**AR VISUALIZATION IN MEDICAL SCIENCE**

**Minor Project II**

Submitted by:

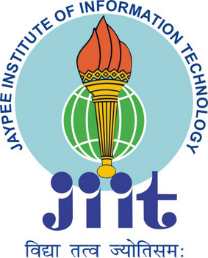
**Shubham Garg (9919103057)**

**Pranjal Tiwari (9919103010)**

**Rahul Sharma (9919103173)**

Under the supervision of:

**Dr. Shilpa Budhkar**

****

**Department of CSE/IT**

**Jaypee Institute of Information Technology University, Noida**

**May 2022**

**ACKNOWLEDGEMENT**

We would like to place on record my deep sense of gratitude to Dr. Shilpa Budhkar, Assistant Professor of the Department of Computer Science & Engineering and Information Technology, Jaypee Institute of Information Technology, India for her generous guidance, help and useful suggestions.

We also express our thanks Dr. Shikha Mehta, Dr. Himanshu Mittal and other classmates for their insightful comments and constructive suggestions to improve the quality of this project work.



### Signatures of Students:

Shubham Garg (9919103057)

Pranjal Tiwari (9919103010)

Rahul Sharma (9919103173)

**DECLARATION**

We hereby declare that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment has been made in the text.

### Place: Jaypee Institute of Information Technology, Noida, India

Date: **10th May 2022**

Name: **Shubham Garg** Enrolment No.: **9919103057** Name: **Pranjal Tiwari** Enrolment No.: **9919103010** Name: **Rahul Sharma** Enrolment No.: **9919103173**

**CERTIFICATE**

This is to certify that the work titled **“AR VISUALIZATION IN MEDICAL SCIENCE”** submitted by **Shubham Garg, Pranjal Tiwari** and **Rahul Sharma** of B.Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of any other degree or diploma.

Digital Signature of Supervisor:

### Dr. Shilpa Budhkar

**Assistant Professor (Grade-II)**

**CSE Department,**

**JIIT 128,**

**Noida**

Date: **10th May 2022**

**ABSTRACT**

The purpose of this project is to upgrade the existing theoretical study methods to make it more interactive using Augmented Reality. Junior high school students’ imaginative abilities are not yet mature. As a result, they are not able to visualize microstructures and concepts correctly during the beginning stage of learning Science. Using Augmented Reality in the classroom can turn an ordinary class into an engaging experience. AR technology provides virtual examples and adds gaming elements to support textbook materials. As a result, classes become more interactive. AR helps students better remember the information they've just learned. This project can be extremely helpful to learn virtually from home during harsh times like COVID - 19 and also to overcome financial barriers. These reasons make students to lose interest in science. With the help of this project, they can understand concepts in greater depth by visualizing experiments using virtual apparatus. This is a safer method as the student doesn’t handle dangerous apparatus on his own. This is also a cheaper method to grasp a concept using virtual experiments.

# **TABLE OF CONTENTS**

Abstract i

[List of Tables iv](#_TOC_250028)

[List of Figures iv](#_TOC_250027)

List of Abbreviations v

List of Nomenclature v

[Chapter 1: INTRODUCTION 1](#_TOC_250026)

* 1. [AUGMENTED REALITY 1](#_TOC_250025)
  2. [UNITY ENGINE 2](#_TOC_250024)
  3. [VUFORIA 2](#_TOC_250024)

[Chapter 2: BACKGROUND STUDY 3](#_TOC_250021)

[Chapter 3: REQUIREMENT ANALYSIS 6](#_TOC_250020)

[Chapter 4: DETAILED DESIGN 7](#_TOC_250018)

[Chapter 5: IMPLEMENTATION 9](#_TOC_250014)

* 1. [STREPS INVOLVED 9](#_TOC_250013)

Chapter 6: RESULTS AND ANALYSIS 12

Chapter 7: CONCLUSION AND FUTERE SCOPE 18

### 

### List of Figures

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| 1 | CanvasRefrenceResollution Script | 9 |
| 2 | DeactivateUI Script | 9 |
| 3 | MainUI Script | 9 |
| 4 | NamingButtonClick Script | 12 |
| 5 | Inspector Window | 12 |
| 6 | PinchZoom Script | 13 |
| 7 | SceneManagerUI Script | 13 |
| 8 | WireResistance Script | 14 |
| 9 | Titration Script | 16 |
| 10 | Chromatography Script | 17 |
| 11 | Image Target for Animal cell | 18 |
| 12 | Output Screenshot of Animal Cell | 19 |
| 13 | Image Target for Plant cell | 19 |
| 14 | Output Screenshot of Plant Cell | 20 |
| 15 | Image Target for Heart | 20 |
| 16 | Output Screenshot of Heart | 21 |
| 17 | Image Target for Titration | 22 |
| 18 | Output Screenshot of Titration | 22 |
| 19 | Image Target for Chromatography | 22 |
| 20 | Output Screenshot of Chromatography | 23 |
| 21 | Image Target for Meter Bridge | 23 |
| 22 | Output Screenshot of Meter Bridge | 23 |

**ABBREVIATIONS**

AR Augmented Reality

SDK Software Development Kit

UI User Interface

### INTRODUCTION

#### **Augmented Reality**

Augmented reality (AR) is an enhanced version of the real physical world that is achieved through the use of digital visual elements, sound, or other sensory stimuli delivered via technology. Unlike virtual reality, which creates its own cyber environment, augmented reality adds to the existing world as it is. It is a growing trend among companies involved in mobile computing and business applications in particular.

#### **Characteristics**

**Wearables:** If a user is viewing the “world” of an augmented reality game via their phone, it’s hard not to notice the surrounding, “real” world beyond the screen. This is distracting, and robs the product of its realism.Headsets will remove such limitations. Thus, developers for augmented reality are going to focus more of their attention on creating experiences with wearable technology in mind.

**Improved Tracking:** software engineers have been working to create and release programs which improve on the tracking capabilities of current AR software development kits. Primary goals include extracting more detail from the real world for tracking, and boosting a software’s ability to easily track fast movements.

**Mobile Products:** Although augmented reality developers will likely pay more attention to wearables throughout the remainder of 2017 and beyond, that doesn’t mean they’ll neglect mobile devices. On the contrary, the success of mobile AR games like Pokemon Go, and implementing augmented reality into mobile apps such as Snapchat make it reasonable to assume that developers will strive for that level of success.

#### **Types Of AR**

The best part of Augmented Reality is that it is accessible for the ordinary user. Augmented reality companies are probably helping users to experience augmented reality with the help of smartphones. Let's look at 4 main types of AR-

##### **Marker-Based AR**

The other name for Marker-Based AR is also called Image Recognition or Recognition based AR. This type of AR provides us more information about the object after it focuses on the recognition of objects. Marker-based AR technology has diverse uses according to market purposes. It detects the object in front of the camera and provides information about the object on the screen. The recognition of the object is based on the marker where it replaces the marker on the screen with a 3D version of the corresponding object. Therefore, the user can view the object in more detail and from various angles. Apart from that while rotating the marker, the user can also rotate the 3D imagery as well. This acts as a reference for the AR app running on the system.

##### **Markerless Augmented Reality**

Markerless augmented reality is one of the most widely implemented applications in the industry. It is also known as Location-based AR for the reason for the easy availability of the features in the smartphones that provide location detection. This type of app is mostly used to help travelers. Apart from that, it helps users to discover interesting places within their current location. This method works by reading data from the mobile’s GPS, digital compass and accelerometer while predicting where the user is focusing. This AR is all about adding location information on screen about the objects that can be seen from the user’s camera.

##### **Projection Augmented Reality**

This is one of the simplest types of AR which is the projection of light on a surface. Projection-based AR is appealing and interactive where light is blown onto a surface and the interaction is done by touching the projected surface with your hand. The widespread uses of projection-based AR techniques can be used to create deception about the position, orientation, and depth of an object. In such a case this allows the user to take different objects into consideration and its structure in order to study in-depth. This technology offers a whole lot more in every sense. This piece of tech is used for creating a virtual object for much larger deployments for experiencing Augmented Reality.

##### **Superimposition Based Augmented Reality**

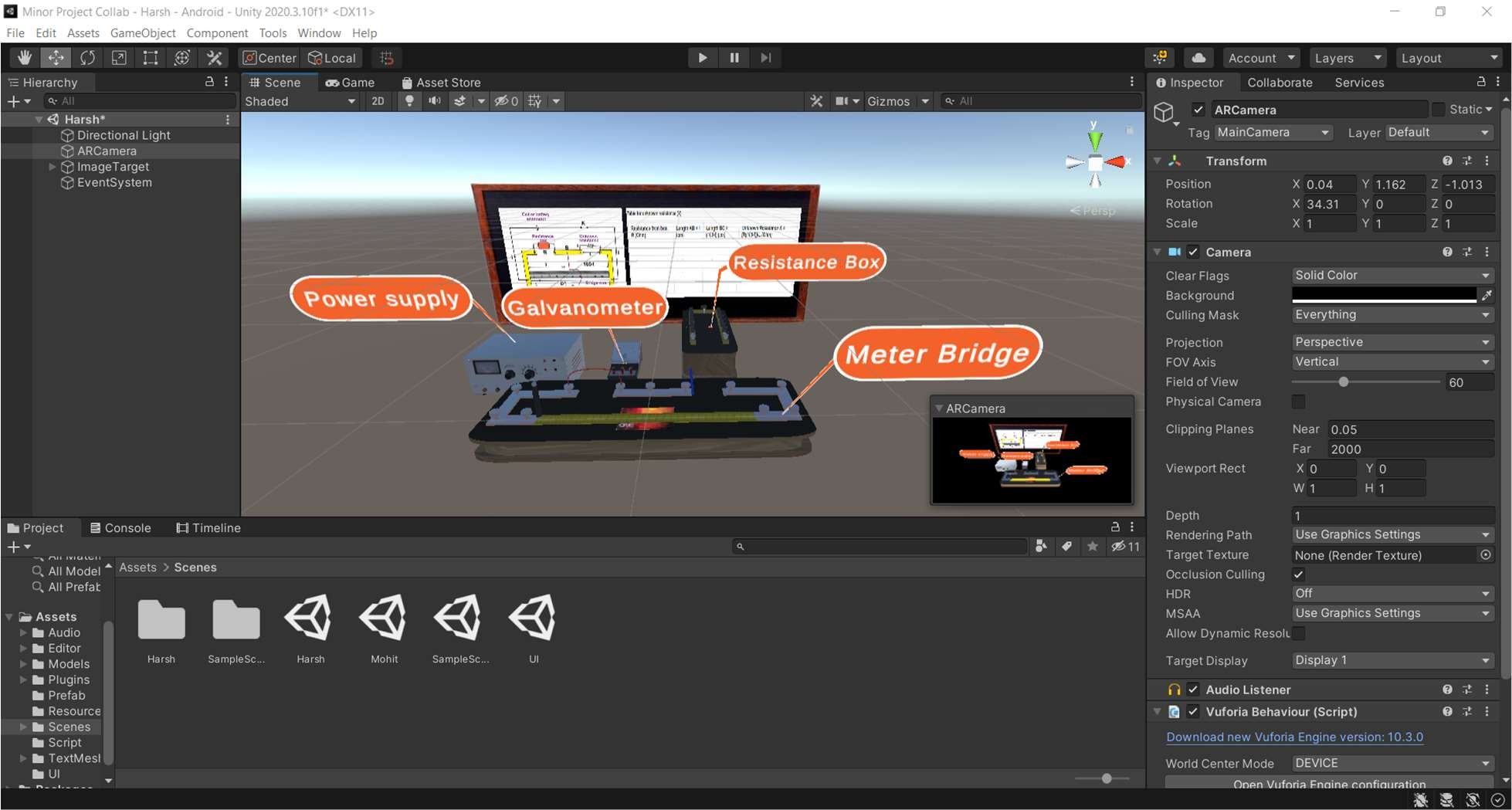
As the word itself explains the superimposition of the objects. This AR provides a replacement view of the object in focus. This is done by replacing the entire or partial view with an augmented view of the object. Here object recognition plays a vital role where replacing a view of an object with an augmented view is done.

#### **Uses**

* + - 1. AR-powered Art Exhibits
      2. AR in Automotive & Manufacturing
      3. Use of AR in Social Media
      4. AR Indoor Navigation
      5. AR Avatars
      6. AR-driven healthcare apps
      7. AR-driven wearable app
      8. AR-driven travel app
      9. AR-driven interior designing app
      10. AR-driven face tuning app

#### **Unity Engine**

Unity is a **cross-platform** game engine developed by **Unity Technologies**, first announced and released in June 2005 at Apple Inc.'s Worldwide Developers Conference as a Mac OS X- exclusive game engine. As of 2019, the engine had been extended to support more than 25 platforms. The engine can be used to create three-dimensional, two-dimensional, virtual reality, and augmented reality games, as well as simulations and other experiences. The engine has been adopted by industries outside video gaming, such as film, automotive, architecture, engineering and construction.



#### **1.2 Overview**

Unity gives users the ability to create games and experiences in both 2D and 3D, and the engine offers a primary scripting API in C#, for both the Unity editor in the form of plugins, and games themselves, as well as drag and drop functionality. Prior to C# being the primary programming language used for the engine, it previously supported Boo, which was removed with the release of Unity 5, and a version of JavaScript called UnityScript, which was deprecated in August 2017, after the release of Unity 2017.1, in favor of C#.

Within 2D games, Unity allows importation of sprites and an advanced 2D world renderer. For 3D games, Unity allows specification of texture compression, mipmaps, and resolution settings for each platform that the game engine supports, and provides support for bump mapping, reflection mapping, parallax mapping, screen space ambient occlusion (SSAO), dynamic shadows using shadow maps, render-to-texture and full-screen post-processing effects.

#### **Supported Platforms**

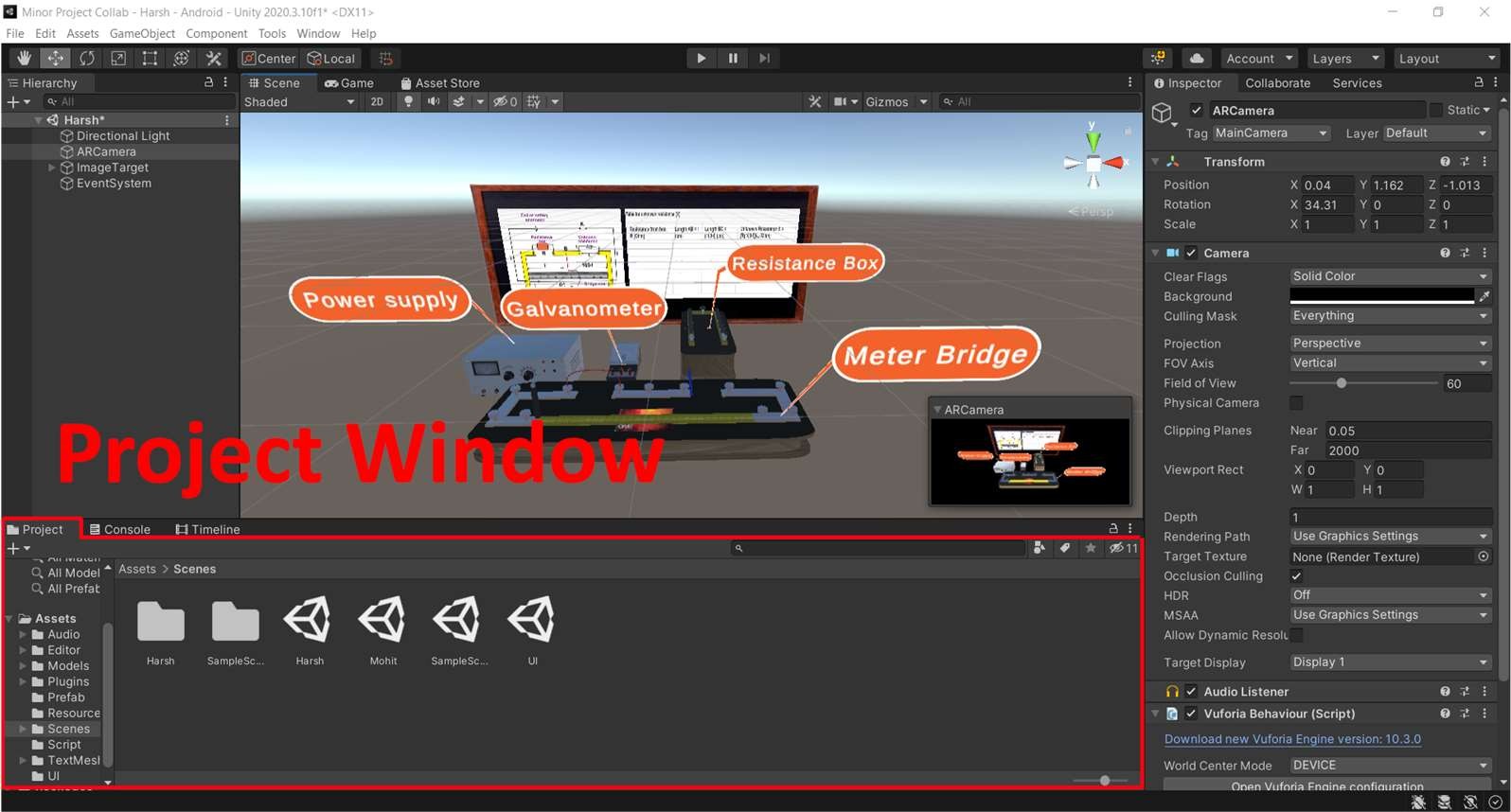
Unity is a cross-platform engine. The Unity editor is supported on Windows, macOS, and

the Linux platform, while the engine itself currently supports building games for more than 25 different platforms, including mobile, desktop, consoles, and virtual reality devices.

#### **Unity Engine’s Interface**

##### **The Project Window**

The Project window displays all of the files related to your Project and is the main way you can navigate

and find Assets and other Project files in your application.

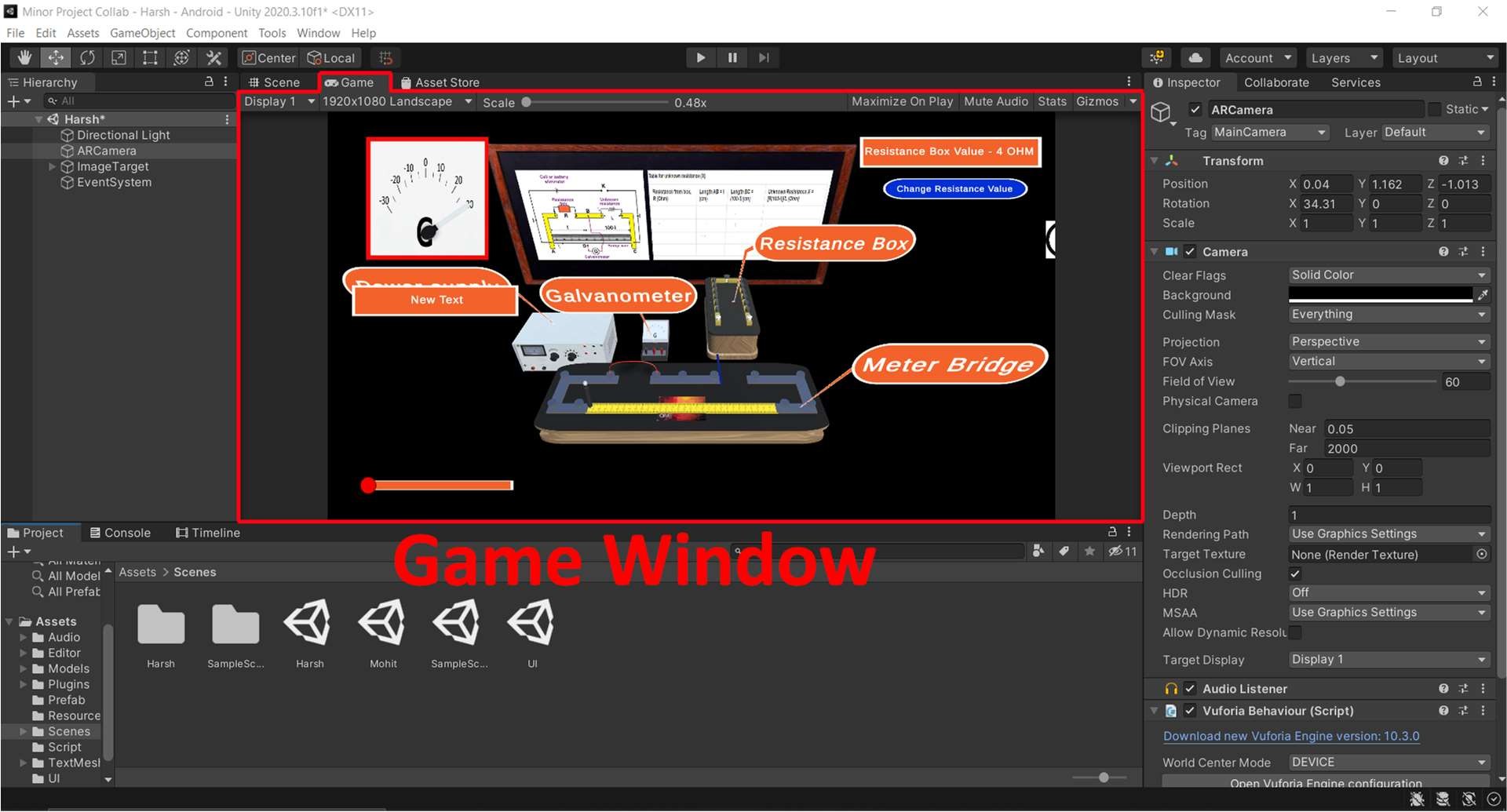
##### **The Scene View**

The Scene view is your interactive view into the world you are creating. You can use the Scene view to select and position scenery, characters, Cameras, lights, and all other types of Game Objects. Selecting, manipulating, and modifying Game Objects in the Scene view are some of the first skills you must learn to begin working in Unity.



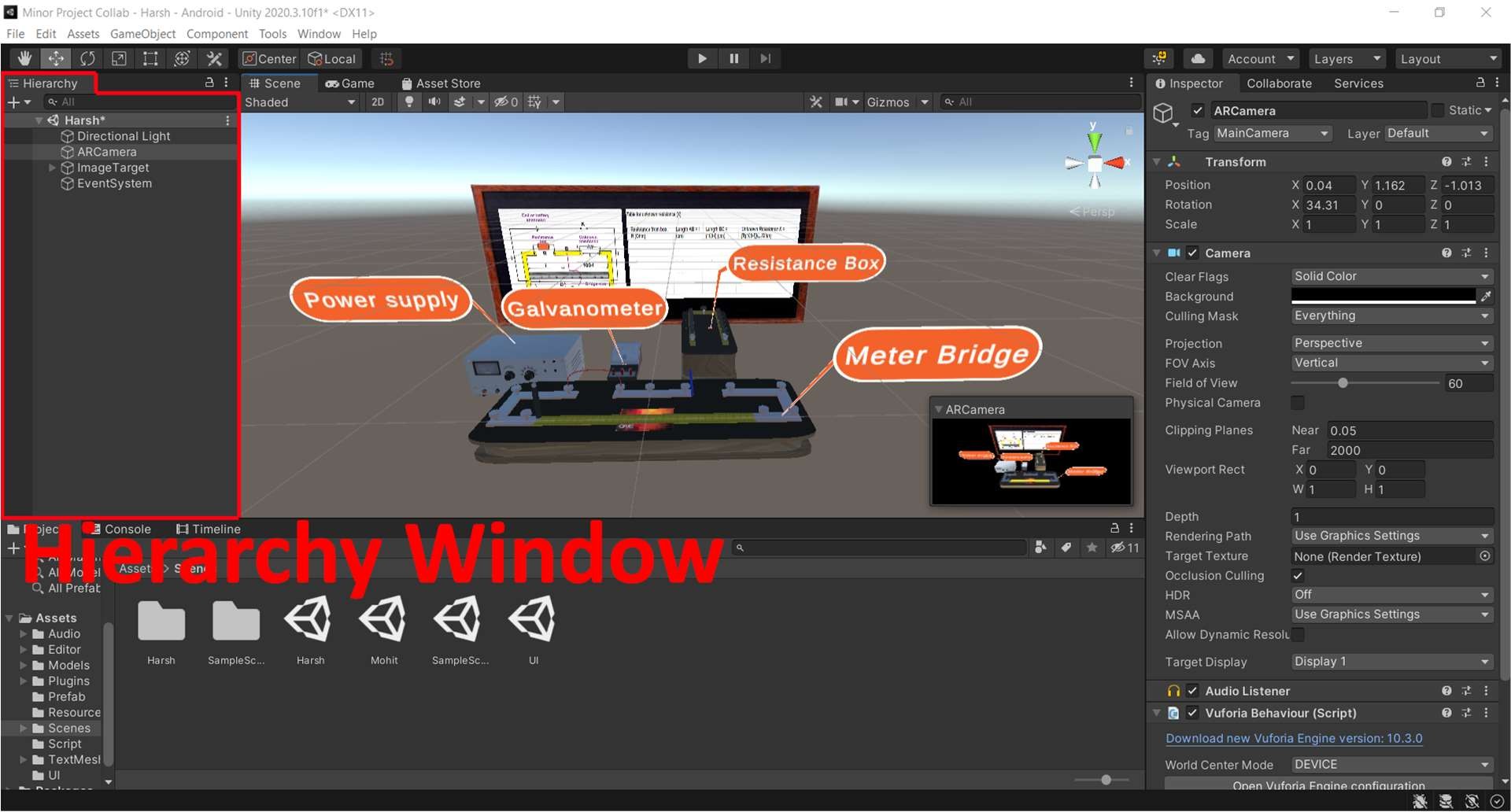
##### **The Game View**

The Game view is rendered from the Camera(s) in your application. It represents your final, published application. You need to use one or more Cameras to control what the player sees when they are using your application.



##### **The Hierarchy Window**

The Hierarchy window contains a list of every Game Object in the current Scene. Some of these are direct instances of Asset files (like 3D models), and others are instances of Prefabs, which are custom Game Objects that make up most of your game.



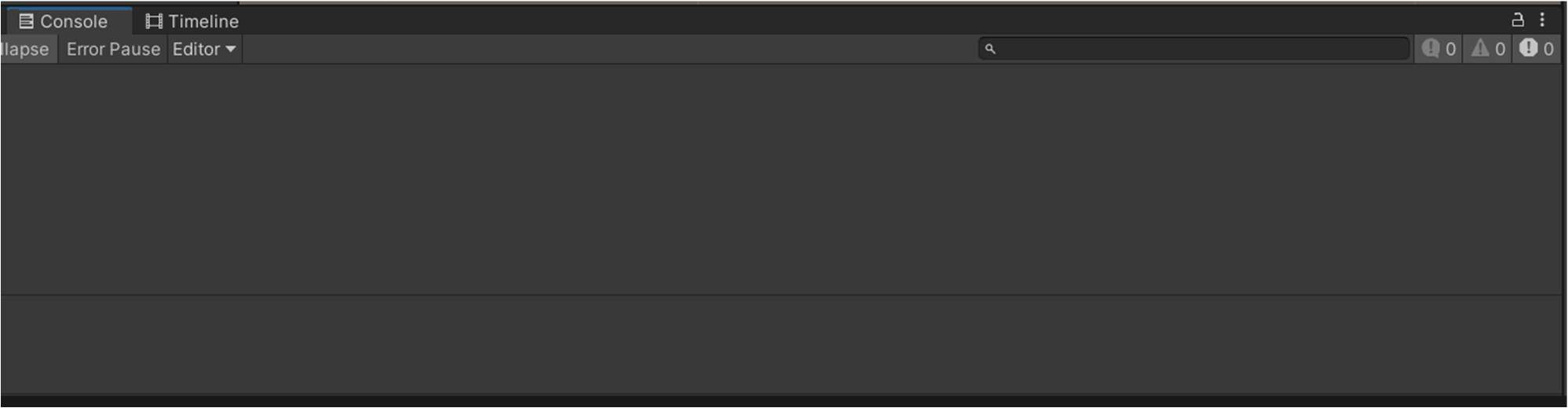
##### **The Inspector Window**

The Inspector window (sometimes referred to as “the Inspector”) displays detailed information about the currently selected Game Object, including all attached components and their properties, and allows you to modify the functionality of Game Objects in your Scene.



##### **Console Window**

The Console Window shows errors, warnings and other messages generated by Unity. You can also show your own messages in the Console using the Debug class.



##### **Meshes, Materials, Shaders and Textures**

Rendering in Unity uses Meshes, Materials, Shaders and Textures. They have a close relationship.

* Meshes are the main graphics primitive of Unity. They define the shape of an object.
* Materials define how a surface should be rendered, by including references to the Textures it uses, tiling information, Color tints and more. The available options for a Material depend on which Shader the Material is using.
* Shaders are small scripts that contain the mathematical calculations and algorithms for calculating the Color of each pixel rendered, based on the lighting input and the Material configuration.
* Textures are bitmap images. A Material can contain references to textures, so that the

Material’s Shader can use the textures while calculating the surface color of

a Game Object. In addition to basic Color (Albedo) of a Game Object’s surface, Textures can represent many other aspects of a Material’s surface such as its reflectivity or roughness.

#### **Vuforia**

Vuforia is an augmented reality software development kit (SDK) for mobile devices that enables the creation of augmented reality applications. It uses computer vision technology to recognize and track planar images and 3D objects in real time. This image registration capability enables developers to position and orient virtual objects, such as 3D models and other media, in relation to real world objects when they are viewed through the camera of a mobile device. The virtual object then tracks the position and orientation of the image in real-time so that the viewer's perspective on the object corresponds with the perspective on the target. It thus appears that the virtual object is a part of the real-world scene.

The Vuforia SDK supports a variety of 2D and 3D target types including ‘marker less’ Image Targets, 3D Model Target, and a form of addressable Fiducial Marker, known as a VuMark. Additional features of the SDK include 6 degrees of freedom device localization in space, localized Occlusion Detection using ‘Virtual Buttons’, runtime image target selection, and the ability to create and reconfigure target sets programmatically at runtime.

Vuforia provides Application Programming Interfaces (API) in C++, Java, Objective-C++, and the .NET languages through an extension to the Unity game engine. In this way, the SDK supports both native development for iOS, Android, and UWP while it also enables the development of AR applications in Unity that are easily portable to both platforms.

### BACKGROUND STUDY

To complete the project “Educational AR application” a detailed study of unity engine and its development and deployment process was done. We Collected all information regarding AR and decided to proceed with markered AR for easy marker access and meaningful projections of models. For AR development we searched for various AR frameworks and found Vuforia easy to implement and widely used and economically cheaper than others. A detailed study on Vuforia was done and implemented with unity engine to develop this project.

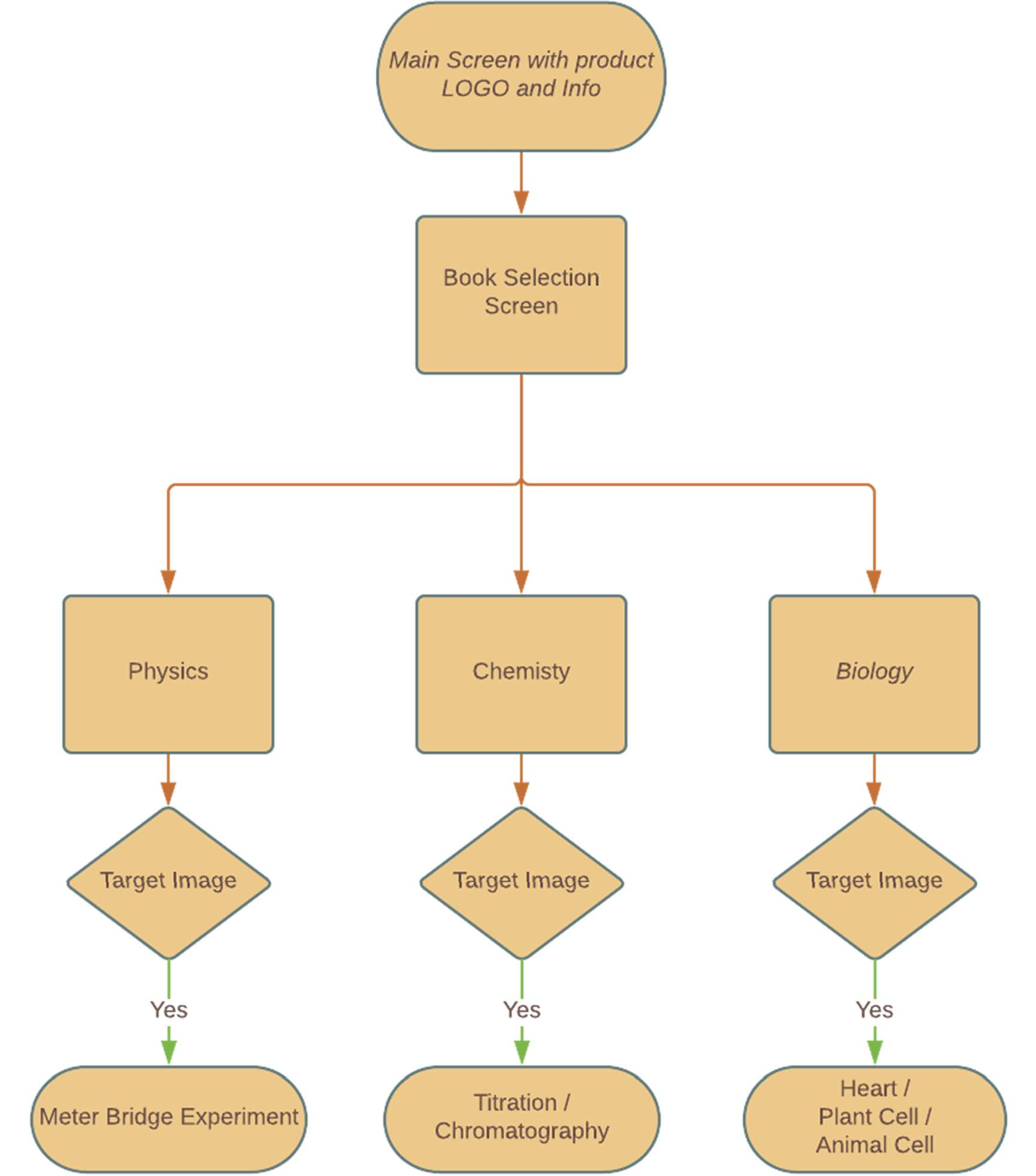
### REQUIREMENT ANALYSIS

Hardship faced by students in understanding and imagining an interesting and eye-catching topic with just theoretical sentences and a dull diagram provided by NCERT books calls for the requirement of such products which helps students in visualising everything with greater ease.

Augmented Reality is fascinating in itself and using it for educational purposes is not a unique idea now, but NCERT books which haven't been updated for a long time can be transformed into something new

with upcoming technologies like AR. None of the educational AR applications targets NCERT books, this is where our product comes in picture.

### DETAILED DESIGN

workflow of application is given in a flowchart below

### IMPLEMENTATION

using UnityEngine; using UnityEngine.UI;

public class CanvasRefrenceResollution : MonoBehaviour

{

public CanvasScaler scaler;

void Start() { scaler.referenceResolution = new

Vector2(Screen.currentResolution.width, Screen.currentResolution.height);

}

private void Update(){ scaler.referenceResolution = new

Vector2(Screen.currentResolution.width, Screen.currentResolution.height);}}

Figure 1. The function of this script is to perfectly fit everything in an application with different

resolutions of different devices.

using System.Collections;

using System.Collections.Generic; using UnityEngine;

public class DeactivateUI : MonoBehaviour

{

[SerializeField] GameObject[] UIComponents; bool Isactive = false;

public void DeactivateUIAll() { Isactive = !Isactive;

foreach(GameObject x in UIComponents)

{x.SetActive(Isactive);}}}

Figure 2. The function of this script is to deactivate or activate all the UI markings in the scene to perform the experiment without any hindrance.

using System.Collections;

using System.Collections.Generic; using UnityEngine;

using DG.Tweening; using UnityEngine.UI;

public class MainUI : MonoBehaviour

{

[SerializeField]GameObject MainPanel, SelectBookPanel; void Start() {

MainPanel.SetActive(true); SelectBookPanel.SetActive(true);

}

public void TapToContinue(){ SelectBookPanel.GetComponent<RectTransform>().DOAnchorPos(new

Vector2(0, 0), 0.5f);}}

Figure 3. The function of this script is to handle the main menu UI, which activates the book selection panel to select different book scenes.

using System.Collections;

using System.Collections.Generic; using UnityEngine;

using DG.Tweening; using UnityEngine.UI;

using UnityEngine.Playables; using TMPro;

public class NamingButtonClick : MonoBehaviour

{

[SerializeField] GameObject BottomText, RaycastBlocker, PlantCell, AnimalCell;

[SerializeField] GameObject[] PlantCellComponent, AnimalCellComponent; [SerializeField] string[] AnimalCellInfo, PlantCellInfo, HeartInfo; public int index, type;//type 0->Animal cell ,1->Heart ,2->Plant cell [SerializeField] AudioSource audiosource;

[SerializeField] AudioClip[] PlantCellAudio, AnimalCellAudio, HeartAudio;

[SerializeField] PlayableDirector PlantCelldirector, AnimalCelldirector; [SerializeField] PlayableAsset[] PlantCellAsset, AnimalCellAsset; [SerializeField] PlayableAsset PlantCellTimeline, AnimalCellTimeline; [SerializeField] TextMeshProUGUI TextBox,Heading;

string[] HeadingtextP = { "Cytoplasm", " Chloroplast" , "Mitochondrion", "Vacuole" , "Cell Wall", "Golgi Apparatus", "Lysosomes", "Endoplasmic Reticulam", "Necleus"};

string[] HeadingtextA = { "Golgi Apparatus", " Centriole", "Lysosomes", "Endoplasmic Reticulam", "Necleus", "Plasma Membrane ", "Cytoplasm", "Mitochondrion" };

string[] HeadingtextH = { "Aorta", "Pulmonary Artery" , "Pulmonary Veins", "Left Atrium", "Left Ventrical", "Right Ventrical", "Right Atrium","Vena Cava"};

void Start(){ BottomText.GetComponent<RectTransform>().DOAnchorPos(new Vector2(0, -

400) ,0.5f);

RaycastBlocker.SetActive(false); PlantCell.SetActive(true); AnimalCell.SetActive(true); PlantCelldirector.playableAsset = PlantCellTimeline;

AnimalCelldirector.playableAsset = AnimalCellTimeline;

}

void SetComponentFalse(bool Bool){ foreach(GameObject var in PlantCellComponent)

{var.SetActive(Bool);}

foreach (GameObject var in AnimalCellComponent)

{var.SetActive(Bool);}

}

public void Findtype(int temp){

if (temp < 10) { index = temp;type = 0;}

else if(temp>9 && temp < 20) { index = temp-10; type = 1; } else { index = temp - 20;type = 2; }

}

public void NamingButtonTap(int ind){ Findtype(ind); RaycastBlocker.SetActive(true); if (type == 0){

audiosource.clip = AnimalCellAudio[index]; AnimalCelldirector.playableAsset = AnimalCellAsset[index]; AnimalCell.SetActive(false);

AnimalCelldirector.Play(); TextBox.text = AnimalCellInfo[index]; Heading.text = HeadingtextA[index];

}

else if (type == 1){

audiosource.clip = HeartAudio[index]; TextBox.text = HeartInfo[index]; Heading.text = HeadingtextH[index];

}

else{

audiosource.clip = PlantCellAudio[index]; PlantCelldirector.playableAsset = PlantCellAsset[index]; PlantCell.SetActive(false);

PlantCelldirector.Play(); Heading.text = HeadingtextP[index]; TextBox.text = PlantCellInfo[index];

}

BottomText.GetComponent<RectTransform>().DOAnchorPos(new Vector2(0, 400),0.5f);

audiosource.Play();

}

public void PlayAgainButton(){ audiosource.Stop(); audiosource.Play();

}

public void CloseButton(){ BottomText.GetComponent<RectTransform>().DOAnchorPos(new Vector2(0,

-400), 0.5f);

RaycastBlocker.SetActive(false); audiosource.Stop();

PlantCelldirector.playableAsset = PlantCellTimeline; AnimalCelldirector.playableAsset = AnimalCellTimeline; PlantCell.SetActive(true);

AnimalCell.SetActive(true); SetComponentFalse(false);

}}

Figure 4. The function of this script is to handle all the data or information about biology that will be

showcased when the particular image target is visible.

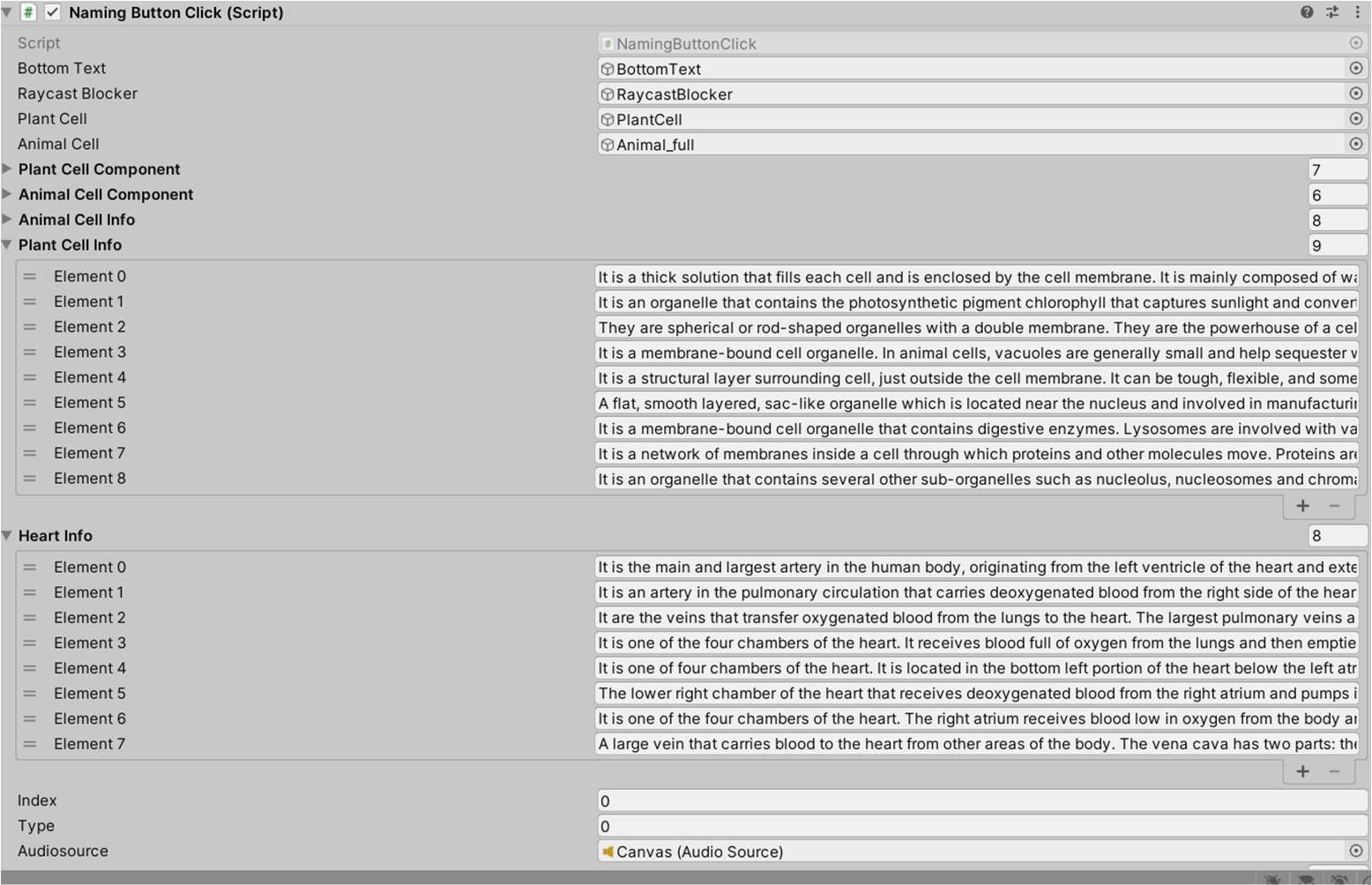


Figure 5. Data is stored in the inspector window

using System.Collections;

using System.Collections.Generic; using UnityEngine;

public class PinchZoom : MonoBehaviour

{

float initialDistance; private Vector3 initialScale;

public GameObject spawnedObject; void Update(){

if (Input.touchCount == 2){

var touchZero = Input.GetTouch(0); var touchOne = Input.GetTouch(1);

if (touchZero.phase == TouchPhase.Ended || touchZero.phase == TouchPhase.Canceled || touchOne.phase == TouchPhase.Ended || touchOne.phase == TouchPhase.Canceled)

{return;}

if (touchZero.phase == TouchPhase.Began || touchOne.phase == TouchPhase.Began){ initialDistance = Vector2.Distance(touchZero.position, touchOne.position); initialScale = spawnedObject.transform.localScale;

}

else {

var currentDistance = Vector2.Distance(touchZero.position, touchOne.position);

if (Mathf.Approximately(initialDistance, 0)) return;

var factor = currentDistance / initialDistance; spawnedObject.transform.localScale = initialScale \* factor; }}}}

Figure 6. The function of this script is to scale the size of the model to desired size if necessary.

using System.Collections;

using System.Collections.Generic; using UnityEngine;

using UnityEngine.SceneManagement;

public class SceneManagerUI : MonoBehaviour

{

public void ChangeScene(int sceneNumber)

{SceneManager.LoadScene(sceneNumber);}

}

Figure 7. The function of this script is to control screen workflow by changing scene when correct button is pressed

using System.Collections;

using System.Collections.Generic; using UnityEngine;

using UnityEngine.UI; using TMPro;

public class Resistance

{

public int ResistanceValue; public int ScaleLenght;

public void TakeInput(int resistance, int lenght){ ResistanceValue = resistance;

ScaleLenght = lenght;

}

public int GetLength()

{return ScaleLenght;}

};

public class WIreResiistance : MonoBehaviour

{

public Slider slider; public GameObject Jockey;

public RectTransform neddle;

public TextMeshProUGUI DisplayLenght;

public TextMeshProUGUI ResistanceBoxValue; public bool R1Check;

public Resistance R1;

public GameObject ResistanceWire; void Update() {

Jockey.transform.localPosition = new Vector3(0, 0.0899f,(-slider.value

+ 0.4861f));

if(Jockey.transform.localPosition.z < -0.4861f) { Jockey.transform.localPosition = new Vector3(0, 0.0899f,(-0.4861f));

}

DisplayLenght.text = "Scale Reading - " + (Mathf.InverseLerp(0,.787f, slider.value) \* 100).ToString("0.00")+ " CM";

if(R1Check){

neddle.rotation = Quaternion.Euler(0, 0,55 - (slider.value \* 100)

);}

else{

neddle.rotation = Quaternion.Euler(0, 0, 55 - (slider.value \*

90));}

}

public void ChangeResistance(){ R1Check = !R1Check; slider.value = 0; if(R1Check){

ResistanceBoxValue.text = "Resistance Box Value - 2 OHM";} else{

ResistanceBoxValue.text = "Resistance Box Value - 4 OHM";

}}}

Figure 8. The function of this script is to properly animate all UI in meter bridge experiments like

changing the rotation of the needle in the galvanometer. And also to provide correct reading in UI when jokey is moved to a different location.

using System.Collections;

using System.Collections.Generic; using UnityEngine;

using UnityEngine.Playables;

public class Titration : MonoBehaviour

{

public PlayableDirector director; public PlayableAsset PutSalt; public PlayableAsset ClickSalt; public PlayableAsset ClickFlask; public PlayableAsset Mixsalt; public PlayableAsset Clickwater; public PlayableAsset Final;

public PlayableAsset FinalHalf; public GameObject SaltContainer; public GameObject Spoon;

public GameObject SaltonPaper; public GameObject FilterPaper; public GameObject WeightMachine; public GameObject Funnel;

public GameObject Flask; public GameObject Water; public GameObject Acid; public GameObject Pipet; void Update(){

if (Input.GetMouseButtonDown(0)){ RaycastHit hit;

Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition); if (Physics.Raycast(ray, out hit, Mathf.Infinity)){

if (hit.collider.name == "Salt"){ director.playableAsset = PutSalt; director.Play();

}

else if (hit.collider.name == "Flask\_04"){ director.playableAsset = Mixsalt;

director.Play();

}

else if (hit.collider.name == "Flask\_03 (1)") { director.playableAsset = Final; director.Play();

}

else if (hit.collider.name == "burettecap") { director.playableAsset = Final; director.Pause();

director.time = 41.15f; director.Resume(); director.Play();

}

}}}

public void DisableSaltContainer() { SaltContainer.SetActive(false); Spoon.SetActive(false); SaltonPaper.SetActive(true); Funnel.SetActive(true); Flask.SetActive(true); director.playableAsset = ClickFlask; director.Play();

}

public void DetachChild() { WeightMachine.SetActive(false); Water.SetActive(true); Acid.SetActive(true); Pipet.SetActive(true); director.playableAsset = Clickwater; director.Play();

}

public void DisableWaterAcidCon(){ Funnel.SetActive(false); Water.SetActive(false); Acid.SetActive(false); Pipet.SetActive(false); director.Pause(); director.playableAsset = FinalHalf; director.time = 0;

director.Play();

}

public void ClickSalts(){ director.playableAsset = ClickSalt; director.Play();

}}

Figure 9. The function of this script is to handle all the animation of titration and take input by

raycasting on the object.

using UnityEngine;

using UnityEngine.Playables;

public class Chromatography : MonoBehaviour

{

public PlayableDirector director; public PlayableAsset DrawLine; public PlayableAsset Pickpencil; public PlayableAsset Pickpen; public PlayableAsset DrawDot; public PlayableAsset ClickPaper; public PlayableAsset FitPaper; public GameObject Pen;

public GameObject Pencil; public GameObject Container; public GameObject Line;

void Update(){

if (Input.GetMouseButtonDown(0)){ RaycastHit hit;

Ray ray = Camera.main.ScreenPointToRay(Input.mousePosition); if (Physics.Raycast(ray, out hit, Mathf.Infinity)) {

if (hit.collider.name == "Pencil") { director.playableAsset = DrawLine; director.Play();

}

else if (hit.collider.name == "Pen") { director.playableAsset = DrawDot; director.Play();

}

else if (hit.collider.name == "Paper") { director.playableAsset = FitPaper; director.Play();

}

}

}

}

public void ActivatePen() { Pen.SetActive(true); Pencil.SetActive(false); Line.SetActive(true); director.playableAsset = Pickpen; director.Play();

}

public void DisablePen() { Pen.SetActive(false); Container.SetActive(true); director.playableAsset = ClickPaper; director.Play();

}

public void PickPencil() { director.playableAsset = Pickpencil; director.Play();

}

}

Figure 10. The function of this script is to handle all the animation of Chromatography and take input by

raycasting on the object, also showing the reading of the results we get.

### EXPERIMENTAL RESULTS AND ANALYSIS



Figure 11. Image Target for Animal cell



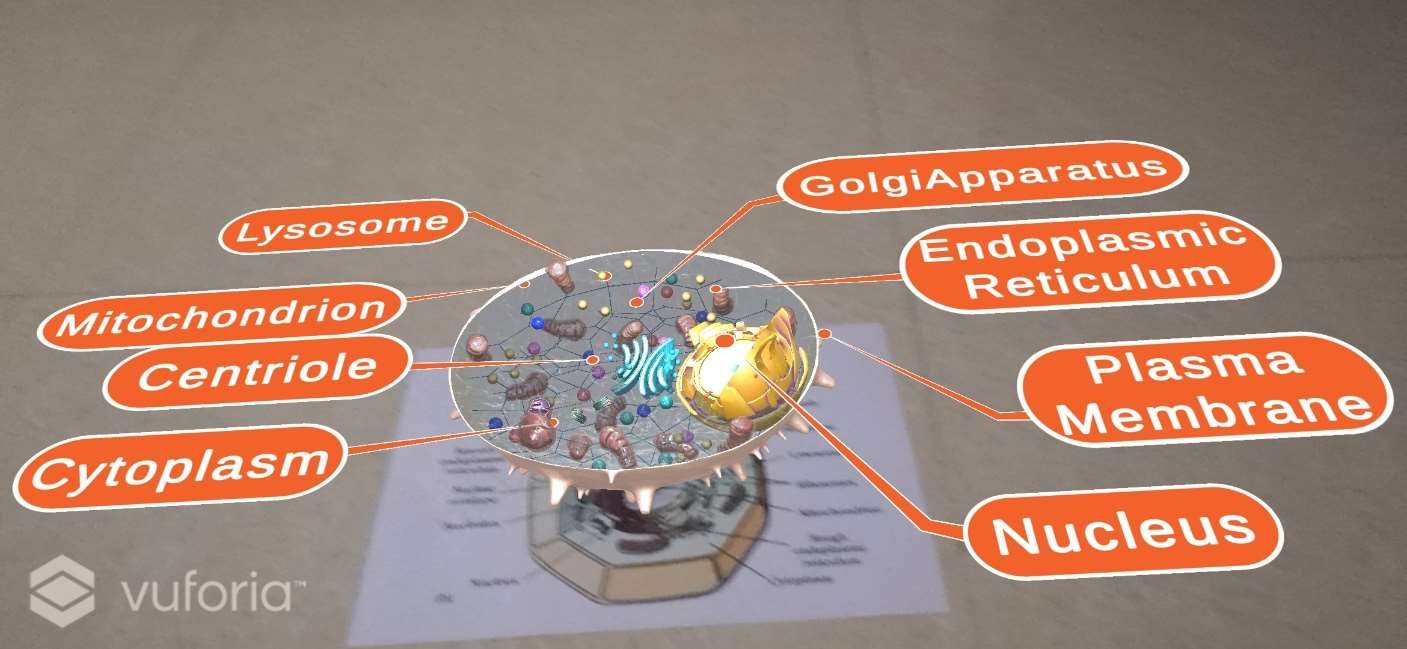


Figure 12. Output Screenshot of Animal Cell

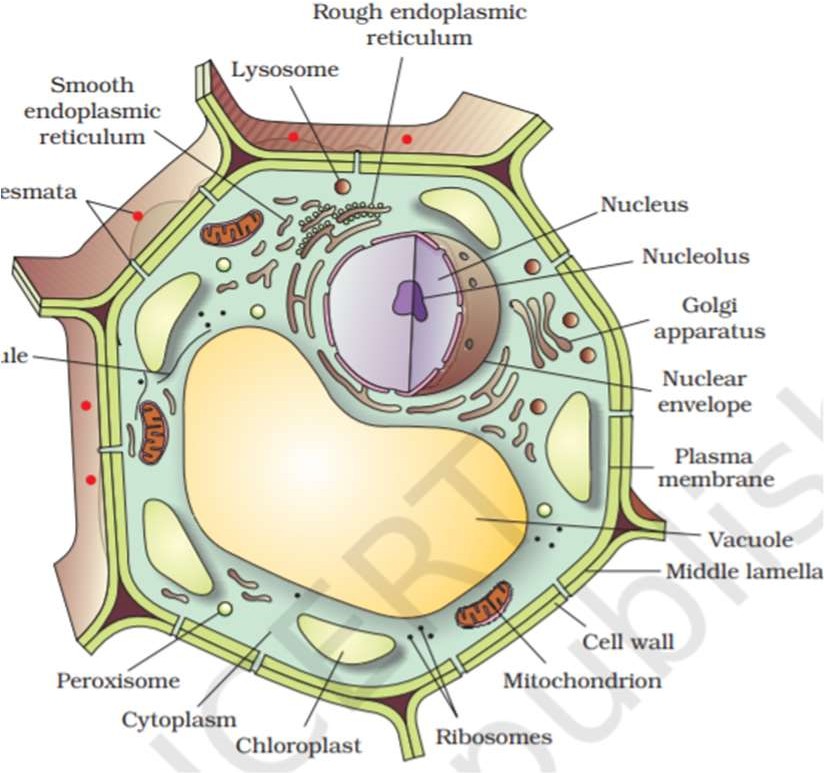


Figure 13. Image Target for Plant cell



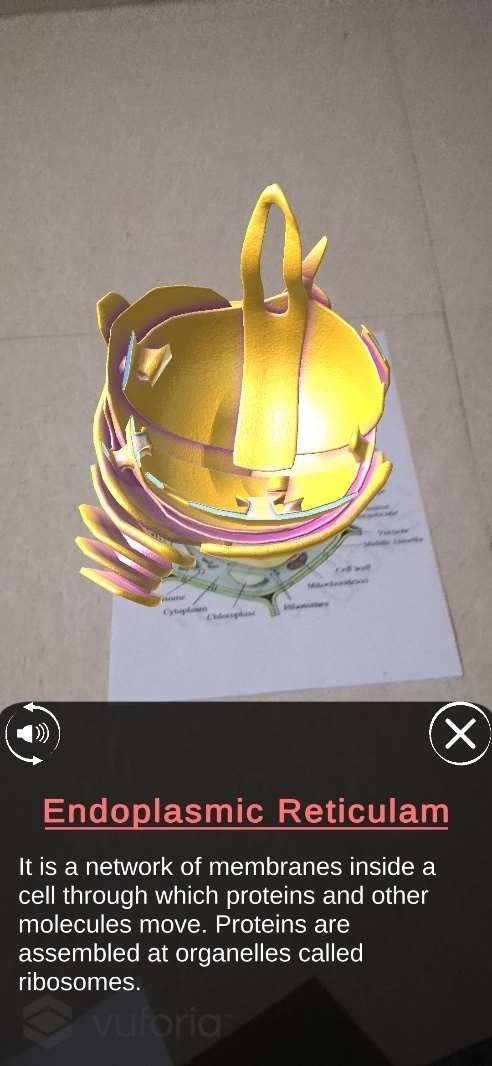


Figure 14. Output Screenshot of Plant Cell

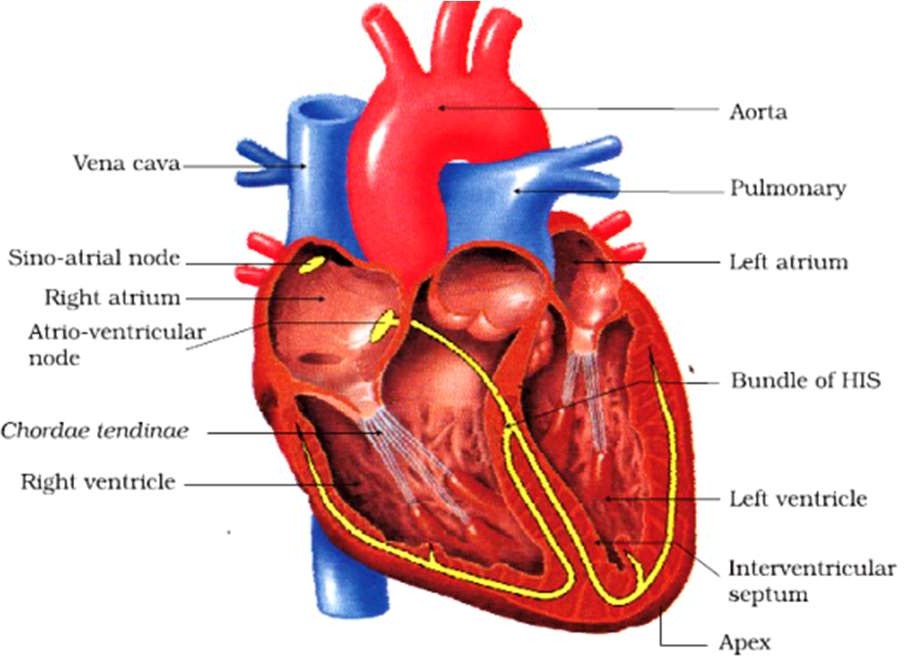
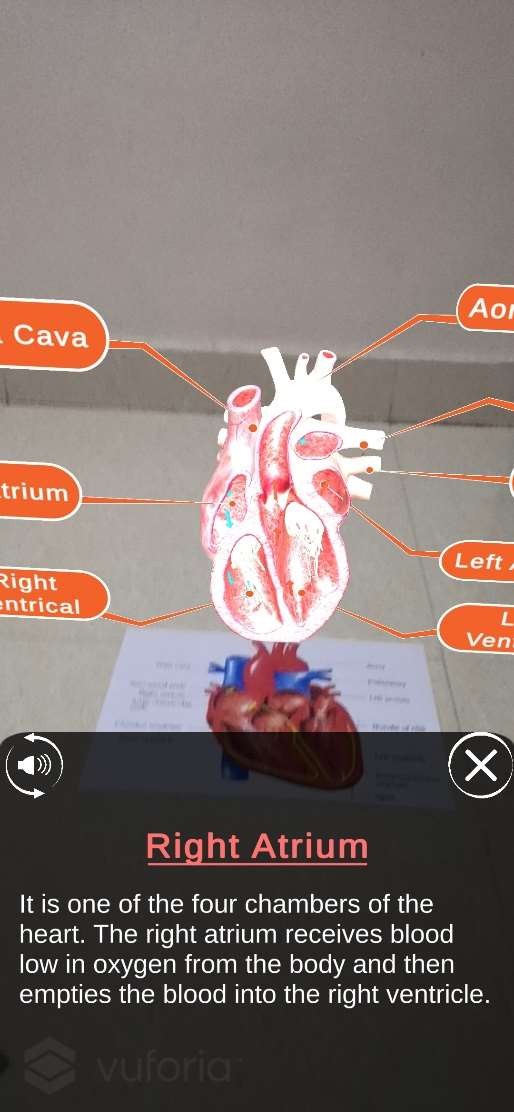


Figure 15. Image Target for Heart



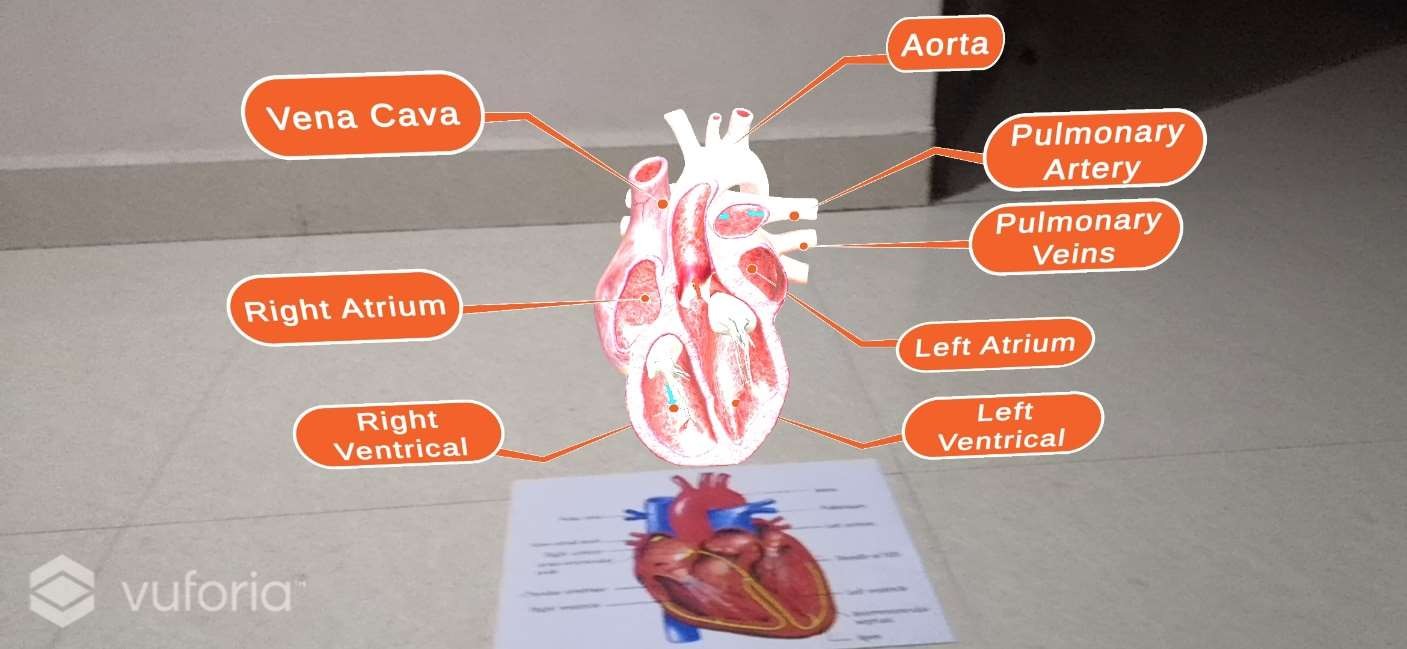


Figure 16. Output Screenshot of Heart

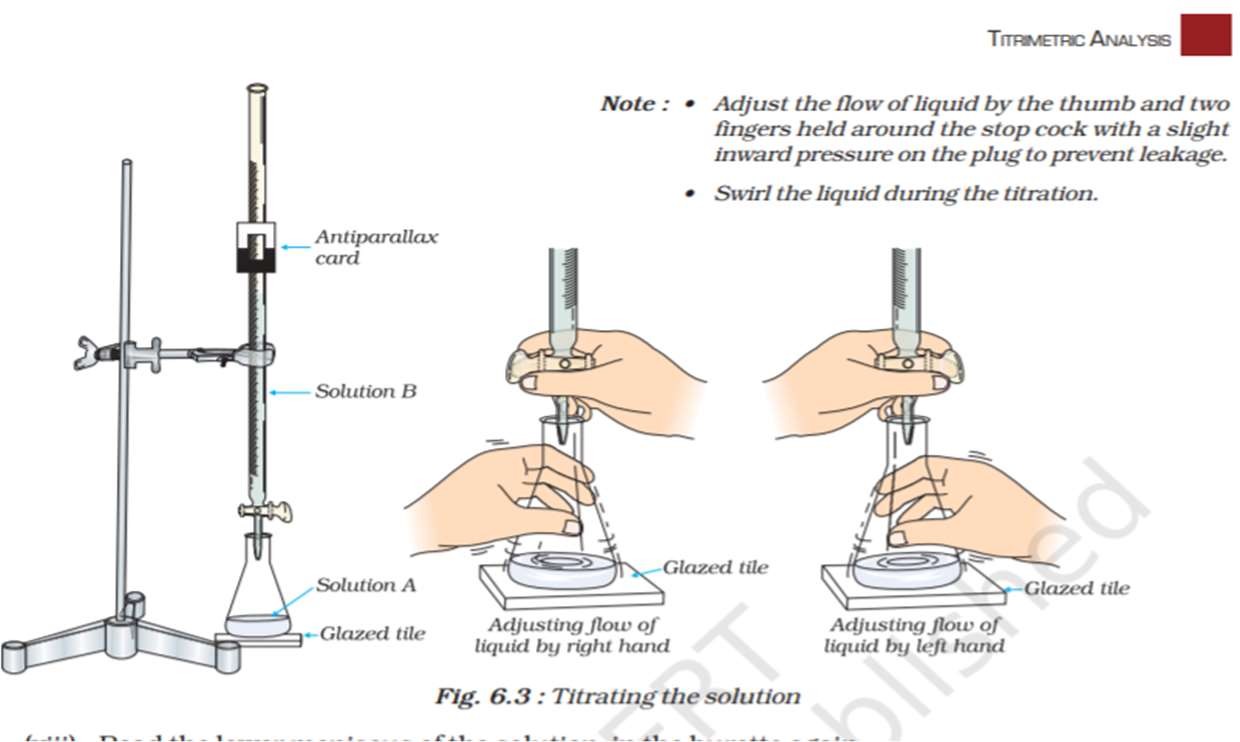


Figure 17. Image Target for Titration

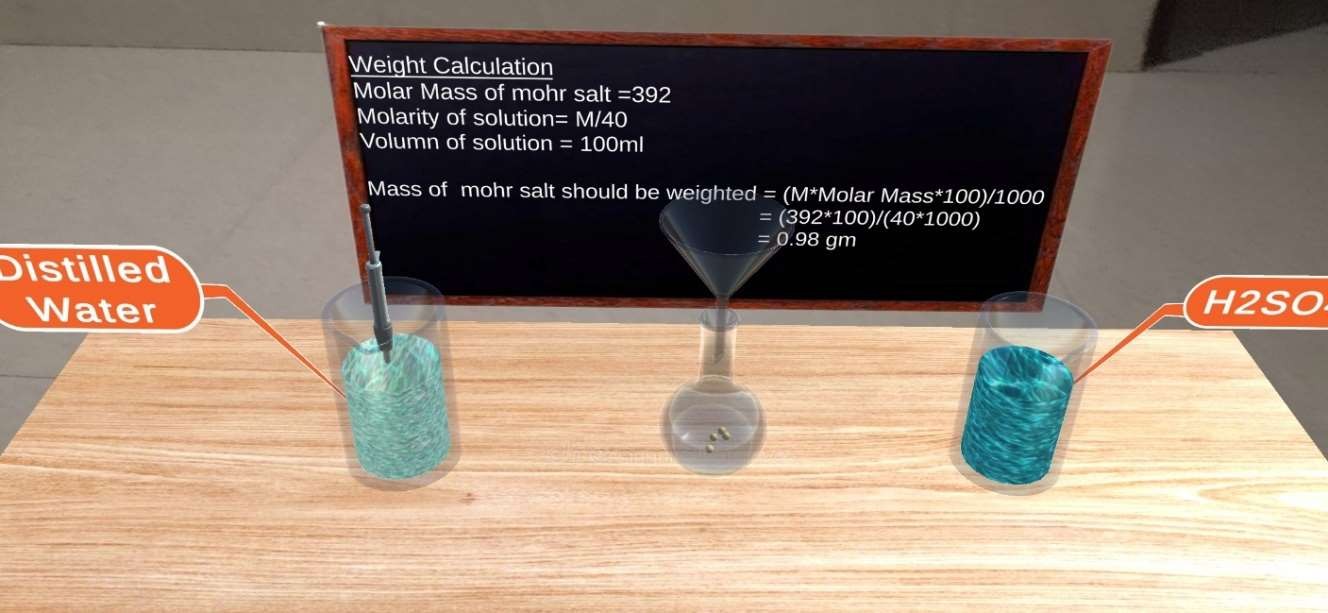


Figure 18. Output Screenshot of Titration

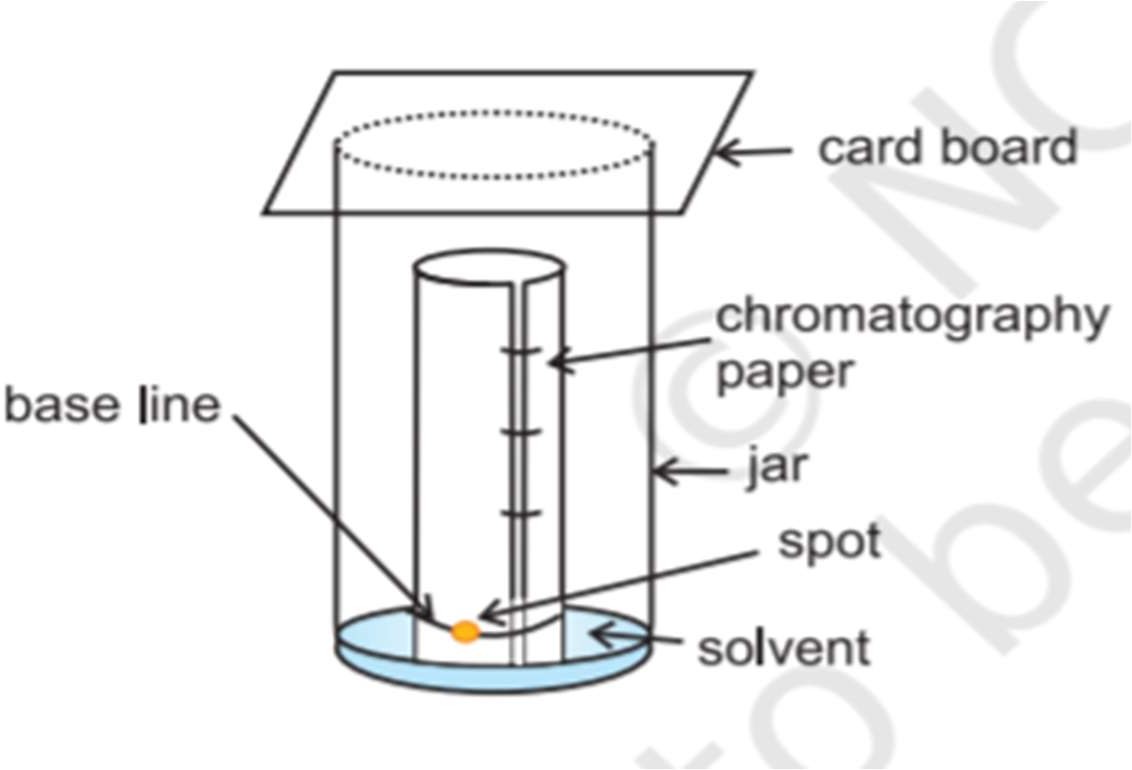


Figure 19. Image Target for Chromatography

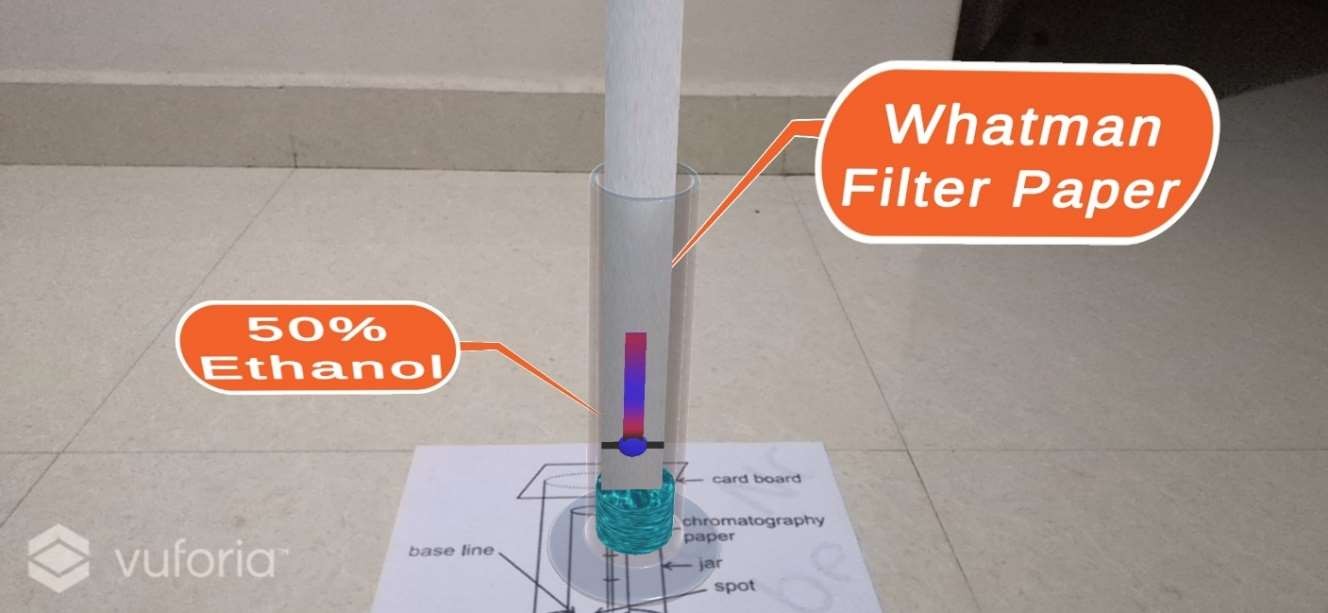


Figure 20. Output Screenshot of Chromatography

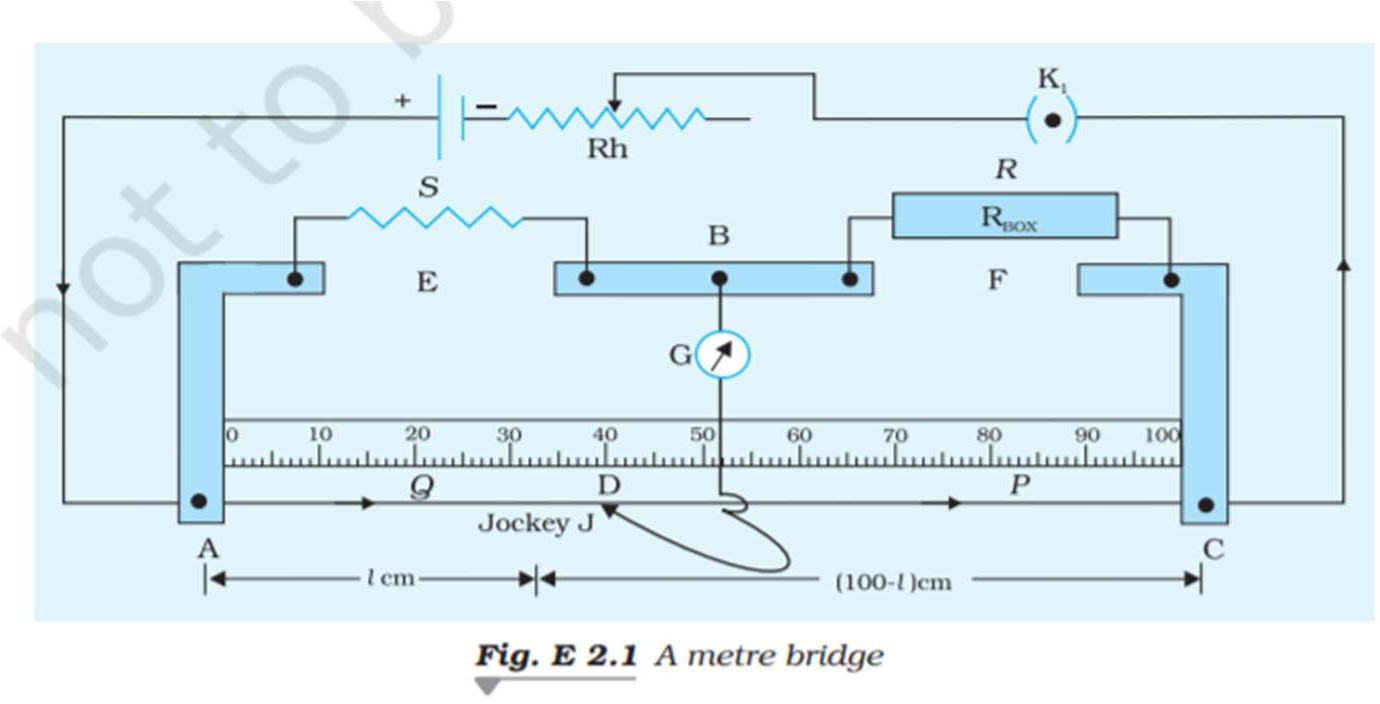


Figure 21. Image Target for Meter Bridge

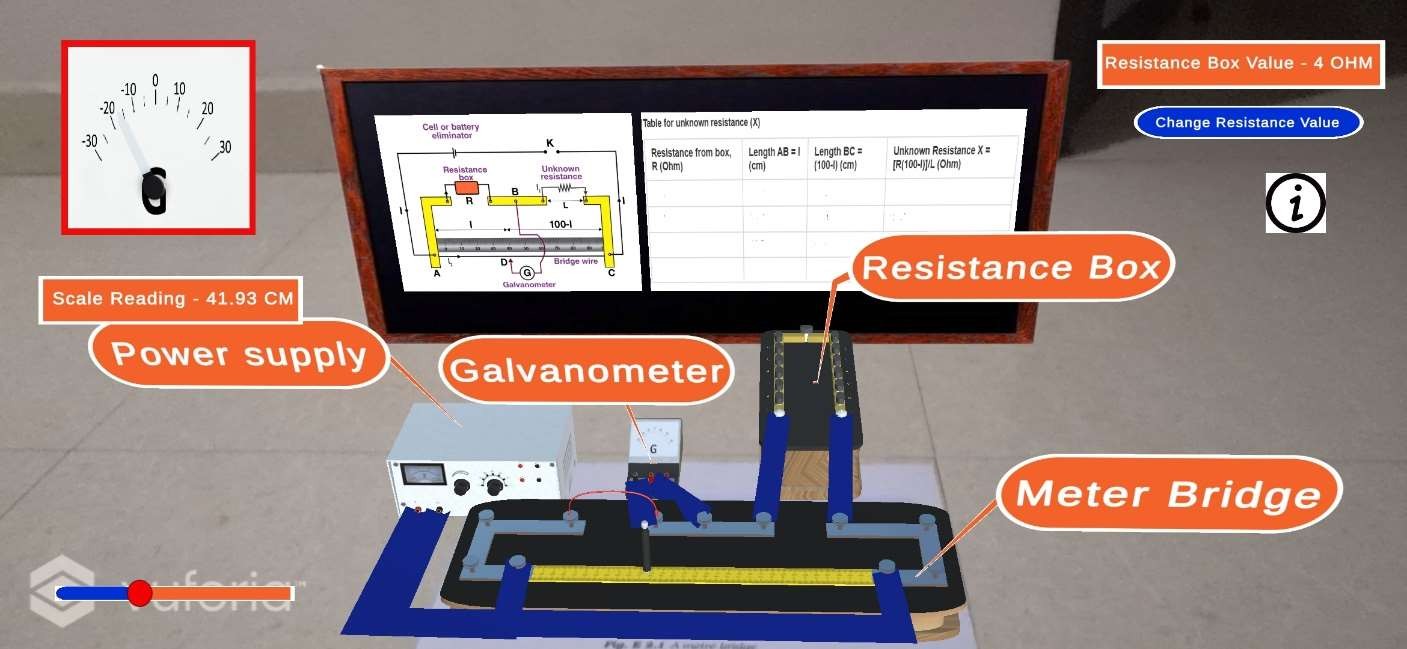


Figure 22. Output Screenshot of Meter Bridge

### CONCLUSION OF THE REPORT AND FUTURE SCOPE

AR opens up the possibility of learning that we have never experienced before and it has the potential to change the way we educate and learn forever. With the technology steadily becoming more advanced, students will become more engaged with the classroom and therefore be able to absorb information more effectively.

This Project can be extended for all classes and all subjects. It can be a great product which can be released by NCERT which will provide a better education and will help them to be in a competitive market where many education institutes like Byjus and Whitehat are using technologies to demonstrate concepts in better fashion than old school methods.

### REFERENCES:

#### Books:

1. P.M. Morse and H. Feshback, Methods of Theoretical Physics. New York: McGraw Hill, 1953.
2. Unity 2018 Augmented Reality Projects Book by Jesse Glover, Packt Publishing, 30 July 2018.
3. Complete Virtual Reality and Augmented Reality Development with Unity Book by Jesse Glover and Jonathan Linowes, Packt Publishing,17 April 2019.

# Objective

* Better Learning Experience created with augmented reality.
* Providing more practical knowledge rather than theory.
* 360-degree view of each educational resources using Augmented Reality. E.g.- Heart, Cell, etc.

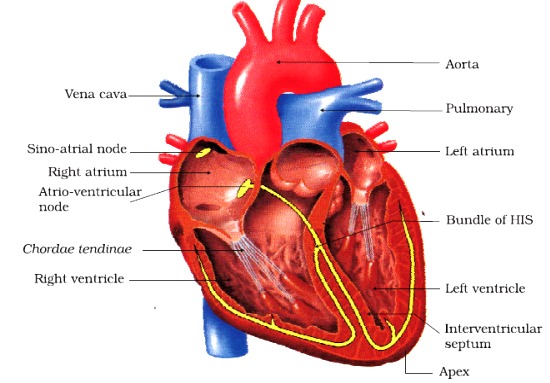
# Work Done Till 20th Apr 22

* Setting up AR environment inside Unity3D project using Vuforia SDK.
* Detecting image targets using Vuforia.
* Using these image targets to spawn the appropriate 3D models in real World using AR.
* Visualization that can be seen by moving the device around the target.
* Integrated model interaction for more information.
* Inner components will be visible by tapping on the 3D model.
* Added voice for reading the description of the components.
* Made an Android build for convenience.

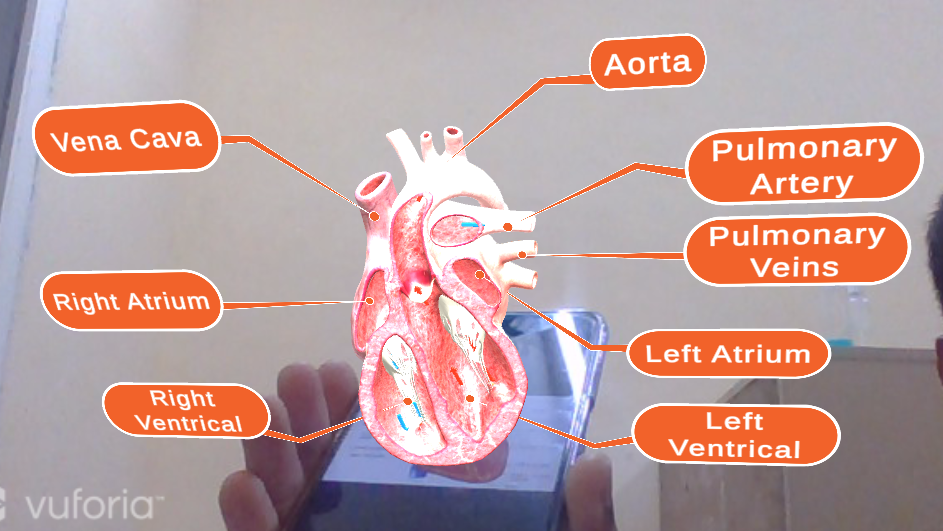
Below are the examples of image targets taken from Internet and their AR projections:

1. **HEART**

**IMAGE TARGET:**

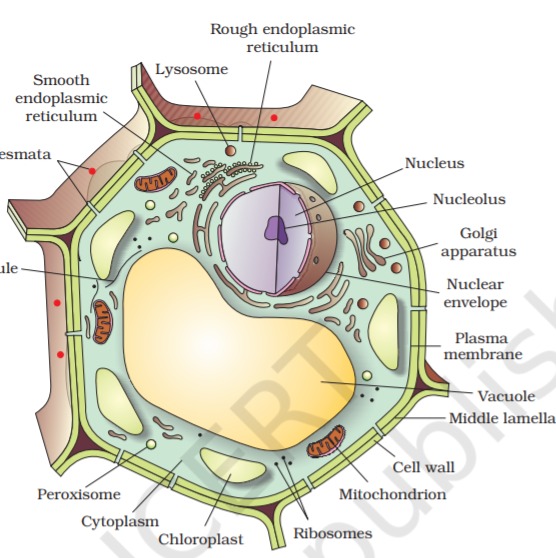


**WORKING AR:**

****

1. **PLANT CELL**

**IMAGE TARGET:**

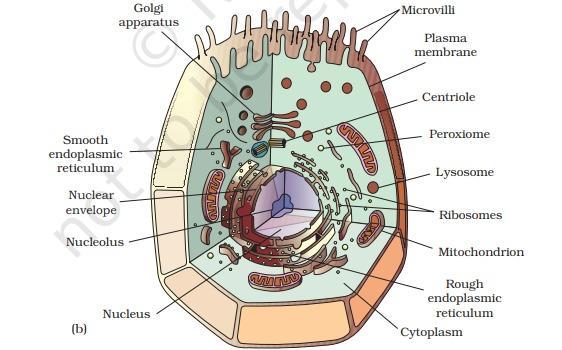
****

**WORKING AR:**

****

1. **ANIMAL CELL**

**IMAGE TARGET:**

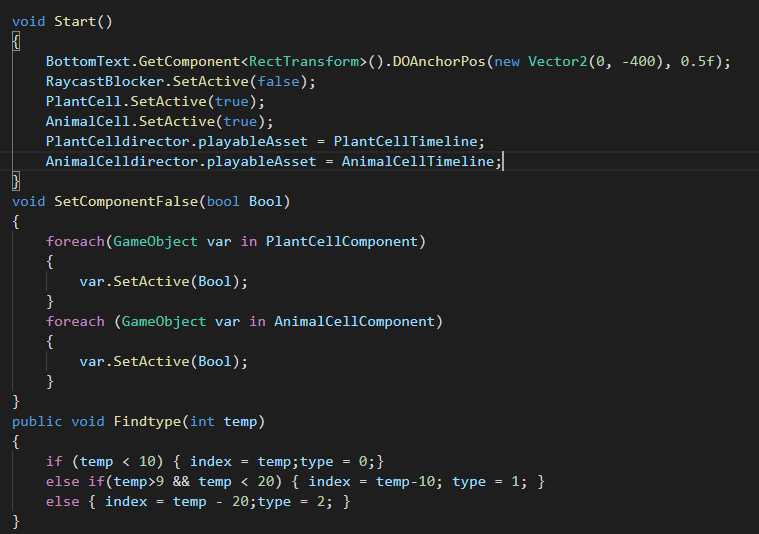


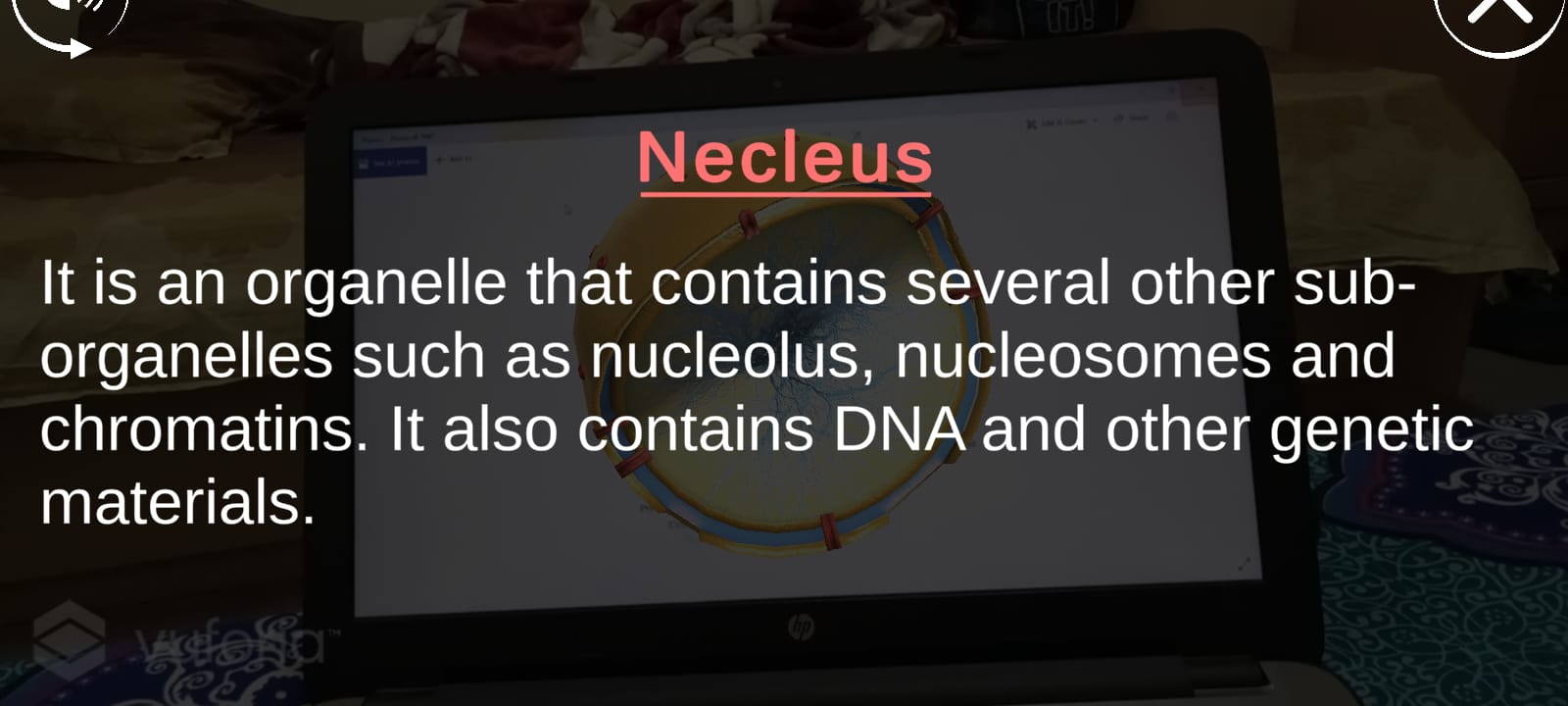
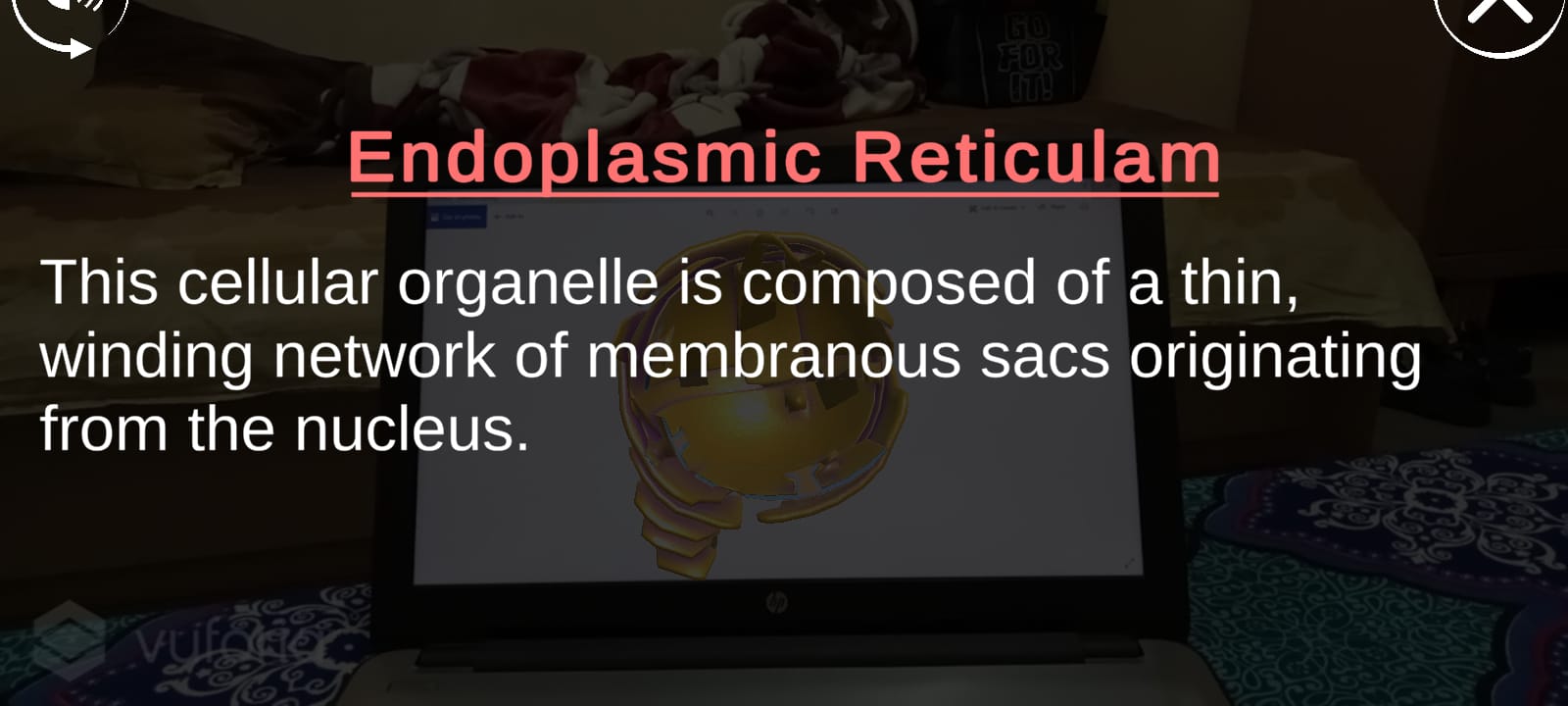
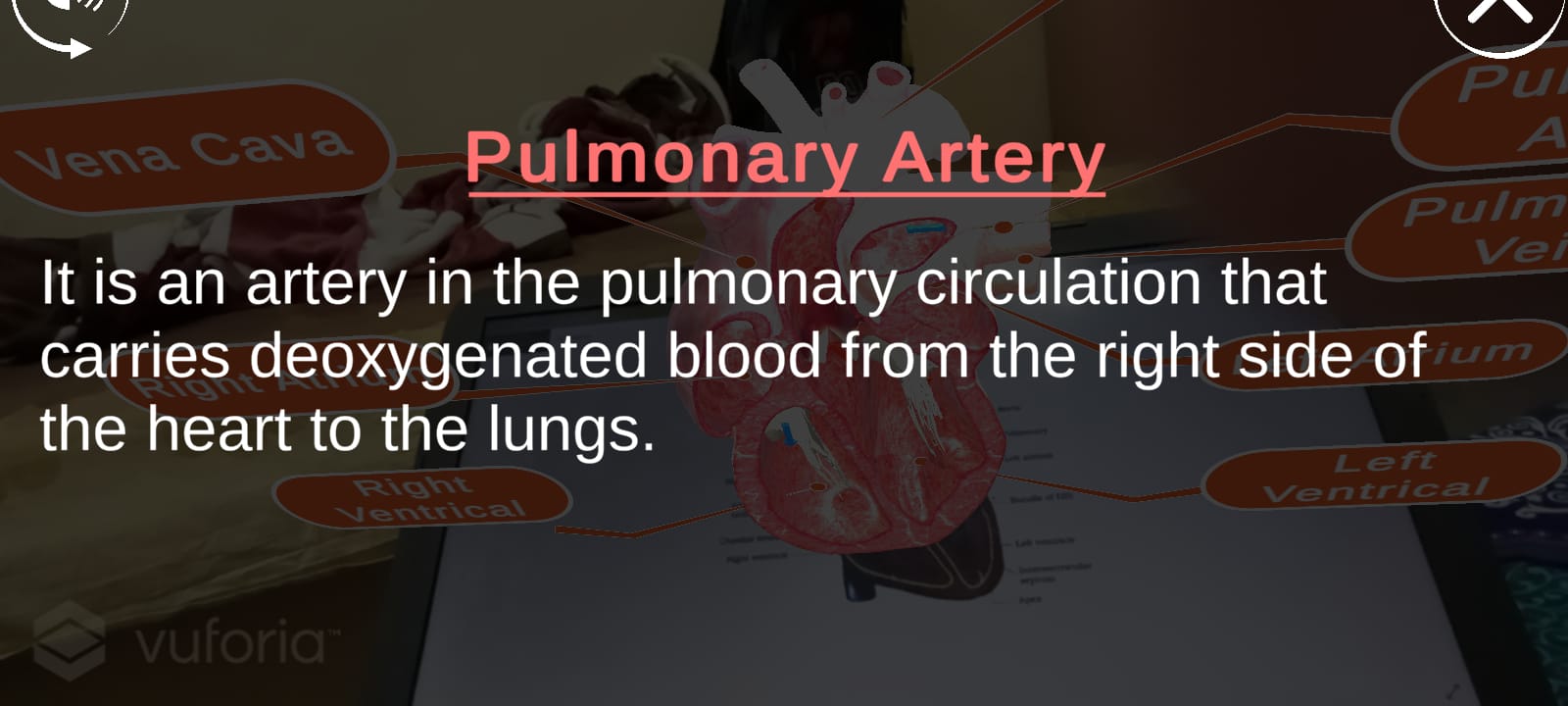
**WORKING AR:**

****

**CODE:**

****

****



**Work to be done:**

* Content(Theory + Photos + Code)
* Formatting(Spacing + Page no. + Font size & Style)
* Sequence
* Index update
* References
* SS of Unity, Code, etc.