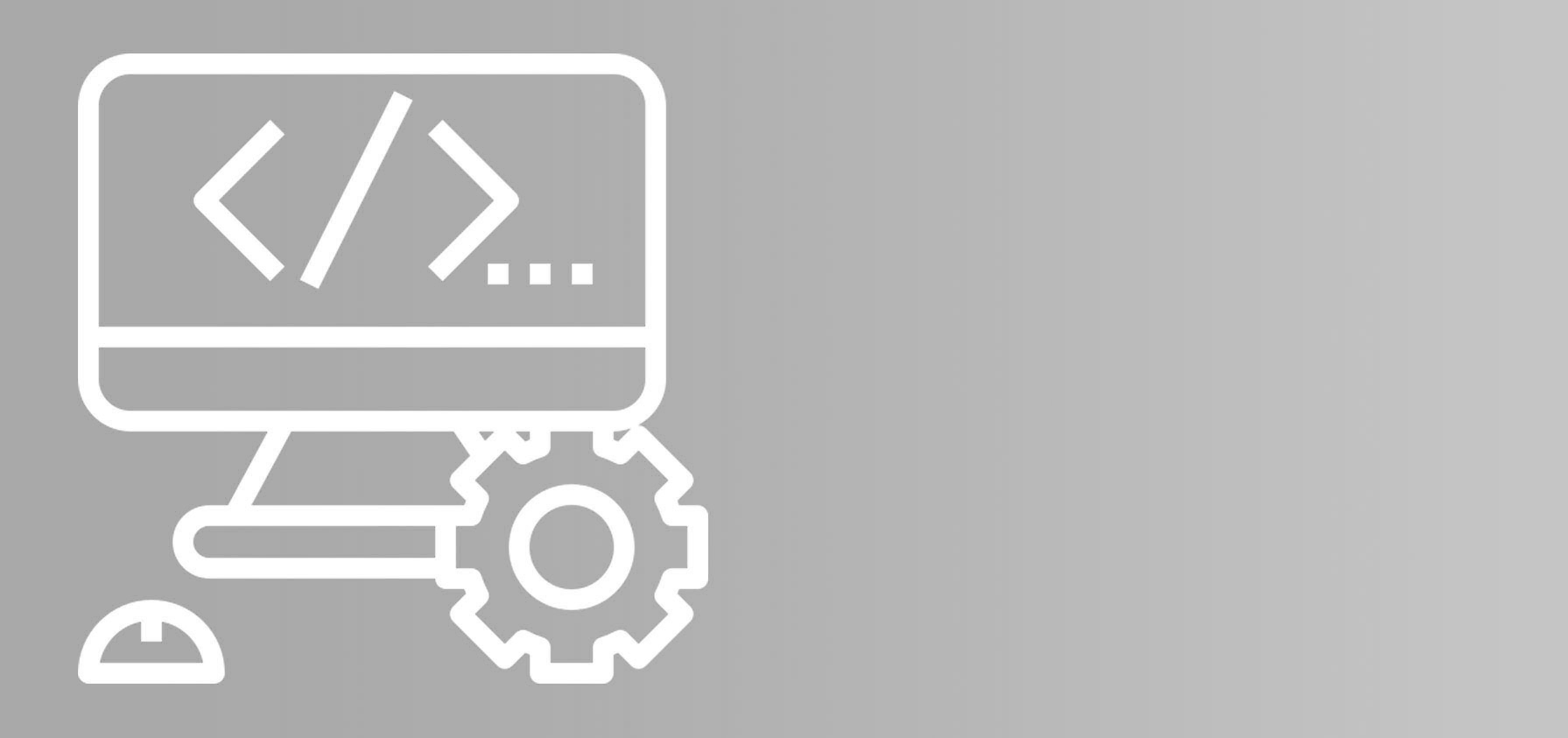
|  |  |  |
| --- | --- | --- |
|  |  | DOKUMENTAtION  MODUL 318 |



|  |  |  |
| --- | --- | --- |
|  |  | Tamila Govduchanova  Maihofstrasse 87  6006 Luzern  S-INF18s  Michael Zurmühle  24.04.2020 |

Inhaltsverzeichnis

[1. Einleitung 4](#_Toc38611014)

[1.1. Modulidentifikation 4](#_Toc38611015)

[1.1.1. Handlungsziele & Handlungsnotwendige Kenntnisse 4](#_Toc38611016)

[2. Management Summary 5](#_Toc38611017)

[3. Zweck des Dokuments 5](#_Toc38611018)

[4. Programmierrichtlinien für C, C++, C# 5](#_Toc38611019)

[4.1. Wartbarkeit, Lesbarkeit, Ästhetik 5](#_Toc38611020)

[4.2. Software-Aktualität 6](#_Toc38611021)

[4.3. Sprache 6](#_Toc38611022)

[4.4. Dokumentation 6](#_Toc38611023)

[4.5. Notationen 6](#_Toc38611024)

[4.6. Zeilenlänge, Sonderzeichen und Umlaute 7](#_Toc38611025)

[4.7. Einrückungen (Indents) 7](#_Toc38611026)

[4.8. Verwendung der geschweiften Klammern 8](#_Toc38611027)

[4.9. Deklarationen und Zuweisungszeichen in derselben Spalte 8](#_Toc38611028)

[4.10. Enumerationen 9](#_Toc38611029)

[5. Welche Funktionen wurden umgesetzt? 10](#_Toc38611030)

[A001 10](#_Toc38611031)

[A002 10](#_Toc38611032)

[A003 10](#_Toc38611033)

[A004 10](#_Toc38611034)

[A005 10](#_Toc38611035)

[Eigene Funktion 10](#_Toc38611036)

[Eigene Funktion 10](#_Toc38611037)

[6. Visualisierung & diagramme 11](#_Toc38611038)

[6.1. Mockup Gui 11](#_Toc38611039)

[6.2. UseCase-Diagramm 12](#_Toc38611040)

[6.3. Aktivitätsdiagramm 15](#_Toc38611041)

[6.4. UML-Diagramm 16](#_Toc38611042)

[7. Warum gehen IT-projekte schief? 17](#_Toc38611043)

[8. Testfälle 18](#_Toc38611044)

[8.1. Test „Stationssuche“ 18](#_Toc38611045)

[8.2. Test „aktuelle Verbindungen suchen“ 18](#_Toc38611046)

[8.3. Test „Verbindungen nach bestimmter Station suchen“ 18](#_Toc38611047)

[8.4. Test „Verbindungen nach bestimmter Zeit suchen“ 18](#_Toc38611048)

[8.5. Test „Reset Button“ 19](#_Toc38611049)

[8.6. Test „Abfahrt & Ankunft tauschen“ 19](#_Toc38611050)

[9. Testprotokoll 19](#_Toc38611051)

[9.1. Test „Stationssuche“ 19](#_Toc38611052)

[9.2. Test „aktuelle Verbindungen suchen“ 20](#_Toc38611053)

[9.3. Test „Verbindungen nach bestimmter Station suchen“ 20](#_Toc38611054)

[9.4. Test „Verbindungen nach bestimmter Zeit suchen“ 21](#_Toc38611055)

[9.5. Test „Reset Button“ 21](#_Toc38611056)

[9.6. Test „Abfahrt & Ankunft tauschen“ 22](#_Toc38611057)

[10. Installationsanleitung 22](#_Toc38611058)

[11. Persönliche Reflexion 22](#_Toc38611059)

[Abbildungsverzeichnis 22](#_Toc38611060)

[Literaturverzeichnis 23](#_Toc38611061)

[SelbststänDigskeitserklärung 23](#_Toc38611062)

# Einleitung

## Modulidentifikation

*Modul 318 - Analysieren und objektbasiert programmieren mit Komponenten*

*Objekt:* Mit einer Komponenten-Bibliothek implementiertes Programm

Lektionen: 40 *Kompetenzfeld:* Application Engineering

### Handlungsziele & Handlungsnotwendige Kenntnisse

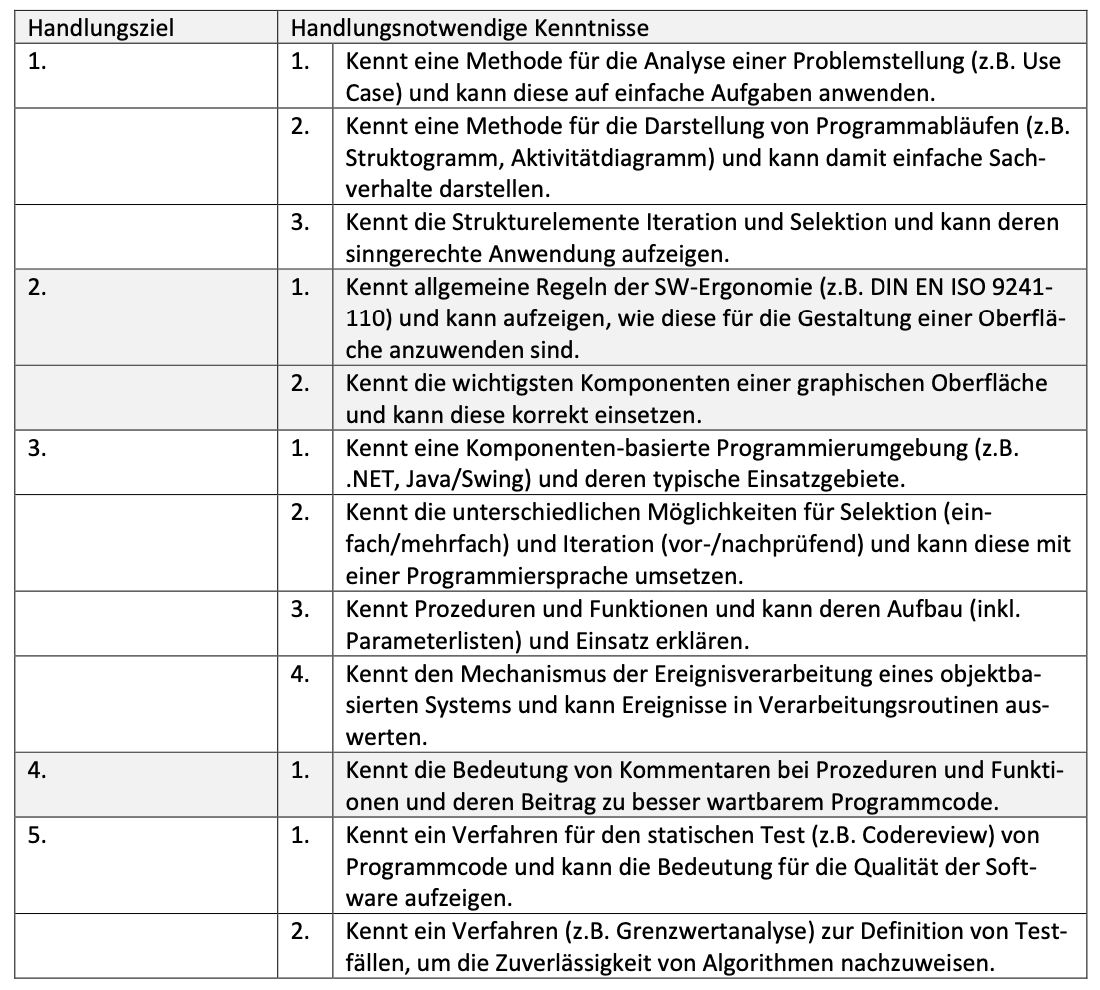


Abbildung 1: Modul\_318\_Modulidentifikation und Handlungsnotwendige Kenntnisse.pdf, 20.04.2020

# Management Summary

Dieses Dokument habe ich während dem ganzen ÜK318 geschrieben und fortlaufend aktualisiert. Wir haben das Thema „objektorientiertes Programmieren“ behandelt. Zuerst haben wir ein paar Testübungen programmiert, die mir später fürs Projekt geholfen haben. Zusätzlich haben wir noch einige Inputs erhalten zum Thema Projekte, Design und Testing. Das Ziel bzw. unser Projekt ist es eine Applikation zu entwickeln, die anhand von einer Datenbank mit öffentlichen Fahrplandaten, Verbindungen raussucht und diese anzeigt. Zusätzlich konnten/mussten wir noch andere Feautures einbauen: nur Abfahrten oder Ankünfte anzeige, Stationen auslesen, etc.

# Zweck des Dokuments

Das Dokument dient zum besseren Verständnis der ganzen Projektarbeit und damit der Instruktor oder jemand anderes nachvollziehen kann, was es mit der ganzen Applikation auf sich hat. In diesem Dokument sind verschiedene Dinge beschrieben: Die umgesetzten Anforderungen, Funktionen, Programmierrichtlinien, Diagramme, Testfälle, etc.

# Programmierrichtlinien für C, C++, C#

**Auszüge aus:** [**https://www.team-electronics.com/Download/D0577-03.pdf**](https://www.team-electronics.com/Download/D0577-03.pdf)

Die folgenden Zeilen beschreiben die grundsätzlichen Standards, welche bei der Software-Erstellung und Wartung berücksichtigt werden sollten. Diese Standards beziehen sich besonders auf die Programmierung in den Sprachen C, C++ und C# - die meisten Regeln gelten aber auch für alle anderen Programmiersprachen.

## Wartbarkeit, Lesbarkeit, Ästhetik

Bei der Erstellung von Software sollten immer folgende Faktoren in Erinnerung gerufen werden:

• Kann ich diesen Source-Code auch in einem Jahr noch verstehen?

• Ist meine Software ordentlich strukturiert/modularisiert oder gibt's haufenweise dubiose Querverbindungen zwischen den Modulen?

• Was passiert, wenn jemand anderer meinen Code betrachtet ("Code–Review") oder Änderungen vornehmen muss – wird er sich zurechtfinden?

• Erscheint meine Software aus "einem Guss" (konsequentes Anwenden von Kommentierungen, Einrückungen, Variablennamen...)?

## Software-Aktualität

Wird eine Software erstellt oder geändert, so muss diese wie auch alle Dokumente und andere Unterlagen auf dem Server gespeichert werden. Nach einer eventuellen Software-Änderung ausser Hause und muss diese Software sofort bei der Rückkehr (und nicht erst Stunden später oder nie) auf den gemeinsamen Server .

## Sprache

Grundsätzlich orientieren sich viele Variablennamen an der englischen Sprache und die Kommentare sind in Deutsch verfasst, sofern nicht Kundenanforderungen dies verhindern.

Wie in der Programmierung üblich, ergibt sich oft ein Mix aus deutsch/englisch, was aber im Rahmen der natürlichen Empfindung zulässig ist. Es gelten allgemein die Regeln der neuen deutschen Rechtschreibung.

## Dokumentation

Lästig, aber notwendig: Inbetriebnahmeanleitungen, Beschreibung von Funktionsbibilotheken, Bedienungsanleitungen, Berechnungs-Dokumentationen (Übersetzungsverhältnisse...) usw. Verweise auf diese Dokumente finden sich im jeweiligen Info-File.

## Notationen

Grundsätzlich existieren diese Notationen (Schreibstile) von Bezeichnern:

|  |  |  |
| --- | --- | --- |
| Notation | Bemerkung | Beispiel |
| Pascal | Der erste Buchstabe jeweils gross. | MyIndentifier |
| Kamel | Am Anfang klein, danach jeweils der erste Buchstabe gross. | myIdentifier |
| Ungarisch | Wie Pascal, aber mit einem vorangestellten kleinengeschriebenen Typ-Präfix | nMyIdentifier |
| Gross | Alles groß, Worttrennung durch Unterstrich | MY\_IDENTIFIER |
| Klein | Alles klein, Worttrennung durch Unterstrich | my\_identifier |

Je nach Bezeichnertyp (Methode, Konstante, Variable...) werden unterschiedliche Notationen angewendet. Die Notation "Klein" sollte nur noch bei Standard C-Programmen verwendet werden und keinesfalls in C++ oder C# - Code Einzug finden.

## Zeilenlänge, Sonderzeichen und Umlaute

Die Zeilenlänge (Breite des Codes) sollte maximal 100 Zeichen betragen. Üblich und historisch bedingt (DOS- Fenster, Nadeldrucker...) sind hingegen 80 Zeichen, was auch im Zeitalter hochauflösender Bildschirme eine sinnvolle Begrenzung darstellt. Sonderzeichen und Umlaute wie "ß", "@", "ä" oder "$" sind natürlich nur innerhalb von Strings oder Kommentaren zulässig und auch erlaubt.

Ausnahme: Info-Files und "alte C-Files" sollten keines dieser Zeichen verwenden, das diese oft mit alten Editoren bearbeitet werden, welche ein anderes Sonderzeichen-Format besitzen.

## Einrückungen (Indents)

Einrückungen erfolgen in Schritten von 2 Leerzeichen, wobei diese nicht als Tabulatorzeichen definiert sein sollten. Dies kann in den meisten Entwicklungsumgebungen mit Einstellungen wie "Insert spaces for tabs" vorgenommen werden.

Beispiel:

namspace Test

{

public class MyClass

{

private int Sum (int nA, int nB)

{

return (nA + nB);

}

}

}

und niemals so:

namspace Test

{

public class MyClass

{

private int Sum (int nA, int nB)

{

return (nA + nB);

}

} }

## Verwendung der geschweiften Klammern

|  |  |
| --- | --- |
| gut | schlecht |
| if (nX > 0)  {  for (int i = 0; i < 10; i++)  {  adValueX[i]++;  adValueY[i]--: }  } | if (nX > 0) {  for (int i = 0; i < 10; i++) {  adValueX[i]++;  adValueY[i]--: }  } |

Code-Blöcke, welche nur aus einer einzigen Anweisung bestehen, dürfen auf die geschweiften Klammern verzichten:

|  |  |
| --- | --- |
| Richtig | Ebenso richtig |
| if (nX > 0)  dValue = 1; | if (nX > 0)  {  dValue = 1;  } |

## Deklarationen und Zuweisungszeichen in derselben Spalte

Typdeklarationen, Klasseninstanzen, Zuweisungszeichen usw. sollten für zusammenhängende Code- Abschnitte in derselben Spalte stehen.

|  |  |
| --- | --- |
| gut | schlecht |
| double dTestVar1;  float fTestVar2;  long nValue;  int nValueLessOne;  bool bEnable;  dTestVar1 = 1.89;  fTestVar2 = -17.89;  nValue = (long)dTestVar1;  nValueLessOne = (int) nValue - 1;  bEnable. = true; | double dTestVar1;  float fTestVar2;  long nValue;  int nValueLessOne;ß  bool bEnable;  dTestVar1 = 1.89; fTestVar2 = -17.89; nValue = (long)dTestVar1; nValueLessOne = nValue - 1; bEnable = true; |

## Enumerationen

Enumerationen erhalten das Präfix E. Die einzelnen Enumerationswerte werden je nach Typ entweder in den Notationen "Groß" oder "Pascal" definiert. "Groß" wird verwendet, wenn die Enumertationswerte den Charakter von Konstanten haben, ansonsten kann die Notation Pascal benutzt werden.

Instanzen von Enumerationen erhalten das Präfix e.

Beispiel:

|  |  |
| --- | --- |
| Enumeration mit Konstanten | Wertunabhängige Enumeration |
| public enum EDigitalInputs : ushort  {  START = 1<<0, n  STOP = 1<<1,  nNOTAUS = 1<<2,  }  EDigitalInputs eDiBoard1;  if ((eDiBoard1 & EDigitalInputs.START) != 0)  {  // ...  } | public enum EColor  {  Red,  Blue,  Green,  }  EColor eColor;  eColor = EColor.Red; |

# Welche Funktionen wurden umgesetzt?

A001 Als ÖV-Benutzer möchte ich Start- und Endstation mittels Textsuche suchen können, damit ich nicht alle Stationsnamen auswendig lernen muss.

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.

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.

A004 Als ÖV-Benutzer möchte ich, dass schon während meiner Eingabe erste Suchresultate erscheinen, damit ich effizienter nach Stationen suchen kann.

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.

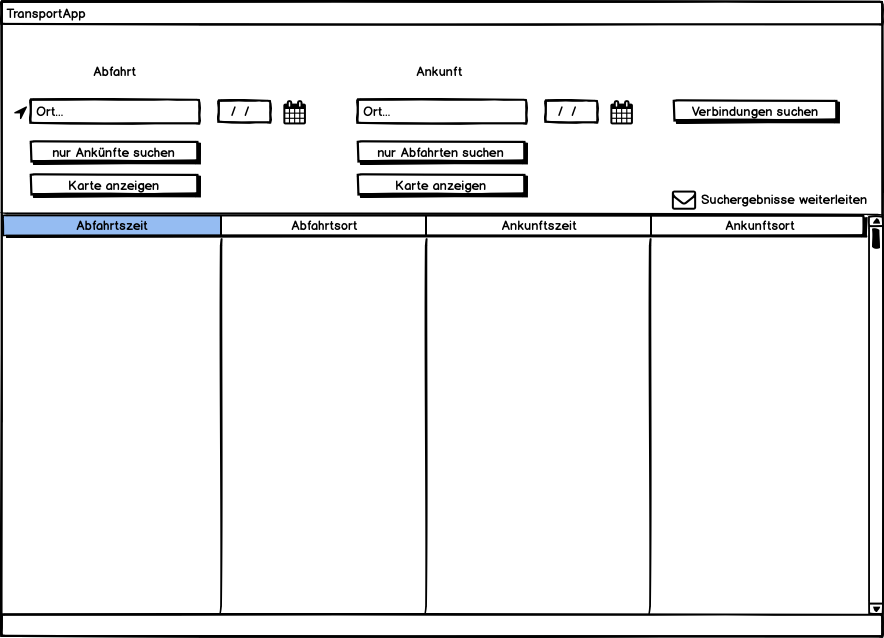
Eigene Funktion: Als ÖV-Benutzer möchte ich einen „Reset“-Button haben.

Eigene Funktion: Als ÖV-Benutzer möchte ich Start- & Endstation tauschen können.

# Visualisierung & diagramme

## Mockup Gui

„Ein Vorführmodell ist ein komplettes Produkt oder eine Attrappe, die genutzt wird, um Design und/oder Funktionen eines geplanten oder bereits eingeführten Produktes zu demonstrieren. Es ist oft ein maßstäbliches Modell bzw. eine Nachbildung zu Präsentationszwecken.» (Wikipedia, Vorführmodell)





= zusätzlich im Verlauf der Arbeit hinzugefügt



RESET

= schlussendlich umgesetzt



Abbildung 2: Mockup Gui erstellt mit "Balsamiq", 21.04.2020

## UseCase-Diagramm

„Ein Anwendungsfalldiagramm, auch Nutzfalldiagramm, ist eine der 14 Diagrammarten der Unified Modeling Language, einer Sprache für die Modellierung der Strukturen und des Verhaltens von Software- und anderen Systemen. Es stellt Anwendungsfälle und Akteure mit ihren jeweiligen Abhängigkeiten und Beziehungen dar.“ (Wikipedia, Anwendungsfalldiagramm)

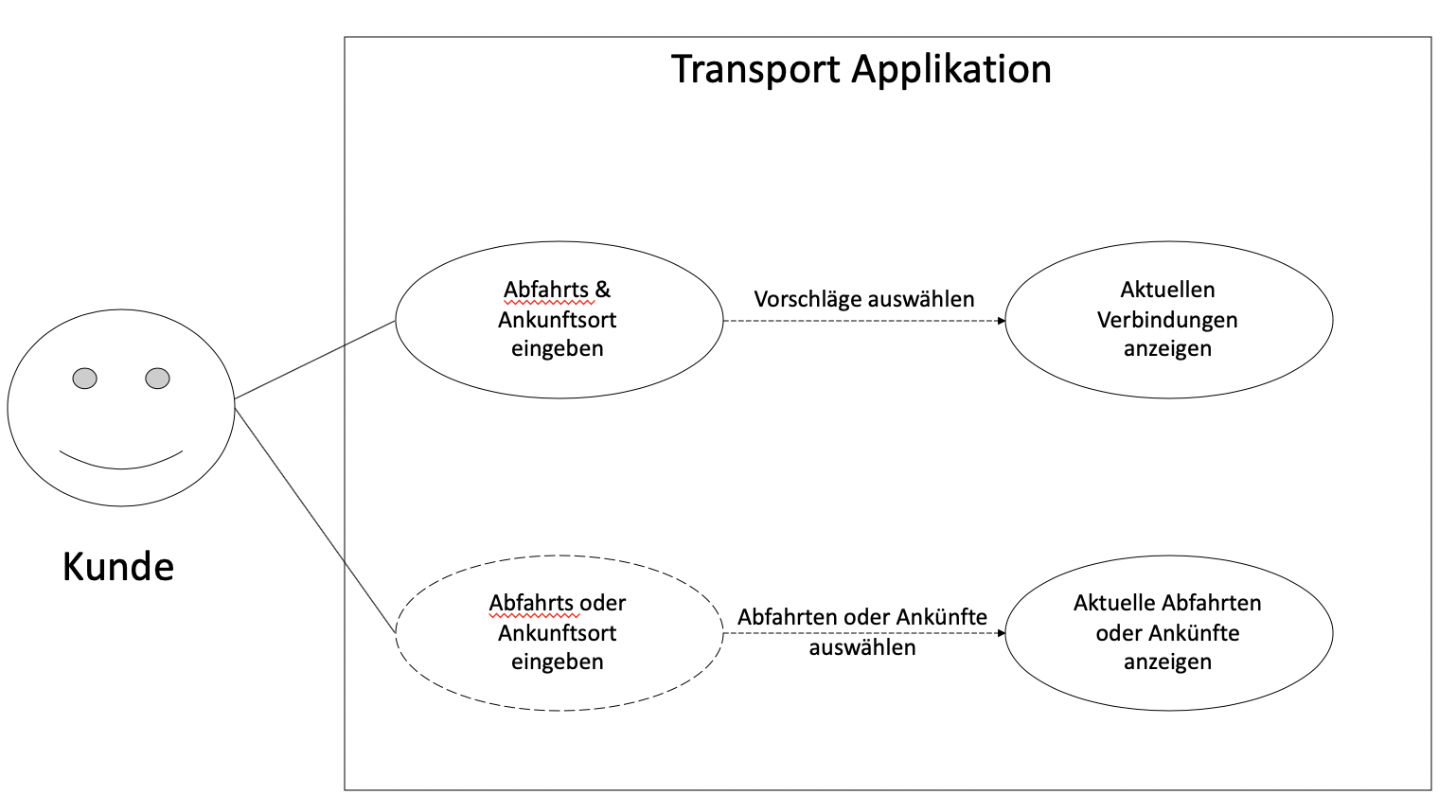


Abbildung 3: UseCase-Diagramm, 21.04.2020

|  |  |
| --- | --- |
| **Use Case 1** | **Abfahrts & Ankunftsort eingeben** |
| **Beschreibung** | Der Benutzer gibt seinen Abfahrts- und Ankunftsort in das vorgegebene Feld ein. |
| **Akteur(e)** | ÖV-Benutzer |
| **Auslöser** | Der ÖV-Benutzer möchte von A nach B reisen. |
| **Vorbedingungen** | Der ÖV-Benutzer muss Abfahrts- & Ankunftsort kennen. |
| **Ablauf** | 1. Programm öffnen 2. Abfahrtsort eingeben (Vorschläge auswählen) 3. Ankunftsort eingeben (Vorschläge auswählen) 4. «Verbindungen suchen» drücken |
| **Alternativer Ablauf #1** | Kein |
| **Ergebnis** | Verbindungen werden angezeigt. |

|  |  |
| --- | --- |
| **Use Case 2** | **Aktuelle Verbindungen anzeigen** |
| **Beschreibung** | Dem Benutzer werden alle Verbindungen angezeigt, die zwischen A und B verkehren. |
| **Akteur(e)** | ÖV-Benutzer |
| **Auslöser** | Der ÖV-Benutzer möchte seine nächsten Verbindungen sehen. |
| **Vorbedingungen** | Der ÖV-Benutzer muss Abfahrts- & Ankunftsort kennen und den «Verbindungen suchen» -Button gedrückt haben. |
| **Ablauf** | 1. Use Case Fall «Abfahrts & Ankunftsort eingeben» 2. Verbindungen werden angezeigt. |
| **Alternativer Ablauf #1** | Kein |
| **Ergebnis** | Verbindungen werden angezeigt. |

|  |  |
| --- | --- |
| **Use Case 3** | **Abfahrts oder Ankunftsort eingeben** |
| **Beschreibung** | Der Benutzer gibt seinen Abfahrts- oder Ankunftsort in das vorgegebene Feld ein. |
| **Akteur(e)** | ÖV-Benutzer |
| **Auslöser** | Der ÖV-Benutzer möchte alle Abfahrten oder Ankünfte sehen. |
| **Vorbedingungen** | Der ÖV-Benutzer muss Abfahrts- oder Ankunftsort kennen. |
| **Ablauf** | 1. Programm öffnen 2. Abfahrts- oder Ankunftsort eingeben (Vorschläge auswählen) 3. «nur Ankünfte suchen» oder «nur Abfahrten suchen» drücken |
| **Alternativer Ablauf #1** | Kein |
| **Ergebnis** | Abfahrten oder Ankünfte werden angezeigt. |

|  |  |
| --- | --- |
| **Use Case 4** | **Aktuelle Abfahrten oder Ankünfte anzeigen** |
| **Beschreibung** | Dem Benutzer werden alle Abfahrten oder Ankünfte angezeigt, die von/nach A und B verkehren. |
| **Akteur(e)** | ÖV-Benutzer |
| **Auslöser** | Der ÖV-Benutzer möchte seine nächsten Abfahrten/Ankünfte. |
| **Vorbedingungen** | Der ÖV-Benutzer muss Abfahrts- oder Ankunftsort kennen und den entsprechenden Button gedrückt haben. |
| **Ablauf** | 1. Use Case Fall «Abfahrts oder Ankunftsort eingeben» 2. Abfahrten/Ankünfte werden angezeigt. |
| **Alternativer Ablauf #1** | Kein |
| **Ergebnis** | Abfahrten oder Ankünfte werden angezeigt. |

## Aktivitätsdiagramm

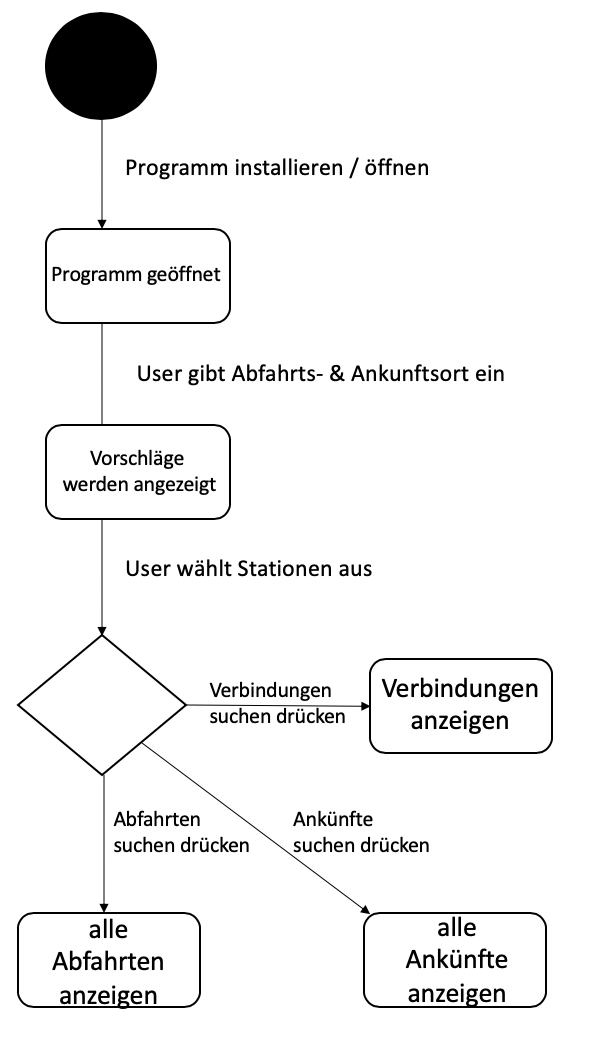


Abbildung 4: Aktivitätsdiagramm, 21.04.2020

„Ein Aktivitätsdiagramm ist ein Verhaltensdiagramm der Unified Modeling Language, einer Modellierungssprache für Software und andere Systeme, und stellt die Vernetzung von elementaren Aktionen und deren Verbindungen mit Kontroll- und Datenflüssen grafisch dar.“ (Wikipedia, Aktivitätsdiagramm)

## UML-Diagramm

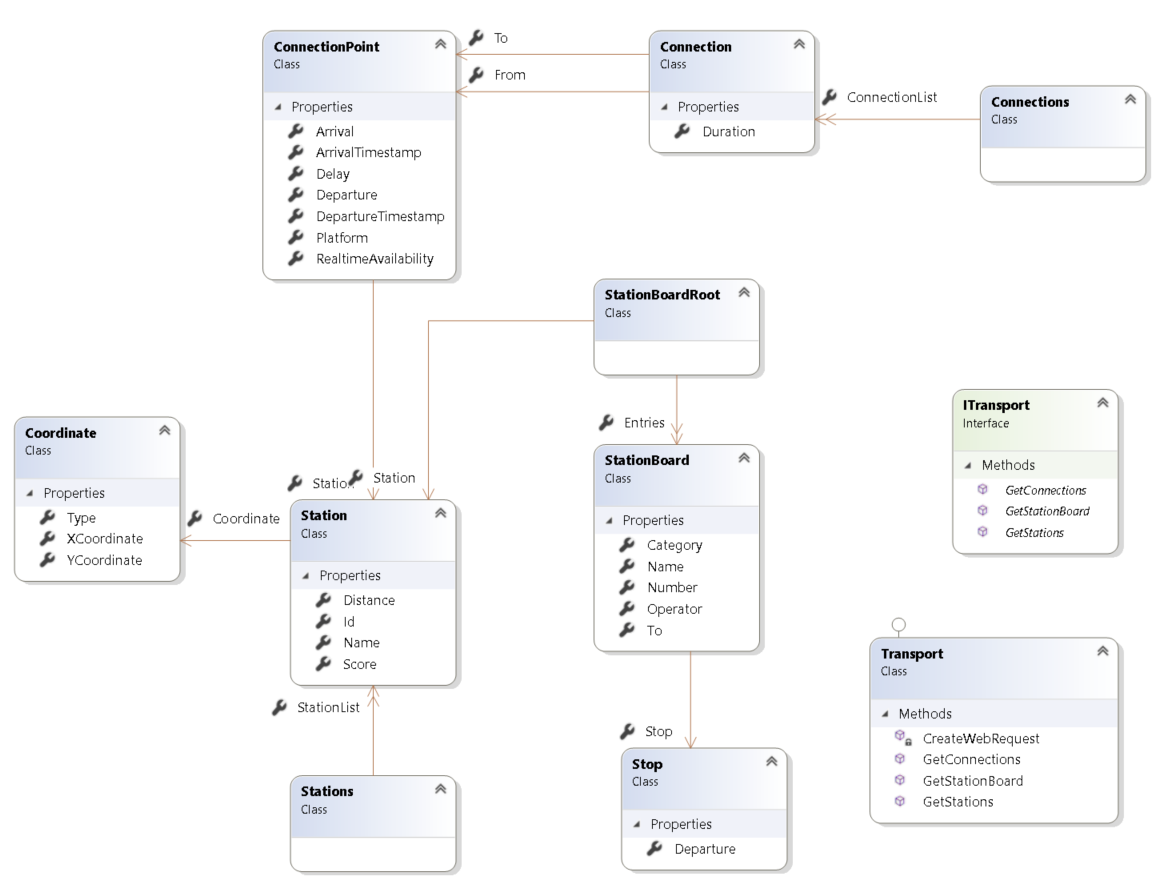


Abbildung 5: UML-Diagramm, 21.04.2020

„Die Unified Modeling Language, kurz UML, ist eine grafische Modellierungssprache zur Spezifikation, Konstruktion, Dokumentation und Visualisierung von Software-Teilen und anderen Systemen. Sie wird von der Object Management Group entwickelt und ist sowohl von ihr als auch von der ISO genormt.“ (Wikipedia, Unified Modeling Language)

# Warum gehen IT-projekte schief?

|  |  |
| --- | --- |
| **Problem** | **Lösung** |
| fehlende Erfahrung in der Leitung | verstärkte Zusammenarbeit, flache “Rangordnung” |
| zu wenig Planung | mehr Zeit ins Planen investieren (verhindert Scheitern aufgrund Kosten, etc.) |
| nicht genug intensiv getestet | mehr Tests machen |
| wenig Engagement | kann super geplant sein, aber bringt nichts wenn kein Engagement gezeigt wird |
| schlechte Kommunikation (Hierarchie) | bei schlechter Organisation funktioniert auch die Kommunikation nach oben/unten nicht |
| kein klares Ziel | Überlegen, was und wer man mit der Software erreichen möchte |
| unklare Anforderungen | von Anfang an definieren |

# Testfälle

## „Ein Testfall (engl. Test case) beschreibt einen elementaren, funktionalen Softwaretest, der der Überprüfung einer z. B. in einer Spezifikation zugesicherten Eigenschaft eines Testobjektes dient. Testfälle werden unter Anwendung von Testmethoden und geeigneten Software-Hilfsmitteln erstellt.“ (Wikipedia, Testfall)

## Test „Stationssuche“

Der User möchte mit dem Textfeld Abfahrts- & Ankunftsort suchen.

|  |  |  |
| --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |

## Test „aktuelle Verbindungen suchen“

Der User möchte die aktuellen Verbindungen zwischen Abfahrts- & Ankunftsort sehen.

|  |  |  |
| --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |
| **3** | alle Verbindungen suchen drücken | Es werden die nächsten 4-5 Verbindungen zwischen „Luzern, Bahnhof“ und „Luzern, Maihof“ angezeigt |

## Test „Verbindungen nach bestimmter Station suchen“

Der User möchte die Verbindungen ab Abfahrtsort angezeigt bekommen.

|  |  |  |
| --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |
| **2** | „alle Abfahrten anzeigen“ drücken | alle aktuellen Verbindungen ab „Luzern, Bahnhof“ werden angezeigt. |

Der User möchte die Verbindungen an Ankunftsort angezeigt bekommen.

|  |  |  |
| --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** |
| **1** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |
| **2** | „alle Ankünfte anzeigen“ drücken | alle aktuellen Verbindungen an „Luzern, Maihof“ werden angezeigt. |

## Test „Verbindungen nach bestimmter Zeit suchen“

Der User möchte die Verbindungen zwischen Abfahrts- & Ankunftsort nach Zeit suchen.

|  |  |  |
| --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |
| **3** | Abfahrtszeit eingeben | Abfahrtszeit wurde im dafür vorgesehenen Feld eingegeben. |
| **4** | „alle Verbindungen suchen“ drücken | Es werden die aktuellen Verbindungen zwischen „Luzern, Bahnhof“ und „Luzern, Maihof“ angezeigt. |

## Test „Reset Button“

Der User möchte seine Ortseingaben nicht manuell löschen müssen, er fordert einen Button.

|  |  |  |
| --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |
| **3** | „RESET“-Button drücken | „Luzern, Bahnhof“ und „Luzern, Maihof“ werden aus dem Feld gelöscht. |

## Test „Abfahrt & Ankunft tauschen“

Der User möchte den Abfahrts- & Ankunftsort tauschen.

|  |  |  |
| --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |
| **3** | „SWAP“Button drücken | „Luzern, Bahnhof“ und „Luzern, Maihof“ werden getauscht. |

# Testprotokoll

## Test „Stationssuche“

Durchgeführt von: Tamila Govduchanova

Durchgeführt am: 23.04.2020

Der User möchte mit dem Textfeld Abfahrts- & Ankunftsort suchen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** | **tatsächliches Resultat** | **Status** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |

## Test „aktuelle Verbindungen suchen“

Durchgeführt von: Tamila Govduchanova

Durchgeführt am: 23.04.2020

Der User möchte die aktuellen Verbindungen zwischen Abfahrts- & Ankunftsort sehen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** | **tatsächliches Resultat** | **Status** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **3** | alle Verbindungen suchen drücken | Es werden die nächsten 4-5 Verbindungen zwischen „Luzern, Bahnhof“ und „Luzern, Maihof“ angezeigt |  | Häkchen |

## Test „Verbindungen nach bestimmter Station suchen“

Durchgeführt von: Tamila Govduchanova

Durchgeführt am: 23.04.2020

Der User möchte die Verbindungen ab Abfahrtsort angezeigt bekommen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** | **tatsächliches Resultat** | **Status** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **2** | „alle Abfahrten anzeigen“ drücken | alle aktuellen Verbindungen ab „Luzern, Bahnhof“ werden angezeigt. |  | Häkchen |

Durchgeführt von: Tamila Govduchanova

Durchgeführt am: 23.04.2020

Der User möchte die Verbindungen an Ankunftsort angezeigt bekommen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** | **tatsächliches Resultat** | **Status** |
| **1** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **2** | „alle Ankünfte anzeigen“ drücken | alle aktuellen Verbindungen an „Luzern, Maihof“ werden angezeigt. |  | Häkchen |

## Test „Verbindungen nach bestimmter Zeit suchen“

Durchgeführt von: Tamila Govduchanova

Durchgeführt am: 23.04.2020

Der User möchte die Verbindungen zwischen Abfahrts- & Ankunftsort nach Zeit suchen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** | **tatsächliches Resultat** | **Status** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **3** | Abfahrtszeit eingeben | Abfahrtszeit wurde im dafür vorgesehenen Feld eingegeben. |  | Häkchen |
| **4** | „alle Verbindungen suchen“ drücken | Es werden die aktuellen Verbindungen zwischen „Luzern, Bahnhof“ und „Luzern, Maihof“ angezeigt. |  | Häkchen |

## Test „Reset Button“

Durchgeführt von: Tamila Govduchanova

Durchgeführt am: 23.04.2020

Der User möchte seine Ortseingaben nicht manuell löschen müssen, er fordert einen Button.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** | **tatsächliches Resultat** | **Status** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **3** | „RESET“-Button drücken | „Luzern, Bahnhof“ und „Luzern, Maihof“ werden aus dem Feld gelöscht. |  | Häkchen |

## Test „Abfahrt & Ankunft tauschen“

Durchgeführt von: Tamila Govduchanova

Durchgeführt am: 23.04.2020

Der User möchte den Abfahrts- & Ankunftsort tauschen.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schritt** | **Aktivität** | **erwartetes Resultat** | **tatsächliches Resultat** | **Status** |
| **1** | Abfahrtsort „Luzern, Bahnhof“ eingeben | „Luzern, Bahnhof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **2** | Ankunftsort „Luzern, Maihof“ eingeben | „Luzern, Maihof“ wird vom Autocomplete erkannt und eingegeben. |  | Häkchen |
| **3** | „SWAP“Button drücken | „Luzern, Bahnhof“ und „Luzern, Maihof“ werden getauscht. |  | Häkchen |

# Installationsanleitung

# Abbildungsverzeichnis

[Abbildung 1: Modul\_318\_Modulidentifikation und Handlungsnotwendige Kenntnisse.pdf, 20.04.2020 4](#_Toc38531268)

[Abbildung 2: Mockup Gui erstellt mit "Balsamiq", 21.04.2020 10](#_Toc38531269)

[Abbildung 3: UseCase-Diagramm, 21.04.2020 11](#_Toc38531270)

[Abbildung 4: Aktivitätsdiagramm, 21.04.2020 14](#_Toc38531271)

[Abbildung 5: UML-Diagramm, 21.04.2020 15](#_Toc38531272)

# SelbststänDigskeitserklärung

*Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne unerlaubte fremde Hilfe verfasst und keine anderen Hilfsmittel als die angegebenen verwendet habe.*

*Insbesondere versichere ich, dass ich alle wörtlichen und sinngemässen Übernahmen aus anderen Werken als solche kenntlich gemacht habe.*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Ort, Datum Unterschrift*