**Evaluation Prototyp – wanderoo**

Contents

[Vorwort 1](#_Toc52312690)

[Log-In 1](#_Toc52312691)

[Passwort vergessen 1](#_Toc52312692)

[Entscheidungsbaum 2](#_Toc52312693)

[Menü 2](#_Toc52312694)

[Karte 2](#_Toc52312695)

[Intelligente Karte 3](#_Toc52312696)

[Notfallknopf 3](#_Toc52312697)

[QR-Code-Scanner 3](#_Toc52312698)

[Augmented Reality 3](#_Toc52312699)

[Blog 3](#_Toc52312700)

[Attraktionen 3](#_Toc52312701)

[Hundebesitzer 4](#_Toc52312702)

[Mountainbiking 4](#_Toc52312703)

[Geocaching 4](#_Toc52312704)

[Livestream-Möglichkeit 4](#_Toc52312705)

[Internationalisierung 4](#_Toc52312706)

# Vorwort

Vorab möchten wir betonen, dass einige „Features“ in der Applikation „wanderoo“ nur zur Anschaulichkeit und nicht zur realen Anwendung implementiert werden, beispielsweise die Karte oder der Notfallknopf. Das Java-Projekt wird nicht auf Smartphones angepasst sein, wir simulieren dies jedoch durch die Fenstermaße und das User Interface, die die Applikation besitzen wird.

# Log-In

Das Team hat sich dafür entschieden, den Log-In-Screen aus dem Prototypen zu implementieren, da Accounts eine wichtige Rolle für unsere Applikation spielen. Ohne Accounts ist ein individuelles App-Erlebnis kaum möglich. Jeder Benutzer hat andere Präferenzen, die in dem Mein Profil-Screen eingestellt werden können. Diese Daten sollen auf einer Datenbank gespeichert werden. Die Grundlage hierfür bildet in dem Falle der Log-In-Screen.

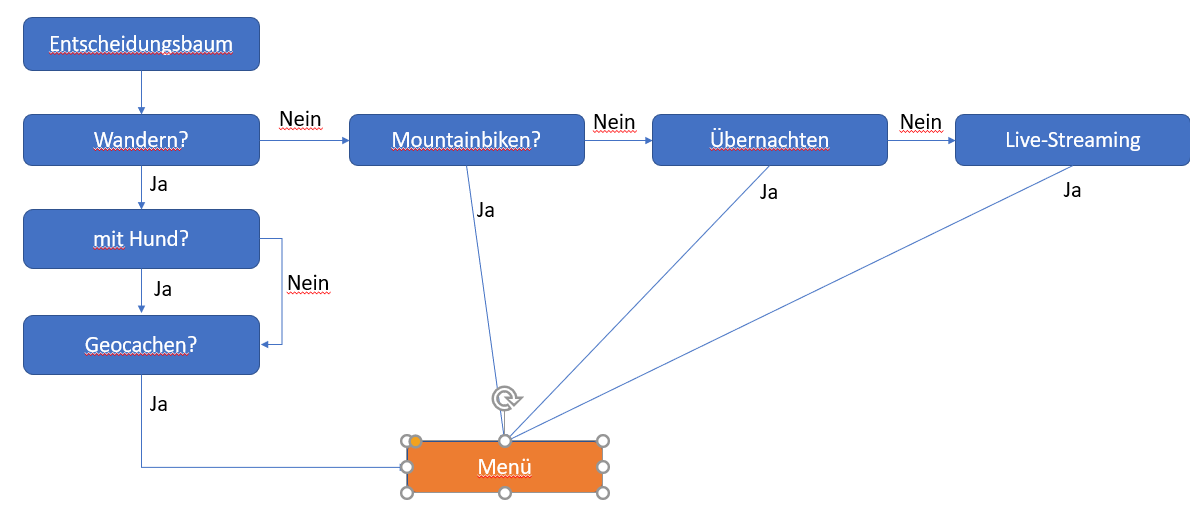
# Passwort vergessen

Die „Passwort vergessen“-Funktion wurde nicht ausimplementiert, da dort eine E-Mail versendet werden würde, um dem Benutzer einen Code zuzuschicken. Dies scheint unserer Ansicht nach nicht wichtig für den Grundsatz unseres Projekts zu sein und wurde deshalb vernachlässigt.

# Entscheidungsbaum

Der Entscheidungsbaum hat es leider nicht in den „integrated Prototype“ geschafft. Ein solcher Entscheidungsbaum soll nach dem Login mithilfe von verschiedenen Abfragen auf dem UI ausgeführt werden.

Vereinfacht sieht dieser wiefolgt aus:



So wird deutlich, dass nach dem „Einloggen“ der Benutzer abgefragt wird, wie sein Erlebnis aussehen soll für die Wanderung. So kann sichergestellt werden, dass ein noch individuelleres Erlebnis der wanderoo-Applikation zur Verfügung gestellt wird. Nach Ausführen dieser Abfragen sollte das UI auf die vorigen Eingaben angepasst sein: Jemand der nicht wandern möchte, sollte also nicht den Wandern-Screen angezeigt bekommen, sondern dann je nach Eingabe beispielsweise den Mountainbiking-Screen oder den Übernachtungsscreen sehen.

# Menü

Das Menü spielt ebenfalls eine essentielle Rolle in der Applikation. Zu einer intuitiven Navigation gehört unserer Meinung nach ein Menü, aus dem man in die verschiedenen „Services“ der Applikation springen kann. Im Prototypen haben wir uns bewusst nicht für eine Side-Bar oder eine Bottom Icon Navigation Bar entschieden, um die App schlicht und übersichtlich zu halten. Zudem lässt sich ein eigner Menü-Screen leichter in einer Java-Applikation implementieren.

# Karte

Eine Karte, wie von Google Maps bekannt, würden wir ebenfalls gerne umsetzen, da sie das Hauptmerkmal unserer Applikation darstellt. Wanderrouten sollen über diese Karte angezeigt werden. Im Prototyp ist ebenfalls zu sehen, dass mit Hilfe von Schiebereglern ausgewählt werden kann, wie lang die Wanderstrecke sein soll und wie viele Höhenmeter mit inbegriffen sein sollen. Wir haben uns dafür entschieden, diese Schieberegler aus dem Prototypen nicht mit zu implementieren, da die Komplexität für uns innerhalb der gegebenen Zeit nicht umsetzbar scheint. Auch bei der Umsetzung der Karte kürzen wir die Implementation. Es scheint so, dass Google Maps keine API für Java anbietet, sondern nur für WebApps. Um unsere App trotzdem anschaulich zu gestalten, werden wir versuchen mithilfe von Web-Integrationen in Java eine Google Maps-Seite darzustellen.

Eine eigene Kartenimplementation scheint schlichtweg zu komplex.

Nach heutiger Sicht (Stand: 01.09.2020) scheint auch die Implementation von der oben genannten Swing-Browser-Komponente nicht erreichbar, da man für die Bibliothek, die wir dafür genutzt hätten, eine Lizenz benötigt. Somit wird die Karte nur „generisch“ in Form eines Bildes oder Ähnlichem in der Applikation zu finden sein.

## Intelligente Karte

Für die „Intelligente Karte gilt Ähnliches. Es scheint zu komplex zu sein, intelligente Features in eine Karte zu bauen, bei der die Sorge besteht, dass sie nur als Browser-Swing-Komponente implementiert wird. Eine intelligente Karte war von der Gruppe als Idee gedacht, und auch im Prototypen wird diese Idee gerne veranschaulicht, es scheitert leider an der realen Umsetzungsmöglichkeit.

# Notfallknopf

Der Notfallknopf soll implementiert werden, es soll jedoch nur ein Anruf simuliert werden. Da die Applikation nicht nativ für ein Smartphone entwickelt wird, lässt sich ein Notruf nur simulieren. Vom ausführenden PC wird keine SIM-Karte vorausgesetzt.

# QR-Code-Scanner

Der QR-Code-Scanner wurde aus drei Gründen nicht integriert.

1. Da die Java-Applikation nicht auf dem Handy, sondern auf einem PC läuft, kann keine Kamera vorausgesetzt sein. Dazu kommt, dass sowohl bei iPhones, als auch Android-Geräten ein QR-Code-Scanner mitgeliefert wird.
2. Bei der Applikation legt die Gruppe Wert auf die Datenbanknutzung mit Benutzern und die Karte. Der Fokus liegt unserer persönlichen Abwägung nach nicht auf der Implementation eines Features, dass standardmäßig mitgeliefert wird.
3. Die Aufnahme von Informationsvideos zu den gescannten Inhalten ist unserer Gruppe zu viel Arbeit und Zeitaufwand.

# Augmented Reality

Augmented Reality kann leider von uns nicht implementiert werden, da keine Gyrosensoren oder andere Mittel vorliegen, mit denen Gerätebewegungen analysiert werden können. Die Implementierungskomplexität scheint ebenfalls enorm hoch zu sein.

# Blog

Der Blog ist aus Zeitgründen nicht umsetzbar. Zudem stellt er nicht eins der Hauptinteressen unseres Projekts dar. Der Austausch der Mitglieder muss zunächst in den Gruppen stattfinden, die man gründen kann. Der Arbeitsaufwand wäre für die Zeit des Programmierens zu groß.

Es würde ein nettes Feature für die Applikation darstellen, jedoch hat sich die Gruppe dagegen entschieden, da solche Plattformen bereits bestehen.

# Attraktionen

Bei den Attraktionen hat die Gruppe ebenfalls entschieden, dass das Einpflegen von der Daten verschiedener Hütten etc. nicht unser Hauptfokus sein sollte. Dies wäre eventuell ein Punkt, der in dem Zeitraum vom 13.09.2020 bis zur Präsentation eingarbeitet werden könnte.

# Hundebesitzer

Die Idee eines Hundebesitzer-Features, wo man seinen Standort teilen und andere „Hundebesitzer“-Standorte ansehen kann wurde nicht weiter ausgearbeitet, da man sich dort mit rechtlichen Wissen zum Standort-Tracking beschäftigen muss.

# Mountainbiking

Mountainbiking wurde, ebenfalls wie Wandern, ebenfalls mit in die Applikation mit übernommen, da Wandern und Mountainbiking die Hauptbeschäftigungen im Pfalzer Wald darstellen. Somit entschied die Gruppe, soll dies annähernd ähnlich wie die Wanderfunktion implementiert werden.

Auch hier steht noch aus, in welchem Maße dies von uns ausimplementiert werden kann.

# Geocaching

Da auch beim Geocaching Standorte verwenden werden, haben wir uns mit der Implementation dieses Themas nicht weiter auseinandergesetzt, da wir uns dort rechtlich nicht auskennen. Die Idee finden wir trotzdem so gut, dass sie mit in den Prototypen übernommen wurde.

# Livestream-Möglichkeit

Die Livestream-Möglichkeit für Menschen mit Behinderung etc. wurde auch nicht ausgearbeitet, da die Live-Video-Übertragung über Java zu komplex zum Implementieren erscheint.

# Internationalisierung

Die Internationalisierung wird leider nicht umgesetzt werden können.

*Wir prüfen derzeit, ob die theorethische Internationlisierung mit ins UML-Diagramm übernommen wird.*

Wie man sehen kann, ist der Prototyp komplett auf deutsch. Um die Region der Süd-West-Pfalz aber auch Touristen (touristisch gut erschlossenes Heidelberg ist in der Nähe) zugänglich zu machen sollte die Applikation auch in einer englischen Version vorhanden sein. Dies sollte mithilfe mehrerer Ressource-Klassen gehandhabt werden die Beispielsweise so heißen: „resources/HelloWorld\_en.properties“

„resources/HelloWorld\_de.properties“

Dabei können die UI-String-Werte aus diesen Klassen geladen werden und je nach Systemsprache wird hier dann die …en.properties-Datei oder die …de.properties-Datei gewählt, die dann die englischen oder deutschen Strings enthalten haben.