Leuchter AG Eingereicht bei: Markus Estermann Eingereicht am 04.12.2018

ÖV-Application

Francesco Casamassa S-Inf 17dL Tulpenweg 8b 6060 Sarnen

Inhalt

[1 Einführung 3](#_Toc531702101)

[1.1 Einleitung 3](#_Toc531702102)

[1.2 Zweck 3](#_Toc531702103)

[1.3 Ziel 3](#_Toc531702104)

[2 Funktionen 3](#_Toc531702105)

[2.1 angeforderte Funktionen 3](#_Toc531702106)

[2.2 umgesetzte Funktionen 4](#_Toc531702107)

[2.3 teilweise umgesetzte Funktionen 4](#_Toc531702108)

[2.4 nicht umgesetzte Funktionen 4](#_Toc531702109)

[3 Diagramme 5](#_Toc531702110)

[3.1 Klassendiagramm 5](#_Toc531702111)

[3.2 UseCase Diagramm und Beschreibung 5](#_Toc531702112)

[3.3 Aktivitätsdiagramm 7](#_Toc531702113)

[3.4 MockUp 8](#_Toc531702114)

[4 Testfälle 9](#_Toc531702115)

[4.1 allgemeine Vorbedingungen 9](#_Toc531702116)

[4.2 Testfall «Station suchen» 9](#_Toc531702117)

[4.2.1 Testszenario 9](#_Toc531702118)

[4.3 Testfall «Verbindung zwischen zwei Stationen suchen» 9](#_Toc531702119)

[4.3.1 Testszenario 9](#_Toc531702120)

[4.4 Testfall «Verbindung ab einer Station suchen» 9](#_Toc531702121)

[4.4.1 Testszenario 9](#_Toc531702122)

[5 Code Guidelines 10](#_Toc531702123)

[5.1 Naming Conventions 10](#_Toc531702124)

[5.2 Comments 10](#_Toc531702125)

[5.3 Declarations 10](#_Toc531702126)

[6 Guide 11](#_Toc531702127)

[6.1 Applikation installieren 11](#_Toc531702128)

[6.2 Applikation deinstallieren 12](#_Toc531702129)

[6.2.1 Mit Windows 12](#_Toc531702130)

[6.2.2 Mit erneuten Setupausführung 12](#_Toc531702131)

[7 Routenplaner 13](#_Toc531702132)

[8 Bildverzeichnis 13](#_Toc531702133)

# Einführung

## Einleitung

In diesem Kurs mussten wir eine «ÖV-Application» erstellen, welche mit einer externen API kommuniziert. Diese API wird von OpenData verwaltet und gepflegt. Wir sollten eine Applikation erstellen, die Aufrufe an eine API macht und Verbindungen, Stationsnamen etc. zurück liefert.

## Zweck

Die Applikation sollte einige Anforderungen erfüllen, die später im Dokument erwähnt werden, welche und vorgegeben wurden. Wir sollten diese Anforderungen so gut wie möglich umsetzten.

## Ziel

Mein persönliches Ziel der Arbeit war, dass ich versuche so viele Anforderung, wie möglich, erfülle und diese mit Exception-Handling zu sichern, sodass das Programm nicht abstürzten kann.

# Funktionen

## angeforderte Funktionen

|  |  |  |
| --- | --- | --- |
| ID | Beschreibung | Priorität |
| A001 | Als ÖV-Benutzer möchte ich Start- und Endstation mittels Textsuche suchen können, damit ich nicht alle Stationsnamen auswendig lernen muss. | 1 |
| 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. | 1 |
| A003 | Als ÖV-Benutzer möchte ich sehen, welche Verbindungen ab einer bestimmten Station voran den sind, damit ich bei mir zuhause eine Art Abfahrtstafel haben kann. | 1 |
| A004 | Als ÖV-Benutzer möchte ich, dass schon während meiner Eingabe erste Such-Resultate erscheinen, damit ich effizienter nach Stationen suchen kann. | 2 |
| 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. | 2 |
| A006 | Als ÖV-Benutzer möchte ich sehen, wo sich eine Station befindet, damit ich mir besser vorstellen kann, wie die Situation vor Ort aussieht. | 3 |
| A007 | Als ÖV-Benutzer möchte Stationen finden, die sich ganz in der Nähe meiner aktuellen Position befinden, damit ich schnell einen Anschluss erreichen kann. | 3 |
| A008 | Ich möchte meine gefundenen Resultate via Mail weiterleiten können, damit auch andere von meinen Recherchen profitieren können. | 3 |

## umgesetzte Funktionen

Die Funktionen A001-A005 wurden vollstätig umgesetzt:

Die Anforderung A001 habe ich mit der Anforderung A004 verknüpft und direkt eine «autoComplete»-Methode erstellt die, wie der Name schon sagt, Vorschläge gibt (siehe Rot§). Dies ist ein «TextChanged»-Event, welches durchgehend Verbindungen erstellt und diese zurückgibt und dadurch Vorschlägt. Dies ist natürlich nicht die «schönste» Art dies zu lösen, aber mir kam keine andere Lösung in den Sinn, ausser einer Liste welche alle Einträge hat und diese filtert, aber dies würde der Performace schaden. Also habe ich mich für die Lösung mit mehreren Verbindungen entschieden.

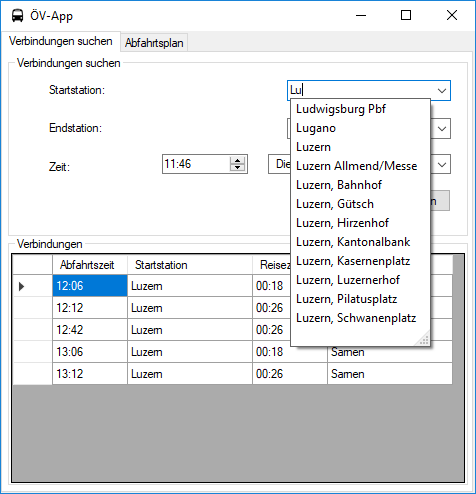


Abb. 1 ÖV-App UI

Um vier bis fünf Verbindungen zurück zu bekommen, musste ich die API abändern, dass sie mir immer fünf Verbindungen zurück liefert. Diese habe ich in eine DataGridView hinzugefügt um diese Auszugeben. Im DataGridView wird die Abfahrtszeit, Startstation, Reisezeit und Ankunftsstation ausgegeben. Ich habe diese Anforderung ebenfalls mit einer anderen Verknüpft, mit der Anforderung: A005, und habe nicht zwei DateTimePicker hinzugefügt (siehe Grün), einen für das Datum und der andere für die Zeit. Um ein DateTime als Parameter der Abfrage mitgeben zu können, habe ich die API nochmals abgeändert. Aber dieses Mal musste ich noch deinen Test anpassen.

Für die Abfahrtstafel gab es zum Glück schon eine Funktion, welche mir alle Verbindungen zu einer Station zu bekommen. Ich habe mir beim GUI-MockUp vorgestellt, dass ich ein zweites Panel mache, welche durch TabControl erreichbar ist.

## teilweise umgesetzte Funktionen

Die einzige Anforderung die nur teilweise umgesetzt wurde ist: A006.

Die Anforderung war meiner Meinung nach, nicht klar deklariert. Ich habe dies so umgesetzt, dass der Standart-Browser des Users ausgeführt wird und mit Google Maps Routenplaner eine Route mit der Start- und Endstation erstellt wird und diese anzeigt. Meiner Meinung nach ist diese Anforderung nur teilweise erfüllt, da man keine Streetview hat, wie bei Google Earth, dies ist aber Ansichtssache.

## nicht umgesetzte Funktionen

Die letzten zwei Anforderung konnte ich wegen Zeitlichen Gründen nicht implementieren. Ich habe zwar die Mailfunktion zu implementieren, aber diese funktionierte nicht, also habe ich sie vorübergehend auskommentiert, aber schlussendlich habe ich.

# Diagramme

## Klassendiagramm

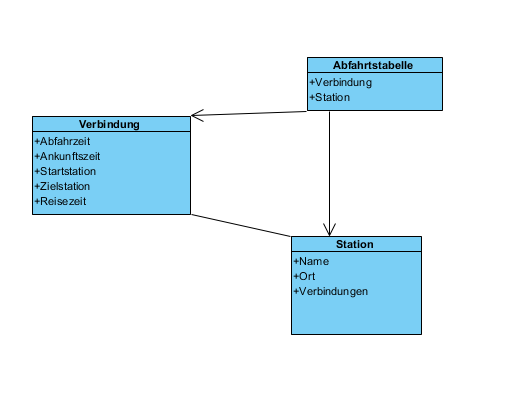


Abb. 2 Klassendiagramm ÖV-App

## UseCase Diagramm und Beschreibung

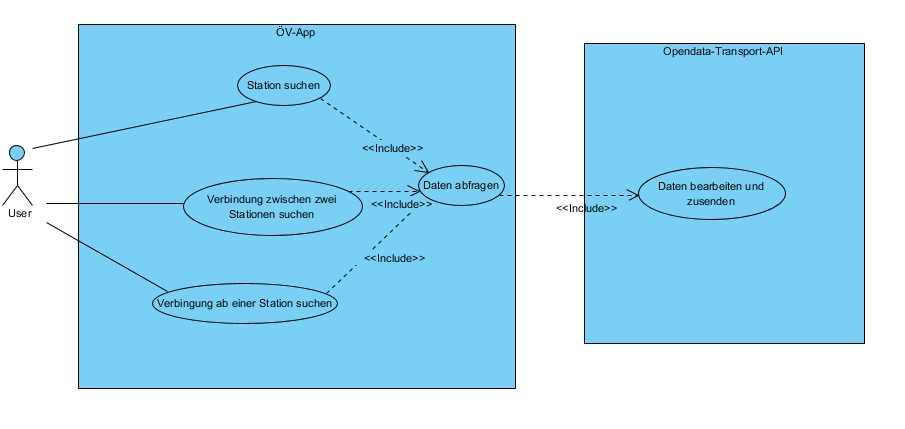


Abb. 3 UseCase-Diagramm

|  |  |
| --- | --- |
| UseCase Name | Station suchen |
| Beschreibung | Der User sucht nach dem Namen der Station |
| Vorbedingungen | Aktive Internetverbindung, ÖV-App installiert |
| Auslöser | Der Nutzer möchte den genauen Namen der Station |
| Akteur | User |
| Ablauf | 1. Programm starten 2. Text eingeben 3. Station auswählen |
| Varianten | 1. Startstation suchen  2. Endstation suchen |
| Ergebnis | Der vollständige Name der gesuchten Station |

|  |  |
| --- | --- |
| UseCase Name | Verbindung zwischen zwei Stationen suchen |
| Beschreibung | Der User gibt eine Start- und Endstation ein und erhält die Daten (Zeit, Gleis etc. )der Verbindung |
| Vorbedingungen | Aktive Internetverbindung, ÖV-App installiert |
| Auslöser | Der User möchte genaue Angaben der nächsten Verbindungen |
| Akteur | User |
| Ablauf | 1. Programm starten 2. Startstation eingeben 3. Endstation eingeben 4. Button klicken |
| Varianten | Keine |
| Ergebnis | Angaben der nächsten fünf Verbindungen |

|  |  |
| --- | --- |
| UseCase Name | Verbindung ab einer Station suchen |
| Beschreibung | Der User gibt den Namen einer Station ein und erhält sämtliche Verbindung der gesuchten Station |
| Vorbedingungen | Aktive Internetverbindung, ÖV-App installiert |
| Auslöser | Verbindungsinformationen einer Station |
| Akteur | User |
| Ablauf | 1. Programm starten 2. Stationsnamen eingeben 3. «Info» Button betätigen |
| Varianten | Keine |
| Ergebnis | Angaben der nächsten fünf Verbindungen |

## Aktivitätsdiagramm

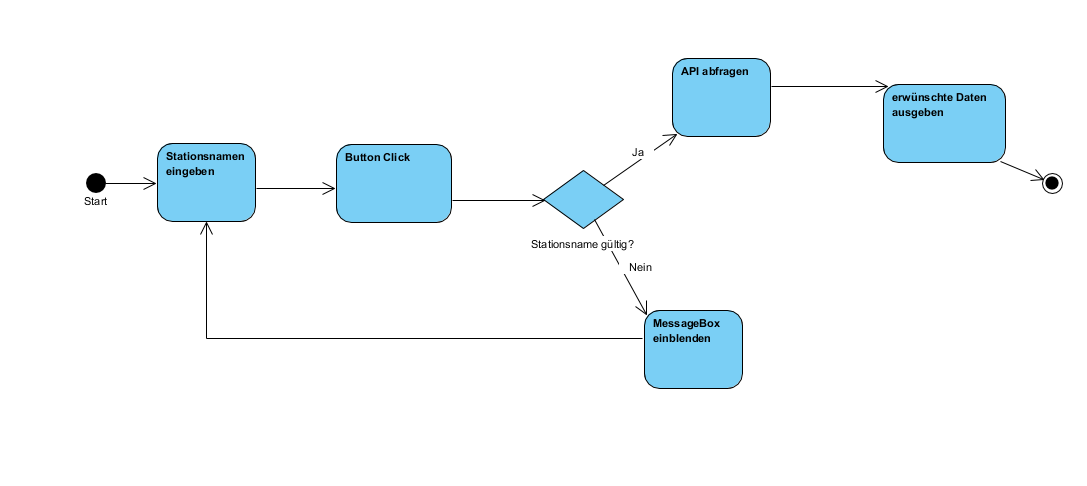


Abb. 4 Aktivitätsdiagramm für UseCase " Station suchen "

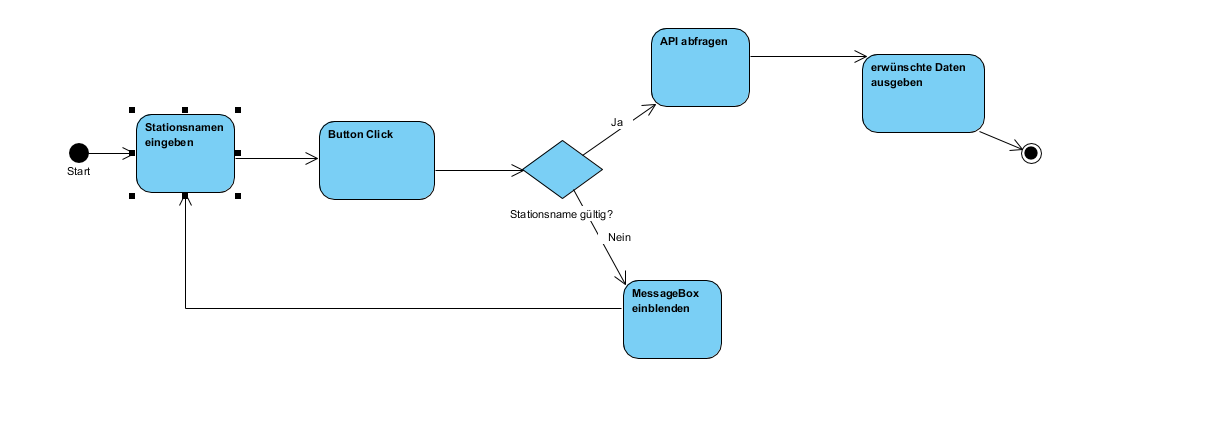


Abb. 5 Aktivitätsdiagramm für UseCase "Verbindung zwischen zwei Stationen suchen"

## MockUp

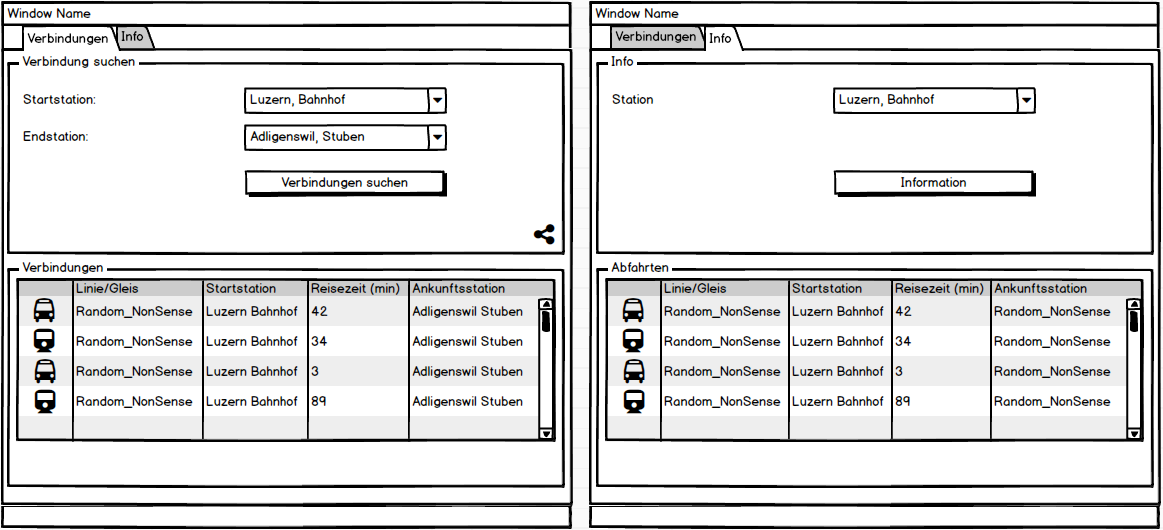


Abb. 6 erstes MockUp

Ich habe versucht mich möglichst ans GUI zu halten, was ich geschafft habe. Da ich im MockUp nur die ersten drei Anforderungen berücksichtigt habe, habe ich kein DateTimePicker. Ich habe die Icons bei der Verbindungstabelle weggelassen, da ich herausgefunden habe, dass die zur Verfügung gestellte API nicht weiss, ob es ein Bus oder Zug ist.

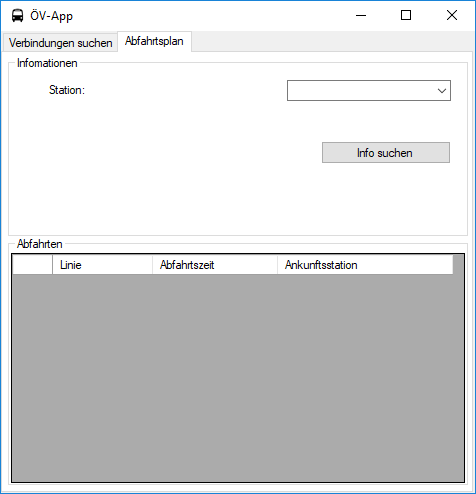
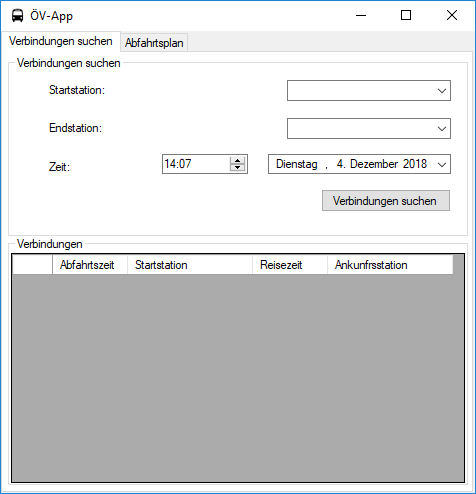


Abb. 7 MockUp Resultat

# Testfälle

## allgemeine Vorbedingungen

Da die Vorbedingungen bei allen Test die gleichen sind, habe ich sie als allgemeine Vorbedingungen definiert:

Der User muss eine aktive Internetverbindung haben und muss das Programm vollständig installiert haben.

## Testfall «Station suchen»

### Testszenario

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Programm starten | Programmstart ohne Errormeldungen |
| 2 | Text eingeben | Bei der Combobox sollte ein DropDown mit möglichen Stationen erscheinen |
| 3 | Station auswählen | Stationsname sollte in Combobox erscheinen |

## Testfall «Verbindung zwischen zwei Stationen suchen»

### Testszenario

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Programm starten | Programmstart ohne Errormeldungen |
| 2 | Startstation eingeben | Bei der Combobox sollte ein DropDown mit möglichen Stationen erscheinen |
| 3 | Endstation eingeben | Bei der Combobox sollte ein DropDown mit möglichen Stationen erscheinen |
| 4 | Button klicken | DataGridView sollte mir fünf Verbindungen gefüllt werden |

## Testfall «Verbindung ab einer Station suchen»

### Testszenario

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Programm starten | Programmstart ohne Errormeldungen |
| 2 | Stationsnamen eingeben | Bei der Combobox sollte ein DropDown mit möglichen Stationen erscheinen |
| 3 | «Info» Button betätigen |  |

# Code Guidelines

Obwohl ich keine spezifische Code-Guidelines festgelegt habe, habe ich wegen Angewohnheit an diese Punkte gehalten.

## Naming Conventions

* Schlagkräftige Variablen-, Methodennamen
* Variablen und Methoden immer mit englischen Namen
* GUI-Komponente immer abkürzen
  + Beispiele
    - Button 🡪 btn
    - DataGridView 🡪 dgv
    - Textbox 🡪 txt
    - Label 🡪 lbl
    - Etc.

## Comments

* Immer mehrzeiligen Codecoments verwenden, auch wenn nur eine Zeile kommentiert wird. (/\* Comment \*/)

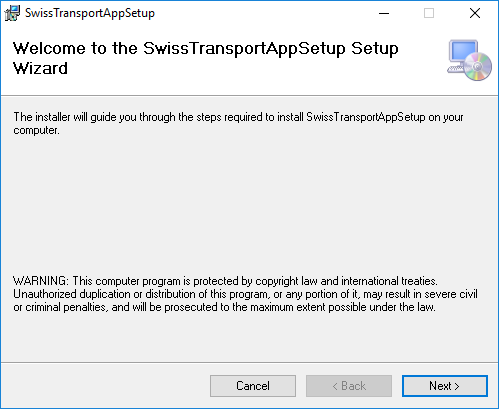
## Declarations

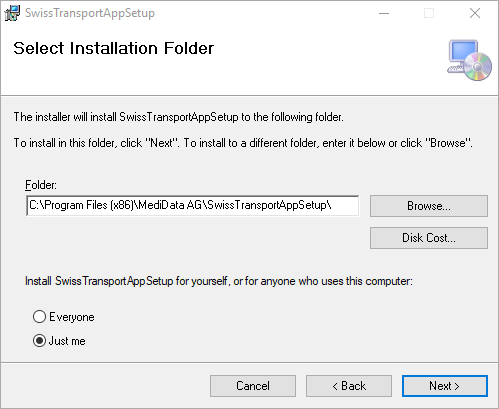
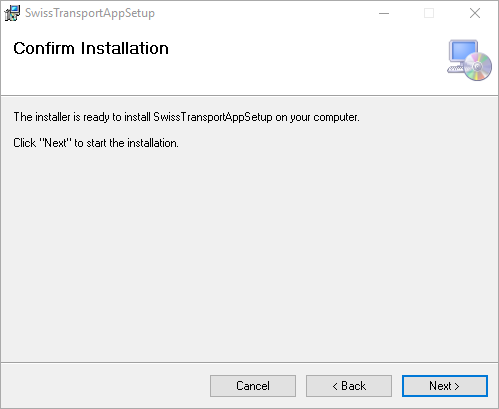
* Variablen immer nach der Methode/Statement deklarieren

# Guide

Um das Programm zu installieren braucht man Adminrechte.

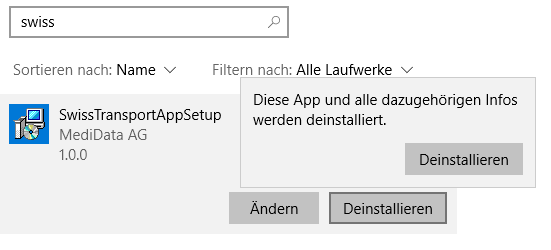
## Applikation installieren



1. Sobald das \*.exe oder \*.msi gestartet wird, wird das Startfenster erscheinen (siehe Bild). Drücken Sie auf «Next».
2. Nachdem Sie auf «Next» gedrückt haben können Sie den Installations-Ordner auswählen, bzw. den Ort beidem das Programm installiert werden sollte.
3. Nachdem Sie den Installations-Ordner ausgewählt haben, müssen Sie nur noch bestätigen und Sie sind fertig. Nach der Installation sollte eine Verknüpfung auf dem Desktop erscheinen, welche Sie ausführen können.

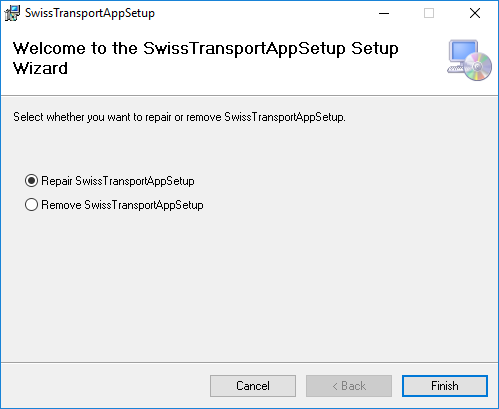
## Applikation deinstallieren

### Mit Windows

Bei ***Windows*** > ***Apps und Features.***

Kann man bei der Suche «Swiss», das Programm finden. Durch das anklicken der Applikation. Werden die zwei Knöpfe «Ändern» und «Deinstallieren» erscheinen, nach erneuten drücken des «Deinstallieren»-Knopfs (insgesamt zwei-Mal). Wird das Programm automatisch deinstalliert.

### Mit erneuten Setupausführung

Durch erneutes Ausführen des \*.exe oder \*.msi Files wird das Fenster (siehe Bild) erscheinen. Danach kann man die Option «Remove SwissTransportAppSetup» ausführen und nach dem anklicken des «Finish»-Knopfes wird die App durch erneute «Ja» drücken deinstalliert.

# Routenplaner

Ich habe einen Routenplaner für die DataGridView implementiert, sodass wenn man einen Eintrag in der DataGridView anklickt, dass der Standart-Browser gestartet wird und eine Route mit Start- und Endstation erstellt. In diesem Beispiel ist es eine Route mit Startstation 🡪 Luzern und Endstation 🡪 Sarnen.

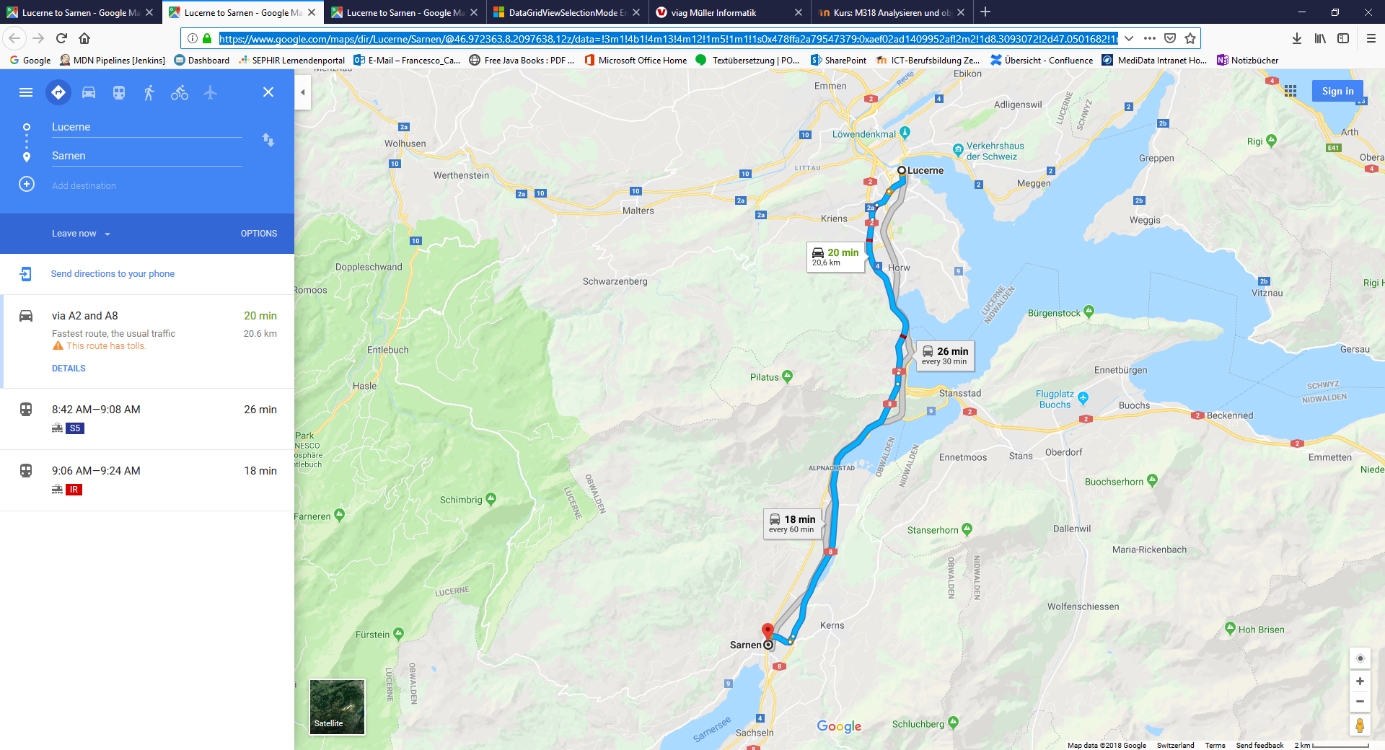


Abb. 8 Routenplaner von Luzern nach Sarnen

# Bildverzeichnis

Titelblatts Bild: <https://www.materialui.co/icon/directions-bus>