[Document title]

[Document subtitle]

BBZW-Sursee; Ziltener Noah

[Year]

Table of Contents

[Einleitung 2](#_Toc34727662)

[Aufgabenstellung 2](#_Toc34727663)

[Anforderungen 2](#_Toc34727664)

[Zweck 2](#_Toc34727665)

[Funktionen 2](#_Toc34727666)

[Bugs 2](#_Toc34727667)

[Planung 2](#_Toc34727668)

[Use Cases 2](#_Toc34727669)

[Aktivitäten Diagramm 4](#_Toc34727670)

[Mockups 4](#_Toc34727671)

[Programmrichtlinien 4](#_Toc34727672)

[Naming 4](#_Toc34727673)

[Decleation 4](#_Toc34727674)

[Comments 4](#_Toc34727675)

[Statements 4](#_Toc34727676)

[Testfälle 5](#_Toc34727677)

[Testprotokoll 5](#_Toc34727678)

[Installations Anleitung 5](#_Toc34727679)

# Einleitung

## Aufgabenstellung

Unsere Aufgabe ist es eine Desktop Applikation mit Hilfe der Transport API zu Programmieren. Wir haben mehrere Anforderungen bekommen, die nach Wichtigkeit priorisiert sind. Uns ist es selber überlasen welche wir umsetzten möchten doch sollten die Anforderungen mit Priorität 1 auf jeden Fall umgesetzt werden. Mein Ziel ist es bis zur Abgabe ein fertiges Programm zu haben und die Code Konventionen einzuhalten. Auch möchte ich Die Anforderung mit Priorität 1 und 2 abgeschlossen haben.

## Anforderungen

## Zweck

Die Dokumentation dient dazu die Projektarbeit im Rahmen des ÜK 318 zu dokumentieren. Das Dokument enthält die Planung, allgemeine Informationen über das Projekt und das Testing der Applikation. Auch ist eine Installationsanleitung am Ende des Dokumentes enthalten.

# Funktionen

# Bugs

# Planung

## Use Cases

|  |  |
| --- | --- |
| USe Case | 1, Start- & End-Station suchen |
| Beschreibung | Damit der User sich keine Stationen merken muss kann er nach Stationen suchen. Die suche muss über eine Texteingabe erfolgen. Es werden auch schon Stationen angezeigt, wenn der Name noch nicht den ganzen Namen eingegeben hat. |
| Akteur | SBB User |
| Auslöser | SBB User möchte, Namen der Station wissen und nicht selber auswendig lernen. |
| Vobedingungen | Internetverbindung |
| Ablauf | 1. User öffnet Programm 2. User klickt entweder auf «Abfahrstafel» oder «Verbindungen anzeigen» 3. User gibt Namen von Station ein. 4. User klickt auf Suchen 5. User bekommt Resultat auch wenn er nicht den kompletten Namen eingegeben hat |
| Ergebnis | Stationen werden trotzdem gefunden auch wenn der User nicht den Kompletten Namen eingegeben hat. z.B. Meierskappel wird zu Meierskappel Dorfplatz. |

|  |  |
| --- | --- |
| USe Case | 1, Verbindungen anzeigen |
| Beschreibung | Dem User werden die nächsten 5 Verbindungen zwischen der Start- und End-Station angezeigt. |
| Akteur | SBB User |
| Auslöser | SBB User möchte, die wissen was die besten Verbindungen zwischen den eingegebenen Stationen wissen. |
| Vobedingungen | Internetverbindung |
| Ablauf | 1. User öffnet Programm 2. User klickt entweder auf «Verbindungen anzeigen» 3. User gibt Namen von Start- und Endstation ein. 4. User klickt auf Verbindungen anzeigen 5. User bekommt Liste mit den 5 nächsten Verbindungen zwischen den 2 Stationen angezeigt. |
| Ergebnis | User bekommt Liste mit den 5 nächsten Verbindungen zwischen den 2 Stationen angezeigt. |

|  |  |
| --- | --- |
| USe Case | 1, Abfahrtstafel |
| Beschreibung | Dem User werden alle Anschlussmöglichkeiten einer Station angezeigt. Es sollen alle Verbindungen, egal in welche Richtung, welche an dieser Station anhalten angezeigt werden. |
| Akteur | SBB User |
| Auslöser | SBB User möchte alle Verbindungen einer Station in alle Richtungen wissen. |
| Vobedingungen | Internetverbindung |
| Ablauf | 1. User öffnet Programm 2. User klickt entweder auf «Abfahrstafel» 3. User gibt Namen von Startstation ein. 4. User klickt auf Abfahrstafel anzeigen 5. User bekommt Liste mit den nächsten Verbindungen die von der eingegeben Stationen ausgehen angezeigt. |
| Ergebnis | User bekommt Liste mit den nächsten Verbindungen die von der eingegeben Stationen ausgehen angezeigt. Die Tafel funktioniert genauso wie die Tafel die an den Bahnhöfen stehen. |

|  |  |
| --- | --- |
| USe Case | 1, Abfahrtstafel |
| Beschreibung | Dem User werden alle Anschlussmöglichkeiten einer Station angezeigt. Es sollen alle Verbindungen, egal in welche Richtung, welche an dieser Station anhalten angezeigt werden. |
| Akteur | SBB User |
| Auslöser | SBB User möchte alle Verbindungen einer Station in alle Richtungen wissen. |
| Vobedingungen | Internetverbindung |
| Ablauf | 1. User öffnet Programm 2. User klickt entweder auf «Abfahrstafel» 3. User gibt Namen von Startstation ein. 4. User klickt auf Abfahrstafel anzeigen 5. User bekommt Liste mit den nächsten Verbindungen die von der eingegeben Stationen ausgehen angezeigt. |
| Ergebnis | User bekommt Liste mit den nächsten Verbindungen die von der eingegeben Stationen ausgehen angezeigt. Die Tafel funktioniert genauso wie die Tafel die an den Bahnhöfen stehen. |

## Aktivitäten Diagramm

## Mockups

# Programmrichtlinien

## Naming

|  |  |  |
| --- | --- | --- |
| Stelle | Regel | Beispiel |
| Allgemein | Englische Bezeichnungen, aussagen was für Zweck sie haben. |  |
| Variablen | CamelCase, Lokale und Membervariabel werden nicht unterschieden. | numberOfConnections |
| Methoden | PascalCase | AreStationFieldsFilled() |
| Klassen | Gross | Station |
| GUI | Fängt mit Kürzel von GUI-Element an. | btnSearchConnections |

## Decleation

|  |  |
| --- | --- |
| Regel | Beispiel |
| Globale Variablen unterhalb des Klassenkopfes definieren und im Konstruktor deklarieren | int varOne;  Konstruktor() {  varOne = 0  } |
| Lokale variablen unterhalb des Methodenkopfes definieren und deklarieren | Methode() {  Int varOne = 0;  } |

## Comments

|  |  |
| --- | --- |
| Stelle |  |
| Code sollte auch ohne Kommentare lesbar sein. Keine Kommentare, nur bei Komplexen Code Komentare | while (x == y)  {  somethingSimple();  // Erklährung  somethingComplicated();  } |

## Statements

|  |  |
| --- | --- |
| Regel |  |
| Allman, Klammern immer auf neu Line bei Statements | while (x == y)  {  something();  somethingelse();  } |

# Testfälle

## Testprotokoll

# Installations Anleitung