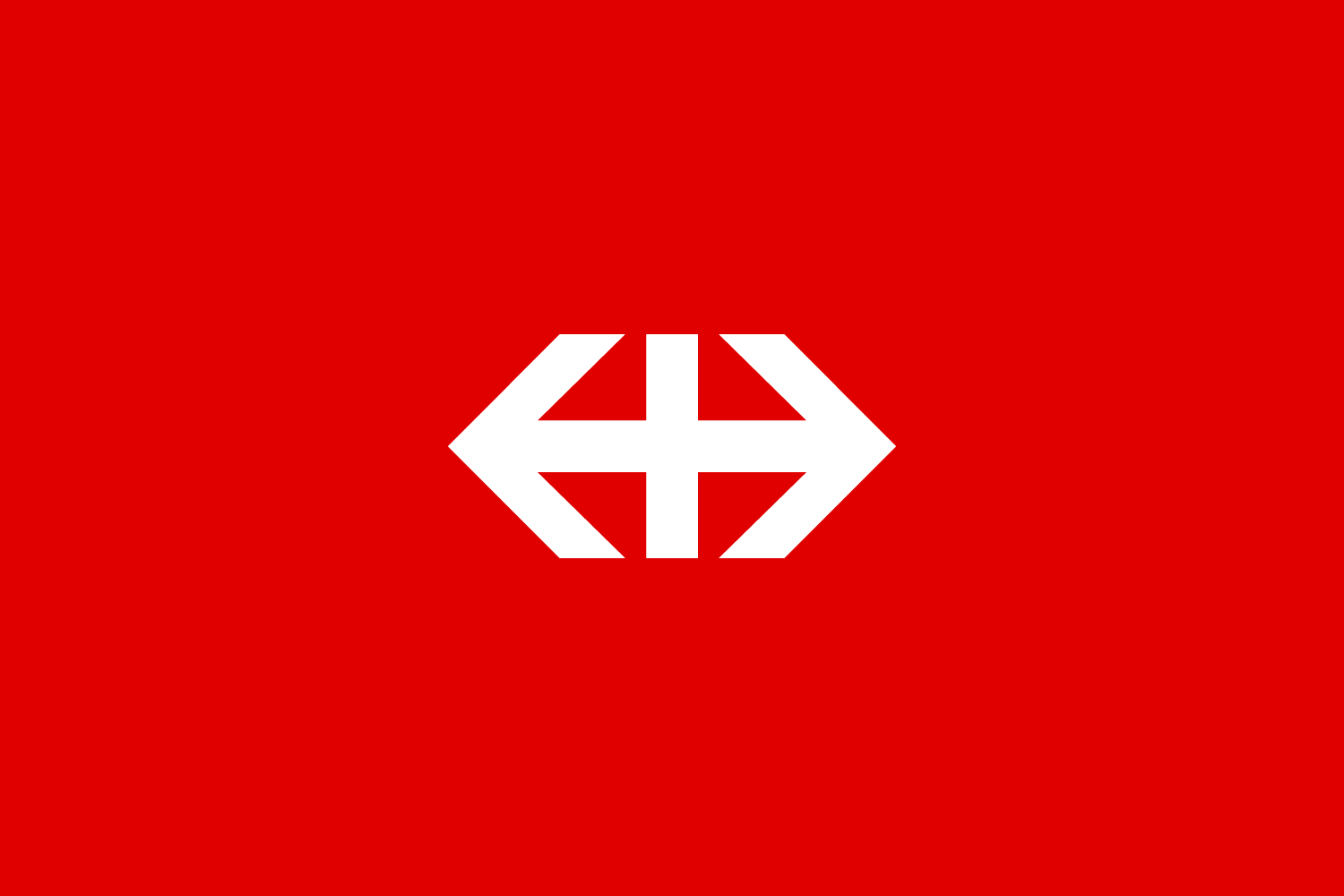
**SBB-Applikation**



**Nevin Helfenstein**

**Inf17aL / BMLT18B**

**Gruobrain 6**

**6372 Ennetmoos**

Inhalt

[1 Einführung 3](#_Toc531789102)

[1.1 Einleitung 3](#_Toc531789103)

[1.2 Zweck 3](#_Toc531789104)

[1.3 Ziel 3](#_Toc531789105)

[2 Programmrichtlinien 3](#_Toc531789106)

[2.1 Einführung 3](#_Toc531789107)

[2.2 Einrückung 3](#_Toc531789108)

[2.3 Kontrollstrukturen 4](#_Toc531789109)

[2.4 Funktionsaufrufe 4](#_Toc531789110)

[2.5 Kommentare 4](#_Toc531789111)

[2.6 Namenskonventionen 5](#_Toc531789112)

[3 GUI-Design 6](#_Toc531789113)

[3.1 Einleitung 6](#_Toc531789114)

[3.2 Altes GUI 6](#_Toc531789115)

[3.3 Neues GUI 8](#_Toc531789116)

[4 Funktionen 9](#_Toc531789117)

[4.1 Vorgegebene Funktionen 9](#_Toc531789118)

[4.2 Umgesetzte Funktionen 10](#_Toc531789119)

[5 Testfälle 11](#_Toc531789120)

[5.1 Einleitung 11](#_Toc531789121)

[5.2 Station suchen 11](#_Toc531789122)

[5.3 Verbindung zwischen zwei Stationen suchen 11](#_Toc531789123)

[5.4 Alle Verbindungen ab einer Station suchen 11](#_Toc531789124)

[6 Installation 12](#_Toc531789125)

[7 Deinstallation 13](#_Toc531789126)

[8 Diagramme 14](#_Toc531789127)

[8.1 Use-Case Diagramm 14](#_Toc531789128)

[8.2 Aktivitätsdiagramm 1](#_Toc531789129)

[8.3 Klassendiagramm 1](#_Toc531789130)

# Einführung

## Einleitung

In diesem ÜK haben wir alle notwendigen Sachen gelernt um dieses Projekt zu vervollständigen. Das Ziel ist es die öffentliche API-Schnittstelle der SBB anzuziehen und mit diesen Daten zu arbeiten. Um diese Projektarbeit zu verwirklichen mussten viele Theorieblöcke gemacht werden. Zum einen mussten zuerst die Grundlagen der Programmiersprache C# erlernt werden, mit der das Projekt realisiert wurde. Danach mussten auch noch verschiedene Diagramme besprochen und gemacht werden. Das sind Aktivitätsdiagramme, Use Cas Diagramme und Klassendiagramme. Dies war wichtig um solche Diagramme am Schluss für die Dokumentation zu erstellen.

## Zweck

Dieses Dokument ist keine Dokumentation zu meiner Projektarbeit sondern soll nur das wichtigste meiner Projektarbeit beschreiben und verständlich machen. Ansonsten sind in diesem Dokument auch noch meine Entwicklungsrichtlinien vorhanden, an denen ich mich den ganzen Programmcode durch orientiert habe.

## Ziel

Das Ziel bei meiner Projektarbeit war es eine Applikation zu erstellen, die den Anforderungen entspricht oder sogar besser ist. Die Applikation sollte aber nicht nur funktionieren sondern auch noch weitere Punkte wie das Exception Handling sauber machen.

# Programmrichtlinien

## Einführung

Die folgenden Programmierrichtlinien wurden erstellt von Nevin Helfenstein und Lino Meyer.

Der Zweck ist die Vereinheitlichung des Programmcodes.

Wenn eine Gruppe von Applikationsentwickler an einer App arbeiten, dann kann das zu Problemen führen wenn es keine Richtlinien gibt. Also müssen allgemeine Richtlinien erstellt werden. So Programmiert jeder nach den Richtlinien und nicht mit seinem eigenen Style.

## Einrückung

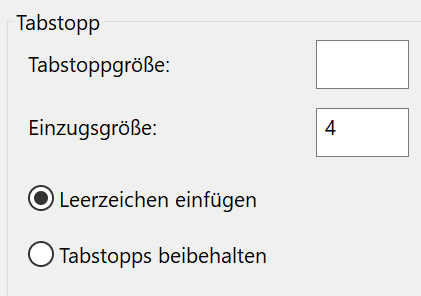
Einrückungen gehören zu den wichtigsten Mitteln um den Code strukturiert zu halten. Wir haben uns für Einrückungen in Höhe von **vier** Leerzeichen entschieden. Das heißt, im VisualStudio muss eingestellt sein, dass ein Tab vier Leerzeichen entspricht. Ausserdem sollte jeder einstellen, dass die Tabs beim Einfügen/Speichern durch Leerzeichen ersetzt werden.

Abbildung : Tabulator definieren

## Kontrollstrukturen

Mit Kontrollstrukturen sind folgende Elemente gemeint:

* if
* for
* while
* switch

Hierbei sollte darauf geachtet werden, dass die geschweiften Klammern immer auf der Höhe der zugehörigen Kontrollstruktur in einer separaten Zeile stehen!

Bei if-Anweisungen ist die kurze Schreibweise erlaubt.

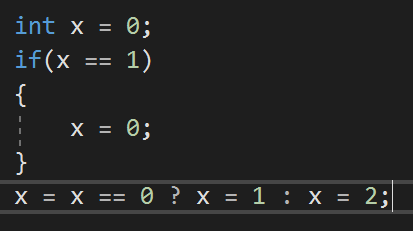
Gutes Beispiel:

Abbildung : Gutes Beispiel Kontrollstrukturen

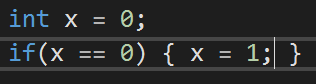
Schlechtes Beispiel:

Abbildung : Schlechtes Beispiel Kontrollstrukturen

## Funktionsaufrufe

Ein Funktionsaufruf geschieht immer ohne Leerzeichen zwischen dem Funktionsnamen, der öffnenden Klammer und dem ersten Übergabeparameter. Die einzelnen Parameter werden mit Komma und Leerzeichen voneinander getrennt. Hinter dem letztem Parameter wird ebenfalls kein Leerzeichen mehr eingesetzt.

Gutes Beispiel:

Abbildung : Gutes Beispiel Funktionsaufruf

Schlechtes Beispiel:

Abbildung : Schlechtes Beispiel Funktionsaufruf

## Kommentare

Damit auch andere die eventuell vorhanden Bugs in eurem Code fixen können, muss dieser nicht nur übersichtlich, sondern auch verständlich sein. Daher ist der ein oder andere Kommentar unumgänglich. Hier ist der richtige Mittelweg zu wählen. Zu viel Kommentar verschränkt den Blick auf das wesentliche (den Code), zu wenig Kommentar lässt ihn undurchdringbar erscheinen…

Kommentare sollten in der Regel in englisch erfasst werden, so dass der Sourcecode auch von nicht deutschsprachigen Entwicklern gelesen werden kann.

Kommentare direkt im Sourcecode (keine Funktions- und Methodenbeschreibung) müssen eingerückt werden. So dass sie auf der selben Ebene wie das zu kommentierende Konstrukt stehen. Ein- bis zweizeilige Kommentare werden wie folgt dargestellt:

Gutes Beispiel:

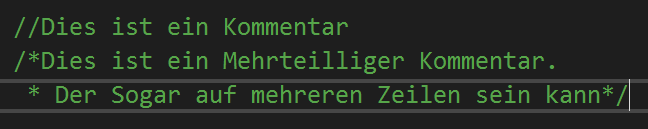


Abbildung : Gutes Beispiel Kommentar

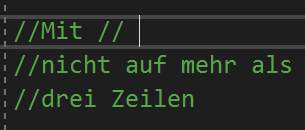
Schlechtes Beispiel:

Abbildung : Schlechtes Beispiel Kommentar

## Namenskonventionen

**Klassen**  
Klassen sollen selbstsprechende Namen erhalten. Kryptische Abkürzungen sollte man wenn möglich vermeiden. Klassen sollten immer mit einem Großbuchstaben beginnen.

**Funktionen**

Bei Funktionen ist der erste Buchstabe und alle nächsten Anfangsbuchstaben der folgenden Wörter gross geschrieben.

Gutes Beispiel:

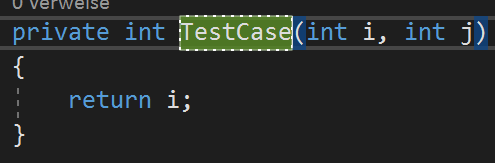


Abbildung : Gutes Beispiel Funktionsname

Schlechtes Beispiel:

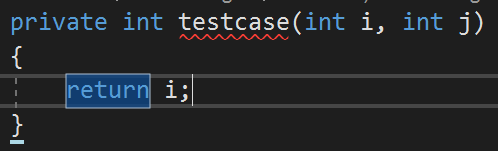


Abbildung : Schlechtes Beispiel Funktionsname

**Variablen**  
Variablen sollten ähnlich den Funktionsnamen im camelStyle (vielen auch als [camelCaps](http://de.wikipedia.org/wiki/CamelCaps" \o "http://de.wikipedia.org/wiki/CamelCaps) oder laOlaStyle bekannt) geschrieben werden. Nur bei Variabeln ist der Anfangsbuchstabe des ersten Wortes klein.

Gutes Beispiel:



Abbildung : Gutes Beispiel Variabel Name

Schlechtes Beispiel:

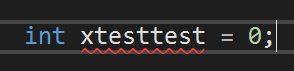


Abbildung : Schlechtes Beispiel Variabel Name

**Konstanten**

Bei Konstanten werden alle Buchstaben groß geschrieben und die einlenen Worte innerhalb de Namens durch Unterstriche getrennt.

Gutes Beispiel:



Abbildung : Gutes Beispiel Konstanten Name

Schlechtes Beispiel:

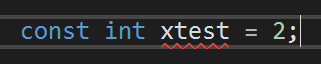


Abbildung : Schlechtes Beispiel Konstanten Name

# GUI-Design

## Einleitung

Zuerst habe ich versucht ein möglichst gutes GUI zu gestalten, bevor ich angefangen habe zu programmieren. Da aber immer wieder Änderungen zu meinem Programm hinzu kamen, musste ich das GUI anpassen. Deswegen habe ich ein neues und ein Altes GUI.

## Altes GUI

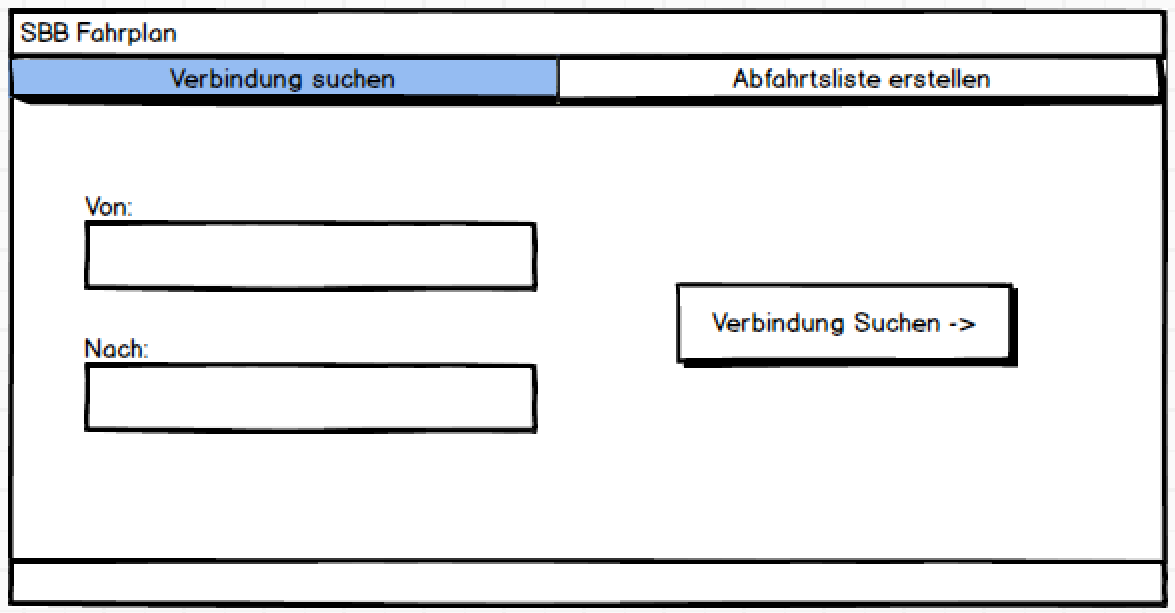


Abbildung : Altes GUI

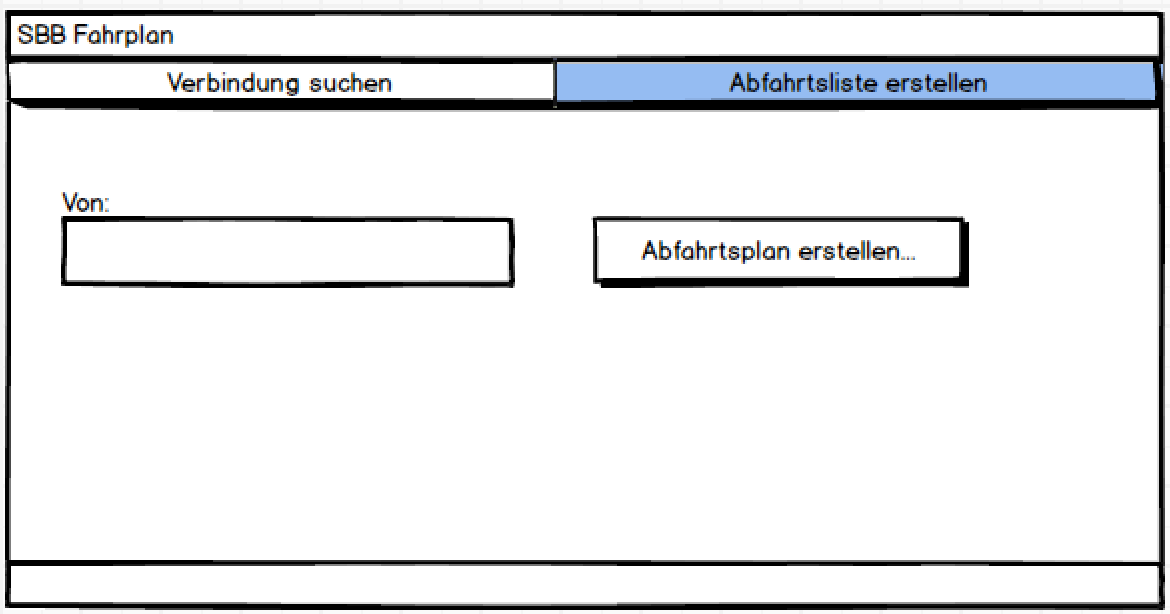


Abbildung : Altes GUI

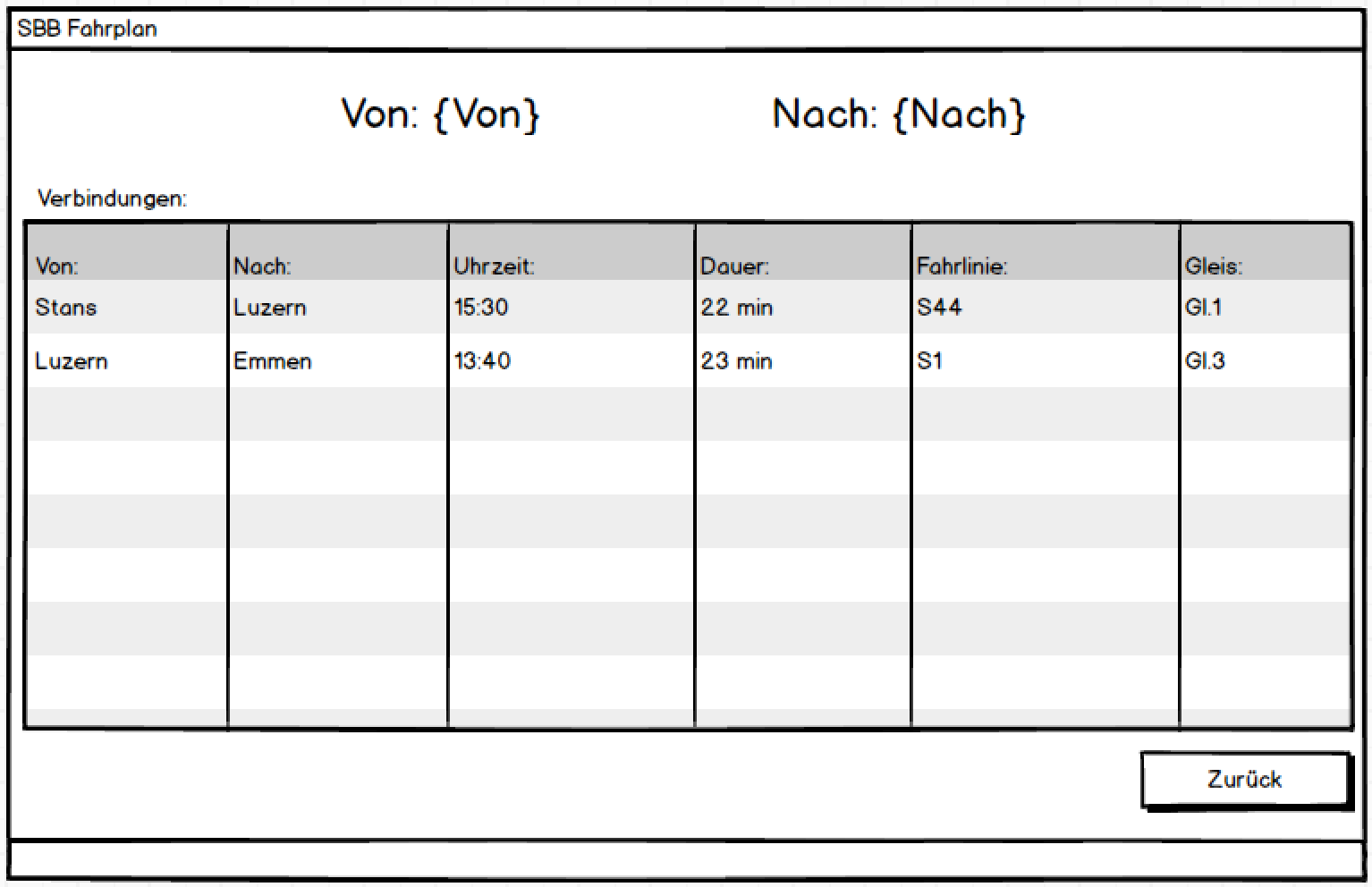


Abbildung : Altes GUI

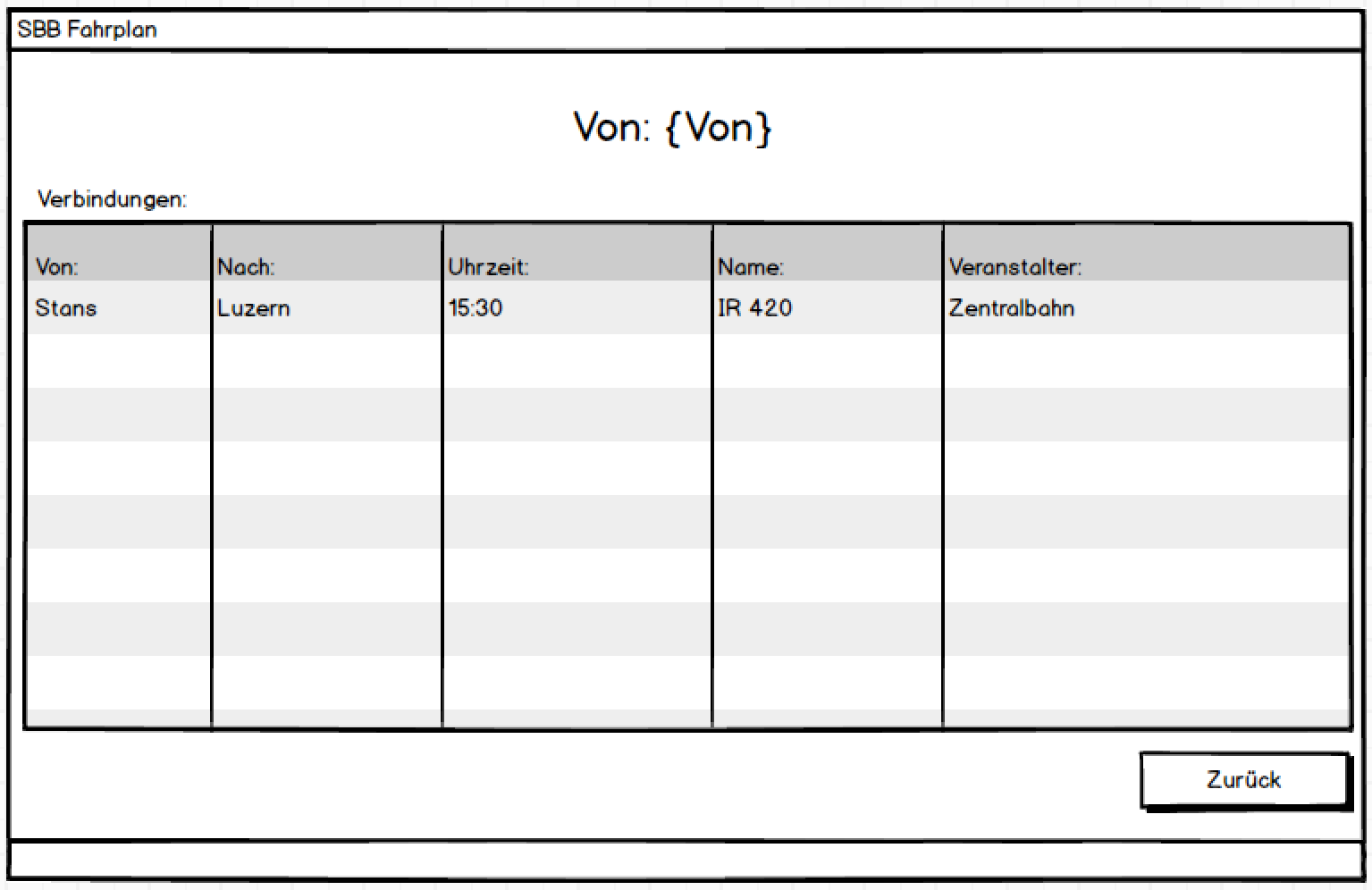
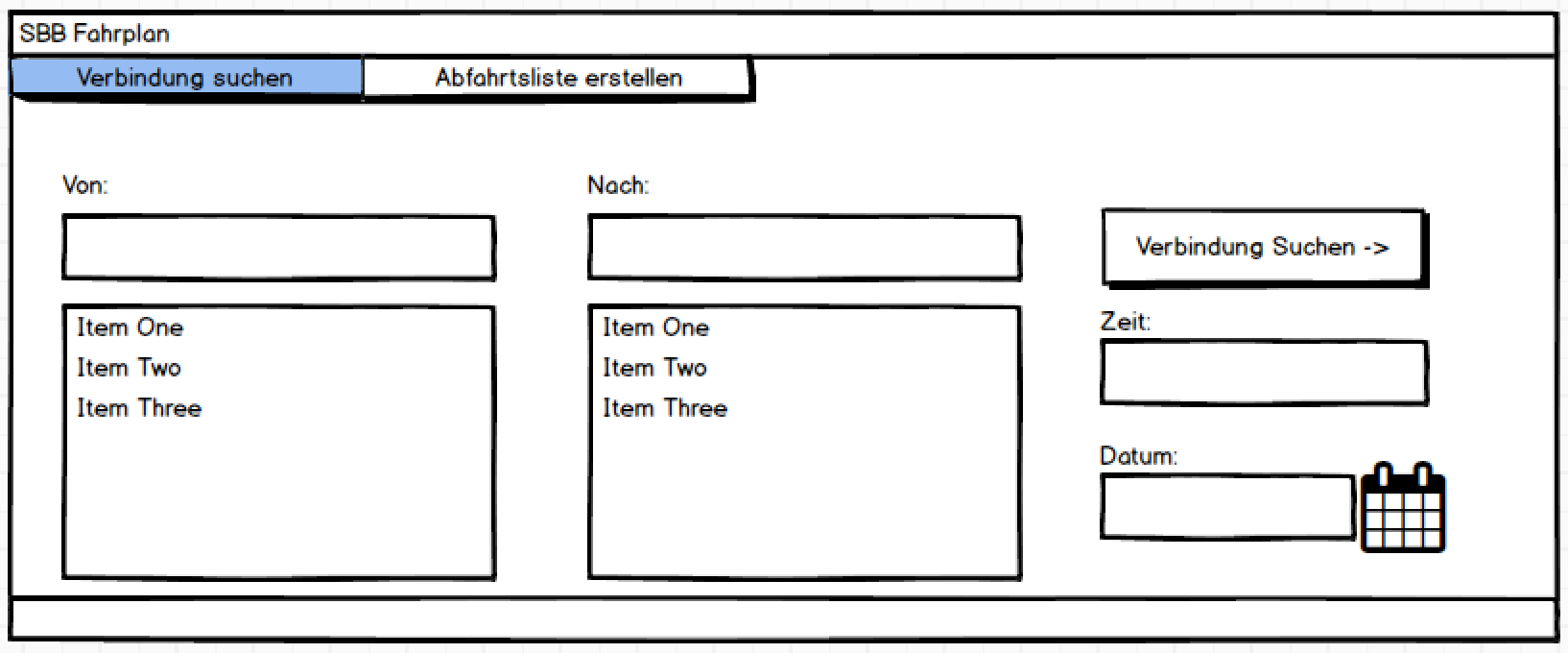


Abbildung : Altes GUI

## Neues GUI



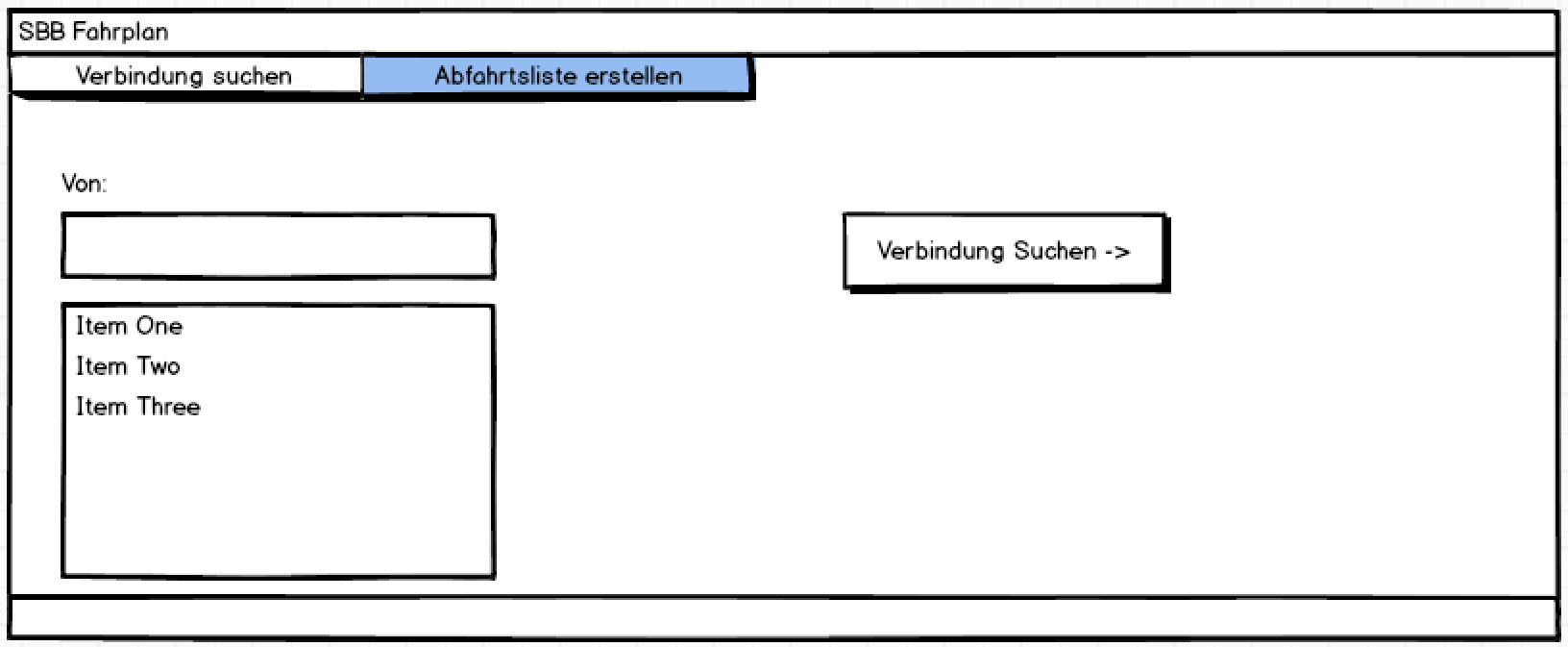
Abbildung 18: Neues GUI

Abbildung : Neues GUI

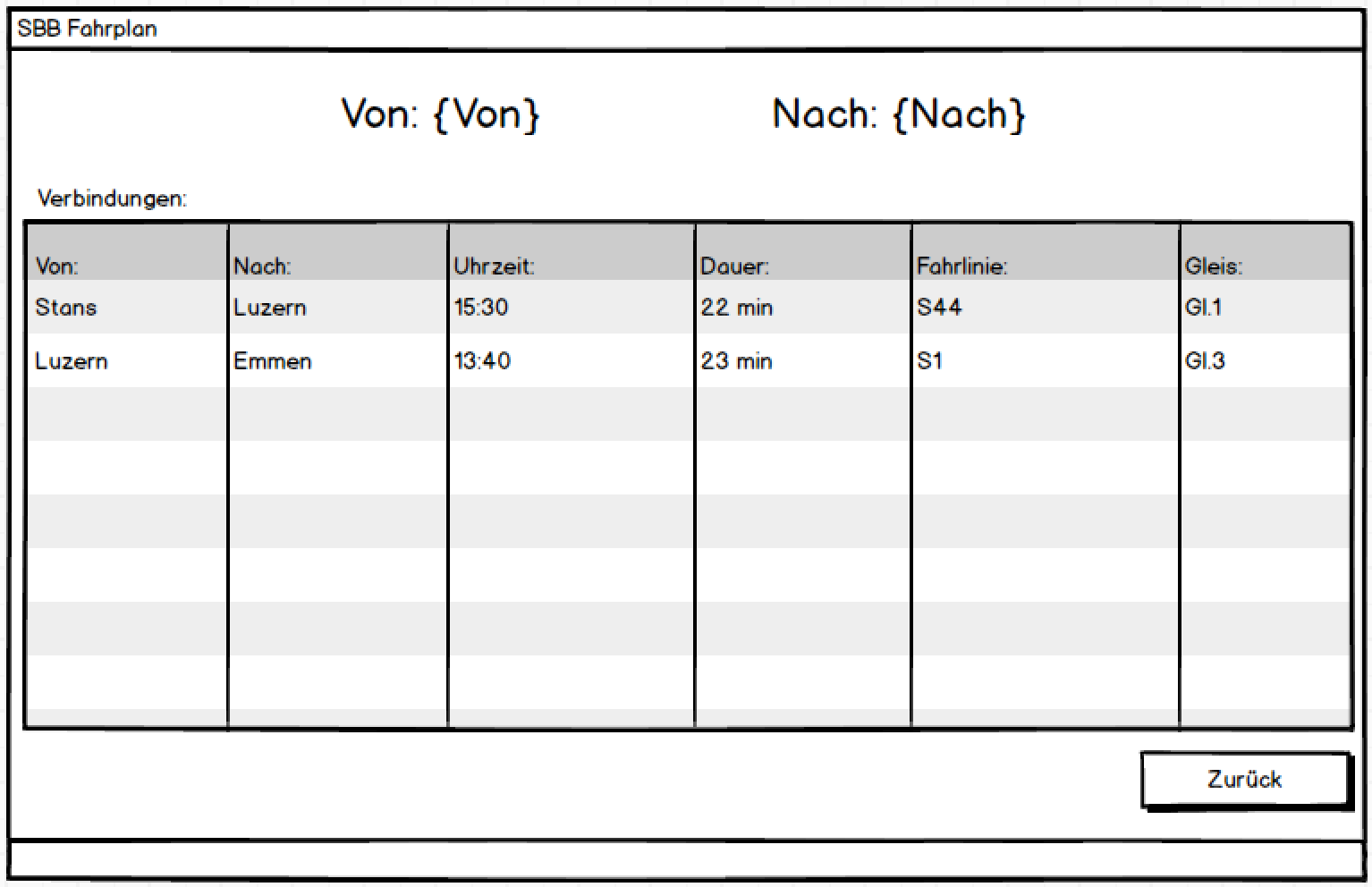


Abbildung : Neues GUI

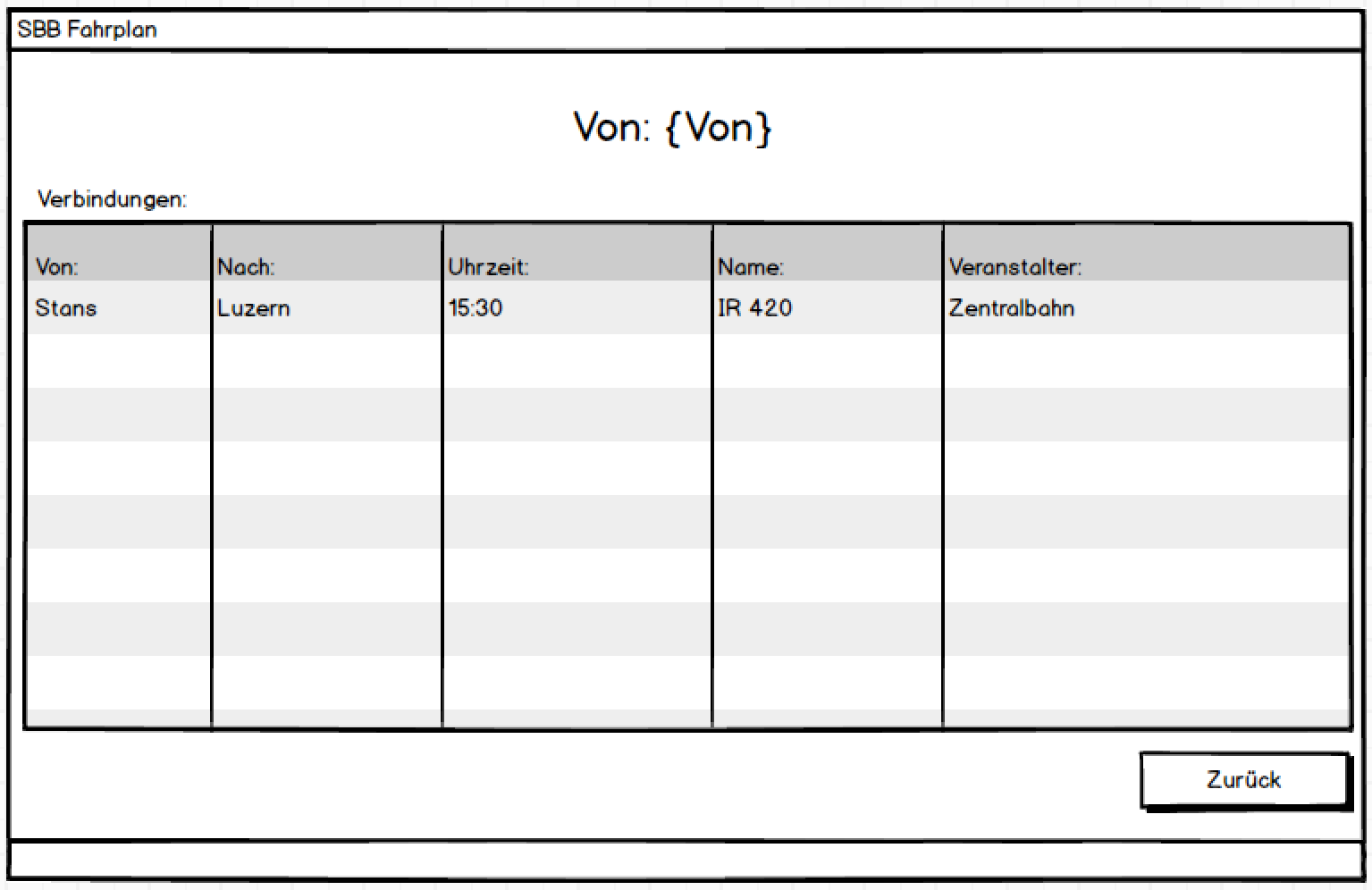


Abbildung : Neues GUI

Beim neuen GUI habe ich einfach die beiden Eingabefenster verändert. Die Ausgabefenster sind genau die gleichen.

# Funktionen

## Vorgegebene 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 |

Bei den vorgegebenen Funktionen hat es Kategorien von eins bis drei. Die Kategorie eins und zwei sollten eigentlich gemacht sein, die Kategorie drei ist «nice to have».

## Umgesetzte Funktionen

In meiner Applikation sind die ersten Fünf Funktionen vollständig erfüllt. Die sechste Funktion, wo der User Google Maps mit Doppelklick auf eine bestimmte Verbindung öffnen kann, funktioniert aber nicht genau so wie sie in Nummer 6 beschrieben ist. Ich habe die Nummer 6 ein wenig anders gelöst als vorgegeben.

|  |  |  |
| --- | --- | --- |
| ID | Beschreibung | Priorität |
| A001 | In meiner Applikation funktioniert die Suche einer Station, dies jedoch nicht mit einem Button sondern direkt mit Autocomplete. | 1 |
| A002 | Das suchen von Verbindungen funktioniert auch Reibungslos. | 1 |
| A003 | Das Erstellen eines Stationboards funktioniert auch ohne Probleme. | 1 |
| A004 | Autocomplete funktioniert mit der Hilfe einer Listbox. | 2 |
| A005 | Das suchen mit einem Datum funktioniert mit einem Datetimepicker. | 2 |
| A006 | Dieser Punkt ist in einem anderen Stil als beschrieben erfüllt. Der User sieht nicht die Station direkt sondern den Googlemaps Routenplaner. | 3 |

Die letzten beiden Anforderungen konnte ich leider nicht erfüllen aus Zeittechnischen Gründen.

# Testfälle

## Einleitung

Für die das Testen gibt es gewisse Vorbedingungen. Diese Vorbedingungen sind bei allen Tests gleich.

Der Benutzer muss eine laufende Internetverbindung haben und das Programm muss vollständig und korrekt auf dem Gerät des Users Installiert sein.

Ich habe für die ersten drei Funktionen ein Testfall erstellt. Für das Suchen einer Station, das Suchen einer Verbindung und dem suchen alles Verbindungen ab einer bestimmten Station.

## Station suchen

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität des Users | Erwartetes Resultat des Programmes |
| 1 | Applikation starten | Die Applikation soll starten ohne abzustürzen oder eine Fehlermeldung zu zeigen. |
| 2 | Stationsnamen eingeben | Direkt bei der Eingabe sollte sich Autocomplete aktivieren und Vorschläge für Stationen machen. Die richtige Station sollte erscheinen. |
| 3 | Stationsnamen auswählen | Mit doppelklickt soll eine Station aus der Listbox ausgewählt werden können. |

## Verbindung zwischen zwei Stationen suchen

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität des Users | Erwartetes Resultat des Programmes |
| 1 | Applikation starten | Die Applikation soll starten ohne abzustürzen oder eine Fehlermeldung zu zeigen. |
| 2 | Beide Stationsnamen eingeben | Direkt bei der Eingabe sollte sich Autocomplete aktivieren und Vorschläge für Stationen machen. Die richtige Station sollte erscheinen. |
| 3 | Stationsnamen auswählen | Mit doppelklickt soll eine Station aus der Listbox ausgewählt werden können. |
| 4 | Verbindung suchen- Button klicken | Mit dem Klick auf den Button soll eine Liste von Verbindungen von den eingegebenen Orten erscheinen. |

## Alle Verbindungen ab einer Station suchen

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität des Users | Erwartetes Resultat des Programmes |
| 1 | Applikation starten | Die Applikation soll starten ohne abzustürzen oder eine Fehlermeldung zu zeigen. |
| 2 | Stationsnamen eingeben | Direkt bei der Eingabe sollte sich Autocomplete aktivieren und Vorschläge für Stationen machen. Die richtige Station sollte erscheinen. |
| 3 | Button klick auf Verbindungen suchen | Alle Abfahrten ab einer bestimmten Station sollen gezeigt werden. |

# Installation

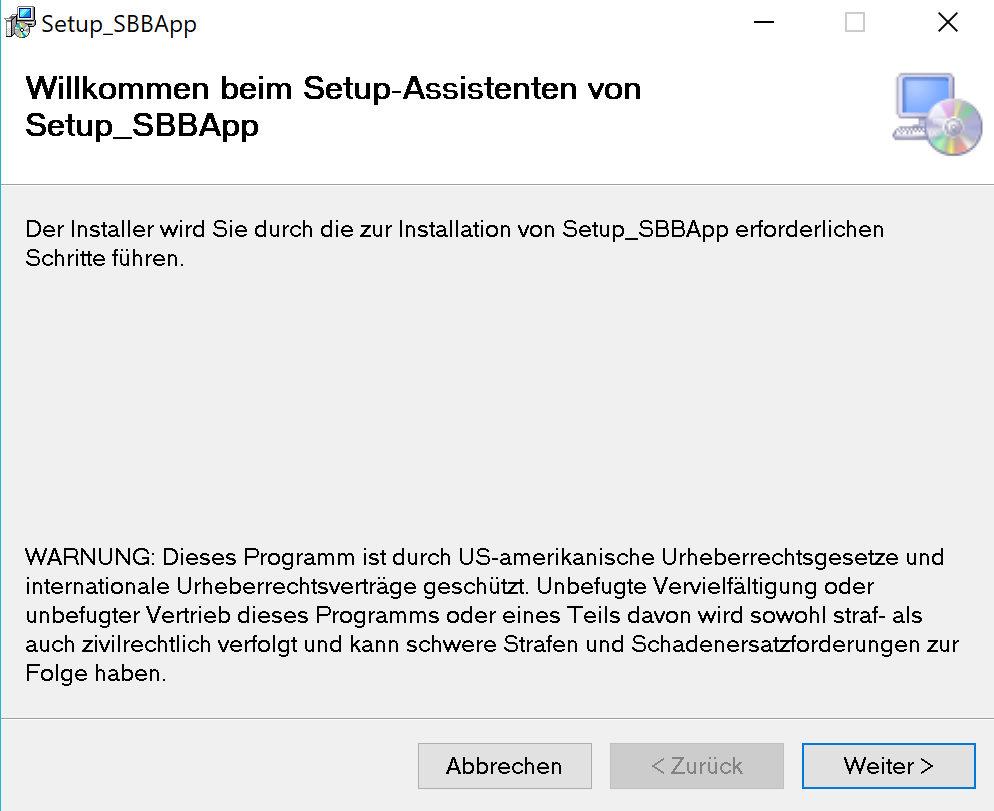


Abbildung : Start Setup

Wenn die .exe Datei ausgeführt wird, erscheint dieses Bild.

Hier einfach auf «Weiter klicken».

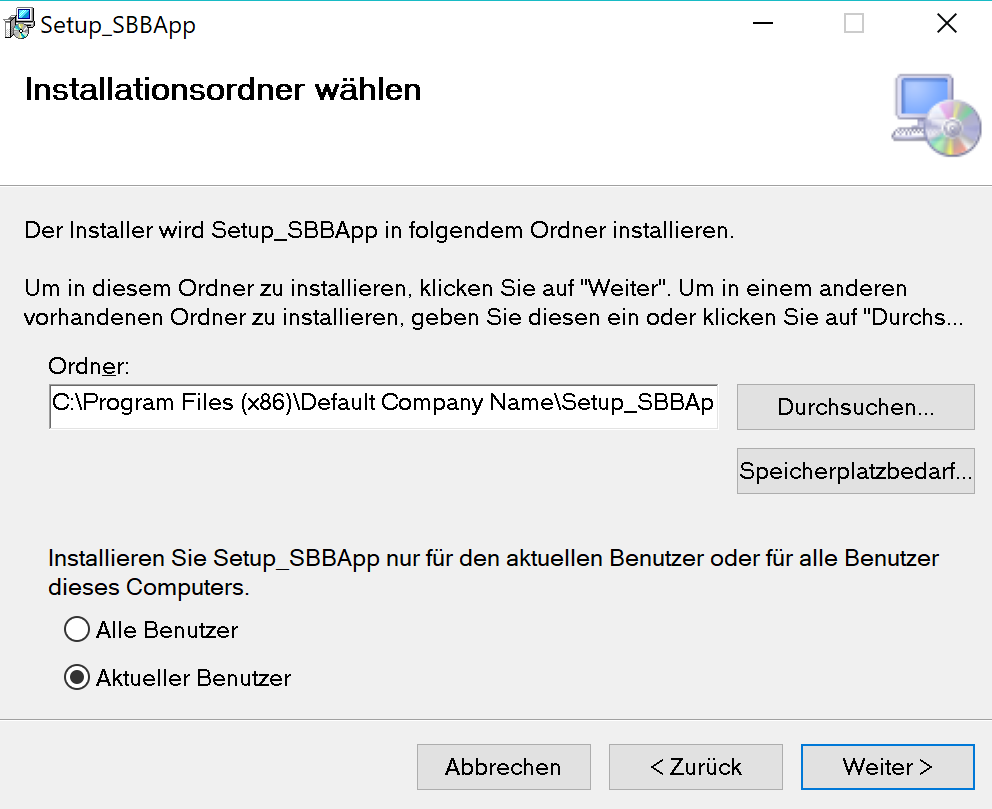


Abbildung : Installationsordner

Danach erscheint ein Fenster indem der Installationspfad angegeben werden muss.

Hier kann der Standardpfad gelassen werden und einfach auf «Weiter» geklickt werden.

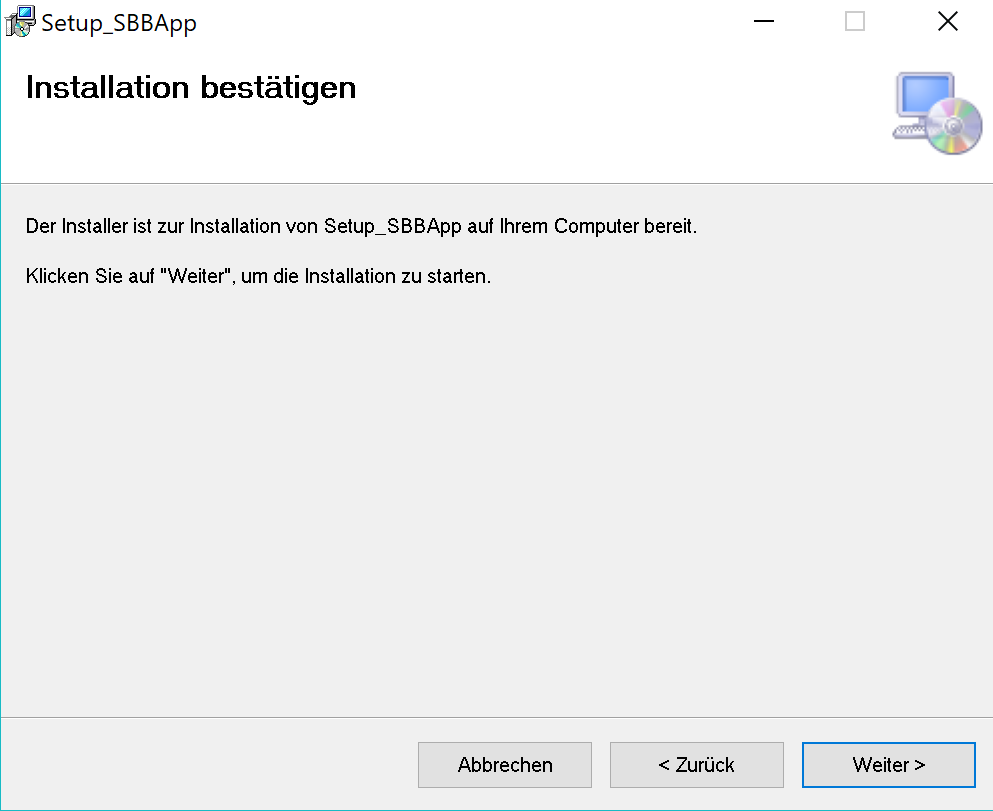


Abbildung : Installation bestätigen

Dann muss noch die Installation bestätigt werden.

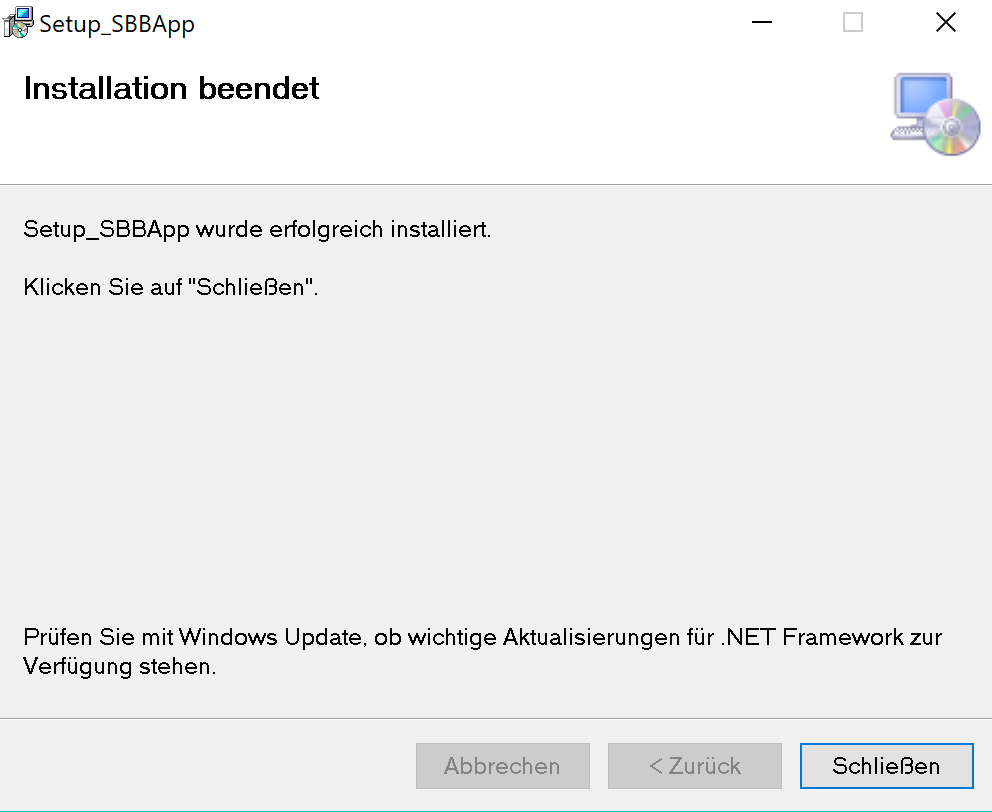


Abbildung : Installation beendet

Dann erscheint einfach nur noch das Infofenster das die Installation Erfolgreich durchgelaufen ist.

# Deinstallation

Um das Programm zu Deinstallieren muss die Systemsteuerung geöffnet werden.

Danach muss zu folgendem Pfad navigiert werden: Systemsteuerung\Programme\Programme und Features.

Danach auf das richtige Programm rechtsklick drücken und auf «Deinstallieren» klicken. Danach noch einmal bestätigen dass das Programm wirklich deinstalliert werden soll.

# Diagramme

## Use-Case Diagramm

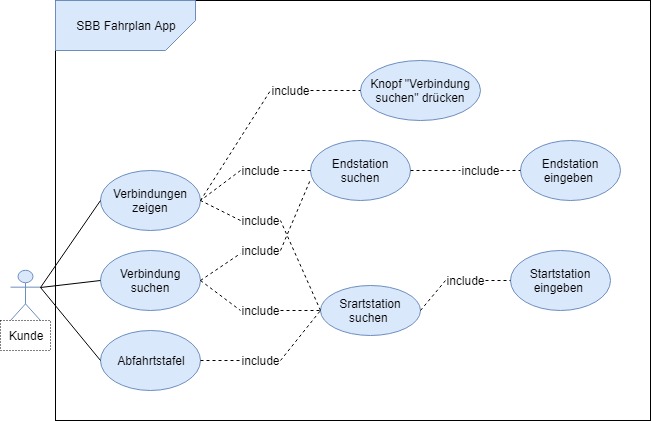


Abbildung : Use-Case Diagramm

## Aktivitätsdiagramm

Abbildung : Aktivitätsdiagramm

## Klassendiagramm

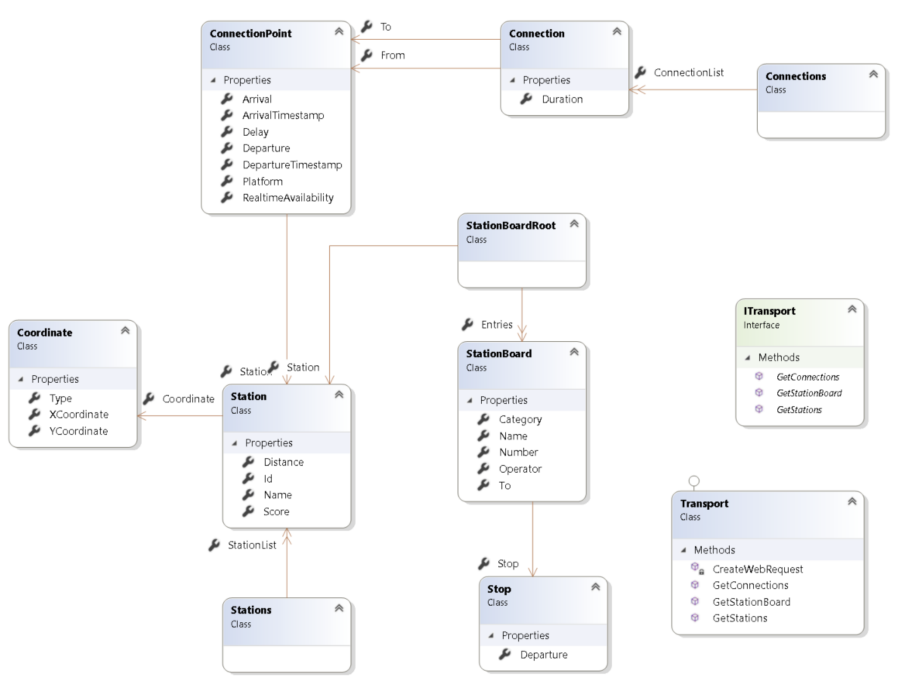


Abbildung : Klassendiagramm