1. Thiết kế cơ sở dữ liệu

|  |  |
| --- | --- |
| users | bookings |
| tutors | ratings |
| students | feedbacks |
| services | complaints |

Bảng users

|  |
| --- |
| from sqlalchemy import Column, Integer, String, Boolean  from sqlalchemy.ext.declarative import declarative\_base  Base = declarative\_base()  class User(Base):  \_\_tablename\_\_ = 'users'  id = Column(Integer, primary\_key=True)  name = Column(String, nullable=False)  email = Column(String, unique=True, nullable=False)  password = Column(String, nullable=False)  role = Column(String, nullable=False) # 'student', 'tutor', 'admin' |

**Bảng** Bookings (liên kết với Students và Tutors)

|  |
| --- |
| from sqlalchemy import ForeignKey, DateTime  class Booking(Base):  \_\_tablename\_\_ = 'bookings'  id = Column(Integer, primary\_key=True)  student\_id = Column(Integer, ForeignKey('users.id'))  tutor\_id = Column(Integer, ForeignKey('users.id'))  service\_id = Column(Integer, ForeignKey('services.id'))  date\_time = Column(DateTime, nullable=False)  status = Column(String, nullable=False) # 'pending', 'confirmed', 'completed' |

Bảng Ratings

|  |
| --- |
| from sqlalchemy import Column, Integer, ForeignKey, Float  class Rating(Base):  \_\_tablename\_\_ = 'ratings'  id = Column(Integer, primary\_key=True)  student\_id = Column(Integer, ForeignKey('users.id'), nullable=False) # Liên kết với học sinh  tutor\_id = Column(Integer, ForeignKey('users.id'), nullable=False) # Liên kết với gia sư  score = Column(Float, nullable=False) # Điểm số đánh giá (1-5)  comment = Column(String, nullable=True) # Nhận xét tùy chọn |

Bảng Feedbacks

|  |
| --- |
| from sqlalchemy import Column, Integer, ForeignKey, String, DateTime  from datetime import datetime  class Feedback(Base):  \_\_tablename\_\_ = 'feedbacks'  id = Column(Integer, primary\_key=True)  user\_id = Column(Integer, ForeignKey('users.id'), nullable=False) # Người gửi phản hồi  content = Column(String, nullable=False) # Nội dung phản hồi  created\_at = Column(DateTime, default=datetime.utcnow) # Thời gian phản hồi |

Bảng Complaints

|  |
| --- |
| from sqlalchemy import Column, Integer, ForeignKey, String, DateTime  class Complaint(Base):  \_\_tablename\_\_ = 'complaints'  id = Column(Integer, primary\_key=True)  user\_id = Column(Integer, ForeignKey('users.id'), nullable=False) # Người gửi khiếu nại  description = Column(String, nullable=False) # Mô tả khiếu nại  status = Column(String, default='pending') # Trạng thái xử lý ('pending', 'resolved', 'rejected')  created\_at = Column(DateTime, default=datetime.utcnow) # Thời gian gửi khiếu nại |

Bảng Services

|  |
| --- |
| from sqlalchemy import Column, Integer, ForeignKey, String, Float  class Service(Base):  \_\_tablename\_\_ = 'services'  id = Column(Integer, primary\_key=True)  tutor\_id = Column(Integer, ForeignKey('users.id'), nullable=False) # Gia sư cung cấp dịch vụ  subject = Column(String, nullable=False) # Môn học của dịch vụ  hourly\_rate = Column(Float, nullable=False) # Giá thuê theo giờ  description = Column(String, nullable=True) # Mô tả dịch vụ |

Tích hợp các bảng với ORM

Lưu ý : Định nghĩa tất cả các bảng trong file models ,tạo các migration và áp dụng chúng ,kiểm tra dữ liệu bằng cách thêm một số bản ghi mẫu qua giao diện hoặc script Python.

|  |
| --- |
| flask db migrate -m "Add Ratings, Feedbacks, Complaints, Services"  flask db upgrade |

1. Tích hợp chức năng chat

Tạo server WebSocket

Cài đặt Flask-SocketIO:

|  |
| --- |
| pip install flask-socketio |

Tạo file app.py

|  |
| --- |
| from flask import Flask  from flask\_socketio import SocketIO  app = Flask(\_\_name\_\_)  app.config['SECRET\_KEY'] = 'your\_secret\_key'  socketio = SocketIO(app)  @socketio.on('message')  def handle\_message(msg):  print(f"Received message: {msg}")  socketio.emit('response', {'data': msg})  if \_\_name\_\_ == '\_\_main\_\_':  socketio.run(app) |

Kết nối frontend

Sử dụng thư viện Socket.IO trên frontend để kết nối(html)

|  |
| --- |
| <script src="https://cdn.socket.io/4.0.1/socket.io.min.js"></script>  <script>  const socket = io('http://localhost:5000');  socket.on('connect', () => {  console.log('Connected to server');  });  socket.on('response', (data) => {  console.log(data);  });  socket.emit('message', 'Hello Tutor!');  </script> |

1. Bảo mật và xác thực

Sử dụng JWT

Cài đặt thư viện:

|  |
| --- |
| pip install PyJWT |

Triển khai mã nguồn JWT

|  |
| --- |
| import jwt  from datetime import datetime, timedelta  SECRET\_KEY = 'your\_secret\_key'  def create\_token(user\_id):  payload = {  'user\_id': user\_id,  'exp': datetime.utcnow() + timedelta(hours=1)  }  return jwt.encode(payload, SECRET\_KEY, algorithm='HS256')  def verify\_token(token):  try:  payload = jwt.decode(token, SECRET\_KEY, algorithms=['HS256'])  return payload['user\_id']  except jwt.ExpiredSignatureError:  return None |

Mã hóa mật khẩu

Sử dụng thư viện bcrypt

|  |
| --- |
| from bcrypt import hashpw, gensalt, checkpw  def hash\_password(password):  return hashpw(password.encode(), gensalt())  def verify\_password(password, hashed):  return checkpw(password.encode(), hashed) |

Lưu trữ tài liệu chứng minh

Sử dụng dịch vụ AWS S3 hoặc thư viện boto3

|  |
| --- |
| pip install boto3 |

Mã lưu trữ tài liệu

|  |
| --- |
| import boto3  s3 = boto3.client('s3', aws\_access\_key\_id='your\_access\_key', aws\_secret\_access\_key='your\_secret\_key')  def upload\_file\_to\_s3(file\_name, bucket\_name):  s3.upload\_file(file\_name, bucket\_name, file\_name) |