

Christen-Travel

Reisen sie sicher

Elias Christen |M118 | 21.11.2017

Inhalt

[Einleitung: 3](#_Toc499043192)

[Analyse: 3](#_Toc499043193)

[Zweck des Dokuments: 3](#_Toc499043194)

[Aktuelle Situation: 3](#_Toc499043195)

[Ziel definieren: 3](#_Toc499043196)

[Anforderungen ermitteln: 4](#_Toc499043197)

[Qualitätsziele: 6](#_Toc499043198)

[Programmierrichlinien: 7](#_Toc499043199)

[Naming Conventions: 7](#_Toc499043200)

[Declaration: 7](#_Toc499043201)

[Comments: 7](#_Toc499043202)

[Statements (New Lines, Indentation): 7](#_Toc499043203)

[Design: 7](#_Toc499043204)

[Lösung erarbeiten: 7](#_Toc499043205)

[GUI Entwerfen 1: 8](#_Toc499043206)

[Vorteile: 8](#_Toc499043207)

[Nachteil: 8](#_Toc499043208)

[GUI Entwerfen 2: 9](#_Toc499043209)

[Vorteile: 9](#_Toc499043210)

[Nachteil: 9](#_Toc499043211)

[Lösungen: 9](#_Toc499043212)

[Activity-Diagram: 10](#_Toc499043213)

[Aoo1: 10](#_Toc499043214)

[A002: 10](#_Toc499043215)

[A003: 10](#_Toc499043216)

[Use-Case-Diagram(A001-A003): 11](#_Toc499043217)

[Umsetzung: 12](#_Toc499043218)

[Umsetzung des GUI: 12](#_Toc499043219)

[Umsetung Code: 12](#_Toc499043220)

[Probleme: 12](#_Toc499043221)

[Testfälle: 13](#_Toc499043222)

[Testfall A001: 13](#_Toc499043223)

[Testfall A002: 13](#_Toc499043224)

[Testfall A003: 13](#_Toc499043225)

[Testfall A004: 14](#_Toc499043226)

[Testfall A005: 14](#_Toc499043227)

[Testfall A006: 14](#_Toc499043228)

[Einführung Programm: 15](#_Toc499043229)

[Installation: 15](#_Toc499043230)

[Hilfe: 0](file:///C:\Users\Elias\Documents\ÜK\LokalesRepo\doc\Christen-Travel.docx#_Toc499043231)

[0](#_Toc499043232)

[Fazit: 0](#_Toc499043233)

[Resultat: 0](#_Toc499043234)

# Einleitung:

Wir haben im ÜK M118 den Auftrag bekommen eine Windows Applikation zu erstellen, mit Visual Studio. Diese sollte Verschiedene Anforderungen erfüllen, welche den Grundaufbau darstellen und weiterunten definiert sind. Aber kurz gesagt uns wurde eine öffentliche API zur Verfügung gestellt mit deren Hilfe wir ein Reiseprogramm erstellen können. Wie zum Beispiel SBB.ch einfach als Windows Applikation.

# Analyse:

## Zweck des Dokuments:

Wir haben einen ÜK in dem wir das Objekt Orientierte Programmieren lernen. Der Zweck des Dokuments ist es das wir lernen wie man eine Arbeit protokolliert und ausführt. Ich habe in diesem Dokument beschrieben wie mein Arbeitsvorgehen war und wo es Probleme gab. Ausserdem sind noch Installationsanweisung und Hilfestellungen vorhanden.

## Aktuelle Situation:

* Programm welche die Fahrplandaten des Schweizerischen öffentlichen Verkehrs benutzt
* Programm mit dem man Verkehrsverbindungen zwischen zwei Stationen suchen kann
* Programm mit dem die Gewünschten Anforderungen

## Ziel definieren:

* Eine funktionierende Applikation
* Anforderungen sollen, wenn möglich erfüllt sein
* Qualität-Ziele sollen erfüllt sein
* Hohe Qualität soll vorhanden sein
* Design soll ansprechend und modern sein

## Anforderungen ermitteln:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Beschreibung | Priorität | Erfüllt |
| A001 | Als Christen-Travel-Benutzer möchte ich Start- und Endstation mittels Textsuche suchen können, damit ich nicht alle Stationsnamen auswendig lernen muss. | 1 | X |
| A002 | Als Christen-Travel-Benutzer möchte ich die aktuellen, d.h. mindestens die nächsten 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 | X |
| A003 | Als Christen-Travel-Benutzer möchte ich sehen, welche Verbindungen ab einer bestimmten Station vorhanden sind, damit ich bei mir zuhause eine Art Abfahrstafel haben kann. | 1 | X |
| A004 | Als Christen-Travel-Benutzer möchte ich, dass schon während meiner Eingabe erste Suchresultate erscheinen, damit ich effizienter nach Stationen suchen kann. | 2 | X |
| A005 | Als Christen-Travel-Benutzer möchte ich nicht nur aktuelle Verbindungen suchen können, sondern auch solche zu einem beliebigen anderen Zeitpunkt, damit ich zukünftige Reisen planen kann. | 2 | X |
| A006 | Als Christen-Travel-Benutzer möchte ich sehen, wo sich eine Station befindet, damit ich mir besser vorstellen kann, wie die Situation vor Ort aussieht. | 3 | X |
| A007 | Als Christen-Travel-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 |  |

## Qualitätsziele:

Funktionalität:

* Angeforderte Funktionalitäten sollen entsprechend laut Besprechung umgesetzt werden.
* Die Funktionen sollen korrekt umgesetzt sein.
* Die Funktionsweise soll durch Testfälle sichergestellt sein.

Zuverlässigkeit:

* Die Software soll zuverlässig funktionieren und tun, was man von ihr erwartet.
* Die Software soll in Fehlerfällen den Benutzer korrekt auf Fehler hinweisen.
* Nach Auftreten eines Fehlers sollte die Software, wenn möglich, normal weiterlaufen.

Benutzbarkeit:

* Die Bedienung der Software soll selbsterklärend sein.
* Benutzereingaben sollen validiert werden.
* Die grafische Benutzeroberfläche soll übersichtlich gestaltet sein.
* Das GUI soll sich an UI Standards halten.

Wartbarkeit:

* Der Source Code soll sich an die Programmierrichtlinien halten.
* Variablen, Klassen, Methoden und andere Elemente sollen sprechende Namen haben.
* Der Source Code soll verständlich kommentiert sein, insbesondere sollen alle öffentlichen Methoden inklusive ihrer Parameter kommentiert sein.
* Es soll einfach und problemlos möglich sein, Änderungen an der Software zu machen. Dazu gehört: kein repetitiver Code, kein Spaghetticode, Beachtung des Kohäsionszprinzip und eine möglichst lose Kopplung der einzelnen Module und Klassen.

Portierbarkeit:

* Die Software soll einfach zu installieren sein.
* Die Software soll auch einfach wieder entfernt werden können.
* Die Software soll nach der Installation auch auf einem Rechner ohne Entwicklungsumgebung lauffähig sein.
* Zusammen mit der Software soll eine Installationsanleitung ausgeliefert werden.

## Programmierrichlinien:

### Naming Conventions:

* CamelCase
* Variablen werden immer klein geschrieben, ob lokal oder global
* Methoden werden immer klein geschrieben
* Klassen werden immer grossgeschrieben und sind klar definiert
* GUI-Controls werden ID’s klar definiert und passend gewählt und in CamelCase geschrieben zum Beispiel wäre eine Button Variable so buttonAdd

### Declaration:

* Globale Variablen gleich nach der Klasse
* Lokale Variablen immer am Anfange der Methode
* Variablen müssen nicht immer initialisiert sein

### Comments:

* Wichtige Zeilen und Methoden werden kommentiert
* Wird vor oder neben dem Code kommentiert
* Es wird mit einfachen Worten erklärt was die Eigenschaften der Methoden sind
* Jede öffentliche Methode und Konstruktoren muss kommentiert sein mit:

///<summery>

///Diese Methode macht etwas.

///</summary>

### Statements (New Lines, Indentation):

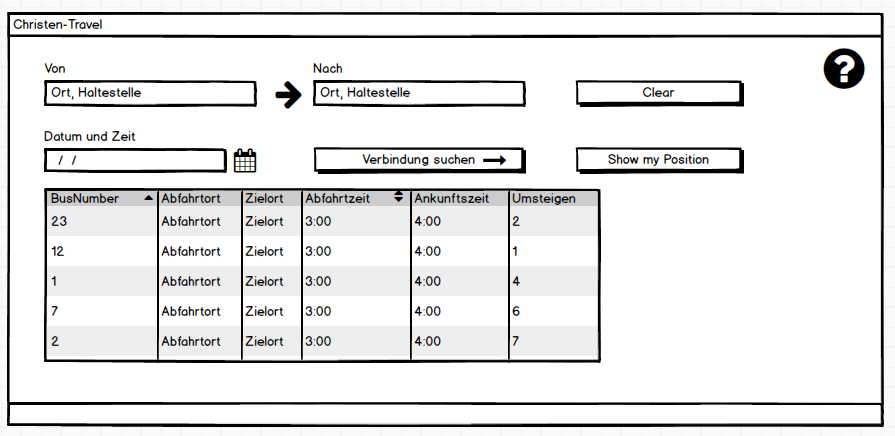
* Klammer « {« ist immer auf gleicher Höhe wie das Statement
* Klammer «}» ist immer eine Zeile unter dem letzten Code
* Operation «==» wird möglichst verhindert besser «<=»
* Höchstens dreifache innere Verschachtelung eines Statements
* Geläufige Variablen z.B. bei for-Schleife verwenden wie zum Beispiel int i=0;

# Design:

## Lösung erarbeiten:

Ich brauche ein GUI welches all den Anforderungen entspricht. Es muss mindesten drei Eingabemöglichkeiten geben (Von, Nach, Zeit und Datum). Ausserdem werden sicher auch zwei Button benötigt einer für die Eingabe und einer für das Verlassen. Ich möchte mich am SBB-Design orientieren, weil es mir persönlich sehr zuspricht. Ausserdem soll ein Hilfsbutton vorhanden sein.

## GUI Entwerfen 1:



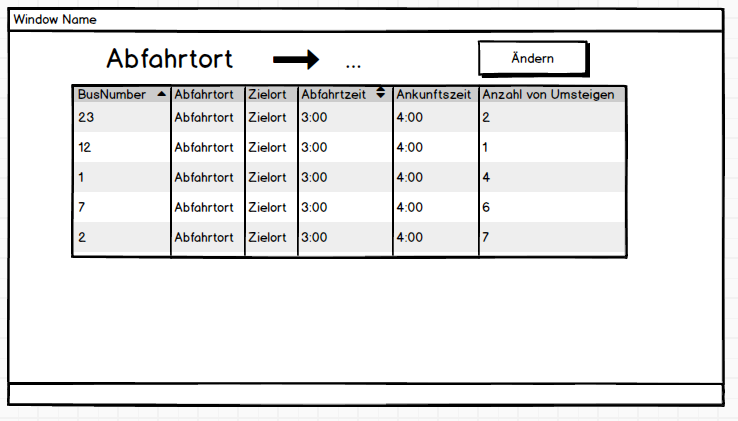
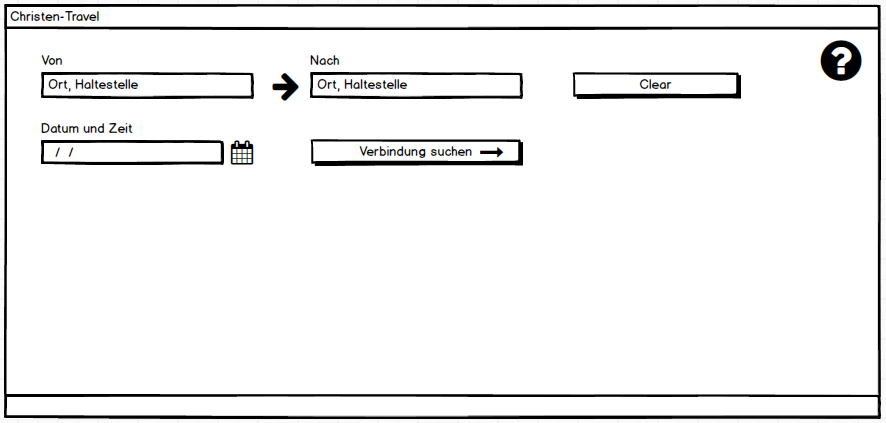
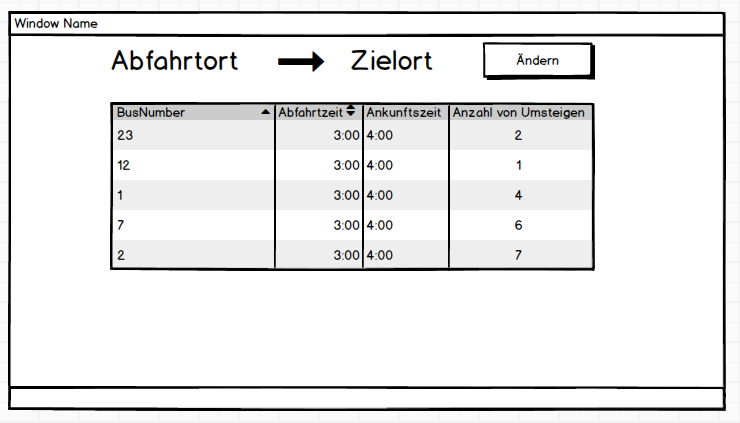
### Vorteile:

Ein Vorteil ist sicher, dass ich mit einer Form arbeiten kann und nicht zwischen den Formen hin und her wechseln muss. Ausserdem finde ich dieses Beispiel sehr ansprechend.

### Nachteil:

Es könnte schlussendlich überladen wirken und die Übersichtlichkeit verlieren.

## GUI Entwerfen 2:



### Vorteile:

Ist sehr schön unterteilt und ausserdem auch übersichtlich und gut verständlich.

### Nachteil:

Wird sehr schnell komplex und man müsste zwischen den einzelnen Formen hin und her wechseln was sehr lästig werden könnte.

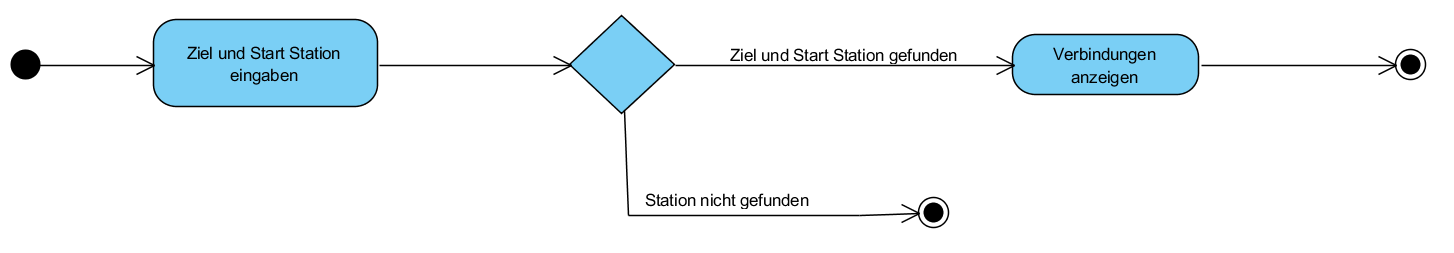
### Lösungen:

Ich habe mich schlussendlich für den ersten Entwurf entschieden und werde diesen verfolgen, da er einfacher in der Umsetzung ist und ich falls die Zeit noch reicht das GUI immer noch teilen kann.

## Activity-Diagram:

Ein Aktivitätsdiagramm ist ein Verhaltensdiagramm der Kategorie UML. Wir hatten den Auftrag zu den ersten drei Anforderungspunkten ein solches Activity-Diagram zu machen.

### Aoo1:

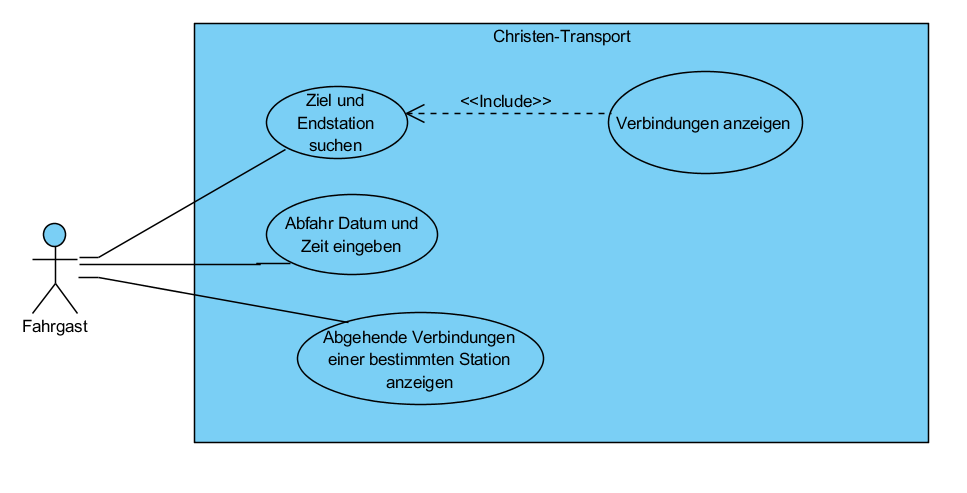


### A002:

### A003:

## Use-Case-Diagram(A001-A003):

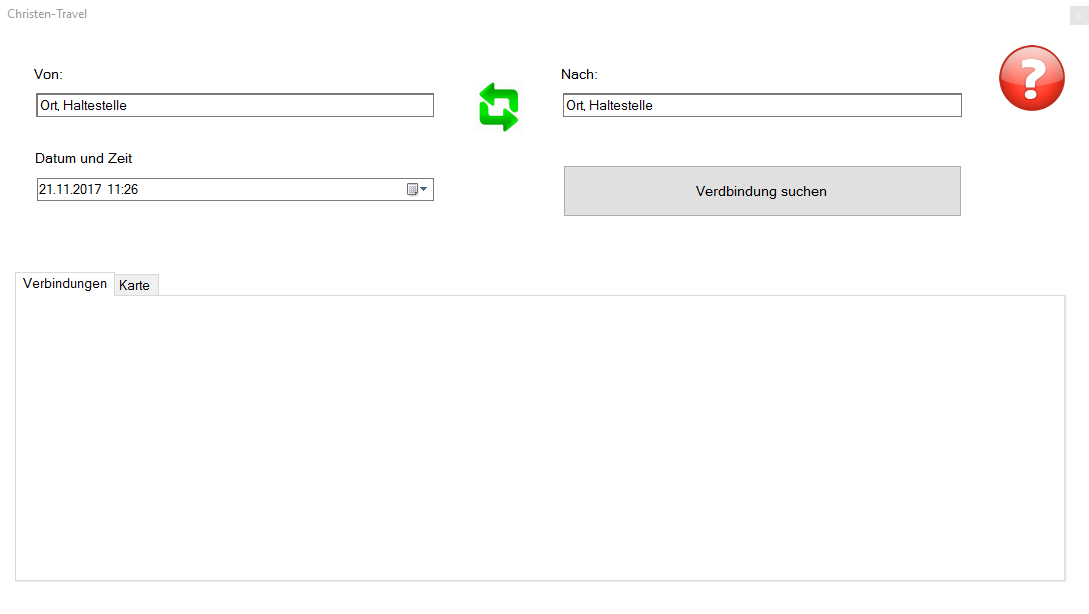
Ein Use-Case-Diagram oder zu Deutsch Anwendungsfalldiagramm ist eines der UML Diagrammarten. Ein solches Diagramm stellt die Anwendungsfälle und Akteure mit ihren jeweiligen Abhängigkeiten und Beziehungen dar. Wir hatten den Auftrag zu den ersten drei Anforderungspunkten ein Use-Case-Diagramm zu machen.



# Umsetzung:

## Umsetzung des GUI:

Das erste was ich in Angriff nahm war die Umsetzung des GUI, welche mir dann das Grundfundament bildete. Dies konnte ich auch erreichen und es ähnelte meinem geplanten Designe sehr aber hatte ein paar Abweichungen.



## Umsetung Code:

Das Coden ging sehr gut, ich kam auch sehr gut voran, doch ich hatte auch einige Probleme welche weiterunten aufgelistet sind. Weitere Infos zum Code entnehmen Sie bitte dem Sourcecode.

### Probleme:

* Combobox hatte Probleme mit Texteingabe und Dropdown-Funktion
  + Lösung: TextBox und darunter eine Combobox die nur für das Dropdown verantwortlich ist.
* DataGrid hatte Probleme mit Befüllung der Header
  + Lösung: Eine eigene Klasse und Header überschreiben
* Meinen Standort auslesen
  + Lösung: Keine
* Probleme mit Hintergrundbild, plötzliche Ruckler
  + Lösung: Ohne Hintergrundbild ohne Transparenz

## Testfälle:

### Testfall A001:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Kunde klickt in die TextBox welches das Label «Von:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Beckenried, Po ein | Das Erste Resultat ist Beckenried, Post im Dropdown |

### Testfall A002:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Kunde klickt in die TextBox welches das Label «Von:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Beckenried, Post ein |  |
| 2 | Kunde klickt in die TextBox welches das Label «Nach:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Stans ein |  |
| 3 | Kunde klickt auf den Button «Verbindung suchen» | Spalte Von: Beckenried, Post  Spalte Nach: Stans, Bahnhof  Abfahrt: Zeiten  Ankunft: Zeiten  Dauer: Zeiten |

### Testfall A003:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Kunde klickt in die TextBox welches das Label «Von:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Beckenried, Post ein. Der Kunde lässt das andere Feld leer oder mit dem Standard Text |  |
| 2 | Kunde klickt auf den Button «Verbindung suchen» | Es sollten mehrere Ergebnisse angezeigt werden im DataGrid, welche alle bei der Spalte «Von» Beckenried Post haben und bei der Spalte «Nach» entweder «Emmetten, Post», «Seelisberg, Bahnhof», «Stans, Bahnhof» und «Stans, Länderpark». Ausserdem sollte bei allen die Nummer 311 haben. Die Abfahrzeit sollte verschieden sein. |

### Testfall A004:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Kunde klickt in die TextBox welches das Label «Von:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Beckenried, Po ein | Der erste Vorschlag im Dropdown sollte Beckenried, Post sein |

### Testfall A005:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Kunde klickt in die TextBox welches das Label «Von:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Beckenried, Post ein |  |
| 2 | Kunde klickt in die TextBox welches das Label «Nach:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Stans ein |  |
| 3 | Kunde gibt noch im Zeit Feld (unter Text «Datum und Zeit») das Abfahrdatum 20.11.2017 15:27 ein |  |
| 4 | Kunde klickt auf den Button «Verbindung suchen» | Es sollten Resultate erscheinen:  Alle Werte in der Spalte Abfahrt sollten nach dem eingegebenen Abfahrwert sein. |

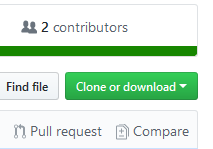
### Testfall A006:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Kunde klickt in die TextBox welches das Label «Von:» darüber hat und den Standard Text Ort, Haltestelle beinhaltet und gibt den Text Beckenried, Post ein |  |
| 2 | Kunde klickt auf den Button «Verbindung suchen» |  |
| 3 | Kunde drückt im tabControll auf Karte | Karte mit einem Kreuz auf dem Standort Beckenried Post |

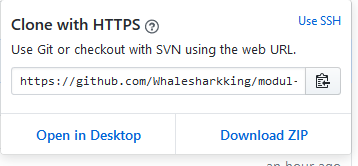
# Einführung Programm:

## Installation:

1. <https://github.com/Whalesharkking/modul-318-student> im Internet öffnen
2. Clone or Download Knopf drücken (grüner Button)



1. Download ZIP anklicken (Wenn Nachricht erscheint das Sie Administrator Rechte brauchen, wenden Sie sich bitte and den Administrator)



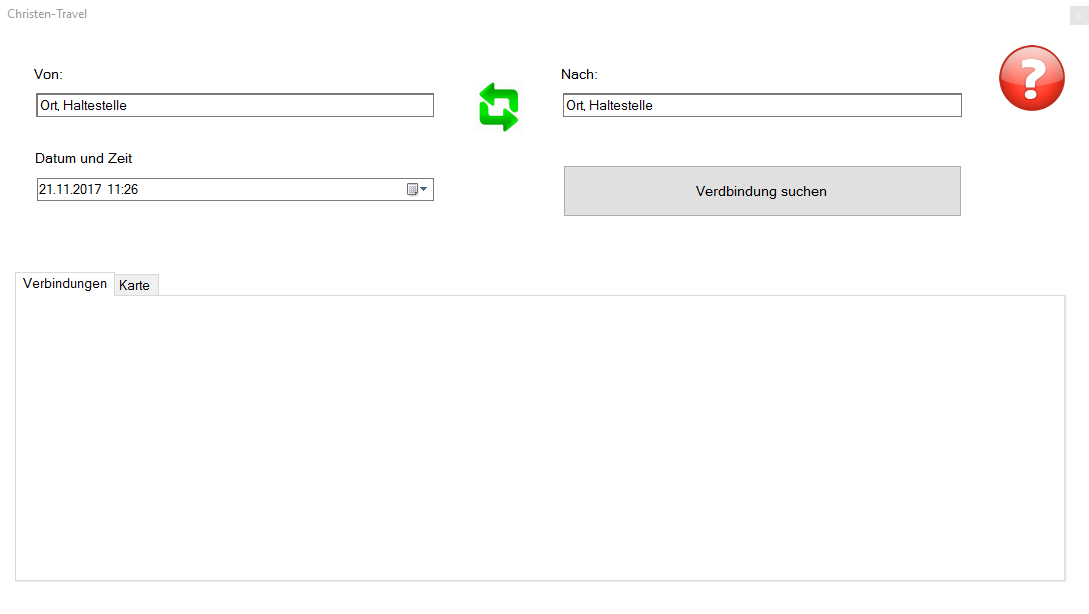
1. Mit OK bestätigen
2. Öffnen Sie den Downloads Ordner
3. Öffnen Sie Christen-Travel.zip
4. Klicken Sie auf den blauen Strich mit dem Text «Entpacken» darunter



1. Bestätigen sie mit der Taste OK
2. Schliessen Sie das Fenster Christen-Travel.zip
3. Öffnen Sie im Ordner Downloads den Ordner Christen-Travel
4. In diesem Ordner Öffnen Sie bitte den Ordner Setup
5. Machen Sie Doppelklick auf die Datei «Christen-Travel.msi»



1. Folgen Sie den Installationsanweisungen
2. Gehen Sie auf den Desktop und öffnen Sie die Datei Christen-Travel



Hilfe-Knopf

Eingabe Abfahrstation

Eingabe Zielstation

Eingabe Abfahrt Datum und Zeit

Verbindungen anzeigen

Stationsposition auf Karte anzeigen

Verbindungen suchen Knopf

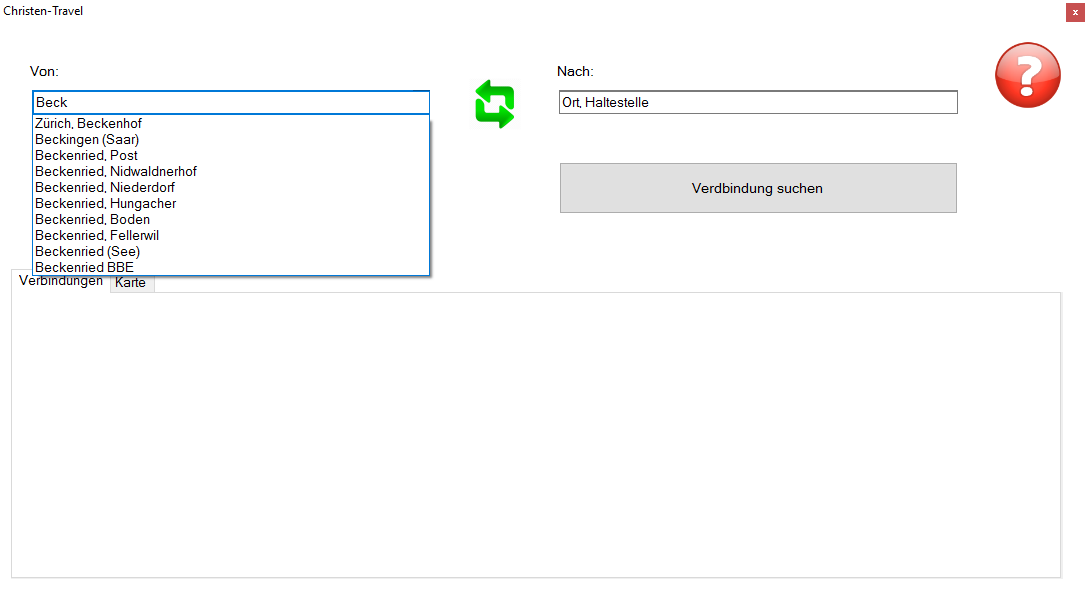
Von Station und nach Station Text tauschen

Programm schliessen

## Hilfe:

Bei Hilfe für das Programm bitte verfügbaren Hilfebutton gebrauchen.

# 



Mögliche Vorschläge

#### Tastenfunktionen:

Pfeiltaste nach unten(Down): Damit wählt man ein vorgeschlagenes Element aus.

Enter: Mit der Taste Enter bestätigt man seine Auswahl von den vorgeschlagenen Elementen.

# Fazit:

Mir hat dieses Projekt sehr Spass gemacht, auch wenn es ab und zu sehr frustrieren war. Ich habe sehr vieles gelernt, was ich auch sicher wieder einmal verwenden kann. Für mich war es ausserdem eine sehr spezielle Erfahrung denn ich arbeite in meinem Geschäft mit Java und da war das Programmieren mit C# gleich noch eine andere Herausforderung. Es kam vor das ich an einigen Punkten hängenblieb, doch ich konnte früher oder später fast immer das Problem lösen und weiterarbeiten. Mein Lernfortschritt war sehr gross, denn ich habe mich am Wochenende auch noch sehr stark damit befasst.

## Resultat:

Ich bin mit dem Resultat zufrieden auch wenn es nicht all Anforderungen abdeckt. Dafür bin ich mit denen Punkten die ich gemacht sehr zufrieden, auch wenn sie noch nicht perfekt sind. Wenn ich wieder ein solches Projekt machen würde wäre mein Vorgehen ziemlich ähnlich. Ich würde wahrscheinlich mehr Zeit ins Design stecken, damit die Applikation moderner aussieht.