**ÜK 318-1 - Analysieren und objektbasiert programmieren**

Autor: Adrian Cerdeira  
Firma: viag Müller informatik  
Gruppe: (BIVO 2014) SJ 18/19 - Gruppe 1 APP  
Datum: 04.12.2018

Inhaltsverzeichnis

[**1.** **Einleitung** 3](#_Toc531702098)

[1.1 Zielsetzung 3](#_Toc531702099)

[**2.** **Funktionen** 4](#_Toc531702100)

[2.1 Umgesetzte und teilweise umgesetzte Funktionen 4](#_Toc531702101)

[**3.** **Applikations-Features** 5](#_Toc531702102)

[3.1 Navigation zu anderen Verbindungen/Abfahrten 5](#_Toc531702103)

[3.2 Chromium-Browser & Google-Maps Routenplaner 5](#_Toc531702104)

[**4.** **Use Cases, Aktivitätsdiagramme und Klassendiagramm** 6](#_Toc531702105)

[4.1 Use Case Diagramm 6](#_Toc531702106)

[4.1.1 Use Case Beschreibung 6](#_Toc531702107)

[4.2 Aktivitätsdiagramm 7](#_Toc531702108)

[4.3 Klassendiagramm 7](#_Toc531702109)

[**5.** **Code Guidelines** 8](#_Toc531702110)

[5.1 Naming Conventions 8](#_Toc531702111)

[**6.** **Mockup** 9](#_Toc531702112)

[**7.** **Testing** 10](#_Toc531702113)

[**6.** **Anleitung** 11](#_Toc531702114)

[6.1 Applikation installieren 11](#_Toc531702115)

[6.2 Applikation über Windows deinstallieren 12](#_Toc531702116)

[6.3 Applikation über Setup deinstallieren 13](#_Toc531702117)

[**7.** **Abbildungsverzeichnis** 14](#_Toc531702118)

[**8.** **Anhang** 14](#_Toc531702119)

[8.1 Applikations-Anforderungen 14](#_Toc531702120)

# **Einleitung**

Im Überbetrieblichen Kurs muss ich eine Dokumentation für unseres Projekt «Öv-Applikation» in Bezug auf das Thema «Analysieren und objektbasiert programmieren» erstellen.

Ich werde in meinem Dokumentation über die Umsetzung der Funktionen, Use Cases Diagramm, Mockup und Tests sprechen. Zum Schluss gibt’s noch eine Anleitung, wie man mein Programm installiert und deinstalliert.

## Zielsetzung

Ich möchte diese Arbeit natürlich mit einer erfolgreichen Note abschliessen. Auch möchte ich zeigen, dass man mit einer strukturierten Vorgehensweise ein erfolgreiches Programm schreiben und veröffentlichen kann. Natürlich sollte das Programm keine Fehler auslösen und möglichst Benutzerfreundlich sein.

# **Funktionen**

## Umgesetzte und teilweise umgesetzte Funktionen

Die meisten Anforderungen konnte ich erfolgreich umsetzen (alle Anforderungen im Anhang). Es war vieles kompliziert und mühsam. Aber zum Schluss konnte ich alles wie gewünscht umsetzen. Was mir teilweise geglückt ist, sind folgende Anforderungen:

* A006: Als ÖV-Benutzer möchte ich sehen, wo sich eine Station befindet, damit ich mir besser vorstellen kann, wie die Situation vor Ort aussieht.
* A007:Als ÖV-Benutzer möchte Stationen finden, die sich ganz in der Nähe meiner aktuellen Position befinden, damit ich schnell einen Anschluss erreichen kann.

Dabei war bei beiden das Problem die Google-Maps-Integration.

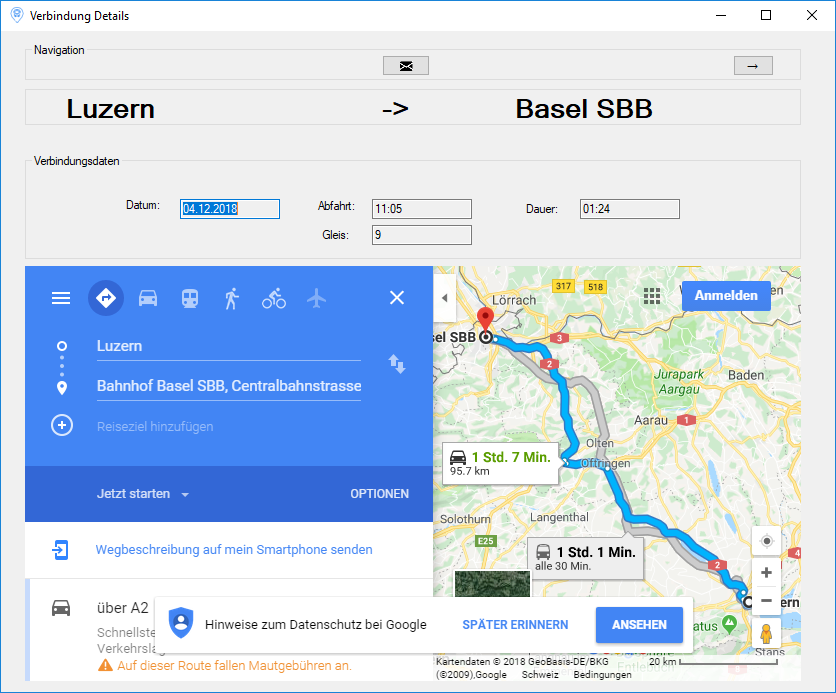
Für die Anforderung A006 habe ich extra eine Chrome-Engine mit CefSharp integriert. Damit die Route/Standort nicht per Link abgerufen wird, sondern per Webbrowser angezeigt wird. Leider lief der Webbrowser mit Internet-Explorer, der den Link-Parameter «Mein Standort» nicht unterstützt und allgemein eine sehr schlechte Performance hatte. Deshalb der umstieg auf Chrome-Engine «Chromium-Browser». Der sehr viele Vorteile hatte: Chrome muss nicht installiert sein, simpler Code und sehr gute Performance. Der einzige Nachteil war das Importieren der Assemby. Die im Debug und schlussendlich im Setup integriert werden mussten.

Abbildung 1: Integrierung Chromium Browser

Bei der Anforderung A007 habe ich ein einfachen Link zu Google-Maps mit dem Parameter «Mein Standort» und der Zielstation. Dies ist wie bereits erwähnt für Benutzer die keinen Chrome haben, suboptimal.

# **Applikations-Features**

## Navigation zu anderen Verbindungen/Abfahrten

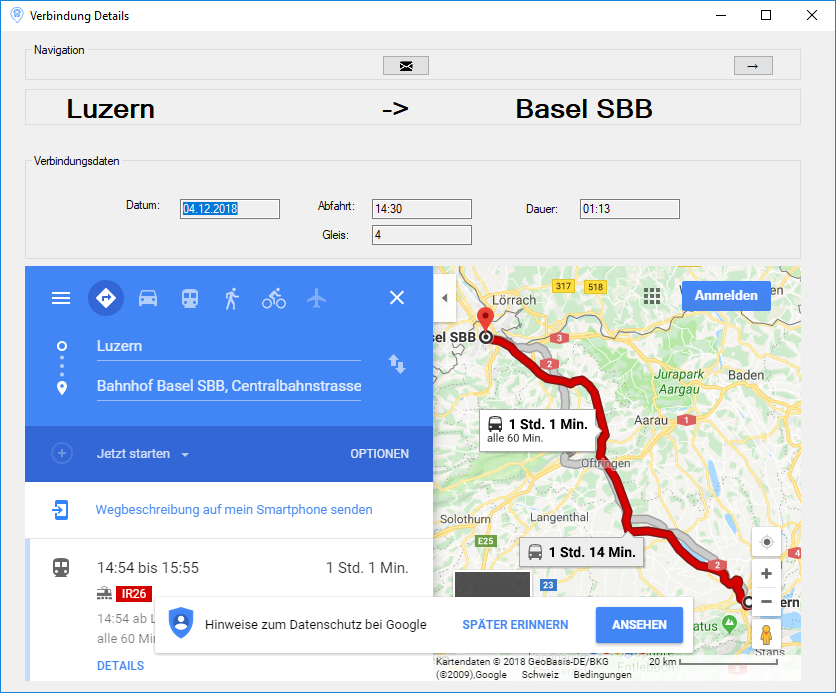
Mit den Klicken auf den Button -> und <- kann man zwischen den anderen Verbindungen / Abfahrten navigieren. Dabei wurde programmiertechnisch zwei separate Formulare verwendet. Eine für die Verbindung-Details und eine für die Abfahrt-Details. Ich konnte mit separaten Funktionen viel Code sparen und dies einfach wiederverwenden. Der Code bei den Formularen gefällt mir persönlich am Besten und ist sehr übersichtlich. Damit konnte ich die Daten Aktualisierung, Buttons beim letzten und ersten Element unsichtbar machen, Mail-Versendung und die Google-Maps Chrome-Engine-Integration komplett automatisieren.

Abbildung 2: Chromium-Browser

## Chromium-Browser & Google-Maps Routenplaner

Die meisten haben einen einfachen Link zu Google-Maps gemacht. Ich ermögliche den User mit diesem Feature die Details der Verbindung oder Abfahrt mit Google-Maps direkt anzuzeigen. Dabei konnte dies mit den Chromium-Browser Performance schonend anzeigen.

Auch eine Route zum Zielort wird direkt vorgeschlagen. Dies konnte man einfach mit den Stationen Namen dies im Google-Map Link definieren.

Dabei ist leider das Risiko, dass die Namen nicht gefunden werden oder nicht existieren. Weil dies mit den Koordinaten kompliziert und wieder eine API-Abfrage benötigte.

Mehr Infos zu Chromium-Browser: <https://de.wikipedia.org/wiki/Chromium_(Browser)>

Mehr Infos über CefSharp:  
<https://github.com/cefsharp/CefSharp/tree/cefsharp/57>

# **Use Cases, Aktivitätsdiagramme und Klassendiagramm**

Hier wurden Use Cases und Aktivitätsdiagramme für die Anforderungen A001 bis A003 umgesetzt.

## Use Case Diagramm

Abbildung 3: Use Case Diagramm

### Use Case Beschreibung

|  |  |
| --- | --- |
| Use case | Beschreibung |
| Startstation eingeben | Eingabe/Suche der Startstation. |
| Endstation eingeben | Eingabe/Suche der Endstation. |
| Passende Station filtern | Suche der Startstation/Endstation wird passend bei jeder Text-Eingabe gefiltert. Also passende Stationen werden vorgeschlagen. |
| Verbindungen anzeigen | Anzeigen der Verbindungen durch die Eingabe der Start- und Endstation. |
| Stationen per API abrufen | Bei der Eingabe wird eine Abfrage zu API gesendet und gefiltert, die alle Stationen als Vorschlag zu einer Listbox integriert. |
| Abfahrttafel von Station anzeigen | Alle Abfahrten zur aktuellen Zeit der ausgewählten Station wird angezeigt. |
| Öv-Benutzer | User der das Programm nutzt. Als Voraussetzung muss das Programm gestartet werden. |
| Ergebnis | Anzeigen aller verfügbaren Verbindungen oder die Erstellung der Abfahrttafel/Abfahrtmonitor. |

## Aktivitätsdiagramm

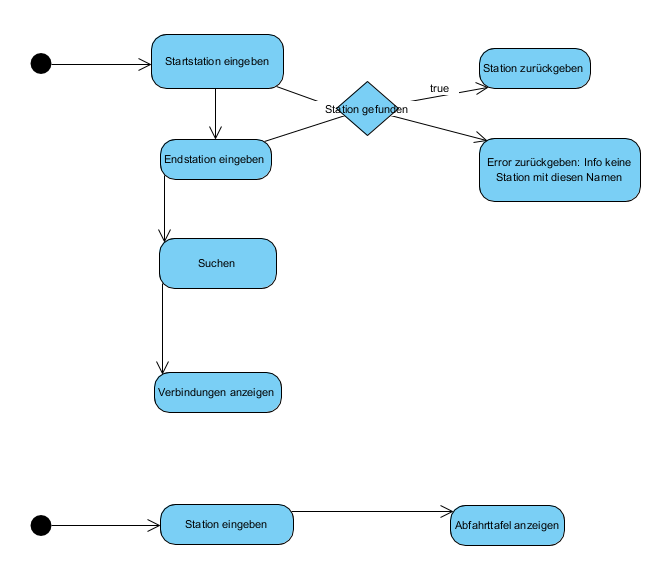


Abbildung 4: Aktivitätsdiagramm

## Klassendiagramm

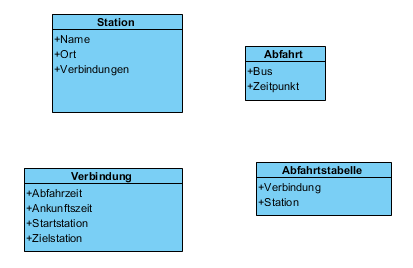
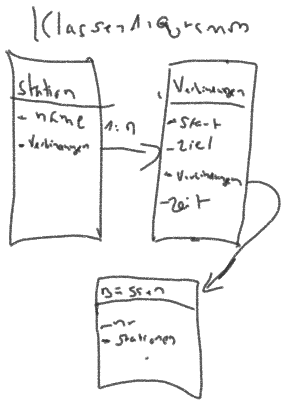


Abbildung 5: Klassendiagram Final

Abbildung 6: Klassendiagramm Skizze

# **Code Guidelines**

Eigentlich hatte ich keine richtige Code Guidelines. Meistens für die Variable-Namen. Dabei war es mir wichtig das ich das Prinzip von Don’t repeat yourself (DRY) einhalte.

## Naming Conventions

* Alles auf Englisch
* Passende Variable- und Methodennamen
* Variablen/Klassen/Methoden: Camel-case z.B. myClass
  + Ausnahme konstante Variablen: const MY\_VAR
* Bei Klassen immer Nomen verwenden
* Bei Methoden immer ein Verbprefix z.B. getVariables()

Mehr Infos zu DRY-Prinzip:

<https://de.wikipedia.org/wiki/Don%E2%80%99t_repeat_yourself>

# **Mockup**

Hier das Mockup für die Anforderungen A001 bis A003 umgesetzt.

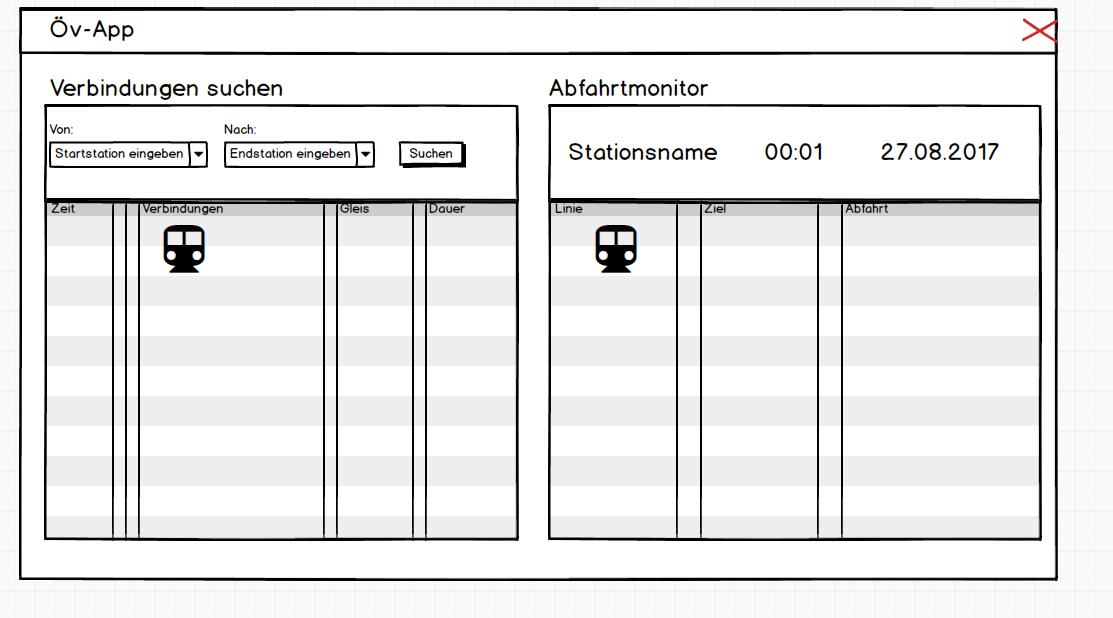


Abbildung 7: Mockup Version 1

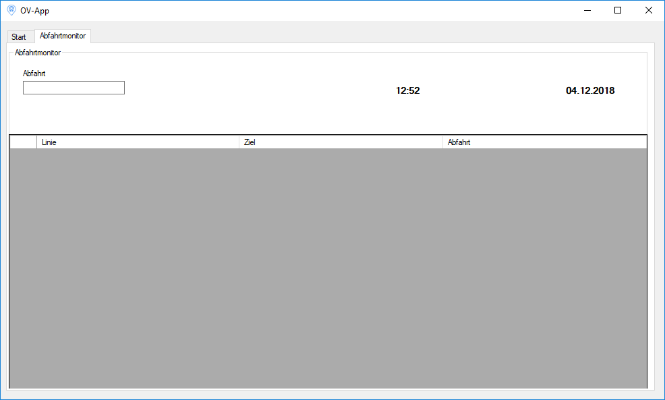
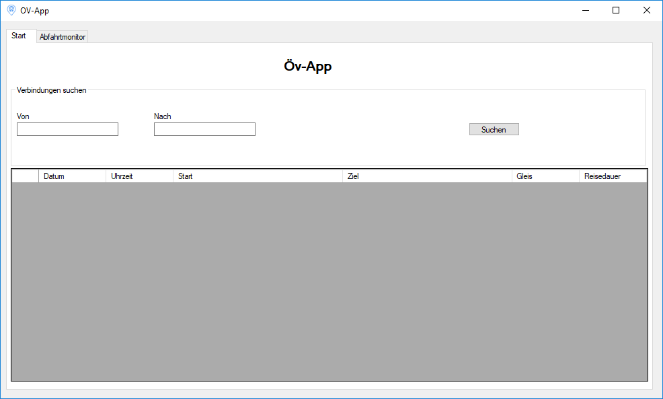
Dabei hat sich das Design während des Programmierens komplett verändert. Zum Beispiel wurde hier die Eingabe der Zeit und Datum gar nicht beachtet. Die wichtigste Veränderung war das Splitten auf Tabs, dass alles schöner und der Performance sehr half.

Abbildung 8: Öv-App Start

Abbildung 9: Abfahrt-Linie Details

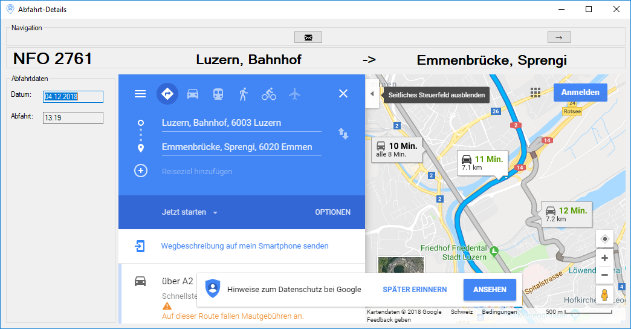


Abbildung 10: Öv-App Abfahrt-Monitor

# **Testing**

Hier wurde Testfälle für die Anforderungen A001 bis A003 umgesetzt.

|  |  |
| --- | --- |
| **Testfall** | **Ablauf** |
| Eingabe Startstation/Endstation und Rückgabe der Liste von den vorgeschlagenen Stationen | 1. Programm starten 2. Zu Abfahrtmonitor oder Start navigieren 3. Stationname in Textfeld eingeben 4. Werden Vorschläge in Liste angezeigt (Ja / Nein) 5. Falls Ja, Test beenden 6. Falls Nein, Issue im Githhub erstellen |
| Verbindungen suchen (Button suchen) | 1. Programm starten 2. Zu Start navigieren 3. Start- und Endstation in Textfeld eingeben 4. Start- und Endstation auswählen 5. Datum auswählen/eingeben 6. Zeit auswählen/eingeben 7. Auf Button «Suchen» klicken 8. Werden Verbindungen angezeigt und wird kein Error ausgelöst? 9. Falls Ja, Test beenden 10. Falls Nein, Issue im Githhub erstellen |
| Abfahrttafel/monitor von Station anzeigen | 1. Programm starten 2. Zu Abfahrtmonitor navigieren 3. Stationsname in Textfeld eingeben 4. Station auswählen 5. Werden Abfahrten/Verbindungen angezeigt und wird kein Error ausgelöst? 6. Falls Ja, Test beenden 7. Falls Nein, Issue im Githhub erstellen |

# **Anleitung**

## Applikation installieren

1. Unter <https://github.com/ac-webdesign/modul-318-student/blob/master/setup.exe> das Setup herunterladen und dies schlussendlich ausführen.
2. Falls der Windows-Defender die Installation abbricht, auf «Weitere Informationen» klicken und schlussendlich Installation mit «Trotzdem ausführen» bestätigen:



Abbildung 11: Windows-Defender Error

1. Beim Installationsmanager immer auf «Weiter klicken» bis das Programm installiert wird:

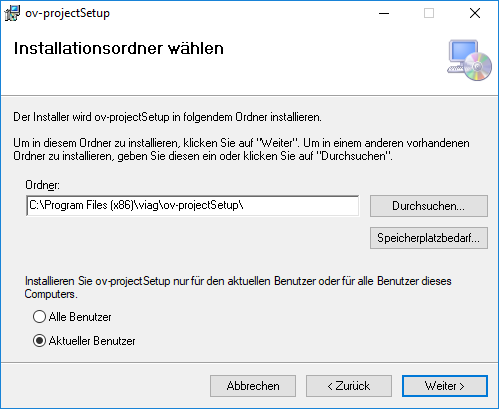


Abbildung 12: Installations-Tool

1. Programm ist als Shortcut im Desktop vorhanden und kann gestartet werden

## Applikation über Windows deinstallieren

1. Suchen Sie über die Windows-Suche «Programme hinzufügen und entfernen» und starten Sie diese.
2. Geben Sie im Filter den Programmname ein.
3. Wählen Sie das Programm aus und klicken Sie auf deinstallieren:

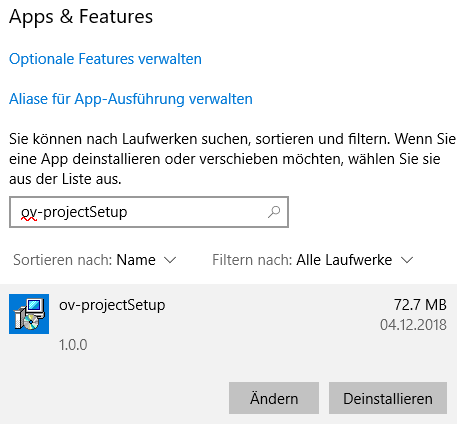


Abbildung 13: Programm deinstallieren

1. Dann müssen Sie mit den Button Ja den Vorgang bestätigen (falls Admin-Recht).
2. Warten Sie bis Vorgang beendet ist.
3. Programm wurde erfolgreich deinstalliert.

## Applikation über Setup deinstallieren

1. Das Setup unter <https://github.com/ac-webdesign/modul-318-student/blob/master/setup.exe> herunterladen und nachher starten oder falls Sie dies schon im Explorer vorhanden haben, von dort das Setup starten.
2. «Programmname entfernen» auswählen und auf «Fertig stellen» klicken:

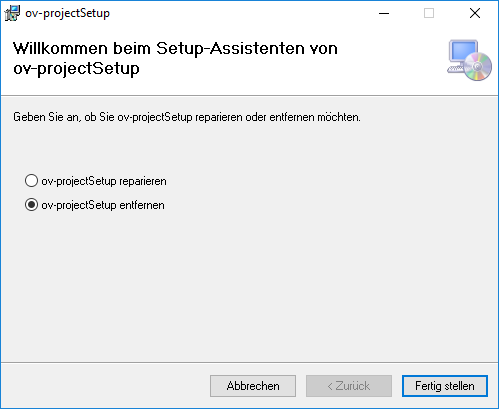


Abbildung 14: Programm über Setup deinstallieren

1. Dann müssen Sie mit den Button Ja den Vorgang bestätigen (falls Admin-Recht).
2. Warten Sie bis Vorgang beendet ist.
3. Programm wurde erfolgreich deinstalliert.

# **Abbildungsverzeichnis**

[Abbildung 1: Integrierung Chromium Browser 4](file:///C:\Users\AC-BYOD\Desktop\git\modul-318-student\docs\Dokumentation%20Projekt.docx#_Toc531700635)

[Abbildung 2: Chromium-Browser 5](file:///C:\Users\AC-BYOD\Desktop\git\modul-318-student\docs\Dokumentation%20Projekt.docx#_Toc531700636)

[Abbildung 3: Use Case Diagramm 6](file:///C:\Users\AC-BYOD\Desktop\git\modul-318-student\docs\Dokumentation%20Projekt.docx#_Toc531700637)

[Abbildung 4: Aktivitätsdiagramm 7](#_Toc531700638)

[Abbildung 5: Klassendiagram Final 7](file:///C:\Users\AC-BYOD\Desktop\git\modul-318-student\docs\Dokumentation%20Projekt.docx#_Toc531700639)

[Abbildung 6: Klassendiagramm Skizze 7](file:///C:\Users\AC-BYOD\Desktop\git\modul-318-student\docs\Dokumentation%20Projekt.docx#_Toc531700640)

[Abbildung 7: Mockup Version 1 9](#_Toc531700641)

[Abbildung 8: Öv-App Start 9](file:///C:\Users\AC-BYOD\Desktop\git\modul-318-student\docs\Dokumentation%20Projekt.docx#_Toc531700642)

[Abbildung 9: Abfahrt-Linie Details 9](#_Toc531700643)

[Abbildung 10: Öv-App Abfahrt-Monitor 9](file:///C:\Users\AC-BYOD\Desktop\git\modul-318-student\docs\Dokumentation%20Projekt.docx#_Toc531700644)

[Abbildung 11: Windows-Defender Error 11](#_Toc531700645)

[Abbildung 12: Installations-Tool 11](#_Toc531700646)

[Abbildung 13: Programm deinstallieren 12](#_Toc531700647)

[Abbildung 14: Programm über Setup deinstallieren 13](#_Toc531700648)

# **Anhang**

## Applikations-Anforderungen

|  |  |  |
| --- | --- | --- |
| 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 |