**Dokumentation**

Swiss Transport

|  |  |
| --- | --- |
| Inhalt | **Dokumentation Swiss Transport** |
| Erstelldatum | **Dienstag, 26. November 2019** |
| Änderungsdatum | **Mittwoch, 4. Dezember 2019** |
| Autor | **Janis Kneubühler** |

**Inhaltsverzeichnis**

[1 Einleitung 3](#_Toc26359977)

[1.1 Aufgabenstellung 3](#_Toc26359978)

[1.2 Anforderungen des Kunden 3](#_Toc26359979)

[1.3 Zweck des Dokumentes 4](#_Toc26359980)

[2 Programmierrichtlinien 4](#_Toc26359981)

[2.1 Naming 4](#_Toc26359982)

[2.2 Declaration 4](#_Toc26359983)

[2.3 Comments 4](#_Toc26359984)

[3 Planung 5](#_Toc26359985)

[3.1 Use-Cases 5](#_Toc26359986)

[3.2 Mockups 6](#_Toc26359987)

[4 Funktionen des Programmes 6](#_Toc26359988)

[4.1 Vollständig umgesetzt 6](#_Toc26359989)

[4.2 Teilweise umgesetzt 6](#_Toc26359990)

[4.3 Nicht umgesetzt 6](#_Toc26359991)

[4.4 Bugs 6](#_Toc26359992)

[5 Systemtests 6](#_Toc26359993)

[6 Testprotokoll 6](#_Toc26359994)

[7 Installationsanleitung 6](#_Toc26359995)

Es wird keine Dokumentation im eigentlichen Sinne verlangt. Es braucht jedoch im Minimum ein Dokument,

das die folgenden Informationen enthält.

OK - Autor, Datum.

- Eine Einleitung (Management Summary).

- Zweck des Dokuments.

- Was (d.h. welche Funktionen) wurde umgesetzt?

- Falls bestimmte Funktionen nur teilweise umgesetzt wurden: Welche? Welcher Teil der

Funktionalität fehlt noch? Bekannte Fehler/Bugs?

- Mockups für alle Formulare/Dialoge (GUI) , die benötigt werden um die Anforderungen A001, A002

und A003 abzudecken.

- Use Cases, Use Case Beschreibungen und Aktivitätendiagramme mindestens für die mit Priorität 1

klassifizierten Anforderungen.

- Testfälle (Systemtests), verständlich und eindeutig nachvollziehbar geschrieben, so dass ein Tester

diese ohne weiteren Erklärungen durchführen kann.

- Testprotokoll mit Durchführungsdatum der Tests, Name des Testers und Protokollierung der

tatsächlichen Testresultate.

- Installationsanleitung: Wie wird die Software installiert? Wie wird die Software deinstalliert?

- Andere spannende Informationen für die Bewertung.

Die Dokumentation soll sauber gestaltet, inhaltlich korrekt und übersichtlich strukturiert sein. Ein

Inhaltsverzeichnis soll vorhanden sein. Achte auf Typographie und Rechtschreibung!

# Einleitung

## Aufgabenstellung

In dieser Arbeit wird ein Programm erstellt, welches ÖV-Verbindungen anzeigen soll. Diese Verbindungen werden mithilfe der vom ÜK zur Verfügung gestellten API gesucht. Der Kunde hat verschiedene Anforderungen, die das Programm haben soll, aufgeschrieben und priorisiert. Diese sind im Punkt 1.1 aufgeführt. Welche Anforderungen eingebaut werden, ist jedem selber überlassen. Jedoch sollten diese mit der Priorität 1 sicher vorhanden sein! Zudem können aber auch noch eigene Ideen eingebaut werden.  
Bis am Dienstag, 3. Dezember, möchte ich sicher mit den Anforderungen, mit der Priorität 1, fertig sein. Danach geht es an die Verschönerung des GUIS, an das Testen des Programms und ans Schreiben der Dokumentation. Erst dann werden zusätzliche Funktionen ins Programm eingebaut.  
Am Schluss möchte ich ein Programm ohne Bugs haben, falls nicht alle Funktionen eingebaut sind, ist dies kein Problem. Mir ist es wichtiger, dass die Qualität des Codes stimmt und dafür das Programm nicht ganz so umfangsreich ist.

## Anforderungen des Kunden

|  |  |
| --- | --- |
| Beschreibung | Priorität |
| Als ÖV-Benutzer möchte ich Start- und Endstation mittels Textsuche suchen können, damit ich nicht alle Stationsnamen auswendig lernen muss. | 1 |
| Als ÖV-Benutzer möchte ich die aktuellen, d.h. mindestens die nächsten vier bis fünf Verbindungen zwischen den beiden gefundenen und ausgewählten Stationen sehen, damit ich weiss wann ich zur Station muss, um den für mich idealen Anschluss zu erwischen. | 1 |
| Als ÖV-Benutzer möchte ich sehen, welche Verbindungen ab einer bestimmten Station vorhanden sind, damit ich bei mir zuhause eine Art Abfahrtstafel haben kann. | 1 |
| Als ÖV-Benutzer möchte ich, dass schon während meiner Eingabe erste Suchresultate erscheinen, damit ich effizienter nach Stationen suchen kann. | 2 |
| Als ÖV-Benutzer möchte ich nicht nur aktuelle Verbindungen suchen können, sondern auch solche zu einem beliebigen anderen Zeitpunkt, damit ich zukünftige Reisen planen kann. | 2 |
| Als ÖV-Benutzer möchte ich sehen, wo sich eine Station befindet, damit ich mir besser vorstellen kann, wie die Situation vor Ort aussieht. | 3 |
| Als ÖV-Benutzer möchte Stationen finden, die sich ganz in der Nähe meiner aktuellen Position befinden, damit ich schnell einen Anschluss erreichen kann. | 3 |
| Ich möchte meine gefundenen Resultate via Mail weiterleiten können, damit auch andere von meinen Recherchen profitieren können. | 3 |

## Zweck des Dokumentes

Dieses Dokument zeigt, was im ÜK 318 alles gemacht wurde. Es soll einerseits zeigen, welche Funktionen und Fehler die Swiss Transport App. Aber auch Programmierrichtlinien, Test-Cases und die ganze Planung befinden sich in diesem Dokument.

# Programmierrichtlinien

## Naming

|  |  |  |
| --- | --- | --- |
| Stelle | REgel | Beispiel |
| Allgemein | Namen auf Englisch, aussagen welchen Zweck sie haben,Camel Case |  |
| Variablen | Anfang klein, lokale und Membervariablen werden nicht unterschieden | numberOfConnections |
| Methoden | Anfang gross | AreStationFieldsFilled() |
| Klassen | Gross | Station |
| GUI | Vor Name Kürzel des GUI-Elements (Angegeben in Modul 305 Dokument) | btnSearchConnections |

## Declaration

|  |  |
| --- | --- |
| REgel | Beispiel |
| Variablen: am Anfang deklarieren und im Konstruktor definieren | int numberOfConnections;  Konstruktor() {  numberOfConnections = 0  } |

## Comments

|  |  |
| --- | --- |
| REgel | Beispiel |
| Code soll durch richtiges naming selbsterklärend sein -> keine comments im code, ausser bei sehr komplizierten methoden | int numberOfConnections;  Konstruktor() {   numberOfConnections = 0  } |

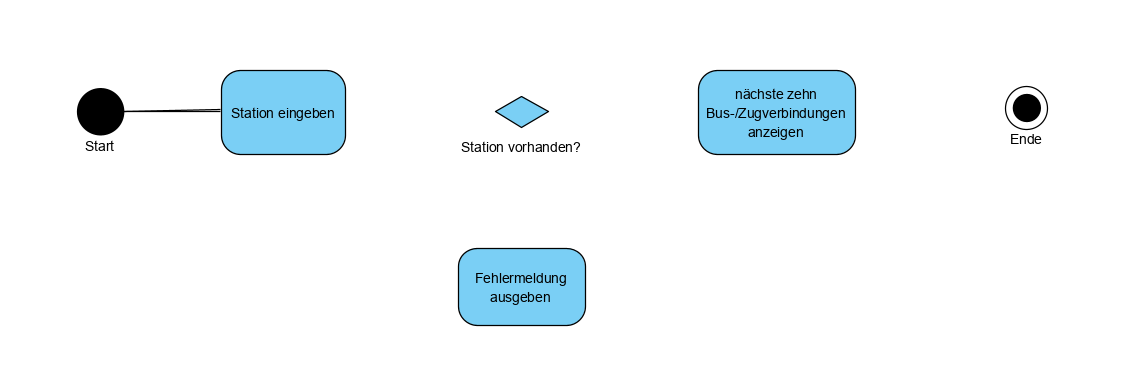
# Planung

## Use-Cases

|  |  |
| --- | --- |
| Use Case | Anschlussmöglichkeiten anzeigen |
| Beschreibung | Dem Benutzer werden alle Anschlussmöglichkeiten einer Station angezeigt wie dies auch bei Abfahrtstafeln an den Bahnhöfen der Fall ist. Es sollen alle Verbindungen, egal in welche Richtung, welche an dieser Station anhalten angezeigt werden. |
| Akteur | Benutzer des Programms (meist ÖV-Reisende) |
| Vorbedingung | Internetverbindung |
| Ablauf | 1. Anschlussmöglichkeiten anzeigen anklicken 2. Station eingeben 3. Suchen drücken |
| Ergebnis | Alle Verbindungen, welche in nächster Zeit in dieser Station verfügbar sind, werden angezeigt. Egal wohin diese führen. |
| Erweiterungen | - |

|  |  |
| --- | --- |
| Use Case | Verbindungen suchen |
| Beschreibung | Dem Benutzer werden die nächsten fünf ÖV-Verbindungen zwischen eingegebenen Start- und Zielort aufgelistet. |
| Akteur | Benutzer des Programms (meist ÖV-Reisende) |
| Vorbedingung | Internetverbindung |
| Ablauf | 1. Verbindungen suchen anklicken 2. Start- und Zielort eingeben 3. Suchen drücken |
| Ergebnis | Die nächsten fünf Verbindungen mit Uhrzeiten, welche vom Start- bis zum Zielort führen, werden angezeigt. |
| Erweiterungen | Zusätzlich zu Start- und Zielort soll auch noch die Abfahrtszeit angegeben werden, damit auch Verbindungen in Zukunft nachgeschaut werden können |

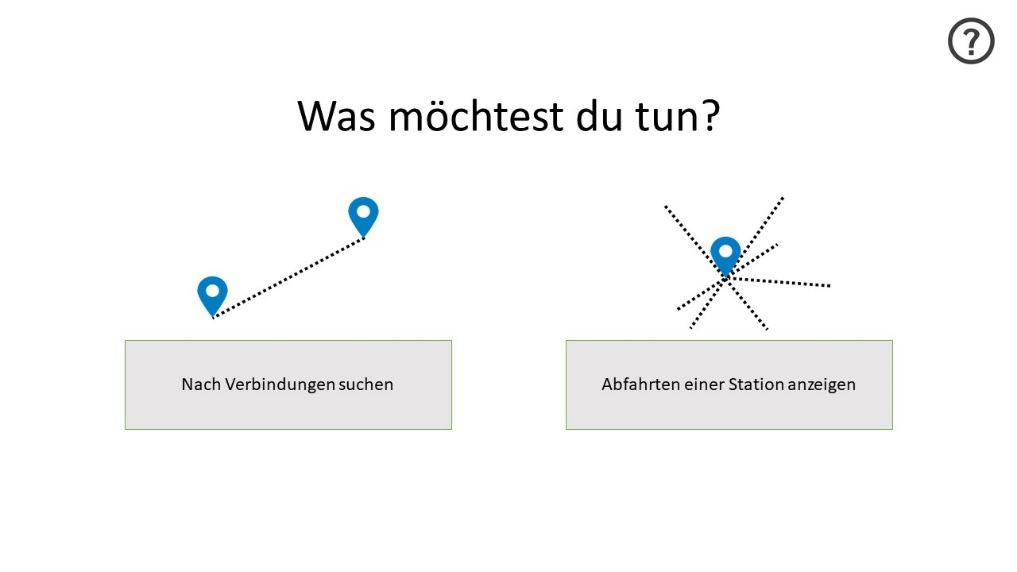
|  |  |
| --- | --- |
| Use Case | Optimierte Stationssuche |
| Beschreibung | Damit die Suche nach Stationen dem Benutzer leichtfällt, soll nicht exakt nach dem richtigen Stationsnamen gesucht werden müssen. Die Stationen sollen auch schon gefunden werden wenn nur ein Teil davon steht. |
| Akteur | Benutzer des Programms (meist ÖV-Reisende) |
| Vorbedingung | Internetverbindung |
| Ablauf | 1. Nach einer Station suchen (integriert in Use Cases Verbindungen suchen & Anschlussmöglichkeiten anzeigen) 2. Text eingeben 3. Stationen werden aufgelistet |
| Ergebnis | Die Station wird trotzdem gefunden, auch wenn diese nicht exakt so eingegeben wurde wie sie richtig heisst. Das bedeutet, wenn der Benutzer z.B. nach «Ufhusen Käserei» sucht, wird «Ufhusen Käserei Dorf» gefunden. |
| Erweiterungen | Wenn der Benutzer einen Text in die Suchleiste eingibt, wird bereits nach möglichen Stationen gesucht und diese werden vorgeschlagen. |



## Mockups

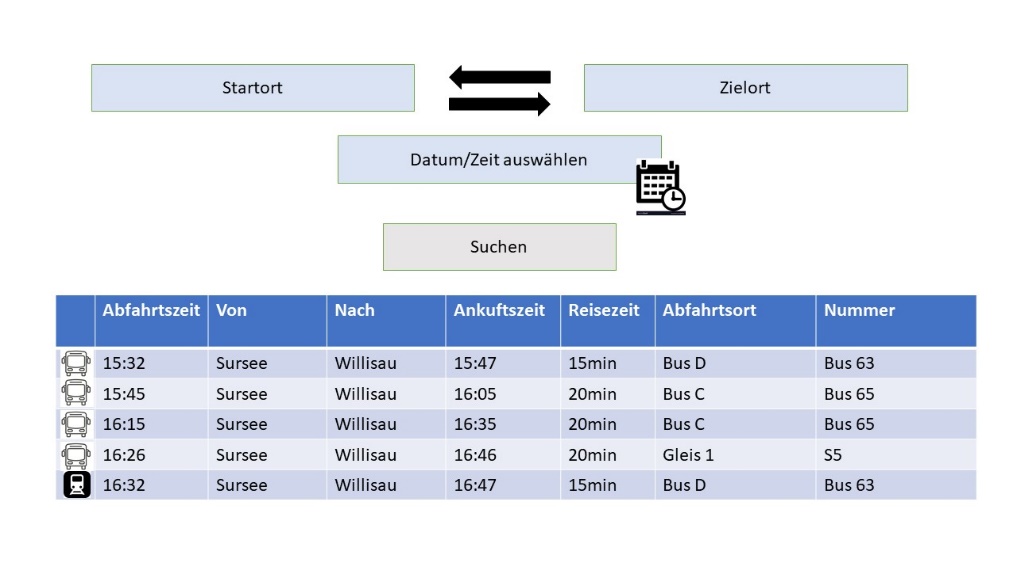
Ich habe die Designs ziemlich ähnlich wie Mockup gemacht. Für die Icons in den Tabellen hat mir leider die Zeit nicht gereicht. Zudem ist das Startmenü nicht wirklich gut herausgekommen. Dies habe ich aber auch schon im Mockup so designt.

### Startmenü



### Verbindungen

Dieses Fenster ist mir am besten gelungen. Ausser den Farben und den «Datum/Zeit» Knopf ist es ziemlich ähnlich zum Programm.



### Abfahrten

Abgesehen der Icons und den Farben ist auch dieses ziemlich ähnlich zum GUI im Programm.



# Funktionen des Programmes

## Vollständig umgesetzt

## Teilweise umgesetzt

## Nicht umgesetzt

## Bugs

# Systemtests

# Testprotokoll

# Installationsanleitung