# Резюме на дипломен проект: Разработка на REST API с Django REST Framework

# 1. Въведение и цел

Представяне на съвременната архитектура REST за уеб услуги.

Цел: реализация на демонстрационно приложение за управление на курсове за деца, като учебен модел за изграждане и тестване на REST API.

# 2. Основи на REST архитектурата

Представяне като архитектурен стил за комуникация чрез HTTP.

Основни принципи: статус на заявки, уеднаквен интерфейс, слоеста структура.

Предимства: мащабируемост, яснота, сигурност.

# 3. Области на приложение

Уеб и мобилни приложения, интеграция между системи, IoT устройства.

Важност на простия и стандартен начин за обмен на данни.

# 4. HTTP методи в REST API

GET: взима данни.

POST: създава нов ресурс.

PUT/PATCH: актуализира съществуващ ресурс.

DELETE: изтрива ресурс.

Използване на статус кодове за комуникация.

# 5. Дизайн и структура на REST API

Ясна номенклатура на ресурсите.

Стандартизирано маршрутизиране и документация чрез Swagger и OpenAPI.

Важност на добрите практики и ясна структура.

# 6. Сигурност в REST API

Автентикация и авторизация (токен-базирана).

HTTPS протокол.

Валидация на входните данни, предпазване от SQL инжектиране и XSS.

# 7. Потенциални проблеми и ограничения

CORS ограничения при различни домейни.

Версиониране и поддръжка на старите API.

Ограничения по скорост и честота на заявките.

Нееднородност между различните реализации.

# 8. Тестване

Важност на unit, интеграционни и сигурностни тестове.

Инструменти: Postman, Swagger.

Провеждане на проверки за функционалност и сигурност.

# 9. Описание на приложението

Демонстрационен проект за управление на курсове за деца.

Възможност за CRUD операции върху курсове и записвания.

Включва потребителски интерфейс с Vu.js и API комуникация.

# 10. Използвани технологии

Python: лесен за научаване, широко използван.

Django: мощен уеб фреймуърк за организиране на приложението.

Django REST Framework (DRF): за автоматизация на създаването на REST API.

Vu.js: модерен фронтенд фреймуърк за потребителски интерфейс.

Axios: за HTTP заявки към API.

# 11. Архитектура на приложението

Модели: Курсове и Записвания.

Сериализатори: превръщане към/от JSON.

Изгледи: логика за CRUD операции.

Маршрути: автоматично генериране чрез Routers.

# 12. Реализация на REST API

Дефиниране на сериализатори за модели.

Изгледи чрез ModelViewSet.

Автоматично маршрутизиране с DefaultRouter.

Примери за заявки и отговори.

# 13. Потребителски интерфейс

Реализиран като отделна HTML страница с Vu.js и Axios.

Обслужва заявки към API.

Тестиране на процесите чрез форма за регистрация и визуална обратна връзка.

# 14. Предимства и резултати

Възможност за развитие и разширяване.

Надеждност и бързина при работа.

Демонстрация на интеграция между фронтенд и бекенд.

Смесване на академични и практически аспекти.

# 15. Тестове и резултати

Провеждане на тестове през уеб интерфейса на DRF.

Коректна работа на CRUD операции.

Валидация на данни, безопасна комуникация с API.

# 16. Заключение

Успешна реализация на модерен, гъвкав и сигурен REST API.

Възможности за разширяване и интеграция.

Добър учебен модел за бъдещи разработчици.