

Hochschule Bremen
City University of Applied Sciences



Fallstudie Bürgerregister Light

Bilal Özsoy, Felix Hinrichs, Jan Plachetka, Nick Eilers

Agenda

1. Fallstudie
2. Use Case Diagramm
3. Klassendiagramm
4. Sequenzdiagramm
5. Ordnerstruktur
6. Code Vorstellung
7. Demo



Erklärung über das eigenständige Erstellen der Arbeit

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Die Stellen der Arbeit, die anderen Werken dem Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht.

Diese Erklärung erstreckt sich auch auf die in der Arbeit enthaltenen Grafiken, Skizzen sowie bildlichen Darstellungen. Die Arbeit habe ich in gleicher oder ähnlicher Form auch auszugsweise noch nicht als Bestandteil einer Prüfungs- oder Studienleistung vorgelegt.

Ich versichere, dass die eingereichte elektronische Version der Arbeit vollständig mit der Druckversion übereinstimmt und stimme einer elektronischen Überprüfung der Arbeit mittels Plagiatssoftware zu.

Bitte ankreuzen: Eine der zwei Optionen ist in Absprache zwischen Prüfenden und Geprüften verbindlich auszuwählen.

Option 1: Verwendung von KI-basierten Tools ohne Kennzeichnungspflicht

Ich versichere, dass ich keine KI-basierten Tools verwendet habe, deren Nutzung mit der prüfenden Person nicht schriftlich verabredet wurde. Ich bin mir bewusst, dass die Verwendung von Texten oder anderen Inhalten und Produkten, die durch KI-basierte Tools generiert wurden, keine Garantie für deren Qualität darstellt. Ich übernehme die Verantwortung für die abgegebene Arbeit und die darin enthaltenen Inhalte. Ich versichere zudem, dass in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt.

Option 2: Verwendung von KI-basierten Tools mit Kennzeichnungspflicht

Ich versichere, dass ich keine KI-basierten Tools verwendet habe, deren Nutzung mit der prüfenden Person nicht schriftlich verabredet wurde. Ich bin mir bewusst, dass die Verwendung von Texten oder anderen Inhalten und Produkten, die durch KI-basierte Tools generiert wurden, keine Garantie für deren Qualität darstellt. Ich übernehme die Verantwortung für die abgegebene Arbeit und die darin enthaltenen Inhalte. Ich versichere zudem, dass in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Sämtliche wörtlichen oder sinngemäßen Übernahmen und Zitate, sowie alle Abschnitte, die mithilfe von KI-basierten Tools entworfen, verfasst und/oder bearbeitet wurden, sind von mir kenntlich gemacht und nachgewiesen. Die Form der Kennzeichnung wird zwischen der prüfenden Person und Prüfling abgestimmt.

Mir ist bekannt, dass ein Verstoß gegen die genannten Punkte prüfungsrechtliche Konsequenzen haben und insbesondere dazu führen kann, dass die Prüfungsleistung mit „nicht ausreichend“ bzw. die Studienleistung mit „nicht bestanden“ bewertet wird und bei mehrfachem oder schwerwiegendem Täuschungsversuch eine Exmatrikulation erfolgen kann.

Vor- und Nachname **Felix Hinrichs**

Matrikelnummer **5383630**

Bremen, den **05.12.2024**

Unterschrift 

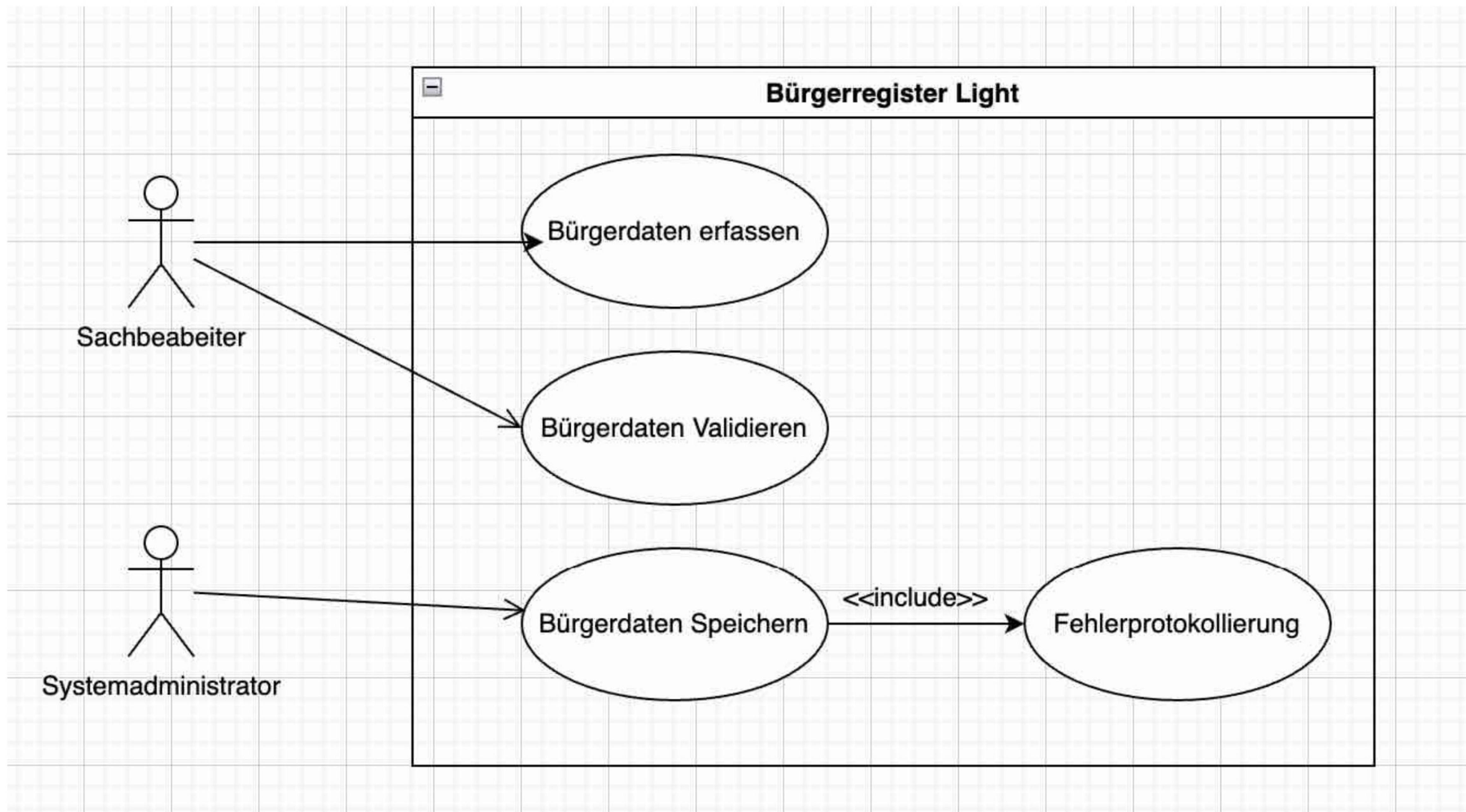
Hochschule Bremen
Abgestimmt in der Studiendekanerunde vom 29. Oktober 2024

Ausgangslage

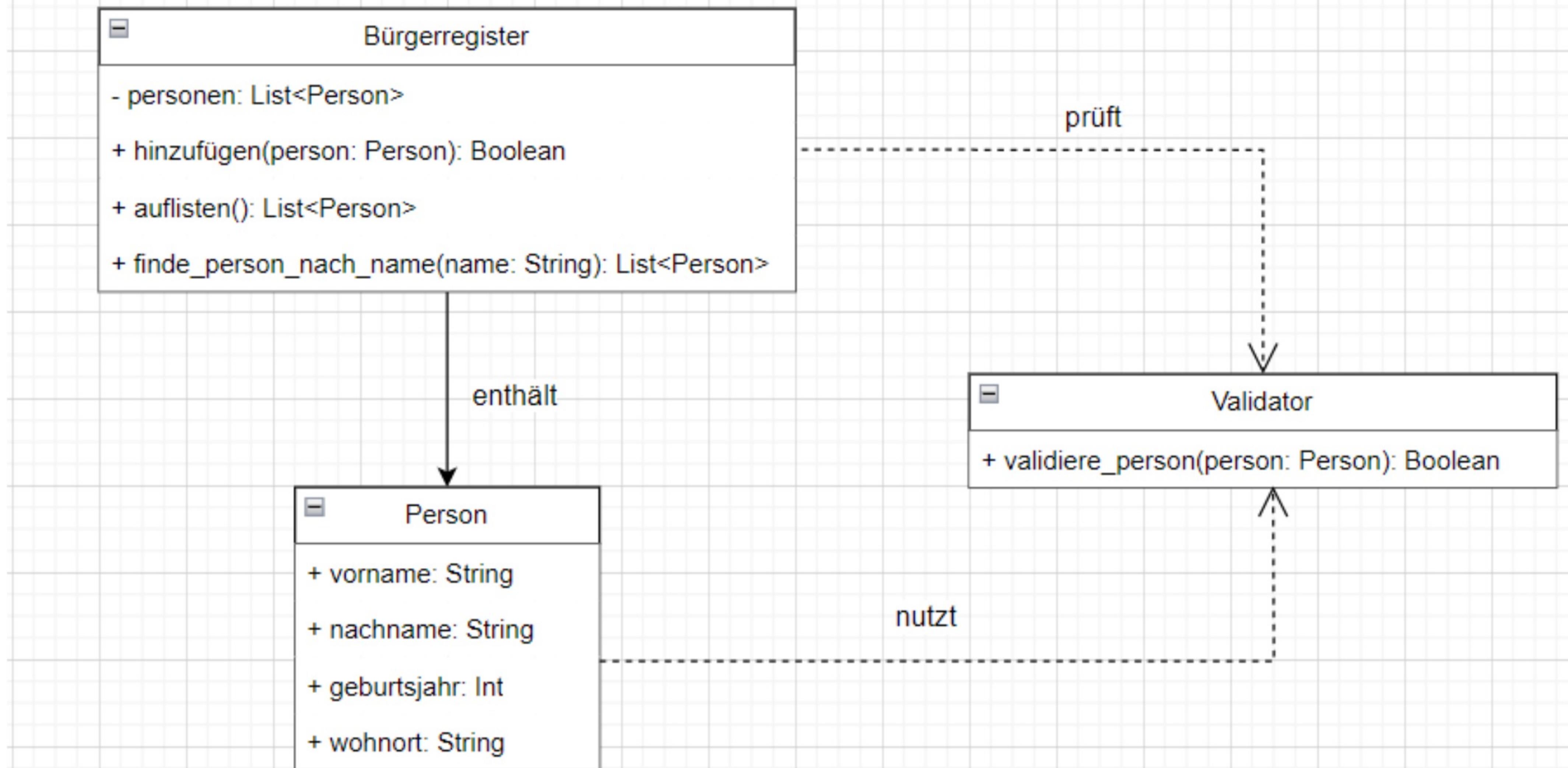
- Kommunen und Verwaltungen stehen vor der Aufgabe, ihre Prozesse zu digitalisieren; besonders zentral ist dabei die korrekte Erfassung und Verwaltung von Bürgerdaten wie Melderegister, Anträgen, Ausweisen und Wahllisten.
- Viele kleinere Gemeinden arbeiten noch mit veralteten Desktopprogrammen oder Excel-Lösungen, die weder benutzerfreundlich noch sicher oder skalierbar sind und somit den modernen Anforderungen nicht gerecht werden.
- Für das Modul Software Engineering II entsteht daher ein vereinfachtes, realitätsnahe Fallbeispiel namens „Bürgerregister Light“, das diese Problemlage aufgreift.

Zielsetzung des Projekts

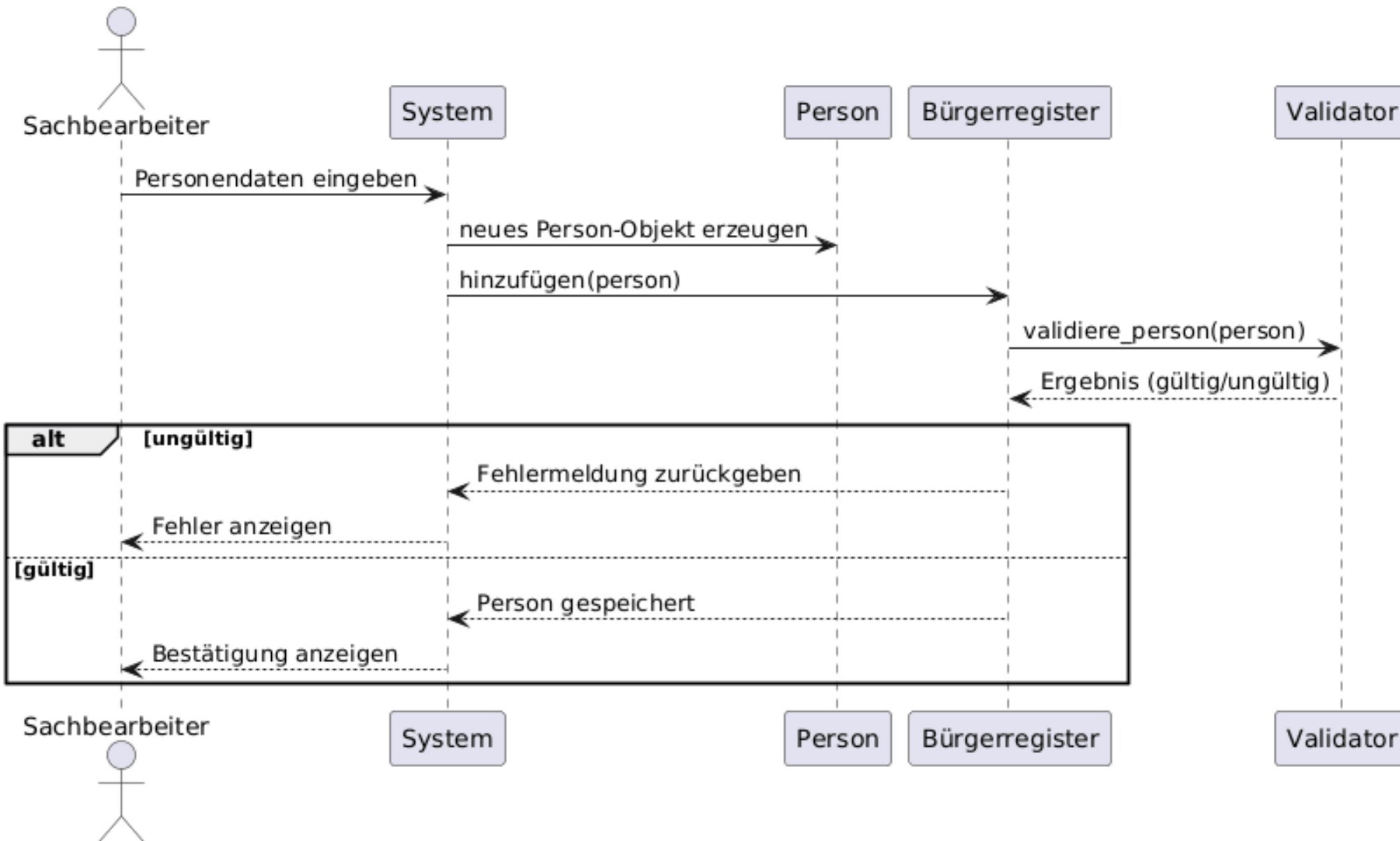
- Das „Bürgerregister Light“ soll eine moderne, vereinfachte Verwaltungsanwendung darstellen, die die grundlegenden Funktionen eines Bürgerregisters abbildet und gleichzeitig als didaktische Übungsplattform dient.
- Im Projekt wird der komplette Softwareentwicklungsprozess exemplarisch durchlaufen – von Anforderungsanalyse, Modellierung und Entwurf bis hin zu Implementierung und Test.
- Ziel ist nicht die Entwicklung eines produktiven Systems, sondern eine verständliche, funktionale Softwarearchitektur mit klaren Zuständigkeiten und einem transparenten Datenfluss.

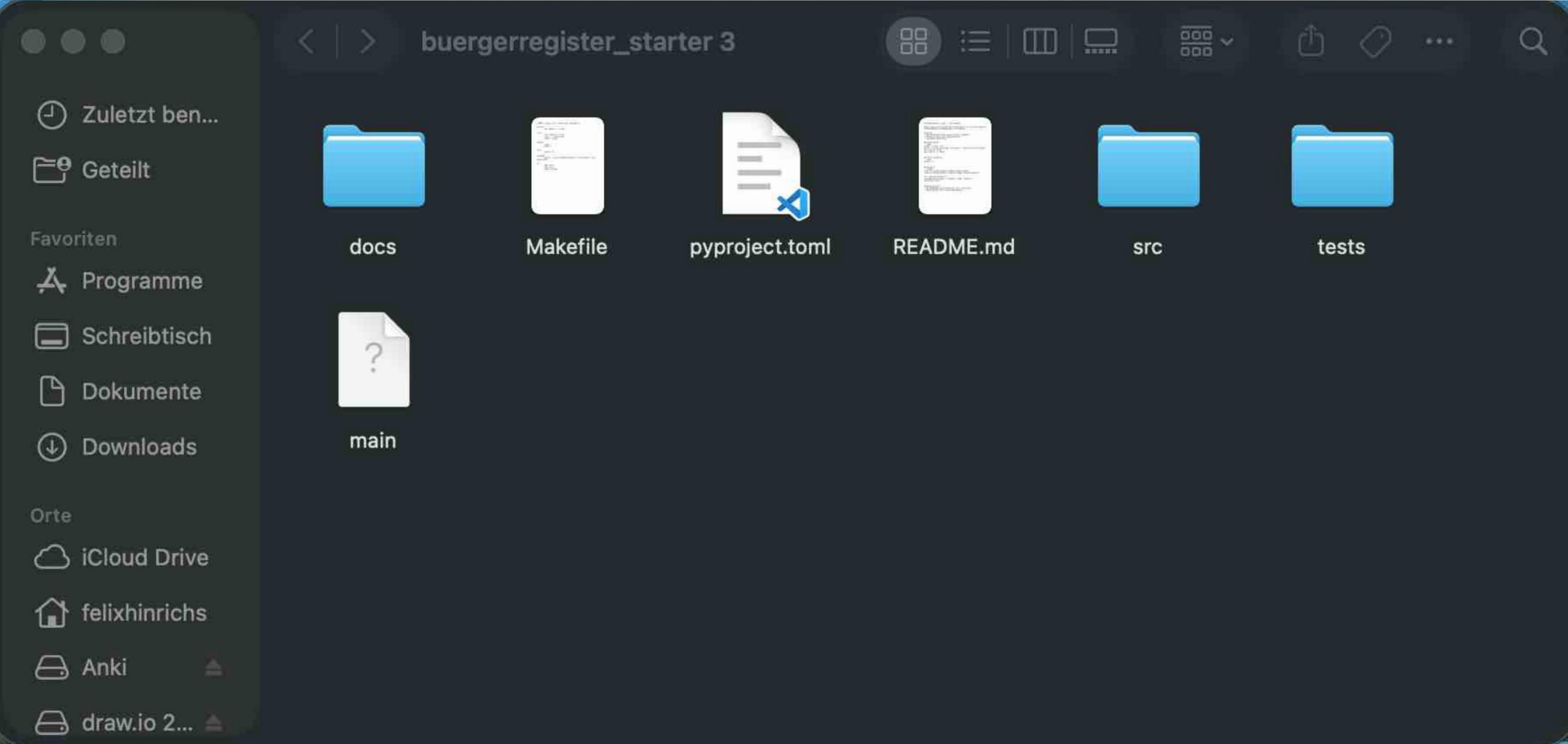


Klassendiagramm - Bürgerregister Light



Sequenzdiagramm - Person anlegen mit Validierung





__init__.py X

Users > felixhinrichs > Desktop > buergerregister_starter 3 > src > buergerregister > __init__.py > ...

```
1 from .models import Person
2 from .register import Buergerregister
3 from .validation import validiere_person
4
5 __all__ = ['Person', 'Buergerregister', 'validiere_person']
6
```

models.py ×

Users > felixhinrichs > Desktop > buergerregister_starter_3 > src > buergerregister > models.py > ...

```
1 from dataclasses import dataclass
2
3 @dataclass
4 class Person:
5     """
6     Repräsentiert eine Bürgerin bzw. einen Bürger als einfaches Datenobjekt.
7     """
8     vorname: str
9     nachname: str
10    geburtsjahr: int
11    wohnort: str
12
13    # Hinweis: Hier wurde bewusst auf __post_init__ Validierung verzichtet,
14    # damit die Logik zentral in validation.py gesteuert werden kann.
15
```

validation.py X

Users > felixhinrichs > Desktop > buergerregister_starter 3 > src > buergerregister > validation.py > validiere_person

```

1  from typing import Any, Dict, List, Tuple
2
3  def validiere_person(daten: Dict[str, Any]) -> Tuple[bool, List[str]]:
4      """
5          Prüft ein Personen-Dictionary auf Pflichtfelder und Plausibilität.
6
7      Args:
8          daten (dict): Ein Dictionary mit den Schlüsseln 'vorname', 'nachname',
9                          'geburtsjahr' und 'wohnort'.
10
11     Returns:
12         tuple: (ist_valide (bool), fehlerliste (list[str]))
13         """
14     errors: List[str] = []
15
16     # Prüfung auf leere Strings (Pflichtfelder)
17     if not daten.get("vorname"):
18         errors.append("Der Vorname darf nicht leer sein.")
19
20     if not daten.get("nachname"):
21         errors.append("Der Nachname darf nicht leer sein.")
22
23     if not daten.get("wohnort"):
24         errors.append("Der Wohnort darf nicht leer sein.")
25
26     # Plausibilitätsprüfung Geburtsjahr
27     jahr = daten.get("geburtsjahr")
28     if not isinstance(jahr, int):
29         errors.append("Das Geburtsjahr muss eine Zahl sein.")
30     elif not (1900 <= jahr <= 2025):
31         errors.append(f"Das Geburtsjahr {jahr} ist unplausibel (muss zwischen 1900 und 2025 liegen).")
32
33     # Ergebnis zurückgeben: True wenn keine Fehler da sind
34     return (len(errors) == 0, errors)

```

- Validiert die Personen
- Prüft auf leere Eingaben
- Prüft das Geburtsjahr auf Plausibilität

```

register.py ✘

Users > felixhinrichs > Desktop > buergerregister_starter_3 > src > buergerregister > register.py > ...
1  from typing import List, Tuple
2  from .models import Person
3  from .validation import validiere_person
4
5  class Buergerregister:
6      """
7          Verwaltet eine Liste von Personen im Arbeitsspeicher (In-Memory).
8      """
9
10     def __init__(self):
11         # Interne Liste zur Speicherung der Person-Objekte
12         self._personen: List[Person] = []
13
14     def add(self, person: Person) -> Tuple[bool, str]:
15         """
16             Fügt eine Person hinzu, sofern sie valide ist und noch nicht existiert.
17
18             Returns:
19                 Tuple[bool, str]: (Erfolg, Nachricht/Fehlergrund)
20         """
21
22         # 1. Validierung aufrufen (Umwandlung in Dict für den Validator)
23         ok, errors = validiere_person(person.__dict__)
24         if not ok:
25             # Fehlerliste zu einem String zusammenbauen
26             return False, "Validierungsfehler: " + ", ".join(errors)
27
28         # 2. Duplikatprüfung (Vorname + Nachname)
29         # Wir nutzen einen Generator-Ausdruck für Effizienz
30         if any(p.nachname == person.nachname and p.vorname == person.vorname for p in self._personen):
31             return False, "Warnung: Diese Person existiert bereits (Duplikat)."
32
33         # 3. Speichern
34         self._personen.append(person)
35         return True, "Person erfolgreich hinzugefügt."
36
37     def list(self) -> List[Person]:
38         """Gibt eine Kopie der aktuellen Personenliste zurück."""
39         return list(self._personen)
40
41     def find(self, nachname: str) -> List[Person]:
42         """Sucht Personen anhand des Nachnamens (Groß-/Kleinschreibung wird beachtet)."""
43         # List Comprehension für Filterung
44         return [p for p in self._personen if p.nachname == nachname]

```

Funktion des Registers:

- Validierung
- Duplikatenprüfung
- Speicherung
- Auflistung

```

  main 4 X
Users > felixhinrichs > Desktop > buergerregister_starter 3 > main > ...
1  ~ from src.buergerregister.models import Person
2  from src.buergerregister.register import Buergerregister
3
4  def print_menu():
5      print("\n--- Bürgerregister Light ---")
6      print("1. Neue Person anlegen")
7      print("2. Alle Personen auflisten")
8      print("3. Person suchen")
9      print("4. Beenden")
10
11 def main():
12     register = Buergerregister()
13
14     while True:
15         print_menu()
16         wahl = input("Ihre Wahl: ")
17
18         if wahl == "1":
19             print("\n[Person anlegen]")
20             try:
21                 v_name = input("Vorname: ").strip()
22                 n_name = input("Nachname: ").strip()
23                 ort = input("Wohnort: ").strip()
24                 # Eingabe sicher in Zahl umwandeln
25                 jahr_str = input("Geburtsjahr: ").strip()
26                 jahr = int(jahr_str) if jahr_str.isdigit() else 0
27
28                 # Person-Objekt erstellen
29                 neue_person = Person(
30                     vorname=v_name,
31                     nachname=n_name,
32                     geburtsjahr=jahr,
33                     wohnort=ort
34                 )
35
36                 # Logik aufrufen (End-to-End: Validierung & Speicherung passiert im Register)
37                 erfolg, meldung = register.add(neue_person)
38
39

```

```

  main 4 X
Users > felixhinrichs > Desktop > buergerregister_starter 3 > main > ...
12  def main():
13      ~
14      if erfolg:
15          print(f"✓ {meldung}")
16      else:
17          print(f"✗ {meldung}")
18
19      except Exception as e:
20          print(f"Fehler bei der Eingabe: {e}")
21
22      elif wahl == "2":
23          print("\n[Liste aller Personen]")
24          personen = register.list()
25          if not personen:
26              print("Das Register ist leer.")
27          else:
28              print(f"{len(personen)} Personen gefunden")
29              print("-" * 50)
30              for p in personen:
31                  print(f"{p.nachname}<15} {p.vorname}<15} {p.wohnort}<15} {p.geburtsjahr}<15")
32
33      elif wahl == "3":
34          such_name = input("Nachname gesucht: ").strip()
35          treffer = register.find(such_name)
36          print(f"{len(treffer)} Treffer gefunden.")
37          for p in treffer:
38              print(f"--> {p.vorname} {p.nachname} aus {p.wohnort}")
39
40      elif wahl == "4":
41          print("Programm beendet.")
42          sys.exit()
43
44      else:
45          print("Ungültige Eingabe.")
46
47      if __name__ == "__main__":
48          main()

```

PROBLEMS 11 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
/usr/bin/python3 "/Users/felixhinrichs/Desktop/buergerregister_starter 3/main"  
○ (base) felixhinrichs@MacBook-Air-von-Felix-4 ~ % /usr/bin/python3 "/Users/felixhinrichs/Desktop/buergerregister_starter 3/main"
```

--- Bürgerregister Light ---

1. Neue Person anlegen
2. Alle Personen auflisten
3. Person suchen
4. Beenden

Ihre Wahl:



```
/usr/bin/python3 "/Users/felixhinrichs/Desktop/buergerregister_starter 3/main"  
○ (base) felixhinrichs@MacBook-Air-von-Felix-4 ~ % /usr/bin/python3 "/Users/felixhinrichs/Desktop/buergerregister_starter 3/main"
```

--- Bürgerregister Light ---

1. Neue Person anlegen
2. Alle Personen auflisten
3. Person suchen
4. Beenden

Ihre Wahl: 2

[Liste aller Personen]
Das Register ist leer.

Fazit (Was haben wir aus der Übung mitgenommen?)



Vielen Dank für Ihre Aufmerksamkeit!

Bilal Özsoy, Felix Hinrichs, Jan Plachetka, Nick Eilers