

Dokumentation zur Projektarbeit



# **Schülerparkausweise**

## **Teilprojekt Onlineverwaltungstool**

*Webanwendung für das Verwalten der Schülerparkausweise für die  
Lehrkräfte*

Abgabetermin: Regensburg, 19.04.2023

### **Gruppe 4**

Marco Pfab

Jonas Blümel

Frank Sylla

Sebastian Knobel

Maksim Hermann



### **Betrieb:**

IT Service GmbH

Service Straße 1

42069 Seewen

## Inhaltsverzeichnis:

- 1. Einleitung (Frank)**
  - 1.1. Projektbeschreibung
  - 1.2. Projektziele
  - 1.3. Projektbegründung
  - 1.4. Projektschnittstellen
- 2. Projektplanung (Sebastian)**
  - 2.1. Projektphasen
  - 2.2. Ressourcenplanung
  - 2.3. Entwicklungsprozess
- 3. Analysephase (Sebastian)**
  - 3.1. IST-Analyse
  - 3.2. Kostenanalyse
- 4. Entwurfsphase (Maksim)**
  - 4.1. Zielplattform
  - 4.2. Architekturdesign
  - 4.3. Entwurf der Benutzeroberfläche
  - 4.4. Datenbankmodell
  - 4.5. Maßnahmen zur Qualitätssicherung
  - 4.6. Pflichtenheft
- 5. Implementierungsphase (Marco)**
  - 5.1. Implementierung Python
  - 5.2. Implementierung der Benutzeroberfläche
  - 5.3. Testen der Anwendung
- 6. Dokumentationsphase (Jonas)**
- 7. Fazit (Jonas)**
  - 7.1. SOLL-/IST-Analyse
  - 7.2. Lesson Learned
  - 7.3. Ausblick
- 8. Anhang**

## Abkürzungsverzeichnis:

OVT – Onlineverwaltungstool

BS3 – Berufliches Schulzentrum Matthäus Runtinger

## 1. Einleitung (Frank)

In der folgenden Projektdokumentation wird der Ablauf des Teilprojektes OVT, das durch die Autoren im Rahmen ihres Berufsschulunterrichts durchgeführt wurde, erläutert. Das Projekt wird in der IT Service GmbH durchgeführt, welche der Projektbetrieb der Autoren ist. Bei der GmbH sind zurzeit 5 Mitarbeiter beschäftigt. Die IT Service GmbH hat sich auf Web-Development und Datenverwaltung spezialisiert. Ziel dieser Dokumentation ist es, die durchzuführenden Schritte des Projektes von der Planung bis zum Deployment zu erläutern und dies mit geeigneten Diagrammen und Dokumenten zu unterstützen.

### 1.1. Projektbeschreibung

Im Rahmen des Projektes zur Neugestaltung der Schülerparkplatzverwaltung und zur Entlastung der Lehrkräfte, soll in unserem Teilprojekt eine Webanwendung mit Datenbankzugriff erstellt werden, die es den Lehrern ermöglicht die Daten der Schülereinträge einzusehen, diese zu bearbeiten und zu löschen, aber auch neue Einträge zu erstellen. Den Lehrkräften soll außerdem die Möglichkeit geschaffen werden, die zugehörigen Parkausweise in dieser Webapplikation zu generieren.

Das Teilprojekt hat zum Ziel, eine Webanwendung mit Datenbankzugriff zu erstellen, um die Verwaltung des Schülerparkplatzes zu erleichtern und die Lehrkräfte zu entlasten. Die Webanwendung ermöglicht es den Lehrern, auf die Schülereinträge zuzugreifen, diese zu bearbeiten und zu löschen, aber auch neue Einträge zu erstellen. Die Webanwendung soll die Möglichkeit bieten, die zugehörigen Parkausweise in der Anwendung zu generieren. Um die Lehrkräfte zu unterstützen, wird die Webanwendung mit einem benutzerfreundlichen Interface gestaltet, das eine einfache Navigation und intuitive Benutzeroberfläche bietet. Darüber hinaus wird die Anwendung eine einfache, effiziente und schnelle Art und Weise für Lehrer bereitstellen, um auf die notwendigen Informationen zuzugreifen und Änderungen an den Schülereinträgen vornehmen zu können.

Die Webanwendung soll auch eine sichere und zuverlässige Datenbank für die Speicherung von Schülerdaten und Informationen bereitstellen. Eine angemessene Verschlüsselung und Sicherheitsmaßnahmen werden in der Anwendung implementiert, um sicherzustellen, dass alle Daten sicher und geschützt gespeichert und übertragen werden.

Die Webanwendung soll auch die Möglichkeit bieten, Parkausweise für die Schülerinnen und Schüler zu generieren, um sicherzustellen, dass sie berechtigt sind, den Parkplatz zu nutzen. Die Lehrer können diese Parkausweise in der Webanwendung generieren und ausdrucken, um eine schnelle und effiziente Verteilung zu ermöglichen.

Insgesamt zielt das Teilprojekt darauf ab, eine benutzerfreundliche, sichere und effektive Webanwendung zu schaffen, die den Lehrkräften eine einfache Möglichkeit zur Verwaltung der Schülerparkplatzdaten bietet und die Verwaltung des Schülerparkplatzes im Allgemeinen verbessert.

## 1.2. Projektziel

Ziel dieses Projektes ist die erfolgreiche Umsetzung der neuen Webanwendung, mit der automatisiert verschiedene Daten für den Nutzer dargestellt und die Parkausweise gedruckt werden können. Dadurch sollen die Lehrkräfte entlastet werden.

## 1.3. Projektbegründung

Die Berufsschule III Regensburg benötigt eine neue Webanwendung zur Verwaltung der Kfz-Kennzeichen und Parkausweise von Schülerinnen und Schülern, da die neue Schulverwaltungssoftware ASV keine entsprechende Funktion bietet. Die derzeitige Übergangslösung über eine Umfrage auf der Lernplattform Mebis hat mehrere Schwächen, wie zum Beispiel das Fehlen einer Speicherung der Parkausweise und den Bedarf an vielen manuellen Schritten. Das Ziel des Projekts ist es, eine automatisierte, datenbankgestützte Webanwendung zu entwickeln, welche die Erfassung, Verwaltung und den Druck der Parkausweise erleichtert und das Personal der Schulverwaltung entlastet. Die neue Anwendung soll einfach zu bedienen und mittels einer Login-Funktion gesichert sein, um den Datenschutz der Schülerinnen und Schüler zu gewährleisten. Durch eine erfolgreiche Umsetzung des Projekts wird die Verwaltung der Kfz-Kennzeichen und Parkausweise an der Berufsschule III Regensburg effizienter und fehlerfreier gestaltet.

## 1.4. Projektschnittstellen

Um an die Daten zu gelangen, muss die Anwendung mit einer PostgreSQL-Datenbank kommunizieren und Daten abfragen können. Diese befindet sich im schulinternen Verwaltungsnetz und kann direkt angesprochen werden. Zwar ist dieses Schulprojekt dem schulinternen Projekt Schülerparkplatzverwaltung zugeordnet, lässt sich davon aber weitestgehend abgrenzen, da es als einzige Schnittstelle die gemeinsame Datenbank gibt. Jedoch ist eine Abstimmung mit den Projektbeteiligten notwendig. Auch die Lehrerschaft soll bei der Oberflächengestaltung, vertreten durch die projektverantwortlichen Lehrkräfte, mit in den Entwicklungsprozess einbezogen werden.

## 2. Projektplanung (Sebastian)

### 2.1. Projektphasen

Für die Umsetzung des Projektes standen 50 Stunden zur Verfügung, wie es im Schulunterricht vorschrieben wurde. Bevor mit dem Projekt gestartet wurde, fand eine Aufteilung auf verschiedene Phasen statt, die den kompletten Prozess der Softwareentwicklung abdecken. Eine grobe Zeitplanung mit den Hauptphasen lässt sich aus Tabelle 1 entnehmen. Eine detailliertere Zeitplanung mit den einzelnen Schritten der unterschiedlichen Phasen findet sich im **Anhang A.1: Detaillierter Zeitplan**.

Projektphase		Geplante Zeit
Planungsphase		7 h
Entwurfsphase		4 h
Implementierungsphase		33 h
Dokumentationsphase		6 h
Gesamt		50 h

Tabelle 1. Grobe Planung

### 2.2. Ressourcenplanung

Im **Anhang A.2: Verwendete Ressourcen** sind alle Ressourcen aufgelistet, die im Rahmen des Projektes genutzt wurden, einschließlich Hard- und Softwareressourcen sowie Personal. Bei der Auswahl der verwendeten Software wurde darauf geachtet, dass keine Lizenzkosten anfallen und dass das erforderliche Know-how vorhanden ist, um Kosten zu minimieren.

### 2.3. Entwicklungsprozess

Als letzter Schritt vor dem tatsächlichen Beginn des Projektes musste durch die Autoren ein geeigneter Entwicklungsprozess gewählt werden. Durch diesen wird die Vorgehensweise bei der Umsetzung des Projektes definiert. Grundsätzlich soll sich die Umsetzung des Projektes an den Phasen des Wasserfall-Modells orientieren, da die Anforderungen durch das VVG überwiegend eindeutig definiert sind. Beim Wasserfall-Modell werden die einzelnen Phasen der Software-Entwicklung nacheinander durchlaufen, wobei eine Rückkehr in eine vorherige Phase jederzeit möglich ist. Im Bereich der Oberflächengestaltung soll es jedoch durch regelmäßige Rücksprache mit der Lehrerschaft einen agilen Prozess geben.

Zudem werden die Tests in die Implementierungsphase integriert, um eine stetige Überprüfung der Funktionsfähigkeit der Software zu gewährleisten. Dabei handelt es sich um Black-Box Tests. Durch diesen Ansatz soll eine hohe Qualität und Funktionalität der Software sichergestellt werden. Die

Ergebnisse der Tests werden dokumentiert und dienen als Grundlage für mögliche Anpassungen im Verlauf des Entwicklungsprozesses.

### 3. Analysephase (Sebastian)

#### 3.1. IST-Analyse

Zum Schuljahr 2022/2023 hat die Berufsschule III Regensburg die neue Amtliche Schulverwaltung (ASV) eingeführt und dabei die vorherige Schulverwaltungssoftware Atlantis ersetzt. Die ASV und das dazugehörige Datenbanksystem bieten keine Möglichkeit, Kfz-Kennzeichen von Schülerinnen und Schülern sowie die dazugehörigen Parkausweise zu verwalten. Vor der Einführung der ASV konnten volljährige Schülerinnen und Schüler ihre Fahrzeuge über ein Online-Formular registrieren, wobei die Kennzeichen in einer Datenbank gespeichert wurden. Der Ausweisdruck erfolgte über einen Serienbrief in Microsoft Word. Derzeit wird die Erfassung der Kfz-Kennzeichen mithilfe einer Umfrage auf der Lernplattform Mebis vorgenommen, die dann in eine CSV-Datei exportiert wird, welche als Grundlage für den Serienbrief dient. Die Parkausweise werden jedoch nicht gespeichert, was dazu führt, dass Schüler\*innen ihre Fahrzeuge jedes Schuljahr erneut registrieren müssen, um einen neuen Parkausweis zu erhalten. Die Umfrageerstellung, der Export und die Aufbereitung der CSV-Datei sowie der Druck und die Verteilung der Parkausweise an die Klassenlehrkräfte erfordern viele manuelle Schritte und die Lernplattform Mebis ist nicht für Schulverwaltungsaufgaben konzipiert.

#### 3.2. Kostenanalyse

Im Folgenden sollen die Kosten, die im Laufe des Projektes anfallen, kalkuliert werden. Dabei wurden die anfallenden Personalkosten der Entwickler berücksichtigt. Da die tatsächlichen Personalkosten nicht herausgegeben werden dürfen, wird die Kalkulation anhand von Stundensätzen durchgeführt, die durch die Personalabteilung festgelegt wurden. Die aufgeführten Stundensätze beinhalten zum Großteil das Bruttogehalt sowie die Sozialaufwendungen des Arbeitgebers. Für einen Mitarbeiter wird ein Stundensatz von 70,00 € eingeplant. Der Stundensatz für einen Auszubildenden ist auf 25,00 € festgelegt. Die Entwicklungszeit des Projektes beträgt 33 Stunden. Diese belaufen sich auf 3300,00 €

## 4. Entwurfsphase (Maksim)

### 4.1. Zielplattform

Das Teilprojekt soll als eigenständige Webanwendung laufen. Die Datenbank und ein großer Teil der Datensätze bestehen bereits. In der Laufzeit der Anwendung, sollen die Datensätze ergänzt werden.

Als Programmiersprache des Back-Ends, wird Python mit der Grundlage des Django Frameworks verwendet. Da Bereits Anwendungen und die Datenbank auf Grundlage dieses Frameworks erstellt wurden, macht es das Warten der Anwendung für den Kunden deutlich einfacher. Für das Front-End (Grafische Oberfläche), wird HTML mit JavaScript verwendet, welche mit dem Zusammenspiel von CSS eine benutzerfreundliche und intuitive Oberfläche bietet.

### 4.2. Architekturdesign

Für die Basis der Architektur, wird auf das bekannte und weitverbreitete Schema für das Django Framework zurückgegriffen. Das basiert auf der Model-View-Template (MVT) Architektur. Das erleichtert die Arbeit und Implementierung, da bereits viele Informationen im Internet zur Verfügung gestellt werden.

Aufgebaut wird die Anwendung als erstes mit dem Model, welches auch die Datenbank im Hintergrund darstellt. Dadurch wird es ermöglicht die Daten für die Geschäftslogik zu verwenden. Das Model kümmert sich um die Aktualisierungen und das Abrufen der Daten, welche im Zusammenhang mit der Benutzung der Webanwendung entstehen.

Dabei wird jede Tabelle als eigene Klasse modelliert. Den Spalten werden die Attribute, bzw. die Datentypen zugewiesen, die es ermöglichen die Daten richtig zu manipulieren. Das lässt es auch einfacher nachzuvollziehen und dementsprechend leichter zu implementieren. Für den ganzen Aufbau der Models, werden die bereits Verfügbaren Module verwendet.

Die View kümmert sich um das Rendern des Templates und das Bereitstellen der Daten von den Models. Diese werden an das Template weitergegeben und dementsprechend zur Darstellung verwendet.

Das Template ist das Zusammenspiel aus einem HTML-Dokument und dem dazugehörigen JavaScript- und CSS-Teil. Dort werden die Daten, die von der View weitergegeben werden, so dargestellt und aufbereitet, wie es dem Endnutzer präsentiert werden soll. Eingaben werden an die View geschickt und diese schickt Anforderungen an das Model, welche dann die Daten manipuliert.

Diese Architektur baut auf der, von Django vorgegebenen Struktur auf. Das heißt es werden pro APP (also eine Aufgabe), eine eigene Instanz erstellt, welche die Views, Models und Templates umfassen. Dort wird das Rooting innerhalb der Webanwendung angegeben und verschiedene Views angelegt. Zusätzlich werden die notwendigen Models modelliert

### 4.3. Entwurf der Benutzeroberfläche

Die Oberfläche wurde nach dem Design des Beruflichen Schulzentrum Matthäus Runtinger (BS3) erstellt. Dort werden die Farben und Formen der grafischen Oberfläche nach dem Wiedererkennungswert der BS3 erstellt und angepasst. Hierfür wurden Entwürfe angefertigt (**Anhang A.3: Entwürfe**). Zuerst wurde eine Skizze (**Anhang A.4: Skizze**) erstellt, die die groben Ausprägungen für die spätere Webanwendung darstellen soll. Nachdem das Konzept finalisiert wurde und wir Ideen und weitere Anforderungen zusammenfassen konnten, wurde das Mockup (**Anhang A.5: Mockup**) erstellt. Dieses Mockup wurde vorgestellt und als Vorlage, unter Berücksichtigung von kleineren Änderungen für das Erstellen der Oberfläche der Webanwendung verwendet. Die zu sehenden Elemente, sollten für den Grundaufbau dienen und eine positive Benutzererfahrung hervorbringen.

Für den Prototypen wurden die bereits definierten Elemente in der Skizze und in dem Mockup als Vorlage verwendet. Es wurden, für die Endnutzer der Webanwendung Verbesserungen vorgenommen, die zur Abweichung des Mockups geführt haben. Das Ziel wurde aber eingehalten und weiterverfolgt, die Entwürfe als Modell für das Entwickeln der Webanwendung zu verwenden.

### 4.4. Datenbankmodell

Das OVT wird auf einem bereits bestehenden Datenbankmodell aufgebaut. Es wurden kleine Änderungen an dem bestehenden Datenbankmodell gemacht, damit die Funktionen, die vom Kunden angefordert wurden, auch eingehalten werden können.

Die Tabelle *isv\_parkausweise* (**Anhang A.6: Datenbankmodell**) wurde hinzugefügt. Diese Tabelle ist notwendig, um die Parkausweise einem Schüler und einem KFZ-Kennzeichen zuzuordnen. Informationen ob der Parkausweis bereits gedruckt wurde, für die Benutzerfreundlichkeit, falls ein Schüler einen neuen Ausweis braucht oder neue Kennzeichen erfasst wurden, nicht alle wieder ausgewählt werden müssen werden gespeichert. Die Tabelle steht in einer 1 zu 1 Beziehung zur *isv\_kennzeichen* Tabelle, weil ein Kennzeichen einen Parkausweis benötigt und ein Parkausweis nur für ein Kennzeichen gilt. Des Weiteren besteht eine n zu 1 Beziehung zu der



*isv\_schueler* Tabelle. Ein Parkausweis ist einem Schüler zugeordnet, aber ein Schüler kann mehrere Parkausweise besitzen.

#### 4.5. Maßnahmen zur Qualitätssicherung

Für die Sicherung der Qualität wurde sich voraussichtlich im Team für das Black-Box-Testing und Unit-Testing entschieden.

Durch die Aufteilung der Arbeitspakete untereinander, konnten die Black-Box Tests innerhalb des Teams durchgeführt werden. Dort wurden fertiggestellte Teile dem Team vorgestellt und zur Verfügung gestellt. Im Anschluss konnte sich das Team durch das Ausführen und Anschauen der Änderungen und das Feststellen von Fehlern zur Qualität der Webanwendung beitragen und sichern.

Geplant waren auch Unit-Tests, die für einzelnen Funktionen erstellt werden sollten. Da das Projekt nicht fertiggestellt werden konnte, wurden keine Unit-Tests, durch fehlende Funktionen erstellt.

#### 4.6. Pflichtenheft

Auf der Grundlage des Lastenhefts und den beschriebenen Entwürfen, wurde das Pflichtenheft erstellt. Dieses Pflichtenheft dient als Leitfaden für die Entwicklung und als Vergleich für das Ergebnis der Soll- und Ist-Analyse.

### 5. Implementierungsphase (Marco)

Für die Implementierungsphase des Projekts haben wir zunächst eine virtuelle Umgebung mit Python eingerichtet und das Django-Framework installiert.

Anschließend haben wir ein neues Django-Projekt erstellt und eine App innerhalb des Projekts entworfen.

#### 5.1. Implementierung Python

Wir haben eine virtuelle Umgebung erstellt, um unser Python-Projekt von anderen Projekten zu isolieren und Abhängigkeitskonflikte zu vermeiden. Dazu haben wir das Virtualenv-Modul verwendet, das über den Befehl „py -m venv .venv“ installiert wurde. Diese Umgebung wurde anschließend mit dem Befehl „& ../../.venv/Scripts/Activate.ps1“ aktiviert. Nach der Einrichtung der virtuellen Umgebung haben wir Django installiert, indem wir den Befehl „pip install django“ ausgeführt haben. Mit Django können wir ein Projekt mit dem Befehl „django-admin startproject StaffManagementTool“ erstellen. Hier haben wir unser Projekt mit dem Namen " StaffManagementTool " erstellt. Mit Django können wir eine App innerhalb des Projekts erstellen, die bestimmte Funktionen enthält. Wir haben Apps mit dem Befehl „py manage.py startapp appname“ erstellt. Dabei haben wir die Apps Login, new\_pw, reset\_pw und overview erstellt. In der Implementierungsphase haben wir die Logik und Funktionen der Apps in Python-Code implementiert. Wir haben URLs erstellt,

Views definiert und Templates erstellt, um unsere App funktionsfähig zu machen (**Anhang A7: Viewcode**).

## 5.2. Implementierung der Benutzeroberfläche

Auf Grundlage der unter 4.3 (Entwurf der Benutzeroberfläche) beschriebenen Mockups konnte die Benutzeroberfläche implementiert werden. Umgesetzt wurde die Oberfläche mit der Hypertext Markup Language (HTML), damit die Anwendung im Browser angezeigt werden kann. Um die Einheitlichkeit der einzelnen Seiten zu gewährleisten, ist es in Python-Django möglich, Vorlagen anzulegen und diese dann zu verwenden. Diese Vorgehensweise wurde bei den Apps login, new\_pw und reset\_pw genutzt.

Die Gestaltung der Oberfläche wurde dann mit eingebundenen Cascading Style Sheets (CSS)-Dateien umgesetzt (**Anhang A.8: Css**).

Bei der Gestaltung der Oberfläche wie auch während der gesamten Entwicklung wurden die Softwareergonomie-Richtlinien beachtet. Dazu zählt vor allem die Aufgabenangemessenheit durch eine Minimierung der Interaktionen und geeigneter Funktionalität und die Fehlertoleranz.

## 5.3. Testen der Anwendung

In der Testphase unseres Django-Projekts haben wir verschiedene Tests durchgeführt, um sicherzustellen, dass die Anwendung funktioniert und alle Anforderungen erfüllt sind. Hier sind die Arten von Tests, die wir durchgeführt haben:

### Blackbox-Tests

Bei Blackbox-Tests wird das System als Blackbox betrachtet, d.h. es werden nur die Ein- und Ausgabeparameter der Anwendung betrachtet. Wir haben verschiedene Szenarien entworfen, um die Funktionalität unserer Anwendung zu testen. Zum Beispiel haben wir getestet, ob das Hinzufügen von Daten in die Datenbank und das Abrufen dieser Daten korrekt funktioniert, ob Benutzer sich erfolgreich anmelden und abmelden können und ob Fehlermeldungen korrekt angezeigt werden.

### Unit-Tests

Unit-Tests sind spezifische Tests, die auf bestimmte Einheiten des Codes angewendet werden, z.B. auf einzelne Funktionen oder Methoden. Wir haben verschiedene Unit-Tests geschrieben, um die Funktionen unserer App auf korrekte Funktionalität zu überprüfen. Hier sind einige Beispiele:

Testen der Validierungsfunktionen, um sicherzustellen, dass nur korrekte Daten in die Datenbank eingefügt werden.

Testen der Login-Funktion, um sicherzustellen, dass nur registrierte Benutzer auf bestimmte Funktionen zugreifen können.

Testen der Suchfunktion, um sicherzustellen, dass die Ergebnisse korrekt und vollständig sind.

### Integrationstests

Integrationstests überprüfen, ob verschiedene Einheiten oder Komponenten der Anwendung korrekt zusammenarbeiten. Wir haben Integrationstests durchgeführt, um sicherzustellen, dass die verschiedenen Komponenten unserer App, wie Datenbanken, Front-End und Back-End, korrekt miteinander interagieren.

Wir haben auch automatisierte Tests erstellt, um sicherzustellen, dass die Anwendung kontinuierlich getestet wird, während wir Änderungen am Code vornehmen. Dazu haben wir verschiedene Test-Frameworks wie Django-Test und Python-Unit-Test verwendet.

Zusammenfassend haben wir während der Testphase unserer Django-Anwendung verschiedene Tests durchgeführt, um sicherzustellen, dass sie korrekt funktioniert und alle Anforderungen erfüllt. Durch die Verwendung von Blackbox-Tests, Unit-Tests und Integrationstests konnten wir Fehler schnell identifizieren und korrigieren, was zur Entwicklung einer stabilen und zuverlässigen Anwendung beigetragen hat.

## 6. Dokumentationsphase (Jonas)

Die Dokumentation der Anwendung beinhaltet neben der Projektdokumentation, in der die einzelnen Phasen des Projektes beschrieben werden, auch eine Benutzer- und Entwicklerdokumentation.

Die Benutzerdokumentation soll dem Anwender helfen, sich in der Anwendung zurechtzufinden. Diese macht Angaben zur Funktionsweise und Navigation der Anwendung. Anhand von Screenshots wird so die gedachte Benutzung erklärt. Ein Ausschnitt befindet sich im **Anhang A.23**.

Aufgrund Zeitlichem Mangels besteht die Entwicklerdokumentation lediglich aus dem Inhalt einer Readme zum Navigieren der Entwicklerumgebung, die recht hilfreich für neue Benutzer sein kann, jedoch keine ist detaillierte Beschreibung zu Methoden, Variablen oder ähnlichem liefert, da diese noch nicht implementiert sind.

Ein Screenshot der erstellten Entwicklerdokumentation lässt sich im **Anhang A.24** finden.

## 7. Fazit (Jonas)

### 7.1. SOLL-/IST-Analyse

In einem Rückblick soll die Planung des Projektes abschließend überprüft werden. Dazu wird im Vorfeld geschätzte Zeit mit der tatsächlich benötigten verglichen. Die Zeitplanung konnte nicht eingehalten werden, da die sehr begrenzte Zeit dem enormen Umfang des Projektes nicht angepasst war. Es fehlt dementsprechend ein Großteil der Implementierungsphase.

Projektphase	Geplante Zeit	Tatsächliche Zeit	Differenz
Planungsphase	7 h	7h	-
Entwurfsphase	4 h	6h	+2h
Implementierungsphase	33 h	33h	-
Dokumentationsphase	6 h	4h	-2h
Gesamt	50 h	50h	0h

Tabelle 2.: Soll-/Ist-Analyse

Die Entwurfsphase fiel um zwei Stunde länger aus, da sehr lange auf benötigte Datenbanken gewartet werden musste.

In der Implementierungsphase würde deutlich mehr Zeit benötigt werden, um die Funktionalität des Projektes zu implementieren.

### 7.2. Lesson Learned

Anhand der Umsetzung dieses Abschlussprojektes konnten die Autoren wertvolle Erfahrungen über die Arbeit an einem Projekt mit allen zugehörigen Arbeitsschritten sammeln. Vor allem in der Planung und Entwurfsphase. Auch in Bezug auf die Programmierung konnten mit dem Konzept Full-Stack und dem Framework Django wichtige Grundlagen im Bereich Web Development vermittelt werden. Zudem konnten die Autoren Erfahrung im Bereich Datenbank Anbindung sammeln. Jedoch ist zu notieren, dass der Rahmen des Projektes weit über die verfügbare Zeit hinausging und leider vieles nicht zu Ende geführt werden konnte.

### 7.3. Ausblick

Auch wenn nicht alle an dieses Projekt gestellte Anforderungen umgesetzt werden konnten, wurde von den Autoren reichlich Erfahrung gesammelt, die in ähnlichen Projekten und Aufgaben zukünftig einfacher und effizienter eingesetzt werden kann. Es wurde ein stabiles Gerüst anhand von Views und Rooting für Weiterentwicklung und Implementierung der Funktionen geschaffen.

## 8. Anhang

### A.1 Detaillierter Zeitplan

### A.2 Verwendete Ressourcen

#### Hardware:

- Laptops

#### Software:

- VS-Code als IDE

- XAMPP zur Simulation der Datenbank

- Git

- PostgreSQL – Datenbanksystem

- Windows 10 – Betriebssystem

#### Personal:

- Auszubildende – Umsetzung des Projekts

- Lehrkräfte – Fachbetreuung

## A.3 Entwürfe

Logo BSZ Logo Stadt

Login

Email

Password

Passwort vergessen?

Logo BSZ Logo Stadt

Passwort zurücksetzen



e-mail



Logo BSZ Logo Stadt

Neues PW



#### A.4 Skizze



Logo BSZ Logo Stadt




ITAE12D | ▾ +  

<input type="checkbox"/>		S-ID	Vorname	Nachname	Klasse	Kennzeichen	Gesperrt	Verwarnungen
<input type="checkbox"/>		245	Marco	Pfab	ITAE12d	R-MP-187	Nein	0

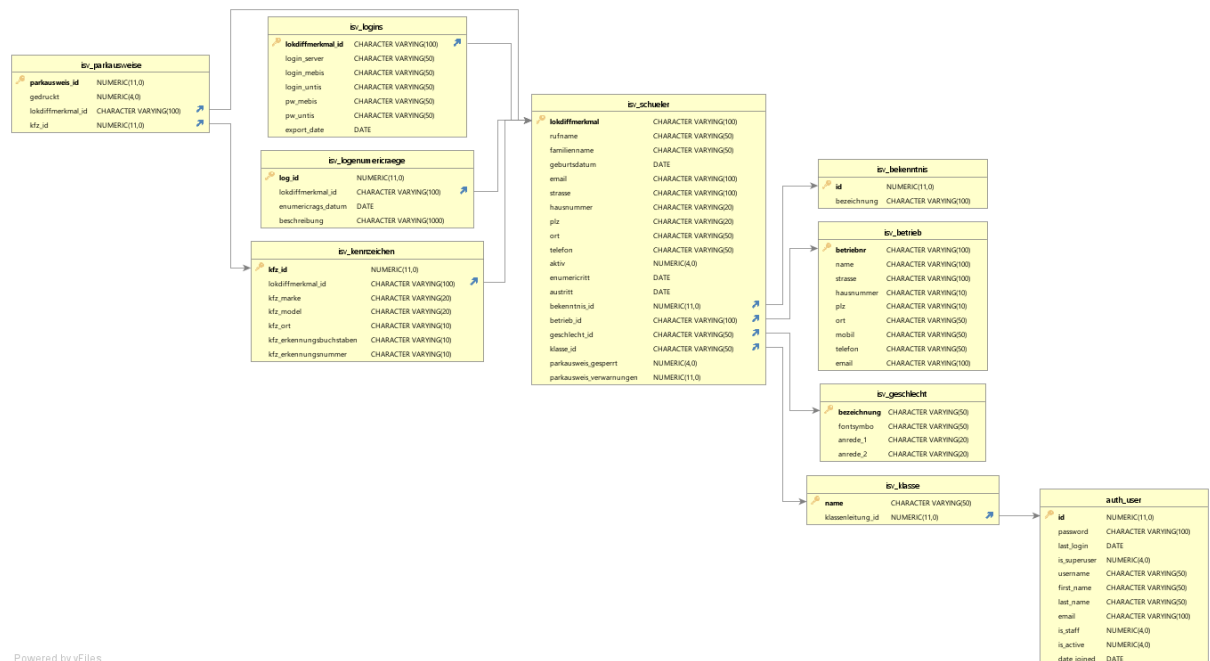
#### A.5 Mockup

 BSZ Matthäus  
Runtinger 

ITAE12D | ▾ +  

<input type="checkbox"/>		S-ID	Vorname	Nachname	Klasse	Kennzeichen	Gesperrt?	Verwarnungen	Notizen
<input type="checkbox"/>		00345	Maksim	Hermann	ITAE12D	HF-MH-123	Nein	2	
<input type="checkbox"/>		00467	Michael	Hoelzer	ITAE12D	R-H-345	Nein	0	

## A.6 Datenbankmodell



## A.7 Viewcode

```

urls.py
01_repo > StaffManagementTool > overview > urls.py > ...
1  from django.contrib import admin
2  from django.urls import path
3
4  # importing views from views..py
5  from .views import overview_view
6
7  urlpatterns = [
8      path('', overview_view, name='home'),
9  ]

```

```

views.py
01_repo > StaffManagementTool > overview > views.py > overview_view
1  from django.shortcuts import render
2
3  # Create your views here.
4  def overview_view(request):
5      return render(request, 'overview.html')

```



## A.8 Css

```
01_repo > StaffManagementTool > login > static > # styles.css > .anmeldeprozess h1
1  /* Standardschriftart */
2  body {
3    font-family: 'Open Sans', sans-serif;
4  }
5
6  /* Header */
7  header {
8    position: fixed; /* fixiere die Position des Headers */
9    top: 5px;        /* setze den Header an die obere Kante des Viewports */
10   left: 0;          /* setze den Header an die linke Kante des Viewports */
11   width: 100%;       /* setze die Breite des Headers auf 100% des Viewports */
12   background-color: #dd0000;
13   display: flex;
14   justify-content: center;
15 }
16
17 .logo {
18   max-width: 100%;
19   height: auto;
20 }
21
22 .logo-item-schule {
23   position: absolute;
24   left: 50%;
25   transform: translateX(-300px);
26 }
27
28 .logo-item-stadt{
29   position: absolute;
30   left: 50%;
31   transform: translateX(-100%) translateX(300px);
32 }
33
34 /* Formular */
35 .anmeldeprozess {
36   max-width: 600px;
37   border-color: #ccc;
38   border-width: 1px;
39   border-style: solid;
40   margin: 85px auto 0;
41   display: flex;
42   flex-direction: column;
43   align-items: flex-start;
44   background-color: #fff;
45   padding: 20px;
46   box-shadow: 0px 0px 5px 0px rgba(0,0,0,0.25);
47 }
48
49 .anmeldeprozess h1 {
50   font-size: 24px;
51   font-weight: bold;
52   margin: 0 0 5px;
53   text-align: center; /* Zentriere die Überschrift horizontal */
54 }
```

## A.7 Entwicklerdokumentation

----- READ ME for Web App (Staff Management Tool) -----

----- Requirements: -----

VSCode Download: <https://code.visualstudio.com/download>

Python 3.11: <https://www.python.org/ftp/python/3.11.0/python-3.11.0-amd64.exe>

-On Windows, make sure the location of your Python interpreter is included in your PATH environment variable. You can check the location by running path at the command prompt. If the Python interpreter's folder isn't included, open Windows Settings, search for "environment", select Edit environment variables for your account, then edit the Path variable to include that folder

NPM: <https://nodejs.org/dist/v18.12.1/node-v18.12.1-x64.msi>

Github bash download: <https://github.com/git-for-windows/git/releases/download/v2.38.1.windows.1/Git-2.38.1-64-bit.exe>

Github account creation: <https://github.com/join>

Needed Extensions:

Name: HTML CSS Support Id: ecmel.vscod.html-css Description: CSS Intellisense for HTML Version: 1.13.1 Publisher: ecmel VS Marketplace Link: <https://marketplace.visualstudio.com/items?itemName=ecmel.vscod.html-css>

Name: Python Id: ms-python.python Description: IntelliSense (Pylance), Linting, Debugging (multi-threaded, remote), Jupyter Notebooks, code formatting, refactoring, unit tests, and more. Version: 2022.18.2 Publisher: Microsoft VS Marketplace Link: <https://marketplace.visualstudio.com/items?itemName=ms-python.python>

Name: Djaneiro - Django Snippets Id: thebarkman.vscod-djaneiro Description: A collection of snippets for django templates, models, views, fields & forms. Ported from Djaneiro for SublimeText Version: 1.4.2 Publisher: Scott Barkman VS Marketplace Link: <https://marketplace.visualstudio.com/items?itemName=thebarkman.vscod-djaneiro>

....

Needed input:

- Data structure from group 3

Output to:

- Controllgroup QR-Code group 1

----- Creating a Venv: -----

1.On your file system, create a project folder for this tutorial, such as hello\_django.

2.In that folder create a folder called ".gitignore", use the following command to create a virtual environment named venv based on your current interpreter:

```
# Linux
sudo apt-get install python3-venv # If needed
python3 -m venv .venv
source .venv/bin/activate

# macOS
python3 -m venv .venv
source .venv/bin/activate

# Windows
py -m venv .gitignore\venv

In powershell use:

set-executionpolicy remotesigned, followed by J

to activate the scripts use '<path of python.exe>'
```

Interpreter auswählen & "c:/Users/q506869/OneDrive - BMW Group/Dokumente/07\_Programmieren/02\_Python/Django/.venv/Scripts/Activate.ps1"

3.Open the project folder in VS Code by running code .

4.In VS Code, open the Command Palette (View > Command Palette or (Ctrl+Shift+P)). Then select the Python: Select Interpreter command:

5.Select the virtual environment in your project folder that starts with ./venv or ..venv

6.Create New Terminal (Ctrl+Shift+)

7.Update pip: python -m pip install --upgrade pip

8.Install django: python -m pip install django

----- Using Django: -----

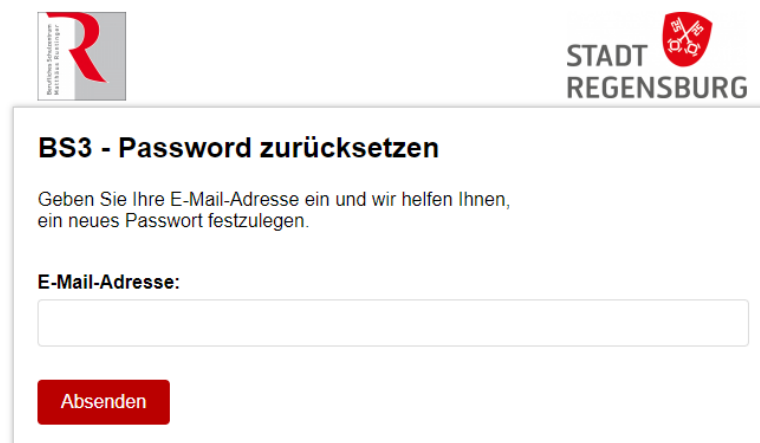
<https://code.visualstudio.com/docs/python/tutorial-django>

## A.8 Benutzerdokumentation (Auszug)



The screenshot shows the login interface for the BS3 system. At the top left is a logo with a red 'R' and the text 'Berufliches Schulzentrum Regensburg'. At the top right is the 'STADT REGENSBURG' logo. The main heading is 'BS3 - Login'. Below it, a message says 'Bitte melden Sie sich mit Ihrer E-Mail-Adresse und Passwort an.' There are two input fields: 'E-Mail-Adresse:' and 'Passwort:'. Below the password field is a link 'Passwort vergessen?'. At the bottom is a red 'Login' button.

**Login-Maske**, nach Aufruf der Webapplikation. Nach erfolgreicher Eingabe der E-Mail-Adresse im ersten Eingabefeld (1) und Passwort im zweiten Eingabefeld (2) und anschließendem Bestätigen mit dem Login Button (3) wird man auf die Übersichtsseite weitergeleitet. Mit einem Mausklick auf den grauen Text „Passwort vergessen?“ (4) wird der Benutzer auf eine neue Seite geleitet, um dieses zurückzusetzen.



The screenshot shows the password reset interface for the BS3 system. At the top left is the same logo with a red 'R'. At the top right is the 'STADT REGENSBURG' logo. The main heading is 'BS3 - Password zurücksetzen'. Below it, a message says 'Geben Sie Ihre E-Mail-Adresse ein und wir helfen Ihnen, ein neues Passwort festzulegen.' There is one input field: 'E-Mail-Adresse:'. At the bottom is a red 'Absenden' button.

Passwort zurücksetzen Maske, um ein neues Passwort festzulegen wird zunächst die E-Mail-Adresse des Benutzers gefordert. Nach Eingabe dieser E-Mail-Adresse im Eingabefeld (5) muss der rote „Absenden“ Button betätigt werden. Anschließend wird eine E-Mail mit weiteren Anleitungen zum Festlegen des neuen Passworts an diese geschickt. Dieser Prozess kann einige Minuten dauern.