

# IPA-Bericht

## Code-Generator



|        |                                    |
|--------|------------------------------------|
| Autor  | Luan Caduff                        |
| Klasse | ISO-20                             |
| Datum  | 03. Mai 2024                       |
| Firma  | Evernex IT Services Switzerland AG |

## Table of Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Teil 1</b>                                     | <b>4</b>  |
| 1.1      | Dokumenteninformationen                           | 4         |
| 1.1.1    | Historie  | 4         |
| 1.1.2    | Eigenschaften                                     | 4         |
| 1.2      | Aufgabenstellung                                  | 5         |
| 1.3      | Projektorganisation                               | 7         |
| 1.4      | Projektmethode                                    | 7         |
| 1.5      | Deklaration der Vorkenntnisse                     | 8         |
| 1.6      | Deklaration der Vorarbeiten                       | 8         |
| 1.7      | Deklaration der benutzten Firmenstandards         | 8         |
| 1.8      | Organisation der Arbeitsergebnisse                | 9         |
| 1.9      | Zeitplan  | 10        |
| 1.10     | Arbeitsjournal                                    | 11        |
| 1.10.1   | Tag 1, 18. April 2024                             | 11        |
|          | Tag 2, 19. April 2024                             | 12        |
| 1.10.2   | Tag 3, 23. April 2024                             | 13        |
| 1.10.3   | Tag 4, 24. April 2024                             | 14        |
| 1.10.4   | Tag 5, 25. April 2024                             | 15        |
| 1.10.5   | Tag 6, 26. April 2024                             | 16        |
| 1.10.6   | Tag 7, 30. April 2024                             | 17        |
| 1.10.7   | Tag 8, 01. Mai 2024                               | 18        |
| 1.10.8   | Tag 9, 02. Mai 2024                               | 18        |
| 1.10.9   | Tag 10, 03. Mai 2024                              | 18        |
| 1.10.11  | Zusammenfassung Zeitplan                          | 19        |
| <b>2</b> | <b>Teil 2</b>                                     | <b>20</b> |
| 2.1      | Management Summary (Kurzfassung des IPA-Berichts) | 20        |
| 2.1.1    | Ausgangslage                                      | 20        |
| 2.1.2    | Vorgehen  | 20        |
| 2.1.3    | Ergebnis  | 20        |
| 2.2      | Informieren                                       | 21        |
| 2.2.1    | Aufgabestellung                                   | 21        |
| 2.2.2    | Technologien                                      | 21        |
| 2.2.3    | Use-Cases   | 22        |
| 2.3      | Planen  | 24        |
| 2.3.1    | GUI MockUps                                       | 24        |
| 2.4      | Entscheiden                                       | 27        |

|            |                                |           |
|------------|--------------------------------|-----------|
| <b>2.5</b> | <b>Realisieren.....</b>        | <b>29</b> |
| 2.5.1      | Tabellen-Auswahl Frontend..... | 29        |
| 2.5.2      | Config-Formular.....           | 30        |
| 2.5.3      | Grundaufbau Backend.....       | 31        |
| 2.5.4      | Datenbank Verbindungen.....    | 33        |
| 2.5.5      | Tabellen-Auswahl Backend.....  | 35        |
| 2.5.6      | Snippet-Auswahl Frontend.....  | 35        |
| <b>2.6</b> | <b>Kontrollieren.....</b>      | <b>37</b> |
| 2.6.1      | Testfälle .....                | 37        |
| 2.6.2      | Testprotokolle.....            | 44        |
| <b>2.7</b> | <b>Auswerten .....</b>         | <b>45</b> |
| <b>2.8</b> | <b>Verzeichnisse.....</b>      | <b>46</b> |
| 2.8.1      | Abbildungsverzeichnis.....     | 46        |
| 2.8.2      | Tabellenverzeichnis.....       | 46        |
| 2.8.3      | Links.....                     | 47        |
| 2.8.4      | Glossar / Abkürzungen .....    | 47        |
| <b>2.9</b> | <b>Anhang.....</b>             | <b>48</b> |

# 1 Teil 1

## 1.1 Dokumenteninformationen

### 1.1.1 Historie

| Version  | Gültig ab  | Dokumentenhistorie / Änderungshinweis | Q           |
|----------|------------|---------------------------------------|-------------|
| 0.1.0.1  | 17.04.2024 | Erstellung der Grundstruktur          | Luan Caduff |
| 0.2.0.2  | 18.04.2024 | Zeitplan, Informieren und Planen      | Luan Caduff |
| 0.3.0.3  | 19.04.2024 | Informieren, Entscheiden              | Luan Caduff |
| 0.4.0.4  | 23.04.2024 | Realisieren                           | Luan Caduff |
| 0.5.0.5  | 24.04.2024 | Realisieren                           | Luan Caduff |
| 0.6.0.6  | 25.04.2024 | Realisieren                           | Luan Caduff |
| 0.7.0.7  | 26.04.2024 | Realisieren                           | Luan Caduff |
| 0.8.0.8  | 30.04.2024 | Realisieren                           | Luan Caduff |
| 0.9.0.9  | 01.05.2024 | Realisieren, Kontrollieren            | Luan Caduff |
|          |            |                                       |             |
| 1.0.0.11 | 03.05.2024 | Version bei der Abgabe                | Luan Caduff |

Tabelle 1 - Dokumentenhistorie

### Versionierung

A.B.C.D

A = Eine Veröffentlichung / bereit zum Druck / Abgabe

B = Inhaltliche Änderungen am Dokument

C = Korrekturen (keine inhaltlichen Änderungen)

D = Laufnummer (wird bei jeder Änderung erhöht)

### 1.1.2 Eigenschaften

| Bezeichnung    | Detailinformationen          |
|----------------|------------------------------|
| Status         | In Arbeit                    |
| Autor          | Luan Caduff                  |
| Ausbildung zum | Eidg. Dipl. Informatiker EFZ |
| Fachrichtung   | Applikationsentwickler       |
| Version        | 1.0.0.XX                     |
| Versionsdatum  | 03.05.2024                   |
| Seiten         | 48                           |

Tabelle 2 - Dokumenteneigenschaften

## 1.2 Aufgabenstellung

Die Firma S + O AG ist Partnerfirma der Evernex IT Services Switzerland AG und zuständig für den Betrieb und die Weiterentwicklung deren Web-Applikationen. Die bestehende Applikation SWO (Simple Web Office) soll abgelöst werden. In einem ersten Schritt ist geplant, auf der bestehenden MariaDB Datenbank-Struktur neue Web-Views (CRUDs) für die Administration zu erstellen. Um diese Arbeit zu erleichtern und zu beschleunigen soll ein Code-Generator erstellt werden, welcher anhand der Struktur einer Datenbank-Tabelle Code-Snippets erstellt.

Der Code-Generator soll mit PHP sowie HTML, CSS und JavaScript umgesetzt werden. Es ist eine neue, stand-alone Applikation, welche entsprechend unabhängig läuft, ohne Einbettung in ein bestehendes Umfeld.

Da diese Applikation lediglich lokal von uns verwendet werden soll, benötigt es kein Authentifizierungssystem.

Der Benutzer dieses Code-Generators soll in einem Web-GUI den Datenbank-Server aus einer Selectbox auswählen können. Für die Auswahl des Datenbank-Servers sollen die zum Verbindungsaufbau nötigen Informationen aus einer Konfigurations-Datei gelesen werden (CSV, eine Zeile pro Datenbank-Server mit Strichpunkt getrennte Informationen wie Host, Login, Passwort).

Nach Auswahl des Datenbank-Servers werden dessen Datenbanken in einer weiteren Selectbox zur Auswahl angezeigt. Nachdem eine Datenbank gewählt wurde, werden dessen Tabellen ebenso zur Auswahl angezeigt. Nachdem eine Tabelle ausgewählt wurde, kann mit dem jeweiligen Button ein Code-Snippet erstellt werden.

Das generierte Code-Snippet soll im Web-GUI angezeigt werden und mittels eines Copy-Buttons in die Zwischenablage kopiert werden können.

Das Ziel dieser Web-Applikation ist es also ein sauber formatiertes (übliche Einrückungen, Zeilen-Abstände zur besseren Lesbarkeit) Code-Snippet in der Zwischenablage zur weiteren Verarbeitung bereit zu stellen.

Nachfolgende Code-Snippets sollen generiert werden können:

| Snippet Art         | Zweck   |
|---------------------|---|
| PHP-Model-Klasse    | Abbild der Tabelle (Attribute), Konstruktor, Getter/Setter, JSON-Serialize, Objekt-Erstellung aus Daten |
| PHP-Gateway-Klasse  | Hinzufügen bzw. Anpassen eines neuen Datensatzes anhand des Model-Objektes                              |
| ExtJS-Model         | Laden bzw. mappen via JSON übertragener Datensätze  |
| ExtJS-Grid-List     | Liste der geladenen Datensätze darstellen   |
| ExtJS-Create-Dialog | Eingabe-Formular für einen neuen Datensatz  |
| ExtJS-Edit-Dialog   | Eingabe-Formular für die Anpassung eines bestehenden Datensatzes  |
| ExtJS-Info-Dialog   | Darstellen aller Informationen (nicht editierbare Datenfelder) eines bestehenden Datensatzes            |

*Tabelle 3 - Code-Snippet Arten*

Als Vorlage für die Code-Snippets dient das vorhandene Test-CRUD der Tabelle «erp\_article\_service», an dem ich in den letzten Monaten gearbeitet habe. Diese Vorlagen werden aber hier nicht mitgeliefert bzw. hochgeladen, da es sich dabei um die gesamte Test-Applikation handelt und daraus allgemeine (bzw. für die gewählte Tabelle), sinnvolle Code-Snippets (nach obiger Auflistung) erstellt werden sollen. Die Auswahl für «sinnvoll» ist Bestandteil der IPA und soll auch entsprechend begründet werden.

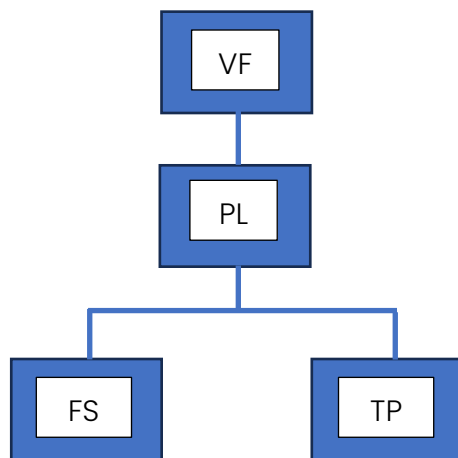
Die Architektur soll MVC nach Firmenusanz abbilden: Klassen mit entsprechenden Funktionalitäten in entsprechenden Verzeichnissen. Dazu existieren keine dokumentierten Firmenstandards.

Die technische Dokumentation zum Aufbau der Applikation soll mittels Use Case und eines Klassen-Diagramms erstellt werden.

Das Code-Styling soll wie folgt sein: Die Namensgebung ist einfach gut gewählt. Die Struktur des Codes ist ebenfalls einfach übersichtlich gemacht. Es ist eine gewisse Einheit zu sehen in der Art und Weise, wie der Code strukturiert ist (d.h. es ist überall etwa gleich gemacht).

Die Applikation soll manuell, anhand von sinnvollen Testfällen getestet werden. Die Definition der Testfälle ist Bestandteil der IPA und die einzelnen Tests müssen dann auch entsprechend protokolliert werden.

## 1.3 Projektorganisation



VF – Roman Born (Verantwortliche Fachkraft)

PL – Luan Caduff (Projektleiter)

FS – Luan Caduff (Fachspezialist)

TP – Luan Caduff (Testperson)

## 1.4 Projektmethode

Dieses Projekt wird mit IPERKA durchgeführt. Dies ist eine simple Projektmanagement-Methode zur strukturierten Planung und Umsetzung eines Projektes.

IPERKA ist ein Akronym und bedeutet folgendes:

I – Informieren – Relevante Informationen sammeln

P – Planen – Damit einen Plan erstellen

E – Entscheiden – Sich für eine spezifische Vorgehensweise entscheiden

R – Realisieren – Das Projekt durchführen

K – Kontrollieren – Endprodukt testen und überprüfen

A – Auswerten – Über den Arbeitsprozess und das Resultat reflektieren

Weitere Informationen zur IPERKA-Methode finden Sie unter [https://www.ict-berufsbildung-bern.ch/resources/Iperka\\_OdA\\_200617.pdf](https://www.ict-berufsbildung-bern.ch/resources/Iperka_OdA_200617.pdf)

## 1.5 Deklaration der Vorkenntnisse

Alle geplanten Tätigkeiten/Produkte/Techniken sind bekannt und wurden während der gesamten Praktikumszeit eingesetzt.

| Technologie         | Erfahrung   |
|---------------------|---|
| PHP                 | Sehr gute Kenntnisse<br>Seit 2.5 Jahren aktiv genutzt.      |
| HTML                | Sehr gute Kenntnisse<br>Seit 3.5 Jahren aktiv genutzt.      |
| JavaScript + jQuery | Ziemlich gute Kenntnisse<br>Seit 2.5 Jahren öfters genutzt. |
| CSS + Bootstrap     | Ziemlich gute Kenntnisse<br>Seit 3.5 Jahren öfters genutzt. |
| MariaDB             | Sehr gute Kenntnisse<br>Seit 3.5 Jahren aktiv genutzt.      |

## 1.6 Deklaration der Vorarbeiten

In den letzten Monaten habe ich ein Test-CRUD der Tabelle «erp\_article\_service» aufgebaut. Dieses CRUD dient als Grundlage für die Code-Snippets, die der Generator erstellen soll.

In direktem Zusammenhang mit dieser Arbeit habe ich ein GIT-Repository auf GitHub eingerichtet.

Des Weiteren habe ich die Dokumentenvorlage inklusive Verzeichnisstruktur am Vortag zum IPA-Start erstellt.

## 1.7 Deklaration der benutzten Firmenstandards

Es existieren keine dokumentierten Firmenstandards.

Der Code wird nach der «PHP Standard Recommendation 1» aufgebaut, worin Grundlegende Coding Spezifikationen festgehalten werden.

Die Spezifikationen sind unter folgendem Link dokumentiert:

<https://www.php-fig.org/psr/psr-1/>

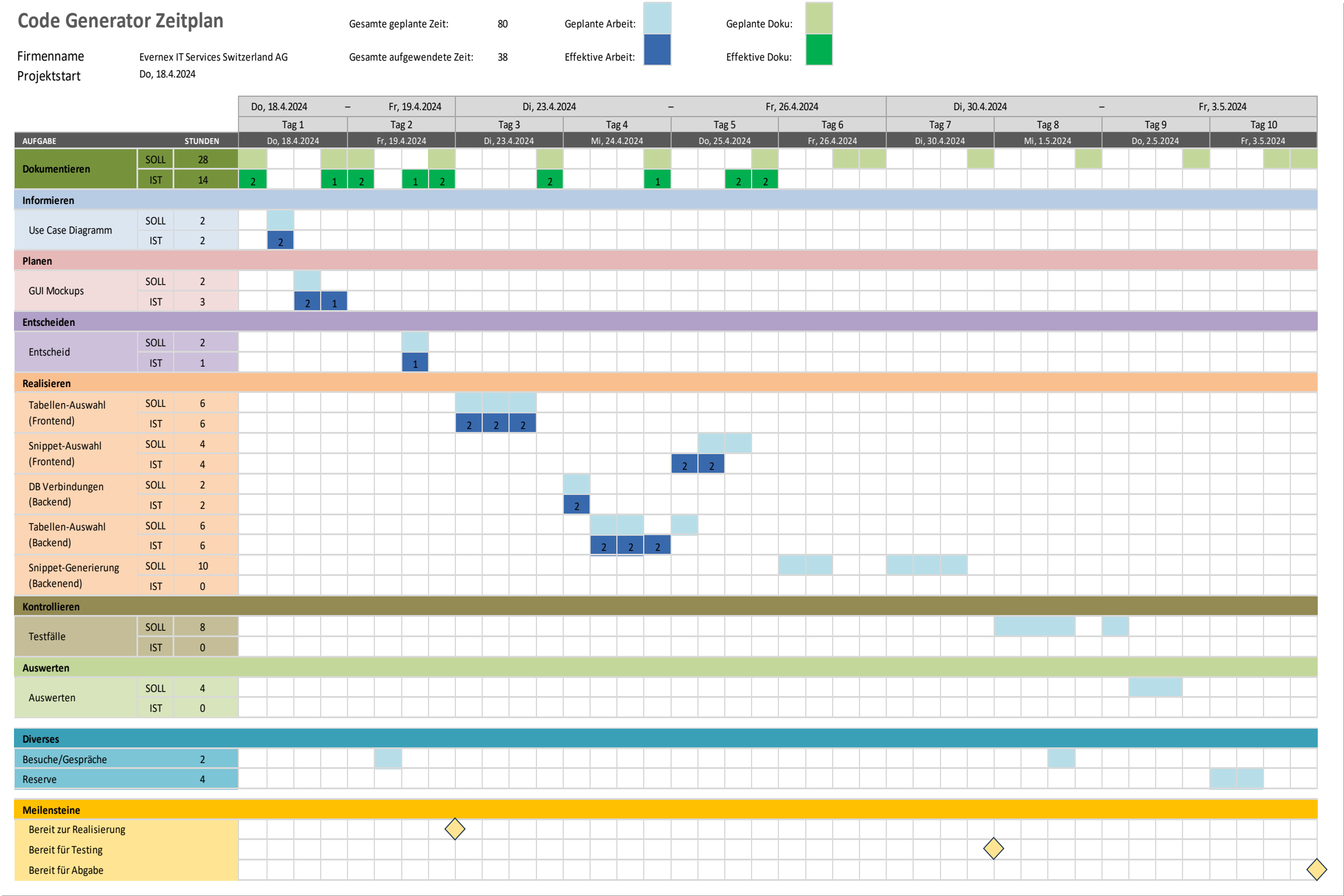


## 1.8 Organisation der Arbeitsergebnisse

Der erarbeitete Code wird in das lokale GIT-Repository committed.  
Das IPA-Dokument wird ebenfalls – jeweils vor Arbeitsende – im lokalen GIT-Repository in das dafür vorgesehene Verzeichnis (000\_Dokumentation) sowohl als Word-Dokument als auch als PDF kopiert und committed.

Das Repository wird täglich (ebenfalls vor Arbeitsende) auf GitHub gepusht:  
<https://github.com/xivia/ipa-luan---code-generator>

1.9 Zeitplan



## 1.10 Arbeitsjournal

### 1.10.1 Tag 1, 18. April 2024

|                     |  |
|---------------------|--|
| Arbeiten            | <p>Dokumentation</p> <ul style="list-style-type: none"> <li>▪ Zeitplan erstellt</li> <li>▪ Im IPA Dokument Teil 1 abgefüllt</li> </ul> <p>Informieren</p> <ul style="list-style-type: none"> <li>▪ Use Case Diagramm erstellt</li> </ul> <p>Planen</p> <ul style="list-style-type: none"> <li>▪ MockUps erstellt</li> </ul>  |
| Probleme            | Ich habe die MockUps erstellt, jedoch wurden diese nicht richtig gespeichert und sind verloren gegangen. Aus diesem Grund habe ich sie dann erneut erstellen müssen.   |
| Hilfestellungen     | Zeitplan kurz mit VF besprochen.   |
| Überzeiten          | Aufgrund meines Problems habe ich für die MockUps eine Stunde mehr als geplant aufgewendet.  |
| Ungeplante Arbeiten | Keine.   |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.  |
| Misserfolge         | Keine.   |
| Reflektion          | <p>Ich habe die MockUps wie bereits erwähnt mit demselben Web-Tool erstellt, wie zuvor die Use Cases. Mit diesem Tool konnte ich die MockUps effizient zusammensetzen. Leider habe ich aus Versehen meinen Browser geschlossen, bevor das Tool die Änderungen speichern konnte. Zukünftig werde ich besser darauf Acht geben, Dateien öfters abzuspeichern.</p> <p>Da ich nun etwas in Verzug bin, da die Use Case Beschreibungen noch nicht dokumentiert wurden, muss ich morgen im geplanten Dokumentieren-Block als erstes diese noch verfassen, um so wieder auf Kurs zu kommen.</p> |

Tag 2, 19. April 2024

|                     |   |
|---------------------|---|
| Arbeiten            | <p>Erstes Gespräch mit HEX</p> <p>Dokumentation</p> <ul style="list-style-type: none"> <li>▪ Generell Dokumentation verschönert</li> <li>▪ Use Case Beschreibungen verfasst</li> </ul> <p>Entscheiden</p> <ul style="list-style-type: none"> <li>▪ GUI-Varianten Entscheid gefällt</li> </ul>   |
| Probleme            | Keine.  |
| Hilfestellungen     | Vorbereitung mit VF auf HEX-Gespräch.   |
| Überzeiten          | Keine.  |
| Ungeplante Arbeiten | Keine.  |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.   |
| Misserfolge         | Keine.  |
| Reflektion          | <p>Die Use-Case Beschreibungen waren schnell gemacht, da das Diagramm bereits existiert und ich die Use-Cases bereits gut kenne.</p> <p>Die Entscheidung zwischen den beiden GUI-MockUps war schwerer als gedacht, da die Vergleichskriterien schwierig zu finden sind bzw. nicht immer objektiv sind. Vorerst wurde der Vergleich textuell gemacht, aber dies sollte zukünftig noch in eine Entscheidungsmatrix/Tabelle integriert werden.</p> |

## 1.10.2 Tag 3, 23. April 2024

|                     |   |
|---------------------|---|
| Arbeiten            | Dokumentation <ul style="list-style-type: none"><li>▪ Realisierung dokumentiert</li></ul> Realisieren <ul style="list-style-type: none"><li>▪ Frontend Tabellen-Auswahl</li></ul>   |
| Probleme            | Keine.  |
| Hilfestellungen     | Keine.  |
| Überzeiten          | Keine.  |
| Ungeplante Arbeiten | Keine.  |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.   |
| Misserfolge         | Keine.  |
| Reflektion          | <p>Ich konnte die geplanten 6 Stunden für die Tabellen-Auswahl effizient ausnützen.</p> <p>Für die verschiedenen Selectboxen musste ich Dummy-Daten hartkodieren, da noch kein Backend existiert.</p> <p>Zudem habe ich die Formularmaske zur Einrichtung neuer Datenbankverbindungen (Configs) erstellt.</p> <p>Ich finde die Fade-In und Fade-Out Effekte visuell ansprechend und habe sie implementiert, da ich noch genügend Zeit dafür hatte.</p> <p>Das Resultat entspricht dem MockUp soweit ziemlich genau.</p> |

## 1.10.3 Tag 4, 24. April 2024

|                     |  |
|---------------------|--|
| Arbeiten            | Dokumentation <ul style="list-style-type: none"><li>▪ Realisierung dokumentiert</li></ul> Realisieren <ul style="list-style-type: none"><li>▪ Backend DB-Verbindungen und Tabellen-Auswahl</li></ul>   |
| Probleme            | Keine.   |
| Hilfestellungen     | Keine.   |
| Überzeiten          | Eine Stunde Überzeit zur Fertigstellung der Tabellen-Auswahl.  |
| Ungeplante Arbeiten | Keine.   |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.  |
| Misserfolge         | Keine.   |
| Reflektion          | Zuerst habe ich die Grundstruktur des Backends aufgebaut, um eine Datenbankverbindung mithilfe einer Config herstellen zu können. Ich konnte einen flüssigen Übergang zur Tabellen-Auswahl machen, und habe diese heute vorfristig abgeschlossen, in dem ich sie anstelle der geplanten Dokumentation gemacht habe, und dafür eine Stunde Überzeit zum Dokumentieren investiert habe. Somit bin ich nun vorne im Zeitplan, und kann morgen bereits mit der Snippet-Auswahl beginnen. |

## 1.10.4 Tag 5, 25. April 2024

|                     |   |
|---------------------|---|
| Arbeiten            | Dokumentation <ul style="list-style-type: none"><li>▪ Realisierung dokumentiert</li></ul> Realisieren <ul style="list-style-type: none"><li>▪ Frontend Snippet-Auswahl</li></ul>  |
| Probleme            | Kopieren in die Zwischenablage komplizierter als gedacht.   |
| Hilfestellungen     | Keine.  |
| Überzeiten          | Keine.  |
| Ungeplante Arbeiten | Keine.  |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.   |
| Misserfolge         | Keine.  |
| Reflektion          | Viel dokumentiert – verfeinert.<br>Frontend umgesetzt, zusätzlich visuelle Hilfe (rechter Teil der Applikation grau überdeckt) zur Erläuterung der Eingaben<br>Reihenfolge (zuerst DB-Tabelle wählen).<br>Da ich die Zwischenablage |

## 1.10.5 Tag 6, 26. April 2024

|                     |   |
|---------------------|---|
| Arbeiten            | Dokumentation <ul style="list-style-type: none"><li>▪ Realisierung dokumentiert</li></ul> Realisieren <ul style="list-style-type: none"><li>▪ Backend Snippet-Generierung PHP</li></ul>   |
| Probleme            | Keine.  |
| Hilfestellungen     | Keine.  |
| Überzeiten          | Keine.  |
| Ungeplante Arbeiten | Keine.  |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.   |
| Misserfolge         | Keine.  |
| Reflektion          | Heute habe ich im Backend die PHP Snippets fertiggestellt. Dies wurde noch ein bisschen stressig da zuerst noch alle grundlegende String-Manipulationsfunktionen in einer Basis-Klasse definiert werden mussten, um die Snippets alle mit den gleichen Funktionen zu erstellen. |



## 1.10.6 Tag 7, 30. April 2024

|                     |   |
|---------------------|---|
| Arbeiten            | Dokumentation <ul style="list-style-type: none"><li>▪ Realisierung dokumentiert</li></ul> Realisieren <ul style="list-style-type: none"><li>▪ Backend Snippet-Generierung ExtJs</li></ul>             |
| Probleme            | Keine.  |
| Hilfestellungen     | Keine.  |
| Überzeiten          | Keine.  |
| Ungeplante Arbeiten | Keine.  |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.   |
| Misserfolge         | Keine.  |
| Reflektion          | Heute habe ich die ExtJs Snippets, und somit den «Realisieren» Teil der Arbeit, abgeschlossen. Das Dokumentieren ist zeitaufwändiger als gedacht, und ich bin mit meinem Fortschritt nicht zufrieden. |

## 1.10.7 Tag 8, 01. Mai 2024

|                     |  |
|---------------------|--|
| Arbeiten            | Dokumentation <ul style="list-style-type: none"><li>▪ Kontrollieren dokumentiert</li><li>▪ Realisieren Dokumentiert</li></ul> Kontrollieren <ul style="list-style-type: none"><li>▪ Testfälle definiert.</li></ul> |
| Probleme            | Keine.   |
| Hilfestellungen     | Keine.   |
| Überzeiten          | Keine.   |
| Ungeplante Arbeiten | Keine.   |
| Erfolge             | Alle Arbeiten wurden zufriedenstellend abgeschlossen.  |
| Misserfolge         | Keine.   |
| Reflektion          | Heute habe ich alle Testfälle definiert, und weiterhin die fehlenden Teile der Realisierung fertiggestellt.<br>Die verschiedenen Tests für die Applikation zu erfinden war..                                       |

## 1.10.8 Tag 9, 02. Mai 2024

## 1.10.9 Tag 10, 03. Mai 2024

## 1.10.11 Zusammenfassung Zeitplan

Hier zählen wir alle Stunden pro Tag zusammen um einen guten, übersichtlichen Soll-/Ist- Vergleich zu erhalten.

| Datum                 | Soll [h] | Ist [h] |
|-----------------------|----------|---------|
| Tag 1, 18. April 2024 | 8        | 8       |
| Tag 2, 19. April 2024 | 8        | 8       |
| Tag 3, 23. April 2024 | 8        | 8       |
| Tag 4, 24. April 2024 | 8        | 9       |
| Tag 5, 25. April 2024 | 8        | 8       |
| Tag 6, 26. April 2024 | 8        | 8       |
| Tag 7, 30. April 2024 | 8        | 8       |
| Tag 8, 1. Mai 2024    | 8        | 9       |
| Tag 9, 2. Mai 2024    | 8        |         |
| Tag 10, 3. Mai 2024   | 8        |         |
| Total                 | 80       |         |

Tabelle 4 - Zeitplan Zusammenfassung

Begründung → siehe Arbeitsjournal

## 2 Teil 2

### 2.1 Management Summary (Kurzfassung des IPA-Berichts)

#### 2.1.1 Ausgangslage

Text

#### 2.1.2 Vorgehen

Text

#### 2.1.3 Ergebnis

Text

## 2.2 Informieren

### 2.2.1 Aufgabestellung

Siehe Kapitel 1.2 Aufgabenstellung

Nach Analyse dieser Aufgabenstellung habe ich bemerkt, dass nicht definiert ist, wie diese Konfigurations-Datei abgefüllt werden soll. Nach Rücksprache kann sie manuell bearbeitet werden. Ich habe mich dann aber dazu entschlossen ein kleines Formular zum Hinzufügen eines Eintrages umzusetzen, da dies keinen grossen Zusatzaufwand mit sich trägt, jedoch einen Mehrwert bietet.

Weil die Applikation auf Englisch umgesetzt wird, werden auch alle technischen Diagramme und Dateinamen Englisch benannt.

#### **Begrifflichkeiten:**

Die Konfigurations-Datei wird fortan Config-Datei, und die darin enthaltenen Verbindungsinformationen Configs genannt.

### 2.2.2 Technologien

Als Vorgabe sind folgende Technologien zu benützen:

- ➔ PHP – Personal Home Page Hypertext Preprocessor (rekursives Akronym)
  - Verwendet wird die Version 8.2
- ➔ HTML – Hypertext Markup Language
  - Verwendet wird HTML5
- ➔ JavaScript mit der jQuery Library
  - Hier kommen ECMAScript 2023 und jQuery Version 3.7.1 zum Einsatz
- ➔ CSS mit dem Bootstrap Framework – Cascading Style Sheets
  - Hier verwenden wir auch die aktuellsten Versionen der jeweiligen Technologien; CSS 3 & Bootstrap 5.3.2

## 2.2.3 Use-Cases

Mithilfe des Web-Tools «Lucidchart» habe ich das Use-Case-Diagramm erstellt.  
Mehr zum Tool finden Sie unter <https://www.lucidchart.com/pages/>

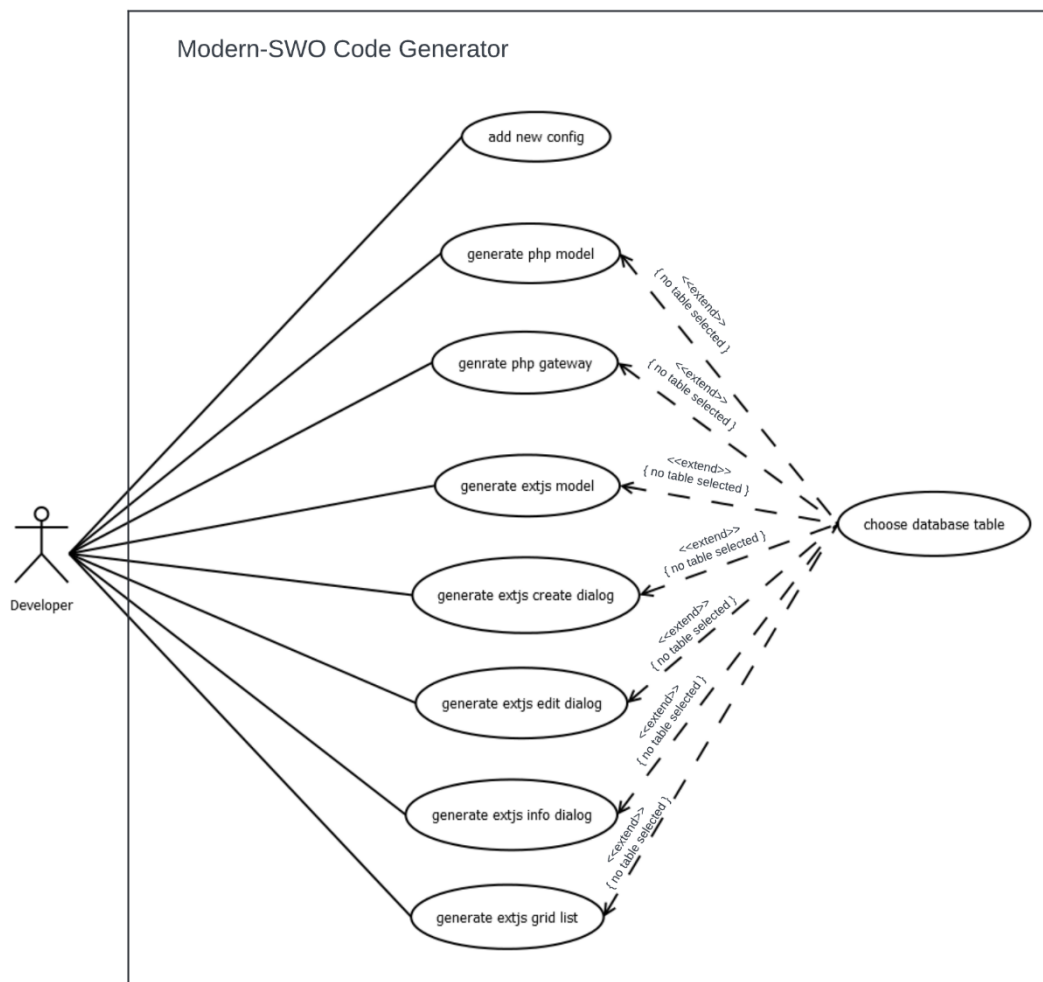


Abbildung 1 - Use-Case-Diagramm

## Beschreibungen:

|               |  |
|---------------|--|
| Use Case      | add new config   |
| Akteur        | Developer  |
| Vorbedingung  | Keine  |
| Beschreibung  | Durch Klick auf einen Button öffnet sich ein Formular, in welchem eine neue Datenbankverbindung (Eintrag in der Config-Datei) hinzugefügt werden kann. Es müssen Host, User, Passwort und der Port angegeben werden. Danach werden diese in der Config-Datei als neue Zeile gespeichert. |
| Alternative   | Kann die Config nicht hinzugefügt werden, wird eine Fehlermeldung ausgegeben.  |
| Nachbedingung | Eine neue Config wurde in der Config-Datei angelegt.   |

Tabelle 5 - Use-Case Beschreibung «add new config»

|               |   |
|---------------|---|
| Use Case      | choose database table   |
| Akteur        | Developer   |
| Vorbedingung  | Keine   |
| Beschreibung  | Erstens kann eine Datenbankverbindung gewählt werden, woraufhin versucht wird, diese Verbindung herzustellen. Bei Erfolg kann als nächstes eine darin enthaltene Datenbank ausgewählt werden. Danach kann eine Tabelle der Datenbank ausgewählt werden. |
| Alternative   | Kann die Datenbankverbindung nicht hergestellt werden, erscheint eine Fehlermeldung.  |
| Nachbedingung | Es können Code-Snippets generiert werden.   |

Tabelle 6 - Use-Case Beschreibung «choose database table»

|               |   |
|---------------|---|
| Use Case      | generate Code-Snippet<br>(php model, php gateway, extjs model, extjs grid, extjs add, extjs edit, extjs info)   |
| Akteur        | Developer   |
| Vorbedingung  | Es muss eine Datenbank-Tabelle ausgewählt sein.   |
| Beschreibung  | Durch Klick auf den entsprechenden Button wird das Code-Snippet generiert.  |
| Alternative   | Wenn in der Tabelle ein unbekannter Datentyp vorkommt, wird eine Fehlermeldung mit dem Namen des Datentyps ausgegeben. Sonst erscheint eine generische Fehlermeldung. |
| Nachbedingung | Das Code-Snippet ist im GUI ersichtlich und kopierbar.  |

Tabelle 7 - Use-Case Beschreibung «generate Code-Snippet»

## 2.3 Planen

### 2.3.1 GUI MockUps

Ich möchte zwei Versionen eines möglichen Frontend-Aufbaus dieser Applikation entwerfen. Dazu verwende ich dasselbe Web-Tool wie bereits für die Use-Cases.

#### Entwurf 1

Config Label  
<selectbox>

DB Label  
<selectbox>

Table Label  
<selectbox>

new

Modern-SWO Code Generator

PHP  
gen gen

ExtJS  
gen gen gen gen gen

Output:

copy

output box

Abbildung 2 - MockUp 1



## Entwurf 2

### Modern-SWO Code Generator

new

Config Label

<selectbox>

DB Label

<selectbox>

Table Label

<selectbox>

---

PHP

gen

gen

ExtJS

gen

gen

gen

gen

gen

Output:

copy

output box

Abbildung 3 – MockUp 2

## Beschreibung der Elemente:

Der Button «new» sollte den Dialog zum Erfassen einer neuen Config öffnen. Entweder als Pop-Up (MockUp 1), oder als versteckte Formfelder, die dann auftauchen (MockUp 2).

Die Selectboxen gewährleisten jeweils die Auswahl der Config, der Datenbanken und der Tabellen.

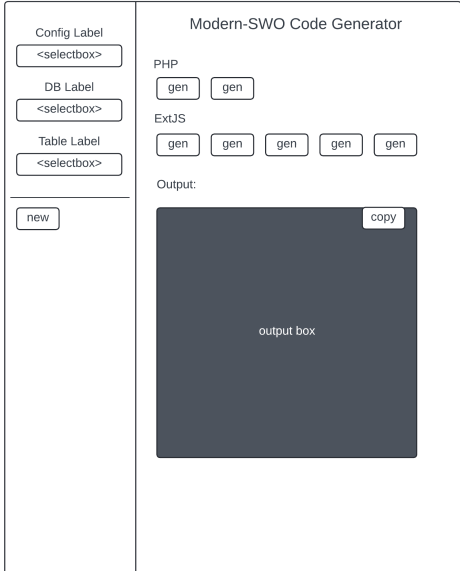
Die Buttons, die mit «gen» gekennzeichnet sind, werden die Code-Snippets generieren und in der Output box darstellen.

Um das Kopieren in die Zwischenablage zu erleichtern habe ich noch einen Copy-Button in die Output box integriert.

## 2.4 Entscheiden

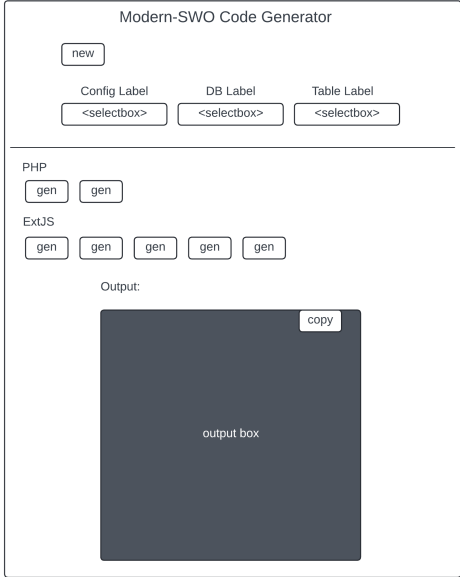
Nun wird sich für eine der beiden GUI-Varianten entschieden.

Variante 1



oder

Variante 2



### Kriterien/Tabelle/Punktevergabe

Variante 1 hat ein horizontales Layout, wo die Selectboxen links ziemlich eng zusammen sind, und unterhalb dieser der «new» Button mit einem Trennstrich isoliert ist. Dies liegt daran, dass der Platz unter dem «new» Button dafür vorgesehen ist, ein Formular anzuzeigen, wenn der Button gedrückt wird.

Bei der Variante 2 wäre solch ein Formular vom Platz her nicht umsetzbar, und man müsste ein modales Pop-Up verwenden.

Bei der Variante 1 gibt es vertikal mehr Platz für das Code-Snippet als bei Variante 2 und grundsätzlich wird das Code-Snippet eher länger als breiter.

Der new-Button impliziert bei Variante 2, dass zuerst eine Config erstellt werden muss, was nicht der Fall ist. Und wenn der new-Button unter den Selectboxen ist, muss man diesen jeweils „überspringen“.

Der «Arbeits-Prozess»: Wähle Config -> DB -> Table -> Snippet -> Copy geht bei Variante 1 ein wenig „locker von der Hand“: von links oben nach rechts versus links-rechts-links

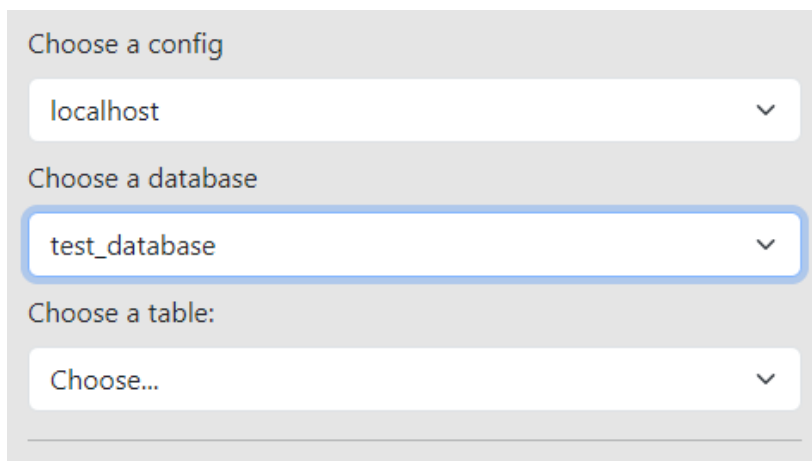
Entscheid: Es wird Variante 1 umgesetzt.

## 2.5 Realisieren

Bei der Realisierung ist geplant, die Applikation schrittweise aufzubauen. Zuerst ein Abschnitt des Frontends und danach der entsprechende Backend Teil dazu.

### 2.5.1 Tabellen-Auswahl Frontend

Um eine Datenbank Tabelle auswählen zu können, muss zuerst eine Datenbank ausgewählt sein. Um jedoch solch eine zu wählen, muss man eine Datenbankverbindung, also eine Config, auswählen.



Choose a config

localhost

Choose a database

test\_database

Choose a table:

Choose...

Abbildung 4 - Selectboxen

Dies habe ich mithilfe des JavaScript onChange Events umgesetzt. Sobald die oberste Selectbox (Config) geändert wird, beginnt die automatische Abfüllung der nächsten Selectbox (Datenbank). Bei der letzten Selectbox (Tabelle) ist keine onChange Event Funktion definiert.

Damit dies für den Benutzer klar ersichtlich und intuitiv ist, erscheint zuerst lediglich die oberste Selectbox. Erst so bald dort eine Option gewählt wurde, erscheint die Datenbank Selectbox. Das gleiche geschieht mit der Tabellen-Selectbox, sie erscheint erst, wenn eine Datenbank ausgewählt wurde.

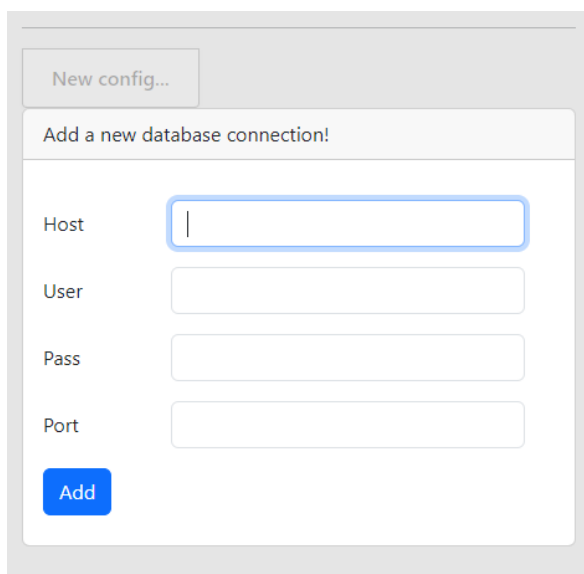
Falls im Backend beim Datenbankverbindungsversuch ein Fehler auftritt, wird eine Meldung unterhalb der Config Selectbox erscheinen.

## 2.5.2 Config-Formular

Zur erleichterten Anlegung einer neuen Config habe ich definiert, ein Formular dafür zu integrieren.

Dieses erscheint, nachdem man auf den «New config» Button drückt, und verschwindet, wenn der Fokus das Formular verlässt und kein Feld abgefüllt wurde oder eine Config hinzugefügt wurde.

Das Formular:



The screenshot shows a modal window titled "Add a new database connection!". In the top-left corner of the modal is a button labeled "New config...". The main area of the modal contains four labeled input fields: "Host", "User", "Pass", and "Port". The "Host" input field is currently active, indicated by a blue border. At the bottom-left of the modal is a blue button labeled "Add".

Abbildung 5 - Config-Formular

## 2.5.3 Grundaufbau Backend

Grundsätzlich ist die Applikation nach MVC gestaltet. Alle Klassen, die nicht in dieses Schema passen, sind in einem vierten Verzeichnis namens «Utils» abgelegt. Die verschiedenen Module kommunizieren folgendermassen:

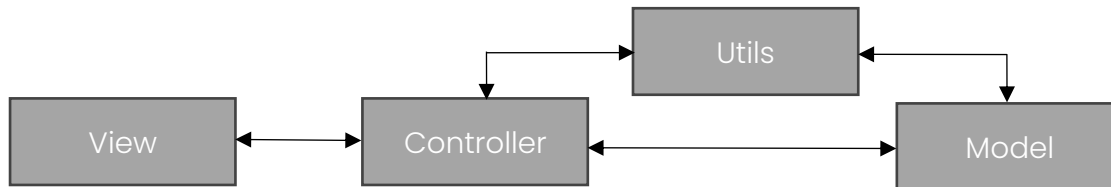


Abbildung 6 – Backend-Struktur

Im Controller existiert eine PHP-Datei namens «router». Diese ist die einzige Backend Datei, welche keine Klasse ist. Sie ist die Kommunikationsschnittstelle zwischen dem Front- und Backend, und wird auch verwendet, um einen Autoloader zu definieren und Error-Reporting einzuschalten.

Jede AJAX-Anfrage geht direkt an diese Datei, worin das entsprechende Controller Objekt anhand des übergebenen «action» Parameters erstellt wird und die gesendeten Daten (falls vorhanden) an eine Methode des erstellten Objekts übergeben werden.

```

1  <?php
2  namespace Controller;
3
4  error_reporting(E_ALL);
5  ini_set('display_errors', 'On');
6
7  spl_autoload_register(function ($class_name) {
8      $class_name = str_replace('\\', DIRECTORY_SEPARATOR, $class_name);
9      require_once('..' . $class_name . '.php');
10 });
11
12
13
14 if(isset($_GET['action']) && $_GET['action'] == 'getConfigs') {
15
16     $controller = new ConfigController();
17     $controller->getConfigs();
18     exit();
19 }
20
21
22 if(isset($_GET['action']) && $_GET['action'] == 'getDatabases') {
23
24     $configId = isset($_GET['configId']) ? $_GET['configId'] : 0;
25
26     $controller = new ConfigController();
27     $controller->getDatabases($configId);
28
29     exit();
30 }
31
32 // . . .
33
34 echo(['msg': 'fallthrough']);
35
36 ?>
  
```

Abbildung 7 – Backend router Ausschnitt

Jede Controller-Methode, welche die Sichtbarkeit «public» hat, sorgt für die Ausgabe (Antwort ans Frontend) der gewünschten Daten.

Mithilfe der Model und/oder Utils Klassen wird eine Antwort ausgearbeitet.

Zur Ausgabe wird die Utils-Klasse «Response» verwendet. Sie definiert eine Einheitliche Antwort im JSON-Format.

```

33 public function respond() {
34     $this->prepareResponse();
35
36     echo json_encode($this->response, JSON_UNESCAPED_UNICODE);
37 }
38
39 private function prepareResponse() {
40
41     http_response_code($this->httpResponseCode);
42
43     $this->response = [
44         'status' => $this->status,
45         'message' => $this->message,
46         'data' => $this->wrapData($this->data)
47     ];
48 }
49
50 private function wrapData(mixed $data): array {
51     // if data is not already an array put it in one
52     if(gettype($data) != 'array' && gettype($data) != 'object') {
53         return [$data];
54     } else {
55         return $data;
56     }
57 }
58

```

Abbildung 8 - Response-Klasse Ausschnitt

So sieht das im Einsatz aus:

```

public function getDatabases($configId) {
    $response = new Response();

    $config = ConfigManager::getInstance()->getConfigById($configId);
    $databases = Database::list($config);

    $response->setStatus(Response::STATUS_OK);
    $response->setMessage('');
    $response->setHttpResponseCode(Response::HTTP_STATUS_OK);
    $response->setData($databases);

    $response->respond();
}

```

Abbildung 9 - Controller-Klasse Ausgabe

Die einzige andere Klasse, welche eine Ausgabe ausführt, ist die Utils-Klasse «ErrorThrower», welche lediglich eine statische Methode zur Fehlerausgabe und zum Ausführungsabbruch definiert.

```

1 <?php
2 namespace Utils;
3
4 class ErrorThrower {
5
6     public static function throw($msg, $data = []) {
7         $response = new Response(Response::STATUS_ERROR, $msg, Response::HTTP_STATUS_BAD_REQUEST);
8         $response->setData($data);
9         $response->respond();
10        exit();
11    }
12 }

```

Abbildung 10 - Utils-Klasse ErrorThrower



## 2.5.4 Datenbank Verbindungen

Ein weiteres Beispiel einer Utils Klasse ist «MysqliDB».

Sie ist ein Singleton und wurde erstellt, um mysqli-Objekte zu erstellen und zu verwalten, welche als Datenbankverbindung verwendet werden.

In PHP ist die mysqli-Klasse ein Datenbanktreiber der als Schnittstelle zu MySQL-Datenbanken verwendet wird.

Die Klasse benötigt zur Instanziierung eines mysqli-Objekts ein Config-Objekt. Ein Config-Objekt ist ein Abbild einer Zeile der Config-Datei, und zählt somit als Model. Zusätzlich sind die zwei weiteren Models für eine Datenbank und eine Tabelle definiert. Diese drei Models bilden unsere Selectboxen ab.

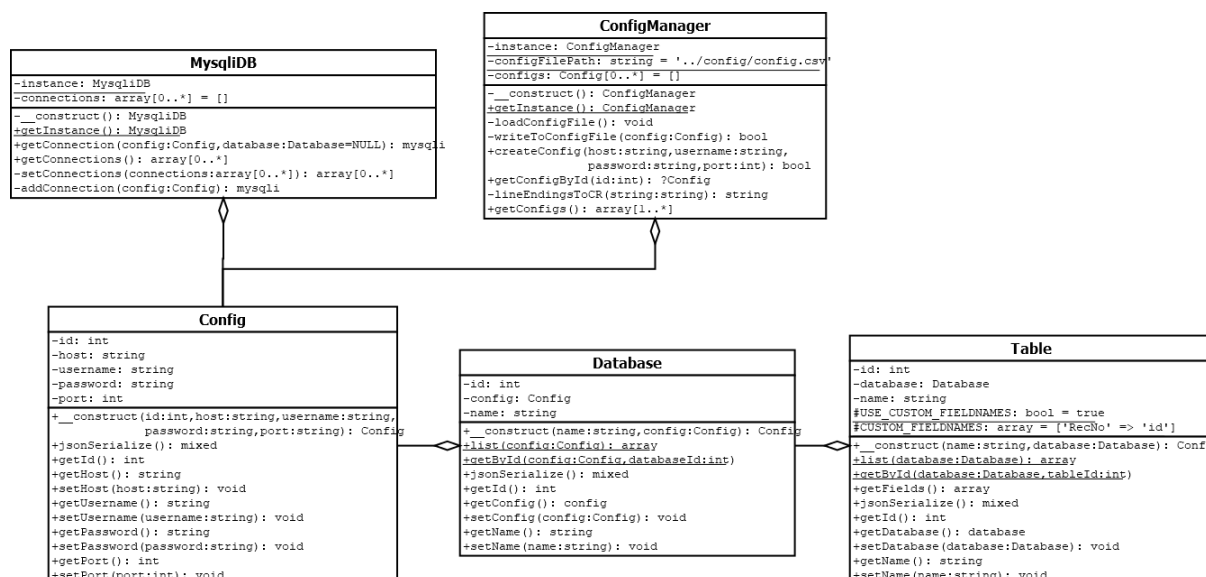


Abbildung 11 - Klassendiagramm DB Verbindungen

Alles, was direkt mit der Config-Datei zu tun hat, wird im ConfigManager, einer weiteren Utils-Singleton-Klasse, geregelt.

Der ConfigManager ist dafür verantwortlich beim Start der Applikation automatisch jede Zeile der Config-Datei auszulesen, Config-Objekte daraus zu erstellen und diese in einem eigenen Array abzuspeichern.

Ebenfalls unternimmt er alle Änderungen an der Config-Datei, z.B. wenn eine neue Config hinzugefügt wird.

Zwei Zeilen in der Config-Datei würden folgendermassen aussehen:

```

1;mySQLServer.com;root;myPssword2024*;3306
2;differentServer.ch;admin;admin; 1433
  
```

Eine Datenbankverbindung wird erst hergestellt, wenn es nötig ist.

Wenn lediglich die möglichen Configs angezeigt werden müssen, wird auch keine Verbindung hergestellt.

Als beispiel wird der Ablauf angezeigt, wenn alle Tabellen einer Datenbank abgefragt werden.

Request mit der configId und der databaseId wird an router-Datei geschickt.

Request wird an Controller übergeben:

```
public function getTables($configId, $databaseId) {
    $response = new Response();

    $config = ConfigManager::getInstance()->getConfigById($configId);
    $database = Database::getById($config, $databaseId);
    $tables = Table::list($database);

    $response->setStatus(Response::$STATUS_OK);
    $response->setMessage('');
    $response->setHttpStatusCode(Response::$HTTP_STATUS_OK);
    $response->setData($tables);

    $response->respond();
}
```

Abbildung 12 - ConfigController getTables Funktion

Wie man sehen, kann wird zuerst über den ConfigManager die Config aus den geladenen Configs die im ConfigManager abgespeichert sind geholt.

Mit diesem Config Objekt können wir dasselbe mit der Datenbank machen.

Nun aber müssen wir

### Hinzufügen einer Config

Die Config-Datei muss bei jeder Installation manuell erstellt werden, da sie aufgrund ihres heiklen Inhalts (Passwörter) in der gitignore-Datei ist, und das Formular „New config“ lediglich Config Zeilen hinzufügen kann, jedoch das benötigte File nicht erstellt.

Wird das Formular zum Hinzufügen einer Config abgeschickt, kommt der ConfigController zum Einsatz.

| ConfigController  |
|---|
| <pre>+getConfigs(): void +getDatabases(configId:int): void +getTables(configId:int,databaseId:int): void +createConfig(host:string,username:string,               password:string,port:int): void</pre> |

Abbildung 14 - ConfigController  
Klassendiagramm

```
public function createConfig($host, $user, $password, $port) {
    $response = new Response();

    $manager = ConfigManager::getInstance();
    if($manager->createConfig($host, $user, $password, $port)) {
        $response->setStatus(Response::$STATUS_OK);
        $response->setMessage('');
        $response->setHttpStatusCode(Response::$HTTP_STATUS_OK);
    } else {
        $response->setStatus(Response::$STATUS_ERROR);
        $response->setMessage('Failed to add new config');
        $response->setHttpStatusCode(Response::$HTTP_STATUS_SERVER_ERROR);
    }

    $response->respond();
}
```

Abbildung 13 - ConfigController createConfig Funktion

Wie man sehen kann, leitet der Controller die Aufgabe an den ConfigManager weiter. Der nimmt diese Aufgabe so entgegen:

```
private function writeToConfigFile(Config $config): bool {
    $configLine = chr(ASCII::LINE_FEED);
    $configLine .= "{ $config->getId()};{$config->getHost()};{$config->getUsername()};{$config->getPassword()};{$config->getPort()}";
    return (bool) file_put_contents(self::$configFilePath, $configLine, FILE_APPEND);
}

public function createConfig($host, $user, $password, $port): bool {
    $id = count($this->configs) + 1;
    $config = new Config($id, $host, $user, $password, $port);

    $this->configs[] = $config;
    return $this->writeToConfigFile($config);
}
```

Abbildung 15 - ConfigManager Dateibearbeitung

Wie man in der writeToConfigFile-Funktion sehen kann, wird die Utils-Klasse ASCII verwendet, um mit der PHP-Funktion chr ein ASCII-Zeichen zu erzeugen. Diese Klasse ist also nur ein Mapping von Namen zu ASCII-Codes:

| ASCII                 |
|-----------------------|
| +LINE_FEED = 10       |
| +CARRIAGE_RETURN = 13 |
| +SEMICOLON = 59       |
| +                     |

Abbildung 16 - ASCII Klassendiagramm

Dank dieser Lösung mit einer lokalen Datei muss für die Anwendung keine Datenbank erstellt werden.

Sie ist unabhängiger von ihrer Umgebung und kann schneller auf einem System integriert werden.

## 2.5.5 Tabellen-Auswahl Backend

Nachdem eine Config per Selectbox ausgewählt wurde erhält das Backend eine Anfrage zur Abfüllung der nächsten Selectbox. Es wird versucht, eine Verbindung zum Datenbank-Server herzustellen. Ist dies erfolgreich werden alle verfügbaren Datenbanken auf dem Server abgefragt, um diese dann in einer Response an das Frontend zu geben. Wenn dann eine der Datenbanken ausgewählt wird, passiert dasselbe Spiel nochmals, nun wird aber direkt auf die ausgewählte Datenbank verbunden, und es werden alle Tabellen ausgelesen und zurückgegeben.

## 2.5.6 Snippet-Auswahl Frontend

Copy Problem!

### 2.5.6.1 PHP-Model-Klasse

Die Model-Klasse ist ein Abbild der Datenbank Tabelle. Sie wird verwendet, um einen Datensatz der entsprechenden Tabelle zu laden und manipulieren.

```
<?php
namespace Model;

use JsonSerializable;

class KLASSENNAME extends BaseModel implements JsonSerializable {

    private ?int      $id      = NULL;
    private string    $articleNo = '';
    private ?DateTime $validBegin = NULL;
    private ?DateTime $validEnd  = NULL;
    .....

    public function jsonSerialize(): array {
        return [
            'id'          => $this->id,
            'articleNo'   => $this->articleNo,
            'validBegin'  => $this->validBegin,
            'validEnd'    => $this->validEnd,
            .....
        ]
    }

    public static function createObject(array $data): ERParticleService {
        $obj->id      = isset($data['id'])      ? $data['id']      : NULL;
        $obj->articleNo = isset($data['ArticleNo']) ? $data['ArticleNo'] : '';
        $obj->validBegin = isset($data['ValidBegin']) ? $data['ValidBegin'] : NULL;
        $obj->validEnd  = isset($data['ValidEnd']) ? $data['ValidEnd'] : NULL;
        .....
    }

    public function getId(): ?int {
        return $this->id;
    }

    public function setId(?int $id) {
        $this->id = $id;
    }

    .....
}
}
```

**Attribute**

**JSON-Serialize**

**Objekterstellung**

**Getters & Setters**

## 2.6 Kontrollieren

Zur Qualitätssicherung ist folgendes Testprotokoll definiert, welches alle notwendigen Funktionen der Applikation abdeckt.

Die Tests werden manuell durchgeführt.

### 2.6.1 Testfälle

|                     |   |
|---------------------|---|
| ID                  | T_001   |
| Beschreibung        | Eine neue Config über das Formular hinterlegen.   |
| Testvoraussetzung   | Keine.  |
| Testschritte        | <ol style="list-style-type: none"><li>1. «New config» Button drücken</li><li>2. Formular mit Daten befüllen</li><li>3. Formular mittels «Add» Button abschicken</li></ol> |
| Erwartetes Ergebnis | Die Config wird richtig in der Config-Datei hinterlegt und steht in der Config-Selectbox zur Auswahl.   |

Tabelle 8 – Testfall 1

|                     |  |
|---------------------|--|
| ID                  | T_002  |
| Beschreibung        | Eine Config aus der Config-Selectbox auswählen.  |
| Testvoraussetzung   | Es wurde mindestens eine Config hinterlegt.  |
| Testschritte        | <ol style="list-style-type: none"><li>1. Auf der Selectbox mit Label «Choose a config» eine Config auswählen</li></ol> |
| Erwartetes Ergebnis | Unterhalb dieser Selectbox erscheint nun eine weitere, mit dem Label «Choose a database»                               |

Tabelle 9 – Testfall 2

|                     |  |
|---------------------|--|
| ID                  | T_003  |
| Beschreibung        | Eine Config aus der Config-Selectbox auswählen, mit welcher keine Datenbankverbindung aufgebaut werden kann.   |
| Testvoraussetzung   | Es wurde mindestens eine Config hinterlegt, auf welche eine Datenbankverbindung unmöglich ist (z.B. mit Host, der kein Server ist oder absichtlich falschem Passwort). |
| Testschritte        | 1. Auf der Selectbox mit Label «Choose a config» eine fehlerhafte Config auswählen   |
| Erwartetes Ergebnis | Unterhalb dieser Selectbox erscheint ein Ladezeichen, welches durch eine Fehlermeldung ersetzt wird, sobald der Server Antwort gibt.                                   |

Tabelle 10 – Testfall 3

|                     |   |
|---------------------|---|
| ID                  | T_004   |
| Beschreibung        | Eine Datenbank aus der Datenbank-Selectbox auswählen.                                 |
| Testvoraussetzung   | Es wurde eine Config in der Config-Selectbox ausgewählt.                              |
| Testschritte        | 1. Auf der Selectbox mit Label «Choose a database» eine Datenbank auswählen           |
| Erwartetes Ergebnis | Unterhalb dieser Selectbox erscheint nun eine weitere, mit dem Label «Choose a table» |

Tabelle 11 – Testfall 4

|                     |   |
|---------------------|---|
| ID                  | T_005   |
| Beschreibung        | Eine Tabelle aus der Tabellen-Selectbox auswählen.  |
| Testvoraussetzung   | Es wurde eine Datenbank in der Datenbank-Selectbox ausgewählt.  |
| Testschritte        | 1. Auf der Selectbox mit Label «Choose a table» eine Tabelle auswählen  |
| Erwartetes Ergebnis | Der Rechte Bereich des Layouts verliert die graue Überdeckung, und man kann damit interagieren (Buttons drücken). |

Tabelle 12 – Testfall 5

|                     |  |
|---------------------|--|
| ID                  | T_006  |
| Beschreibung        | Ein PHP-Model-Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation bekannt sind. |
| Testschritte        | 1. Button «Model Class» unter PHP klicken.   |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem das Code-Snippet dargestellt wird.                |

Tabelle 13 – Testfall 6

|                     |   |
|---------------------|---|
| ID                  | T_007   |
| Beschreibung        | Ein PHP-Model-Snippet generieren.   |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation unbekannt sind.  |
| Testschritte        | 1. Button «Model Class» unter PHP klicken.  |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem eine Fehlermeldung und der Unbekannte Dateityp, der den Fehler ausgelöst hat, ausgegeben werden. |

Tabelle 14 – Testfall 7

|                     |  |
|---------------------|--|
| ID                  | T_008  |
| Beschreibung        | Ein PHP-Gateway-Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation bekannt sind. |
| Testschritte        | 1. Button «Gateway Class» unter PHP klicken.   |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem das Code-Snippet dargestellt wird.                |

Tabelle 15 – Testfall 8

|                     |   |
|---------------------|---|
| ID                  | T_009   |
| Beschreibung        | Ein PHP-Gateway-Snippet generieren.   |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation unbekannt sind.  |
| Testschritte        | 1. Button «Gateway Class» unter PHP klicken.  |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem eine Fehlermeldung und der Unbekannte Dateityp, der den Fehler ausgelöst hat, ausgegeben werden. |

Tabelle 16 – Testfall 9

|                     |  |
|---------------------|--|
| ID                  | T_010  |
| Beschreibung        | Ein ExtJs-Model-Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation bekannt sind. |
| Testschritte        | 1. Button «Model Class» unter ExtJs klicken.   |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem das Code-Snippet dargestellt wird.                |

Tabelle 17 – Testfall 10

|                     |   |
|---------------------|---|
| ID                  | T_011   |
| Beschreibung        | Ein ExtJs-Model -Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation unbekannt sind.  |
| Testschritte        | 1. Button «Model Class» unter ExtJs klicken.  |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem eine Fehlermeldung und der Unbekannte Dateityp, der den Fehler ausgelöst hat, ausgegeben werden. |

Tabelle 18 – Testfall 11



|                     |  |
|---------------------|--|
| ID                  | T_012  |
| Beschreibung        | Ein ExtJs-GridList-Snippet generieren.   |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation bekannt sind. |
| Testschritte        | 1. Button «Grid List» unter ExtJs klicken.   |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem das Code-Snippet dargestellt wird.                |

Tabelle 19 – Testfall 12

|                     |   |
|---------------------|---|
| ID                  | T_013   |
| Beschreibung        | Ein ExtJs-GridList-Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation unbekannt sind.  |
| Testschritte        | 1. Button «Grid List» unter ExtJs klicken.  |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem eine Fehlermeldung und der Unbekannte Dateityp, der den Fehler ausgelöst hat, ausgegeben werden. |

Tabelle 20 – Testfall 13

|                     |  |
|---------------------|--|
| ID                  | T_014  |
| Beschreibung        | Ein ExtJs-Add-Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation bekannt sind. |
| Testschritte        | 1. Button «Add Dialog» unter ExtJs klicken.  |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem das Code-Snippet dargestellt wird.                |

Tabelle 21 – Testfall 14

|                     |   |
|---------------------|---|
| ID                  | T_015   |
| Beschreibung        | Ein ExtJs- Add -Snippet generieren.   |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation unbekannt sind.  |
| Testschritte        | 1. Button «Add Dialog» unter ExtJs klicken.   |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem eine Fehlermeldung und der Unbekannte Dateityp, der den Fehler ausgelöst hat, ausgegeben werden. |

Tabelle 22 – Testfall 15

|                     |  |
|---------------------|--|
| ID                  | T_016  |
| Beschreibung        | Ein ExtJs-Edit-Snippet generieren.   |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation bekannt sind. |
| Testschritte        | 1. Button «Edit Dialog» unter ExtJs klicken.   |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem das Code-Snippet dargestellt wird.                |

Tabelle 23 – Testfall 16

|                     |   |
|---------------------|---|
| ID                  | T_017   |
| Beschreibung        | Ein ExtJs- Edit -Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation unbekannt sind.  |
| Testschritte        | 1. Button «Edit Dialog» unter ExtJs klicken.  |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem eine Fehlermeldung und der Unbekannte Dateityp, der den Fehler ausgelöst hat, ausgegeben werden. |

Tabelle 24 – Testfall 17

|                     |  |
|---------------------|--|
| ID                  | T_018  |
| Beschreibung        | Ein ExtJs-Info-Snippet generieren.   |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation bekannt sind. |
| Testschritte        | 1. Button «Info Dialog» unter ExtJs klicken.   |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem das Code-Snippet dargestellt wird.                |

Tabelle 25 – Testfall 18

|                     |   |
|---------------------|---|
| ID                  | T_019   |
| Beschreibung        | Ein ExtJs- Info -Snippet generieren.  |
| Testvoraussetzung   | Es wurde eine Tabelle in der Tabellen-Selectbox ausgewählt, die Dateitypen beinhaltet, die der Applikation unbekannt sind.  |
| Testschritte        | 1. Button «Info Dialog» unter ExtJs klicken.  |
| Erwartetes Ergebnis | Unterhalb des Titels «Output:» erscheint ein schwarzes Fenster, in dem eine Fehlermeldung und der Unbekannte Dateityp, der den Fehler ausgelöst hat, ausgegeben werden. |

Tabelle 26 – Testfall 19

## 2.6.2 Testprotokolle

## 2.7 Auswerten

Text

Bezug auf die ganze Arbeit

Persönliches Fazit

## 2.8 Verzeichnisse

### 2.8.1 Abbildungsverzeichnis

|  |    |
|--|----|
| Abbildung 1 - Use-Case-Diagramm.....                       | 22 |
| Abbildung 2 - MockUp 1.....                                | 24 |
| Abbildung 3 - MockUp 2.....                                | 25 |
| Abbildung 4 - Selectboxen.....                             | 29 |
| Abbildung 5 - Config-Formular.....                         | 30 |
| Abbildung 6 - Backend-Struktur.....                        | 31 |
| Abbildung 7 - Backend router Ausschnitt.....               | 31 |
| Abbildung 8 - Response-Klasse Ausschnitt.....              | 32 |
| Abbildung 9 - Controller-Klasse Ausgabe.....               | 32 |
| Abbildung 10 - Utils-Klasse ErrorThrower.....              | 32 |
| Abbildung 11 - Klassendiagramm DB Verbindungen.....        | 33 |
| Abbildung 12 - ConfigController getTables Funktion.....    | 34 |
| Abbildung 13 - ConfigController createConfig Funktion..... | 34 |
| Abbildung 14 - ConfigController Klassendiagramm.....       | 34 |
| Abbildung 15 - ConfigManager Dateibearbeitung.....         | 35 |
| Abbildung 16 - ASCII Klassendiagramm.....                  | 35 |

### 2.8.2 Tabellenverzeichnis

|  |    |
|--|----|
| Tabelle 1 - Dokumentenhistorie.....                            | 4  |
| Tabelle 2 - Dokumenteneigenschaften.....                       | 4  |
| Tabelle 3 - Code-Snippet Arten.....                            | 6  |
| Tabelle 4 - Zeitplan Zusammenfassung.....                      | 19 |
| Tabelle 5 - Use-Case Beschreibung «add new config».....        | 23 |
| Tabelle 6 - Use-Case Beschreibung «choose database table»..... | 23 |
| Tabelle 7 - Use-Case Beschreibung «generate Code-Snippet»..... | 23 |
| Tabelle 8 - Testfall 1.....                                    | 37 |
| Tabelle 9 - Testfall 2.....                                    | 37 |
| Tabelle 10 - Testfall 3.....                                   | 38 |
| Tabelle 11 - Testfall 4.....                                   | 38 |
| Tabelle 12 - Testfall 5.....                                   | 38 |
| Tabelle 13 - Testfall 6.....                                   | 39 |
| Tabelle 14 - Testfall 7.....                                   | 39 |
| Tabelle 15 - Testfall 8.....                                   | 39 |
| Tabelle 16 - Testfall 9.....                                   | 40 |
| Tabelle 17 - Testfall 10.....                                  | 40 |
| Tabelle 18 - Testfall 11.....                                  | 40 |
| Tabelle 19 - Testfall 12.....                                  | 41 |

|                                |    |
|--------------------------------|----|
| Tabelle 20 – Testfall 13 ..... | 41 |
| Tabelle 21 – Testfall 14 ..... | 41 |
| Tabelle 22 – Testfall 15 ..... | 42 |
| Tabelle 23 – Testfall 16 ..... | 42 |
| Tabelle 24 – Testfall 17 ..... | 42 |
| Tabelle 25 – Testfall 18 ..... | 43 |
| Tabelle 26 – Testfall 19 ..... | 43 |
| Tabelle 27 – Glossar .....     | 47 |

## 2.8.3 Links

[https://www.ict-berufsbildung-bern.ch/resources/lperka\\_OdA\\_200617.pdf](https://www.ict-berufsbildung-bern.ch/resources/lperka_OdA_200617.pdf)

Website der Berufsbildung Bern...

Heruntergeladen am: 18.04.2024

<https://www.lucidchart.com/pages/>

Web-Tool für MockUps

Erstellt am: 18.04.2024

## 2.8.4 Glossar / Abkürzungen

|      |                                 |
|------|---------------------------------|
| CRUD | Create, Read, Update, Delete    |
| SWO  | Simple Web Office               |
| AJAX | Asynchronous JavaScript and XML |
|      |                                 |
|      |                                 |
|      |                                 |

Tabelle 27 – Glossar

## 2.9 Anhang

Besprechungsprotokolle

Quellcode