**INSTRUCTOR’S COMMENTS**

**REVIEWER’S COMMENTS**

**ABSTRACT**

Project name: Buiding an application for Home Furnishing using Augmented Reality

Student: Nguyen Thi Tinh

Student ID: 102130137

Class: 13T3

The aim of this project was to create an application which simplifies the choosing furniture in the house, room. The users can use mobile devices, using the camera of the phone to decorate their house, choose the stuff which is suitable for their house. After that, they can take that furniture they want without going to the furniture store. Any manipulation with 3D models is always through the camera. To finish decorate their house, they can take a picture or record a video and share them to the social network to get ideas from the others.

|  |  |
| --- | --- |
| THE UNIVERSITY OF DANANG **UNIVERSITY OF SCIENCE AND TECHNOLOGY**  INFORMATION TECHNOLOGY FACULTY | **SOCIALIST REPUBLIC OF VIET NAM**  Independence – Freedom – Happiness |

**GRADUATION PROJECT REQUIREMENTS**

Student Name: …..…………….………….…….. Student No. : ……………….....

Class:……………Department:...................................Major:………………............

1. *Name of project:*

………………………………………………..…………………………………….

……………………………………………………………………………………...

1. *This project’s results:* ☐ *Are protected by an intellectual property agreement*
2. *Initial data:*

……………………………………..……………………………………………….

...……………………………………………………………………………………

…..……………………………….…..………………………..……………………

1. *Project contents:*

…...…………………………………………………………………………………

…...…………………………………………………………………………………

…...…………………………………………………………………………………

…...…………………………………………………………………………………

…...…………………………………………………………………………………

1. *Charts and drawings (please specify the drawing type and size):*

…...…………………………………………………………………………………

…...…………………………………………………………………………………

…...…………………………………………………………………………………

…...…………………………………………………………………………………

*6. Supervisor(s):* …………………………………..……………………

1. *Date of assignment: ……../……./201…..*
2. *Date of completion: ……../……./201…..*

|  |  |
| --- | --- |
| *Da Nang, day month year 201* | |
| **Head of Division** …………………….. | **Supervisor(s)** |

**ACKNOWLEDGEMENTS**

I am highly indebted to our project mentors, Ms. Le Thi My Hanh for their continuous support, supervision, motivation, and guidance throughout the tenure of our project in spite of their hectic schedule. They remained a driving spirit in my project and their experience gave me the understanding in handling research projects as well as helping us to clarify the abstruse concepts, requiring knowledge and perception, handling critical situations and in understanding the objective of our work.

I also want to thank our families and friends, who gave us the strength and confidence during our time of learning and during the implementation phase of this project. They have given a lot of love and encouragement for us which helped pass over the difficulties and fatigues.

I am also grateful to the directions of the management board and engineers of Enclave ODC, Ltd who have facilitated our studying about technology, programming, workplace skill, enhance the best soft skills such as teamwork, attitude and especially English.

Without your generous help, our senior year would not have been successful.

Sincerely,

Nguyen Thi Tinh

# **GUARANTEE**

I guarantee:

1. The contents of this senior project are performed by ourselves following the guidance of lecturer Ms. Le Thi My Hanh.

2. All references used in this senior project thesis, are quoted with the author’s name, project name, time and location to publish clearly and faithfully.

3. All invalid copies, educated statute violation or cheating will be borne the full responsibility by myself.

Students,

Nguyen Thi Tinh

**TABLE OF CONTENTS**

[**INSTRUCTOR’S COMMENTS** 1](#_Toc514250156)

[**REVIEWER’S COMMENTS** 2](#_Toc514250157)

[**ABSTRACT** 3](#_Toc514250158)

[**ACKNOWLEDGEMENTS** i](#_Toc514250159)

[**GUARANTEE** ii](#_Toc514250160)

[**TABLE OF CONTENTS** iii](#_Toc514250161)

[**LIST OF TABLES AND FIGURES** v](#_Toc514250162)

[**LIST OF SYMBOLS AND ABBREVIATIONS** vi](#_Toc514250163)

[**Chapter 1: INTRODUCTION** 1](#_Toc514250164)

[**1.1.** **Background and Context** 1](#_Toc514250165)

[**1.2.** **Purpose** 2](#_Toc514250166)

[**1.3.** **Scope** 3](#_Toc514250167)

[**Chapter 2: THEORIES AND TECHNOLOGIES** 4](#_Toc514250168)

[**2.1. iPhone Operating System** 4](#_Toc514250169)

[2.1.1. Introduction 4](#_Toc514250170)

[2.1.2. iOS layered architecture 5](#_Toc514250171)

[2.1.3. Features 6](#_Toc514250172)

[**2.2.** **Swift programming language** 6](#_Toc514250173)

[2.2.1 Introduction 6](#_Toc514250174)

[2.2.2 Architectural overview 7](#_Toc514250175)

[2.2.3 Features 8](#_Toc514250176)

[**2.3.** **Augmented Reality** 9](#_Toc514250177)

[**2.4.** **ARKit Framework** 10](#_Toc514250178)

[**2.5.** **Realm Database** 12](#_Toc514250179)

[**2.6.** **Blender** 14](#_Toc514250180)

[**Chapter 3: ANALYSIS AND DESIGN** 15](#_Toc514250181)

[**3.1.** **Analysis** 15](#_Toc514250182)

[3.1.1. Detecting Plane 15](#_Toc514250183)

[3.1.2. Choose 3D models 16](#_Toc514250184)

[3.1.3. Manipulate with 3D models 16](#_Toc514250185)

[3.1.4. Take a picture and share 17](#_Toc514250186)

[3.1.5. Manage Utility mode 17](#_Toc514250187)

[3.1.6. Manage Gallery 18](#_Toc514250188)

[**3.2.** **Use-case diagram** 18](#_Toc514250189)

[3.2.1. Overall diagram 18](#_Toc514250190)

[3.2.2. Detect Plane 19](#_Toc514250191)

[3.2.3. Choose 3D models 20](#_Toc514250192)

[3.2.4. Manipulate with 3D models diagram 20](#_Toc514250193)

[3.2.5. Take a picture diagram 21](#_Toc514250194)

[3.2.6. Utility mode diagram 21](#_Toc514250195)

[3.2.7. Manage Gallery diagram 22](#_Toc514250196)

[**3.3.** **Sequence diagram** 23](#_Toc514250197)

[3.3.1 Manipulate with 3D models sequence diagram 23](#_Toc514250198)

[3.3.2 Manage Gallery sequence diagram 24](#_Toc514250199)

[**Chapter 4: IMPLEMENTATION RESULTS** 27](#_Toc514250200)

[**4.1.** **Detecting Plane** 27](#_Toc514250201)

[**4.2.** **Choose 3D models** 30](#_Toc514250202)

[**4.3.** **Manipulate with 3D models** 33](#_Toc514250203)

[**4.4.** **Take a picture and sharing** 36](#_Toc514250204)

[**4.5.** **Manage Utility mode** 39](#_Toc514250205)

[**4.6.** **Manage Gallery** 41](#_Toc514250206)

[**CONCLUSION AND FUTURE WORK** 44](#_Toc514250207)

[**1. Achieve results** 44](#_Toc514250208)

[**2.** **Future works** 45](#_Toc514250209)

[**REFERENCES** 1](#_Toc514250210)

[**APPENDICES** 2](#_Toc514250211)

# **LIST OF TABLES AND FIGURES**

[Figure 1.1 Background 1](#_Toc514251550)

[Figure 1.2 Layered architecture 5](#_Toc514251551)

[Figure 1.3 ARKit Framework 10](#_Toc514251552)

[Figure 1.4 Realm database 12](#_Toc514251553)

[Figure 1.3 Realm platform 13](#_Toc514251554)

[Figure 1.5 Blender 14](#_Toc514251555)

[Figure 3.1 System architecture 15](#_Toc514251556)

[Figure 3.2 Overall usecase diagram 19](#_Toc514251557)

[Figure 3.3 Detect plane usecase 19](#_Toc514251558)

[Figure 3.4 Choose 3D models usecase 20](#_Toc514251559)

[Figure 3.5 Manipulate 3D models usecase 20](#_Toc514251560)

[Figure 2.6 Detect plane usecase 21](#_Toc514251561)

[Figure 3.7 Utility mode usecase 22](#_Toc514251562)

[Figure 3.8 Manage gallery usecase 23](#_Toc514251563)

[Figure 3.9 Manipulate 3D models’s sequence diagram 24](#_Toc514251564)

[Figure 3.10 Add picture’s sequence diagram 25](#_Toc514251565)

[Figure 3.11 View gallery’s sequence diagram 25](#_Toc514251566)

[Figure 3.12 Delete 3D models’s sequence diagram 26](#_Toc514251567)

[Figure 4.1 Detecting plane 27](#_Toc514251568)

[Figure 4.2 Plane Node after having detected Plane 28](#_Toc514251569)

[Figure 4.3 Add (+) Button will be appeared 29](#_Toc514251570)

[Figure 4.4 Choose a category 30](#_Toc514251571)

[Figure 4.5 List furniture of the table 31](#_Toc514251572)

[Figure 4.6 List the others 32](#_Toc514251573)

[Figure 4.7 Add a 3D model on the screen 33](#_Toc514251574)

[Figure 4.8. Keep and drag to another position 34](#_Toc514251575)

[Figure 4.9 Rotate to see another side 35](#_Toc514251576)

[Figure 4.10 Scale smaller (left) and bigger (right) 36](#_Toc514251577)

[Figure 4.11 Take a picture 37](#_Toc514251578)

[Figure 4.12 The picture after taking the screen 38](#_Toc514251579)

[Figure 4.13 Click Share button 39](#_Toc514251580)

[Figure 4.14 Utility mode 40](#_Toc514251581)

[Figure 4.15 Record a video 41](#_Toc514251582)

[Figure 4.16 Gallery screen 42](#_Toc514251583)

[Figure 4.17 The detail of the picture 43](#_Toc514251584)

# **LIST OF SYMBOLS AND ABBREVIATIONS**

|  |  |  |
| --- | --- | --- |
| **No.** | **Items** | **Description** |
| 1 | AR | Augmented reality |
| 2 | UX/UI | User Experience/User Interface |

# **Chapter 1: INTRODUCTION**

* 1. **Background and Context**

Nowadays, the more technology is developed, the human life is more convenient. Once step into the era of technology, requirements for quality of life is increased. To enhance the image of your house, the interior decoration is one of elements which is extremely important.

Follow some recent research on the social network, with a new married couple, the first work they concerned is shopping stuff for their new house.

But with a busy life like today, measuring the dimension, calculating, choosing what kind of furniture is suitable for their house, where is the best place to put them, how are they look like in their house is not an easy job. After discussion, consult the others, they will go to the furniture store to pick out furniture for their house. But, what happened if the selected furniture is not fit the room or they are not suitable when putting at that place?

It will make people feel frustrating, waste of time.



Figure 1.1 Background

So, I apply an achievement of technology to build a comfortable and easy life for everyone. One of these things is a smartphone. I use the AR technology combine with the smartphone to help you to design and decorate your house. Here is a new way to resolve your headache.

For that reason, I decided to do project named "***Building an application for Home Furnishing using Augmented Reality***".

* 1. **Purpose**

This is an interior design application that allows you decorate and get ideas for your house.

With this application, designing and decorating your house has never been so quick and easy!

Whether you want to decorate the house of your dreams, here is the perfect application for you:

* You can use the camera of the phone to decorate the interior of your home with furniture which is virtual objects (we called 3D models) are the same as the real furniture and see how they look like in your rooms without going too far.
* There are many choices from over a list of pieces of furniture and accessories such as tables, lamps, chairs, beds, pictures, etc. it helps you consider what is suitable for your style.
* You also can interact with any object and edit them, by changing its size, position, and rotate to how they look better or make sure where is the good place to put them.
* After decorating you can take a picture or record the video and share to media social network to get ideas from the others for your room.

It is quick, it is fun and you can connect with a vibrant creative community while learning about diverse decor styles - thus improving your design skills and gaining inspiration that you can even apply in your real life.

Besides, this application can associate with third parties (furniture stores). After the people choose items, they can go to the store and buy them. It can help people save a lot of time. The 3D models are built the same as the real furniture.

Accessible to everyone from home decor enthusiasts to students and professionals, this application is the reference interior design application for a professional result at your fingertips!

* 1. **Scope**

Accompany me to develop this project is my friend (named Dinh Huu Quan - class: 13T1, is instructed by Mr. Truong Ngoc Chau). With the permission of Mr. Truong Ngoc Chau and Ms. Le Thi My Hanh, we have built this application together. So, here is the list of features and assignment of each member:

|  |  |
| --- | --- |
| **Dinh Huu Quan – Class: 13T1** | **Nguyen Thi Tinh – Class: 13T3** |
| * Create UI for the whole application. * Detect the plane such as the floor, the table, the wall, etc. And after that, create the visual plane to be easier to see. * Control gestures when manipulating with 3D models: use two fingers to rotate them on horizontal plane. * Implement all gestures on vertical plane. * Create the database for the whole system. * Build 3D models (the furniture such as tables, beds, lamps, pictures...). | * Build 3D models (the furniture such as tables, beds, lamps, pictures...). * Put 3D models after scan the plane and create an anchor to put 3D models to make sure it will be stable on the plane. * Control gestures when manipulate with 3D models: double tap an object to remove, keep and drag an object to another position using one finger. And using two fingers to scale an object. * Take a picture/ a video and save to local database, share to social networks. * Make an audio when manipulating with the 3D objects. * Import 3D models to the project. |

# **Chapter 2: THEORIES AND TECHNOLOGIES**

This application will bring the AR experience to the users, it combines the real world with virtual objects.

After researching and comparing all of the current framework and library such as ARKit (developed by Apple), ArCore (developed by Google), because we want to build this project on iOS devices so we choose using Swift language and ARKit framework and furthermore, ARKit is stable and it releases version 1.5 to help people can detect the vertical Plane which previous versions do not have.

To have the 3D models look like the real furniture and import them to our project, we had to build them by ourselves, but the time is limited so I have just built some 3D models, and another were downloaded from the Internet. We did use Blender software to do that.

This application requires an iOS device with an A9 or later processor and the iOS version of 11.3 or later.

Bellow is theories of technologies we did use in developing our project.

**2.1. iPhone Operating System**

### 2.1.1. Introduction

iOS (formerly iPhone OS) is a mobile operating system created and developed by Apple Inc. exclusively for its hardware. It is the operating system that presently powers many of the company's mobile devices, including the iPhone, iPad, and iPod Touch. It is the second most popular mobile operating system globally after Android.

The iOS user interface is based upon direct manipulation, using multi-touch gestures. Interface control elements consist of sliders, switches, and buttons. Interaction with the OS includes gestures such as swipe, tap, pinch, and reverse pinch, all of which have specific definitions within the context of the iOS operating system and its multi-touch interface. Internal accelerometers are used by some applications to respond to shaking the device (one common result is the undo command) or rotating it in three dimensions (one common result is switching between portrait and landscape mode). Apple has been significantly praised for incorporating thorough accessibility functions into iOS, enabling users with vision and hearing disabilities to properly use its products.

Major versions of iOS are released annually. The current version, iOS 11, was released on September 19, 2017. It is available for all iOS devices with 64-bit processors; the iPhone 5S and later iPhone models, the iPad (2017), the iPad Air and later iPad Air models, all iPad Pro models, the iPad Mini 2 and later iPad Mini models, and the sixth-generation iPod Touch.

* + 1. ***iOS layered architecture***

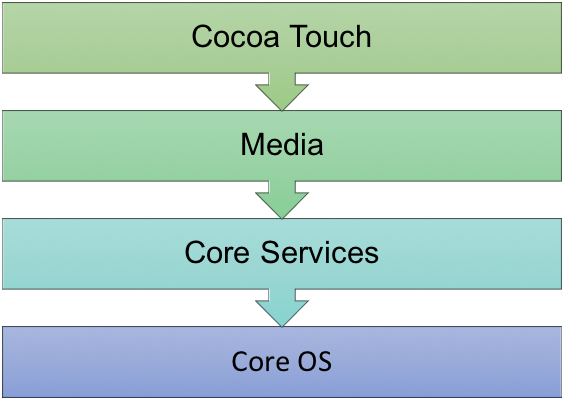
At the highest level, iOS acts as an intermediary between the underlying hardware and the apps you create. Apps do not talk to the underlying hardware directly. Instead, they communicate with the hardware through a set of well-defined system interfaces. These interfaces make it easy to write apps that work consistently on devices having different hardware capabilities.

Figure 1.2 Layered architecture

The implementation of iOS technologies can be viewed as a set of layers. Lower layers contain fundamental services and technologies. Higher-level layers build upon the lower layers and provide more sophisticated services and technologies.

Apple delivers most of its system interfaces in special packages called *frameworks*. A *framework* is a directory that contains a dynamic shared library and the resources (such as header files, images, and helper apps) needed to support that library. To use frameworks, you add them to your app project from Xcode.

* + 1. ***Features***

The home screen, rendered by SpringBoard, displays application icons and a dock at the bottom where users can pin their most frequently used apps. The home screen appears whenever the user unlocks the device or presses the physical "Home" button whilst in another app. Before iOS 4 on the iPhone 3GS (or later), the screen's background could be customized only through jailbreaking, but can now be changed out-of-the-box. The screen has a status bar across the top to display data, such as time, battery level, and signal strength. The rest of the screen is devoted to the current application. When a passcode is set and a user switches on the device, the passcode must be entered at the Lock Screen before access to the Home screen is granted.

iOS offers various accessibility features to help users with vision and hearing disabilities. One major feature, VoiceOver, provides a voice reading information on the screen, including contextual buttons, icons, links and other user interface elements, and allows the user to navigate the operating system through gestures. Any apps with default controls and developed with a UIKit framework gets VoiceOver functionality built in. One example includes holding up the iPhone to take a photo, with VoiceOver describing the photo scenery.

Multitasking for iOS was first released in June 2010 along with the release of iOS 4. Only certain devices—iPhone 4, iPhone 3GS, and iPod Touch 3rd generation—were able to multitask. The iPad did not get multitasking until iOS 4.2.1 in November. Currently, multitasking is supported on iPhone 3GS+, iPod Touch 3rd generation+, and all iPad models. Implementation of multitasking in iOS has been criticized for its approach, which limits the work that applications in the background can perform to a limited function set and requires application developers to add explicit support for it.

* 1. **Swift programming language**
     1. ***Introduction***

Swift is a general-purpose programming language built using a modern approach to safety, performance, and software design patterns. Swift is designed to work with Apple's Cocoa and Cocoa Touch frameworks and the large body of existing Objective-C (ObjC) code written for Apple products.

The goal of the Swift project is to create the best available language for uses ranging from systems programming, to mobile and desktop apps, scaling up to cloud services. Most importantly, Swift is designed to make writing and maintaining correct programs easier for the developer. To achieve this goal, we believe that the most obvious way to write Swift code must also be: *safe*, *fast*, and *expressive*.

On December 3, 2015, the Swift language, supporting libraries, debugger, and package manager were published under *the Apache 2.0 license* with a Runtime Library Exception. Swift is now free to be ported across a wide range of platforms, devices, and use cases.

* + 1. ***Architectural overview***

The features of Swift are designed to work together to create a language that is powerful, yet fun to use. Some additional features of Swift include:

* + - * Closures unified with function pointers.
      * Tuples and multiple return values.
      * Generics.
      * Fast and concise iteration over a range or collection.
      * Structs that support methods, extensions, and protocols.
      * Functional programming patterns, e.g., map and filter.
      * Powerful error handling built-in.
      * Advanced control flow with *do*, *guard*, *defer*, and *repeat* keywords.

The Swift language is managed as a collection of projects, each with its own repositories. The current list of projects includes:

* + - * The Swift compiler command line tool.
      * The standard library bundled as part of the language.
      * Core libraries that provide higher-level functionality.
      * The LLDB debugger which includes the Swift REPL.
      * The Swift package manager for distributing and building Swift source code.

Xcode playground support to enable playgrounds in Xcode.

* + 1. ***Features***
* Syntactic sugar

Under the Cocoa and Cocoa Touch environments, many common classes were part of the Foundation Kit library. This included the NSString string library (using Unicode), the NSArray and NSDictionary collection classes, and others. Objective-C provided various bits of syntactic sugar to allow some of these objects to be created on-the-fly within the language, but once created, the objects were manipulated with object calls.

* Access control

Swift supports five access control levels for symbols: open, public, internal, fileprivate, and private. Unlike many object-oriented languages, these access controls ignore inheritance hierarchies: private indicates that a symbol is accessible only in the immediate scope, fileprivate indicates it is accessible only from within the file, internal indicates it is accessible within the containing module, public indicates it is accessible from any module, and open (only for classes and their methods) indicates that the class may be subclassed outside of the module.

* Optionals and chaining

Swift supports five access control levels for symbols: open, public, internal, fileprivate, and private. Unlike many object-oriented languages, these access controls ignore inheritance hierarchies: private indicates that a symbol is accessible only in the immediate scope, fileprivate indicates it is accessible only from within the file, internal indicates it is accessible within the containing module, public indicates it is accessible from any module, and open (only for classes and their methods) indicates that the class may be subclassed outside of the module.

* Value types

In many object-oriented languages, objects are represented internally in two parts. The object is stored as a block of data placed on the heap, while the name (or "handle") to that object is represented by a pointer. Objects are passed between methods by copying the value of the pointer, allowing the same underlying data on the heap to be accessed by anyone with a copy. In contrast, basic types like integers and floating point values are represented directly; the handle contains the data, not a pointer to it, and that data is passed directly to methods by copying. These styles of access are termed pass-by-reference in the case of objects, and pass-by-value for basic types.

* Protocol-oriented programming

A key feature of ObjC is its support for categories, methods that can be added extend classes at runtime. Categories allow extending classes in-place to add new functions with no need to subclass or even have access to the original source code. An example might be to add spell checker support to the base NSString class, which means all instances of NSString in the application gain spell checking. The system is also widely used as an organizational technique, allowing related code to be gathered into library-like extensions. Swift continues to support this concept, although they are now termed extensions, and declared with the keyword extension.

* 1. **Augmented Reality**

Augmented reality, or AR for short, is a human computer interface paradigm within the general computer graphics discipline. AR aims at moving digital information into the physical world, thereby blurring the border between the physical and the virtual in a way that appears natural to the user. It enables a more intuitive, yet complex interface between man and machine. There is, however, no one widely accepted definition of what AR really is. AR was started by Sutherland with his seminal work on head mounted displays (Sutherland 1968). That work presented the first AR system (see Figure 1). But it was not until the 1990s that an attempt at clearly defining AR was given. Caudell and Mizell coined the term “augmented reality” in 1992 (Caudell and Mizell 1992) and two years later, a first attempt at defining AR was made. There exist two widely known definitions of AR today. The first, by Milgram and Kishino (see Figure 2), defines AR within the “Reality–Virtuality” continuum (Milgram and Kishino 1994). The second, by Azuma, gives more detailed criteria on what the prerequisites for AR are (Azuma 1997). Azuma’s original definition requires:

* + A combination of physical and virtual data
  + Registration in the physical world in 3D
  + Computed interactively/on-the-fly.

Combining physical and virtual information is what most people understand for AR. Adding the constraint of three-dimensional registration intentionally rules out all applications that merely display information over a video feed with a disregard of the underlying data, such as the news ticker on news shows. Requesting that the system should be computed on-the-fly differentiates AR from offline computer augmented movies.

To do AR project, nowadays Apple and Google provide for us two libraries to support us do that: ARKit ( from Apples and the projects will be run on the iOS device with iOS 11.3 and A9 processor or later) and ARCore ( from Google, the projects will be run on the devices which are produced by Google such as Google Pixel and Samsung Galaxy S7 or later).

* 1. **ARKit Framework**

****

Figure 1.3 ARKit Framework

* **ARKit concept**

ARKit (Apple ARKit) is Apple’s augmented reality (AR) development platform for iOS mobile devices.

ARKit allows developers to build high-detail AR experiences for iPad and iPhone. Environments captured through the device can have animated 3D virtual text, objects and characters added to them. AR scenes made by one individual are persistent and can be seen by others visiting the location later.

ARKit was introduced along with iOS 11. As ARKit is specified to run on Core A9 and higher iOS devices, the AR experiences can have more detailed content and maintain better environmental awareness. With iPhone X, ARKit can perform real-time face scanning and use this data to drive facial expressions of 3D characters.

Using the iOS device’s camera, [accelerometers](https://whatis.techtarget.com/definition/accelerometer), [gyroscope](https://whatis.techtarget.com/definition/gyroscope) and [context awareness](https://whatis.techtarget.com/definition/context-awareness), ARKit performs environment mapping as the device is moved. [Sensor fusion](https://whatis.techtarget.com/definition/sensor-fusion) of the inertial sensor data with the data from the camera allows for highly accurate location awareness and mapping. The software picks out visual features in the environment such as planes and tracks motion in conjunction with information from the inertial sensors. The camera is also used to determine light sources by which AR objects are lit. Apple’s solution to the increased detail and therefore memory usage is a sliding map where old data disappears for new. Users can place anchors to mark creations they want to save.

* **ARKit 1.5**

The latest update of ARKit in iOS 11.3 delivers new features that let you create an even more realistic user experience. With improved scene understanding, your app can see and place virtual objects on vertical surfaces, and more accurately map irregularly shaped surfaces. Real world images, such as signs, posters, and artwork can be integrated into the AR experience, so your app can fill a museum with interactive content or bring a movie poster to life. And now, the pass-through camera view of the real world is higher resolution and supports auto-focus for a sharper view in more situations.

* **Hardware and Software Integration**
* TrueDepth Camera: iPhone X and ARKit enable a revolutionary capability for robust face tracking in augmented reality apps. Using the TrueDepth camera, your app can detect the position, topology, and expression of the user’s face, all with high accuracy and in real time, making it easy to apply live selfie effects or use facial expressions to drive a 3D character.
* Scene Understanding and Lighting Estimation: With ARKit, iPhone and iPad can analyze the scene presented by the camera view and find horizontal and vertical planes in the room and can track and place objects on smaller feature points as well. ARKit also makes use of the camera sensor to estimate the total amount of light available in a scene and applies the correct amount of lighting to virtual objects.
* Visual Inertial Odometry: ARKit uses Visual Inertial Odometry (VIO) to accurately track the world around it. VIO fuses camera sensor data with Core Motion data. These two inputs allow the device to sense how it moves within a room with a high degree of accuracy, and without any additional calibration.
* High Performance Hardware and Rendering Optimizations: ARKit runs on the Apple A9, A10, and A11 processors. These processors deliver breakthrough performance that enables fast scene understanding and lets you build detailed and compelling virtual content on top of real-world scenes. You can take advantage of the optimizations for ARKit in Metal, SceneKit, and third-party tools like Unity and Unreal Engine.
  1. **Realm Database**



Figure 1.4 Realm database

Realm is an open-source object database management system, initially for mobile (Android/iOS), also available for platforms such as Xamarin or React Native, and others, including desktop applications (Windows), and is licensed under the Apache license.

In 2016 September, the Realm Mobile Platform was announced, followed by the first stable release in January 2017. It allows two-way synchronization between the Realm Object Server, and the client side databases that belong to the given logged-in user. Both a developer, and a commercial edition was released, along with a business license for integrating with other database management systems such as PostgreSQL.

The most notable features of Realm are the following:

* As Realm is an object store, its typed language-specific APIs map typed objects directly into the Realm file - therefore classes are used as the schema definition.
* Relationships between objects are allowed via "links". Each "link" creates a "backlink" as an inverse relationship to whichever objects are linking to the current object.
* The query results returned by Realm are thread-local views to the current "database version" (as Realm handles concurrency with [MVCC architecture](https://en.wikipedia.org/wiki/Multiversion_concurrency_control)), and these views "automatically update" when a transaction is committed from any thread, as long as Realm is able to update its instance version (which is possible on threads that are able to receive change notifications). When this happens, Realm calls change listeners that are added to its query results (if they've changed).
* Each thread-local view returns proxy objects that only read from/write to the database when an accessor method is called, meaning all database access is lazy-loaded. Please note that writes are allowed only while in a write transaction.
* As each query result and each proxy object is a view to the underlying data, any change made to the database is reflected in all objects that point to the same data. Realm generally calls this behavior "zero-copy architecture" (along with the previously mentioned lazy-loaded data access).

Realm Database is an alternative to SQLite and Core Data. Thanks to its zero-copy design, Realm Database is much faster than an ORM, and often faster than raw SQLite.

Realm Database:

F:\Users\QuanDinh\Desktop\Capture.PNG

Figure 1.3 Realm platform

* 1. **Blender**



Figure 1.5 Blender

Blender is a professional, free and open-source 3D computer graphics software toolset used for creating animated films, visual effects, art, 3D printed models, interactive 3D applications and video games. Blender's features include 3D modeling, UV unwrapping, texturing, raster graphics editing, rigging and skinning, fluid and smoke simulation, particle simulation, soft body simulation, sculpting, animating, match moving, camera tracking, rendering, motion graphics, video editing and compositing. It also features an integrated game engine.

# **Chapter 3: ANALYSIS AND DESIGN**

* 1. **Analysis**

Follow the name of the project, the application must interact with the real world through using the camera. Then, we can put 3D models on the screen and interact with them. It also needs some miscellaneous features to make the app work smoother and more accurate.

Here is the architecture of the project:

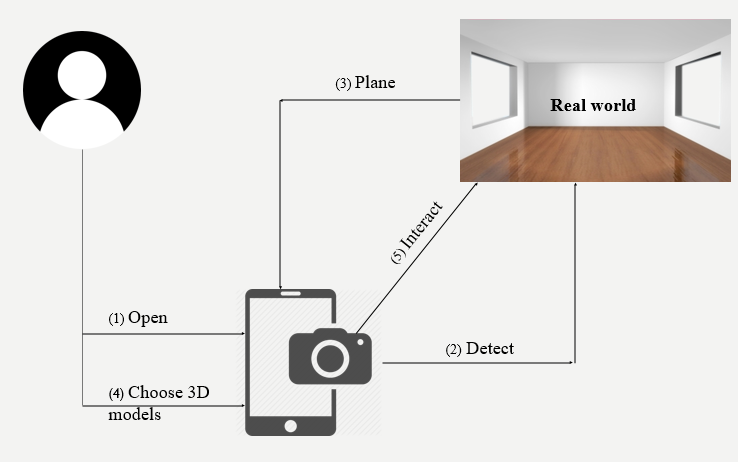


Figure 3.1 System architecture

To clarify this system, we have these features below:

* + 1. ***Detecting Plane***

The whole concept behind Augmented Reality is blending together the reality around us with virtual objects that exist only within our App. In order to be able to do that successfully, we need to be aware of the geometry of our surroundings.

In other words, we need to be able to identify the ceilings, walls, tables and other physical objects.

Then, we can start adding objects to the scene so that they will look real. Of course, size and geometry are just one part of the equation. There are many other important factors, such as the lighting of the scene, that need to be taken into account to place a realistic virtual object.

Detecting planes from a scene in ARKit are possible thanks to featuring points detection and extraction.

During the augmented reality session, ARKit continuously reads video frames from the device’s camera. Then, it tries to extract feature points to identify objects in the scene such as ceilings or furniture.

These feature points can be anything that helps identify objects: corners, structure lines, fabric characteristics, gradients, changes in color or form, edges of objects, etc…

ARKit does an amazing job detecting plane. Nevertheless, some work is required in order to recognize and update these planes properly. We use the Plane are detected and based the dimension of that Plane to make a new Plane for own and it will be visible easier to see.

So, when the user clicks Design and Decoration button at the Home screen, it will open the camera and inform to the user to detect the plane.

* + 1. ***Choose 3D models***

To manipulate with 3D models through the camera. Firstly, After detecting the plane. the user can click Add button in the middle of the screen to choose the item**.** But before that, we must choose a category.

* + 1. ***Manipulate with 3D models***

This feature is the main feature of this application. This feature will include three sub-features:

* Put 3D models on the screen: When the user wants to decorate their room. They just open the application choose Decoration Screen button, give this application the permission to access the Camera and the Microphone and then detect the plane. After having the plane, the user can choose any items on the list and the items will be put at the center of the camera. So, they can see that furniture in the 3D space through the camera of the phone.
* Control gesture when interacting with the 3D models: When the user wants to change the place of that item, they just need to use their finger to keep and drag it to another position. They also can rotate that item. If the 3D model is too small or too big, they can scale that item.
* Remove 3D models: When the user doesn't want to use that item, they can double tap it and the application will show Delete button, the system will show the confirm message, so they can click Yes button to delete that item out of the screen so that they can choose another item.
  + 1. ***Take a picture and share***

After user decorates the room, the user can take a picture of the screen (this picture will show the virtual models in the real world) and they automatically save in gallery database picture and show Share button and Cancel button. Users can click the Share button to save that picture into the phone or share the picture/video to media social network such as Google plus, Facebook, … or personal message, email, …. or click Cancel button to cancel that action.

* + 1. ***Manage Utility mode***

This feature is like utilities for the user.

The user can click the Options button on the screen. It will show the list of option, such as:

* Reset the place: This feature means the system will remove all the objects on the screen and the user can start to create a new one.
* Use Audio: If the user chooses this feature so when the user interacts with the objects on the screen, the system will have the sound.
* Record video: If taking a picture is not enough, the user can click to record the video after decorating, and the also can save or share to media network. This function is the same take a picture, it is automatically saved in gallery database after stopping record a video. They can also click to watch the video which has just recorded.
  + 1. ***Manage Gallery***

After the user can take a picture or record a video, they will be saved in the Gallery database. In the Home Screen, when the user clicks Gallery button, it will open the Manage Gallery Screen and show all of the images, videos of which are taken, recorded by the user before. And when the user clicks one of that pictures/videos, it will be shown detail and the user can play the video or see the detail (about the recored date, dimension, … of them) and user can click Delete button to delete them.

* 1. **Use-case diagram**
     1. ***Overall diagram***

The application includes these features:

* Detecting Plane.
* Choose 3D models.
* Manipulate with 3D models on the screen.
* Take a picture and share to social network.
* Manage Utility mode for the user.
* Manage Gallery.

With these features above (the details of the features will be explained below), the application has an actor (the users) interact with the system. Here is overall use-case.

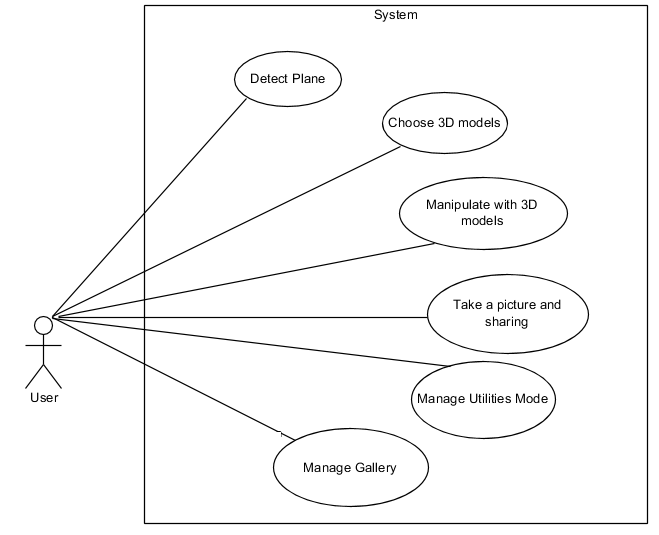


Figure 3.2 Overall usecase diagram

* + 1. ***Detect Plane***

When the user opens the application and chooses Start Decoration button at Home Screen. The system will show the text "Detect Plane", the user must let the phone scan around the floor, wall to find the surfaces (it's called Plane). When they have detected Plane, they can put 3D models on the screen.

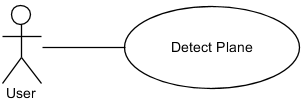


Figure 3.3 Detect plane usecase

* + 1. ***Choose 3D models***

After the users have detected Plane, the user can choose 3D models from the system. But before that, they must choose a category to correspond with 3D models they want to choose.

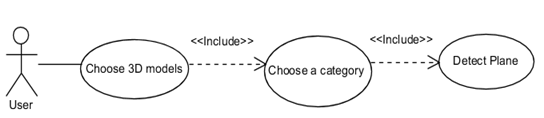


Figure 3.4 Choose 3D models usecase

* + 1. ***Manipulate with 3D models diagram***

When the users choose one 3D model from the system. It will be put at the center of the screen (and that position is a detected plane) so they can interact with items directly such as keep the item to move to another position, rotate the item, scale the item and tap to delete the item.

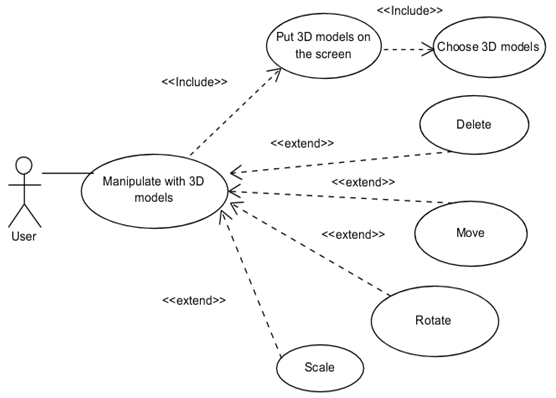


Figure 3.5 Manipulate 3D models usecase

* + 1. ***Take a picture diagram***

To share social network, the user must take screen by clicking camera button.

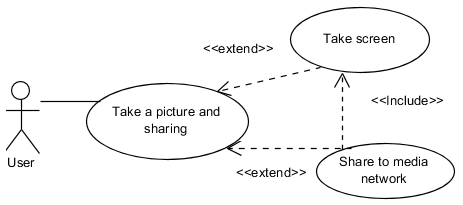


Figure 2.6 Detect plane usecase

* + 1. ***Utility mode diagram***

When the user clicks the Setting button, it will have three features use audio, reset session and record a video.

When the user records a video, they can continue choose to share to social network.

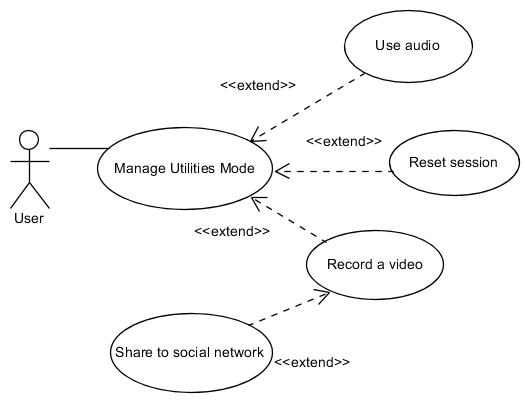


Figure 3.7 Utility mode usecase

* + 1. ***Manage Gallery diagram***

Manage Gallery includes these features: Add, View, View Detail and Delete.

After the user take a picture/video, it can save in the database automatically. And the user can click to see detail or delete them.



Figure 3.8 Manage gallery usecase

* 1. **Sequence diagram**

With these use-case diagrams, here are these sequence diagram of those features above.

* + 1. ***Manipulate with 3D models sequence diagram***

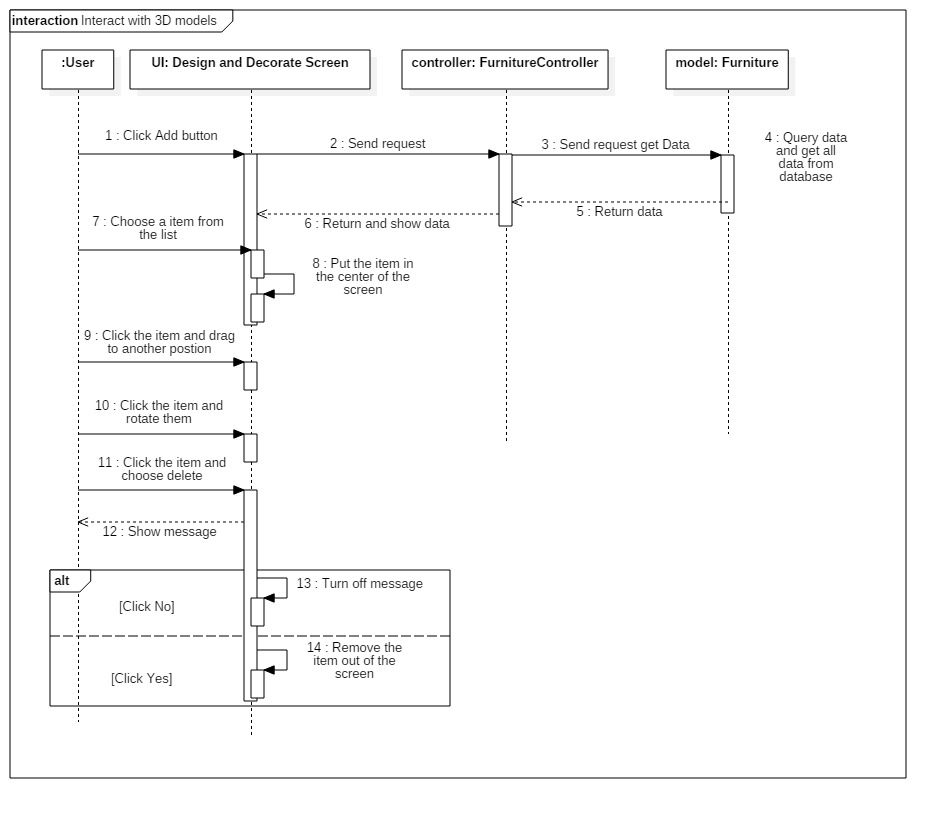
****

Figure 3.9 Manipulate 3D models’s sequence diagram

* + 1. ***Manage Gallery sequence diagram***

Manage Gallery includes three sub features: Add, View, Delete

* Add the picture into the gallery.

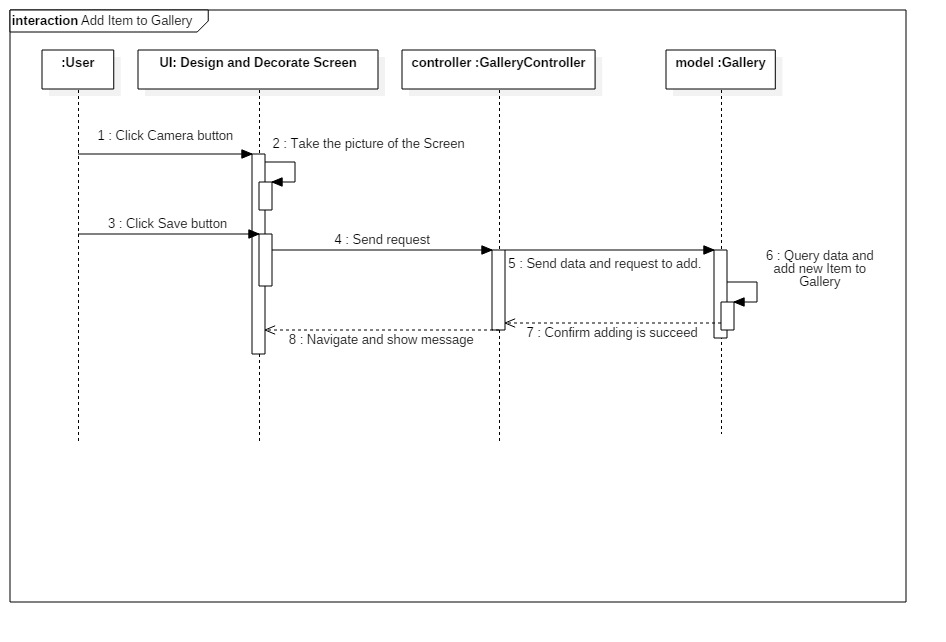
****

Figure 3.10 Add picture’s sequence diagram

* View Gallery

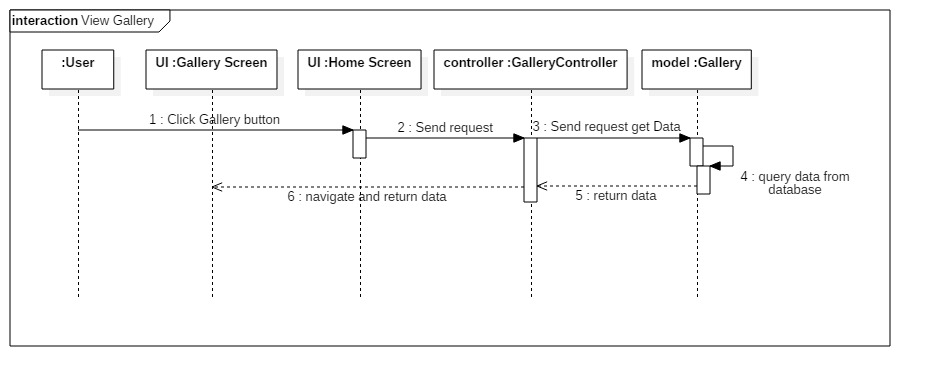


Figure 3.11 View gallery’s sequence diagram

* Delete an item of the gallery

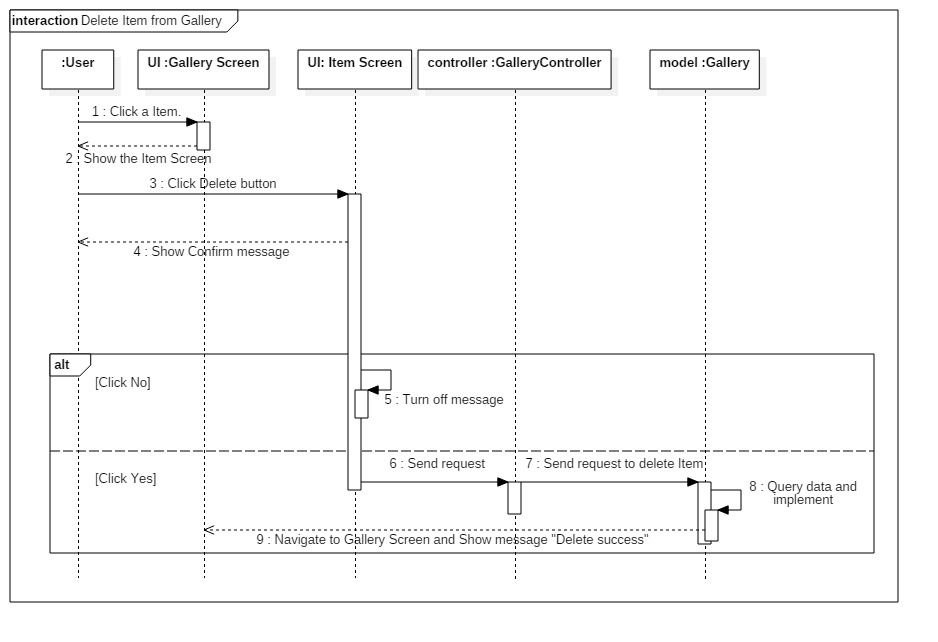


Figure 3.12 Delete 3D models’s sequence diagram

# **Chapter 4: IMPLEMENTATION RESULTS**

Demo main features

* 1. **Detecting Plane**

When the user uses the camera of the phone to detect around the floor, the wall… it will have the dimension of that plane (that plane doesn’t have any geometries, materials) so based on that dimension, we will create an own Plane (we call Plane Node) for users are easier to see.

* The first, the application will show the yellow point ( Feature Point) when users detect plane. If there are many feature points, it means that we will easier to create a plane Node which is stable when we interact with 3D models.



Figure 4.1 Detecting plane

* When as much Feature Point as possible, it will have the Plane so based on the dimension of that Plane, we create an own Plane Node likes bellow.

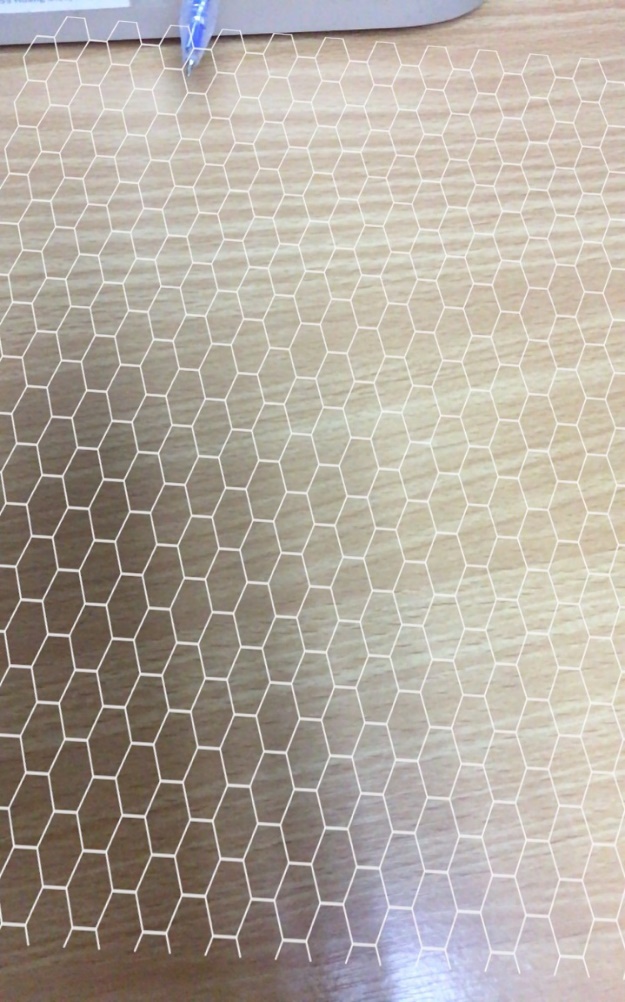


Figure 4.2 Plane Node after having detected Plane

* When the users have detected Plane, the Choose 3D models button will be appeared.

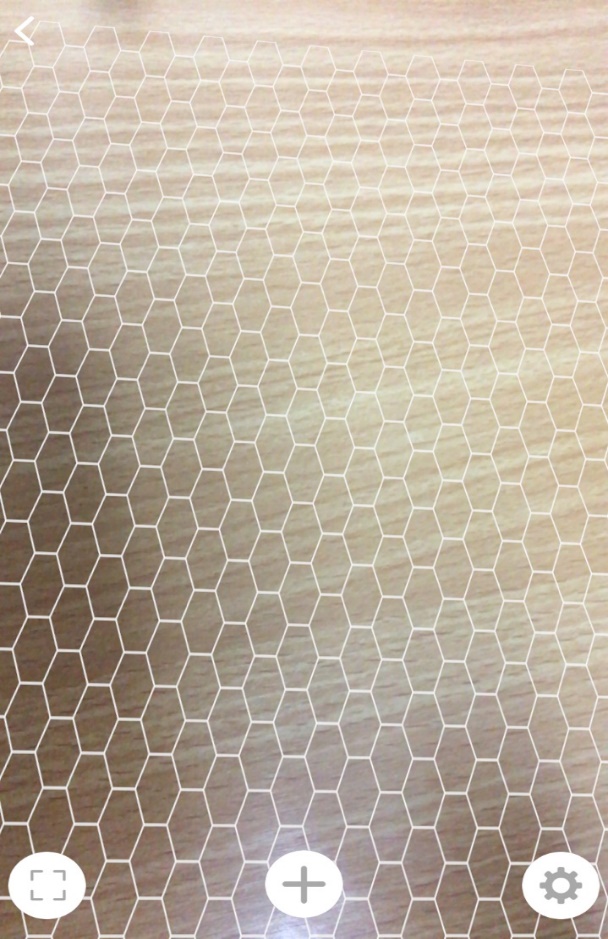


Figure 4.3 Add (+) Button will be appeared

* 1. **Choose 3D models**

As Choose button has appeared above, the users can click it and it will show the category of the furniture.

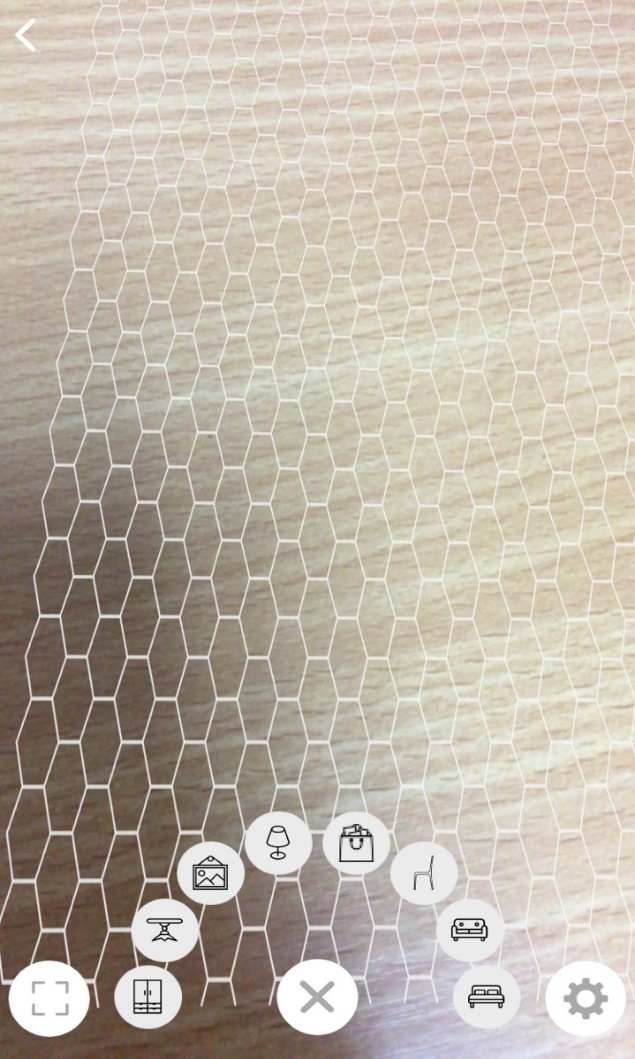


Figure 4.4 Choose a category

* The users choose a category, the list of that category will be showed on the screen.

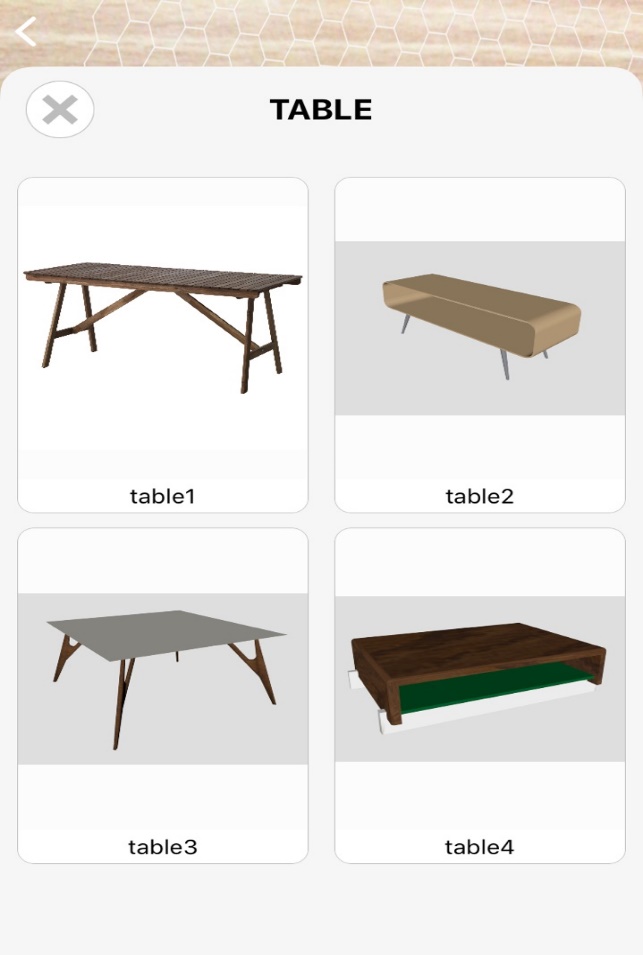


Figure 4.5 List furniture of the table

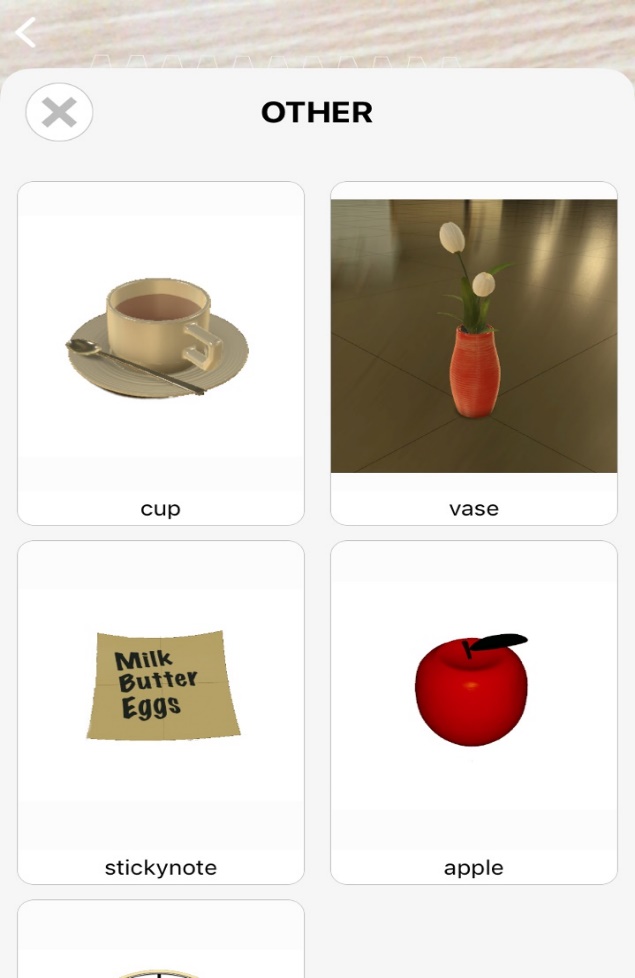


Figure 4.6 List the others

* The users choose one of them so it will be added on the center position of the screen (that position is sub of the Plane Node).



Figure 4.7 Add a 3D model on the screen

When the 3D models are added on the screen, The Plane Node will be removed out of that place to see the place is more beautiful.

* 1. **Manipulate with 3D models**

After adding the 3D models, we can manipulate with them.

* Keep and drag them to move another position.



Figure 4.8. Keep and drag to another position

* Using two fingers to rotate 3D models to see another side of them.



Figure 4.9 Rotate to see another side

* Scale 3D models for smaller or bigger.



Figure 4.10 Scale smaller (left) and bigger (right)

* 1. **Take a picture and sharing**

After decorating, the users can take a picture by clicking the left bottom button on the screen. And the picture will be added to Gallery database.



Figure 4.11 Take a picture

* So, then we can share the picture to the others via message (SMS) or via social network such as facebook by clicking the right top button. If the users do not want to share, just click cancel, it will reject that action.



Figure 4.12 The picture after taking the screen



Figure 4.13 Click Share button

* 1. **Manage Utility mode**

As I explained before, in the utility mode for the user, we have three options:

- Use audio: The audio will be turned on automatically on this application. However, the users can switch the button to turn off the sound.

- Reset place: This mode will remove all 3D models and restart new session (include detect plane).



Figure 4.14 Utility mode

* Record the video: It will the same as taking a picture function, after recording, the video will be saved in Gallery database and we can play it.



Figure 4.15 Record a video

* 1. **Manage Gallery**

On the Home Screen, the user click Gallery button, it will be showed all of the picture and video which are taken before.

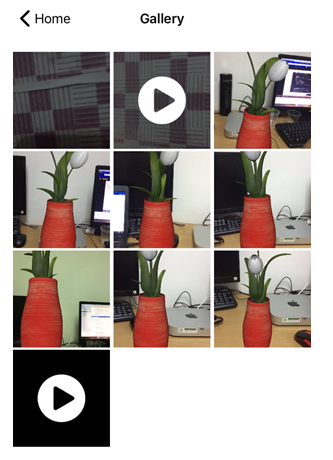


Figure 4.16 Gallery screen

* The users can click to see detail and delete them.



Figure 4.17 The detail of the picture

# **CONCLUSION AND FUTURE WORK**

## Achieve results

In this project, I did learn Swift language and I could understand how to work with ARKit library.

I learned about build 3D models and import them to project.

I knew the way put 3D models on the real world using the camera of the phone and the way to convert 3-dimension to 2-dimension and vice versa.

I also improved researching skills, technical skills, presentation skills, English and many other soft skills.

With gained bits of knowledge, I and my friend built the project meet initial requirements with achievements as below:

* The application brings unparalleled augmented reality experiences for users with many vivid images. It helps the life is better. With the simple UI but include many features, it will help people feel easier to control and choose the furniture which is suitable for their house, room. No time-consuming for finding, measuring,... the users just need to open the application and get their experience and invite the others, get the other's idea by sharing the picture, the video via message, mail, social network.
* The 3D models will be saved in the local database so it will be faster. It doesn't lose time to connect server.
* The users can use their fingers to control the 3D models. Look through the camera, it looks like they have magic because they can move the furniture easily.

However, with advantages, this application still have foibles:

* We did not connect to the server so when we want to import the new 3D models, it will have quite much complicated.
* The 3D model data is temporary. It is not the same as any real furniture.
* This application only works well in the idea environment: rough surfaces or on the wood table which is not too smooth or bright and the light is not too shining. If we scan on the smooth floor, the application cannot detect the Plane.

1. **Future works**

With disadvantages which are mentioned above. If we can continue to do this application, we will create a server and put 3D models to the server. And after that, we will release the server for the others they need to download 3D models.

We also find the way to can detect Plane on the bright wall or we will combine some libraries or use algorithms for detecting Plane.

# **REFERENCES**

[1] Swift for Apple Developer. Available on website: <https://swift.org/>

[2] Realm database for multiple languages. Available on website: <https://realm.io>

[3] ARKit library for Augmented Developer. Available on website: <https://developer.apple.com/arkit/>

# **APPENDICES**