Minor Project - 1

Submitted in partial fulfilment of the
Requirement for the award of the degree of
Masters of Computer Application

“Trivia App”

Submitted To
Dr. Indu Sahu
Associate Professor

Submitted By
Yashovardhn
00117704421

(Batch: 2021-2023)

Vivekananda Institute of Professional Studies
(Affiliated to Guru Gobind Singh Indraprastha University)

MINUTIAE
Trivia Game

Planning

Initial research

There are a variety of trivia game iOS apps in App Store. Those apps have great features and functionalities to satisfy users. However, according to my research, only a few of them have friendly user interface and human centred interactions, which means that a lot of them might be difficult to be navigated even though they provide enough functionalities. It is not convenient for new users.

Quiz Contest is an iOS application that has general questions related to current afairs and computer or according to the api set for the questions. It has multiple choice questions with time limit and it also calculate scores of each correct answer. It is good for students of every age group it helps in increasing general knowledge about world ,Sports and computer etc. Don't need register simply give any user name and password it will saved automatically and you can login again with same user name and password don't have to worry about the past score. The application helps the user to increase his/her knowledge. Since Smartphone mobiles are being widely used by general population and students, the Quiz Contest application can provide on the Student's mobile.

Development environment: Xcode

Open Weather Map API documentation:

https://developer.forecast.io/

Project Specification

- Real time weather forecasting.
- System : MacOS MONTEREY V12.3
- System Specification : 1.4 GHz Quad-Core Intel Core i5
- Graphics : Intel Iris Plus Graphics 645 1536 MB
- Memory : 8 GB 2133 MHz LPDDR3
- Platform: iOS.
- IDE: Xcode.
- Takes user's geolocation as input to provide weather forecast by using apple's CLocation Manager .
- Displays detailed weather information for the current.
- JSON data from the site.
  i) API used to fetch data fromwww.openweathermap.org

Scope of the project

The scope of this iOS app will not be too broad. On the contrary, I will narrow down and only focus on a few functionalities which have high usage frequency. They are basic and also they can totally satisfy users' needs.

The main functionalities include:

- Welcome View.
- Attempt questions.
- Locking the answers.
- Generating scores and displaying them in the leaderboard.
- Displaying if the answer is right or wrong then displaying the correct answer in real time.
- Multiple lives to answer in a set of questions.
- Rather fast interface.

Possible risks involve

- API

I got Question information from open OpenTB API, due to I failed bunch of times accessing to the OpenTB's API. I have no idea why I failed. I registered and got an API key; however, I still failed.

- Development

I have little experience with apple iOS app development, even with swift programming, therefore, making those elements work definitely will be a big challenge for me. I will do research first, or watch some tutorial videos online, practice by myself and then apply them on my app.

- Testing

Different devices have different layouts and screen sizes. The GUI design and development might not fit all kinds of devices.

Meanwhile, different users might have different opinions for this app and I need to consider which feedback I should listen and which I should not.

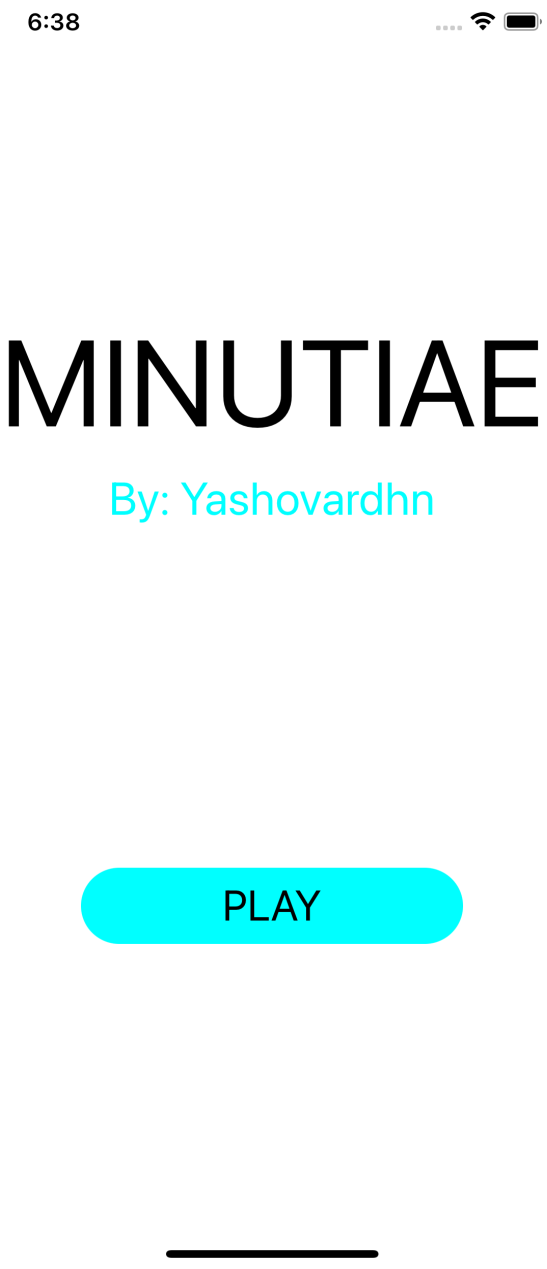
- Debug

It might take a long time to debug and the iOS app might have few unexpected bugs. It takes time to do research and fix. I am not sure how long it will take, but I will try to make them be solved in a certain time and this project is still under control.

Navigation and User Interface Design

Wireframes

I designed wireframes for this weather iOS app which basically can display the information processes and how it works.



Analysis

IOS user analysis

Basically, every iOS users could be the users of my app. However, especially for people who would like to see the weather every day and decide what to wear tomorrow will be the target users for this app.

- The main target users include:
- Business men/women
- For business men or women, they need to view the weather information before they go to work every day, so that they will know what to wear, such as long sleeve t-shirt or short sleeve t-shirt.
- Mothers
- Every mother cares about their children, so another group of target users could be mothers. They will view weather information every morning to know how to prepare clothes for their children.
- Travellers
- Travellers will view destinations' weather information before they depart. Also, before they go back to home, they need to view weather information again.

Scenario analysis

- Screen and Interaction Analysis

The users will use this iOS app on iOS devices . All the information of this iOS app will be displayed full screen.

- Usage Analysis

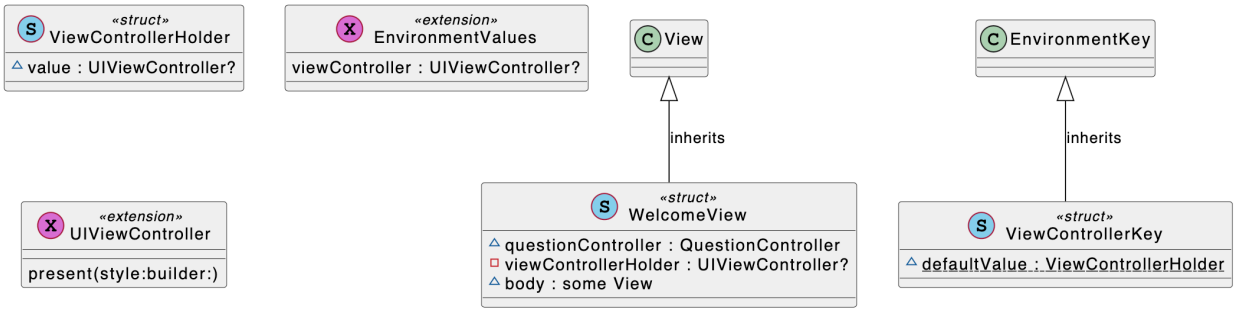
Users can use this iOS app on the morning every day at home, on their way to travel, and other situations as long as they want to know weather information.

- Environment Analysis

This iOS app only can be used on iPhones , iPads . It will access to the open weather API to get the weather information. It sends requests, and then get responses from the API through the internet.

Design

Architectural design

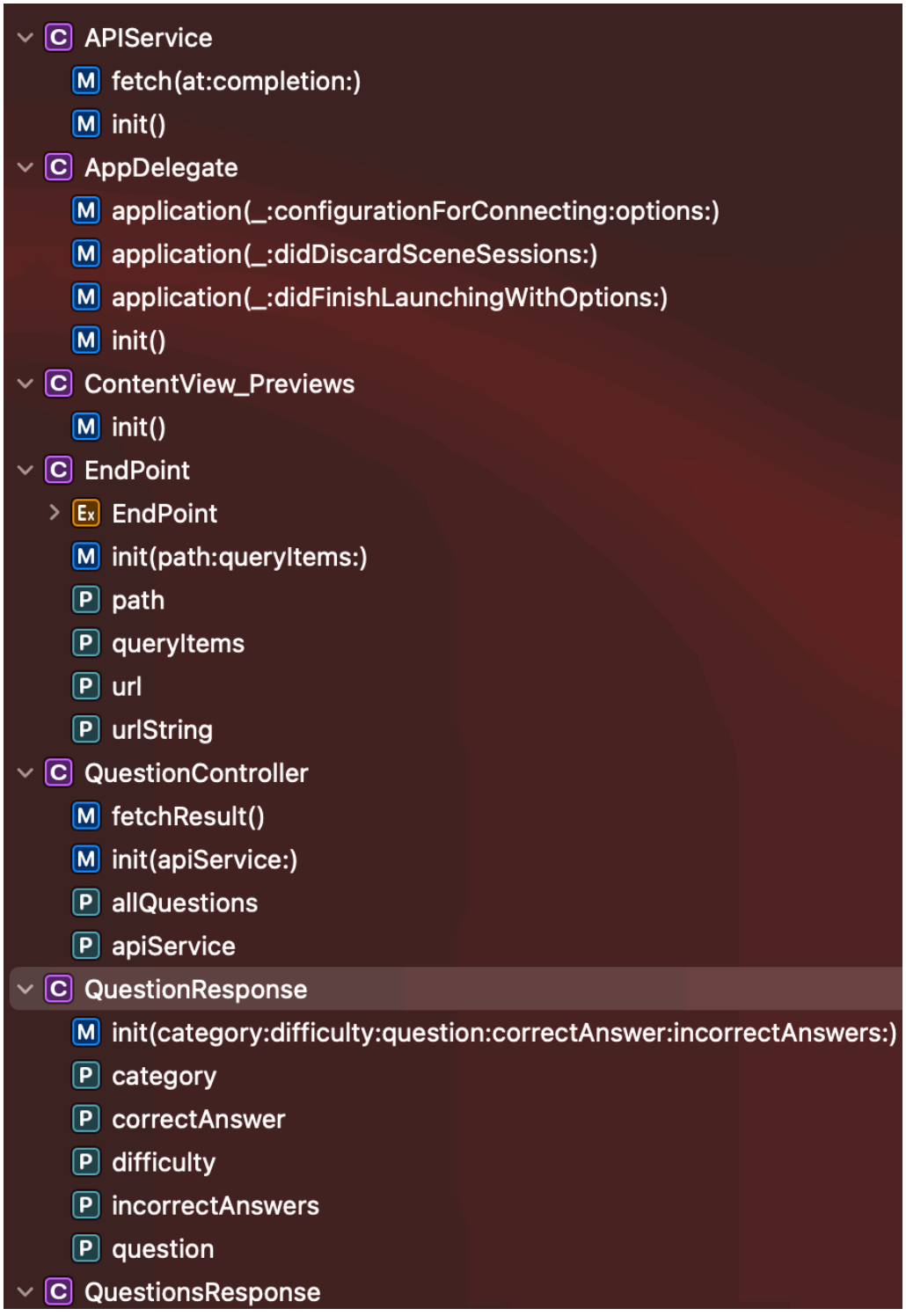


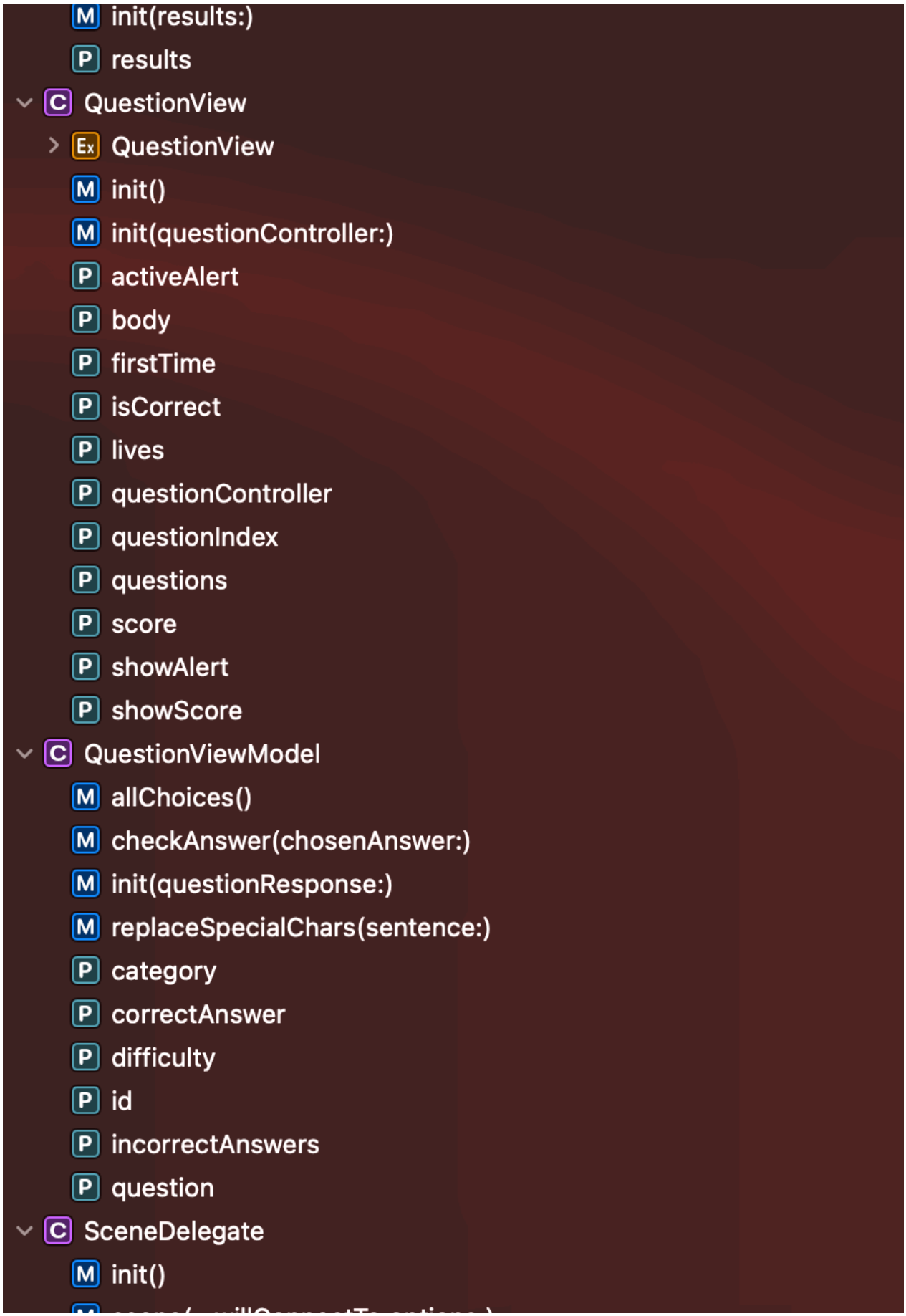
Swift files

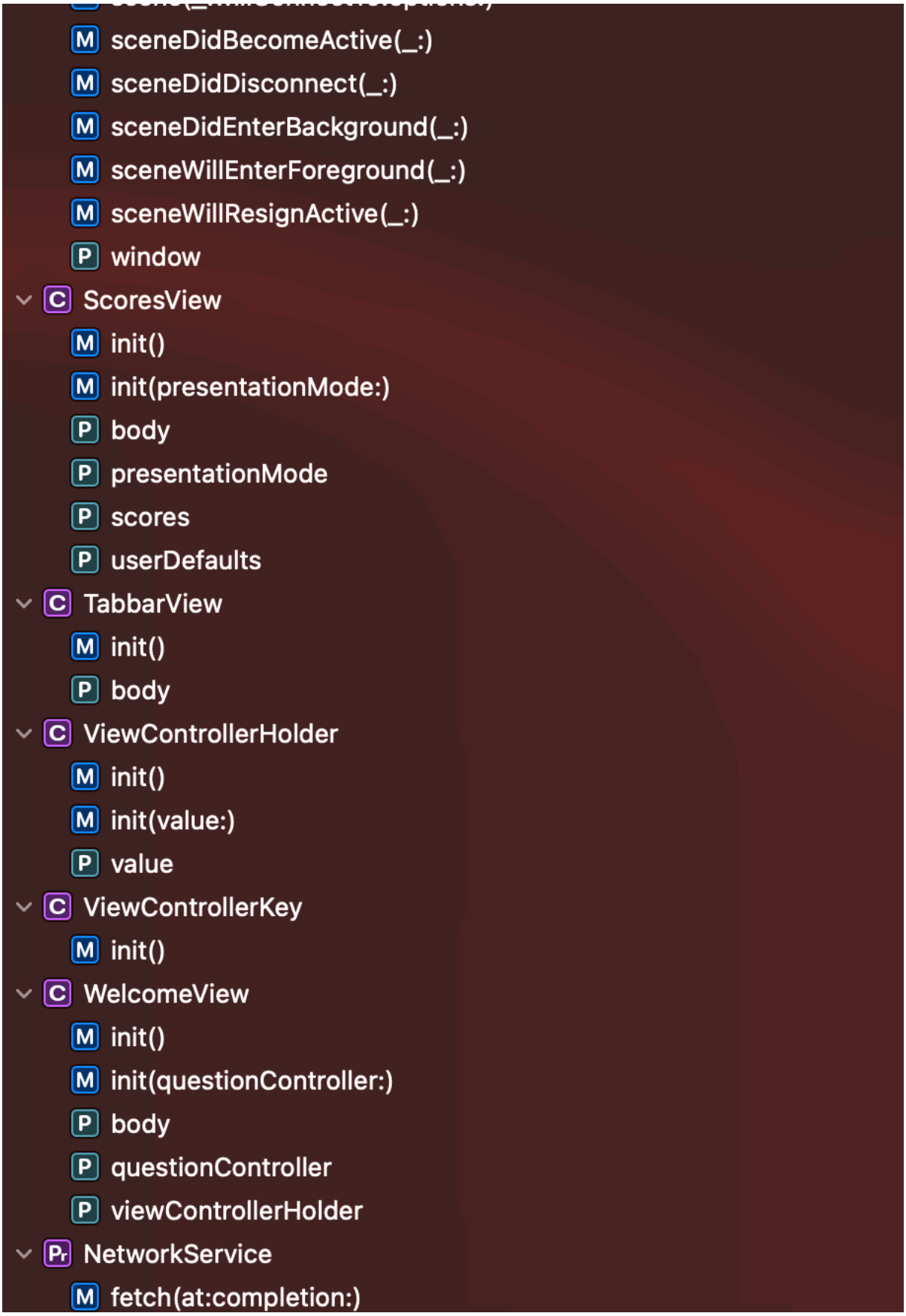
```
@startuml
' STYLE START

skinparam shadowing false
' STYLE END



class "WelcomeView" as WelcomeView << (S, SkyBlue) struct >> {
  ~questionController : QuestionController
  -viewControllerHolder : UIViewController?
  ~body : some View
}
class "ViewControllerHolder" as ViewControllerHolder << (S, SkyBlue) struct >> {
  ~value : UIViewController?
}
class "ViewControllerKey" as ViewControllerKey << (S, SkyBlue) struct >> {
  ~{static} defaultValue : ViewControllerHolder
}
class "EnvironmentValues" as EnvironmentValues << (X, Orchid) extension >> {
  viewController : UIViewController?
}
class "UIViewController" as UIViewController << (X, Orchid) extension >> {
  present(style:builder:)
}
View <|-- WelcomeView : inherits
EnvironmentKey <|-- ViewControllerKey : inherits
```

Implementation Details

Class files




init(results:)
results
QuestionView
QuestionView
init()
init(questionController:)
activeAlert
body
firstTime
isCorrect
lives
questionController
questionIndex
questions
score
showAlert
showScore
QuestionViewModel
allChoices()
checkAnswer(chosenAnswer:)
init(questionResponse:)
replaceSpecialChars(sentence:)
category
correctAnswer
difficulty
id
incorrectAnswers
question
SceneDelegate
init()


sceneDidBecomeActive(_:)
sceneDidDisconnect(_:)
sceneDidEnterBackground(_:)
sceneWillEnterForeground(_:)
sceneWillResignActive(_:)
window
ScoresView
init()
init(presentationMode:)
body
presentationMode
scores
userDefaults
TabbarView
init()
body
ViewControllerHolder
init()
init(value:)
value
ViewControllerKey
init()
WelcomeView
init()
init(questionController:)
body
questionController
viewControllerHolder
NetworkService
fetch(at:completion:)

APIs, code examples or libraries to be used API

String forecastUrl = "https://opentdb.com/api.php?amount=10"

Code examples

Xcode files

```
import Foundation

struct EndPoint {
    var path: String
    var queryItems = [URLQueryItem]()

    var urlString: String {
        url?.absoluteString ?? "errorURL"
        //    return  "https://" + Keys.baseURLPath + path
    }

    var url: URL? {
        var components = URLComponents()
        components.scheme = "https"
        components.host = Keys.baseURLPath
        components.path = path
        components.queryItems = queryItems
        return components.url
    }

}


extension EndPoint {

    static func questions(queryItems : [URLQueryItem]) -> Self {
        EndPoint(path: Keys.questionPath, queryItems: queryItems)
    }
}
```

Testing plan/Deployment

Planned test

I conducted a testing plan to see if the iOS app can connect to the server and get JSON data in while I am programming. Also, I will fix bugs during the development progress as many as possible. Besides, there is another test plans that let me test the entire iOS app.

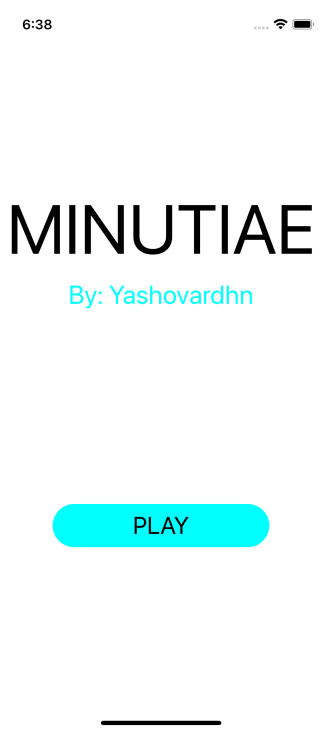
Here is the basic schedule: White Box Testing

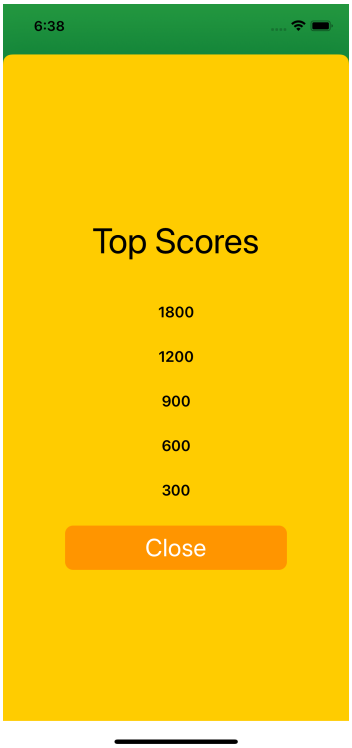
Testing Case	Testing Result
Launching the App	Working
Splash/Launch Screen	Working
JSON data for questions and answer	Working
Exit Button that Resets the game	Working
High Score Stacks	Working
On simulator of different sizes	Some working
Displaying correct answers if users gets it wrong	Working
Lives system	Working
Loading screen	Working

Functional testing

As I said above, few functionalities work well.

- The iOS app can connect to the internet/server
- The iOS app can send request to the API server
- The iOS app can get JSON data from the API server
- The iOS app can parse JSON data and display all the data on the screen
- Users can select the desired answers
- Users can view recent scores
- Users can view their lives remaining in a set of question
- The data is displayed accurately without any difficulties or hiccups
- Working and integrating splash screen using storyboard





User 1	Result	Feedback
Option registration	Working	Not very fluid
Exit function	Working	No content
Score updating	Working	This is good

User 2	Result	Feedback
Option registration	Working	Yes
Exit function	No……	
Score updating	Working	The colors are too light

User 3	Result	Feedback
Option registration	Not Working	Ipad doesn't support this app as of now
Exit function	No……	
Score updating	Not Working	

Adjustment/corrections to be conducted

- UI design:

The UI design does not look good due to the size of the icon. I will modify those icon or reset the size of them to make it look appropriately according to the size of the screen.
Besides, the xml file is using relative layout, so some elements will look differently after modifications. I will do more tests to ensure this iOS app has friendly user interfaces.

Other testing activities to be conducted

- User experience

The user experience might be different when I test the app on a real device, due the resolution and the screen size. I will definitely do adjustments based on the data that I get as stated above with the iPad and maybe a phone not of the aspect ratio of 18:9.

- Functionalities

Few functionalities might be changed due to technical reasons. For an instance, I planned to display the basic weather information in a small widget in the home screen of the device. If this functionality cannot be implemented eventually, I probably will cancel it.

Future Perspectives

Few functionalities can be improved or accomplished

- Make the multi languages functionality works
- Users are able to select type of questions
- Users are able to add names for the scores
- Users will be able to see history of questions and answers with their own answers selected.

Things I Learned

- Xcode UI and its working
- Using API on a project or app
- Swift and Swiftui
- Storyboard and its working
- Apple API
- SF Symbolsk

Limitations

- Unpaid APIs provide incomplete services.Many details cannot be fetched
- Often tuples of upcoming days remain empty once again due to free APIs
- Language diversity could have been implemented.Multilingual apps make it easy for users worldwide

Summary

After taking this Project, I got much more confidence in iOS development. I am so proud that I have learned a lot from this project.

I used to self-learning iOS development and did not make any apps during that time. However, in this project, I developed a functional iOS app, even though it is not perfect.

Working on assignments, I learned more about a variety of elements of iOS and how to make them work. I learned Xcode. This project was a good opportunity to practice this skill and now I also know how to continue self-learning after this course.

Working on the project, I practiced the rapid development process, doing testing and enhancement. Meanwhile, I developed the second version, which has many improvements. The value of designing and developing this iOS app are to clarify the design process, created those files by myself, do research to fix bugs and test it with target users.