



import click

```
from flask_sqlalchemy import SQLAlchemy
from flask_login import UserMixin
from datetime import date, timedelta
from flask_bcrypt import Bcrypt
from app import app

app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///helferbaer.sqlite'

bcrypt = Bcrypt(app)
db = SQLAlchemy()
db.init_app(app)

class User(db.Model, UserMixin): #Quelle UserMixin: Flask-Login, Abschnitt "Your User Class"
    tablename = "user"
    userId = db.Column(db.Integer, primary_key=True, index=True)
    name = db.Column(db.String, nullable=False)
    firstName = db.Column(db.String, nullable=False)
    birthday = db.Column(db.Date, nullable=False)
    email = db.Column(db.String, nullable=False, unique=True)
    password = db.Column(db.String, nullable=False)
    phone = db.Column(db.String)
    role = db.Column(db.String, nullable=False)
    registered_date = db.Column(db.Date, default=date.today())

    jobs_created = db.relationship(
        "Job",
        foreign_keys='Job.kundeld',
        back_populates="kunde"
    )

    jobs_taken = db.relationship(
        "Job",
        foreign_keys='Job.helferId',
        back_populates="helfer"
    )
) meine class User sieht bisher so aus. Brauche ich also eine def get_id?
```

Nein, du brauchst keine explizite `def get_id(self)`-Methode, wenn du `UserMixin` importierst und korrekt konfigurierst – der Fehler liegt am fehlenden `user_loader` und möglicherweise am Primary-Key-Namen. [\[1\]](#) [\[2\]](#)

Problemursachen

UserMixin sucht standardmäßig nach einem Attribut namens id für get_id(), aber dein Primary Key heißt userId. Flask-Login wirft den Fehler, weil login_user die ID nicht findet. Zusätzlich fehlt der @login_manager.user_loader-Callback, der den User aus der Session lädt.^[3] ^[1]

Schnelle Fixes

Füge in app.py (oder wo LoginManager initialisiert wird) hinzu:

```
from flask_login import LoginManager
login_manager = LoginManager()
login_manager.init_app(app)
@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id)) # Lädt per userId (PK)
```

Das passt zu deinem userId-Feld, da SQLAlchemy query.get() den Primary Key verwendet.^[2]
^[3]

User-Modell-Optimierung

- Um Konflikte zu vermeiden, benenne den Primary Key zu id um: id = db.Column(db.Integer, primary_key=True). Passe Relationships an (foreign_keys='Job.kundeId' → 'Job.kunde_id').
^[4] ^[2]
- Oder überschreibe explizit:

```
def get_id(self):
    return str(self.userId)
```

Das erzwingt Kompatibilität mit UserMixin.^[5] ^[6]

Nach Migration (db.create_all()) und Neustart löst sich der Fehler in login_user(user). Teste mit current_user.is_authenticated.^[1]

✱

1. <https://flask-login.readthedocs.io>
2. <https://www.digitalocean.com/community/tutorials/how-to-add-authentication-to-your-app-with-flask-login>
3. <https://stackoverflow.com/questions/73977948/flask-login-using-usermixin-no-id-attribute-error-despite-being-implemented>
4. <https://www.freecodecamp.org/news/how-to-setup-user-authentication-in-flask/>
5. <https://github.com/mattupstate/flask-security/issues/472>
6. <https://stackoverflow.com/questions/34951134/flask-login-only-works-if-get-id-returns-self-email>
7. <https://www.mitchellbusby.com/2016/08/21/tutorial-roll-your-own-user-authorisation-management-with-flask-login/>
8. <https://www.freekb.net/Article?id=4545>

9. <https://stackoverflow.com/questions/50138426/how-to-get-name-of-logged-user-flask-sqlalchemy>
10. <https://groups.google.com/g/google-web-toolkit/c/JVaYlul7v6Q>
11. <https://hackersandslackers.com/flask-login-user-authentication/>
12. https://flask-user.readthedocs.io/en/latest/data_models.html
13. <https://overiq.com/flask-101/authentication-in-flask/index.html>
14. https://flask-login.readthedocs.io/_/downloads/en/0.4.0/pdf/
15. <https://www.youtube.com/watch?v=2dEM-s3mRLE>