**Dharmsinh Desai University, Nadiad**



**Faculty of Technology**

**Department of Computer Engineering**

**B. Tech**, Computer Engineering (**CE**)

**Sem** - **VI**

**Subject**: System Design Practice

**Project** **Title**: *Augmented Reality Use Cases and Applications*.

**By**: Moxank Patel CE-001 (17CEUON061)

Aatish Chaudhari CE-003 (17CEUBN031)

Jaimin Chauhan CE-018 (17CEUBS084)

**Guided By**: Prof. Jigar M Pandya

**CERTIFICATE**



**Dharmsinh Desai University**

**Faculty of Technology**

**Department of Computer Engineering**

This is to certify that the project work titled

***Augmented Reality Use Cases and Applications***

is bona fide work done by

***Moxank Patel (17CEUON061),***

***Aatish Chaudhari (17CEUBN031),***

***Jaimin Chauhan (17CEUBS084)***

carried out in the partial fulfilment of term work for the subject System Design Practice in

The academic session

2019-2020

**Assistant Professor Jigar. P** **DR C.K. Bhensdadia**

Dept. Of Computer Eng. Dept. of Computer Eng.

**Table of Contents**

1. Abstract (What is AR?) 4
2. Technologies Used 5
3. Software Requirement Specifications (SRS) 6
4. Diagrams 16
5. Deployment Descriptor 20
6. Test Case 23
7. Versioning 24
8. Screenshots 25
9. Conclusion 55
10. Limitation and Future Extension 59
11. References 60

**What is Augmented Reality (AR)???**

* It’s an interactive experience of a real-world environment where the objects that reside in the real world are enhanced by computer-generated perceptual information, sometimes across multiple sensory modalities, including visual, auditory, haptic and olfactory (smell).
* Augmented reality is the result of using technology to superimpose information —sounds, images and text.
* In simple words adding something virtually to real world object. It can be visual, audio or any other similar stuff.
* 

You might have come across this popular game ‘Pokémon Go’, it may come as surprise but it is uses Mixed Reality Concept which has partial use of Augmented Reality.

**Technologies (Behind the scene)**

We have majorly used Google AR Core library for implementing our use cases. We have used Google AR Core because it is open source and currently in trend. Other library such as Vuforia is also used to demonstrate a simple use case of object augmentation. Vuforia wasn’t used in other use cases because of its limited support.

Unity is used to develop application making our application almost platform independent.

C# is language behind development as unity development can be done with help of C#.

Google Cloud Service is being internally used by Google AR Core library to retain some of the information.



*Google AR-Core Vuforia Unity*

**Table of Contents of SRS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Type** | **Why it is/not created** | **Phase in which it is generated** | **Is Implemented** |
| SRS | It is created to represent project idea to developers. So that, they can have clear cut idea about what they are developing and how they proceed towards development. | Initial phase.  During planning, requirement gathering stage it was being prepared. | Yes |
| ER Diagram | It is entity relationship diagram which is prepared to represent dependency and relationship among table.  And in our project, there no major uses of database, so it is omitted. |  | No |
| DFD Diagram | It is created to represent data flow among class.  Here in our project there are no classes which are implemented externally, all classes in out project are derived from unity or google arcore library.  And we just integrated unity classes and google arcore classes with the help of function and unity life cycle.  So, we don’t require DFD. |  | No |
| Class Diagram | As mentioned above that, we didn’t create any external classes.  So, no Class Diagram would there.  There were no classes created because while developing apps in unity it follows certain life cycle.  It is,  Initialization,  Update,  Destroy.  And only that classes were created with no meaning to entity. |  | No |
| USE Case Diagram | It is created to represent interaction of particular functionality with a certain type of user and also to show dependency among them. | It created right after SRS is created. | Yes |
| ACTIVITY Diagram | It is used to represent the flow of system.  As mentioned earlier unity follows certain life cycle.  So, every time same activity/pattern is followed, hence no activity diagram. |  | No |
| Sequence Diagram | It is created to represent flow among objects or in other words how they interact.  And we don’t have any external meaningful classes and also the classes/script which are there aren’t interacting with each other as they loosely coupled. |  | No |
| Component  Diagram | It is used to represent abstract view of classes. |  | No |
| Deployment Diagram | It is used to describe system hardware level |  | Yes |

**SRS**

SRS for **LiveNewsPaper**

#List of Users

1. End User

2. Admin

#SRS

\* Higher Level Requirements

1. Add Video

2. View Video

##Functional Requirements

1. Add Video

1.1 Scan Image

I/p: Input Image

O/p: options of browsing corresponding video is displayed

1.2 Attach Video

I/p: Path to video

O/p: successful added message or error message

2. View Video

I/P: Scan Image

O/P: Image is replaced by video

SRS for **ObjectAugmentation**

#List of Users

1. End User

#SRS

\* Higher Level Requirements

1. View Object

##Functional Requirements

1. View Video

I/P: Scan Image

O/P: Image is replaced by object (Cube)

SRS for **TreasureHunt**

#List of Users

1. Client
2. Host

#SRS

\* Higher Level Requirements

1. Add Room

2. Join Room

3. Add Anchor

4. Add Star

5. Add Text

##Functional Requirements

1. Add Room

I/P: Host clicks on button

O/P: Room is created and displayed

1. Join Room

I/p: Client enter room id

o/p: successful message or retry message is rendered

1. Add Anchor

I/P: Client click on screen

o/p: anchor point object is spawned

1. Add Star

I/P: Client click on screen

o/p: star object is spawned

1. Add Text

I/P: Client click on text area

o/p: text is updated

SRS for **FaceFilt**

#List of Users

1. End User

#SRS

\* Higher Level Requirements

1. View Thought

##Functional Requirements

1. View Thought
   1. Get Face

I/P: Scan Image

O/P: Image is scanned and set coordinates for face are returned.

* 1. Augment Object

I/P: Game object [Filter Mesh] from unity

O/P: Augmented Face

* 1. Render Thought

I/P: Coordinates of face

O/P: Randomly selected thought

**Use Case Diagram**

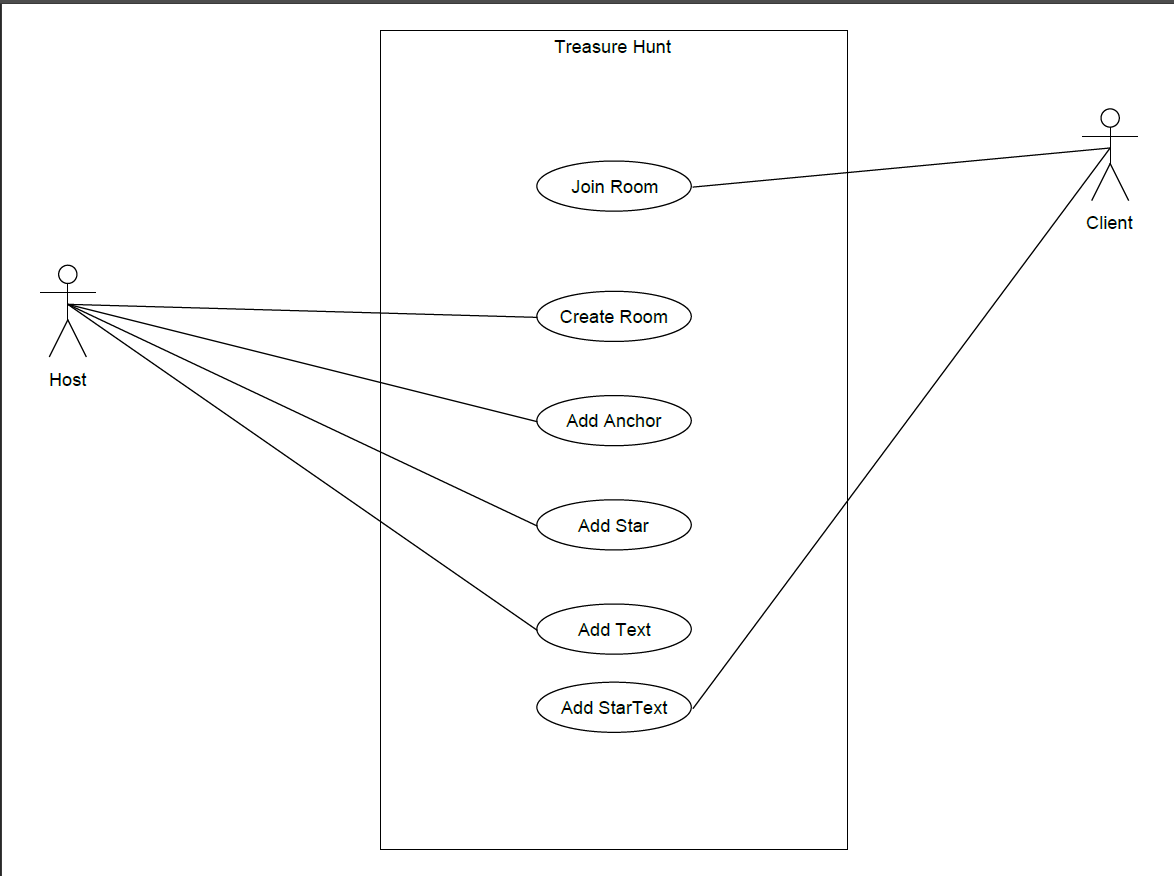


Figure 1. Treasure Hunt Use-Case

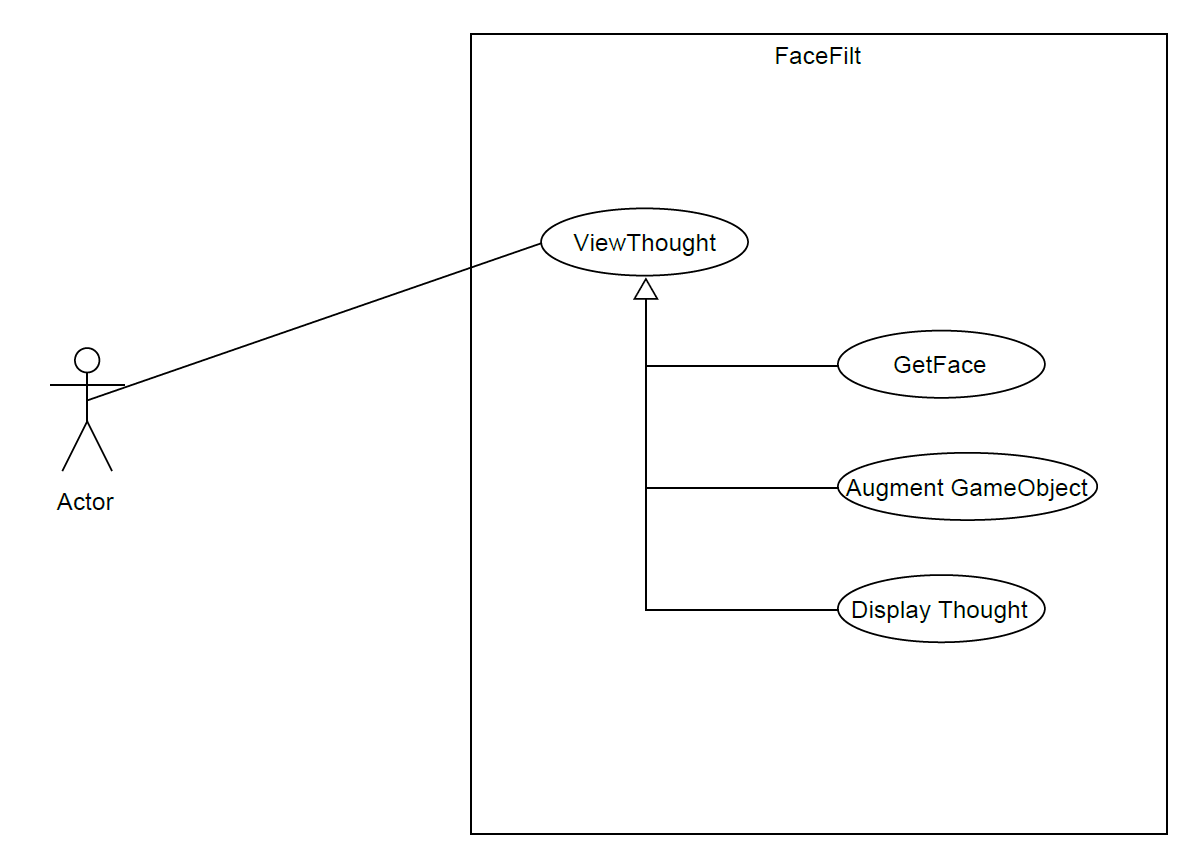


Figure 2. Face Filter Use-Case

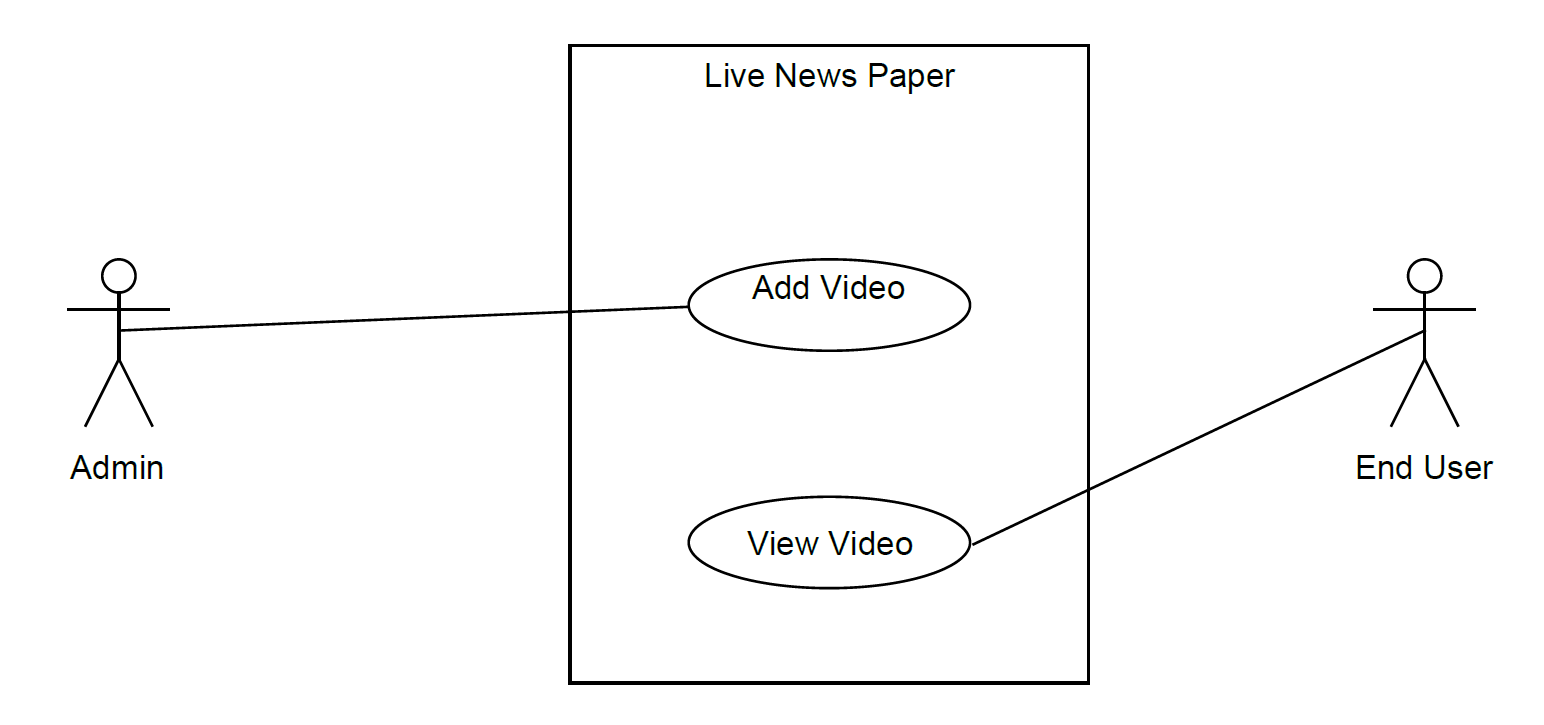


Figure 3. Live News Paper Use-Case

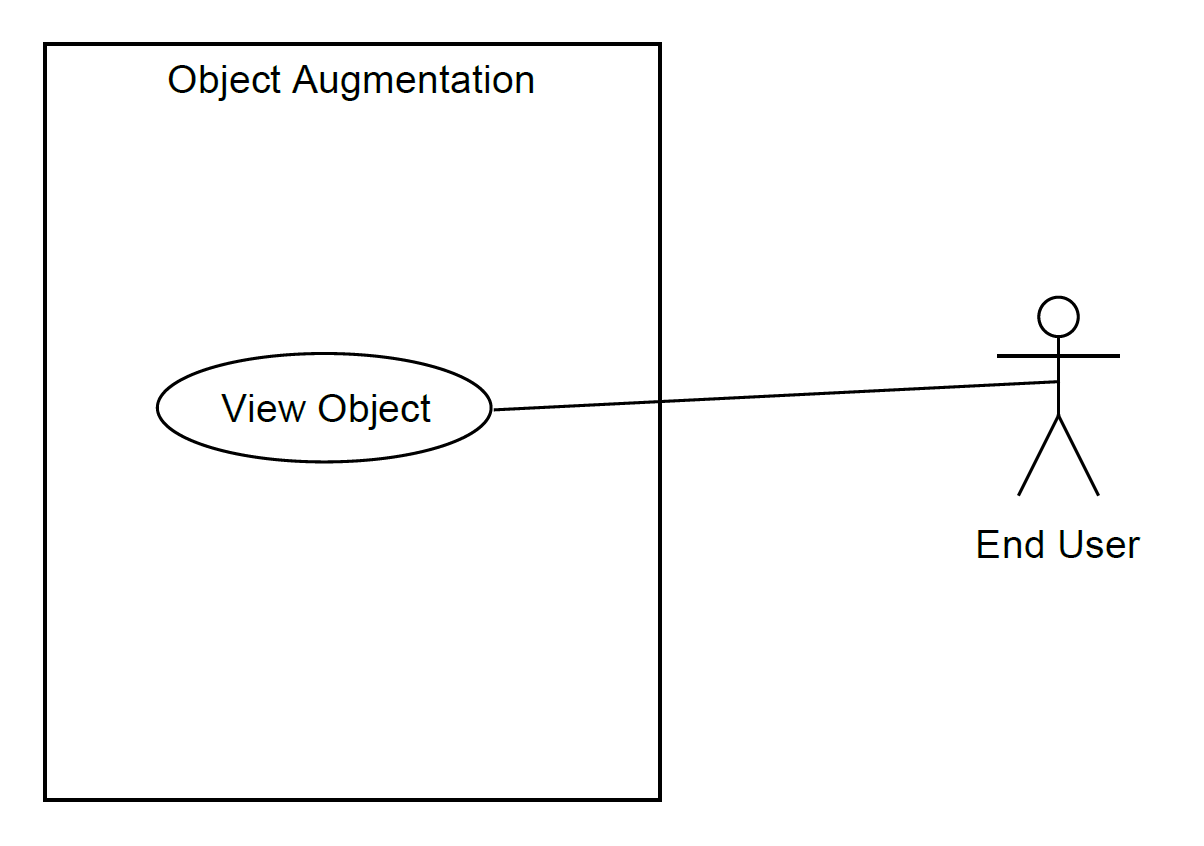


Figure 4. Object Augmentation Use-Case

**Deployments Diagram**

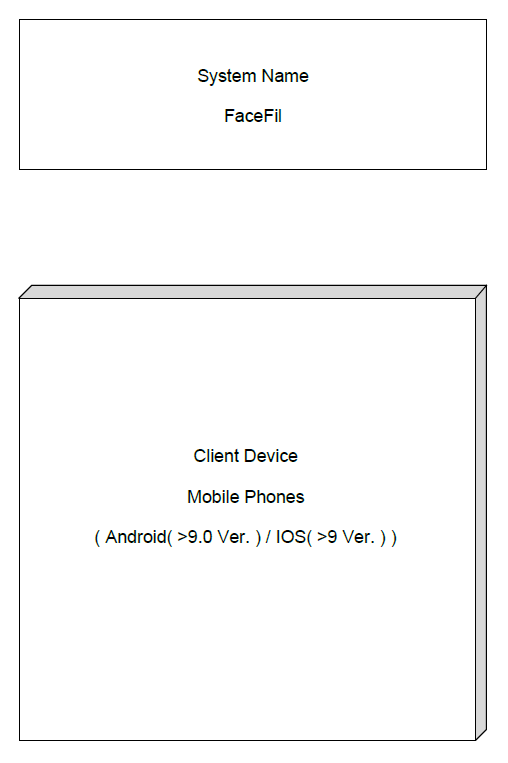


Figure 5. Deployment Diagram for Face Filter

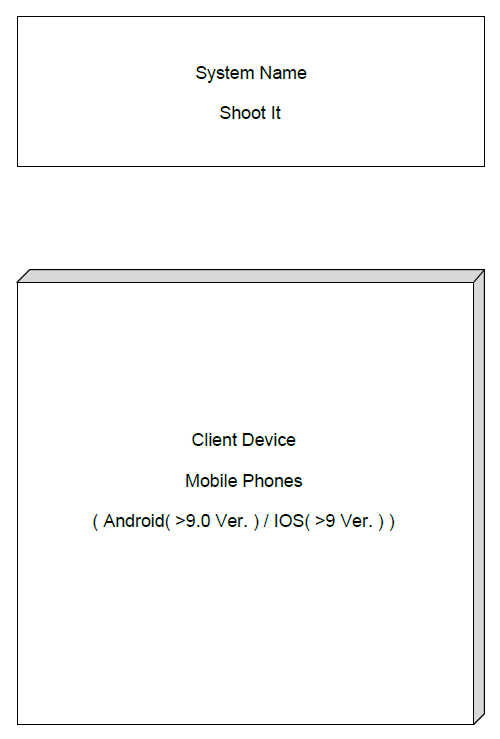


Figure 6. Deployment Diagram for Face Filter

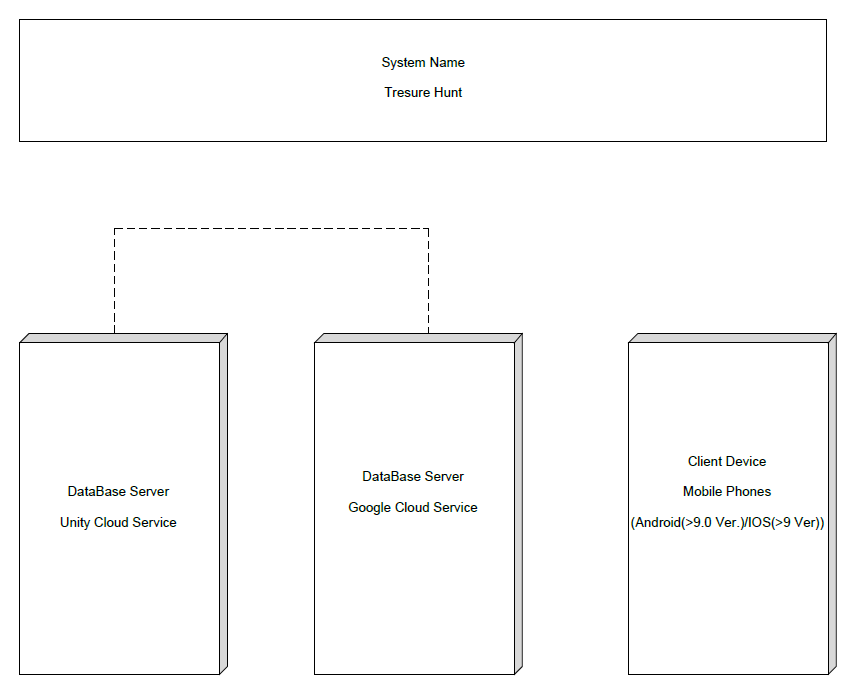


Figure 7. Deployment Diagram for Treasure Hunt

**Deployment Descriptor**

1. Generate signed .apk/. abb using .jsk file
2. Make Play Store Developers account.
3. Create new project in account.
4. Add Screen Shot of application and privacy policy
5. Create release on the Play Console
6. Upload .apk or .abb file
7. Enter your app details.
8. Apply for approval.
9. After approval select devices on which you want to publish your app
10. After approval app would be published.

**Test Case**:

Manual Testing was done.

Those cases are depicted in below section of screenshots.

**Implementation Details:**

In total we have developed 5 applications out of which first 2 basic application of image augmentation and video augmentation is performed by all three members.

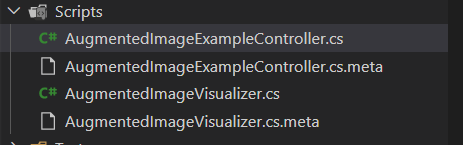
And rest of three application were leaded by each of the members as follows:

Treasure Hunt is by Aatish Chaudhari,

Shoot-It by Jaimin Chauhan and

FaceT by Moxank Patel

Unity scripts (which are actually classes derived from MonoBehavior, they are written in C#) were created for object manipulation.



Major functionality which we have used are inherited for Google AR-Core library and therefore using those classes and implementing them in our files.

The function which were created were under the script for manipulating the models, button, resizing the object and etc.

We have just integrated Google AR-Core library service with unity app creation module.

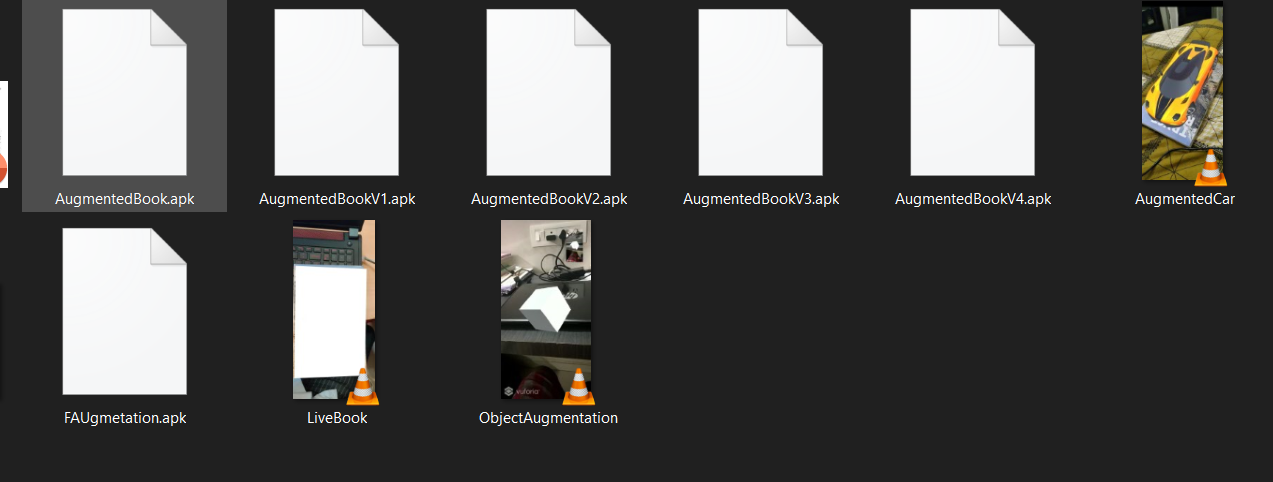
Some of the major functionality which were needed for completion of task were implemented under lifecycle of unity script.

For example, in Treasure Hunt we needed a feature like,

Feature: When user is in certain range of the clues, then only they would be able to view the clue else they clue remain hidden. It was implemented under one the method that is inherited method of MonoBehavior (Unity Script).

Major Functionalities are described in below section of screenshots.

**Versioning**



We needed to maintain version in Live Book and for Treasure Hunt.

Their deployed apk were stored according to their version as we can see above.

**Screenshots**

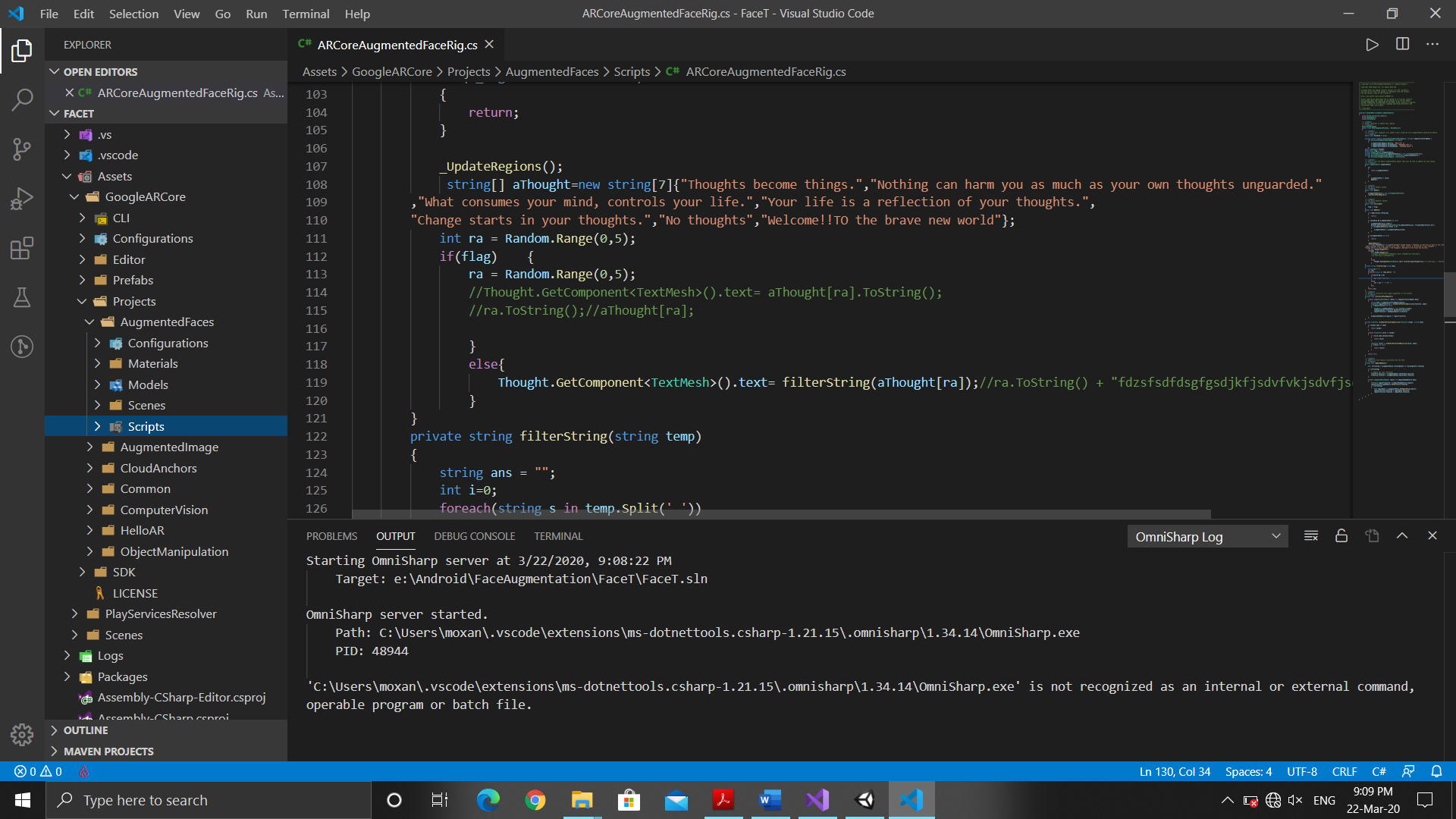


Figure 8. Hierarchy of Project

Google AR-Core: This folder provides modules implementation for some of functionality of AR like augmenting images and likewise. Moreover, it detects images and gives us coordinate. Some of the other functionality provided by Google-AR Core are:

1. Provide Database
2. Provide Cloud DB Service
3. Detects Image
4. Provides Support for Real Time Sharing, ultimately making it easier to manage multiple players in a game if any.

**Scenes**:

It contains the environments and menus of the game and we can think of it as unique level represents a unique scene.

**Scripts:**

It is folder containing script for manipulating objects of unity. Mainly contains C# code files.

**Models:**

Model are the files that are placed in the Assets folder in project and are automatically imported and stored as Unity Assets. A model file can contain a 3D model, such as a character, a building, or a piece of furniture. The model is imported as multiple Assets.

**Configuration**:

As name itself suggest. Contains configuration for project and is created while unity project is created and developer can edit it according their needs.

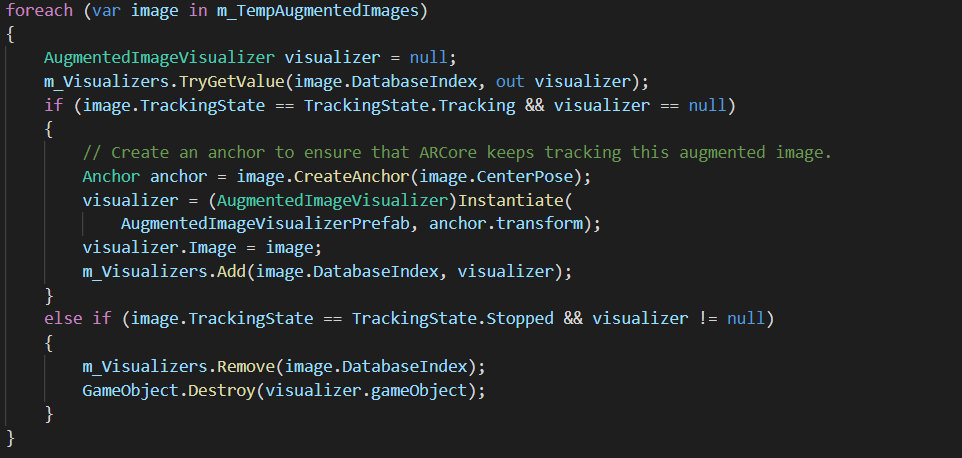
**Materials**:

This module is created whenever unity project is created and it contains texture, objects, etc.…

* Every other use case developed contains same hierarchy and the things that changes are functional implementation, material, prefabs, models.
* And the flow remains same for unity application lifecycle.

Modules

Detecting Image Module



Here above code gets image from camera and stores it in AugmentedImageVisualizer Class object. The class is provided by Google AR Core Library. After creation of object Anchor are created on camera space to track movements of images without retracking whole image.

M\_Visualizer is object of another utility class.

Here, we try to retrieve image if already stored else we store it.

After a new image is stored, the utility class perform different logic based on application.

Like in Live-Book:

It would augment video.

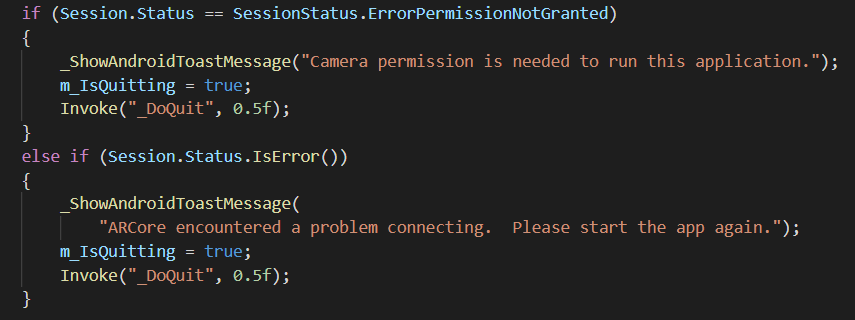


Fig. Z Getting Camera Permission

Here above is common module to show error message if camera permission isn’t acquired.

**In Treasure Hunt:(By Aatish Chaudhari)**

Using the cloud anchor feature in Google Arcore Library

We added new features in the augmented object which is associated with the script which controls the object property

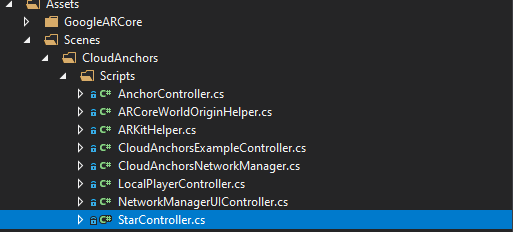


Fig A.1: Hierarchy of Project

StarController.cs is our custom code to manage different properties

Of the chest object.

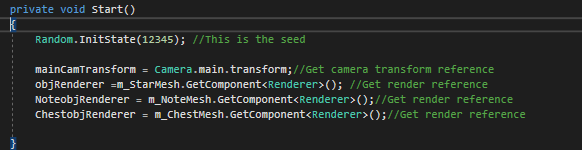


Fig A.2: Start function

Start function is called in unity life cycle when an game object in

Unity is Instantiated.

* mainCamTransform is to get camera position to we can track how far is user form our chest object
* ChestobjRenderer is to controller the display property of chest object so we can hide and unhide the object

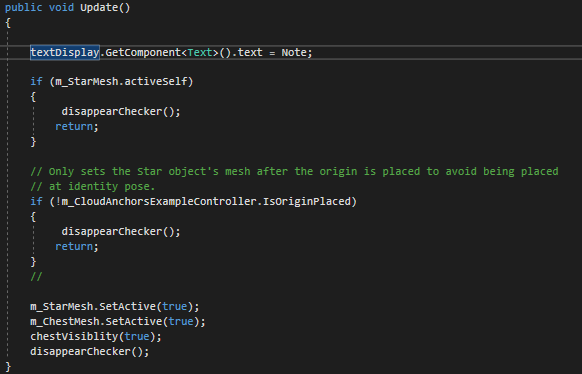


Fig A.3: Update Function

To display the note on the text object

Continuously check for the position of user form the chest object

And based on that decide the visibility of the object

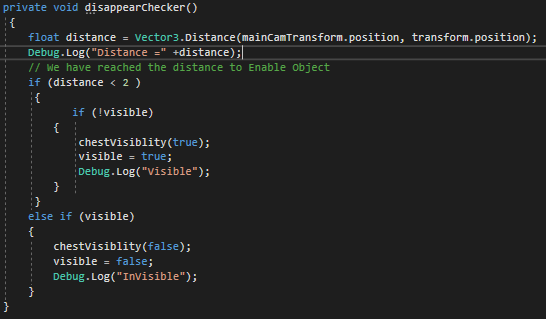


Fig A.4: Function to calculate distance between user and chest

Using Vector3 class present in the unity library we calculate the distance between phone camera and the chest after that we compare it with some constant distance like 2 i.e. 2 meter in real world, we decide the visibility of the chest

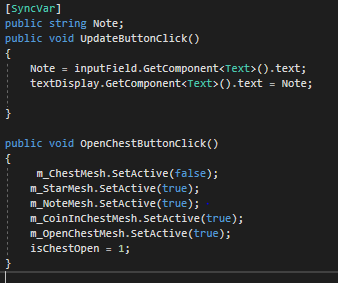


Fig A.5: Function to Update Note string

Annotation [syncvar] is used to update string object note over the network function updateButtonClick is associated with click event of button object of the chest and openChestButtonClick is associated with open button object in chest object.

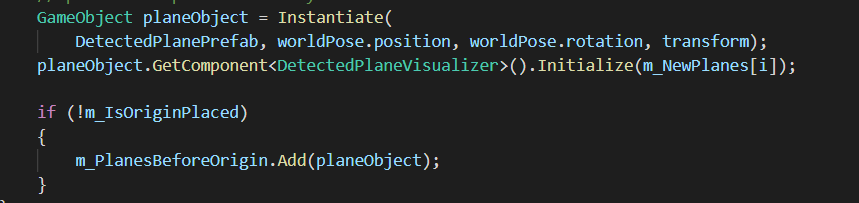


Fig. A.6 Adding clues in Treasure Hunt

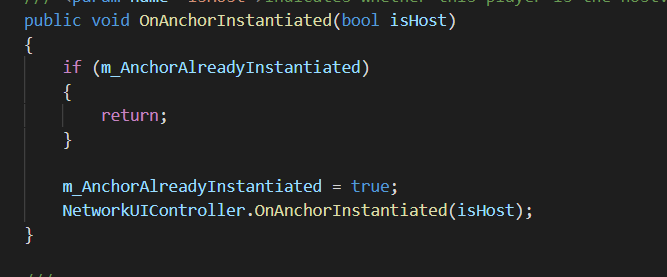


Fig. A7 Adding Anchor Points

Here, we first scan plane and store it in DetectedPlanePrefab.

After that in if condition we check if anchor point has been placed. If placed then only add star else don’t add star.

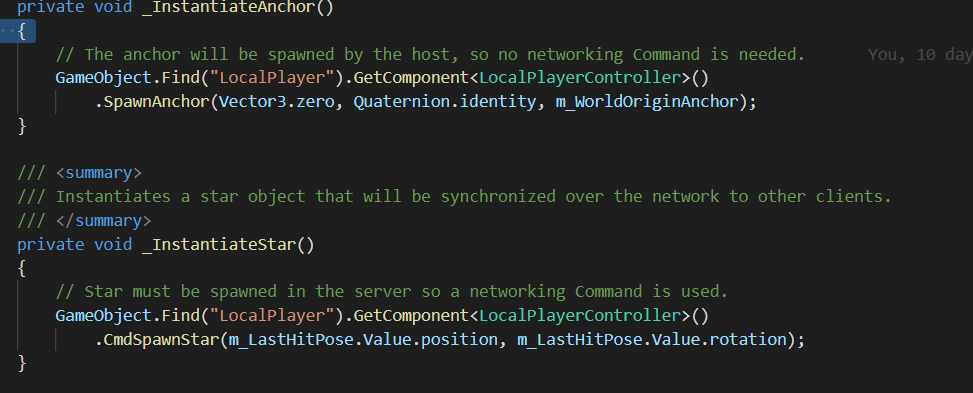


Fig. A8 Instantiating stars and anchors.

Here we simply find current player who spawn the star from dictionary and then pass the coordinated of star to the other player through unity utilities.

**In FaceT:(By Moxank Patel)**

Application finds coordinates of face features. It then attaches fox ear and augment thoughts on face.

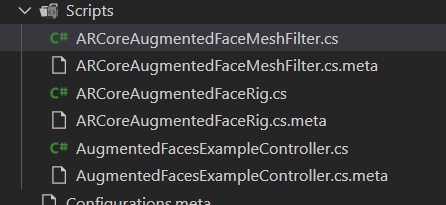


Fig B0. Scripts Created for FaceT

Here Controller script is main script which runs continuously on loop to find face from given image. If it detects face then it passes object of face to Face Mesh and Face Rig where they attach model of fox and thoughts on given face.

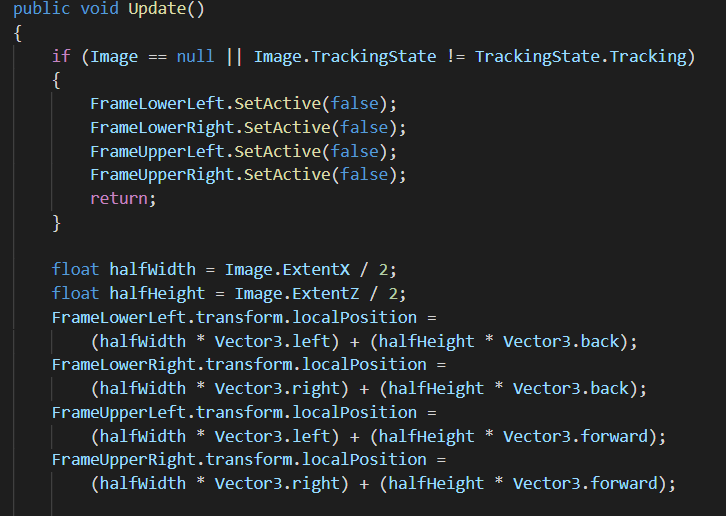
Mesh script forms invisible grid layout of face so that attaching fox model would be easier

Fig B1. Adding Fox Ear to Face

Here above code has update method which is part of lifecycle of Unity Class MonoBehavior.

This update method is invoked whenever we get face coordinates.

It then attaches Fox Ears and Thought to those coordinates.

Here Above “Image” is an object of Visualizer which class of AR Core library. It simply stores image and its attribute. If no image then removes Augmented Object else attach it to new Coordinates.

These co-ordinates are stores in list of face (It can be observed in Fig.B5).

Here, above method is similar for rest of the applications only variables (unity model objects) would be differing while logic remains same.

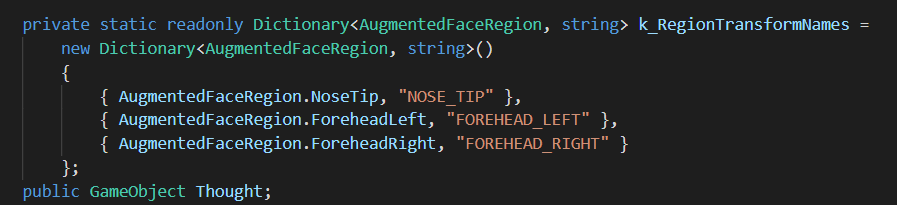


Fig B2. Declaration of variable

These variables are declared to detect face feature and augment thoughts.

AugmentedFaceRegion is class which provided by google AR core library. And those variables are static int of those class i.e. AugmentedFaceRegion.NoseTip.

And whenever AR-Core detects those features it assign those int to respective regions.

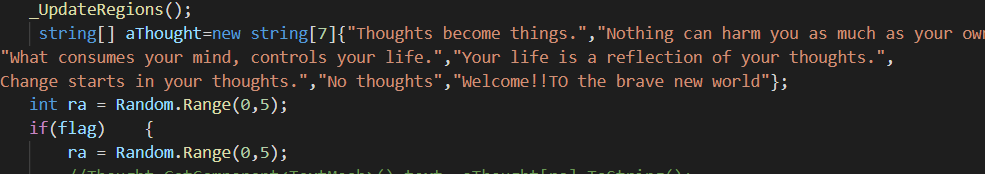


Fig B3. Generating Random Thoughts (Update Function in Mesh)

Here a random number is generated and on the basis of generated random number one of the thoughts is picked from list as shown above and render onto current face (Zeroth Face).

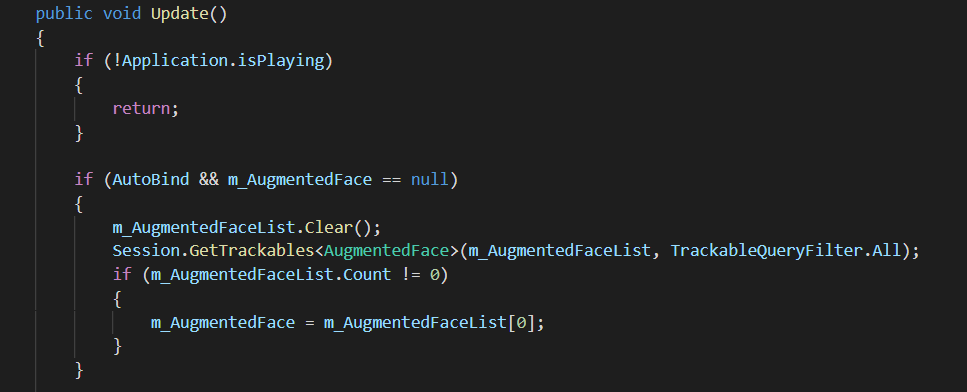


Fig B4. Selection of current face

What if two face are detected by camera.

No worries, here, our application apply thoughts to face which appeared first (i.e. FIFO policy). It can be seen in above declared var m\_AugmentedFaceList [0]. This face list updated in controller script frequently, that is whenever it detects face.

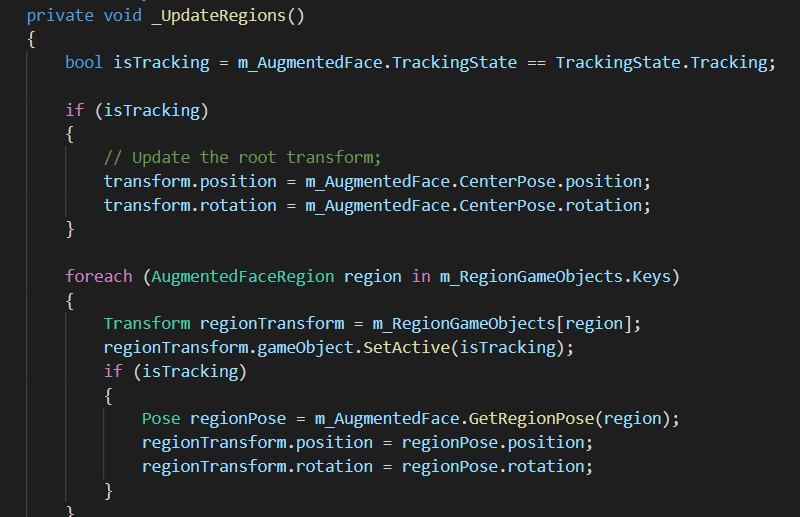


Fig B5. Applying ears and thoughts to appropriate position.

Here system now calls \_UpdateRegion () from Update () after successful detection of a face and then it updates region according to its anchor point.

Here m\_RegionGameObject is dictionary object which declared (Fig A1) earlier and now according to attributes i.e. nose, left-ear, right-ear, head.

It attaches object onto it.

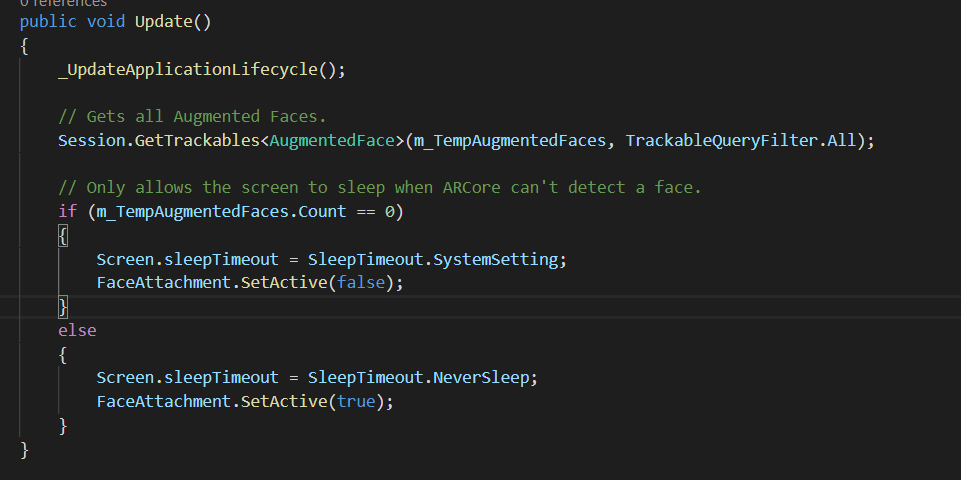


Fig B6. Update Method in Controller

Here we get all faces detected by camera through AR Core library

And augment model on it through Face Mesh and Rig.

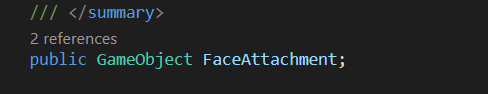


Fig. B7 Model of face.

Here FaceAttachment is actually fox ears which would be a model in unity Models.

**In Shoot it (By Jaimin Chauhan)**

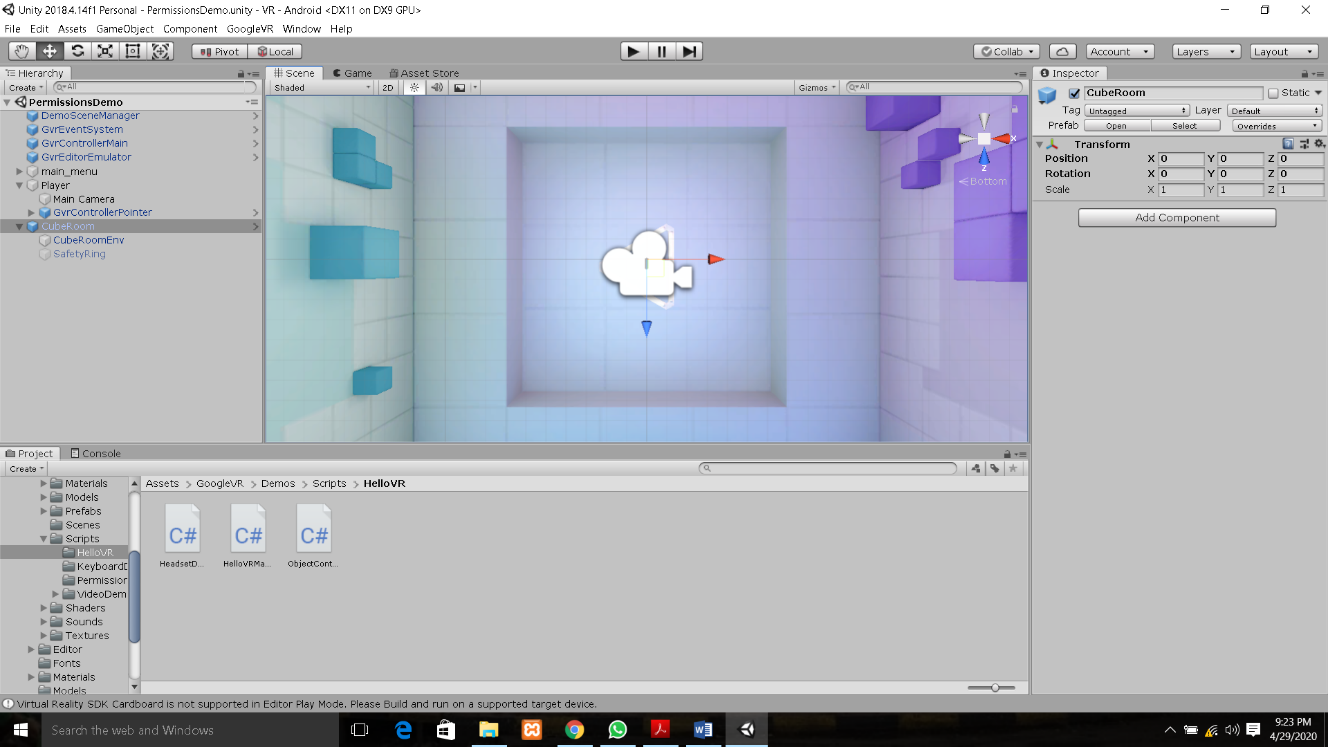


Fig C1. Unity Project Structure

In this figure left panel display a file structure of a project. Right panel shows the properties of an object used in project like Camera, room etc… Bottom panel shows how many files contains by particular folder. Centre panel show the graphical view of objects.

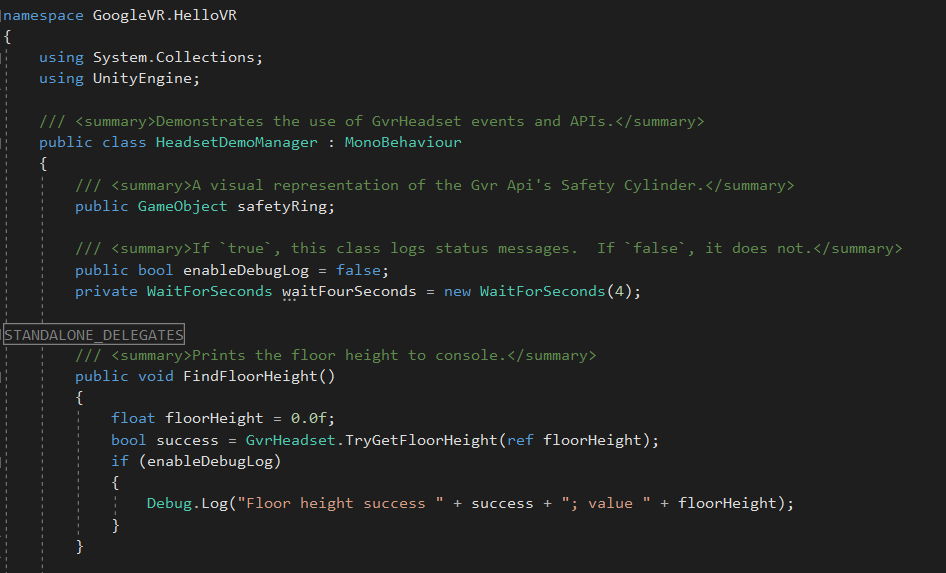


Fig C2. FindFloorHeight Function.

This class is use full to set up VR handset. It calculates all parameter like Height of floor.

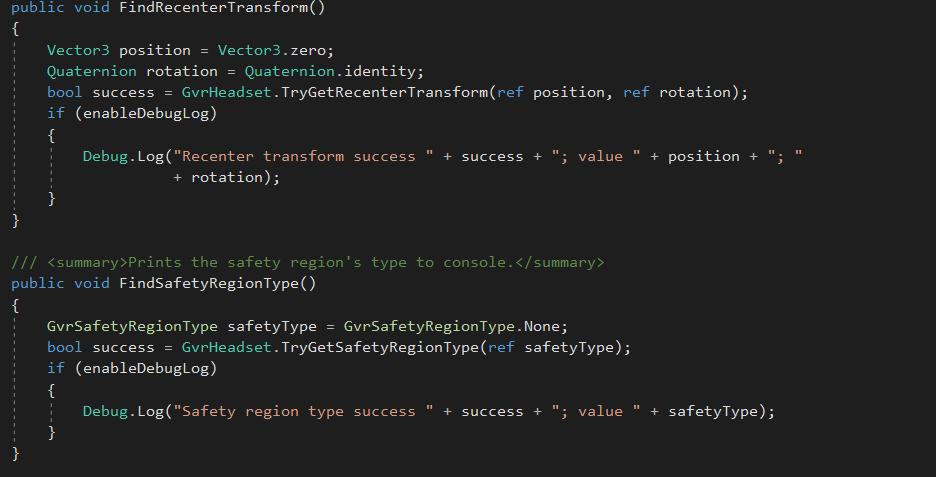


Fig C3. Function that find a centre point.

This function is used to find the centre. So, that we can find other associated parameters with it.

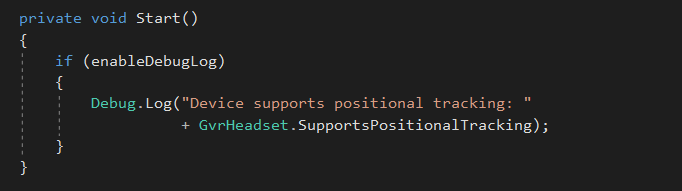


Fig C4. Start function.

When we launch application so first this function call and it start position tracing.

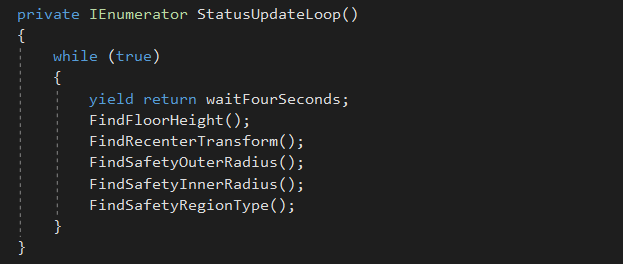


Fig C5. It is infinite loop that is continuously takes all parameters using above functions.

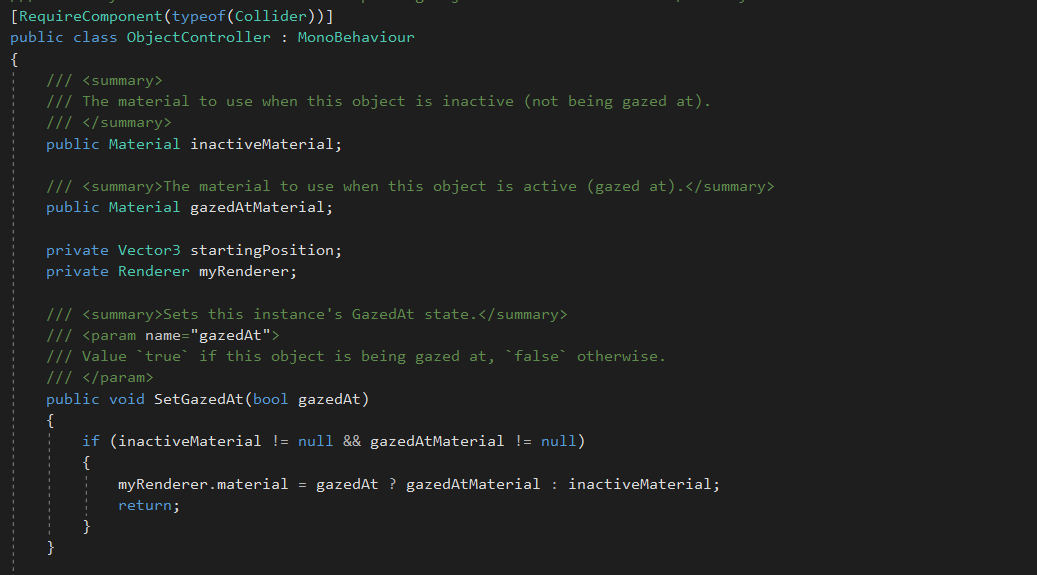


Fig C6. It is an object controller class.

This class manages all object in canvas. Like room ball. Touch response etc…

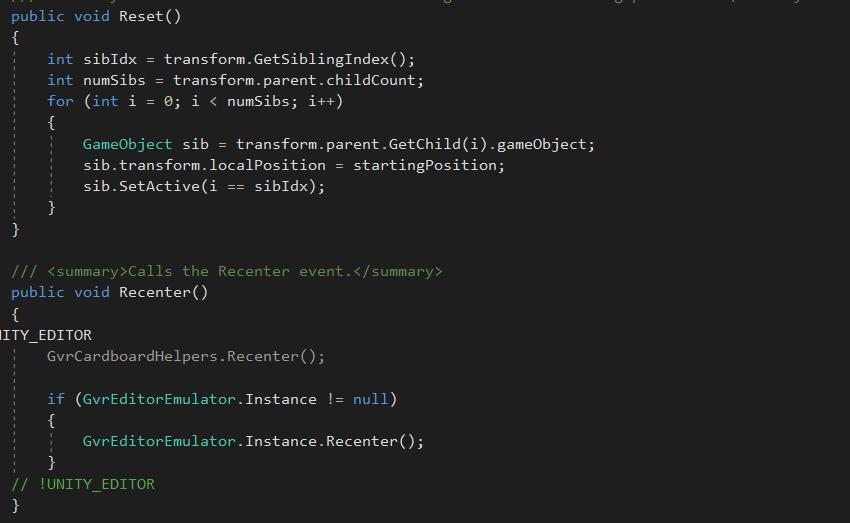


Fig C7. In this you can see two functions. 1.Reset: - so it reset all game object at that initial position. 2. Recentre: - It restart an event.

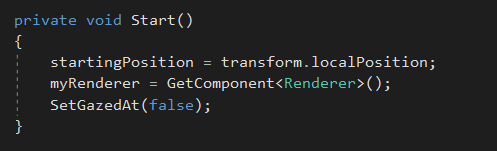


Fig C8. When application in launched then this start function is called and it initialized all game objects.

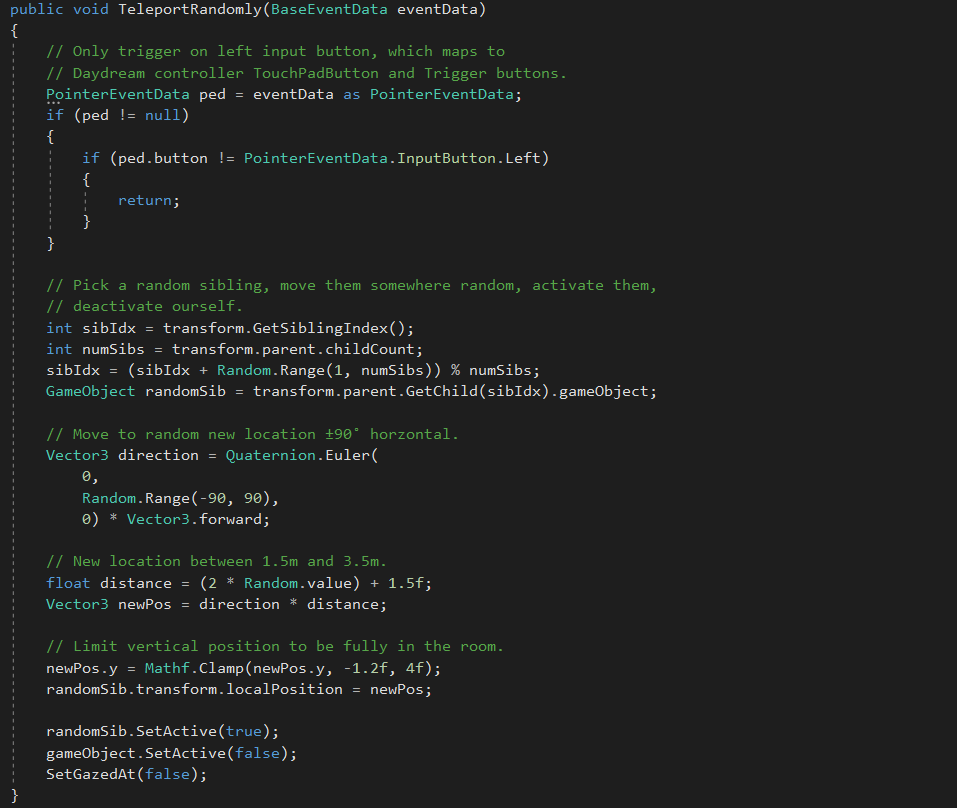


Fig C9. Object teleport function.

When we hit object(ball) it randomly places somewhere else and all these things are managed by this function.

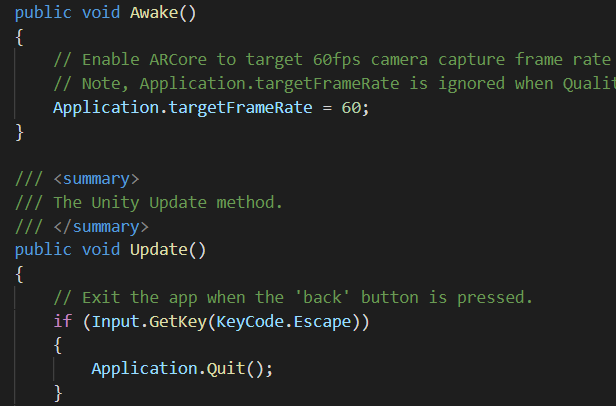


Fig F. Unity Life Cycle

Here awake method is like initialize method and after that continuously update method is being called implicitly.

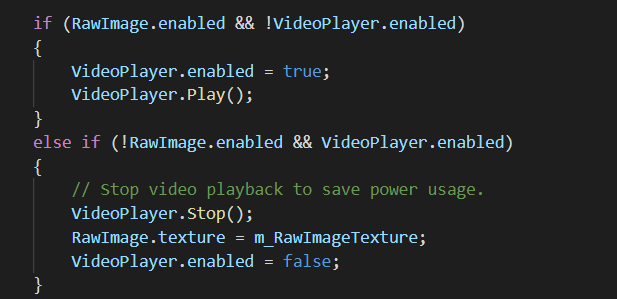
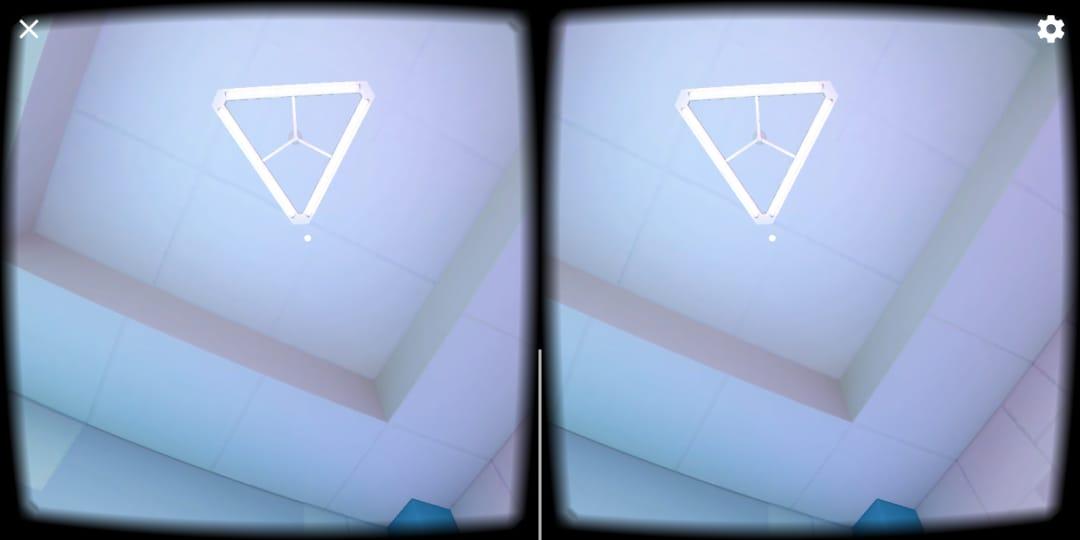
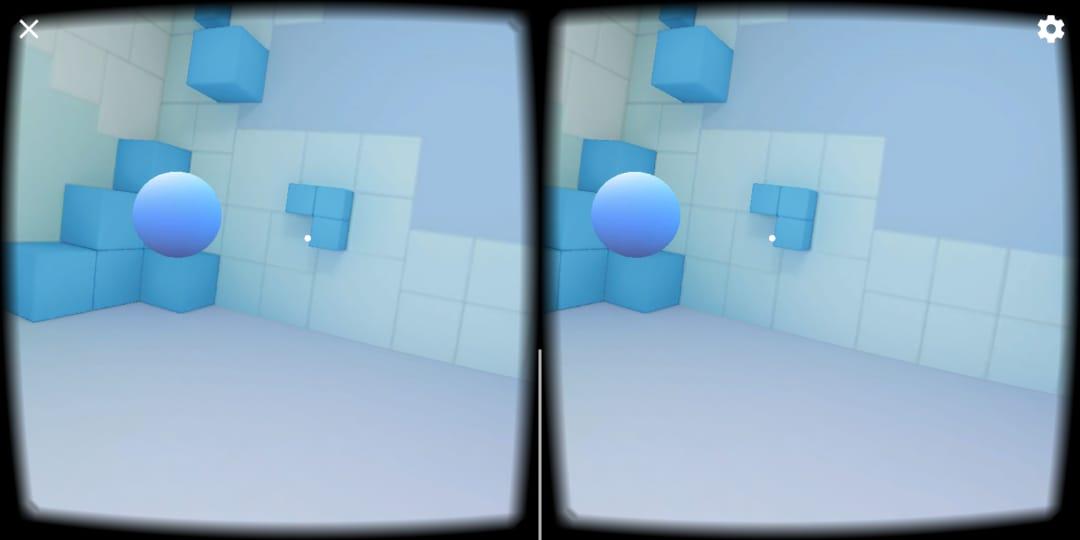


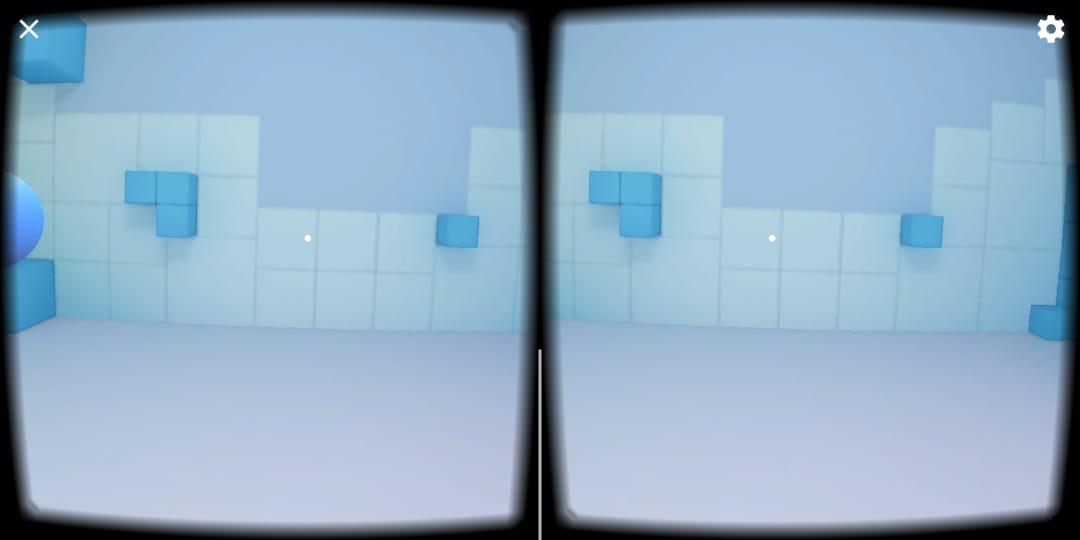
Fig G. Play/Pause Video in Live-Book

Proof of Working

***Shoot-It***

 9.A

 9.B

 9.C

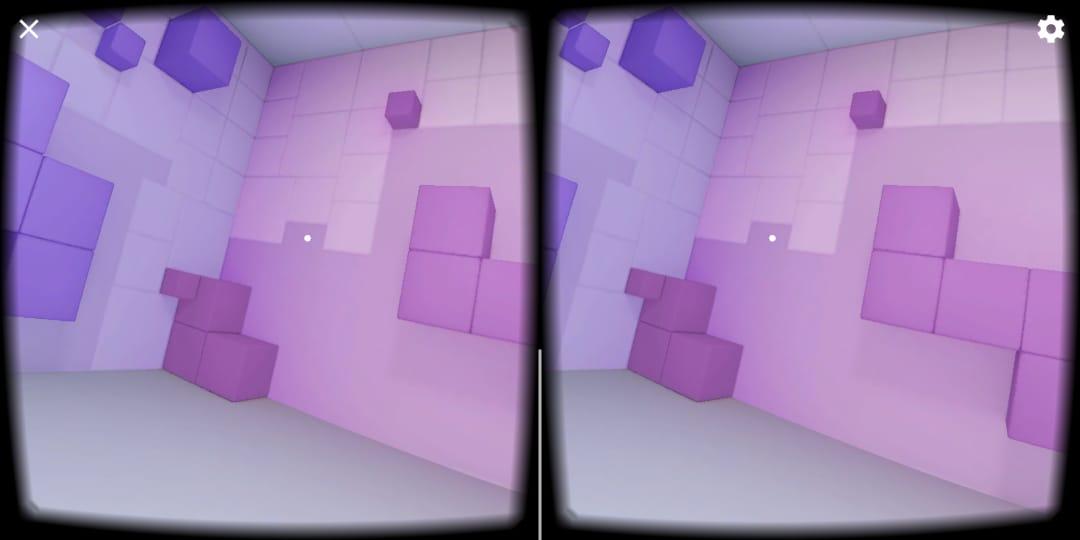
 9.D

Figure 9. Model of Shoot-It

Here in Figure-9 we can see that there is room and white ball is present. The white ball pop up randomly and user needs to shoot it and next ball appears and it continues till end.

***Treasure-Hunt***



Figure 10. Creation of Anchor Point and Adding Treasure/Clue

Immediately after creating room admin who has created room can add anchor point to the game and set up the clue in different room.



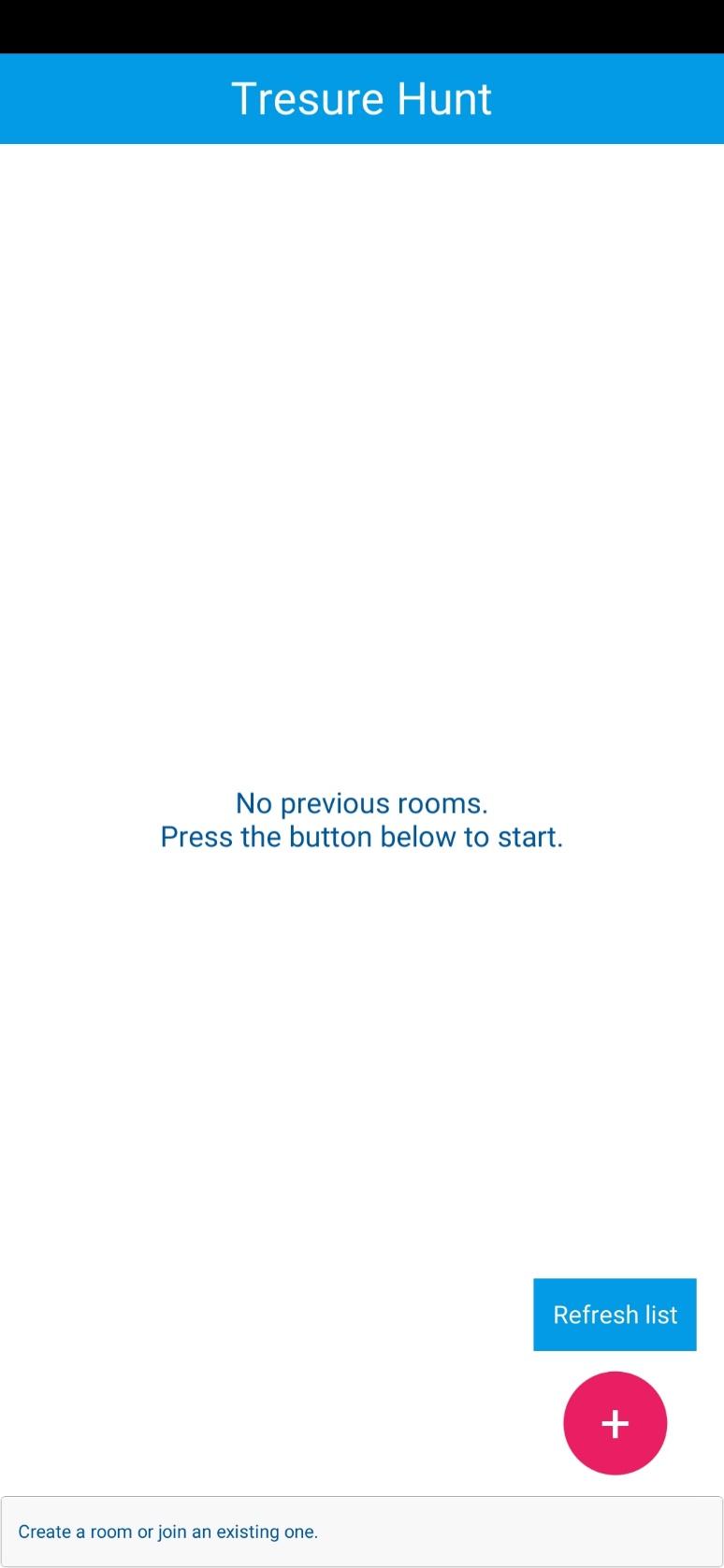
***Figure 11. Adding Clue to Treasure***

Here we add clue in the treasure. It is only possible after creation of anchor points. The clue and anchor point can only be added by admin, who have hosted the game. Others players cannot add them.



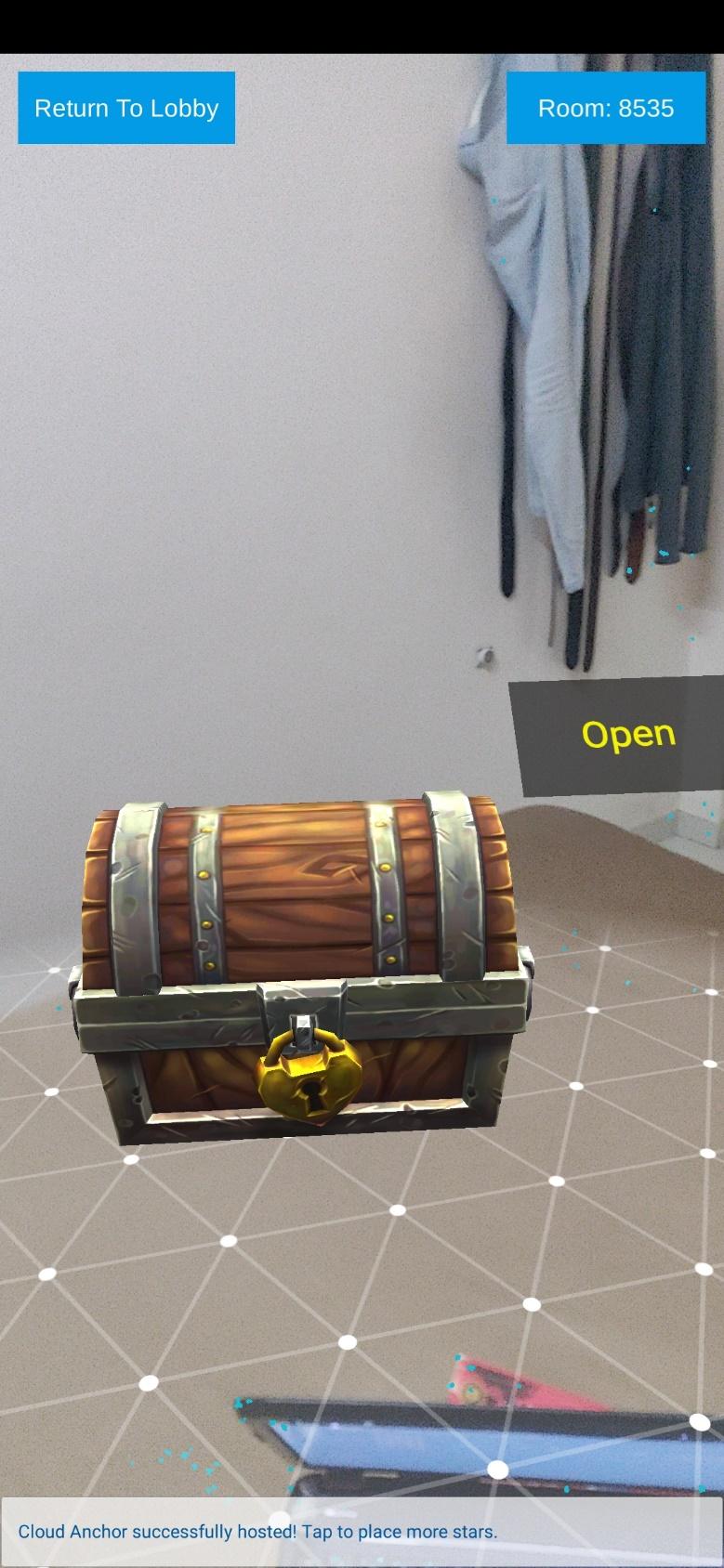
***Figure 12. Updating Clue of Treasure***

We are human so making mistake is part of our system, hence whenever admin entered wrong clue, they could update to clue by just simply tapping on treasure box and updating the text by entering correct text.



***Figure 13. Creating Room/ Joining Room***

It is home page/screen of Treasure Hunt. Here user can either create room by clicking on button with plus sign. Or join the room by refreshing the list and joining with appropriate id.



***Figure 14. Hunting Treasure (Client Side)***

After joining rooms client now need to scan anchor point and after they scan anchor point, they need to find clues by just walking in random direction or trusting your gut feeling.

**Working of Treasure Hunt!!**

Here in this game there are mainly two parties involved: Organizer/Host and other teams.

Here host sets up a treasure hunt trail i.e. they create clue which leads to another clue and it to other and so on. These continue until client reaches end.

**Setting Up Game**

1. Create Room
2. Create Anchor Point
3. Tap on screen to add treasure
4. Add clue in Treasure

**Playing Game**

1. Join Appropriate Room
2. Scan Anchor Point
3. Find Clues

* Clue only appears when user in certain distance of the clue (1 meter)

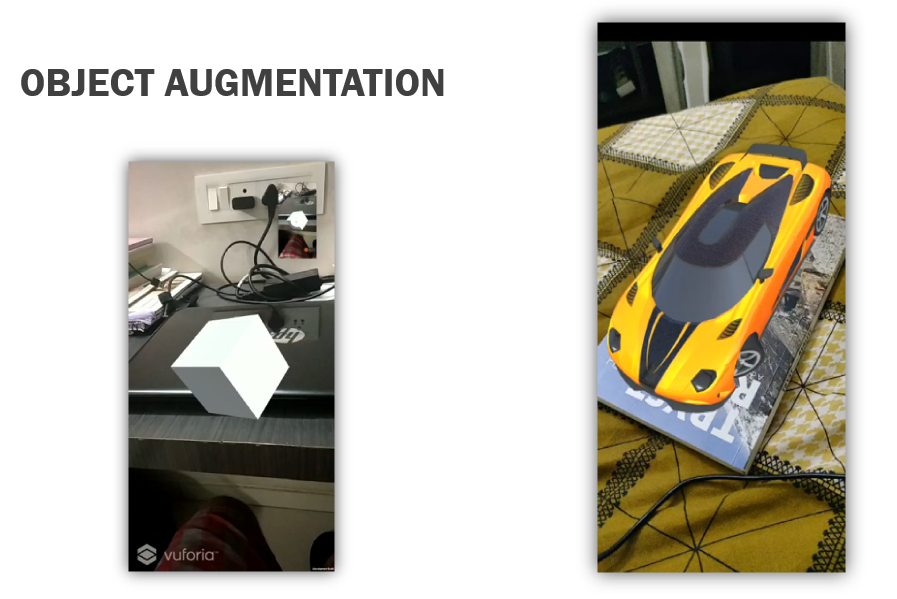


Figure 15. Object Augmentation

Here simply when screen starts the object (3D Model) pop up and user can see it.

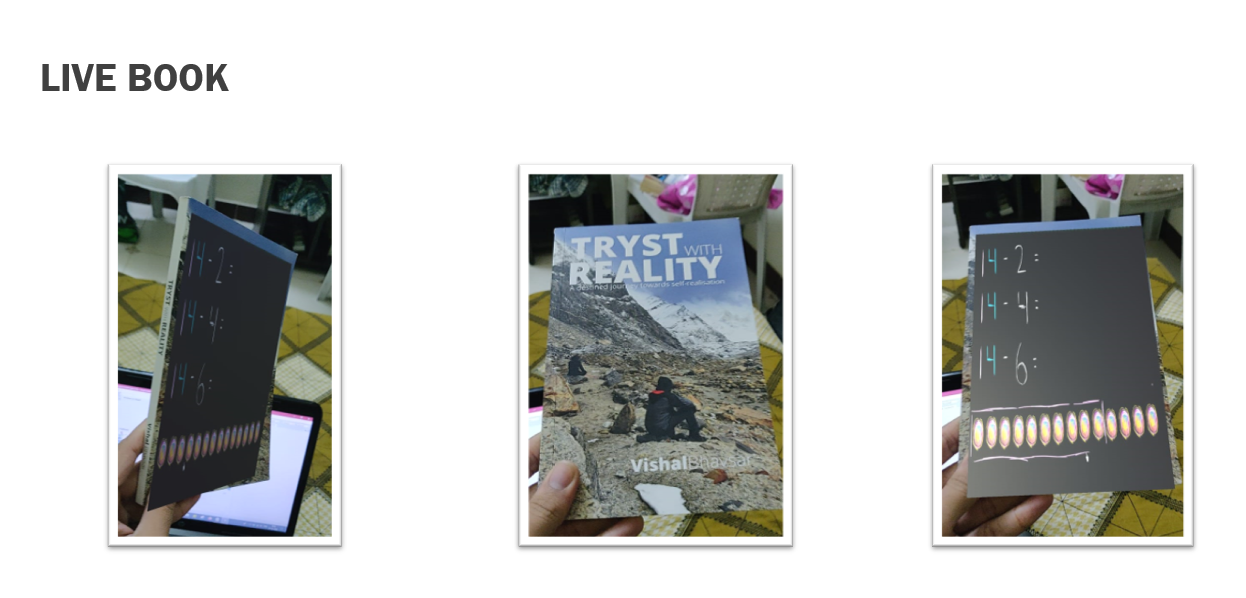


Figure 16. Live Book

Here whenever an appropriate image which is present in DB appears in front of camera after starting application then video pops up.

After tapping on video, it starts playing.

Video can be paused/resumed just tapping on video.

**FaceT**

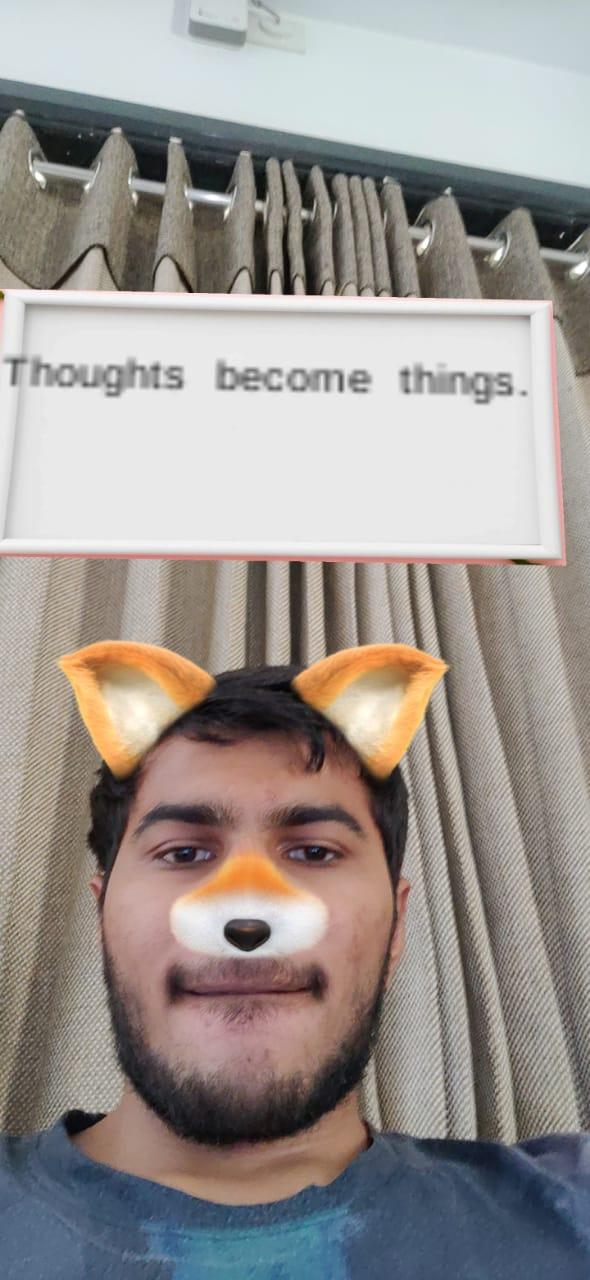


Figure 17. FaceT (Augmenting Thoughts)

Here after starting application whenever face is detected by front camera then their face is augmented with fox ears and random thoughts starts appearing and after a tap on it, it pauses.

It’s mostly fun app to display random thoughts.

**Conclusion**

By developing the uses case of Augmented Reality, we came across various difficulties, technologies and many various solutions to develop a thing using more efficient ways.

Problems Faced:

*In Live-Book:*

For a certain group of images, whenever we tried playing video based on it, different type of video would be rendered.

It was because those images didn’t have features (This are patterns present in image which is to be detected by Artificial Intelligence in Google AR core library)

Later after knowing this problem we allowed to add images having feature above 50 points, these points can be been detected within unity.

*In Kill-Shot:*

Here we had faced quite a challenge because of AR Core’s limited support. AR Core works only on Mobile and not on emulators with live camera.

So, we needed to improvised and that’s when we decided to integrate it with VR.

Here, now we didn’t require camera as everything (models) were switched to Virtual Background.

Now we could run it on emulators. However, it wasn’t the only problem, after switching to VR, we were supposed to sense direction in which user was moving and that was a quite the challenge. And then we learned about gyroscope in mobile and it was integrated with the help of google AR Core library into our project.

*In Treasure-Hunt:*

Initially in this application both client (teams) and host could add stars(clues). It was a conflict as both shouldn’t be able to add clues.

Hence, we needed a mechanism through which both the parties could be distinguished, and that when we divide our application into two types of users. After dividing it the problem was almost solve, as now we only needed to write a code containing traditional if/else

i.e. if admin then add clue, else don’t allow.

After solving this problem, we realised that out application needed to hide clues from client as they are visible from initial point.

Then again, we learned something new about AR Core Library, that it provides interface to detect length of object from mobile/camera.

With the help of it we were able to hide clues from teams.

We also found out that our application works well within two rooms.

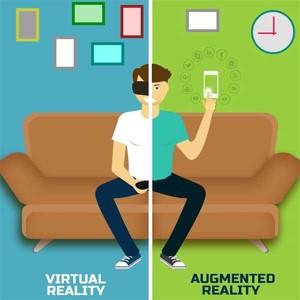
But if distance of clues from anchor points is increased slightly by 10 meters then application anchor points starts deviating. These could because some of the co-ordinates could be omitted while scanning in between in the plane. We researched, read document and learned that AR Core supports only limited range while other libraries such as AR Kit has longer range support. But then again AR-Kit is used to developed application of AR in Apple devices and we have been developing it in android.

Developing FaceT became quite easier as we had encountered all the hurdles while developing other application.

On the way of developing AR apps, we learned two new concepts VR and MR.

VR – Virtual Reality

MR – Mixed Reality



In VR there is virtual space which may corresponds to real world and on the basis of map/theme/scene user can virtualize and simulate real environment.

For example, there are some application which lets doctors simulate the behaviour of human body and they could perform a critical operation on simulation and it could life saviour and helps doctors prepare for original operation and ultimately minimize the risk.

Mixed Reality is combination of VR and AR.

These technologies will revolutionize the modern world in every aspect. From children to industries every single thing/being may become dependent on it. Nowadays children use AR to learn new things and also it makes easier for them to perceive things.



Figure 18. Children using AR to learn globe.

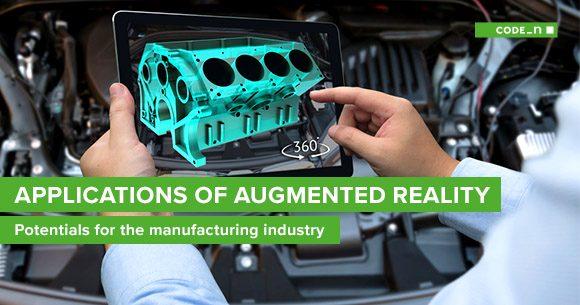


Figure 19. Use of AR in industry to visualize engine

Augmented Reality with artificial intelligence could be the next big step of the century because of its variety of application.

**Limitation and Future Extension**

We have developed use case and they are limitation itself because they individual but along with some real application it has great potential.

For example, if FaceT is developed as service than it could be used to target user on social media so that they can have fun

Apart from it in treasure hunt we have limited range of room and after a certain range the objects starts displacing.

And in Live Book we don’t feature where we could add new video for image dynamically without recompiling or re-analysing all the images, detection time for first time is unusually higher because of initialize of heavy database into main memory and also the video doesn’t remain steady.

All the above limitation could be Future Extension.

**References**

<https://developers.google.com/ar>

<https://library.vuforia.com/content/vuforia-library/en/reference/unity/index.html>

<https://docs.unity3d.com/Manual/index.html>

<https://www.coursera.org/search?query=ar&>

Note: Some of the image were taken from Shutterstock.

<https://www.shutterstock.com/?kw=image%20website&msclkid=66e5d9d57f701967f8268772a5e445ff&utm_source=bing&utm_medium=cpc&utm_campaign=IN_en_image_generic&utm_term=image%20website&utm_content=image_website_exact&gclid=CNLitfrLiukCFWPN1AodULEOPw&gclsrc=ds>