# Отчёт по практике

09.09.2025.

**Изучение FastAPI и Vue.js**

**Общее описание**

Сегодня мною было созданно полноценное веб-приложение с backend на FastAPI и frontend на Vue.js
Приложение включает в себя:
- Систему регистрации и авторизации пользователей
- Создание постов пользователями
- Работу с базой данных SQLite через SQLAlchemy
- REST API с валидацией данных

**Backend (FastAPI)**

**database.py - Настройка базы данных**

SQL_DB_URL = 'sqlite:///./itproger.db' engine = create_engine(SQL_DB_URL, connect_args={'check_same_thread': False})

Создание подключения к SQLite

**models.py - Модели данных**

class User(Base): **tablename** = "users" id = Column(Integer, primary_key=True, index=True) name = Column(String, index=True) age = Column(Integer)

Определение таблицы пользователей с полями

author = relationship("User")

Создание связи между постами и пользователем

**schemas.py - Валидация данных**

```
class UserCreate(UserBase):
    pass
```

Pydantic модель для валидации данных при создании пользователя

```
class Config:
    orm_mode = True
```

Включение режима работы с ORM объектами

**main.py - Основное приложение**

**CORP настройка:**

```
app.add_middleware(CORSMiddleware, allow_origins=origins, ...)
```

**Зависимости базы данных:**

```
def get_db():
    db = session_local()
    try:
        yield db
    finally:
        db.close()
```

Создание и закрытие соединения с БД для каждого запроса

**Создание пользователя:**

```
@app.post("/users/", response_model=DbUser) async def create_user(user: UserCreate, db: Session = Depends(get_db)) -> DbUser:
```

Post endpoint для создания пользователя с валидацией данных и автоматическим подключением к БД

**Обработка ошибок:**

```
raise HTTPException(status_code=404, detail="User not found")
```

Возврат понятной ошибки если  пользватель не найден

**Frontend (Vue.js)**

**AuthComponent.vue - Компонент авторизации**

**Двусторонее связывание данных:**

<input type="text" id="registerName" v-model="registerName">

Связывает значение input с переменной registerName  в компоненте

**Обработка формы:**

<form @submit.prevent="register">

Предотвращает стандартную отправку формы и вызывает метод register

**HTTP запросы:**

const response = await axios.post('http://127.0.0.1:8000/users/', {...})

Отправка POST запроса на создание пользователя

**Передача событий:**

this.$emit('handleAuth');

Отправка событий родительскому компоненту об успешной авторизации

**App.vue - Главный компонент**

**Условный рендеринг:**

<auth-component v-if="!isAuth" @handleAuth="handleAuth" />

<div v-else>Вы авторизованы

Показывает компонент авторизации если пользователь не авторизован, иначе показывает сообщение

**Ключевые основаные концепции:**

1. REST API dising - создание эндпоинтов для CRUD операций

2. ORM работы - использование SQLAlchemy для работы с БД

3. Валидация данных - Pydantic модели для проверки входящих данных

4. Зависимости - система Dependency Injection в FastAPI

5. CORS - настройка межсайтовых запросов

6. Vue.js основы - компоненты, данные, методы, события

7. HTTP клиент - работа с axios для API запросов

8. Состояние приложения - управление авторизацие через Vue