

Philip Baumann

Public transport application

ÜK 318

Table of Contents

[2 Einleitung 2](#_Toc34920397)

[3 Anforderungen 2](#_Toc34920398)

[4 Bekannte Bugs 2](#_Toc34920399)

[5 GUI-Entwurf 2](#_Toc34920400)

[5.1 Abfahrtstafel 3](#_Toc34920401)

[5.2 Station suchen 3](#_Toc34920402)

[5.3 Connection suchen 4](#_Toc34920403)

[6 Analyse 5](#_Toc34920404)

[6.1 Verbindungen abfragen 5](#_Toc34920405)

[7 Code-Richtlinien 11](#_Toc34920406)

[8 Programmierrichtlinien 11](#_Toc34920407)

[8.1 Naming Conventions 11](#_Toc34920408)

[8.2 Declaration 11](#_Toc34920409)

[8.3 Comments 11](#_Toc34920410)

[8.4 Statements 11](#_Toc34920411)

[9 Testfälle 12](#_Toc34920412)

[9.1 Testfall Anforderung A001 12](#_Toc34920413)

[9.2 Testfall Anforderung A002 13](#_Toc34920414)

[9.3 Testfall Anforderung A003 13](#_Toc34920415)

[9.4 Testfall Anforderung A004 13](#_Toc34920416)

[9.5 Testfall Anforderung A005 14](#_Toc34920417)

[9.6 Testfall Anforderung A006 14](#_Toc34920418)

[9.7 Test Auswertung 14](#_Toc34920419)

[10 Installationsanleitung 15](#_Toc34920420)

[11 Deinstallation 15](#_Toc34920421)

# Einleitung

Dieses Dokument ist im Rahmen des überbetrieblichen Kurses, Modul 318 entstanden. Hier werden sämtliche Projektvorbereitungen dokumentiert. Das Dokument bezieht sich auf die von mir entwickelte Applikation (Public Transport Application). Ebenfalls sind hier die Testfälle für die obligatorischen Anforderungen abgelegt.

# Anforderungen

Die Public Transport Application beinhaltet die Funktionen der Anforderungen eins bis sechs. Dies bedeutet das alle Anforderungen der Kategorie eins und zwei vollständig beinhaltet sind und eine weitere Funktion der Kategorie drei vorhanden ist. Die restlichen Anforderungen wurden vernachlässigt, damit ich die Qualität der Software auf ein höheres Niveau bringen konnte. Als zusätzliches Feature verfügt das Programm über einen Installations-Wizard

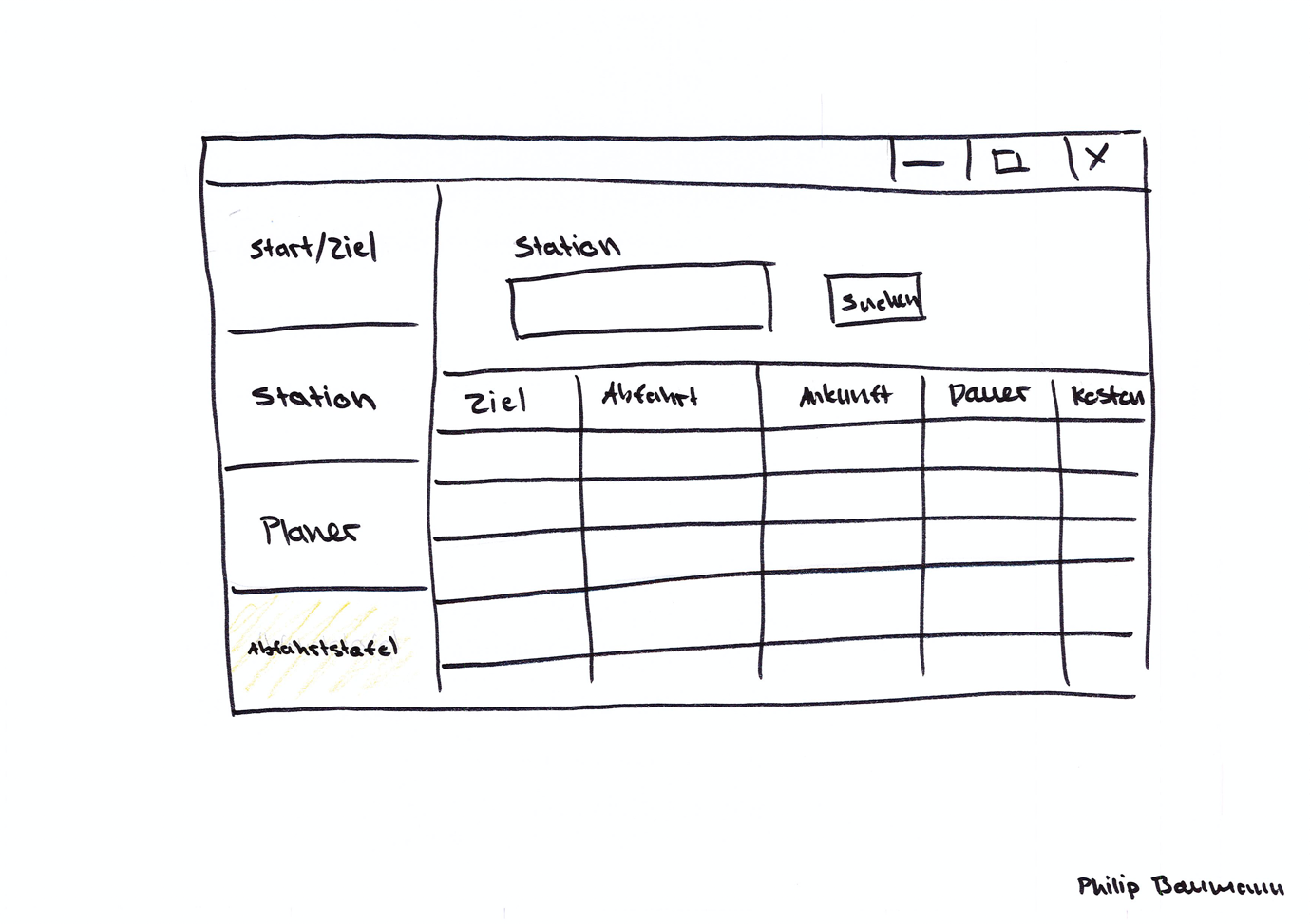
# Bekannte Bugs

* Aus ComboBoxen kann nicht durch einen Enterschlag oder Tabulator klick gesprungen werden.
* DatePicker bei eigener Eingabe und falschem Format kann es zu falschen Ergebnissen führen.

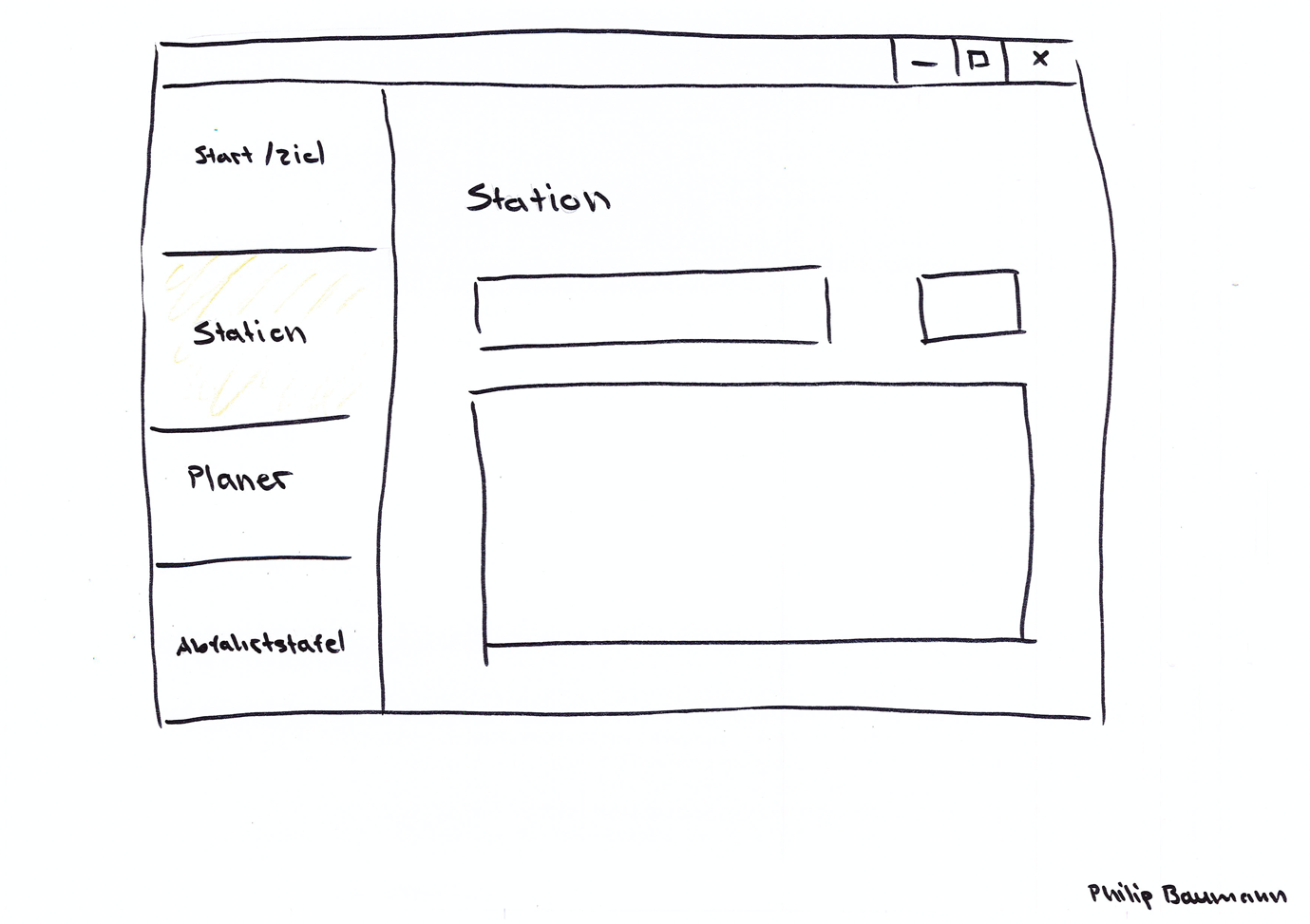
# GUI-Entwurf

Anbei befinden sich Mockups der Anforderungen A001, A002 und A003. Diese wurden im Vorhinein ausgearbeitet und legten einen Grundstein des Layouts dar. Im weiteren Verlauf der Entwicklung mussten einige Anpassungen am GUI gemacht werden, sodass das finale Design ein wenig von den Mockups abschweift. Die Mockups wurden von Hand gezeichnet.

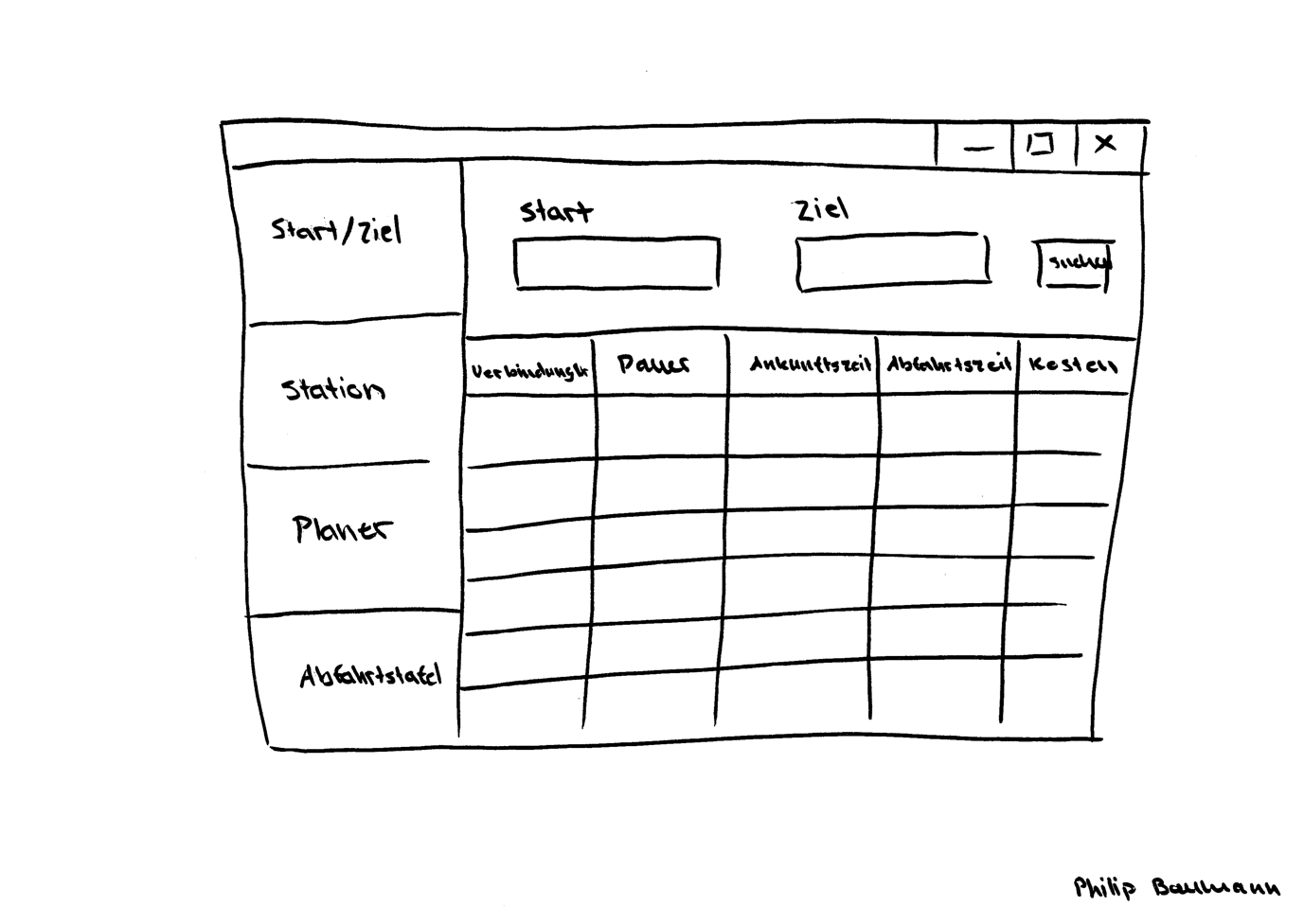
## Abfahrtstafel



## Station suchen



## Connection suchen



# Analyse

## Verbindungen abfragen

|  |  |
| --- | --- |
| Use Case | 1, Station suchen |
| Beschreibung | ÖV-Benutzer sucht eine bestimmte Station. Zur Hilfe kann er eine Teil-Zeichenkette des vollständigen Namens der Station eingeben. |
| Akteur(e) | Benutzer des öffentlichen Verkehrs |
| Auslöser | Ein Benutzer möchte eine bestimmte Station suchen, weiss allerdings den exakten Namen der Station |
| Vorbedingungen | Der Benutzer hat einen Teil des vollständigen Namens eingegeben |
| Ablauf | 1. Public Transport Application öffnen 2. Zum Tab «Station» navigieren 3. Teil des vollständigen Namens im Eingabefeld eingetragen 4. Auf den Button «Suchen» klicken |
| Ergebnis | Sämtliche Verbindungen, auf welche die Teil-Zeichenkette zutrifft, werden auf dem GUI dargestellt. |

A picture containing indoor, white, table, black

Description automatically generated

**Diagramm Beschreibung**

Das Diagramm zeigt den Ablauf einer Suchabfrage, welche die entsprechende Station zurück liefert. Dieses Vorgehen wird einerseits bei der Stationssuche, als auch bei der automatischen Vervollständigung verwendet.

Nach öffnen der Applikation und dem Navigieren in den Station Tab, soll ein Eingabefeld, in welchem der Benutzer der Name einer gewünschten Station eingeben kann. Anschliessend wird diese Eingabe validiert. Bei invalider Eingabe wird dem Benutzer ein konstruktives Feedback gegeben. Ist die Eingabe valide, wird der Webservice aufgerufen und die Daten entgegengenommen. Ist die Abfrage ohne Ergebnisse zurückgekommen, so wird dem Benutzer dies mitgeteilt. Ansonsten werden die Daten im GUI dargestellt.

|  |  |
| --- | --- |
| Use Case | 2, Verbindungen abfragen |
| Beschreibung | ÖV-Benutzer fragt für die Verwendung des öffentlichen Verkehrs den Fahrplan einer bestimmten Verbindung ab. |
| Akteur(e) | Benutzer des öffentlichen Verkehrs |
| Auslöser | Ein ÖV-Benutzer möchte die Verbindungen einer bestimmten Verbindung einsehen. |
| Vorbedingungen | Der Benutzer hat eine valide Start- und Endstation im Eingabefeld eingegeben. |
| Ablauf | 1. Public Transport Application öffnen 2. Zum Tab «Verbindungen» navigieren 3. Eine valide Start- und Endstation in den jeweiligen Eingabefelder angeben 4. Auf den Button «Suchen» klicken |
| Ergebnis | Die Informationen zu den einzelnen Verbindungen wurden in der Applikation dargestellt. |

A close up of a map

Description automatically generated

**Diagramm Beschreibung**

Das nebenanstehende Diagramm zeigt die abzuarbeitenden Schritte für eine Verbindungsabfrage. Dieser Ablauf wird bei der Verbindungssuche verwendet.

Die Applikation soll geöffnet und in den Tab Verbindungen navigiert werden. Anschliessend soll der Benutzer zwei Eingabefelder befüllen. Diese werden daraufhin validiert. Bei fehlerhafter Eingabe wird ein konstruktives Feedback zurückgegeben. Sind die Eingaben valide, wird der Webservice aufgerufen und die Daten entgegengenommen. Ist die Abfrage ohne Treffer zurückgekommen, so wird dies dem Benutzer mitgeteilt. Ansonsten werden die Daten auf dem GUI dargestellt.

|  |  |
| --- | --- |
| **Use Case** | 3, Abfahrtstafel einsehen |
| **Beschreibung** | ÖV-Benutzer fragt für die Verwendung der des öffentlichen Verkehrs alle anstehenden Verbindungen einer bestimmten Station ab. |
| **Akteur(e)** | Benutzer des öffentlichen Verkehrs |
| **Auslöser** | Ein ÖV-Benutzer möchte alle Verbindungen einer bestimmten Station einsehen. |
| **Vorbedingungen** | Der Benutzer hat eine valide Station im Eingabefeld eingegeben. |
| **Ablauf** | 1. Public Transport Application öffnen 2. Zum Tab «Abfahrtstafel» navigieren 3. Eine valide Station im Eingabefeld eingeben 4. Auf den Button «Suchen» klicken |
| **Ergebnis** | Einige Verbindungen einer Station werden dem Benutzer auf dem GUI aufgeführt. |

A close up of text on a white surface

Description automatically generated

**Diagramm Beschreibung**

Nach öffnen der Applikation wird in das Tab Abfahrtstafel navigiert. Dort befindet sich ein Eingabefeld, welches mit dem Namen einer Station befüllt werden soll. Bevor der Webservice aufgerufen wird, werden die Eingabe validiert. Bei inkorrekten Eingaben wird dem Benutzer ein Feedback gegeben. Sind die Eingaben korrekt, wird der Webservice aufgerufen und die Daten werden verarbeitet. Anschliessend werden die Daten im GUI dargestellt.

# Code-Richtlinien

# Programmierrichtlinien

## Naming Conventions

* Variablen werden im Camel Case benannt.
* Methoden werden im Camel Case geschrieben und nachdem Namen wird mit einem Leerschlag von dem Klammern getrennt.
* Konstante und statische Variablen werden grossgeschrieben und jedes Wort durch einen Unterstrich getrennt.
* Der Klassenname wird nach dem Pascal Case benannt.
* Für jede Klasse ist ein separates File vorgesehen.
* Der Namen eines GUI-Elementes beginnt immer mit dem Namen des Elementes und ihrer Funktion

## Declaration

* Die geschweiften Klammern einer Methode werden immer auf derselben Zeile wie der Methodenname angelegt.
* Membervariablen sind standardmässig auf privat gesetzt.
* Membervariabeln sind nach dem Klassennamen im Klassenbody anzugeben.
* Nach den Membervariablen folgen die Konstruktoren.
* Anschliessend werden die Eigenschaften, die C# spezifische «getter/setter» Konstrukte, verwendet.
* Am Schluss sind dann die spezifizierten Methoden zu schreiben.

## Comments

* Der Code soll blockweise kommentiert werden.
* Kommentare sollen auf einer Linie festgehalten werden und mit «//» eingeleitet werden

## Statements

* Die öffnende geschweifte Klammer wird bei if-else Statements auf derselbe Zeile wie das if-Schlüsselwort angelegt. Dasselbe gilt auch für for, foreach, while, switch Statements, etc. Bei if-Konstrukten wird nach dem else-Schlüsselwort die öffnende Klammer angelegt.
* Bei for-Anweisungen wird für die inkrementierende Zahl i für Index verwendet. Ausgenommen bei foreach Anweisung, da soll die iterierende Variable aussagekräftigt benannt werden.
* Bei jeder Verschachtelung soll der Code um einen Tabulator schlag eingerückt werden.

# Testfälle

In diesem Kapitel sind die Testfälle für die obligatorischen Testfälle (A001, A002 und A003), sowie die entsprechenden Testergebnisse festgehalten. In der Spalte «Given» ist die Ausgangslage, in welcher sich der Benutzer befindet, beschrieben. In einer weiteren Spalte «TODO» ist das Vorgehen des Testers angegeben. In der letzten Spalte «Then» ist das erwartete Ergebnis, welches die Applikation liefern soll, beschrieben.

## Testfall Anforderung A001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Given** | **TODO** | **Then** | **Status** |
| 1 | Der Benutzer hat die Applikation gestartet und befindet sich im «Station» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld, welches sich unter dem Titel «Station» befindet die Angaben «Luzern, Bahnhof» ein. Anschliessend klickt er auf den Button «Suchen». | Die Listbox beinhaltet drei Elemente:   * Luzern, Bahnhof * Luzern, Inseliquai/Bahnhof * Luzern Bahnhofquai   Beachte hier, dass diese Testresultate vom 12.03.2020 entstanden sind. | Erfolgreich |
| 2 | Der Benutzer hat die Applikation gestartet und befindet sich im «Station» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld, welches sich unter dem Titel «Station» befindet, den Teilnamen (Luzern Kant) ein. Anschliessend klickt er auf den Button «Suchen» | Die Listbox beinhaltet zwei Elemente   * Luzern, Kantonalbank * Luzern, Kantonsspital   Beachte hier, dass diese Testresultate vom 12.03.2020 entstanden sind. | Erfolgreich |
| 3 | Der Benutzer hat die Applikation gestartet und befindet sich im «Station» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld, welches sich unter dem Titel «Station» befindet, den invaliden Station Namen «qwertzuiop» ein. Anschliessend klickt er auf den Button «Suchen» | Der Benutzer erhält eine Messagebox, welche im erklärt, dass seine Suche keine Ergebnisse ergab. | Erfolgreich |

## Testfall Anforderung A002

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Given** | **TODO** | **Then** | **Status** |
| 1 | Der Benutzer hat die Applikation gestartet und befindet sich im «Connection» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld, welches unter dem Titel «Start» befindet den Wert «Luzern, Bahnhof» ein. Im Eingabefeld, welches unter dem Titel «Ziel» steht, gibt der Benutzer den Wert «Obernau, Dorf» ein. Anschliessend klickt er auf den Button «Suchen». | Die Listbox beinhaltet Busverbindungen. Im Format: Start – Ziel – Abfahrtszeit - Duration | Erfolgreich |

## Testfall Anforderung A003

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Given** | **TODO** | **Then** | **Status** |
| 1 | Der Benutzer hat die Applikation gestartet und befindet sich im «Abfahrtsplan» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld die Station «Luzern, Bahnhof ein». Anschliessend klickt er auf den Button «Suchen». | Die Listbox beinhaltet alle Verbindungen zwischen der Station. | Erfolgreich |

## Testfall Anforderung A004

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Given** | **TODO** | **Then** | **Status** |
| 1 | Der Benutzer hat die Applikation gestartet und befindet sich im «Station» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld, welches sich unter dem Titel «Station» befindet, den Wert «Luz» ein. | Die Combobox öffnet sich und es erscheinen alle Stationen, welche auf den Wert «Luz» zutreffen. | Erfolgreich |

## Testfall Anforderung A005

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Given** | **TODO** | **Then** | **Status** |
| 1 | Der Benutzer hat die Applikation gestartet und befindet sich im «Connection» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld, welches unter dem Titel «Start» befindet den Wert «Luzern, Bahnhof» ein. Im Eingabefeld, welches unter dem Titel «Ziel» steht, gibt der Benutzer den Wert «Obernau, Dorf» ein.  Als Datum wählt er den darauffolgenden Tag ein. In der Combobox wählt er die Uhrzeit «12:00» mittags aus. Anschliessend klickt er auf den Button «Suchen». | In der Listbox erscheinen Verbindungen, welche an dem entsprechenden Datum fahren. | Erfolgreich |

## Testfall Anforderung A006

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Given** | **TODO** | **Then** | **Status** |
| 1 | Der Benutzer hat die Applikation gestartet und befindet sich im «Station» Tab. Weiter verfügt der Benutzer über eine funktionstüchtige Internetverbindung | Der Benutzer gibt im Eingabefeld, welches sich unter der dem Titel «Station» befindet, den Wert «Obernau, Dorf» ein. Anschliessend klickt er auf den Button «Google Maps». | Der Standardbrowser öffnet sich und die Google Maps Webseite wird aufgerufen. Auf der Karte ist die Station gekennzeichnet. | Erfolgreich |

## Test Auswertung

Sämtliche Testfälle sind positiv durchgelaufen. Auch die Unit-Tests sind nach Anpassung der API alle auf grün. Somit ist die Erfolgsquote der Tests bei 8/8 was 100% entspricht

# Installationsanleitung

Das Projekt ist auf Github abgelegt und kann zu jederzeit eingesehen und geklont werden. Anbei wird Schritt für Schritt erklärt wie dieses Projekt von Github heruntergeladen und gestartet werden kann.

1. Öffnen Sie ihrem gewünschten Browser und begeben sie in der Adresszeile folgende URL ein: https://github.com/the-philip-baumann/modul-318-student
2. Klicken sie auf den grünen «Clone or Download» und wählen sie anschliessen die Schaltfläche «Download ZIP».
3. Öffne nun den Windows Explorer, in dem sie in der Windows-Suche nach Explorer suchen.
4. Mit einem rechtsklick auf den Ordner und im Kontextmenu auf alle extrahieren können sie den Ordner extrahieren und als gewöhnlichen Ordner verwenden
5. Anschliessend muss den InstallationWizard.msi Datei öffnen.
6. Informationen durchlesen und mit Button «Weiter klicken».
7. Daraufhin muss der Pfad angegeben werden, an welchem die Applikation installiert werden soll. Ebenfalls kann ausgewählt werden, ob die Applikation für alle Benutzer oder nur den eigenen installiert werden soll. Anschliessend wieder auf den Button «Weiter» klicken
8. Mit dem Klick auf «Weiter» wird die Installation gestartet.
9. Zum Schluss das Fenster schliessen und das Programm zu öffnen (Public Transport Application in der Windows-Suche eingeben).

# Deinstallation

In diesem Kapitel wird kurz erklärt wie die Applikation wieder gelöscht werden kann.

1. Öffne den InstallationWizard, welcher sich im Programm Ordner befindet. Falls dieser gelöscht wurde, wiederhole Schritt 1-5 die Installationsanleitung.
2. Wähle «Public Transport Application entfernen wählen» und klicke auf «Fertig stellen»
3. Anschliessend kann der Installer wieder geschlossen werden.