|  |  |  |  |
| --- | --- | --- | --- |
| **CS102** | **Spring 2022/23** |  | S2G7 |
|  | Project Group |
| Instructor: | **Aynur Dayanık** |
| Assistant: | **Vahid Namakshenas** |  |

|  |  |  |
| --- | --- | --- |
| **Criteria** | **TA/Grader** | **Instructor** |
| Presentation |  |  |
| Overall |  |  |

~InsuChef ~

Meal Magicians

## Emine Fidan

**Umay Dündar**

## Edip Emre Dönger

## Elif Ercan

## Yiğit Kemal Can

**Detailed Design Report**

**20 May 2023**

**1. Introduction**

InsuChef is a mobile application designed to enhance and simplify diabetes management. It aims to help by assisting individuals in tracking their daily carbohydrate consumption and accurately calculating the appropriate insulin dosage for each meal. With its simple and user- friendly interface, InsuChef offers an easier experience in managing diabetes.

**2. Details**

# System Information

The InsuChef application will be accessible on the Android Operating System. That is why we use Android Studio to implement our project. The User Interface designs are made on this program. Additionally, since our application needs certain data, we retrieve this data from the USDA food database and write it into a file in JSON format.

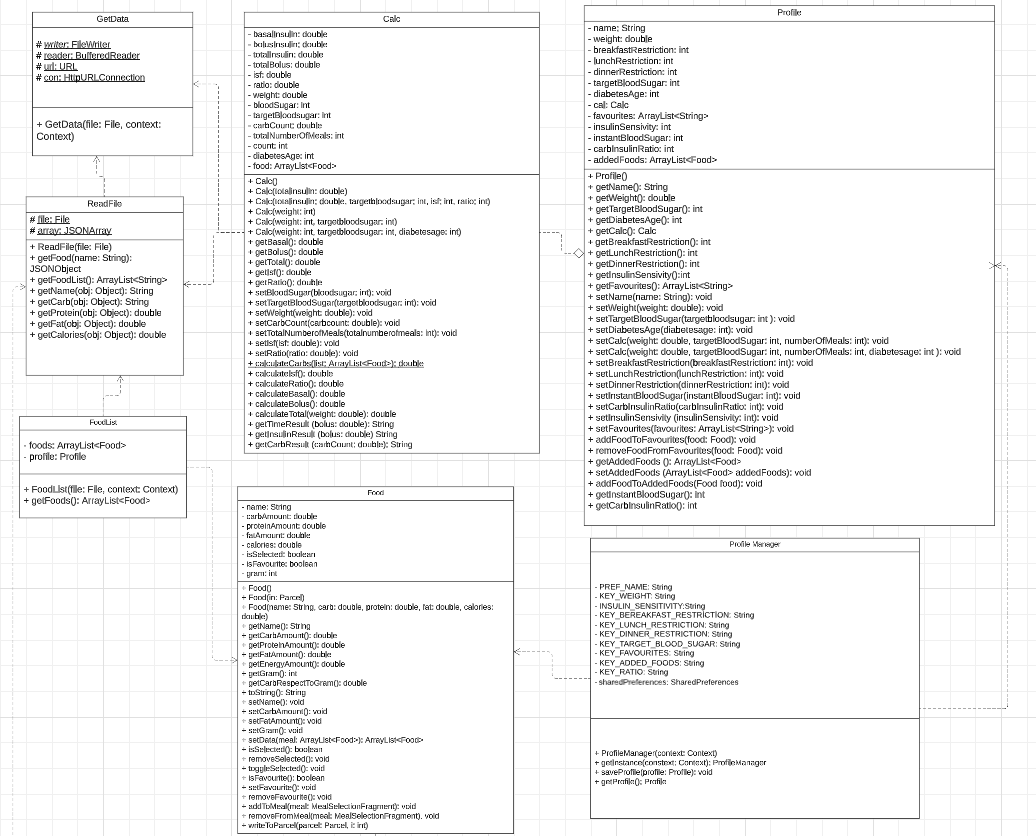
metin, ekran görüntüsü, yazı tipi, logo içeren bir resim

Açıklama otomatik olarak oluşturuldu

“Fig 1 Model-View-Controller Architecture of the InsuChef”

In this project, we use Model-View-Controller architecture. “The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application” (“MVC Framework - Introduction”). In this architectural pattern, users are able to interact with the application via buttons or text boxes. This interaction between the user and the application, which is made by controllers, manipulates the model, and the model updates the view that the user sees.

# UML Diagram and Classes



“Fig 2 UML Diagram of the InsuChef”

**1. Get Data Class**

GetData class contains a constructor that gets the data from the USDA food database and writes it into a file in JSON format. When the application is first installed, we will create a GetData object to get the data from USDA food data central and write it into a file for further usage. We will also create GetData objects once in six months to update our data.

**2. Read File Class**

ReadFile class contains a constructor which reads the JSON file which is previously created and written and stores the info into a JSONArray. It also contains many getters such as getFood, getFoodList, getProtein, getCarb, getFat and getCalories which we will use to get the information about foods for creating food objects in the Food class.

**3. Food Class**

Food class contains information about foods that we use in our application, such as food names, carbohydrate, fat, and protein amounts of foods, and energy of foods. This class also contains information on whether the food is a favourite one or not. In this way, we can store the info that we get from ReadFile class in food objects, and we can use these food objects' properties in our calculations.

**4. Calc Class**

Calc class is where all the calculations are made. This class contains many instances which are set according to the profile info, such as blood sugar, target blood sugar, and weight. It also contains many instances which are used in many calculations, such as isf (insulin sensitization factor) or insulin carbohydrate ratio. The Calc class is an instance in the Profile class, in this way, we can set values to most of the variables easily. Some of the variables, like total carb count, will be set during the meal choice process and at the end. After all the necessary values are set, we will use the Calc class methods to calculate the bolus insulin of the patient, and we will show the results on the results page.

**5. Profile Class**

The Profile class contains information about the patient, such as weight, number of meals, and target blood sugar. It also contains a Calc object which we will use for bolus insulin calculations. Profile class is also very easy to modify. When some user profile info is updated by the user Profile class can set all the variables with setters, and it can create new Calc objects according to the new user info.

**6. Profile Manager Class**

Profile Manager class is used to manage the profile info and send this info between different fragments. It can save and store the profile info and share this info in different fragments of the application such as instant info fragment or distribution fragment. In this way, we can use the profile info which previously set to do our calculations without requesting the user to input the same info again.

**7. Food List Class**

The Food List class creates a food list with food objects that are constructed before in the food class. In this way, we can use this list to show the user a list of foods to choose during the meal selection process.

# 2.3 Achievements

Our Android app project has made significant progress in various areas. Firstly, we successfully implemented the functionality to pull data from the USDA database, allowing us to access essential nutritional information such as carbohydrates, calories, proteins, and fats for different foods. This data retrieval capability provides us with the necessary information for calculations. Furthermore, we have completed the profile features, enabling users to create and manage their personal profiles within the app. This includes essential information like age, weight, and dietary preferences, which can be used to tailor the app's recommendations and suggestions to each user's specific needs. We have also made notable advancements in user interface design. Using fragments instead of separate activities in Android Studio, we have achieved a simple yet cohesive interface that enhances the overall user experience. While actively working on the entire food listing functionality, our current implementation displays the meal's selected foods in a custom list view. Each element in the list consists of an editText, a button, and a textView. This layout allows users to input quantities and view relevant information at a glance. The distribution algorithm is still a work in progress. We have created a calculation class that handles various calculations throughout the app. This class can be easily accessed and utilized for different calculations, such as insulin and carbohydrate calculations. While there is still work to be done, we are confident in delivering a feature-rich and user-friendly app to our audience.

# 2.4 Tasks

1. Emine Fidan: She works on the UI implementation of the application. She is responsible for the UI fragments, such as the distribution fragment and the related adapter and distribution classes to these fragments.
2. Umay Dündar: She works on the Database part of the application. She is responsible for retrieving data from the USDA food database, writing this data into a file in JSON format, and reading this file. She also worked on the classes that were shown in the UML diagram, such as the profile and calculation classes.
3. Edip Emre Dönger: He works on the UI implementation of the application and the classes that were shown in the UML diagram, such as the food class. He is responsible for the UI fragments, such as the meal selection fragment, and the related fragments, which are favorites and show favorites fragments.
4. Elif Ercan: She works on the Database part of the application. She is responsible for the classes that were shown in the UML diagram, such as the GetData and ReadData classes that are used to retrieve data from the USDA database.
5. Yiğit Kemal Can: He works on the UI implementation of the application. He is responsible for the main activity fragment that links all fragments in the application. He also worked on the UI fragments, such as the main, profile, and calculation fragments.

# 2.5 Screenshots

metin, ekran görüntüsü, yemek, gıda içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, kuş, grafik tasarım, grafik içeren bir resim

Açıklama otomatik olarak oluşturuldu

“Fig 5 Profile Page of InsuChef”

“Fig 4 Home Page of InsuChef”

“Fig 3 Launch Page of InsuChef”

metin, elektronik donanım, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu

“Fig 6 Meal Selection Page of InsuChef”

“Fig 8 Calculation Page of InsuChef”

“Fig 7 Distribution Page of InsuChef”

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, elektronik donanım, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

“Fig 9 Food Addition Page of InsuChef”

“Fig 11 Show Favorites Page of InsuChef”

“Fig 10 Change Favorites Page of InsuChef”

# 2.6 Final Reflections

Although we’ve dealt with unexpected issues throughout the project’s process, by researching, we’ve managed to solve the problems, which made working on this project really enjoyable. Since both Android Studio and database structures were concepts we weren’t familiar with, learning about and figuring out how to use them were the steps that we had the hardest time with. But with working together and helping each other we were able to come through the difficulties we faced and achieve our goals. The reports and presentation also made it easier for us to put everything together. By working on this project, we gained a lot of new skills and improved our skills from the lectures.

# 

**3. Summary & Conclusions**

In conclusion, InsuChef is an app that’s designed to help people with diabetes by calculating and keeping track of their carbohydrate intake and also by calculating the amount of insulin needed. Since diabetes is a disease that can affect people from any age, our application’s interface is simple and user-friendly. To create this app, we divided tasks into sub-tasks several times and distributed them between the group members. After everyone managed to do their parts, we were able to create our app InsuChef successfully.

**References**

1. “MVC Framework - Introduction” Tutorials Point, [www.tutorialspoint.com/mvc\_framework/mvc\_framework\_introduction.htm#:~:text=The%20Model%2DView%2DController%20(,development%20aspects%20of%20an%20application](http://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm#:~:text=The%20Model%2DView%2DController%20(,development%20aspects%20of%20an%20application)