

TECHNICAL SPECIFICATIONS

SMART AR SHOPPING

Contents

1.Summary

2.Software & assets required

3.Specifications

3.1. Login and access

- Login using E-mail, Username & Password
- PlayFab's SDK for authentication

3.2. Registering a User

- Register using E-mail, Password & Username

3.3. Taking user input

- User input by selecting toggles
- Storing the options selected in an array

3.4. Algorithm and code

- Algorithm and code for finding shortest path

3.5. Adding virtual elements in AR camera

- Adding virtual arrows in front of AR camera
- Use of AR foundation and AR core

3.6. Showing the shortest path to user

- Showing directions in the AR camera

4.Additional specification

SUMMARY

Smart AR Shopping is an app which can be used to find the shortest possible path inside a mall or supermarket passing through all the places where user has to buy something. The path will be shown to the user by the means of arrows in his/her virtual camera.

For using the app user has to register once before using it for shopping purpose. The app uses **PlayFab**'s user authentication to authenticate the user and save the login info. Once the user gets registered he will be directed to an options menu containing the list of all available stores in the mall as options. Now user has to select the things that he wishes to purchase from the mall. After selecting the options user has to click on the **SUBMIT** button in order to proceed further.

After this user's mobile camera will be opened and as he will see his surrounding in the camera arrows will be shown to him in order to direct along the shortest path passing through all the stores where user has to buy something. Now user has to follow the directions shown by arrows in his mobile camera and he will be notified by a message or alert tone as he reaches the stores where he is supposed to buy something.

SOFTWARE AND ASSETS REQUIRED

For development of this app, following software and SDKs are required:

1.Unity: Unity game engine is the most important software required for development of this app as whole app development will take place on this platform.

2.Unity editor extensions for PlayFab: It's required in order to get started with PlayFab and it's sdk for app development.

3.PlayFab's SDK: This is useful for authentication purpose i.e. for register/login of user and saving login info.

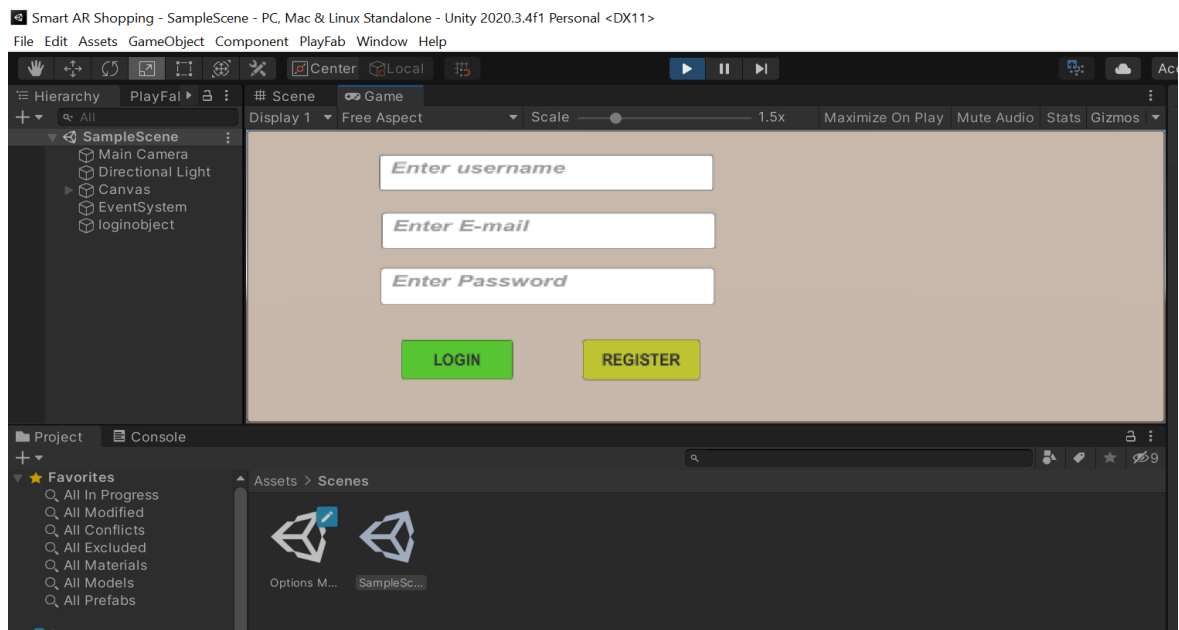
4.C# Scripts: For setting behavior of game objects using code in c# language.

5.Visual Studio IDE: This IDE is required for writing c# scripts .

6.AR core XR plugin: this helps in adding virtual elements in AR camera and making AR apps in unity.

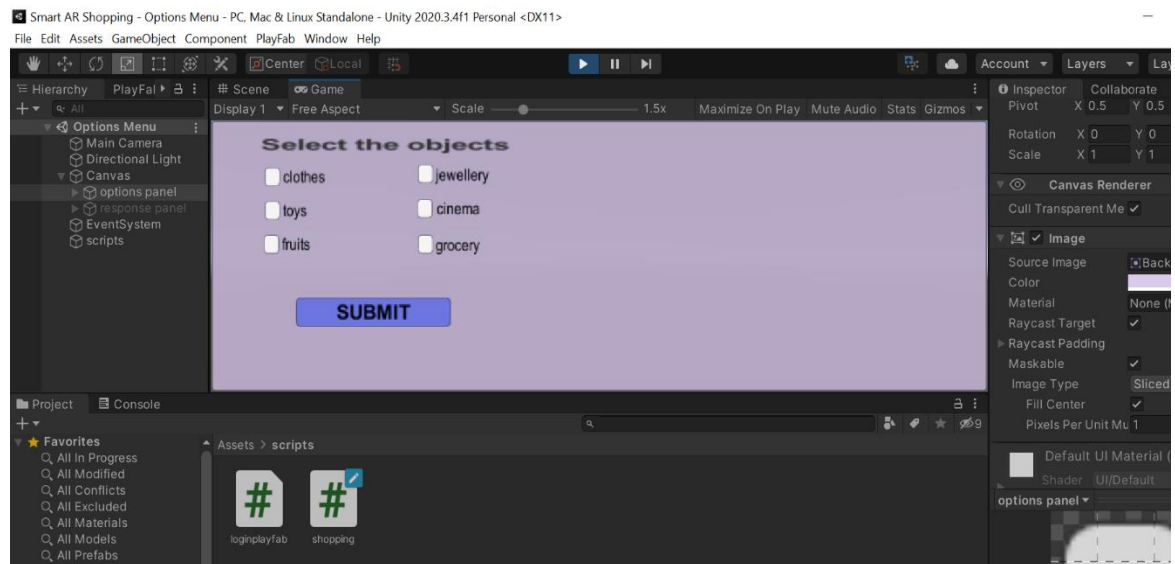
SPECIFICATIONS

- **Login and access:** Already registered users will have access to login to the app and after that he/she can use it for shopping purpose.
- **Registering a User:** A user can register himself in the app by providing a **username**, an **e-mail address** and a password. Once the authentication is done user's login info will be saved in the PlayFab and can be reviewed by the owner of the app.



- ❖ After successful login/registration user will be directed to an options menu where he has to give his input by selecting options.
- **Taking user input:** In the options menu user has to select those stores where he has to buy something. For showing options to user Toggles can

be used in unity which provide the feature of selecting multiple options at a time. After user clicks the submit button the code in c# script will read those options by checking whether the options are marked or not. The Toggles which will be marked by user their corresponding **Label** will be stored in an array of strings.



- **Algorithm and code:** For finding the shortest path passing through all the stores chosen by user, we can assign coordinates to all the stores and taking origin at the initial position of the AR camera i.e. entry gate of the mall and storing them in a **2-D array** (for storing both x and y coordinates). Now the code will start from origin and calculate the **distance** (not displacement) of all other points and hence it will find the point nearest to the origin. The same process can be repeated for each point where the user goes.

For reading the input of user all the store's names will be stored in an array of strings and after creating the array of strings as per the options chosen by the user it can be searched that which strings are chosen by the user. And after that coordinates for those stores can also be find out by searching in the 2-D array of coordinates and can be arranged using algorithm in order to generate shortest possible rectangular path.

A sample is shown below that how algorithm works!!

2-D array storing the coordinates of each store {(0,0) is the starting point}

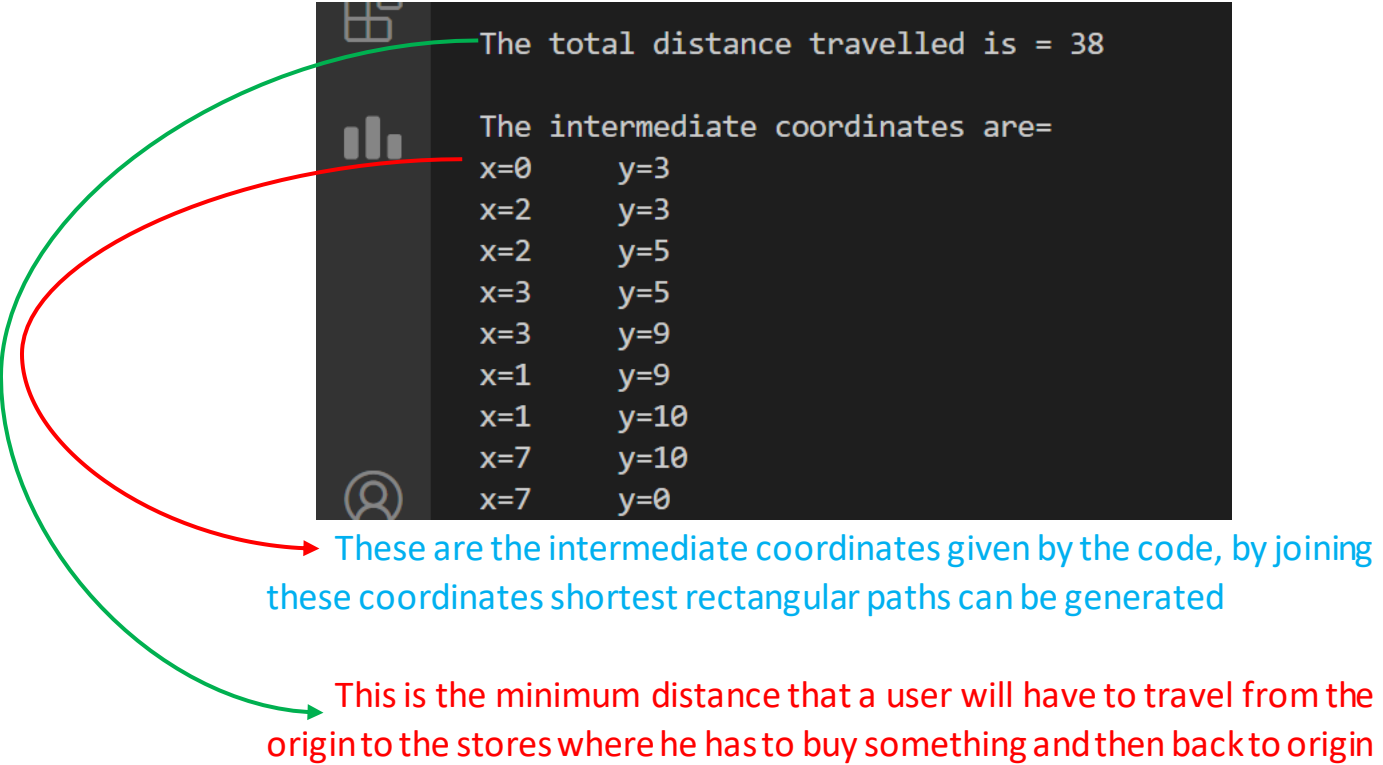
Array of strings in the code, storing the names of stores available in the mall

```
13 string s[7]={"fruits","vegetables","clothes","toys","jewellery","cinema","gaming"}; //array of string for stor
14 cout<<"enter the no of item:";
15 cin>>m;
16 string s1[m]; //array of string for taking input from user
17 for(int i=0;i<m;i++)
18     cin>>s1[i];
19 int arr[8][2]={{0,0},{1,9},{2,3},{4,6},{3,5},{7,10},{5,8},{12,2}}; //2-D array for storing the coordinates of
```

Array of strings for storing user input

```
enter the no of item:4
fruits
toys
vegetables
jewellery
```

User Input (that will be taken by the means of toggles in the actual code)



```
The total distance travelled is = 38
```

```
The intermediate coordinates are=
```

```
x=0      y=3
```

```
x=2      y=3
```

```
x=2      y=5
```

```
x=3      y=5
```

```
x=3      y=9
```

```
x=1      y=9
```

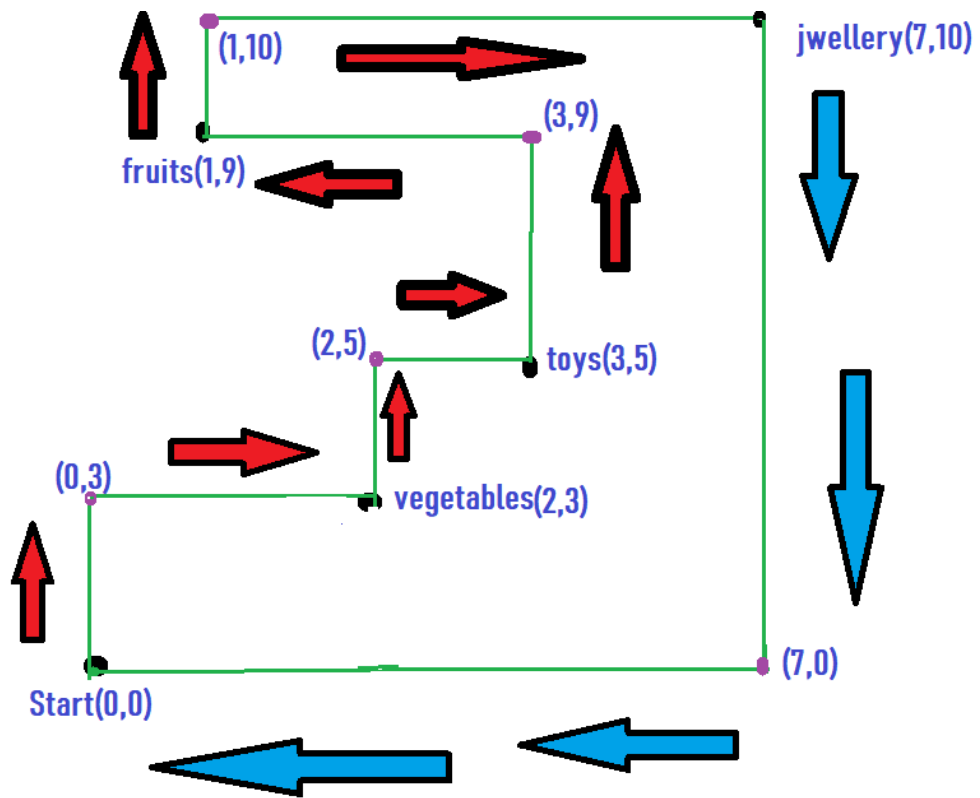
```
x=1      y=10
```

```
x=7      y=10
```

```
x=7      y=0
```

These are the intermediate coordinates given by the code, by joining these coordinates shortest rectangular paths can be generated

This is the minimum distance that a user will have to travel from the origin to the stores where he has to buy something and then back to origin



After joining all the intermediate pts given by code and the origin such a rectangular and shortest path can be generated

- **Adding virtual elements in AR camera:** This can be done using AR Foundation and ARCore XR plugin. Now we set the AR session origin which also includes the AR camera and the remaining coordinates are fixed with respect to this origin. A game object containing c# script will be used to set the coordinates and directions of virtual arrows and other virtual objects in front of AR camera then image sources of the virtual objects will be included in that gameobject.
- **Showing the shortest path to user:** After getting the intermediate coordinates from the code, these coordinates will be fixed in front of AR camera and directions can be shown to the user in the form of arrows as shown in the above figure.

ADDITIONAL SPECIFICATIONS

Adding an alert message or tone:

In the app an additional feature of showing an alert message or an alert tone can be added. It will be shown when the user will be near a place where he has to buy something.

Tracking User's current location:

This feature will record user's current location in order to show him where he had reached up to that time.

Adding data of multiple malls:

This feature will enable the user to use the app in multiple malls for shopping.