2020

Carolina Sanchez

CSS-Versicherung

22.4.2020

M318: SwissTransport Projektarbeit



Inhaltsverzeichnis

[Einleitung 3](#_Toc38636859)

[Diagramme 3](#_Toc38636860)

[Use Case Diagram 3](#_Toc38636861)

[Aktivitätendiagram 3](#_Toc38636862)

[Aktivitätendiagram Anforderung 001 3](#_Toc38636863)

[Aktivitätendiagram Anforderung 002 4](#_Toc38636864)

[Aktivitätendiagram Anforderung 003 4](#_Toc38636865)

[Code Konventionen 4](#_Toc38636866)

[Klammern 4](#_Toc38636867)

[Tabulatoren 4](#_Toc38636868)

[Kommentare 4](#_Toc38636869)

[Namenskonventionen 5](#_Toc38636870)

[Mockups 6](#_Toc38636871)

[Mockup Anforderung 001 6](#_Toc38636872)

[Mockup Anforderung 002 6](#_Toc38636873)

[Mockup Anforderung 003 7](#_Toc38636874)

[Erledigte Anforderungen 7](#_Toc38636875)

[A001 7](#_Toc38636876)

[Implementierung 7](#_Toc38636877)

[A002 8](#_Toc38636878)

[Implementierung 8](#_Toc38636879)

[A003 8](#_Toc38636880)

[Implementierung 8](#_Toc38636881)

[A004 8](#_Toc38636882)

[Implementierung 9](#_Toc38636883)

[A005 9](#_Toc38636884)

[Implementierung 9](#_Toc38636885)

[A006 10](#_Toc38636886)

[Implementierung 10](#_Toc38636887)

[Tests 11](#_Toc38636888)

[A001 11](#_Toc38636889)

[A002 11](#_Toc38636890)

[A003 11](#_Toc38636891)

[A004 11](#_Toc38636892)

[A005 12](#_Toc38636893)

[A006 12](#_Toc38636894)

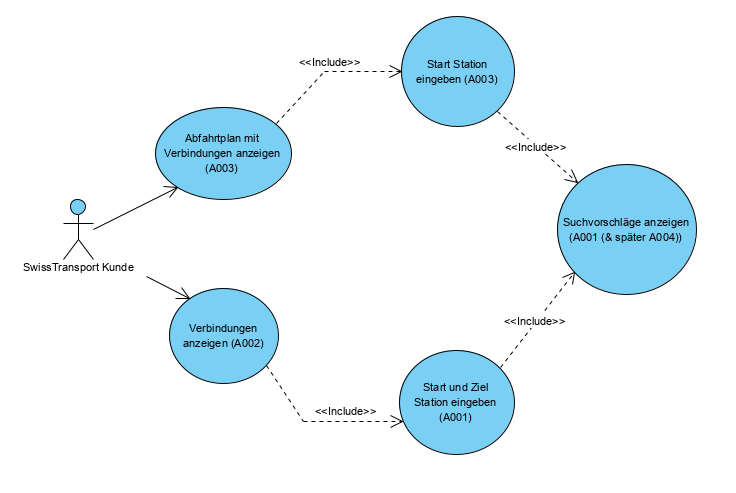
# Einleitung

Im ÜK vom Modul 318 wurde uns aufgetragen das Leben der Leute durch eine App zu erleichtern. Diese Swiss Transport App soll die Daten des ÖV-Fahrplans abrufen können und die dann anschliessend einfach anzeigen lassen.

# Diagramme

## Use Case Diagram

Dieses Use Case Diagramm wurde für die Anforderungen 001-002 erstellt.



## Aktivitätendiagram

### Aktivitätendiagram Anforderung 001

### Aktivitätendiagram Anforderung 002

### Aktivitätendiagram Anforderung 003

# Code Konventionen

Das Coding Standarddokument war unsere Richtlinie, wie wir unseren Code schreiben mussten. Somit ist der Code einfacher zu verstehen ist und jeder Developer kann den Code problemlos lesen und verstehen.

Hier waren die Code Konventionen, auf die ich besonders Wert gelegt habe:

## Klammern

Die Klammern sind mit C# folgendermassen anzuwenden.

public void Method()

{

// Hier kommt der Code hin.

}

## Tabulatoren

Es sollte immer Tabulatoren benutzt werden. Ein Tab ist gleich wie 4 Leerzeichen.

## Kommentare

Die Kommentare werden immer oberhalb der betroffenen Code-Zeile geschrieben. Der Anfangsbuchstabe ist gross zu schrieben und ein Punkt soll der Abschluss sein wie bei einer normalen Satzstruktur.

Beispiel:

// Dies ist ein Kommentar.

Für die Kommentare, die ich oft benutzt habe in diesem Projekt, um zu beschreiben was meine Methoden machten, habe ich die der XML-Dokumentation von C# verwendet.

Beispiel aus meinem Code für die Methode GetStationSuggestions:

/// <summary>

/// Method provides Station suggestions.

/// </summary>

/// <param name="CurrentQuery"></param>

/// <returns> AutoCompleteStringCollection </returns>

Hier sieht man auch welche Parameter die Methode mitgegeben bekommt und welchen Rückwert sie liefert.

# Namenskonventionen

Bei C# wird eine Mischung von den Notationen ‘PascalCase’ und ‘CamelCase’ gebraucht. Es werden keine Underscores/Underlines ausser bei den Field Namen verwendet.

Dazu eine Liste der verwendeten Notationen:

PascalCase:

* Klassen
* Konstruktoren
* Kontante
* Eigenschaften
* Enums

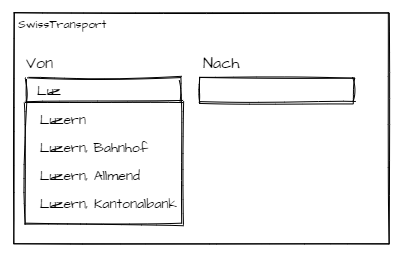
CamelCase:

* Methoden
* Lokale Variablen
* GUI-Komponente

# Mockups

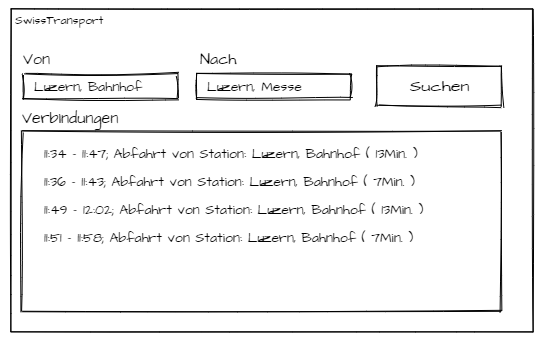
Mockups/Wireframes werden vor dem Programmieren selbst gemacht, um die gewünschten Anforderungen richtig umzusetzen.

## Mockup Anforderung 001



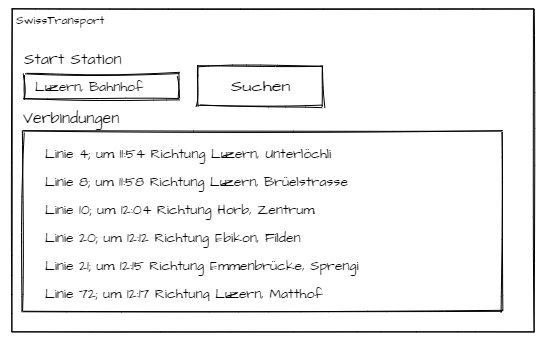
Hier sieht man TextBoxen, in die man die Von- und Nachstation schreibt und die Vorschläge, die während der Eingabe angezeigt werden sollte.

## Mockup Anforderung 002



Hier sieht man Textboxen, in die man die Von- und Nachstation schreibt, den Suchbutton und dann die gefunden Verbindungen.

## Mockup Anforderung 003



Hier sieht man TextBoxen, in die man die Start Station schreibt, den Suchbutton und dann die gefunden Verbindungen von der gesuchten Station aus.

# Erledigte Anforderungen

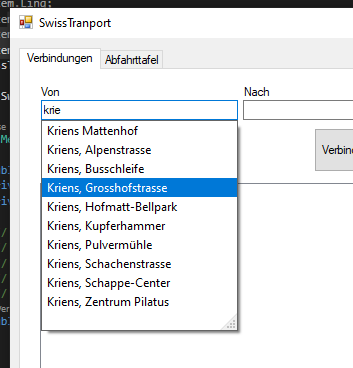
Für das Programm SwissTransport zu schreiben, haben wir 8 Anforderungen erhalten und ich habe die 6 folgenden implementiert.

## A001

Als ÖV-Benutzer möchte ich Start- und Endstation mittels Textsuche suchen können, damit ich nicht alle Stationsnamen auswendig lernen muss.

### Implementierung

Nach 3 Zeichen wird eine Liste mit Stationsvorschlägen angezeigt, die man auch auswählen kann.

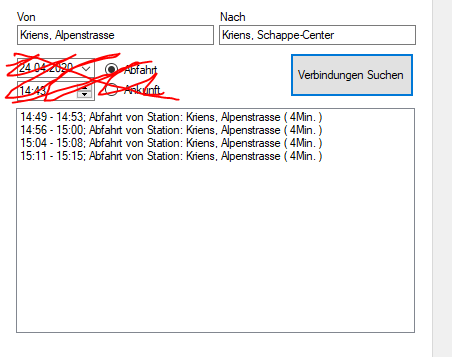


## A002

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.

### Implementierung

Es werden wie verlangt 4 Verbindungen angezeigt.

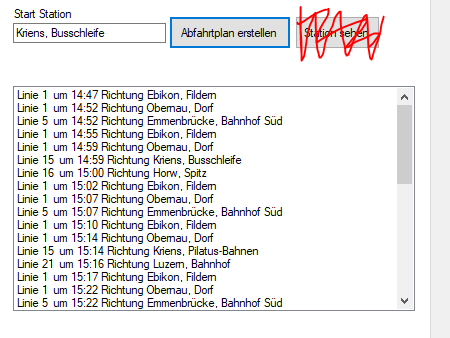


## A003

Als ÖV-Benutzer möchte ich sehen, welche Verbindungen ab einer bestimmten Station vorhanden sind, damit ich bei mir zuhause eine Art Abfahrtstafel haben kann.

### Implementierung

Es werden alle Verbindungen ab der angegebenen Start Station angezeigt.

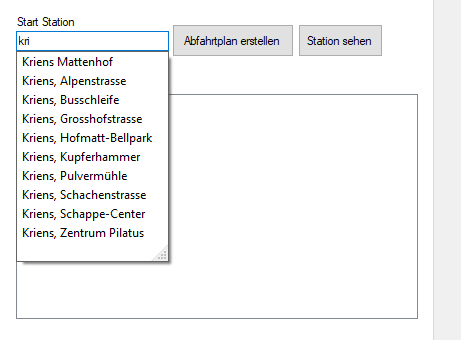


## A004

Als ÖV-Benutzer möchte ich, dass schon während meiner Eingabe erste Suchresultate erscheinen, damit ich effizienter nach Stationen suchen kann.

### Implementierung

Es wird nach 3 Zeichen eine Liste mit Stationsvorschlägen angezeigt.

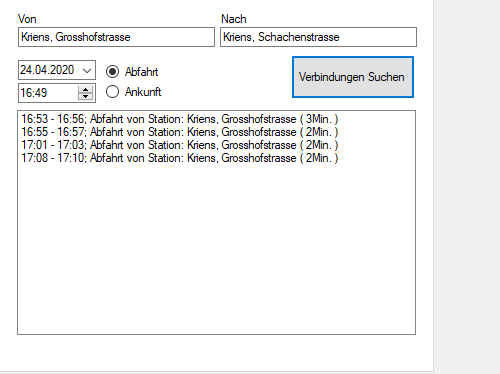


## A005

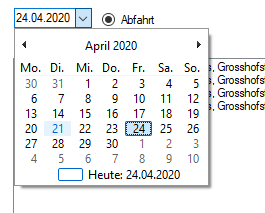
Als ÖV-Benutzer möchte ich nicht nur aktuelle Verbindungen suchen können, sondern auch solche zu einem beliebigen anderen Zeitpunkt, damit ich zukünftige Reisen planen kann.

### Implementierung

Man kann einen beliebigen Tab mit der beliebigen Zeit als Ankunft oder Abfahrt (standartmässig) eingeben und die Verbindungen dazu werden nach der Reihenfolge angezeigt.



Das Auswählen der Zeit funktioniert mit eins addieren (Pfeil gegen oben) oder eins subtrahieren (Pfeil gegen unten) für die Stunden und die Minuten. Das Auswählen des Tages erfolgt dagegen mit einem Kalender, auf dem man den Tag anklicken kann.

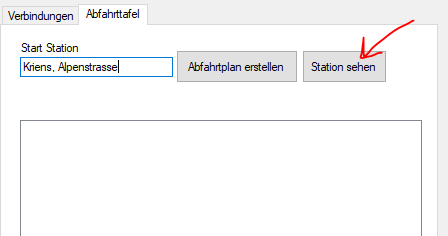


## A006

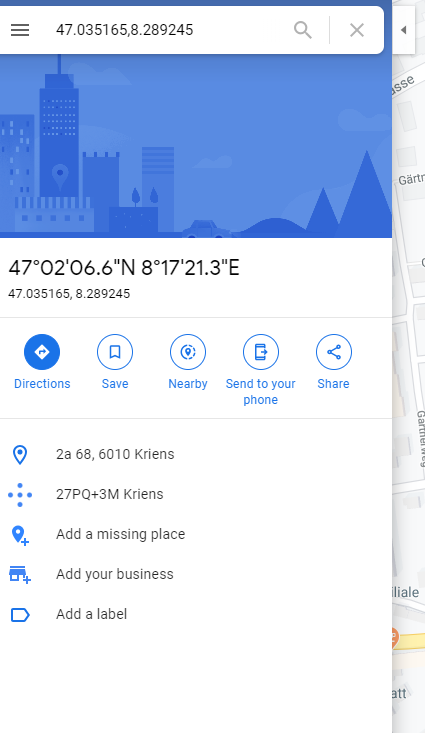
Als ÖV-Benutzer möchte ich sehen, wo sich eine Station befindet, damit ich mir besser vorstellen kann, wie die Situation vor Ort aussieht.

### Implementierung

Sobald auf ‘Station anzeigen’ geklickt wird, öffnet sich der Browser mit der Google Maps Karte und dem Standort der Station.



Rechts wird die Station auf der Karte rot markiert und links wird mit den Koordinaten gesucht.



# Tests

## A001

|  |  |  |  |
| --- | --- | --- | --- |
| **Test:** | **Aktion:** | **Erwartet:** | **Resultat:** |
| 1 | Der Benutzer gibt die 3 Anfangsbuchstaben einer vorhandenen Station ein. | Es werden viele Stationsvorschläge angezeigt, die man auswählen kann. | Wie erwartet. |
| 2 | Der Benutzer gibt die 3 Anfangsbuchstaben einer nicht vorhandenen Station ein. | Es geschieht nichts. | Wie erwartet. |

## A002

|  |  |  |  |
| --- | --- | --- | --- |
| **Test:** | **Aktion:** | **Erwartet:** | **Resultat:** |
| 1 | Der Benutzer sucht Verbindungen von 2 gültigen Stationen mit vorhandenen Verbindungen. | Es werden 4 Verbindungen zwischen den eingegebenen Stationen angezeigt. | Wie erwartet. |
| 2 | Der Benutzer sucht nach ungültigen Stationen. | Es wird eine Fehlermeldung ausgegeben und nichts wird angezeigt. | Wie erwartet. |
| 3 | Der Benutzer sucht nach gültigen Stationen ohne Verbindungen dazwischen. | Es wird nichts angezeigt. | Wie erwartet. |

## A003

|  |  |  |  |
| --- | --- | --- | --- |
| **Test:** | **Aktion:** | **Erwartet:** | **Resultat:** |
| 1 | Der Benutzer sucht nach einer gültigen Station. | Es werden alle möglichen Verbindungen angezeigt. | Wie erwartet. |
| 2 | Der Benutzer sucht nach einer ungültigen Station. | Es wird eine Fehlermeldung ausgegeben und nichts wird angezeigt. | Wie erwartet. |

## A004

|  |  |  |  |
| --- | --- | --- | --- |
| **Test:** | **Aktion:** | **Erwartet:** | **Resultat:** |
| 1 | Der Benutzer gibt die ersten 3 Zeichen einer vorhandenen Station ein. | Es werden viele Stationsvorschläge, die man auswählen kann, angezeigt. | Wie erwartet. |
| 2 | Der Benutzer gibt die ersten 3 Zeichen einer nicht vorhandenen Station ein. | Es geschieht nichts. | Wie erwartet. |

## A005

|  |  |  |  |
| --- | --- | --- | --- |
| **Test:** | **Aktion:** | **Erwartet:** | **Resultat:** |
| 1 | Der Benutzer sucht nach Verbindungen zu einem bestimmten Zeitpunkt.  24.04.2020 um 11:45 | Es werden 4 Verbindungen nach den angegebenen Zeitkriterien, also ab 11:45 vom 24.04.2020 angezeigt. | Wie erwartet. |
| 2 | Der Benutzer sucht nach Verbindungen zu keinem bestimmten Zeitpunkt. | Es werden 4 Verbindungen zu der aktuellen Zeit angegeben. | Wie erwartet. |

## A006

|  |  |  |  |
| --- | --- | --- | --- |
| **Test:** | **Aktion:** | **Erwartet:** | **Resultat:** |
| 1 | Der Benutzer gibt eine gültige Start Station ein und klickt auf Station anzeigen. | Es wird der Browser mit Google Maps und den Koordinaten der gesuchten Station geöffnet. | Wie erwartet. |
| 2 | Der Benutzer gibt eine ungültige Start Station ein und klickt auf Station anzeigen. | Es wird eine Fehlermeldung ausgegeben. | Wie erwartet. |