|  |  |
| --- | --- |
| **Project Case** |  |
| COMP6583 | COMP6583001  Computer Graphics |
| **Computer Science** | **O252-COMP6583-MF01-00** |
| ***Valid on Odd Semester Year 2024/2025*** | **Revision 00** |

1. Kelompok tidak diperkenankan untuk:

*Members of the group are prohibited from:*

* + - Melihat sebagian atau seluruh jawaban kelompok lain,

*Seeing a part or the whole answer from other groups,*

* + - Menyadur sebagian atau seluruh jawaban dari buku, catatan, video, dan jenis referensi lainnya,

*Retell a part or the whole answer from books, notes, videos, and other references,*

* + - Menyadur sebagian atau seluruh jawaban dari internet,

*Retell a part or the whole answer from the internet,*

* + - Mengumpulkan jawaban yang tidak sesuai dengan tema soal,

*Submitting an answer with a different theme from the given case,*

* + - Melakukan tindakan yang menyebabkan jawaban dicontek oleh orang lain atau kelompok lain, baik disengaja maupun tidak disengaja,

*Doing action that could result the answer being copied by someone or other groups, intentionally or unintentionally,*

* + - Melakukan tindakan kecurangan lainnya.

*Committing other dishonest actions.*

1. Jika kelompok terbukti melakukan tindakan seperti yang dicantumkan pada butir ke-1, maka nilai mahasiswa dan/atau kelompok yang melakukan kecurangan, baik menyontek atau dicontek, akan dinolkan sesuai dengan peraturan yang berlaku.

*If it has been proven that a group has committed dishonest actions outlined in point 1 above, the whole groups related to the incident, regardless of which one copies or has their answer copied, will be issued a score of zero according to the regulation.*

1. Jawaban yang dapat diterima dan dinilai adalah jawaban yang dikumpulkan sebelum batas waktu yang telah ditentukan.

*The answer must be submitted before the designated deadline to be accepted and graded,*

1. Jawaban akan dinilai berdasarkan teknik atau metode yang diajarkan pada kelas praktikum dengan menggunakan software yang sudah ditentukan.

*The scoring will be based on the materials taught during the practicum classes using the designated software. Using different software than requested may result in your answer not being graded.*

1. Jika Anda tidak membaca peraturan ini, maka Anda dianggap sudah membaca dan menyetujuinya.

*By taking this exam, you agree to these regulations, regardless of whether you have read it or not.*

1. Persentase penilaian untuk matakuliah ini adalah sebagai berikut:

*The score will be distributed as follows:*

|  |  |  |
| --- | --- | --- |
| **Tugas Mandiri**  *Assignment* | **Proyek**  *Project* | **UAP**  *Final Exam* |
| 40% | 60% | - |

1. Perangkat lunak yang digunakan pada matakuliah ini adalah sebagai berikut:

*This course uses the following software:*

|  |
| --- |
| **Software**  *Software* |
| ThreeJS r145  Visual Studio Code  Web Browser (Google Chrome) |

1. Ekstensi file yang harus dikumpulkan untuk matakuliah ini adalah sebagai berikut:

*Your answers must be in the following file extensions:*

|  |  |  |
| --- | --- | --- |
| **Tugas Mandiri**  *Assignment* | **Proyek**  *Project* | **UAP**  *Final Exam* |
| HTML, CSS, JS, Asset Files | HTML, CSS, JS, GLB, Asset Files | - |

1. File yang harus dikumpulkan adalah keseluruhan jawaban beserta dengan aset yang digunakan (gambar, audio, video, dll) dan dokumentasi proyek yang berisikan link referensi aset dan penjelasan mengenai aplikasi yang dibuat (terlampir bersama dengan soal).

*Include other files that can support your project, such as: all files in your project, other files (image, audio, video, etc.) used in your project, \*.doc file (documentation of your project) that contains all pages in your project, reference links of additional files (image, audio, video, etc.) used in your project, the description about how to use your application, etc.*

## Soal

*Case*

**Solar SysteMF**

As a **Three.js** enthusiast, you and your team were thrilled to take on a new project from a university: creating an interactive, 3D solar system. You decided to leverage the power of **Three.js** for this ambitious undertaking, your specific role was to bring the celestial bodies to life, meticulously crafting the **sun**, **eight planets** (**Mercury**, **Venus**, **Earth**, **Mars**, **Jupiter**, **Saturn**, **Uranus**, and **Neptune**), and a **spaceship** that would allow users to explore the vast expanse of the simulated solar system.

1. **Project Structure**

Your project should contain a main **html** file, several **JavaScript** files, **assets**, and the **three.js** library. You are to acquire **three.js** either from the three.js [official website](https://threejs.org/), [github repository](https://github.com/mrdoob/three.js/), or [CDN link](https://cdnjs.com/libraries/three.js).

You are **required to** **include** the following piece of code in your **html** file.

A computer code with black text

Description automatically generated***Figure 1. Required code***

You are **free to split** your code into several different **JavaScript file**, but code the **main logic** for creating the scene inside “**index.js**” file.

1. **Scene**

Create a **full screen** scene that can be **dynamically resized** to fit the window. The scene also has **shadow map enabled** using **PCFShadowMap** (**default**) as the shadow map type and **anti-aliasing** turned on.

1. **Camera**

Create **cameras** which details will be specified below.

1. **Freely Rotating Camera**

* This camera will have the following specifications:

|  |  |
| --- | --- |
| Property | Value |
| Type | Perspective Camera |
| Field of View | 75 |
| Aspect Ratio | Window Ratio |
| Position | Vector3 (640, 480, 240) |
| Near | 0.1 |
| Far | 10000 |

* This camera will focus on **Vector3** (**640, 320, 0**) **position** and can be **rotated** around said position using **OrbitControls.**

1. **Third Person Camera**

This camera will have the following specifications:

|  |  |
| --- | --- |
| Property | Value |
| Type | Perspective Camera |
| Field of View | 90 |
| Aspect Ratio | Window Ratio |
| Position | Spaceship’s Vector3 (x, y + 16, z – 16) position |
| Near | 0.1 |
| Far | 10000 |

This camera will focus on **Spaceship’s** **Vector3** (**x**, **y + 16**, **z - 16**) **position** and can be **rotated** around said position using **Spaceship’s Quaternion**. Since the spaceship can **move freely**, its **Vector3 position** will **change dynamically**. For example, **Spaceship’s Vector3** (**420**, **320**, **60**), then the **Third Person Camera Vector3** position is (**420**, (**320** + **16**), (**60** – **16**))

1. **Light**

There will be two **global lights** to illuminate the entire scene.

1. **Point Light**

* Below are the specifications:

|  |  |
| --- | --- |
| Property | Value |
| Intensity | 1 |
| Color | #FFFFFF |
| Distance | 1280 |
| Cast Shadow | Yes |
| Position | Vector3 (640, 320, 0) |

1. **Spotlight**

* Below are the spefications:

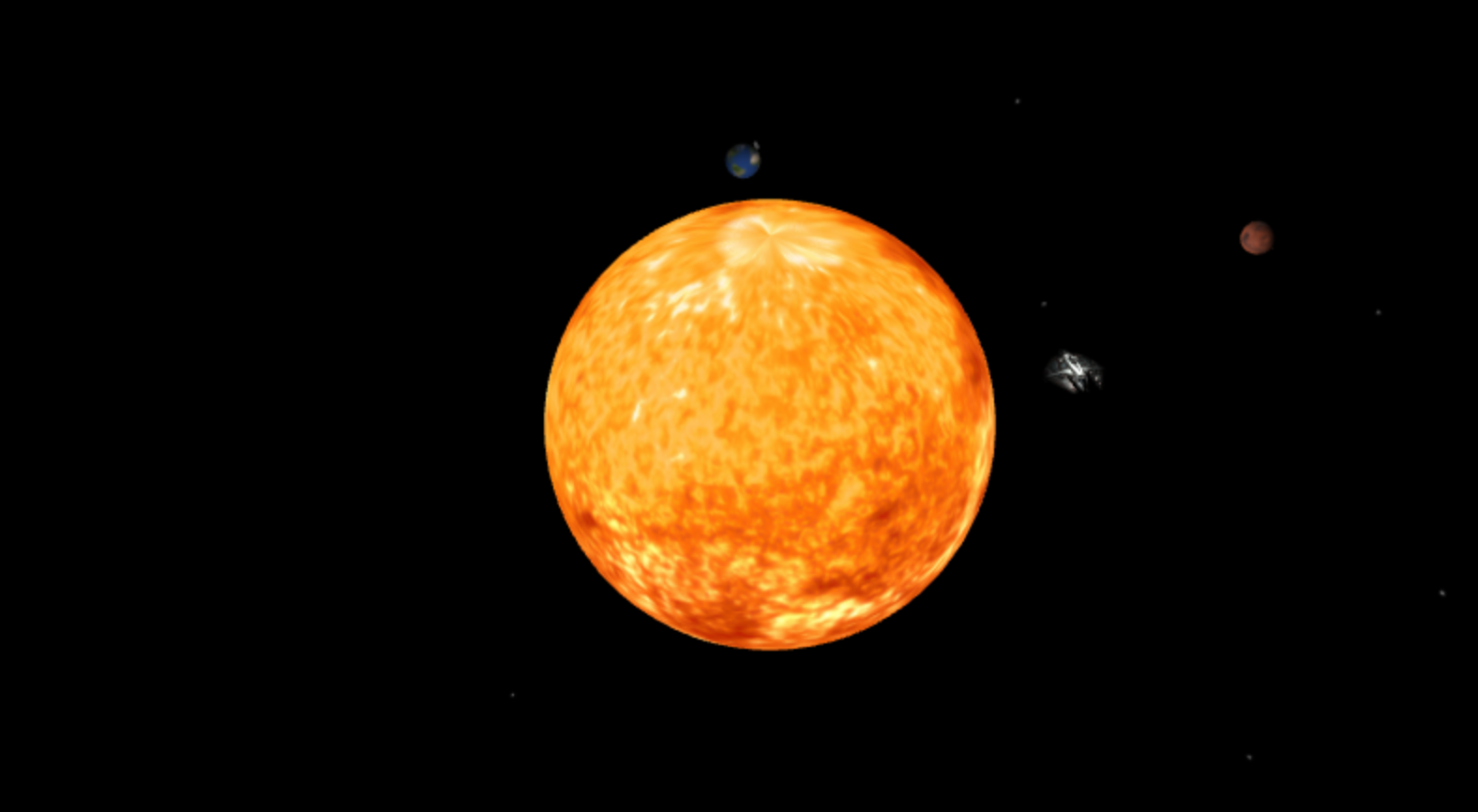
|  |  |
| --- | --- |
| Property | Value |
| Intensity | 8 |
| Color | #FFFFFF |
| Cast Shadow | No |
| Position | Spaceship’s Vector3 (x, y + 6, z) position |
| Distance | 8 |

This **spotlight** will focus on **Spaceship’s** **Vector3** (**x**, **y + 6**, **z**) **position**. Since the spaceship can **move freely**, its **Vector3 position** will **change dynamically**. For example, if the **Spaceship**’**s** Vector3 position is (**400**, **320**, **0**), then the **spotlight** Vector3 position is (**400**, (**320 + 6)**, **0**).

1. **Objects**
2. **Sun**

* Below are the specifications:

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 40 |
| Material Type | Mesh Basic Material |
| Color | #FFFFFF |
| Position | Vector3 (640, 320, 0) |
| Cast Shadow | No |
| Receive Shadow | No |
| Texture Map |  |

  
***Figure 2. Sun***

1. **Planets**

* Below are the specifications:
* **Mercury**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 3.2 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (58, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map | A grey and white surface  Description automatically generated |

* **Venus**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 4.8 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (80, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map |  |

* **Earth**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 4.8 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (100, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map |  |

* **Mars**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 4 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (130, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map |  |

* **Juptiter**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 13 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (175, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map |  |

* **Saturn**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 10 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (240, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map |  |

* **Uranus**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 8 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (280, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map |  |

* **Neptune**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Sphere |
| Radius | 6 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Vector3 (320, 320, 0) |
| Cast Shadow | Yes |
| Receive Shadow | Yes |
| Texture Map |  |

A group of planets in space

Description automatically generated  
**Figure 3. Planets**

1. **Spaceship**

* **Place** the object near earth
* **Load** model from the **GLTF** file “**src**/**assets**/**model**/**spaceship**/**textures**/**scene.gltf**”
* Below are the specifications:

|  |  |
| --- | --- |
| **Property** | **Value** |
| **Cast Shadow** | Yes |
| **Receive Shadow** | Yes |

  
***Figrue 4. Spaceship***

1. **Planet’s Ring**

* Below are the speficitaions:
  + **Saturn’s Ring**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Ring |
| Inner Radius | 16 |
| Outer Radius | 32 |
| Theta Segments | 64 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Saturn’s Vector3 (240, 320, 0) |
| Cast Shadow | No |
| Receive Shadow | Yes |
| Texture Map | A circular object with a black background  Description automatically generated |

Because of Saturn’sVector3 Position is **dynamically moving**, make sure that the Saturn’s ring will **move along with the Saturn’s position**.

* + **Uranus’ Ring**

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Ring |
| Inner Radius | 16 |
| Outer Radius | 20 |
| Theta Segments | 64 |
| Material Type | Mesh Standard Material |
| Color | #FFFFFF |
| Position | Uranus’ Vector3 (280, 320, 0) |
| Cast Shadow | No |
| Receive Shadow | Yes |
| Texture Map |  |

Because of Uranus’s Vector3 Position is **dynamically moving**, make sure that the Saturn’s ring will **move along with the Uranus’ position**.

1. **Satelite**

* Below are the specifications:

|  |  |
| --- | --- |
| Property | Value |
| Geometry Type | Cylinder |
| Radius Top | 1 |
| Radius Bottom | 0.5 |
| Height | 0.4 |
| Radial Segments | 8 |
| Material Type | Mesh Standard Material |
| Color | #CCCCCC |
| Metalness | 0.5 |
| Roughness | 0.5 |
| Position | Earth’s Vector3 (100 + 8, 320, 0) |
| Cast Shadow | No |
| Receive Shadow | Yes |

A white object in the sky

Description automatically generated ***Figure 5. Satelite***

1. **Texts**

* When a user is **hovering over** a sun or any planet, then **display a text** that shown the current object hovered **name** and **change the color** of the object **randomly**. And then when user **click the sun or planet**, make the **rotation spin faster** for a while and then **return the rotation speed to original**. Apply this interaction by using **Raycast**. Make sure that the **Raycast** only applied to **Free Rotating Camera**.
* Below is the color list:

|  |  |
| --- | --- |
| Color | Value |
| Electric Blue | #00FFFF |
| Neon Green | #00FF00 |
| Gold Yellow | #FFCC00 |
| Lavender Purple | #E6E6FA |
| Bright Pink | #FF69B4 |
| Bright Orange | #FF8C00 |
| Pink Pastel | #FFB6C1 |
| Cyan | #00FFFF |
| Sky Blue | #87CEEB |
| Green | #A8FFB2 |
| Bright Purple | #EE82EE |
| Bright Blue | #ADD8E6 |

**A green planet with planets in the background

Description automatically generated  
*Figure 6. Raycast***

***A planet with a ring in the center

Description automatically generated  
Figure 7. Raycast***

1. **Skybox**

* Below are the specifications:

|  |  |
| --- | --- |
| **Property** | **Value** |
| **Size** | 4260 x 4260 x 4260 |
| **Texture**  (In sequence: px, nx, py, ny, pz, nz) |  |