

# Bericht des Projekts einer Radtour in Vorarlberg

SS 2025

Mortitz Plattner: 11720451

Lukas Jehler: 11845123

Elisabeth Erdozain: 12120608

## Inhaltsverzeichnis

1 Projektbeschreibung .....	2
2 Startseite .....	2
HTML: .....	2
CSS: .....	5
3 Übersichtskarte: .....	7
4 Etappenseiten .....	10
Index.html .....	11
main.js .....	11
5 Fazit:.....	14
6 Literaturverzeichnis .....	15

# 1 Projektbeschreibung

Bei unserem Projekt haben wir eine dreitägige Radtour in Vorarlberg dargestellt und beschrieben. Die Route ist in ihrer gesamten Länge auf einer Übersichtskarte zu erkennen, für die drei einzelnen Tagesetappen gibt es jeweils drei extra Seiten. Zudem existiert eine Startseite mit Informationen zu dem Projekt und den Teammitgliedern. Entlang der Route sind Supermärkte dargestellt, die zur Verpflegung dienen. Noch dazu gibts mögliche Übernachtungsmöglichkeiten mit Campingplätzen in der Umgebung.

Auf der Startseite gibt es relevante Bilder als Sehenswürdigkeiten zu dem Tourgebiet in einer Slideshow mit einer navigierbaren Auswahl an Buttons, um verschiedene Seiten zu besuchen. Eine kurze Beschreibung von unserem Projekt könnte auf der Startseite gesehen werden, und den Teammitgliedern mit ihren Projektaufgaben.

# 2 Startseite

Für den Startseite war es prominent in HTML und CSS gemacht. Hier wird den erste Blick auf das Projekt gemacht. Unter der Headerbild der Vorarlberg Gebirgskette steht der Titel mit den verschiedenen Buttons zur Übersichtskarte und Etappen der Tour. Hier kann man leicht zu verschiedenen Seiten navigieren. Die verschiedenen color-coded Buttons dienen noch zu leichteren und schnelleren navigieren und helfen bei der Organisation der Seite. Zunächst kommt das Leistungsbild von der Vorarlberg Bergen mit Radfahrerinnen auf einem Berg auf der Straße. Diese unterstützt das Projekt bei einer Präsentation von was man in der Tour erwarten kann, in denen man oft bergauf und ab entlang der verschiedenen Etappen fährt. Unter steht eine kurze Projektbeschreibung zu der Tour mit ein paar Beispiele von der integrierte Features. Danach kommt die Sehenswürdigkeiten zu sehen. Ein paar werden in einer Slideshow präsentiert. Am Ende kommt der verschiedene Teammitglieder, die dieses Projekt gebaut haben. Unter dem Teammitglieds Icon stehen der Name und seine Aufgaben in dem Projekt in Bullet Points. Das Konzept für das Projekt steht unter diese in einem hellblauen Button.

## **HTML:**

Zu der HTML Teil des Codes sind der folgenden gemacht. Den nötige Plugins sind installiert, wie Font Awesome für den Icons und Swiperjs für den Slideshow. Den nötige Verknüpfung zu dem CSS Code sowie auch dem json Code ist wie normal gemacht. Der Rest ist in der Body der HTML gemacht. Für den Header ist ein Bild aus Pixabay, das frei zu verfügen ist. Es dient

der Seite, ein gewisses Thema zu stellen. Unter dem Header kommt der Titel, indem ein Rad Icon dazu gestellt ist. Das Icon kommt von Font Awesome. Unter dem Titel kommt der verschiedenen Etappen-Buttons, wobei ein bestimmte Klasse für die gebaut ist ("etappen-links"). Nach jeder Etappen link kommt noch eine spezifische Klasse abhängig, ob es eine Etappe ist oder den Übersichtskarte ist. Den Klassen hilft das Styling in der CSS später.

Um Platz zwischen den Etappen Buttons und den Leistungsbild, ist `<span></span>` benutzt. Mit dem Leistungsbild, ist es mit vorliegendem Text zusammen gemacht. Dies war ein neuer Schritt, wobei es eine interessante Website mit einem unterschiedlichen Layout war. Für diesen Schritt war es gemeinsam mit der Hilfe den Webpage: Codedamn gemacht (Codedamn, 2022). Der erste Teil wird auf der HTML Seite geschrieben, den zweite Teil zu den CSS wird in der CSS Teil dieses Berichtes beschrieben. Abbildung 1 zeigt den Teil des HTML zu diesem Schritt.

```
....<div class="container_above">
....|  <div class="bikers">
....||  
....|  </div>
....|  <div class="bikers_text">
....||  <h1>Rennradfahren in die Vorarlberg Berge</h1>
....|  </div>
....</div>
```

Abbildung 1: Screenshot aus dem index.html der Startseite. Quelle: eigene Erstellung

`<div>` wird gestellt, sodass dieser Teil separat betrachtet wird. Den `<div>` bekommt auch eine class: "container-above", sodass bestimmte Regeln in der CSS spezifisch nur auf die Teil des Codes führen werden. Der Name ist der obere Container, der mehr als einen Teil beinhaltet. Den class "bikers" ist zu dem Image gegeben und den Text, der darüber das Bild legen wird, bekommt der class: "bikers\_text". Am Ende kommt der `</div>`s der schließt diese Container Class.

Danach kommt die Projektbeschreibung, die in `<h4>` steht und der Text zu der Beschreibung steht in `<p>`. Zunächst kommt der Sehenswürdigkeiten, die zu der Vorarlberg Tour gehören. Unter einem kurzen Satz findet eine Slideshow statt, wobei man durch den verschiedenen Bilder klicken kann. Dies war auch ein neuer Schritt mit Hilfe der Webpage: [w3schools.com](https://www.w3schools.com) ([w3schools.com](https://www.w3schools.com), 2025). Den Teil des HTML Code kann in Abbildung 2 gesehen werden.

```

<div class="container">
    <!-- Slider main container -->
    <div class="swiper">
        <!-- Additional required wrapper -->
        <div class="swiper-wrapper">
            <!-- Slides -->
            <div class="swiper-slide">Bregenzerwald</div>
            <div class="swiper-slide">Feldkirch</div>
            <div class="swiper-slide">Formarinsee</div>
            <div class="swiper-slide">Vorarlberg Gebirge</div>
            <div class="swiper-slide">Vorarlberg Gebirge</div>
            ...
        </div>
        <!-- If we need pagination -->
        <div class="swiper-pagination"></div>

        <!-- If we need navigation buttons -->
        <div class="swiper-button-prev"></div>
        <div class="swiper-button-next"></div>
    </div>
</div>

```

Abbildung 2: Screenshot aus dem index.html der Startseite. Quelle: eigene Erstellung

Zuerst war der gesamte Bereich, der mit der Slideshow in der `<div>` begrenzt ist. In der `<div>` ist der `class: "container"` gegeben, um der CSS Styling besser zu anpassen. Danach kommt der allgemeine `class: "swiper"`, der spezifisch mit der allgemeinen Gestaltung der Slideshow handelt. Der `swiper-wrapper` ist der Klasse, die alle der unterliegenden Klassen steuert. Zunächst kommt der `class: "swiper-slider"`. Hier steht jede Slide für den Slideshow. Für jede Slide gibt es eine neue Abgrenzung mit `<div class: "swiper-slider">`. Danach kommt der Name, der zu dem Bild gehört (z.B. das Bild von Feldkirche hat den Übertitel: "Feldkirche"). Zunächst folgt das bestimmte Bild, das auf dieser Slide stehen sollte. Hier werden den normalen Referenzen von Bilder gemacht, wobei das gewünschte Bild vom Imagefolder aufgerufen wird. Den Line von Code ist mit `</div>` beschlossen. Nachdem alle Bilder zu der "swiper-slider" gerufen sind, kommt ein `</div>` der diesen Teil der Code grenzt. Unter steht die Möglichkeit, Pagination hinzuzufügen. Diese ist mit den kleinen Punkten am Rand des Bildes zu navigieren. Da es noch den Pfeilen zu navigieren steht, habe ich die Pagination weggelassen. Den letzte Teil dieses Code Abschnittes zeigt der zwei verschiedene Pfeilen der links und rechts zu Bild in Slideshow stehen. Diese dient zur Navigation der Slideshow. Den `</div>`s schließt diesen Code Abschnitt ab.

Darunter steht den Teammitgliedern mit ihren Icon, Name und Projektaufgaben in einer Liste. Hier werden die Profile Icons von Github als Screenshot heruntergeladen und weiter in Gimp produziert. In Gimp wird das Image zu 100 x 100 Pixeln skaliert, sodass sie die richtige Größe auf der Webseite sind. Bei jedem Teammitglied, ist der `class` in `<div> : "team-member"` gegeben. Für den Text sind der Name in Bold Schrift, und der Rest in normalen Schrift. Diese ist durch die Nutzung von `<ul>` um eine ungeordnete Liste zu erstellen. Den Bulletpoints sind

mit <li> gestellt. Zuletzt kommt der Link zu dem Konzept, der seinen eigenen Button und das Icon von Awesome Font hat.

## CSS:

In diese Teil der Startseite Code, wird den CSS Code kurz erläutert. Zuerst ist der Font Open Sans von [fonts.googleapis.com](https://fonts.googleapis.com) importiert. Den verschiedenen Umstellungen zu der Header Bild, Bilder, Body, Slideshow und Etappen wird geschrieben. Zu den Slideshow, wird den swiper class gestyled sodass der Breite der Slideshow zu 100% der Seitemargin fühlt. Der Höhe der Slideshow ist zu 500 Pixeln gestellt, sodass es gut auf der Seite passt, aber groß genug ist, um der Bilder gut zu vorstellen. Den Figures wird allgemein gestaltet mit display zu flex, der justify-content ist centre um der Bild richtig zu orientieren, der gap ist zu 100 Pixeln um gut genug Platz zwischen der Bilder zu geben, den flex-wrap ist zu wrap gestellt (den Images steht fließend neben einander, ohne Überlappung) und am Ende kommt der margin-top:2em der richtet der Platz der Bilder in Bezug auf den gestellt Margin (die wird 2 em von der Top stehen). Jeder Button bekommt seine eigene Farbe abhängig von der Klasse mit ein bestimmten Breite, Höhe und Größe, der balanciert auf den Seite steht. Den Text zu der obere

```
99
100 .concept {
101   padding: 0.75em 4.5em;
102   font-size: 1.1rem;
103   background-color: #lightblue;
104   color: #white;
105   text-decoration: none;
106   border-radius: 8px;
107   width: 12em ;
108   transition: background-color 0.2s ease;
109   margin:auto;
110 }
111
112 .bikers_text [
113   font-size:small;
114   float:left;
115   color:#aliceblue;
116   width: 15em;
117   margin:center;
118 ]
119
120 .container_above {
121   display:grid;
122   align-items:start;
123   grid-template-columns: 1fr 1fr 1fr;
```

Abbildung 3: Screenshot aus dem main.css der Startseite. Quelle: eigene Erstellung

Bild gehört bekommt ein Farbe aliceblue, und ist in Verknüpfung mit der .container\_above class verknüpft. (Abbildung 3)

Den grid-template-columns in der .container\_above erlaubt den Text(.bikers\_text) drüber zu stehen.

Für die Gestaltung der Teammitgliedernseite, habe ich es originalweise ohne KI gestaltet. Zuerst habe ich den Image Format umgewandelt, sodass die Icons geründet mit ein Radius von 50% sind. Den Größe war mit ein Breite von 120px der ein bisschen breite wie der originale 100px und der Höhe war auf auto gesetzt sodass es zu der Breite passt (auch 120px, da es ein perfekter Kreis ist). Nach der Image Formatierung, der Figcaption zu Bold Schrift gestaltet und mit der Positionierung auf der Seite mit margin-top Wert von 0.5em. Der Liste der Teammitgliedern, hat es die gleiche margin-top als der Schrift sodass die homogen zueinander auf der Webseite liegt. Den padding-left: 1em und text-align:left richtet der Text, sodass es ein bisschen links im Vergleich mit der Überschrift und Image steht. Sondern diese in eine balancierte Homogen gestaltete Sicht. Zuerst hatte ich Schwierigkeiten, alle drei Komponenten zusammen zu bilanzieren. Ein Änderung der zu einer der drei führt zu Störungen bei der Andere Zwei. Deswegen habe ich KI auf diese Teil benutzt, um der Seite zusammenzusetzen.

```

/*KI_Begin*/
.team-member {
    display: flex;
    flex-direction: column;
    align-items: center;
    text-align: center;
    max-width: 220px;
    padding: 1em;
}

.team-member img {
    width: 120px;
    height: auto;
    border-radius: 50%;
    margin-bottom: 0.5em;
}

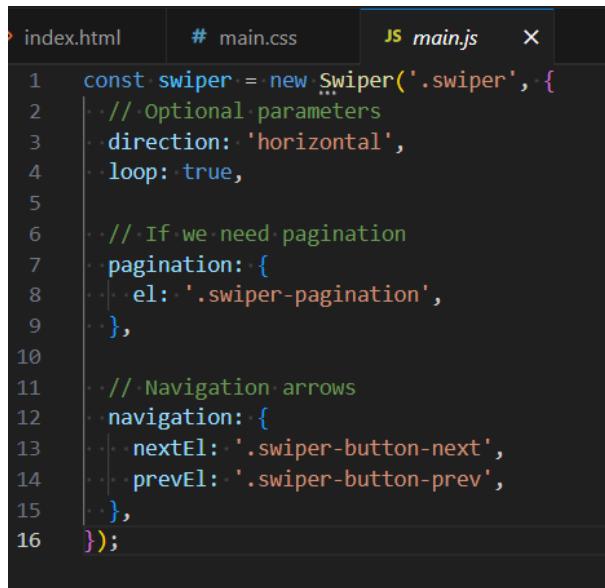
.team-member figcaption {
    font-weight: bold;
    margin-top: 0.5em;
}

.team-member ul {
    text-align: left;
    padding-left: 1em;
    margin-top: 0.5em;
}
/*KI_End*/

```

Abbildung 4: Screenshot aus dem main.css der Startseite.  
Quelle: eigene Erstellung

Der JSON Teil des Code war sehr klein, da es nur eine Funktion zur Gestaltung der Slideshow enthält. Diese könnte in der Abbildung 5 gesehen werden.



```
index.html # main.css JS main.js X
1 const swiper = new Swiper('.swiper', {
2   // Optional parameters
3   direction: 'horizontal',
4   loop: true,
5
6   // If we need pagination
7   pagination: {
8     el: '.swiper-pagination',
9   },
10
11   // Navigation arrows
12   navigation: {
13     nextEl: '.swiper-button-next',
14     prevEl: '.swiper-button-prev',
15   },
16 });


```

Abbildung 5: Screenshot des main.js der Startseite.  
Quelle: eigene Erstellung

Hier den Function: swiper ist gegeben, wobei die Richtung der Slide ist horizontal und der Bewegung könnte unendlich durchgehen (loop: true). Pagination erlaubt den Punkten unter der Slideshow zu deuten, wie viel und an welchen Stellen man ist, aber nicht für die physische Navigation benutzt. Den Navigations-Pfeilen erlaubt es, die Bilder in der Slideshow durch zu klicken.

### 3 Übersichtskarte:

Die Übersichtskarte dient zur Darstellung der gesamten Tour. Im oberen Teil der Seite kann man mithilfe der interaktiven Buttons durch die verschiedenen Seiten navigieren. Die Buttons wurden mit Hilfe von Links erzeugt und werden im Kapitel Etappenseiten näher beschrieben. Indem den Buttons unterschiedliche classes zugewiesen wurden, können sie im css separat angepasst werden. Die aktuelle Seite soll immer blau erscheinen, die anderen sind grün. (Abbildung 6)

Unterhalb dieser Navigationsleiste findet man Informationen und Daten zur ganzen Tour. Ähnlich wie beim Übungsbeispiel Bike-Tirol wurde mittels div Klassen und "Font Awesome" Übersicht Icons erstellt, die Informationen zu der Tour beinhalten. Unterhalb befindet sich eine weitere Navigationsleiste, sie wurde mit Hilfe von W3schools erstellt ([w3schools.com](https://www.w3schools.com), 2025). Die Tabs „Beschreibung“, „Anreise“, „Unterkunft“ und „Verpflegung“ erlauben es dem Benutzer, durch Klicken zwischen den drei verschiedenen Inhaltsbereichen zu wechseln.

```

.etappen-button {
    padding: 0.75em 3em;
    font-size: 1.1rem;
    background-color: green;
    color: white;
    text-decoration: none;
    border-radius: 8px;
    transition: background-color 0.2s ease;
}

.etappen-button-v2 {
    padding: 0.75em 3em;
    font-size: 1.1rem;
    background-color: blue;
    color: white;
    text-decoration: none;
    border-radius: 8px;
    transition: background-color 0.2s ease;
}

```

Abbildung 6: Screenshot des main.css der Übersichtskarte. Quelle: eigene Erstellung

Jede Schaltfläche hat eine Klasse tablinks und hat ein sogenanntes onclick Attribut. Dieses sorgt dafür, dass beim Klick eine JavaScript-Funktion aufgerufen wird. Der Funktion wird übergeben, welches Tab geöffnet werden soll – zum Beispiel „Anreise“ oder „Verpflegung“, der Rest bleibt ausgeblendet. (Abbildung 7)

```

<div class="tab">
    <button class="tablinks" onclick="openCity(event, 'Beschreibung')>Beschreibung</button>
    <button class="tablinks" onclick="openCity(event, 'Anreise')>Anreise</button>
    <button class="tablinks" onclick="openCity(event, 'Verpflegung')>Verpflegung</button>
    <button class="tablinks" onclick="openCity(event, 'Unterkünfte')>Unterkünfte</button>

```

Abbildung 7: Screenshot aus dem index.html der Übersichtskarte. Quelle: eigene Erstellung

Die vier verschiedenen Inhalte werden unterhalb definiert, für den Reiter Anreise wurde KI verwendet, da ansonsten die Karte nicht mehr richtig dargestellt wurde. Mit „Scotty“ wurde der ÖBB Fahrplan dort eingefügt, da diese Applikation recht groß ist, wurde die Karte vermutlich falsch dargestellt. (Abbildung 8)

```

<!-- KI_BEGIN -->
<div id="Anreise" class="tabcontent">
    <h3>Zufahrt mit öffentlichen Verkehrsmitteln</h3>
    <iframe src="https://fahrplan.oebb.at/webapp/?style=widget&arr=Dornbirn%20Bahnhof" width="100%" height="400" style="border: none" loading="lazy" title="ÖBB Fahrplan Widget">
</iframe>
</div>
<!-- KI_END -->

```

Abbildung 8: Screenshot aus dem index.html der Übersichtskarte. Quelle: eigene Erstellung

Mit iframe gab es keine Konflikte mehr, da Karte und das Scotty-Widget unabhängig voneinander laufen. Unterhalb befindet sich die Hauptkarte mit der Darstellung der gesamten Tour. Das Höhenprofil wurde mit css angepasst, damit es nicht mehr flächig gefärbt ist, sondern als schwarze Linie angezeigt wird.

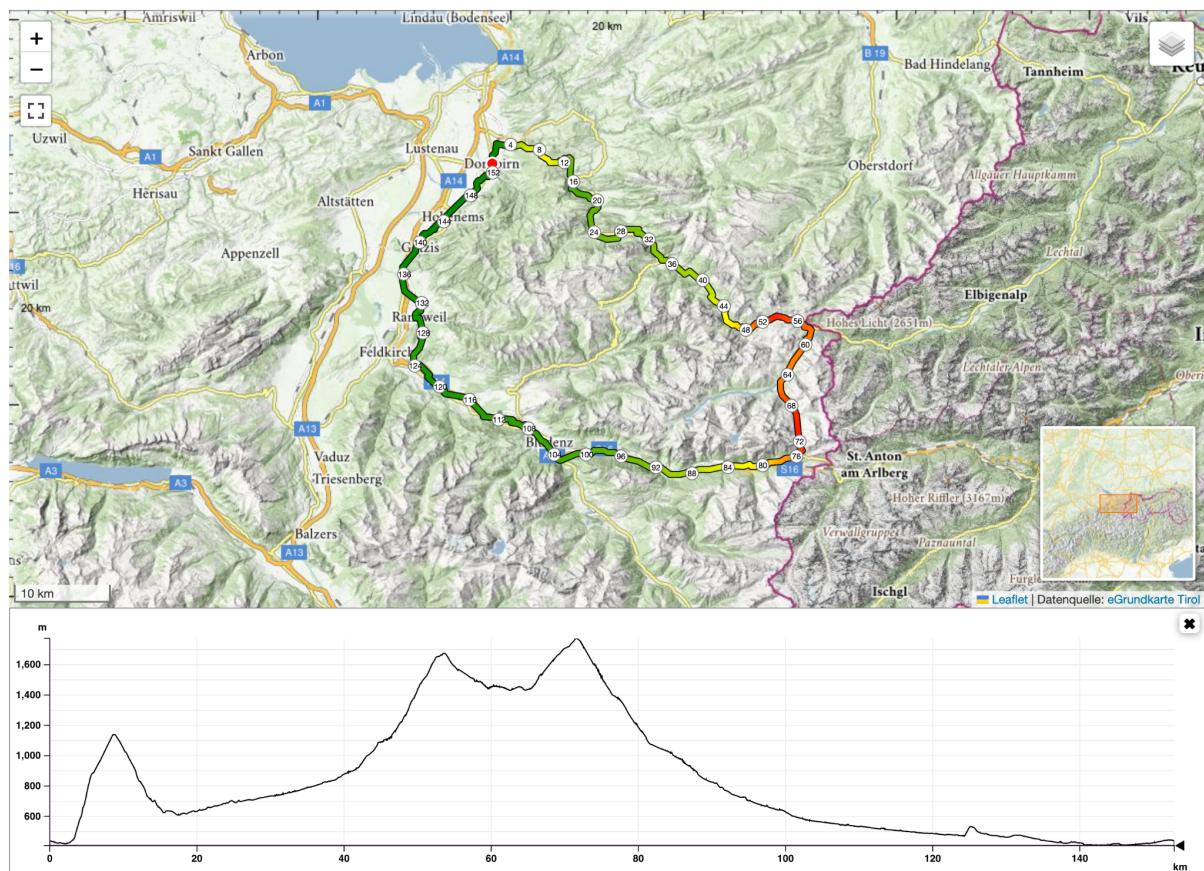


Abbildung 9: Ansicht der Webseite der Übersichtskarte. Quelle: eigene Erstellung

Für die Karte wird zunächst eine Variable definiert, die die geografischen Koordinaten eines ungefähr zentralen Punkts in Vorarlberg enthält. Diese Koordinaten werden dann verwendet, um die Startposition der Karte festzulegen. Mit L.map wird die eigentliche Karte dann erstellt. Danach werden verschiedene Hintergrundlayer vorbereitet. Diese stammen von der eGrundkarte Tirol und bieten unterschiedliche Darstellungen. Auf overlays wie Supermärkte und Campingplätze wurde in der Hauptkarte verzichtet, sie befinden sich in den einzelnen Etappen.

Mithilfe von Outdooractive wurde die Strecke erstellt. Hierfür wurde zunächst die gesamte Etappe eigenhändig, also nicht durch ein bestehendes GPX-File definiert. Da die Einzeletappen ebenfalls von Bedeutung sind, wurden sie ebenfalls definiert und anschließend heruntergeladen. Mit dem leaflet-elevation Plugin kann das GPX file der ganzen Tour eingelesen werden. Ebenfalls wird das dazugehörige Höhenprofil erzeugt. Als kleine Übersichtskarte wird eine Minimap hinzugefügt, ein Plugin, das eine kleine Karte im unteren rechten Bereich der Karte einblendet. Ein Fullscreen stylesheet dient für die Darstellung der Karte im Vollbild Modus. Im letzten Teil des Javascript-Codes steht die Funktion, die für die Steuerung der in Abbildung 10 dargestellten Leiste benötigt wird.

Abbildung 10: Screenshot aus der Webseite der Übersichtskarte. Quelle: eigene Erstellung

Diese Funktion wird beim Klick auf einen Tab-Button aufgerufen. Sie sorgt dafür, dass beim Klicken auf einen der Tab-Buttons der entsprechende Inhalt angezeigt und alle anderen Inhalte ausgeblendet werden. Zuerst werden alle bestehenden Inhalte mit der Klasse tabcontent versteckt. Danach werden alle Tab-Buttons mit der Klasse tablinks durchsucht und deren „active“-Markierung entfernt, damit sie optisch nicht mehr als aktiv erscheinen. Anschließend wird der Inhalt des Tabs, der zur gedrückten Schaltfläche gehört (z. B. „Anreise“), wieder eingeblendet, und der entsprechende Button bekommt die Klasse „active“, um als aktiv dargestellt zu werden. Dadurch sieht der Nutzer nur den aktuell ausgewählten Bereich, und der zugehörige Button wird hervorgehoben.

## 4 Etappenseiten

❖ Etappe 1: Von Dornbirn nach Mellau



Abbildung 11: Screenshot der Ansicht der Etappenseite 1. Quelle: eigene Erstellung

### Allgemeine Beschreibung:

Neben der Hauptkarte gibt es für jede Etappe eine eigene Seite, auf der nur die jeweilige Etappe dargestellt ist. (Abbildung 11) Die Karte ist hier der Zentrale Teil der Seite und umfasst die volle Breite der Seite, um einen möglichst guten Überblick zu erhalten. Oberhalb der Karte bleibt die Navigation wie auf der Hauptkarte erhalten. Man kann in eine andere Etappe wechseln oder zurück auf die Übersichtskarte navigieren. Die Seite, auf der man sich gerade befindet, wird in blauer Farbe dargestellt. In der Karte ist die Etappe mit Hilfe einer .gpx Datei dargestellt. Jede Etappe stellt einen Teil der Gesamt-Tour dar. Zusätzlich erhält man in der Etappen- Detailansicht zusätzliche Informationen entlang der Strecke. Es werden Campingplätze angezeigt und ein Popup beschreibt den Namen und Telefonnummer des jeweiligen Campingplatzes. Zusätzlich kann ein Layer "shops" aktiviert werden. Nun werden

alle Einkaufsmöglichkeiten in Vorarlberg und entlang der Strecke als Marker angezeigt. In dieser Karte kann auch der aktuelle Standort angezeigt werden. Dafür muss man den Standort am verwendeten Endgerät freigeben. Im unteren Teil der Seite wird ähnlich wie bei der Gesamtkarte ein Höhenprofil der jeweiligen Karte dargestellt.

## **Index.html**

Im head des index.html werden einige Metadaten definiert und der Titel der Seite vergeben. Anschließend werden alle Verbindungen zu den Plugin Providern hergestellt. Es werden anhand von links und scripts verschiedene .css und .js Dateien eingebunden. Darunter befinden sich Font Awesome für das Styling, Leaflet für die Erstellung der Karten, das leaflet elevation Plugin für die Erstellung der Höhenprofile, das Plugin leaflet marker cluster, für das zusammenlegen von mehreren Markern zu einem Cluster und das Plugin locate control zum Bestimmen des eigenen Standorts. Abschließend werden noch die eigene main.css Datei und [main.js](#) Datei eingebunden.

Im body des index.html wird der Titel der Seite in einem h1 Element definiert. Anschließend wird in einem div Element die Navigation auf die weiteren Seiten behandelt. Das div Element erhält die Klasse etappen-links, die im main.css definiert wird. Die Option “display: flex” macht aus den Links einen Button, den man wie einen Link anklicken kann, um auf die jeweilige Seite zu gelangen. Die weiteren Optionen betreffen das Styling. Die weiteren Optionen betreffen das Styling. Sie definieren den Abstand an den Seiten, oben und unten und den Abstand zwischen den Buttons. In dem div Element befinden sich vier links in Form eines a Elements. Href verweist auf die index.html Seite der Seite, auf die man durch klicken erreichen möchte. Die einzelnen Elemente erhalten eine weitere Klasse, die im main.css definiert wird. Die Klasse nennt sich “etappen-button” und definiert die Größe der Buttons, die Schriftgröße, die Hintergrund- und Schriftfarbe, verhindert Text Dekorationen wie das Unterstreichen und definiert die Form des Randes. Es gibt eine weitere Klasse “etappen-button-2”, mit denselben Eigenschaften nur in blauer, statt grüner Farbe. Diese Klasse wird jeweils jenem Link zugewiesen, auf der man sich gerade befindet. Somit kann man auf jeder Seite schnell erkennen, wo in der Navigation man gerade ist. Abschließend wird im index.html die Höhe und Position der Karte und des Höhenprofils definiert. Initialisiert werden sie dann im [main.js](#), das im nächsten Abschnitt beschrieben wird.

## **main.js**

In diesem script wird die Karte der Etappen Seiten erstellt. Grundlegend ist die Initialisierung der Karte mit L.map. Die Karte wird mit Hilfe von Leaflet erstellt. Das Zoom-Level wird festgelegt und mit setView definiert, wo man beim ersten Aufrufen der Karte hinkommt. In

diesem Fall werden die Koordinaten von Vorarlberg verwendet. Anschließend werden wie in der Übersichtskarte verschiedene Hintergrundlayer definiert. Neu dazu kommt im nächsten Schritt ein Overlay. Es werden zwei Overlays definiert. Das Overlay "campings" wird als "featureGroup" erstellt und default in der Karte angezeigt. Ein weiteres Overlay "shops" kann optional als "markerClusterGroup" angezeigt werden. Die Karten und die Overlays werden in ein Layer Control geschrieben und sind somit auf der Karte zu finden. Nach dem Layer Control wird auch für die Etappen Karten ein Maßstab erstellt.

Nun folgen zwei async Funktionen, die auf die Daten der Shops und Campings zugreifen und diese in den Overlay schreiben:

1. Campings: Leider war unsere Recherche nach einem verfügbaren Datensatz mit Campingplätzen in Vorarlberg erfolglos. Daher musste ein eigener Datensatz erstellt werden. Die Namen und Adressen der Campingplätze wurden auf der Seite <https://www.campingclub.at> entnommen. Mit Hilfe der Adressen wurden auf der Seite [www.gps-coordinates.net/gps-coordinates-converter](http://www.gps-coordinates.net/gps-coordinates-converter) die jeweiligen Koordinaten in WGS 84 erstellt. Diese wurden in Arcgis in Form eines Shapefiles digitalisiert. Als Attribute wurden der Name des Campingplatzes und die Telefonnummer gespeichert. Das Shapefile wurde mit dem Tool "Feature to json" in ein .geojson verwandelt und für die Darstellung in der Etappenkarte verwendet. Die Datei wurde auf der Git Projektseite gespeichert und von dort durch eine url geladen. Das icon für die Campingplätze wurde auf der Seite [mapicons.mapsmarker.com](http://mapicons.mapsmarker.com) heruntergeladen und wird an den Positionen der Campingplätze als Marker angezeigt. Zusätzlich wurde mit layer.bindPopup ein Popup erzeugt, das beim draufklicken den Namen und die Telefonnummer des Campingplatzes anzeigt. Hier hätte man noch weitere

```
// Campingplätze Vorarlberg
async function loadCampings(url) { // funktion wird definiert
    //console.log(url);

    let response = await fetch(url); // anfrage an server
    let jsondata = await response.json(); // in variable schreiben
    //console.log(jsondata);

    L.geoJSON(jsondata, {
        attribution: "Datenquelle: <a href='https://www.campingclub.at/campingplaetze/vorarlberg'>Camping Club</a>",
        pointToLayer: function (feature, latlng) {
            return L.marker(latlng, {
                icon: L.icon({
                    iconUrl: '../icons/camping.png',
                    iconAnchor: [16, 37],
                    popupAnchor: [0, -37] // popup um Bildhöhe nach oben verschieben
                })
            });
        },
        onEachFeature: function (feature, layer) {
            console.log(feature.properties)
            layer.bindPopup(
                '

#### ${feature.properties.name}</h4> <div>Tel.: ${feature.properties.tel}</div> '); } }).addTo(overlays.campings); // mit leaflet in karte hinzufügen! };


```

Abbildung 12: Screenshot aus dem main.js der Etappenseite 1. Es ist die Funktion zur erstellung der Marker mit Popup für die Campingplätze dargestellt. Abbildung 13: Screenshot der Ansicht der Etappenseite 1. Quelle: eigene Erstellung

Informationen implementieren können, was aufgrund der händischen Erstellung der .geojson Datei weggelassen wurde. Die gesamte Funktion ist in Abbildung 12 dargestellt.

2. Shops: Im Gegensatz zu den Campingplätzen waren wir erfreut, ein bestehendes Shapefile auf der Seite [data.gv.at](http://data.gv.at) vorzufinden. Es sind alle Einkaufszentren in Vorarlberg abgebildet. Der Datensatz enthält die Flächen der Einkaufszentren. Die Flächen wurden in Arcgis in einen Punktlayer verwandelt, um die Größe der Datei zu minimieren. Leider sind die Attribute für diesen Datensatz nicht sehr detailliert dokumentiert. Es sind lediglich Kürzel für den Flächenwidmungsplan und ein Attribut "legende" mit unterschiedlichen Informationen zur Einkaufsmöglichkeit beschrieben (z.B. Fläche des Ladens). Da der Aufwand sehr groß wäre, für jede Einkaufsmöglichkeit wertvolle Informationen wie Name, angebotene Produkte oder Öffnungszeiten zu erhalten, werden für die Einkaufszentren nur die Position mit Hilfe eines Markers dargestellt. Auch dieser Marker wurde von der Seite [mapicons.mapsmarker.com](http://mapicons.mapsmarker.com) bezogen.

Beide Funktionen wurden zuerst definiert und anschließend aufgerufen, indem die url übergeben wird. Nach dem Laden der Funktionen wird das leaflet Pluging locate control implementiert. Es erstellt in der Karte eine Möglichkeit, den live Standort zu aktivieren und auf der Karte anzuzeigen. Abschließend wird mit dem Plugin "control elevation" die .gpx Datei für die jeweilige Etappe in der Karte dargestellt. Zusätzlich wird automatisch das interaktive Höhenprofil unterhalb der Karte erstellt.

Damit sind alle drei Seiten der Etappen erstellt worden und ermöglichen eine detaillierte Ansicht für den jeweiligen Streckenabschnitt. Besonders unterscheiden sie sich von der Hauptkarte, indem die Karte noch präsenter dargestellt ist, zusätzliche Overlays angezeigt werden können und der Live- Standort aktiviert werden kann. Somit hat man zusätzliche Informationen, die man für die Navigation der einzelnen Etappen verwenden kann.

## 5 Fazit

Grundsätzlich ist die Navigation durch die Seiten über die verschiedenen Link Buttons gelungen. Auch dass die aktuelle Seite dann in einem blau gefärbten Button erscheint, macht die Navigation übersichtlich. Natürlich könnte mit aufwändigem css styling das Design noch verbessert werden. Auch die Tableiste oberhalb der Karte ist nützlich um zusätzliche Informationen anzuzeigen. Dadurch, dass nicht alle Daten gleichzeitig erscheinen, bleibt die Übersichtskarte übersichtlich.

Die größte Schwierigkeit für die Umsetzung der initialen Ideen war die Verfügbarkeit von bestehenden Datensätzen. Wir haben uns bewusst für den Raum Vorarlberg entschieden, da wir hier in einer kurzen Recherche am meisten Datensätze gefunden haben. Bei einer vertieften Analyse fanden wir aber heraus, dass viele Datensätze nicht zu gebrauchen sind oder keine wertvollen Informationen enthalten. Zum Beispiel sah der Datensatz Einkaufszentren zunächst vielversprechend aus, um ein Popup mit vielen Informationen zu erstellen. Leider enthielten die Attribute des Datensatzes, abgesehen von den Koordinaten, aber gar keine nützliche Information für das Projekt. Durch die mangelnde Verfügbarkeit waren wir gezwungen, eigene Datensätze zu erstellen. Das ist uns mit den .gpx Dateien für die Darstellung der Fahrradtour und dem Datensatz der Campingplätze gelungen. Die selbstständige Erstellung der Daten war eine gute Übung und hat das Verständnis für die erstellten Datentypen stark vertieft.

Leider haben wir erst zu spät von einem Kollegen erfahren, dass man in GitHub mit verschiedenen branches arbeiten kann/sollte, das hätte gleichzeitiges Arbeiten vermutlich einfacher gemacht.

## 6 Literaturverzeichnis

Codedamn. (2022). *How to put image and text side-by-side in HTML?* Abgerufen am 24. Juni 2025, von <https://www.codedamn.com>

Font Awesome. (o. J.). *Font Awesome*. Abgerufen am 24. Juni 2025, von <https://fontawesome.com>

GPS Coordinates Converter. (o. J.). [www.gps-coordinates.net](http://www.gps-coordinates.net). Abgerufen am 24. Juni 2025, von <https://www.gps-coordinates.net/gps-coordinates-converter>

Leaflet. (o. J.). *Plugins*. Abgerufen am 24. Juni 2025, von <https://leafletjs.com/plugins.html>

Leaflet-Extras. (o. J.). *Leaflet Provider Demo*. Abgerufen am 24. Juni 2025, von <https://leaflet-extras.github.io/leaflet-providers/preview/>

Map Icons Collection. (o. J.). *mapicons.mapsmarker.com*. Abgerufen am 24. Juni 2025, von <https://mapicons.mapsmarker.com>

Offene Daten Österreich. (o. J.). *data.gv.at*. Abgerufen am 24. Juni 2025, von <https://www.data.gv.at>

Outdooractive. (o. J.). *Outdooractive*. Abgerufen am 24. Juni 2025, von <https://www.outdooractive.com/de/>

Österreichischer Camping Club. (o. J.). *Campingclub.at*. Abgerufen am 24. Juni 2025, von <https://www.campingclub.at>

W3Schools. (o. J.). *How TO – Code snippets for HTML, CSS and JavaScript*. Abgerufen am 24. Juni 2025, von <https://www.w3schools.com/howto/default.asp>

W3Schools.com. (2025). *How to create a slideshow*. Abgerufen am 24. Juni 2025, von <https://www.w3schools.com>

Vereinzelt ChatGPT: <https://chatgpt.com>. Abgerufen am 24. Juni 2025