DOKUZ EYLÜL UNIVERSITY
ENGINEERING FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

AUGMENTED REALITY IN REAL ESTATE MOBILE APPLICATION

by
Şefika Özlem PUL
Ece KAZAN

Advisor
Asst. Prof. Dr. Yunus DOĞAN

June, 2022
İZMİR

AUGMENTED REALITY IN REAL ESTATE MOBILE APPLICATION

A Thesis Submitted to the
Dokuz Eylül University, Department of Computer Engineering
In Partial Fulfillment of the Requirements for the Degree of B.Sc.

by
Şefika Özlem PUL
Ece KAZAN

Advisor
Asst. Prof. Dr. Yunus DOĞAN

June, 2022
İZMİR

SENIOR PROJECT EXAMINATION RESULT FORM

We have read the thesis entitled **"AUGMENTED REALITY IN REAL ESTATE MOBILE APPLICATION"** completed by **Ece KAZAN** and **Şefika Özlem PUL** under advisor of **Assist. Prof. Dr. Yunus DOĞAN** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of B.Sc.

.....

Advisor

.....

Committee Member

.....

Committee Member

Prof. Dr. Yalçın ÇEBİ

Chair

Department of Computer Engineering

ACKNOWLEDGEMENTS

We would like to thank our esteemed consultant Yunus DOĞAN, who did not leave us alone during this difficult process of the graduation project, did not spare us his support and knowledge in every matter, found a solution to every problem and constantly motivated us.

We would like to thank all of our teachers who tried to find answers to the questions we asked about the project when appropriate.

We would like to express our gratitude to our esteemed families who shared with us the stress, anxiety, worry, sadness and joy we experienced during this project and supported us.

We would like to thank each other for completing this project successfully as a team, as it should be.

Ece KAZAN
Şefika Özlem PUL

AUGMENTED REALITY IN REAL ESTATE MOBILE APPLICATION

ABSTRACT

The aim of the AUGMENTED REALITY project in the REAL ESTATE MOBILE APPLICATION is to see the postings added to the android mobile application on the camera screen of the phone using AR technology at the specified location. In today's technology age, people carry out almost all their work with smartphones. Augmented Reality also allows the physical environment of the real world to be combined with the virtual environment via the phone. Therefore, the project was developed by combining augmented reality and mobile application. When the user opens the application via the mobile application, he sees the advertisements added to the home page as listed. When you click on any ad, the ad's information, pictures and location on the map are displayed. You can add the advertisement to your favourites, and view your favorite advertisements on a separate page. The user can update the postings he/she has added. With the AR application, the advertisement information can be seen in the marked position with the phone camera.

For the AR application, the object on which the advertisement details are displayed was placed in certain locations and it was tested whether the location information was correctly detected. Later, thanks to the database connection, the posting information and location information added to the android application are received by the AR application. Thus, objects are dynamically created for the advertisements recorded in the database. Advertisements are reflected on the screen with the camera angle according to the distance between the designated phone and the advertisement location.

A common database was needed in order to use the posting and location information added to the Android application on the Unity side. For this, the database was shared in both applications with a remote database connection using a server. Information that is not required to be used by Unity is kept in the SQLite database.

EMLAK MOBİL UYGULAMASINDA ARTIRILMIŞ GERÇEKLİK

ÖZET

Emlak Mobil Uygulamasında Artırılmış Gerçeklik projesinin amacı, android mobil uygulamasına eklenen ilanları, belirtilen lokasyonda AR teknolojisi kullanılarak telefonun kamera ekranında görülmektedir. Günümüz teknoloji çağında insanlar neredeyse tüm işlerini akıllı telefonlar ile yürütmektedir. Artırılmış Gerçeklik, gerçek dünyanın fiziksel ortamının telefon aracılığıyla sanal ortamla birleştirilmesine de olanak tanır. Bu nedenle proje, artırılmış gerçeklik ve mobil uygulama birleştirilerek geliştirildi. Kullanıcı, mobil uygulama üzerinden uygulamayı açtığında ana sayfaya eklenen ilanları listelenmiş olarak görmektedir. Herhangi bir ilana tıkladığında ilanın bilgilerini, resimlerini ve harita üzerindeki konumu görüntülenir. İlanı favorilerinize ekleyebilir, favorilediği ilanları ayrı bir sayfada görüntüleyebilirsiniz. Kullanıcı kendi eklediği ilanları güncelleyebilir. AR uygulaması ile ilan bilgilerini telefon kamerası ile işaretli konumda görülebilmektedir.

AR uygulaması için öncelikle ilan detaylarının görüntülendiği nesne belirli konumlara yerleştirildi ve konum bilgilerinin doğru algılanıp algılanmadığı test edildi. Daha sonra veritabanı bağlantısı sayesinde android uygulamasına eklenen ilan bilgileri ve konum bilgileri AR uygulaması tarafından alınmaktadır. Böylece veri tabanına kaydedilen ilanlar için dinamik olarak nesneler oluşturulmuştur. Belirlenen telefon ile ilan konumu arasındaki mesafeye göre ilanlar kamera açısıyla ekrana yansımaktadır.

Unity tarafında android uygulamasına eklenen ilan ve konum bilgilerinin kullanılabilmesi için ortak bir veri tabanına ihtiyaç duyuldu. Bunun için veritabanı, bir sunucu kullanılarak uzak veritabanı bağlantısı ile her iki uygulamada da paylaşıldı. Unity tarafından kullanılması gerekmeyen bilgiler SQLite veritabanında tutulmaktadır.

CONTENTS

Page

LIST OF TABLES **Page**

LIST OF FIGURES **Page**

CHAPTER ONE

INTRODUCTION

1.1. Background Information

The project is to develop a mobile application in the field of real estate using augmented reality. In today's technology age, people carry out almost all their work with smartphones. Augmented Reality also allows the physical environment of the real world to be combined with the virtual environment via the phone. Therefore, the project was developed by combining augmented reality and mobile application.

By using the developed mobile application, location information and phone camera, the information of the places for sale or rental was shown to the users. In addition, users were able to make their advertisement visible to other users with the advertisement sharing feature.

1.2. Problem Definition

Today, people find it difficult to find a place for sale or rent. People have to spend a lot of time and energy trying to find the advertisements with the features they want by walking around the street. People generally want to search for information and price about that place instead of immediately holding the place in the first advertisement they come across. While doing this, they try to reach that advertisement on other websites or applications because they do not want to disturb the advertiser. For this reason, they have difficulty in finding that advertisement among the many advertisement they have reached by filtering too much on other websites and applications. Since most people do not know the name of the street or neighborhood where the advertisement is located, they cannot use this information in this filtering feature, so they will encounter more than necessary advertisements. For this reason, they will spend a lot of time on websites and applications to find the

advertisement they are looking for. People have difficulty in finding advertisements that are similar to or close to the location of the advertisement they are interested in. When they go to see the details of the advertisement and they think that the place does not have the features they want, they are not aware of the existence of the advertisement with the desired features in the immediate vicinity. That's why they don't see those advertisements or they notice it later.

1.3. Motivation/Related Works

Amira B. Sallow, Sarkawt R. Hussain (2020) have developed a mobile application on real estate using artificial intelligence. While developing this application, MYSQL was used as a database, and a client server was used with PHP libraries while providing communication between users on the application. Users of the application can log in and out of the application and create their own user profile. Users can make a price offer for where to buy or sell. While the communication between the seller and the buyer is provided by chat or VoIP, their notifications are also sent to each other. There is no feature filtering, only feature selection when posting. There are buying and selling transactions within the application.

A. C. Aydinoglu, R. Bovkir (2020) have developed a mobile application that finds real estate using location information. While developing the application, the Java programming language was used in the Android Studio environment. It has been used as a database for PostgreSQL and PostGIS geographic datasets. On the main screen of the application, there is a map of the region where the user is located by detecting the location and a search box on the map. While using the search box, the user can perform the search by selecting the word or specific words he wants to search for. By using the user filtering feature, the values such as education, health, industry, population around the place with the desired features are displayed on the graph. In this way, users decide whether that place has the feature they want or not.

Daniel V. de Macedo, Maria Andréia F. Rodrigues, João J.V.P. Furtado, Elizabeth S. Furtado, and Daniel A. Chagas (2014) have developed a real estate mobile

application using augmented reality technology. This application programming language, which is based on IOS, is preferred as C language and PHP as server. The location of the user is determined by GPS, and the places that meet the filtering criteria or in the immediate vicinity are displayed on the map. Clicking on the desired advertisement displays the details of that advertisement. The user can add the advertisements he likes to his favorites and view them on the favorites screen. Searches and filters are recorded, so users don't have to choose filtering options again and again.

1.4. Goal/Contribution

In the application, the buyer only contacts the seller with the phone number in the ad. The reason why there are no features such as chat is that the chat feature can be used by users for other than its intended use and this situation is difficult to detect. If detected, it can cause serious problems for both users and app owners.

In the application, users want to add various features to the advertisement while adding an advertisement. Because when buyers look at the details of the advertisement, seeing various information about the advertisement accelerates the buyer's decision. For example, when general information such as property type and location is included in the advertisement, too many advertisement options are offered to the buyer, which causes the buyer's boredom and loss of time. Instead, having detailed information such as the age of the property, the floor it is located on, the type of heating makes it easier for the user to make a decision and saves time. In the application to be made, ease of use is aimed by considering both the buyer and the seller.

In the application made, it was aimed to show the nearest advertisements to the user by determining the current street location of the user. Thanks to the augmented reality with the phone camera, the advertisements on the street where the user is located were shown to the user with a sign. The display of advertisements to the user on a large map by the application did not provide any benefit for the user at that time.

Because this application is already designed for the user to see instant advertisements on the street or very near.

In the application, similar advertisements in the immediate environment that the user looks at are listed as suggestions and presented to the user. Thus, the user can look at these advertisements and get the chance to reach the desired ad.

1.5. Project Scope

In the project, users register to the application and create their own profiles. General information such as name, surname, e-mail address, phone number, password is requested from the user and if this information meets the verification criteria, it is saved in the database. When logging into the application, the registered e-mail address and password are requested from the user. If the e-mail address and password match in the database is correct, the user logs into the application.

Users add the advertisement of the place they want to sell or rent to the application and share this advertisement with other users. Various information such as picture, location information, price, square meter of the advertisement are filled in by the user and this information is saved in the database.

In the project, augmented reality was used to display advertisements using location information where the person was pointing at the phone camera. In order to see the advertisements, aim the phone camera in the desired direction, close to the user's location, and if there is any advertisement in the camera's field of view, the advertisement details are displayed where the advertisement is.

1.6. Methodology/Tools/Libraries

Unity has recently been used for Augmented Reality technology that allows the user to interact between the real world and the virtual environment. Unity is a game engine used to develop video games and simulations for computers, consoles, mobile

devices. Unity game engine; It is also used by different industries apart from video games such as the movie industry, automotive industry, architecture, engineering and construction. Unity is a development platform that supports 2D and 3D graphics, drag and drop functionality, and the C# programming language. It also offers the opportunity to develop applications for mobile devices. Therefore, the Augmented reality and other functions of the mobile application for the project will be developed in the Unity environment with the C# programming language.

It is an open-source, object-oriented, multi-functional, high-level language, high-throughput, platform-independent, step-by-step application. Java is a program that can run independently of any computer architecture and platform. In the project, various libraries and methods in Java programming language will be used to detect location information, display the street and mark the places where the advertisements are located in Android Studio.

Android Studio is an environment developed for programmers. In this environment, software developers can create applications suitable for devices with Android operating system. Java is used as the programming language in Android Studio. Also, Android Studio's operating system is Cross-platform. This means multiplatform software. In this way, software distribution can be realized for more than one operating system.

It will be used in the project and the data received from the user will be stored in a database. SQLite was used for database operations in the project. SQLite is basically a system where data is stored. It is an enterprise-wide relational database management system that allows data to be stored and accessed by multiple users at the same time. Transactions were made by connecting to the Java programming language SQLite database.

CHAPTER TWO

LITERATURE REVIEW

Since augmented reality is a new technology, it has attracted attention in recent years, and accordingly, many applications have been developed using augmented reality and a lot of work has been done on this subject. Successful applications have been developed in the field of real estate by using camera, GPS and navigation technologies. In addition, augmented reality applications have been successful and beneficial in fields such as education, health, tourism and entertainment. These applications have become more effective and useful with classification and clustering algorithms. Thanks to the classification algorithms, a more comfortable use is provided by helping the user with features such as making predictions and offering suggestions within the application. Thanks to the clustering algorithms, the results for filtering and user requests are displayed within the applications.

Bhorkar, G. (2017) has conducted studies and research to develop augmented reality navigation systems. Maps and navigation systems with different technologies have been developed to make them more useful and robust. In an augmented reality navigation system, the screen should not impede or distract the user. In augmented reality, there are many image options on how to transfer images to the user. Different image options such as video see-through, optical see-through, projective are preferred in different areas. When considering navigation for the car, the optical see-through display model is recommended, in which the car's windshield acts as a binder. For pedestrian or indoor navigation, video see-through is recommended. In augmented reality navigation, direction sensors are used for augmented reality to work properly when the smartphone is pointed anywhere. Location information is obtained with GPS sensors, or location information is obtained by making calculations using the direction and accelerometer sensor data of the smartphone. The real-world view is obtained with the camera on the smartphone. The navigation information created afterwards is combined with the camera and displayed on the smart screen phone.

Brata, K. C., & Liang, D. (2019) did a study with location-based augmented reality and location marking. Location-based augmented reality allows correlating the physical location coordinate with data processing environments allowing the user to see the POIs and descriptions of a particular landmark in a real environment. The aim is to use location-based augmented reality in navigation to reduce pedestrian travel time and greatly reduce user action, a user journey mapping method is proposed. Since there will be more than one landmark and information for these places on the screen, it may not be very confusing and useful for those who do not know that area. It is a journey map application that quickly transforms the user experience into a tool for designing its prototype with mobile location-centered augmented reality that will enable it to intuitively find its destination. Augmented reality application is not a virtual or magic application, the environment used is physical and real life. User tested for navigation and augmented reality performance. Blind search and guided scenarios are tested with both standard augmented reality and the proposed prototype. As a result, since the user only follows one point to get to the destination, the user will see the next waypoint and final destination with navigation in the augmented reality prototype. With standard location-based augmented reality, there are no right and left turn signs when the user is going to the destination.

Lam, K. Y., Lee, L. H., & Hui, P. (2021, October) have developed an augmented reality supported web browser (AR To Web) using a mobile headset with augmented reality technology. Augmented Reality (AR) enables the interaction between physical life and the digital world. AR To Web (A2W) is offered to provide user-centered web browsing driven by Augmented Reality headsets. AR To Web browser displays AR-supported web content to the user using a content-based filtering model. When a user with an AR headset encounters the places and objects he encounters in his daily life, this data is recorded by the interaction with the headset. User-based web browsing on mobile headsets provides a personalized web experience enabled by MAR (Mobile Augmented Reality). As a result of the studies, it has been observed that the A2W context-sensitive web experience performs better than smartphones, and accordingly Augmented Reality web browsing offers a more useful

opportunity. It has been deemed appropriate to use the WebXR API standard to provide a uniform abstraction layer for interacting with the interacting devices in the A2W architecture and for its real-time operation. Hand gestures and other gestures supported by GestureML are controlled. The movements of the user on the web page such as page changes, clicks, scrolling up and down the page are recorded and analyzed and presented to the user as a suggestion according to the user's interaction with the web content.

Asraf, S. M. H., Hashim, A. F. M., & Idrus, S. Z. S. (2020, April) aimed to develop an outdoor navigation mobile application using location-based services and augmented reality technologies. Many applications have been developed with the use of navigation and augmented reality in location-based services (LSB). There are processes commonly used by developers in the development of Augmented Reality. At the System Planning stage, the design adjustment is made, the tools and equipment to be used in the software are determined at the AR Tools stage (For example, Unity AR, GPS Location), the augmented reality application made in the Augmented Reality Apps stage is exported. Augmented Reality location-based services require permission access to pre-installed sensors from the mobile device. Unity is a platform that provides 3D modeling, Augmented Reality GPS location transfer to mobile devices. In Unity AR GPS software, latitudes and longitudes are shown clearly for better understanding of the location. The reason why the IOS platform is preferred in location production in AR GPS to develop applications on smartphones is that Iphone's ARCore SDK is more compatible with Unity AR. As a result, the techniques used in GPS location determination can be applied in augmented reality.

Jia, J., Elezovikj, S., Fan, H., Yang, S., Liu, J., Guo, W., ... & Ling, H. (2021) used Street view, tag placement technology for AR with scenarios of various events. A special map, which is a guide map, is placed with labels identifying important regions and places. When a map or image is added as input, its specific information is aligned with the map thanks to augmented reality. There is also a tagging system to tag the places that the user deems important. Computer-generated tags in the AR

app are saved with their geolocation as points of interest along with the GPS location. AR tech needs to see the main subject because of the clutter in the background. Data selection and simulation with AR-Head Up Display (AR-HUD) is used in AR-based navigation to indicate places of interest such as markets, restaurants, hospitals, banks by marking them on the map. Thanks to the AR navigation system, restaurants, cafes and places of interest on the street are labeled and displayed to users. This data selection process can be a photograph, such as a cityscape or a street view. Therefore, it does not necessarily have to make markings on the image of a street.

Cui, B. B. (2017) has designed the movie recommendation system using the Knn classification algorithm and analyzed the results by applying it. Personalized recommendation systems are important in terms of facilitating the user's choice when the user does not have a clear choice. Information such as what kind of person the user is, what he likes, what kind of things he likes and how he makes choices in which situations are the information analyzed in the recommendation systems. Personalized recommendation systems carry the filtering feature visibly, and for example, while the user is choosing a movie, all kinds of movies are presented to him, but with the recommendation systems, movies that appeal to the user are shown to him. While developing this system, the KNN classification algorithm greatly helps. While making KNN classification, suggestions can also be made according to the similarity between users. The data to be analyzed is kept in the MYSQL database and the calculations are made according to the formula of the KNN algorithm, taking into account the 'K' value, which shows the most similarity, that is, the results that show the least similarity, and as a result, a suggestion is presented to the user.

Bao, L.Q., LV, L.X., & Li, J.L. (2017) optimized the Naïve Bayes algorithm for SMS spam filtering on mobile phone to reduce resource consumption. Nowadays, mobile phones receive unnecessary spam messages and they need to be filtered. Although the spam filtering algorithms for SMSs sent to mobile phones are successful in spam filtering, the effect of mobile phones on the consumption of hardware resources has not been emphasized. For this, Naive Bayes algorithm is

proposed for SMS spam filtering in order to minimize the consumption of hardware resources of the mobile phone and to reflect the personal preferences of the user. As a result of the examinations, it was concluded that the algorithm can be widely used on mobile platforms, while maintaining its accuracy in classification, accelerating the classification work and reducing the storage space. In the SMS spam filtering experiment conducted in China, a data set consisting of 706 spam messages and 813 normal messages was used. Depending on the results of the examination, a graph was created and some analyzes were made. With the optimized Naive Bayes SMS spam filtering algorithm, it has been observed that while the number of attributes of each message is between 9 -13, it provides an accuracy rate of over 90%, while the accuracy rate increases further and reaches the maximum value of 11 when the number of attributes increases. If the number of features is higher than 13, the accuracy rate decreases, and in this case, the performance of the filtering algorithm also decreases. If we want to achieve SMS spam filtering with high classification accuracy, low hardware resource consumption, and low processing time, it would be appropriate to use the Naive Bayes algorithm.

Andri, C., Alkawaz, M.H. and Sallow, A. B. (2018) have developed a campus tour mobile application using augmented reality technology. Today, many universities use the mobile campus tour application using augmented reality technology to promote their campus. This application includes features such as navigation, location search, 3D campus tour and virtual genre. AR technology combines virtual objects or information with the real environment through computer graphics technology, thus making it easier for people who come to visit the campus. It provides a 3D tour service of buildings and artifacts to campus visitors using HMD, a display with augmented reality GPS and orientation tracking. Location information is obtained via the internet using GPS to find specific points (POIs) within the campus. 3D navigation apps like Sunmap+ create a real-time virtual tour guide with a smartphone camera. The camera will place live step-by-step navigation indicators and this application will direct the user to the user's target by monitoring the user's movements. For example, when the user points the phone camera towards the library building, a marker for the library building appears on the screen, and when this

marker is clicked, the activity details about the library are shown to the user. The user scans the building during the campus visit, and when this image matches the database, the user is shown the building information and a 360-degree virtual view.

Chung, C. O., She, Y., & Jung, H. K. (2016) have minimized augmented reality-based navigation system errors on smartphones supported by android operating systems and presented the user with reality information about their location. With the rapid development of smart phones today, internet access is facilitated, and users can use the virtual tracking technology of GPS location-based technologies to reach basic information about a building they are looking for or in their immediate surroundings. With the Augmented Reality technology, certain signs have been placed on the users' smartphones while the places and areas are displayed on the screen. By clicking on these signs, while the information is presented to the user, they will not be able to choose to get information about the areas or places that are not displayed on the screen. Augmented Reality is divided into two as marker augmented reality and unsigned augmented reality. The marker uses ARToolkit, while augmented reality magnifies the object using a special marker, allowing for easier calculation of coordinates. By adapting the mobile device to various other devices and sensors via the Android operating system API, it can display information such as location, direction and inclination degree on the user's device with GPS, camera, digital compass and sensors via these APIs. The general name of the buildings and stores displayed on digital maps such as navigation is called POI, and Google Earth, Google Maps and KML are used to obtain POI data. Augmented Reality technology and navigation systems can provide viewable information to the user not only about a limited road, but also about the current location and surrounding places/buildings.

Köse, M., İncel, O. D., and Ersoy, C. (2012, April) have identified the most useful algorithm when using online classification algorithms for people's activities in daily life on smart phones. Naive Bayes, Clustered KNN and KNN classification algorithms are used for online activity recognition and improvement on smartphones. Basic movements of the user such as running, walking, sitting and standing are tried to be detected and classified using accelerometers embedded in smartphones. By

testing the classification algorithms on five different subjects, the clustered KNN approach performed better than other classification algorithms in terms of processing, time and accuracy. KNN (nearest neighbor algorithm) is not very suitable for use alone in online classification, which requires high computational processing and limited resources in smartphones. Instead, in the clustered KNN algorithm, the data is first processed as mean, minimum, maximum and standard deviation, and the second step is classification according to the obtained values. The obtained values are compared with the values in the data set one by one at the classification stage, and after the closest K sample is selected, we label the data as the activity for which we have the most data in the last K set. The calculated standard deviation of the activities and the standard deviation values from the relevant part are compared and the activity is selected as the closest standard deviation result. Looking at the processing time of the clustered KNN performance, when the K parameter decreases, the classification time is greatly reduced. When examined in terms of resource consumption, the same amount of memory usage is observed in the clustered KNN and Minimum Distance classifier, while the Naive Bayes algorithm has a very high CPU usage. When all the evaluations are considered, the clustered KNN algorithm provides much higher performance than the Naive Bayes algorithm in terms of accuracy and performance on mobile platforms.

Skorokhodov, D., Seliverstov, Y., Seliverstov, S., Burov, I., Vydrina, E., Podoprigora, N., ... & Cheremisina, A. (2018, November) have developed an application for marking the routes of public transport vehicles with augmented reality mobile technology. The reason for this is to reduce the confusion of a person who does not know or does not know that city, region, about choosing a vehicle while using public transportation. If a trist wants to use public transport, the public transport to any bus stop tells him where he wants to go and asks if public transport will go there, or he tells a person at the stop where to go and tries to go. Find out which public transport number to go to. However, in order to prevent this, a phone camera will be kept on the number of the public transportation vehicles coming to the stop, so that the route of that public transportation will be marked on the map. In other words, the user's location is found by the mobile application. The camera of the

phone with the Android / IOS operating system is turned on and pointed to the number of the public transport vehicle and that number is introduced. The route of public transport is now displayed on the mobile device after the user checks the correctness of the vehicle number. Thus, it is observed that the quality of service in public transport journeys has increased with augmented reality technology. A mobile application was developed in Unity and Vuforia environments using Yandex API to mark the public transport route.

Carrasco, B., Guamán, V., Delgado, J., & Ruiz, F. (2019, November) have also thought of using it in the field of tourism as it brings the objects in the digital world to the real world with the augmented reality technology. It is used not only for entertainment purposes such as games, but also in many fields such as education, medicine and tourism. AR-TOUR application, prepared with augmented reality technology in tourism, allows visitors/tourists to discover touristic places and have information about that place. Thanks to this application, the person does not help in guiding the tourists in the city, thanks to the application, he has information about every part of the city. City attractions such as parks and museums will be displayed in 3D. Thanks to augmented reality, when a church is scanned with a camera, it is estimated which church is in the field of view of the camera, along with the surface, and information about the interior and exterior of the church is given. At the end of the touristic trip, users were evaluated about the accuracy of the application, and the application was improved over time. Selection menus of the application such as choosing a location and choosing a language were developed with Flutter and Vuforia, and the augmented reality part was developed with Unity.

Repaka, A. N., Ravikanti, S. D., & Franklin, R. G. (2019, April) thought that data mining applications in the medical field could benefit clinical diagnoses. With the development of technology, mobile health technology has also started to be used in the field of medicine. Diagnosis occurs with reference to previous data and information. Smart Heart Disease Prediction (SHDP) used Naive Bayes to predict disease diagnosis and risk factors. Information such as age, blood pressure, cholesterol, gender, blood sugar is collected to estimate the probability of heart

disease. This information is entered from the Naive Bayes classification and 80% of this data is used for decision and 20% for testing. The output is an estimate of heart disease risk factors. Heart disease prediction is made using data mining method. It is important to make a quality and closest diagnosis due to the degree of risk that will arise as a result of the Naive Bayes classification, the treatments to be given to the patient, the drugs to be given to the patient, and their doses. medications and recovery time are determined. As a result of this study, it worked successfully with Naive Bayes classification by providing high accuracy.

Meyer-Lee, G., Shang, J., and Wu, J. (2018, November) have identified the problems that cause location leakage through network traffic in mobile augmented reality applications and produced solutions to this issue. Recently, Mobile Augmented Reality applications allow users to interact with smart devices such as smartphones and tablets, between the real world and virtual objects. While augmented reality technology is in the fields of education, tourism, entertainment, health and virtual modeling, it is mostly used in game applications. The most popular of the location-based Mobile Augmented Reality games is Pokemon Go. In the game, AR puts virtual objects in the real world environment and gives the game very close to reality by giving geolocation. Researches have determined that there may be security and privacy violations related to the collection and storage of information from device sensors in location-based AR applications. Location-based augmented reality applications simply transmit data about the user's location, as well as virtual content, to a server to create it. Mobile AR application developers draw attention to packet-filling defenses to network traffic analysis attacks, and if the developers fix location-related downloads to a fixed size, a location analysis attack from network traffic can be avoided. In mobile AR applications, the hacked person has the potential to access and leak information about the neighborhood, city, or region. Developers of AR technology should take measures to protect network traffic from the physical location of their users and keep all the data they obtain within the framework of security and confidentiality.

Kurniawan, M. H., & Witjaksono, G. (2018) observed that while medical students were studying, they had difficulty in understanding the department they were studying. The reason for this is that they want to see that object in 3D in some subjects. For example, they want to learn by seeing the anatomy of the human body. Therefore, the purpose of this application is to visualize the anatomy of the human body in 3D for medical students with augmented reality technology. Photos are taken in augmented reality with the Phone camera. Then, this image is divided into parts and it is checked whether the parts match the images stored in the database. Thus, thanks to augmented reality, that object image turns into a 3-dimensional form. While human anatomy models are costly, the cost is minimized thanks to this application. At the same time, the user is informed about the human anatomy, which has become 3D thanks to augmented reality, and questions are asked about it. Thanks to this 3D anatomy created by AR, the information will stay in the minds of the students more. Augmented reality has also managed to work successfully in this area. Floating Euphoria Framework and SQL database are used.

López-Faican, L., & Jaen, J. (2020) aim to develop children's sensory intelligence, socialization and communication skills with the games developed in today's technology. In addition, it is aimed to develop basic behaviors such as competitive and cooperative behaviors of children over time. Thanks to the augmented reality technology, games and scenarios that will attract the attention of children will be created, and they will be able to develop their skills while having fun. This game is evaluated with a mobile augmented reality application containing scenarios. It harmonizes unsigned Mobile Augmented Reality (MAR) technology to create a geolocation with unlimited physical space with mobile augmented reality technology. Children perform some skills by predicting the emotions, movements and expressions of emojis created with augmented reality. As a result of this research, it was found that cooperative behavior developed more than emotions such as love and social. Unity is used as a game development platform and is compatible with Apple IOS smartphone as mobile device.

CHAPTER THREE

REQUIREMENTS/REQUIREMENT ENGINEERING

3.1. Functional Requirements

3.1.1. Application Landing Page

userMailAndPasswordCheck(): When the login button is clicked, the compatibility of the mail and password from the user is checked in the database. Boolean is returned as true for mail and password match and false if not matched.

clickLogin() According to the value returned from the 'userMailAndPasswordCheck()' function, if it's true, it stays on the homepage, if it's false, it stays on the same page.

clickRegister() : If the user wants to register for the application, it redirects to the registration page.

3.1.2. Register Page

mailControl(string mail) : It is checked whether the e-mail entered by the user is empty, whether there is an e-mail and whether it is in the database. A boolean value is returned as true or false.

nameAndSurnameControl(string name, string surname): Whether the name and surname entered by the user are empty, the character is checked. A boolean value is returned as true or false.

passwordControl(string password) : Checking whether the password entered by the user is blank, checking that it is a minimum of 10 characters, checking whether there is a minimum of one lowercase letter, uppercase letter and number, foreign character and space checks are made. A boolean value is returned as true or false.

clickSaveUserInfo(): If all of the mailControl, nameAndSurnameControl, passwordControl functions return true, the user's information is saved in the database and directed to the homepage.

3.1.3. Home Page

clickFilter(): It directs the user to the screen to select the desired features.

clickSort() : When the user clicks on the sorting combobox, he/she is confronted with ascending and descending options according to price. If he chooses ascending according to price, the listings are listed in ascending order of price. If he chooses the one that decreases by price, the listings are listed in descending order of price.

advView() : Clickable listing of advertisements on the homepage. If the user views the homepage, they are listed in the order of their addition, if he clicks on the "My Favorites" button, the advertisements that the user has added to his/her favorites are listed, if he clicks on the Private Me button, advertisements similar to the advertisements that the user has viewed before are listed, and as a result of the filtering process, the advertisements are listed.

clickAdv(int advId): The user is directed to the advertisement Detail Viewing Page in order to show the detailed information of the advertisement that she/he clicked.

3.1.4. Advertisement Detail View Page

getImage(int advId) : Displaying the photos saved in the database of the advertisement with the id taken as a parameter.

getInfoAdv(int advId) : Displaying the database saved information of the advertisement with the id taken as a parameter on the screen.

clickFavorite(int advId) : If the user has not added the advertisement with this id to their favourites, the advertisement with the id taken as a parameter is saved as the user's favorite advertisement with the user's id in the database. If the user has previously added the advertisement with this id to his favourites, the advertisement with the id taken as a parameter is deleted as the user's favorite advertisement with the id of the user in the database.

displayAdvLocationInMap(int advId) : The location of the advertisement on the map is displayed with the x and y coordinate information, which are the map information saved in the database of the advertisement with the id taken as a parameter.

3.1.5. Adding Advertisement Page

advInputControl(): While the user is adding an ad, it is checked whether the advertisement information is empty, foreign character control and integer whether it is a number. If there is no error in the checks made, a boolean value is returned as true. If any errors are detected, a boolean value is returned as false.

addAdvImage(): The user uses it to add photos of the ad.

predictionAdvPrice(string advStatus, string roomNum, int squareMeters, int floorLocation, ,int buildingAge, string warmType, int buildingFloors, string eligibilittForCredit, string itemStatus, int numberOfBathrooms, string buildingType, string stateOfBuilding, string usingStatus, int dues, string swap, string front, int rentalIncome, string fuelType) : The information that the user enters while adding an advertisement offers a price suggestion for the price of the advertisement with the classification algorithm.

markingAdvOfLocationMap (): The user marks the location of the advertisement that he has added on the map on the map. As a result of this selection, the x and y coordinates of the location are returned.

clickAdvInsert (): After all addition and control operations, all information is saved in the database.

3.1.6. Filtering Page

inputFilterControl(): It is checked whether the advertisement properties selected by the user are empty, foreign characters are checked, and integers are numbers. If there is no error in the checks made, a boolean value is returned as true. If any errors are detected, a boolean value is returned as false.

clickFilterAdv(): The advertisements in the database and the advertisements that match the advertisement features entered by the user are directed to the homepage and displayed there.

3.1.7. My Account Page

clickUserInfo(): Redirects to the page where user information is displayed.

clickUserAdvFavorite(): It directs the user to the page that lists the advertisements that the user has added to their favourites.

clickSpecialForMe(): It redirects to the page that lists the postings that the user has previously viewed.

clickMyAdv(): The user is directed to the page with the postings.

clickExit(): When the user logs out, she/he is redirected to the login page.

3.1.8. My Information Page

getUserInfo(): The user's information is displayed in the correct places in the input boxes.

inputUserInfoControl(): It is checked whether the data in the input boxes containing the information of the users contain foreign characters, the correctness of the mail and whether it exists in the database. As a result, a boolean value is returned as true if all inputs are true and false if false.

clickUpdateUserInfo(): If the value from the inputUserInfoControl function is true, the user information is updated in the database, if false, an error message is given.

3.1.9. My Advertisements Page

userAdvView(): Clickable listing of ads.

clickUserAdv(int advId): It is directed to the advertisement Detail Display Page to show detailed information of the advertisement that the user clicked.

clickUpdateAdv(int advId): The user is directed to the posting update page to update the post that he has added.

3.1.10. Advertisement Update Page

inputUpdateControl(): It is checked whether the data in the input boxes containing the properties of the advertisement contain foreign characters, whether they are empty and string integer. As a result, a boolean value is returned as true if all inputs are true and false if false.

clickUpdateAdv(): If the value from the inputUpdateControl function is true, the information of the announcement is updated in the database, if false, an error message is given.

3.1.11. AR Page

updateLocationFile(): The x and y coordinate information of the advertisements are taken from the database and written to the file in xml format.

3.1.12. Layout Class

Thanks to the "back" button in the upper left corner of the screen, it is directed to the homepage from all pages.

It is directed to the AR page with the "AR" button in the lower left corner of the screen.

With the "Add Ad" button in the lower middle of the screen, you are directed to the page for adding an ad.

It is directed to the My Account page with the "My Account" button in the lower right corner of the screen.

3.1.13. Database

3.1.13.1. Advertisements Table

AdvertisementId (integer primary key): The id of the advertisement

AdvertisementTitle (varchar): Title of the ad

AdvertisementStatus (varchar): Advertisement status

RoomNum(varchar): Number of rooms in the ad

SquareMeters(integer): How many square meters is the ad

FloorLocation (integer): Floor information of the ad

BuildingAge (integer): The age of the building where the advertisement is located

WarmType (varchar) : Warmup type

BuildingFloors (integer): Total floor number of the building where the advertisement is located

eligibilittForCredit (varchar): advertisement eligibility status for credit

ItemStatus(varchar): Item information of the ad

NumberOfBathrooms (integer) : Number of bathrooms in the ad

BuildingType(varchar): The type of the building where the advertisement is located

StateOfBuilding(varchar) : State of the structure where the advertisement is located

UsingStatus(varchar): Ad's usage status

Dues (integer): Dues information of the ad

Swap (varchar): Swap information of the ad

Front (varchar): Ad's front information

RentalIncome(integer): Rental income of the ad

FuelType(varchar): Fuel information of the ad

Price (integer): Price information of the ad

Date(date): The date the posting was added

Address(varchar): The address of the ad

xCoordinate(double): The x coordinate information of the ad

yCoordinate(double): The y coordinate information of the advertisement

3.1.13.2. UserInformation Table

UserId (integer primary key): User's id

Name(varchar): User's name

Surname(varchar): User's last name

Mail(varchar): User's mail

Password(varchar): User's password

3.1.13.3. Favorite Table

UserId(integer foreign key): User's id
AdvertisementId (integer foreign key): The id of the advertisement
FavoriteStatus(bool): The status of the advertisement in favorites

3.1.13.4. UserAdvertisement Table

UserId(integer foreign key): User's id
AdvertisementId (integer foreign key): The id of the advertisement

3.1.13.5. UserPastAdvertisement Table

UserId(integer foreign key): User's id
AdvertisementId (integer foreign key): The id of the advertisement

3.1.13.6. AdvertisementImage Table

AdvertisementId (integer foreign key): The id of the advertisement
ImageId(integer primary key): Photo id of the ad
ImageName(varchar): The photo url of the ad

3.2. Non-Functional Requirements

Performance and Scalability: Since there is not much data in the data set, functions such as filtering, sorting and listing are fast.

Portability and Compatibility: The software is developed and run in the Android Studio environment on the Windows operating system. The application can be transferred and run on phones with Android operating system.

Reliability, Availability, Maintainability: Since the data set of the application is not very large and there are not very detailed and tiresome functions on the application, critical failures are not experienced in the system.

Security: In the application, sensitive information is stored in the database, the personal information of the users and the information of the advertisements are not shared with anyone.

Usability: Our application has a user friendly interface. Switching between the screens in the application is easy, the most used functions are highlighted and added to a visible place that the user can easily reach, the operations to be performed on the buttons are clearly and clearly indicated.

CHAPTER FOUR

DESIGN

4.1. Architectural View

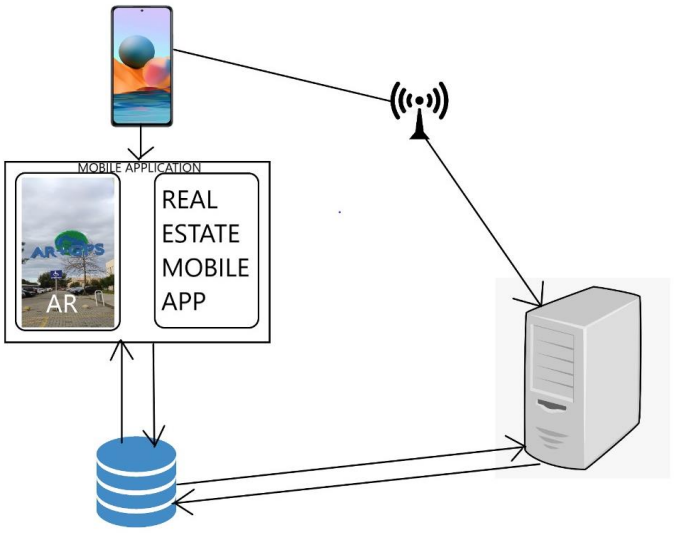


Figure 4.1.1 Architectural View of the Project

The architectural view of the mobile application project is as in Figure 4.1.1. The project has two main components. These are real estate and AR project. The data in the application is stored through the database. This mobile application connects to the database with the server.

4.2. Database Design/ER Diagram

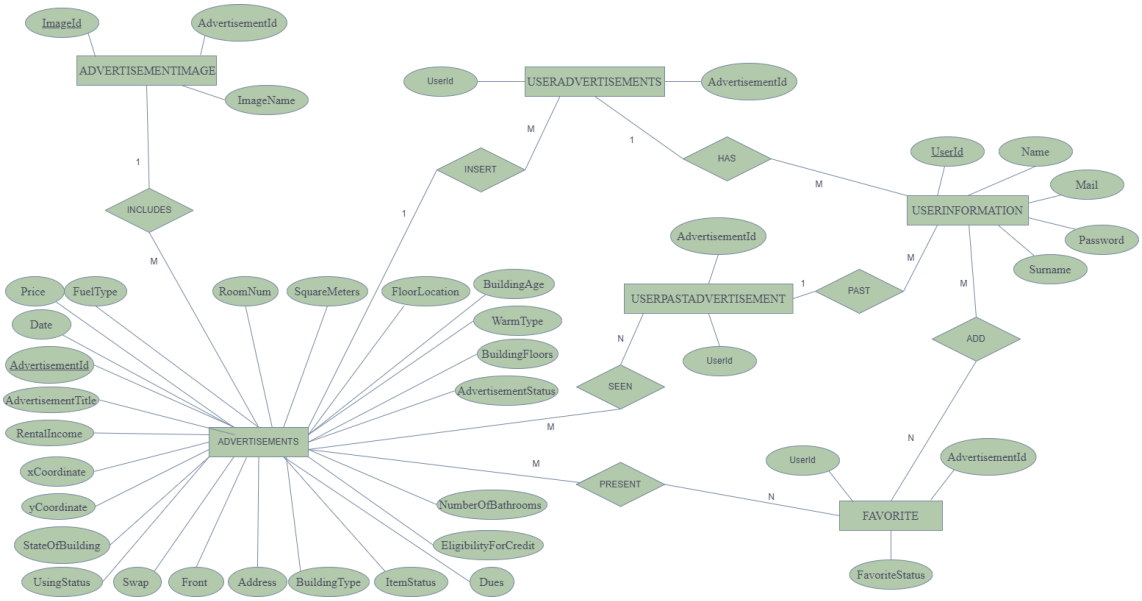


Figure 4.2.1 Database Design/ER Diagram of the Project

The database design/ER diagram of the mobile application project is as in Figure 4.2.1. The database of the project has six tables: Advertisement Table, Advertisement Image Table, User Advertisements Table, User Information Table, User Past Advertisement Table, Favorite Table. In the Advertisement Table, the information of the advertisements is kept. In the Advertisement Image Table, the names of the advertisement photos are kept together with the advertisement id obtained from the Advertisement Table. In the User Information Table, the information of the users registered to the application is kept. In the User Advertisements Table, the id of the user is kept together with the id of the advertisement that the user added. In the User Past Advertisement Table, the id of the advertisement that the user viewed before and the user's id are kept. In the Favorite Table, the user's id is kept together with the id of the advertisement that the user added to his/her favorite.

4.3. Use Case Diagram

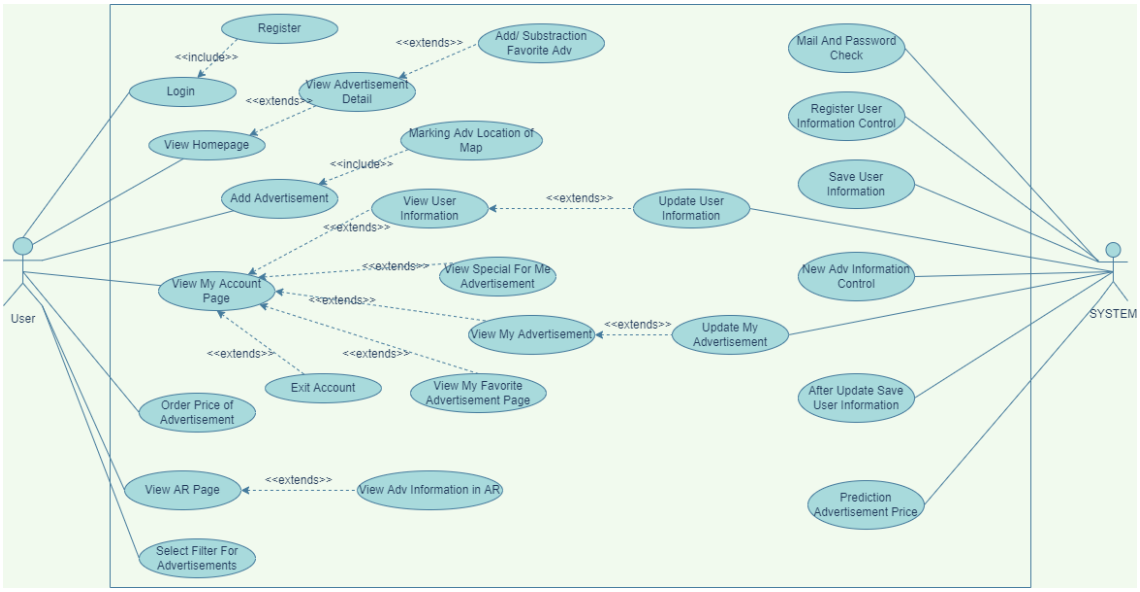


Figure 4.3.1 Use Case Diagram of the Project

The use case diagram of the mobile application project is as in Figure 4.3.1. There are user and system as actors in the project. If the user is registered to the application, the login process is performed, if not, it logs in after the registration process. You can select one of the advertisements listed on the homepage of the application, view the details of the ad, and add/remove it to your favourites. The user adds an advertisement and marks the location of the advertisement on the map. The user can view and update their own information. It displays the list of advertisements specific to the user, the list of advertisements that he/she has added and the list of advertisements he/she has added to his/her favourites. Thanks to the AR page, the user can see the advertisements close to his/her own location by means of a marker on the phone camera. The system checks the accuracy of all the information entered or updated by the user and saves it in the database.

4.4. Sequence Diagrams

4.4.1. Sequence Diagram of Add Advertisement

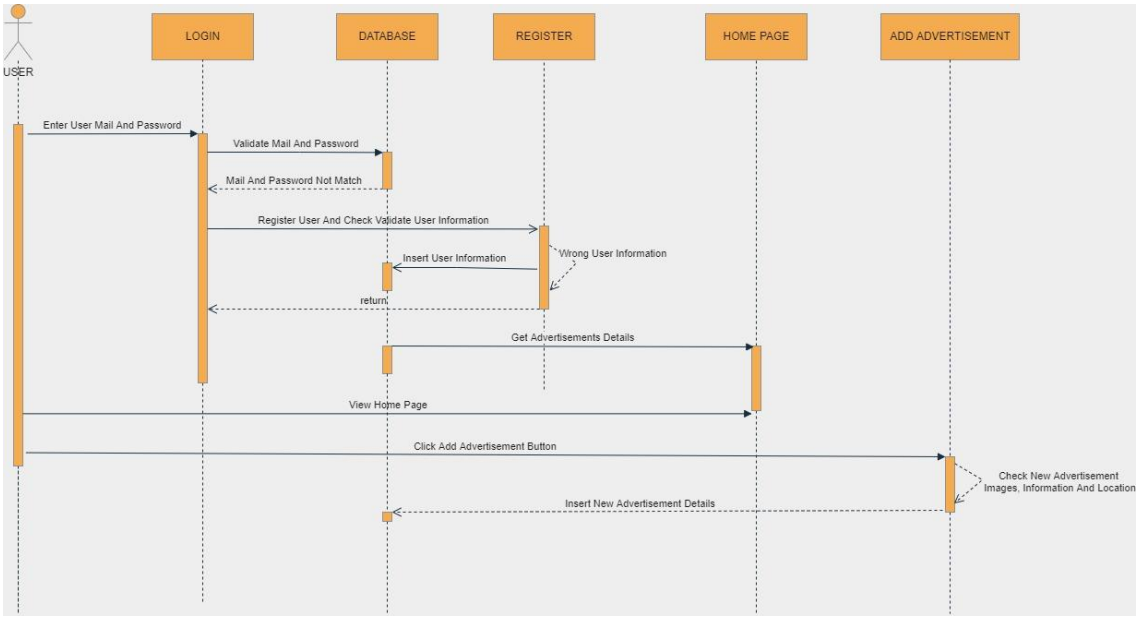


Figure 4.4.1.1 Sequence Diagram of Add Advertisement in the Project

The sequence diagram of add advertisement in the project is as in Figure 4.4.1.1. The user enters his/her mail and password to log in to the application. If the user's mail and password do not match in the database, he/she is asked to register. After registration, you are directed to the login screen. If the correct e-mail address and password are provided in the database on the login screen, it is directed to the homepage of the application where the advertisements are listed. The user clicks on the add advertisement button on the homepage of the application and displays the advertisement add page. On the advertisement add advertisement page, the user enters the advertisement's information, adds the photos of the advertisement, and after marking the advertisement's location on the map, all this information is checked. As a result of this control, if the information is correct, a new advertisement is added to the database.

4.4.2. Sequence Diagram of AR View

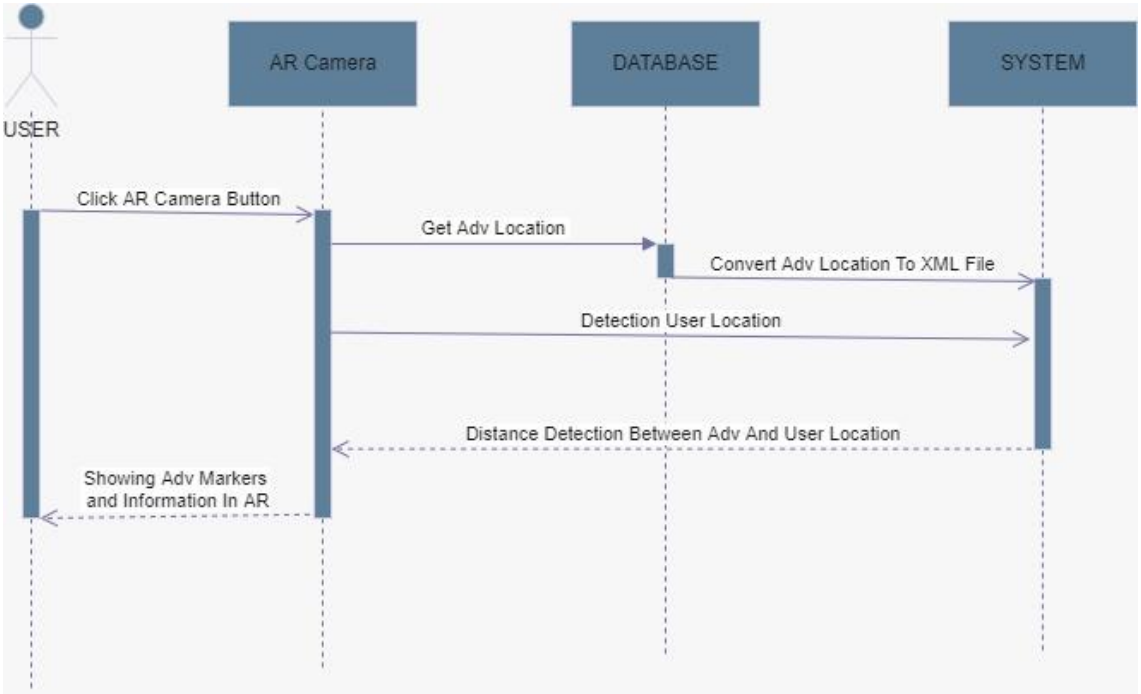


Figure 4.4.2.1 Sequence Diagram of AR View in the Project

The sequence diagram of AR view in the project is as in Figure 4.4.2.1. The user clicks on the AR camera button in the application. The location information of all the advertisements' x and y coordinates is retrieved from the database. The location information of the advertisements is converted to XML file. The AR camera detects the location of the user. The distance between the location of the advertisements and the location of the user is calculated. As a result of this calculation, advertisements close to the user and at the camera angle are shown to the user with markers on the AR camera.

4.5. Activity Diagrams

4.5.1. Activity Diagram of Add Advertisement

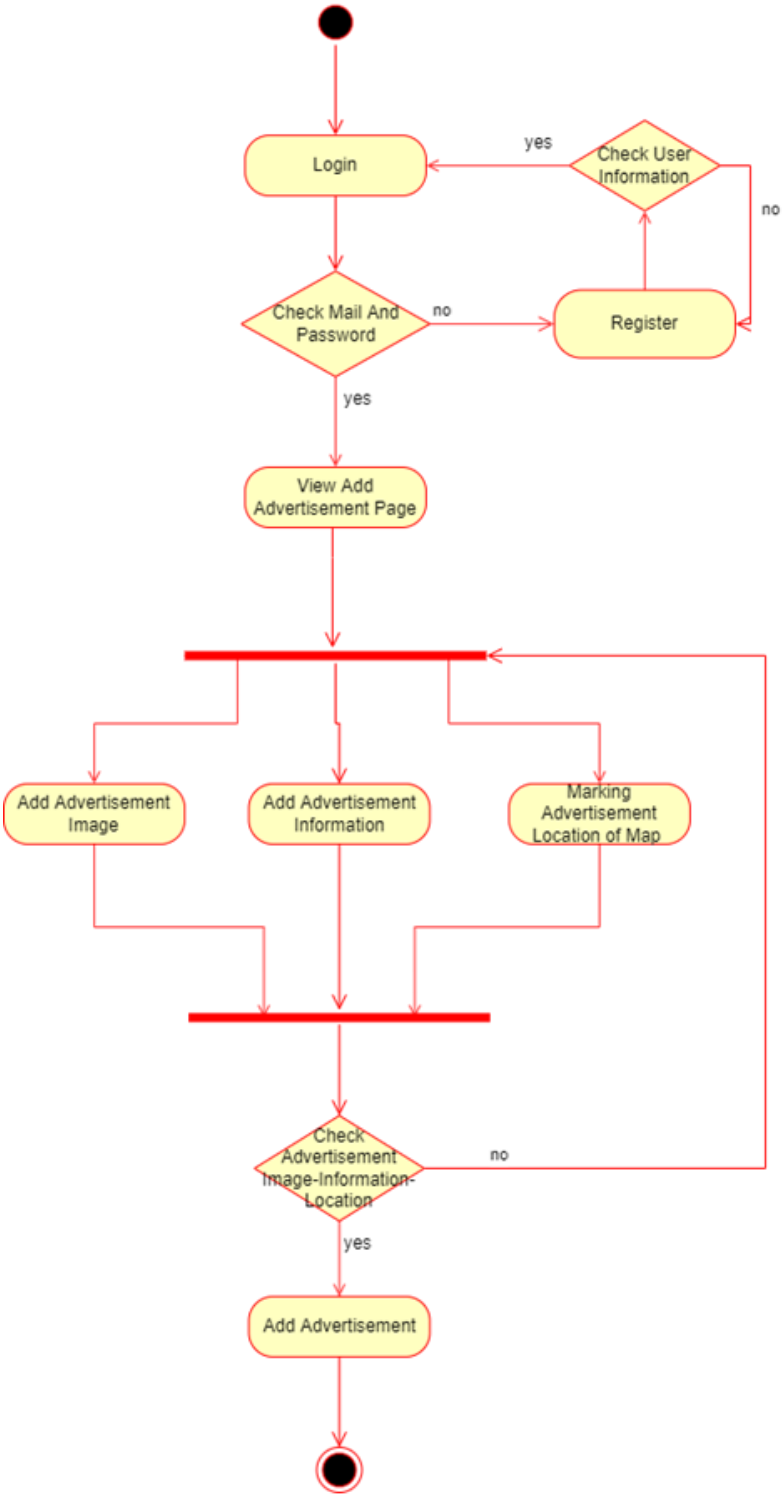


Figure 4.5.1.1 Activity Diagram of Add Advertisement in the Project

The sequence chart of adding advertisements in the project is as in Figure 4.5.1.1. The user comes to the login page and enters his e-mail and password to log in. Mail and password match is checked. If the mail and password do not match, the user cannot log in to the application and the user is asked to register in this drum. After the user registers, they are directed to the login screen. After entering the e-mail address and password on the login screen, the matching of these information is checked in the database. If there is no error as a result of the control, the user is directed to the main page of the application where the advertisements are listed. The user is directed to the add advertisement page with the add advertisement button on the main page of the application. On the add advertisement page, the user enters the advertisement information, adds the photos of the ad, and after marking the place of the advertisement on the map, all this information is checked. If no error is found as a result of the control, a new advertisement is added to the database.

4.5.2. Activity Diagram of AR Camera View

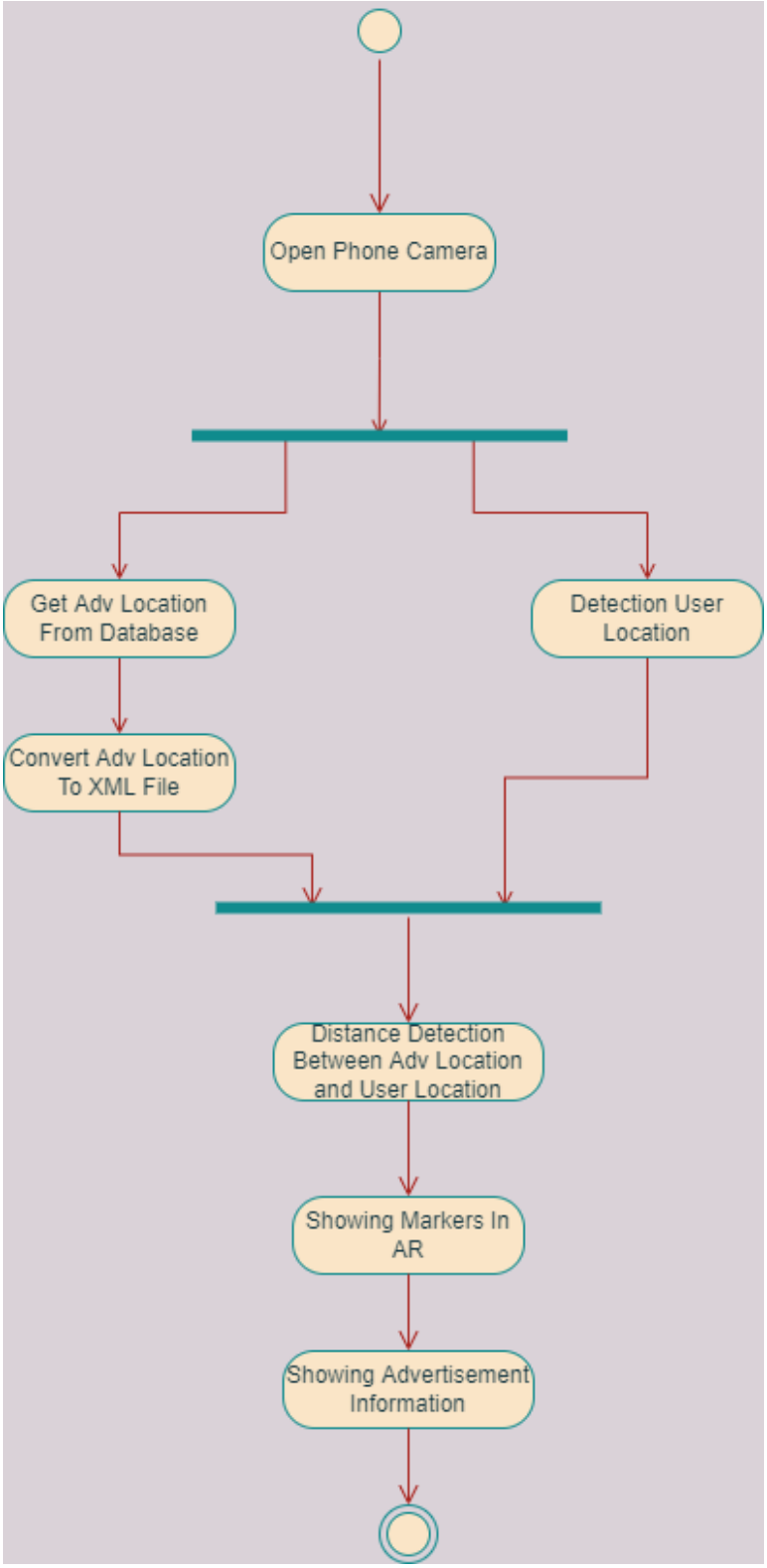


Figure 4.5.2.1 Activity Diagram of AR Camera View in the Project

The sequence chart of AR camera view in the project is as in Figure 4.5.2.1. User clicks AR camera button to open AR camera. The x - y coordinates, which are the location information of all advertisements in the database, are taken and the location information of these advertisements is converted into an XML format file. The distance between the location of the advertisements and the location of the user is measured, as the AR camera detects the user's location. As a result of this measurement, the advertisements at the camera angle are shown to the user with the markers on the AR camera according to the proximity to the user, and the size of these markers varies according to the distance.

4.6. Mockups

Some screen designs of the mobile application to be made have been made.

4.6.1. Log in Page Screen

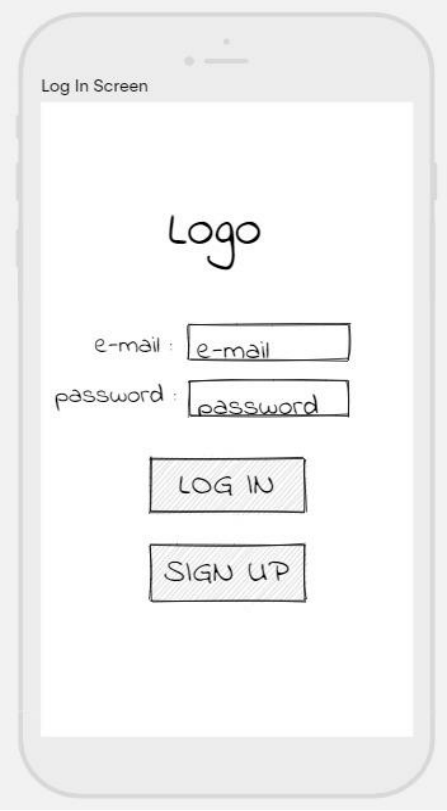


Figure 4.6.1.1 Log In Screen

When the user opens the application for the first time, the login screen in Figure 4.6.1.1 appears. It asks the user for an e-mail and password to log into the application. If an error occurs when clicking the Log In button after entering mail and password, or if the user is not registered with the application, he or she is directed to the application registration screen by clicking the Sign Up button.

4.6.2. Register Page Screen

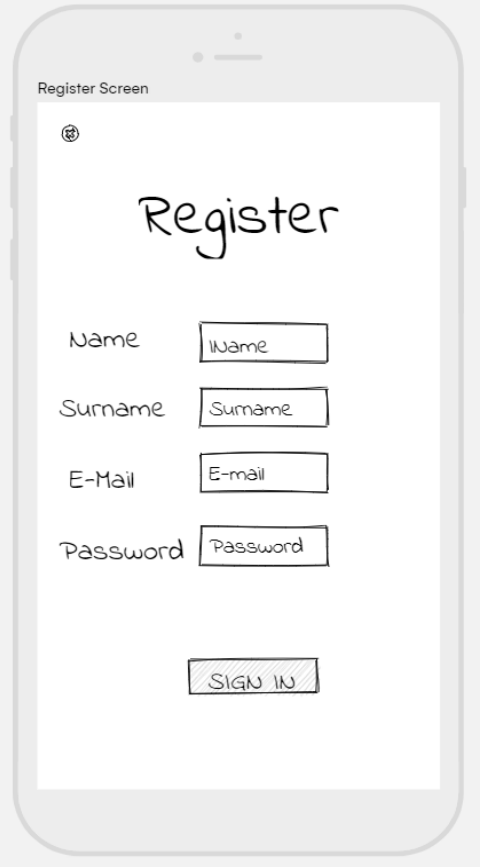


Figure 4.6.2.1 Register Screen

If the user is not registered to the application, he/she registers to the application by entering the requested information on the registration page in Figure 4.6.2.1.

4.6.3. Home Page Screen

Figure 4.6.3.1 Home Screen

After logging into the application, the user encounters the Figure 4.6.3.1 screen. On this page, advertisements are listed as clickable. Filtering and sorting can be done in the ads. Clicking the AR button will be directed to the AR page, clicking the add advertisement button will be directed to the add advertisement page, and clicking the my account button will be directed to the my account page.

4.6.4. Advertisement Detail View Page Screen

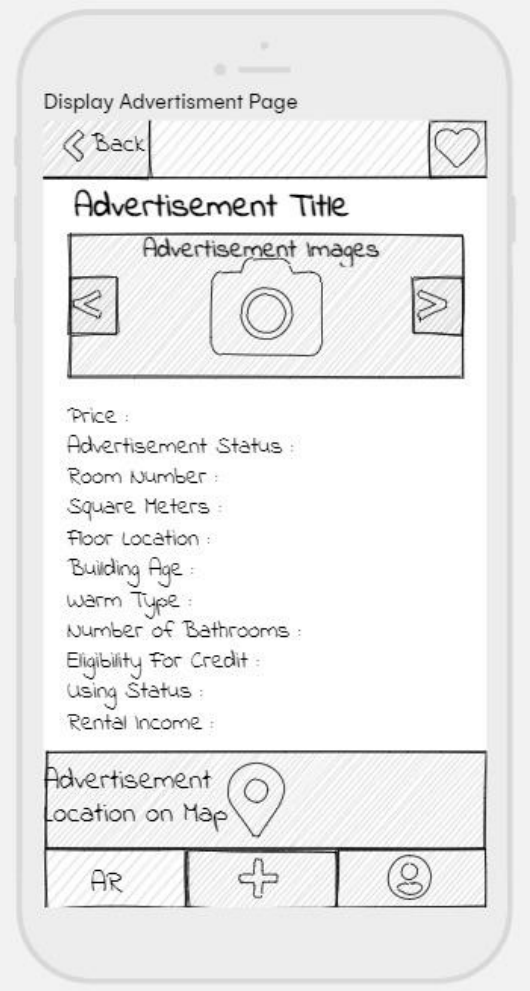


Figure 4.6.4.1 Advertisement Detail View Screen

If the user wants to see an advertisement in detail, when he/she clicks on the advertisement, he/she sees the details of the advertisement with the screen in Figure 4.6.4.1. He/She can change the photos of the advertisement with the arrow keys. It can also display the location of the advertisement on the map.

4.6.5. Adding Advertisement Page Screen

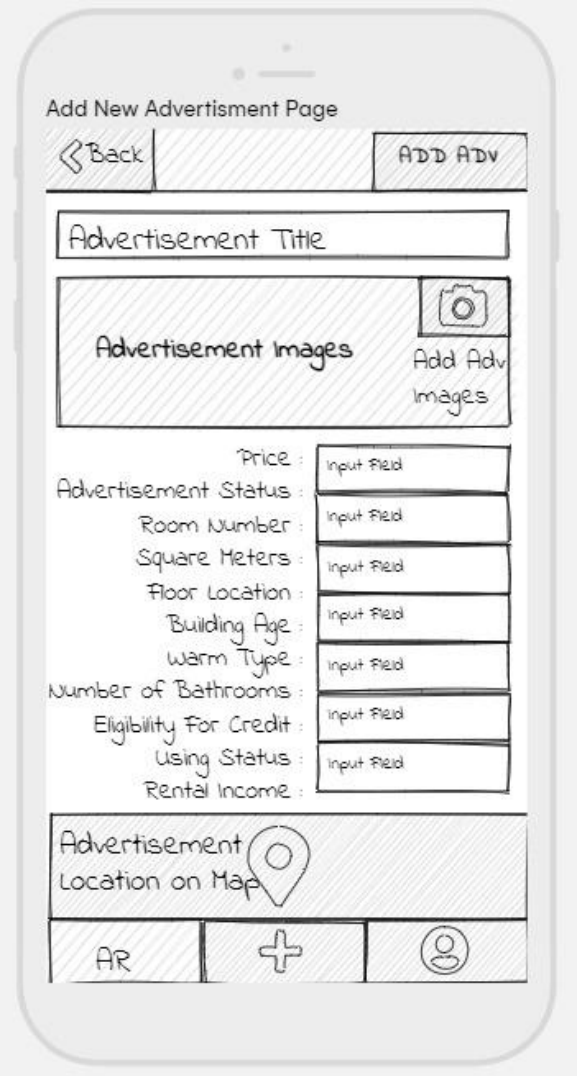


Figure 4.6.5.1 Adding Advertisement Screen

When the user wants to add a new advertisement, the Figure 4.6.5.1 screen appears. The user advertisements the photos of the ad, enters its properties in the necessary places, and after selecting the location of the advertisement on the map, adds the advertisement with the add advertisement button.

4.6.6. AR Page

Figure 4.6.6.1 AR Screen

With the AR screen in Figure 4.6.6.1, the user can easily reach the location of the advertisement and the details of the advertisement thanks to a marker at the location of the rental and sale advertisements at the angle of the camera.

4.6.7. My Account Page

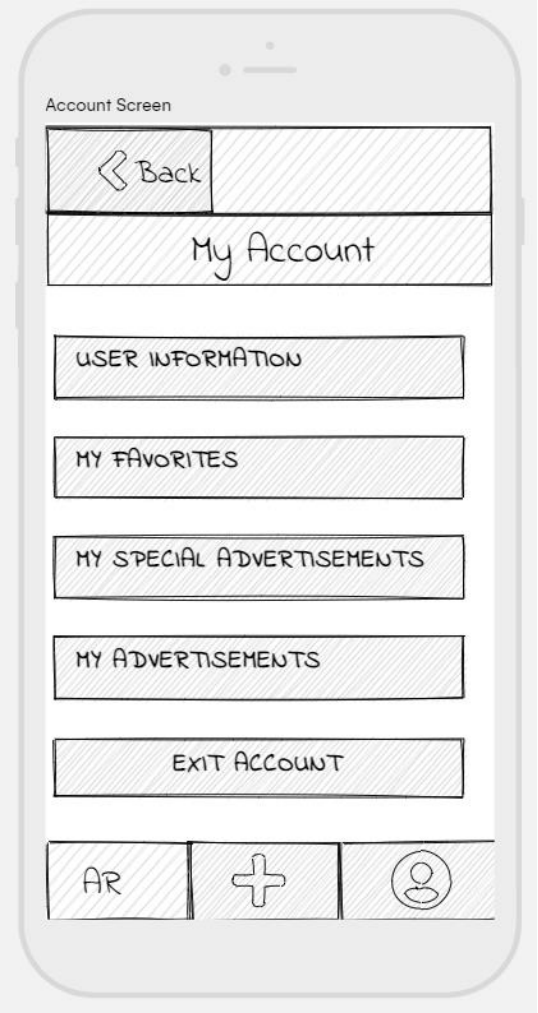


Figure 4.6.7.1 My Account Screen

On the screen in Figure 4.6.7.1, there are buttons that the user will use to update the user information, view the advertisements he/she has added to his/her favorites, view the advertisements that are special to him/her, view the advertisements he/she has added and exit his/her account.

4.6.8. My Information Page

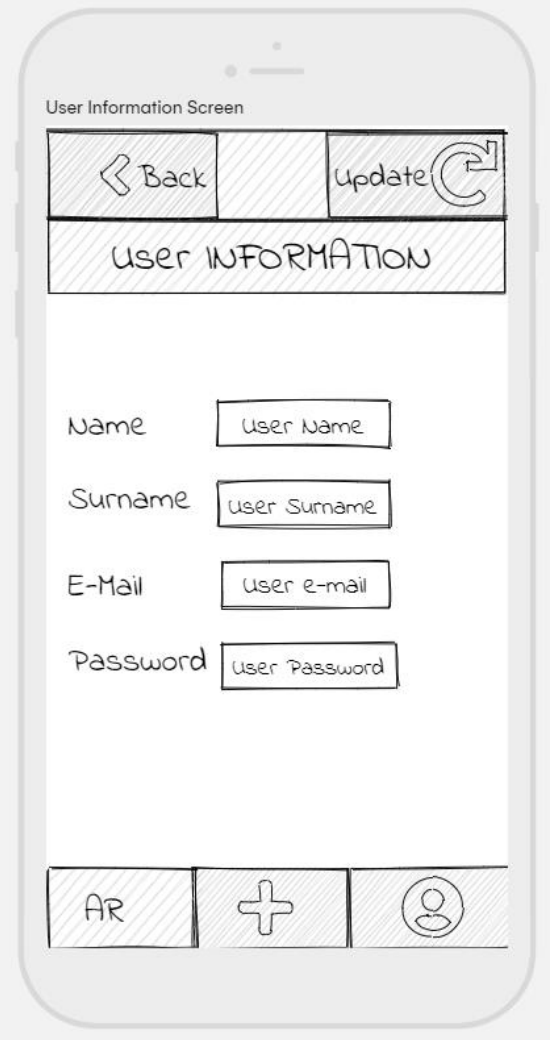


Figure 4.6.8.1 My Information Screen

The user can view and update their own information on the screen in Figure 4.6.8.1.

CHAPTER FIVE

IMPLEMENTATION

5.1. Android Application

5.1.1. Codes Within Classes

5.1.1.1. MainActivity Class

MainActivity Class is the first class that works in the application. Therefore, the "Database" class, which is the class in which the database and tables are created, is called here first. Thus, the database and tables are created when the application is first opened. "HomeFragment" is called because the first screen that the user sees when the application runs is the homepage.

At the top of the homepage are filtering and sorting options. Advertisements are listed in the "HomeFragment" according to the conditions of the options that increase or decrease according to the price in the order. In the filtering option, "FilterAdvFragment" is directed.

```
binding.navView.setOnItemSelectedListener(item -> {
    switch (item.getItemId()) {
        case R.id.navigation_ar:
            replaceFragment(new ARFragment());
            break;
        case R.id.navigation_addadvertisement:
            navViewToolbar.setVisibility(View.INVISIBLE);
            replaceFragment(new AddAdvFragment());
            break;
        case R.id.navigation_home:
            navViewToolbar.setVisibility(View.VISIBLE);
            MyAdvertisementFragment.clickMyAdvDetail=false;
            replaceFragment(new HomeFragment());
            break;
        case R.id.navigation_notifications:
            Intent intent=getIntent();
            User user=(User)intent.getSerializableExtra( name: "UserInformation");
            if(user==null){
                replaceFragment(new LoginFragment());
            }
            else{
                replaceFragment(new MyAccountFragment());
            }
            navViewToolbar.setVisibility(View.INVISIBLE);
            break;
    }
    return true;
});
```

Figure 5.1.1.1.1 MainActivity Navigation Bar

Page directions with the buttons on the Navigation Bar is done as in Figure 5.1.1.1.1. If the "Add Advertisement" button is clicked, it is directed to "AddAdvFragment", if the "Home" button is clicked to "HomeFragment", if the "User" button is clicked to "LoginFragment" or "MyAccountFragment", if the "AR" button is clicked to "ARFragment".

5.1.1.2. Database Class

```
@Override
public void onCreate(SQLiteDatabase db) {

    db.execSQL("CREATE TABLE IF NOT EXISTS Advertisements (AdvId INTEGER PRIMARY KEY AUTOINCREMENT,AdvTitle TEXT NOT NULL UNIQUE," +
            "AdvImage BLOB,Price INTEGER,AdvStatus TEXT,RoomNum TEXT,SquareMeter INTEGER,BuildingFloors INTEGER," +
            "FloorLoc INTEGER,BuildAge INTEGER,BuildType TEXT,ItemStatus TEXT,WarmType TEXT,NumOfBathrooms INTEGER," +
            "ElgCredit TEXT,UsingStatus TEXT,StateBuilding TEXT,RentalIncome INTEGER,Dues INTEGER,Swap TEXT,Front TEXT," +
            "FuelType TEXT,Date TEXT,Address TEXT,Cities TEXT,Town REAL,xCoordinate TEXT,yCoordinate REAL );");

    db.execSQL("CREATE TABLE IF NOT EXISTS  AdvertisementImage( ImageId  INTEGER PRIMARY KEY AUTOINCREMENT, " +
            "advImage  BLOB, AdvId INTEGER,  FOREIGN KEY (AdvId) REFERENCES Advertisements(AdvId));");

    db.execSQL("CREATE TABLE IF NOT EXISTS UserInformation (UserId INTEGER PRIMARY KEY AUTOINCREMENT,UserName TEXT NOT NULL," +
            " UserSurname TEXT NOT NULL, MailAddress TEXT NOT NULL,Password TEXT NOT NULL);");

    db.execSQL("CREATE TABLE IF NOT EXISTS Favorite (UserId INTEGER,AdvId INTEGER,FavoriteStatus INTEGER,FOREIGN KEY (UserId) " +
            "REFERENCES UserInformation(UserId), FOREIGN KEY (AdvId) REFERENCES Advertisements(AdvId));");

    db.execSQL("CREATE TABLE IF NOT EXISTS UserPastAdvertisement (UserId INTEGER,AdvId INTEGER,FOREIGN KEY (UserId) " +
            "REFERENCES UserInformation(UserId), FOREIGN KEY (AdvId) REFERENCES Advertisements(AdvId));");

    db.execSQL("CREATE TABLE IF NOT EXISTS Cities (CityId INTEGER PRIMARY KEY AUTOINCREMENT,CityName TEXT NOT NULL UNIQUE);");

    db.execSQL("CREATE TABLE IF NOT EXISTS District(DistrictId INTEGER PRIMARY KEY AUTOINCREMENT,DistrictName TEXT NOT NULL," +
            "CityId INTEGER NOT NULL);");

    db.execSQL("CREATE TABLE IF NOT EXISTS  UserAdvertisement( UserId INTEGER,AdvId INTEGER, FOREIGN KEY (UserId) " +
            "REFERENCES UserInformation(UserId), FOREIGN KEY (AdvId) REFERENCES Advertisements(AdvId));");
}
```

Figure 5.1.1.2.1 Create Tables in Database

SQLite database is used. It is the class in which "RealEstate" database and "Advertisements", "AdvertisementImage", "UserInformation", "Favorite", "UserPastAdvertisement", "Cities", "District", "UserAdvertisement" tables are created as in Figure 5.1.1.2.1.

```
public User loginUser(String email, String password){
    User user=null;
    try{
        SQLiteDatabase sqLiteDatabase=getReadableDatabase();
      Cursor cursor=sqLiteDatabase.rawQuery( sql: "SELECT * FROM UserInformation " +
              "WHERE MailAddress = ? AND Password = ?",new String[]{email,password});
      if(cursor.moveToFirst()){
          user=new User();
          user.setUserId(cursor.getInt( i: 0));
          user.setUserName(cursor.getString( i: 1));
          user.setUserSurname(cursor.getString( i: 2));
          user.setMailAddress(cursor.getString( i: 3));
          user.setPassword(cursor.getString( i: 4));
      }
    }catch (Exception e){
        user=null;
    }
    return user;
}
```

Figure 5.1.1.2.2 User Create in Database Class

As seen in Figure 5.1.1.2.2, a user object is created by fetching the user information from the database according to the mail and password of the logged in user.

```
public int updateMyAdv(int advId, String advTitle, byte[]  advImage, int price, String advStatus, String roomNum,
                    int squareMeter, int buildingFloors, int floorLoc,int buildAge,String buildType,String itemStatus,
                    String warmTpe,int numOfBathrooms,String elgCredit, String usingStatus,String stateBuilding,
                    int rentalIncome, int dues, String swap,String front,String fuelType,String date,
                    String address, String city,String town, double xCoordinate, double yCoordinate){
    try{
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put("AdvTitle",advTitle);                values.put("AdvImage",advImage);
        values.put("Price",price);                      values.put("AdvStatus",advStatus);
        values.put("RoomNum",roomNum);                  values.put("SquareMeter",squareMeter);
        values.put("BuildingFloors",buildingFloors);    values.put("FloorLoc",floorLoc);
        values.put("BuildAge",buildAge);                values.put("BuildType",buildType);
        values.put("ItemStatus",itemStatus);            values.put("WarmType",warmTpe);
        values.put("NumOfBathrooms",numOfBathrooms);    values.put("ElgCredit",elgCredit);
        values.put("UsingStatus",usingStatus);          values.put("StateBuilding",stateBuilding);
        values.put("RentalIncome",rentalIncome);        values.put("Dues",dues);
        values.put("Swap",swap);                        values.put("Front",front);
        values.put("FuelType",fuelType);                values.put("Date",date);
        values.put("Address",address);                  values.put("Cities",city);
        values.put("Town",town);                        values.put("xCoordinate",xCoordinate);
        values.put("yCoordinate",yCoordinate);
        db.update( table: "Advertisements", values,  whereClause: "AdvId=?", new String[]{String.valueOf(advId)});
        db.close();
    }catch (Exception e){
        return 0;
    }
    return 1;
}
```

Figure 5.1.1.2.3 Update Advertisement in Database Class

When the user's previously processed post is updated, it is sent as a parameter to the "updateMyAdv" function. Parameters are placed in the columns in the "Advertisements" table in the database as in Figure 5.1.1.2.3. If the update was successful, the value "1" is returned.

5.1.1.3. Home Page

In HomeFragment, the advertisements in the database are displayed. The way these advertisements are displayed is in the feature of clickable card. When these cards are clicked, the details of the advertisement are displayed.

Thanks to the object created with "RecyclerView", all advertisements in the database are dynamically displayed on the homepage. Thanks to the object created with the "AdvertisementAdapter", the advertisements are kept on the card and the clickability feature is called.

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    MyFavoritesFragment.clickMyFavDetail=false;
    MyAccountFragment.clickMyAdv=false;
    View view=inflater.inflate(R.layout.fragment_home,container, attachToRoot: false);
    recyclerView=(RecyclerView) view.findViewById(R.id.recview);
    recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));
    advAdapter=new AdvertisementAdapter(getData(getContext()),getContext());
   recyclerView.setAdapter(advAdapter);
    advAdapter.setOnItemClickListener(adv ->  {
        advDetail=new AdvDetail(adv.getAdvId(),adv.getAdvTitle(),adv.getAdvImage(),adv.getPrice(),adv.getAdvStatus(),
                adv.getRoomNum(),adv.getSquareMeters(),adv.getBuildingFloors(),adv.getFloorLoc(),
                adv.getBuildAge(),adv.getBuildType(),adv.getItemStatus(),adv.getWarmType(),adv.getNumOfBathr(),
                adv.getElgForCredit(),adv.getUsingStatus(),adv.getStateBuilding(),adv.getRentalIncome(),
                adv.getDues(),adv.getSwap(),adv.getFront(),adv.getFuelType(),adv.getDate(),adv.getAddress(),adv.getCity(),adv.getTown());

        MainActivity.navViewToolbar.setVisibility(View.INVISIBLE);
        replaceFragment(new AdvDetailFragment());
    });
    return view;
}
```

Figure 5.1.1.3.1 HomeFragment onCreateView Function

As can be seen in Figure 5.1.1.3.1, the advertisement cards created with the "advAdapter" object are displayed dynamically from the database by using the “recyclerView” object. When the advertisement is clicked, the details of the clicked advertisement are created by means of the "get"

functions, and the "advDetail" object is created and directed to the "AdvDetailFragment" class to display the advertisement detail.

5.1.1.4. User Class

```
import java.io.Serializable;

public class User implements Serializable {
    private int UserId;
    private String UserName;
    private String UserSurname;
    private String MailAddress;
    private String Password;

    public int getUserId() { return UserId; }

    public void setUserId(int userId) { UserId = userId; }

    public String getUserName() { return UserName; }

    public void setUserName(String userName) { UserName = userName; }

    public String getUserSurname() { return UserSurname; }

    public void setUserSurname(String userSurname) { UserSurname = userSurname; }

    public String getMailAddress() { return MailAddress; }

    public void setMailAddress(String mailAddress) { MailAddress = mailAddress; }

    public String getPassword() { return Password; }

    public void setPassword(String password) { Password = password; }
}
```

Figure 5.1.1.4.1 User Class

The properties defined in the User class are for the information requested when registering the user. The name, surname, e-mail address and password of the user in Figure 5.1.1.4.1 are taken from the user. UserId, on the other hand, keeps the user unique in the database with automatic incrementing.

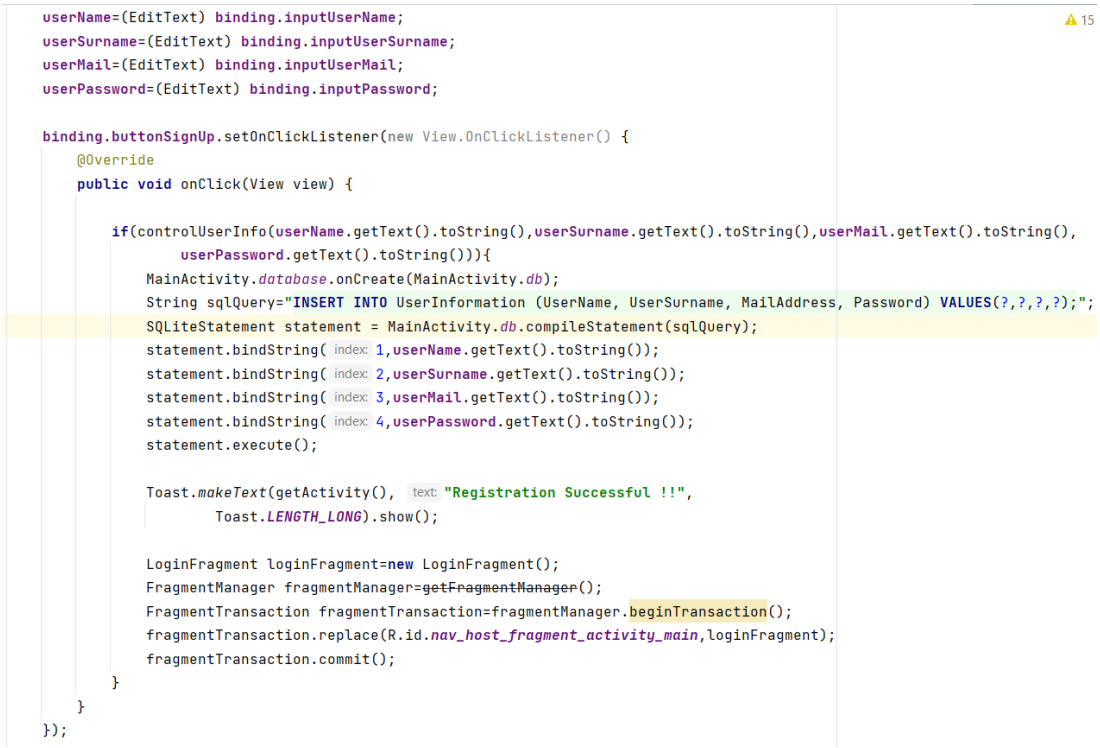
5.1.1.5. SignUp Page

In order for the user to register with the application, name, surname, e-mail and password information are obtained from the user.

```
public boolean controlUserInfo(String userName, String userSurname, String userMail, String userPassword){
    Database database=new Database(getContext());

    if(TextUtils.isEmpty(userName)||TextUtils.isEmpty(userSurname)||TextUtils.isEmpty(userMail)||TextUtils.isEmpty(userPassword)){
        Toast.makeText(getActivity(), text: "Please fill in all the blanks !!", Toast.LENGTH_SHORT).show();
        return false;
    }
    else if(!database.checkEmailExist(userMail)){
        Toast.makeText(getActivity(), text: "You cannot use this e-mail address !!", Toast.LENGTH_SHORT).show();
        return false;
    }
    else if(!userMail.matches(regex)){
        Toast.makeText(getActivity(), text: "E-mail address format is not match !!", Toast.LENGTH_SHORT).show();
        return false;
    }
    else if(userPassword.length()<8){
        Toast.makeText(getActivity(), text: "Password is too short !!", Toast.LENGTH_SHORT).show();
        return false;
    }

    return true;
}
```

Figure 5.1.1.5.1 User Information Control

In Figure 5.1.1.5.1 the uniqueness of the e-mail address, the suitability of the e-mail format, the length of the password and the completeness of the information are checked.



```
userName=(EditText) binding.inputUserName;
userSurname=(EditText) binding.inputUserSurname;
userMail=(EditText) binding.inputUserMail;
userPassword=(EditText) binding.inputPassword;

binding.buttonSignUp.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        if(controlUserInfo(userName.getText().toString(),userSurname.getText().toString(),userMail.getText().toString(),
                userPassword.getText().toString())){
            MainActivity.database.onCreate(MainActivity.db);
            String sqlQuery="INSERT INTO UserInformation (UserName, UserSurname, MailAddress, Password) VALUES(?,?,?,?);";
            SQLiteStatement statement = MainActivity.db.compileStatement(sqlQuery);
            statement.bindString( index: 1,userName.getText().toString());
            statement.bindString( index: 2,userSurname.getText().toString());
            statement.bindString( index: 3,userMail.getText().toString());
            statement.bindString( index: 4,userPassword.getText().toString());
            statement.execute();

            Toast.makeText(getActivity(),  text: "Registration Successful !!",
                    Toast.LENGTH_LONG).show();

            LoginFragment loginFragment=new LoginFragment();
            FragmentManager fragmentManager=getFragmentManager();
            FragmentTransaction fragmentTransaction=fragmentManager.beginTransaction();
            fragmentTransaction.replace(R.id.nav_host_fragment_activity_main,loginFragment);
            fragmentTransaction.commit();
        }
    }
});
```

Figure 5.1.1.5.2 Insert User Information in Database

If the control is successful, the user is added to the database with this information, and the registration process is completed and directed to the "Login Fragment" in Figure 5.1.1.5.2.

5.1.1.6. Login Page

While the user is logging into the application, the e-mail address and password are obtained and the information received in the database is checked to match a user. If the user's mail and password do not match the data in the database, an error message is displayed to the user.

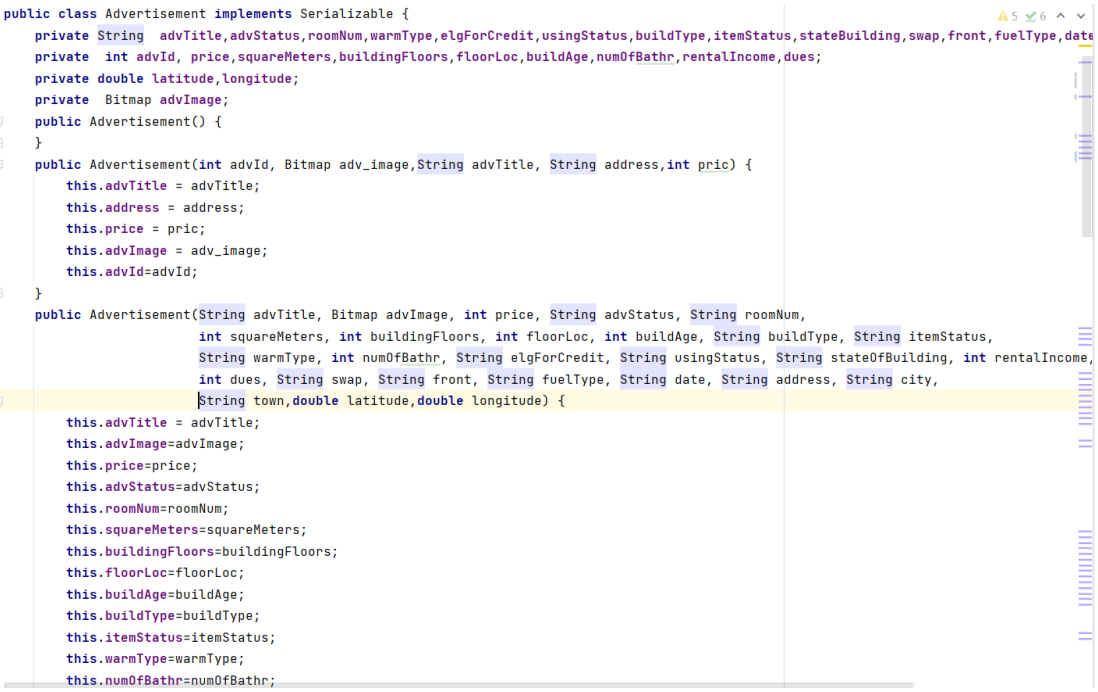
```
inputUserMail=(EditText) binding.inputUserMail;
inputPassword=(EditText) binding.inputPassword;
buttonLogin=(Button) binding.buttonLogin;

binding.buttonLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        if(TextUtils.isEmpty(inputUserMail.getText().toString()) || TextUtils.isEmpty(inputPassword.getText().toString())){

            Toast.makeText(getActivity(), text: "Please fill in all the blanks !!",
                    Toast.LENGTH_SHORT).show();
        }
        else if(!TextUtils.isEmpty(inputUserMail.getText().toString()) && !TextUtils.isEmpty(inputPassword.getText().toString())){
            Database database=new Database(getContext());
            String userMail=inputUserMail.getText().toString();
            String userPassword=inputPassword.getText().toString();
            User user= database.loginUser(userMail,userPassword);

            if(user!=null){
                Intent intent=new Intent(getActivity().getBaseContext(),MainActivity.class);
                intent.putExtra( name: "UserInformation",user);
                getActivity().startActivity(intent);
            }
            else{
                Toast.makeText(getActivity(), text: "E-mail or password is wrong !!",
                        Toast.LENGTH_SHORT).show();
            }
        }
    }
});
```

Figure 5.1.1.6.1 User Information Control for Login

If the user information matches as in Figure 5.1.1.6.1, a user named "UserInformation" is created. Thus, when the user's id is needed, the user information is accessed with this "UserInformation".

5.1.1.7. Advertisement Class

The properties defined in the Advertisement class are for creating advertisements. The object created with this class is used in transactions related to advertisements.



```
public class Advertisement implements Serializable {
    private String  advTitle,advStatus,roomNum,warmType,elgForCredit,usingStatus,buildType,itemStatus,stateBuilding,swap,front,fuelType,date
    private  int advId, price,squareMeters,buildingFloors,floorLoc,buildAge,numOfBathr,rentalIncome,dues;
    private double latitude,longitude;
    private  Bitmap advImage;
    public Advertisement() {
    }
    public Advertisement(int advId, Bitmap adv_image,String advTitle, String address,int pric) {
        this.advTitle = advTitle;
        this.address = address;
        this.price = pric;
        this.advImage = adv_image;
        this.advId=advId;
    }
    public Advertisement(String advTitle, Bitmap advImage, int price, String advStatus, String roomNum,
                         int squareMeters, int buildingFloors, int floorLoc, int buildAge, String buildType, String itemStatus,
                         String warmType, int numOfBathr, String elgForCredit, String usingStatus, String stateOfBuilding, int rentalIncome,
                         int dues, String swap, String front, String fuelType, String date, String address, String city,
                         String town,double latitude,double longitude) {
        this.advTitle = advTitle;
        this.advImage=advImage;
        this.price=price;
        this.advStatus=advStatus;
        this.roomNum=roomNum;
        this.squareMeters=squareMeters;
        this.buildingFloors=buildingFloors;
        this.floorLoc=floorLoc;
        this.buildAge=buildAge;
        this.buildType=buildType;
        this.itemStatus=itemStatus;
        this.warmType=warmType;
        this.numOfBathr=numOfBathr;
```

Figure 5.1.1.7.1 Advertisement Class

As in Figure 5.1.1.7.1, advertisement title, advertisement status, number of rooms, heating type, credit eligibility, usage status, building type, property status, state building, clearing status, facade, fuel type, date of addition, address, city, town, price, square meters, number of floors of the building, the floor where it is located, age of the building, number of bathrooms, rental income, dues, location information and pictures.

5.1.1.8. AdvertisementAdapter Class

In the “AdvertisementAdepter” class, there are methods that provide the ability to display the advertisement cards created while listing the advertisements and to click on these cards. The picture, advertisement title, address and price are displayed on the advertisement cards.

```
class myviewholder extends RecyclerView.ViewHolder{
    ImageView img;
    TextView titlee, addresss, pricee;
    LinearLayout clickDetail;
    public myviewholder (@NonNull View itemView){
        super(itemView);
        img=itemView.findViewById(R.id.image);
        titlee=itemView.findViewById(R.id.title);
        addresss=itemView.findViewById(R.id.address);
        pricee=itemView.findViewById(R.id.price);
        itemView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                int position=getAdapterPosition();

                if(listener!=null && position!=RecyclerView.NO_POSITION ){
                    listener.onItemClick(adv.get(position));
                }
            }
        });
    }
}
```

Figure 5.1.1.8.1 AdvertisementAdapter Class MyViewHolder Method

As seen in Figure 5.1.1.8.1, notice cards are assigned to the object created with the "View" class. With the "ViewHolder" feature, the parameters defined in the announcement card are given a clickable feature with the object created with the "View" class.

5.1.1.9. Add Advertisement Page

"AddAdvFragment" is used to add advertisements to the database. From user name title, advertisement status, number of rooms, heating type, credit eligibility, usage status, building type, property status, state building, clearing status, facade, fuel type, date of addition, address, city, town, price, square meters, number of floors of the building, the floor where it is located, age of the building, number of bathrooms, rental income, dues, location information and pictures are requested.

```
ArrayAdapter<CharSequence>adapterAdvStatus=ArrayAdapter.createFromResource(getActivity().getBaseContext(),
        R.array.Adv_Status, android.R.layout.simple_spinner_item);
adapterAdvStatus.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinnerAdvStatus.setAdapter(adapterAdvStatus);
spinnerAdvStatus.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) {
        advStatus = adapterView.getItemAtPosition(i).toString();
    }
    @Override
    public void onNothingSelected(AdapterView<?> adapterView) {

    }
});

ArrayAdapter<CharSequence>adapterBuildType=ArrayAdapter.createFromResource(getActivity().getBaseContext(),
        R.array.BuildingType, android.R.layout.simple_spinner_item);
adapterBuildType.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

spinnerBuildType.setAdapter(adapterBuildType);
spinnerBuildType.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) {
        buildType=adapterView.getItemAtPosition(i).toString();
    }
    @Override
    public void onNothingSelected(AdapterView<?> adapterView) {

    }
});
```

Figure 5.1.1.9.1 Add Advertisement Dropdown Input

As in Figure 5.1.1.9.1 "Advertisement Status", "Room Number", "Build Type", "Item Status", "Warm Type", "Eligibility For Credit", "Using Status", "State Of Building", "Swap ", "Front", "Fuel Type", "City", "Town" inputs, the user chooses from the dropdown options. Other information is requested from the user as input. The location of the advertisement is marked on the map for the location information on the map, and all the values received from the user, together with the x-y coordinates of the marked place, are saved in the database.

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    binding = FragmentAddAdvBinding.inflate(inflater, container,  attachToParent: false);
    Intent intent=getActivity().getIntent();
    User user=(User)intent.getSerializableExtra( name: "UserInformation");
    if(user==null){
        FragmentManager fragmentManager=getActivity().getSupportFragmentManager();
        FragmentTransaction fragmentTransaction=fragmentManager.beginTransaction();
        fragmentTransaction.replace(R.id.nav_host_fragment_activity_main,new LoginFragment());
        fragmentTransaction.commit();
    }
    else{
        userId=user.getUserId();
        SupportMapFragment supportMapFragment=(SupportMapFragment) getChildFragmentManager().findFragmentById(R.id.google_map);
        supportMapFragment.getMapAsync( callback: this);
        init();
        binding.addAdvImage.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if(ContextCompat.checkSelfPermission(getActivity().getBaseContext(),
                        Manifest.permission.READ_EXTERNAL_STORAGE)!= PackageManager.PERMISSION_GRANTED){
                    ActivityCompat.requestPermissions(getActivity(),new String[]{Manifest.permission.READ_EXTERNAL_STORAGE},
                            imgNoPermissionCod);
                }
                else{
                    Intent intent = new Intent();
                    intent.setType("image/*");
                    intent.putExtra(Intent.EXTRA_ALLOW_MULTIPLE,  value: true);
                    intent.setAction(Intent.ACTION_GET_CONTENT);
                    startActivityForResult(Intent.createChooser(intent,  title: "Select images"), imgPermissionCod);
                }
```

Figure 5.1.1.9.2 Select Multi Image in Gallery

As seen in Figure 5.1.1.9.2, this method is used to select more than one picture from the gallery on the phone. When you click on the photo icon, you will be directed to the gallery if access to the gallery is allowed.

```
public boolean advInputControl(){
    Boolean checkEmpty =true;
    if(MainActivity.database.checkAdvTitleExist(advTitle)==false) {
        checkEmpty=false;
        Toast.makeText(getActivity(),  text: "You cannot use this adv title", Toast.LENGTH_SHORT).show();
    }
    if(advStatus.equals("Select Adv Status") || buildType.equals("Select Build Type") || itemStatus.equals("Select Item Status") ||
            elgForCredit.equals("Select Elg Credit") || usingStatus.equals("Select Using Status") || stateBuilding.equals("Select State Building")||
            swap.equals("Select Swap")|| front.equals("Select Front") || fuelType.equals("Select Fuel Type") ||
            roomNum.equals("Select Room Number") || warmType.equals("Select Warm Type") || city.equals("Select City") ){
        checkEmpty =false;
        Toast.makeText(getActivity(), text: "Please fill in the blanks",Toast.LENGTH_SHORT).show();
    }else if(TextUtils.isEmpty(advTitle)  || TextUtils.isEmpty(address)|| TextUtils.isEmpty(editTxtPrice.getText().toString())||
            TextUtils.isEmpty(editTxtSquareMt.getText().toString())||TextUtils.isEmpty(editTxtBuildingFloors.getText().toString())||
            TextUtils.isEmpty(editTxtFloorLoc.getText().toString()) ||TextUtils.isEmpty(editTxtBuildAge.getText().toString())||
            TextUtils.isEmpty(editTxtNumofBath.getText().toString())||TextUtils.isEmpty(editTxtRentalIncome.getText().toString())||
            TextUtils.isEmpty(editTxtDues.getText().toString())){
        checkEmpty=false;
        Toast.makeText(getActivity(), text: "Please fill in the blanks",Toast.LENGTH_SHORT).show();
    }else if(!TextUtils.isDigitsOnly(editTxtPrice.getText()) || !TextUtils.isDigitsOnly(editTxtSquareMt.getText()) ||
            !TextUtils.isDigitsOnly(editTxtBuildingFloors.getText()) || !TextUtils.isDigitsOnly(editTxtFloorLoc.getText()) ||
            !TextUtils.isDigitsOnly(editTxtBuildAge.getText()) || !TextUtils.isDigitsOnly(editTxtRentalIncome.getText())||
            !TextUtils.isDigitsOnly(editTxtDues.getText()) || !TextUtils.isDigitsOnly(editTxtNumofBath.getText())) {
        checkEmpty =false;
        Toast.makeText(getActivity(), text: "Please Incorrect Inputs",Toast.LENGTH_SHORT).show();
    }
    return checkEmpty;
}
```

Figure 5.1.1.9.3 Add Advertisement Input Control

The function in which the inputs entered while adding an advertisement are empty, selections are made from the dropdown, and the types of inputs entered are checked are as in Figure 5.1.1.9.3. If the result of this function returns false, an error message is displayed to the user.

```
public void saveAdv(View view) {
    try {
        advTitle=editTxtTitle.getText().toString();
        date=getTodayDate();
        address=editTxtAddress.getText().toString();
        price= Integer.parseInt( editTxtPrice.getText().toString());
        squareMeters=Integer.parseInt(editTxtSquareMt.getText().toString());
        buildingFloors=Integer.parseInt(editTxtBuildingFloors.getText().toString());
        floorLoc=Integer.parseInt(editTxtFloorLoc.getText().toString());
        buildAge=Integer.parseInt(editTxtBuildAge.getText().toString());
        numOfBathr=Integer.parseInt(editTxtNumofBath.getText().toString());
        rentalIncome=Integer.parseInt(editTxtRentalIncome.getText().toString());
        dues=Integer.parseInt(editTxtDues.getText().toString());
    }catch (Exception e ){
        System.out.println("ERROR");
    }
    ByteArrayOutputStream outputStream =new ByteArrayOutputStream();
    smallestedImg=imageSmall(firstSelectedImage);
    smallestedImg.compress(Bitmap.CompressFormat.PNG, quality: 75,outputStream);
    byte[] Image=outputStream.toByteArray();
    if(advInputControl()==true){
        try {
            MainActivity.database.onCreate(MainActivity.db);
            String sqlQuery="INSERT INTO Advertisements (AdvTitle,AdvImage,Price,AdvStatus,RoomNum,SquareMeter,BuildingFloors,FloorLoc" +
                    ",BuildAge,BuildType,ItemStatus,WarmType,NumOfBathrooms,ElgCredit,UsingStatus,StateBuilding,RentalIncome,Dues,Swap,Front," +
                    "FuelType,Date,Address,Cities,Town,xCoordinate,yCoordinate)VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
            SQLiteStatement statement = MainActivity.db.compileStatement(sqlQuery);
            statement.bindString( index: 1,advTitle);
            statement.bindBlob( index: 2,Image);
            statement.bindString( index: 3, String.valueOf(price));
            statement.bindString( index: 4,advStatus);
            statement.bindString( index: 5,roomNum);
```

Figure 5.1.1.9.4 Insert Advertisement in Database

As in Figure 5.1.1.9.4, the values received from the user as input are converted to string type and assigned to variables. The images selected from the gallery are transferred to the byte type array to save the bitmap type database. If the input control is provided correctly, all the information received from the user is added to the "Advertisements" table in the database.

```
final ServiceManage serviceManage=new ServiceManage();
Thread thread=new Thread(new Runnable() {
    @Override
    public void run() {
        serviceManage.addAdvertisement(advTitle,price,advStatus,roomNum,squareMeters,buildingFloors,floorLoc,buildAge,
                buildType,itemStatus,warmType,numOfBathr,elgForCredit,usingStatus,stateBuilding,rentalIncome,dues,
                swap,front,fuelType,endDate,address,city,town,latitude,longitude);
    }
});
thread.start();
```

Figure 5.1.1.9.5 Insert Advertisement in Server Database

In order to use the advertisement information in the application on the Unity side, advertisement information is added to the database on the server common to both applications in Figure 5.1.1.9.5.

```
Object_Clear();
Toast.makeText(getActivity(), text: "Adding Advertisement Successful ",Toast.LENGTH_SHORT).show();
add_Adv=true;
int lastInsertedAdvId = 0;
Cursor c=MainActivity.db.rawQuery( sql: "select last_insert_rowid()", selectionArgs: null);
if (c!= null && c.moveToFirst()) {
    lastInsertedAdvId = c.getInt( i: 0);
}
c.close();
advId=lastInsertedAdvId;
for (int i=0;i<imageCount;i++) {
    ByteArrayOutputStream outputStreamMulti =new ByteArrayOutputStream();
    smallestedImg=imageSmall(imagesSelect.get(i));
    smallestedImg.compress(Bitmap.CompressFormat.PNG, quality: 75,outputStreamMulti);
    byte[] imageMulti=outputStreamMulti.toByteArray();
    String sqlQueryImage = "INSERT INTO AdvertisementImage (AdvImage,AdvId)VALUES(?,?)";
    SQLiteStatement statementImg = MainActivity.db.compileStatement(sqlQueryImage);
    statementImg.bindBlob( index: 1, imageMulti);
    statementImg.bindString( index: 2, String.valueOf(lastInsertedAdvId));
    statementImg.execute();
}
String sqlQueryUserAdv="INSERT INTO UserAdvertisement (UserId,AdvId)VALUES(?,?);";
SQLiteStatement statementUserAdv = MainActivity.db.compileStatement(sqlQueryUserAdv);
statementUserAdv.bindString( index: 1,String.valueOf(userId));
statementUserAdv.bindString( index: 2,String.valueOf(advId));
statementUserAdv.execute();
```

Figure 5.1.1.9.6 AdvImage and UserAdv Insert in Tables

The id information of the last added advertisement is obtained. The pictures selected from the gallery are added to the AdvertisementImage table with the byte type advertisement id information with the for loop. The user who added the advertisement and the id of the advertisement are added to the UserAdvertisement table as in Figure 5.1.1.9.6.

If the advertisement save operation is successful, the details of the newly added advertisement will be shown by being directed to the "AdvDetailFragment".

```
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int [] grantResults){

    if(requestCode==imgNoPermissionCod)
    {
        if(grantResults.length> 0 && grantResults[0]==PackageManager.PERMISSION_GRANTED){
            Intent imageGet = new Intent(Intent.ACTION_PICK,MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
            startActivityForResult(imageGet,imgPermissionCod);
        }
    }
    super.onRequestPermissionsResult(requestCode,permissions,grantResults);
}
```

Figure 5.1.1.9.7 Image Permission Media Store

As seen in Figure 5.1.1.9.7, when adding an advertisement, permission control to select an image from the media store.

```
if(requestCode==imgPermissionCod){
    if(resultCode==-1 && data !=null){
        try {
            Uri imgUrl=data.getData();
            if (Build.VERSION.SDK_INT >= 28) {

            } else {
                if(data.getClipData()!=null){
                    count = data.getClipData().getItemCount();
                    for (int i = 0; i < count; i++) {
                        Uri imageUri = data.getClipData().getItemAt(i).getUri();
                        mArrayUri.add(imageUri);
                    }
                    for (int j = 0; j < mArrayUri.size(); j++) {
                        imagesSelect.add(MediaStore.Images.Media.getBitmap(getActivity().getContentResolver(), mArrayUri.get(j)));
                        firstSelectedImage = imagesSelect.get(0);
                        selectedImg = imagesSelect.get(j);
                        imageAdv.setImageBitmap(selectedImg);
                        imageCount++;
                    }
                }
                else if(data.getClipData()==null) {
                    imageCount=1;
                    imagesSelect.add(MediaStore.Images.Media.getBitmap(getActivity().getContentResolver(),imgUrl));
                    selectedImg= imagesSelect.get(0);
                    firstSelectedImage = imagesSelect.get(0);
                    imageAdv.setImageBitmap(selectedImg);
                }
            }
            btnSubmitAdv.setEnabled(true);
        } catch (IOException e) {
            e.printStackTrace();
```

Figure 5.1.1.9.8 Image Select SetImageBitMap

In Figure 5.1.1.9.8, the content of the "data" of type Intent that comes as a parameter is assigned to the imgUrl variable of type "Uri". If more than one photo is selected in the gallery, the number of selection is kept as "count". The "Uri" type variable of each image selected with the for loop is added to the "mArrayUri" array. With another for loop, each item in "mArrayUri" is added to the "imagesSelect" array of "BitMap" array type. It

is assigned to the "selectedImg" variable of type Bitmap for further processing. This variable is set to the "imageAdv" variable of type "ImageView". If only one image is selected, the first element of the "imagesSelect" array is taken for processing.

```
private void Object_Clear(){
    editTxtTitle.setText("");
    editTxtAddress.setText("");
    editTxtPrice.setText("");
    editTxtDues.setText("");
    editTxtBuildingFloors.setText("");
    editTxtBuildAge.setText("");
    editTxtFloorLoc.setText("");
    editTxtRentalIncome.setText("");
    editTxtSquareMt.setText("");
    editTxtNumofBath.setText("");
    spinnerAdvStatus.setSelection(0);
    spinnerBuildType.setSelection(0);
    spinnerFront.setSelection(0);
    spinnerElgbCredit.setSelection(0);
    spinnerFuelType.setSelection(0);
    spinnerItemStatus.setSelection(0);
    spinnerStateOfBuilding.setSelection(0);
    spinnerSwap.setSelection(0);
    spinnerUsingStatus.setSelection(0);
    spinnerCity.setSelection(0);
    spinnerWarmType.setSelection(0);
    spinnerRoomNum.setSelection(0);
    imageAdv.setImageBitmap(firstImage);
    btnSubmitAdv.setEnabled(false);
}
```

Figure 5.1.1.9.9 Clear Input Boxes After Save Advertisement

After the advertisement insertion process is completed, the inside of the input boxes that the user will enter is cleaned and restored to its original state, as in Figure 5.1.1.9.9.

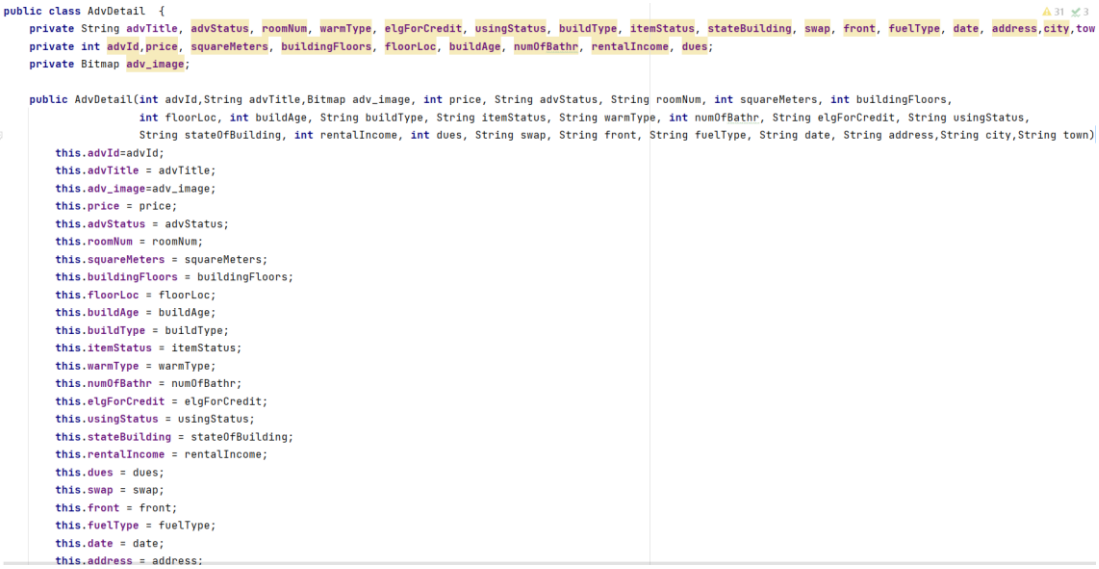
```
@Override
public void onMapReady(@NonNull GoogleMap googleMap) {
    gMap=googleMap;
    gMap.setOnMapClickListener(new GoogleMap.OnMapClickListener() {
        @Override
        public void onMapClick(LatLng latLng) {
            MarkerOptions markerOptions=new MarkerOptions();
            markerOptions.position(latLng);
            markerOptions.title(latLng.latitude+" : "+latLng.longitude);
            System.out.println(latLng.latitude+" : "+latLng.longitude);
            latitude=  latLng.latitude;
            longitude= latLng.longitude;
            gMap.clear();
            gMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latLng, zoom: 10));
            gMap.addMarker(markerOptions);
        }
    });
}
```

Figure 5.1.1.9.10 Advertisement Coordinates Marker in Google Map

The location of the advertisement is marked on the map and the latitude-longtitude information of the marked place is obtained, as seen in Figure 5.1.1.9.10.

5.1.1.10. AdvDetail Class

The properties defined in the *AdvDetail* class are for creating advertisement details. The object created with this class is used in transactions related to advertisement details.



```
public class AdvDetail  {
    private String advTitle, advStatus, roomNum, warmType, elgForCredit, usingStatus, buildType, itemStatus, stateBuilding, swap, front, fuelType, date, address,city,tow
    private int advId,price, squareMeters, buildingFloors, floorLoc, buildAge, numOfBathr, rentalIncome, dues;
    private Bitmap adv_image;

    public AdvDetail(int advId,String advTitle,Bitmap adv_image, int price, String advStatus, String roomNum, int squareMeters, int buildingFloors,
                     int floorLoc, int buildAge, String buildType, String itemStatus, String warmType, int numOfBathr, String elgForCredit, String usingStatus,
                     String stateOfBuilding, int rentalIncome, int dues, String swap, String front, String fuelType, String date, String address,String city,String town)
        this.advId=advId;
        this.advTitle = advTitle;
        this.adv_image=adv_image;
        this.price = price;
        this.advStatus = advStatus;
        this.roomNum = roomNum;
        this.squareMeters = squareMeters;
        this.buildingFloors = buildingFloors;
        this.floorLoc = floorLoc;
        this.buildAge = buildAge;
        this.buildType = buildType;
        this.itemStatus = itemStatus;
        this.warmType = warmType;
        this.numOfBathr = numOfBathr;
        this.elgForCredit = elgForCredit;
        this.usingStatus = usingStatus;
        this.stateBuilding = stateOfBuilding;
        this.rentalIncome = rentalIncome;
        this.dues = dues;
        this.swap = swap;
        this.front = front;
        this.fuelType = fuelType;
        this.date = date;
        this.address = address;
```

Figure 5.1.1.10.1 Advertisement Detail Class

As in Figure 5.1.1.10.1, advertisement title, advertisement status, number of rooms, heating type, credit eligibility, usage status, building type, property status, state building, clearing status, facade, fuel type, date of addition, address, city, town, price, square meters, number of floors of the building, the floor where it is located, age of the building, number of bathrooms, rental income, dues and pictures.

5.1.1.11. Advertisement Detail Page

It is the page where all the information of the advertisement is displayed after clicking on the pages where the advertisements are listed or after adding the ad.

```
else if(MyFavoritesFragment.clickMyFavDetail==true){
    advId=MyFavoritesFragment.advDetail.getAdvId();
    advTitle=MyFavoritesFragment.advDetail.getAdvTitle();
    advImagePng=MyFavoritesFragment.advDetail.getAdv_image();
    price=MyFavoritesFragment.advDetail.getPrice();
    advStatus=MyFavoritesFragment.advDetail.getAdvStatus();
    roomNum=MyFavoritesFragment.advDetail.getRoomNum();
    squareMeters=MyFavoritesFragment.advDetail.getSquareMeters();
    buildingFloors=MyFavoritesFragment.advDetail.getBuildingFloors();
    floorLoc=MyFavoritesFragment.advDetail.getFloorLoc();
    buildAge=MyFavoritesFragment.advDetail.getBuildAge();
    buildType=MyFavoritesFragment.advDetail.getBuildType();
    itemStatus=MyFavoritesFragment.advDetail.getItemStatus();
    warmType=MyFavoritesFragment.advDetail.getWarmType();
    numOfBathr=MyFavoritesFragment.advDetail.getNumOfBathr();
    elgForCredit=MyFavoritesFragment.advDetail.getElgForCredit();
    usingStatus=MyFavoritesFragment.advDetail.getUsingStatus();
    stateBuilding=MyFavoritesFragment.advDetail.getStateBuilding();
    rentalIncome=MyFavoritesFragment.advDetail.getRentalIncome();
    dues=MyFavoritesFragment.advDetail.getDues();
    swap=MyFavoritesFragment.advDetail.getSwap();
    front=MyFavoritesFragment.advDetail.getFront();
    fuelType=MyFavoritesFragment.advDetail.getFuelType();
    date=MyFavoritesFragment.advDetail.getDate();
    address=MyFavoritesFragment.advDetail.getAddress();
    city=MyFavoritesFragment.advDetail.getCity();
    town=MyFavoritesFragment.advDetail.getTown();
}
```

Figure 5.1.1.11.1 Advertisement Details Get Adv Attributes

If one of the advertisements on the home page is clicked, the properties of the "advDetail" object created in "AdvDetail" type in the "HomeFragment" are called and assigned to the relevant variables. If a new advertisement is added, the properties of the "advDetailLast" object created in "AdvDetail" type are assigned to the relevant variables after adding. If the user wants to view the details of one of the advertisements he added, the properties of the "advDetail" object created in the "AdvDetail" type created in the "MyAdvertisementFragment" are assigned to the relevant variables. As in Figure 5.1.1.11.1, if the user wants to view the details of one of the advertisements that he has added to his favourites, the properties of the "advDetail" object created in the "AdvDetail" type created in the "MyFavoritesFragment" are assigned to the relevant variables.

```
@Override
public void onMapReady(@NonNull GoogleMap googleMap) {
    String sqlQuery="SELECT * FROM Advertisements WHERE AdvId = '"+advId+"'";
    Cursor cursor=MainActivity.db.rawQuery(sqlQuery, selectionArgs: null);
    int xCoordinateIndex = cursor.getColumnIndex( columnName: "xCoordinate");
    int yCoordinateIndex = cursor.getColumnIndex( columnName: "yCoordinate");
    while (cursor.moveToNext()) {
        xCoordinateAdv= cursor.getFloat(xCoordinateIndex);
        yCoordinateAdv= cursor.getFloat(yCoordinateIndex);
    }
    mapAPI=googleMap;
    LatLng advCoordinate=new LatLng(xCoordinateAdv,yCoordinateAdv);
    mapAPI.addMarker(new MarkerOptions().position(advCoordinate).title(advTitle));
    mapAPI.moveCamera(CameraUpdateFactory.newLatLng(advCoordinate));
    mapAPI.moveCamera(CameraUpdateFactory.newLatLngZoom(advCoordinate, zoom: 10f));

}
```

Figure 5.1.1.11.2 Show Map in Adv Detail Page

As in Figure 5.1.1.11.2, Google Map has been added to show the location of the advertisement on the map on the advertisement detail page. According to the id information of the advertisement, the x and y coordinate information is given to the latitude and longitude values, and the position of the advertisement on the map is marked.

```
ArrayList<Bitmap>images=new ArrayList<>();
String sqlQuery="SELECT * FROM AdvertisementImage WHERE AdvId = '"+advId+"'";
Cursor imageCursor=MainActivity.db.rawQuery(sqlQuery, selectionArgs: null);
int indexImage=imageCursor.getColumnIndex( s: "advImage");
while (imageCursor.moveToNext()){
    count++;
    byte[] imageByte = imageCursor.getBlob(indexImage);
    Bitmap imageAdv = BitmapFactory.decodeByteArray(imageByte, offset: 0, imageByte.length);
    images.add(imageAdv);
}
txtAdvTitle.setText(advTitle);
advImageView.setImageBitmap(images.get(0));
binding.next.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if(position<(count-1)){
            position++;
            advImageView.setImageBitmap(images.get(position));
        }
    }
});
binding.previous.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if(position>0){
            position--;
            advImageView.setImageBitmap(images.get(position));
        }
    }
});
```

Figure 5.1.1.11.3 View Advertisement of All Images

Other images added for the advertisement can be viewed by using the direction keys on the advertisement details display page. Button type variables called Next and Previous are given the ability to be clicked. In Figure 5.1.1.11.3, If the "next" button is clicked, the "position" is increased and the next picture is displayed. If the "previous" button is clicked, the "position" is reduced and the previous picture is shown.

```
Intent intent=getActivity().getIntent();
User user=(User)intent.getSerializableExtra( name: "UserInformation");
MainActivity.database.onCreate(MainActivity.db);
db = MainActivity.database.getWritableDatabase();
Drawable addFav = getResources().getDrawable(R.drawable.ic_baseline_favorite_24);
Drawable notFav = getContext().getDrawable(R.drawable.ic_baseline_favorite_border_24);
if(user==null){
    binding.buttonFav1.setImageDrawable(notFav);
}
else{
    Cursor c1 = db.rawQuery( sql: "SELECT * FROM Favorite WHERE UserId=?  AND AdvId=? AND FavoriteStatus=? ;",
            new String[]{String.valueOf(user.getUserId()), String.valueOf(advId), "1"});
    if(c1.moveToFirst()){
        binding.buttonFav1.setImageDrawable(addFav);
    }
    else{
        binding.buttonFav1.setImageDrawable(notFav);
    }
}
```

Figure 5.1.1.11.4 Favorite Button in Advertisement Detail

In Figure 5.1.1.11.4, if no user has logged in to the application, the favorite icon appears empty on the detail page of all advertisements. If any user is logged into the application, the favorite icon appears in the details of the advertisements in the favorite.

```
binding.buttonFav1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if(user==null){
            Toast.makeText(getActivity(),  text: "Please login to your account!!", Toast.LENGTH_SHORT).show();
        }
        else{
            Cursor c = db.rawQuery( sql: "SELECT * FROM Favorite WHERE UserId=?  AND AdvId=? AND FavoriteStatus=? ;",
                    new String[]{String.valueOf(user.getUserId()),  String.valueOf(advId), "1"});
            if(c.moveToFirst()){
                binding.buttonFav1.setImageDrawable(notFav);
                db.execSQL("UPDATE Favorite SET FavoriteStatus = 0 WHERE UserId = "+ String.valueOf(user.getUserId())+
                        " AND AdvId="+String.valueOf(advId)+";");
                Toast.makeText(getActivity(),  text: "It has been deleted to the list of favorites !!",
                        Toast.LENGTH_SHORT).show();
            }
            else{
                binding.buttonFav1.setImageDrawable(addFav);
                c = db.rawQuery( sql: "SELECT * FROM Favorite WHERE UserId=?  AND AdvId=? AND FavoriteStatus=? ;",
                        new String[]{String.valueOf(user.getUserId()), String.valueOf(advId), "0"});
                if(c.moveToFirst()) {
                    db.execSQL("UPDATE Favorite SET FavoriteStatus = 1 WHERE UserId = "+ String.valueOf(user.getUserId())+
                            " AND AdvId= "+String.valueOf(advId)+";");

                    Toast.makeText(getActivity(),  text: "It has been added to the list of favorites !!", Toast.LENGTH_SHORT).show();
                }
                else{
                    MainActivity.database.onCreate(MainActivity.db);
                    String sqlQuery="INSERT INTO Favorite (UserId ,AdvId ,FavoriteStatus)  VALUES(?,?,?);";
                    SQLiteStatement statement = MainActivity.db.compileStatement(sqlQuery);
                    statement.bindString( index: 1,String.valueOf(user.getUserId()));
                    statement.bindString( index: 2,String.valueOf(advId));
```

Figure 5.1.1.11.5 Insert or Delete Advertisement Favorites

If a user who is not logged into the application clicks the favorite button on the advertisement detail page, he or she will receive a warning message to log in to the application. If the user has logged into the application and clicked on the advertisement favorite button on the detail page when it is empty, it is checked whether the user has added this advertisement to their favorites before. If this advertisement has been added to its favorites before, the "status" variable is set to "1" and the favorite button is filled. If he has not added this advertisement to his favourites, it is added to the "Favorite" table with the advertisement id and user id information, and the favorite button is filled. If the user wants to remove the advertisement from his favorites, when he clicks on the favorite button, the "status" value is set to "0" and the button is emptied in Figure 5.1.1.11.5.

5.1.1.12. My Favorites Page

```
Intent intent=getActivity().getIntent();
User user=(User)intent.getSerializableExtra( name: "UserInformation");
clickMyFav=true;
MainActivity.database.onCreate(MainActivity.db);
db = MainActivity.database.getWritableDatabase();
View view=inflater.inflate(R.layout.fragment_my_favorites,container, attachToRoot: false);
recyclerView=(RecyclerView) view.findViewById(R.id.recviewMyFavs);
recyclerView.setLayoutManager(new LinearLayoutManager(getContext()));
Cursor c1 = db.rawQuery( sql: "SELECT * FROM Favorite as fav, Advertisements as adv, UserAdvertisement as usAdv "+
            " WHERE fav.AdvId=usAdv.AdvId AND fav.FavoriteStatus=1 AND fav.AdvId=adv.AdvId AND fav.UserId=?;",
                new String[]{String.valueOf(user.getUserId())});
adv=new ArrayList<>();
int advTitleIndex = c1.getColumnIndex( s: "AdvTitle");
int ImageIndex = c1.getColumnIndex( s: "AdvImage");
int priceIndex = c1.getColumnIndex( s: "Price");
int advStatusIndex = c1.getColumnIndex( s: "AdvStatus");
int roomNumIndex = c1.getColumnIndex( s: "RoomNum");
int squareMeterIndex = c1.getColumnIndex( s: "SquareMeter");
int buildingFloorsIndex = c1.getColumnIndex( s: "BuildingFloors");
int floorLocIndex = c1.getColumnIndex( s: "FloorLoc");
int buildAgeIndex = c1.getColumnIndex( s: "BuildAge");
int buildTypeIndex = c1.getColumnIndex( s: "BuildType");
int itemStatusIndex = c1.getColumnIndex( s: "ItemStatus");
int warmTypeIndex = c1.getColumnIndex( s: "WarmType");
int numOfBathroomsIndex = c1.getColumnIndex( s: "NumOfBathrooms");
int elgCreditIndex = c1.getColumnIndex( s: "ElgCredit");
int usingStatusIndex = c1.getColumnIndex( s: "UsingStatus");
int stateBuildingIndex = c1.getColumnIndex( s: "StateBuilding");
int rentalIncomeIndex = c1.getColumnIndex( s: "RentalIncome");
int duesIndex = c1.getColumnIndex( s: "Dues");
```

Figure 5.1.1.12.1 My Favorites Listing

It is the page that lists the advertisements that the user likes. As in Figure 5.1.1.12.1, the user's id information from the "Favorite" table, the advertisements that the user has added to their favorites and the "status" value is "1" are brought from the table and assigned to the variables. According to the column indexes of the table, the advertisement properties are set to the relevant properties of the "newAdv" object created with the "Advertisement" class. This "newAdv" object is assigned to the "adv" list. Thanks to this "adv" list, it is possible to reach the details of the advertisement.

5.1.1.13. Filter Advertisement Page

It is for the user to view the advertisements with the features they want when they click on the "Filter" button in the topbar menu on the first page opened in the application.

```
binding.FilterBtnApply.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        try {
            priceMin = Integer.parseInt(editTxtPriceMin.getText().toString());
            priceMax = Integer.parseInt(editTxtPriceMax.getText().toString());
            squareMeterMin = Integer.parseInt(editTxtSquareMin.getText().toString());
            squareMeterMax = Integer.parseInt(editTxtSquareMax.getText().toString());
            buildingFloorsMin = Integer.parseInt(editTextBuildFloorMin.getText().toString());
            buildingFloorsMax = Integer.parseInt(editTextBuildFloorMax.getText().toString());
            floorLocMin = Integer.parseInt(editTxtFloorLocMin.getText().toString());
            floorLocMax = Integer.parseInt(editTxtFloorLocMax.getText().toString());
            buildAgeMin = Integer.parseInt(editTextBuildAgeMin.getText().toString());
            buildAgeMax = Integer.parseInt(editTextBuildAgeMax.getText().toString());
            numOfBathrMin = Integer.parseInt(editTxtNumofBathMin.getText().toString());
            numOfBathrMax = Integer.parseInt(editTxtNumofBathMax.getText().toString());
            rentalIncomeMin = Integer.parseInt(editTxtRentalIncomeMin.getText().toString());
            rentalIncomeMax = Integer.parseInt(editTxtRentalIncomeMax.getText().toString());
            duesMin = Integer.parseInt(editTxtDuesMin.getText().toString());
            duesMax = Integer.parseInt(editTxtDuesMax.getText().toString());
        }catch (Exception e){
            System.out.println("Error");
        }
        if(filterInputControl()==true){
            applButton=true;
            HomeFragment homeFragment = new HomeFragment();
            FragmentManager fragmentManager = getFragmentManager();
            FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
            fragmentTransaction.replace(R.id.nav_host_fragment_activity_main, homeFragment);
            fragmentTransaction.commit();
        }
    }
```

Figure 5.1.1.13.1 Filter Button Apply

In this page, the selections made by the user among the numerical input in Figure 5.1.1.13.1 value "min" and "max" value ranges and the dropdown options are assigned to the relevant variables. These variables are given as parameter values in the "Advertisements" table in the database, using the "BETWEEN" key, and other selections in the "new String{ }". Advertisements that comply with these conditions are shown to the user.

5.1.1.14. My Account Page

```
binding.myFav.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        clickMyFav=true;
        FragmentManager fragmentManager=getActivity().getSupportFragmentManager();
        FragmentTransaction fragmentTransaction=fragmentManager.beginTransaction();
        fragmentTransaction.replace(R.id.nav_host_fragment_activity_main,new MyFavoritesFragment());
        fragmentTransaction.commit();
    }
});
binding.exitAccount.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View view) {
        AlertDialog.Builder builder=new AlertDialog.Builder(getContext());
        builder.setMessage("Are you sure?").setPositiveButton( text: "Yes", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                Intent intent=getActivity().getIntent();
                User user=(User)intent.getSerializableExtra( name: "UserInformation");
                user=null;
                intent.putExtra( name: "UserInformation",user);
                FragmentManager fragmentManager=getActivity().getSupportFragmentManager();
                FragmentTransaction fragmentTransaction=fragmentManager.beginTransaction();
                fragmentTransaction.replace(R.id.nav_host_fragment_activity_main,new LoginFragment());
                fragmentTransaction.commit();
            }
        }).setNegativeButton( text: "No", listener: null );
        AlertDialog alert=builder.create();
        alert.show();
    }
});
```

Figure 5.1.1.14.1 Directed Pages in My Account Page

The user who is logged into the application displays his/her own user information, advertisements he/she added, favorite advertisements and exit buttons. As in Figure 5.1 1.14.1, these buttons are directed to the relevant pages.

5.1.1.15. MyAdvertisementAdapter Class

The MyAdvertisementAdapter class has similar methods and properties as the AdvertisementAdapter class. In this class, in addition to the advertisement card, there is a "three-dot" option menu in the my advertisement card in order to display the advertisements of the user.

```
@Override
public void onBindViewHolder(MyAdvertisementAdapter.ViewHolder holder, int position) {
    holder.img.setImageBitmap(adv.get(position).getAdvImage());
    holder.titlee.setText(adv.get(position).getAdvTitle());
    holder.addresss.setText(adv.get(position).getAddress());
    holder.pricee.setText(String.valueOf(adv.get(position).getPrice()));
   int pos=position;
    holder.buttonViewOption.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            PopupMenu popup = new PopupMenu(context, holder.buttonViewOption);
            popup.inflate(R.menu.option_myadv);
            popup.setOnMenuItemClickListener(new PopupMenu.OnMenuItemClickListener() {
                @Override
                public boolean onMenuItemClick(MenuItem menuItem) {
                    if(menuItem.getItemId()==R.id.adv_update){
                        clickAdvUpdate=true;
                        advId=adv.get(pos).getAdvId();
                        advTitle=adv.get(pos).getAdvTitle();
                        advStatus=adv.get(pos).getAdvStatus();
                        roomNum=adv.get(pos).getRoomNum();
                        warmType=adv.get(pos).getWarmType();
                        elgForCredit=adv.get(pos).getElgForCredit();
                        usingStatus=adv.get(pos).getUsingStatus();
                        buildType=adv.get(pos).getBuildType();
                        itemStatus=adv.get(pos).getItemStatus();
                        stateBuilding=adv.get(pos).getStateBuilding();
                        swap=adv.get(pos).getSwap();
                        front=adv.get(pos).getFront();
                        fuelType=adv.get(pos).getFuelType();
                        date=adv.get(pos).getDate();
                        address=adv.get(pos).getAddress();
```

Figure 5.1.1.15.1 My Advertisement Adapter Create an Option Menu

As seen in Figure 5.1.1.15.1, the picture, the title of the ad, the address of the ad, the price and the option menu with three dots are displayed. When you click on this selection button using the properties and methods of the "PopupMenu" class, "advertising update" and "posting deletion" options appear. If the "update" option is selected, the properties of the advertisement clicked on the selection menu are called according to the "position" value, and the properties of the object of type "Advertisement" are called and assigned to the relevant variables. These variables defined as public static are redirected to "MyAdvUpdateFragment" to be processed. If the user wants to delete the advertisement that he clicked on the selection menu, the advertisement id information is found according to the "position" value of

the advertisement and the deletion process is performed according to the "AdvId" property.

5.1.1.16. My Advertisement Update Page

"MyAdvUpdateFragment" works if "update" in the "option menu" button of the advertisements is selected on the page where the advertisements added by the user are listed. As on the advertisement posting page, there are selection inputs in the form of input boxes and dropdowns according to the relevant features of the ad.

```
public void initFirst(){
    binding.addAdvEditTextAdvTitle.setText(MyAdvertisementAdapter.advTitle);
    String sqlQuery="SELECT * FROM AdvertisementImage WHERE AdvId = '"+MyAdvertisementAdapter.advId+"'";
    Cursor imageCursor=MainActivity.db.rawQuery(sqlQuery, selectionArgs: null);
    int indexImage=imageCursor.getColumnIndex( s: "advImage");
    int indexImageId=imageCursor.getColumnIndex( s: "ImageId");
    int indexAdvId=imageCursor.getColumnIndex( s: "AdvId");
    while (imageCursor.moveToNext()){
        count++;
        advIdImages.add(Integer.parseInt(imageCursor.getString(indexAdvId)));
        imagesId.add(Integer.parseInt(imageCursor.getString(indexImageId)));
        byte[] imageByte = imageCursor.getBlob(indexImage);
        Bitmap imageAdv = BitmapFactory.decodeByteArray(imageByte,  offset: 0, imageByte.length);
        images.add(imageAdv);
    }
    binding.addAdvImage.setImageBitmap(images.get(position));
    binding.addAdvEditTextPrice.setText(String.valueOf(MyAdvertisementAdapter.price));
    spinnerAdvStatus=(Spinner)binding.addAdvSpinnerAdvStatus;
    ArrayAdapter<CharSequence> adapterAdvStatus=ArrayAdapter.createFromResource(getActivity().getBaseContext(),
            R.array.Adv_Status, android.R.layout.simple_spinner_item);
    adapterAdvStatus.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    spinnerAdvStatus.setAdapter(adapterAdvStatus);
    if (MyAdvertisementAdapter.advStatus != null) {
        int spinnerPosition = adapterAdvStatus.getPosition(MyAdvertisementAdapter.advStatus);
        spinnerAdvStatus.setSelection(spinnerPosition);
        tempStatus=adapterAdvStatus.getItem(spinnerPosition).toString();
    }
```

Figure 5.1.1.16.1 Setting Advertisement Current Information

As can be seen in Figure 5.1.1.16.1, the current advertisement information of the advertisement to be updated is set in the relevant input box of the advertisement, in which the information of the advertisement that was clicked on the "selection menu" in the "MyAdvertisementAdapter" class is stored within the "initFirst" method. According to the id information of the advertisement to be updated, the properties of the pictures of the

advertisement are retrieved from the "AdvertisementImage" table and this information is added to the appropriate type arrays with a while loop. Each image is taken in byte type and then assigned to a variable of type "Bitmap" using the "BitmapFactory.decodeByteArray" method. This bitmap type variable is added to an array defined as "Bitmap" for display.

```
    elgForCredit=spinnerElgbCredit.getSelectedItem().toString();
    usingStatus=spinnerUsingStatus.getSelectedItem().toString();
    stateBuilding=spinnerStateOfBuilding.getSelectedItem().toString();
    swap=spinnerSwap.getSelectedItem().toString();
    front=spinnerFront.getSelectedItem().toString();
    fuelType=spinnerFuelType.getSelectedItem().toString();
    city=spinnerCity.getSelectedItem().toString();
    town=spinnerTown.getSelectedItem().toString();
}catch (Exception e ){
    System.out.println("Error");
}
ByteArrayOutputStream outputStream =new ByteArrayOutputStream();
if(selectedİmg==null)
{
    System.out.println("There is not image");
    selectedİmg=MyAdvertisementAdapter.selectedİmg;
}
for(int a=0;a<imagesSelect.size();a++)
{
    ByteArrayOutputStream outputStreamMulti =new ByteArrayOutputStream();
    smallestedImg=imageSmall(imagesSelect.get(a));
    smallestedImg.compress(Bitmap.CompressFormat.PNG, quality: 75,outputStreamMulti);
    byte[] ImageMulti=outputStreamMulti.toByteArray();
    updateAdvImages=database.updateMyAdvImages(imagesId.get(a),ImageMulti,advIdImages.get(0));
}
smallestedImg=imageSmall(firstSelectedImage);
smallestedImg.compress(Bitmap.CompressFormat.PNG, quality: 75,outputStream);
byte[] Image=outputStream.toByteArray();
updateAdv=database.updateMyAdv(advId,advTitle,Image,price,advStatus,roomNum,squareMeters,buildingFloors,floorLoc,
        buildAge,buildType,itemStatus,warmType,numOfBathr,elgForCredit,usingStatus,stateBuilding,rentalIncome,dues,
        swap,front,fuelType,date,address,city,town,latitude,longitude);
```

Figure 5.1.1.16.2 Updating Advertisement of Images and Information

When the user clicks the "Update Advertisement" button after changing, the latest features of the advertisement are assigned to the relevant variables.

As in Figure 5.1.1.16.2, the "updateMyAdvImages" function in the "Database" class is called with the pictures of the advertisement and the pictures selected in the for loop, and each selected picture is given as a parameter to the function with the id information of the picture and the advertisement in a byte array type. Variables that hold the final information of the advertisement are also sent as parameters to the "updateMyAdv" function in the "Database" class. If both of these two functions return the

value "1" meaning the update process is successful, a message is given to the user asking if he is sure about the update process. If "Yes" is selected, all the features will be updated with the posting, and then "MyAccountFragment" will be redirected.

As in the advertisement add page, selecting an image from the gallery, activating the permission to select an image from the gallery, adding the selected images to arrays for display, displaying the next or previous image, getting input from the user, implementing dropdown selection inputs, advertisement input control, location on the map Methods for marking and obtaining coordinate information of the marked location are also available on this page.

5.1.1.17. My Advertisement Page

This page is used to show the advertisements added by the user who logs in to the application, in the type of advertisement cards, just like on the home page.

```
sqlQuery = "SELECT AdvId FROM UserAdvertisement WHERE UserId= ?";
cursor = MainActivity.db.rawQuery(sqlQuery, new String[]{String.valueOf(MyAccountFragment.userMyId)});
while (cursor.moveToNext()) {
    int advertisementId = Integer.parseInt(cursor.getString( i: 0));
    sqlQuery = "SELECT * FROM  Advertisements WHERE AdvId = ? ";
    Cursor cursorTemp = MainActivity.db.rawQuery(sqlQuery, new String[]{String.valueOf(advertisementId)});
    int advTitleIndex = cursorTemp.getColumnIndex( s: "AdvTitle");
    int ImageIndex = cursorTemp.getColumnIndex( s: "AdvImage");
    int priceIncex = cursorTemp.getColumnIndex( s: "Price");
    int advStatusIndex = cursorTemp.getColumnIndex( s: "AdvStatus");
    int roomNumIndex = cursorTemp.getColumnIndex( s: "RoomNum");
    int squareMeterIndex = cursorTemp.getColumnIndex( s: "SquareMeter");
    int buildingFloorsIndex = cursorTemp.getColumnIndex( s: "BuildingFloors");
    int floorLocIndex = cursorTemp.getColumnIndex( s: "FloorLoc");
    int buildAgeIndex = cursorTemp.getColumnIndex( s: "BuildAge");
    int buildTypeIndex = cursorTemp.getColumnIndex( s: "BuildType");
    int itemStatusIndex = cursorTemp.getColumnIndex( s: "ItemStatus");
    int warmTypeIndex = cursorTemp.getColumnIndex( s: "WarmType");
    int numOfBathroomsIndex = cursorTemp.getColumnIndex( s: "NumOfBathrooms");
    int elgCreditIndex = cursorTemp.getColumnIndex( s: "ElgCredit");
    int usingStatusIndex = cursorTemp.getColumnIndex( s: "UsingStatus");
    int stateBuildingIndex = cursorTemp.getColumnIndex( s: "StateBuilding");
    int rentalIncomeIndex = cursorTemp.getColumnIndex( s: "RentalIncome");
    int duesIndex = cursorTemp.getColumnIndex( s: "Dues");
    int swapIndex = cursorTemp.getColumnIndex( s: "Swap");
    int frontIndex = cursorTemp.getColumnIndex( s: "Front");
    int fuelTypeIndex = cursorTemp.getColumnIndex( s: "FuelType");
    int dateIndex = cursorTemp.getColumnIndex( s: "Date");
    int addressIndex = cursorTemp.getColumnIndex( s: "Address");
    int cityIndex = cursorTemp.getColumnIndex( s: "Cities");
    int townIndex = cursorTemp.getColumnIndex( s: "Town");
    int xCoordinateIndex = cursorTemp.getColumnIndex( s: "xCoordinate");
```

Figure 5.1.1.17.1 User's Advertisement from Database

As seen in Figure 5.1.1.17.1, "AdvId"s are taken according to the user id information in the "UserAdvertisement" table in the database. According to the "AdvId" value received, each feature of the advertisements belonging to the user in the database from the "Advertisement" table is set using the while loop, and the properties of the advertisement are set with the relevant variables. It is added to the arraylist named "advertisementList" of type "Advertisement".

With the object created with the "RecyclerView" class, the advertisements added by the user are dynamically retrieved from the database. With the object created with the "MyAdvertisementAdapter" class, the ability to click on the advertisement is brought, and the "advDetail" object is created by bringing the properties in the advertisement

detail with the "AdvDetail" class. As in the advertisement card, each advertisement is listed on the My Advertisement page as clickable.

5.1.1.18. User Profile Page

In the user profile page, the id of the logged in application user and the user information with that id from the "Users" table in the database are displayed in the input boxes. Thanks to the "Update" button under the input boxes, the user information is updated.

```
binding= FragmentUserProfileBinding.inflate(inflater,container, attachToParent: false);
Intent intent=getActivity().getIntent();
User user=(User)intent.getSerializableExtra( name: "UserInformation");
binding.inputUserName.setText(user.getUserName());
binding.inputUserSurname.setText(user.getUserSurname());
binding.inputUserMail.setText(user.getMailAddress());
buttonUpdate=(Button) binding.buttonUpdate;
binding.buttonUpdate.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        int userId=user.getUserId();
        String userPassword=user.getPassword();
        inputUserName=(EditText)binding.inputUserName;
        inputUserSurname=(EditText)binding.inputUserSurname;
        inputUserMail=(EditText) binding.inputUserMail;
        inputPassword=(EditText) binding.inputPassword;
        inputNewPassword=(EditText) binding.inputNewPassword;

        if(!TextUtils.isEmpty(inputUserName.getText().toString()) && !TextUtils.isEmpty(inputUserSurname.getText().toString()) &&
                !TextUtils.isEmpty(inputUserMail.getText().toString()) && !TextUtils.isEmpty(inputPassword.getText().toString()) ){

            Database database=new Database(getContext());

            if(!database.checkEmailExist(inputUserMail.getText().toString())){
                Toast.makeText(getActivity(), text: "You cannot use this e-mail address !!", Toast.LENGTH_SHORT).show();
            }
            else if(inputPassword.getText().toString().equals(userPassword)){
                int updateUserInfo=0;
                String tempPassword="";
                int temp=0;
```

Figure 5.1.1.18.1 Input Control in User Information Update

If the user updates the name, surname, mail and password information as in Figure 5.1.1.18.1, some checks are made. All inputs must not be empty. Since the e-mail address will be unique to each user, the status of the changed e-mail address in the database is checked. Format control of e-mail address is done. If the password is to be updated, the length of the new password is checked. For security, the user's password is requested to be re-

entered while updating the information. If the entered password and the user's password in the database match, the update process is performed.

```
if(!TextUtils.isEmpty(inputNewPassword.getText().toString())){
    if(inputNewPassword.getText().toString().length()<8){
        temp=1;
    }
    tempPassword=inputNewPassword.getText().toString();
}
else{
    tempPassword=inputPassword.getText().toString();
}

updateUserInfo= database.updateUser(userId,inputUserName.getText().toString(),inputUserSurname.getText().toString(),
        inputUserMail.getText().toString(),inputPassword.getText().toString(),tempPassword);

if(updateUserInfo==1&&temp==0){
    AlertDialog.Builder builder=new AlertDialog.Builder(getContext());
    String finalTempPassword = tempPassword;
    builder.setMessage("Are you sure?").setPositiveButton( text: "Yes", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialogInterface, int i) {
            user.setPassword(finalTempPassword);
            user.setUserName(inputUserName.getText().toString());
            user.setUserSurname(inputUserSurname.getText().toString());
            user.setMailAddress(inputUserMail.getText().toString());

            UserProfileFragment userProfileFragment=new UserProfileFragment();
            FragmentManager fragmentManager=getFragmentManager();
            FragmentTransaction fragmentTransaction=fragmentManager.beginTransaction();
            fragmentTransaction.replace(R.id.nav_host_fragment_activity_main,userProfileFragment);
            fragmentTransaction.commit();
        }
    }).setNegativeButton( text: "No", listener: null );
```

Figure 5.1.1.18.2 User Information Update in Database

When the information will be updated, the user is asked to confirm again. If the user approves the updating of the information, the user information in the database is updated and the user profile page is displayed with the updated information in Figure 5.1.1.18.2.

5.1.1.19. ServiceManage Class

ServiceManage class is a class in which we write functions to connect to the database on the remote server. As seen in Figure 5.1.1.19.1, the url of the server, the namespace and the name of the method we will use are defined on the server. SoapObject, SoapSerializationEnvelope and HttpTransportSE objects are defined to be used.

```
public class ServiceManage {

    private static final String NAMESPACE="http://tempuri.org/";
    private static final String METHOD_NAME_ADDAdvertisement="addAdvertisement";
    private static final String URL="http://193.140.150.95/deuarSrv/WebService1.asmx";
    private SoapObject soapObject;
    private SoapSerializationEnvelope soapSerializationEnvelope;
    private HttpTransportSE httpTransportSE;
```

Figure 5.1.1.19.1 ServiceManage Class Define Variables

The "addAdvertisement" method is used to add database advertisements on the server. This method takes many properties related to the ad, such as the advertisement title, price, number of rooms, as parameters. The SoabObject object is created according to the namespace and method name. By using the "addProperty" property of this object, properties related to the advertisement are added to certain columns in Figure 5.1.1.19.2 .

```
public void addAdvertisement(String advTitle, int price, String advStatus, String roomNum, int squareMeter, int buildingFloors,
          int floorLoc, int buildAge, String buildType, String itemStatus, String warmType, int numOfBathRooms,
          String elgCredit, String usingStatus, String stateBuilding, int rentalIncome, int dues, String swap,
          String front, String fuelType, Date date, String address, String city, String town, double xCoordinate, double yCoordinate)
{
    soapObject=new SoapObject(NAMESPACE,METHOD_NAME_ADDAdvertisement);
    soapObject.addProperty( name: "advTitle", advTitle);
    soapObject.addProperty( name: "price", price);
    soapObject.addProperty( name: "advStatus", advStatus);
    soapObject.addProperty( name: "roomNum", roomNum);
    soapObject.addProperty( name: "squareMeter", squareMeter);
    soapObject.addProperty( name: "buildingFloors", buildingFloors);
    soapObject.addProperty( name: "floorLoc", floorLoc);
    soapObject.addProperty( name: "buildAge", buildAge);
    soapObject.addProperty( name: "buildType", buildType);
    soapObject.addProperty( name: "itemStatus", itemStatus);
    soapObject.addProperty( name: "warmType", warmType);
    soapObject.addProperty( name: "numOfBathRooms", numOfBathRooms);
    soapObject.addProperty( name: "elgCredit", elgCredit);
    soapObject.addProperty( name: "usingStatus", usingStatus);
    soapObject.addProperty( name: "stateBuilding", stateBuilding);
    soapObject.addProperty( name: "rentalIncome", rentalIncome);
    soapObject.addProperty( name: "dues", dues);
    soapObject.addProperty( name: "swap", swap);
    soapObject.addProperty( name: "front", front);
    soapObject.addProperty( name: "fuelType", fuelType);
```

Figure 5.1.1.19.2 Add Advertisement in Server Database

The "soapSerializationEnvelope" object is created according to SoapEnvelope.VER11, the dotNet of this object is set to true and the encodingStyle is set to utf-8. The previously created soapObject object is sent to the setOutputSoapObject function of the

"soapSerializationEnvelope" object. The "httpTransportSE" object is created with the previously defined URL information. Namespace, method name and soapSerializationEnvelope are sent as parameters to the "call" function of this object in Figure 5.1.1.19.3.

```
    PropertyInfo pi = new PropertyInfo();
    pi.setName("date");
    pi.setValue(date.toString());
    pi.setType(Date.class);
    soapObject.addProperty(pi);
    soapObject.addProperty( name: "address", address);
    soapObject.addProperty( name: "city", city);
    soapObject.addProperty( name: "town", town);
    soapObject.addProperty( name: "xCoordinate", Double. toString(xCoordinate));
    soapObject.addProperty( name: "yCoordinate",Double. toString(yCoordinate));

    soapSerializationEnvelope=new SoapSerializationEnvelope(SoapEnvelope.VER11);
    soapSerializationEnvelope.dotNet=true;
    soapSerializationEnvelope.encodingStyle = "utf-8";
    soapSerializationEnvelope.setOutputSoapObject(soapObject);

    httpTransportSE=new HttpTransportSE(URL);
    httpTransportSE.debug=true;

    try{
        httpTransportSE.call( soapAction: NAMESPACE+METHOD_NAME_ADDAdvertisement,soapSerializationEnvelope);
        SoapObject response=(SoapObject) soapSerializationEnvelope.getResponse();
    }catch (Exception e){
        e.printStackTrace();
    }
}
```

Figure 5.1.1.19.3 Sending Parameters To The Method On The Server

5.1.1.20. AR Fragment Class

When AR button is pressed, AR fragment class is called. In this class, the Unity application is run by giving the package name of the Unity application as the package name in Figure 5.1.1.20.1.

```java
@Override
public void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); }

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    binding= FragmentArBinding.inflate(inflater,container, attachToParent: false);
    binding.button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            PackageManager manager = getContext().getPackageManager();
            Intent intent = manager.getLaunchIntentForPackage( packageName: "com.arRealEstate.ARRealEstate");
            intent.addCategory(Intent.CATEGORY_LAUNCHER);
            startActivity(intent);
        }
    });
    return binding.getRoot();
```

Figure 5.1.1.20.1 Call Unity Application

5.1.2. Application Interfaces

5.1.2.1. Home Page

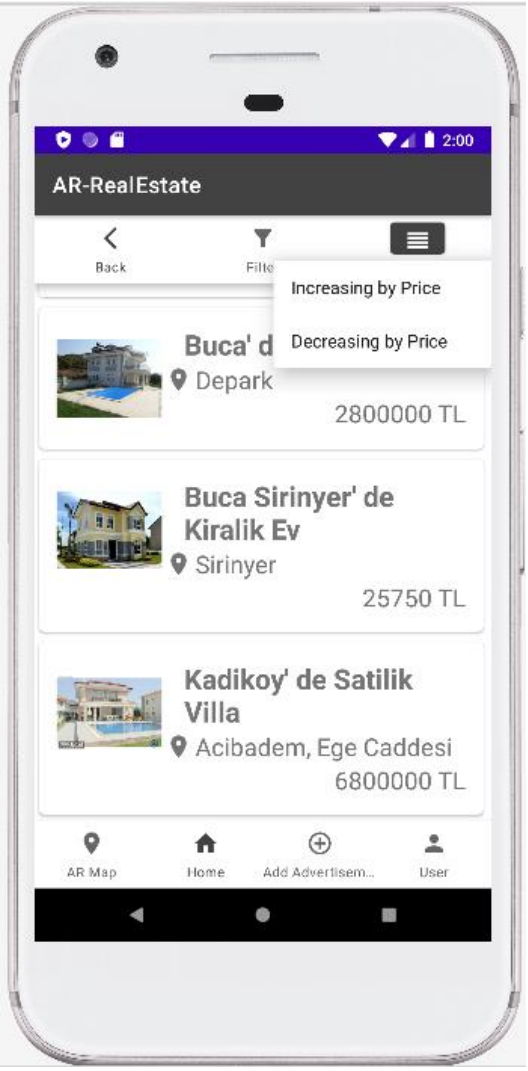


Figure 5.1.2.1.1 Application Home Page

The screen that will be encountered when the application is entered for the first time is the screen in Figure 5.1.2.1.1. Here, all the advertisements in the database are displayed. When the advertisements are clicked, the details of that advertisement are shown. At the top of the screen are the filtering and sorting options. In the lower navigation bar, there are "AR Map", "Home", "Add Advertisement" and "User" buttons. Clicking on the filter button allows the user to select the desired features and list the

advertisements with these features. When you click on the Order button, "Ascending by price" and "Descending by price" options appear. According to these options, it is ensured that the advertisements are listed in order according to the price. Clicking on the "AR Map" button will direct you to the AR application. When you click on the "Home" button, the user is directed to the home page from the other pages, with the "Add Advertisement" button, the user is directed to the add advertisement page. Clicking on the "User" button will go to the login screen, if the user has logged in before, they will be directed to the "My Account" page.

5.1.2.2. Login Page

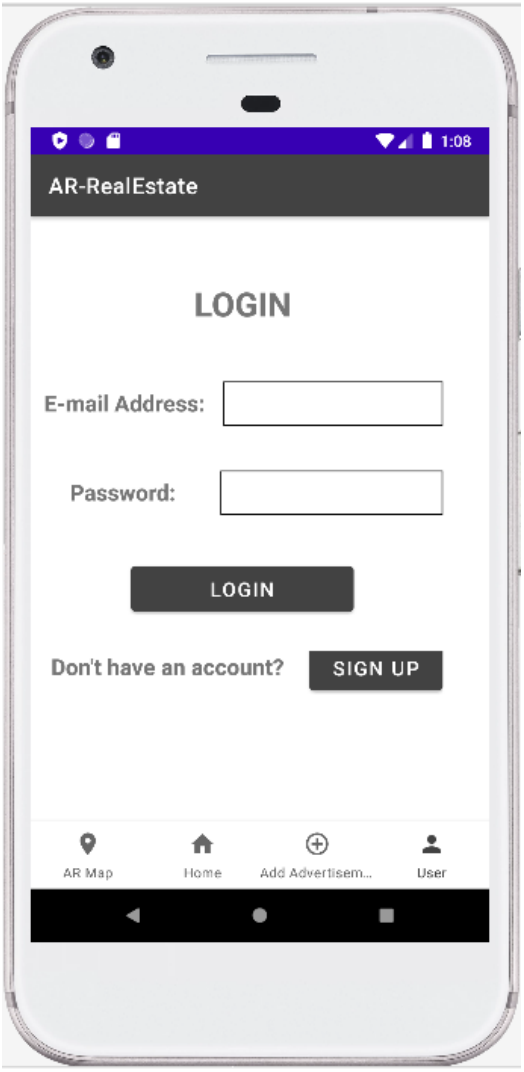


Figure 5.1.2.2.1 Application Login Page

When the user button in the navigation bar is clicked, the screen in Figure 5.1.2.2.1 is displayed. The user must be registered to the application for all operations other than viewing the homepage and advertisement details, and if registered, he must log in. Email and password information is requested for the user to log in. If the correctness and matching of the information entered in the database is correct, he can log in. If the user does not have an account, they are directed to the registration screen with the "Sign Up" button.

5.1.2.3. Sign Up Page

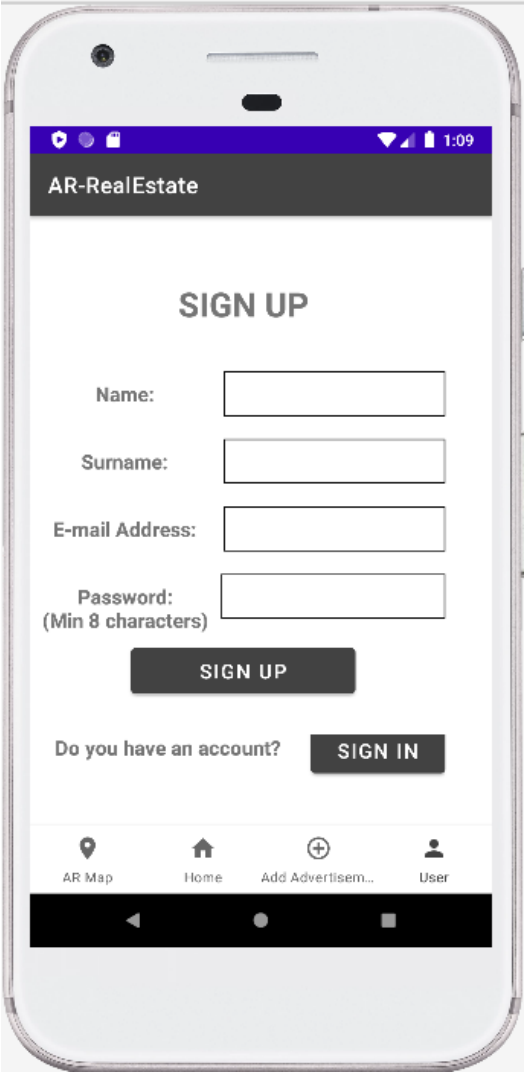


Figure 5.1.2.3.1 Application Sign Up Page

The user must log in to the application for operations other than viewing the homepage and advertisement details. If the user is not registered with the application, name, surname, e-mail address and an eight-character password are requested as seen on the screen in Figure 5.1.2.3.1. Email and password must be in the appropriate format. If the registration is successful, it is directed to the login page. If the registration is not successful, an error message is given to the user.

5.1.2.4. Add Advertisement Page

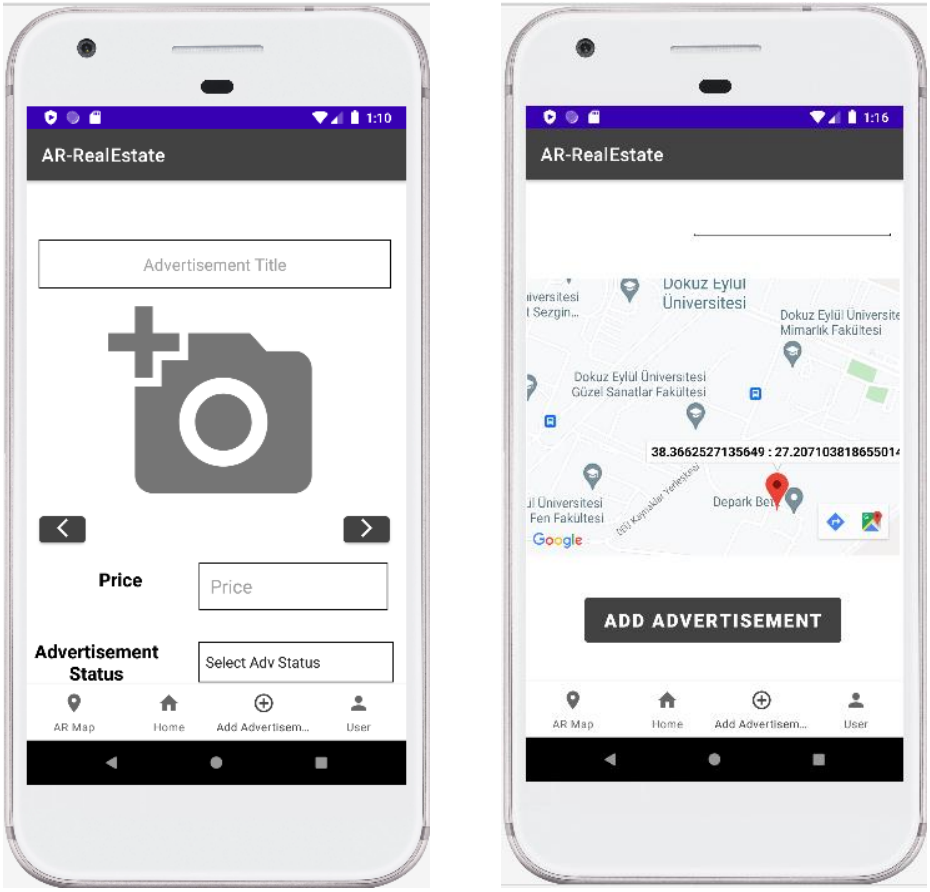


Figure 5.1.2.4.1 Application Add Advertisement Page

If the user is logged into the application and wants to add an advertisement, when he clicks on the "Add Advertisement" button on the navigation bar on the homepage, he is directed to the adding advertisement screen in Figure 5.1.2.4.1. The title of the ad, price information, how many square meters, the number of floors of the building, the floor of the house in the ad, the age of the building, the number of bathrooms, rental income, dues and address information are requested from the user as input. It is requested to choose one of the multiple options, which are listed as the status of the advertisement, number of rooms, type of building, property, heating type, suitability for credit, usage status, state of building, exchange, facade, fuel type, city, town information. When the photo icon is clicked to add the photos of the ad, permission to access the gallery is requested and if the user gives permission, they are directed to the phone's gallery. One or more pictures are selected in the gallery and displayed to the user as photos of the ad. The location of the advertisement is marked on the map at the bottom of the page.

5.1.2.5. Advertisement Detail Page

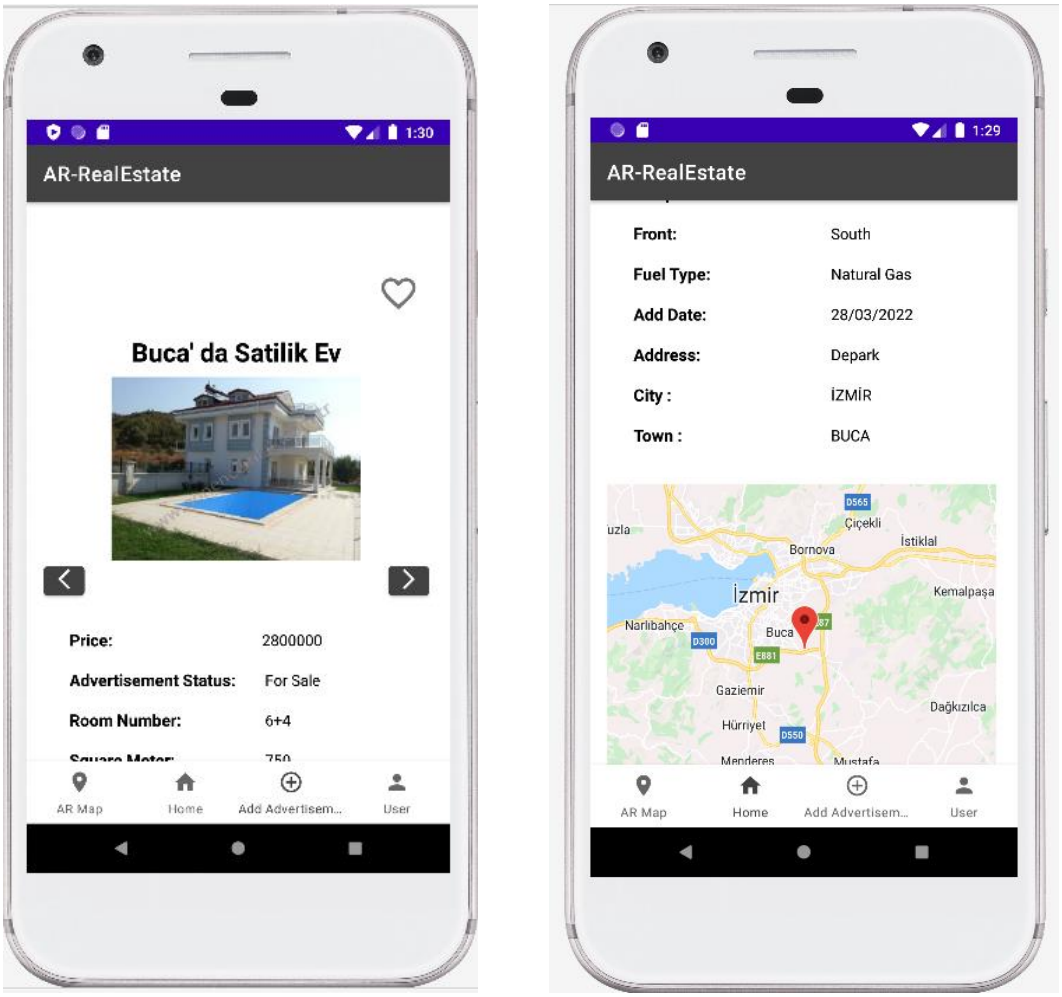


Figure 5.1.2.5.1 Application Advertisement Detail Page

The screen that the user will see immediately after adding an advertisement or when he clicks on the advertisement that he wants to see the details of is a screen like Figure 5.1.2.5.1. All information about the advertisement is listed, the photos of the advertisement can be viewed by using the arrow keys. If the user wants to add this advertisement to his favourites, he clicks on the hollow favorite icon in the upper right corner of the page. If the user has added this advertisement to their favorites before, the favorite icon will be filled.

5.1.2.6. Filter Advertisement Page

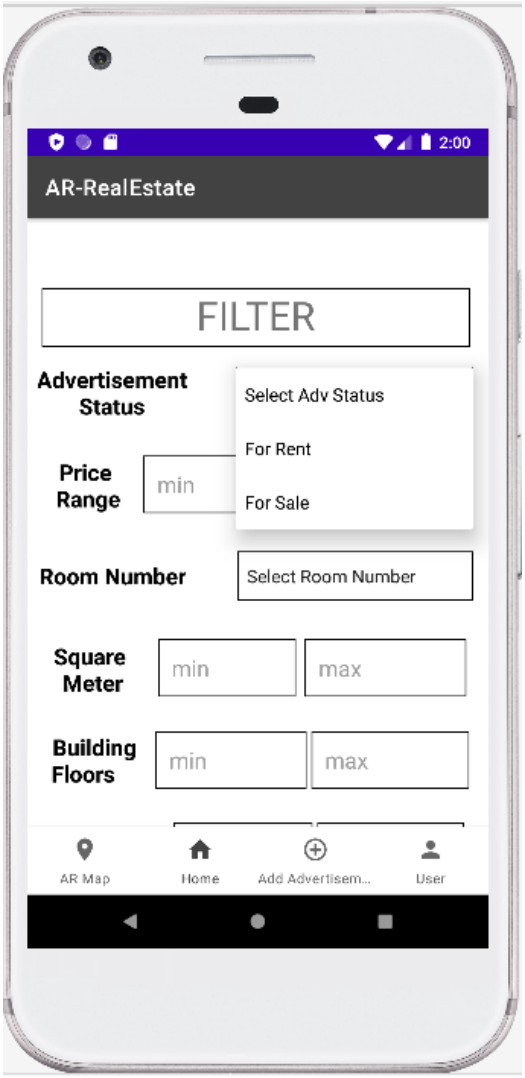


Figure 5.1.2.6.1 Application Filter Advertisement Page

When the "Filter" button at the top of the homepage is clicked, it is directed to the screen in Figure 5.1.2.6.1. The user fills in the input boxes according to their wishes in order to be able to view the advertisements with the desired feature and sees the desired advertisements by making a selection. If the user enters the input boxes blank or incorrectly, an error message is displayed.

5.1.2.7. My Account Page

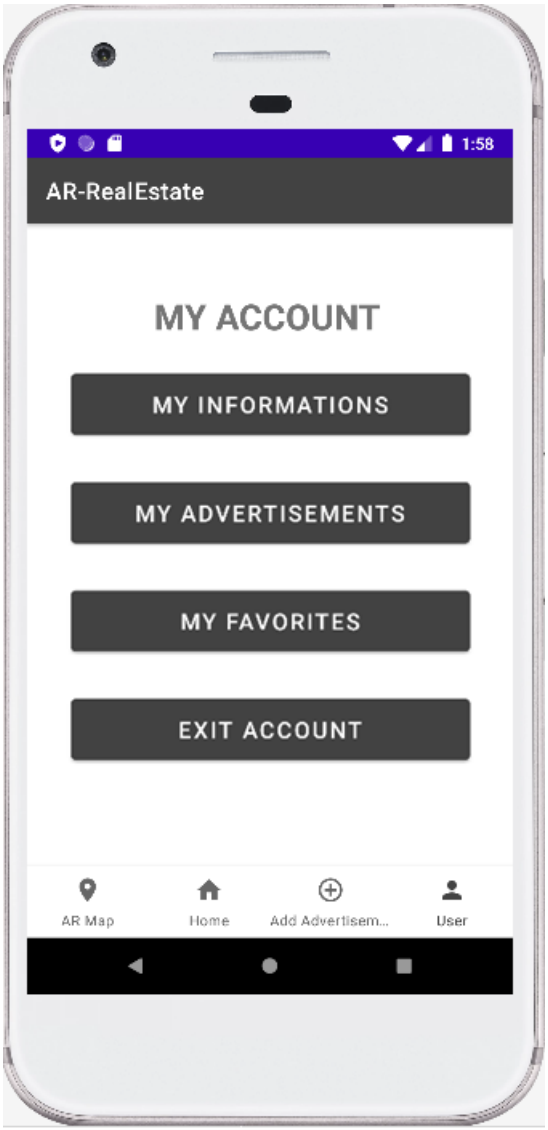


Figure 5.1.2.7.1 Application My Account Page

If the user has logged into the application and clicked on the "User" button on the navigation bar on the homepage, they are directed to the screen in Figure 5.1.2.7.1. On this screen, there is a "My Information" button that leads to the page where the user's information is displayed and updated, a "My Advertisements" button that leads to the viewing page of the advertisements if he has added an advertisement, a "My Favorites" button to view the advertisements he has added to his favorites, and an "Exit Account" button if he wants to leave his account.

5.1.2.8. My Information Page

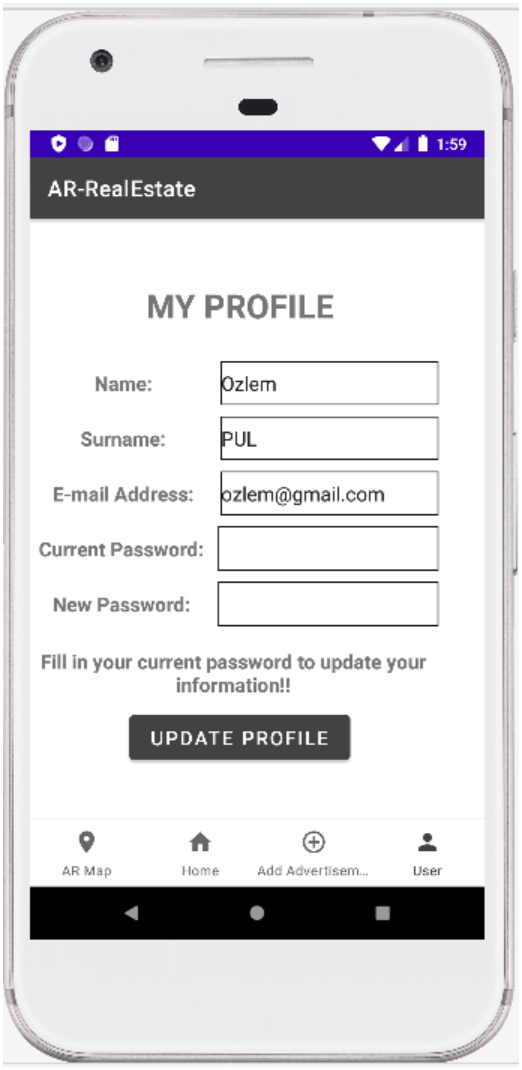


Figure 5.1.2.8.1 Application My Information Page

If the user wants to view or update their profile information, they will see a screen like Figure 5.1.2.8.1. The user who wants to update his information must also enter his current password.

5.1.2.9. My Advertisements Page

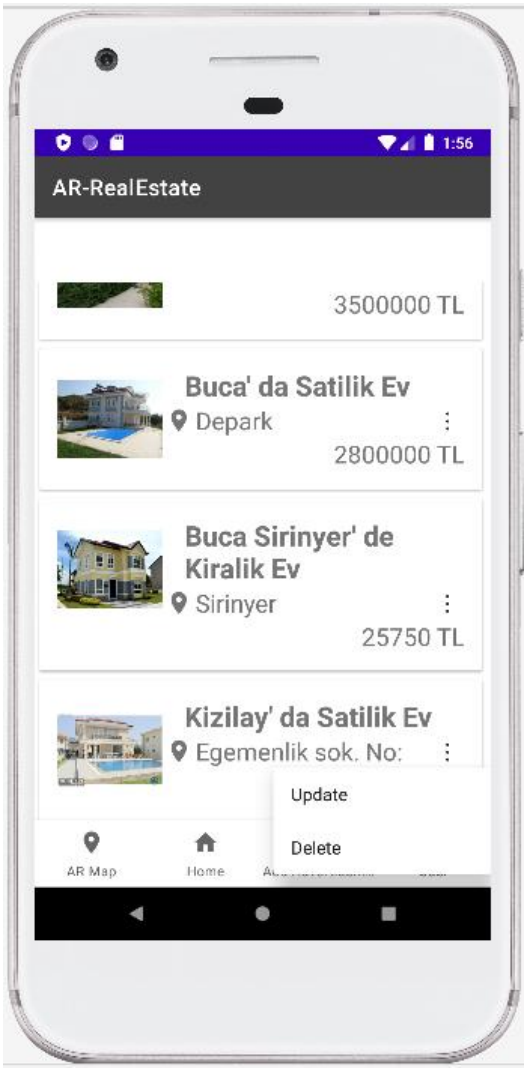


Figure 5.1.2.9.1 Application My Advertisements Page

If the user is logged into the application and wants to view, delete or update the advertisements he has added, the screen in Figure 5.1.2.9.1 appears. When you click on any of the advertisements you have added, you can view the details of that ad. When you click on the three dots on the right of each ad, "update" and "delete" options appear. When the Update option is clicked, that posting is directed to the update screen, if the delete option is clicked, that posting is deleted.,

5.1.2.10. Update Advertisements Page

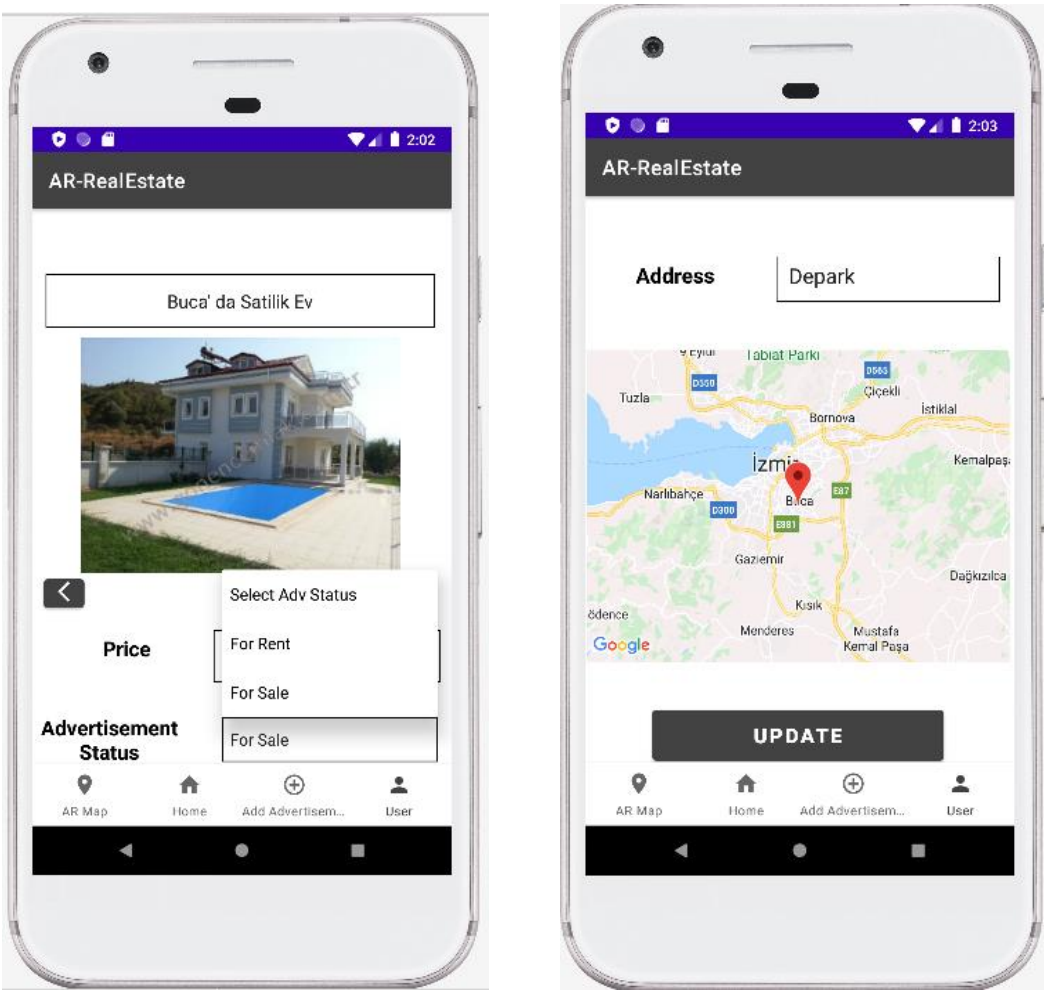


Figure 5.1.2.10.1 Application Update Advertisement Page

If the user wants to update from the advertisements he has added, he is directed to the page in Figure 5.1.2.10.1 with the update option at the three points on the right of the "My Advertisement" page. On this page, the current features of the advertisement are displayed, and he can change the feature he wants to change from these features. By clicking the Update button, the information of the announcement is updated with new information. If there is a feature that is not selected or entered on this page, or if the picture is blank, an error message is given to the user.

5.1.2.11. My Favorites Page

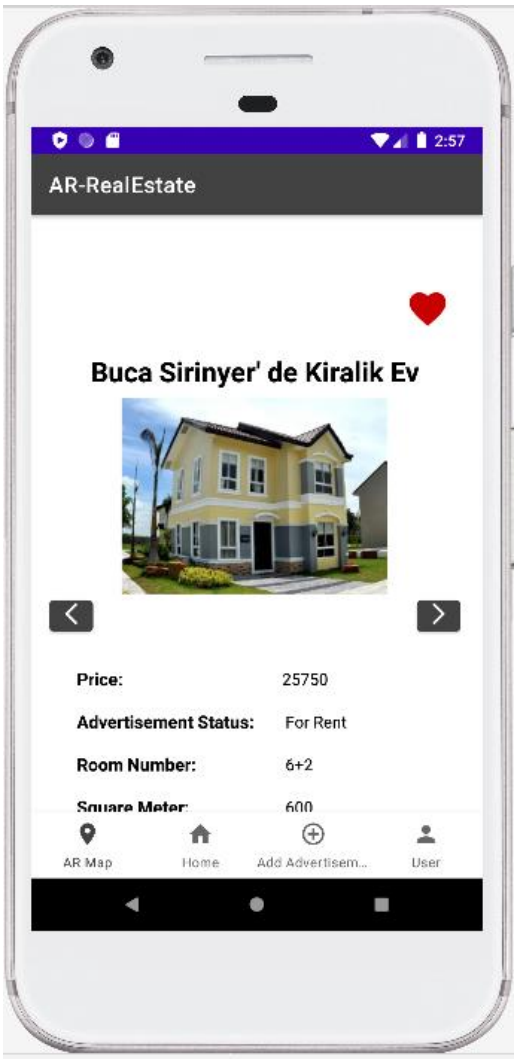


Figure 5.1.2.11.1 Application Add Advertisement Favorites List

If the user wants to add the ad, of which the detail is viewed, to his favorites, on the page where the user is viewing the advertisement details, he/she will add it to his/her favorite advertisements by clicking the hollow favorite icon in the upper right corner of the page in Figure 5.1.2.11.1. If the favorite icon is filled, it has already been added to your favorites. If he wants to remove it from his favorites, he clicks on the filled icon and removes it from his favorites.

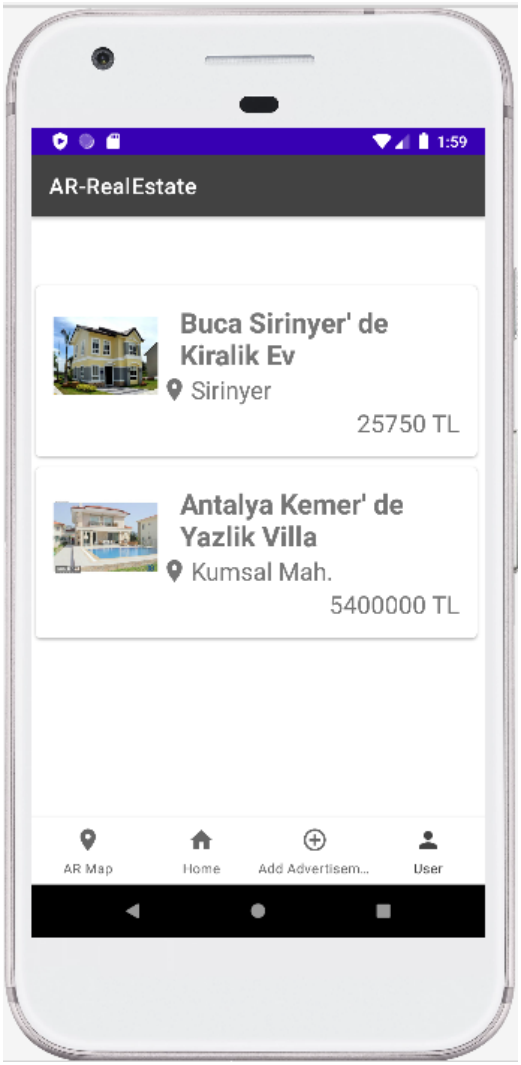


Figure 5.1.2.11.2 Application My Favorites Advertisement List Page

When the user wants to view the list of his favorite ads, he can see it on the screen in Figure 5.1.2.11.2. You can view the details of the advertisement when you click on it, and the advertisements you remove from your favorites cannot be viewed on this screen because they are removed from your favorites.

5.1.2.12. AR Unity Page

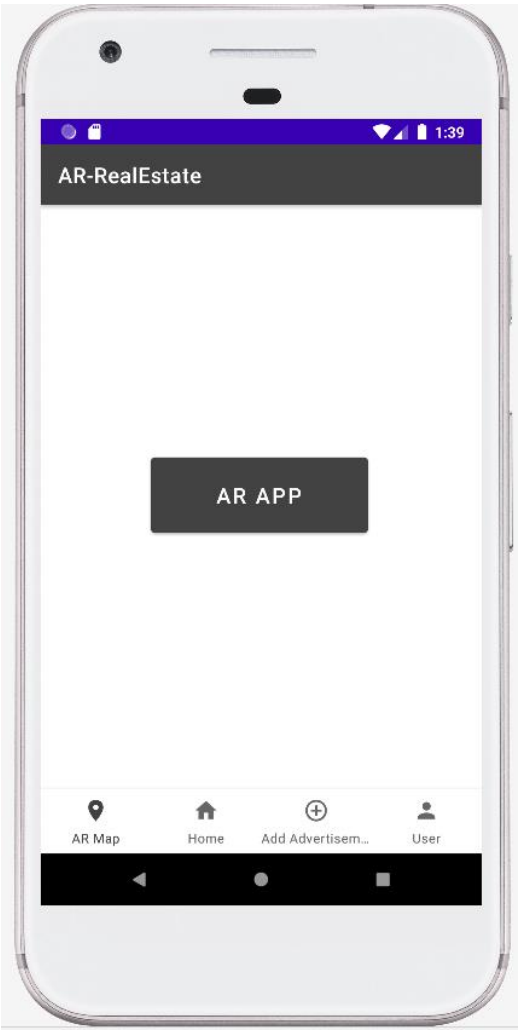


Figure 5.1.1.12.1 Direction AR Application

In the application, this page comes up when the "AR Map" button in the navigation bar is clicked. When the "AR APP" button is pressed, the application directs us to the Unity application in Figure 5.1.1.12.1.

5.2. Database

5.2.1. SQLite Database

There are "Advertisements", "Advertisement Image", "User Information", "Favorite", "Cities", "District" and "UserAdvertisement" tables in the database we use.

In the Advertisements table, the id information of the advertisement is kept as the primary key autoincrement. advertisement title, advertisement status, number of rooms, buildtype, item status, warm type, credit eligibility status, using status, state building, swap, front, fuel type, date, address, city, town information are kept in the table in text type. Price, square meter, building floors, floor, building age, number of bathrooms, rental income, dues information are kept in the table in integer type. The picture of the advertisement is kept in blob type and the x-y coordinates are kept in real type in the table. When a new advertisement is added, it is recorded in this table, when the advertisement is deleted, it is deleted from this table, and when the advertisement is updated, it is updated in this table.

The pictures of the advertisements are kept in the AdvertisementImage table. Image id information is kept as integer type as primary key auto increment and advertisement id information is taken from Advertisements table as foreign key. Each of the pictures of the advertisements is kept in a blob-type table. When the advertisement is added, all the pictures of the advertisement are recorded in this table.

The UserInformation table keeps the information of the users who registered in the application. User id information is kept in integer type as primary key auto increment. The user's name, surname, e-mail address and password are kept in this table in text type. When user information is updated, it is updated in this table.

In the Favorite table, the advertisements that the user has favorited are kept. The user id information is taken from the "UserInformation" table as a foreign key, and

the advertisement id information is taken from the "Advertisements" table. If the user wants to post advertisements from favorites, their status is changed.

While adding or updating an advertisement from the Cities table, the names of the cities in Turkey are kept in order to select the city information of the advertisement. While adding or updating the advertisement from the District table, the names of the districts in Turkey are kept in order to select the district information of the cities in the advertisement.

If the user who entered the UserAdvertisement table has added an advertisement, the id information of the advertisement and the user id information are assigned as foreign key. Advertisements added by the user are kept in this table.

5.2.2. Server Database

Database created in MSSQL Server. This database was used while adding an ad. It is a common database for Android and Unity.

In the Advertisements table, the id information of the advertisement is kept as the primary key autoincrement. advertisement title, advertisement status, number of rooms, buildtype, item status, warm type, credit eligibility status, using status, state building, swap, front, fuel type, date, address, city, town information are kept in the table in text type. Price, square meter, building floors, floor, building age, number of bathrooms, rental income, dues information are kept in the table in integer type. The picture of the advertisement is kept in blob type and the x-y coordinates are kept in real type in the table. When a new advertisement is added.

5.3. Unity AR Application

A start was made by creating an Augmented Reality project on the Unity platform. "AR+GPS Location" package was imported in order to use coordinate location information in augmented reality. Augmented reality related packages were also imported into the application. An object has been created to extract the logo according to the coordinate information. A script for obtaining and processing coordinate information has been integrated into this object. The logo that will be displayed temporarily has also been integrated into this object. For now, a static xml type file has been created with a certain number of x-y coordinates and integrated into this object. Android environment is selected in Build settings. Then "Vulkan" was removed in the player settings screen, "AR Core" was selected and the minimum "API Level" to work was selected and built. Afterwards, the Unity AR project we created was run by transferring the built file apk to a real android mobile phone.

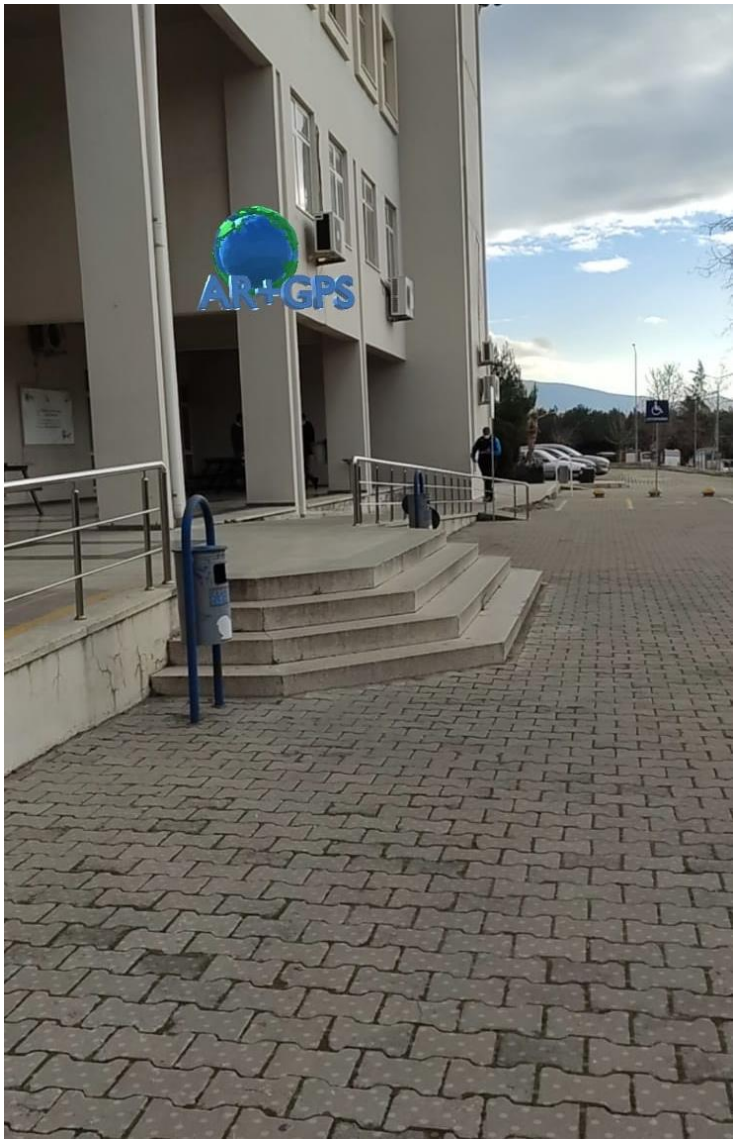


Figure 5.3.1 AR Application AR+GPS Logo

The file containing the x-y coordinates was prepared by entering the coordinate information of the main gate of the Dokuz Eylül University Computer Engineering Department into the data xml file. The project built using this file was run on a mobile android phone. By scanning around with the phone camera, locations at a certain distance are detected, the "AR+GPS" logo, which we have determined at the location we have put, appears on the phone screen as in Figure 5.3.1.

5.3.1. Create Dynamic Object in Unity AR Application

First of all, dynamic objects were created in Unity with the C# programming language in order to show the details of the advertisements where the advertisements are located. As in Figure 5.3.1.1, with the getAdvertisement function on the server, the information of all advertisements is brought in XML format. This XML is parsed and thrown into the array.

```
Unity İletisi | 0 başvuru
IEnumerator Start()
{
    WWW www = new WWW("http://193.140.150.95/deuarSrv/WebService1.asmx/getAdvertisements");
    yield return www;
    string allData = www.text;

    allData = allData.Substring(allData.IndexOf("<Advertisements>") + 16).Replace("\t", "").Replace("\n", "").Replace("\r", "").Replace("    ", " ").Replace("  ", " ").
    allData = allData.Replace("</ArrayOfAdvertisements>", "").Replace("</Advertisements>", "").Replace("</YCoordinate>", "").Replace("</XCoordinate>", "").Replace("</D

    String[] spearator = { "<Advertisements>" };
    String[] spearatorX = { "<Advertisements>", "<YCoordinate>", "<XCoordinate>", "<Date>", "<Town>", "<Cities>", "<Address>", "<FuelType>", "<Front>", "<Swap>", "<Sta

    string[] scenes = allData.Split(spearator, StringSplitOptions.RemoveEmptyEntries);
    string[] scenesX = allData.Split(spearatorX, StringSplitOptions.RemoveEmptyEntries);

    advertisementList = new Advertisements[scenes.Length];
    int index = 0;
    for (int i = 0; i < advertisementList.Length; i++)
    {
        advertisementList[i] = new Advertisements();
        advertisementList[i].AdvId = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].Price = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].SquareMeter = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].BuildingFloors = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].FloorLoc = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].BuildAge = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].NumOfBathrooms = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].RentalIncome = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].Dues = Convert.ToInt32(scenesX[index++]);
        advertisementList[i].AdvTitle = scenesX[index++];
        advertisementList[i].AdvStatus = scenesX[index++];
        advertisementList[i].RoomNum = scenesX[index++];
        advertisementList[i].BuildType = scenesX[index++];
        advertisementList[i].ItemStatus = scenesX[index++];
        advertisementList[i].WarmType = scenesX[index++];
```

Figure 5.3.1.1 Call Get Advertisement Function in Server

By creating a text object, the text property of this object and the advertisement information are assigned to the text and displayed on the camera in Figure 5.3.1.2.

```
string forLat = scenesX[index++].ToString();
string forLong = scenesX[index++].ToString();
double outLat;
double outLong;
double.TryParse(forLat, NumberStyles.Any, CultureInfo.InvariantCulture, out outLat);
double.TryParse(forLong, NumberStyles.Any, CultureInfo.InvariantCulture, out outLong);

advertisementList[i].XCoordinate = outLat;
advertisementList[i].YCoordinate = outLong;

GameObject textGameObject = new GameObject();
textGameObject.name = "Text" + advertisementList[i].AdvId.ToString();
TextMesh textMeshObject = textGameObject.AddComponent<TextMesh>();

textMeshObject.text = advertisementList[i].AdvTitle.ToString() + "\n" + advertisementList[i].AdvStatus + "\n"
    + advertisementList[i].Price + "\n" + advertisementList[i].SquareMeter + "\n" + advertisementList[i].BuildingFloors + "\n" +
    advertisementList[i].FloorLoc + "\n" + advertisementList[i].BuildAge + "\n" + advertisementList[i].NumOfBathrooms + "\n" +
    advertisementList[i].RoomNum + "\n" + advertisementList[i].RentalIncome;

textMeshObject.color = Color.white;
textMeshObject.fontSize = 12;
```

Figure 5.3.1.2 Create Text Object

The latitude, longitude, height, elevationMode values, which are the location information of these objects, are dynamically created by taking from the database. The values of the constant parameters that the PlaceAtOptions() method in the PlacaAtLocation script should take are given. The AddPlaceAtComponent function in PlaceAtLocation has location information, PlaceAtOptions and variables of the created game object as parameters, and an object with location information is created in Figure 5.3.1.3. To store this object in Unity, a previously created object is assigned to its parent. In this way, the object created in Unity became visible.

```
    var locText = new Location
    {
        Latitude = outLat,
        Longitude = outLong,
        Altitude = 0,
        AltitudeMode = AltitudeMode.GroundRelative
    };

    var optsText = new PlaceAtLocation.PlaceAtOptions()
    {
        HideObjectUntilItIsPlaced = true,
        MaxNumberOfLocationUpdates = 2,
        MovementSmoothing = 0.05f,
        UseMovingAverage = false
    };

    var getParent = GameObject.Find("GameObjectParent");
    var instanceTextObject = PlaceAtLocation.AddPlaceAtComponent(textGameObject, locText, optsText);
    textGameObject.transform.parent = getParent.transform;
    instanceTextObject.Location = locText;
    textGameObject.transform.position = new Vector3(0, 0, 0);
}
```

Figure 5.3.1.3 Text Object Place at Location

5.3.1.1. AR Application View

Figure 5.3.1.1.1 One Advertisement Display on Camera

It looks like in figure 5.3.1.1.1, close to the user and at the camera view. As the user approaches the advertisement, the text containing the advertisement information does not change, it remains in the marked position. If the camera is turned in the other direction, the ad will not be visible if it comes out from the camera view. The user again points the camera towards the place where the ad is located.

Figure 5.3.1.1.2 Display of Multiple Advertisements on Camera

As in Figure 5.3.1.1.2, if there is more than one advertisement in the camera view, the text of the one closest to the user appears larger, the farthest one is smaller. If the advertisements at a certain distance from the user are in the camera view, they appear on the screen. Thus, more than one advertisement can appear on the screen at the same time.

5.3.2. Database Connection in Unity AR Application

```
[DataContract]
6 references
public class Advertisements
{
    int advId;
    string advTitle;
    System.Byte[] advImage;
    System.Data.Linq.Binary advImageBinary;
    int price;
    string advStatus;
    string roomNum;
    int squareMeter;
    int buildingFloors;
    int floorLoc;
    int buildAge;
    string buildType;
    string itemStatus;
    string warmType;
    int numOfBathrooms;
    string elgCredit;
    string usingStatus;
    string stateBuilding;
    int rentalIncome;
    int dues;
    string swap;
    string front;
    string fuelType;
    DateTime date;
    string address;
    string cities;
    string town;
    double xCoordinate;
    double yCoordinate;
```

Figure 5.3.2.1 Table Attributes

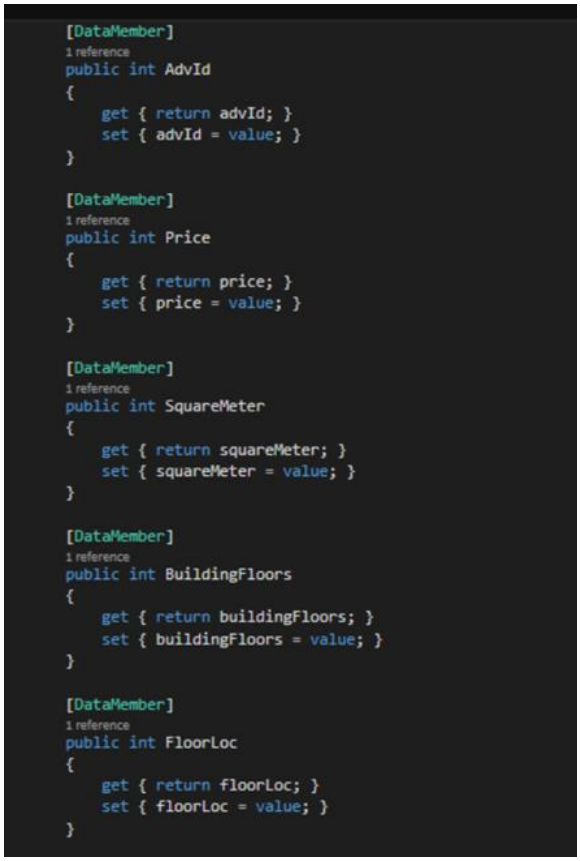
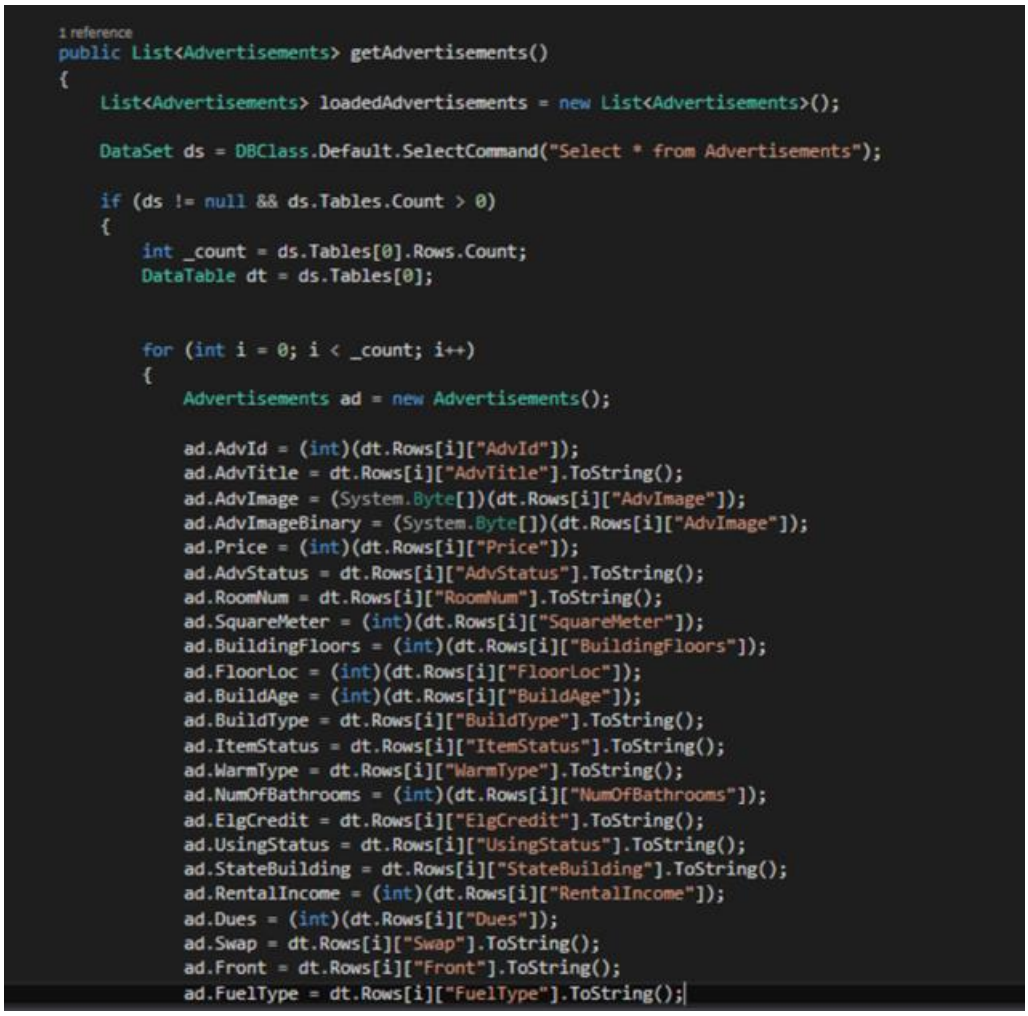


Figure 5.3.2.2 Get and Set For Table Attributes

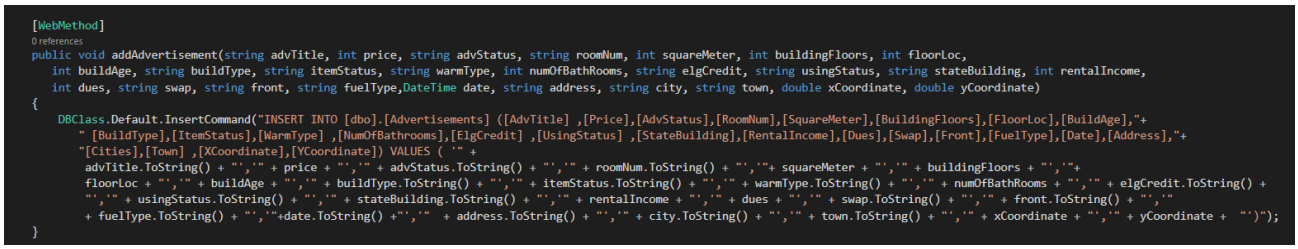
Instead of connecting directly to the database where we want the advertisements to be found, a web service was written as a layer in between. This is because it is not safe to connect directly to the database. Public classes have been created for each table in the database. The attributes of these classes as in Figure 5.3.2.1 are the columns in the tables. For each of these columns, get and set methods are added as in Figure 5.3.2.2 The get method is to fetch the value in the column, and the set method is to update this value.



```
1 reference
public List<Advertisements> getAdvertisements()
{
    List<Advertisements> loadedAdvertisements = new List<Advertisements>();

    DataSet ds = DBClass.Default.SelectCommand("Select * from Advertisements");

    if (ds != null && ds.Tables.Count > 0)
    {
        int _count = ds.Tables[0].Rows.Count;
        DataTable dt = ds.Tables[0];


        for (int i = 0; i < _count; i++)
        {
            Advertisements ad = new Advertisements();

            ad.AdvId = (int)(dt.Rows[i]["AdvId"]);
            ad.AdvTitle = dt.Rows[i]["AdvTitle"].ToString();
            ad.AdvImage = (System.Byte[])(dt.Rows[i]["AdvImage"]);
            ad.AdvImageBinary = (System.Byte[])(dt.Rows[i]["AdvImage"]);
            ad.Price = (int)(dt.Rows[i]["Price"]);
            ad.AdvStatus = dt.Rows[i]["AdvStatus"].ToString();
            ad.RoomNum = dt.Rows[i]["RoomNum"].ToString();
            ad.SquareMeter = (int)(dt.Rows[i]["SquareMeter"]);
            ad.BuildingFloors = (int)(dt.Rows[i]["BuildingFloors"]);
            ad.FloorLoc = (int)(dt.Rows[i]["FloorLoc"]);
            ad.BuildAge = (int)(dt.Rows[i]["BuildAge"]);
            ad.BuildType = dt.Rows[i]["BuildType"].ToString();
            ad.ItemStatus = dt.Rows[i]["ItemStatus"].ToString();
            ad.WarmType = dt.Rows[i]["WarmType"].ToString();
            ad.NumOfBathrooms = (int)(dt.Rows[i]["NumOfBathrooms"]);
            ad.ElgCredit = dt.Rows[i]["ElgCredit"].ToString();
            ad.UsingStatus = dt.Rows[i]["UsingStatus"].ToString();
            ad.StateBuilding = dt.Rows[i]["StateBuilding"].ToString();
            ad.RentalIncome = (int)(dt.Rows[i]["RentalIncome"]);
            ad.Dues = (int)(dt.Rows[i]["Dues"]);
            ad.Swap = dt.Rows[i]["Swap"].ToString();
            ad.Front = dt.Rows[i]["Front"].ToString();
            ad.FuelType = dt.Rows[i]["FuelType"].ToString();
```

Figure 5.3.2.3 Get Advertisement Data in Server Database

The details of the advertisements are kept in the "advertisements" table in the database. The data in this table has been put into the dataset object with the select query on the web service side. A for loop is created for the length of this dataset. As in Figure 5.3.2.3, an object of "Advertisements" type was created and each advertisement was added to this object with its properties, and then a list of advertisement type was added to this object. The function that creates this list is called in the script where the objects on the Unity side are created and transferred to the relevant array. As a result, text objects are created dynamically with the data pulled from the database.



```
[WebMethod]
0 references
public void addAdvertisement(string advTitle, int price, string advStatus, string roomNum, int squareMeter, int buildingFloors, int floorLoc,
   int buildAge, string buildType, string itemStatus, string warmType, int numOfBathRooms, string elgCredit, string usingStatus, string stateBuilding, int rentalIncome,
   int dues, string swap, string front, string fuelType,DateTime date, string address, string city, string town, double xCoordinate, double yCoordinate)
{
    DBClass.Default.InsertCommand("INSERT INTO [dbo].[Advertisements] ([AdvTitle] ,[Price],[AdvStatus],[RoomNum],[SquareMeter],[BuildingFloors],[FloorLoc],[BuildAge],"+
       " [BuildType],[ItemStatus],[WarmType] ,[NumOfBathrooms],[ElgCredit] ,[UsingStatus] ,[StateBuilding],[RentalIncome],[Dues],[Swap],[Front],[FuelType],[Date],[Address],"+
       "[Cities],[Town] ,[XCoordinate],[YCoordinate]) VALUES ( '" +
        advTitle.ToString() + "','" + price + "','" + advStatus.ToString() + "','" + roomNum.ToString() + "','"+ squareMeter + "','" + buildingFloors + "','"+
        floorLoc + "','" + buildAge + "','" + buildType.ToString() + "','" + itemStatus.ToString() + "','" + warmType.ToString() + "','" + numOfBathRooms + "','" + elgCredit.ToString() +
        "','" + usingStatus.ToString() + "','" + stateBuilding.ToString() + "','" + rentalIncome + "','" + dues + "','" + swap.ToString() + "','" + front.ToString() + "','"
        + fuelType.ToString() + "','"+date.ToString() +"','"  + address.ToString() + "','" + city.ToString() + "','" + town.ToString() + "','" + xCoordinate + "','" + yCoordinate +  "')");
}
```

Figure 5.3.2.4 Add Advertisement in Server Database

Posts added to the SQLite database are added to the database on the server using the Figure 5.3.2.4 function.

CHAPTER SIX

TEST/EXPERIMENTS

Tablo 6.1 Android Studio Project Test Cases

It is checked that the advertisements added to the database are listed dynamically on the home page.	✓
It is checked that the advertisement cards listed on the home page are clickable.	✓
When one of the advertisement cards listed on the home page is clicked on, it is checked that it is directed to the page where the details of that advertisement are shown.	✓
When the filter button is clicked from the buttons at the top of the home page, it is checked that it is directed to the advertisement filtering page.	✓
When the sort button is clicked on the top of the home page, it is checked that the options that increase according to price / decrease according to price are seen.	✓
Clicking on the sort button at the top of the home page and selecting ascending according to price, it is checked that the advertisements are listed in ascending order according to their prices.	✓
Clicking the sort button from the buttons at the top of the home page and selecting descending according to price, it is checked that the advertisements are listed in descending order according to their prices.	✓
When you click on the "AR Map" button in the Navigation Bar, it is checked that it directs it to the Unity application.	✓
When you click on the "Home" button in the Navigation Bar, it is checked that it is directed to the home page.	✓
When you click on the "Add Advertisement" button in the Navigation Bar, it is checked that if it is a logged in user, it is directed to the add advertisement page.	✓

When you click on the "Add Advertisement" button in the Navigation Bar, it is checked that if the user is not logged in, they are directed to the login page.	✓
When you click on the "User" button in the Navigation Bar, it is checked that if the user is not logged in, they are directed to the login page.	✓
When you click on the "User" button in the Navigation Bar, it is checked that if it is a logged in user, it is directed to the user profile page.	✓
If the user wants to register to the application when he/she comes to the login screen in the application, when he/she clicks on the "Sign Up" button on the login screen, it is checked that he/she is directed to the registration page.	✓
If the user is registered to the application when he comes to the login screen in the application, if he has written the "email" and "password" fields correctly, it is checked that he is directed to the home page when he clicks the "Login" button on the screen.	✓
When the user comes to the login screen in the application, if he is registered in the application, if he wrote the "email" and "password" fields incorrectly, it is checked that the error message is displayed when he clicks the "Login" button on the screen.	✓
When the user fills in the input fields on the registration screen in the correct format and clicks on the "Sign Up" button on the screen, if the registration is successful, it is checked that he is directed to the login page and the database is reflected.	✓
When the user fills in the input fields on the registration screen in the correct format and clicks on the "Sign Up" button on the screen, it is checked that an error message is displayed if the registration is not successful.	✓
In the advertisement insertion screen, it is checked that one or more images can be selected from the gallery in the image area.	✓

In the advertisement insertion screen, it is checked that the selected pictures are reflected in the picture area.	✓
When the right and left direction buttons are clicked on the advertisement posting screen, it is checked that the selected pictures are seen.	✓
In the advertisement insertion screen, it is checked that the input fields are entered correctly.	✓
When the "Add Advertisement" button is clicked after the input fields are entered correctly on the advertisement add screen, it is checked that the added advertisement is directed to the page where the detail is displayed.	✓
If there is an empty or incorrect input field on the advertisement add screen and the "Add Advertisement" button is clicked, it is checked that the error message is displayed.	✓
If adding an advertisement is successful, when the "Add Advertisement" button is clicked, it is checked that the information and pictures of the advertisement have been added to the database.	✓
It is checked that the information of the relevant advertisement is reflected correctly on the advertisement details screen.	✓
It is checked that the pictures of the relevant advertisement are reflected on the advertisement details screen and that other pictures of the advertisement are displayed with the direction buttons.	✓
If the advertisement is not favored by that user on the advertisement details screen, it is checked that the favorite icon is blank.	✓
If the advertisement is favorited by that user on the advertisement details screen, it is checked that the favorite icon appears full.	✓
When the logged in user comes to the "User" screen, it is checked that he/she is directed to the relevant pages when the "My Profile", "My Advertisements", "My Favorites", "Exit" buttons are clicked.	✓
On the My profile screen, it is checked that the current information of the user is reflected and that these fields are editable.	✓

On the My advertisement screen, if the user has added an advertisement, it is checked that the advertisements he/she has added are listed.	✓
In the My advertisement screen, if the user has added an advertisement, it is checked that the "three dots" icon is seen on the advertisement card of the advertisements and when clicked, "update" and "delete" options appear.	✓
On the user advertisement update screen, it is checked that the current features of the relevant advertisement are reflected in the input fields and that these fields are editable.	✓
When the user clicks on the "Advertisement Update" button on the advertisement update screen, it is checked that the advertisement information is updated in the database.	✓
If the user has pressed the "delete" option on the advertisement card for the relevant advertisement, it is checked that that advertisement has been deleted from the database.	✓
If the user is on the "My Favorites" screen, it is checked that the advertisements they have favorited are listed.	✓
If the user has removed a favorite advertisement from his favorites, it is checked that it has been removed from the database from the favorites table.	✓

Tablo 6.2 Unity Project Test Cases

On the Unity side, it is checked that the GameObject can be created without a script and its x and y coordinates can be given.	✓
When a GameObject is created without a script in Unity and its x and y coordinates can be given, it is checked that the object is seen on the phone camera in those coordinates.	✓
When more than one GameObject is created without a script in Unity and the x and y coordinates are given, it is checked that the objects in those coordinates do not overlap on the phone camera.	✓

When a TextMesh is created without a script in Unity and the x and y coordinates are given, it is checked that the textmesh is seen on the phone camera in those coordinates.	✓
It is checked that the text information of the textmesh created without a script by Unity is reflected correctly on the phone camera.	✓
When more than one TextMesh is created without script in Unity and given x and y coordinates, it is checked that the textmeshes in these coordinates do not overlap on the phone camera.	✓
It is checked that the gameobject created in the script in Unity is created under its parent when it runs on unity.	✓
When a GameObject is created with a script in Unity and its x and y coordinates can be given, it is checked that the object is seen on the Unity in those coordinates.	✓
When a GameObject is created with a script in Unity and its x and y coordinates can be given, it is checked that the object is seen on the phone camera in those coordinates.	✓
When more than one GameObject is created with script in Unity and given x and y coordinates, it is checked that the GameObject in these coordinates do not overlap on the phone camera.	✓
When a TextMesh is created with a script in Unity and its x and y coordinates can be given, it is checked that the object is seen on the Unity in those coordinates.	✓
When a TextMesh is created with a script in Unity and its x and y coordinates can be given, it is checked that the object is seen on the phone camera in those coordinates.	✓
When more than one TextMesh is created with script in Unity and given x and y coordinates, it is checked that the TextMesh in these coordinates do not overlap on the phone camera.	✓
In Unity, it is checked that it can connect to the previously created database in the script.	✓
It is checked that the objects are created dynamically when they are run in Unity with the information received from the database in Unity.	✓

In Unity, it is checked that the objects are created dynamically when run on the phone with the information obtained from the database.	✓
In Unity, it is checked that more than one textmesh is seen on the phone camera at the desired locations by looping without a database connection in the script.	✓
Using the database connection in the script in Unity, it is checked that more than one textmesh is seen by looping on the phone camera at desired locations.	✓
It is checked from the database that the information of each advertisement is correct.	✓
According to the location information of the advertisements in the database, it is checked that the detail of the advertisement is displayed at the location of that advertisement on the phone screen.	✓

CHAPTER SEVEN

CONCLUSION

Advertisement added to the AR-RealEstate android mobile application can be seen on the camera screen of the phone using AR technology at the specified location. An android mobile application was developed over Android Studio for adding advertisements and detailed operations regarding advertisements. AR application was developed in Unity environment in order to display advertisements on the camera according to their location. When the user opens the mobile application, he sees the advertisements added to the home page as listed, and only logged in users can add advertisements and favorite the ad. When he clicks on an ad, he can view the ad's information, images, and location. You can add the advertisement to your favorites, and remove it from your favorites, and view the adverts that it likes on a page. The user can update and delete the advertisements that he/she added. It can be accessed from the Android application, and the AR application can also be operated independently of the Android application by the user.

For the AR application, objects such as cubes and cylinders were created in certain locations in the script, and it was tested with the phone camera whether the location information was detected correctly or not. Later, instead of these objects, text type objects were created and tested in this way. Finally, the advertisement information and location information added to the android application are received by the AR application, thanks to the database connection. Thus, dynamic text objects containing the details of the advertisements were created for the advertisements saved in the database. According to the distance between the phone and the advertisement position, the advertisements are reflected on the screen at the camera level. The database on the server, which is a common database, was used to use the posting information added in the Android application by the Unity application. Only the information to be used by the android mobile application was kept in the SQLite database. Thus, the slowness of the process was prevented by connecting to the common database on the server every time.

REFERENCES

Andri, C., Alkawaz, M. H., & Sallow, A. B. (2018). Adoption of mobile augmented reality as a campus tour application. Int. J. Eng. Technol, 7(4.11), 64.

Asraf, S. M. H., Hashim, A. F. M., & Idrus, S. Z. S. (2020, April). Mobile Application Outdoor Navigation Using Location-Based Augmented Reality (AR). In Journal of Physics: Conference Series (Vol. 1529, No. 2, p. 022098). IOP Publishing.

Aydinoglu, A. C., & Bovkir, R. (2020). Developing a Mobile Application for Smart Real Estate Information. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 44, 89-94.

Bao, L. Q., LV, L. X., & Li, J. L. (2017). Optimizing Naïve Bayes Algorithm for SMS Spam Filtering on Mobile Phone to Reduce the Consumption of Resources. Journal of Computers, 28(3), 174-183.

Bhorkar, G. (2017). A survey of augmented reality navigation. arXiv preprint arXiv:1708.05006.

Brata, K. C., & Liang, D. (2019). An effective approach to develop location-based augmented reality information support. International Journal of Electrical & Computer Engineering (2088-8708), 9(4).

Carrasco, B., Guamán, V., Delgado, J., & Ruiz, F. (2019, November). ARTOUR: augmented reality for tourism-a case study in Riobamba, Ecuador. In 2019 International Conference on Information Systems and Computer Science (INCISCOS) (pp. 116-123). IEEE.

Chung, C. O., He, Y., & Jung, H. K. (2016). Augmented reality navigation system on android. International Journal of Electrical and Computer Engineering, 6(1), 406.

Cui, B. B. (2017). Design and implementation of movie recommendation system based on Knn collaborative filtering algorithm. In ITM web of conferences (Vol. 12, p. 04008). EDP Sciences.

de Macedo, D. V., Rodrigues, M. A. F., Furtado, J. J., Furtado, E. S., & Chagas, D. A. (2014). Using and evaluating augmented reality for mobile data visualization in real estate classified ads. International Journal of Computers and Applications, 36(1), 7-14.

Jia, J., Elezovikj, S., Fan, H., Yang, S., Liu, J., Guo, W., ... & Ling, H. (2021). Semantic-aware label placement for augmented reality in street view. The Visual Computer, 37(7), 1805-1819.

Kurniawan, M. H., & Witjaksono, G. (2018). Human anatomy learning systems using augmented reality on mobile application. Procedia Computer Science, 135, 80-88.

Kose, M., Incel, O. D., & Ersoy, C. (2012, April). Online human activity recognition on smart phones. In Workshop on mobile sensing: from smartphones and wearables to big data (Vol. 16, No. 2012, pp. 11-15).

Lam, K. Y., Lee, L. H., & Hui, P. (2021, October). A2w: Context-aware recommendation system for mobile augmented reality web browser. In Proceedings of the 29th ACM International Conference on Multimedia (pp. 2447-2455).

López-Faican, L., & Jaen, J. (2020). EmoFindAR: Evaluation of a mobile multiplayer augmented reality game for primary school children. Computers & Education, 149, 103814.

Meyer-Lee, G., Shang, J., & Wu, J. (2018, November). Location-leaking through network traffic in mobile augmented reality applications. In 2018 IEEE 37th International Performance Computing and Communications Conference (IPCCC) (pp. 1-8). IEEE.

Repaka, A. N., Ravikanti, S. D., & Franklin, R. G. (2019, April). Design and implementing heart disease prediction using naives Bayesian. In 2019 3rd International conference on trends in electronics and informatics (ICOEI) (pp. 292-297). IEEE.

Sallow, A. B., & Hussain, S. R. (2020). Multi-Agent System for Supporting and Managing Real Estate Marketing. Academic Journal of Nawroz University, 9(3), 54-62.

Skorokhodov, D., Seliverstov, Y., Seliverstov, S., Burov, I., Vydrina, E., Podoprigora, N., ... & Cheremisina, A. (2018, November). Using augmented reality technology to improve the quality of transport services. In International Conference on Convergent Cognitive Information Technologies (pp. 339-348). Springer, Cham.