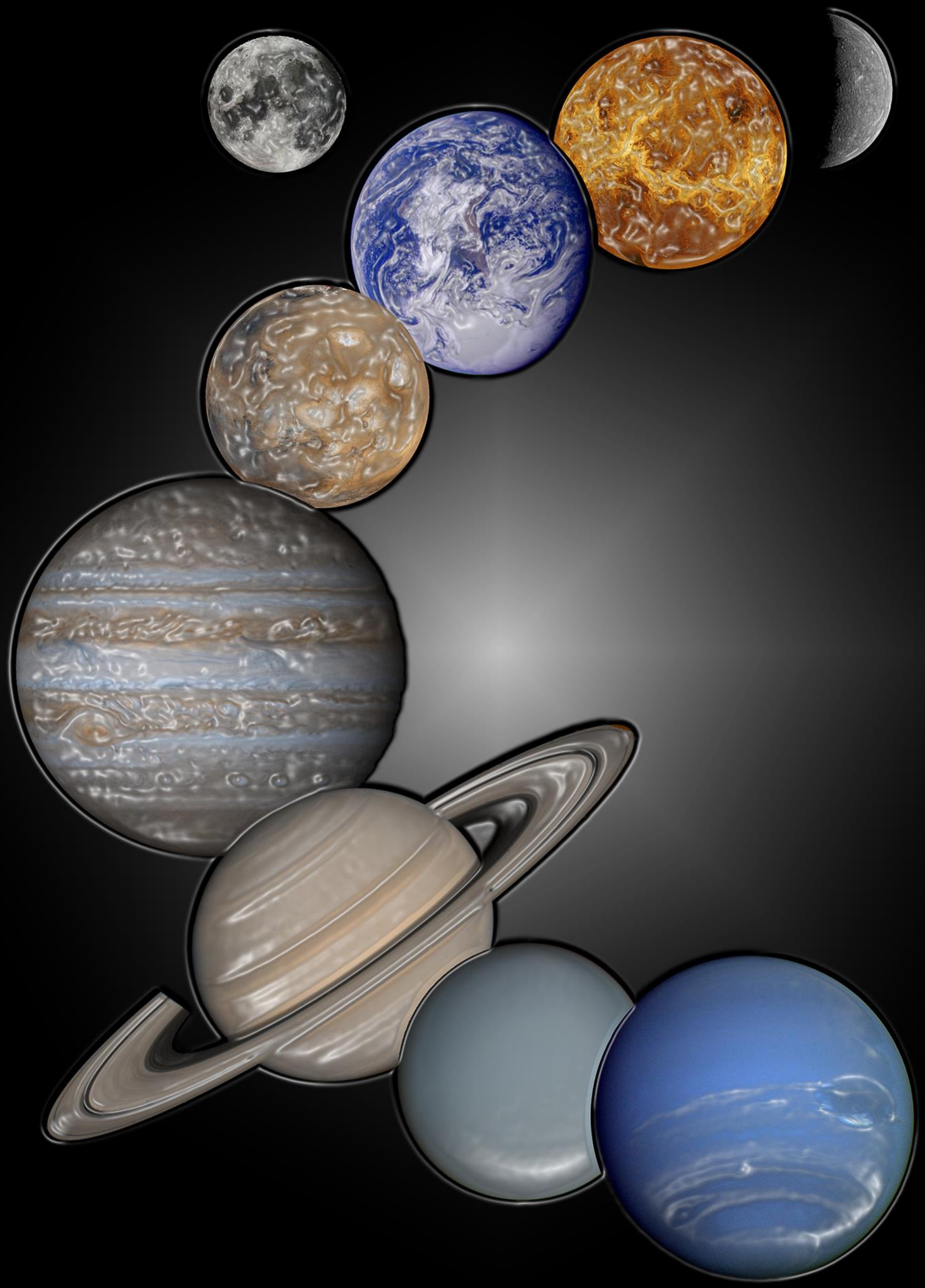


Sistema Solar 3D

António Moreira, 93279
Introduction to Computer Graphics – 2023/2024

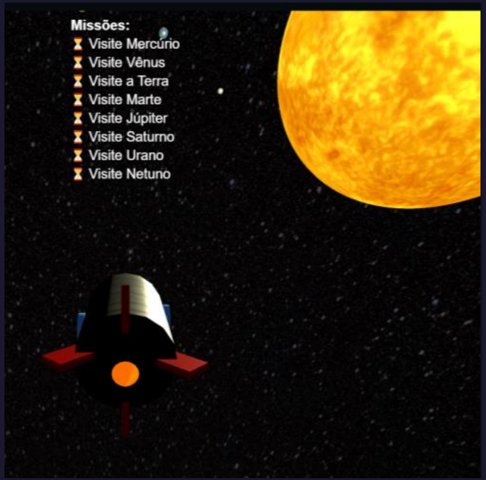
Main Idea

Create an interactive solar system in 3D, allowing the user to explore planets, learn about the solar system, and play two mini-games (Asteroids and Exploration).

Built with Three.js and custom JavaScript logic.

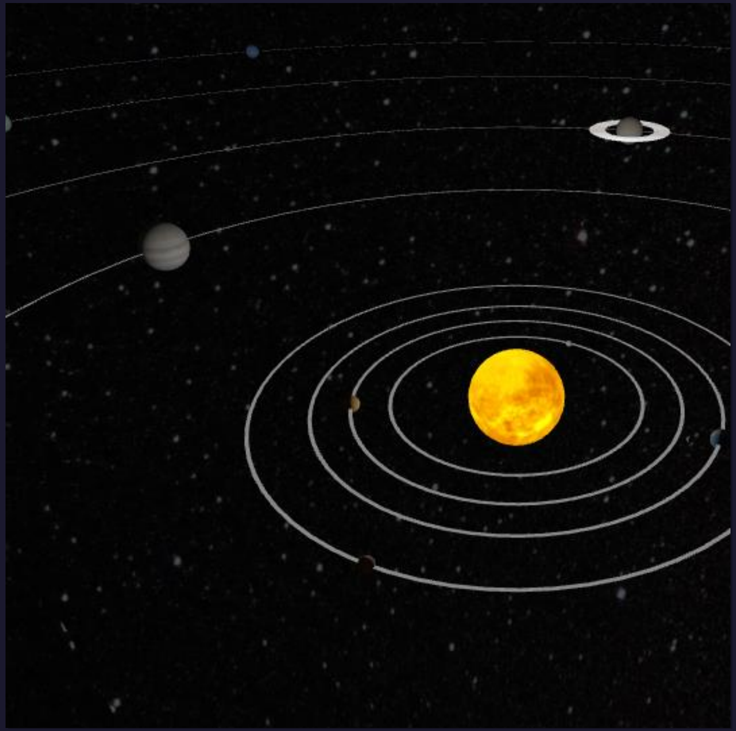
Demo:

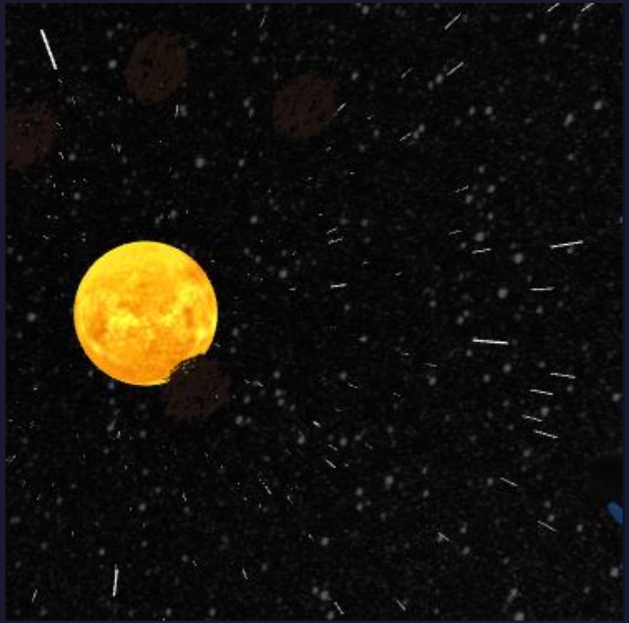
antoniomspmoreira13.github.io/ICG-projeto/



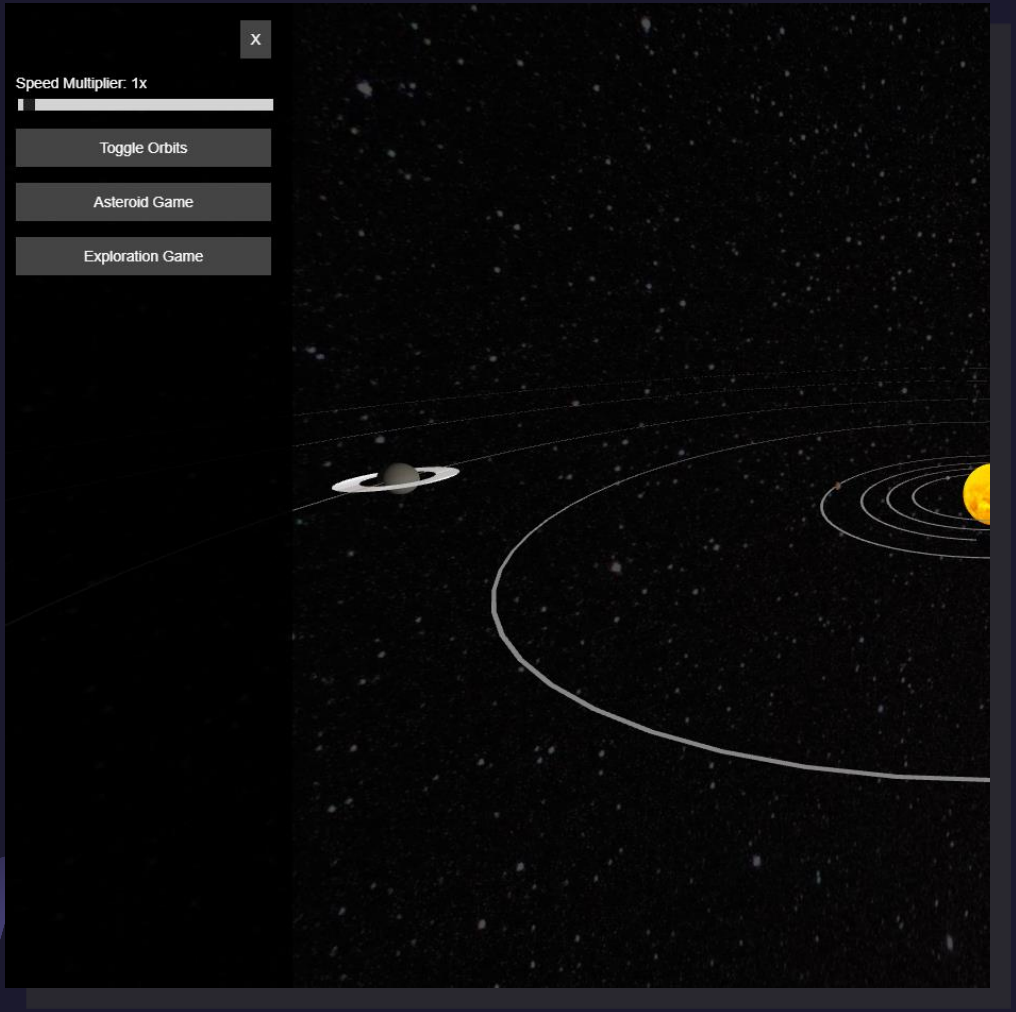
Missões:

- ⏳ Visite Mercúrio
- ⏳ Visite Vênus
- ⏳ Visite a Terra
- ⏳ Visite Marte
- ⏳ Visite Júpiter
- ⏳ Visite Saturno
- ⏳ Visite Urano
- ⏳ Visite Netuno





Initial Menu and Navigation

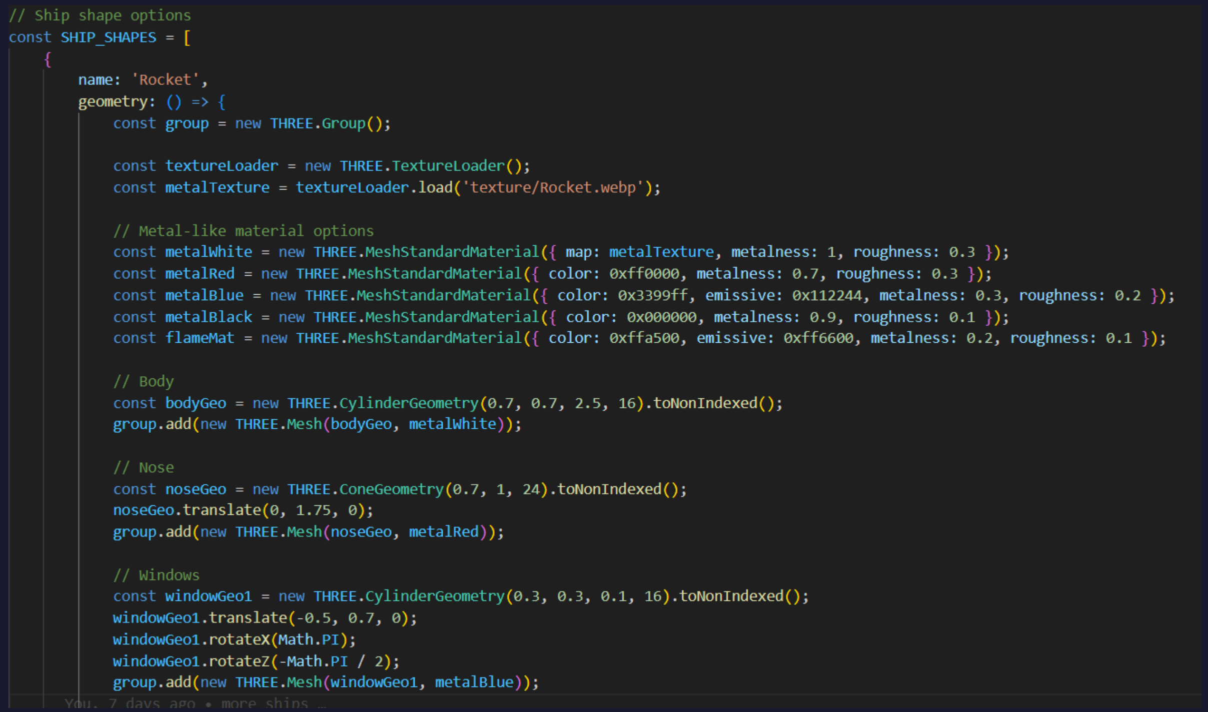


Floating menu (Drawer) allows the user to:

- Adjust orbital speed
- Toggle planet orbits
- Access Asteroid Game
- Access Exploration Game
- Instructions and controls available for each mode.

Models and Scene Graph

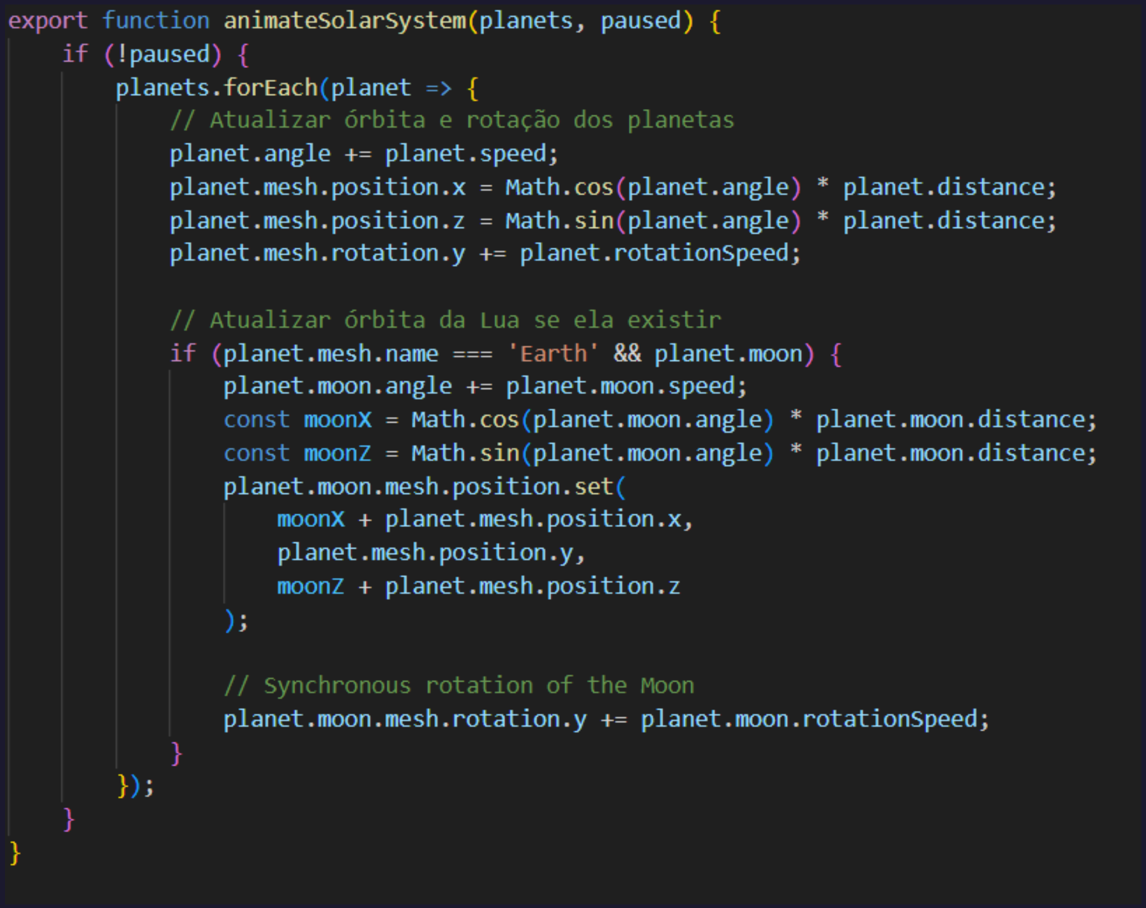
- Sun, 8 planets, and the Moon, each with realistic textures
- Saturn’s dynamic ring
- Custom rocket model for games
- Asteroids, coins, and backgrounds all generated in Three.js
- Planets are objects in the scene graph, with their own position, orbit, and rotation logic



```
// Ship shape options
const SHIP_SHAPES = [
    {
        name: 'Rocket',
        geometry: () => {
            const group = new THREE.Group();

            const textureLoader = new THREE.TextureLoader();
            const metalTexture = textureLoader.load('texture/Rocket.webp');

            // Metal-like material options
            const metalWhite = new THREE.MeshStandardMaterial({ map: metalTexture, metalness: 1, roughness: 0.3 });
            const metalRed = new THREE.MeshStandardMaterial({ color: 0xff0000, metalness: 0.7, roughness: 0.3 });
            const metalBlue = new THREE.MeshStandardMaterial({ color: 0x3399ff, emissive: 0x112244, metalness: 0.3, roughness: 0.2 });
            const metalBlack = new THREE.MeshStandardMaterial({ color: 0x000000, metalness: 0.9, roughness: 0.1 });
            const flameMat = new THREE.MeshStandardMaterial({ color: 0xffa500, emissive: 0xff6600, metalness: 0.2, roughness: 0.1 });

            // Body
            const bodyGeo = new THREE.CylinderGeometry(0.7, 0.7, 2.5, 16).toNonIndexed();
            group.add(new THREE.Mesh(bodyGeo, metalWhite));

            // Nose
            const noseGeo = new THREE.ConeGeometry(0.7, 1, 24).toNonIndexed();
            noseGeo.translate(0, 1.75, 0);
            group.add(new THREE.Mesh(noseGeo, metalRed));

            // Windows
            const windowGeo1 = new THREE.CylinderGeometry(0.3, 0.3, 0.1, 16).toNonIndexed();
            windowGeo1.translate(-0.5, 0.7, 0);
            windowGeo1.rotateX(Math.PI);
            windowGeo1.rotateZ(-Math.PI / 2);
            group.add(new THREE.Mesh(windowGeo1, metalBlue));
        You, 7 days ago • more ships …
```

Animation

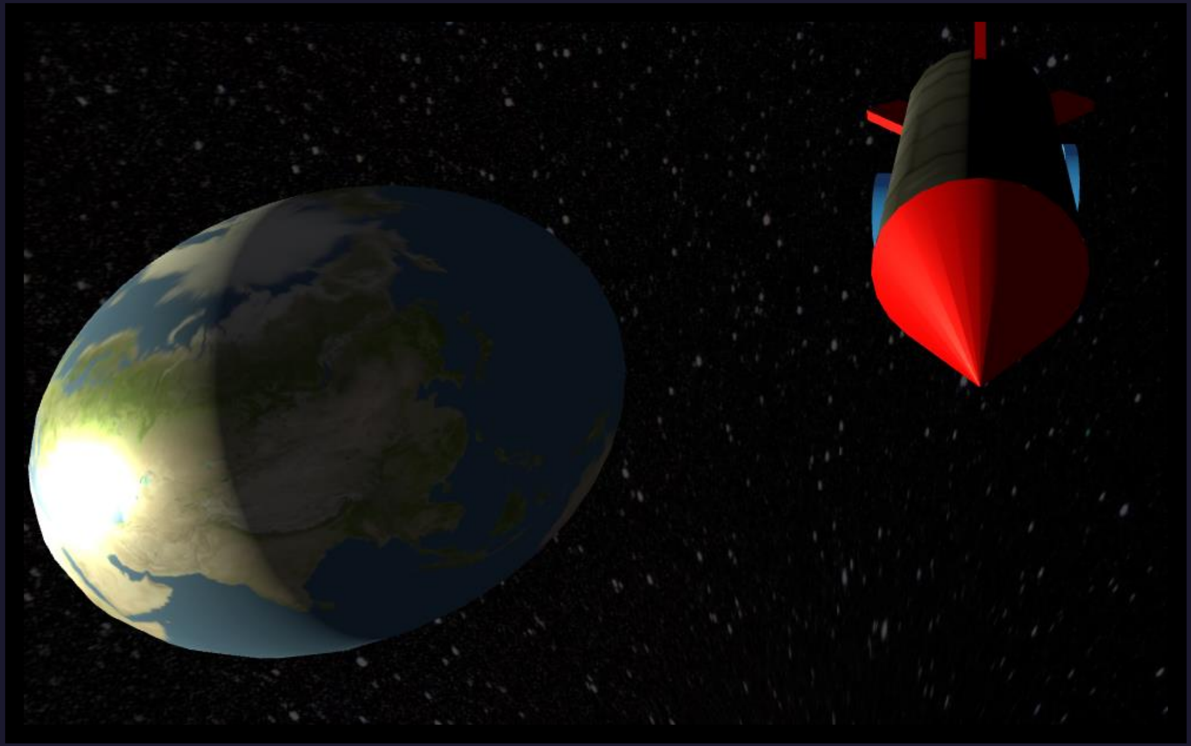
- Orbital and rotational movement of all planets
- Moon orbits Earth in real time
- Rockets, asteroids, and coins are animated in their respective games
- Camera follows selected planet or rocket in a smooth third-person perspective



```
export function animateSolarSystem(planets, paused) {
    if (!paused) {
        planets.forEach(planet => {
            // Atualizar órbita e rotação dos planetas
            planet.angle += planet.speed;
            planet.mesh.position.x = Math.cos(planet.angle) * planet.distance;
            planet.mesh.position.z = Math.sin(planet.angle) * planet.distance;
            planet.mesh.rotation.y += planet.rotationSpeed;

            // Atualizar órbita da Lua se ela existir
            if (planet.mesh.name === 'Earth' && planet.moon) {
                planet.moon.angle += planet.moon.speed;
                const moonX = Math.cos(planet.moon.angle) * planet.moon.distance;
                const moonZ = Math.sin(planet.moon.angle) * planet.moon.distance;
                planet.moon.mesh.position.set(
                    moonX + planet.mesh.position.x,
                    planet.mesh.position.y,
                    moonZ + planet.mesh.position.z
                );

                // Synchronous rotation of the Moon
                planet.moon.mesh.rotation.y += planet.moon.rotationSpeed;
            }
        });
    }
}
```

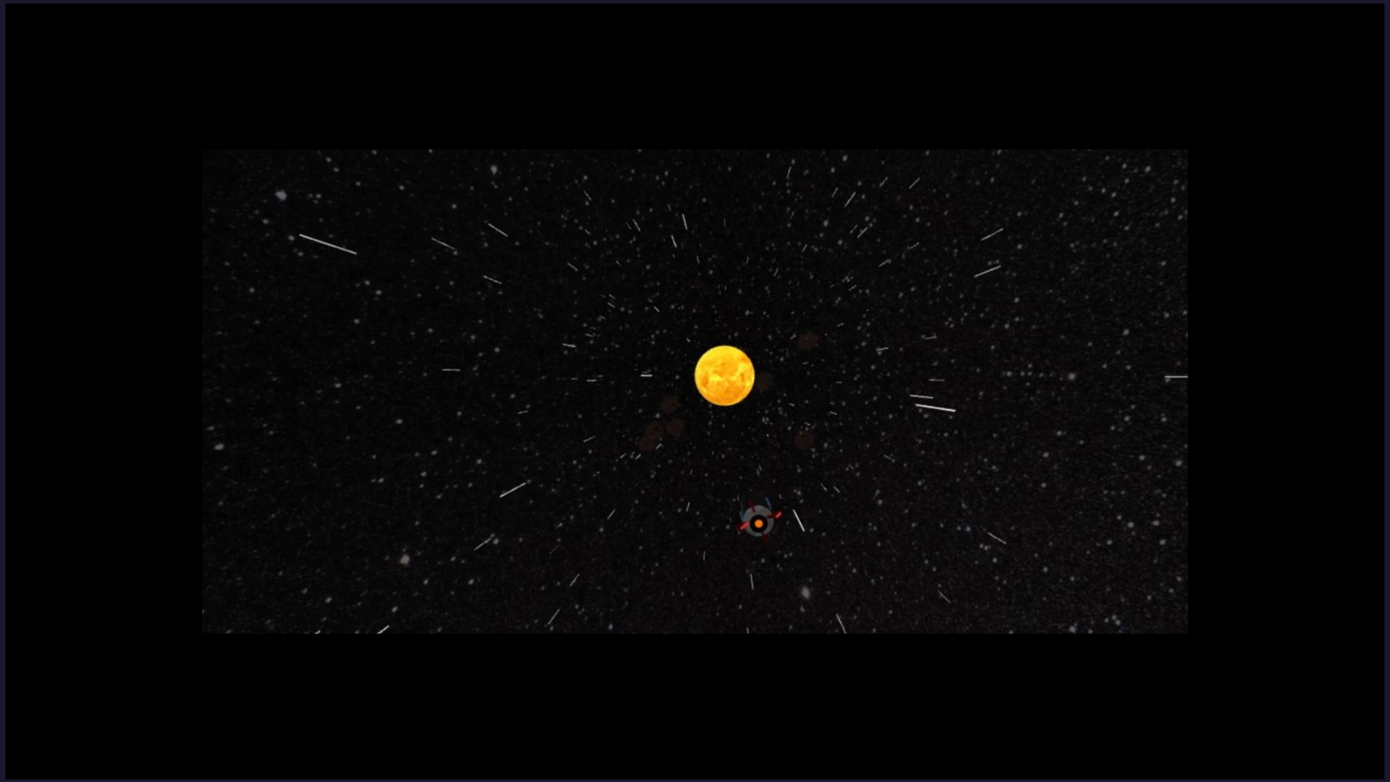
Illumination

- Point light representing the Sun
- Ambient light for overall scene
- Emissive materials for Sun and some objects
- Dynamic shadows on planets and objects



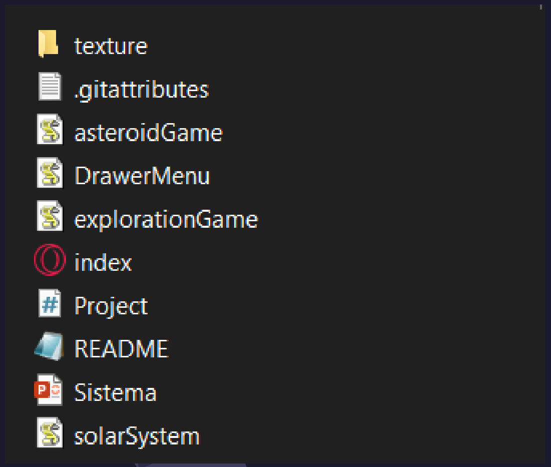
Effects

- Collisions, coin collection, and thrust (optional/bonus)
- Visual feedback: invulnerability flashing, score display, HUD for missions
- Star background for immersion



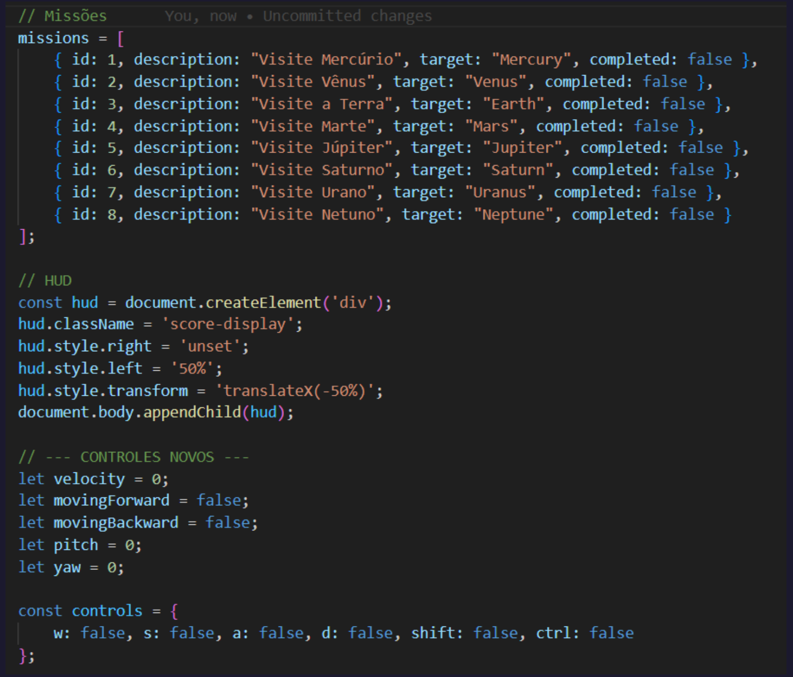
Development

Code modularized: solarSystem.js, DrawerMenu.js, asteroidGame.js, explorationGame.js



texture
.gitattributes
asteroidGame
DrawerMenu
explorationGame
index
Project
README
Sistema
solarSystem

- State management for menu, pause, and active game



```
// Missões       You, now • Uncommitted changes
missions = [
    { id: 1, description: "Visite Mercúrio", target: "Mercury", completed: false },
    { id: 2, description: "Visite Vênus", target: "Venus", completed: false },
    { id: 3, description: "Visite a Terra", target: "Earth", completed: false },
    { id: 4, description: "Visite Marte", target: "Mars", completed: false },
    { id: 5, description: "Visite Júpiter", target: "Jupiter", completed: false },
    { id: 6, description: "Visite Saturno", target: "Saturn", completed: false },
    { id: 7, description: "Visite Urano", target: "Uranus", completed: false },
    { id: 8, description: "Visite Netuno", target: "Neptune", completed: false }
];

// HUD
const hud = document.createElement('div');
hud.className = 'score-display';
hud.style.right = 'unset';
hud.style.left = '50%';
hud.style.transform = 'translateX(-50%)';
document.body.appendChild(hud);

// --- CONTROLES NOVOS ---
let velocity = 0;
let movingForward = false;
let movingBackward = false;
let pitch = 0;
let yaw = 0;

const controls = {
    w: false, s: false, a: false, d: false, shift: false, ctrl: false
};
```

Main challenges:

- Managing large scale and camera transitions
- Ensuring performance with many objects
- Handling user inputs for multiple game modes

Conclusions

- Achieved an interactive, educational and fun solar system project

- Explored advanced 3D graphics, animation, and game logic

- Next steps:

  - More missions and items
  - Multiplayer or leaderboard online
  - Deeper educational content