**VIRTUAL COLLEGE TOUR**

**A Minor Project Report**

**Submitted in Partial fulfillment for the award of**

**Bachelor of Engineering in Computer Science & Engineering**

Submitted to

**RAJIV GANDHI PROUDYOGIKI VISHWAVIDYALAYA**

**BHOPAL (M.P)**



**MINOR PROJECT REPORT**

Bharat Malviya[0103CS171044]Ayush Rathore[0103CS171040]**rt**Aditya Ghule[0103CS171013] Pallavi Soni[0103CS171094]

Under the supervision of

Guide Name

Designation



**Department of Computer Science & Engineering**

**Lakshmi Narain College of Technology, Bhopal (M.P.)**

**Session 2018-19**



LAKSHMI NARAIN COLLEGE OF TECHNOLOGY, BHOPAL

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

### CERTIFICATE

This is to certify that the work embodied in this project work entitled **”Virtual College Tour”** has been satisfactorily completed by the **Bharat Malviya** (0103CS171044), **Ayush Rathore** (0103CS171040), **Pallavi Soni** (0103CS171094) , **Aditya Ghule** (0103CS171013). It is a bonafide piece of work, carried out under the guidance in **Department of Computer Science & Engineering**, **Lakshmi Narain College of Technology, Bhopal** for the partial fulfillment of the **Bachelor of Technology** during the academic year 2018-19.

**Guided By**

## (Guide Name)

Designation

**Approved By**

**Prof. & Head**

# Department of Computer Science & Engineering



LAKSHMI NARAIN COLLEGE OF TECHNOLOGY, BHOPAL

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Acknowledgement**

We express our deep sense of gratitude to Prof. ……………………. (Guide) department of Computer Science &Engineering L.N.C.T., Bhopal whose kindness valuable guidance and timely help encouraged me to complete this project.

A special thank goes to Dr. Sadhna K. Mishra (HOD) who helped me in completing this project work. She exchanged her interesting ideas & thoughts which made this project work successful.

We would also thank our institution and all the faculty members without whom this project work would have been a distant reality.

Bharat Malviya[0103CS171044]

Pallavi Soni[0103CS171094]

Ayush Rathore[0103CS171040]

Aditya Ghule[0103CS171013]

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **TOPICS** | **PAGES** |
|  | Problem Domain | 1-2 |
|  | Literature Survey | 2-3 |
|  | Major objective & scope of project. | 2-3 |
|  | Problem Analysis and requirement specification | 4-5 |
|  | Detailed Design(Modeling and ERD/DFD ) | 5-6 |
|  | Hardware/Software platform environment | 2-3 |
|  | Snapshots of Input & Output | 5-6 |
|  | Coding () | 8-10 |
|  | Project limitation and Future scope | 2-3 |
|  | References | 2-3 |
|  | Total Approx. 30-40 Pages | |

**Instruction for Submission of Project Report**

1. Chapter Name - Number : 14 (Times New Roman & All Capital Latters)
2. Heading : 16 (Times New Roman & First Latter Capital)
3. Sub Heading : 14 (Times New Roman & FLC)
4. Matter : 12 (Times New Roman )
5. Line Spacing : 1.5 pt
6. Text Alignment : Justified (cnt + j)
7. Page Margin : Left:1.25”, Right:0.75” ,Top:.1.00”,Bottom:0.25”
8. The report should be indexed and pages must be numbered all the figures and diagrams must be numbered and labeled.

**Note: Next page gives Template for making project file**

**CHAPTER 1**

**INTRODUCTION**

1. **Heading**

This template, modified in MS Word 2007 and saved as a “Word 97-2003 Document” for the PC, provides authors with most of the formatting specifications needed for preparing electronic versions of their papers.

**1.1.1 Sub Heading**

The template is used to format your paper and style the text. All margins, column widths, line spaces, and text fonts are prescribed; please do not alter them. You may note peculiarities.

## *A) Sub Sub Heading*

Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures; table heads should appear above the tables. Insert figures and tables after they are cited in the text. Use the abbreviation “Fig. 1.1”, even at the beginning

1. Table Type Styles

| Table Head | Table Column Head | | |
| --- | --- | --- | --- |
| Table column subhead | Subhead | Subhead |
| copy | More table copya |  |  |

http://cellbio.emory.edu/bnanes/figures/img/lut2.png

**CHAPTER 1**

**PROBLEM DOMAIN**

**Citizen Services in VR**

People like travelling and exploring locations, out of work, travel or entertainment. They make plans, and try to visit the places of their choice, but many a times are disappointed when the location doesn’t turn out quite as what they expected it to be.They quite often rely on the images available on the internet and expect the place to be similar to the images, but are often left displeased.Also, it itself isa tedious work of surveying or exploring just any location, especially if it is as big as a college campus, a company headquarters or say any archaeological site.

People would liketo have a visual hint of how any campus looks like, how the locality feels like, without actually physically being there, before they could plan a visit to the place. For the handicapped community, it has always beena real challenge of visiting locations as they can’talways simply go out due to obvious reasons. So there is a need of some sort of a way so that such people can easily find out how the location looks like.

**An example**

Suppose any student’s parents need to find out how the new college campus of their ward actually looks like, theneither they can visit the place themselves- which can be a tough task if the college is not of the same state, or even same country, or they can look through the images available on the internet. But the 2D images are just not sufficient enough to give a proper idea of how the campus would actually look like. How easy it could have been if one could simply stay at home and have a slight experience of their place of choicevirtually, in 3D?

**CHAPTER 2**

**LITERATURE SURVEY**

Since the first time the term "Virtual Reality" (VR) has been used back in the 60s, VR has evolved in different manners becoming more and more similar to the real world. Two different kinds of VR can be identified: non-immersive and immersive. The former is a computer-based environment that can simulate places in the real or imagined worlds; the latter takes the idea even further by giving the perception of being physically present in the non-physical world.

While non-immersive VR can be based on a standard computer, immersive VR is still evolving as the needed devices are becoming more user friendly and economically accessible. In the past, there was a major difficulty about using equipment such as a helmet with goggles, while now new devices are being developed to make usability better for the user. VR, which is based on three basic principles: Immersion, Interaction, and User involvement with the environment and narrative, offers a very high potential in education by making learning more motivating and engaging.

Up to now, the use of immersive-VR in educational games has been limited due to high prices of the devices and their limited usability. Now new tools like the commercial "Oculus Rift", make it possible to access immersive-VR in lots of educational situations. This paper reports a survey on the scientific literature on the advantages and potentials in the use of Immersive Virtual Reality in Education in the last two years . It shows how VR in general, and immersive VR in particular, has been used mostly for adult training in special situations or for university students. It then focuses on the possible advantages and drawbacks of its use in education with reference to different classes o/f users like children and some kinds of cognitive disabilities (with particular reference to the Down syndrome). It concludes outlining strategies that could be carried out to verify these ideas.

**CHAPTER 3**

**MAJOR OBJECTIVE AND**

**SCOPE OF PROJECT**

**Major Objective**

The major objective of our project is to provide one convenient way to the citizens so that they can simply stay at home and are still able to have a virtual experience of the location they plan or wish to visit to.

Our project will help citizens to get a virtual tour of any concerned campuses or organizations they plan on visiting and will help them to “visit” far-off places without actually stepping a foot out of their house. This project will thus help in saving that time and money that would be spent on an actual visit to the place.

One important objective of this project is to provide the facility of exploring places to the handicapped community.

**Scope of Project**

Currently our project can provide you with tour of our college. But with some quality work we can extend this to tours of a street, a locality, or even an entire city. In the current stage, the visuals are close to being realistic but with proper assets, the graphic quality and 3D textures can still be improved without making the application much bulkier.

At this stage, we only provide user with the external tours of the campus, but in the coming future few features can be added like interior of the buildings, navigation systems, finding shortest route, etc.

**CHAPTER 4**

**Detailed Design(Modeling and ERD/DFD )**

Since the first time the term "Virtual Reality" (VR) has been used back in the 60s, VR has evolved in different manners becoming more and more similar to the real world. Two different kinds of VR can be identified: non-immersive and immersive. The former is a computer-based environment that can simulate places in the real or imagined worlds; the latter takes the idea even further by giving the perception of being physically present in the non-physical world.

While non-immersive VR can be based on a standard computer, immersive VR is still evolving as the needed devices are becoming more user friendly and economically accessible. In the past, there was a major difficulty about using equipment such as a helmet with goggles, while now new devices are being developed to make usability better for the user. VR, which is based on three basic principles: Immersion, Interaction, and User involvement with the environment and narrative, offers a very high potential in education by making learning more motivating and engaging.

Up to now, the use of immersive-VR in educational games has been limited due to high prices of the devices and their limited usability. Now new tools like the commercial "Oculus Rift", make it possible to access immersive-VR in lots of educational situations. This paper reports a survey on the scientific literature on the advantages and potentials in the use of Immersive Virtual Reality in Education in the last two years . It shows how VR in general, and immersive VR in particular, has been used mostly for adult training in special situations or for university students. It then focuses on the possible advantages and drawbacks of its use in education with reference to different classes o/f users like children and some kinds of cognitive disabilities (with particular reference to the Down syndrome). It concludes outlining strategies that could be carried out to verify these ideas.

**CHAPTER 5**

**PROBLEM ANALYSIS AND**

**REQUIREMENT SPECIFICATION**

**Problem Analysis**

The problem involved the inability of a citizen to visit a particular location/campus, due to various reasons like the place being very far away from the citizen’s city/area, lack of time and money, disabilities like being handicapped etc.

We worked out a smart solution for the stated problem and were able to develop a virtual reality tour for the college campus, which included the main structural objects of the campus, like the college buildings, hostels, temple, trees, etc. Any person with a VR headset on and a smartphone with the application installed upon it can easily experience the virtual tour of the college campus.

With the help of the hand-held controllers, user can move according to his will within the virtual campus.

http://cellbio.emory.edu/bnanes/figures/img/lut2.png

**CHAPTER 6**

**HARDWARE/SOFTWARE**

**PLATFORM ENVIRONMENT**

**1. Hardware**

Since the project is a virtual tour, a VR headset along with its hand-held controllers is a must paired with a smartphone which has enough CPU power and memory to handle a large application.

## *A) VR headset and controllers*

Various VR headsets like VR Box, Oculus Go, Oculus Rift, HTC Vive, and many more can be used for the tour. The only necessity is that of the hand-held controllers which will be used to move around in the tour.

## *B) Smartphone*

The application is a bit on the heavier side, so a smartphone with at least 4 GBs of RAM and a mid-tier CPU is needed for seamless performance of the tour.

**2. Software**

The entire project was build using Unity 3D (for platform), Blender (for models) and C# scripts (for VR compatibility).

## *A) Unity 3D*

Unity3D is a commercially available multiplatform game engine used for the production of 2d and 3D video games as well as non-game interactive simulations and visualizations. The base of our project is provided by Unity3D

## *B) Blender*

Blender is an open-source 3D computer graphics software toolset used for creating animated films, visual effects, art, 3D models, motion graphics, interactive 3D applications, and computer games. We used blender to create the models of various buildings, parks, pathways, etc. which were then imported in Unity.

## *C) C#*

C# can be used to create almost anything but is particularly strong at building Windows desktop applications and games. We used C# as our major scripting tool.

http://cellbio.emory.edu/bnanes/figures/img/lut2.png

**CHAPTER 8**

**CODING**

**First Person Controller (Modified API)**

using System;

usingUnityEngine;

usingUnityStandardAssets.CrossPlatformInput;

usingUnityStandardAssets.Utility;

using Random = UnityEngine.Random;

namespaceUnityStandardAssets.Characters.FirstPerson

{

[RequireComponent(typeof (CharacterController))]

[RequireComponent(typeof (AudioSource))]

public class FirstPersonController : MonoBehaviour

{

[SerializeField] private bool m\_IsWalking;

[SerializeField] private float m\_WalkSpeed;

[SerializeField] private float m\_RunSpeed;

[SerializeField] [Range(0f, 1f)] private float m\_RunstepLenghten;

[SerializeField] private float m\_JumpSpeed;

[SerializeField] private float m\_StickToGroundForce;

[SerializeField] private float m\_GravityMultiplier;

[SerializeField] private MouseLookm\_MouseLook;

[SerializeField] private bool m\_UseFovKick;

[SerializeField] private FOVKickm\_FovKick = new FOVKick();

[SerializeField] private bool m\_UseHeadBob;

[SerializeField] private CurveControlledBobm\_HeadBob = new CurveControlledBob();

[SerializeField] private LerpControlledBobm\_JumpBob = new LerpControlledBob();

[SerializeField] private float m\_StepInterval;

[SerializeField] private AudioClip[] m\_FootstepSounds;

[SerializeField] private AudioClipm\_JumpSound; // the sound played when character leaves the ground.

[SerializeField] private AudioClipm\_LandSound; // the sound played when character touches back on ground.

private Camera m\_Camera;

private bool m\_Jump;

private float m\_YRotation;

private Vector2 m\_Input;

private Vector3 m\_MoveDir = Vector3.zero;

privateCharacterControllerm\_CharacterController;

privateCollisionFlagsm\_CollisionFlags;

private bool m\_PreviouslyGrounded;

private Vector3 m\_OriginalCameraPosition;

private float m\_StepCycle;

private float m\_NextStep;

private bool m\_Jumping;

privateAudioSourcem\_AudioSource;

[HideInInspector] public Vector2 runaxis;

//[HideInInspector] public bool jumpaxis;

private void Start()

{

m\_CharacterController = GetComponent<CharacterController>();

m\_Camera = Camera.main;

m\_OriginalCameraPosition = m\_Camera.transform.localPosition;

m\_FovKick.Setup(m\_Camera);

m\_HeadBob.Setup(m\_Camera, m\_StepInterval);

m\_StepCycle = 0f;

m\_NextStep = m\_StepCycle/2f;

m\_Jumping = false;

m\_AudioSource = GetComponent<AudioSource>();

m\_MouseLook.Init(transform , m\_Camera.transform);

}

private void Update()

{

RotateView();

if (!m\_Jump)

{

m\_Jump = CrossPlatformInputManager.GetButtonDown("Jump");

}

if (!m\_PreviouslyGrounded&&m\_CharacterController.isGrounded)

{

StartCoroutine(m\_JumpBob.DoBobCycle());

PlayLandingSound();

m\_MoveDir.y = 0f;

m\_Jumping = false;

}

if (!m\_CharacterController.isGrounded&& !m\_Jumping&&m\_PreviouslyGrounded)

{

m\_MoveDir.y = 0f;

}

m\_PreviouslyGrounded = m\_CharacterController.isGrounded;

}

private void PlayLandingSound()

{

m\_AudioSource.clip = m\_LandSound;

m\_AudioSource.Play();

m\_NextStep = m\_StepCycle + .5f;

}

private void FixedUpdate()

{

float speed;

GetInput(out speed);

Vector3 desiredMove = transform.forward\*m\_Input.y + transform.right\*m\_Input.x;

RaycastHithitInfo;

Physics.SphereCast(transform.position, m\_CharacterController.radius, Vector3.down, out hitInfo,

m\_CharacterController.height/2f, Physics.AllLayers, QueryTriggerInteraction.Ignore);

desiredMove = Vector3.ProjectOnPlane(desiredMove, hitInfo.normal).normalized;

m\_MoveDir.x = desiredMove.x\*speed;

m\_MoveDir.z = desiredMove.z\*speed;

if (m\_CharacterController.isGrounded)

{

m\_MoveDir.y = -m\_StickToGroundForce;

if (m\_Jump)

{

m\_MoveDir.y = m\_JumpSpeed;

PlayJumpSound();

m\_Jump = false;

m\_Jumping = true;

}

}

else

{

m\_MoveDir += Physics.gravity\*m\_GravityMultiplier\*Time.fixedDeltaTime;

}

m\_CollisionFlags = m\_CharacterController.Move(m\_MoveDir\*Time.fixedDeltaTime);

ProgressStepCycle(speed);

UpdateCameraPosition(speed);

m\_MouseLook.UpdateCursorLock();

}

private void PlayJumpSound()

{

m\_AudioSource.clip = m\_JumpSound;

m\_AudioSource.Play();

}

private void ProgressStepCycle(float speed)

{

if (m\_CharacterController.velocity.sqrMagnitude> 0 && (m\_Input.x != 0 || m\_Input.y != 0))

{

m\_StepCycle += (m\_CharacterController.velocity.magnitude + (speed\*(m\_IsWalking ? 1f : m\_RunstepLenghten)))\*

Time.fixedDeltaTime;

}

if (!(m\_StepCycle>m\_NextStep))

{

return;

}

m\_NextStep= m\_StepCycle + m\_StepInterval;

PlayFootStepAudio();

}

private void PlayFootStepAudio()

{

if (!m\_CharacterController.isGrounded)

{

return;

}

int n = Random.Range(1, m\_FootstepSounds.Length);

m\_AudioSource.clip = m\_FootstepSounds[n];

m\_AudioSource.PlayOneShot(m\_AudioSource.clip);

m\_FootstepSounds[n] = m\_FootstepSounds[0];

m\_FootstepSounds[0] = m\_AudioSource.clip;

}

private void UpdateCameraPosition(float speed)

{

Vector3 newCameraPosition;

if (!m\_UseHeadBob)

{

return;

}

if (m\_CharacterController.velocity.magnitude> 0 &&m\_CharacterController.isGrounded)

{

m\_Camera.transform.localPosition =

m\_HeadBob.DoHeadBob(m\_CharacterController.velocity.magnitude +

(speed\*(m\_IsWalking ? 1f : m\_RunstepLenghten)));

newCameraPosition = m\_Camera.transform.localPosition;

newCameraPosition.y = m\_Camera.transform.localPosition.y - m\_JumpBob.Offset();

}

else

{

newCameraPosition = m\_Camera.transform.localPosition;

newCameraPosition.y = m\_OriginalCameraPosition.y - m\_JumpBob.Offset();

}

m\_Camera.transform.localPosition = newCameraPosition;

}

private void GetInput(out float speed)

{

// Read input

//android

float horizontal = runaxis.x;

float vertical = runaxis.y;

//windows

//float horizontal = CrossPlatformInputManager.GetAxis("Horizontal");

//float vertical = CrossPlatformInputManager.GetAxis("Vertical");

boolwaswalking = m\_IsWalking;

#if !MOBILE\_INPUT

m\_IsWalking= !Input.GetKey(KeyCode.LeftShift);

#endif

speed = m\_IsWalking ? m\_WalkSpeed :m\_RunSpeed;

m\_Input = new Vector2(horizontal, vertical);

if (m\_Input.sqrMagnitude> 1)

{

m\_Input.Normalize();

}

if (m\_IsWalking != waswalking&&m\_UseFovKick&&m\_CharacterController.velocity.sqrMagnitude> 0)

{

StopAllCoroutines();

StartCoroutine(!m\_IsWalking ? m\_FovKick.FOVKickUp() : m\_FovKick.FOVKickDown());

}

}

private void RotateView()

{

m\_MouseLook.LookRotation (transform, m\_Camera.transform);

}

private void OnControllerColliderHit(ControllerColliderHit hit)

{

Rigidbody body = hit.collider.attachedRigidbody;

if (m\_CollisionFlags == CollisionFlags.Below)

{

return;

}

if (body == null || body.isKinematic)

{

return;

}

body.AddForceAtPosition(m\_CharacterController.velocity\*0.1f, hit.point, ForceMode.Impulse);

}

}

}

**Gaze**

usingSystem.Collections;

usingSystem.Collections.Generic;

usingUnityEngine;

public class Gaze : MonoBehaviour

{

publicGameObject reticle;

public Color inactivecolor = Color.grey;

public Color activecolor = Color.green;

privateGazeableObjectscurrentGazeObject;

privateGazeableObjectscurrentselectedobject;

privateRaycastHitlasthit;

privateintgazecounter;

privateGameObject menu;

privateGameObject child1;

void Start()

{

SetReticleColor(inactivecolor);

}

void Update()

{

Gazein();

CheckForInput(lasthit);

}

public void Gazein()

{

Ray raycastray = new Ray(transform.position, transform.forward);

RaycastHithitInfo;

Debug.DrawRay(raycastray.origin, raycastray.direction \* 100);

if (Physics.Raycast(raycastray, out hitInfo))

{

GameObjecthitobj = hitInfo.collider.gameObject;

GazeableObjectsgazeobj = hitobj.GetComponent<GazeableObjects>(); //what it do

if (gazeobj != null)

{

if (gazeobj != currentGazeObject)

{

ClearCurrentObject();

currentGazeObject = gazeobj;

currentGazeObject.OnGazeEnter(hitInfo);

SetReticleColor(activecolor);

}

else

{

gazecounter++;

if (gazecounter == 100)

{

string x = gazeobj.name;

menu = GameObject.Find("Canvas");

child1 = menu.transform.Find(x).gameObject;

//Debug.Log(x);

//Debug.Log(menu.name);

//Debug.Log(child1.name);

child1.SetActive(true);

}

currentGazeObject.OnGaze(hitInfo);

}

}

else

{

ClearCurrentObject();

gazecounter = 0;

child1.SetActive(false);

}

lasthit = hitInfo;

}

else

{

ClearCurrentObject();

}

}

public void SetReticleColor(Color reticleColor)

{

reticle.GetComponent<Renderer>().material.color = reticleColor;

}

private void CheckForInput(RaycastHithitInfo)

{

if (Input.GetMouseButtonDown(0) &&currentGazeObject != null)

{

currentselectedobject = currentGazeObject;

currentselectedobject.OnPress(hitInfo);

}

else if (Input.GetMouseButton(0) &&currentGazeObject != null)

{

currentselectedobject.OnHold(hitInfo);

}

else if (Input.GetMouseButtonUp(0) &&currentGazeObject != null)

{

currentselectedobject.OnRelease(hitInfo);

currentselectedobject = null;

}

}

private void ClearCurrentObject()

{

if (currentGazeObject != null)

{

currentGazeObject.OnGazeExit();

SetReticleColor(inactivecolor);

currentGazeObject = null;

}

}

}

**Gazeable objects**

usingSystem.Collections;

usingSystem.Collections.Generic;

usingUnityEngine;

public class GazeableObjects : MonoBehaviour

{

public virtual void OnGazeEnter(RaycastHithitInfo)

{

Debug.Log("Gaze entered on " + gameObject.name);

}

public virtual void OnGaze(RaycastHithitInfo)

{

Debug.Log("Gaze hold on " + gameObject.name);

{

public virtual void OnGazeExit()

{

Debug.Log("Gaze exited on " + gameObject.name);

}

public virtual void OnPress(RaycastHithitInfo)

{

}

public virtual void OnHold(RaycastHithitInfo)

{

}

public virtual void OnRelease(RaycastHithitInfo)

{

}

}

**GYRO CONTROLS**

usingSystem.Collections;

usingSystem.Collections.Generic;

usingUnityEngine;

public class GyroCpntrole : MonoBehaviour

{

private bool gyroEnabled;

private Gyroscope gyro;

privateGameObjectcameraContainer;

private Quaternion rot;

void Start()

{

cameraContainer = new GameObject("Camera Container");

cameraContainer.transform.position = transform.position;

transform.SetParent(cameraContainer.transform);

gyroEnabled = EnableGyro();

}

private bool EnableGyro()

{

if (SystemInfo.supportsGyroscope)

{

gyro = Input.gyro;

gyroEnabled = true;

cameraContainer.transform.rotation = Quaternion.Euler(90f, 90f, 0f);

rot = new Quaternion(0, 0, 1, 0);

return true;

}

return false;

}

void Update()

{

if (gyroEnabled)

{

transform.localRotation = gyro.attitude \* rot;

}

}

}

**Joystick Control**

usingSystem.Collections;

usingSystem.Collections.Generic;

usingUnityEngine;

usingUnityStandardAssets.Characters.FirstPerson;

usingUnityStandardAssets.CrossPlatformInput;

usingUnityStandardAssets.Utility;

public class joystickcontrol : MonoBehaviour

{

// Start is called before the first frame update

void Start()

{

}

// Update is called once per frame

void Update()

{

var fps = GetComponent<FirstPersonController>();

fps.runaxis.x = Input.GetAxis("Vertical")\*-1;

fps.runaxis.y = Input.GetAxis("Horizontal")\*-1;

Debug.Log(fps.runaxis.x + " " + fps.runaxis.y);

}

}

//for android

**CHAPTER 9**

**PROJECT LIMITATION**

**AND FUTURE SCOPE**

**Project Limitations**

One of the major limitations of our project is that for creating proper models, various images of the location, blueprints, layouts, etc. are needed which might not be always available or legal for public use. Without proper blueprints, the virtual campus may turn out to be a lot different than original in terms of layout and orientations.

Another limitation may include the need of a VR headset, which itself is not a cheap item and not everyone can afford it. Also the smartphone on which the application would run needs to be of one the high-tier models because without adequate RAM availability and CPU performance the virtual tour becomes very laggy which will affect the visual experience that we aim to provide the user.

Currently our project can provide people with a close-to-real virtual experience of the campus, but it still isn’t “real” enough. Increasing the quality of textures and shadows will make the application run even heavier than what it originally is, and not every smartphone would be able to run it without lags.

**Future Scope`**

Currently our project can provide you with tour of our college. But with some quality work we can extend this to tours of a street, a locality, or even an entire city. In the current stage, the visuals are close to being realistic but with proper assets, the graphic quality and 3D textures can still be improved without making the application much bulkier.

At this stage, we only provide user with the external tours of the campus, but in the coming future few features can be added like interior of the buildings, navigation systems, finding shortest route, etc.