

Computergrafik

Programmentwurf

über die Theoriephasen des dritten Studienjahrs

an der Fakultät für Technik
im Studiengang Informatik

an der DHBW Ravensburg
Campus Friedrichshafen

von

Johannes Brandenburger, Lukas Braun, Henry Schuler

21. November 2022

Bearbeitungszeitraum: 01.10.2022 - 21.11.2022

Kurs: TIT20

Dozent der Hochschule: Prof. Dr. Jürgen Schneider

Gender Erklärung

Aus Gründen der besseren Lesbarkeit wird in dieser Bachelorarbeit auf die gleichzeitige Verwendung der Sprachformen männlich, weiblich und divers (m/w/d) verzichtet. Sämtliche Formulierungen gelten gleichermaßen für alle Geschlechter.

Selbstständigkeitserklärung

gemäß Ziffer 1.1.13 der Anlage 1 zu §§ 3, 4 und 5 der Studien- und Prüfungsordnung für die Bachelorstudiengänge im Studienbereich Technik der Dualen Hochschule Baden-Württemberg vom 29.09.2017.

Wir versichern hiermit, dass wir unsere Bachelorarbeit (bzw. Projektarbeit oder Studienarbeit bzw. Hausarbeit) mit dem Thema:

Computergrafik

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben. Wir versichern zudem, dass die eingereichte elektronische Fassung mit der gedruckten Fassung übereinstimmt.

Friedrichshafen, 21. November 2022

Ort, Datum



Johannes Brandenburger

Friedrichshafen, 21. November 2022

Ort, Datum



Lukas Braun

Friedrichshafen, 21. November 2022

Ort, Datum



Henry Schuler

Inhaltsverzeichnis

Genderklärung	II
Selbstständigkeitserklärung	III
Abbildungsverzeichnis	V
Tabellenverzeichnis	VI
1 Einleitung	1
1.1 Aufgabenstellung	1
1.2 Aufbau der Arbeit	1
2 Tools	2
3 Designkonzept	3
3.1 Klassenzimmer	3
3.2 Flugsimulator	7
4 Realisierung	8
4.1 Klassenzimmer	8
4.2 Flugsimulator	9
4.3 Architektur	10
5 Final Product	12
5.1 Klassenzimmer	12
5.2 Flugsimulator	14
6 Installationsanleitung	15
A Anhang	A

Abbildungsverzeichnis

3.1 Klassenzimmer Skizze	3
3.2 Klassenzimmer Entwurf mit Bemaßung	4
3.3 Klassenzimmer Entwurf mit Fenster	5
3.4 Klassenzimmer Entwurf der Lampen	5
4.1 Flowchart des kompletten Projekts	10
4.2 Flowchart des Klassenzimmers	11
4.3 Flowchart des Flugsimulators	11
5.1 Klassenzimmer alle Lampen	12
5.2 Klassenzimmer nur ein Lampencluster	12
5.3 Klassenzimmer ohne Schatten	13
5.4 Klassenzimmer Aussicht	13
5.5 Flugsimulator	14
5.6 Flugsimulator Game Over	14

Tabellenverzeichnis

3.1 Modell-Maße in Meter	6
3.2 Modelle aus dem Internet	6

1 Einleitung

1.1 Aufgabenstellung

Die Prüfungsleistung der Vorlesung Computergrafik beinhaltet die Erstellung eines Programm-entwurfs.

Dieser Programmentwurf besteht aus der Erstellung einer animierten 3D-Computergrafik. Hierzu sollen HTML, CSS, JavaScript und WebGLv2 verwendet werden. Zur Modellierung der Szenen darf außerdem die three.js Bibliothek verwendet werden. Der Programmentwurf muss folgende Punkte enthalten:

- Szene ist dreidimensional
- Einzelne Objekte in der Szene sind animiert
- Kamera kann sich durch die Szene bewegen
- Mindestens eine Lichtquelle mit Phong-Beleuchtungmodell
- Control Panel zur Steuerung der 3D-Grafik

Das Controlpanel kann auch durch Interaktionen mit der Szene ersetzt werden.

1.2 Aufbau der Arbeit

Im folgenden werden zunächst die verwendeten Hilfsmittel erläutert, im Anschluss wird ein Konzept für die 3D-Szene erarbeitet und in verschiedenen Diagrammen dargestellt. Abschließend wird das finale Produkt dargestellt und eine Installationsanleitung zur Verfügung gestellt.

2 Tools

Wie in der Einleitung erwähnt werden für den Programmentwurf die Programmiersprachen HTML, CSS, JavaScript und WebGLv2 verwendet. Die 3D Szenenmodellierung wird mit der three.js Bibliothek realisiert. Zum Erstellen der 3D Modelle wird Blender verwendet. Diese können anschließend als glTF (GL Transmission Format) in three.js importiert werden.

Zur Entwicklung des Sourcecodes wird der Editor Visual Studio Code verwendet. Um die aktuelle Website zu starten wird ein node-Server verwendet. Der Sourcecode wird in einem Git Repository auf GitHub verwaltet. Zur Dokumentation des Quellcodes wird JSDoc verwendet, dieses veranschaulicht sämtliche Funktionen und deren Kommentare in einer automatisch generierten Website. Ergänzend dazu wird das Python Paket code2flow verwendet, welches Flowcharts, die die Aufrufhierarchie der verwendeten Funktionen darstellt, erstellt.

Um die Szene zu entwickeln und Entwürfe grafisch darzustellen wird Microsoft Visio verwendet. Händische Zeichnungen werden mit Microsoft OneNote oder GoodNotes abhängig vom Teammitglied erstellt, da GoodNotes nur auf Apple Geräten verfügbar ist.

Alle verwendeten Hilfsmittel werden in der folgenden Auflistung dargestellt:

- Blender
- code2flow
- GitHub
- GoodNotes
- JsDoc
- Microsoft OneNote
- Microsoft Visio
- Node Server
- Visual Studio Code

3 Designkonzept

3.1 Klassenzimmer

Als grundlegende Idee wurde zunächst ein Vorlesungssaal vorgeschlagen. Um weitere Komponenten aus den Anforderungen an diese Arbeit sinnvoll umzusetzen, wurde die grundlegende Idee überdacht und neu definiert als Klassenzimmer einer Flugschule.

Um eine erste Vorstellung des Klassenzimmers zu bekommen wurde zunächst eine händische Zeichnung angefertigt. Diese ist in Abbildung 3.1 dargestellt.

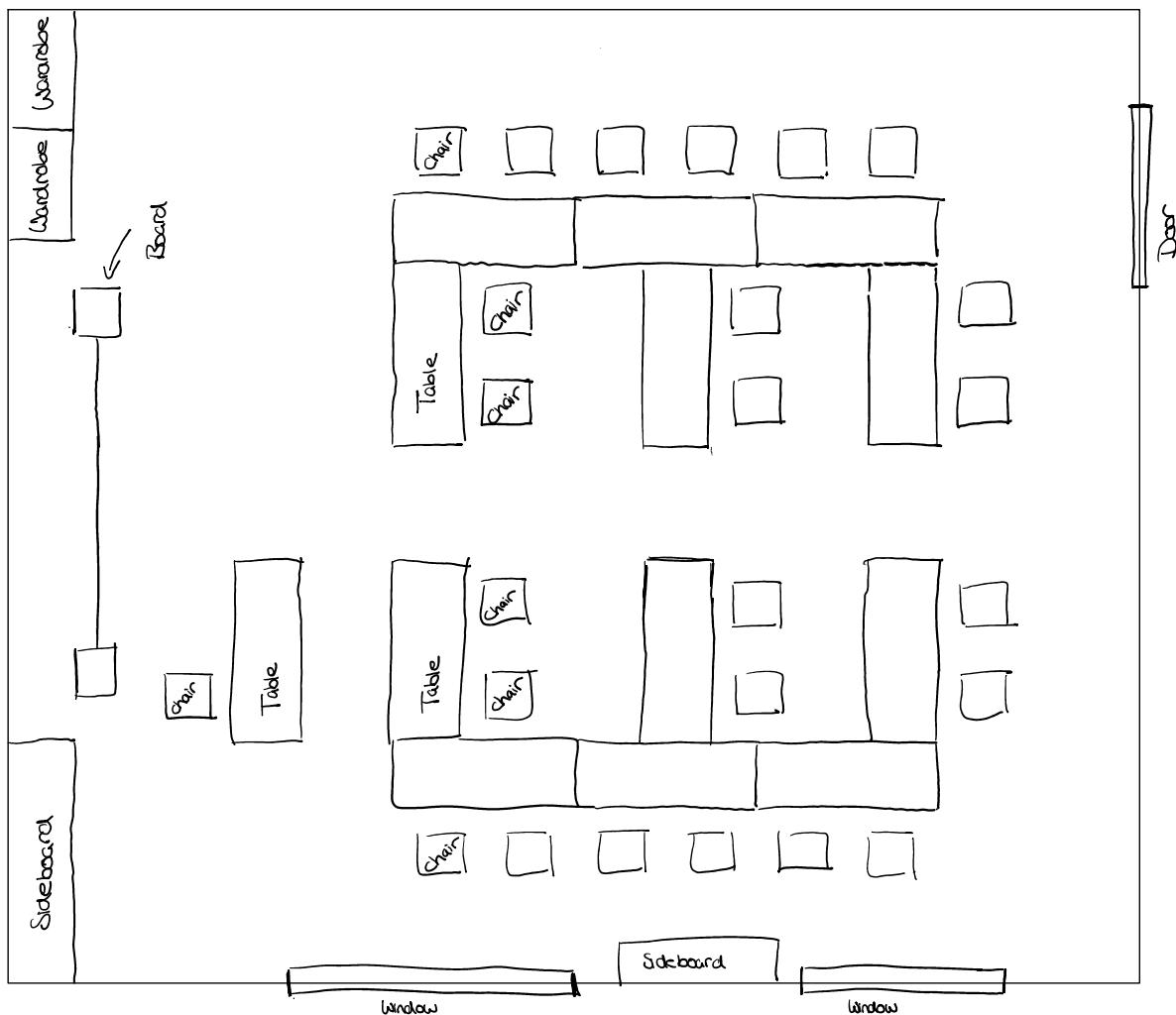


Abb. 3.1: Klassenzimmer Skizze

Anschließend wurde der Raum maßstabsgetreu in einem Bauplan gezeichnet um so die Abstände und Maße zu definieren. Diese Zeichnung ist in Abbildung 3.2 dargestellt.

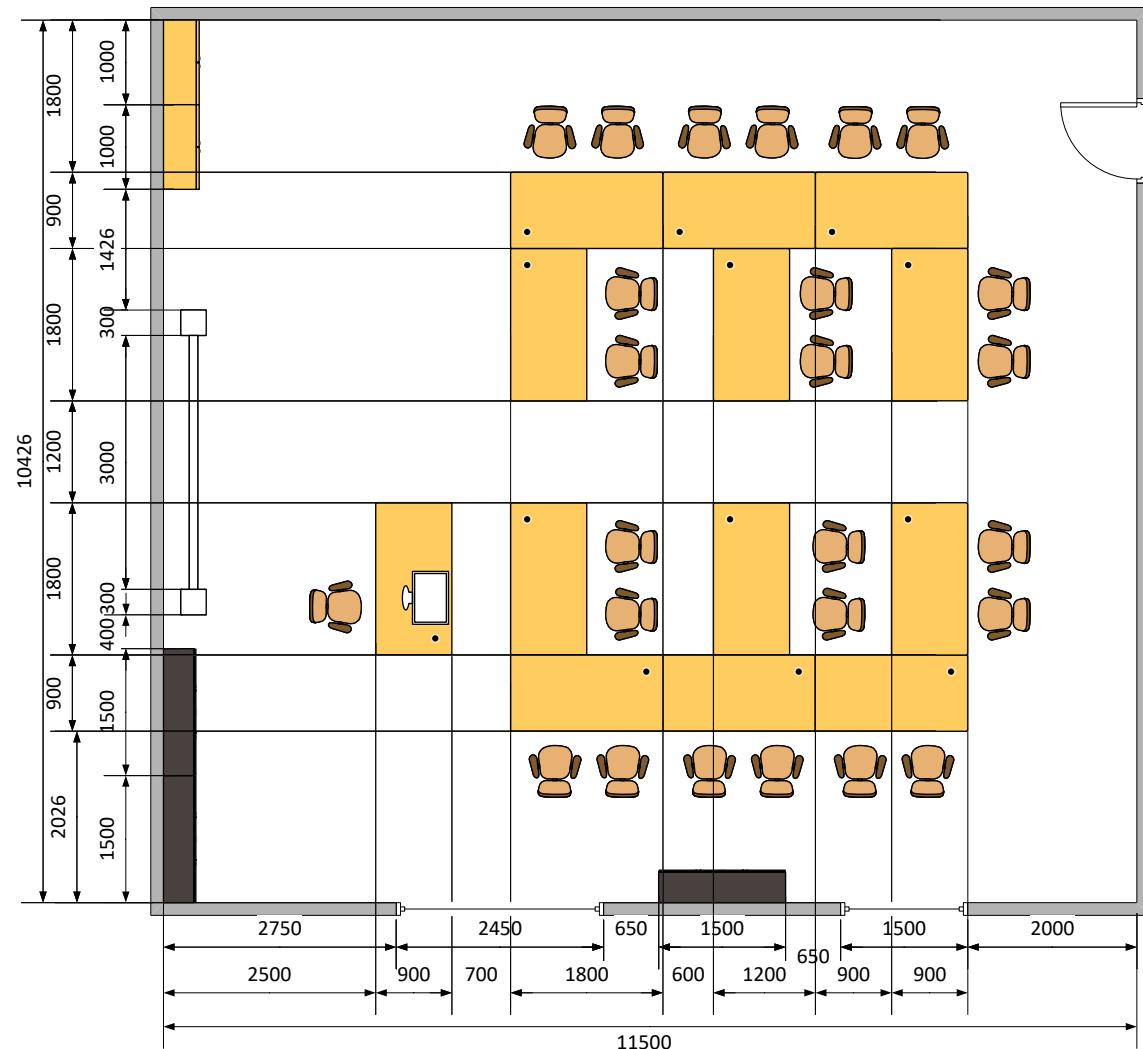


Abb. 3.2: Klassenzimmer Entwurf mit Bemaßung

Um die Höhen der Fenster zu bestimmen wurde zusätzlich eine Seitenansicht erstellt. Diese ist in Abbildung 3.3 zu sehen.

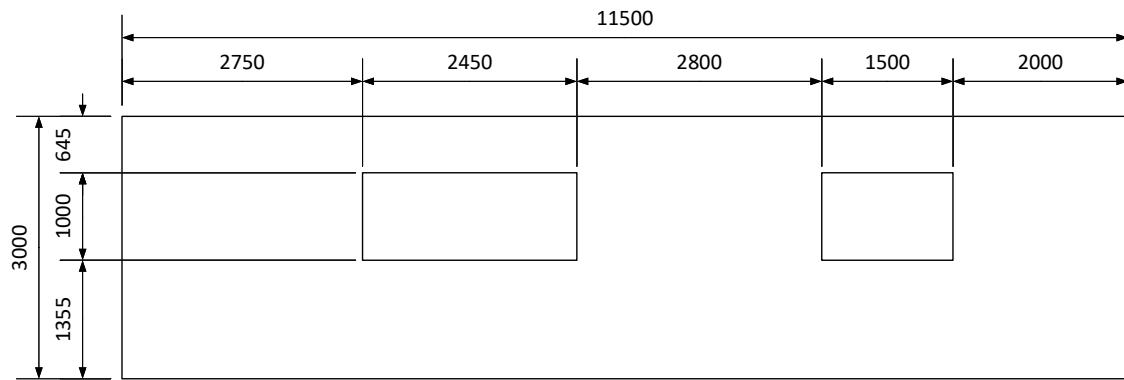


Abb. 3.3: Klassenzimmer Entwurf mit Fenster

Abschließend wurde eine weitere Zeichnung zur Platzierung der Lampen erstellt. Diese ist in Abbildung 3.4 dargestellt.

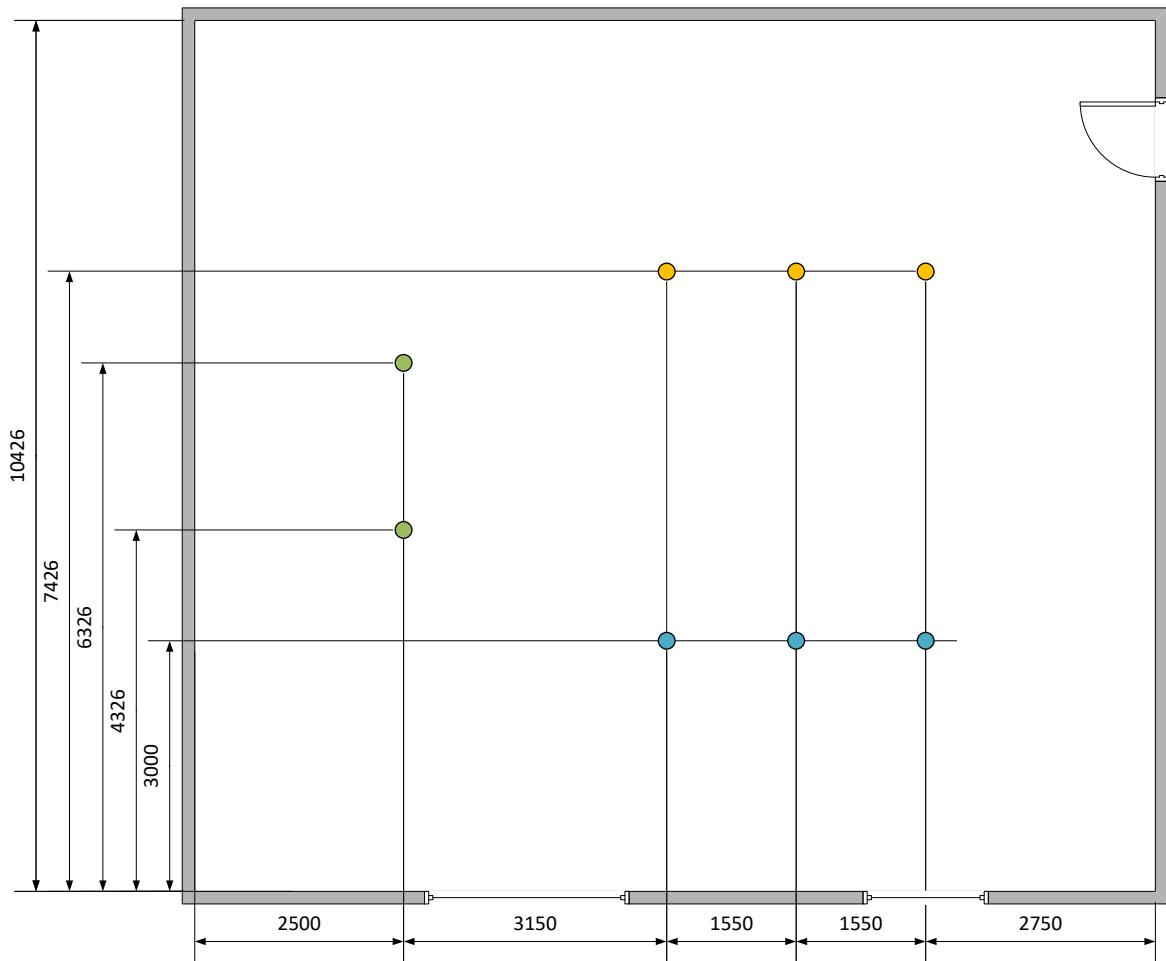


Abb. 3.4: Klassenzimmer Entwurf der Lampen

Um die Anforderungen vollständig zu erfüllen müssen Interaktionen mit der 3D Szene möglich sein, diese werden im Folgenden beschrieben.

Im Klassenzimmer ist es möglich zu laufen, bei einer Kollision mit einem Gegenstand wird die Bewegung angehalten. Das Licht im Klassenzimmer kann durch drei Lichtschalter neben der Tür per Mausklick gesteuert werden. Mit einem Schalter kann jeweils ein Cluster angesteuert werden, diese sind in Abbildung 3.4 farblich gekennzeichnet. Außerdem können die Stühle auf- und abgestellt werden. Zusätzlich können die Schränke geöffnet und geschlossen werden und die Tafel nach oben bzw. nach unten geschoben werden. Bei einem Blick aus dem Fenster soll der Ausblick aus der DHBW in Richtung Bodensee dargestellt werden.

Aus den beschriebenen Animationen und den Plänen aus den Abbildungen 3.2, 3.3 und 3.4 ergeben sich alle Komponenten der Szene. Einige Maße werden bereits durch den Plan vorgegeben in einem weiteren Schritt werden diese nun vollständig definiert. Diese Definitionen sind in der Tabelle 3.1 abgebildet.

Komponente	x	y	z	file
Blackboard	03,600	00,240	02,500	/blender/blackboard.glb
Chair	00,470	00,480	00,870	/blender/chair.glb
Closet	01,000	00,725	02,200	/blender/closet.glb
Lamp	00,328	00,328	00,700	/blender/lamp.glb
LightSwitch	00,080	00,014	00,080	/blender/lightswitch.glb
Sideboard	00,600	01,500	00,700	/blender/sideboard.glb
Table	01,800	00,900	00,790	/blender/table.glb
Room	10,710	11,820	03,300	/blender/room_door.glb

Tab. 3.1: Modell-Maße in Meter

Mit den definierten Größen aus Tabelle 3.1 können anschließend die Blender Modelle erstellt werden. Um das Klassenzimmer ansprechender darzustellen werden weitere Elemente aus dem Internet eingefügt, diese werden mit ihren Quellen in Tabelle 3.2 aufgezählt.

Komponente	Quelle	file
Monitor	https://poly.pizza/m/e8cELDeDuTr	/blender/monitor.glb
Keyboard	https://www.thingiverse.com/thing:4230507	/blender/keyboard.glb
Plane	https://sketchfab.com/3d-models/low-poly-airplane-65cc7c4349174f7bbb20ed70206377b5	/flight-simulator/low-poly_airplane.glb-low

Tab. 3.2: Modelle aus dem Internet

3.2 Flugsimulator

Als Erweiterung der Flugschule, wird ein Flugsimulator-Spiel integriert, welches folgende Funktionen beinhaltet:

Durch einen Klick auf den Monitor in der Flugschule wird auf eine Unterseite weitergeleitet und der Flugsimulator gestartet. Ein Flugzeug-Modell wird in der Mitte der Szene platziert. Dessen Flugrichtung ist mit der Maus veränderbar, wodurch eine Steuerung des Flugzeugs ermöglicht wird. Die Geschwindigkeit des Flugzeugs wird nicht durch den Benutzer verändert, sondern passt sich anhand der Neigung des Flugzeugs an. Mit einem Mausklick kann dem Flugzeug ein kleiner Boost gegeben werden.

Beim Starten des Flugsimulator-Spiels wird ein Countdown gestartet, welcher 60 Sekunden lang dauert. In dieser Zeit kann der Benutzer durch zufällig generierte Ringe fliegen und dabei (je nach Art des Rings) einen oder fünf Punkte sammeln. Zur zusätzlichen Herausforderung werden Hindernisse generiert. Wenn das Flugzeug in eines dieser Hindernisse oder gegen einen der Ringe fliegt oder die Zeit abläuft, wird das Spiel beendet. Der Benutzer bekommt hierbei die erreichte Punktzahl angezeigt und kann das Spiel erneut starten oder zur Flugschule zurückkehren.

Im Hintergrund des Spiels wird ein Ozean und ein Himmel dargestellt. Die Kamera bewegt sich während des Fluges mit dem Flugzeug mit. Die Steuerung kann durch das Drücken der Taste „I“ invertiert werden, um allen Benutzern das bestmögliche Spielerlebnis zu ermöglichen.

4 Realisierung

4.1 Klassenzimmer

Zunächst werden alle Modelle aus Blender in das GL Transmission Format (.glb) exportiert. Anschließend werden diese Modelle in three.js eingebunden und gemäß dem Plan in Abbildung 3.2 platziert. Um bei einem Blick aus dem Fenster den Ausblick aus der DHBW Richtung Bodensee zu sehen wurde ein 3D Bild erstellt. Dieses wird ebenfalls in three.js eingebunden. Abschließend müssen die Scheiben in die Fenster eingefügt werden, da dies nicht mit einem Export aus Blender möglich ist.

Nachdem die Szene vollständig erstellt wurde, müssen nachfolgend die definierten Interaktionen hinzugefügt werden. Um sich im Raum zu bewegen, wird ein unsichtbarer Quader hinzugefügt, der die Person darstellt. Dieser Quader enthält auch die Kamera und kann mit den Tasten „W“, „A“, „S“ und „D“ durch den Raum bewegt werden. Um Kollisionen zu erkennen wird vor Aufführung der Bewegung überprüft, ob die Bewegung ausgeführt werden darf. Würde der Quader innerhalb eines anderen Objekts sein, wird die Bewegung nicht ausgeführt.

Damit die Tafel nach oben bzw. nach unten bewegt werden kann, wird die Tafel mit der Maus angewählt und verschoben. Hierzu wird das Modul threex verwendet, welches bei einem Mausklick ein Ray aussendet und überprüft ob dieses das Tafelobjekt trifft. Ist dies der Fall, wird diese anschließend abhängig von der Mausbewegung in die Y-Achsen Richtung verschoben. Das Auf- und Abstühlen der Stühle funktioniert ähnlich. Es wird ebenfalls überprüft ob bei einem Mausklick auf einen Stuhl geklickt wurde und anschließend wird dieser auf- oder abgestuhlt. Das Öffnen oder Schließen der Schranktüren funktioniert analog zu den Stühlen. Allgemein gilt für alle Aktionen, dass der Abstand zwischen Personen-Quader und dem Objekt maximal vier Meter betragen darf damit die Animation ausgeführt wird.

Das Rendern der Schatten aller Elemente führt zu hohen Auslastungen mancher Systeme, deshalb wurde ein Performance Mode hinzugefügt, so können über die Tasten „C“ bzw. „T“ die Schatten für die Stühle bzw. Tische aktiviert oder deaktiviert werden.

Das Laden der verschiedenen Modelle in die Szene benötigt einen längeren Zeitraum, weshalb zusätzlich eine Landingpage hinzugefügt wurde, sodass die Modelle im Hintergrund geladen werden können. Die Landingpage gibt einen Ausblick über die Features des erstellten Programms und listet die Interaktionen und Hotkeys im Klassenzimmer und im Flugsimulator. Die Landingpage kann jederzeit mit dem Hotkey „H“ aufgerufen werden und dient somit als Hilfe.

4.2 Flugsimulator

Um das Flugsimulator-Spiel zu realisieren, wird zunächst eine Umgebung benötigt. Diese wird mit Hilfe von vorgefertigten three.js Elementen wie einem Meer und einem Himmel erstellt. Anschließend wird ein Flugzeugmodell im .glb Format eingelesen und der Szene hinzugefügt.

Da die FlyControls von three.js nicht den Ansprüchen des Spiels genügen, wird eine eigene Steuerung für das Bewegen des Flugzeugs implementiert. Dabei wird die Cursorposition mit Abstand zum Mittelpunkt des Bildschirms verglichen und die Flugrichtung (Vektor) entsprechend angepasst. Für diese Anpassung werden die Helper-Funktionen `turnVectorAroundVerticalAxis` und `turnVectorAroundHorizontalAxis` implementiert und in jeder Animationsschleife aufgerufen. Speziell für das Drehen eines Vektors um die horizontale Achse, ist die Mathematik nicht trivial, da zunächst eine Rotationsachse berechnet werden muss, die senkrecht zur Flugrichtung steht und bei jeder Position des Flugzeugs die gleiche Richtung hat. Eine besondere Herausforderung bei der Flugsteuerung stellen die Überschläge des Flugzeugs dar. Diese müssen vom Programm erkannt werden, um aktiv die Kamera, das Flugzeug und die Steuerung zu invertieren. Um die Geschwindigkeit des Flugzeugs möglichst realistisch zu gestalten, wird diese über die Beschleunigung des Flugzeugs, der Erdbeschleunigung (proportional zur Neigung des Flugzeugs) und des Luftwiderstands berechnet und in jeder Animationsschleife angepasst.

Die Ringe werden in einer Schleife als three.js Torus-Objekte erstellt und zufällig in der Szene platziert. Außerdem werden der Szene Hindernisse hinzugefügt, die das Flugzeug nicht berühren darf. Diese werden aus Dodekaeder, Icosahedron, Oktaeder und Tetraeder Elementen erstellt und ebenfalls zufällig in der Szene angeordnet. Da es sich bei den genannten Elementen um native three.js Elemente handelt, unterstützen diese (im Gegensatz zu den importierten Blender Modellen) die Verwendung des Phong-Beleuchtungsmodells. In Kombination mit der nativen three.js Lichtquelle, welche in der Flugsimulator-Szene zum Einsatz kommt, wird somit das Phong-Beleuchtungsmodell verwendet.

Um zu erkennen, ob das Flugzeug erfolgreich durch einen Ring geflogen ist oder mit einem Ring oder einem sonstigen Hindernis kollidiert ist, wird Kollisionserkennung benötigt. Dazu wird zunächst das Element mit dem geringsten Abstand zum Flugzeug ermittelt. Von diesem Objekt wird anschließend eine BoundingBox bzw. ein BoundingSphere erstellt und geprüft, ob diese die Flugzeugposition beinhaltet. Bei den Ringen wird zusätzlich auf den Abstand zum Mittelpunkt des Torus-Objekts geprüft und so determiniert, ob das Flugzeug den Ring durchfliegt, in den Rand des Rings fliegt oder außerhalb am Rand des Rings vorbeifliegt. Die Kollisionserkennung wird in jeder Animationsschleife aufgerufen und die Punktzahl entsprechend angepasst.

Bei einem Kollisionsereignis oder beim Ablauen der Zeit, wird die Flug-Animation unterbrochen, die aktuelle Punktezahl in einem „Game Over“-Overlay angezeigt und der User bekommt die Möglichkeit, das Spiel neu zu starten oder zurück zur Flugschule zu gelangen.

4.3 Architektur

Im folgenden wird die Architektur des Programmentwurfs in Form von Flowcharts beschrieben. Die Abbildung 4.1 beschreibt die Funktionsaufrufe des kompletten Programmentwurfs, die Gräfiken 4.2 und 4.3 beschreiben die Funktionsaufrufe im Klassenzimmer bzw. im Flugsimulator.

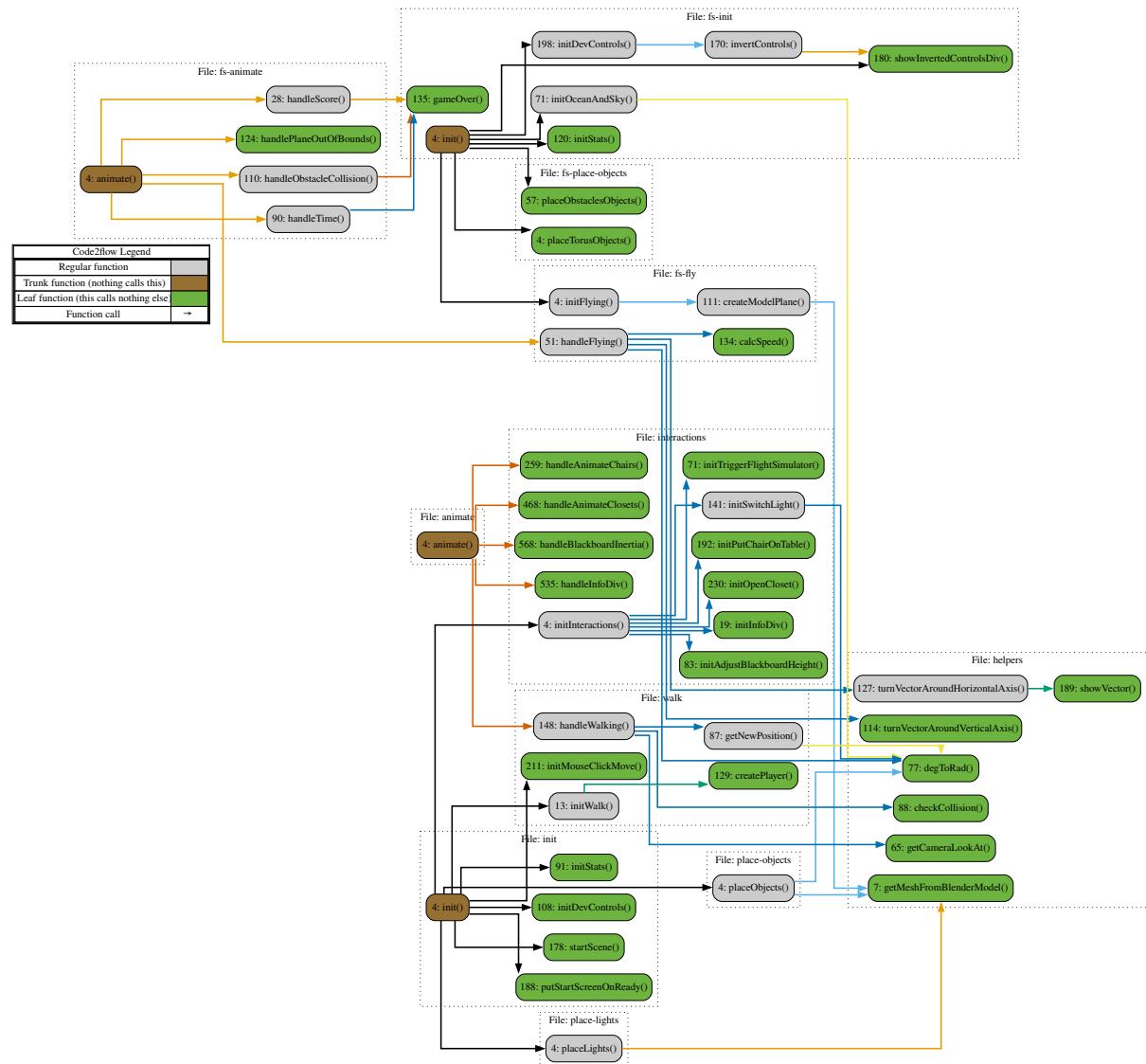


Abb. 4.1: Flowchart des kompletten Projekts

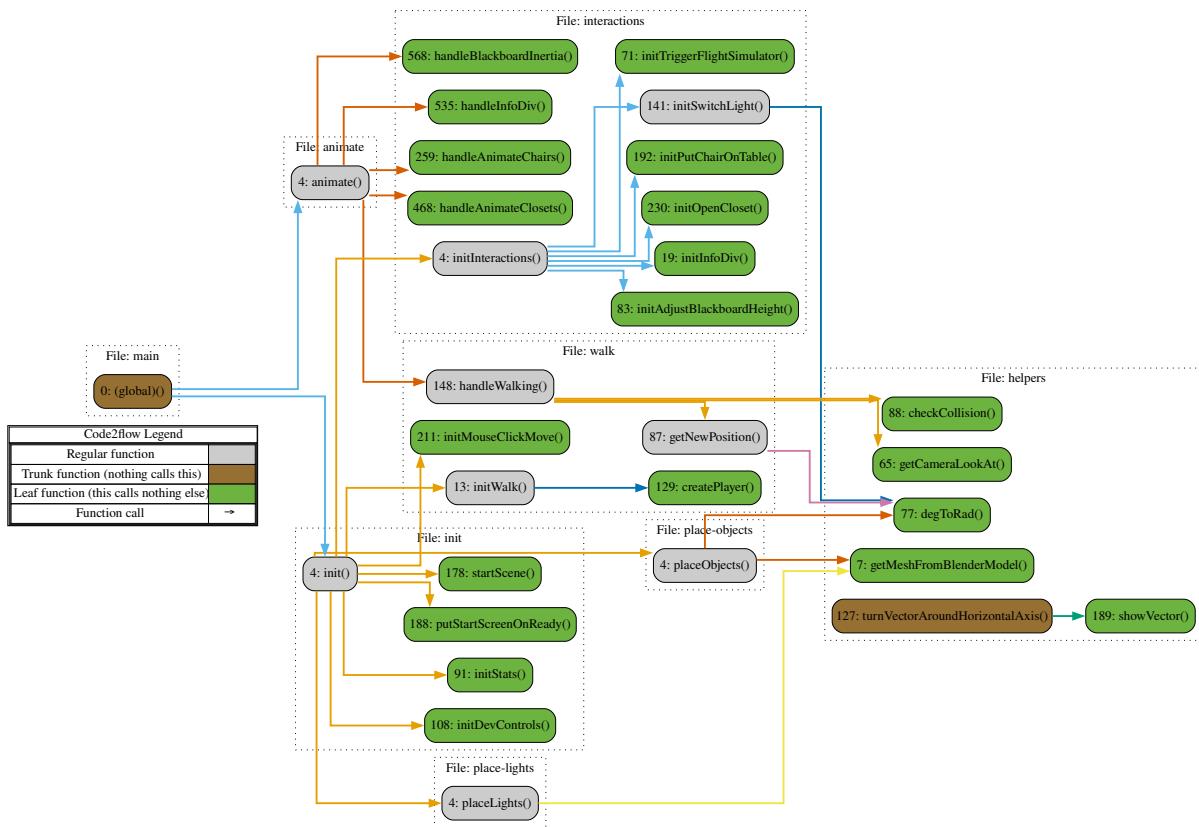


Abb. 4.2: Flowchart des Klassenzimmers

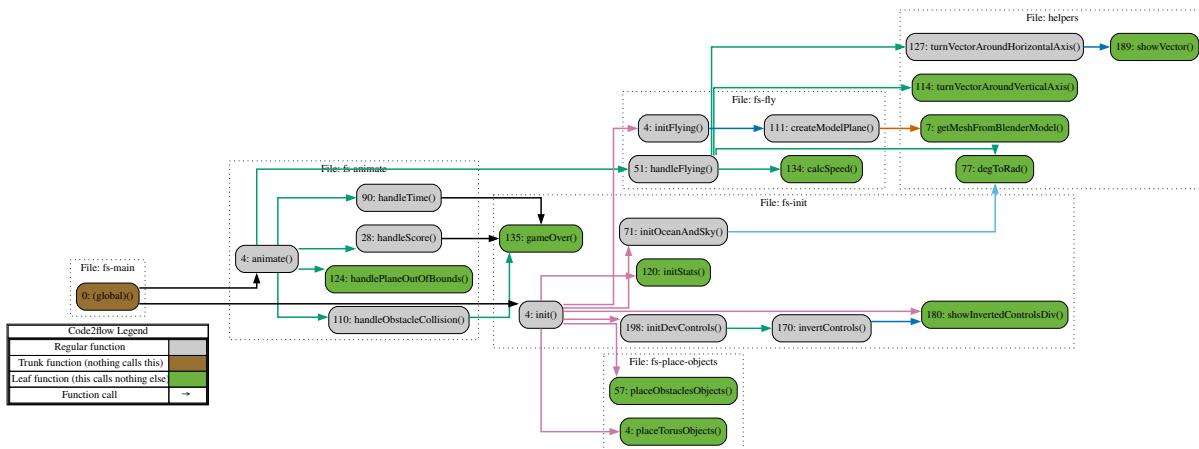


Abb. 4.3: Flowchart des Flugsimulators

5 Final Product

In diesem Kapitel wird das Ergebnis des Programmentwurfs durch mehrere Screenshots dargestellt.

5.1 Klassenzimmer



Abb. 5.1: Klassenzimmer alle Lampen



Abb. 5.2: Klassenzimmer nur ein Lampencluster



Abb. 5.3: Klassenzimmer ohne Schatten



Abb. 5.4: Klassenzimmer Aussicht

5.2 Flugsimulator

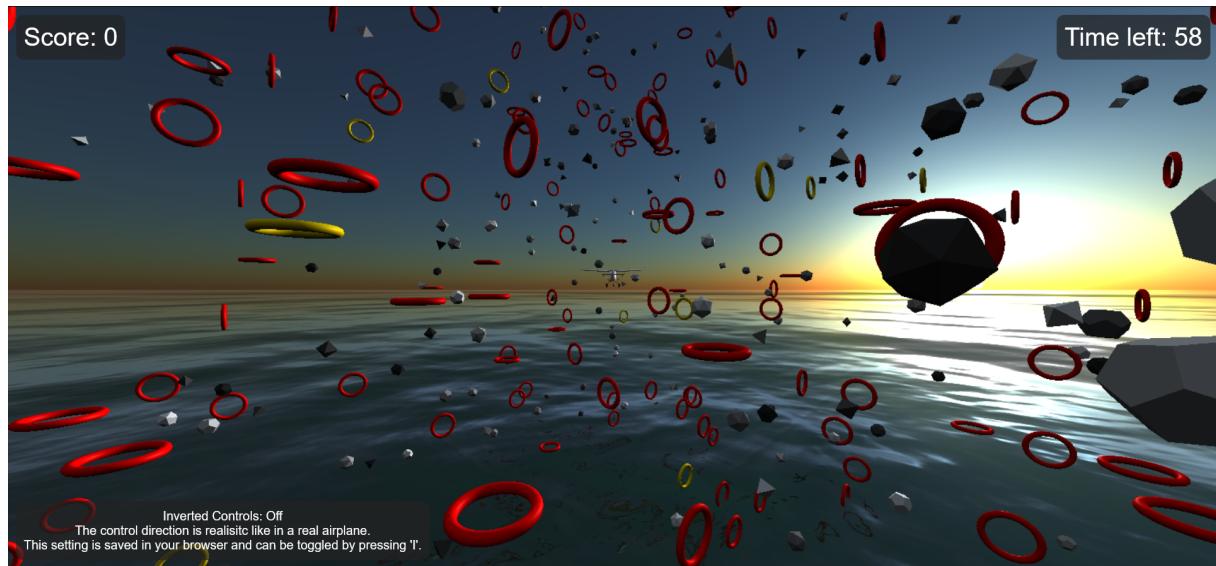


Abb. 5.5: Flugsimulator

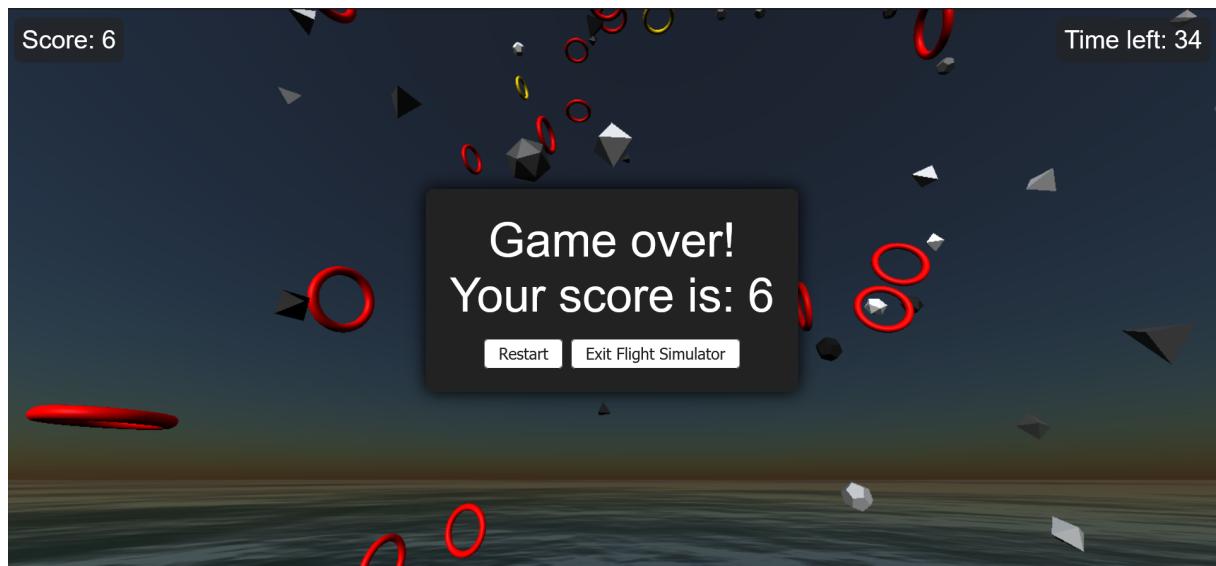


Abb. 5.6: Flugsimulator Game Over

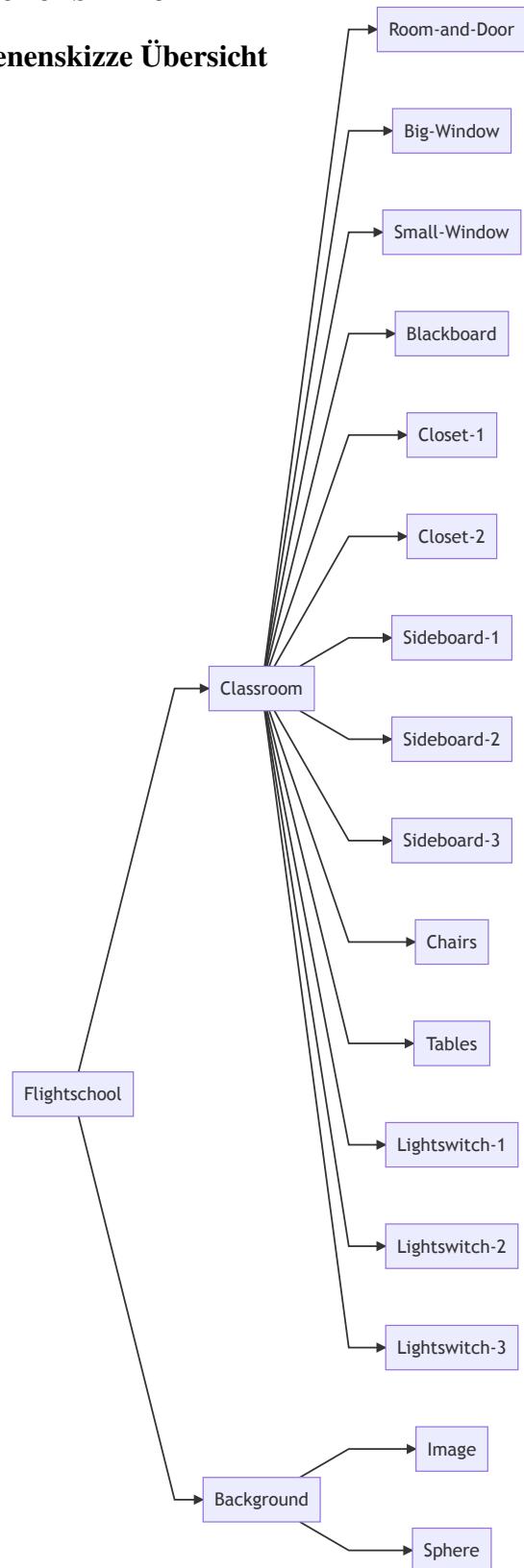
6 Installationsanleitung

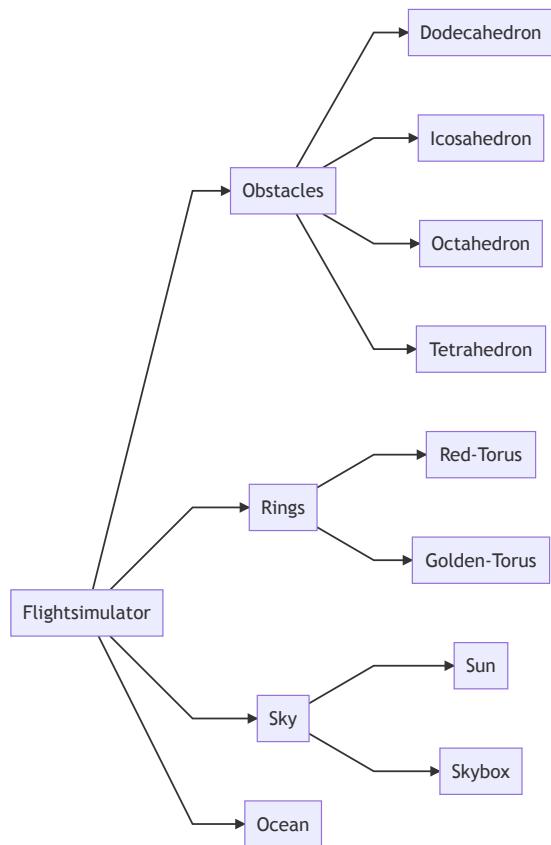
1. Wenn node und npm noch nicht installiert sind, installieren Sie diese von nodejs.org.
2. Führen Sie `npm run start` aus, um alle Abhängigkeiten zu installieren und den Webserver auf Port 3000 zu starten.
3. Öffnen Sie `http://localhost:3000` in Ihrem Browser (Chrome).

A Anhang

Szenenskizze

Szenenskizze Übersicht





Szenenskizze Detailliert