

Technische Hochschule Ingolstadt - Fakultät Informatik

User Experience Design Master

Virtual and Augmented Reality Applications



**xSpace: an educational virtual reality application – project report**

Winter Term 2022/23

Submitted by: Nurkhon Akhmedov, Emely Rosbach, Tingnan Li

Submission date: 18.01.2023

Supervisor: Prof. Dr. Munir Georges, Andreas Löcken

## Meta data

### Biographies

The xSpace project group (Group B) consists of Nurkhon Akhmedov, Emely Rosbach and Tingnan Li, who all have different backgrounds and experience levels with Unity and virtual reality applications. By coming together to achieve a common goal, we were not only able to share our personal expertise with each other, but also acquire new knowledge and skills during the project development cycle.

The table depicted below contains a brief bio for each project group member including their name, role, relevant experience and newly obtained knowledge and skills, acquired through working on the xSpace project.

Name	Role	Relevant experience	What we've learned
Nurkhon Akhmedov	Project Manager, Unity Specialist, UI/UX Designer	Unity (2D and 3D) using C#, Audio Editing, Project Management (Software projects)	Conducting Usability Tests for a VR project, GitHub using Terminal
Tingnan Li	3D Modeling, Assets Management, UI/UX Designer, Audio, and Video Rendering	3D Modeling and rendering (Cinema 4D, Fusion 360, Redshift), Media editing. UI/UX Design.	VR game development in Unity.

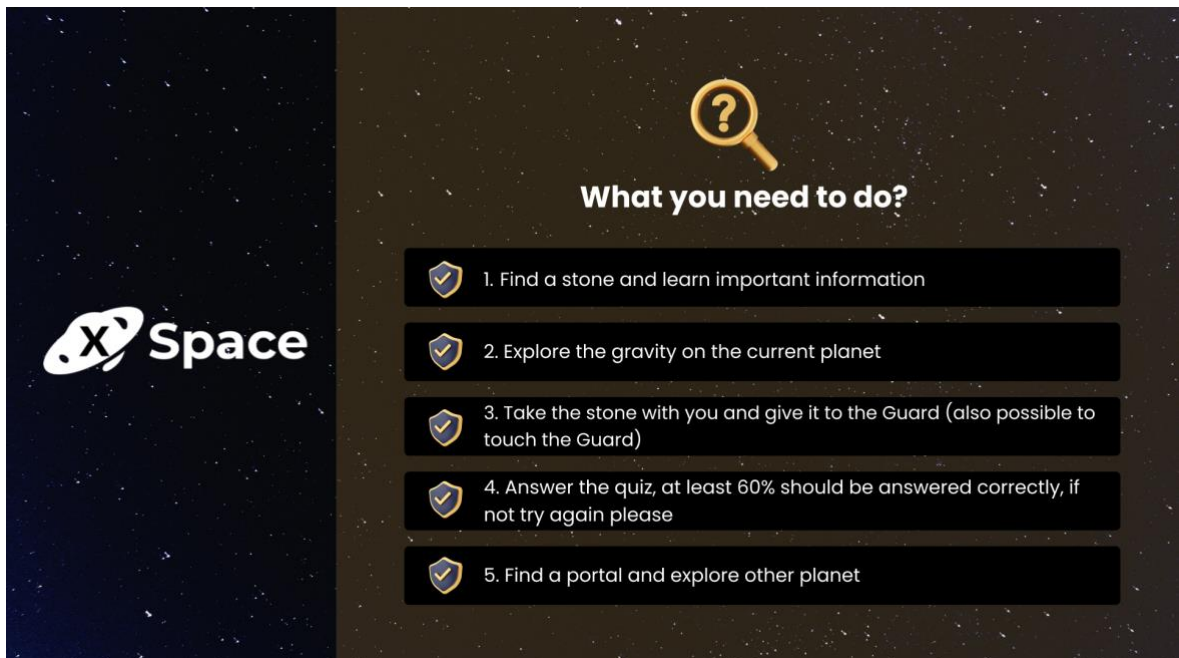
Emely Rosbach	Unity Developer (C#), GitHub Master, UX Specialist	Java Developer, UX-Specialist, Practical experience with GitHub	Developing in Unity and C# (with no prior experience), Conducting /Usability Tests for a VR project
---------------	--	---	---

## Instructions

The xSpace GitHub repository can be accessed via the following link:

<https://github.com/emelyrosbach/xSpace>

The “master” branch holds the latest version of the xSpace application, while the project report and the .zip-file containing the executable can be found inside the “final” branch. All presentations are located inside the “presentation” branch.



The xSpace application can be either started through the Unity Editor (scene “xSpace”) or by double-clicking the executable file. The figure above depicts the

tasks the player must complete, which is also shown alongside additional controller explanations as a tutorial inside the xSpace game.

# Table of Contents

## **Meta data**

Biographies

Instructions

## **1. Introduction**

1.1. General project overview

1.2. Problems addressed by the project

1.3. Project objectives

## **2. Project conceptualization and implementation**

2.1. Project conceptualization

2.2. User personas

2.3. Methods used to create content (modeling, animation, media, sound)

2.4. Technical integration

2.4.1 Technical requirements

2.4.2 Development tools and environments

2.4.3 Code implementation

2.5. Usability testing and results (plan, conduction, result analysis)

2.6. Improvements (Guidance)

2.7. Limitations (technical, gamification)

2.8. Reflection on objectives

## **3. Summary and Self-Evaluation**

## **4. List of references**

## **5. Appendices**

## List of Abbreviations

Abbreviation	Term
VR	Virtual Reality
UX	User Experience
UI	User Interface
Unity	Unity Game Engine
OC	Oculus
3D	Three-Dimensional
3D Modeling	The process of creating a three-dimensional mathematical representation of a structure or object using 3D modeling software.
IDE	Integrated Development Environment
API	Application Programming Interface
VRQ	Virtual Reality Quiz
DGR	Different Gravities
DS	Different Sounds
Activity Diagram	Graphical representation of the flow
C#	C Sharp Programming Language

GM	The Game Manager is a crucial component in the development of any game, it is responsible for managing and controlling various aspects of the game, such as player movement, collision detection, score keeping, and game state management.
ETD	Estimated Time to Develop
NASA	National Aeronautics and Space Administration

# 1.Introduction

## 1.1. General project overview

Have you ever wondered what standing on a different planet would feel like? This human desire is also the basis for the educational virtual reality application “xSpace”, which allows users to explore different planets and moons in our solar system and uses a gamification approach to educate people about our galaxy and peculiarities of specific planets. The xSpace project was implemented employing the Unity game engine and Oculus headset, allowing for an immersive experience, where users could not only see different planets, but also experience the unique sounds and gravity of each specific planet. As space travel is currently not possible for most people, the future of space exploration for the masses lies in virtual- and augmented reality. XSpace reflects this sentiment and provides a way for users to explore and learn about the solar system in a realistic and engaging manner through the usage of VR. XSpace is designed to allow users to explore different planets from our solar system (in multiplayer mode), complete with corresponding sound effects and gravity conditions, several interactive elements and quiz questions to gauge one’s learning progress. The project aims to create an immersive and educational experience that will enhance the user's understanding and appreciation of the solar system and its planets.

## 1.2. Problems addressed by the project

The problem that the xSpace project aims to address is the lack of opportunities for people to personally experience outer space and being on another planet. Space travel is not feasible for most people, making it challenging for them to gain a realistic understanding of the solar system. Additionally, traditional forms of learning about space, such as textbooks or 2D images, can lack the immersive and interactive elements that are necessary for deep engagement and understanding. Thus, many people may struggle to fully grasp the size, scale, and conditions of different planets and moons in the solar system and miss the chances to have a deeper understanding about it.



The xSpace VR project offers an alternative method of exploring space that can be both realistic, engaging, and educational. It provides an immersive experience that allows users to explore and experience the scale, conditions, and environment of different planets, and enables them to gain a deeper understanding and appreciation of the solar system. This is particularly important in an educational context, where students with a visual learning style can increase their interest in astronomy by interacting with realistic simulations and interactive elements on several planets in multiplayer mode.

### **1.3. Project objectives**

Setting clear project objectives is one of the most critical steps in the project lifecycle. Well-defined measurable project objectives ensure the deadline is met, improve the Key Performance Indicators (KPIs) and the overall project development process. The objectives for xSpace have been categorized into academic, business, system, and personal categories.

#### **Academic objectives:**

1. To develop an educational VR experience that enhances students' understanding of the solar system and its different planets.
2. To use this experience as a case study to explore the potential of VR in education and the challenges that need to be addressed.
3. To provide a learning tool for students with a visual learning style

#### **Business objectives:**

1. To provide an innovative solution that can be used in educational settings to enhance students' understanding of the solar system.
2. To create a unique and engaging experience that can attract users and generate revenue.

3. To use this project as a steppingstone to further explore and develop other VR educational experiences.

**Technical objectives:**

1. To develop a VR application using Unity and the Oculus SDK, that is compatible with the Oculus headset.
2. To incorporate realistic sound effects and gravitational acceleration, which correspond to the real-life conditions on the specific planet in the solar system.
3. To create a user-friendly interface that allows users to easily navigate and interact with the application.
4. To allow multiplayer usage through the employment of the Photon Multiplayer API
5. To integrate voice-over using AI tools

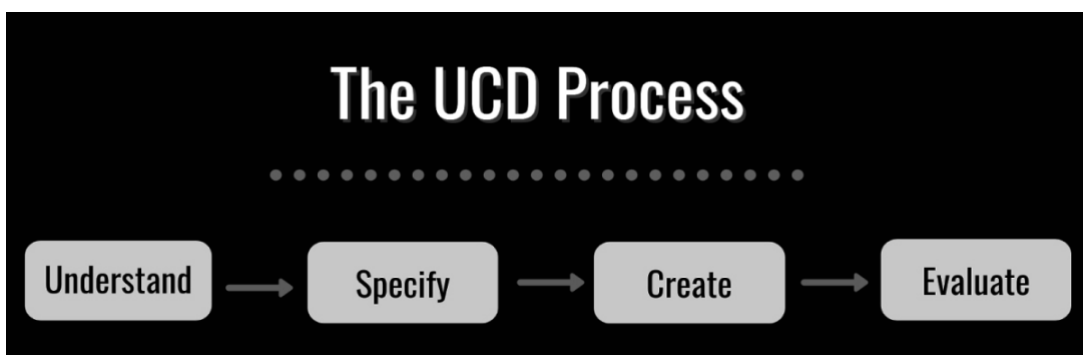
**Persona's objectives:**

1. To offer a fun and interactive experience, that is engaging and easy to understand for all the target users.
2. To individually enhance C# knowledge and 3D modeling experience during the project implementation.
3. To create a stunning VR Project for one's personal portfolio.

## 2. Project conceptualization and implementation

### 2.1. Project conceptualization

xSpace project followed a user centered design approach, which involves placing the needs and wants of the user at the center of the design process.



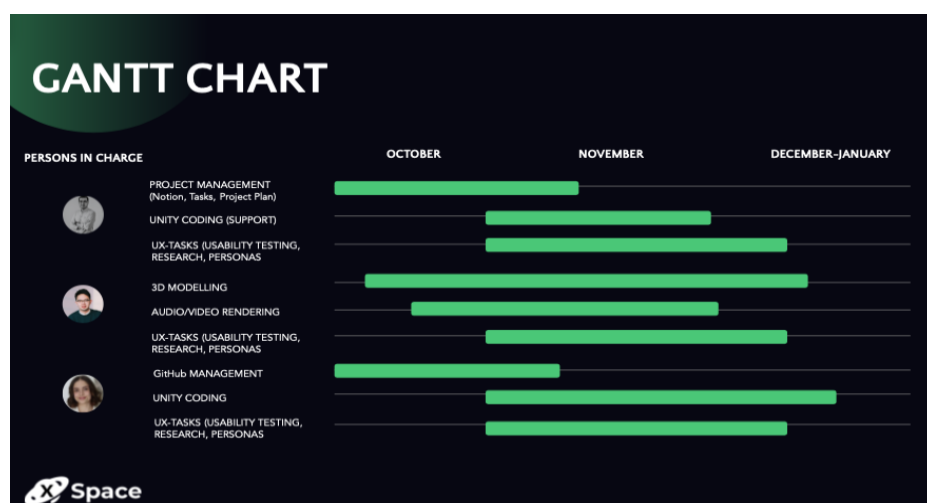
The following examples showcase how we incorporated this approach in each step of the project cycle:

1. Research: We conducted research on the current educational resources available for studying space and planets, as well as user needs and wants for interactive and immersive educational experiences. We also made an investigation with peers to gather feedback on what they would like to see in a VR space exploration experience.
2. Design: We used the information gathered from our research to inform the design of the VR experience. This included the interactive elements, quizzes, and varying gravities that were included in the project. We also worked closely with educators to ensure that the content aligned with their curriculum and learning objectives.
3. Development: During the development phase, we continuously involved users by testing the VR experience at various stages and incorporating

feedback to improve the overall usability. This included conducting user testing sessions with students to gather feedback on the VR experience.

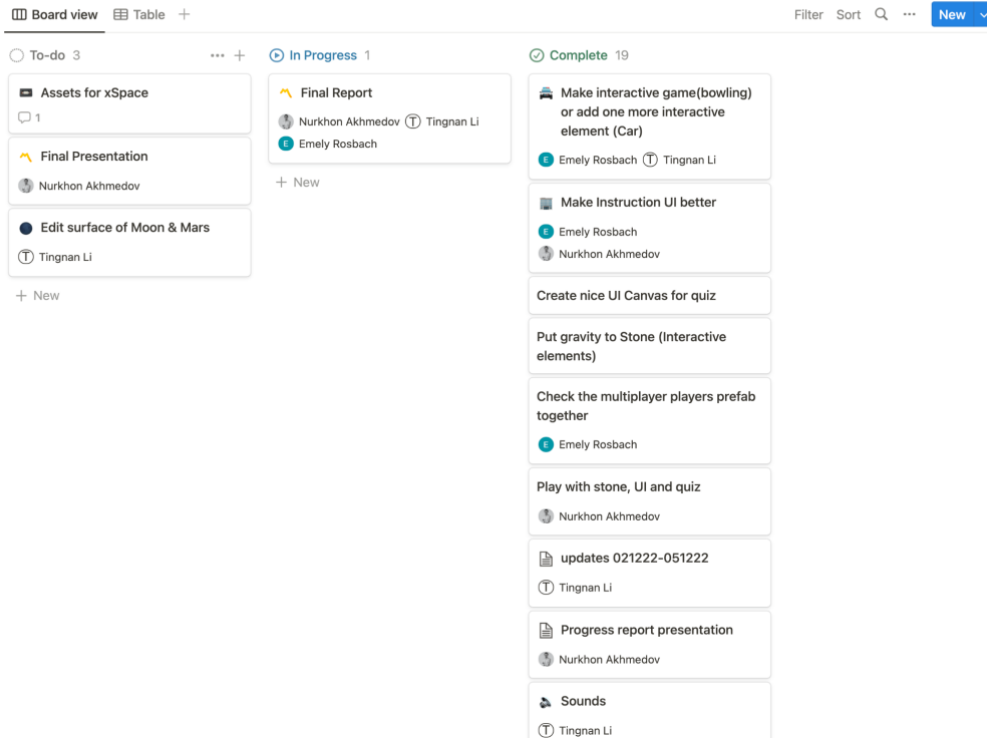
4. Evaluation: We evaluated the effectiveness of our VR experience through usability testing and user interviews. Based on the feedback we received, we made changes and adaptations to improve the overall user experience.
5. Implementation: After making the necessary changes and adaptations, we implemented the final version of the VR experience in an educational setting, where students were able to use it to supplement traditional classroom instruction.

By following a user centered design approach, we were able to create a VR experience that effectively addressed user needs and wants and was well received by students and professors. This approach allowed us to create a more engaging and educational experience that was closely aligned with the needs of the users. Regarding the general project management approach, as a team of individuals with a different personal backgrounds and varying experiences and skills, the first step was the creation of a Gantt Chart to roughly plan out the project timeline and ensure individual competencies are optimally used to advance the project by assigning the best fitting roles to each team member.

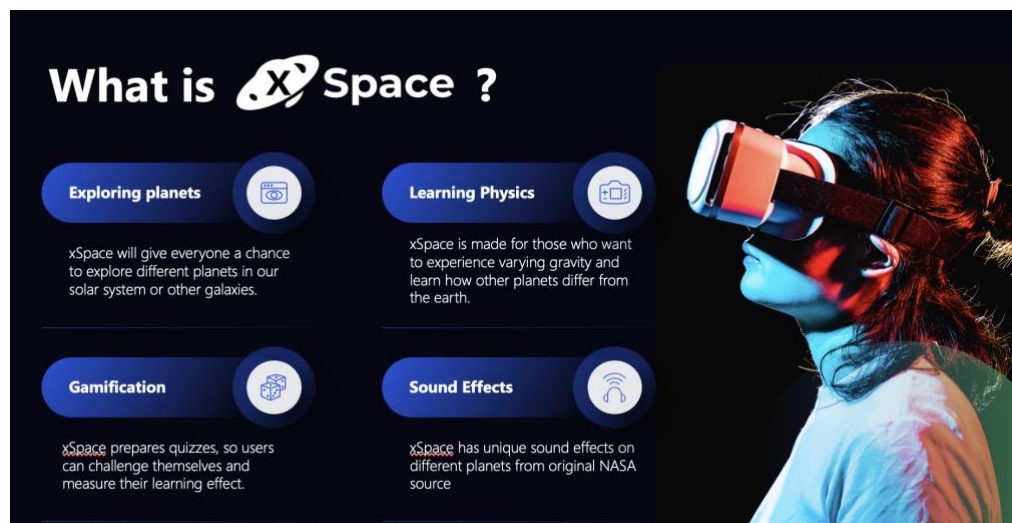


Furthermore, as a team, it was decided to use the Notion app for better collaboration and communication, where the Kanban board is used:

### xSpace VR Project



Regarding the conceptualization of xSpace, firstly it was decided by the team to focus on the four aspects, depicted in the figure below:



Moreover, we have created a storyline for the current project, where the main purpose of the main character, functionalities and stunning moments were detailed as can be seen below:

“



*The story follows the main character, James Mars, who is stranded in space after his spacecraft malfunctions. He is determined to find a way back to Earth, but as he travels through space, he stumbles upon an incredible opportunity to explore and learn about different planets.*

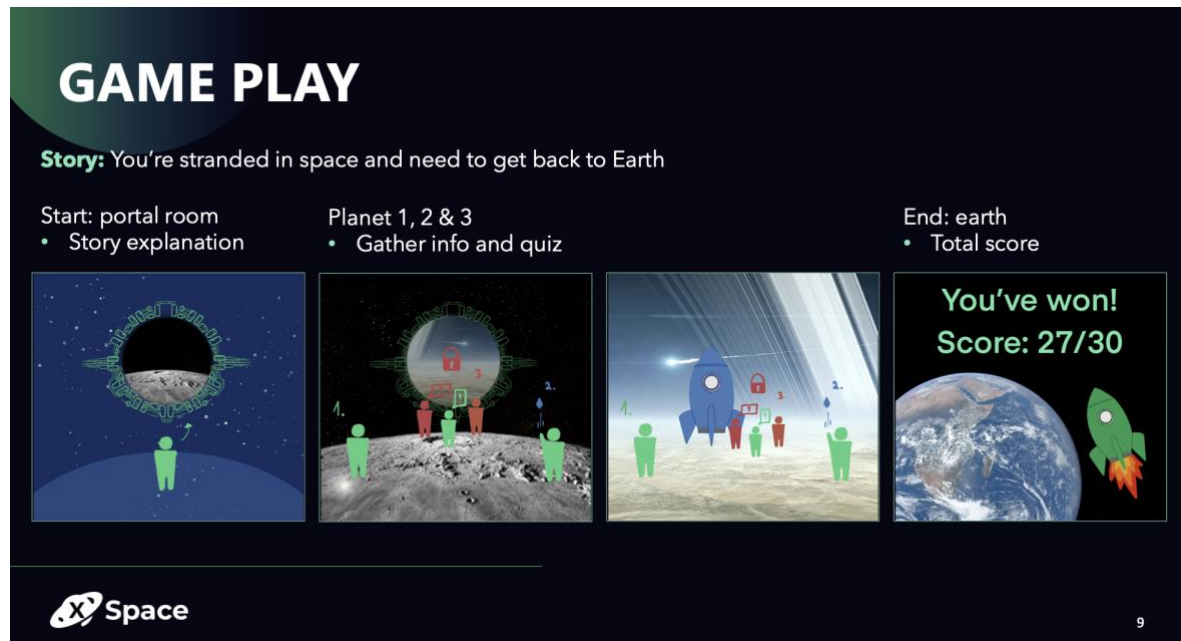
*As James travels from planet to planet, he is greeted by a special guard standing near each portal. The guard asks him different questions to unlock the portal to the next planet. Here he must answer a series of quizzes about the planet's characteristics, including its gravity, atmosphere, and unique interactive elements.*

*James begins his journey by exploring the Moon, where he encounters different gravitational conditions, amazing environmental sounds and discovers that it is the closest celestial body to Earth. He finds interactive unique elements and tries to remember important details to complete the quiz and moves on to the next planet, Mars. There he finds out why Mars has a red surface and experiences yet another gravity. Furthermore, he completes the quiz, and the guard unlocks the Spacecraft to the final destination, Earth. James enters the spacecraft and finds himself arriving back on Earth. He is relieved and grateful for the opportunity to learn and explore these incredible planets. He returns home with a newfound understanding of the universe and a deeper appreciation for the beauty and complexity of our solar system.*

*The end.*

”

The overall gameplay with its storyline is illustrated below:



Additionally, to prioritize the functionalities, that we wanted to implement for the xSpace project, the Moscow Prioritization framework has been utilized to rank tasks based on “Must-have”, “Should-have”, “Could-have”, “Wouldn't have” in the following table:

Functionality	Must-have	Should-have	Could-have	Wouldn't-have
Interactive Elements	+			
Gravity of the objects	+			
Multiplayer		+		
Sound	+			

effects				
Portal (Scene Management )	+			
Quiz	+			
Voice Over			+	
Planet's Modeling and Texture		+		
Scoring System		+		
Spaceman and Spacecraft creation			+	

Generally, considering the aspects mentioned above, we as a team had a clear vision and goals including the project approach, storyline, and prioritization of the tasks.

To summarize the planned user activities are the exploration of different planets, gathering information through interaction with game objects, completing quizzes and using a portal to teleport. Furthermore, the following multi-user interactions were also scheduled: voice chat through Photon, seeing player avatars and their movement and joint interaction with game elements e.g., throwing stones to each other.



## **2.2. User personas**

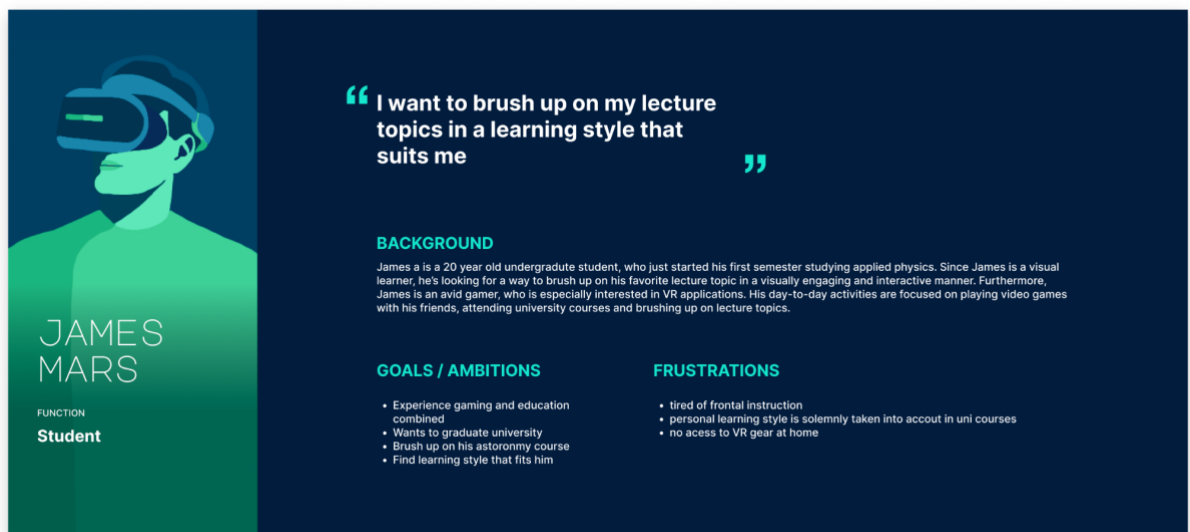
First and foremost, user personas are an essential aspect of any project, and they play a crucial role in ensuring that the final product is designed to meet the needs and wants of the target users. In the case of our xSpace VR project, user personas are important for the following reason:

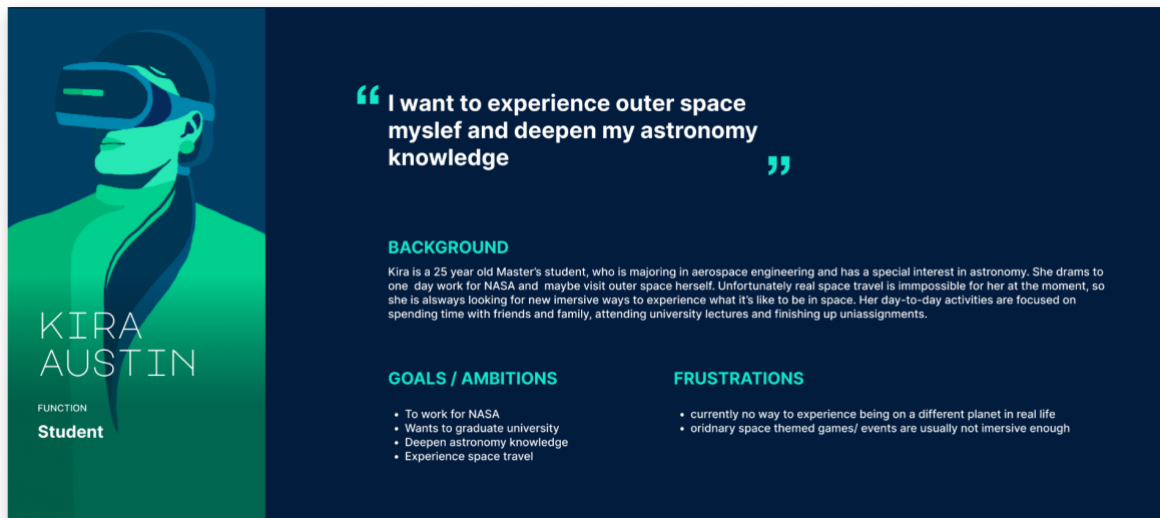
1. Identifying users requires: User personas help to identify the needs and wants of the target users, which is essential for ensuring that the final product meets the user expectations. For our VR project, user personas allowed us to identify the needs of our potential users, which helped to guide the design and development of the VR experience.
2. Creating empathy: User personas help to create empathy with the target users, which allows the team to understand their perspective and experience. This can lead to a better understanding of the user's pain points and challenges and help in identifying the best solutions.
3. Improving communication: User personas can be used as a shared language between the project team members and stakeholders, which can help to improve communication and align the project goals with the target users.
4. Prioritizing features: User personas can be used to prioritize features and functionalities to ensure that the most important and relevant features are included in the final product.
5. Improving usability: User personas can be used to evaluate the usability of the final product and identify any areas that need to be improved. This can help to ensure that the final product is easy to use and understand for the target users.

Based on these important factors, it was decided to use three major User Personas:

- Undergraduate student studying applied physics with a visual learning style
- Postgraduate student studying aerospace engineering with an interest in astronomy
- Professor in applied science, specialized in Virtual Reality experiences

However, after receiving feedback from Prof. Dr. Munir, who mentioned that it is better to focus on students as the target audience, the “Professor” user persona was removed. Thus, it was decided to only use two user personas, which are the students with a visual learning style and interest in astronomy. The results of the created personas are illustrated below:





## 2.3. Methods used to create content (modeling, animation, media, sound)

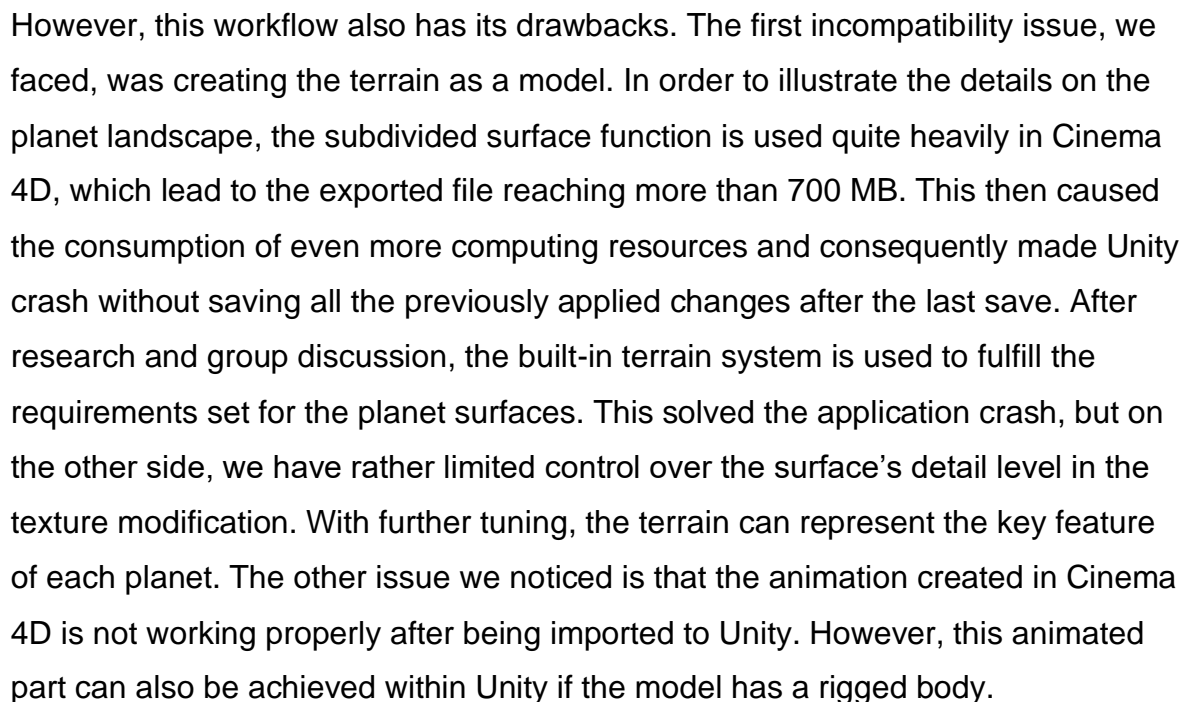
Before creating the actual content, different tools had to be agreed upon by the team. To create the VR experience, we used Unity as our game engine coupled with the Oculus Rift VR headset and controllers. All the content the team creates had to be compatible with the aforementioned soft- and hardware. Therefore, we used Cinema 4D for 3D modeling, and Audacity for editing the sound effects. However, some additional assets have been taken from the official Unity asset store and redesigned/edited to fit the project needs. For instance, the terrain of the moon has been taken from the Unity store [3] and then combined with the Cinema 4D model, which will be described in detail later on.

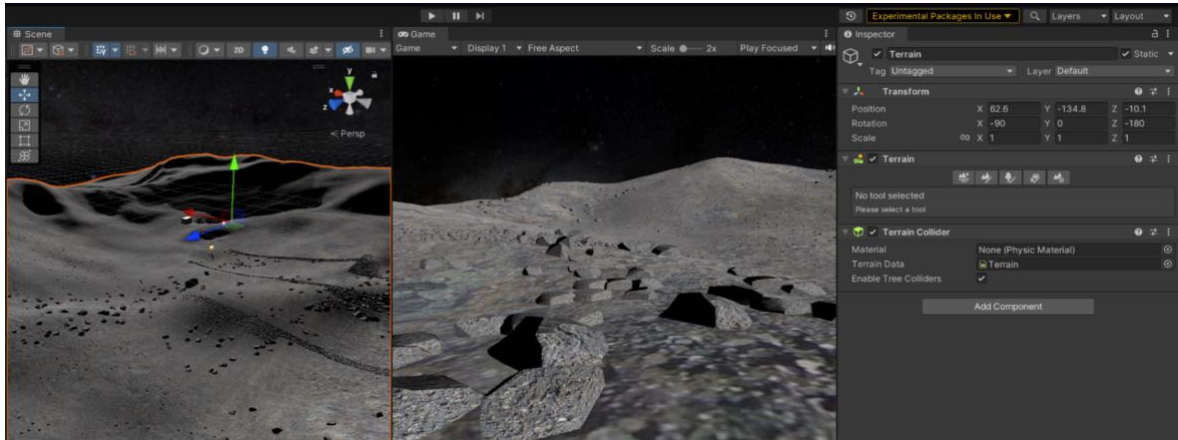
### Cinema 4D

Cinema 4D is a professional 3D modeling, animation, and rendering software solution. Its toolset makes the 3D workflow in the project easier and allows for a more efficient design process, idea generation, and enables modelling especially for AR/MR/VR environments to boost the team's game development in Unity. In

the xSpace project, Cinema 4D was mainly used to create models such as the stone and other interactive element. Moreover, in the early stages of the project some terrains e.g. the Mars and Moon landscapes were also implemented manually in Cinema 4D. The inter format is “.fbx” and the version is Maxon Cinema 4D 2023.1.3 Educational.

In the very early stages of planning, the team already kept the 3D design workflow in mind. Various methods of creating content were tried and tested. The Cinema 4D-.fbx file-Unity was the winning solution for its efficiency and compatibility. The process begins with basic functions and tools, such as modeling, texturing, displacer, subdivided surface, and physic-based-material tool set. Then the model can be exported as. fbx format without the unnecessary elements like lighting and camera settings. Therefore, the data including the material can be transmitted correctly into Unity and the reference is kept the same, which makes it possible to make quick changes and then apply them to Unity. There are various options available for the Cinema 4D export as well as the Unity import. By testing and playing with the relevant options the transit can be made more seamless for specific scenes and requirements. This is also the case when martials must be created in Unity instead of in Cinema 4D, especially when interactive objects are concerned, whose material is depended on the code assigned.

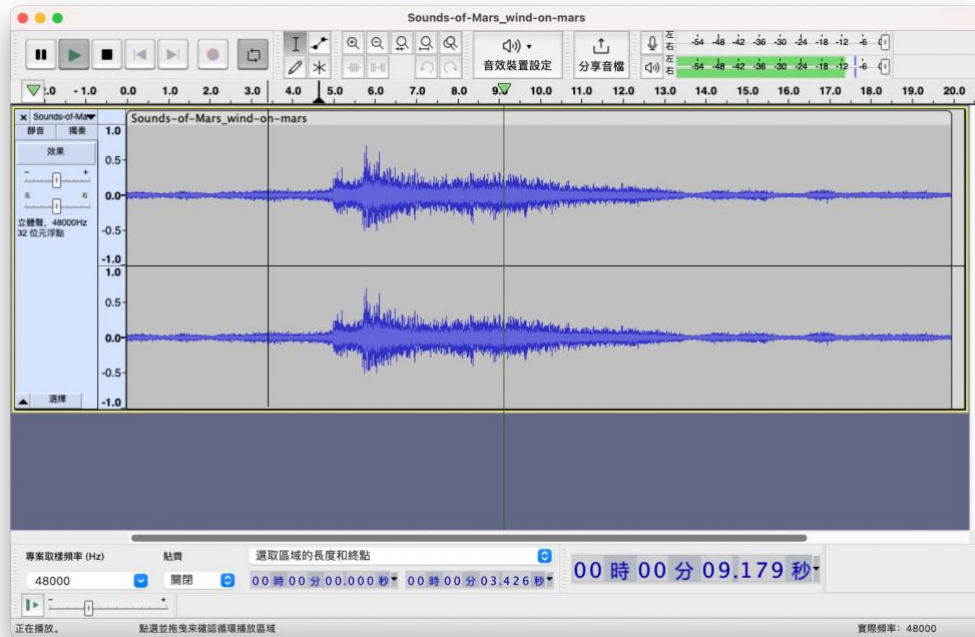




It's important to note that creating assets for Unity VR game development is a complex process, and it requires a lot of knowledge and skills in both Cinema 4D and Unity. And the conversion can additionally result in problems concerning performance, compatibility, and completeness. But these issues can often be fixed by using another format or setup.

### **Audacity**

As an open-source, cross-platform audio software, Audacity 3.2 is an easy-to-use, multi-track audio editor for creating the background sound for each planet to increase the immersiveness of the environment.



Our team began by researching different types of background sounds, that would be appropriate for our VR game. We considered factors such as the game's setting, atmosphere, and overall theme to determine the best type of background sound to use.

With the sources of sound from Mars provided by NASA, the team had an inspiration and starting point to create background sounds in space. As commonly known, vacuum is not a medium that transmits sound waves, but the astronaut would most likely have a supporting system in the suit, which gives the concept a reason. In Audacity, the raw sounds were filtered and boosted with the built-in functions, which made the sounds less harsh and irritating. This also included using the software's built-in effects such as equalization and reverb to create a sense of depth and space in the sound. We also used Audacity's time-stretching and pitch-shifting tools to adjust the tempo and pitch of the samples as needed. The sounds were exported in stereo setup in 320 KBPS bit rate to maintain the quality and suitability for VR scenarios. We then imported the audio files into Unity and used them as background sound for our VR game.

## 2.4. Technical integration

### 2.4.1 Technical requirements

Before the actual implementation was set in motion, technical requirements in accordance with the Moscow Prioritization Framework (explained in more detail in the “Project conceptualization” chapter) were established by the team.

Functionality	Must-have	Should-have	Could-have	Wouldn't-have
Runnable product	+			
Multiplayer functionality (Photon)		+		
Interactive elements	+			
Quiz and score system	+			
Teleportation/portal logic	+			
More than one planet for		+		



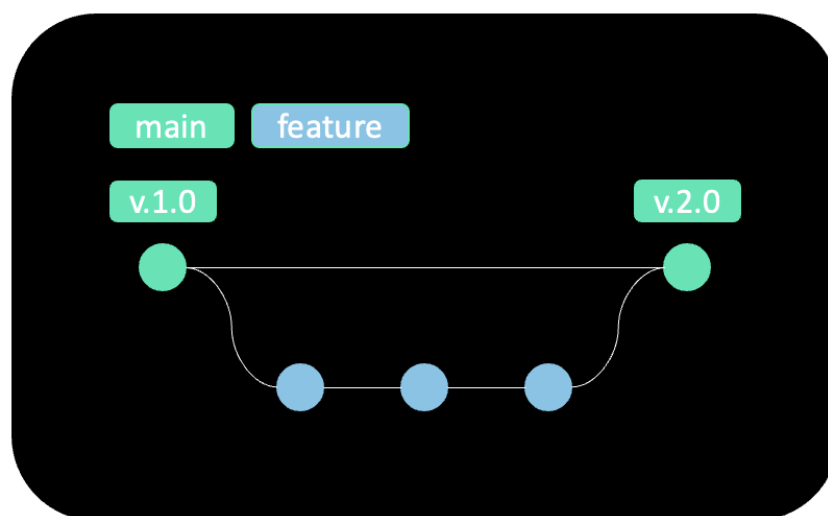
exploration (excluding end planet)				
--	--	--	--	--

## 2.4.2 Development tools and environments

### GitHub

To make the coding workflow and collaboration as smooth as possible, version control is often utilized to manage and document source code changes.

For the xSpace project a new GitHub repository was initialized, where the master branch always contains the latest, error free version of the application, to help keep track of source code changes. As direct pushes to the master branch were purposefully prevented, a feature branch workflow, which is depicted in the figure below, was used to ensure that only high-quality, runnable code was committed to the master branch. This source code branching pattern dictates that each new feature must be implemented on a new separate branch. Furthermore, only successful alterations are allowed to be merged back onto the master branch [2].



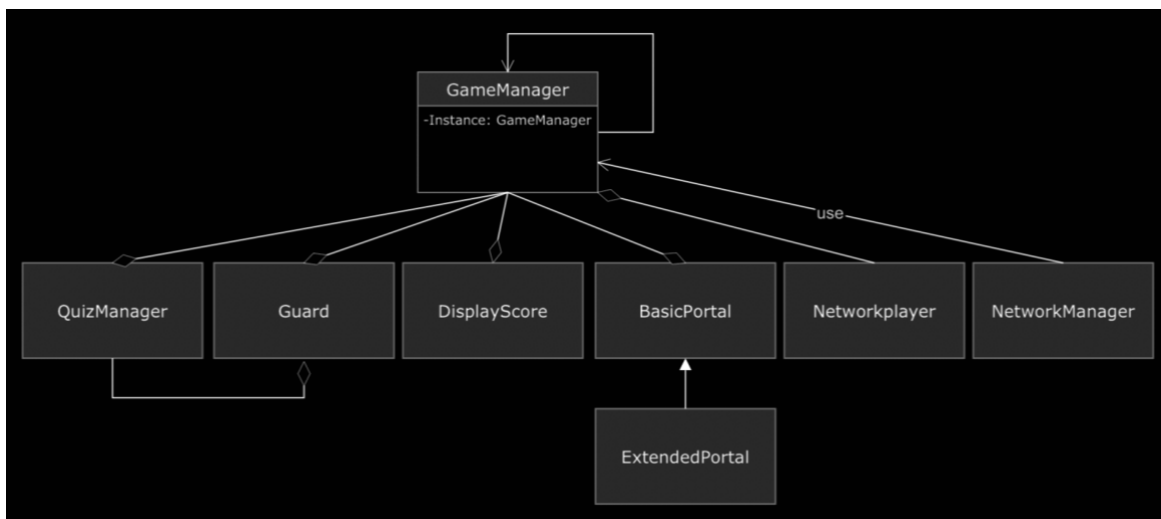
## Unity

The cross-platform game engine Unity (version 2021.3.9f1), which is considered easy to use for beginner developers and is popular amongst the indie game development community, was employed to create the xSpace VR application. The xSpace Unity project was set up with a VR core, which allows one to directly target virtual reality devices from Unity without needing to import any external plugins, and additionally employs the Vive input utility toolkit to ensure a seamless functionality of the Oculus VR glasses and controllers. To enable multiplayer gaming, the Photon PUN, and Photon Voice Unity packages, each assigned with a specific identifier, were also included in the project.

### 2.4.3 Code implementation

#### Architecture Overview

Before singular functionalities of the VR application will be elaborated upon, its overall architecture shall be analyzed shortly based on the figure depicted below, which portrays the major C# classes and their interaction partners.



The “GameManager.cs” class is the heart of the application, responsible for initiating or controlling different processes and keeping track of important

game data such as the current level. The singleton creation design pattern is utilized to restrict instantiation of the “GameManager” class to only one object [1]. Since this single “GameManager” object coordinates major actions across the application, the employment of the Singleton pattern is crucial to ensure that other C# classes access and modify the same data, stored inside GM instance variables.

```
void Awake()
{
    if (Instance == null)
    {
        Instance = this;
        DontDestroyOnLoad(gameObject);
    }
    else
    {
        Destroy(gameObject);
    }
}
```

### Photon Connection

The “NetworkManager.cs” class, which was provided in the Multi-User VR example project, was slightly modified and is now solely responsible for establishing a connection to the Photon server and initiating the player character instantiation by calling the GM’s “InstantiatePlayer()” method as soon as the user successfully joins the specified room.

```
void Start()
{
    PhotonNetwork.GameVersion = GAME_VERSION;
    PhotonNetwork.ConnectUsingSettings();
}

public override void OnConnectedToMaster()
{
    PhotonNetwork.JoinOrCreateRoom("Portal-Room", ROOM_OPTIONS, null);
}

public override void OnJoinedRoom()
{
    GameManager.Instance.InstantiatePlayer();
}
```

### Instantiate & Move Player

The actual player instantiation happens inside the “GameManager.cs” class. Here, a new NetworkPlayer object is created, whose data and attributes such as the current position are tracked by Photon.

```
public void InstantiatePlayer()
{
    networkPlayer = PhotonNetwork.Instantiate(this.networkPlayer.name,
new Vector3(0, 3f, 0), Quaternion.identity, 0);
    np = networkPlayer.GetComponent<NetworkPlayer>();
    networkPlayer.transform.parent = transform;
    cameraRig.transform.parent = networkPlayer.transform;
}
```

The “NetworkPlayer.cs” script is attached to an avatar prefab in Unity, which is used to visualize the user in the Unity scene. Moving the avatar works in one of the ways:

Option1: The position and rotation of the VR glasses, stored inside the VROrigin Unity game object, are mapped to the avatar. Meaning, whenever the user moves their body or head in real life, the character in Unity follows suit. This is implemented inside the “NetworkPlayer.cs” class, where, after verifying that the player prefab belongs to the current user (with “if (photonView.IsMine)” , the player prefab position is modified in the “Update()” method.

```
void Update()
{
    if (photonView.IsMine)
    {
        CameraOffset.y = -2;
        MapPosition(Head, headRig, CameraOffset);
    }
}
```

```
void MapPosition(Transform target, Transform rigTransform, Vector3 cameraoffset)
{
    target.position = rigTransform.position + cameraoffset;
    target.position = new Vector3(target.position.x, GetComponent<Rigidbody>()
.position.y, target.position.z);

    target.rotation = rigTransform.rotation;
    target.rotation = Quaternion.Euler(0.0f, target.eulerAngles.y, 0.0f);
}
```

Option 2: Additionally, the player prefab can be moved via the joystick on the right VR controller. This is accomplished in the “MovePlayerController.cs” script attached to an empty Unity game object, where depending on the joystick input the VROrigin and therefore the player prefab position is transformed inside the “Update ()” method.

```

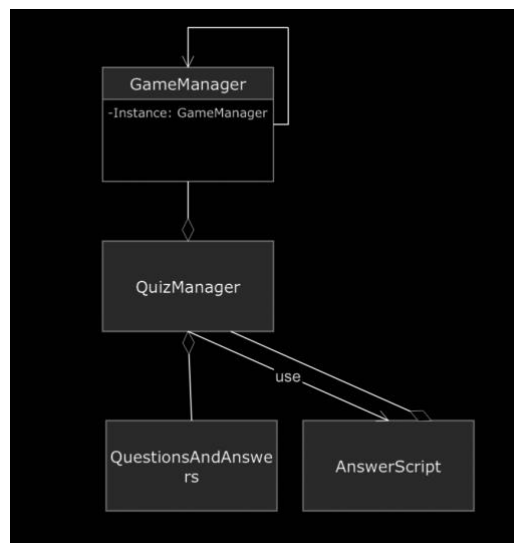
void Update()
{
    if (!_targetDevice.isValid)
    {
        TryInitialize();
    }
    else
    {
        float speed = movementSpeed;

        if (_targetDevice.TryGetFeatureValue(CommonUsages.primary2DAxisClick,
out bool buttonPressed) && buttonPressed == true)
        {
            speed = movementSpeed * 5;
        }
    }
    ...
}
}

```

## Quiz

As mentioned in the customer journey chapter, the player must complete a brief quiz to unlock the portal and enter a new planet/level. This Quiz is created and controlled by the “QuizManager.cs” script attached to the empty “QuizManager” game object.



Here the quiz questions and answers are stored inside a List of the type “QuestionsAndAnswers”, which then is used to generate and set the answers options/ questions, visible to the user, in the form of UI elements.

```
private void generateQuestion()
{
    if (QnA.Count > 0)
    {
        currentQuestion = UnityEngine.Random.Range(0, QnA.Count);
        QuestionTxt.text = QnA[currentQuestion].Question;
        setAnswers();
    }
    else
    {
        Debug.Log("out of questions");
        GameOver();
    }
}
```

```
private void setAnswers()
{
    for (int i = 0; i < options.Length; i++)
    {
        options[i].GetComponent<AnswerScript>().setNeutral();
        options[i].GetComponent<AnswerScript>().isCorrect = false;
        options[i].transform.GetChild(0).GetComponent<Text>().text =
QnA[currentQuestion].Answers[i];

        if (QnA[currentQuestion].CorrectAnswer == i + 1)
        {
            options[i].GetComponent<AnswerScript>().isCorrect = true;
        }
    }
}
```

To ensure the correct questions and answer options are loaded in each level, a switch-case function is used inside the “setUp ()” method, which loads the right strings depending on the current level. Furthermore, the score of the current quiz round is stored inside an instance variable and passed on to the GM, when the player changes planets and is then added to the total player score.

```
List<QuestionsAndAnswers> quiz1 = new List<QuestionsAndAnswers>();

switch (gameManager.getCurrentLevel())
{
    case 0:
        question1 = "What is the gravity of the moon?";
        answers1 = new string[] { "1.625 m/s2", "9.255 m/s2", "3.547 m/s2",
"7.881 m/s2" };
        correct1 = 1;
        questionAndAnswers1 = new QuestionsAndAnswers(question1, answers1,
correct1);
        ...
}
```

## Portal

The portal on each planet is locked by default and must be unlocked by answering at least 60% of the quiz questions correctly. This is achieved by changing the GM’s instance variable “isCurrentPortalActive” from false to true and checking for

changes inside the “Update ()” method of the portal super- and subclasses. Here, inheritance is used to pass the attributes of the “BasicPortal.cs” superclass to the “ExtendendPortal.cs” subclass, but to additionally expand the portal logic by adding a material change in the subclass to showcase whether the portal is currently locked.

```
public class ExtendedPortal : BasicPortal
{
    public GameObject portalLiquid;
    public Material open;
    public Material closed;

    void Update()
    {
        if (gameManager.isCurrentPortalActive())
        {
            portalLiquid.GetComponent<Renderer>().material = open;
        }
        else
        {
            portalLiquid.GetComponent<Renderer>().material = closed;
        }
    }
}
```

## Change Levels

To initiate the level change, after the portal was successfully unlocked, the GM’s “nextLevel()” method is called, when a collision with the portal is detected. For this a Box Collider was added to the portal Unity game object.

```
public void OnCollisionEnter(Collision other)
{
    if (gameManager.isCurrentPortalActive())
    {
        Debug.Log("next level");
        gameManager.nextLevel();
    }
}
```

When the GameManager “nextLevel()” method gets called, the current level is increased by one and game settings such as the boolean variable “isCurrentPortalActive” are reverted to their default states. Moreover, three dictionaries are employed to output the correct spawning position for the character, environment sound and terrain depending on the current level.

```
public void nextLevel()
{
    if (level < 2)
    {
        level++;
        levels[level].SetActive(true);
        setPlayerScore(currentScore);
        currentScore = 0;
        currentPortalActive = false;
        XRrigPosition.transform.position = spawns[level].position;
        audio.Stop();
        audio.clip = sounds[level];
        audio.loop = true;
        audio.Play();
        levels[level - 1].SetActive(false);
    }
}
```

## 2.5. Usability testing and results (plan, conduction, result analysis)

Usability testing is an important aspect of any project because it helps to evaluate the effectiveness of the user interface and the overall usability of the product. By conducting usability testing, we can gather feedback from real users and identify any issues or areas that need to be improved.

1. Identifying usability issues: Usability testing allows us to identify any usability issues that users may experience while using the product. By identifying these issues early on, we can make changes to the product before it is released, which will result in a better user experience.
2. Measuring user satisfaction: Usability testing allows us to measure the level of user satisfaction with the product. By conducting surveys and interviews with users, we can gather feedback on the overall usability and engagement of the product. This feedback can be used to make improvements to the product and increase user satisfaction.
3. Improving product usability: Usability testing helps us to identify areas that need to be improved in terms of usability. By making changes to the product based on user feedback, we can improve the overall usability and make the product more user-friendly.



4. Prioritizing features: Usability testing helps us to prioritize features and functionalities to ensure that the most important and relevant features are included in the final product.

The planned usability test consisted of a single player Lab Test and a follow-up survey using Google Forms, which both followed the following requirements:

- Team roles: 1) moderator, responsible for guiding the participant through the xSpace application 2) observer, in charge of protocoling the user answers 3) Survey master, responsible for the follow-up survey
- Number of participants: 5
- Target group: THI students with interest in space/ visual learners

### **Lab usability test**

A lab usability test is often conducted on an especially pre-configured device inside a lab. Where test subjects' complete tasks while the moderator observes and asks questions [4]. This is exactly what we did: we set up a PC inside the VR-lab with the current version of our application running and after a short introduction, asked our 5 participants to complete 4 tasks each inside the VR game. We encouraged them to think aloud and noted down their remarks.

Tasks:

- Looking and moving around inside the VR scene
- Using controllers to interact with game objects e.g., stone
- Completing the quiz
- Changing planets via the portal

The goal was to assess interactive elements and the functionality of the game logic, while receiving live and unfiltered feedback. The lists below show up a summary of the lab usability test results, the whole experiment transcript containing the individual feedback form the participants, can be found in the appendix to this paper.

**Positive feedback:**

- Immersive experience, sound- and world design
- Well working interactions and game logic
- Selected quiz questions were appropriate and not too difficult
- Great showcase of differing gravity

**Negative feedback:**

- Need for a tutorial or contextual help at game start
- Blurry UI elements

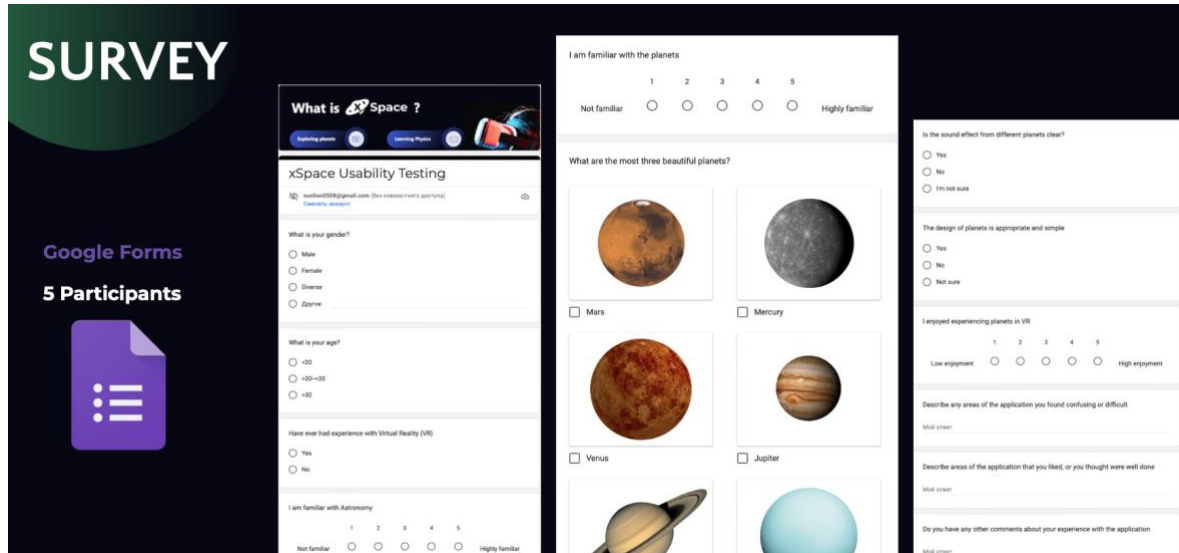
**Survey**

After successfully finishing the lab usability test participants were asked to complete a Google Forms survey to record the following data:

- Demographic data (gender, age etc.)
- Previous experience in VR
- Fundamental knowledge of astronomy
- Likert scale of interest in planets (from 1 to 5)
- Feedback of the overall user experience (including sound, visuals)
- Confusing areas
- Mostly liked features
- Additional comments

The questions can be viewed in detail by following the provided link:

<https://docs.google.com/forms/d/16uh7ekQtw0kkBAwjKmfcVeJ15dhtb9zhsWiRcfzJgDk/edit>



The lists below offer a short summary of the survey findings, detailed result views can be found in the appendix to this paper.

### Overall suggestions for improvements:

- Guidance at the beginning
- Blurry quiz

### Overall good points:

- Sounds
- Movement
- Gravity
- Portal

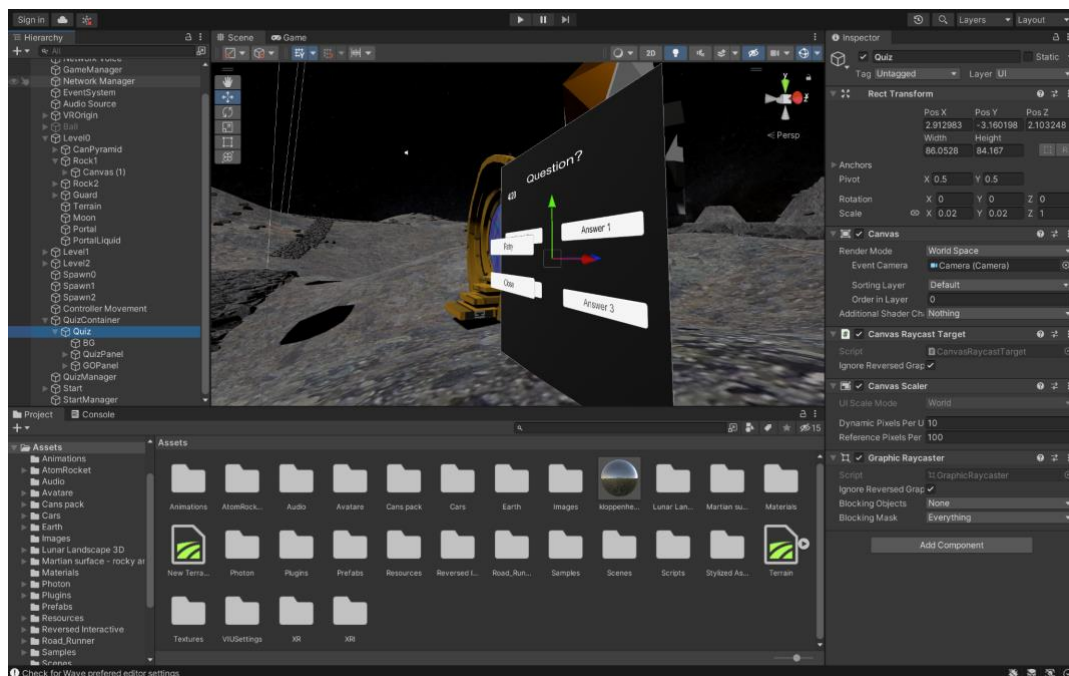
After analyzing and summarizing all the feedback, the next steps for us were working on the user centered improvements, which will be the focus of the next chapter.

## 2.6. Improvements (Guidance)

Based on the results of usability testing provided above, it was decided to immediately start making xSpace even better. The list of issues with improvements are provided below:

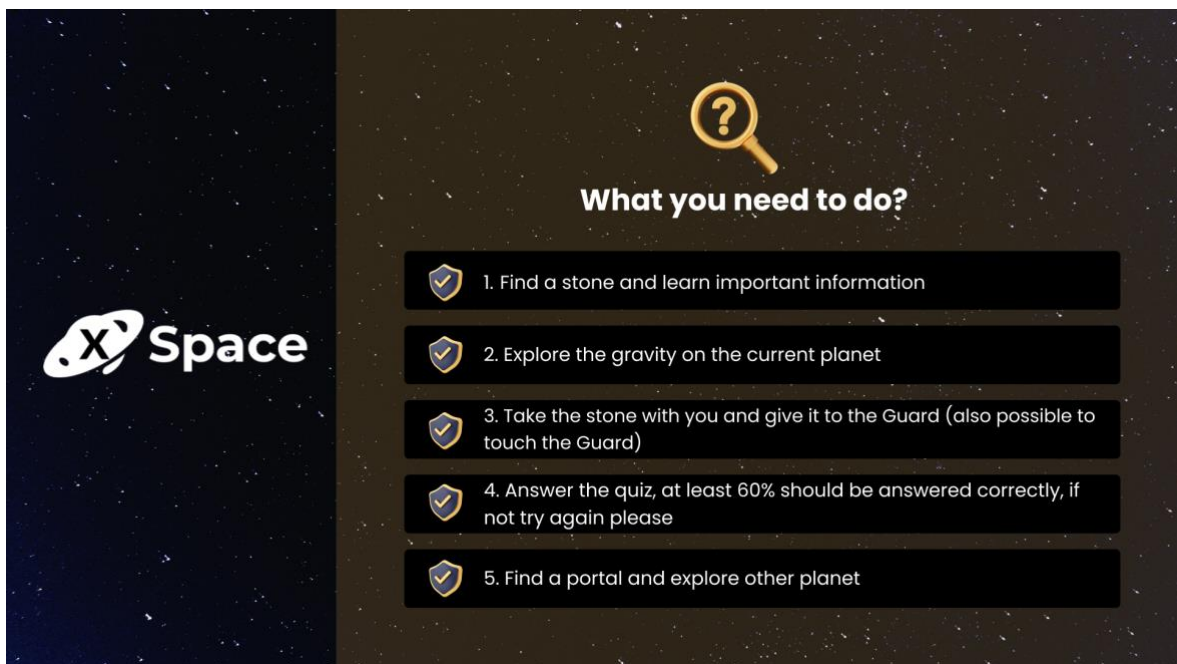
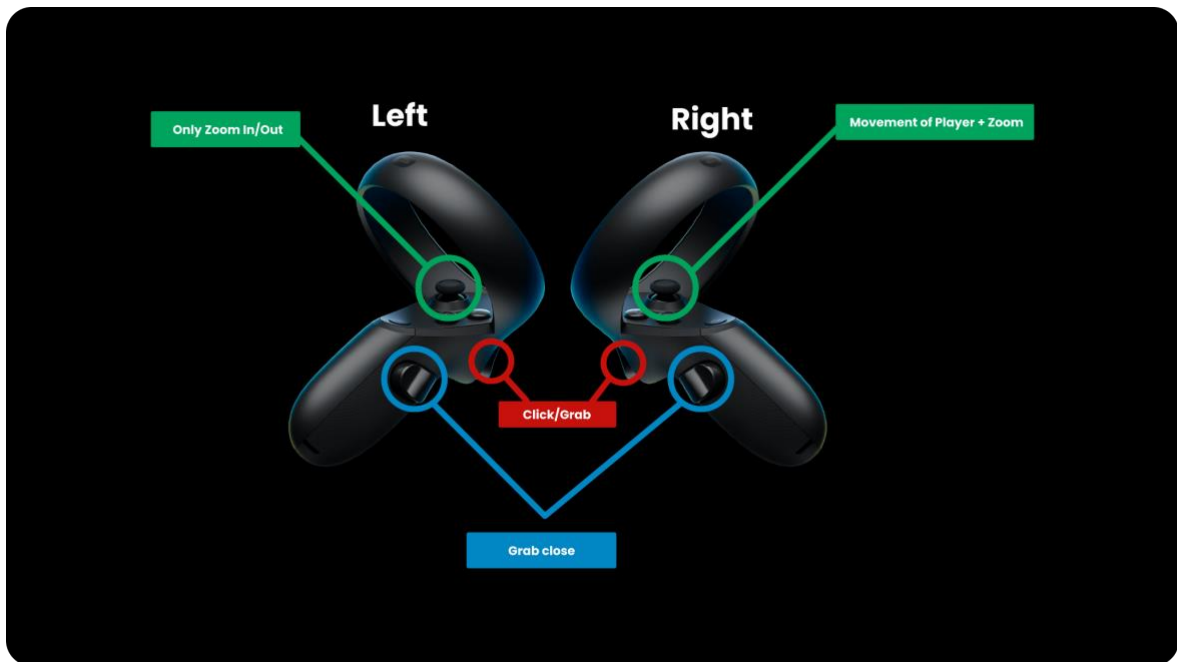
### 1. Blurry User Interface

The problem with the UI canvas was, that the scaling parameters were wrong. Thus the additional component called “Canvas Scaler” was added with proper settings which are illustrated below:



### 2. Guidance in the beginning of a project

As it was mentioned above in this report, the player can move and interact with the objects by grabbing, throwing, and clicking them using the controller. However, during the usability testing, participants mentioned, that it would be great, if there was an instruction on how to use the controllers. Also, the main mission of the project should be illustrated as an instruction. Thus, the instructions have been created using the Adobe Illustrator as a vector image and added as a tutorial inside the game.



### 3. UI feedbacks on button

To fix this UI issue and provide feedback, when a quiz answer is clicked, additional scripts have been implemented including the change of the button material depending on whether the quiz question is answered correctly.

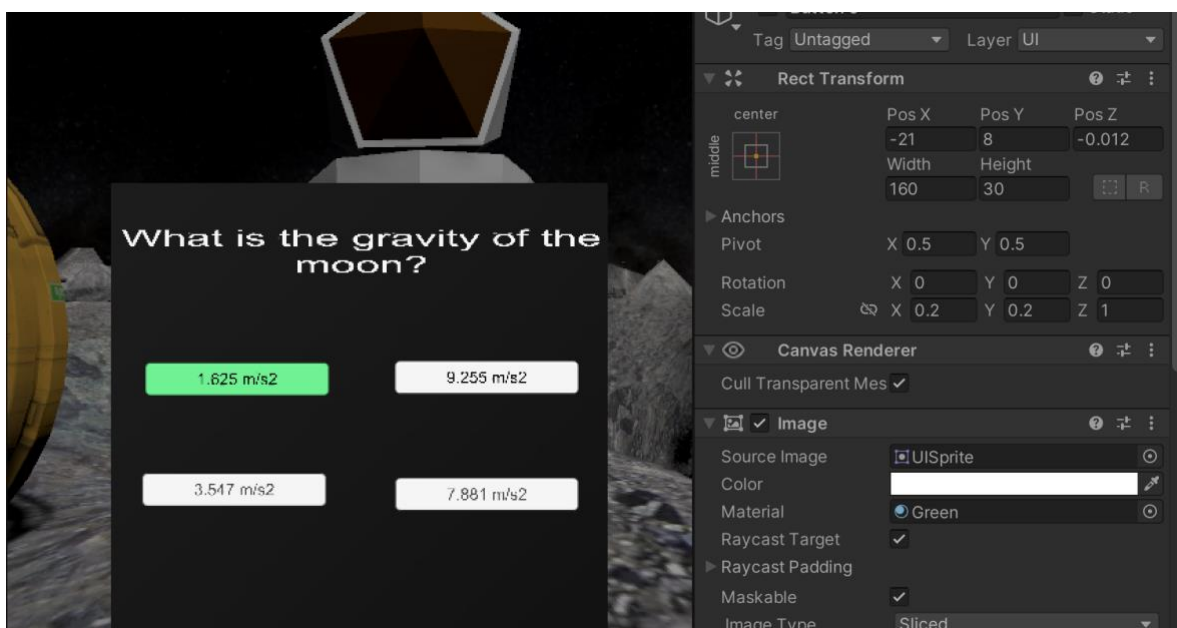
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class AnswerScript : MonoBehaviour
{
    public bool isCorrect = false;
    public QuizManager quizManager;
    public Material red;
    public Material green;
    public Material neutral;

    public void Answer()
    {
        StartCoroutine(wait());
    }

    public void setNeutral()
    {
        Image w = GetComponent<Image>();
        w.material = neutral;
        Debug.Log("neutral");
    }

    IEnumerator wait()
    {
        Debug.Log("wait");
        if (isCorrect)
        {
            Image g = GetComponent<Image>();
            g.material = green;
            Debug.Log("Correct");
            yield return new WaitForSeconds(1);
            quizManager.correct();
        }
        else
        {
            Image r = GetComponent<Image>();
            r.material = red;
            Debug.Log("Wrong");
            yield return new WaitForSeconds(0.25f);
            quizManager.wrong();
        }
    }
}
```



## 2.7. Limitations (technical, gamification)

The xSpace project was completed fully as it was planned, however, there were some limitations and lessons learned while developing and implementing it in Unity. The provided table indicates all the challenges we faced and how we tried to solve them:

Challenge/Limitation	Description	How we solve/solved it
Multiplayer	The project was initially created without Photon multiplayer; thus it was difficult to set up correctly later on.	We have set up Photon using our own PUN and created “spawning points” to place the location of users depending on their current level.
Scene Management (in Multiplayer, Offline)	The Scene Management has been built. However, firstly did not work as intended while having multiple users. For example, when the first player answers the quiz and moves on to the next planet, the second player (still not done exploring) is also teleported there.	We have solved this issue by building all planets in the same scene, however, the position of them differs, thus when the first user is traveling to the next planet, the second player stays on their current planet which is better in terms of Usability Experience. In the end, every player will have their own Score.

Voice over	The guard should have voice-over generated by AI and whenever the user triggers the guard, the voice-over turns on and explains how to answer questions	For the current sprint, the voice-over has not been implemented yet, which is considered as a limitation for the xSpace project. However, if we could manage to allocate time for it, we can do it before publishing it to our portfolio.
Simultaneously working on one project	"Prior to the implementation of a repository and the establishment of proper branching protocols, the process of collaborating on the same project was problematic due to the lack of consistency in versions, code, and modifications	One of us (Emely) managed to set up a GitHub repository with several branches, where we can push/pull changes to our project, which makes our development cycle much easier.

In conclusion it can be said that the project was implemented successfully and offers the necessary structure to add more planets from our solar system. However, if the integration of even more celestial bodies or even galaxies is intended for the future, then the one scene/ one Photon room structure of the project, has to be expanded accordingly.



## 2.8. Reflection on objectives

In the following part of this report, the reflection on the objectives (as mentioned in project objectives section) and whether they were achieved will be provided:

Status	Academic	Business	Technical	Personal
	Achieved all	Achieved all	One was not achieved → voice-over implementation using artificial intelligence. However, it will be implemented after the final report is submitted.	Mostly achieved. The project should be stacked in a more visual way and published on Behance, Dribble or other platforms as a portfolio.

### **3. Summary and Self-Evaluation**

In conclusion, our VR project in Unity was able to create an interactive and educational experience for users to explore different planets, complete quizzes, and experience different gravities using an Oculus headset. The feedback from the usability tests was generally positive, with users reporting that they enjoyed the interactive elements and quizzes and felt that the VR experience enhanced their understanding of the planets and astronomy. The project was deemed to be practical and could be used in educational settings to supplement traditional classroom instruction, however, the cost of the Oculus headset may be a limitation. Future work could include expanding the number of planets included and expanding the quizzes to cover a wider range of topics, voice-over implementation and finalization of all our additional nice-to-have features. Then we could publish it to the Unity Store as a free asset for educational use.

## 4. List of references

- [1] M. Fowler, FeatureBranch, 2020. [online]. Available at: <https://martinfowler.com/bliki/FeatureBranch.html> (Retrieved: 13.01.2023).
- [2] Technopedia, Singleton, 2011. [online]. Available at: <https://www.techopedia.com/definition/15830/singleton> (Retrieved: 12.01.2023).
- [3] Unity, Unity Store, 2023. [online]. Available at: [https://assetstore.unity.com/publishers/19123?gclid=CjwKCAiAwomeBhBWEiwAM43YIF5Eqlz8pmuFJLYBWDSZLMb5SQP1WYNR56Il76Ilb8Te6FUZv7kqdh oCNu0QAvD\\_BwE](https://assetstore.unity.com/publishers/19123?gclid=CjwKCAiAwomeBhBWEiwAM43YIF5Eqlz8pmuFJLYBWDSZLMb5SQP1WYNR56Il76Ilb8Te6FUZv7kqdh oCNu0QAvD_BwE) (Retrieved: 12.01.2023).
- [4] Hotjar, The different types of usability testing methods for your projects, 2022. [online]. Available at: <https://www.hotjar.com/usability-testing/methods/> (Retrieved: 13.01.2023).

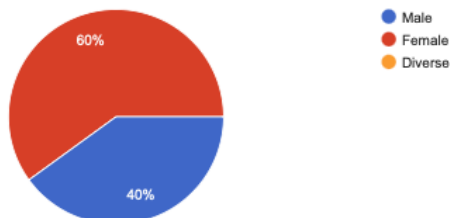
## 5. Appendices

Lab usability test - Tasks	P1	P2	P3	P4	P5
1. Look and move around inside the VR scene	visually pleasing surface design	movement works well, but is a bit dizzying when moving with the joy stick	amazed by environment, moving around works well , nice world, space sound fitting	Planet texture looks good, natural movement, sound is immersive, no rotation with joystick (only with oculus)	sound is very immersive, great planet and props design
2. Use controllers to interact with game objects e.g. stone	Stone UI is a bit blurry	Stone UI is hard to read (especially in the distancy)	UI is a bit pixely, but readable. Likes the gravity/floating effect	Interactions work well	Intuitive interactions, good functionality
3. Complete the quiz	UI is blurry, but the quiz works well	Questions a bit too easy, no hover effect when button pressed/selected	appropriate quiz questions with cool fact and humorous answers	good	UI a bit blurry (from glasses maybe)
4. Change plantes via the portal	great teleportation functionality	natural feeling during scene change	Cool scene change, but second UI is placed too high	Portal looks nice	scene change done well
Additional feedback	more instruction before/ during game	feedback for UI e.g. sound/ highlight	Scoreboard too high	More hints/ screen inputs. Highlight grabbed elements	Tutorial when starting

What is your gender?

5 ответов

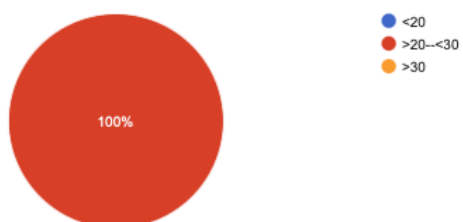
 Копировать



What is your age?

5 ответов

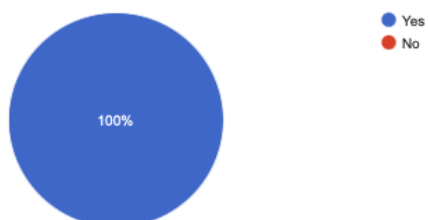
 Копировать



Have ever had experience with Virtual Reality (VR)

5 ответов

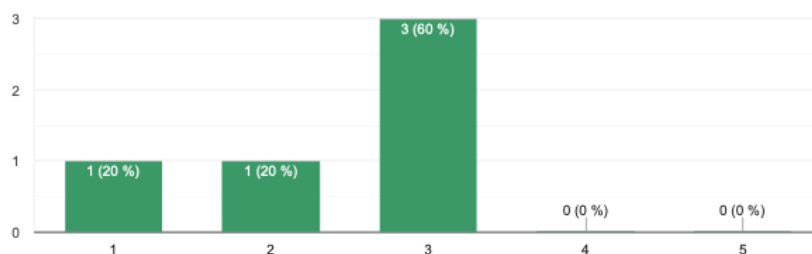
 Копировать



I am familiar with Astronomy

5 ответов

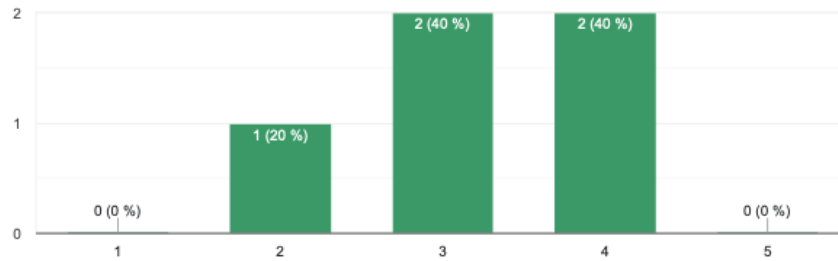
 Копировать



I am familiar with the planets

 Копировать

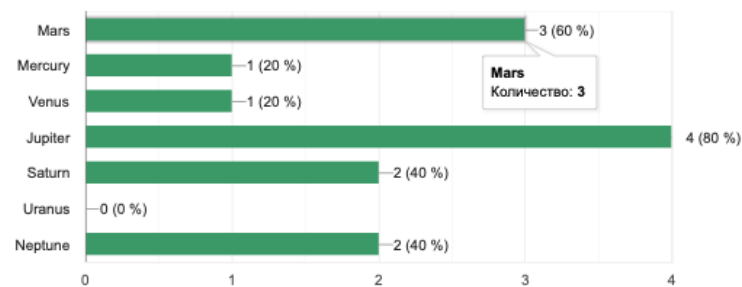
5 ответов



What are the most three beautiful planets?

 Копировать

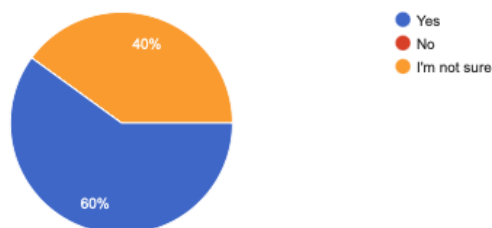
5 ответов



Is the sound effect from different planets clear?

 Копировать

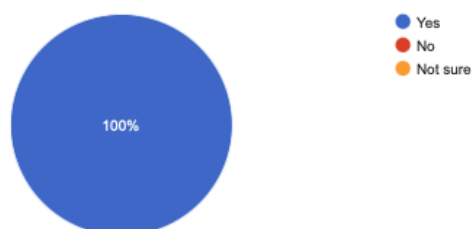
5 ответов



The design of planets is appropriate and simple

 Копировать

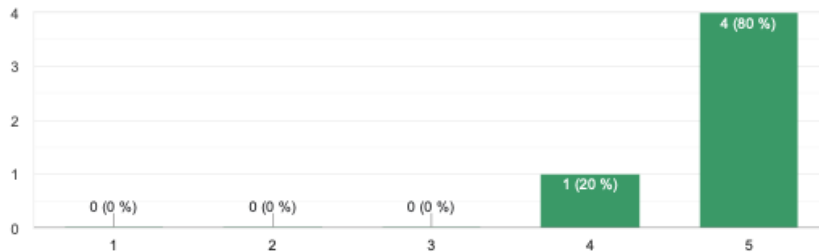
5 ответов



I enjoyed experiencing planets in VR

Копировать

5 ответов



Describe any areas of the application you found confusing or difficult

5 ответов

A bit more guidance at the beginning would be cool e.g. where to go, etc as well as an introduction in the gestures in the game. Further, for people who do not have preknowledge information would be good before the quiz so that you can apply your learning in the quiz.

didn't know what to do with the port

walking through the ground

Instruction UI Canvas was a bit blurry, maybe the size of the text can be increased

Grabbing a stone and navigate at the same

Describe areas of the application that you liked, or you thought were well done

5 ответов

I loved the sound and the environment of the planets which gives you the feeling that you're actually on the planet. As well as the movement by joystick so you can move around as you like without the restrictions of the room.

liked the quiz, learned some facts about the moon which I didn't know

overall ui, sounds, tapping into the astronaut, going through the portal

Movement, Sound, Planets and Gravity

Minimalist visuals

Do you have any other comments about your experience with the application

5 ответов

Really like the idea of experiencing different planets. I think that's a topic where several students can learn a lot!

really liked the style and that I could move around

Good Job!

Great project!

Maybe add some fun stuff like jump, some short tutorials in game like what I can do say can I throw a stone or take it to another portal