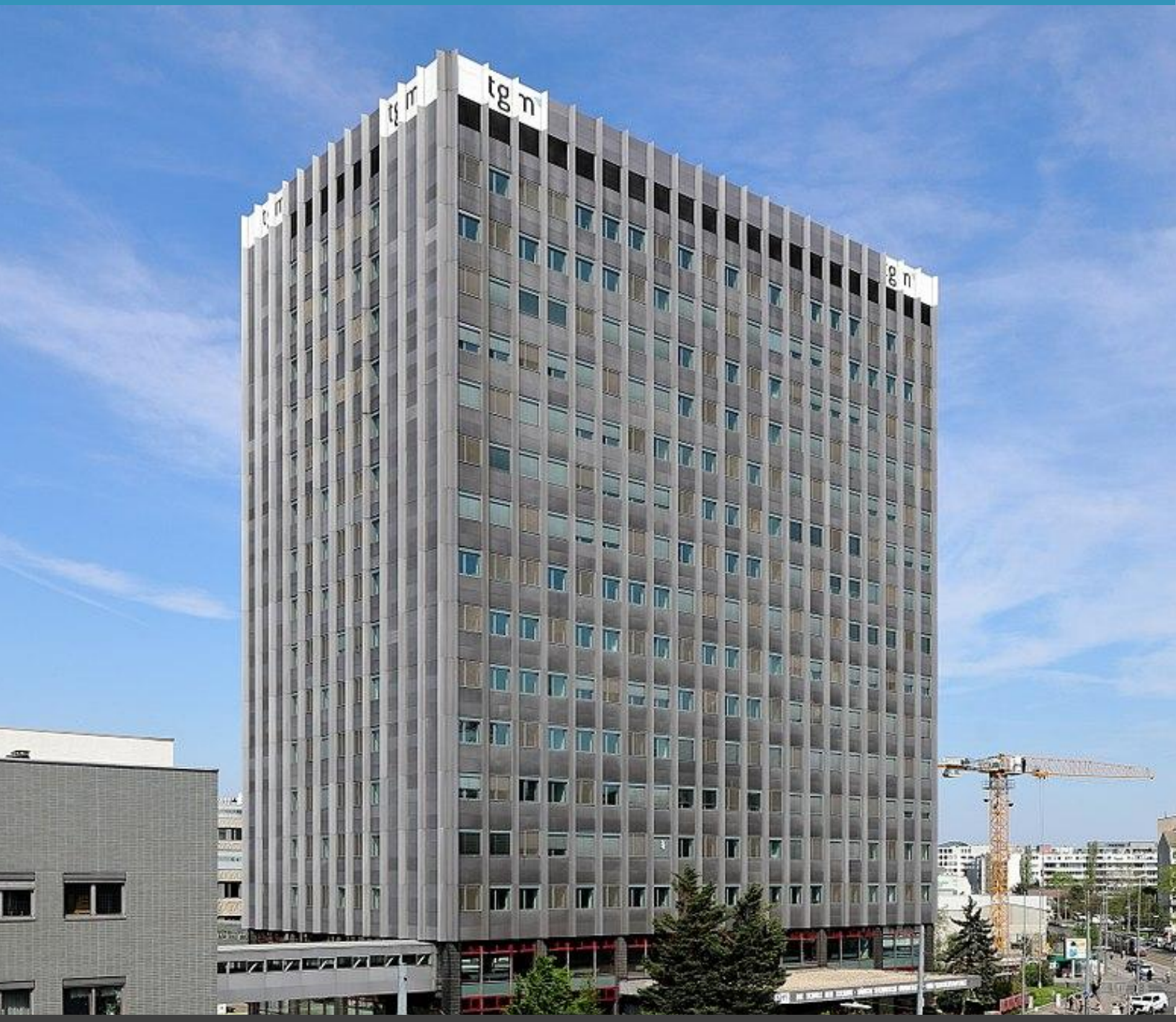
SWP WiSe 2025/26



Projektbeschreibung

Refactoring & Weiterentwicklung
Lohn- & Personalverrechnung

Projektbeschreibung

Refactoring – Lohn- & Personalverrechnung

I. Einleitung

In der Softwareentwicklung wird es immer häufiger notwendig alte Programme und Methoden neu aufzuarbeiten. Dies setzt voraus, dass die Entwickler auch geschult im Anwendungs- und Methodenrefactoring sind.

Refactoring bedeutet hierbei im engeren Sinne, dass alte Strukturen den aktuellen Programmierparadigmen unterworfen und damit für zukünftige Entwickler auch aufbereitet werden. Aufgrund der immer volatileren Jobsituation im IT-Sektor kann man damit im weiteren Sinn auch eine ausreichende Dokumentation sehen, in Form von Kommentaren im Code, die es einfacher machen den Code von wechselnden Mitarbeitern warten lassen zu können oder eben einer gesonderten Dokumentation samt Ablaufdiagrammen, die den Code gut erklären.

II. Situationsanalyse

Nachdem in einer Softwareentwicklungsfirma der Personalverrechner pensioniert wurde, übernahm ein junger, motivierter, neuer Mitarbeiter dessen Verantwortungsbereich. In der ersten Woche stieß er dabei auf zahlreiche alte Verfahren und ein altes benutztes Programm, das ein Entwickler vor einiger Zeit für den pensionierten Personalverrechner erstellt hatte.

Ihr wurdet in dieser Firma gerade als Junior-Developer angestellt und in Absprache mit der Firmenführung wurde ein Team aus mehreren neuen Mitarbeitern gebildet, dass nun seinem Bereich zugeordnet wurde, um das alte Programm auf einen neuen Standard zu heben.

Um die Entwickler für ihre Aufgabe zu schulen, gibt der Personalverrechner dem Team einen ausreichend tiefen Einblick in die Grundlagen der Personalverrechnung. Eure Aufgabe als Team ist es der Firmenführung euren Wert zu beweisen, um einen guten Firmeneinstand zu haben.

III. Datenbankenanforderungen

Datenbanken stellen die beste Möglichkeit dar, Daten professionell zu sammeln, zu verarbeiten und gesichert zu lagern. Es gibt dabei zahlreiche Möglichkeiten an Datenbankmodellen, -sprachen und Möglichkeiten diese umzusetzen. (lokale oder öffentliche)

Es wird somit notwendig sein, die Daten der Mitarbeiter in einer Datenbank zu sammeln und diese über eine GUI bearbeitbar zu machen. Es liegt hierbei die barrierefreie Nutzbarkeit im Vordergrund und es soll somit auch der Code bereinigt und weiterentwickelt werden.

Wichtig ist hierbei jedoch, dass Daten nicht einfach gelöscht oder überschrieben werden sollen, da Änderungen im Verlauf der Zeit nachverfolgt werden können müssen. Es ist somit im Bereich der Personalverrechnung (und einigen anderen Bereichen) **dringend notwendig**, sogenannte **historische Datenbanken** zu erstellen, bei denen jeweils auch ein **zeitlicher Geltungsbereich der Daten** mit angegeben wird.

Etwaige Daten, die in der Datenbank gesammelt werden sollen, sind:

- **Stammdaten**: Personalnummer, Name, Geburtsdatum, Sozialversicherungsnummer, Adresse, Geschlecht, Kinder, …
- **Kinder**: Personalnummer (Elternteil), Name, Geburtsdatum, Geschlecht, Sozialversicherungsnummer, …
- **Dienstvertragsdaten**: Personalnummer, Dienstvertragsart, Bruttogehalt, vereinbarte Wochenarbeitszeit, Einstufung im Kollektivvertrag, All in?,
- **Steuerliche Vorteile**: Personalnummer, Familienbonus(+), Freibetrag, Pendlerpauschale, …
- **Lohnverrechnung (DN)**: Personalnummer, Abrechnungsmonat, Brutto, Netto, Mehrarbeit, Überstunden, SV-Abgaben, Lohnsteuer, Sonderzahlungen, Sachbezüge, …
- **Lohnverrechnung (DG)**: Personalnummer, Dienstgeber-SV-Abgaben, Kommunalsteuer, Ubahn-Steuer, …
- **Kollektivvertragsdaten**: Kollektivvertrag, Jahr, Stufen, Mindestgehalt, …
- **Urlaubsanspruch (DN)**: Personalnummer, Jahr, verbrauchte Urlaubstage, Urlaub (genehmigter Zeitraum)

IV. User Interface / User Experience

Benutzeroberflächen sind heute ein unverzichtbarer Bestandteil nahezu jeder Software. Kaum jemand arbeitet noch direkt mit Datenbankbefehlen oder schreibt SQL-Statements manuell, um Daten an ein Datenbankmanagementsystem (DBMS) zu übermitteln. Stattdessen ist es Standard, dass Anwendungen **Eingabemasken** bereitstellen, in denen Daten komfortabel erfasst oder geändert werden können. Die eigentliche Generierung und Ausführung der Datenbankbefehle erfolgt dabei automatisiert im Hintergrund – ausgelöst durch einfache Interaktionen wie das Klicken auf einen Button.

Warum ist das wichtig?

- **Benutzerfreundlichkeit**: Ein gutes UI reduziert die Einstiegshürde und minimiert Fehler.
- **Effizienz**: Komplexe Abläufe werden für den Anwender vereinfacht.
- **Akzeptanz**: Anwendungen mit intuitiver Bedienung werden eher genutzt.

Aktueller Trend:
Viele Softwareprojekte verabschieden sich von klassischen Desktop-GUIs (z. B. Windows-Programme) und setzen stattdessen auf **Web- und Cloud-Lösungen**, die direkt im Browser laufen.

Vorteile:

- Plattformunabhängigkeit (Windows, macOS, Linux, mobile Endgeräte)
- Einfachere Updates und Wartung (kein lokales Installieren)
- Bessere Integration in moderne IT-Infrastrukturen

Für unser Projekt bedeutet das:

- Die bestehende Desktop-Anwendung wird nicht nur refaktoriert, sondern **architektonisch modernisiert**, um eine zukunftsorientierte **Oberfläche** bereitzustellen.
- Dabei sollen **UI/UX-Prinzipien** berücksichtigt werden: klare Navigation, konsistente Gestaltung, Barrierefreiheit (z. B. sinnvolle Labels, Kontraste).

Verpflichtend bleibt jedoch die Umsetzung der in den Arbeitspaketen enthaltenen Punkte. Ihr entscheidet, ob ihr ein Framework (z. B. Django, Flask) oder ein leichtes Setup nutzt.

V. Aufgabenstellung

Im Rahmen dieses Projekts soll wie bereits in der Situationsanalyse beschrieben das bestehende Programm modernisiert und um zahlreiche Verbesserungen erweitert werden.

1 Anforderungen an das Programm

- **Datenbank**: Für Mitarbeiter, die Abrechnungen der Jeweiligen, evtl. der jeweiligen Daten die relevant für die Abrechnung usw sind.
- **UI**: Eine Lösung für eine Oberfläche für die Bearbeitung der Datenbankeinträge, sowie zur Abrechnung der Mitarbeiter, Erhöhung der Kollektivverträge usw. (Umsetzung via moderner GUI, UI als Cloudlösung via Render/Heroku mittels GIT)
- **PDF-Ausgabe:** Der Druck von Stammdatenblättern oder Lohn- und Gehaltszetteln sollte automatisiert möglich sein.
- **Lohnverrechnung:** Eine automatische Lohnverrechnung sollte möglich sein. (Das bedeutet ausgehend vom bisherigen Skript werden die aktuellen KV-Daten aus der Datenbank herangezogen und die Berechnung des Netto-Lohns durchgeführt.)
- **(bei Gruppengröße von 4 Personen) Zeiterfassung:** Die Erfassung von Arbeits-, Kranken-, sowie Urlaubstagen sollte in einer Tabelle möglich sein.
- **Login & Startseite:** Authentifizierung berechtigter Nutzer
- **Testing**: Testen des DBMS, Einrichtung der Schnittstellen, Überprüfung der Lohnverrechnung auf korrekte Ergebnisse.

Projektsprache ist Python! (Ausnahme Django mit html/css/js/… für das Frontend)

2 Fristen

Abgabe Aufgabenverteilung	**16.09.2025** 09:50
Abgabe Projektmanagement	**30.09.2025** 23:59
PAP-Besprechung	**30.09.2025** 08-00 – 10:40
Projektabgabe (Abgabe des Portfolios, Kontrolle Peerfeedback)	**24.10.2025** 23:59
Projektpräsentation	**04.11.2025** 08:15

3 Wöchentliche Mitarbeit

Neben den Fristen zur Abgabe wird auch an den Terminen zur Bearbeitung des Projekts der Fortschritt und die laufende Mitarbeit dokumentiert. Die Mitarbeit fließt direkt am Ende der Bewertung als Multiplikator der Gesamtbewertung ein und kann somit, je nach Mitarbeit, die Bewertung um **bis zu 10% verbessern oder verschlechtern**.

$$Mitarbeit = 0{,}9 - 1{,}1$$

4 Gruppenaufteilung

Die Aufgabe ist **kein Einzelprojekt**, sondern für drei bis vier Personen vorgesehen. Folgende Punkte sollen bei Projektstart auf die Gruppenmitglieder aufgeteilt werden. Ausgehend davon erfolgt dann die Beurteilung der Einzelleistungen.

MUSS (Pflicht)

- **Projektmanagement**: Erstellung eines Projektplans mit Beschreibung des Projektauftrages, Organigramm und Gruppeneinteilung, Zeitanalyse (SOLL), Aufgabenverteilung, PSP & Gantt-Diagramm, …
- **Datenbank**: Stammdaten, Lohnabrechnungen, KV-Daten, Historisierung (keine Löschungen).
- **UI** : Oberfläche für Stammdaten, Lohnverrechnung, KV-Anpassungen.
- **Lohnverrechnung**: Automatische Berechnung von Brutto/Netto inkl. Abgaben.
- **Login & Startseite**: Authentifizierung (mindestens einfache Benutzerverwaltung).
- **Testing (Basis)**: Überprüfung der Lohnverrechnung und DB-Anbindung.

SOLL (empfohlen)

- **PDF-Ausgabe**: Stammdatenblätter oder Lohnzettel.
- **UI/UX-Verbesserungen**: Responsives Design, klare Navigation.
- **Such- und Filterfunktionen** Datenbankfilter in der Oberfläche.
- **Erweiterte Tests**: Unit- und Integrationstests.

KANN (Bonus)

- **CI/CD-Pipeline**: Automatisierte Tests und Builds via GitHub Actions.
- **Deployment in der Cloud**: Bereitstellung auf Render oder Heroku.
- **Zeiterfassung**: Arbeits-, Kranken- und Urlaubstage.
- **Erweiterte Security**: Rollen, Passwort-Hashing.
- **Automatisierte PDF-Reports**: Monatsberichte (Monatszusammenfassung neu eingestellter Mitarbeiter, monatlicher Auszahlung an Löhnen/Gehältern, etc.)

5 Entwicklungsportfolio

Jede:r Schüler:in erstellt ein **persönliches Entwicklungsportfolio** als Teil der Bewertung. **Ziel:** Reflexion über den eigenen Beitrag und die Lernfortschritte, nicht nur „fertiger Code“. (siehe Vorlage)

Inhalt

- **4 Codefragmente** (mind. je 10–20 Zeilen, keine ganzen Dateien, bitte Syntax-Highlighting)
- Zu jedem Fragment:
    - **Warum habe ich dieses Beispiel gewählt?**
      (z. B. Durchbruch, besonders knifflig, zeigt meine Kompetenz)
    - **Was macht der Code?**
      (Funktion, Kontext im Projekt)
    - **Warum ist er so implementiert?**
      (Entscheidungen, Patterns, Alternativen)
    - **Wie habe ich die Qualität sichergestellt?**
      (Tests, Reviews, Refactoring)
    - **Was habe ich dabei gelernt?**
      (Technisch und methodisch)
- Abschließende Reflexion zum Gesamtprojekt
    - Was habe ich gelernt?
    - Wo sehe ich meine eigenen Stärken im Projekt?
    - Was könnte man noch verbessern?
    - Wo möchte ich mich beim nächsten Projekt verbessern?

Form

- **Dokument (PDF)**, ca. 4-7 Seiten. (kurz und knapp)
- Screenshots oder Codeblöcke + erklärender Text.
- Abgabe gemeinsam mit dem Projekt.

6 Beurteilung

Kriterium	**Anteil**
Programmierung	25%
Projektmanagement	35%
Entwicklungsportfolio	20%
Peerfeedback	10%
Abschlusspräsentation	10%
Projektbewertung	100%

Aufgrund von Kapitel 3 ergibt sich für die Endbewertung noch ein Multiplikator (zwischen 0,9 und 1,1) als Faktor zur Projektbewertung:

$$Endbewertung = Projektbewertung \cdot Mitarbeit$$

6.1 Programmierung

Diese Punkte fallen weg, falls sich der Code nicht ausführen lässt. Testen Sie ihren Code auch außerhalb von Spyder, VS-Code, o.ä., z.B. mit der Windows-PowerShell!

Weitere Abzüge ergeben sich durch auftretende Fehler, die sich durch einen der folgenden Punkte ergeben:

- Fehlerhafter Datentyp wird übergeben / eingegeben / verarbeitet
- Schnittstellen arbeiten nicht miteinander
- Allgemein auf keine Fehlerbehandlung geachtet
- Fehlende Module beim User (nicht in Dokumentation/readme.md erwähnt)

Die angegebenen Prozent sind jeweils die maximale mögliche Bewertung. Diese werden nur vergeben, wenn die Aufgabe perfekt umgesetzt wurde:

- 1. Sinnvolle Variablen-Benennung
- 2. Kurzer und schlanker Code, übersichtlich und angemessen für die Aufgabe
- 3. Kein redundanter Code, alles wird benötigt (sonst löschen)
- 4. Sinnvolles Programmierparadigma (objektorientiert, prozedural, …) und entsprechender Einsatz von Klassen, Listen und Dictionaries

a) Nutzung von KI zur Unterstützung

Aufgrund der vermehrten Nutzung von sogenannter KI-Software wird eine Überprüfung der Eigenleistungen erforderlich.

Der Einsatz zur Unterstützung bei der Programmierung ist nicht reglementiert, jedoch sollte jederzeit der erzeugte Code auch auf Fehler oder fälschliche Veränderung von bestehendem Code überprüft werden. Dementsprechend sollte KI immer mit Vorsicht genossen werden!

Weiters sollte wie üblich der Code verstanden und erklärt werden können. Ein Bezug auf reine Funktionalität ist nicht zweckmäßig und würde eine Aberkennung der fachlich technischen Beurteilung mit sich ziehen.

6.2 Projektmanagement

Das Projektmanagement hat innerhalb von ein bis zwei Wochen abgeschlossen zu sein und ist somit als Teilleistung fällig. Es sind die geforderten Inhalte aus der Aufgabenstellung auf jeden Fall zu erfüllen, sowie folgende Formvorgaben. (**Eine Auswertung je Gruppe**)

- Titelblatt (mit Namen und Aufgaben der Teilnehmer)
- Inhaltsverzeichnis
- Seitenbeschriftungen & -nummerierungen
- Quellen
- Abbildungsbeschriftung
- Standard / Absatztext wird in **Calibri**, 11-12pt mit 1,5-fachen Zeilenabstand verfasst. Überschriften nicht größer als 20pt
- Sinnhafte Erklärungen, sowie ein angemessenes Fachvokabular

6.3 Abschlusspräsentation

Die Präsentation des Projekts dient in erster Linie die eigenen Präsentationsfähigkeiten zu üben und vor allem fachlich zu testen. Es wird dabei auf folgende Punkte geachtet:

- Ein*e Teilnehmer*in präsentiert kurz das Team und das Projekt
- Die Gruppenmitglieder präsentieren ihre Teilaufgaben am Gesamtprojekt
- Kurze Vorführung der Aufgabe
- Diskussion mit dem Publikum
- Der Vortrag dauert nicht länger als 5 min/Mitglied

6.4 Zusätzliche Punkteabzüge

Wie bereits unter 6.1a) erklärt, ist die Nutzung von Online-Tools erlaubt, jedoch gilt diese Nutzung als Plagiarismus, sofern die erstellten Elemente 1:1 verwendet werden. Bei solchen Fällen handelt es sich um keine Eigenleistung und die Bewertung der Aufgabe entfällt.

Dies beinhaltet nicht nur jederzeit die Erklärung der eigenen Codezeilen, sondern auch die Mitarbeit im Unterricht bei der der aktuelle Stand jederzeit vorgezeigt werden soll. Sollte es Unstimmigkeiten geben, kann gesondert nach Abschluss der vorläufigen Bewertung noch ein Feedbackgespräch stattfinden, bei dem im Gespräch klar werden soll, inwiefern eine Eigenleistung vorliegt. Sollte im Zweifel sein, ob die Leistung selbst erbracht wurde, so werden die Punkte diese(r) Teilaufgabe(n) gestrichen.

VI. Abgabe

1 Fristen

Abgabe Aufgabenverteilung	**16.09.2025** 09:50
Abgabe Projektmanagement	**30.09.2025** 23:59
PAP-Besprechung	**30.09.2025** 08-00 – 10:40
Projektabgabe (Abgabe des Portfolios, Kontrolle Peerfeedback)	**24.10.2025** 23:59
Projektpräsentation	**04.11.2025** 08:15

2 Umfang

In der Abgabe via eLearning sollten folgende Punkte enthalten sein:

- Je Gruppe einmalig das komplette Projektverzeichnis
    - …, sowie eine readme.md in der mögliche Testversionen angegeben werden die online sind.
- Je Schüler:in selbst jeweils das Entwicklungsportfolio.
- Alle Feedbacks im jeweiligen eLearning-Punkt für die Teilnehmer:innen.