

SEA MORE: DIE ZERSTÖRUNG UNSERER MEERE

Eine interaktive Reise



Studiengang: Interaktive Mediensysteme
Vorlesung: Webtechnologien
Dozent: Fabian Ziegler
Semester: Sommer 2021
Hochschule Augsburg

Gruppenmitglieder:
Sarah Schneller
Veronika Remiger
Benjamin Adolph



Hochschule Augsburg
University of Applied Sciences

INHALTSVERZEICHNIS

KAPITEL	AUTOR	SEITE
1. THEMA & MOTIVATION	BENJAMIN	S. 3
2. PROJEKTMANAGEMENT	SARAH	S. 4
2.1. Aufgabenverteilung und Herangehensweise		S. 4
2.2. Methoden und Tools		S. 5
2.3. Timeline		S. 6
3. KONZEPT UND DESIGN	VERONIKA	S. 7
3.1. Zielgruppe		S. 7
3.2. Creative Challenge		S. 9
3.3. Storytelling		S. 9
3.4. Styleguide		S. 12
3.5. Umsetzung Design		S. 13
3.6. Designübergabe		S. 14
4. CONTENT	VERONIKA	S. 14
5. UMSETZUNG		S. 15
5.1. Backend	Benjamin	S. 15
5.2. Zusammenspiel Backend und Frontend	Sarah	S. 16
5.3. Frontend	Benjamin & Sarah	S. 16
5.4. Dev-Umgebung	Benjamin	S. 21
6. DATENSCHUTZ	SARAH	S. 22
7. HERAUSFORDERUNGEN	SARAH	S. 23
8. LEARNINGS	ALLE	S. 24
9. AUSBLICK	SARAH	S. 25
10. QUELLEN		S. 26

1. THEMA UND MOTIVATION (BENJAMIN)

“Am Umgang mit den Meeren entscheidet sich, ob es der Menschheit gelingt, dass die Erde auch in Zukunft lebenswert bleibt.” – Johan Rockström

Dieses Zitat vom Exekutivdirektor des Stockholm Resilience Centre (SRC), welches sich unter anderem mit dem Klimawandel und Meeressystemen beschäftigt, beschreibt das aktuelle Problem unserer Ozeane sehr gut. Leider bekommen meist nur die häufig thematisierten Probleme wie z. B. Plastik im Meer oder Überfischung Aufmerksamkeit. Es gibt jedoch noch viele weitere Probleme, welche oft in Vergessenheit geraten oder verdrängt werden. Mit unserer Webseite wollen wir diesem Trend entgegenwirken und einen Ort schaffen, an dem der Nutzer auf einen Großteil der Probleme in unseren Ozeanen aufmerksam gemacht wird.

Folgende Ziele wollen wir mit unserer Webseite erreichen:

1. Dem Nutzer einen Überblick über alle Probleme geben.
2. Dem Nutzer weitere Informationen zu diesen Problemen zur Verfügung stellen.
3. Den Nutzer dazu zu animieren, die Probleme durch sein eigenes Handeln zu bekämpfen.

Natürlich war es außerdem wichtig, uns von anderen Webseiten abzuheben und eine spannende Nutzererfahrung zu schaffen. Deshalb haben wir uns dazu entschlossen, das Thema mithilfe einer interaktiven Webseite, welche per Smartphone steuerbar ist, und Gamification aufzuarbeiten.



2. PROJEKTMANAGEMENT (SARAH)

2.1. AUFGABENVERTEILUNG UND HERANGEHENSWEISE

Sarah und Benjamin haben bereits einige Webprojekte umgesetzt. Dadurch gab es schon viele Erfahrungswerte, auf welche wir aufbauen konnten. Da beide sehr breit aufgestellt sind, war es zu Projektstart nicht notwendig eine strikte Aufgabentrennung zwischen Projektmanagement, Front- und Backend durchzuführen. Unsere ganze Gruppe hat sich gegenseitig sehr stark unterstützt und die Tickets wurden immer gleichmäßig verteilt. Im Verlauf des Projekts hat sich dadurch nebenstehende Aufteilung ergeben.

Zu Beginn haben wir unser Thema festgelegt und unsere Zielgruppe definiert. Anschließend haben wir eine Creative Challenge entwickelt und mit der Recherche für die Inhalte begonnen. Nachdem wir uns auf den Inhalt festgelegt haben, wurden die Texte, Bilder und Videos zusammengetragen. Auch für das Design wurden parallel schon Moodboards erstellt. In dieser ersten Phase haben wir vor allem mit Google Docs und Miro gearbeitet.

In einem eintägigen Workshop haben wir sowohl das Branding mit Namen, Logo und Design-Richtung als auch den grundlegenden Aufbau der Webseite erarbeitet.

Um Veronika beim Design zu unterstützen, haben Benjamin und Sarah gemeinsam an den Wireframes für die Infoseiten und das Menü gearbeitet.

Parallel zur Konzeption unserer Webseite haben wir mit der Umsetzung des Front- und Backends begonnen. Dieser Schritt war enorm wichtig, da bisher niemand aus unserer Gruppe Erfahrungen mit Socket.io oder PixiJS sammeln konnte. Durch die Erstellung eines technischen Prototypen während der Konzeption konnten wir sicherstellen, dass unser Konzept mit den gewählten Technologien umsetzbar ist. Im Anschluss konnten wir das Design relativ einfach auf unseren technischen Prototypen übertragen.

VERONIKA:

- Designsystem (Farben, Schriftarten, Gestaltung der Elemente)
- Screendesigns
- Erstellung kleinerer Animationen
- Content-Erstellung
- Storytelling

BENJAMIN:

- Design Wireframes Infoseiten
- Frontend PixiJS
- Verknüpfung Controller mit PixiJS (Backend & Frontend)
- Content-Erstellung

SARAH:

- Projektmanagement
- Aufbau Backend
- Design Wireframes Infoseiten
- Frontend Unterseiten & Menü

2.2. METHODEN UND TOOLS

PROJEKT-MANAGEMENT	DESIGN	ENTWICKLUNG BACKEND	ENTWICKLUNG FRONTEND
Jira Zoom Google Drive Slack Miro	Illustrator Figma	Editor: Visual Studio Code Javascript NodeJs express socket.io Webpack	Editor: Visual Studio Code Javascript PixieJS Sass HTML5 jQuery Gsap3

Tab. 1: Tools sortiert nach Aufgabenbereichen

Unser Weekly hatten wir immer Freitags über Zoom. Dabei haben wir eine Retrospektive der letzten Woche abgehalten, den aktuellen Stand des Projekts besprochen und die Aufgaben für den nächsten Sprint aus unserem Backlog verteilt.

Weitere Termine zur Konzepterstellung oder für Code-Reviews haben wir nach Bedarf ausgemacht. Meistens waren diese auch online, jedoch haben wir uns falls möglich in der Hochschule getroffen. Unsere Hauptkommunikationstools waren Zoom und Slack.

Unsere Designs haben wir in Figma erstellt. So konnte man jederzeit den aktuellen Stand einsehen und gemeinsam an Designs arbeiten und darüber diskutieren. Außerdem ermöglichte Figma es den Entwicklern sich bei Bedarf auch selbst Assets zu exportieren.

All unsere Aufgaben haben wir mithilfe von Tickets in Jira organisiert. Zur Verwaltung unseres Quellcodes haben wir das Hochschul-GitLab verwendet. Unsere Branches haben wir anhand der Tickets erstellt und diese nach Fertigstellung in den Master gemerged. Zu Beginn wollten wir nach Fertigstellung jedes Ticket an den anderen Entwickler übergeben. Dieser sollte dann ein Review des Tickets machen und es in den

Master mergen. Dieser Prozess ist jedoch sehr zeitaufwendig. Wir sind nach einiger Zeit dazu übergegangen, die Tickets nach einem kurzen Review selbst in den Master zu mergen. Durch eine enge Absprache und da für jedes Ticket ein eigener Branch erstellt wurde, kam es nur selten zu Konflikten.

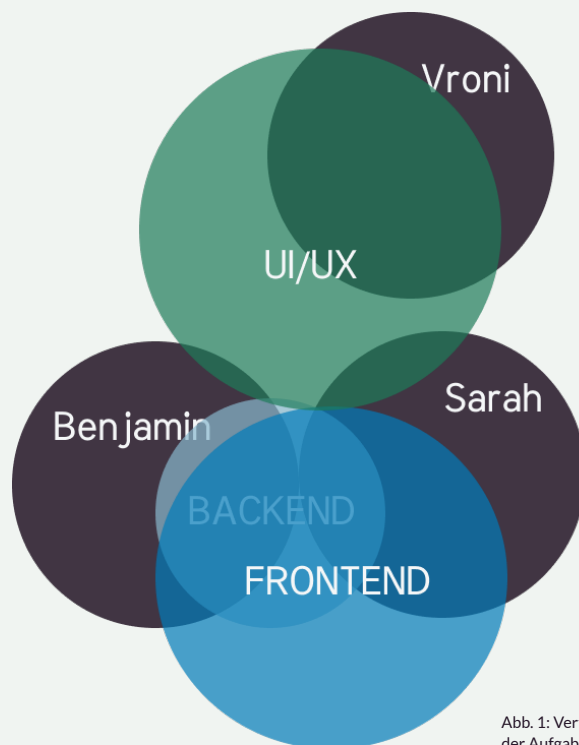


Abb. 1: Verteilung der Aufgaben

2.3. TIMELINE

Zu Beginn des Projekts haben wir Meilensteine festgelegt. Auf dieser Basis haben wir unser Semester geplant. Bis auf ein paar kleine Ausreißer, haben wir den Zeitplan sehr gut einhalten können. Falls notwendig, haben wir unseren Zeitplan an die aktuellen Gegebenheiten angepasst. So hatten wir bei unserem wöchentlichen Meeting zur Aufgabenverteilung immer eine gute Diskussionsgrundlage und konnten anhand des aktuellen Stands die wichtigen Aufgaben priorisieren.

Alles in allem, haben wir alle unsere Ziele erreicht und konnten sogar noch einige zusätzliche Funktionen einbauen.

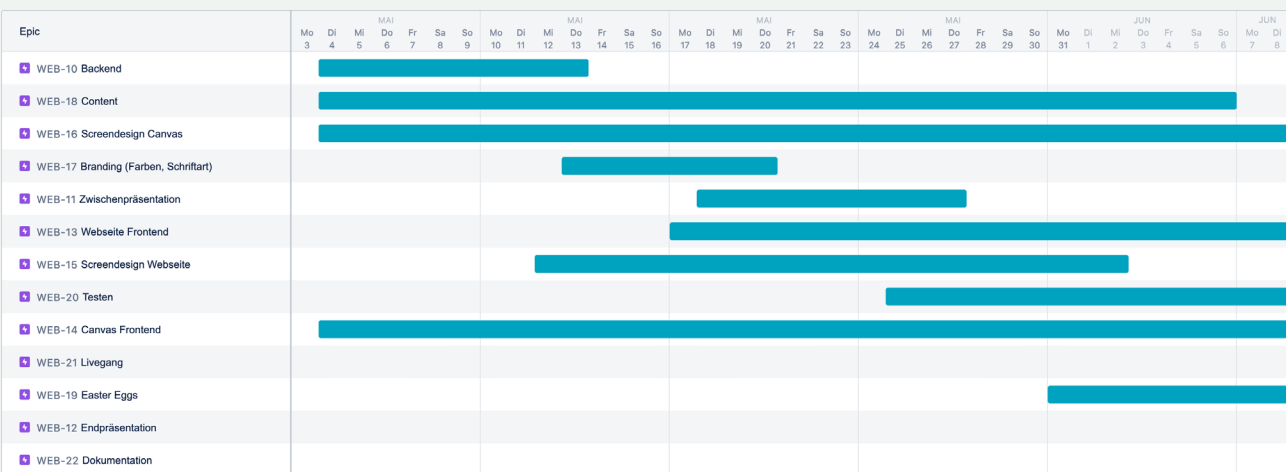


Abb. 2: Screenshot Jira Roadmap: Epics für die erste Semesterhälfte

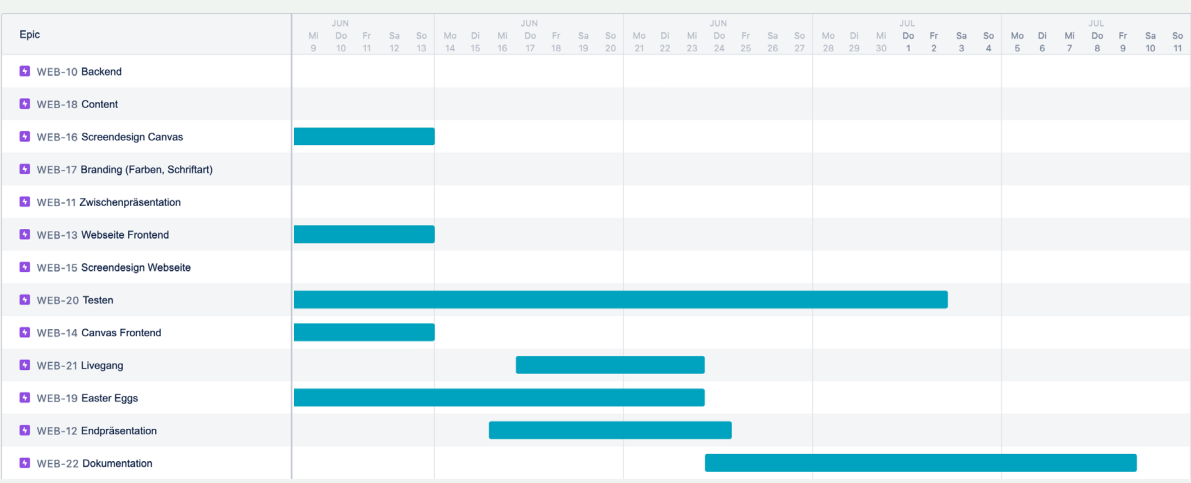


Abb. 3: Screenshot Jira Roadmap: Epics für die zweite Semesterhälfte

3. KONZEPT (VERONIKA)

Zu Beginn der Konzeption mussten wir uns für eine Design-Richtung entscheiden. Dazu war es wichtig, zu ermitteln, wen wir mit unserer Anwendung ansprechen wollen und welche Probleme es zu lösen gilt. Deshalb haben wir als erstes unsere Zielgruppe bestimmt und anschließend eine Creative Challenge entwickelt. Die Erkenntnisse haben wir in den Styleguide einfließen lassen. Da wir mit Storytelling unsere Webseite aufwerten wollten, war es wichtig, dies bei allen Design-Entscheidungen zu berücksichtigen.

3.1. ZIELGRUPPE

Zur Definition unserer Zielgruppe haben wir drei Personas erstellt: Olaf, Pia und Justin Aurelius. Unsere Personas sind mit einem Alter zwischen 12-30 Jahren noch relativ jung. Dies muss natürlich bei der Gestaltung und Umsetzung der Webseite berücksichtigt werden. Unsere Personas haben unterschiedliche Gründe, weshalb das Thema für sie von Interesse ist. Teilweise suchen sie nach einer guten Zusammenfassung oder Ideen zum Schutz der Meere. Manche Personas haben aber auch das Ziel, andere Menschen in ihrem Umfeld für den Schutz der Meere zu begeistern. Allerdings haben wir auch Personen, welche sich schnell von trockenen Themen schnell gelangweilt fühlen und sich noch keine Gedanken zum Schutz der Ozeane gemacht haben.



Olaf, 27, Erzieher

Olaf wohnt in einer WG in Berlin und ist Aktivist bei Fridays for Future. Er versucht sich hauptsächlich vegan zu ernähren und sein Lieblingsessen sind Falafel. Gelegentlich geht er mit Freunden containern, da er etwas gegen Lebensmittelverschwendung tun will.

Was er tun will:

- hat den inneren Drang etwas für den Schutz der Meere zu unternehmen
- andere Menschen inspirieren die Ozeane zu schützen

Was ihn hindert:

- er weiß nicht, wo er anfangen soll
- er hat keine konkrete Lösung
- er traut manchen Hilfsorganisationen nicht

Was er braucht:

- zuverlässige Quelle, die ihm hilft die Meere zu schützen
- Übersicht über alle Problemfelder der Zerstörung der Ozeane

Abb. 4: Persona Olaf



Pia, 21, Studentin

Pia ist vor kurzem für ihr Master-Studium nach Freiburg gezogen. Sie isst sehr gerne Spaghetti Bolognese und trainiert zwei mal in der Woche mit ihrer Handballmannschaft. Unter der Woche ist sie fleißig und lernt viel. Am Wochenende hat sie Handballspiele oder trifft sich mit Freunden. Außerdem schaut sie gerne Netflix. Dort ist sie auch auf die neue Dokumentation seaspiracy gestoßen.

Was sie tun will:

Sie will sich mehr über das Thema Fischfang informieren.

Was sie hindert:

- sie hat noch keine gute Wissensgrundlage
- viele Seiten im Internet bereiten die Infos nicht gut auf, weshalb sich Pia schnell langweilt.

Was sie braucht:

- eine Grundlage um mit ihren Freunden über das Thema diskutieren zu können
- Mehr Infos in Richtung seaspiracy

Abb. 5: Persona Pia

Was er tun will:

Er will eine gute Note in seinem Referat bekommen.

Was ihn hindert:

- er interessiert sich nicht besonders für das Thema Umweltschutz
- seine Freunde finden das Thema langweilig

Was er braucht:

Er braucht gute Quellen für sein Referat.



Justin-Aurelius, 15, Schüler

Justus Aurelius wohnt mit seinen Eltern in München und besucht eine Realschule. Am liebsten isst er Fischstäbchen mit Pommes. Für die Schule muss er ein Referat zu Thema Umweltschutz machen.

Abb. 6: Persona Justin-Aurelius

3.2. CREATIVE CHALLENGE

Die Webseite soll unsere Zielgruppe über die Zerstörung der Ozeane informieren und ihnen Lösungsvorschläge aufzeigen, wie sie die Meere aktiv schützen können. Außerdem wollen wir sie dazu ermutigen ihr Umfeld aufzuklären und zu inspirieren. Besonders wichtig ist uns dabei, dass unsere Zielgruppe einen guten Überblick über die wichtigsten Themen hat und nicht gelangweilt wird. Wir wollen durch Spaß Wissen vermitteln und dadurch den Lerneffekt verstärken. Basierend auf diesen Punkten haben wir unsere Creative Challenge formuliert:

“Wie können wir Olaf, Pia und Justin-Aurelius helfen
sich selbst über den Schutz der Meere zu informieren,
oder sie dazu inspirieren die Meere zu schützen,
ohne die Übersicht zu verlieren,
oder gelangweilt zu werden?”

3.3. STORYTELLING

Unsere Antwort auf die Creative Challenge lautet: Storytelling. Wir erzählen uns täglich Geschichten, um Erlebnisse wieder- und Wissen weiterzugeben. Mit einer guten Story können wir unser Gegenüber fesseln und Spannung erzeugen. Indem wir Storytelling in unsere Anwendung integrieren, schicken wir den Nutzer selbst auf eine Reise durch den Ozean. Dadurch kann er sich selbst das Wissen erarbeiten und hat Spaß beim Lernen. Folgende Schritte sind für das Storytelling sehr wichtig:

EMOTIONEN

Wir als Entwickler und Designer bestimmen die Art und Weise, wie wir die Botschaft an das Publikum weitergeben. Indem wir bestimmte Gefühle bei unseren Nutzern hervorrufen, können wir die Wirkung unserer Botschaft beeinflussen. Es ist sehr wichtig, dass unsere Zielgruppe nicht gelangweilt wird und Freude verspürt. Deshalb wollen wir mit unserer Anwendung eine Spannung erzeugen und Spaß vermitteln. ¹

DIE BOTSCHAFT

In jeder guten Geschichte gibt es eine Moral, die man vermitteln möchte. In unserem Fall ist das der Aufruf die Meere zu schützen und mit der Zerstörung der Ozeane aufzuhören. ²

DER HELD DER GESCHICHTE

Im klassischen Storytelling erlebt man die Geschichte aus der Sicht der Hauptperson. Dieser wird als Held bezeichnet und er vermittelt im Laufe der Geschichte die Moral. Der Held in unserer Anwendung ist der Nutzer selbst, in Form von einer Schildkröte (Abbildung 7). Er kann durch den Ozean schwimmen und dabei lernen und verstehen, warum es wichtig ist die Meere zu schützen.³



Abb. 7: Turtle

OPENING SCENE

Um die Aufmerksamkeit der Nutzer zu erhalten, braucht man einen starken Einstieg. Dieser soll dem Nutzer ein Problem vorstellen, welches er überwinden muss und ihn so ermutigen auf der Seite zu bleiben. Sobald der Nutzer unsere Webseite aufruft, bekommt er eine kurze Einleitung in das Thema. So erfährt er, um was es bei der Anwendung geht und welches Problem gelöst werden soll (Abbildung 8). Um die Aufmerksamkeit hoch zu halten, wird der Nutzer aufgefordert einen QR-Code zu scannen. Dieser ermöglicht es ihm per Smartphone durch den Ozean zu navigieren. Außerdem wird erklärt, wie er mit den Controller auf dem Smartphone zu bedienen hat (Abbildung 9).⁴

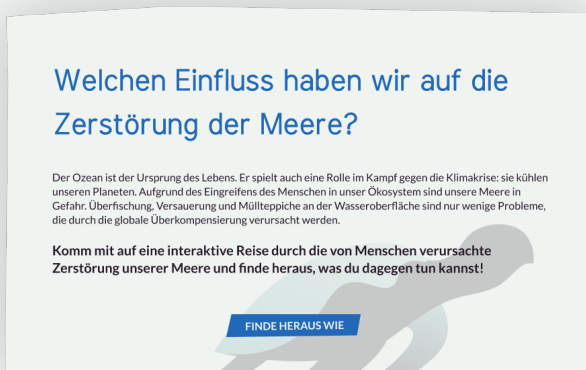


Abb. 8: Opening Scene



Abb. 9: Tutorial

STORY DEVELOPING

Nachdem der Nutzer mit dem Theam vertraut gemacht wurde, erhält er mehr Informationen in den Unterseiten unserer Anwendung. Zu jedem Thema auf unserem Canvas gibt es einen Info-Button, welcher auf die dazugehörige Unterseite mit weiterleitet.

Um die Aufmerksamkeit des Nutzers auch während der Bedienung der Seite aufrecht zu erhalten, haben wir eine Gamification eingebaut. Im Einführungstutorial bekommt er die Information, dass im Ozean fünf Münzen versteckt sind, die gesammelt werden müssen. Dadurch wird der Nutzer dazu animiert durch den Ozean zu schwimmen und mehr Themenbereiche zu entdecken. Hat man alle fünf Münzen gesammelt, bekommt man eine Belohnung. Man kann sich Visuals herunterladen,



die man auf Social Media teilen oder als Screensaver verwenden kann. Dadurch wollen wir es unserer Zielgruppe möglichst einfach machen, andere über den Schutz der Meere aufzuklären. Die Visuals könnten als Einstieg für ein Gespräch dienen oder durch den Link auf unsere Website führen.

Das Menü bietet dem Nutzer eine Übersicht über alle Themen auf unserer Webseite. Außerdem kann der Nutzer von dort direkt auf die jeweilige Unterseite navigieren.⁵

CALL TO ACTION

Nachdem der Nutzer Informationen über die Zerstörung der Meere erhalten hat, wollen wir ihm zeigen, was er selbst dagegen unternehmen kann. Dazu haben wir zwei Lösungen entwickelt: Auf jeder Infoseite geben wir Tipps, die man einfach im Alltag integrieren kann. Zudem haben wir eine "Unternimm was"-Seite gestaltet, auf der der Nutzer Links zu Kampagnen oder Spendenaktionen von Greenpeace oder WWF findet (Abbildung 9, 10 & 11). So kann er selbst aktiv werden.⁶



Abb. 10: Aktionsbutton Greenpeace



Abb. 11: Aktionsbutton WWF

Was kann ich unternehmen?

Wenn du Lebensmittel einkaufst, wähle Bio-Produkte, bei denen der Einsatz von nitrathaltigem Kunstdünger verboten ist.

Achte beim Waschen darauf, dass du die richtige Menge an Waschmittel verwendest. Beim Kauf solltest du auf phosphatfreie Waschmittel verzichten und ökologische Alternativen wählen. Oder mache dir dein Waschmittel einfach selber!

Auch für deinen Garten gibt es ökologische Alternativen. Verzichte auf künstlichen Dünger und verwende stattdessen Bio-Dünger. Außerdem kannst du umweltfreundlichen Dünger für Pflanzen ganz natürlich selber machen.

3.4. STYLEGUIDE

Unser Design muss zur Zielgruppe passen. Da wir sehr junge Menschen ansprechen wollen, haben wir die Formsprache modern und verspielt gehalten. Wir verwenden viele Unregelmäßigkeiten, abstrakte Formen und teils unrealistische Proportionen. Die Hauptfarben Grün, Blau und Violett sind an die Farbpalette unseres Ozeans angelehnt. Wir haben uns für zwei Schriften entschieden. Die Schrift Meera Inimai dient uns als Schmuckschrift. Sie passt sehr gut zu dem unregelmäßigen, verspielten Gesamteindruck, den die Webseite vermittelt. Für Fließtext verwenden wir die Schrift Lato. Sie passt gut zu unserer Schmuckschrift und ist besonders auf Bildschirmen gut lesbar.

LATO

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum.

ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890'?"!")(%)[#] {@}/&\<-++÷×=>®©\$€£¥¢;,:,*

MEERA INIMAI

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum.

ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890'?"!")(%)[#] {@}/&\<-++÷×=>®©\$€£¥¢;,:,*

Abb. 12: Schriftarten

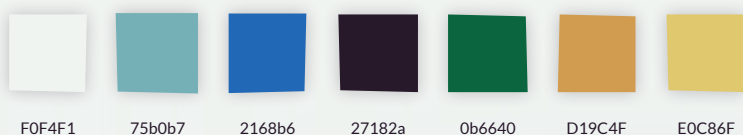


Abb. 13: Farben

Unternimm etwas

Abb. 14: Button



Abb. 15: Formensprache

3.5. UMSETZUNG DESIGN

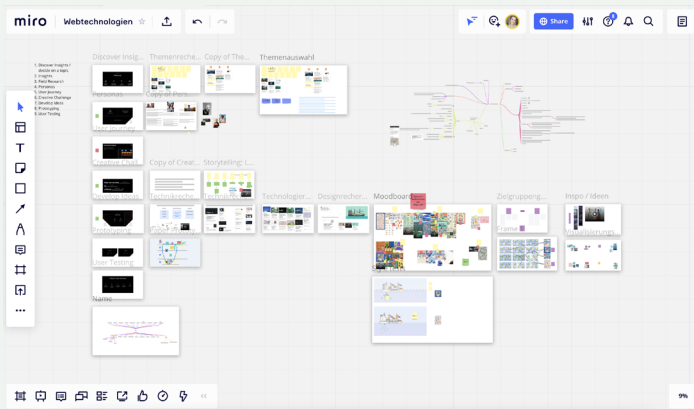


Abb. 16: Miroboard Kick-Off Workshop



Abb. 17: Canvas

Zu Beginn haben wir all unsere Ideen in Miro zusammengetragen (Abbildung 16). So konnten wir gemeinsam an unseren Personas arbeiten, recherchierte Daten zusammenfügen und die erste Anordnung von unseren Themen auf dem Canvas testen.

Für die Erstellung der Grafiken haben wir Illustrator verwendet. Dadurch konnten wir alle Elemente auf einer Zeichenfläche anordnen und sehen, ob sie stimmig zueinander wirken. Dabei haben wir uns als erstes auf unsere Hauptelemente, über welche wir den Nutzer aufklären wollen, fokussiert. Beispielsweise das Fischernetz, welches auf das Thema Überfischung aufmerksam machen soll (Abbildung 17).

Danach haben wir Dekoelemente gestaltet, welche den Canvas füllen und dem Nutzer Orientierung bieten sollen. Für die Übergabe an die Entwickler wurde alle Elemente logisch gruppiert und als SVG exportiert.

Die Unterseiten haben wir in Figma gestaltet. So konnten wir sehr einfach einen Prototypen für unsere Webseite erstellen. Hier haben wir auch verschiedene Varianten für die Anordnung von Elementen auf der Landingpage getestet und die Visuals zum Download gestaltet (Abbildung 18).

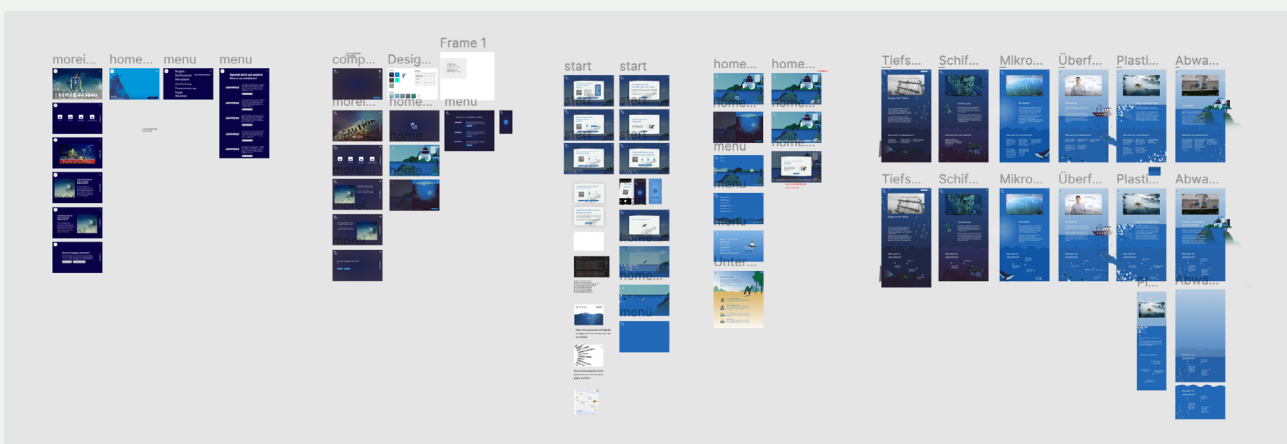


Abb. 18: Figma-board mit den Screendesigns der Webseite

3.6. DESIGN ÜBERGABE

Um die Zusammenarbeit zwischen Design und Entwicklung möglichst effektiv zu gestalten haben wir einige Punkte beachtet:

1. Dateien wurden gleich von Anfang an einheitlich und eindeutig benannt. Dafür haben wir ausschließlich Kleinbuchstaben verwendet und Wörter mit Bindestrichen voneinander getrennt (z. B. superturtle-down-right).
2. Beim Export von SVGs für den Canvas war es wichtig, dass diese für das Web optimiert und mit Höhen- und Breitenangabe exportiert wurden.
3. Um Zeit zu sparen hat der Designer auch direkt die Positionen in X- und Y-Koordinaten für die Platzierung auf dem Canvas mit angegeben.
4. Für den Austausch von Daten haben wir Google-Drive verwendet.
5. Durch ein gemeinsames Figma-Board konnten die Entwickler bei Bedarf auch Grafiken selbst exportieren.



4. CONTENT (VERONIKA)

Viele Webseiten beschäftigen sich immer nur mit einem Thema aus dem Bereich Meeresschutz. Unser Ziel war es eine gute Übersicht über den Großteil der Themen zu geben. Dabei haben wir eine Auswahl der relevantesten Themen zusammengestellt und sie gleichmäßig auf unserem Canvas verteilt. Unsere Themen sind:

1. Müll (Plastik, Fischernetze)
2. Überfischung
3. Abwasser und Überdüngung
4. Mikroplastik
5. Bergbau in der Tiefsee
6. Erdöl (Schiffswracks, Ölplattformen)



5. UMSETZUNG

Im Folgenden werden wir genauer auf die im Front- und Backend verwendeten Technologien und deren Zusammenspiel eingehen.

5.1. BACKEND (BENJAMIN)



Abb. 19: Node.js

NODE.JS

Für den Server fiel die Entscheidung schnell auf Node.js, da wir so im gesamten Projekt dieselbe Programmiersprache (JavaScript) nutzen konnten. Auch die Vielzahl an Erweiterungen, eine sehr gute Dokumentation und eine große Community sprachen für die Verwendung.

Node.js ist eine JavaScript-Laufzeitumgebung und ermöglicht die serverseitige Programmierung in JavaScript. Sie wird in unserem Fall im Backend zur Erstellung eines Webserver genutzt. Die Installation von Node.js beinhaltet den Paketmanager npm. Dieser erlaubt das einfache Hinzufügen von JavaScript-Modulen zum eigenen Projekt. Unter <https://www.npmjs.com/> findet man eine Bibliothek mit einer großen Auswahl an Modulen, welche unterschiedlichste Funktionalitäten anbieten.^{7,8}

Zur Erstellung unseres Webserver haben wir Express.js genutzt. Express ist ein Webanwendungsframework, welches in Kombination mit Node.js meist als Serverframework zur Erstellung von Webanwendungen genutzt wird. Zusätzlich ermöglicht Express.js den Einsatz von sogenannten Template Engines, welche Variablen in statischen Files zur Laufzeit durch den entsprechenden Code ersetzen und daraus ein HTML-File erstellen können. In unserem Fall haben wir dafür EJS benutzt, worauf im Abschnitt "Frontend" aber noch genauer eingegangen wird.^{9,10}



Abb. 20: Logo Socket.io

SOCKET.IO

Um die Steuerung der Webseite per Smartphone zu ermöglichen, haben wir Socket.io verwendet. Socket.io ist eine JavaScript-Bibliothek die eine bidirektionale, eventbasierte Echtzeitverbindung zwischen unserer Anwendung und unserem Node.js-Server ermöglicht. Socket.io basiert dabei auf dem WebSocket-Protokoll, welches für die bidirektionale Verbindung zwischen Anwendung und Server entworfen wurde.¹¹

Socket.io bringt ein wichtiges Feature mit, welches von Websockets nicht geboten wird und das sind sogenannte Rooms. Sie ermöglichen die Erstellung von separate Kanälen auf dem Server, welche von Sockets betreten und verlassen werden können. Dies ist in unserem Fall wichtig, damit jeder Nutzer seine "eigene" Webseite steuern kann, wenn mehrere Leute auf den Server zugreifen. Weitere Infos zu unserer Architektur werden im nächsten Kapitel erläutert.^{12,13}

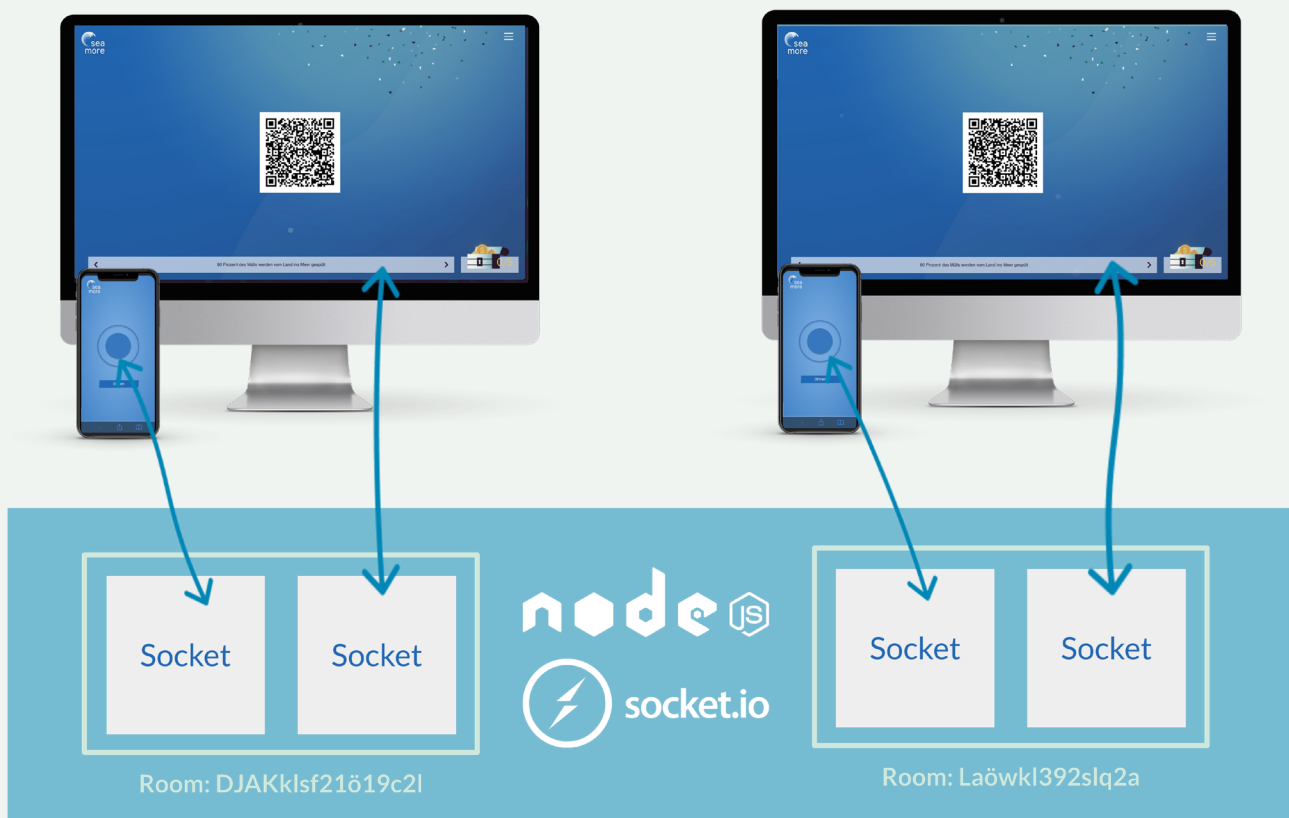


Abb. 21: Zusammenspiel Backend und Frontend

5.2. ZUSAMMENSPIEL BACKEND UND FRONTEND (SARAH)

Bei jedem Aufruf der Webseite wird eine Verbindung mit dem Server hergestellt. Dieser erstellt mit Hilfe von Socket.io einen Socket. Dieser wird dann automatisch einem Raum zugewiesen. Verbindet sich der Nutzer nun durch Scannen des QR-Codes mit dem Smartphone, wird ein neuer Socket in demselben Raum erstellt (Abbildung 21). Der QR-Code wird automatisch aus der URL der Webseite und einem zufälligen, 16-stelligen String generiert. Dies stellt sicher, dass nur ein Nutzer die Webseite steuern kann. Der Controller kann nun mit der Webseite interagieren und unsere Schildkröte steuern.

Ein weiterer Vorteil ist, dass die Webseite so von mehreren Nutzern gleichzeitig verwendet werden kann. Da für jeden Nutzer neue Sockets angelegt werden, können sie sich nicht in die Quere kommen. Wenn natürlich zwei Smartphones den gleichen QR-Code scannen, können beide Handys auch die gleiche Webseite steuern.

5.3. FRONTEND (BENJAMIN UND SARAH)

Das Frontend teilt sich in unserem Fall in zwei Bereiche auf. Wir haben unseren interaktiven Canvas, welcher mit PixiJS erstellt wurde und die Unterseiten, auf denen weitere Informationen zu den verschiedenen Themen gegeben werden.

SEITENAUFBAU (BENJAMIN)

Unsere Webseite baut sich um den PixiJS Canvas (Abbildung 22). Beim Aufrufen der Seite wird der Nutzer durch ein kurzes Tutorial geleitet. Dieses erklärt ihm das Ziel der Webseite, die Schatzsuche und die Steuerung der Webseite mit dem Controller. Sowohl vom Canvas als auch über das Menü können die Infoseiten zu den verschiedenen Themen erreicht werden. Über das Menü erhält der Nutzer zusätzlich die Möglichkeit das Impressum, die Datenschutzerklärung und die Unternimm Etwas-Seite aufzurufen. Außerdem öffnet sich ein Dialog, wenn der Nutzer die Schatzsuche erfolgreich abgeschlossen hat.

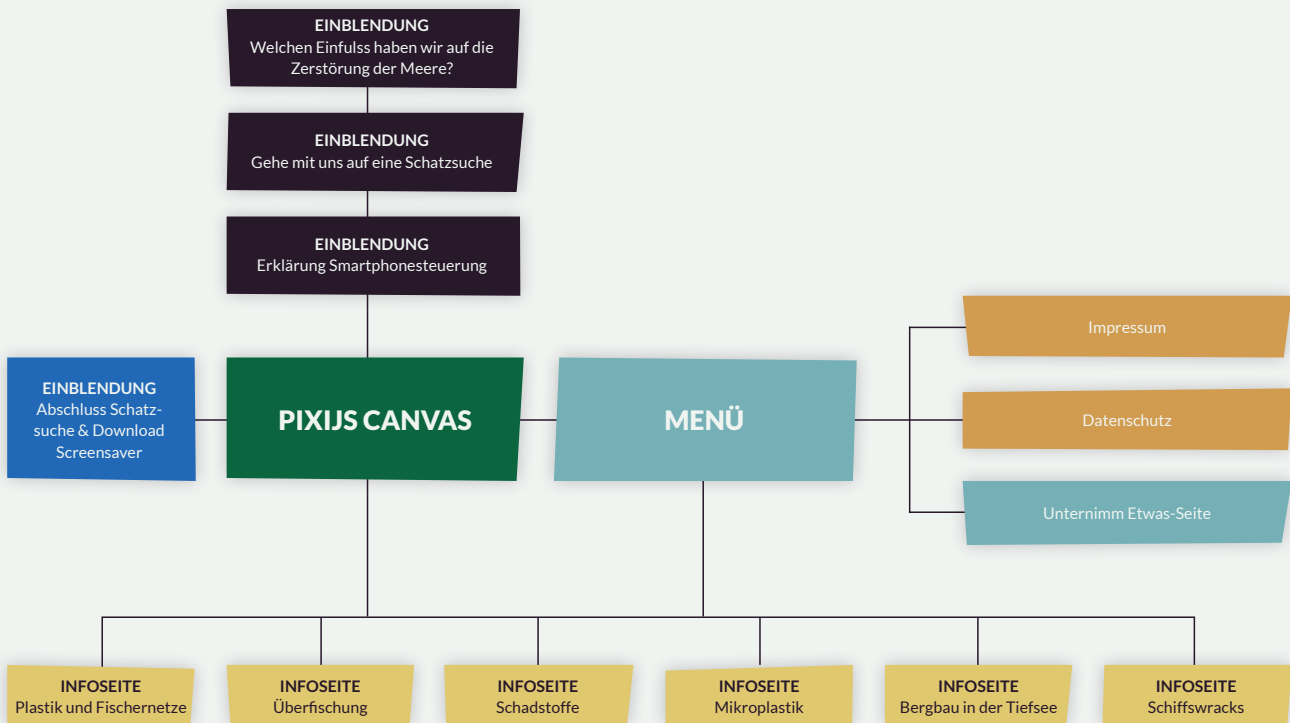


Abb. 22: Seitenaufbau

CANVAS MIT PIXIJS (BENJAMIN)

Nach der Festlegung unseres Themas, haben wir angefangen nach geeigneten Technologien zu recherchieren, mit denen wir unser interaktives Frontend umsetzen können. Dabei sind wir relativ schnell auf PixiJS gestoßen. Das ist eine 2D JavaScript-Rendering Bibliothek, welche auf WebGL basiert. Sie ermöglicht das Erstellen von plattformübergreifenden, interaktiven Apps und Spielen - ohne dabei zu tief in WebGL eintauchen zu müssen. Zu den Vorteilen gehören z. B. eine umfassende Geräte- und Browser-

kompatibilität und sehr gute Performance. Außerdem bietet PixiJS einen eingebauten Canvas-Fallback, falls WebGL nicht unterstützt wird.¹⁴

In unserem Fall verwenden wir PixiJS dazu, eine interaktive, selbst gestaltete 2D-Karte des Ozeans abzubilden. Der Canvas besteht aus einem großen Hintergrund und weiteren Grafiken, welche einzeln darauf platziert werden. Manche Browser, wie z. B. Firefox, haben eine maximale Größe von Texturen,

welche mit WebGL geladen werden können. Deshalb setzt sich unser Hintergrund aus vier Teilbildern zusammen. Auch das Spritesheet für unsere Schildkröte mussten wir aus diesem Grund in mehrere Bilder unterteilen. Alle anderen Objekte auf unserem Canvas werden als SVGs geladen. Die Größe wird erst beim Rendern des Bildes auf unserem Canvas festgelegt. Somit waren wir bei Größenänderungen von Objekten sehr flexibel und mussten nur die Größe im Quellcode anpassen. Das hat uns viel Arbeit erspart, da wir so die Bilder nicht bei jeder Änderung neu exportieren mussten.

Durch Click & Drag oder per Smartphone-Controller ist der Nutzer in der Lage sich auf der Karte umher zu bewegen. Auf die Umsetzung der Steuerung per Smartphone wird im nächsten Abschnitt genauer eingegangen.

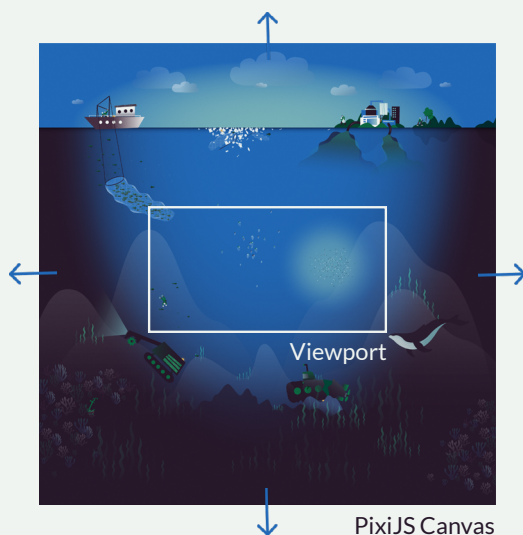


Abb. 23: Navigation mit Viewport



Abb. 24: Turtle Sprites

Pixi Viewport:

Das PixiJS-Plugin Pixi Viewport ist für unser Projekt sehr wichtig gewesen. Die Bibliothek ermöglicht es eine 2D-Kamera zu erzeugen, welche es dem Nutzer erlaubt, sich auf dem Canvas umher zu bewegen. Der Viewport passt sich dabei der Größe des Bildschirms an, während der Canvas eine feste Größe von 8090 x 8127 Pixeln hat (Abbildung 23). Von Haus aus bringt das Plugin verschiedene Funktionalitäten zur Navigation auf dem Canvas mit. In unserem Fall nutzen wir das Plugin für die Click and Drag-Funktionalität zur Navigation und um den Canvas auf den Bereich mit Inhalt zu begrenzen, sodass der Nutzer nicht unendlich weiter navigieren kann. Außerdem ist es möglich den Viewport anhand von X- und Y-Koordinaten zu verschieben. So konnten wir auch die Steuerung per Smartphone ohne große Probleme implementieren.

PixiJS Animationen:

Animationen sind bei unserem Projekt sehr wichtig. Denn nur durch sie wirkt unser Canvas lebendig und gibt dem Nutzer das Gefühl wirklich durch den Ozean zu schwimmen.

Eines der Highlights ist die Animation der Schildkröte, welche sich je nach Schwimmrichtung und -geschwindigkeit entsprechend bewegt. Um diese Animation zu ermöglichen haben wir Spritesheets (Abbildung 24) verwendet. PixiJS stellt für die Erstellung von Animationen mit Spritesheets eine Klasse zur Verfügung. Diese ermöglicht es aus einem oder mehreren Bildern einen Array mit Texturen zu erstellen, welche dann schnell hintereinander abgespielt werden, um so eine Bewegung zu simulieren. In unserem Fall haben wir zwei Arrays mit unterschiedlichen Texturen, zwischen welchen nahtlos gewechselt werden kann. Dies nutzen wir, um die zwischen der schwimmenden Schildkröte und der fliegenden Schildkröte zu wechseln. Der Wechsel findet immer dann statt, wenn sich die Schildkröte aus dem Ozean heraus bewegt. Dies wird anhand der Y-Koordinaten der Schildkröte überprüft.

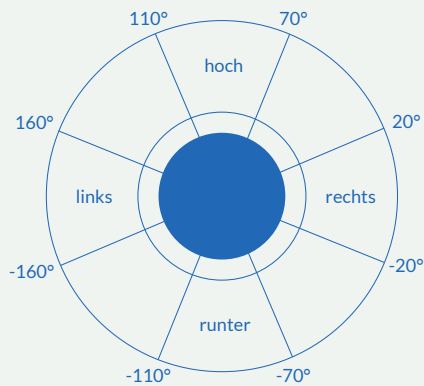


Abb. 25: Animation Schildkröte

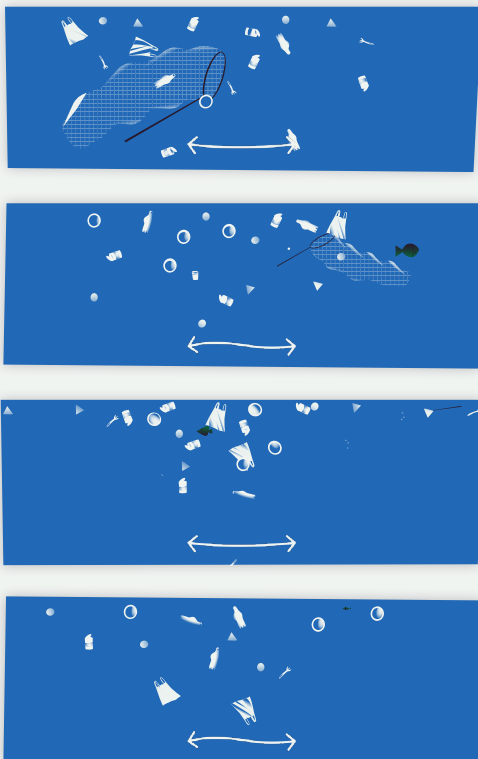


Abb. 26: Animation Müllteppich

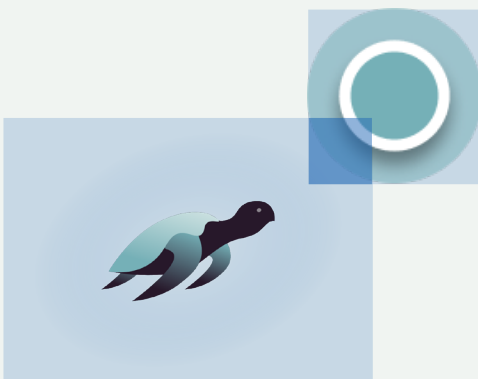


Abb. 27: Interaktion durch HIT-Testing

Damit für jede Bewegungsrichtung die richtigen Spritesheets angezeigt werden, haben wir mithilfe der $\arctan2$ -Funktion und den X- und Y-Werten des Joysticks den Winkel des Vektors berechnet (Abbildung 25). So können wir die Spritesheets immer an die Schwimm- oder Flugrichtung der Schildkröte anpassen. Außerdem ist die Geschwindigkeit der Animation der Schildkröte und der Positionierung des Viewports abhängig von der Ziehstärke des Joysticks. Desto stärker gezogen wird, desto schneller läuft die Animation der Schildkröte ab und der Viewport bewegt sich schneller.

Die meisten anderen Animationen haben wir erstellt, indem wir z. B. den Müllteppich in verschiedene Ebenen aufgeteilt und diese mithilfe von Sinus und Cosinus-Wellen animiert haben (Abbildung 26). So sieht es für den Nutzer aus, als ob der Müllteppich auf dem Meer treibt.

Für die Animation der aufsteigenden Luftblasen haben wir die Bibliothek Pixi Particles verwendet. Sie ermöglicht das einfache Erstellen und Animieren von Partikeln in PixiJS. Außerdem stellen die Entwickler einen interaktiven Editor zur Verfügung, welcher es sehr einfach gestaltet, benutzerdefinierte Partikel-Emitter für sein eigenes Projekt zu entwerfen.

PixiJS Interaktion:

Um dem Nutzer die Interaktion mit Objekten auf dem Canvas zu ermöglichen bietet PixiJS standardmäßig die Möglichkeit diese klickbar zu machen und auf verschiedene Events zu reagieren. So haben wir auch die Funktionalität auf dem Desktop umgesetzt. Wenn der Nutzer jedoch die Steuerung per Smartphone Controller verwendet, kann diese Funktionalität nicht genutzt werden. Hier haben wir das sogenannte Hit-Testing verwendet. Dabei wird geprüft, ob sich die Positionen bestimmter Objekte auf dem Canvas überlappen (Abbildung 27). In unserem Fall prüfen wir, ob die Schildkröte mit einem der Infopage-Buttons oder einem Coin überlappt. Wenn dies der Fall ist, kann der Nutzer per Button mit dem Canvas interagieren. Jenachdem über welchem Objekt er sich befindet, kann der Button auf dem Smartphone Controller anders benutzt werden. So kann man mit ihm z. B. Infopages öffnen und schließen oder Coins sammeln.

UNTERSEITEN UND MENÜ (SARAH)

Zu Beginn des Kapitels wurde bereits genauer auf den Seitenaufbau eingegangen. Neben dem Canvas mit PixiJS haben wir noch die Unterseiten und das Menü, welche sich als HTML-Container über den Canvas legen. Dies ist sehr wichtig, da so beim Schließen der Unterseiten oder des Menüs verhindert wird, dass die Seite neu laden muss.

Zur Umsetzung des Frontends haben wir HTML5, Sass, JavaScript und JQuery verwendet. Wir haben uns für eine Kombination aus JavaScript und jQuery entschieden, da die DOM-Manipulation mit JQuery deutlich einfacher ist.

Für die Animationen im Menü und auf den Unterseiten verwenden wir die Bibliothek GSAP3 von Greensock. Die Vorteile GSAP3 gegenüber anderen Animationsbibliotheken sind eine sehr gute Performance und hohe Browserkompatibilität.

TEMPLATE ENGINE (SARAH)

Als Template Engine verwenden wir EJS. Damit können wir wiederkehrende Elemente, wie z. B. den Header, in einzelne Dateien auslagern und diese importieren. Dies hat den Vorteil, dass Änderungen nur in einer Datei durchgeführt werden müssen und sorgt bei unserem One-Pager für eine übersichtliche Code-Struktur. Außerdem werden die Inhalte erst beim Aufruf der entsprechenden Seite geladen. Dies verkürzt die Ladezeit beim ersten Aufruf der Webseite merklich. Ein weiterer Vorteil von EJS ist, dass beim einbinden eines Skripts Variablen übergeben werden können. So benötigt es z. B. für einen Button nur ein Skript, welches dann mit unterschiedlichen Werten eingebunden werden kann (Abbildung 28).

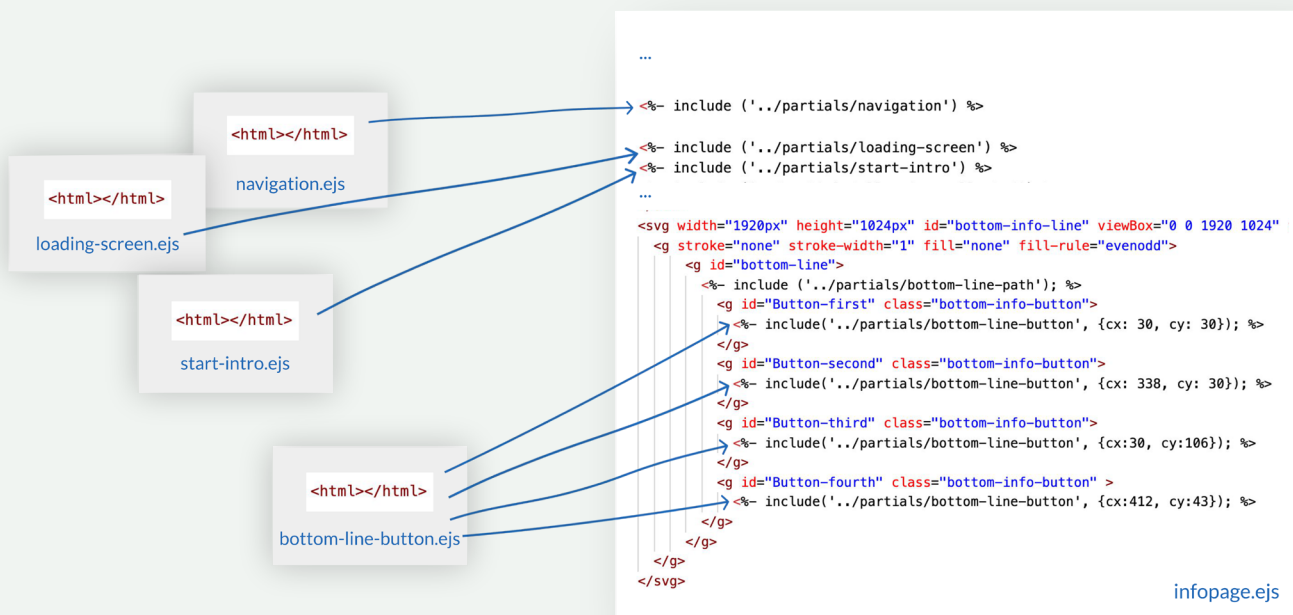


Abb. 28: Template-Engine

5.4. DEV-UMGEBUNG (BENJAMIN)



Abb. 29: Logo Webpack



Abb. 30: Logo ES-Lint



Abb. 31: Logo Heroku

WEBPACK

Wir haben uns direkt zu Beginn der Umsetzung dazu entschieden, Webpack zu verwenden. Der wichtigste Grund dafür war, dass es uns so auch möglich ist, npm-Pakete im Frontend zu benutzen. So können wir z. B. PixiJS und alle weiteren Abhängigkeiten einfach per npm installieren. Außerdem sind wir nicht auf Content Delivery Networks (CDNs) angewiesen und können alle Abhängigkeit selbst auf unserem Server hosten. Weitere Vorteile von Webpack sind, dass wir es auch zum Zusammenfügen und Komprimieren unserer JavaScript-Dateien und zur Kompilierung unserer SCSS-Dateien zu CSS benutzen konnten.

LINTER

Um eine gleichbleibende Codequalität zu gewährleisten, Fehler direkt zu erkennen und um beim Code ein einheitliches Erscheinungsbild zu wahren, haben wir uns für den Einsatz von ES-Lint entschieden. Wir haben die Standards von AirBnB verwendet und diese geringfügig an unsere Bedürfnisse angepasst.

DEPLOYMENT

Unsere Node.js-App haben wir auf Heroku deployed. Dies war ein sehr wichtiger Schritt, um unsere Webseite unter realen Bedingungen testen zu können. Um unsere App auf Heroku zum Laufen zu bringen mussten wir nur kleinere Anpassungen vornehmen, wie z. B. die Node.js- & npm-Version in unser package.json-Script hinzuzufügen und ein Procfile anzulegen. Im Procfile wird festgelegt, welcher Befehl zum Starten unserer Node.js-App ausgeführt werden soll.

Mithilfe der Heroku-CLI wird es dem Entwickler sehr einfach gemacht, den aktuellen Stand auf Heroku zu deployen. So muss man einfach das aktuelle Git-Verzeichnis auf Heroku pushen. Anschließend installiert Heroku automatisch alle Abhängigkeiten und führt auch alle Skripte, wie z. B. Webpack, aus.

6. DATENSCHUTZ (SARAH)

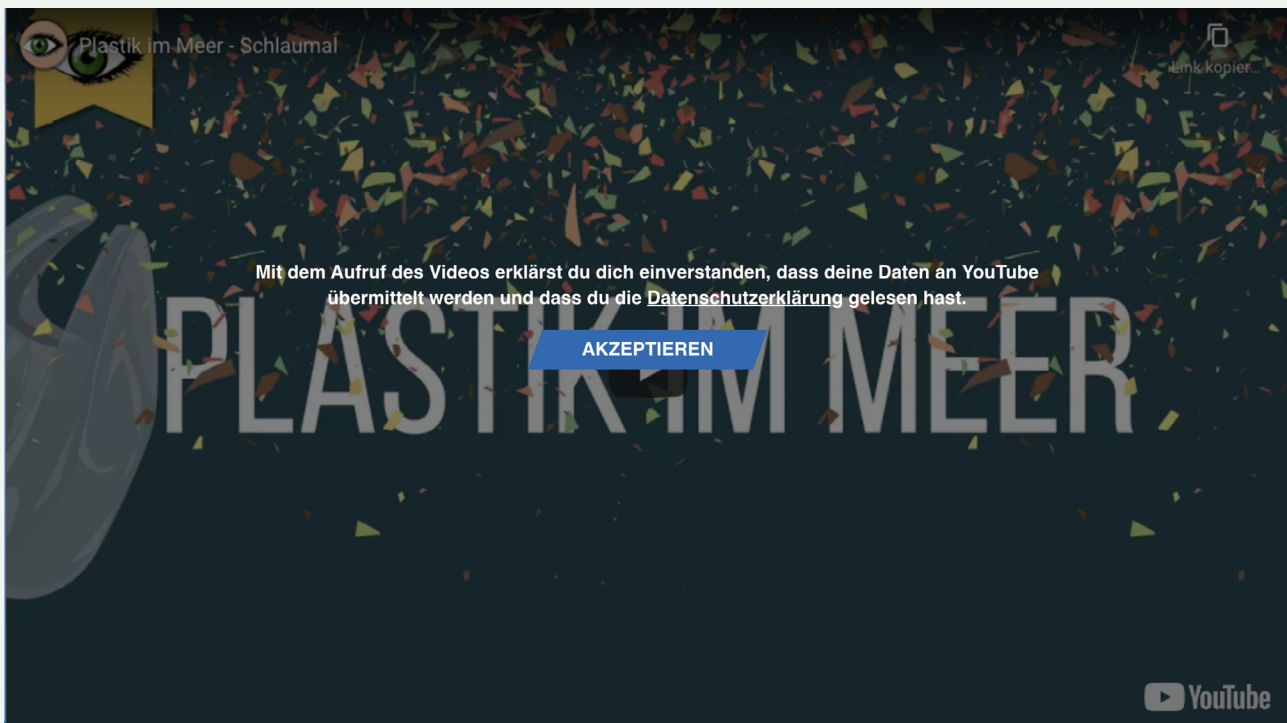


Abb. 32: Datenschutz bei eingebundenen YouTube Videos.

Da unsere Webseite live gegangen ist, mussten wir uns auch mit Datenschutz auseinandergesetzt. Auf unseren Info-Seiten haben wir zu den jeweiligen Themen YouTube Videos eingesetzt. Diese werden von einer externen Webseite (Youtube) geladen und deshalb muss hier die Einwilligung des Nutzers eingeholt werden. Eine Möglichkeit ist es, dies über ein Cookie-Banner zu tun. Da wir darauf verzichten wollten, haben wir uns eine andere Lösung überlegt. Dazu haben wir bei jedem eingebundenen Video ein Overlay, welches den Nutzer nach seiner Einwilligung fragt (Abbildung 32). So wird das Video erst geladen, wenn der Nutzer aktiv zustimmt.



7. HERAUSFORDERUNGEN (SARAH)

ZEIT

Eine unserer größten Herausforderungen war die Zeit. Konzept, Design und Umsetzung in zwei Monaten mit Technologien, in denen keiner von uns Erfahrung hatte, war eine große Hürde. Insbesondere, da wir dieses Semester auch die Umsetzung unseres Masterprojekts und weiteren Projekten meistern mussten. Trotzdem haben wir alles umgesetzt, was wir uns vorgenommen hatten. Dies lag auch daran, dass wir uns bereits von anderen Projekten kennen und so beim Erstellen des Zeitplans realistische Annahmen treffen konnten.

PROJEKTMANAGEMENT

Da unser Team aus nur 3 Personen bestand, war es nicht möglich, dass eine Person nur das Projektmanagement übernimmt. Sarah war für das Projektmanagement verantwortlich, jedoch haben wir Aufgaben meistens zusammen verteilt und geplant. Zwischenzeitlich hätten wir uns dennoch einen Projektleiter gewünscht, welcher den Zeitplan und die Tickets im Auge behält und die Teammitglieder motiviert.

MEETINGS

Die Umsetzung eines Software-Projektes ist auf jeden Fall rein online möglich. Trotzdem haben uns die persönlichen Treffen sehr gut getan und uns motiviert. Dadurch, dass wir uns schon aus dem vorherigen Semester kannten, ist uns die Verteilung der Aufgaben und die Zusammenarbeit deutlich leichter gefallen.



8. LEARNINGS



Abb. 33: Veronika

VERONIKA

In diesem Semester habe ich insbesondere gelernt, welchen Effekt visuelles Storytelling im UX-Design auf die Nutzer haben kann. Durch das gezielte Erzeugen von bestimmten Emotionen, kann die Auffassungsgabe verstärken und erzielt einen größeren Lerneffekt. Storytelling sollte man von Anfang an im Design-Prozess berücksichtigen, da davon viele Gestaltungsentscheidungen abhängen. Beispielsweise musste ich die Farbpalette so anlegen, dass auch die Farben der Münzen aus der Schatzsuche stimmig sind.

Zudem habe ich einige Regeln für die Zusammenarbeit mit Entwicklern gelernt. Ich weiß jetzt, wie wichtig es ist, Daten richtig anzulegen und in den richtigen Formaten zu exportieren. Dies kann den Entwicklern viel Zeit sparen.



Abb. 34: Sarah

SARAH

Zwar hatte ich schon mal mit Node.js gearbeitet, hatte aber vorher noch nie das Backend von Grund auf aufgebaut.

Das Thema Animationen habe ich mir vorher auch nicht genau angeschaut und deshalb fand ich Gsap wirklich interessant.

Besonders faszinierend fand ich aber auch, was man alles mit Storytelling auf einer Webseite erreichen kann. Vorher habe ich einfach nur Webseiten erstellt, wie man es halt so macht. Aber nie habe ich damit eine richtige Geschichte erzählt.

Und ich fand's toll zu sehen, dass auch ein Studentenprojekt funktionieren kann, wenn alle immer wieder sich gegenseitig pushen und man offen und ehrlich miteinander kommuniziert.

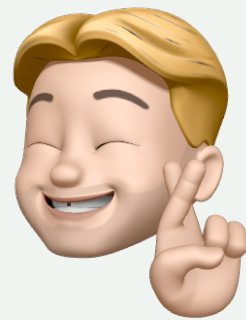


Abb. 35: Benjamin

BENJAMIN

Durch unser Projekt konnte ich mich intensiv mit PixiJS auseinandersetzen. Dabei konnte ich sehr viel über 2D-Rendering-Engines und Animationen lernen. Insbesondere die Logik wie z. B. das Hit-Testing hinter einer interaktiven Webseite waren für mich dabei besonders interessant.

Auch bei der Verwendung von Socket.io konnte ich sehr viele Dinge über das WebSocket-Protokoll und das Zusammenspiel von Client und Server lernen. Auch die Verknüpfung von PixiJS und Socket.io war sehr herausfordernd, aber trotzdem interessant.

Für den Livegang habe ich mich das erste Mal mit dem Deployment einer Node.js-App auf Heroku auseinandergesetzt. Dabei habe ich gelernt, wie einfach das Deployment möglich ist.

9. AUSBLICK (SARAH)

BROWSER- & GERÄTETESTING

Leider haben wir es nicht mehr ganz geschafft das Projekt auf Herz und Nieren zu testen. Wir haben gegen Ende des Semesters noch versucht, möglichst viele Bugs zu fixen. So konnten wir z. B. die in der Präsentation genannten Bugs in Firefox beheben. Für ein ausgiebiges Testing blieb leider keine Zeit mehr. Unsere Webseite sollte in allen gängigen Browsern wie Chrome, Safari und Firefox funktionieren. Die beste Experience erhält der Nutzer derzeit auf Chrome.

RESPONSIVE DESIGN

Unsere Website wurde für den Desktop konzipiert. Für die Darstellung auf Smartphone und Tablet müsste man die Webseite noch einmal stark anpassen. Wir haben unsere Webseite trotzdem so weit es geht angepasst, aber kein eigenes Design für mobile Geräte entwickelt.

VERBESSERUNG DER PERFORMANCE

Nach dem Livegang haben wir die Webseite ausgiebig auf unseren Geräten getestet. Dabei haben wir festgestellt, dass sie nicht auf allen Geräten flüssig läuft. Deshalb haben wir z. B. durch verkleinern von Assets versucht, die Performance noch einmal zu verbessern. Trotzdem gibt es hier noch Luft nach oben. Wir sind uns alle einig, dass man die Performance durch den Einbau von Caching oder das Reduzieren von Javascript-Code (z.B. Weglassen von jQuery), verbessern könnte.

Abschließend kann man sagen, dass wir mit dem Ergebnis des Projekts zufrieden sind. Die Zusammenarbeit lief sehr gut und harmonisch. Wir haben uns gegenseitig unterstützt und motiviert. Außerdem haben wir alle gesetzten Ziele erreicht und viel gelernt.



QUELLEN

- [1-6] Elementor. Using Storytelling in Web Design - Monday Masterclass. <https://www.youtube.com/watch?v=vf7ol5JliDg>
- [7] Nodejs. Über Node.js. <https://nodejs.org/de/about>
- [8] npm. About npm. <https://www.npmjs.com/about>
- [9] Express.js. Using template engines with Express. <https://expressjs.com/>
- [10] Express.js. <https://expressjs.com/>
- [11] Socket.io. Introduction. <https://socket.io/docs/v4/index.html>
- [12] MDN Web Docs. WebSockets. https://developer.mozilla.org/de/docs/Web/API/WebSockets_API
- [13] Socket.io. Rooms. <https://socket.io/docs/v4/rooms/>
- [14] PixiJS. FAQ. <https://www.pixijs.com/faq>