Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования

«Петрозаводский государственный университет» Институт математики и информационных технологий Кафедра прикладной математики и кибернетики

Направление подготовки магистратуры

09.04.02 - Информационные системы и технологии

Магистерская программа “Управление данными”

Отчет по учебному курсу «Верификация программного обеспечения»

Выполнил:

студент группы 22605

Максименко Л.М.

Преподаватель:

к.ф.–м.н., доцент

К. А. Кулаков

Петрозаводск — 2021

Содержание:

1. Объект тестирования

1.1. Описание приложения

Средства автоматизации начинают использоваться повсеместно, не только на производстве, но и в жизни. Тестируемое веб-приложение предоставляет студентам возможность доступа к материалам курса по английскому языку, а преподавателю отслеживать активность студентов и проводить занятия реже используя бумажный носитель. В доступных пользователям возможностях имеется: авторизация по студенческому, грамматический раздел, разнообразие тематических текстов, словари по текстам, тесты по текстам и флеш-карточки проверяющие знание слов из словаря.

Приложение состоит из нескольких страниц:

- Авторизация - на вкладке есть возможность авторизации пользователей, которые уже получили доступ у пользователя. А также возможность быстрой регистрации нового участника.
- Главная страница - на вкладке доступен список тематических текстов, добавленных в систему. А также возможность перейти в раздел грамматики для повторения теории.
- Грамматическая - на вкладке доступен список времен английского языка и заполненной в них теорией.
- Тематическая - на вкладке доступен текст для изучения. А также возможность перейти в раздел тематического словаря и в раздел тестов по данному тесту.
- Тесты - на вкладке доступен тест для проверки внимательности изучения темы.
- Словарь - на вкладке доступен словарь с переводом слов из текста, помогающий и облегчающий процесс изучения темы.
- Флеш-карточки - на вкладке доступны карточки, которые проверяют усвоил ли ученик содержимое словаря.

Неотъемлемой частью жизненного цикла разработки программного обеспечения является тестирование, которое позволяет определить проблемные участи кода и вследствие этого повысить качество разрабатываемого продукта.

Задачами тестирования веб-приложения для занятий по английскому языку являются:

- Повышение качества, как отдельных модулей приложения, так и всего приложения в целом путем выполнения тестирования различных видов;
- Выявление ошибок в системе с целью их дальнейшего исправления.

Объектом тестирования является информационная система «English help!», предназначенная для помощи в обучении студентов и хранения данных для занятий в электронном виде. Информационная система реализована на базе Фреймворка Django версии 3.2.9 с использованием реляционной базы данных SQlite.

1.2. Архитектура приложения

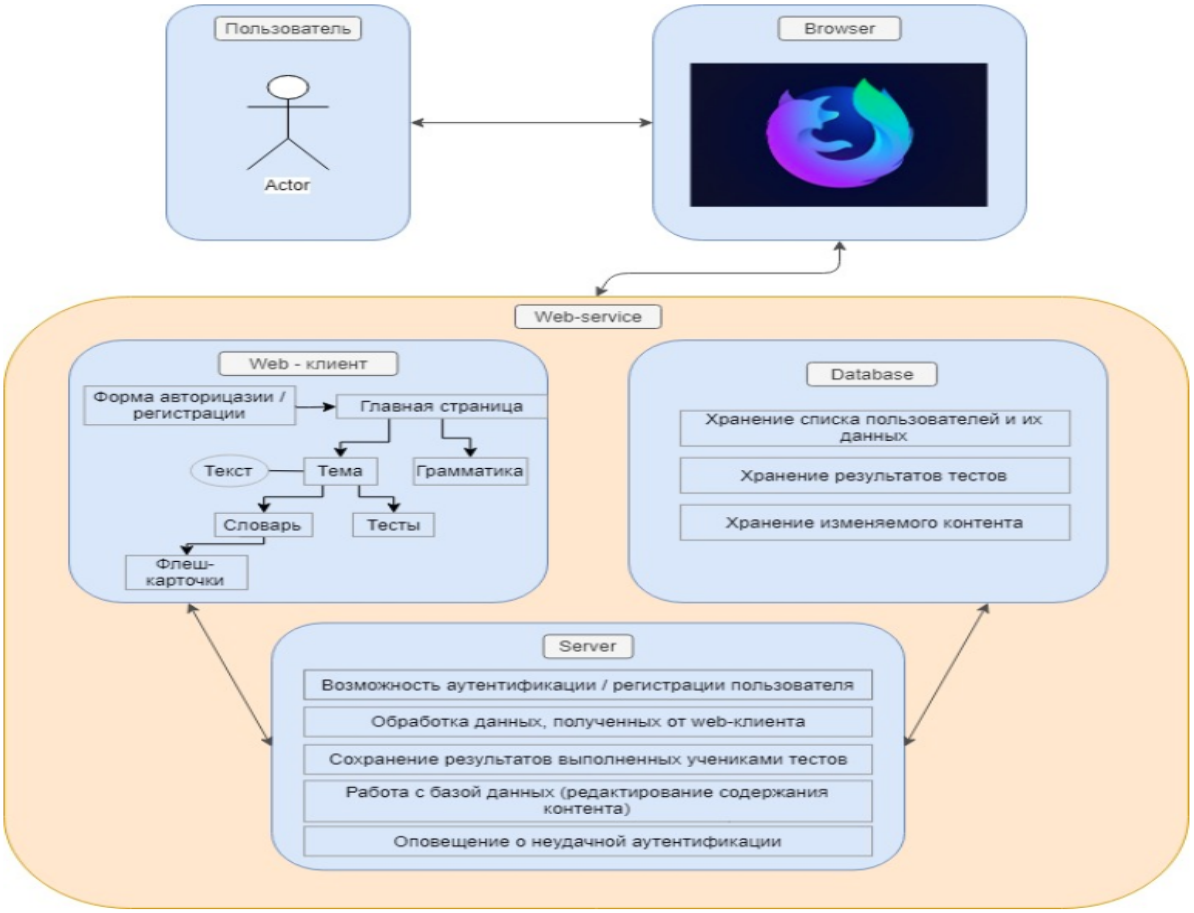


Рисунок 1: Архитектура приложения

Высокоуровневая модель архитектуры (рис. 1) показывает взаимодействие всех элементов системы между собой.

Основными модулями являются:

- Web-Client
    - Отображение формы аутентификации пользователя
    - Отображение главной страницы сервиса
    - Отображение личного кабинета
    - Отображение тематических разделов
    - Возможность выбрать нужный тематический раздел
        - Отображение категорий в данном разделе
        - Возможность выбрать нужную категорию
        - При выборе категории "Теория"

            - Отображение теоретического материала по выбранной тематике
        - При выборе категории "Словарь"
            - Отображение списка слов по выбранной тематике
            - Отображение раздела "Карточки"
            - Отображение флэш-карточек
            - Возможность открывать данные карточки
        - При выборе категории "Тесты"
            - Отображение тестов и вариантов ответов
            - Возможность выбирать вариант ответа
            - Возможность сохранить варианты ответов
            - Возможность посмотреть результаты
- Server
    - Возможность аутентификации пользователя
    - Оповещение о неудачной аутентификации
    - Обработка данных, полученных от web-клиента
    - Работа с базой данных (редактирование содержания контента)
    - Обработка данных отдельно для каждого пользователя, после прохождения тестов
    - Сохранение результатов выполненных учениками тестов
- Database
    - Хранение изменяемого контента
    - Хранение результатов тестов
    - Хранение списка пользователей и их данных

1.3. Описание модулей

Модули Server и Database относятся к внутренней логике Django. Их работу проверять не будем. Структура объекта Web-Client:

1.3.1. manage.py - Скрипт manage.py используется для создания приложений, работы с базами данных и для запуска отладочного сервера.

1.3.2. wsgi.py- используется для налаживания связи между вашим Django приложением и веб-сервером. Вы можете воспринимать его, как утилиту.

1.3.3. settings.py- содержит в себе все настройки проекта. Здесь мы регистрируем приложения, задаём размещение статичных файлов, настройки базы данных и так далее.

1.3.4. urls.py - Функция path принимает на вход URL-адрес и ставит в соответствие функцию из модуля views.py. URL соотношения хранятся в переменной urlpatterns, которая является списком функций path().

1.3.5. views.py - тут размещена «логика» работы веб-приложения. Она запросит информацию из модели, которую мы создали ранее, и передаст её в шаблон.

Состоит из четырехосновных функций: def index, def first, def two, def vote.

Def index - Функция генерирует поля авторизации, затем формирует страницу авторизации. При удачной авторизации производит перенаправление URL на страницу с темами, иначе выводится предупреждающее сообщение.

Для вызова извне доступны следующие методы:

- request.POST.get (name)

Возвращаемое значение: name.

Описание: функция возвращает значение переменной name из запроса от браузера. Аналогично для password.

- get_object_or_404 (users, name = name)

Возвращаемое значение: name_prof.

Описание: получает на вход имя модели из базы данных и id нужного элемента таблицы. Возвращает элемент из базы данных.

- Render (request,"first.html", {"text": test})

Возвращаемое значение: веб страница.

Описание: функция render получает объект запроса request и путь к файлу шаблона в рамках папки templates.

- HttpResponseRedirect (path)

Возвращаемое значение: переход на страницу.

Описание: функция HttpResponseRedirect производит перенаправление URL по пути path.

- UserForm ()

Возвращаемое значение: два поля для регистрации.

Описание: функция генерирует поля для авторизации пользователю.

Def first - функция формирует страницу со списком тем, названия которых загружаются из базы данных. Для вызова извне доступны следующие методы:

- get_object_or_404 (theory, id=1)

Возвращаемое значение: test.

Описание: получает на вход имя модели из базы данных и id нужного элемента таблицы. Возвращает элемент из базы данных.

- Render (request,"first.html", {"text": test})

Возвращаемое значение: веб страница.

Описание: функция render получает объект запроса request и путь к файлу шаблона в рамках папки templates.

Def two - Функция формирует страницу с теоретическим материалом, подгружая соответствующей теме текст из базы данных. Для вызова извне доступны следующие методы:

- get_object_or_404 (theory, id=num)

Возвращаемое значение: test.

Описание: получает на вход имя модели из базы данных и id нужного элемента таблицы. Возвращает элемент из базы данных.

- Render (request,"second.html", {"text": test})

Возвращаемое значение: веб страница.

Описание: функция render получает объект запроса request и путь к файлу шаблона в рамках папки templates.

Def vote - Функция формирует страницу с тестами, загружая список вопросов и ответом из базы данных. После выполнения теста отображает количество правильных ответов. Для вызова извне доступны следующие методы:

- get_object_or_404 (Question, pk=1)

Возвращаемое значение: question.

Описание: получает на вход имя модели из базы данных и id нужного элемента таблицы. Возвращает элемент из базы данных.

- Render (request, 'test.html', {"question": question, "question2": question2})

Возвращаемое значение: веб страница.

Описание: функция render получает объект запроса request и путь к файлу шаблона в рамках папки templates.

- question.choice_set.get (pk=request.POST['choice'])

Возвращаемое значение: selected_choice.

Описание: функция выбирает все ответы (choice), относящиеся к вопросу question по вторичному ключу (который определяет принадлежность к вопросу), полученному из запроса из браузера.

1.3.6. forms.py - Модуль содержит 1 класс – UserForm и отвечает за форму авторизации.

1.3.7. models.py – в нем определяются модели. Они реализуются как подклассы, и могут включать поля, методы и метаданные.

Модель может иметь произвольное количество полей любого типа - каждый представляет столбец данных, который мы хотим сохранить в одной из наших таблиц базы данных. Каждая запись (строка) базы данных будет состоять из одного значения каждого поля.

Модель содержит 4 класса – theory, users, question, choise.

Theory: содержит два текстовых поля CharField и TextField, используется при просмотре на главной странице списка тем и на странице самой темы.

Users: содержит два текстовых поля CharField, используется на странице авторизации для получения данных пользователя.

Question: содержит текстовые поле CharField, используется на странице с тестами для вывода текста вопросов в тестах. Для вызова извне доступны следующие методы:

__str__ (self)

Возвращаемое значение: question_text.

Описание: наполняет текстом созданный тест по теме на странице тесты.

Choise: используется на странице с тестами для заполнения текстом вариантов ответа и содержит поля:

ForeignKey - считывает номер вопроса;

CharField – текст для вариантов ответов;

BooleanField - поле, хранящее значение true/false;

IntegerField - поле, содержащее целые числа разделенные запятыми..

Для вызова извне доступны следующие методы:

__str__ (self)

Возвращаемое значение: choice_text.

Описание: наполняет текстом варианты ответов на созданный тест по теме на странице тесты.

1.3.8. apps.py – в нем определяются модели. Они реализуются как подклассы, и могут включать поля, методы и метаданные.

1.3.9. Templates – папка содержащая шаблоны страниц

Основным модулем в приложении является views.py. В нем размещена «логика» работы веб-приложения. Он запросит информацию из модели, которую мы создали ранее, и передаст её в шаблон.

Рисунок 1 содержит диаграмму, в которой показаны все связи между модулями программы, которые необходимо тестировать.

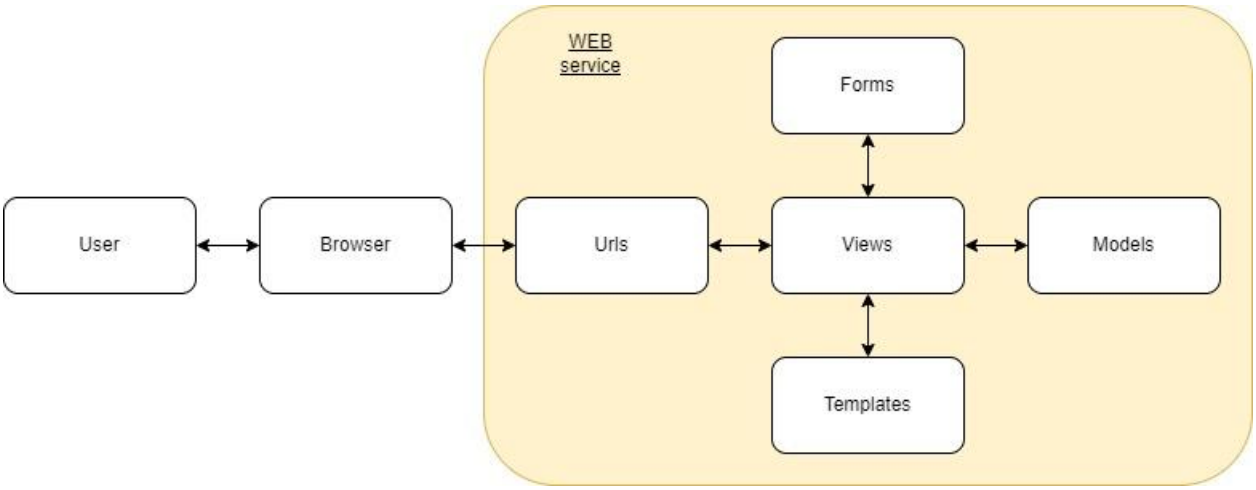


Рисунок 1

1.4. Функциональные возможности веб-приложения

Основные функциональные возможности веб-приложения:

1. Открыть сервис
2. Зарегистрироваться

3. Войти в систему (Авторизироваться)
4. Выбрать требующуюся ему тематику, в котором он может выбрать одну из трёх категорий - тесты, теория и словарь
5. В разделе теории просматривать теоретический материал
6. В разделе словаря просматривать слов по теме и их переводов, а также взаимодействовать с флэш-карточками
7. В разделе тестов проходить тестирование по теме
8. Выйти из сервиса

2. Стратегия тестирования

2.1. Стратегия блочного тестирования

Блочное тестирование будет проводиться встроенными инструментами в django с использованием модуля unittest Python.

Блочное тестирование должно быть применено к следующим модулям: views и models.

Модули urls, templates проверять не будем.

Модули относящиеся к внутренней логике Django проверять не будем: под номерами 1.3.1, 1.3.2, 1.3.3 и 1.3.8.

Должен быть разработан как минимум один тест на каждый модуль.

2.2. Стратегия интеграционного тестирования

Интеграционное тестирование будет проводиться встроенными инструментами в django с использованием модуля unittest Python.

Интеграционное тестирование будет проведено для следующих взаимодействий между модулями: urls, models, views, templates, forms.

Будут рассмотрены следующие комбинации этих модулей: urls, views, templates и urls, views, templates, models, forms; другие комбинации из этих модулей невозможны.

Модули относящиеся к внутренней логике Django проверять не будем: под номерами 1.3.1, 1.3.2, 1.3.3 и 1.3.8.

Для тестирования мы используем тестовый клиент Django Client. Данный класс действует как упрощённый веб-браузер который мы применяем для имитации GET и POST запросов и проверки ответов. Про ответы мы можем узнать почти все, начиная с низкоуровневого HTTP (итоговые заголовки и коды статусов) и вплоть до применяемых шаблонов, которые используются для HTML-рендера, а также контекста, который передаётся в соответствующий шаблон. Кроме того, мы можем отследить последовательность перенаправлений (если имеются), проверить URL-адреса и коды статусов на каждом шаге.

2.3. Стратегия аттестационного тестирования

В ходе аттестационное тестирования будет протестирована работоспособность приложения и его возможность осуществлять заявленный функционал.

Аттестационное тестирование будет проводиться методом «живого человека». Тестирующий человек, по заранее заданным инструкциям, производит требуемые действия и сверяется с заранее заданными результатами. Тест считается пройденным, если ожидаемый результат совпадает с фактическим результатом. В противном случае тест считается не пройденным

2.4. Стратегия нагрузочного тестирования

Нагрузочное тестирование – это процесс умышленной нагрузки системы, с целью определения показателей производительности, времени отклика, проверки соответствия требованиям, которые были предъявлены к данной системе или отдельному устройству.

JMeter имеет высокий порог вхождения. Поэтому был использован простой и удобный софт locust.

Целью данного тестирования является оценка производительности и работоспособности тестируемого модуля. Нагрузочные тесты:

1) Проверка работоспособности сервиса при одновременном прибавлении 4 пользователей в секунду, до тех пор пока не наберется 150 человек.
2) Проверка работоспособности сервиса при одновременном прибавлении 10 пользователей в секунду до тех пор пока не наберется 150 человек.
3) Проверка работоспособности сервиса при одновременном прибавлении 30 пользователей в секунду до тех пор пока не наберется 150 человек.

2.5. Критерии прохождения тестирования

Тест считается успешно пройденным, если ожидаемый и фактический результаты совпадают. Если тест завершается неудачей, то перед принятием решения целесообразно проверить правильность самого теста. Если тест завершился неудачей и тест реализован правильно, то производится заключение о найденной ошибке. Тестирование считается пройденным, если во время его прохождения не выявлено критических ошибок и количество пройденных тестов составляет не менее 85% от общего числа тестов.

2.6. Условия возобновления и приостановки выполнения тестов

Тестирование должно быть приостановлено, если количество не пройденных тестов превысит 15% от их общего количества. Тестирование должно быть приостановлено при обнаружении критических ошибок.

Тестирование возобновляется после исправления ошибок, выявленных при предыдущем тестировании. Повторное тестирование должно быть выполнено с самого начала, начиная с блочных тестов.

3. Детальный план тестирования

3.1. Аттестационное тестирование

Перечень доступных для перехода ссылок (далее #):

http://127.0.0.1:8000/
http://127.0.0.1:8000/theme/
http://127.0.0.1:8000/theme/*,(где * = 1,2,3,4,5,6,7,8,9)
http://127.0.0.1:8000/theme/*/test
http://127.0.0.1:8000/theme/*/dictionary
Итого 29 ссылок.
Перечень доступных для перехода ссылок (далее *):

На странице авторизации: проверка кнопок - вход, регистрация; 2

На главной странице и страницах словарей: проверка кнопок - выхода; 10

На страницах с темами: проверка кнопок - тест, словарь, выход; 27

На страницах с тестами: проверка кнопок - проверка, выход. 18

Итого 57 кнопок.

Таблица 3.1.1 Тест № А-1

№ теста	А-1
Цель теста (описание)	Приложение существует - отображение страницы регистрации в различных браузерах (url-адрес вводится в различных браузерах)
Тип теста	Позитивный
Входные данные	Ввод url-адреса в адресную строку браузер: http://127.0.0.1:8000/
Ожидаемый результат	Отображение сайта во всех имеющихся браузерах

Таблица 3.1.2 Тест № А-2

№ теста	А-2
Цель теста (описание)	Навигация на сайте (пользователь нажимает на различные элементы навигационного меню сайта)
Тип теста	Позитивный
Входные данные	Набор ссылок *
Ожидаемый результат	Переход к элементам сайта (все существует)

Таблица 3.1.3 Тест № А-3

№ теста	А-3
Цель теста (описание)	Работоспособность ссылок (Пользователь нажимает на ссылки)
Тип теста	Позитивный
Входные данные	Набор ссылок #
Ожидаемый результат	Переход по ссылкам (все существует)

Таблица 3.1.4 Тест № А-4

№ теста	А-4
Цель теста (описание)	Масштабируемость страниц (корректность отображения страниц сайта)
Тип теста	Позитивный
Входные данные	Запуск сайта на мониторах, имеющих различную диагональ экрана
Ожидаемый результат	Одинаковое (или с незначительной разницей) отображение страниц сайта

Таблица 3.1.5 Тест № А-5

№ теста	А-5
Цель теста (описание)	Отображение графических материалов (зайти на страницу, содержащую графический материал, проверить корректность отображения рисунков, фотографий и прочих графических материалов)
Тип теста	Позитивный
Входные данные	Страницы с графическим материалом
Ожидаемый результат	Отображение графического материала в предполагаемом масштабе, цвете и т.п.

Таблица 3.1.6 Тест № А-6

№ теста	А-6
Цель теста (описание)	Описание, содержание и свойства страниц (проверка описания страницы, её содержания и свойств)
Тип теста	Позитивный
Входные данные	Набор ссылок #
Ожидаемый результат	Соответствие описания, содержания, свойств страницы между собой и общим планом проекта

Таблица 3.1.7 Тест № А-7

№ теста	А-7
Цель теста (описание)	Вход в аккаунт (проверка существования аккаунта)
Тип теста	Позитивный
Входные данные	Логин и пароль (name='7624998', password='24.01.2000')
Ожидаемый результат	Переход на главную страницу (успешная авторизация)

Таблица 3.1.8 Тест № А-8

№ теста	А-8
Цель теста (описание)	Вход в аккаунт (проверка существования аккаунта)
Тип теста	Негативный
Входные данные	Логин и пароль (name='1', password='1')
Ожидаемый результат	Оповещение о неверной комбинации логина и пароля

Таблица 3.1.9 Тест № А-9

№ теста	А-9
Цель теста (описание)	Регистрация (проверка существования ограничений полей)
Тип теста	Негативный
Входные данные	Логин и пароль (name='111111111111111111', password='1')
Ожидаемый результат	Оповещение о невозможности создания такой комбинации логина пароля

Таблица 3.1.10 Тест № А-10

№ теста	А-10
Цель теста (описание)	Регистрация (проверка существования ограничений полей)
Тип теста	Негативный
Входные данные	Логин и пароль (name='1', password='1111111111111111')
Ожидаемый результат	Оповещение о невозможности создания такой комбинации логина пароля

Таблица 3.1.11 Тест № А-11

№ теста	А-11
Цель теста (описание)	Регистрация (проверка существования ограничений полей)
Тип теста	Негативный
Входные данные	Логин и пароль (name='111111111111111111', password='111111111111111111')
Ожидаемый результат	Оповещение о невозможности создания такой комбинации логина пароля

Таблица 3.1.12 Тест № А-12

№ теста	А-12
Цель теста (описание)	Прохождение теста (проверка вывода результатов)
Тип теста	позитивный
Входные данные	Все ответы проставлены
Ожидаемый результат	Оповещение о результатах

Таблица 3.1.13 Тест № А-13

№ теста	А-13
Цель теста (описание)	Прохождение теста (проверка вывода результатов)
Тип теста	Негативный
Входные данные	Некоторые вопросы не имеют отмеченного ответа
Ожидаемый результат	Оповещение о невозможности завершения теста

Таблица 3.1.14 Тест № А-14

№ теста	А-14
Цель теста (описание)	Проверка отображения информации (должна быть отображена вся необходимая информация)
Тип теста	Позитивный
Входные данные	Набор ссылок #
Ожидаемый результат	Выводится вся существующая информация

Таблица 3.1.15 Тест № А-15

№ теста	А-15
Цель теста (описание)	Вход с неверными данными (Авторизованный пользователь вводит несуществующий адрес)
Тип теста	Негативный

Входные данные	url адрес темы xxx: http://127.0.0.1:8000/theme/xxx/
Ожидаемый результат	Ошибка 404, страница не существует

3.2. Блочное тестирование

Таблица 3.2.1 Тест № Б-1

№ теста	Б-1
Цель теста (описание)	Проверка существования полей name и text бд использованных в классе theory
Тип теста	Позитивный
Объект тестирования	models.py, class theory
Входные данные	Создается объект БД [theory.objects.create (name='first', text='textforfirsttheory')]
Ожидаемый результат	Получаем название полей из раздела БД theory: name, text

Таблица 3.2.2 Тест № Б-2

№ теста	Б-2
Цель теста (описание)	Проверка размера полей name и text бд использованных в классе theory
Тип теста	Позитивный
Объект тестирования	models.py, class theory
Входные данные	Создается объект БД [theory.objects.create (name='first', text='textforfirsttheory')]
Ожидаемый результат	Получаем ограничения размера полей из раздела БД theory: 20, None

Таблица 3.2.3 Тест № Б-3

№ теста	Б-3
Цель теста (описание)	Функция two() принимает на вход id темы в БД. Производит поиск объекта в БД. В случае успеха происходит загрузка содержимого из БД. (Проверка существования значений name и text бд использованных в классе theory)
Тип теста	Негативный
Объект тестирования	views.py/def_two
Входные данные	Поиск объекта БД [theory.objects. (name='test1', text='test1')]
Ожидаемый результат	Error 404, такой записи нету в БД

Таблица 3.2.4 Тест № Б-4

№ теста	Б-4
Цель теста (описание)	Проверка существования полей name и password бд использованных в классе users

Тип теста	Позитивный
Объект тестирования	models.py, class users
Входные данные	Создается объект БД [users.objects.create (name='big', password='bob')]
Ожидаемый результат	Получаем название полей из раздела БД users: name, password

Таблица 3.2.5 Тест № Б-5

№ теста	Б-5
Цель теста (описание)	Проверка размера полей name и password бд использованных в классе users
Тип теста	Позитивный
Объект тестирования	models.py, class users
Входные данные	Создается объект БД [users.objects.create(name='big', password='bob')]
Ожидаемый результат	Получаем ограничения размера полей из раздела БД users:20, 20

Таблица 3.2.6 Тест № Б-6

№ теста	Б-6
Цель теста (описание)	Проверка существования значений name и password бд использованных в классе users
Тип теста	Негативный
Объект тестирования	views.py/def_index
Входные данные	Поиск объекта БД [users.objects.create (name='big', password='bob')]
Ожидаемый результат	Error 404, такой записи нету в БД

Таблица 3.2.7 Тест № Б-7

№ теста	Б-7
Цель теста (описание)	Проверка существования поля question_text бд использованного в классе question
Тип теста	Позитивный
Объект тестирования	models.py, class question
Входные данные	Создается объект БД [users.objects.create(question_text ='firstquestion')
Ожидаемый результат	Получаем название поля из раздела БД users: question_text

Таблица 3.2.8 Тест № Б-8

№ теста	Б-8
Цель теста (описание)	Проверка размера поля question_text бд использованного в классе question
Тип теста	Позитивный
Объект тестирования	models.py, class question

Входные данные	Создается объект БД [users.objects.create (question_text ='firstquestion')
Ожидаемый результат	Получаем ограничения размера поля из раздела БД users: 200

Таблица 3.2.9 Тест № Б-9

№ теста	Б-9
Цель теста (описание)	Проверка существования значений бд использованных в классе question
Тип теста	Негативный
Объект тестирования	views.py/def_vote
Входные данные	Поиск объекта БД [users.objects.create (question_text ='firstquestion')
Ожидаемый результат	Error 404, такой записи нету в БД

Таблица 3.2.10 Тест № Б-10

№ теста	Б-10
Цель теста (описание)	Проверка существования поля question_text бд использованного в классе choise
Тип теста	Позитивный
Объект тестирования	models.py, class choise
Входные данные	Создается объект БД [users.objects.create (question-id='1', choice_text='bob', poper='0')]
Ожидаемый результат	Получаем названия полей из раздела БД questions: question_id, choice_text, poper

Таблица 3.2.11 Тест № Б-11

№ теста	Б-11
Цель теста (описание)	Проверка размера поля question_text бд использованного в классе choise
Тип теста	Позитивный
Объект тестирования	models.py, class choise
Входные данные	Создается объект БД [users.objects.create (question='1', choice_text='bob', poper='0')]
Ожидаемый результат	Получаем ограничения размера полей из раздела БД questions: None,200,1

Таблица 3.2.12 Тест № Б-12

№ теста	Б-12
Цель теста (описание)	Функция vote() принимает на вход id темы в БД и варианты ответов теста. Загружает список вопросов, относящихся к этой теме. В случае успеха происходит отображение тестов из БД и вывод результатов теста.

	(Проверка существования значений бд использованных в классе question)
Тип теста	Негативный
Объект тестирования	views.py/def_vote
Входные данные	Поиск объекта БД [users.objects.create (question='1', choice_text='bob', poper='0')]
Ожидаемый результат	Error 404, такой записи нету в БД

Таблица 3.2.13 Тест № Б-13

№ теста	Б-13
Цель теста (описание)	Функция first() принимает на вход id тем в БД. Загружает список n, относящихся к этой теме. В случае успеха происходит тестов из БД и вывод результатов теста. (Проверка существования значений бд использованных в классе question)
Тип теста	Негативный
Объект тестирования	views.py/def_first
Входные данные	Поиск объекта БД [theory.objects (name='first'); theory.objects (name='second'); theory.objects (name='third')]
Ожидаемый результат	Error 404, такой записи нету в БД

3.3. Интеграционное тестирование

Тестирование производится на описанной ниже конфигурации БД:

- 9 тем с заполненным содержимым;
- 10 наборов имен и паролей для авторизации
- по одному словарю и тесту к каждой теме

Таблица 3.3.1 Тест № И-1

№ теста	И-1
Цель теста (описание)	Взаимодействие между urls, views, templates (Получение списка всех тем). Начало теста -(1)-> test view url exists at desired location (urls:'/theme/') -(2)-> test view url accessible by name (views:theme) -(3)-> Test view uses correct template (templates:"first.html") -(4)-> Test lists all themes(databasetest) -(5)-> Конец теста
Объект тестирования	Urls.py, view.py, templates
Тип теста	Позитивный
Входные данные	адрес URL: http://english-help.com/theme/ + БД

Ожидаемый результат	Указанный путь существует. Загрузка страницы со списком тем. Список из 9 тем: {{ theme_id: 1, theme_name: 'Travelling' }, { theme_id: 2, theme_name: 'Charecter' }, { theme_id: 3, theme_name: 'Hobbies' }, { theme_id: 4, theme_name: 'Food' }, { theme_id: 5, theme_name: 'Work' }, { theme_id: 6, theme_name: 'Nature' }, { theme_id: 7, theme_name: 'Appearance' }, { theme_id: 8, theme_name: 'Education' }, { theme_id: 9, theme_name: 'Home' }}

Таблица 3.3.2 Тест № И-2

№ теста	И-2
Цель теста (описание)	Взаимодействие модулей urls, forms,views, templates (вход пользователя в систему, загрузка страницы с темой) Начало теста -(1)->\ test logged (forms:(name='7624998', password='24.01.2000') -(2)->\ test view url exists at desired location (urls:'/theme/1') -(3)->\ test view url accessible by name (views: Travelling) -(4)->\ Test view uses correct template (templates:"two.html") -(5)->\ Test themes text(databasetest) -(6)->\ Конец теста
Объект тестирования	Urls.py, view.py, templates, forms.py
Тип теста	Позитивный
Входные данные	Существующий пользователь (name='7624998', password='24.01.2000') + адрес URL: http://english-help.com/theme/1 + БД
Ожидаемый результат	Указанный путь существует. Пользователь успешно прошел форму авторизации. Загрузка страницы с первой темой темой Travelling. status_code, 200

Таблица 3.3.3 Тест № И-3

№ теста	И-3
Цель теста (описание)	Взаимодействие модулей urls, forms,views, templates (вход пользователя в систему, загрузка страницы с несуществующей темой) Начало теста -(1)->\ test logged (forms:(name='7624998', password='24.01.2000') -(2)->\

	test view url exists at desired location (urls:'/theme/xxx') -(3)-> test view url accessible by name (views: xxx) -(4)->Test view uses correct template (templates:"two.html") -(5)->Test themes text(databasetest) -(6)->Конец теста
Объект тестирования	Urls.py, view.py, templates, forms.py
Тип теста	Негативный
Входные данные	Существующий пользователь (name='7624998', password='24.01.2000') + адрес URL: http://english-help.com/theme/xxx + БД
Ожидаемый результат	Пользователь успешно прошел форму авторизации. Указанный путь не существует. status_code, 404

Таблица 3.3.4 Тест № И-4

№ теста	И-4
Цель теста (описание)	Взаимодействие модулей urls, forms,views, templates (вход в систему пользователем, запуск теста по теме) Начало теста -(1)->test logged (forms:(name='7624998', password='24.01.2000') -(2)->test view url exists at desired location (urls:'/theme/1/test) -(3)->Test view uses correct template (templates:"test.html") -(4)->Test themes test text(databasetest) -(5)->Конец теста
Объект тестирования	Urls.py, view.py, templates, forms.py
Тип теста	Позитивный
Входные данные	Существующий пользователь (name='7624998', password='24.01.2000') + адрес URL: http://english-help.com/theme/1/test + Выбран раздел тест в 1 теме + БД
Ожидаемый результат	Пользователь успешно прошел форму авторизации. Загрузка страницы с тестами по выбранной теме прошла успешно. Указанный путь существует. status_code, 200

Таблица 3.3.5 Тест № И-5

№ теста	И-5

Цель теста (описание)	Взаимодействие модулей urls, forms,views, templates (вход в систему пользователем, запуск словаря по теме) Начало теста -(1)->\ test logged (forms:(name='7624998', password='24.01.2000') -(2)->\ test view url exists at desired location (urls:'/theme/1/dictionary) -(3)->\ Test view uses correct template (templates:" dictionary.html") -(4)->\ Test themes test text(databasetest) -(5)->\ Конец теста
Объект тестирования	Urls.py, view.py, templates, forms.py
Тип теста	Позитивный
Входные данные	Существующий пользователь (name='7624998', password='24.01.2000') + адрес URL: http://english-help.com/theme/1/dictionary + Выбран раздел словарь в 1 теме + БД
Ожидаемый результат	Пользователь успешно прошел форму авторизации. Указанный путь существует. Загрузка страницы со словарем по выбранной теме прошла успешно. status_code, 200

3.4. Нагрузочное тестирование

Таблица 3.4.1 Тест № Н-1

№ теста	Н-1
Цель теста (описание)	Проверка работоспособности сервиса при появлении 4 новых пользователей в секунду, до тех пор пока не наберется 150 человек. Присутствует разделение нагрузки: 2 пользователя на главную страницу и 2 пользователя на страницу первой темы
Тип теста	Общий
Входные данные	Каждую секунду имитируется появление 4 новых пользователей
Ожидаемый результат	Отсутствие ошибок

Таблица 3.4.2 Тест № Н-2

№ теста	Н-2
Цель теста (описание)	Проверка работоспособности сервиса при появлении 10 новых пользователей в секунду, до тех пор пока не наберется 150 человек.

	Присутствует разделение нагрузки: 5 пользователя на главную страницу и 5 пользователя на страницу первой темы
Тип теста	Общий
Входные данные	Каждую секунду имитируется появление 10 новых пользователей
Ожидаемый результат	Отсутствие ошибок

Таблица 3.4.3 Тест № Н-3

№ теста	Н-3
Цель теста (описание)	Проверка работоспособности сервиса при появлении 30 новых пользователей в секунду, до тех пор пока не наберется 150 человек. Присутствует разделение нагрузки: 15 пользователя на главную страницу и 15 пользователя на страницу первой темы
Тип теста	Общий
Входные данные	Каждую секунду имитируется появление 30 новых пользователей
Ожидаемый результат	Отсутствие ошибок

3.5. Покрытие кода тестами

Расчет тестового покрытия относительно исполняемого кода программного обеспечения проводится по формуле:

$$T_{cov} = \frac{L_{tc}}{L_{code}} \times 100\%$$

Где:

T_{cov} - тестовое покрытие;

L_{tc} - количество строк кода, покрытых тестами;

L_{code} - общее количество строк кода.

Тогда: $\boldsymbol{Tcov}$ = (403/847) * 100% = 47,5 %

3.6. Пример тестов на примере модели users:

test_models.py × test_views.py

C: > Users > dogptz > Downloads > english-help-master > english-help-master > topproj > test > test_models.py

```python
from django.test import TestCase
from tppo.models import users,theory,Question,Choice

class usersModelTest(TestCase):

    @classmethod
    def setUpTestData(cls):
        users.objects.create(name='big', password='bob')

    def test_users_name_label(self):
        user=users.objects.get(id=1)
        field_label = user._meta.get_field('name').verbose_name
        self.assertEquals(field_label,'name')

    def test_users_password_label(self):
        user=users.objects.get(id=1)
        field_label = user._meta.get_field('password').verbose_name
        self.assertEquals(field_label,'password')

    def test_users_name_max_length(self):
        user=users.objects.get(id=1)
        max_length = user._meta.get_field('name').max_length
        self.assertEquals(max_length,20)

    def test_users_password_max_length(self):
        user=users.objects.get(id=1)
        max_length = user._meta.get_field('password').max_length
        self.assertEquals(max_length,20)

    def test_users_object_name(self):
        user=users.objects.get(id=1)
        expected_object_name = '%s, %s' % (user.name, user.password)
        self.assertEquals(expected_object_name,str(user))
class theoryModelTest(TestCase):···
class ChoiceModelTest(TestCase):···
class QuestionModelTest(TestCase):···
```

4. Журнал тестирования

№ теста	Дата	Результат	Отчет об ошибке
A-1	25.12.21	Пройден	
A-2	25.12.21	Пройден	
A-3	25.12.21	Пройден	
A-4	25.12.21	Пройден	
A-5	25.12.21	Пройден	
A-6	25.12.21	Пройден	
A-7	25.12.21	Пройден	
A-8	17.01.22	Пройден	
A-9	17.01.22	Пройден	
A-10	17.01.22	Пройден	
A-11	17.01.22	Пройден	
A-12	17.01.22	Пройден	
A-13	17.01.22	Пройден	
A-14	25.12.21	Пройден	

А-15	25.12.21	Пройден	

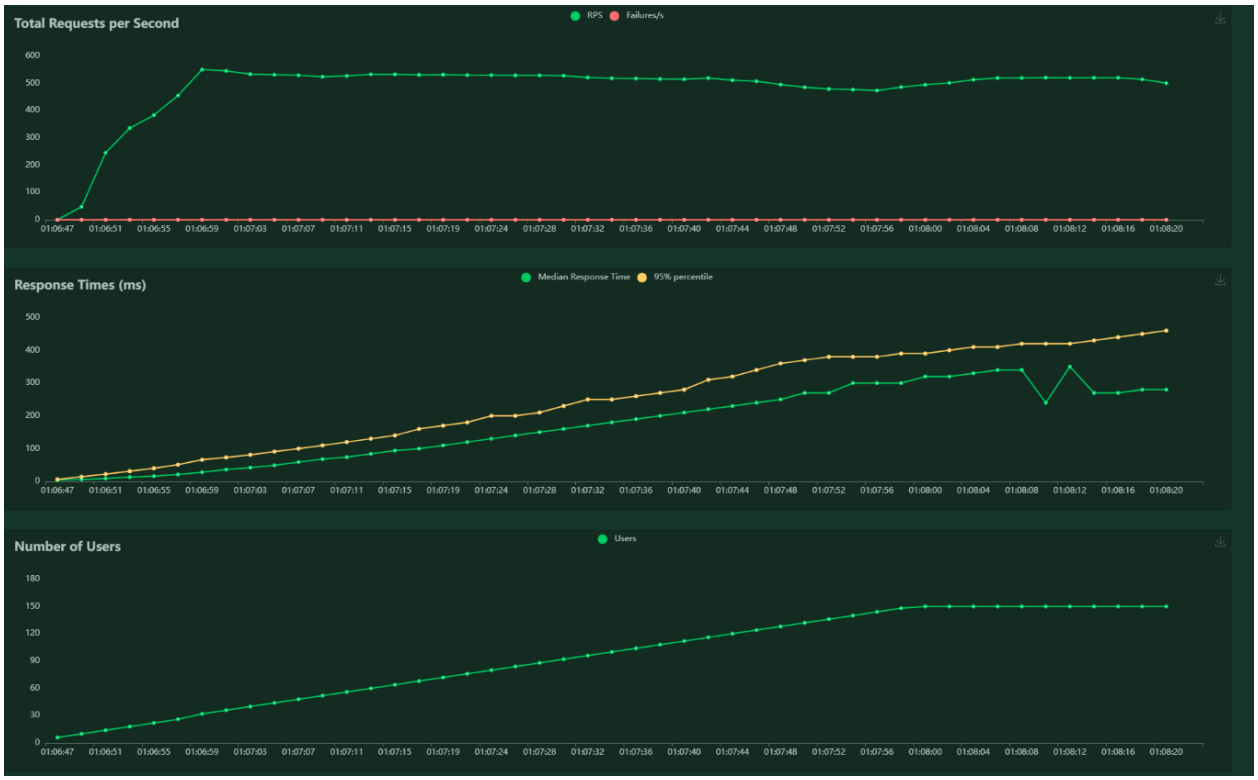
№ теста	Дата	Результат	Отчет об ошибке
Б-1	21.12.21	Пройден	
Б-2	21.12.21	Пройден	
Б-3	21.12.21	НЕ Пройден	1
Б-4	21.12.21	Пройден	
Б-5	21.12.21	Пройден	
Б-6	21.12.21	НЕ Пройден	2
Б-7	21.12.21	Пройден	
Б-8	21.12.21	Пройден	
Б-9	21.12.21	НЕ Пройден	3
Б-10	21.12.21	Пройден	
Б-11	21.12.21	Пройден	
Б-12	21.12.21	НЕ Пройден	4
Б-13	21.12.21	НЕ Пройден	5

№ теста	Дата	Результат	Отчет об ошибке
И-1	26.12.21	Пройден	
И-2	26.12.21	Пройден	
И-3	26.12.21	Пройден	
И-4	26.12.21	Пройден	
И-5	26.12.21	Пройден	

№ теста	Дата	Результат	Отчет об ошибке
Н-1	11.01.22	Пройден	
Н-2	11.01.22	Пройден	
Н-3	11.01.22	Пройден	

Н-1 результаты.

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/theme/	24574	0	120	210	131	2	2092	2802	250.2	0
GET	/theme/1	24469	0	240	400	235	3	498	3294	250.7	0
	Aggregated	49043	0	170	370	183	2	2092	3047	500.9	0

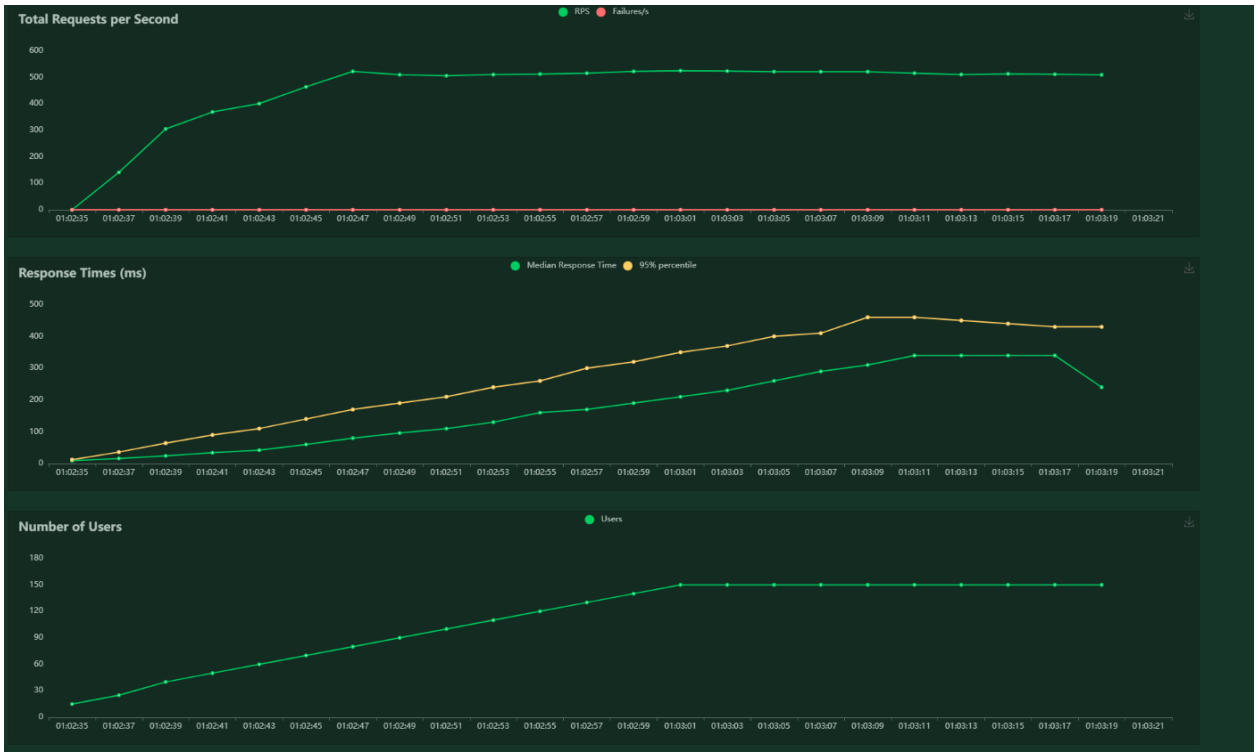



```
Name                                                          # reqs      # fails  |     Avg     Min     Max  Median  |   req/s failures/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET /theme/                                                    24574     0(0.00%)  |     131       2    2091     120  |  253.38    0.00
GET /theme/1                                                   24469     0(0.00%)  |     234       3     498     240  |  252.30    0.00
--------------------------------------------------------------------------------------------------------------------------------------------
Aggregated                                                     49043     0(0.00%)  |     182       2    2091     170  |  505.68    0.00

esponse time percentiles (approximated)
Type     Name                                                             50%    66%    75%    80%    90%    95%    98%    99%  99.9% 99.99%   100% # reqs
-------|--------------------------------------------------------------|--------|------|------|------|------|------|------|------|------|------|------|------|
GET      /theme/                                                          120    170    180    180    210    220    230    260   2100   2100   2100  24574
GET      /theme/1                                                         240    340    360    370    400    420    450    470    490    490    500  24469
-------|--------------------------------------------------------------|--------|------|------|------|------|------|------|------|------|------|------|------|
None     Aggregated                                                       170    200    250    310    370    400    430    460   2100   2100   2100  49043
```

H-2 результаты.

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/theme/	12230	0	150	210	157	4	2082	2802	256.1	0
GET	/theme/1	12125	0	300	410	266	7	492	3294	256.8	0
	Aggregated	**24355**	**0**	**180**	**390**	**212**	**4**	**2082**	**3047**	**512.9**	**0**




```
Name                                                          # reqs      # fails  |     Avg     Min     Max  Median  |   req/s failures/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET /theme/                                                    12230     0(0.00%)  |     157       3    2081     150  |  249.16    0.00
GET /theme/1                                                   12125     0(0.00%)  |     266       6     491     300  |  247.02    0.00
--------------------------------------------------------------------------------------------------------------------------------------------
Aggregated                                                     24355     0(0.00%)  |     211       3    2081     180  |  496.18    0.00

Response time percentiles (approximated)
Type     Name                                                             50%    66%    75%    80%    90%    95%    98%    99%  99.9% 99.99%   100% # reqs
--------|----------------------------------------------------------------|------|------|------|------|------|------|------|------|------|------|------|------|
GET      /theme/                                                          150    180    190    190    210    220    260   2000   2100   2100   2100  12230
GET      /theme/1                                                         300    360    380    380    410    430    450    470    490    490    490  12125
--------|----------------------------------------------------------------|------|------|------|------|------|------|------|------|------|------|------|------|
None     Aggregated                                                       180    220    310    350    390    410    440    470   2100   2100   2100  24355
```

Н-3 результаты.

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/theme/	9773	0	190	230	206	12	2581	2802	253.5	0
GET	/theme/1	9671	0	390	450	353	11	519	3294	255	0
	Aggregated	**19444**	**0**	**220**	**430**	**279**	**11**	**2581**	**3047**	**508.5**	**0**


```
Name                                                          # reqs      # fails  |     Avg     Min     Max  Median  |   req/s failures/s
--------------------------------------------------------------------------------------------------------------------------------------------
GET /theme/                                                     9773     0(0.00%)  |     205      11    2581     190  |  239.28    0.00
GET /theme/1                                                    9671     0(0.00%)  |     353      11     519     390  |  236.78    0.00
--------------------------------------------------------------------------------------------------------------------------------------------
Aggregated                                                     19444     0(0.00%)  |     278      11    2581     220  |  476.06    0.00

Response time percentiles (approximated)
Type     Name                                                     50%    66%    75%    80%    90%    95%    98%    99%  99.9% 99.99%   100% # reqs
--------|----------------------------------------------------|---------|------|------|------|------|------|------|------|------|------|------|------|
GET      /theme/                                                 190    200    210    220    230    240    280   2100   2100   2600   2600   9773
GET      /theme/1                                                390    410    420    420    450    460    490    490    520    520    520   9671
--------|----------------------------------------------------|---------|------|------|------|------|------|------|------|------|------|------|------|
None     Aggregated                                              220    370    390    400    430    450    480    500   2100   2600   2600  19444
```

5. Журнал найденных ошибок

№ отчёта об ошибке	1
Дата составления отчёта	28.12.2021
Номер теста	Б-3
Объект тестирования	views.py/def_two
Ожидаемый результат	Такой записи нету в БД
Фактический результат	AssertionError: 'first, textforfirsttheory' != 'theory object (1)'
Приоритет	Некритичная ошибка

№ отчёта об ошибке	2
Дата составления отчёта	28.12.2021
Номер теста	Б-6
Объект тестирования	views.py/def_index
Ожидаемый результат	Такой записи нету в БД
Фактический результат	AssertionError: 'big, bob' != 'users object (1)'
Приоритет	Некритичная ошибка

№ отчёта об ошибке	3
Дата составления отчёта	28.12.2021
Номер теста	Б-9
Объект тестирования	views.py/def_vote
Ожидаемый результат	Такой записи нету в БД
Фактический результат	AssertionError: ' firstquestion ' != 'question object (1)'
Приоритет	Некритичная ошибка

№ отчёта об ошибке	4
Дата составления отчёта	28.12.2021
Номер теста	Б-12
Объект тестирования	views.py/def_vote
Ожидаемый результат	Такой записи нету в БД
Фактический результат	AssertionError: 'bob' != 'choise object (1)'
Приоритет	Некритичная ошибка

№ отчёта об ошибке	5
Дата составления отчёта	28.12.2021

Номер теста	Б-13
Объект тестирования	views.py/def_first
Ожидаемый результат	Такой записи нету в БД
Фактический результат	AssertionError: 'first' != 'theme object (1)'
Приоритет	Некритичная ошибка

6. Результаты

Данное тестирование помогло работе проекта выявить ошибки в приложении. В ходе блочного, интеграционного, аттестационного и нагрузочного тестирования модулей приложения в рамках выполнения проекта было выявлено 5 некритических ошибок. Предположительно работа системы с технической точки зрения является работоспособной, все заявленные функции выполняются без ошибок. В ходе выполнения тестирования были выявлены недостатки, которые требуют рассмотрения в будущем.