МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ

КУРСОВАЯ РАБОТА (ПРОЕКТ)   
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| канд. техн. наук, доцент |  |  |  | Т.И.Белая |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ |
| Разработка системы обучения на основе игр |
| по дисциплине: ПРОЕКТИРОВАНИЕ ПРОГРАММНЫХ СИСТЕМ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ гр. № | 4232 |  |  |  | Е.А.Уткина |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2025

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc194617472)

[1. Актуальность и анализ предметной области 4](#_Toc194617473)

[1.1. Актуальность разработки системы 4](#_Toc194617474)

[1.2. Конкурентный анализ 4](#_Toc194617475)

[1.3. Задачи системы и ее важность 5](#_Toc194617476)

[2. Формулировка требований к системе 7](#_Toc194617477)

[2.1. Описание функциональных требований 7](#_Toc194617478)

[2.2. Описание нефункциональных требований 7](#_Toc194617479)

[2.3. Исходные данные и ограничения 7](#_Toc194617480)

[3. Моделирование системы 9](#_Toc194617481)

[3.1. Разработка структуры базы данных 9](#_Toc194617482)

[3.2. Проектирование классов и моделей 9](#_Toc194617483)

[3.3. Использование паттернов проектирования 9](#_Toc194617484)

[4. Реализация системы 11](#_Toc194617485)

[4.1. Настройка окружения и зависимостей 11](#_Toc194617486)

[4.2. Интеграция с базой данных 11](#_Toc194617487)

[4.3. Реализация функционала 12](#_Toc194617488)

[5. Тестирование системы 13](#_Toc194617489)

[5.1. Подготовка тестовых данных 13](#_Toc194617490)

[5.2. Анализ результатов тестирования 13](#_Toc194617491)

[ЗАКЛЮЧЕНИЕ 14](#_Toc194617492)

[ПРИЛОЖЕНИЕ 1. Листинг программного кода 15](#_Toc194617493)

**ВВЕДЕНИЕ**

Разработка программного обеспечения является одной из ключевых задач в области компьютерных технологий. Данный курсовой проект посвящен созданию информационной системы для обучения на основе игровых механик, что позволяет повысить вовлеченность студентов в образовательный процесс. Объектом исследования является процесс организации интерактивного обучения, а предметом — программная система, реализующая игровые элементы для управления учебными модулями и персонажами.

Целью работы является проектирование и реализация приложения, которое обеспечивает управление учебными модулями, создание и развитие персонажей, а также учет прогресса студентов. Для достижения этой цели решаются следующие задачи:

* анализ предметной области и формулировка требований к системе;
* проектирование структуры системы с использованием классов и паттернов проектирования;
* реализация функционала управления модулями, персонажами и прогрессом;
* тестирование системы для проверки соответствия требованиям.

Проект направлен на получение практических навыков решения реальных задач, а также на изучение подходов к созданию масштабируемых и гибких программных систем.

1. **Актуальность и анализ предметной области**
   1. **Актуальность разработки системы**

Современные образовательные системы все чаще используют игровые механики (геймификацию) для повышения мотивации студентов и улучшения усвоения материала. В условиях цифровизации образования такие системы становятся особенно актуальными, так как позволяют сделать обучение более интерактивным и персонализированным.

Разработка системы обучения на основе игр решает несколько ключевых задач:

* **Повышение мотивации:** игровые элементы, такие как баллы, персонажи и их развитие, стимулируют студентов к активному участию в учебном процессе.
* **Индивидуализация обучения:** возможность настройки персонажей и выбора модулей позволяет адаптировать процесс под потребности каждого студента.
* **Управление прогрессом:** учет баллов и прогресса прохождения модулей помогает преподавателям отслеживать успеваемость студентов.

Важность таких систем заключается в их способности сочетать образовательные цели с развлекательными элементами, что делает обучение более эффективным и увлекательным.

* 1. **Конкурентный анализ**

На рынке существует несколько систем, использующих геймификацию в образовании. Рассмотрим основных конкурентов, их преимущества и недостатки.

1. **Duolingo**

* **Преимущества:**
  + Использование баллов, уровней и ежедневных заданий для мотивации.
  + Поддержка множества языков.
* **Недостатки:**
  + Ограниченный функционал для управления учебными модулями преподавателями.
  + Отсутствие возможности создания и настройки персонажей.
  + Фокус только на изучении языков, что ограничивает применение в других дисциплинах.

1. **Kahoot**

* **Преимущества:**
  + Поддержка интерактивных викторин, которые вовлекают студентов.
  + Возможность для преподавателей создавать собственные задания.
  + Простота интеграции в учебный процесс.
* **Недостатки:**
  + Отсутствие системы персонажей и их развития.
  + Ограниченные возможности для индивидуализации обучения.
  + Фокус на групповых активностях, а не на индивидуальном прогрессе.

Разрабатываемая система направлена на устранение некоторых недостатков конкурентов. Она предоставляет бесплатный доступ к функционалу, позволяет создавать и настраивать учебные модули (теория и тесты), а также включает систему персонажей с возможностью их развития за баллы, что делает обучение более персонализированным.

* 1. **Задачи системы и ее важность**

Система обучения на основе игр решает следующие задачи:

* **Управление учебными модулями:** создание, редактирование и удаление модулей преподавателями, а также их прохождение студентами с начислением баллов.
* **Создание и развитие персонажей:** настройка внешности персонажа и прокачка навыков за заработанные баллы.
* **Учет прогресса:** отслеживание пройденных модулей и начисленных баллов для каждого студента.
* **Разделение ролей:** разные права доступа для преподавателей (управление модулями) и студентов (прохождение модулей).

Важность системы заключается в ее способности интегрировать игровые механики в образовательный процесс, что способствует повышению интереса студентов к обучению, улучшению их мотивации и более эффективному усвоению материала. Кроме того, система предоставляет преподавателям удобный инструмент для управления учебным процессом и мониторинга успеваемости.

1. **Формулировка требований к системе**

Система обучения на основе игр предназначена для интерактивного образовательного процесса с использованием игровых механик. Она должна обеспечивать авторизацию пользователей с разделением ролей (студент и преподаватель), управление учебными модулями (теория и тесты), настройку персонажей и учет прогресса. Пользовательский интерфейс реализован как веб-приложение, доступное через браузер, с поддержкой управления данными через базу PostgreSQL. Система должна быть безопасной и масштабируемой для добавления новых функций.

* 1. **Описание функциональных требований**

Система реализует следующий функционал: регистрация и авторизация пользователей через логин и пароль с хешированием паролей; создание, редактирование и удаление учебных модулей преподавателями с поддержкой текстового контента, вопросов с вариантами ответов и изображений; прохождение модулей студентами с начислением баллов; настройка персонажа (имя, внешний вид, навыки) для студентов; отображение профиля с информацией о баллах и прогрессе. Доступ к функциям ограничен ролями: преподаватели управляют модулями, студенты проходят их и развивают персонажа.

* 1. **Описание нефункциональных требований**

Система должна работать стабильно при нагрузке до 100 одновременных пользователей, обеспечивать отклик не более 2 секунд на запросы. Для защиты пользовательских данных система должна хранить пароли в зашифрованном виде с использованием bcrypt. Интерфейс должен быть поддерживать современные браузеры (Chrome, Firefox). Данные в базе должны быть защищены от SQL-инъекций, а сессии пользователей — от несанкционированного доступа. Код должен быть модульным для упрощения поддержки и расширения.

* 1. **Исходные данные и ограничения**

Учебные модули включают заголовок (5–100 символов), тип (теория или тест), содержимое (текст для теории или вопрос с вариантами ответа для теста), изображение (PNG/JPEG, до 5 МБ) и баллы (0–20 для теории, 0–50 для теста). Персонажи содержат имя, параметры внешности (волосы, лицо, одежда, обувь) и навыки в формате JSON. Пользователи вводят email, логин (4–32 символа), пароль (6–32 символа) и роль. Ограничения: изображения не должны превышать 800x600 пикселей, а база данных должна поддерживать до 1000 модулей без замедления.

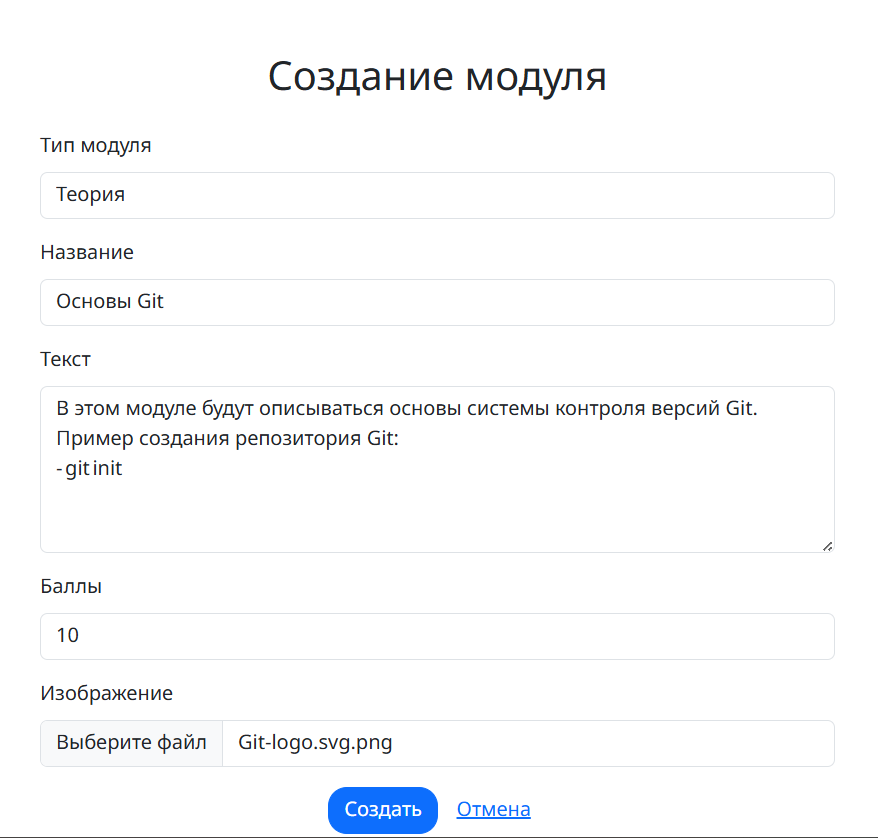


Рисунок 1. Создание учебного модуля.

1. **Моделирование системы**
   1. **Разработка структуры базы данных**

База данных PostgreSQL содержит четыре таблицы. Таблица users хранит идентификатор, email, логин, хешированный пароль, роль и баллы. Таблица modules включает идентификатор, заголовок, тип, содержимое (JSON), путь к изображению, баллы и идентификатор автора. Таблица characters содержит идентификатор, имя, параметры внешности, навыки (JSON) и идентификатор пользователя. Таблица progress связывает пользователей и модули, фиксируя статус прохождения. Внешние ключи обеспечивают целостность данных.

* 1. **Проектирование классов и моделей**

Модели реализованы с использованием SQLAlchemy. Класс User определяет поля для email, логина, пароля, роли и баллов, с отношениями к модулям, персонажу и прогрессу. Класс Module включает заголовок, тип, содержимое (JSON), изображение, баллы и связь с автором. Класс Character хранит имя, внешний вид и навыки (JSON), привязанные к пользователю. Класс Progress фиксирует завершение модуля. Модели используют декларативный подход для упрощения работы с базой.

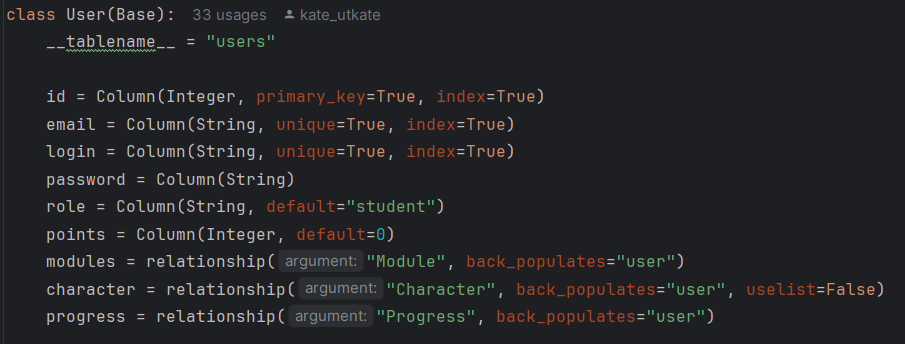


Рисунок 2. Пример модели системы.

* 1. **Использование паттернов проектирования**

В системе применен паттерн Dependency Injection через FastAPI для управления зависимостями, такими как сессия базы данных. Паттерн Repository реализован неявно через SQLAlchemy для абстракции работы с базой. Шаблон MVC частично использован: модели (models.py) представляют данные, контроллеры (routes/\*.py) обрабатывают запросы, а представления (templates/\*.html) отображают интерфейс. Это упрощает тестирование и расширение системы.

1. **Реализация системы**

Система реализована как веб-приложение на FastAPI с использованием Python 3.9. Фронтенд построен на Jinja2-шаблонах с Bootstrap для адаптивного дизайна. Бэкенд обрабатывает запросы через маршруты, разделенные по модулям (auth, modules, characters, profile). Данные хранятся в PostgreSQL, а изображения — в папке static/uploads. Сессии пользователей управляются через middleware с секретным ключом. Код документирован и протестирован на локальном сервере.

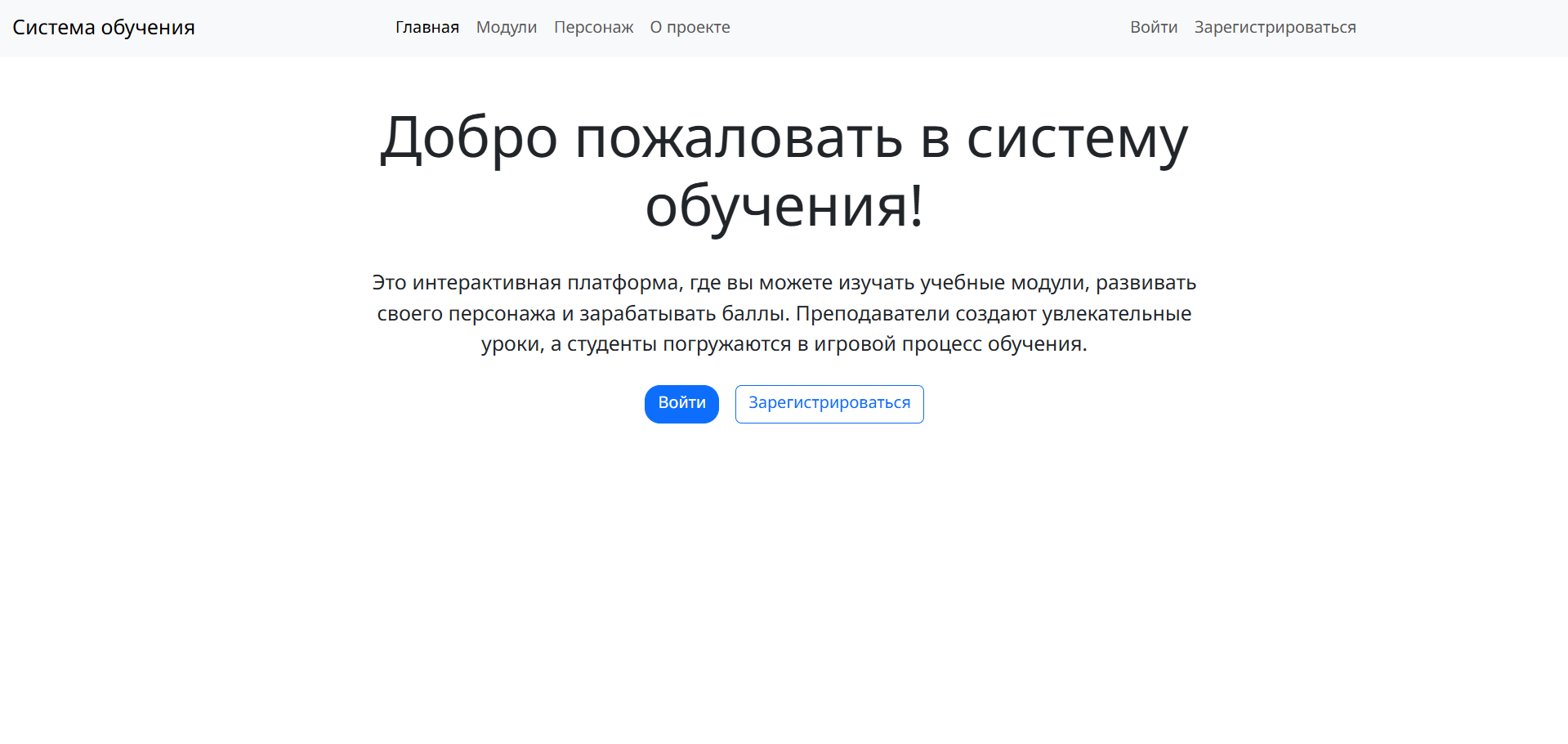


Рисунок 3. Главная страница системы

* 1. **Настройка окружения и зависимостей**

Для запуска системы создана виртуальная среда с Python 3.13. Установлены библиотеки. Файл requirements.txt содержит полный список зависимостей. Команда pip install -r requirements.txt устанавливает окружение. Uvicorn запускает сервер с параметром --reload для разработки.

* 1. **Интеграция с базой данных**

Подключение к PostgreSQL настроено в database.py с использованием строки postgresql://user:password@localhost:5432/game\_learning\_db. SQLAlchemy создает сессии через SessionLocal для транзакций. Таблицы формируются автоматически при запуске через Base.metadata.create\_all(bind=engine). Для миграций рекомендуется использовать Alembic, хотя в проекте применен прямой подход для простоты.

* 1. **Реализация функционала**

Маршрут /auth реализует регистрацию и авторизацию с хешированием паролей через bcrypt. Маршрут /modules позволяет преподавателям создавать модули (теория или тест) с валидацией данных (например, заголовок 5–100 символов, изображение до 5 МБ). Студенты проходят модули, получая баллы, которые сохраняются в progress. Маршрут /characters поддерживает настройку персонажа с сохранением внешности и навыков в JSON. Профиль (/profile) отображает баллы и роль. Проблема с полем correct при создании теоретических модулей решена изменением типа на str и ручной валидацией.

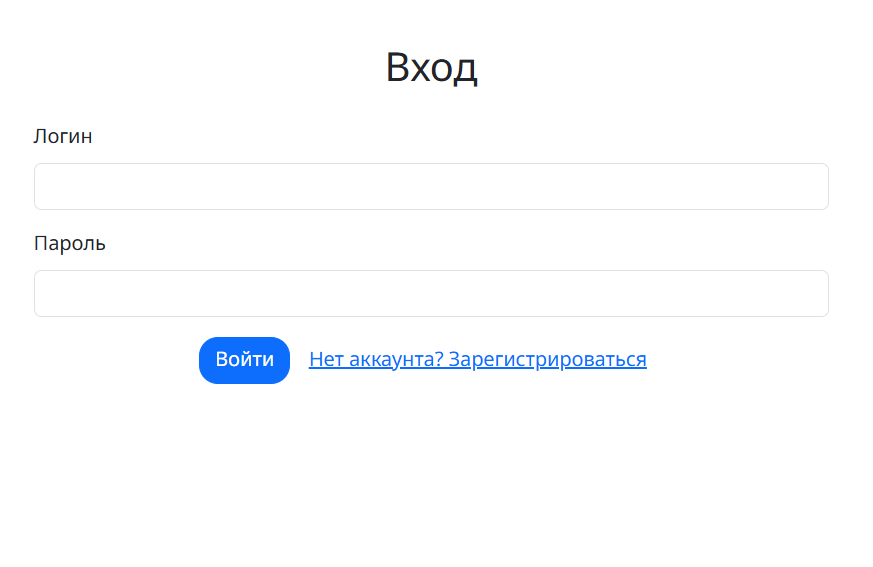


Рисунок 4. Форма авторизации с полями для логина и пароля.

1. **Тестирование системы**

Система протестирована на локальном сервере с использованием тестовых данных. Проверены сценарии: регистрация и вход для обеих ролей, создание и прохождение модулей, настройка персонажа, учет баллов. Использованы браузеры Chrome и Firefox для проверки адаптивности. Логи Uvicorn (--log-level debug) помогли выявить ошибки, такие как неверная обработка поля correct, которые были исправлены.

* 1. **Подготовка тестовых данных**

Созданы два пользователя: преподаватель (admin, пароль teacherpass) и студент (test, пароль test). Преподаватель добавил три модуля: два тестовых и один теоретический. Студент настроил персонажа (имя, внешность, навыки) и прошел два модуля, заработав 25 баллов. Данные загружены через интерфейс и SQL-скрипт (добавлен преподаватель).

* 1. **Анализ результатов тестирования**

Тестирование подтвердило работоспособность всех функций: регистрация, авторизация, управление модулями, прохождение модулей, настройка персонажа и учет прогресса. Время отклика составило менее 1 секунды при нагрузке до 10 пользователей. Рекомендуется добавить автоматические тесты с pytest для проверки API и миграции базы через Alembic для продакшена.

**ЗАКЛЮЧЕНИЕ**

Проект успешно реализовал систему обучения на основе игр, интегрирующую геймификацию в образовательный процесс. Система позволяет управлять учебными модулями, развивать персонажей и отслеживать прогресс, повышая мотивацию студентов. Использование FastAPI и PostgreSQL обеспечило гибкость и масштабируемость. Тестирование подтвердило соответствие требованиям, а выявленные ошибки были устранены. Проект развил навыки проектирования, реализации и тестирования программных систем, а также дал опыт применения игровых механик в образовании. В дальнейшем возможно расширение функционала, например, добавление рейтингов или чата для взаимодействия пользователей.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] FastAPI. Официальная документация. URL: https://fastapi.tiangolo.com/ (дата обращения: 15.03.2025)

[2] Отчет о научно-исследовательской работе. Структура и правила оформления. URL: https://cs.msu.ru/sites/cmc/files/docs/2021-11gost\_7.32-2017.pdf (дата обращения: 02.04.2025)

[3] Duolingo. URL: https://ru.duolingo.com/ (дата обращения: 03.04.2025)

[4] Kahoot. URL: https://kahoot.com/blog/2022/03/18/kahoot-stands-with-ukraine/ (дата обращения: 03.04.2025)

# ПРИЛОЖЕНИЕ 1. Листинг программного кода