FYP-II Final Report

Project Name: Augmented Reality based Tactical Simulator

Project ID: F21-03-D-ARTS

Supervisor

Dr Adnan Tariq

Project Members

18I-0504 Rao Ubaid Ullah

18I-0462 M. Tayyab Ejaz

18I-0517 Tanzeel Ahmed

Anti-Plagiarism Declaration

This is to declare that the above FYP report was produced under the:

Title: Augmented Reality based Tactical Simulator

is the sole contribution of the author(s) and no part hereof has been reproduced on **as it is the** basis (cut and paste) which can be considered as **Plagiarism.** All referenced parts have been used to argue the idea and have been cited properly. I/We will be responsible and liable for any consequence if a violation of this declaration is determined.

Date: October 22, 2021

Student 1

Name: Rao Ubaid Ullah

Signature:_____

Student 2

Name: M. Tayyab Ejaz

Signature:_____

Student 3

Name: Tanzeel Ahmed

Signature:_____

Supervisor (Faculty)

Name: Dr. Adnan Tariq

Signature:_____

Table of Contents

1. Project Vision

1.1 Problem Statement

Augmented Reality (AR) technology allows the combination of the real world with computer-generated images (CGI). This technology can enable endless possibilities in different real-life applications but it is not mainstream yet.

The training process of police officers and special tactical forces can benefit from such technology. Currently, the training process is carried out using man-made courses, props and other officers roleplaying as perpetrators. There is a gap in performance analysis and assistance provided to the officers that can be filled using AR technology.

Apex Officer[1] is a Virtual Reality (VR) based training simulator for police officers that enables the trainees to train in different scenarios. The training is analysed by a senior officer in real-time as well as in post-simulation. However, this technology is not widely used in different police departments and using VR has its downsides. Most people get nauseated very quickly and long term use of VR is very uncomfortable for them. Another downside of VR is that it cannot be implemented in a way to be used in real-life missions.

1.2 Business Opportunity

The basic idea of our application is unique and new, but the AR technology is not mature enough to be used in training simulations yet. However, in the coming years, as AR technology improves, there is a possible market gap that can be filled by this application. The Unique Selling Proposition will be the advantage it has over other VR based training simulators, and no competition in the market.

1.3 Objectives

The main objective is to develop a training simulator for search and rescue forces. The simulation includes a team of multiple members in various scenarios. Team members will communicate with each other using call outs, tagging in 3d space, and can view the location of their allies. Each member is part of the virtual arena connected to a single lobby. A lobby can also have observers analysing the performance of players. A player can see a 3D or 2D view of the arena through their device. AR mesh generation will enable team members to share their 3D views. The simulator will cover different types of scenarios like rescue operations and arena sweep. Post-training analysis will be provided through the app where the performance of each player can be analysed and their responses to callouts can be judged.

1.4 Project Scope

The project is aimed at building an android application using Unity and ARFoundation. The app is targeted towards scenario-based training of search and rescue forces and will not include/guarantee reliable performance in real-world missions. The mentioned app is not a GAME, so it will not be including any scoring, player health, multiple levels, story features. The app is a simulator and will only assist in training by providing different scenarios created by the host. The rules of training would still apply in the real world. For example, if training includes the use of paintball/airsoft guns, players getting shot would have to leave the simulation. Our app wouldn't be keeping track of that. Other such rules are supposed to be followed in the real world, and our AR app wouldn't be catering to those rules.

1.5 Constraints

- Due to inaccuracies in tracking, and error accumulation over long distances in AR, the arena size must be limited to a small area (unless improvements are made to the tracking algorithms). After testing the limitations of the technology, we found out that the AR session origin is prone to errors because of inaccuracies. So there might be some weird results in synchronization between multiple players. Our approach to this problem is discussed in the technical challenges section.
- Since we are aiming for scenarios based training, it only makes sense to have some kind of objectives for trainees to collect or discover. However, AR is not yet capable of generic object recognition (unless a machine learning model is implemented). For this purpose, we intend to use printed images as markers. A marker for AR is like a specific pattern that it looks for in the camera feed. If that pattern is found, logic can be implemented to count it towards the progress of that team. This type of AR is called Marker-based AR[2].

1.6 Stakeholders Description

1.6.1 FYP Group Information

Member	Name	Roll No	CGPA	Email
1	Rao Ubaid Ullah	18I-0504	3.53	i180504@nu.edu.pk
2	M. Tayyab Ejaz	18I-0462	3.51	i180462@nu.edu.pk
3	Tanzeel Ahmed	18I-0517	3.49	i180517@nu.edu.pk

1.6.2 FYP Supervisor

Dr. Muhammad Adnan Tariq

1.6.3 Search and Rescue Forces

- Trainees
- Trainer

1.6.4 Key High-Level Goals

Trainee:

1. Move around in the arena and have the application track different interest points and mark their locations.
2. See their allies position, markers placed by their team in the 3D Space.
3. Place tags or callouts in the real world through the application.
4. Detect markers placed by the host and perceive progress.

Trainer:

1. Setup markers throughout the arena and in the application.
2. Setup the simulation.
3. Monitor performance of trainees after the simulation.

2. Iteration Plan

3. Iteration 1

3.1 User Stories

3.1.1 User Story 1

Name: Detect Planes in Real World

Story: As a user, I want to detect and view planes of the real world so that I can view the real world in AR view.

3.1.2 User Story 2

Name: Place 3D objects on view

Story: As a user, I want to place 3D objects on view planes so that I can interact with the real world using different objects.

3.1.3 User Story 3

Name: Scan Marker for Interaction

Story: As a user, I want to scan AR Markers in the real world so that I can keep track of my progress throughout the simulated scenario.

3.2 Use cases

3.2.1 Use case 1

Use Case:	Detect Planes in Real World
Actors:	Trainee
Type:	Primary
Description:	The trainee joins the arena, where he can detect and view the planes detected by the application augmented into the real world on the screen.

3.2.2 Use case 2

Use Case:	Place 3D Object on view
Actors:	Trainee
Type:	Primary
Description:	The trainee joins the arena, where he can place different 3d objects on view planes in the arena.

3.2.3 Use case 3

Use Case:	Scan Marker for Interaction
Actors:	Trainee
Type:	Primary
Description:	When the trainee joins the arena, he wants to scan AR Markers in the real world so that he can keep track of my progress throughout the simulated scenario.

3.3 Extended Use Cases

3.3.1 Extended Use Case 1

Use Case Name: Detect Planes in Real World

Actor: Trainee

Level: User-Level Goal

User Story: As a user, I want to detect and view planes of the real world so that I can view the real world in AR view.

Preconditions: ARCore and application must be installed on the phone.

Postconditions: The application will be able to perform augmented functionality on these 3D planes.

Special Requirement: The mobile device is AR-enabled.

Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera or mobile turns off.

Extended Use Case:

Actor	System
1. Open AR App	
2. Scan view by Camera	
	3. Detect and Create 3D planes for view
	4. Display them on Camera view

3.3.2 Extended Use Case 2

Use Case Name: Place 3D objects on view

Actor: Trainee

Level: User-Level Goal

User Story: As a user, I want to place 3D objects on view planes so that I can interact with the real world using different objects.

Precondition: 3D view planes are generated.

Postcondition: The user will be able to view and interact with these 3D objects.

Special Requirement: The mobile device is AR-enabled.

Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera or mobile turns off.

Extended Use Case:

Actor	System
1. Open AR App	
2. Scan view by Camera	
	3. Create 3D planes for view
	4. Display planes on Camera view
5. Place the 3D object on view	
	6. Spawn 3D object and place on user-specified coordinate

3.3.3 Extended Use Case 3

Use Case Name: Scan Marker for Interaction

Actor: Trainee

User Story: As a user, I want to scan AR Markers in the real world so that I can keep track of my progress throughout the simulated scenario.

Precondition: 3D view planes are generated.

Postcondition: AR Markers are scanned and placed throughout the arena.

Special Requirement: The mobile device is AR-enabled.

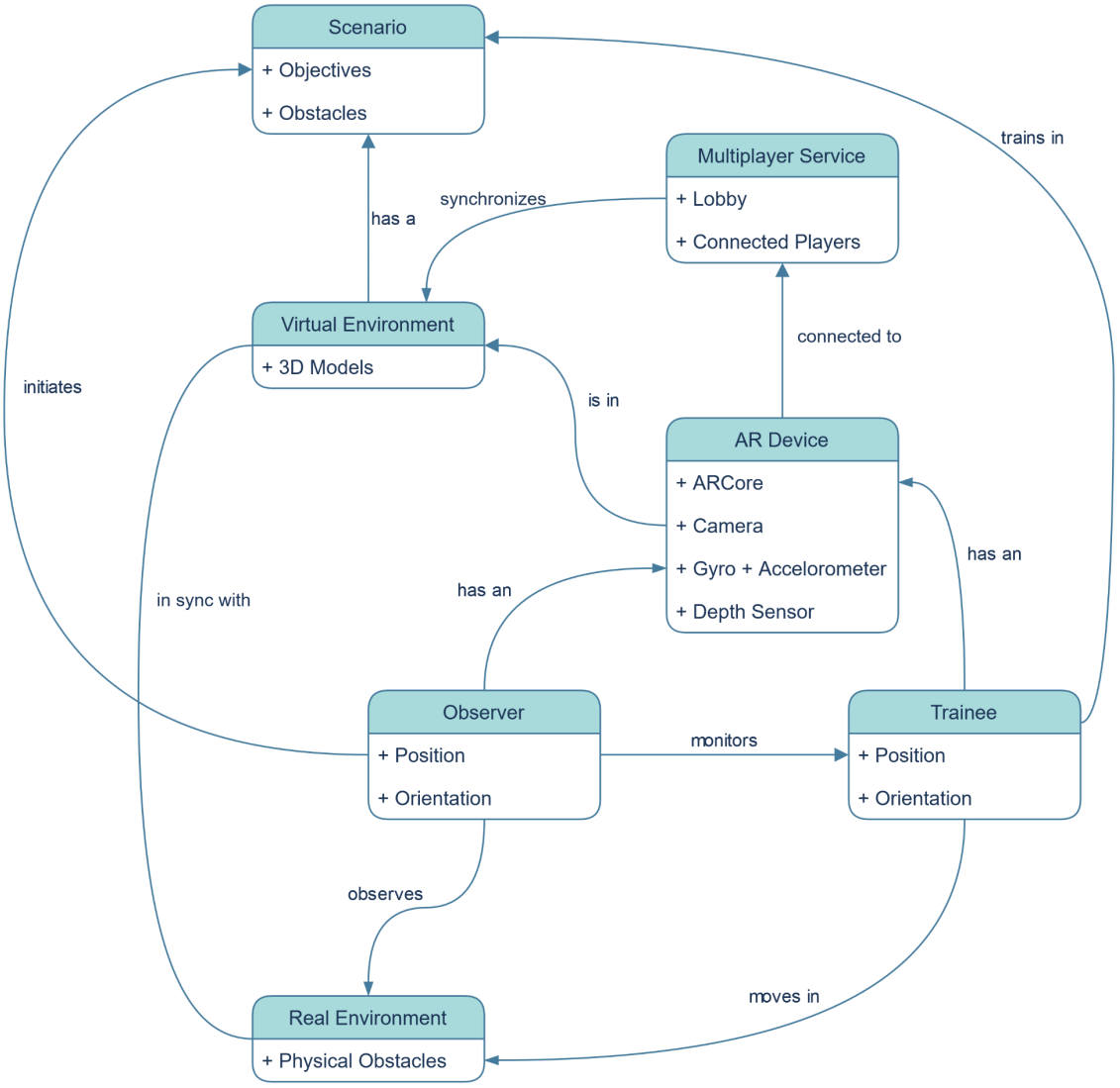
Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera, or mobile turns off.

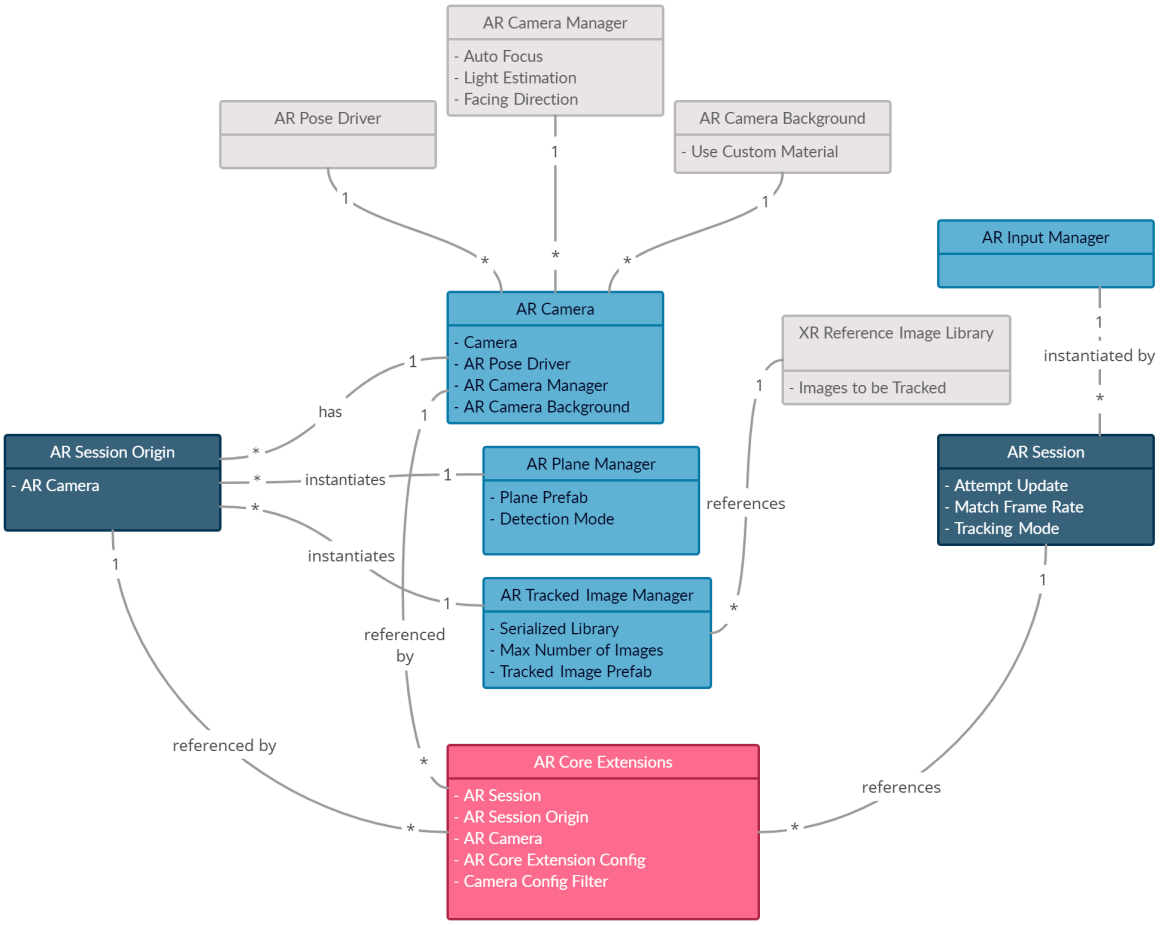
Extended Use Case:

Actor	System
1. Open AR App	
2. Scan AR marker	
	3. Interpret AR marker
	4. Display interaction based on marker

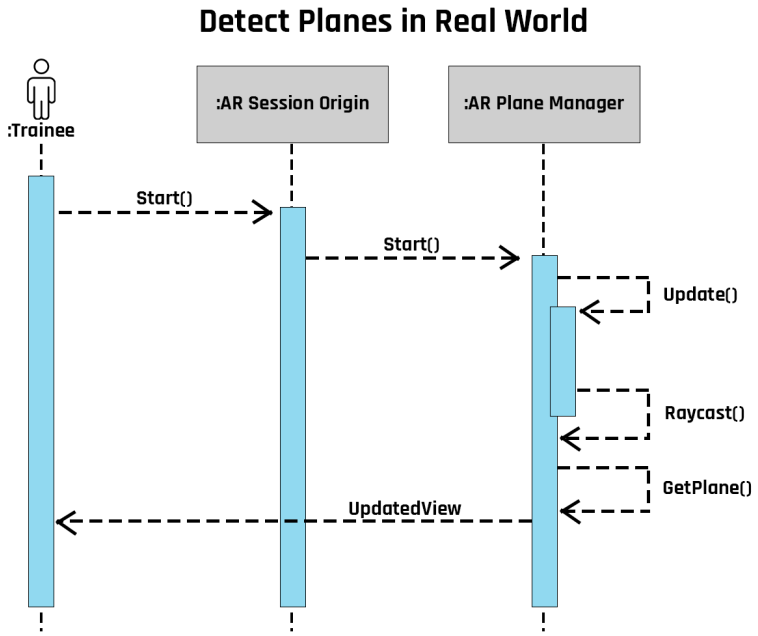
3.4 Domain Model

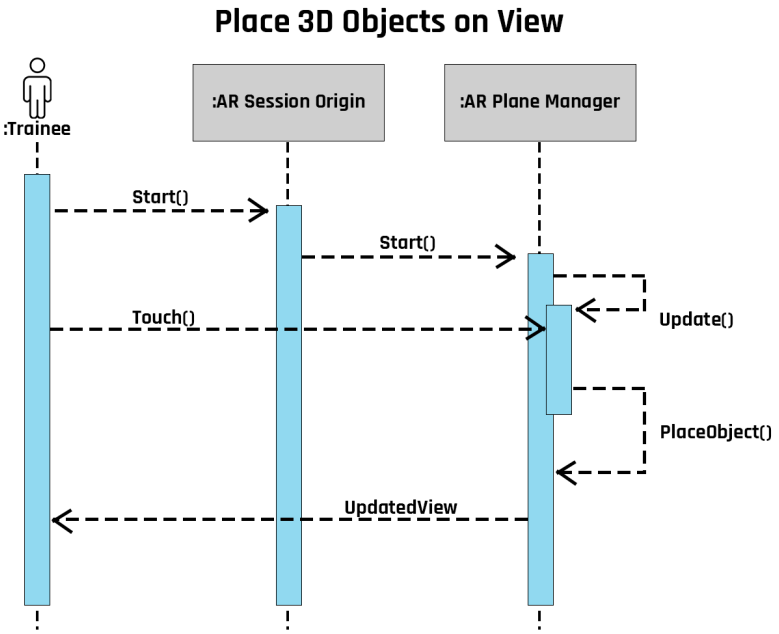


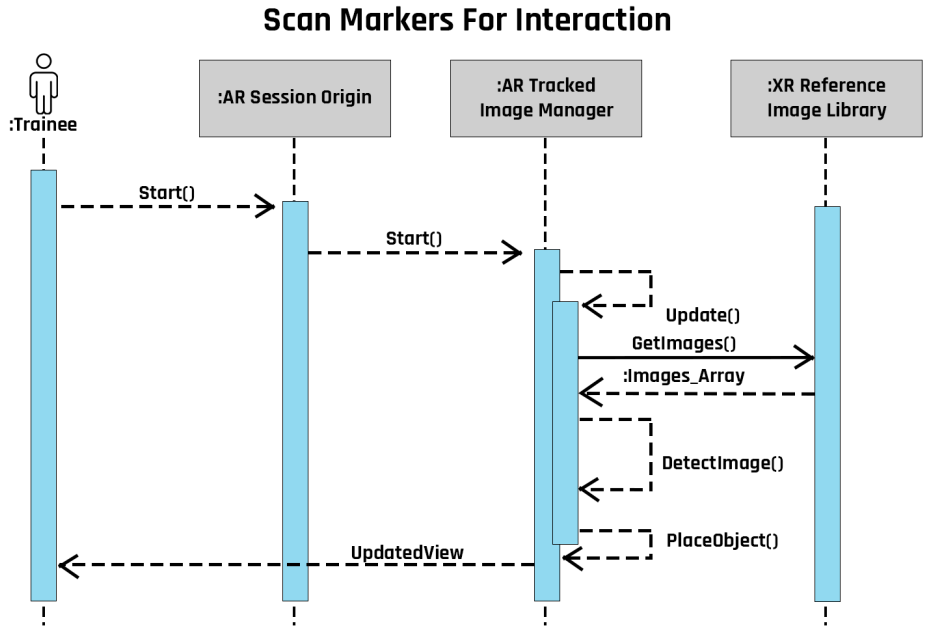
3.5 Class Diagram



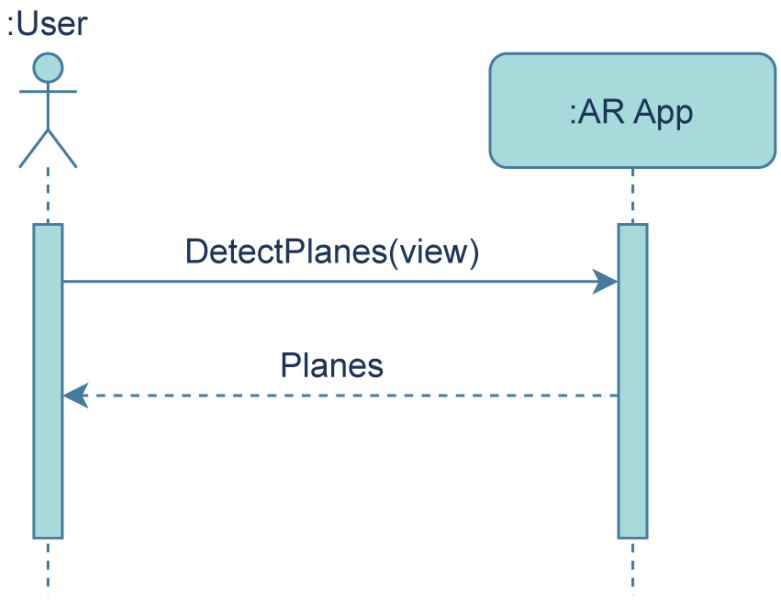
3.7 Sequence Diagram

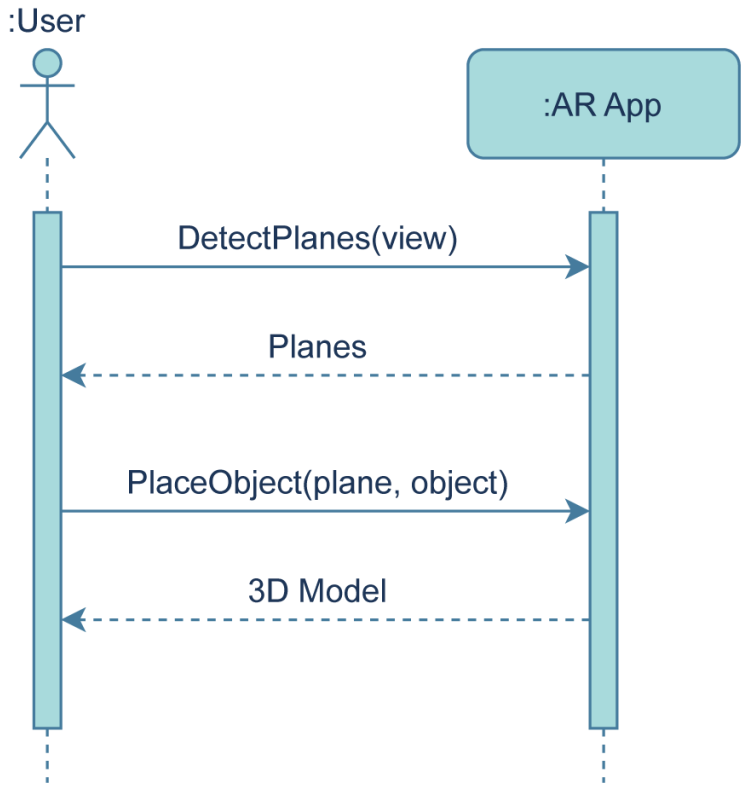


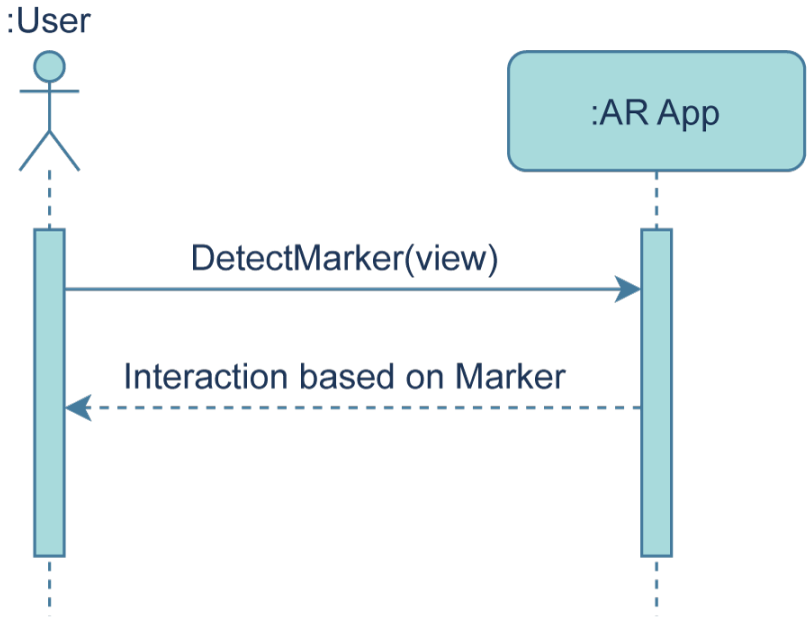

Place 3D Objects on View
:Trainee
:AR Session Origin
:AR Plane Manager
Start()
Start()
Touch()
Update()
PlaceObject()
UpdatedView


Scan Markers For Interaction
:Trainee
:AR Session Origin
:AR Tracked Image Manager
:XR Reference Image Library
Start()
Start()
Update()
GetImages()
:Images_Array
DetectImage()
PlaceObject()
UpdatedView

3.8 System Sequence Diagram






:User
:AR App
DetectMarker(view)
Interaction based on Marker

3.9. Software Requirement Specifications

3.9.1 List of Features

- Augmented Reality
- Multiplayer
- Scenario-based Simulations

3.9.2 Functional Requirements

- The user should be able to tag real-world location
- The user should be able to see team members in AR view
- The user should be able to interact with the environment by scanning markers
- The user should be able to train in a simulated environment by achieving assigned objectives

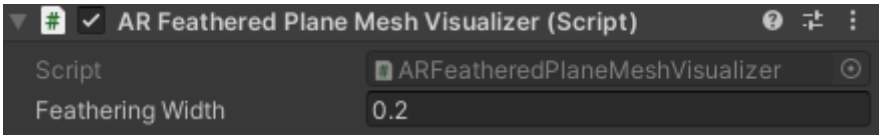
3.9.3 Non-Functional Requirements

- Performance
- Usability
- Efficiency
- Correctness

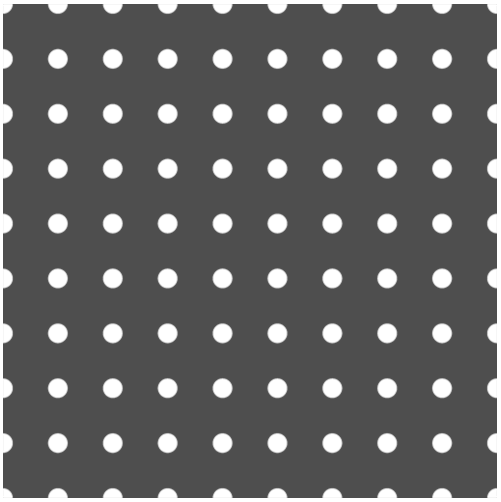
3.10 Work done

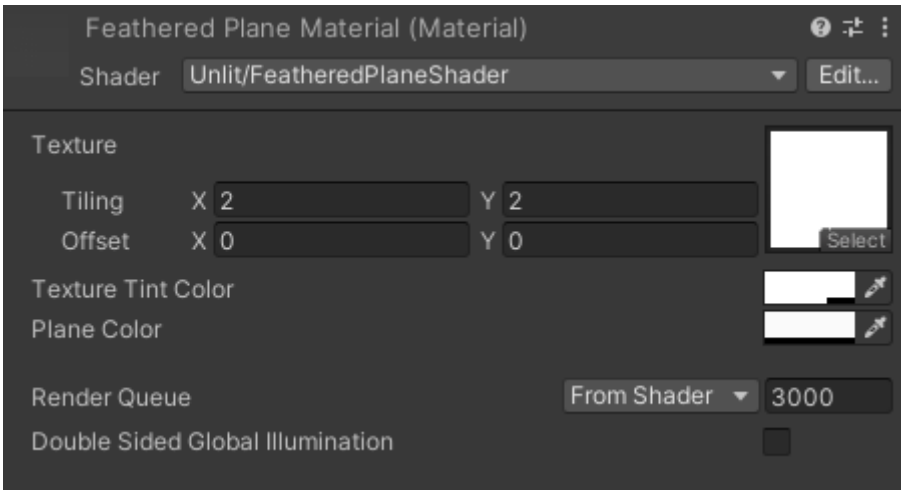
3.10.1 Plane Detection

The first task we did for this iteration was to detect Planes in the real world. For that, we used Unity AR Plane Manager to assist with plane creation. It casts rays on the view and detects a collision. These collisions are then translated to planes. These planes are later visualized in the view using ARFeatheredPlaneMeshVisualizer Script.



Material for the plane visualizer was created using the following texture image and shader settings.





The final output after the plane detection was as such.

3.10.2 Object Placement

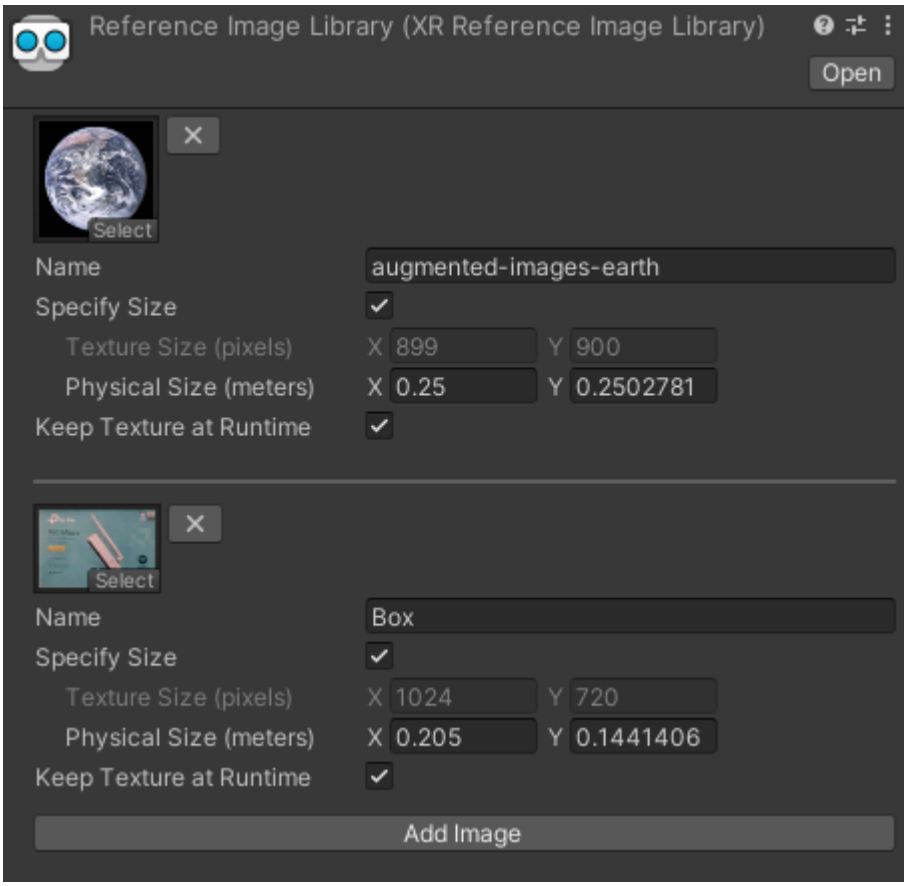
Once the planes are detected in the scene, we use the AR Raycast Manager with our AR Interaction script to place a 3D Land mine into the environment. The mine would be placed anywhere in the scene where the user touches the phone.

Here is the final output:

3.10.3 Marker Detection

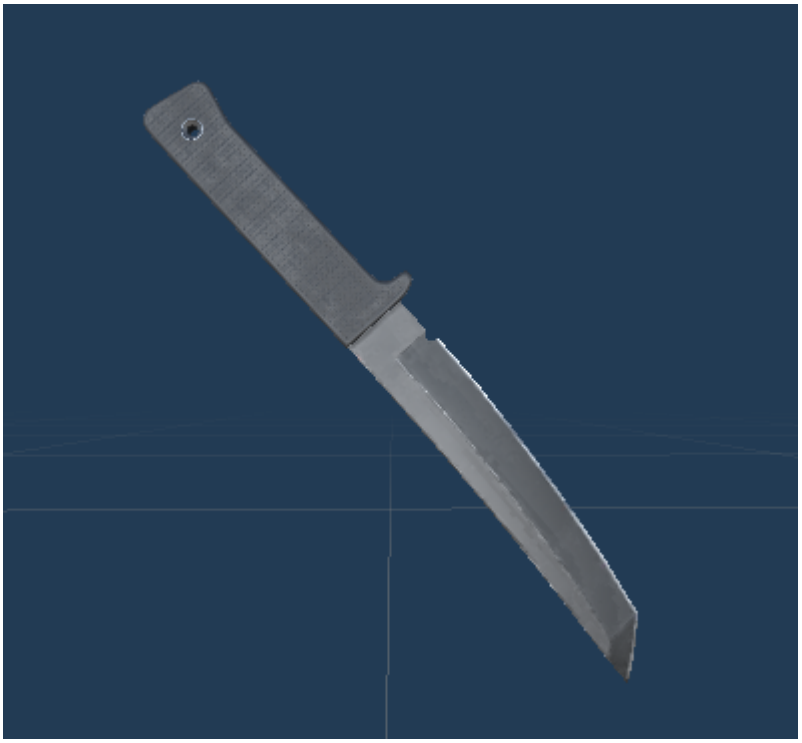
After the plane detection and AR interaction, we created an XR Reference Image Library with a few images taken from different boxes and then added that to AR Tracked Image Manager. While the App is working, if the given images are detected in the scene (at least covering 25% of the camera feed) will spawn a knife prefab on it.

XR Reference Image Library is set up as follows:

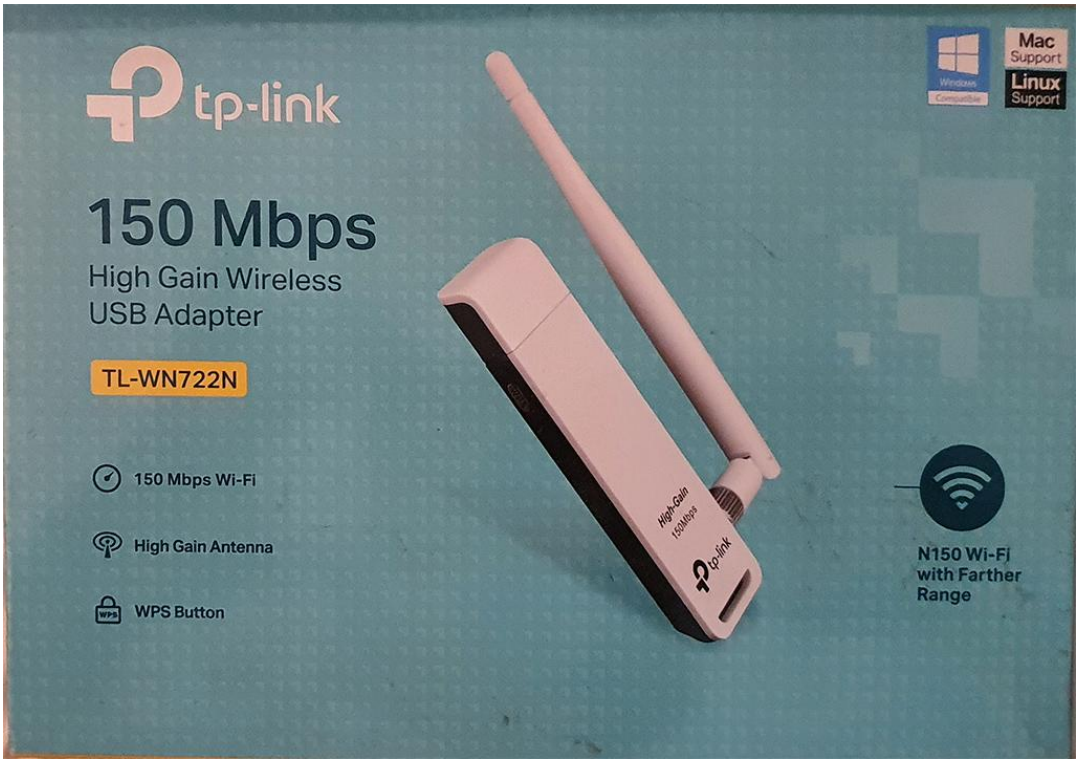


The image's physical size was also provided for accurate tracking in the application.

The prefab to be placed over the tracked image is:



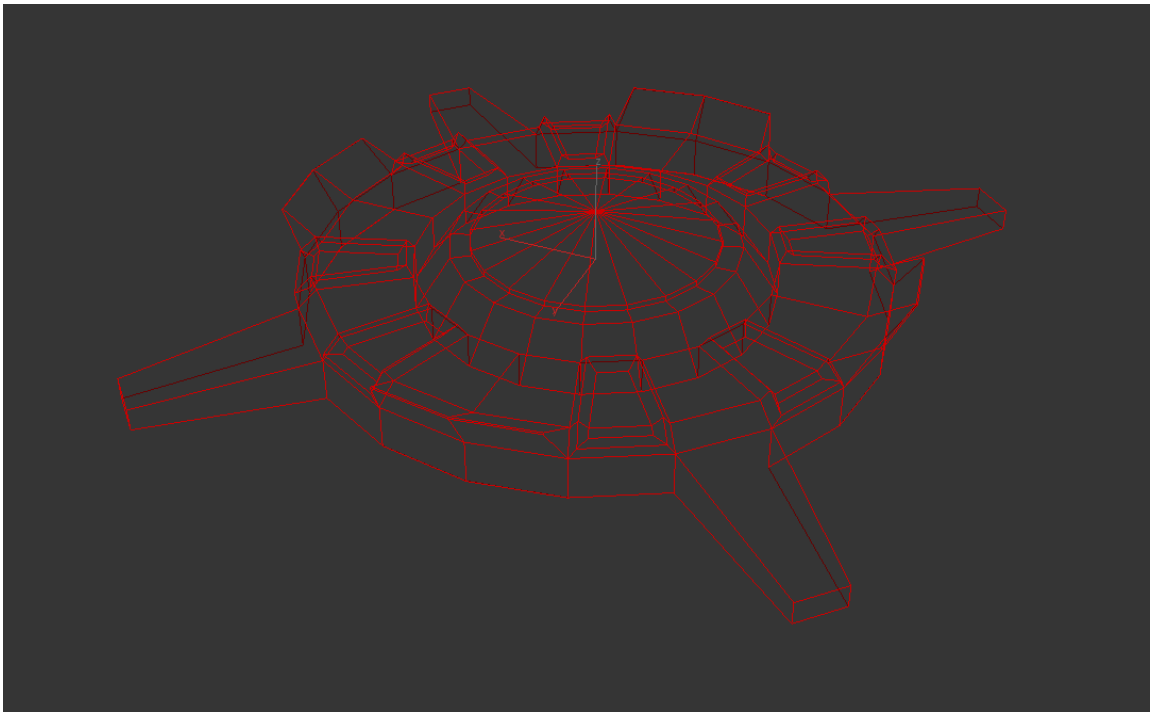
The image being tracked and the final output are as followed:



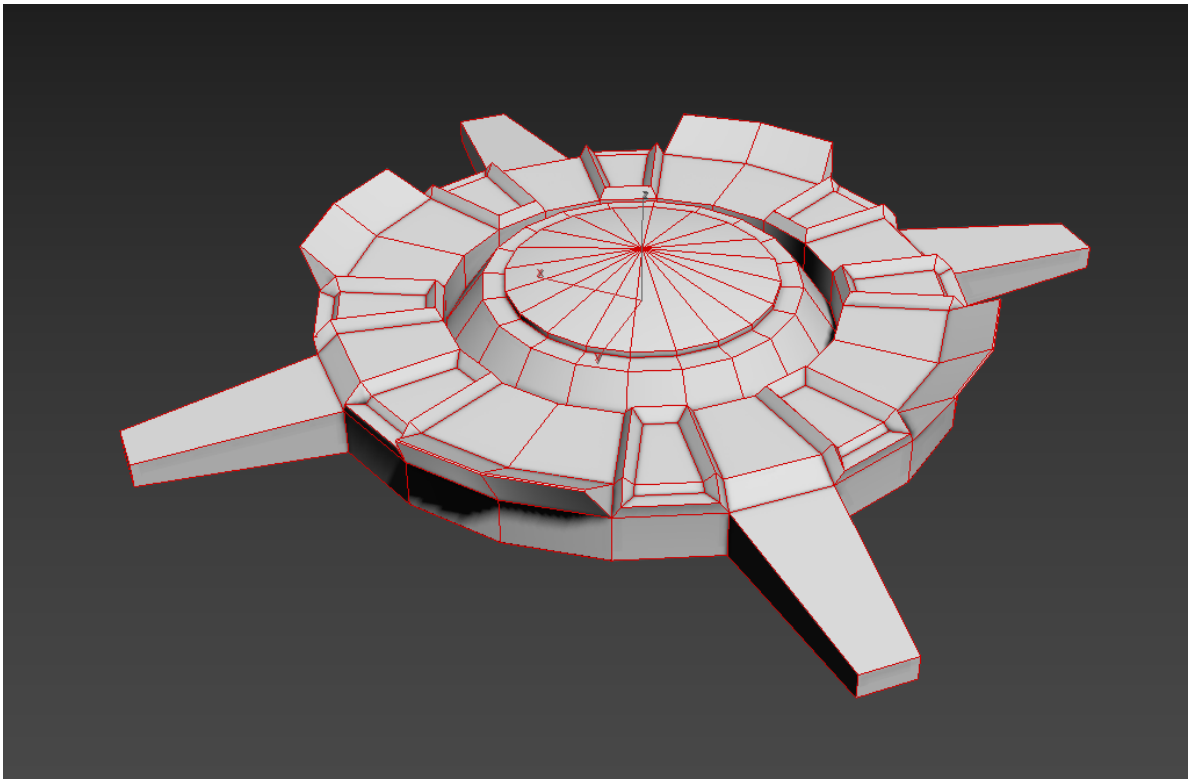
3.10.4 3D Assets Modeling

After completing the above-mentioned work, we moved on to the modelling process of 3d assets. We created the 3d model of the landmine using 3ds max.

Wireframe view of landmine:

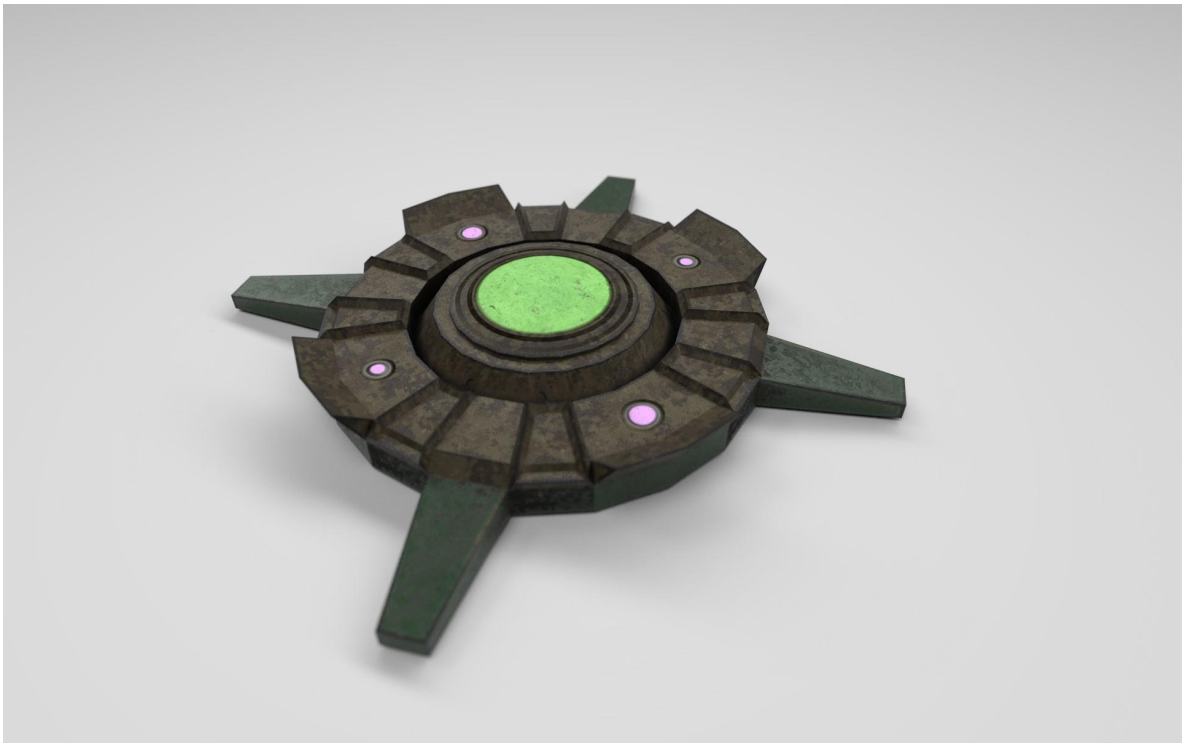


Realistic + wireframe view of landmine:



Then, we painted the land mine using the substance painter.

Textured landmine rendered in a studio environment:

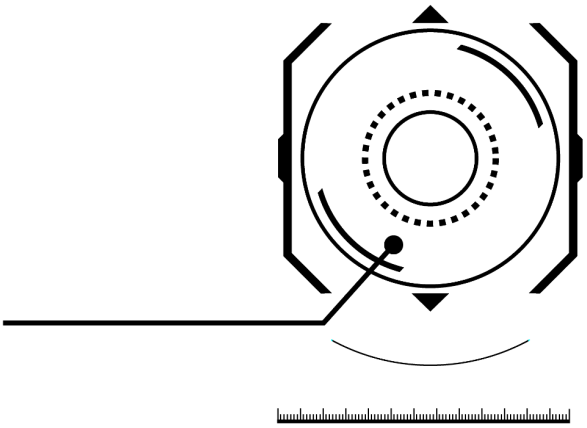


Textured landmine rendered in an outdoor scene:

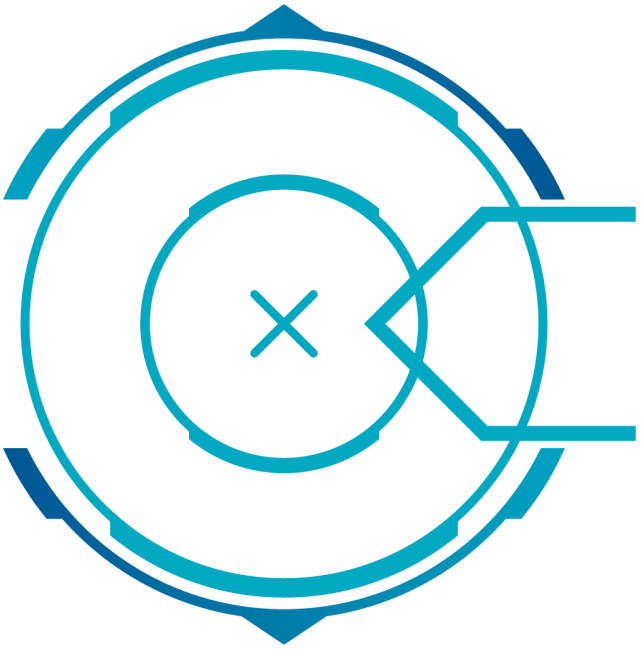
3.10.5 2D Assets Design

For use in later iterations, we designed some 2D UI elements in photoshop and illustrator. These assets are going to be used as tags placed by the trainees during a simulation.

3.10.5.1 Asset# 1



3.10.5.2 Asset# 2



We expect the 2D assets to be used as shown:


3.1 METERS
OBJECT TAGGED
OBJECT TAGGED
2.9 METERS

4. Iteration 2

4.1 User Stories

4.1.1 User Story 1

Name: Tagging Locations

Story: As a user, I want to tag locations to give callouts to my team about certain objectives or mark the objects identified in the arena.

4.1.2 User Story 2

Name: Viewing distance to tags

Story: As a user, I want to view my distance from the tagged location to decide my next move.

4.1.3 User Story 3

Name: Variation in tags

Story: As a user, I want to be able to tag locations with different types of tags based on what type of object is being targeted.

4.2 Use cases

4.2.1 Use case 1

Use Case:	Tagging Location
Actors:	Trainee
Type:	Primary
Description:	The trainee can tag different locations in the arena to give callouts to other teammates about certain objectives or mark the objects identified in the arena.

4.2.2 Use case 2

Use Case:	Viewing distance to tags
Actors:	Trainee
Type:	Primary
Description:	The trainee joins the arena, where he can place different 3d objects on view planes in the arena.

4.2.3 Use case 3

Use Case:	Variation in tags
Actors:	Trainee
Type:	Primary
Description:	The trainee joins the arena and he will be able to tag locations with different types of tags based on what type of object is being targeted.

4.3 Extended Use Cases

4.3.1 Extended Use Case 1

Use Case Name: Tagging Locations

Actor: Trainee

Level: User-Level Goal

User Story: As a user, I want to be able to tag locations with different types of tags based on what type of object is being targeted.

Preconditions: Plane is detected

Postconditions: Places a 2D or 3D object in that location

Special Requirement: The mobile device is AR-enabled.

Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera or mobile turns off.

Extended Use Case:

Actor	System
1. Trainee clicks on some object in the arena.	
	2. System places a tag on that location.
	3. Display tags on camera when camera is scanning that area.

4.3.2 Extended Use Case 2

Use Case Name: Viewing distance to tags

Actor: Trainee

Level: User-Level Goal

User Story: As a user, I want to view my distance from the tagged location to decide my next move.

Precondition: A tagged location is in camera view.

Postcondition: Display distance of the user from the tagged location.

Special Requirement: The mobile device is AR-enabled.

Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera or mobile turns off.

Extended Use Case:

Actor	System
	1. Calculate distance to the tagged location.
	2. Display distance to the tagged location
3. Trainee can see his distance from the tagged location	

4.3.3 Extended Use Case 3

Use Case Name: Variation in tags

Actor: Trainee

User Story: The trainee joins the arena and he will be able to tag locations with different types of tags based on what type of object is being targeted.

Precondition: Camera view contains one or more tags

Postcondition: Specific tag is placed in the 3D environment.

Special Requirement: The mobile device is AR-enabled.

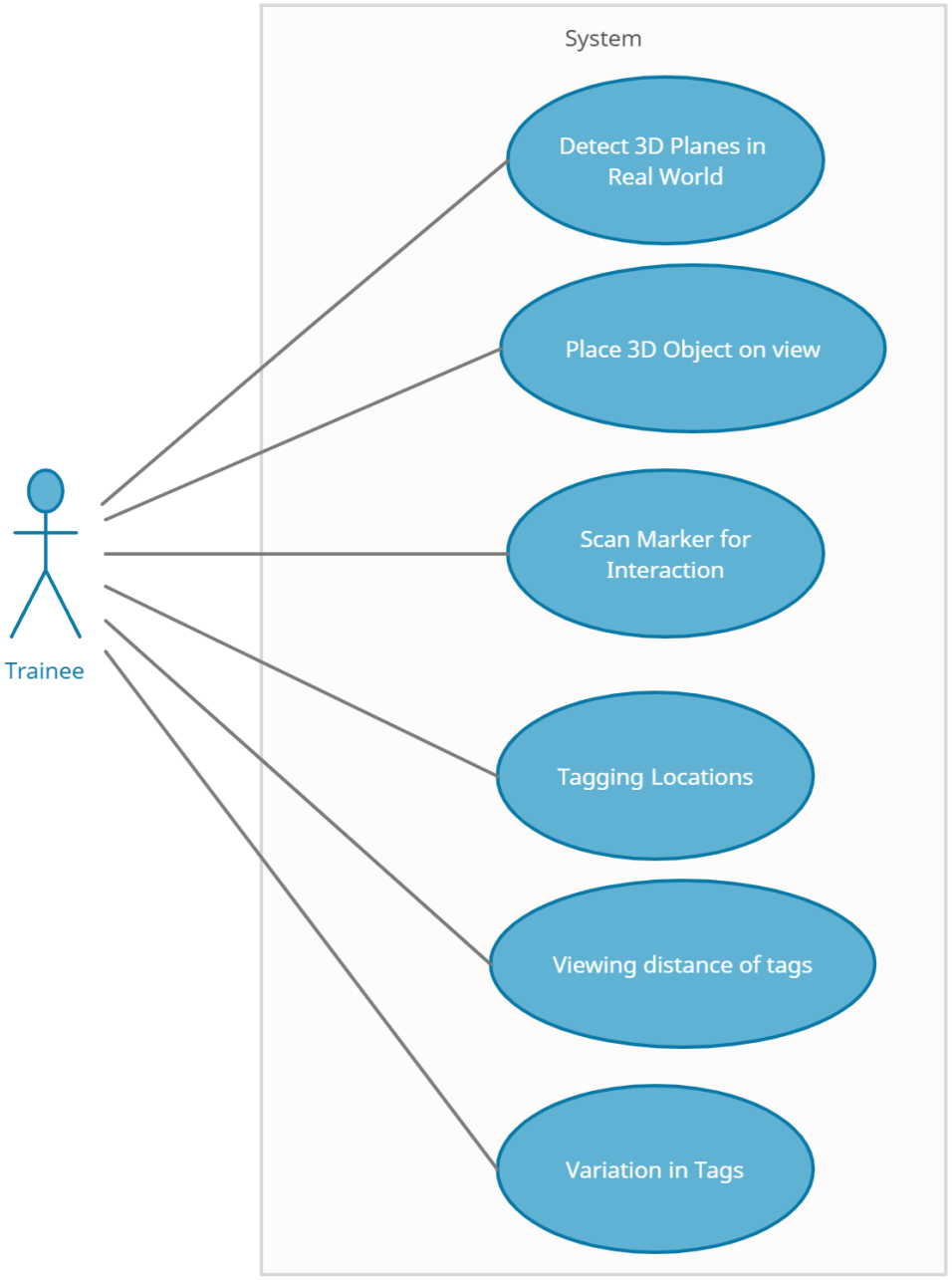
Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera, or mobile turns off.

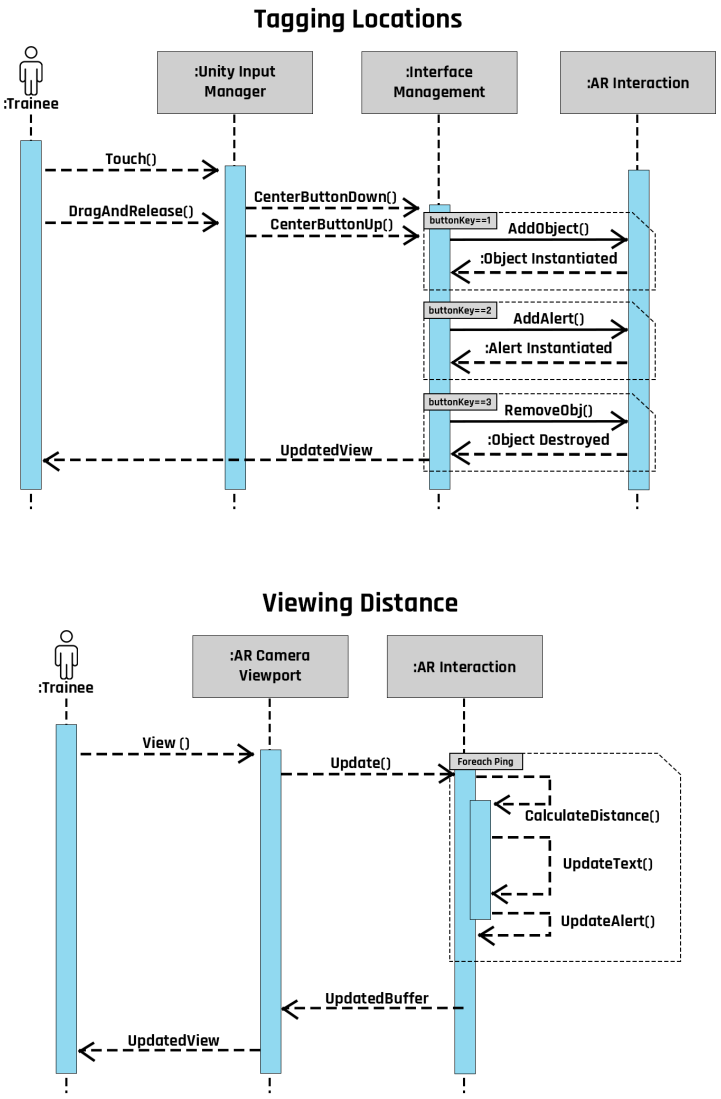
Extended Use Case:

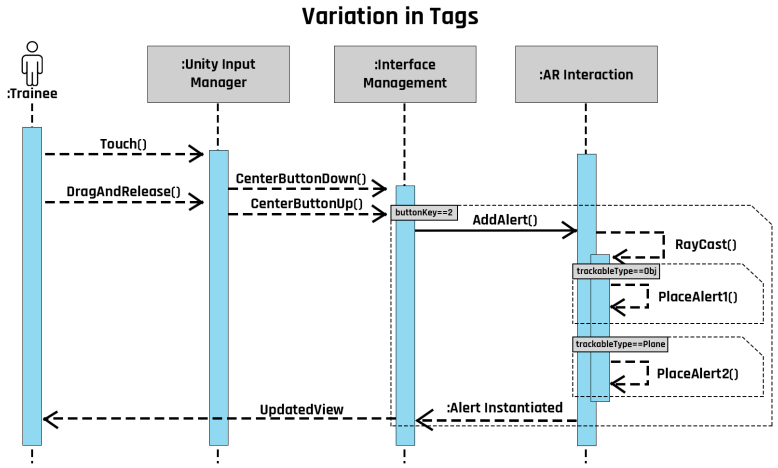
Actor	System
	1. Cast a ray to check the type of tag.
	2. Place the tag depending on which type of trackable is hit by the ray.
	3. Displays tag on camera.

4.4 Use Case Diagram

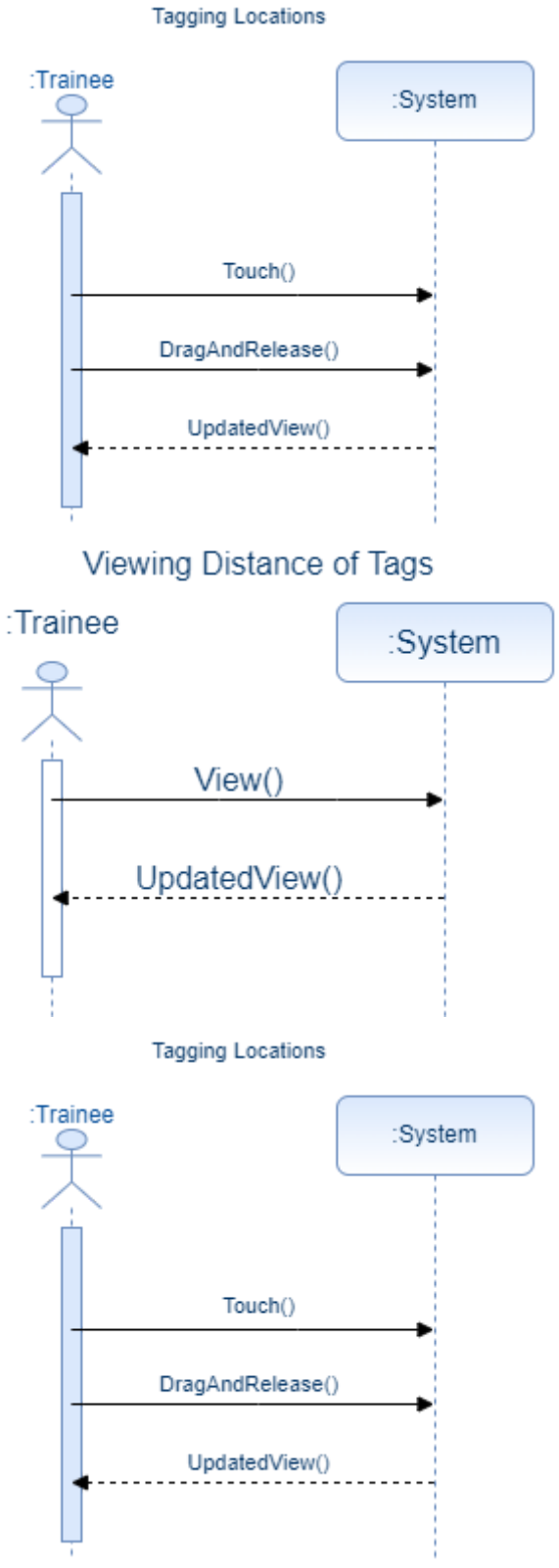


4.5 Sequence Diagram

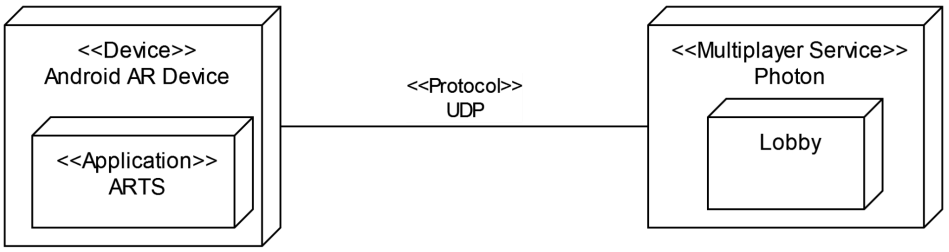



Variation in Tags
:Trainee
:Unity Input Manager
:Interface Management
:AR Interaction
Touch()
DragAndRelease()
CenterButtonDown()
CenterButtonUp()
buttonKey==2
AddAlert()
RayCast()
trackableType==Obj
PlaceAlert1()
trackableType==Plane
PlaceAlert2()
UpdatedView
:Alert Instantiated

4.6 System Sequence Diagram



4.7 Deployment Diagram



5. Iteration 3

5.1: User Stories

5.1.1 User Story 1

Name: Scenario Synchronization

Story: As a host, I want to synchronize the scenario so that trainee's can collaborate with each other in the world.

5.1.2 User Story 2

Name: Player Synchronization

Story: As a user, I want to synchronize my position so that I can have the same origin as other players and view their actual position in the real world.

5.1.3 User Story 3

Name: Object Synchronization

Story: As a user, I want to view the object placed by other users so that I can collaborate with them accordingly.

5.2 Use cases

5.2.1 Use case 1

Use Case:	Scenario Synchronization
Actors:	Host, Trainee
Type:	Primary
Description:	The host decides the synchronization point. The host and trainee's scan the synchronization point to synchronize their origins. They can re-scan the points to reduce the AR redrifting.

5.2.2 Use case 2

Use Case:	Player Synchronization
Actors:	Host, Trainee
Type:	Primary
Description:	The user joins the room, and scans the point decided by the host. Then, his position is sent to other users and their position is also received and displayed to the user.

5.2.3 Use case 3

Use Case:	Object Synchronization
Actors:	Host, Trainee
Type:	Primary
Description:	The host places the objects in the real world. The position of the objects is synchronized with other users as soon as they scan the point. The objects are visible to other users in their own view at approximately same position

5.3 Extended Use Cases

5.3.1 Extended Use Case 1

Use Case Name: Scenario Synchronization

Actor: Host, Trainee

Level: Host-Level Goal

User Story: As a host, I want to synchronize the scenario so that trainee's can collaborate with each other in the world.

Preconditions:

1.) Plane is detected
2.) The host and trainee's are connected to the server

Postconditions: Synchronized view of the world

Special Requirement: The mobile device is AR-enabled.

Frequency of Occurrence: Often

Miscellaneous: What if someone covers the camera or mobile turns off.

Extended Use Case:

Actor	System
1. Host decides a point for synchronization in the real world	
2. Host and other trainee's points the camera on the point and presses the scan button	
	3. System saves the point for each user on his/her device.

5.3.2 Extended Use Case 2

Use Case Name: Player Synchronization

Actor: Host, Trainee

Level: User-Level Goal

User Story: As a user, I want to synchronize my position so that I can have the same origin as other players and view their actual position in the real world.

Precondition:

1.) Plane is detected.
2.) The host and trainee's are connected to the server.
3.) The scenario is synchronized

Postcondition: Displays a 2D sprite on each user's mobile camera in the scenario.

Special Requirement: The mobile device is AR-enabled.

Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera or mobile turns off.

Extended Use Case:

Actor	System
	1. Receives the position of other users in the scenario.
	2. Calculates the distance from other user to current user
	3. Display 2D sprite on the position of other users.
4. Trainee can see the position and distance of other user	

5.3.3 Extended Use Case 3

Use Case Name: Object Synchronization

Actor: Host, Trainee

User Story: As a host, I want to synchronize the scenario so that trainee's can collaborate with each other in the world.

Precondition:

1.) Plane is detected.
2.) The host and trainee's are connected to the server.
3.) The scenario is synchronized

Postcondition: Objects are synchronized for all the users.

Special Requirement: The mobile device is AR-enabled.

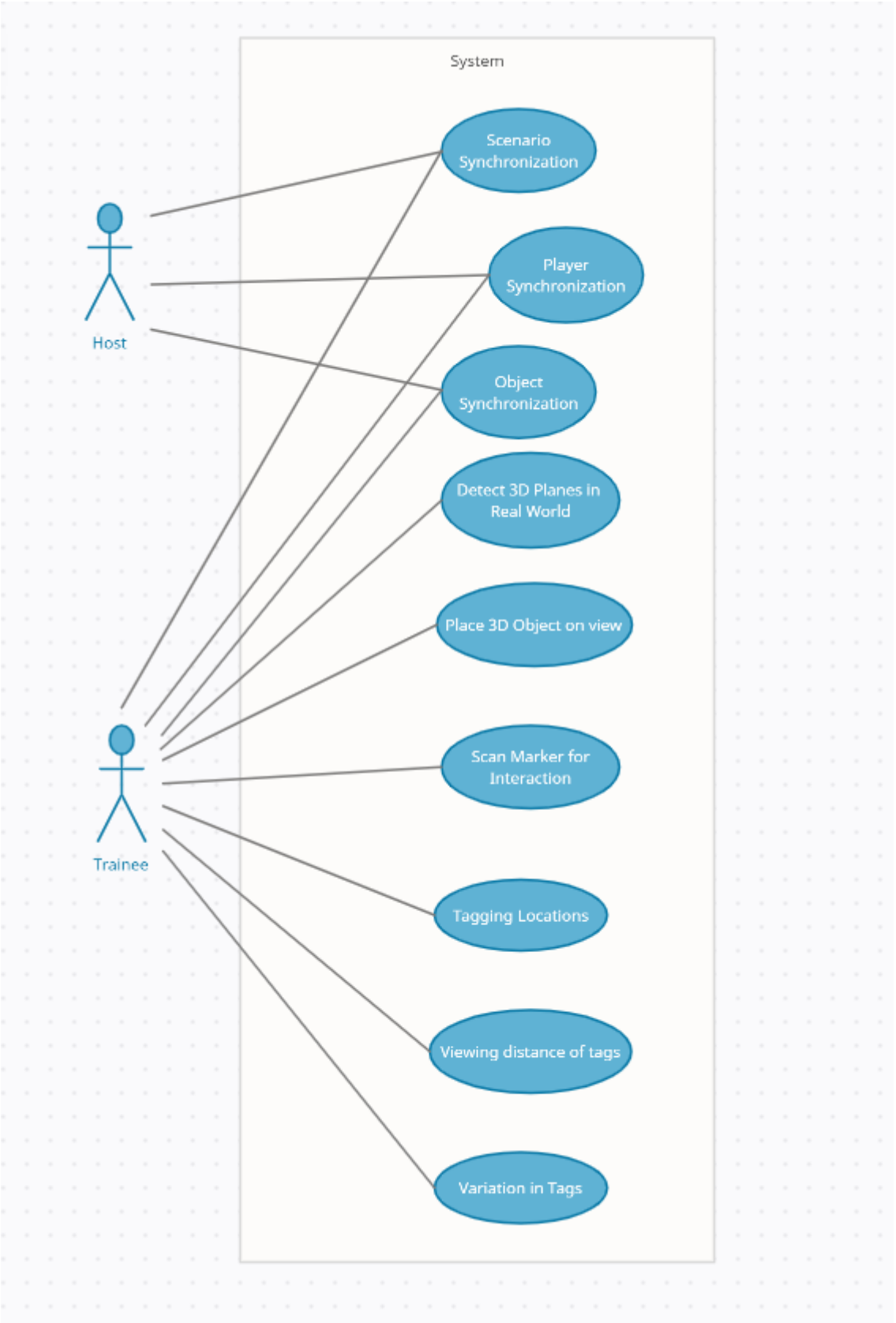
Frequency of Occurrence: Frequent

Miscellaneous: What if someone covers the camera, or mobile turns off.

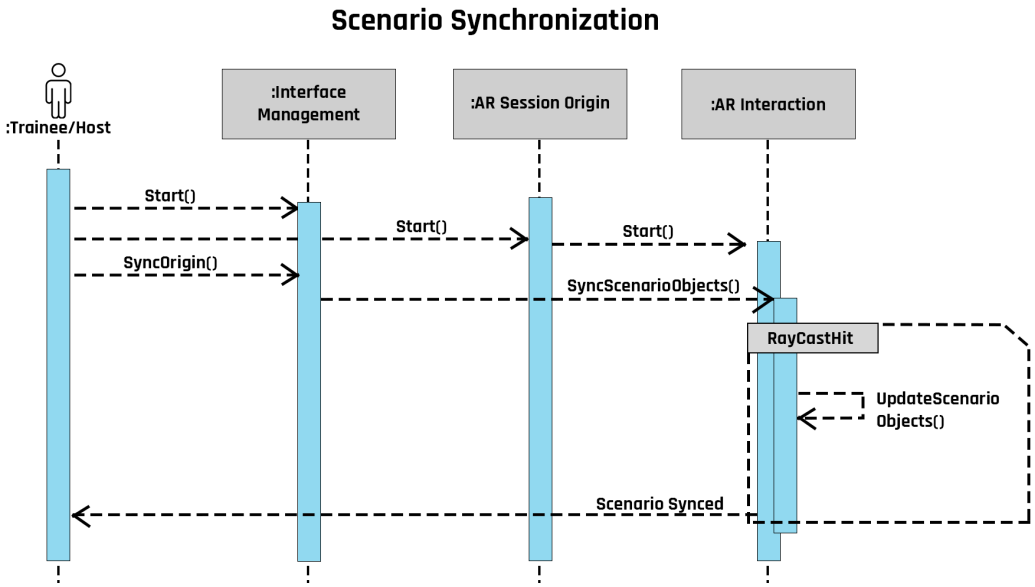
Extended Use Case:

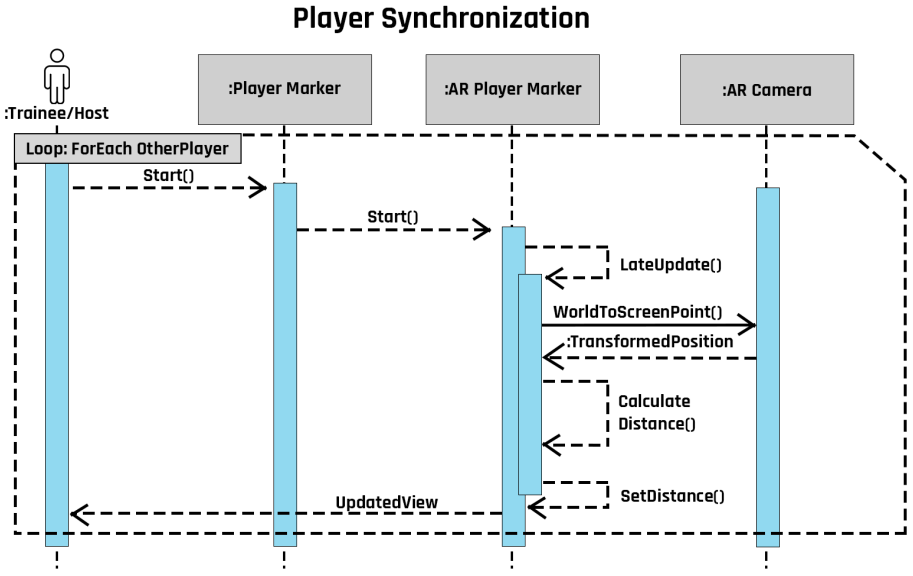
Actor	System
	1. Receives the position of the object placed in the scenario.
	2. Places the object in the world of the user
3. The trainee can view the object placed in the real world	

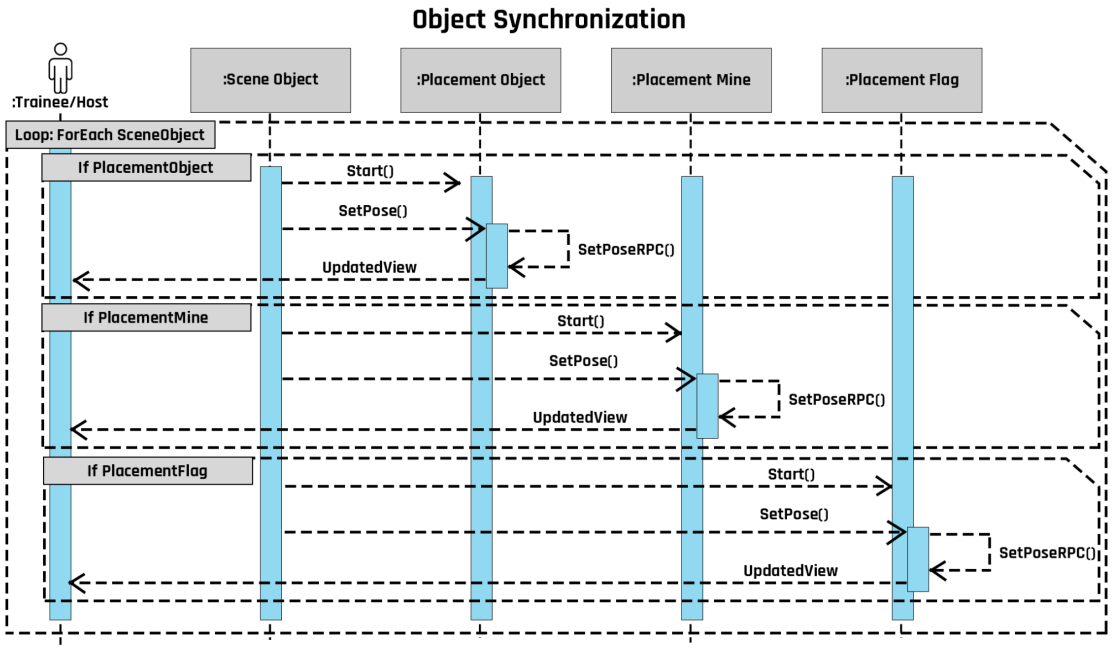
5.4 Use Case Diagram



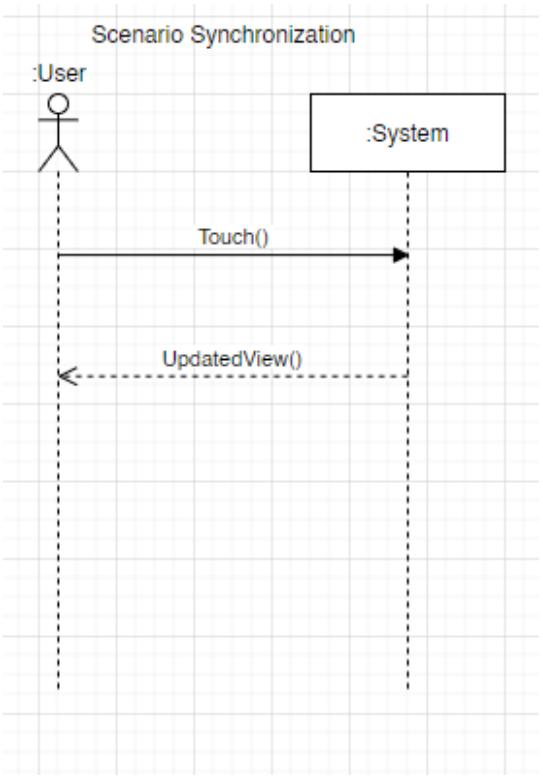
5.5 Sequence Diagram

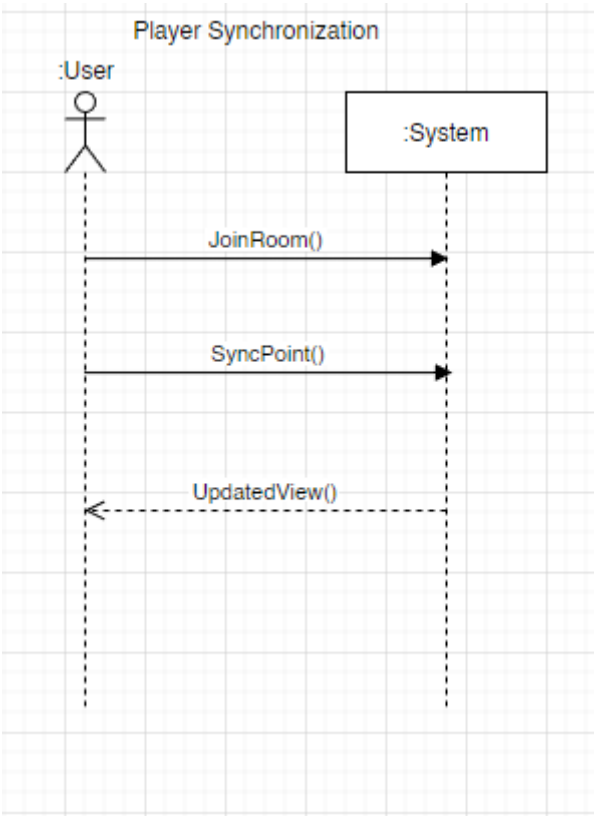


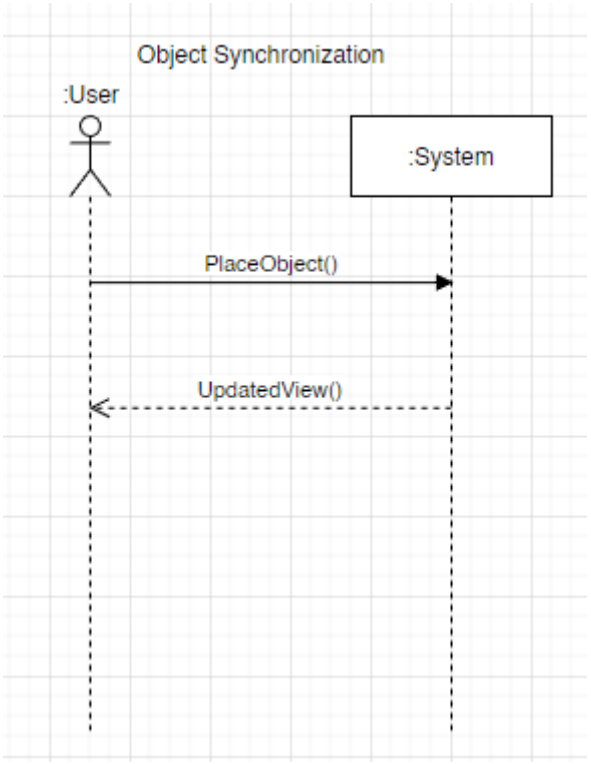




5.6 System Sequence Diagram




Player Synchronization
:User
:System
JoinRoom()
SyncPoint()
UpdatedView()


Object Synchronization
:User
:System
PlaceObject()
UpdatedView()

6. Iteration 4

6.1: User Stories

6.1.1 User Story 1

Name: Saving Scenario

Story: As a host, I want to save a created scenario so that i can load the scenario dynamically at a later time

6.1.2 User Story 2

Name: Loading Scenario

Story: As a host, I want to load a scenario saved earlier so that I dont have to recreate a scenario each time

6.2 Use cases

6.2.1 Use case 1

Use Case:	Saving Scenario
Actors:	Host
Type:	Primary
Description:	The host creates a scenario and saves it for loading it later dynamically and multiple times

6.2.2 Use case 2

Use Case:	Loading Scenario
Actors:	Host
Type:	Primary
Description:	The host loads a scenario saved previously so as to not create the scenario again.

6.3 Extended Use Cases

6.3.1 Extended Use Case 1

Use Case Name: Saving Scenario

Actor: Host

Level: Host-Level Goal

User Story: As a host, I want to save a created scenario so that i can load the scenario dynamically at a later time

Preconditions:

3.) Scenario objects are placed
4.) The host is to the server

Postconditions: Scenario saved in persistent storage

Special Requirement: The mobile device is AR-enabled.

Frequency of Occurrence: Uncommon

Extended Use Case:

Actor	System
1. Connect to server	
2. Create Scenario	
3. Save the scenario	
	4. Serialize scene objects
	5. Store serialized data to persistent storage

6.3.2 Extended Use Case 2

Use Case Name: Loading Scenario

Actor: Host

Level: Host-Level Goal

User Story: As a host, I want to load a scenario saved earlier so that I dont have to recreate a scenario each time

Preconditions:

5.) Scenario was saved earlier
6.) The host is connected to the server

Postconditions: Scenario loaded for each player

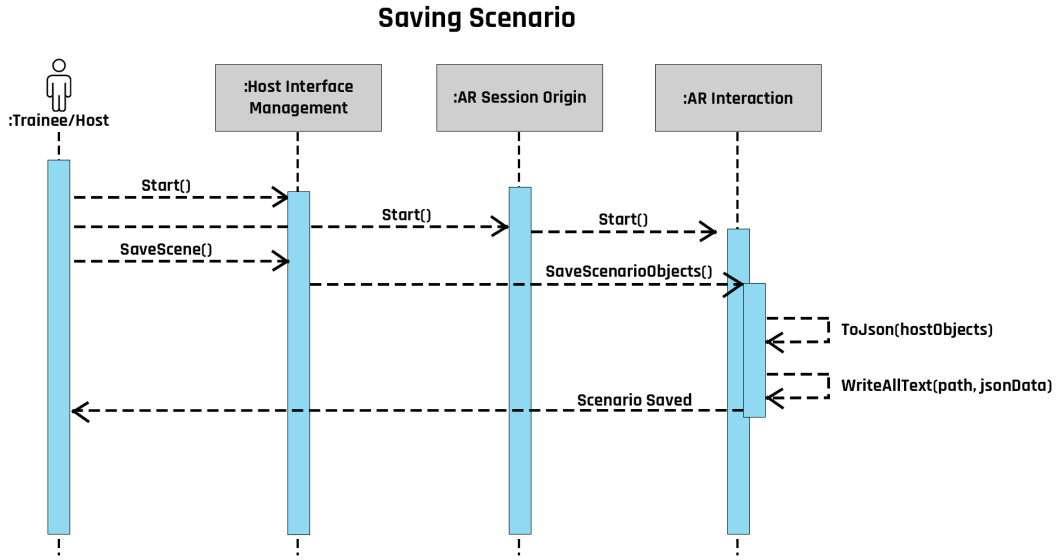
Special Requirement: The mobile device is AR-enabled.

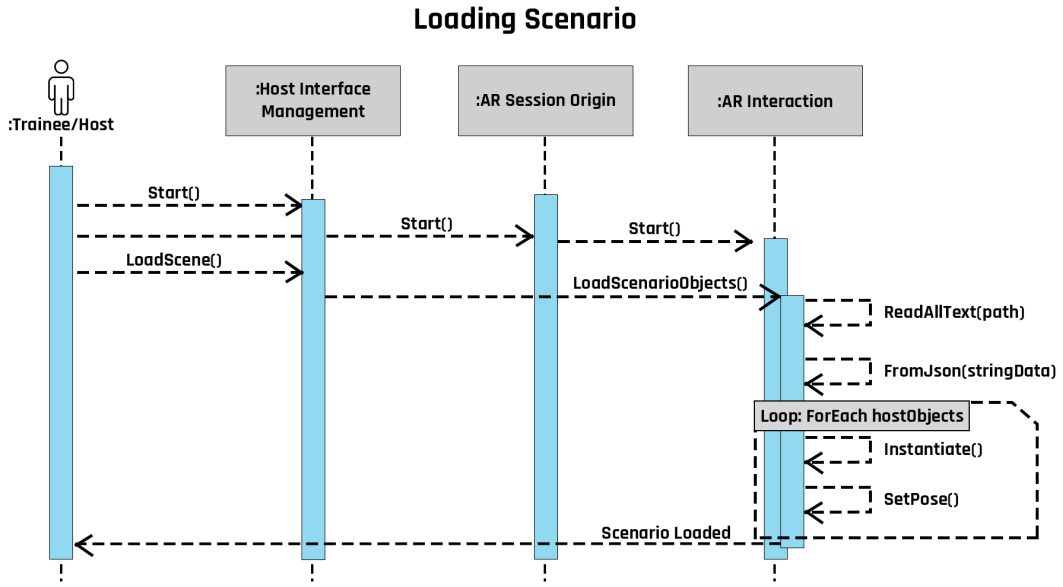
Frequency of Occurrence: Often

Extended Use Case:

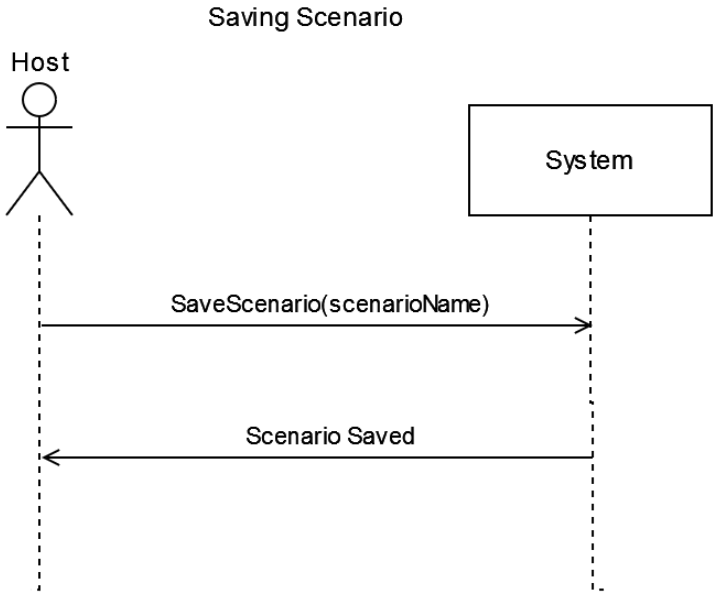
Actor	System
1. Connect to server	
2. Load a scenario	
	3. Deserialize Scene Objects
	4. Spawn Objects on the network

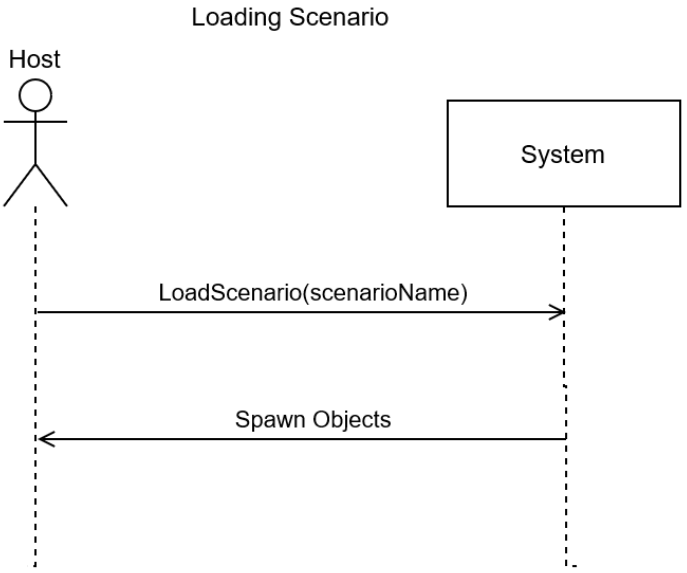
6.4 Sequence Diagram





6.5 System Sequence Diagram





7. User Manual

7.1 Main Menu:

7.1.1 Connecting to server:

The user can connect to sever by pressing START Button on main menu. This will make all the users currently in lobby to join the same room and start the Simulation. The count after the start button indicates the current number of users in lobby.

7.2 Simulation:

7.2.1 Synchronizing Start Point:

The users can synchronize their starting point by selecting center bottom button on the screen.

7.2.2 Adding Host Objects:

The host can add scenario objects by swiping the left button and selecting the respective object i.e flag icon to place a 3D Flag and bomb icon to place a 3D Mine.

7.2.3 Deleting Host Objects:

The host can delete a scenario object by selecting the bin icon from host controls (left button)

7.2.4 Pinging Scenario Objects:

A User can ping a scenario object by swiping the right button and selecting the ping button. The ping will adapt to the object pinged

7.2.5 Multiplayer Positions:

The users can see other players on their screen marked with a UI element

8. References

[1] *Apex Officer - Virtual Reality Police Training Simulator*. (n.d.). Retrieved September 7, 2021, from https://www.apexofficer.com/

[2] *Marker-based vs markerless augmented reality: pros & cons | Overly app*. (n.d.). Retrieved September 7, 2021, from https://overlyapp.com/blog/marker-based-vs-markerless-augmented-reality-pros-cons-examples/