interactDiscussionForum(): handle interactive discussions in discussionforum

### 4.19 Component: ProfileController

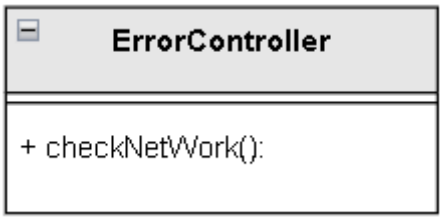



*** ProfileController**

Description: Handling changes to account details

- showDetail(): Display detailed information of the account
- changePassword(): used to handle password changes
- changeInfor(): Function to change information eg: Name, date of birth, gender,...
- showHistory(): Displays the history of the person who took the exam

### 4.20 Component: ErrorController

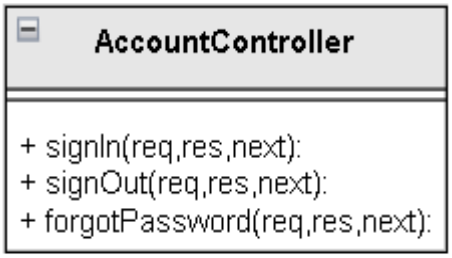



*** ErrorController**

Description : Check the website for errors

- checkNetWork(): Checks whether the network is stable or not and displays connection details

### 4.21 Component: AccountController

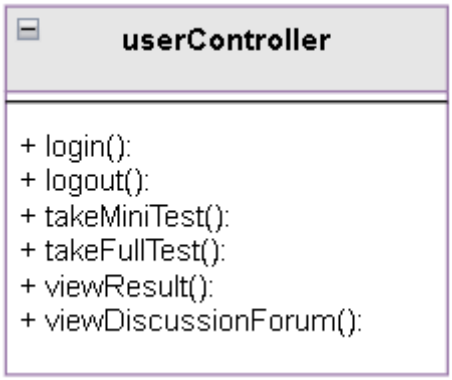



*** AccountController**

Description: handle login, registration, forgot password

- signIn(): Make login request
- signOut(): Make a registration request
- forgotPassword(): perform forgot password request

### 4.22 Component: UserController

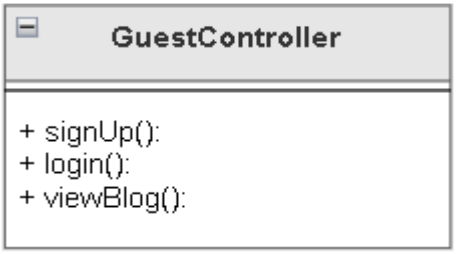



*** UserController**

Description: Handle user requests such as: login, do minitest, view results,...

- login(): Handle logged in users
- logout(): Handle user logout
- takeMiniTest(): used to make miniTest for users
- takeMiniTest(): used to do fulltest for users
- viewResult(): Handles the user viewing his results
- viewDiscussionForum(): Process view of my discussion post

### 4.23 Component: GuestController

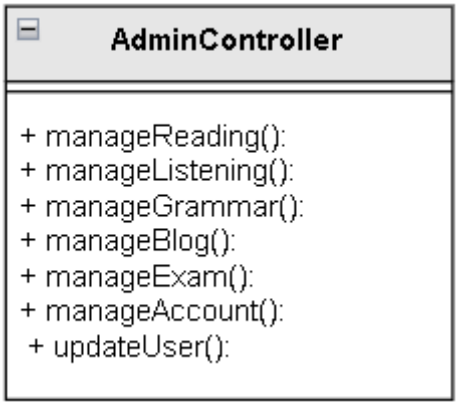



***GuestController**

Description: Handle guest requests such as registration, login, blog viewing

- signUp(): Handling customers who register for an account
- login(): Handle guest login
- viewBlog(): Handle visitors who can see useful blog posts

### 4.24 Component: AdminController

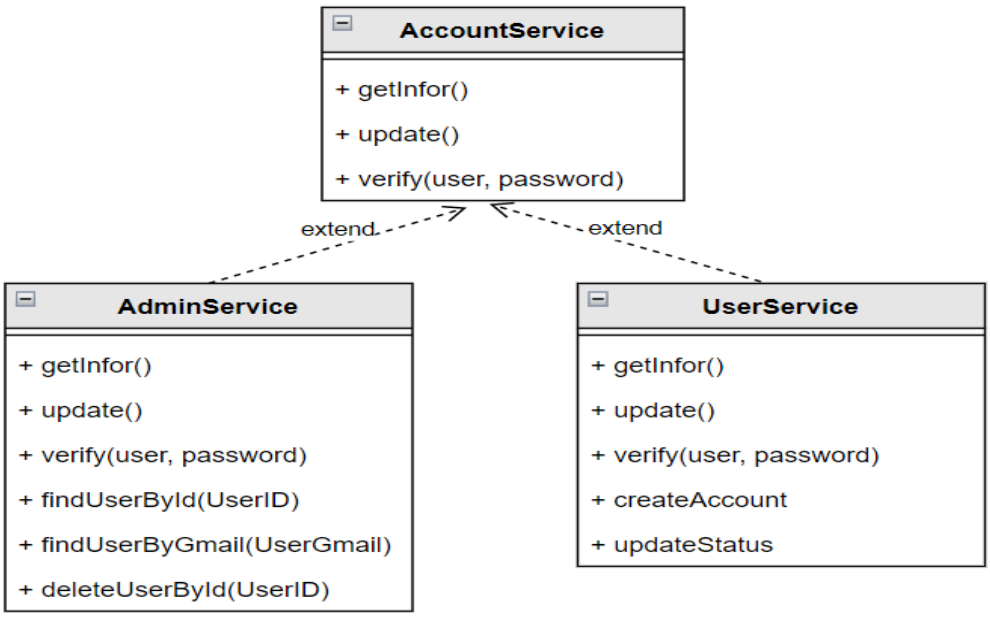


***AdminController**

Description: Handle the requests of the admin on the admin page eg: blog management, listening management, account management,...

- manageReading(): Displays a page to handle redesign of readings such as adding, deleting, editing
- manageListening(): Display the page to handle the redesign of the listening lesson
- manageGrammar() Displays the admin's grammar correction handling page
- manageBlog(): Displays the page for editing Blog posts
- manageExam() Displays the page for editing sentences in the test
- manageAccount(): Show user accounts to handle violations or not
- updateUser(): Remove user add admin's user

## 4.25 Component: AccountService




Description:

- Perform the functions of retrieving, editing, and verifying information about admin and user accounts. -AdminService and UserService are inherited classes, so they will inherit methods from the parent class and define more methods themselves.

### 4.26 Component: ExamService

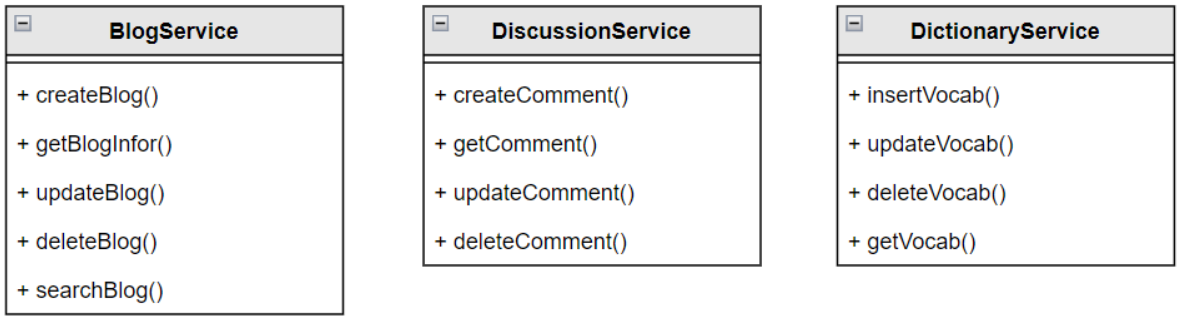
***ReadingService:**

Description: perform the processing functions related to the reading test: find the test by part, save the test, delete the exam.

***ListeningService:**

Description: perform the processing functions related to the listening test: find the exam by part, save the exam question, delete the exam question.

### 4.27 Component: SupportService




*** BlogService:**
Description: perform the functions of handling blog-related articles such as creating blogs, editing blogs, deleting blogs, getting blog information...

- createBlog(): send new blog data to model.
- getBlogInfor(): get blog information from model.
- updateBlog(): edit, update blog content.
- deleteBlog(): delete blog
- searchBlog(): search blog

***DiscussionService:**

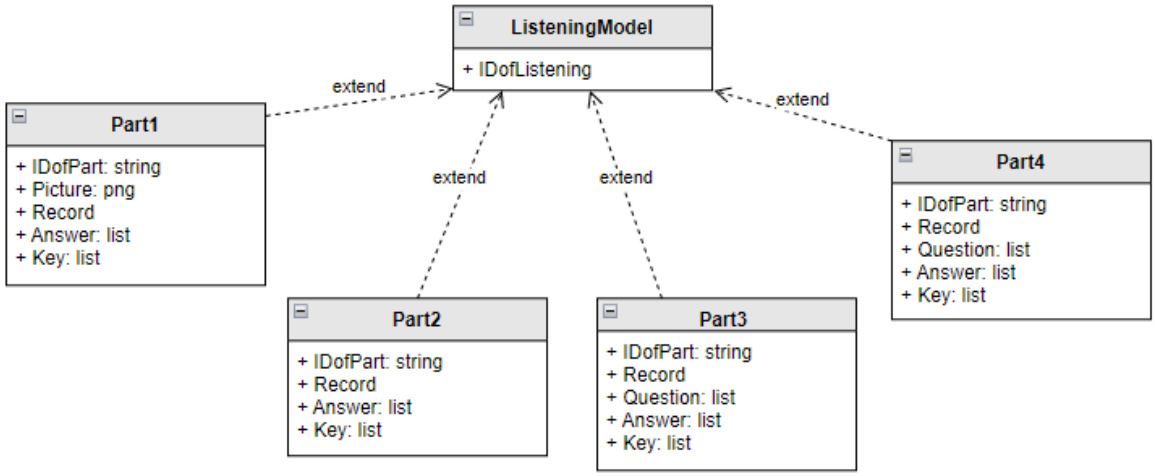
Description: perform discussion-related processing functions such as creating comments, retrieving comment content.

***DictionaryService:**

Description: perform dictionary-related processing functions such as creating new words, editing words, deleting words, etc…

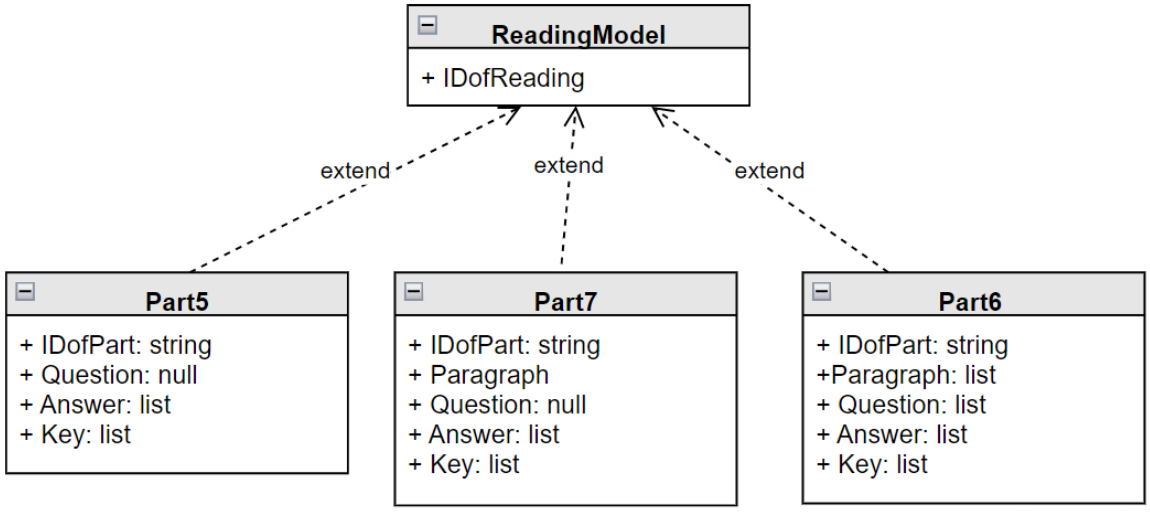
### 4.28 Component: ListeningModel

Description: the class contains properties that store the content of the Listening test, and the classes contain the details of the questions and answers as well as the content.



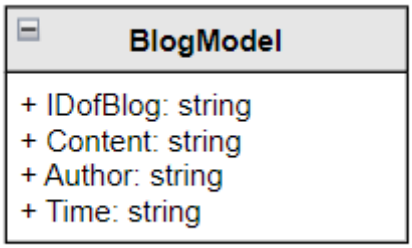

### 4.29 Component: ReadingModel

Description: the class contains properties that store the content of the Reading test, and the classes contain the details of the questions and answers as well as the content.



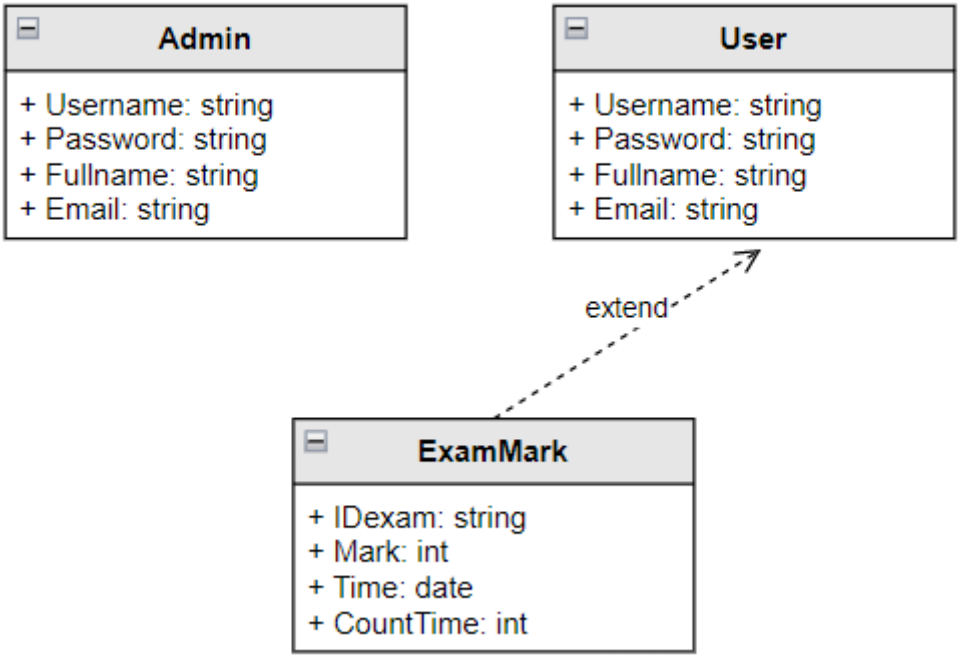

### 4.30 Component: BlogModel

Description: class contains properties that contain detailed information about the blog.



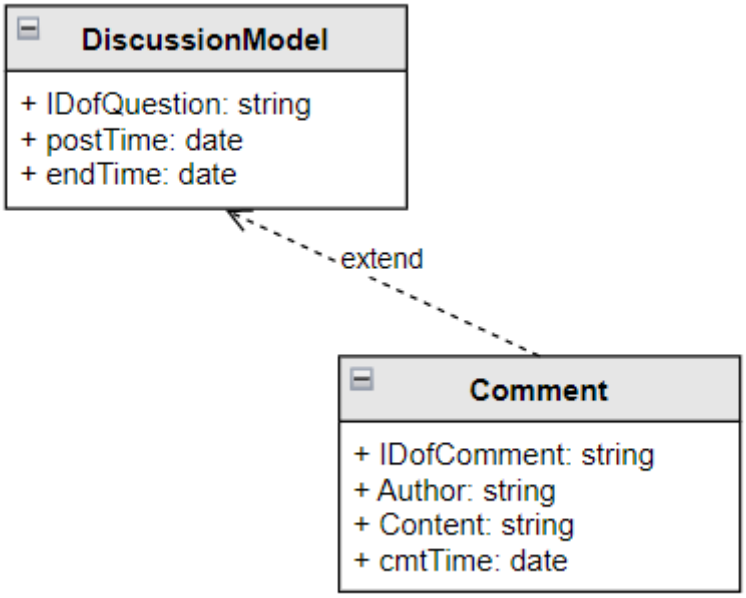

### 4.31 Component: AccountModel

Description: class contains information of admin and user (user will include additional class containing information about the exam taken).



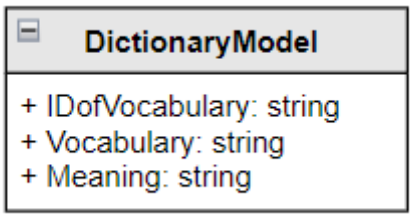

### 4.32 Component: DicussionModel

Description: the class contains the information of the discussion forum, contains a class that contains the information of comments from the users.




### 4.33 Component: DictionaryModel

Description: the class contains information about the vocabulary.




## 5. Deployment

*[Leave this section blank for PA3.]*

## 6. Implementation View

*[Leave this section blank for PA3.]*