Technische Universität Braunschweig

ISF

Software-Entwicklungspraktikum

OfficeMania SEC0

Software-Entwicklungspraktikum (SEP)
Sommersemester 2021

Angebot

Auftraggeber
Technische Universität Braunschweig
Institut für Systemsicherheit
Prof. Dr. Konrad Rieck
Mühlenpfordtstraße 23
38106 Braunschweig

Betreuer: Erwin Quiring, Alexander Warnecke, Jonas Möller

Auftragnehmer:

Name	E-Mail-Adresse
Michael Goslar	m.goslar@tu-bs.de
Paul Hagedorn	paul.hagedorn@tu-bs.de
Johan Kolms	j.kolms@tu-bs.de
Eva Nortmann	e.nortmann@tu-bs.de
Joel Schaub	j.schaub@tu-bs.de
Han Thang	h.thang@tu-bs.de
Fabian Vizi	f.vizi@tu-bs.de
Lukas Wieland	l.wieland@tu-bs.de
Giulia Woywod	g.woywod@tu-bs.de

Braunschweig, 28. April 2021

Bearbeiterübersicht

Kapitel	Autoren	Kommentare
1	Wieland, Thang	...
1.1	Wieland, Schaub	...
1.2	Wieland, Schaub	...
2	Goslar	...
3	Woywod	...
3.1	Woywod	...
3.2	Woywod, Goslar	...
4	Nortmann	...
4.1	Nortmann	...
4.2	Nortmann	...
4.3	Nortmann	...
5	Kolms	...
5.1	Kolms	...
5.2	Kolms, Hagedorn	...
5.3	Kolms, Hagedorn	...
6	Vizi	...
6.1	Vizi	...
6.2	Vizi	...
6.3	Vizi	...
7	Thang	...

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Geboren aus den einzigartigen Umständen der globalen Pandemie COVID19 und der daraus folgenden Notwendigkeit zum Social Distancing, entsteht die Idee, dem Home-Office virtuell mit dem realen Büroleben so nahe wie möglich zu kommen. Wo sich bislang Video- und Soundübertragungen als ausreichend erwiesen haben, ist es die Intention dieses Projektes auch den Rest des Büros, in allen seinen Farben und Möglichkeiten, nach Hause zu holen.

1.1 Ziel

Konkret gilt es, vorerst die räumlichen Gegebenheiten des vorgegebenen Instituts im 2D Format virtuell wiederzugeben. Die Benutzer steuern simultan in dem erschaffenen, interaktiven Raum Charaktere, wobei Kommunikationsmittel, wie Videokonferenzen, existieren. Konzeptionell limitiert aber angepasst an die Bedingungen der Realität, werden zum Beispiel Benutzerinteraktionen, durch ausreichende Annährung ermöglicht. Neben Kommunikationsmitteln sollen dem Benutzer auch andere, an das Büroleben erinnernde, sowie surreale und kreative Interaktionsmöglichkeiten gegeben werden.

1.2 Motivation

Die Grundmotivation leitet sich aus der Aufgabenstellung ab, die das Softwareentwicklungspraktikum (SEP) der TU Braunschweig vorgibt. Sekundär existiert die Motivationsquelle der technischen, konzeptuellen und zwischenmenschlichen Auseinandersetzung mit den projektrelevanten Inhalten und Fertigkeiten, zur Förderung von Erfahrung und Kompetenzen. Auch ein direktes oder nebensächliches Interesse an Web- und Spieleentwicklung ist allgemein vertreten. Letztlich sehen sich die Teammitglieder gereizt, technisch und gedanklich kreative Additionen (Gimmmicks) zu dem Projekt beizutragen und zu implementieren.

2 Formale Grundlagen

Für das Projekt „OfficeMania“ wird TypeScript sowohl für den Server als auch für die Clients als Programmiersprache benutzt. Der Server, auf dem die Software implementiert wird, basiert auf Node.js. Damit TypeScript auf dem Server laufen kann, nutzen wir webpack zur Generierung der Client-Skripts. Um die Synchronisierung zwischen Client und Server zu realisieren, was bei „OfficeMania“ ausschlaggebend ist, benutzen wir das Colyseus-Framework. Das Institut für Systemsicherheit stellt uns für das Projekt eine Virtuelle Maschine auf einem Webserver zur Verfügung, auf dem die Software für den Kunden zugänglich gemacht wird. Über Clients können Nutzer in den Webbrowsern Google Chrome und Mozilla Firefox mit der Software interagieren und so direkt im Browser „OfficeMania“ nutzen, ohne die Software herunterladen zu müssen. Alle Abhängigkeiten werden über npm verwaltet. Ausgeliefert wird die Software in Deutsch und Englisch, wobei die Sprache in der Software umgestellt werden kann.

3 Projektablauf

Zur Organisation und Synchronisation zwischen Teammitgliedern orientieren wir uns am Scrum-Modell. Unser Team ist zwecks optimierter Aufteilung der vielfältigen Aufgaben in drei Gruppen eingeteilt: „Design“, „Movement“ und „Videochat“. Das Design-Team erarbeitet im Wesentlichen das Mapdesign sowie die Gestaltung des User Interface. Das Aufgabenfeld von der Gruppe „Movement“ beschränkt sich auf Benutzereingaben, Movement-Bedingungen und Interaktionen mit Objekten auf der Map. Das Team „Videochat“ hat die Aufgabe, Ton- und Bild-, sowie Chatfunktionen einzurichten. Der erfolgreiche Ablauf des Projekts erfordert eine konstante Absprache zwischen den drei Teams, auch um überschneidende Funktionen des Projekts gemeinsam implementieren und passende Schnittstellen für andere Teams zur Verfügung stellen zu können.

3.1 Meilensteine

Die Tabelle in Abb. 3.1 zeigt einen zeitlichen Ablauf mit wichtigen Meilensteinen. Dazu zählen sowohl Abgaben von Dokumenten an den Kunden, als auch projektinterne Ziele von großer Bedeutung. Zu nennen sind hier insbesondere das Zusammenführen von Videochat und Movement, also das Starten des Videos in Abhängigkeit von der Position des Charakters auf der Map und das Zusammenführen von Movement und Design, was eine Visualisierung der Spielerpositionen und Interaktion mit Objekten auf der Map bedeutet - beides wichtige Komponenten für ein spielerisches Erlebnis.

Nr.	Meilenstein	Dokumente	Abgabetermin
1	Projektstart	-	15.04.21
2	Angebot	Angebot	28.04.21
3	Abgabe Pflichtenheft und Abnahmetestspezifikation	Pflichtenheft, Abnahmespezifikation	19.05.21
4	grobes Mapdesign (Prototyp-Map mit Räumen und Objekten)	-	29.05.21
5	Videochat kann gestartet werden	-	29.05.21
6	Movement und Videochat Zusammenführung	-	03.06.21
7	Zwischenpräsentation	-	31.05 - 04.06.21
8	Abgabe Fachentwurf	Fachentwurf	09.06.21
9	Mapdesign fertig	-	09.06.21
10	Movement und Design Zusammenführung	-	14.06.21
11	Anwendung passt sich an Fenstergröße an	-	21.06.21
12	Notwenige Funktionen fertig	-	28.06.21
13	Abgabe Technischer Entwurf	Technischer Entwurf	30.06.21
14	Wünschenswerte Funktionen fertig	-	07.07.21
15	Abgabe Testdokumentation	Testprotokolle, Testspezifikation	14.07.21
16	TDSE	-	22.07.21

Abbildung 3.1: Meilensteine

3.2 Geplanter Ablauf

Das nachfolgende Gantt-Diagramm stellt die oben genannten Meilensteine graphisch inklusive der Bearbeitungszeiten dar. Es ist ersichtlich, dass an vielen Meilensteinen parallel gearbeitet wird und die Zeiträume nicht an den Rhythmus von Kalenderwochen gebunden sein müssen. Stattdessen wurde für jeden Punkt individuell abgeschätzt, wie viel Zeit benötigt und welches Team daran arbeiten wird. Nur so kann in der kurzen Zeit eine optimale Ausführung der vielfältigen Aufgaben gewährleistet werden.

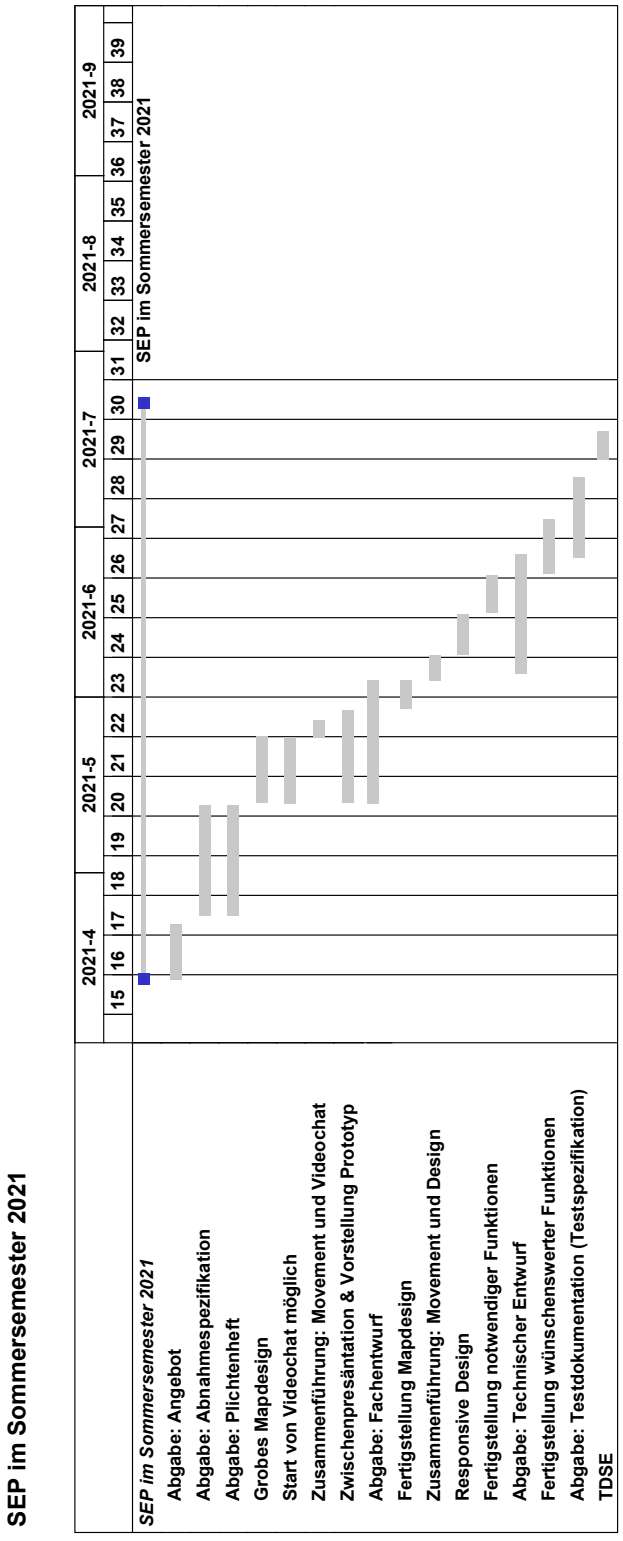

Abbildung 3.2: Gantt-Diagramm

4 Projektumfang

In diesem Kapitel dokumentieren wir den Umfang des Projekts bezüglich der Leistungen und den entsprechenden Kosten, um dem Kunden einen Überblick zu geben, was er von dem Projekt erwarten kann.

4.1 Lieferumfang

Ziel des Projekts ist es, dem Kunden eine funktionsfähige Anwendung inklusive des zugehörigen Quellcodes zu übergeben. Die Abgabe des Codes erfolgt als Git-Projekt. Der Kunde erhält dazu eine Installationsanleitung. Er kann so mit den Befehlen `npm install` und `npm start` selbst eine Session starten.
Außerdem wird dem Kunden zu Beginn des Projekts ein vollständiges Angebot überreicht. Diesem folgt nach Absprache mit ihm das Lasten- und Pflichtenheft inklusive einer Abnahmetestspezifikation. Außerdem wird dem Kunden nach der Hälfte der vorgesehenen Zeit ein Fachentwurf vorgelegt.
Während der Endphase wird dem Kunden ein Technischer Entwurf, welcher als Übersicht vor dem Projektende dient, übergeben. Anschließend erhält er noch eine Testdokumentation, um eine Zusammenfassung von getesteten Funktionen zu erhalten.
Außerdem werden wir dokumentieren, wie mögliche Erweiterungen miteinbezogen werden könnten, sodass das Projekt auch in der Zukunft weiterhin genutzt werden kann, sollte sich zum Beispiel an der Büroumgebung etwas ändern oder mit der Zeit weitere Features gewünscht werden.
Office Mania wird auf einer virtuellen Maschine laufen, zu welcher der Kunde einen Link erhält, um jederzeit Zugang zur aktuellsten Version zu erhalten.

4.2 Kostenplan

Für die Entwicklung von OfficeMania wurden vom Kunden 9 Entwickler angefordert, die jeweils 210 Stunden bis zum Abschluss des Projekts an Arbeitszeit zu vollbringen haben.

Bei einem Arbeitslohn von 100 Euro pro Stunde ergibt das einen Kostenaufwand von 21.000 Euro pro Entwickler, also einen Gesamtkostenaufwand von 189.000 Euro für das Projekt.

4.3 Funktionaler Umfang

Bei dem Projekt soll die natürliche Büroumgebung in Form einer virtuellen Umgebung wie in einem Computerspiel nachempfunden werden.
Dabei soll es möglich sein, das Büro inklusive angrenzender Räume und Flure auf einem Plan zu sehen und wie in der Wirklichkeit umherzulaufen und mit Kollegen interagieren zu können. Dafür soll bei ausreichender Annährung ein Videochat starten, sodass man die Kollegen live sehen und mit ihnen sprechen kann. Es existiert eine Stummschaltoption für andere Teilnehmer und sich selbst.
In einer Seitenleiste werden alle Teilnehmer, die zu dem Zeitpunkt online sind, angezeigt. Außerdem sieht man dort den eigenen selbst gewählten Namen, sowie einen Knopf zum Beenden der Sitzung.

5 Entwicklungsrichtlinien

In den Entwicklungsrichtlinien werden wichtige Eckpfeiler einer geregelten und übersichtlichen Entwicklung der Software festgelegt.

5.1 Konfigurationsmanagement

Der Code wird zentral in GitLab gespeichert. Das ermöglicht die simultane Einsicht und Bearbeitung des Quellcodes. Lädt jemand neuen oder aktualisierten Code hoch, so muss jener zur besseren Nachverfolgbarkeit und Übersichtlichkeit mit einer detaillierten Commit-Message beschrieben werden. Diese Nachricht muss zwecks Einheitlichkeit in englischer Sprache verfasst sein. Wichtig ist ebenfalls, dass zu keiner Zeit Code, der nicht funktioniert, in den Master-Branch hochgeladen wird oder anderen funktionierenden Code überschreibt.
Das GitLab wird außerdem als zentraler Ort zur Sicherung wichtiger Dateien genutzt. Dafür gibt es zwei separate Ordner für Abgaben, wie dieses Angebot, und Organisatorisches, wie eine Sammlung von Fragen an den Kunden.

5.2 Design- und Programmierrichtlinien

Im gesamten Projekt müssen alle Variablen und Funktionen auf Englisch benannt werden. Bestehen diese Namen aus mehreren zusammengesetzten Wörtern, so ist stets die CamelCase-Notation zu verwenden. Dabei ist darauf zu achten, dass die Namen zwar beschreibend, aber auch nicht zu lang sein dürfen. Ergänzend dazu müssen im gesamten Code sinnvolle Kommentare eingefügt werden, ebenfalls in englischer Sprache. Um bei Verständnisfragen zum Code schnell einen Ansprechpartner finden zu können, kann über die IDE, wie z.B. Visual Studio Code, der Autor des Codes herausgefunden werden.

5.3 Verwendete Software

Das Projekt nutzt sowohl bei Server und Client TypeScript als Programmiersprache. Teile des Codes können dabei sowohl vom Client als auch von Server genutzt werden. Die Synchronisation zwischen beiden nutzt Colyseus. Um die Anwendung zu testen, nutzen wir Jest. Der Server basiert auf dem Node.js Framework und Client Scripts werden mithilfe von Webpack generiert.

Als IDE werden von uns Visual Studio Code und IntelliJ IDEA benutzt, ersteres mit der Erweiterung git blame und beide mit einem beautifier für den Code. Für Grafiken und Diagramme wird die Website draw.io genutzt. Um gemeinsam an diesem und weiteren Dokumenten arbeiten zu können, nutzen wir sharelatex, ein Angebot der TU Braunschweig, welches auf Overleaf basiert.

6 Projektorganisation

In diesem Kapitel wird festgehalten, wie die Organisation und Kommunikation dieses Projektes ablaufen wird. Die Kommunikation ist ein fundamentaler Teil eines Projektes von dieser Größe, da sehr viele Entwickler an dem gleichen Thema arbeiten. Schnittstellen müssen intern und auch nach außen zum Kunden immer offen gehalten werden, damit eine reibungslose Organisation möglich ist. Bei einigen Kommunikationsschnittstellen ist es außerdem wichtig Zeitgrenzen festzulegen, andere sollen flexibel gehalten werden.

6.1 Schnittstelle zum Auftraggeber

Mit dem Auftraggeber wird bei organisatorischen, sowie thematischen Fragen und Problemen spontan per E-Mail kommuniziert. Andere Anliegen werden donnerstags bei wöchentlichen Scrum-Meetings angesprochen. Während dieser kann der Kunde ebenfalls Feedback geben und auch eventuelle thematische Missverständnisse ansprechen. Zusätzlich befindet sich im Git-Repository eine Fragen-Datei, in der die Gruppenmitglieder jederzeit Fragen hinzufügen können. Diese Verschriftlichung dient der Erinnerung, da die Datei jeden Donnerstag in der Anwesenheit des Kunden abgearbeitet wird.

6.2 Schnittstelle zu anderen Projekten

Bei diesem Projekt ist nicht vorgesehen, mit anderen Gruppen oder Projekten zusammenzuarbeiten. Deshalb ist vorerst keine Schnittstelle geplant. Sollte sich im Laufe der Entwicklung ergeben, mit der Gruppe SEC1 in Teilprojekten zusammenzuarbeiten, werden neue Schnittstellen erstellt.

6.3 Interne Kommunikation

Innerhalb der Gruppe läuft die Kommunikation sowohl regelmäßig als auch spontan ab. Interne Meetings werden über einen Discord-Server abgehalten. Zur Absprache dieser Meetings wurde eine WhatsApp-Gruppe erstellt. Geplant sind mindestens zwei wöchentliche Treffen, sodass sich die einzelnen Untergruppen (siehe Kapitel 3. Projektablauf) untereinander auf dem neuesten Stand halten. Neben diesen geplanten Treffen sind spontane Meetings sowie Programmier- und Test-Sessions angedacht.
Im Discord-Server besteht die Möglichkeit, Probleme in verschiedene Kanäle zu posten. So kann zum Besipiel jede Teilgruppe gruppeninterne Anliegen innerhalb ihrer Gruppe besprechen. Projektumfassende Probleme können in den Allgemeinen „Probleme“-Kanal gestellt werden. Damit sind die Anliegen differenziert, und jeder kann Kommentare und Lösungsvorschläge zu Problemen abgeben, auch wenn diese in einer anderen Gruppe aufgekommen sind.

7 Glossar

CamelCase-Notation: CamelCase ist eine Notationsart von Variablennamen, bei der man weder Leerzeichen noch Satzzeichen verwendet. Die Trennung von Wörtern wird durch die Großschreibung deren Anfangsbuchstaben verdeutlicht.

Client: Clients sind Programme oder Computer, die auf Server zugreifen können und von denen Daten oder Dienste abrufen.

Colyseus Framework: Das Colyseus Framework ist eine Sammlung von Funktionen, die der Entwicklung von Multiplayer-Anwendungen mit Node.js dient.

Commit-Message: Commit-Messages werden bei Git verwendet. Sie beschreiben vorgenommene Änderungen, die im Programm/Projekt durchgeführt wurden. Der Autor der Änderung gestaltet diese Nachrichten verständlich für andere Teammitglieder.

Discord: Discord ist eine Applikation für Instant Messaging und Sprach- bzw. Videokonferenzen.

Git: Git ist ein Open-Source-Tool zur verteilten Versionskontrolle von Software. Jedes Teammitglied hat jederzeit Zugriff auf die aktuellste Version vom Code.

IDE: IDE (Integrated Development Environment) ist eine Software, welche gängige Entwickler-Tools vereint, um Anwendungen zu programmieren.

Node.js: Node.js ist eine Plattform für die Entwicklung von JavaScript-Programmen.

SCRUM: SCRUM ist ein Vorgehensmodell für die agile Softwareentwicklung. Dabei liegt der Fokus auf regelmäßigen Treffen, intern und mit dem Kunden, bei denen Ziele für die Zeit bis zum nächsten Treffen vereinbart werden.

TypeScript: TypeScript ist eine Programmiersprache zur Entwicklung von webbasierten Anwendungen.