*Федеральное государственное бюджетное образовательное учреждение*

*высшего профессионального образования*

|  |  |
| --- | --- |
|  | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ Информатика и системы управления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**РАСЧЁТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовому проекту на тему:**

«Система сбора и выдачи заданий по английскому языку»

Студент А.А. Пахомов

(Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта Л.Л.Волкова

(Подпись, дата) (И.О.Фамилия)

Москва, 2017

# Введение

В данной работе представлено проектирование и реализация системы выдачи и сбора заданий по английскому языку. Проект является учебным в рамках курса «Распределённые системы обработки информации».

Необходимо выполнить следующие требования:

1. Всё взаимодействие выполняется по REST.
2. Связь пользователей между системами выполняется с помощью UUID.
3. Все системы должны собираться и разворачиваться через CI.
4. Каждая система должна иметь свою собственную базу данных, прямой запрос к сторонней базе запрещен.
5. Предусмотреть возможность масштабирования сервисов.
6. Использовать Git и GitHub.

# Аналитическая часть

## Обзор и анализ предметной области

Проект является узкоспециализированным и может использоваться преподавателями для выдачи заданий зарегистрированным студентам.

Отличительной особенностью является распределённая архитектура серверной части, где микросервисы взаимодействуют с фронтендом по REST. На рисунке 1 представлена общая схема предметной области.

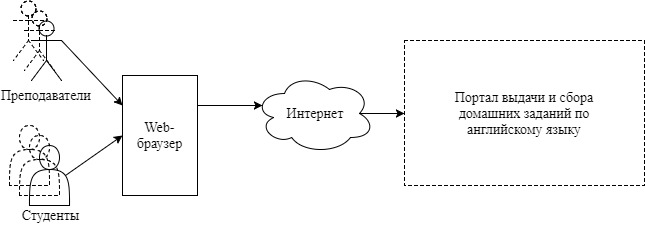


Рисунок 1 Схема предметной области

## Постановка задачи

Система разрабатывается для преподавателей английского языка в технических университетах. Предполагается, что студенты с помощью описываемой системы будут получать домашние задания по английскому языку, выполнять их и загружать на сайт. Преподаватель сможет проверять и выдавать новые задания. Все задания выдаются и собираются в текстовом виде. От системы требуется удобный интерфейс отображения новых уроков с нерешёнными заданиями, а также новых или изменённых ответов, которые преподаватель ещё не проверил.

## Входные параметры системы

**Студент**

* Фотография.
* Имя, фамилия и отчество (при наличии).
* Телефон.
* Электронная почта.
* Пароль.
* Группа.
* Преподаватель.
* Краткая информация о себе
* Выполненное задание – текстовое поле.

**Преподаватель**

* Фотография.
* Имя, фамилия и отчество (при наличии).
* Телефон.
* Электронная почта.
* Пароль.
* Краткая информация о себе
* Задание – текстовое поле.
* Оценка за задание.

### Выходные параметры системы

Выходными параметрами системы являются веб-страницы. Они должны содержать

следующую информацию:

* страница профиля пользователя системы (студента или преподавателя);
* страница урока: номер, задание, список ответивших студентов ам для просмотра их ответа и выставления оценки.

# Конструкторская часть

## Топология системы

На рис. 2 приведена топология разрабатываемой системы.

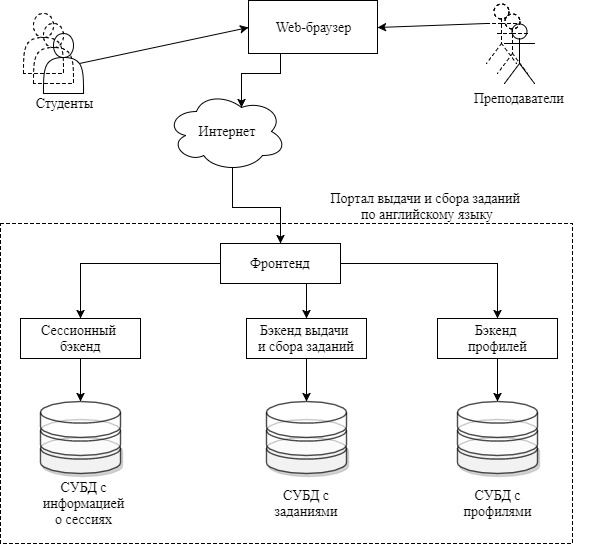


Рисунок 2. Топология системы

Система состоит из фронтенда и трёх бэкендов, что наиболее целесообразно для реализации ее основного назначения.

* **Сессионный бэкенд** отвечает за сессию пользователей портала и реализует

следующие функции:

* + регистрация пользователя;
  + аутентификация (проверка сессии) пользователя;
  + авторизация пользователя (вход, или «логин»);
  + выход из сессии («логаут»).
* **Бэкенд профилей** реализует следующие функции:
  + получение, изменение, удаление конкретного профиля;
  + добавление нового профиля.

В профиле хранится следующая информация о **студенте**:

* имя, фамилия, отчество;
* краткая информация о себе;
* фотография.

В профиле хранится следующая информация о **преподавателе**:

* имя, фамилия, отчество;
* краткая информация о себе;
* фотография.
* **Бэкенд выдачи и сбора заданий** реализует следующие функции:
  + создание нового урока;
  + добавление к уроку задания
  + добавление к уроку выполненного задания;
  + получение задания к уроку;
  + получение ответа студента на задание.

Фронтенд принимает запросы от пользователей по протоколу HTTP и анализирует их. На основе проведенного анализа фронтенд выполняет запросы к бэкендам, агрегирует ответы бекендов и отсылает ответ пользователю.

## Концептуальный дизайн

На рисунке 3 приведена концептуальная модель системы в нотации IDEF0.

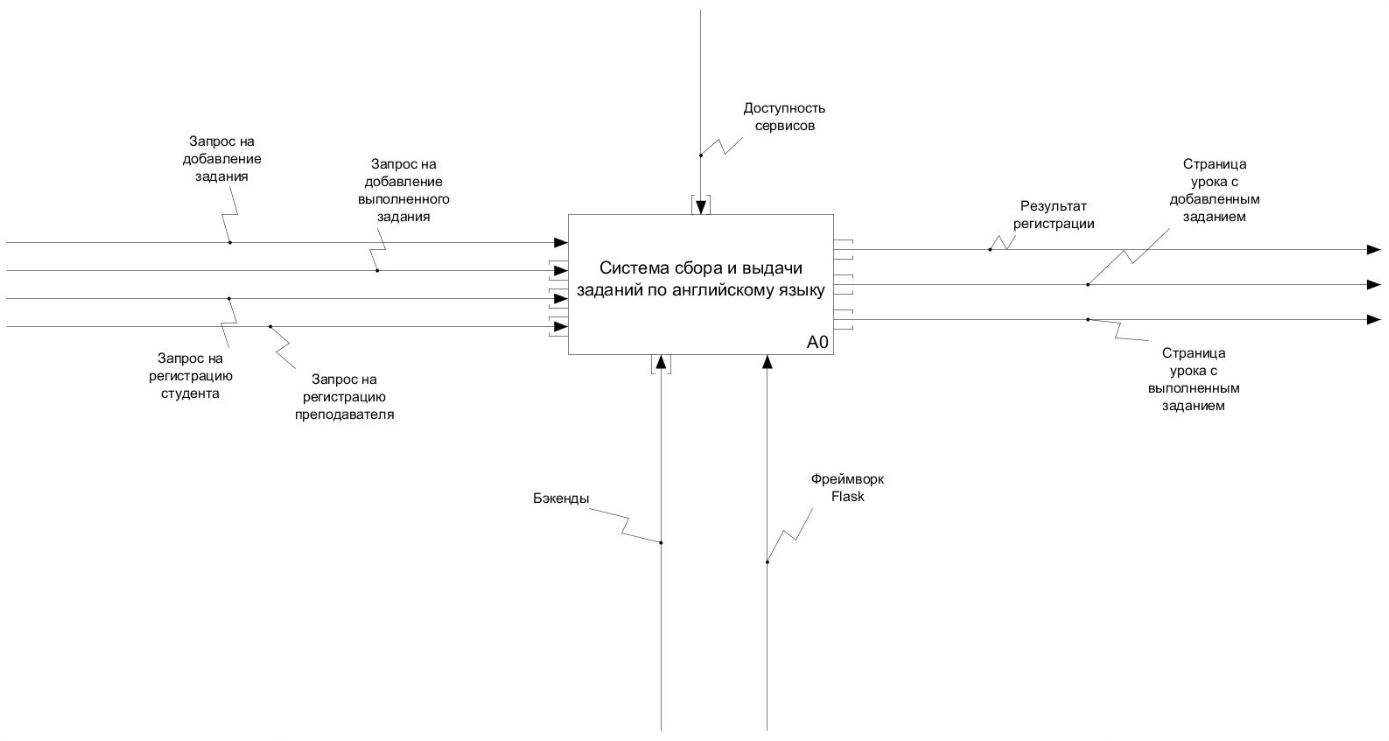


Рисунок 3. Концептуальная модель системы в нотации IDEF0

На рисунке 4 приведена детализированная концептуальная модель.

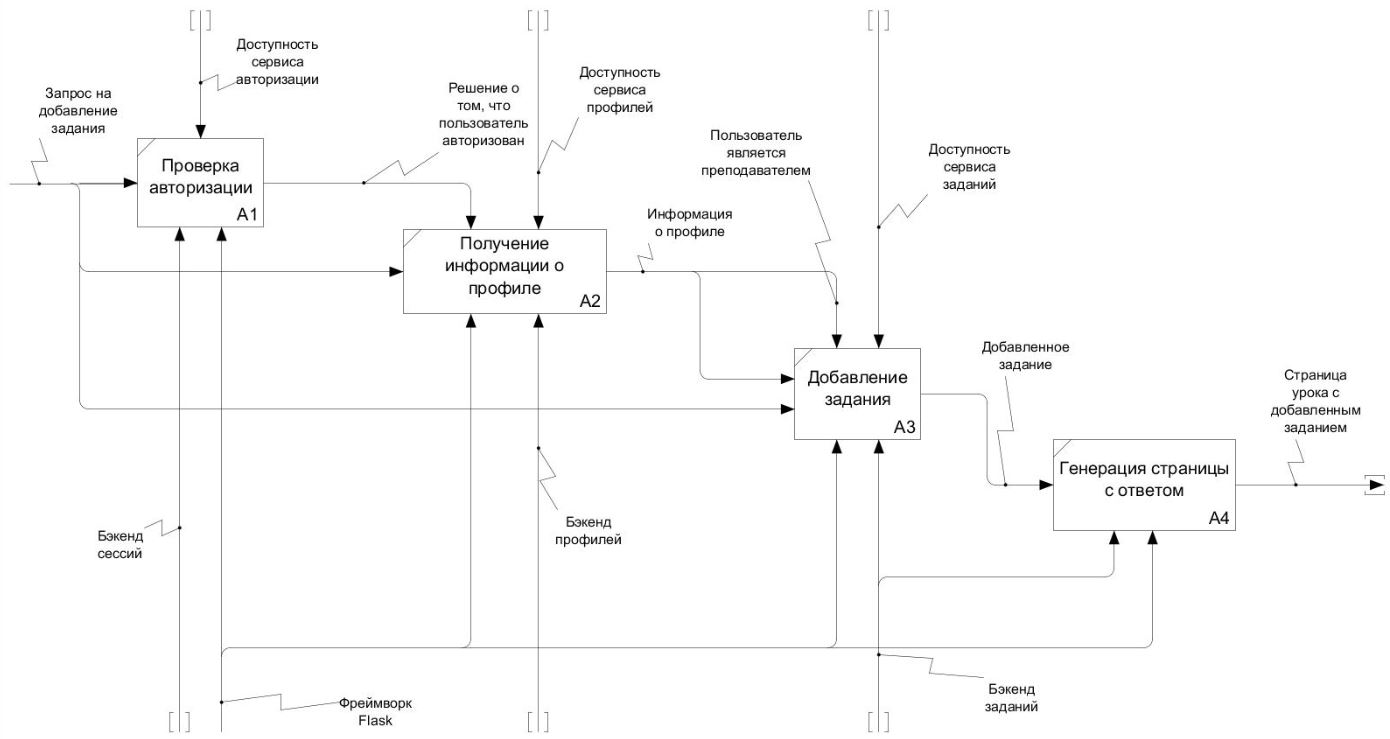


Рисунок 4. Детализированная концептуальная модель системы в нотации IDEF0.

## Варианты использования системы

На рисунке 5 представлена диаграмма прецедентов.



Рисунок 5. Диаграмма прецедентов

Далее представлены спецификации вариантов использования.

## Отправка выполненного задания

**Действующее лицо:**студент

**Цель:**отправить выполненное задание

**Предусловия:**студент осуществил вход в систему, задание к уроку добавлено

**Главная последовательность:**

1. студент заходит на страницу урока;
2. система показывает текстовое поле для записи в него ответа на задание;
3. студент пишет ответ и нажимает кнопку «Изменить решение».

## Добавление задания к уроку

**Действующее лицо:**преподаватель

**Цель:**добавить задание к уроку

**Предусловия:**преподаватель осуществил вход в систему

**Главная последовательность:**

1. преподаватель заходит на страницу урока;
2. система показывает текстовое поле для записи в него задания к уроку;
3. преподаватель пишет задание и нажимает кнопку «Сохранить».

**Альтернативная последовательность**(задание уже было добавлено):

1. преподаватель заходит на страницу урока;
2. система показывает текстовое поле с текущим заданием к уроку;
3. преподаватель вносит в задание изменения и нажимает кнопку «Обновить».

## Проверка задания

**Действующее лицо:**преподаватель

**Цель:**проверить задание к уроку

**Предусловия:**преподаватель осуществил вход в систему

**Главная последовательность:**

1. преподаватель заходит на страницу урока;
2. система показывает список студентов, оставивших ответы;
3. преподаватель нажимает на поле с записью о студенте и разворачивается ответ студента на задание к уроку;
4. ниже ответа студента есть выбор оценки за задание и кнопка «Поставить».

На рисунке 6 представлена диаграмма деятельности для прецедента «Проверить задание».

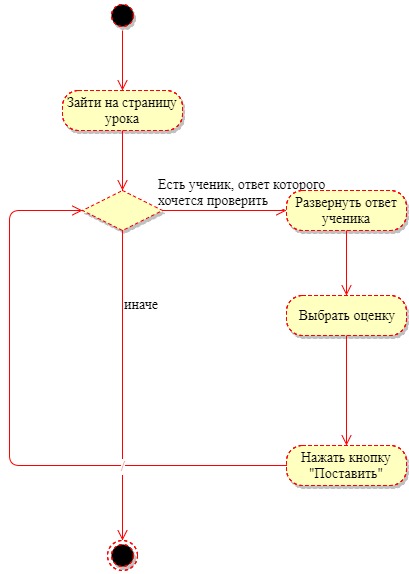


Рисунок 6. Диаграмма деятельности для прецедента "Проверить задание"

# Технологическая часть

## Топология классов

На рисунке 7 представлена диаграмма классов системы.

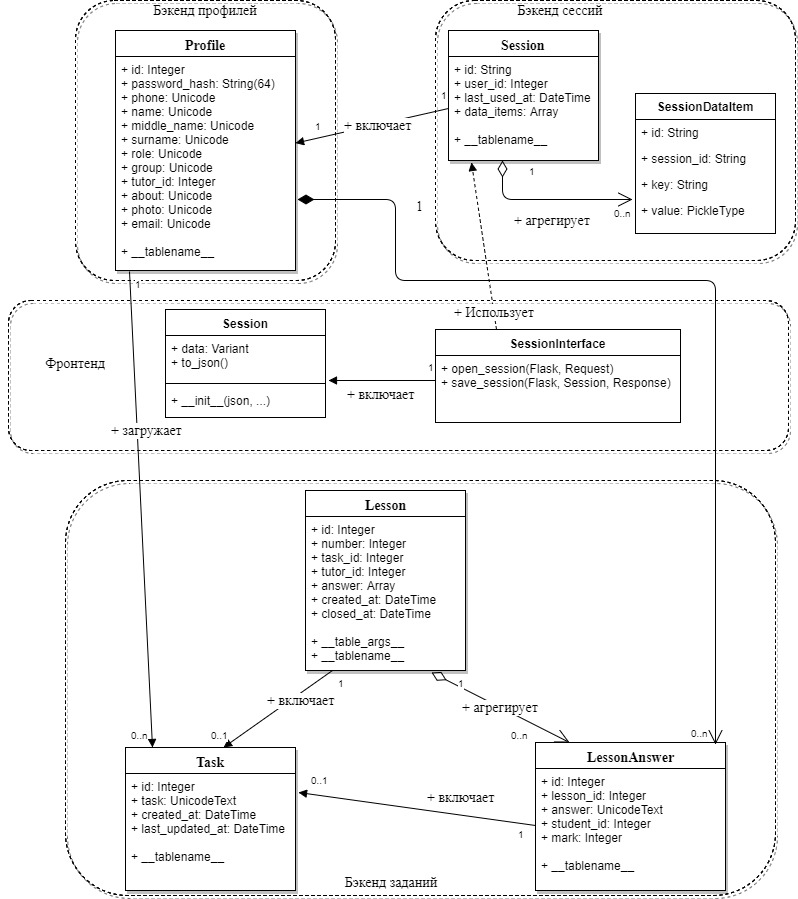


Рисунок 7. Диарамма классов

Класс **Profile** представляет собой профиль преподавателя или студента. Используется в бэкенде профилей.

Класс **Session** представляет собой запись о текущей сессии пользователя. Класс относится к бэкенду сессий.

Класс **Lesson** представляет собой запись об уроке. Имеет ограничение уникальности по паре (tutor\_id ⬄ number). Относится к бэкенду заданий. Связан с массивом ответов на задание, представленных классом **LessonAnswer**.

Класс **Task** представляет задание. Класс относится к бэкенду заданий. Содержит ссылку на преподавателя, добавившего задачу.

Класс **LessonAnswer** представляет ответ на задание. Класс относится к бэкенду задач. Содержит ссылку на студента, выполнившего задание.

## Диаграммы последовательности и потоков данных

На рисунке 8 представлена диаграмма последовательности для регистрации пользователя.

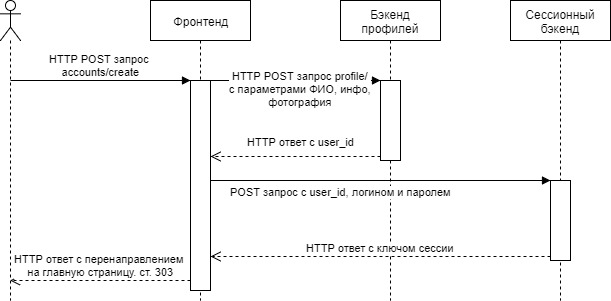


Рисунок 8. Диаграмма последовательности для регистрации пользователя

На рисунке 9 представлена диаграмма потоков данных при добавлении студентом ответа на задание.

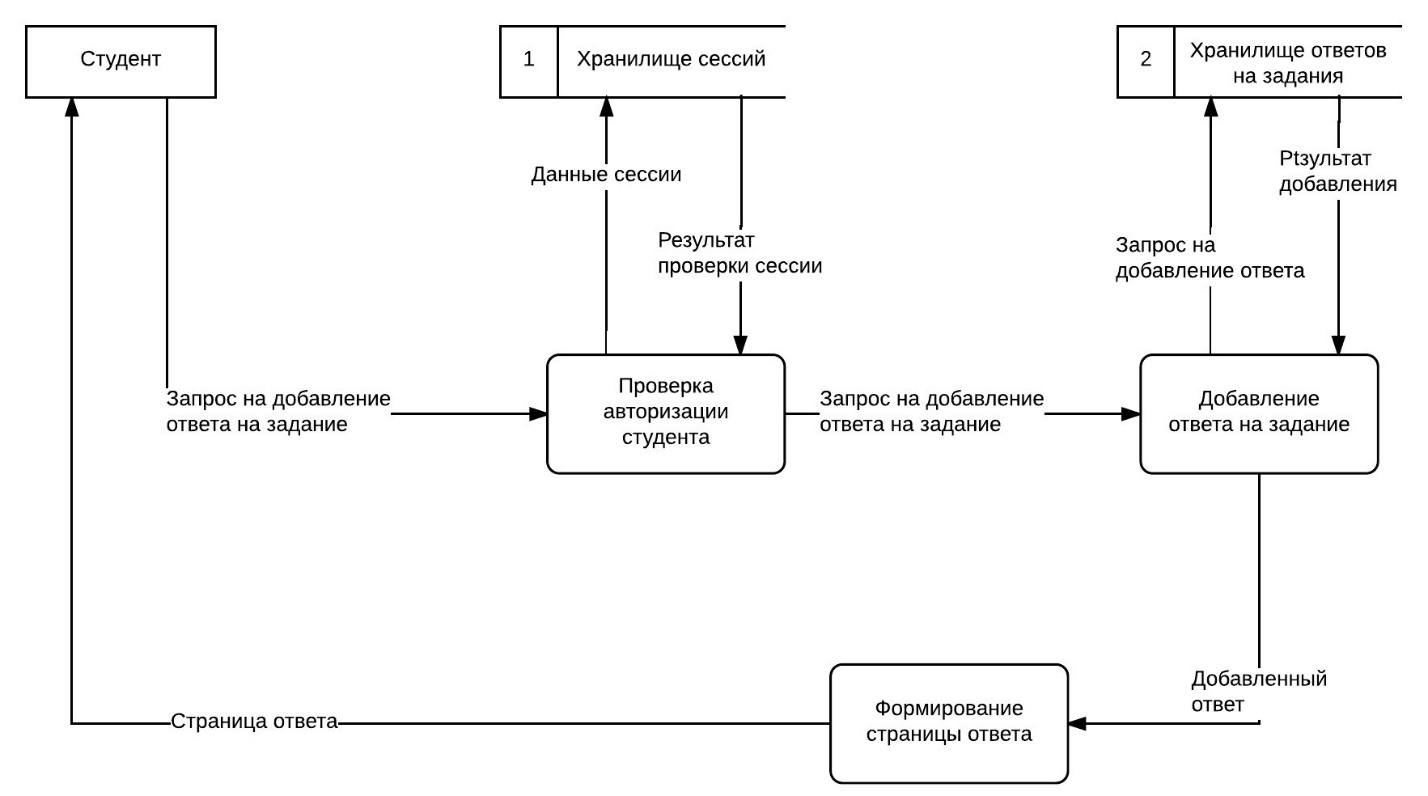


Рисунок 9. Диаграмма потоков данных при добавлении студентов ответа

# Физический дизайн

На рисунке 10 представлена диаграмма развёртывания.

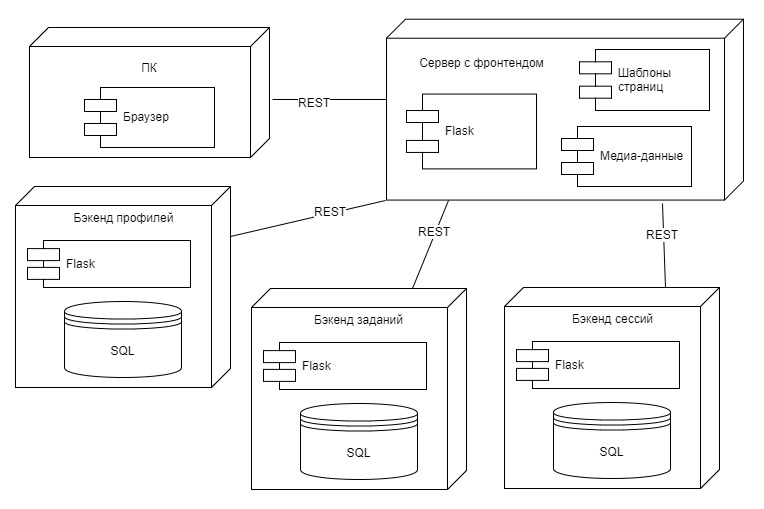


Рисунок 10. Диаграмма развёртывания системы

Исходя из того, что в техническом задании требуется, чтобы система была масштабируемой, необходимо предусмотреть возможность запуска серверов на отдельных машинах. На этапе разработки возможно использовать средства виртуализации для имитации запуска на отдельных машинах. Для этого каждая подсистема запускается в своём виртуальном окружении с собственным интерпретатором и набором зависимостей.

## Выбор ОС

Разрабатывать нужно в системе, виртуальные машины с которой предоставляются CI-сервисами и хостингами, чтобы было проще перенести проект с локальной машины. Обычно, это ОС Ubuntu, поэтому и разработка ведётся на ОС из семейства Ubuntu.

## Выбор СУБД

В виду отсутствия жёстких требований к хранению данных целесообразно использовать легковесную встраиваемую библотеку SQLite, которая компонуется с исполняемым процессом и предоставляет API для работы с БД. Данные хранятся в одном файле, что упрощает создание резервных копий. Для сервиса сессий используется SQLite In-Memory.

## Выбор языка программирования и фреймворков

Язык разработки должен быть совместим с ОС Linux, поддерживать работу с SQLite, а также иметь популярные фреймворки для разработки Web-приложений. Среди таких языков можно выделить Python, Java, Go. Исходя из того, что скорость разработки данной системы важнее, чем количество запросов в секунду, которые она способна обработать, языком для разработки выбран Python.

В языке Python есть два наиболее популярных фреймворка для разработки Web-приложений: Django и Flask. Для разработки приложений с микросервисной архитектурой целесообразнее использовать фреймворк Flask, т.к. от микросервисов требуется только хранить данные и обрабатывать приходящие HTML/REST запросы.

# Описание пользовательского интерфейса

Пользовательский интерфейс в разрабатываемой системе представляет собой Web-интерфейс, доступ к которому осуществляется через браузер.

Страница портала состоит из «навбара» (верхней части страницы, в которой находится логотип и верхнее меню со ссылками на основные разделы портала), основной части и «футера» (нижней части страницы, в которой обычно размещают ссылки на редко посещаемые, но необходимые, страницы, например, страницы с пользовательским соглашением).

Обобщенно структуру страниц портала можно представить следующим образом:

* Главная страница
* Страница уроков
  + Список уроков
* Страница урока
  + Задание
  + Список студентов, выполнивших задание
  + Ответ студента на задание
  + Оценка
* Страница профиля
* Страница авторизации
* Страница регистрации
* Личный кабинет

# Заключение

# Список литературы.

* + - 1. Вишневская Т.И., Романов Т.Н. Методология программной инженерии. Часть 1. – М: Изд-во МГТУ им. Н.Э. Баумана, 2015, 52с.
      2. Проектирование программного обеспечения. [Электронный ресурс] Режим доступа: <https://habrahabr.ru/post/74330/>
      3. Людоговский А. Общие положения IDEF. [Электронный ресурс] Режим доступа: http://dit.isuct.ru/ivt/books/CASE/case10/idef0/met1.htm
      4. Turnbull J. The Docker Book: Containerization is the new virtualization. 160th ed. - James Turnbull, 2014
      5. Xfce. [Электронный ресурс] Режим доступа: <https://ru.wikipedia.org/wiki/Xfce>
      6. SQLite. [Электронный ресурс] Режим доступа: <https://ru.wikipedia.org/wiki/SQLite>
      7. Python. [Электронный ресурс] Режим доступа: [www.python.su](http://www.python.su)
      8. Adrian Holovaty J.K.M. The Definitive Guide to Django: Web Development Done Right. 2nd ed. - Apress, 2009.
      9. Welcome to Flask. [Электронный ресурс] Режим доступа: <http://flask.pocoo.org/docs/0.12/>
      10. W3Schools Online Web Tutorials. [Электронный ресурс] Режим доступа: <https://www.w3schools.com/>