ТЕХНИЧЕСКОЕ ЗАДАНИЕ (ТЗ)

на разработку онлайн-платформы обучения

Версия: 1.0

Дата: [23.02.2025]

1. Введение

1.1. Назначение документа

Настоящее техническое задание определяет общие, функциональные и нефункциональные требования к разработке онлайн-платформы для создания и управления образовательными курсами. Документ предназначен для всех участников проекта (разработчиков, тестировщиков, менеджеров, представителей заказчика).

1.2. Цели проекта

• Создание удобного и функционального инструмента для публикации и прохождения онлайн-курсов.

• Обеспечение эффективного взаимодействия между преподавателями и студентами.

• Автоматизация процесса тестирования, оценки знаний и выдачи сертификатов.

1.3. Термины и определения

• Студент – пользователь, проходящий курсы и выполняющий задания.

• Преподаватель – пользователь, создающий и редактирующий курсы, контролирующий выполнение заданий.

• Администратор – пользователь, управляющий системой, модерирующий контент и контролирующий доступ к ресурсам.

• Курс – структурированный набор образовательных материалов (видео, тексты, задания).

• API – программный интерфейс для взаимодействия между компонентами системы и внешними сервисами.

2. Область применения

2.1. Описание системы

Система предназначена для публикации, управления и прохождения онлайн-курсов. Она обеспечивает:

• Авторизацию и регистрацию пользователей (студенты, преподаватели, администраторы).

• Создание, редактирование и удаление курсов с мультимедийными материалами.

• Автоматизированное тестирование и оценку знаний.

• Интерактивное общение через встроенный форум.

• Интеграцию с внешними API (YouTube для видео-контента, Stripe для платежей, ReportLab для генерации сертификатов).

2.2. Ограничения и предположения

• Система должна поддерживать одновременное подключение большого количества пользователей.

• Требуется высокая безопасность хранения и передачи данных.

• Интерфейс должен быть адаптивным и удобным для пользователей различных устройств.

3. Пользовательские роли

3.1. Студент

• Регистрация и авторизация.

• Просмотр и прохождение курсов, выполнение тестов.

• Участие в обсуждениях на форуме.

• Получение сертификатов после успешного прохождения курса.

3.2. Преподаватель

• Создание и управление курсами (добавление видео, текстов, заданий).

• Просмотр статистики по прохождению курса студентами.

• Модерация обсуждений в рамках своего курса.

3.3. Администратор

• Управление пользователями (регистрация, блокировка, выдача прав).

• Модерация и контроль публикуемого контента.

• Настройка системы, управление интеграциями и обновлениями.

4. Функциональные требования

4.1. Система авторизации и регистрации пользователей

• Возможность регистрации через email и социальные сети.

• Многоуровневая система доступа с разделением по ролям (студент, преподаватель, администратор).

4.2. Управление курсами

• Создание нового курса с возможностью добавления описания, категорий, изображений.

• Редактирование и удаление курса.

• Загрузка и организация учебных материалов:

• Видео: интеграция с YouTube API для встраивания видеоконтента.

• Текстовые материалы и презентации.

• Задания и практические упражнения.

4.3. Система тестирования и оценки

• Автоматическая проверка тестовых заданий.

• Возможность создания различных типов вопросов (множественный выбор, открытый ответ).

• Формирование статистики и отчётов по результатам тестирования.

4.4. Форум для обсуждений

• Раздел для обсуждения курсов с возможностью создания тем, комментариев и ответов.

• Модерация сообщений (администратор и преподаватель курса).

4.5. Генерация сертификатов в PDF

• Автоматическая генерация сертификатов по итогам успешного прохождения курса.

• Настраиваемый дизайн сертификата с возможностью добавления логотипа и подписи.

4.6. Опционально: Онлайн-платежи

• Интеграция с Stripe API для обработки платежей при покупке платных курсов.

4.7. Система уведомлений

• Отправка уведомлений (email, push) о новых курсах, заданиях, результатах тестов.

• Использование Celery и Redis для организации фоновых задач.

5. Нефункциональные требования

5.1. Производительность и масштабируемость

• Обеспечение высокой скорости отклика системы при большом количестве одновременных пользователей.

• Возможность масштабирования компонентов (база данных, сервер приложений) по мере роста нагрузки.

5.2. Безопасность и защита данных

• Шифрование данных пользователей и сертификатов.

• Защита от SQL-инъекций, XSS и CSRF атак.

• Регулярное проведение аудита безопасности.

5.3. Юзабилити (удобство использования)

• Интуитивно понятный интерфейс с адаптивным дизайном (поддержка десктопных и мобильных устройств).

• Возможность кастомизации пользовательских настроек.

5.4. Совместимость и кроссбраузерность

• Поддержка основных браузеров (Chrome, Firefox, Safari, Edge).

• Оптимизированный рендеринг интерфейса на мобильных устройствах.

5.5. Локализация и мультиязычность

• Возможность перевода интерфейса на несколько языков.

6. Описание бизнес-логики

6.1. Процесс создания и публикации курсов

• Преподаватель создаёт курс, добавляет материалы и задания.

• Администратор проверяет и утверждает курс перед публикацией.

• После публикации курс становится доступен для студентов.

6.2. Процесс прохождения курса

• Студент регистрируется и выбирает интересующий курс.

• Система отслеживает прогресс студента, фиксирует результаты тестирования.

• По завершении курса автоматически генерируется сертификат.

6.3. Механизм оценки и тестирования

• Формирование тестовых блоков в рамках курса.

• Автоматическая проверка ответов и вычисление итоговой оценки.

• Возможность повторного прохождения тестов.

6.4. Генерация и выдача сертификатов

• При успешном прохождении курса система генерирует PDF-сертификат с использованием ReportLab.

• Сертификат доступен для скачивания и печати.

6.5. Административное управление

• Управление пользователями, курсам и контентом через административную панель.

• Ведение журналов изменений и действий пользователей.

7. Архитектура проекта

7.1. Общий обзор архитектуры

• Приложение построено по модели MVT (Model-View-Template) с использованием Django.

7.2. Компоненты системы

• Backend:

• Django с Django REST API для обработки запросов.

• Django ORM для работы с базой данных.

• База данных: PostgreSQL

• Фоновые задачи: Celery с Redis для реализации задач (уведомления, генерация отчетов).

• Frontend:

• Шаблоны Django с использованием Bootstrap/Tailwind CSS для адаптивного дизайна.

• Опционально: SPA (например, с использованием React или Vue) для расширенной функциональности.

7.3. Взаимодействие компонентов

• REST API обеспечивает взаимодействие между фронтендом и серверной логикой.

• Интеграция с внешними сервисами (YouTube, Stripe, ReportLab) осуществляется посредством соответствующих API.

7.4. ER-диаграмма и схемы

• Приложить ER-диаграмму базы данных, диаграммы последовательности и классов для наглядного представления архитектуры.

8. Внешние интеграции

8.1. YouTube API

• Встраивание видеоконтента через YouTube API.

• Автоматическое обновление и синхронизация видео-материалов.

8.2. Stripe API

• Обработка онлайн-платежей для покупки курсов (при наличии платных курсов).

8.3. ReportLab

• Генерация PDF-сертификатов на основе шаблонов с динамическим заполнением данных.

9. Интерфейсы

9.1. Пользовательский интерфейс

• Личный кабинет: просмотр курсов, статистика, сертификаты.

• Страница курса: описание, материалы, задания, тесты.

• Форум: обсуждения, комментарии, обратная связь.

• Административная панель: управление контентом и пользователями.

9.2. API-интерфейсы (REST API)

• Эндпоинты для работы с курсами, пользователями, тестами, уведомлениями.

• Документация по API (Swagger/OpenAPI) для разработчиков.

10. План тестирования

10.1. Юнит-тесты

• Тестирование отдельных модулей и функций системы.

10.2. Интеграционное тестирование

• Проверка корректного взаимодействия между компонентами (Backend, Frontend, API).

10.3. Тестирование безопасности

• Проведение аудита безопасности, тестирование уязвимостей (SQL-инъекции, CSRF, XSS).

10.4. Тестирование производительности

• Нагрузочное тестирование для оценки масштабируемости и скорости работы системы.

11. Документация и исходные материалы

11.1. Документ ТЗ

• Текущее техническое задание с описанием требований и архитектуры.

11.2. Проектная документация

• Полный набор документов (Google Docs, PDF) с описанием функционала, процессов и архитектурных схем.

11.3. Глоссарий проекта

• Определение ключевых терминов, сокращений и понятий, используемых в проекте.

11.4. Дополнительные материалы

• ER-диаграммы, диаграммы последовательностей, диаграммы классов, прототипы интерфейсов (Figma или аналог).

12. Этапы разработки и сроки

12.1. Анализ требований и планирование

• Сбор и уточнение требований, конкурентный анализ, формирование ТЗ.

12.2. Дизайн и прототипирование

• Разработка макетов и прототипов пользовательского интерфейса.

12.3. Разработка

• Реализация функционала согласно ТЗ с применением выбранных технологий.

12.4. Тестирование

• Проведение функционального, интеграционного и нагрузочного тестирования.

12.5. Запуск и поддержка

• Развертывание проекта, настройка серверной инфраструктуры, техническая поддержка и обновления.

13. Управление изменениями

13.1. Процедуры внесения изменений

• Все изменения в ТЗ должны согласовываться с ключевыми участниками проекта и документироваться.

13.2. Версионность документации

• Ведение истории версий ТЗ с указанием даты и описанием изменений.

14. Заключение

14.1. Ожидаемые результаты

• Создание функциональной, масштабируемой и безопасной онлайн-платформы для обучения, удовлетворяющей требованиям всех категорий пользователей.

14.2. Критерии приемки системы

• Полное соответствие функциональным и нефункциональным требованиям, успешное прохождение тестирования, положительная оценка конечных пользователей.

Приложения:

A. ER-диаграмма базы данных

B. Примеры API-запросов и ответов

C. Примеры макетов пользовательских интерфейсов