

Abschlusspräsentation SWP-Gruppe 01 (3m5.) SkillMatrix

Aron Gaden, Cornell Ziepel, Bruno Reinhold,
Maximilian Schaller, Wilhelm Pertsch

Agenda

- Aufgabenstellung
- Backend
 - Datenbank
 - Klassendiagramm
 - Authentifizierung
- Frontend
 - React
 - Redux
- Demo
- Gesprächsrunde

Aufgabenstellung

- Webbasierte Anwendung zur Selbsteinschätzung ("Skillmatrix")
 - Anmeldung/Login bei Active Directory
 - Fähigkeiten bewerten
 - Fähigkeiten suchen
 - Fähigkeiten bearbeiten
 - grafische Mitarbeiterentwicklung
 - Interface: Material Design
 - Frontend: React
 - Datenbank: MySQL

Aufgabenstellung - Muss-Kriterien

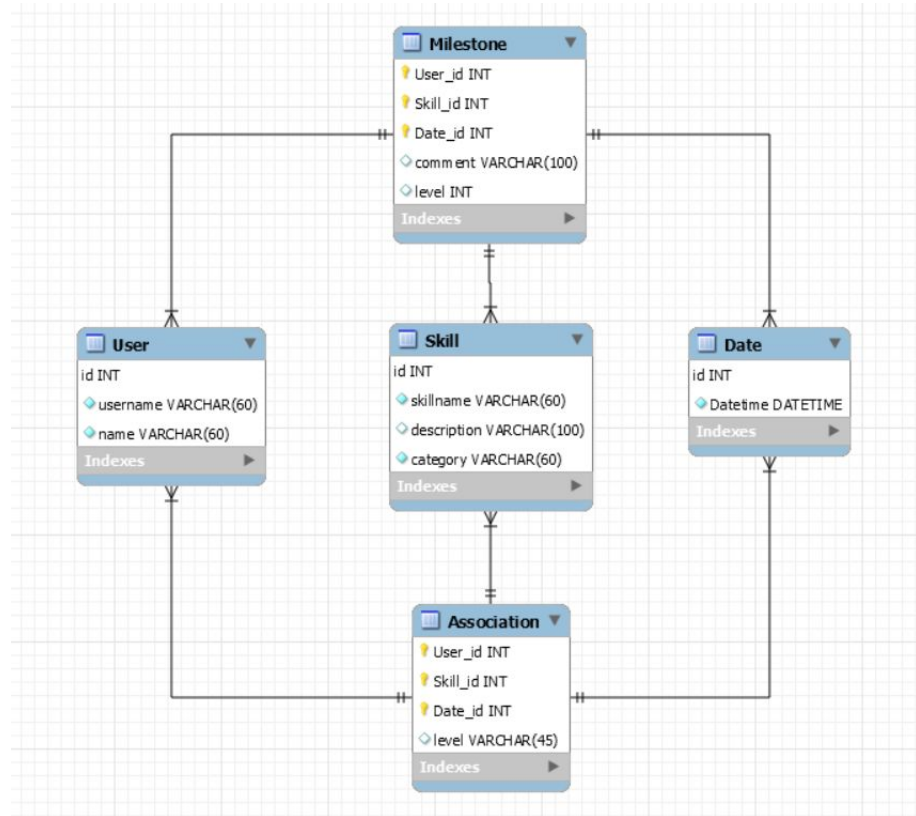
- Authentifizierung der Mitarbeiter über Active Directory
- Hinzufügen und Bearbeiten eigener Skills auf Skala von 1-5 (auch mehrer gleichzeitig)
- Entwicklungsstatistik mit einstellbarem Zeitraum
- Speicherung von Profilen in MySQL-Datenbank
- Unscharfe Suche von Mitarbeitern nach Kriterien (Programmiersprachen, Frameworks, etc.)
- Trennung von Mitarbeitern mit allen gesuchten und Fähigkeiten von Mitarbeitern mit einem Teil dieser
- Hinzufügen von Guidelines für Bewertungsniveau ermöglichen
- zusätzlich: Hierarchie von Skills (Unterkategorien ermöglichen)

Backend - Datenbank

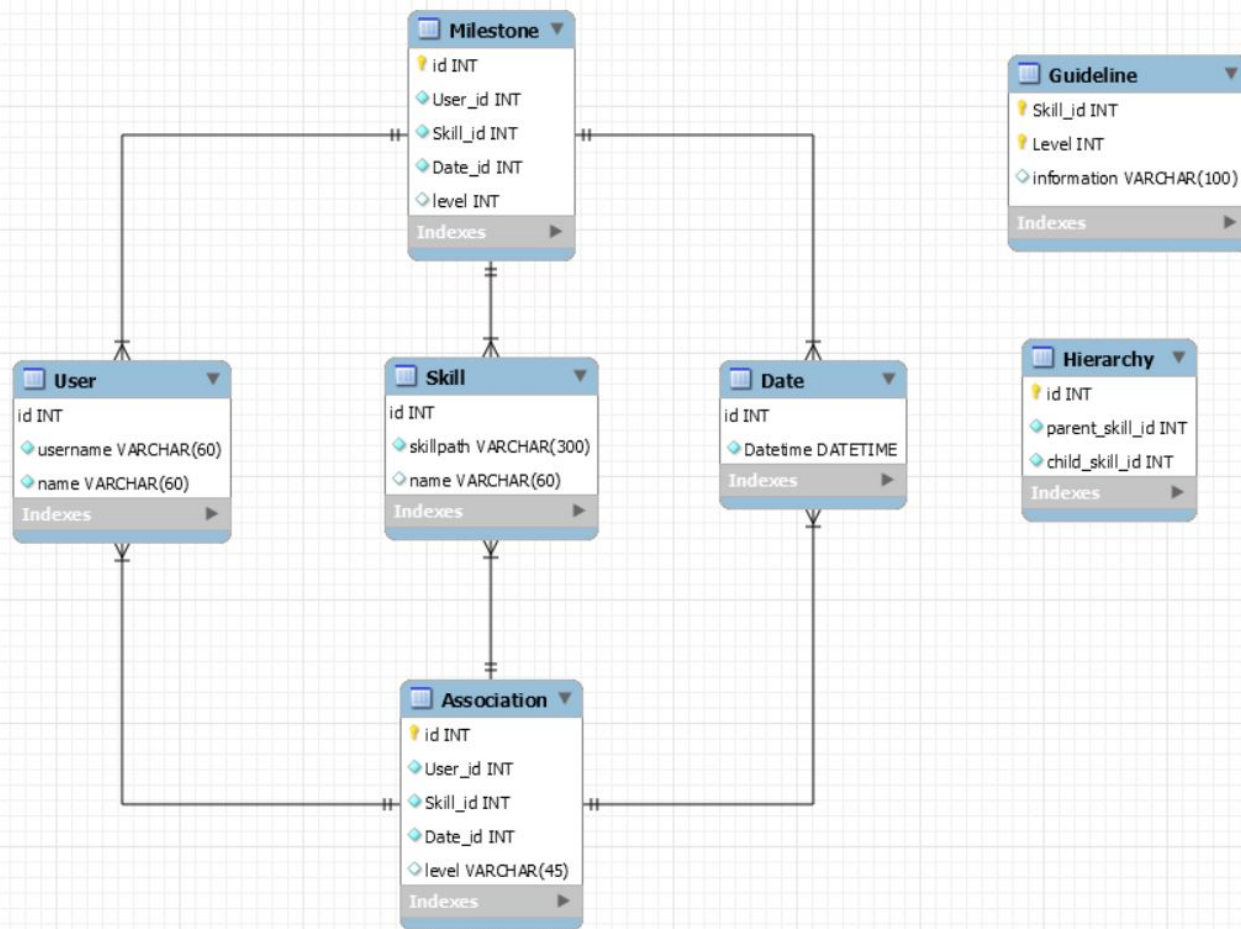
Genutzte Technologien:

- MySQL
- Flask-SQLAlchemy
- SQLAlchemy

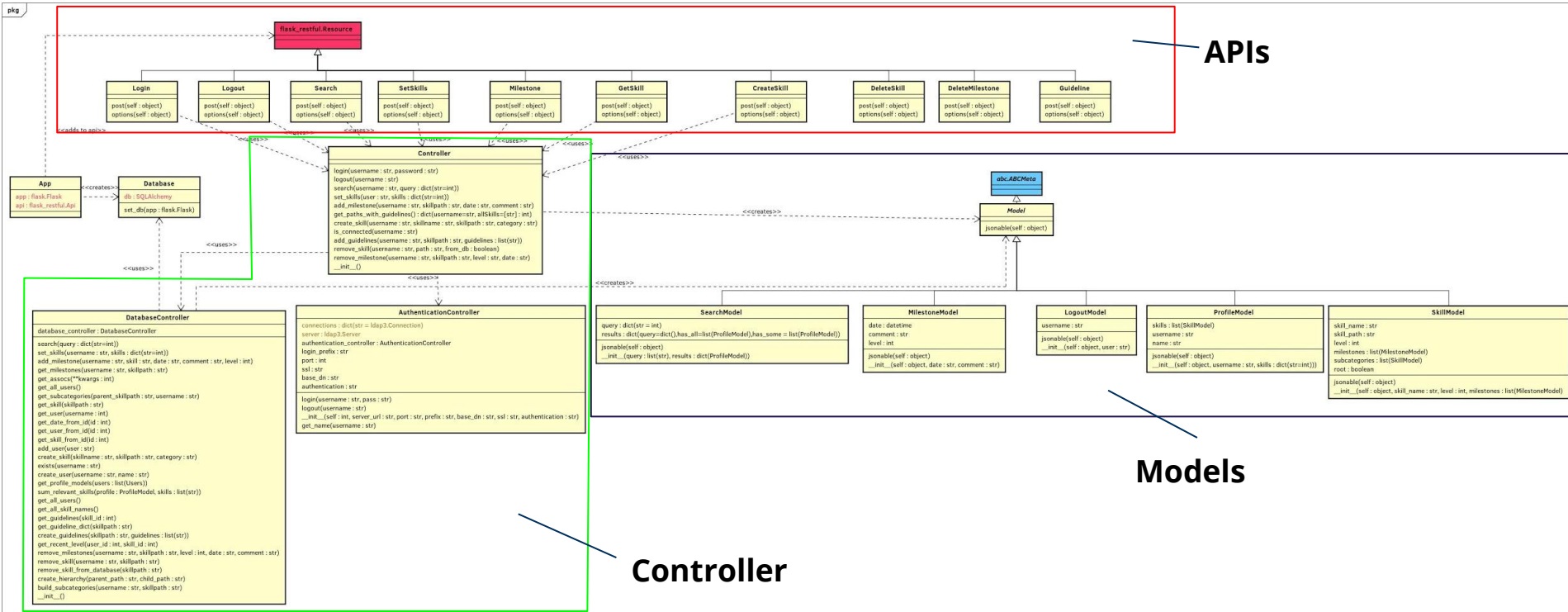
Backend - Datenbank - Früher Entwurf



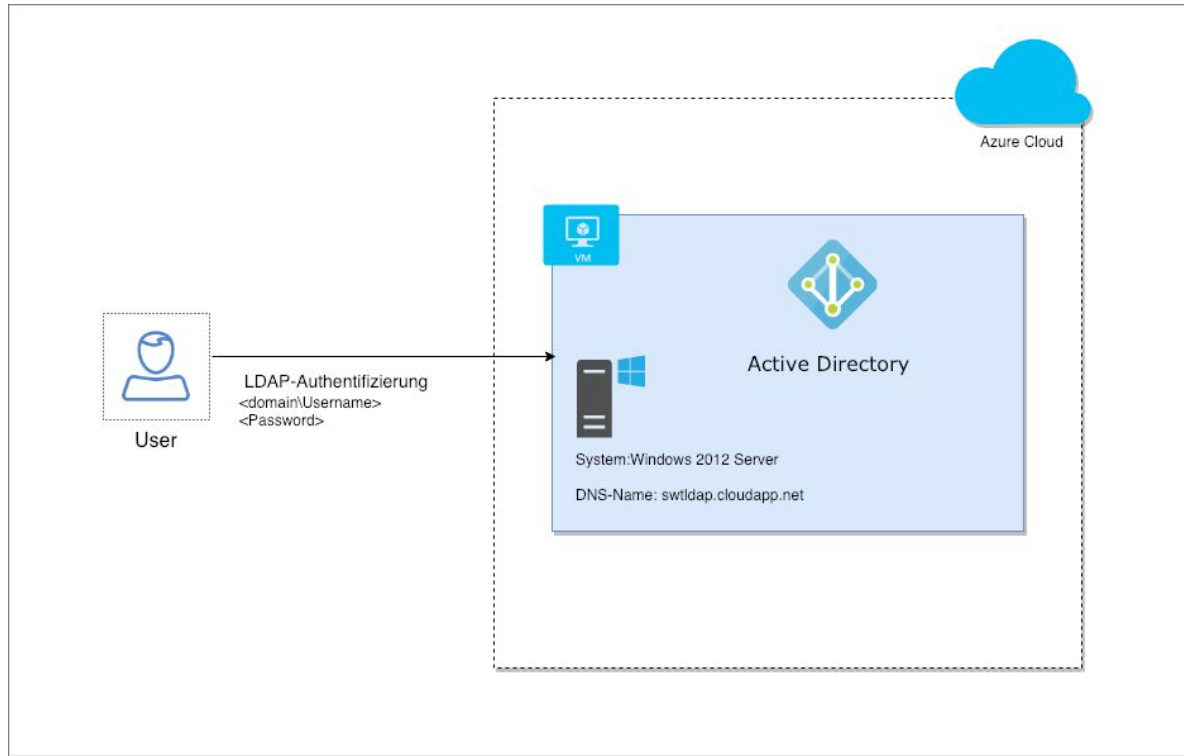
Backend - Datenbank - Endstand



Backend - Klassendiagramm



Active Directory

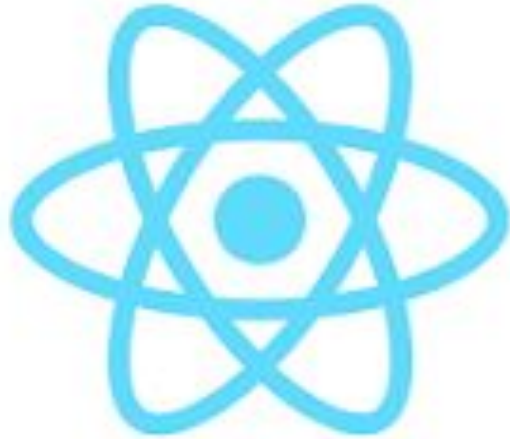


Authentifizierung

```
16 @staticmethod
17 def login(username, password):
18     """Authenticates user with AD.
19     Args:
20         username(`str`): name of the user
21         password(`str`): password of the user
22     Returns:
23         `dict`: profile of the logged in user as `ProfileModel` transformed into dict
24     """
25
26     name = authentication_controller.login(username, password)
27     if not database_controller.exists(username):
28         database_controller.create_user(username, name)
29     user_skills = database_controller.get_skills(username)
30     return dict(user=ProfileModel(username, name, user_skills).jsonable())
```

```
1
2 from ldap3 import Server, Connection, ALL, NTLM
3 server = Server('servername', use_ssl=True, get_info=ALL)
4 conn = Connection(server, user="Domain\\User", password="password", authentication=NTLM, auto_bind=True)
5
```

Frontend - React



```
40 class SkillProfileList extends React.Component {  
41   state = {  
42     expanded: null,  
43   };  
44  
45   render() {  
46     const { categories } = this.props;  
47     <ExpansionPanel categories={categories} >  
48       <ExpansionPanelDetails>  
49         {sortDatastructureRecursive(skill.subcategories, this.props)}  
50       </ExpansionPanelDetails>  
51     </ExpansionPanel>  
52     return panels;  
  }
```

Programming

Programming

Java

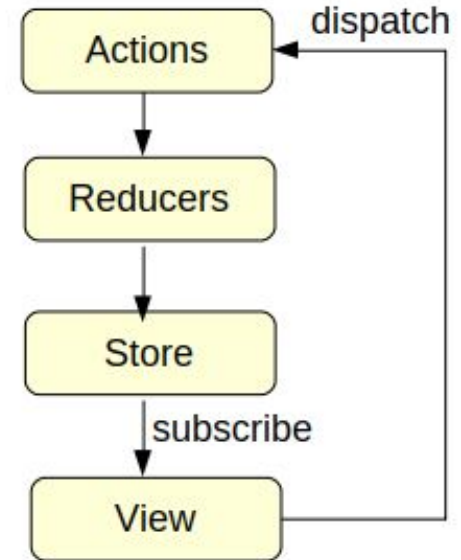
○ 1 ○ 2 ○ 3 ● 4 ○ 5

Frontend - Redux



```
export async function updateAllSkills() {  
  const Rest = new RestCom(RestPoints.getSkills());  
  try {  
    const { allSkills, categories } = await Rest.get();  
    store.dispatch(setAllSkills(allSkills));  
    store.dispatch(setAllCategories(categories));  
  } catch (e) {  
    console.log(e);  
  }  
}
```

```
▼ state: {...}  
  ► allCategories: Array[1]  
  ▼ allSkills: {...}  
    ▼ Programming/Python: {...}  
      1: "Insufficient"  
      2: "Sufficient/Below Average"  
      3: "Satisfactory / Average"  
      4: "Good"  
      5: "Excellent"  
    ☐ drawer: false  
    ► error: {...}  
    ► formState: {...}  
    ☐ loading: false  
    page: "profile"  
    ► profile: {...}  
    ► search: {...}  
    ► user: {...}
```



Demo

Gesprächsrunde - Fragen?