МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждения образования «БЕЛОРУССКИЙ
ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий
Кафедра Информационных систем и технологий
Специальность 1-40 05 01-03 Информационные системы и технологии (издательско-полиграфический комплекс)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту на тему:
«Веб приложение по организации онлайн-курсов»

Дипломник Прихач Евгений Александрович
(Ф.И.О.)

Руководитель проекта асс. О. А. Нистюк
(учен. степень, звание, должность, подпись, Ф.И.О.)

Заведующий кафедрой к. т. н., доц. В. В. Смелов
(учен. степень, звание, должность, подпись, Ф.И.О.)

Консультанты: асс. А. С. Соболевский
(учен. степень, звание, должность, подпись, Ф.И.О.)

Нормоконтролёр асс. О. А. Нистюк
(учен. степень, звание, должность, подпись, Ф.И.О.)

Дипломный проект защищен с оценкой

Председатель ГЭК к. т. н., доц. В. К. Дюбков
(учен. степень, звание, должность, подпись, Ф.И.О.)

Минск 2022

Учреждение образования «БЕЛОРУССКИЙ
ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий
Кафедра Информационных систем и технологий
Специальность 1-40 05 01-03 Информационные системы и технологии (издательско-полиграфический комплекс)

УТВЕРЖДАЮ
Заведующий кафедрой ИСиТ
_____ В.В. Смелов
«____» _____ 2021 г.

ЗАДАНИЕ
на дипломный проект студенту
Прихач Евгению Александровичу
(фамилия, имя, отчество)

1. Тема проекта: Веб приложение по организации онлайн-курсов
утверждена приказом по университету от «01» марта 2021 г. № 46-С
2. Срок сдачи законченного дипломного проекта: 07 июня 2021 г.
3. Функциональные возможности:
– авторизация и регистрация;
– просмотр и создание курсов;
– осуществление записи на курс;
– управление материалами курса;
– создание домашнего задания;
– оценивание домашнего задания;
– поиск курса по названию.
4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов):
1) реферат;
2) содержание;
3) введение;
4) раздел 1: обзор аналогичных решений и постановка задачи;
5) раздел 2: проектирование приложения;
6) раздел 3: разработка приложения;
7) раздел 4: тестирование приложения;
8) раздел 5: анализ информационной безопасности приложения;
9) раздел 6: руководство пользователя;
10) раздел 7: технико-экономическое обоснование проекта;
11) заключение;
12) список использованных источников;
13) приложения и графическая часть;

5. Перечень графического материала (с точным указанием обязательных чертежей):
1) диаграмма вариантов использования;
2) логическая схема базы данных;
3) диаграмма архитектуры приложения;
4) таблица экономических показателей;
6. Консультанты по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант
Теоретическая часть	
Проектирование и разработка	
Экономический раздел	

7. Дата выдачи задания: 01 марта 2021 г.

Руководитель_____О.А. Нистюк
(подпись)
Задание принял к исполнению _____Е.А. Прихач
(подпись)

Календарный план

№ п/п	Наименование этапов дипломного проекта	Срок выполнения этапов проекта	Примечание
1	*обзор аналогов, анализ информационной безопасности*	*22.03.2022 г.*	
2	*Постановка задачи и разработка функциональных требований*	*10.03.2022 г.*	
3	*Проектирование архитектуры программы (разработка структуры системы, алгоритмов и схемы данных)*	*27.03.2022 г.*	
4	*Реализация программы*	*19.04.2022 г.*	
5	*Тестирование и отладка*	*03.05.2022 г.*	
6	*Выполнение расчетов экономического раздела*	*09.05.2022 г.*	
8	*Оформление пояснительной записки и графического материала*	*07.06.2022 г.*	

Дипломник _____ Руководитель проекта_____
(подпись) (подпись)

Содержание

Введение 5
1 Анализ и сравнительный обзор аналогов 6
1.1 Постановка задачи 6
1.2 Обзор аналогичных решений 6
1.2.1 Stepik 6
1.2.2 GeekBrains 8
1.2.3 Skillbox 8
1.3 Выводы по разделу 9
2 Проектирование программного продукта 10
2.1 Диаграмма вариантов использования 10
2.2 Описание используемых технологий 12
2.2.1 Описание языка программирования и технологий для разработки серверной части приложения 12
2.2.2 Описание базы данных 13
2.2.3 Описание языка программирования и фреймворка для разработки клиентской части приложения 13
2.2.4 Описание интегрированной среды разработки 14
2.2.5 Контейнеризация 15
2.3 Проектирование архитектуры информационной системы 16
2.4 Проектирование серверной части приложения 17
2.5 Проектирование клиентской части приложения 17
2.6 Проектирование сущностей базы данных 18
2.7 Вывод по разделу 21
Список использованных источников 22
ПРИЛОЖЕНИЕ А 23
ПРИЛОЖЕНИЕ Б 24

Введение

За последние несколько десятилетий информационные технологии проникли во все области нашей жизни. Появление доступных персональных компьютеров, сети Интернет и клиент-серверных технологий оказали огромное влияние на людей. Наличие необходимых навыков и знаний - важнейшая составляющая на пути становления человека как специалиста.

Современные проекты требуют наличия разнообразных знаний, их выполнение связано с большим количеством рисков: необходимость уложиться в заданные временные рамки, обеспечить качество. Поэтому к сотрудникам предъявляются высокие требования.

Отслеживание собственного прогресса в обучении – одна из главных стратегий, ведь в первую очередь это говорит о собственной компетентности. Необходимо отслеживать свой рост, чтобы выпускаемые продукты были качественными, ведь это самое главное для клиентов.

Данная стратегия подразумевает под собой постоянное развитие и самообучение, что позволяет конкурировать за высшие должностные позиции и, что не мало важно, иметь высокий уровень оценивания собственного труда.

Целью дипломного проекта является создание веб-приложения по осваиванию новых навыков от какого бы ни было языка до фундаментальных наук и имеет следующие возможности:

–сохранение рабочей информации в централизованной базе данных;

–составление материалов курса;

–возможность просмотра личного профиля;

–назначение домашних заданий;

–возможность личной оценки д/з;

–комментировать выполненные задания.

Для достижения цели были поставлены следующие задачи:

–спроектировать и реализовать сущности базы данных;

–обеспечить взаимодействие модуля с базой данных;

–реализовать Back-end часть;

–реализовать Front-end часть;

–спроектировать и реализовать создание и изменения материалов курса, а также добавление студентов и/или других преподавателей;

–разработать необходимые сервисы для работы модуля;

–произвести тестирование программного средства;

1 Анализ и сравнительный обзор аналогов

Перед началом реализации дипломного проекта стоит проанализировать существующие аналоги на рынке, чтобы сформулировать окончательные требования к проектируемому модулю.

1.1 Постановка задачи

Одним из двух элементов научного познавательного процесса, который состоит из постановок и решений, является конкретное решение, которое, в свою очередь, тоже может быть представлено в виде процесса из постановок задач.

Постановка задачи осуществляется в понятиях и терминах какой-либо области науки, поэтому она опирается на весь предшествующий научный опыт, в том числе на опыт, позволяющий выбрать область науки для решения задачи.

Целью дипломного проекта является разработка веб-приложения, с помощью которого пользователь сможет произвести обучение любым представленным дисциплинам.

Модуль должен взаимодействовать с централизованной базой данных, обеспечивающей хранение пользовательской информации.

Взаимодействие между Front-end и Back-end частью должно базироваться на клиент-серверной архитектуре, связь должна осуществляться по протоколу HTTP.

Веб-приложение должно обладать следующими задачами:

–сохранение рабочей информации в централизованной базе данных;

–составление материалов курса;

–возможность просмотра профиля;

–назначение домашних заданий;

–возможность получения оценки и ее обсуждение;

–просмотр материалов и отправка решений выполненных заданий.

Так же модуль обладает бизнес-целями. Так как проект создается для внутреннего пользования компании, то одной из целей дипломной работы является внедрение модуля в рабочий процесс компании.

1.2 Обзор аналогичных решений

1.2.1 Stepik

Первым рассматриваемым приложением среди конкурентов является stepic. Войдя на сайт, первым делом наше внимание акцентируется на предложение о просмотре каталога, на втором плане расположены варианты использования данного ПО, а именно:

–в роли преподавателя;

–в роли студента.

Главная страница конкурентного веб-приложения продемонстрирована на рисунке 1.1.

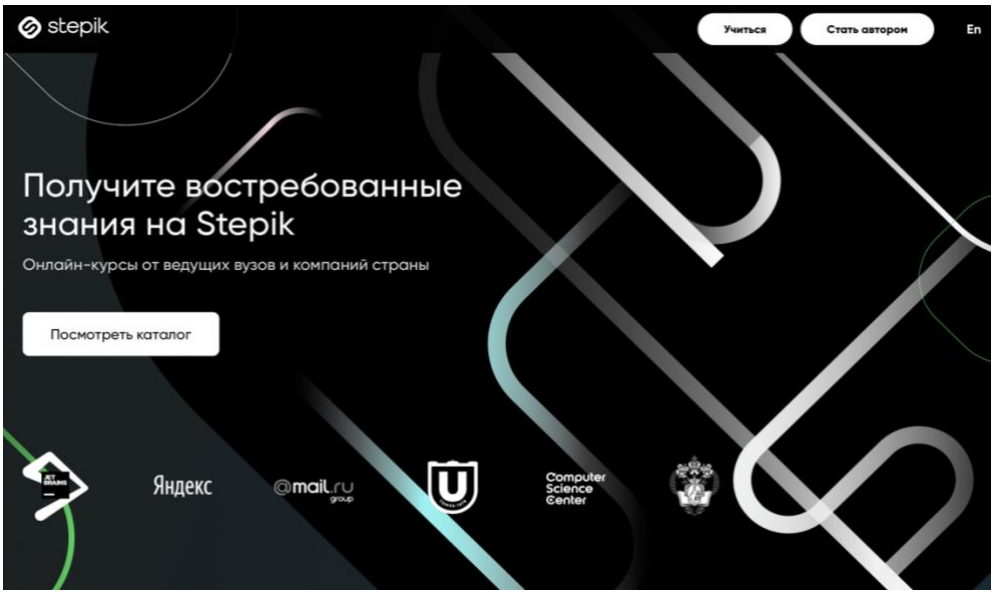


Рисунок 1.1 – Главная страница Stepic[1].

Далее сайт нам предлагает выбрать курс на который мы хотим записаться, как показано на рисунке 1.2.

Мы можем выбрать любой из перечисленных курсов, а также, если он платный, опробовать бесплатный урок для определения того, насколько приведенная в нем информация актуальна.

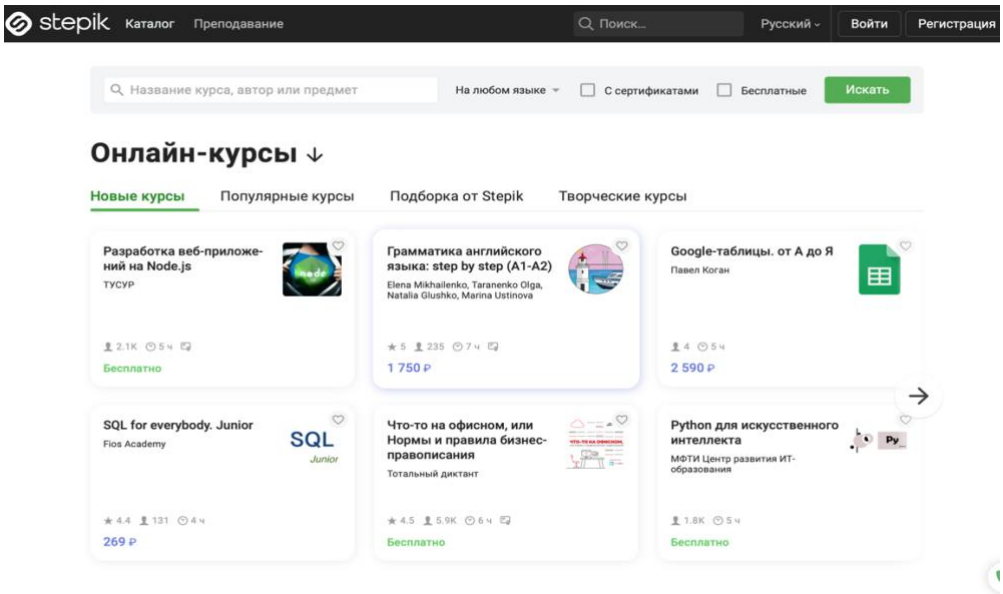


Рисунок 1.2 – Страница с курсами Stepic.

Мы можем выбрать любой из перечисленных курсов, а также, если он платный, опробовать бесплатный урок для определения того, насколько

приведенная в нем информация актуальна. Данное веб-приложение больше подходит для рядового пользователя и имеет удобный интерфейс, также оно является условно-бесплатным предоставляя доступ к наиболее актуальным курсам за отдельную плату и не позволяет проводить частные курсы для сотрудников своей компании.

Ввиду этого, стоит отметить, что рассматриваемое нами приложение не подойдет для наших целей из-за ограниченного функционала.

1.2.2 GeekBrains

Следующий рассматриваемый аналог – приложение GeekBrains[2]. Чтобы воспользоваться услугами данного веб-приложения необходимо зарегистрироваться, что предоставит нам демо-версию к перечисленным на рисунке 1.3 курсам.

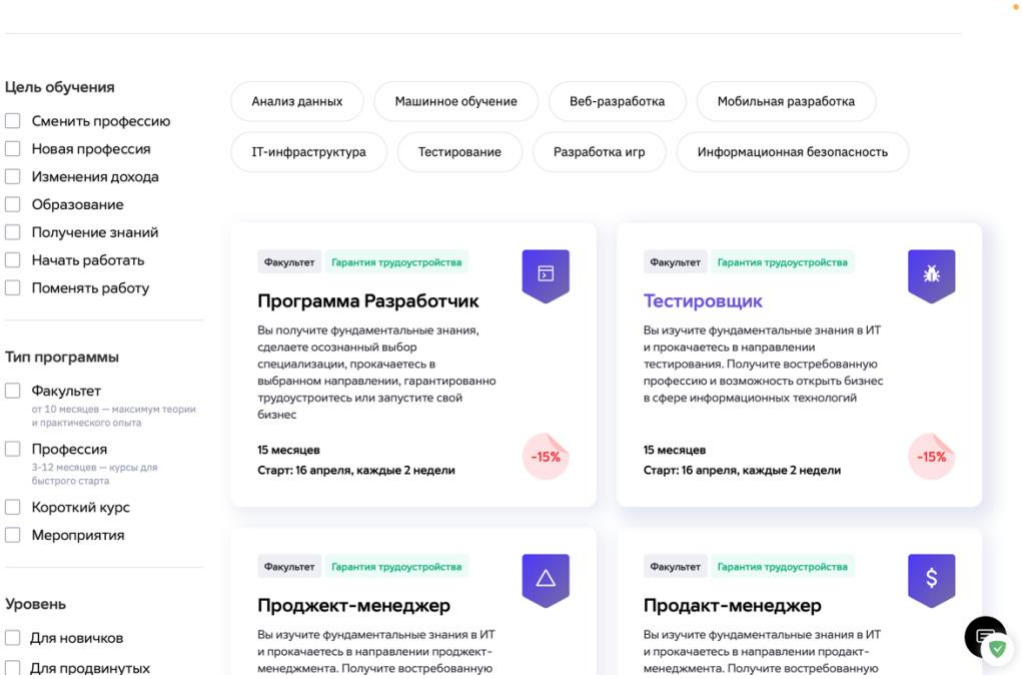


Рисунок 1.3 – перечень курсов GeekBrains.

На данном ресурсе присутствуют курсы только для IT сферы и имеет менее дружелюбный интерфейс, чем приведенный выше аналог. Для данного ресурса характерны все недостатки из предыдущего ресурса и так же не подходит для наших задач.

1.2.3 Skillbox

И последним проектом среди аналогов будет приложение Skillbox[3]. Сайт представляет собой перечень готовых курсов и направлений, которые может выбрать для себя пользователь, имея при этом один из наиболее дружелюбных пользователю интерфейсов. На выбор предлагается более 400 курсов на тематику

IT сферы. Данное веб-приложение ориентировано на одиночного пользователя и имеет гибкую ценовую политику, позволяющую отсрочить платежи более чем на полгода. С самого знакомства с данным приложением нам демонстрируют перечень направлений, по которым пользователь может найти наиболее релевантный для себя курс, что продемонстрировано на рисунке 1.4.

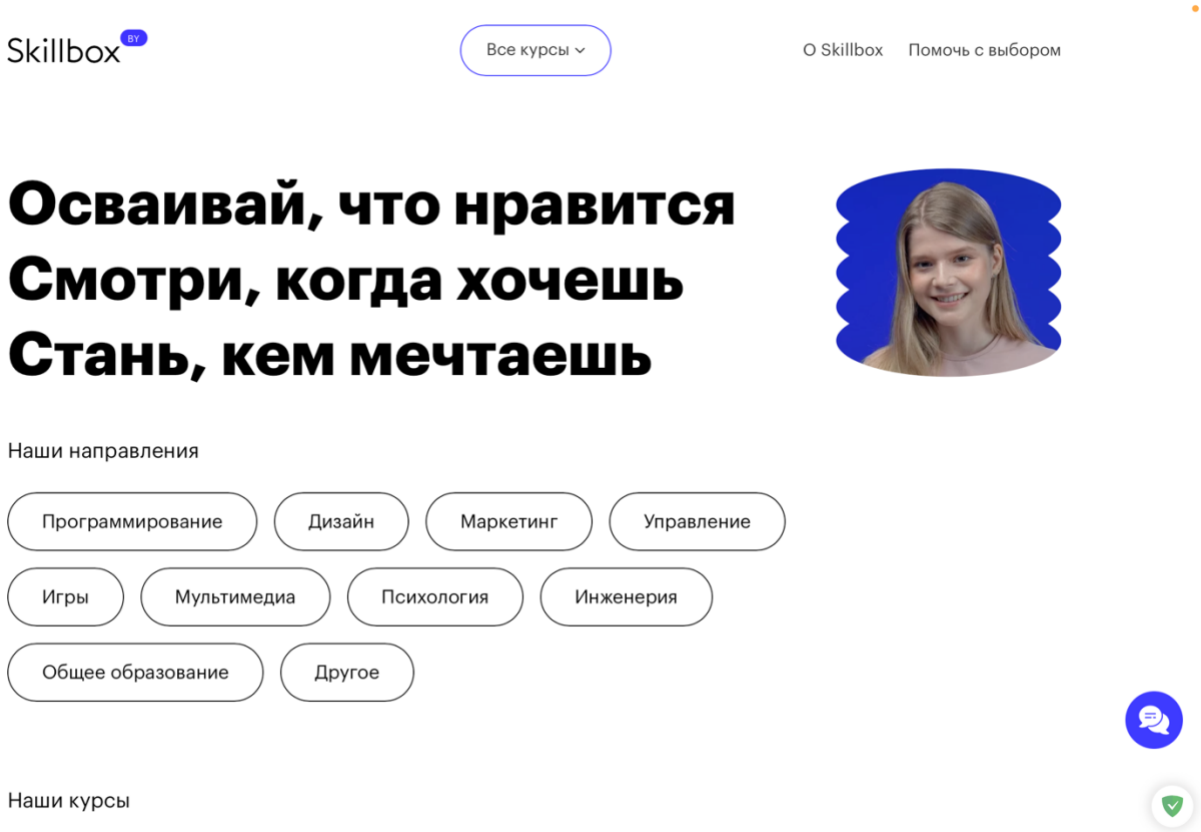


Рисунок 1.4 – Главная страница Skillbox.

Подводя итог по данном приложению, можно смело сказать, что оно приятное, эффективное, обладает необходимым функционалом для углубления либо приобретения знаний в сфере IT. Однако приложение необходимым нам функционалом.

1.3 Выводы по разделу

В данном разделе были подробно описаны и исследованы плюсы и минусы аналогов разрабатываемого программного модуля.

Как видно из представленного краткого обзора, большинство ресурсов имеют много общего в своих функциональных возможностях. Это касается базовой функции – онлайн обучения.

Одновременно с этим все программные средства отличаются между собой как внешне, так и целями их использования

Также отметим, что на основании использования приложений-аналогов были выработаны основные критерии для построения приложения, позволяющего осуществлять наиболее простое и интуитивное его использование.

2 Проектирование программного продукта

При разработке программного средства важно определить необходимый функционал приложения, а также перед началом самой разработки программного продукта установить требуемые программные средства. Поэтому, самым первым этапом в процессе разработки приложения является этап выбора и настройки среды для программного продукта.

Далее идет этап проектирования базы данных проекта. Благодаря этому этапу можно будет также смотреть, в какой инфраструктуре это приложение можно развернуть и где оно будет работать нормально.

Следующим этапом станет выбор подходящего языка и фреймворка для написания Back-end части приложения, или же веб-сервиса. Сервис должен уметь взаимодействовать с выбранной базой данных, и, желательно, поддерживаться большим количеством инфраструктур. Также он должен быть оптимизированным и функциональным для возможности реализации всей требуемой логики модуля.

Последним этапом является реализация Front-End части приложения. Под Front-End частью подразумевается проектирование всех графических элементов на сайте, с которыми может взаимодействовать пользователь, а также написание логики самого сайта.

Стоит также отметить, что для упрощения разработки приложения должна была добавлена возможность виртуализации или контейнеризации, чтобы можно было легко и быстро воссоздать среду окружения, в котором будет размещено приложение. В качестве этой возможности было выбрано приложение с поддержкой контейнеризации Docker.

2.1 Диаграмма вариантов использования

Диаграмма вариантов использования является исходным концептуальным представлением системы в процессе ее проектирования и разработки. Данная диаграмма состоит из актеров, вариантов использования и отношений между ними.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества актеров, взаимодействующих с системой с помощью так называемых вариантов использования. При этом актером называется любой объект, субъект или система, взаимодействующая с моделируемой системой извне.

В свою очередь вариант использования – это спецификация сервисов или функций, которые система предоставляет актеру.

Другими словами, каждый вариант использования определяет некоторый набор действий, совершаемых системой при взаимодействии с актером. При этом в модели никак не отражается то, как будет реализован этот набор действий.

В приложении представлено 2 роли, которые показаны в таблице 2.1.

Таблица 2.1 – Роли пользователей в приложении

Роль	Описание
Студент	Базовый пользователь, имеет доступ к курсам, к которым его пригласили, созданию и изменению решенных д/з, а также возможность создания комментариев .
Преподаватель	Пользователь имеющий права на создание и изменение материалов курсов, создание и выставление оценок д/з, имеет возможность создания комментариев к прикрепленным студентами решениям.

Диаграмму вариантов использования для дипломного проекта можно увидеть в приложении А. На данной диаграмме есть 2 роли, описанные в таблице 2.1.

Студент – слушатель курса, имеет возможность просматривать информацию о себе и о доступных курсах, высылать решения поставленных преподавателем заданий и обсуждать выставленную оценку с преподавателем.

Преподаватель – ключевая роль во всем приложении. Генерирует содержимое курсов, ему предоставлены все возможности по проведению онлайн курсов.

2.2 Описание используемых технологий

При разработке приложения необходимо выбрать соответствующие технологии для серверной и клиентской частей. От выбора языка программирования, фреймворка, интегрированной среды разработки, системы управления базами данных и других вспомогательных инструментов будут варьироваться сложность.проекта, сроки его реализации, а также возможности, которые можно реализовать.

2.2.1 Описание языка программирования и технологий для разработки серверной части приложения

В качестве Backend части в приложении будет выступать REST API, который реализован с помощью Django Rest Framework.

Django Rest Framework построен на основе кроссплатформенной среды Python. Который может быть развернут на всех популярных операционных системах от MacOS до Android и, зачастую, является предустановленным.

На данный рассматриваемый момент самой последней версией языка является Python 3.9.11.

Язык Python является высокоуровневым объектно-ориентированным языком программирования, поэтому он подходит для быстрого конструирования различных компонентов – от высокоуровневой бизнес-логики до машинного обучения. Также следует отметить, что при помощи Python можно создавать различные веб-

приложения. С помощью простых встроенных конструкций языка существует возможность переводить созданные компоненты в веб-сервисы, к которым можно будет обращаться из сети Интернет, используя любой язык на любой операционной системе. Также можно создать полноценные веб-приложения, которые сразу предоставляют интерфейс пользователю. Удобные предоставляемые компаниями методы для разработки веб-приложений позволяют программистам, владеющим навыками объектно-ориентированного программирования, быстро освоиться в разработке веб-приложений.

2.2.2 Описание базы данных

Чтобы хранить данные, нам естественным образом нужна база данных. Для работы с базой данных выбрана популярная реляционная система управления базами данных PostgreSQL.

PostgreSQL - свободная объектно-реляционная система управления базами данных (СУБД). Разработана как open-source проект и написана на C.

Для осуществления связи между базой данных и приложением необходим посредник. И именно таким посредником будет является технология Django ORM. Django ORM представляет собой объектно-ориентированную технологию для работы с данными.

Django ORM - представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от используемого типа хранилища.

Центральной концепцией Entity Framework Core является понятие сущности. Сущность определяет набор данных, которые связаны с определенным объектом. Поэтому данная используемая нами технология предполагает работу не с таблицами, а с объектами и их коллекциями.

2.2.3 Описание языка программирования и фреймворка для разработки клиентской части приложения

Выбор фреймворка, для реализации клиентской части был сделан в пользу Angular, поскольку он является мощным инструментом, использующим компонентный подход, что облегчает его освоение людям, знающим ООП. Одной из ключевых особенностей Angular является то, что он использует в качестве языка программирования TypeScript[4].

TypeScript – это язык программирования, расширяющий существующие возможности базового JavaScript и имеющий следующие преимущества:

- аннотации типов и проверка их согласования на этапе компиляции;
- вывод типов;
- использование интерфейсов;
- использование перечисляемых типов;
- более гибкая система для управления и взаимодействия с классами;

– более удобный синтаксис для объявления анонимных функций.

TypeScript реализует многие концепции, которые свойственны объектно-ориентированным языкам, как, например, наследование, полиморфизм, инкапсуляция и модификаторы доступа и так далее.

Потенциал TypeScript позволяет быстрее и проще писать большие сложные комплексные программы, соответственно их легче поддерживать, развивать, масштабировать и тестировать, чем на стандартном JavaScript.

Для создания компонентов при разработке интерфейса используется Angular Material Design[5]. Material Design – это язык дизайна для веб-приложений и мобильных приложений, который был разработан Google в 2014 году. Использование Material Design упрощает разработчикам настройку UI, сохраняя при этом удобный интерфейс используемых приложений.

2.2.4 Описание интегрированной среды разработки

Для любого разработчика важна среда разработки, от которой зависит скорость и удобство написания программного средства. Выбор интегрированной среды разработки, в свою очередь, зависит от используемой технологии. Для разработки на языке программирования Python существует ряд решений Visual studio code, Pycharm, Eclipse. Рассмотрим наиболее популярный из них.

Pycharm - среда разработки Intellij IDEA.

Плюсами данной среды разработки являются:

– умный редактор PyCharm предназначен для максимально продуктивной разработки на Python, JavaScript, CoffeeScript, TypeScript, CSS и популярных языках шаблонов. Функции автодополнения, обнаружения ошибок и быстрые исправления учитывают особенности каждого из поддерживаемых языков.

– умный поиск позволяет быстро перейти к любому классу, файлу или символу, а также к нужному окну или действию IDE. Переход к вышестоящему методу, тесту, объявлению, вхождению или реализации осуществляется в одно нажатие.

– PyCharm предоставляет широкие возможности реорганизации кода с помощью рефакторингов Rename и Delete, Extract Method, Introduce Variable, Inline Variable, Inline Method и многих других. Рефакторинги учитывают особенности конкретного языка или фреймворка, помогая вносить изменения по всему проекту.

– большое количество встроенных условий рефакторинга кода;

– поддержка нескольких запущенных программ;

– кроссплатформенность;

– удобная система контроля версий;

– большое количество разнообразных плагинов.

Также существует ряд минусов:

– наиболее актуальная версия для Backend разработчиков, а именно Professional – платная;
– большое потребление ресурсов пк.

2.2.5 Контейнеризация

Перед стартом разработки приложения стоит также обратить внимание на необходимую инфраструктуру, которая понадобиться в самом процессе разработке и в дальнейшем для развертывания приложения: это сервер базы данных, сервер для развертывания Front-End и Back-End части. Все это можно разместить на одной физической машине и на физической операционной системе, однако, если в процессе разработки будут участвовать несколько человек, а само приложение будет разворачиваться на популярных облачных сервисах, таких как AWS, Azure, то могут возникнуть проблемы с настройкой инфраструктуры. Чтобы избежать всех этих проблем можно воспользоваться возможностью контейнеризации.

Виртуализация – разновидность виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного. Здесь в качестве базовых элементов выступают контейнеры – приложения, которые потребляют ресурсы конкретно физической ОС, представляют собой изолированное пространство с необходимой для работы средой исполнения. Широкое распространение данная технология получила на семействе ОС Unix-типов, которые преимущественно используются на облачных сервисах для развертывания пользовательских приложений.

Широкое распространение данная технология получила на семействе ОС Unix-типов, которые преимущественно используются на облачных сервисах для развертывания пользовательских приложений.

Виртуализация – такой процесс, в котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного. Здесь в качестве базовых элементов выступают контейнеры – приложения, которые потребляют ресурсы конкретно физической ОС, представляют собой изолированное пространство с необходимой для работы средой исполнения. Виртуализация приложений позволяет запускать отдельное приложение в своей собственной изолированной среде. Такой способ помогает решить множество проблем. А именно безопасность: приложение, запущенное в изолированной среде – не способно нанести вред ОС и другим приложениям.

На данный момент среди самых популярных средств контейнеризации является Docker. Данное средство занимает лидирующую позицию на рынке и интегрировано в облачные сервисы. Оно кроссплатформенно, это означает, что приложение, имеющее поддержку Docker, сможет быть запущено на любой ОС с предустановленным Docker-ом. Помимо обычных средств контейнеризации Docker включает в себя поддержку оркестрации контейнеров, что означает, что можно получить контроль сразу над несколькими контейнерами, управлять их запуском и так далее.

Для корректной работы Docker необходимо подготовить предварительные скрипты для развертывания приложения.

2.3 Проектирование архитектуры информационной системы

Архитектура информационной системы – это структура системы, которая включает программные компоненты, видимые снаружи свойства этих компонентов, а также отношения между компонентами информационной системы.

Для программного средства в дипломном проекте был выбран один из самых распространенных способов разработки программных модулей – клиент-серверная модель, или клиент-серверная архитектура.

В данной архитектуре все задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Используя такой подход, приложение разбивается на 2 части.

Клиентская часть приложения – это графический интерфейс. Графический интерфейс отображается в браузере. Пользователь взаимодействует с приложением именно через браузер, кликая по ссылкам и кнопкам.

Серверная часть веб-приложения – программа на сервере, обрабатывающая запросы пользователя. При каждом переходе пользователя по ссылке браузер отправляет запрос к серверу. А сервер в свою очередь отправляет клиенту ответ, при этом клиент, при запросе на сервер, может передавать параметры вместе с запросом.

Сервер обрабатывает этот запрос, вызывая некоторый скрипт, который формирует веб-страничку, описанную языком HTML, и отсылает клиенту по сети. Браузер тут же отображает результат в виде веб-страницы, которую далее браузер интерпретирует и показывает пользователю.

База данных располагается на сервере. Серверная часть веб-приложения обращается к базе данных, извлекая данные, которые необходимы для формирования страницы, запрошенной пользователем.

В самой базе располагаются различные сущности, которые связаны друг с другом посредством связей первичный-внешний ключ.

Предполагаемая структура состоит из сервера, клиента и базы данных, Обращение к серверу происходит посредством API, а в качестве транспорта данных используется формат передачи данных JSON.

Общая схема взаимодействия клиента и сервера представлена на рисунке 2.1

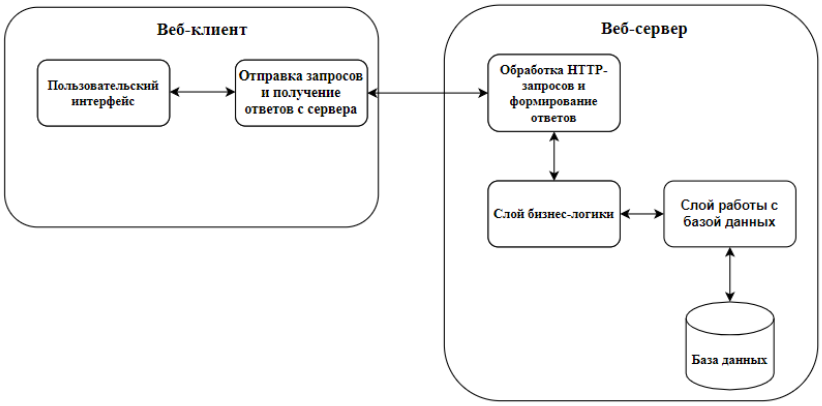


Рисунок 2.1 – Схема взаимодействия компонентов приложения

Разделение на клиент-серверную архитектуру дает возможность заниматься настройкой каждого компонента отдельно и параллельно, что может значительно ускорить работу программного средства.

2.4 Проектирование серверной части приложения

При разработке программного средства по теме дипломного проекта будет применятся трехуровневая архитектура приложения.

Существует достаточное количество различных видов и типов архитектур, но была выбрана трехуровневая, как самая распространенная и наиболее используемая в производстве, а главное, удобная для дальнейшего расширения приложения.

Django использует шаблон MVC и его можно описать следующим образом:

–model – доступ к хранилищу данных;

–view – интерфейс, с которым взаимодействует пользователь;

–controller – связывающий model и view объект.

К преимуществам трехуровневой архитектуры можно отнести:

– масштабируемость программного средства;

– простота конфигурации, за счет того, что можно изменять конфигурацию одного уровня, а не всего программного средства;

–простота тестирования, за счет того, что программное средство разделено на три уровня, можно провести тесты отдельно для каждого элемента, при этом, не отвлекаясь на другой компонент.

2.5 Проектирование клиентской части приложения

При создании клиентской части приложения было принято решение об использовании архитектуры SPA[6] (Single-Page Application) – «одностраничное приложение». Данная архитектура является новым витком в развитии клиентской веб-разработки, представляя более удобный интерфейс для пользователя и преимущества реализации для разработчиков.

Основная логика SPA заключается в том, что приложение является одностраничным по факту. Это значит, что сервер при стартовой загрузке отдает единствен-

ную HTML страницу, которая часто бывает полностью пустой. А все содержимое отрисовывается с помощью JavaScript. Данный способ взаимодействия позволяет практически мгновенно загрузить статическую страницу и отобразить индикатор прогрузки, отрисовывая за ним все необходимые элементы.

Главным преимуществом для пользователя является отказ от стандартных перезагрузок при навигации по сайту. Содержимое приложения изменяется динамически, без перезагрузки, позволяя добавлять новый функционал, независимый от текущей страницы приложения.

Для разработчиков же использование данной архитектуры представляет собой преимущество в разделении. Клиентский код полностью отделен от серверного и, зачастую, реализуется на совершенно ином языке программирования. Это позволяет разрабатывать части приложения независимо друг от друга.

Одним из наиболее популярных фреймворков для построения SPA является Angular, написанный на языке программирования TypeScript и реализующий данный вид архитектуры так же на этом языке.

Angular представляет фреймворк от компании Google для создания клиентских приложений. Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений. В этом плане Angular является наследником другого фреймворка AngularJS. В то же время Angular это не новая версия AngularJS, а принципиально новый фреймворк. Angular предоставляет такую функциональность, как двустороннее связывание, позволяющее динамически изменять данные в одном месте интерфейса при изменении данных модели в другом, шаблоны, маршрутизация.

2.6 Проектирование сущностей базы данных

Одним из ключевых моментов при проектировании и создании сущностей базы данных является грамотный анализ предметной области приложения. Как следствие – составление такой модели данных, которая будет правильно отражать то, как с этими данными и этой моделью подразумевается взаимодействовать.

В результате были созданы 9 сущностей базы данных и 2 роли, которые удовлетворяют поставленным перед нами требованиям.

Таблица User содержит информацию о пользователях.

Описание ее полей представлено в таблице 2.2.

Таблица 2.2 – Структура таблицы User

Название поля	Тип данных	Описание
id	uniqueidentifier	Уникальный идентификатор пользователя
username	Nvarchar(30)	Пользовательский логин
password	Nvarchar(30)	Пароль
email	Nvarchar(30)	Электронная почта
User_status	Nvarchar(1)	Статус пользователя

IsActive	boolean	Проверка на активность пользователя

Таблица Course служит для хранения курсов.

Описание ее полей представлено в таблице 2.3.

Таблица 2.3 – Структура таблицы «Course»

Название	Тип	Описание
Id	uniqueidentifier	Идентификатор курса
name	Nvarchar(100)	Название курса
description	text	Описание курса
teachers	uniqueidentifier	Идентификаторы учителей
students	uniqueidentifier	Идентификаторы студентов

Таблица Teacher_User служит для связи многие ко многим между полем teachers в таблице Course и полем id в таблице User.

Описание ее полей представлено в таблице 2.4.

Таблица 2.4 – Структура таблицы «Teachers_User»

Название	Тип	Описание
Teachers	uniqueidentifier	Идентификатор поля учителей
User	uniqueidentifier	Идентификатор пользователя

Таблица Students_User служит для связи многие ко многим между полем Students в таблице Course и полем Id в таблице User.

Описание ее полей представлено в таблице 2.5.

Таблица 2.5 – Структура таблицы «Students_User»

Название	Тип	Описание
Students	uniqueidentifier	Идентификатор поля студентов
User	uniqueidentifier	Идентификатор пользователя

Таблица Lecture хранит лекционные материалы.

Описание ее полей представлено в таблице 2.6

Таблица 2.6 – Структура таблицы «Lecture»

Название	Тип	Описание
Id	uniqueidentifier	Идентификатор лекции
theme	nvarchar(100)	Тема лекции
presentation	blob	Файл с презентацией
Course	uniqueidentifier	Идентификатор курса

Таблица Homework хранит подготовленное преподавателем д/з.
Описание ее полей представлено в таблице 2.7.

Таблица 2.7 – Структура таблицы «Homework»

Название	Тип	Описание
Id	int	Идентификатор д/з
Homework_theme	nvarchar(70)	Тема д/з
Homework_text	text	Описание задания
Lecture	uniqueidentifier	Идентификатор лекции

Таблица Homework_solution служит для хранения выполненных д/з.
Описание ее полей представлено в таблице 2.8.

Таблица 2.8 – Структура таблицы «Homework_solution»

Название	Тип	Описание
Id	uniqueidentifier	Идентификатор выполненного д/з
solution	blob	Файл с решением
task	uniqueidentifier	Идентификатор задачи
student	uniqueidentifier	Идентификатор студента, отправившего решение

Таблица TaskMark служит для оценивания выполненного пользователем д/з.
Описание ее полей представлено в таблице 2.9.

Таблица 2.9 – Структура таблицы «TaskMark»

Название	Тип	Описание
id	uniqueidentifier	Идентификатор оценки
mark	smallint	Оценка решение
SolutionId	uniqueidentifier	Идентификатор решения

Таблица MarkComment служит для хранения комментариев к оценке.
Описание ее полей представлено в таблице 2.10.

Таблица 2.10 – Структура таблицы «MarkComment»

Название	Тип	Описание
ID	uniqueidentifier	Идентификатор комментария
Comment_author	uniqueidentifier	Идентификатор пользователя, оставившего коментарий
comment	text	Коментарий к оценке
mark	uniqueidentifier	Идентификатор отметки, к которой оставляется коментарий

Структурную схему базы данных можно найти в приложении Б.

2.7 Вывод по разделу

В рамках данного раздела была разработана диаграмма вариантов использования, в которой описан основной функционал разрабатываемого приложения. Также была продемонстрирована общая архитектура программного средства и дано краткое описание серверной и клиентской частей приложения. В данном разделе также была показана модель базы данных с обоснованием всех таблиц и подробным описанием полей каждой таблицы.

Список использованных источников

1 Веб-приложение «Stepic» [Электронный ресурс] – Режим доступа: https://stepik.org – Дата доступа: 20.03.2022.

2 Веб-приложение «GeekBrains» [Электронный ресурс] – Режим доступа: https://gb.ru/ – Дата доступа: 20.03.2022

3 Веб-приложение «Skillbox» [Электронный ресурс] – Режим доступа: https://skillbox.by– Дата доступа: 20.03.2022.

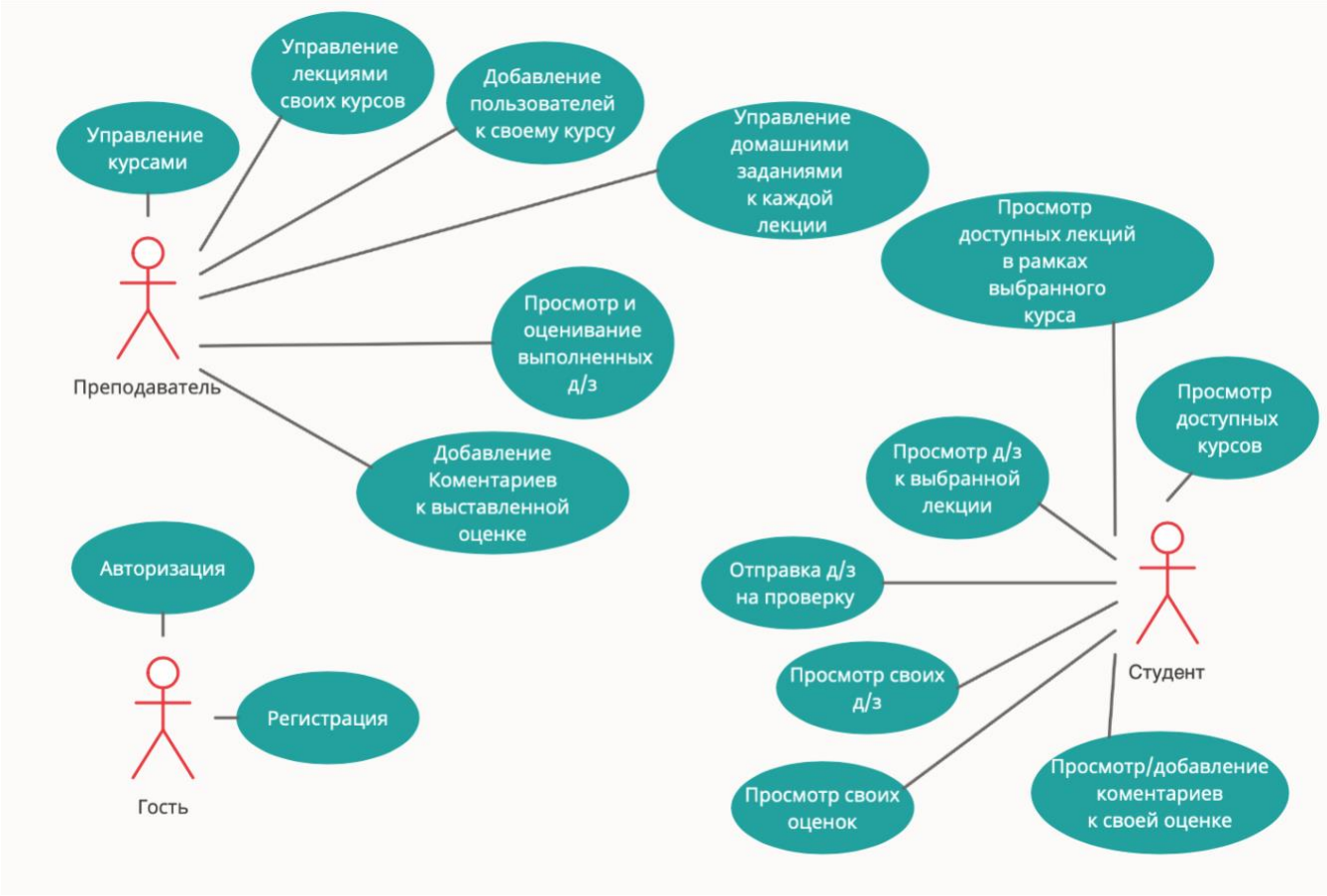
4 Angular [Электронный ресурс] – Режим доступа: https://angular.io/docs – Дата доступа: 20.03.2022.

5 Getting Started with Angular Material [Электронный ресурс] – Режим доступа: https://material.angular.io/guide/getting-started – Дата доступа: 20.03.2022.

6 What Is a Single Page Application? [Электронный ресурс] – Режим доступа: https://www.bloomreach.com/what-is-a-spa.html – Дата доступа: 20.03.2022.

ПРИЛОЖЕНИЕ А

Диаграмма вариантов использования



ПРИЛОЖЕНИЕ Б

Логическая модель базы данных