

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
Факультет компьютерных наук
Департамент программной инженерии**

**Клиент-серверное iOS приложение Ассистента Студента
Итоговый документ**

Исполнители:



/Н.Д. Зубарева/БПИ195



/А.О. Мостачев/БПИ196



/В.А. Поволоцкий/БПИ196

/А.Д. Сальникова/БПИ196

«26» марта 2023 г.

Заказчик:



/А.А. Паринов/
НИУ ВШЭ, Международная лаборатория
интеллектуальных систем и структурного анализа

«26» марта 2023 г.

П	
о	
д	
н	
.	
и	
д	
а	
т	
а	
И	
н	
в	
.	
№	
д	
у	
б	
л	
.	
В	
з	
а	
м	
.	
и	
н	
в	
.	
№	
П	
о	
д	
н	
.	
и	
д	
а	
т	
а	

<i>И</i>	
<i>н</i>	
<i>в</i>	
<i>.</i>	
<i>№</i>	
<i>п</i>	
<i>о</i>	
<i>д</i>	
<i>л</i>	

2023

Содержание

1. Введение.....	5
1.1. Наименование программы.....	5
1.2. Краткая характеристика области применения.....	5
2. Команда.....	5
2.1. Командные роли и распределение ответственности.....	5
2.2. Матрица компетенций.....	5
2.3. Матрица ответственности.....	6
3. Управление проектом.....	7
3.1. Метрики проекта.....	7
3.2. План работ.....	7
3.3. Использование инструментов (Canvas) или систем управления проектом (Trello, Jira).....	8
3.4. Анализ рисков.....	9
4. Предметная область и описание продукта.....	11
4.1. Постановка проблемы.....	11
4.2. Анализ предметной области, рамки и границы проекта.....	12
4.3. Назначение приложения.....	12
4.4. Новые предоставляемые функции.....	12
5. Описание потенциальных пользователей.....	13
5.1. Цель заказчика и ключевые стейкхолдеры (матрица Управления стейкхолдерами).....	13
5.2. План управления коммуникациями.....	13
6. Анализ конкурентов.....	14
7. Функциональные требования.....	15
7.1. Подключение к системе.....	15
7.2. Требования к составу выполняемых функций.....	15
7.3. Требования к реализации.....	16
7.4. Требования к интерфейсу.....	16
7.5. Требования к серверу.....	16
7.6. Требования к базе данных.....	16
7.7. Требования к рекомендательной модели.....	16
8. Другие требования к продукту (нефункциональные).....	17
8.1. Требования к надежности.....	17
8.2. Требования к хранению.....	17
8.3. Требования к безопасности (качеству, стабильности, удобству пользования, проч.).....	17
9. Описание архитектуры приложения и используемых технологий.....	17
9.1. Общая структура.....	17
9.2. Структура мобильного приложения.....	17
9.3. Структура сервера.....	18
9.4. Структура базы данных.....	20
9.5. Структура рекомендательной модели.....	23
10. Первоначальный прототип.....	25
10.1. Прототипы ранней стадии планирования.....	25
10.2. Первые прототипы.....	26

11. Используемые технологии (Front End / Back End / GitHub / прочие).....	27
12. Предложения по монетизации/коммерциализации (либо внутр. эффект от автоматизации)	28
13. Будущее развитие проекта.....	28
14. Отчеты о взаимодействии с заказчиком.....	28
15. Отзыв заказчика.....	29
16. Руководство пользователя.....	29
1. Введение	

1.1. Наименование программы

На русском языке: “Цифровой ассистент студента”

На английском языке: “Digital Student Assistant”

1.2. Краткая характеристика области применения

Областью применения данного приложения является образование, организация учебного процесса. Приложение главным образом предназначено для студентов Высшей Школы Экономики, выполняющих в рамках своего обучения проектную деятельность. Благодаря системе студенты смогут просматривать предлагаемые проекты, получать рекомендации проектов, соответствующих их интересам от модели, основанной на машинном обучении, подавать заявку на проект. Кроме студентов, система может быть применена преподавателями и администрацией НИУ ВШЭ с целью облегчения организации проектной работы студентов.

2. Команда

2.1. Командные роли и распределение ответственности

Зубарева Наталия Дмитриевна	Менеджер проекта, технический писатель, программист блока интеграции с цифровыми системами ВШЭ, программист бэкенда. Генератор идей по тесту Белбина.
Мостачев Андрей Олегович	Аналитик, программист блока системы рекомендаций, ML специалист. Аналитик по тесту Белбина.
Поволоцкий Виктор Александрович	Тестировщик клиентской части приложения, программист IOS. Исполнитель по тесту Белбина.
Сальникова Алиса Дмитриевна	Тестировщик серверной части приложения, программист бэкенда, специалист по базам данных. Эксперт по тесту Белбина.

Все участники проекта занимаются дополнением рабочего пространства на платформе Notion и участвуют в проверке документации.

Распределение участников по ролям было осуществлено на основании пожеланий, технических навыков и готовности и возможности изучать новые технологии.

2.2. Матрица компетенций

Оценки в матрице выставляются по шкале

- 1 - минимальные знания и навыки
- 2 - базовое понимание и умение
- 3 - среднее владение навыками
- 4 - уверенное использование навыков
- 5 - профессиональные знания и навыки

Компетенция	Зубарева Н.Д.	Мостачев А.О.	Поволоцкий В.А.	Сальникова А.Д.
надпрофессиональные навыки				

Управление временем	4	3	4	4
Самостоятельность	4	4	4	5
Командность	5	3	4	4
Обучаемость	5	5	5	5
Ответственность	5	4	5	5
Организационные навыки	4	2	2	2
профессиональные навыки				
Владение навыками мобильной разработки для IOS на Swift	4	1	5	1
Владение навыками разработки бэкенда на Django rest framework	3	1	1	1
Владение навыками создания и поддержания Postgres баз данных, языком SQL	2	1	2	5
Владение навыками создания клиент-серверных API	4	1	4	1
Владение знаниями операционной системы Linux и навыками работы с физическими серверами	2	1	1	3
Владение языком Python	4	5	3	3
Владение навыками и знаниями машинного обучения	3	5	2	2

2.3. Матрица ответственности

Матрица заполнена в соответствии с разделением ответственности:

- К - консультант
- О - ответственный
- Р - разработчик
- И - информируемый

Результат	Зубарева Н.Д.	Мостачев А.О.	Поволоцкий В.А.	Сальникова А.Д.
Интерфейс мобильного приложения	К	И	О, Р	И
Внутренняя логика фронтенда	К	И	О, Р	И
Связь мобильного приложения и сервера через запросы	К	И	О, Р	К

Тестирование мобильного приложения	И	И	О, Р	И
Сервер	О, Р	И	И	К
База данных	К	И	И	О, Р
Интеграция базы данных Postgres и сервера Django rest framework	Р	И	И	О, Р
Тестирование бэкенда	Р	И	И	О, Р
Рекомендационная модель	К	О, Р	И	К
Подготовка данных для модели	Р	О	И	Р
Интеграция модели на сервере	О, Р	К	И	Р
Обеспечение авторизации в системе	О, Р	И	Р	К
Тестирование системы	О, Р	Р	Р	Р
Разработка программной документации	О, Р	К	К	К

Таким образом,

- Зубарева Н.Д. частично отвечает за серверную часть и полностью за часть авторизации пользователя, а также за программную документацию, взаимодействия с заказчиком и внутри команды и организационные вопросы проекта,
- Мостачев А.О. отвечает за создание и содействие рекомендательной системы, а также за сбор и подготовку данных для ее работы,
- Поволоцкий В.А. отвечает за мобильное приложение и его интеграцию с сервером,
- Сальникова А.Д. отвечает за создание базы данных и ее интеграцию с сервером, а также за сервер.

3. Управление проектом

3.1. Метрики проекта

- Точность прогноза на уровне клиента – соответствие рекомендованного проекта интересам студента - **ВЫСОКАЯ**,
- Функциональность – реализация пользовательского интерфейса для авторизации в приложении, взаимодействия с проектами - **ВЫСОКАЯ**,
- Модульность и адаптируемость системы - наличие гибкости, возможность добавить или изменить какой-то один блок с минимальным воздействием на другие блоки - **ВЫСОКАЯ**,
- Степень интеграции с информационной системой ВШЭ - число дополнительных неавтоматизированных действий, которые нужны для согласования выбора проекта, должно быть сведено к минимуму, - **СРЕДНЯЯ**.

3.2. План работ

Работа команды организована в парадигме Agile [29] с двухнедельными спринтами и начинается 02.11.2022.

В соответствии с планом работы на дисциплине “Командный проект”

- постановка задачи, анализ существующих решений и предметной области, анализ и разработка требований должны быть осуществлены до 01.12.2022. В рамках него
 - спринт 1 02.11.2022-16.11.2022: сбор материалов каждым участниками команды по соответствующей области, разработка концепции проекта.

- спринт 2 16.11.2022-30.11.2022: разработка, анализ, систематизация и утверждение требований к системе. Написание и выверка технического задания.
- проектирование архитектуры системы должно быть осуществлено до 15.01.2023. В рамках него
 - спринт 3 30.11.2022-14.12.2022: детальное изучение используемых инструментов каждым участником команды в своей области, формирование предварительного представления о структуре системы.
 - спринт 4 14.12.2022-28.12.2022: разработка архитектуры системы, интеграция всех блоков системы участниками.
 - спринт 5 04.01.2023-15.01.2023: подготовка, выверка и утверждение документации текущего этапа, реализация начальных структур системы участниками.
- разработка прототипа системы и приемочное тестирование должно быть осуществлено до 15.03.2023. В этот срок
 - спринт 6 18.01.2023-01.02.2023: первый этап реализации функций, создание общей структуры мобильного приложения, сервера, базы данных, рекомендательной модели.
 - спринт 7 01.02.2023-15.02.2023: реализация основной части функциональности каждого блока: создание основных экранов мобильного приложения, реализация основных запросов сервера, создание и заполнение основных таблиц в базе данных, создание минимально работающей рекомендательной модели.
 - спринт 8 15.02.2023-01.03.2023: доработка и интеграция всех систем участниками: финализация пользовательского интерфейса, создание рекомендательной модели и ее развертывание на сервере, финализация структуры базы данных, интеграция ее с сервером, добавление всего функционала сервера.
 - спринт 9 01.03.2023-15.03.2023: тестирование блоков системы в соответствии с обязанностями участников, разработка отчета о тестировании, проведение приемки, утверждение документов.
- окончание работы над проектом, финализация системы, разработка программной документации и передача системы заказчику должна быть осуществлена до 26.03.2023.
 - спринт 10 15.03.2023-26.03.2023: корректировка всех недочетов и внесение всех правок, выделенных на предыдущем этапе. Наполнение базы данных данными для демонстрации, отладка интерфейса, остаточная настройка рекомендательной системы. Разработка сопроводительной документации. Передача системы и документации заказчику

3.3. Использование инструментов (Canvas) или систем управления проектом (Trello, Jira)

Для управления проектом используется система Notion, в которой протоколируется информация по проекту, изменение и уточнение требований (рис. 1), а также вносятся задания для участников (рис. 2). Для контроля версий проекта используется система GitHub [8], ссылка на репозиторий проекта: https://github.com/aparinov/2023_DSA_iOS .

Протоколы встреч














-  [Протокол 25.03 14:00](#)
-  [Протокол 25.03 21:00](#)
-  [Протокол 23.03 20:00](#)
-  [Протокол 15.03 19:30](#)
-  [Протокол 14.03 10:00, 23:00](#)
-  [Протокол 13.03 23:00](#)
-  [Протокол 12.03 16:00](#)
-  [Протокол 09.03 18:00](#)
-  [Протокол 11.02 16:00](#)
- + ::  [Протокол 11.01 11:00](#)
-  [Протокол 13.01 22:00](#)
-  [Протокол 16.11 11:00](#)
-  [Протокол 02.11 11:00](#)

Рис. 1 - архив протоколов встреч

Виктор


- ☒ изучить список экранов https://www.kdnuggets.com/2020/11/topic-modeling-bert.html?__cf_chl_tk=NrohmsQviAxelCt_fjVfrAMfBsNA.bdojXiWwLD0yIY-1667376903-0-gaNycGzNB9E
- ☒ сделать макет в фигме на основании функций: 4 основных экрана (вход, личный кабинет, список проектов, ?отдельный проект (каюсь, прослушала)?)
- ☒ сделать проект
- ☒ подумать по поводу архитектуры
- ☒ финализировать архитектуру
- ☒ паттерны
- ☒ классы + компоненты
- ☒ верстка фронта
-  Верстка
- ☒ принять запрос на общий гитхаб
- ☒ связать сервер и клиент
- ☐ связь логина
- ☒ дочинить баги
- ☒ тестирование сетевого слоя

Рис. 2 - пример заполнения задач участника проекта

3.4. Анализ рисков

Шкала воздействия рисков:

Аспекты проекта\Воздействие	Низкое - 0.1	Умеренное - 0.4	Высокое - 0.7
Сроки	Изменение сроков не более чем на 5%	Изменение сроков не более чем на 20%	Изменение сроков более чем на 20%
Содержание	Пренебрежимое изменение функциональности	Существенное изменение функциональности, требует согласования	Фундаментальное изменение функциональности
Качество	Пренебрежимое для заказчика изменение качества	Существенное изменение качества, требует согласования	Фундаментальное изменение качества

Таблица рисков проекта:

Риск	Вероятность [0.1, 0.9]	Уверенность в вероятности	Воздействие	Оценка угрозы	Стратегия управления
негативные риски					
Участники проекта будут критически заняты другими предметами и работой, чтобы работать над проектом	0.9	высокая	0.7	0.63	Приложить усилия для минимизации. Составить планы по времени, учесть, что совместная работа возможна только в выходные дни.
Технические сложности в согласовании используемых технологий	0.9	высокая	0.4	0.36	Приложить усилия для минимизации. Тщательно выбирать инструменты ориентируясь на их совместимость.
Неправильное понимание требований\текущей ситуации в связи с отсутствием опыта в конкретных технологиях	0.7	высокая	0.4	0.28	Минимизировать риск. Изучить технологии перед использованием.
Снижение эффективности коммуникаций с заказчиком	0.6	средняя	0.4	0.24	Минимизировать риск, составить план коммуникаций, активно информировать о прогрессе.
Какие-то из используемых	0.4	низкая	0.4	0.16	По возможности минимизировать

инструментов прекратят свою работу на территории РФ					риск, при всех прочих равных выбирать отечественные аналоги.
Прочие жизненные обстоятельства, временно препятствующие исполнителям работать над проектом	0.9	высокая	0.1	0.09	При случае минимизировать риск, спланировать расписание с запасом, минимизировать ситуации взаимозависимости участников.
Сдвиг сроков в связи с непониманием процедуры отчетности по дисциплине Командный проект	0.6	средняя	0.1	0.06	При случае минимизировать риск, изучить еще раз регламент.
позитивные риски					
Выполнение работы раньше срока в связи с работой Виктора над ВКР	0.5	высокая	0.4	0.2	Максимизировать риск, повысить мотивацию Виктора делать ВКР
Отсутствие необходимости сложной связи используемых технологий в силу их совместимости	0.5	высокая	0.1	0.05	При случае максимизировать риск. Стараться выбирать технологии заведомо совместимые.
Улучшение качества выполнения работы в силу повышения знаний исполнителей	0.3	высокая	0.1	0.03	При случае максимизировать риск. Дополнительно изучать материалы, связанные с проектом, при наличии времени.

4. Предметная область и описание продукта

4.1. Постановка проблемы

Приложение решает существующую проблему выбора студентами проектов для проектной работы в рамках их обучения. В данный момент существующие решения не предоставляют унифицированной, удобной возможности выбрать проект, особенно основываясь на интересах студента. Студентам для поиска и выбора проекта зачастую требуется использовать сразу несколько ресурсов на разных этапах (таблицы с проектами, мессенджеры для связи с руководителями, ЛМС для регистрации на проект, и т.д.), кроме того нередко ни один из ресурсов

не предоставляет достаточную информацию (например, подробное, понятное описание проекта). Часто студенты вынуждены искать интересующие проекты по названию среди десятков проектов в таблице, так как не существует решения, которое позволяло бы, например, сделать фильтрацию проектов по предметной области или используемым технологиям.

Разработанная система имеет своей целью упростить этот процесс для студентов, сделав по крайней мере некоторые его аспекты более удобными (просмотр проектов с подробной информацией, получение только актуальных интересов проектов и т.д.).

4.2. Анализ предметной области, рамки и границы проекта

Основными аналогами системы являются гугл таблицы с темами проектов, системы LMS [22] и SmartLMS [16], а также мобильное приложение HseApp X [11]. Анализ конкурентов отражен в пункте 6.

Внутри экосистемы заказчика разработанная система представляет собой ответвление (хотя в данный момент и независимое) сервиса среди других вариаций (например, приложения для Android, отдельного проекта по изучению и разработке технологий кластеризации для проектов). Сервер размещен на удаленном устройстве заказчика в отдельном контейнере и обращается к отдельной базе данных, поэтому ресурсы различных проектов не пересекаются.

Границы проекта, таким образом, заключаются именно в разработке самостоятельной системы с отдельной базой данных, сервером, собственной рекомендательной системой, основанной на машинном обучении, и мобильным приложением для платформы IOS. Адрес для развертывания сервера предоставляется заказчиком, однако разработка его структуры и функциональности полностью реализуется участниками проекта.

Проект включает в себя разработку преимущественно инструмента для студента-участника проекта. Функциональность для руководителя или создателя проекта дополнительна и не входит в основные границы системы.

Кроме того, дальнейшее развитие приложения (в частности, более тщательное извлечение интересов пользователя и перенос рекомендательной модели на клиентскую часть) будет осуществлено Поволоцким Виктором в рамках выполнения ВКР, поэтому данные функции также не входят в границы текущего проекта.

Более подробную информацию относительно требований и границ системы можно узнать в документе “Функциональные требования и тех. задание”, где требования перечислены в соответствии с методом MOSCOW [21] и, таким образом, очерчивают границы текущего проекта.

4.3. Назначение приложения

Система предназначена для использования в образовательных учреждениях (конкретно НИУ ВШЭ) организаторами учебного процесса, преподавателями и студентами.

Работники учебного офиса и преподаватели могут пользоваться серверной частью системы и ее веб-интерфейсом для создания набора проектов, доступных студентам для выбора на текущем этапе образовательного процесса.

Студенты могут воспользоваться IOS приложением для того, чтобы заполнить свои интересы, просмотреть предложенные проекты, получить рекомендованные проекты, зарегистрироваться на проект и предложить свой проект.

4.4. Новые предоставляемые функции

По сравнению с существующими аналогами разработанная система, с одной стороны, содержит ряд инновационных функций, с другой стороны улучшает пользовательский опыт в существующих функциях.

Главной функцией, не присутствующей в аналогах, является наличие системы рекомендаций - инструмента, который позволяет пользователю получить подборку проектов, соответствующих его научным интересам. Это, с одной стороны, сокращает время, необходимое пользователю для поиска проекта (так как альтернатива - просмотр всех проектов), а с другой стороны повышает вероятность того, что выбранный проект будет действительно интересен студенту, а значит скорее всего будет выполнен лучше.

Также, учитывая что приложение HseApp X не было предназначено для выбора проектов, данное решение предоставляет удобство мобильной платформы для взаимодействия с проектами, что также повышает удобство процесса.

В целом агрегирование описаний проектов с подробной информацией - это повышение качества информационной базы относительно того, что существует сейчас.

5. Описание потенциальных пользователей

5.1. Цель заказчика и ключевые стейкхолдеры (матрица Управления стейкхолдерами)

Цель заказчика: получить систему, с помощью которой можно было бы оптимизировать процесс проектной работы в НИУ ВШЭ.

Стейкхолдеры (рис. 3) и стратегии в отношении них:

1. организаторы дисциплины “Командный проект”. Влияют на проект, но строго говоря не заинтересованы в результате, в соответствии с регламентом курса необходимо адаптировать темп работы над проектом и ключевые результаты каждого этапа.
2. заказчик. Является главным стейкхолдером проекта, с ним необходимо максимальное сотрудничество, информирование, приоритетное выделение требований к системе.
3. цифровой блок НИУ ВШЭ. Практически не участвует в проекте, но может быть заинтересован в результатах - достаточно информировать о наличии системы, имеет смысл ориентироваться на встраиваемость в их экосистему.
4. потенциальные пользователи: студенты и преподаватели, организаторы учебного процесса. Так как они являются конечными пользователями системы, ее необходимо разработать учитывая их потребности, для этого имеет смысл анализ пользовательских историй, опросы о желаемом и полезном функционале, возможно тестирование с этими группами.



Рис. 3 - матрица стейкхолдеров

5.2. План управления коммуникациями

Информирование стейкхолдеров происходит в соответствии с результатами анализа их важности и разработки стратегии взаимодействия.

Так, организаторы “Командного проекта” уведомляются о результатах в соответствии с регламентом дисциплины в требуемом формате через официальные учебные платформы ВШЭ.

Коммуникация с потенциальными пользователями не регламентирована - это могут быть точечные опросы о желаемой функциональности в момент выделения требований или же контакты с целью А/В тестирования системы на этапе тестирования.

Коммуникации с цифровым блоком осуществляются по необходимости через официальные каналы НИУ ВШЭ.

Коммуникация с заказчиком проводится в соответствии с планом спринтов: по завершении каждого спринта заказчику предоставляется отчет и запрашивается обратная связь, в случае корректировок вносятся изменения в продукт. Контакт проводится главным образом через неформальные методы мессенджеров для обеспечения максимальной оперативности, также преимущественно в формате видеозвонков на платформе Zoom. Все документы, изменения и функции согласуются с заказчиком и изменяются в соответствии с соглашениями. Заказчику также предоставлен доступ к системе контроля версий проекта GitHub и платформе управления проектом Notion, где он может наблюдать прогресс, изменения и соответствие разработки требованиям.

6. Анализ конкурентов

Функция	Таблицы для выбора проектных работ на ФКН	<u>ЛМС ВШЭ</u>	<u>HSE App X</u>
Мобильное приложение	Нет	Нет	Есть*
Авторизация через личный кабинет ВШЭ	Нет	Есть	Есть*
Личная информация студента	Нет	Есть	Есть*
Личный кабинет студента	Нет	Есть	Есть*
Общая информация обо всех проектах	Есть*	Есть	Нет
Подробная информация о статусе проекта	Есть*	Нет	Нет
Возможность выбирать проект	Нет+-	Есть*	Нет
Возможность отменять выбор проекта	Нет+-	Нет	Нет
Возможность создавать свой проект	Нет+-	Есть*	Нет
Заполнение своих учебных интересов	Нет	Нет	Нет
Рекомендации от системы	Нет	Нет	Нет
Централизованное хранение всех проектов	Нет+-	Есть*	Нет
Интегрированность в цифровые системы ВШЭ	Нет	Есть	Есть*
Удобный интерфейс	Нет	Нет	Есть*

Примечание: знаком +- отмечены функции, которые реализованы с использованием сторонних ассоциированных систем. Знаком * отмечена реализация функции, являющаяся предпочтением заказчика и образцом для разработки приложения Цифровой ассистент студента.

Ключевыми преимуществами разработанной системы являются, безусловно, наличие рекомендательной модели, позволяющей студентам видеть проекты, наилучшим образом им подходящие, и доступность приложения на мобильном устройстве.

7. Функциональные требования

7.1. Подключение к системе

Мобильное приложение должно быть доступно для установки с помощью APK при наличии ноутбука Macbook. При условии подключения к сети Интернет и запущенного сервера, приложение должно работать и позволять пользователю осуществлять все функциональные действия, описанные ниже.

Сервер должен быть развернут по предоставленному заказчиком адресу. Пользователь с подключением к сети Интернет должен иметь возможность зайти в браузере на адрес, соответствующий серверу, при условии запуска сервера.

Владелец сервера должен иметь возможность запустить его на компьютере из терминала операционной системы Linux с помощью команд, описанных в руководстве пользователя сервера, при условии наличия всех необходимых программных пакетов.

7.2. Требования к составу выполняемых функций

Изначальные требования к системе можно увидеть в документе технического задания. Здесь перечислены финальные требования в зависимости от статуса реализации (требования, не являющиеся необходимыми и нереализованные, очерчивают внешние границы проекта).

Реализованные требования:

1. Авторизация пользователя по данным ЕЛК ВШЭ (email и пароль).
2. Ввод студентом своих научных интересов в виде выбора из предоставленных областей интересов.
3. Ввод студентом прочих личных данных (имя, факультет, группа, курс, номер телефона для связи)
4. Просмотр проектов студентом.
5. Просмотр подробной информации о любом проекте (название, описание, тип, технологические требования, данные руководителя, количество студентов для участия, срок подачи заявки, срок сдачи проекта, форма подачи заявки).
6. Подача студентом заявки на проект.
7. Отмена студентом заявки на проект.
8. Получение студентом рекомендаций проектов в соответствии с интересами и описаниями проектов.
9. Создание студентом инициативного проекта с заполнением информации о проекте из пункта 5 данного списка.
10. Хранение списка проектов в базе данных.
11. Хранение информации о студентах в базе данных.
12. Предоставление возможности получать информацию из базы данных информационным системам ВШЭ.
13. Возможность управления созданным проектом из веб-интерфейса (возможность администратора): просмотр лиц, подавших заявки на проект, отклонение или одобрение их заявки.

Требования за пределами границ проекта (нереализованные):

1. Предоставление студенту дополнительной текстовой информации о проекте (сколько человек его выбрали, данные этих людей, стадия выполнения проекта, статус прохождения этих стадий).
2. Передача решения студента о выборе проекта в информационную систему ВШЭ и отображение этого решения в LMS.
3. Отметка студентом проекта как заинтересовавшего, отмена этой отметки, возможность просматривать список заинтересовавших проектов студентом.
4. Возможность участия студента в переписке с лицами, связанными с проектом, в приложении.

5. Управление созданным проектом из мобильного приложения: просмотр лиц, подавших заявки на проект, отклонение или одобрение их заявки.
6. Расширенное управление созданным проектом из веб-интерфейса: создание контрольных точек.
7. Отображение личного кабинета преподавателя в веб-интерфейсе с какой-либо функциональностью относительно студентов.
8. Отображение контрольных точек по выбранным проектам в личном кабинете студента.
9. Отображение списка дисциплин, успеваемости студента.
10. Повторный или расширенный ввод студентом своих научных интересов.

7.3. Требования к реализации

Подробные требования к реализации можно найти в документе “Функциональные требования и тех. задание”. Вкратце,

- разработка ведется по методологии Agile с спринтами по 2 недели,
- для организации процесса используется система контроля версий GitHub и система управления проектом Notion, к которым должен быть доступ у всех участников,
- программный код должен быть написан в максимальном соответствии с стандартами организации и чистоты кода,
- мобильное приложение должно быть реализовано на языке Swift,
- база данных должна быть разработана на базе технологии PostgreSQL,
- система рекомендаций должна быть разработана на языке Python с использованием алгоритмов кластеризации,
- система должна сопровождаться документацией, которая бы объясняла, как ее эксплуатировать.

7.4. Требования к интерфейсу

Интерфейс должен позволять пользователю осуществлять описанные выше функции и содержать

- экран входа в приложение для авторизации по учетным данным ВШЭ,
- экран добавления интересов студента,
- экран личного кабинета студента для ввода дополнительной информации,
- экран проектов, доступных студенту,
- экран детальной информации о каком-либо конкретном проекте,
- элементы для осуществления ассоциированных функций на экранах, например текстовые поля для ввода информации о студенте на экране личного кабинета студента, кнопка подачи заявки на проект на экране детальной информации проекта и т.д.

7.5. Требования к серверу

Сервер должен располагаться на предоставленном заказчиком адресе и осуществлять поддержку всех необходимых взаимодействий между мобильным приложением, базой данных, рекомендательной моделью, а также осуществлять авторизацию пользователей через сторонние сервисы.

7.6. Требования к базе данных

База данных должна содержать таблицы для информации о студентах, проектах и взаимосвязях между ними.

7.7. Требования к рекомендательной модели

Рекомендательная модель должна принимать на вход список доступных проектов с подробной информацией о них и список интересов пользователя и предоставлять в результате обработки этих данных подборку проектов из изначального списка, соответствующих интересам пользователя.

8. Другие требования к продукту (нефункциональные)

8.1. Требования к надежности

Система должна работать корректно при любом вводе пользователя, не должна завершать свою работу аварийно, в случае ошибки должна завершаться в соответствии с предусмотренным протоколом, уведомляя пользователя (graceful exit).

8.2. Требования к хранению

Система должна быть эксплуатируемой на мобильном устройстве на платформе IOS для мобильного приложения и на компьютере с операционной системой Linux для серверной части. Требования к хранению совпадают с требованиями для эксплуатации этих устройств.

8.3. Требования к безопасности (качеству, стабильности, удобству пользования, проч.)

Для обеспечения безопасности личных данных пользователей система не должна хранить нигде в открытом доступе полные пары учетных записей пользователей в незашифрованном виде.

Система должна работать на устройствах с лицензионным программным обеспечением IOS 13 или более поздней версии, но не обязана работать ни на каких других платформах. Мобильное приложение должно занимать не более 250 Мб памяти устройства и использовать не более 2 Гб оперативной памяти.

Для обеспечения возможности обслуживания система должна быть спроектирована в соответствии с основными принципами ООП и мобильной разработки.

9. Описание архитектуры приложения и используемых технологий

9.1. Общая структура

Система с физической точки зрения состоит из трех компонент:

- мобильного приложения, находящегося на каком-либо пользовательском устройстве,
- веб-интерфейса, доступного по заданному заказчиком адресу,
- серверной части, находящейся на удаленном устройстве заказчика, содержащей сервер, базу данных и рекомендательную модель.

Все элементы взаимодействуют друг с другом через api сервера, предназначенные для вызовов из фронтенда.

Далее будет описана структура отдельно каждого из этих блоков.

9.2. Структура мобильного приложения

Мобильное приложение реализовано на одной из самых популярных архитектур для реализации мобильных приложений для iOS - ModelViewViewModel+Coordinator (MVVM+C) [25] (рис. 4).

В основной директории находятся два главных фолдера: Scenes, отвечающие за реализацию архитектурного паттерна, в сцене каждого экрана находятся следующие разделы: Model(Модели данных), View(Дополнительно реализованные View), ViewModel(ViewModel для работы с сервисами и обновления данных на UI), ViewController(То что в архитектуре является View, основная сущность экрана) и Services, в котором находятся сервис для работы с сетью и вспомогательные модели данных.

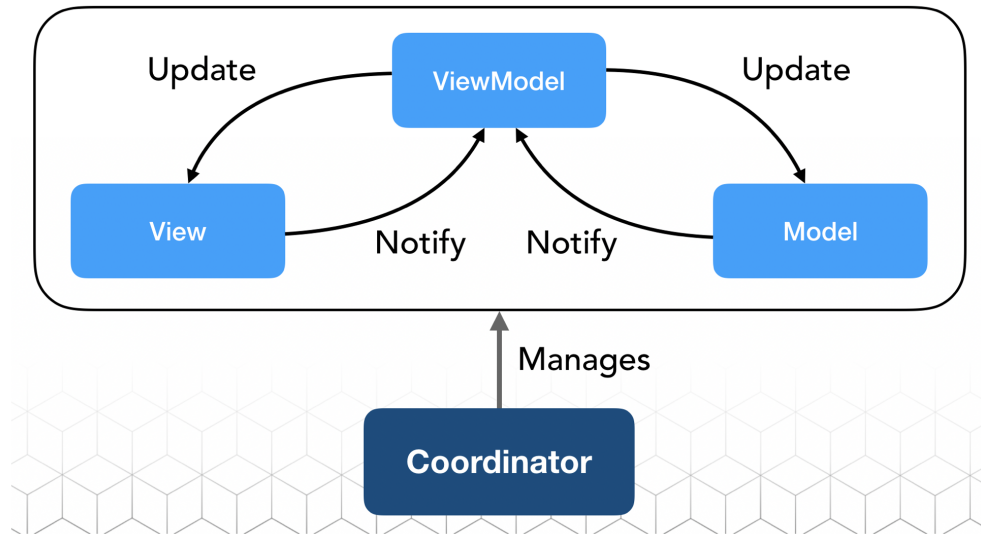


Рис. 4 - структура ModelViewViewModel+Coordinator

9.3. Структура сервера

Сервер расположен в папке nate_kp в директории, предоставленной заказчиком команде (рис. 5). Внутри нее находится директория env с виртуальной средой, нужной для работы Python с проектом, и директория nate_kp, соответствующая всему проекту Django rest framework.

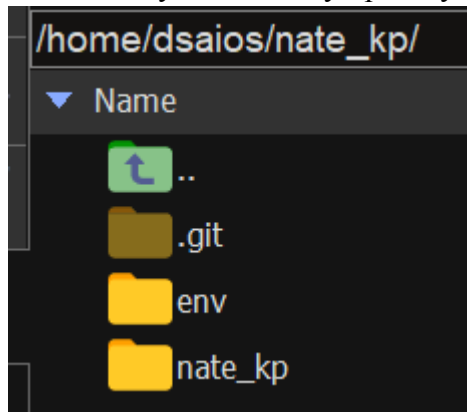


Рис. 5 - общая директория сервера

Внутри этой директории располагаются объекты проекта (рис. 6):

- директория nate_kp с файлами, относящимися ко всему проекту целиком, в частности с настройками,
- директория provider, где расположено приложение для расширения библиотечного решения авторизации django allauth [28], позволяющее подключить провайдера авторизации кроме заданных (например, ЕЛК ВШЭ),
- директория server - относящаяся к приложению, собственно, сервера, используемого в этом проекте и взаимодействующего с базой данных,
- директория templates, содержащая документы верстки html страниц,
- файл manage.py, с помощью которого запускаются приложения проекта.

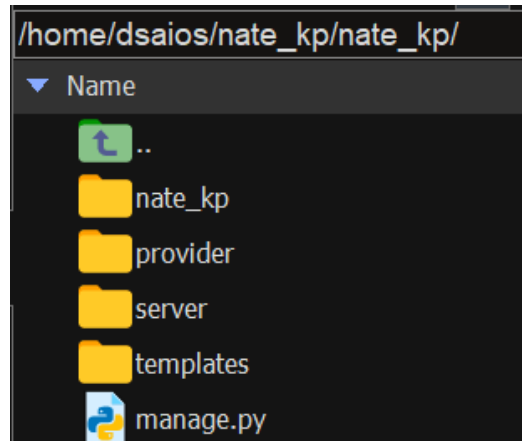


Рис. 6 - директория проекта

Проект связывается с базой данных, расположенной на этом же устройстве, посредством указания ее адреса в файле настроек /home/dsaivos/nate_kp/nate_kp/nate_kp/settings.py (рис. 7).

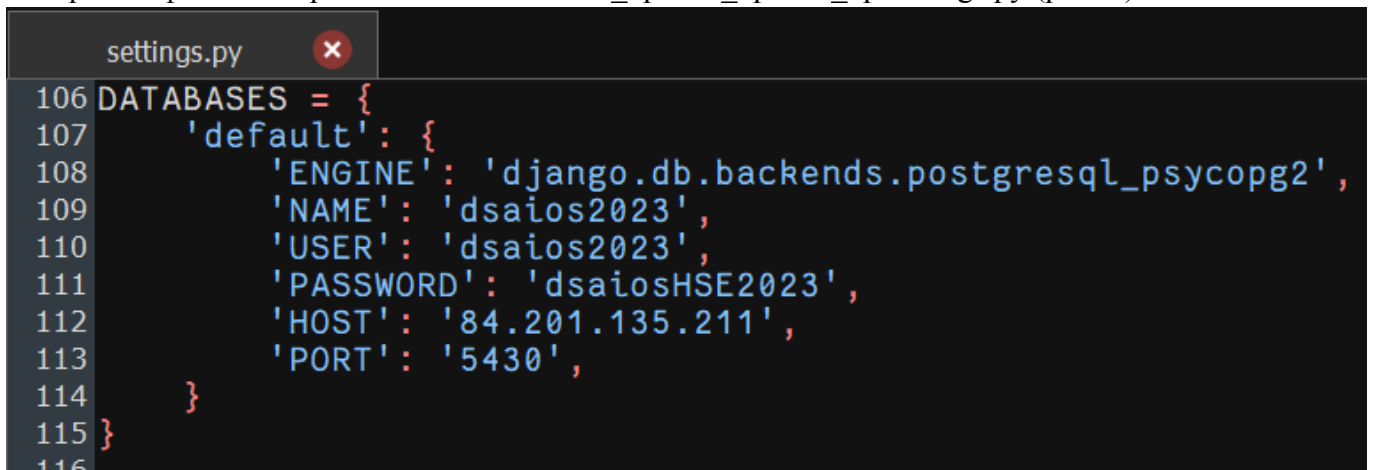


Рис. 7 - связь проекта Django rest framework с базой данных Postgres

Сам основной сервер (приложение server) имеет следующую структуру (рис. 8):



Рис. 8 - организация данных и объектов в приложении server

Таким образом, пользователь имеет доступ к url из файла urls.py, и через соответствующие view, serializer и model получает доступ к тем или иным данным в базе данных, соответствующих этим представлениям.

Рекомендательная модель находится в файле /home/dsaio/nate_kp/nate_kp/server/model.py и запускается из view объекта student_suggestionsDetail, доступного по url /student_suggestions/<int:pk>/, где int:pk - это идентификатор студента, для которого составляется рекомендация.

Полное описание всех url сервера доступно в Руководстве пользователя.

9.4. Структура базы данных

База данных состоит из некоторого количества таблиц, часть из которых генерируется автоматически. Для нас представляет интерес та часть, которая написана вручную и отражает объекты в модели системы. Также в системе присутствуют срезы данных, представляющие собой агрегированные данные из нескольких таблиц. Все они перечислены далее, и их схема представлена на рисунках 9, 10, 11, 12, 13 и 14.

- Таблица student - это таблица студентов, содержит только идентификаторы и email, используемый пользователем для авторизации. Связана с автоматически генерируемой таблицей пользователей user по идентификатору. Триггер на создание студента в таблице прикреплен к авторизации пользователя в первый раз.
- Таблица project - это таблица проектов, содержащее всю информацию о проектах кроме тэгов(технических требований) - кратких описаний технологий, областей знаний и навыков, требуемых от студентов.
- Таблица technical_requirements - это таблица, содержащая идентификаторы тэгов и их названия. Пользователь в приложении не имеет возможности создавать эти тэги, они настраиваются администратором через веб-интерфейс (или вносятся напрямую в базу данных).
- Таблица student_interests - это таблица “многие ко многим” для соответствия студентов и их интересов. Одна запись в такой таблице - это информация о том, что студент с данным идентификатором выбрал тэг с данным идентификатором как свой интерес.
- Срез student_interests_list - это срез, позволяющий агрегировать все интересы студента с их названиями. При получении записи из него запрашивающий увидит идентификатор данного студента, идентификаторы и текстовые названия всех его интересов.
- Таблица student_info - это таблица с подробной информацией о студенте. Она содержит дополнительную информацию, такую как имя, факультет и т.д., а также идентификатор студента из таблицы student, которому эта информация соответствует. Разделение базовой информации о пользователе и детализированных данных на разные таблицы продиктовано тем, что в логике приложения пользователь задает свои данные несколько позже, чем создается объект студента, а также потому что основная информация о пользователе нужна чаще. С помощью разделения можно удобнее создавать и изменять эти данные, а также не использовать их без надобности.
- Срез student_information - это агрегированный срез, представляющий собой полную информацию о студенте - и общую, и базовую, объединенные в один json объект.
- Таблица application - это таблица “многие ко многим” заявок студентов на проекты. Одна запись соответствует информации о том, что студент с данным идентификатором подал заявку на проект с данным идентификатором. Кроме того, она содержит поле статуса заявки - после подачи оно равняется “заявка подана”, далее же, например при отмене заявки, меняется на “заявка отменена”. В случае, если руководитель проекта из веб-интерфейса решит одобрить заявку - статус может быть изменен на “заявка одобрена”. Статус введен для того, чтобы расширить возможную информацию, при этом не добавляя другие сущности и не удаляя записи из этой таблицы при отмене заявок.

- Срез student_applications - это агрегированный срез, позволяющий увидеть все заявки студента с их идентификаторами, статусами, а также информацией о проектах, на которые они были поданы, с полными описаниями.
- Таблица requirements_stack - это таблица “многие ко многим” для соответствия тэгов и проектов. Одна запись из такой таблицы соответствует информации о том, что проекту с данным идентификатором добавлен тэг с данным идентификатором.
- Срез project_requirements - агрегированный срез, позволяющий для данного проекта увидеть все тэги, причем не только их идентификаторы, но и названия.
- Таблица suggestions - это таблица “многие ко многим” соответствия рекомендаций проектов студентам. Одна запись соответствует информации о том, что студенту с данным идентификатором предложен проект с данным идентификатором.
- Срез student_suggestions - это агрегация всех предложенных студенту с данным идентификатором проектов, с полным их описанием.

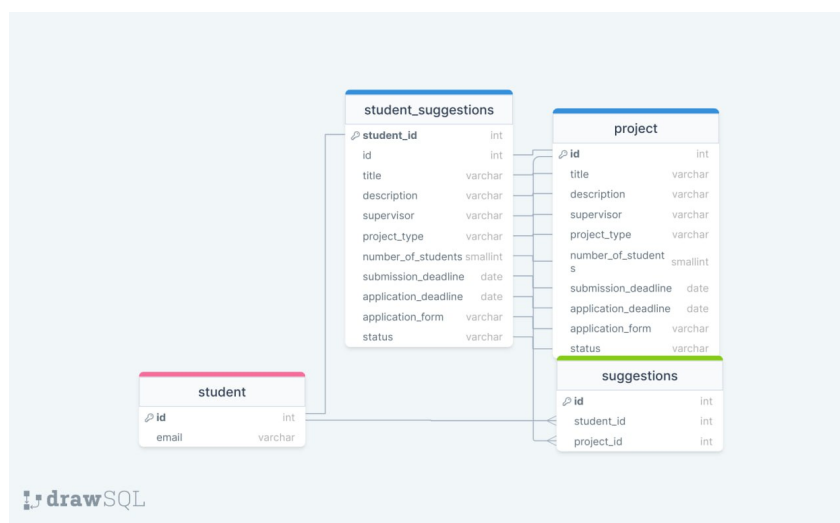


Рис. 9 - модель связи между студентами и предложенными проектами

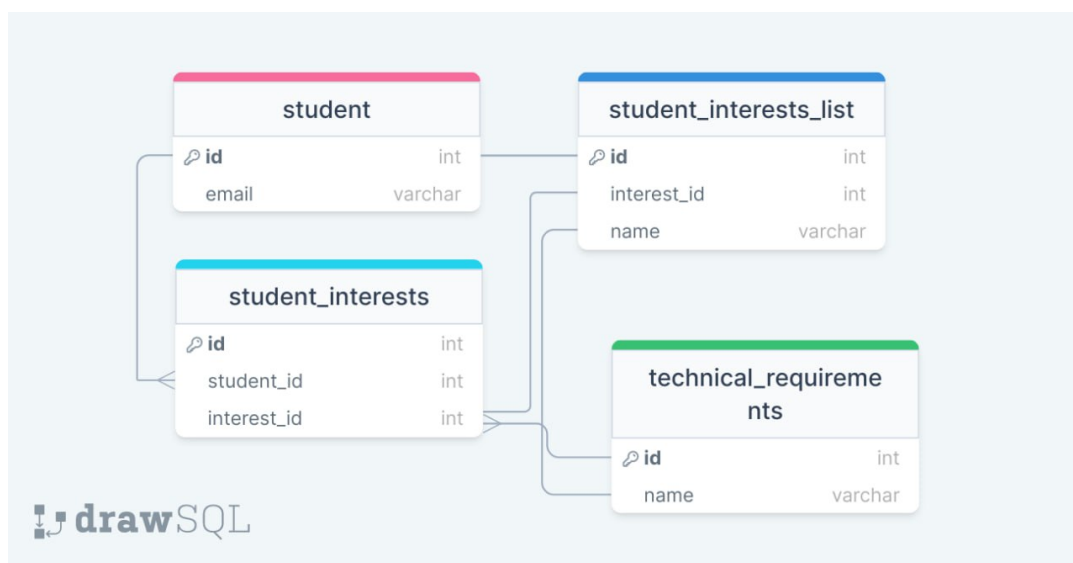


Рис. 10 - модель связи между студентами и тэгами (интересы студента)

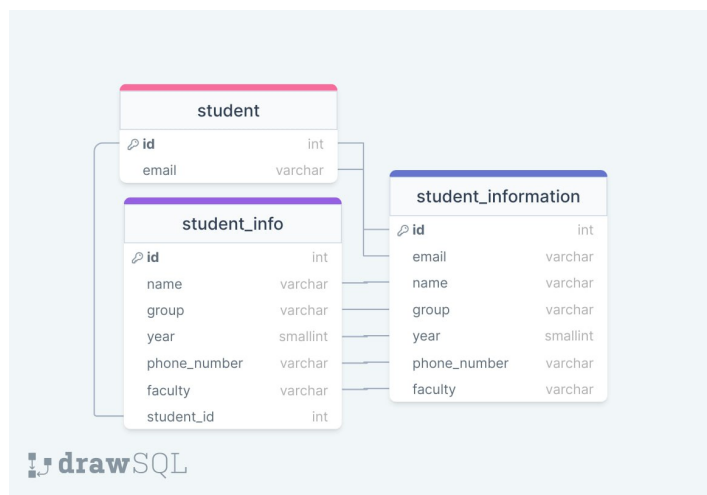


Рис. 11 - модель связи между студентами и подробной информацией о них

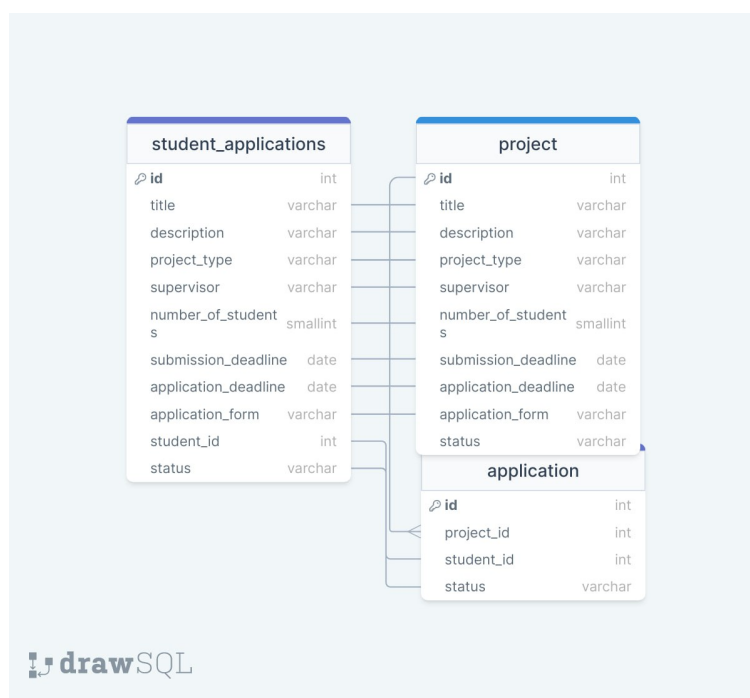


Рис. 12 - модель связи между проектами и заявками студентов на участие в них

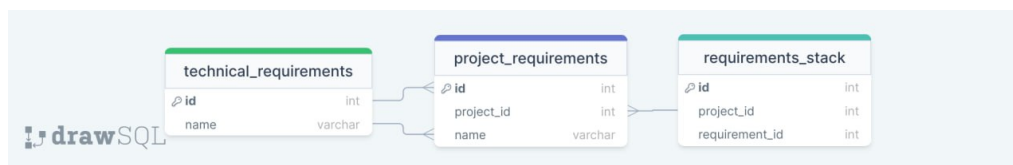


Рис. 13 - модель связи между проектами и тэгами технических требований

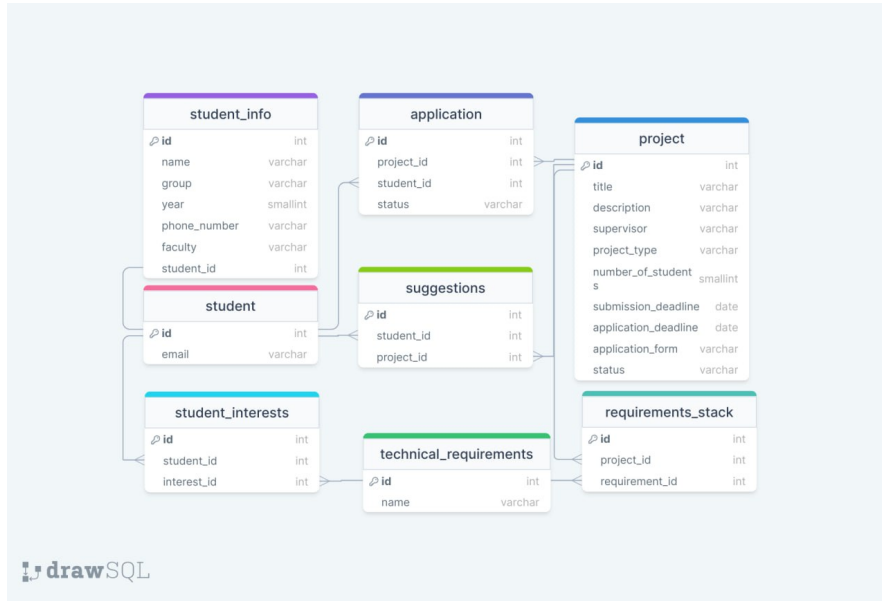


Рис. 14 - модель связи между всеми основными таблицами базы данных (не срезами)

9.5. Структура рекомендательной модели

9.5.1. Генерация данных пользователей

Для оценки качества рекомендаций, выдаваемых моделью, требуется база пользователей с их интересами. По причинам отсутствия размеченной выборки пользователей с тегами, отображающими их интересы, а также во избежание ручной разметки большого количества данных, было принято решение создать алгоритм генерации пользовательских данных. Целью ставилось составление реалистичных интересов и избежание большого разброса, при котором теги интересов пользователя мало связаны друг с другом. Для достижения поставленной цели был выбран итеративный подход на основе матрицы связности тегов.

[illegible]

В матрице представлены попарные скоры связи между тегами, выбранными для MVP. Матрица создана вручную. Возможности использовать корпус курсовых и иных работ ПИ ВШЭ не было, так как связность тегов в рамках одной работы и соответствующие показатели для пользователя принципиально различаются. Также каждому тегу был проставлен скор популярности.

Алгоритм генерации пользователей представляет собой хождение по графу тегов. В таком случае, значения матрицы связности представляют собой величины, обратные длине путей в графе. На первом этапе, в качестве зерна случайно выбирается один тег, на основе скоров популярности - то есть чем популярнее тег, тем большая вероятность того, что он окажется начальным. Адрес каждого следующего шага по графу определяется в соответствии со связностью всех возможных следующих вершин с предыдущим (а с третьего шага - предыдущими двумя) тегом а также скоры популярности. В общем случае вероятность шага из вершины i в вершину j определяется следующей формулой:

$$P_{ij} = \frac{0,8 (0,6 w_{ij} + 0,4 w_{kj}) + 0,2 (p_j)}{\sum_{m \neq i} P_{im}}$$

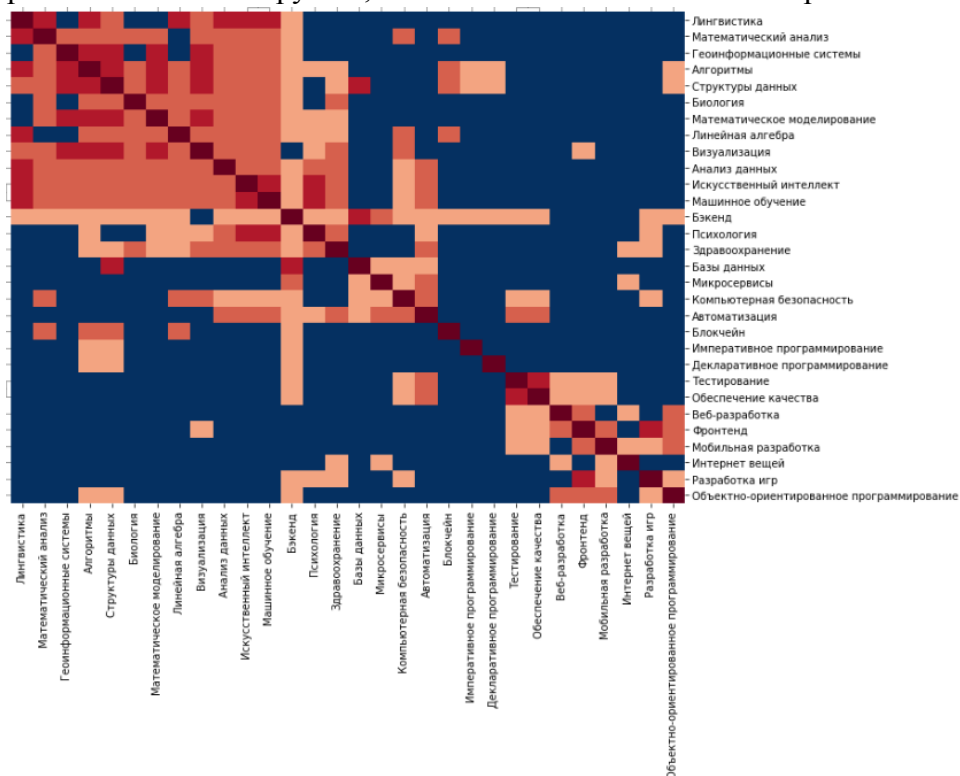
где k - индекс предыдущего взятого тега. Такой подход позволяет избежать подсчета общей метрики реалистичности созданного пользователя и дает достаточно неоднородный результат.

9.5.2. Алгоритм составления рекомендаций

Составление рекомендаций состоит из следующих шагов:

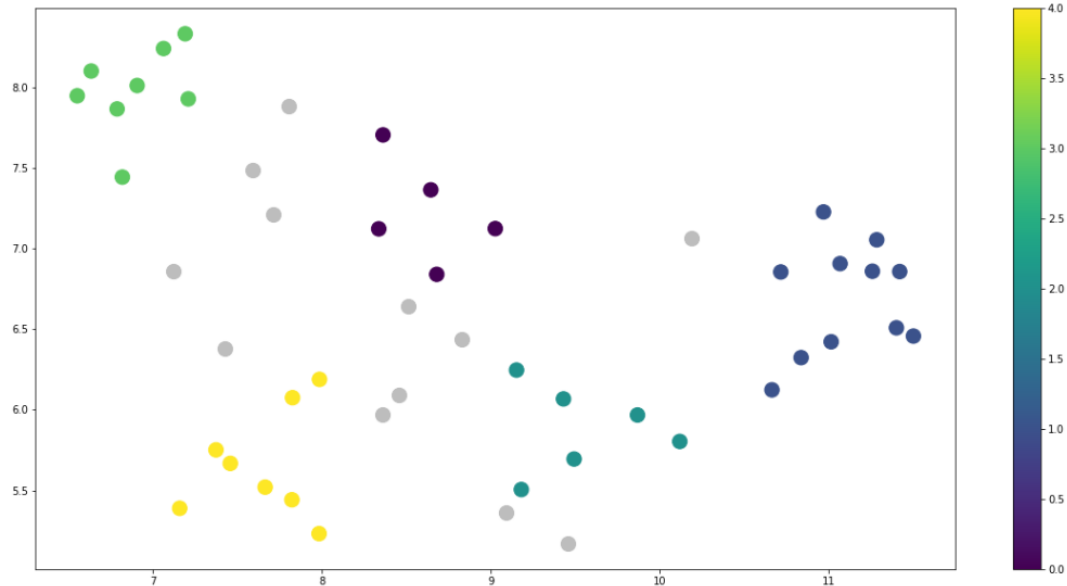
- 1) Кластеризация тегов
- 2) Кластеризация проектов
- 3) Подсчет связности по тегам
- 4) Составление итоговых рекомендаций

Кластеризация тегов была произведена с помощью функции `clustermap`. Выбранные теги были разделены на три высокосвязные группы, остальные считались не кластеризованными.



Кластеризация тегов была мотивирована тем, что наличие совпадающих тегов у пользователя и проекта внутри высокосвязных кластеров должна рассматриваться отдельно от соответствующего показателя для некластеризованных тегов. Внутри каждого кластера подсчитывалась метрика `Intersection Over Union`, определяющая долю совпадающих тегов среди общего набора тегов пользователя и проекта внутри данной категории. Для некластеризованных тегов подсчитывалась модифицированная метрика, где вместо объединения наборов тегов пользователя и проекта использовались только теги работы. Это было сделано, чтобы не штрафовать пользователей, имеющих разносторонние интересы. Среди четырех групп (трех кластеров и некластеризованных тегов) выбирались две с наиболее высокими показателями соответствия, из них составлялась итоговая метрика.

Проекты также были кластеризованы, но уже в соответствии с их названиями на английском языке. Для этого сначала были созданы эмбединги слов, используемых в названиях проектов, с помощью предобученной модели BERT. Затем к векторам названий проектов был применен алгоритм понижения размерности UMAP. В полученном 10-мерном пространстве проекты были разделены на высокосвязные кластеры с помощью алгоритма DBSCAN. Двумерное представление 50 рассмотренных проектов представлено ниже:



В итоге рекомендации строятся следующим образом:

- 1) Для каждого проекта подсчитывается скор соответствия пользовательским интересам на основе тегов.
- 2) При превышении барьера в 0.5 у проекта с наивысшим скором соответствия рассматриваются проекты наиболее близкие к нему в пространстве названий.
- 3) Для остальных проектов внутри кластера к метрике соответствия пользователю добавляется величина, обратная расстоянию внутри кластера до проекта с наивысшим скором
- 4) В итоге в дополнение к изначальному выбору одного проекта добавляется еще 4 с наивысшими значениями комбинированной метрики.

Использование такого подхода позволяет избавиться от необходимости использования больших объемов размеченных данных. По сути, принципиальные части алгоритма рекомендаций реализованы в качестве обучения без учителя. Итоговые результаты работы модели были оценены эмпирически, за неимением объективной метрики качества. Результаты работы модели более чем удовлетворительные.

10. Первоначальный прототип

10.1. Прототипы ранней стадии планирования

Прототипы интерфейса изначально были разработаны в программе Miro [23] (рис. 15), и затем адаптированы программно.

Прототип сервера, еще не интегрированный с Postgres базой данных, был разработан на основании модели данных (рис. 16).

Финальная реализация как интерфейса, так и серверной части несколько отличается от прототипов этапа планирования. Так, в интерфейсе были изменены описания проекта и пользователя в личном кабинете, в серверной же части например было принято решение отказаться от отдельной таблицы интересов пользователей и объединить ее с тэгами проектов. Все существенные изменения отражены в протоколах Notion.

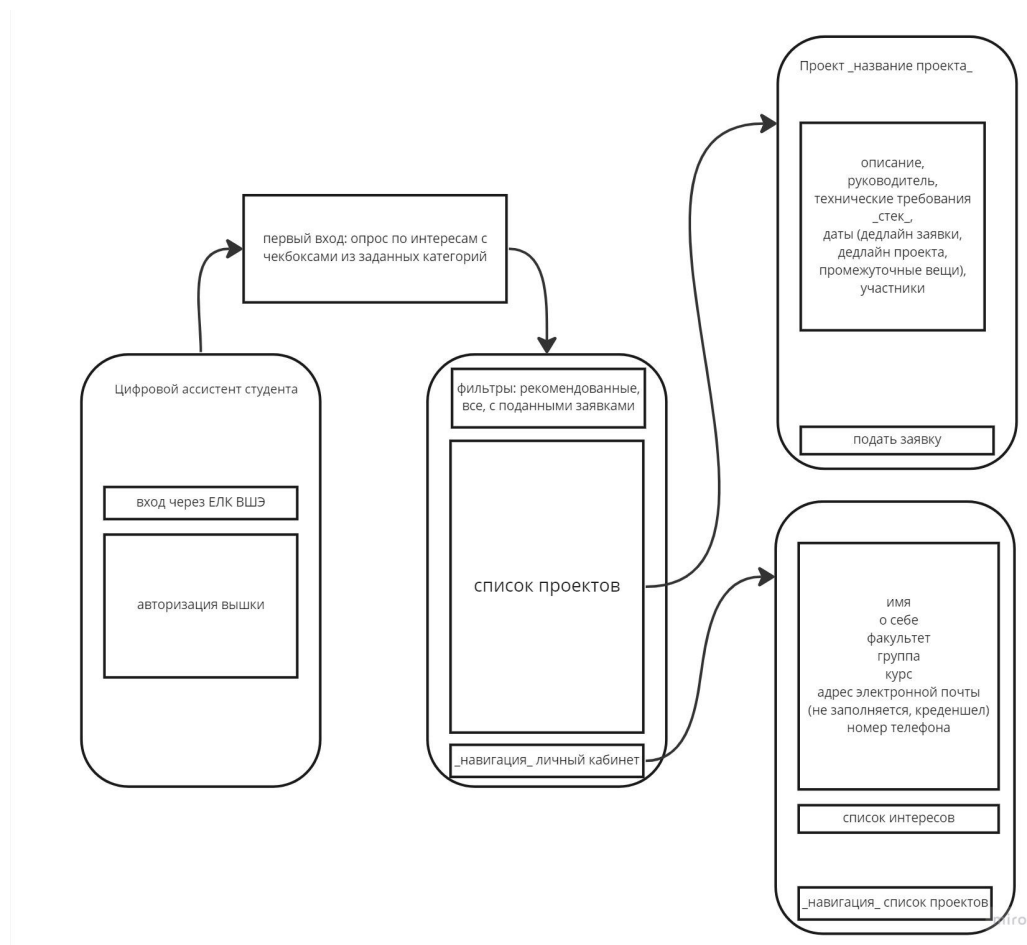


Рис. 15 - первичный прототип интерфейса

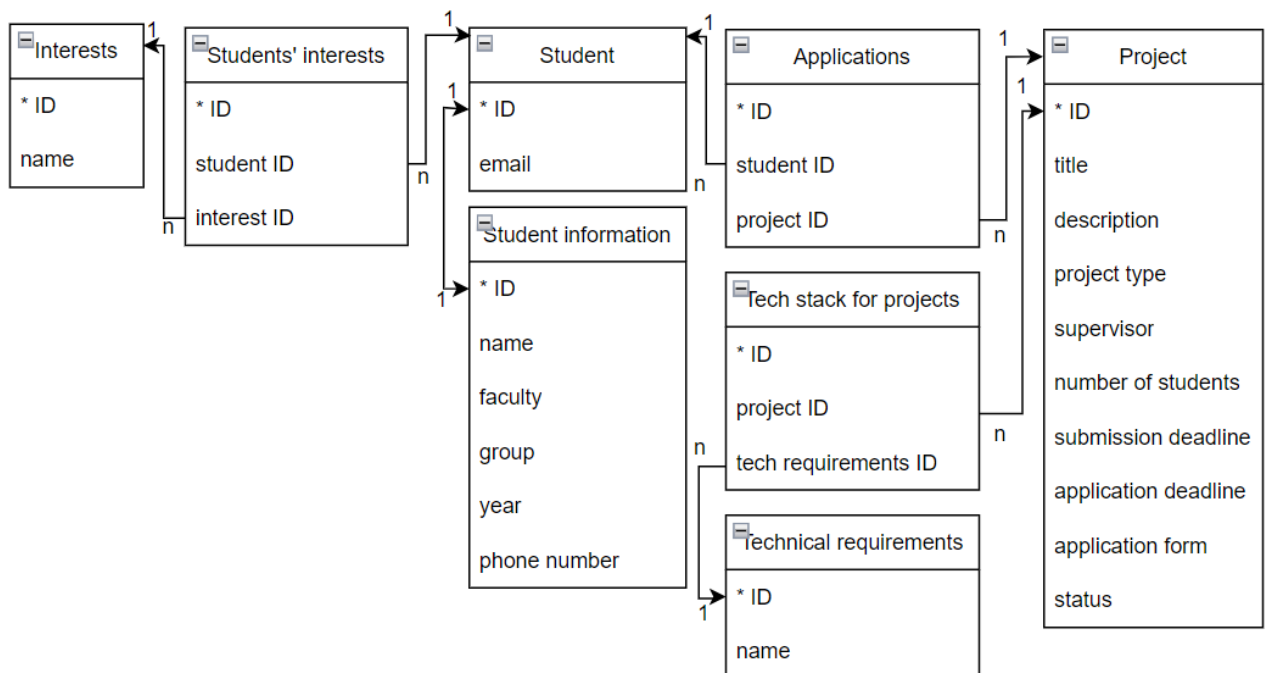


Рис. 16 - модель данных системы - основа для разработки прототипа сервера

10.2. Первые прототипы

Прототип мобильного приложения (рис. 17) и серверной системы были реализованы в первую неделю марта 2023 года и были в дальнейшем протестированы и доработаны. На тот момент они

выполняли только базовые функции, в частности еще не была реализована авторизация пользователя по учетным данным ЕЛК ВШЭ и не была подключена рекомендательная модель. Более подробное описание свойств прототипа от 15.03.2023 можно найти в документе “Тестирование и приемка работ”.

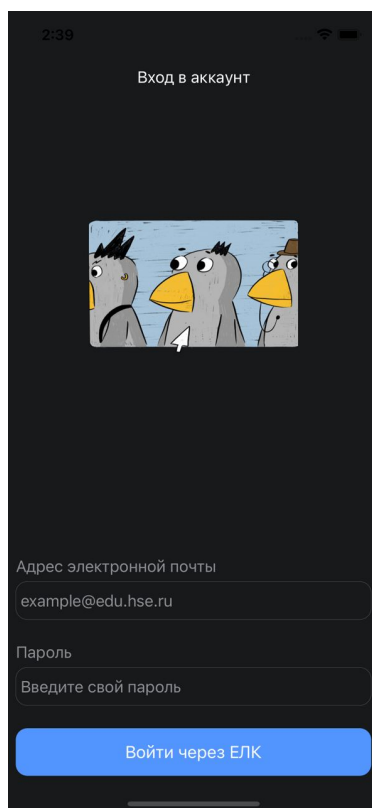


Рис. 17 - версия экрана авторизации из первого прототипа

11. Используемые технологии (Front End / Back End / GitHub / прочие)

11.1. Front End

Основополагающими технологиями при разработке мобильного приложения были стандартные инструменты Apple, такие как:

1. UIKit [27] и SwiftUI [19] для написания пользовательского интерфейса,
2. Foundation [6] для работы с основными сущностями языка Swift [18],
3. Combine [2] для реактивного программирования,
4. SwiftPM [13] был использован для подключения сторонней библиотеки Alamofire,
5. Alamofire [1] - библиотека для работы с сетью, то есть связи с сервером.

Для разработки использовалась среда XCode [30].

11.2. Back End

Для подключения к удаленному рабочему столу с сервером использовалось приложение MobaXterm [24].

Для разработки сервера был использован Django rest framework [5] с использованием библиотеки django allauth [28] для осуществления аутентификации. Разработка осуществлялась на языке Python [15] и html [12].

Базы данных написаны на языке SQL [17] на базе PostgreSQL [14]. Для разработки использовалась среда DataGrip [3].

11.3. ML

1. Python 3.8 - основной язык разработки
2. Pandas - фреймворк для работы с данными
3. Sklearn - фреймворк машинного обучения

4. NumPy - фреймворк обработки числовых данных
5. Matplotlib - фреймворк для визуализации результатов работы моделей
6. Jupyter Notebook - среда разработки

11.4. Общие инструменты

Для контроля версий разработки были использованы системы git [7] и веб-сервис GitHub [8], для управления проектом - Notion [26]. Приложения Discord [4] и Zoom [31] использовались для осуществления командных звонков, мессенджер Telegram [20] - для оперативной коммуникации. Графический веб-сервис Miro [23] был использован для создания макетов и диаграмм, также для написания документации были использованы Google Docs [9], а для централизованного хранения документов и ресурсов с доступом всей команды - Google Drive [10].

12. Предложения по монетизации/коммерциализации (либо внутр. эффект от автоматизации)

Так как приложение имеет своей целью встроиться в цифровую экосистему НИУ ВШЭ, основные модели монетизации связаны с университетскими фондами. Возможно спонсирование в рамках конкурса студенческих проектов и стартапов.

Также, так как многие внутренние процессы организации студенческой проектной деятельности будут автоматизированы, унифицированы и упрощены с помощью данной системы, ее поддержание выгодно для университета.

13. Будущее развитие проекта

Дальнейшее развитие проекта может быть связано как с развитием в ширину, так и в глубину.

С первой точки зрения, можно адаптировать приложение для других платформ: например, Android, веб-интерфейс. Имеет смысл более плотная интеграция с цифровой системой ВШЭ: получение более доступа к данным пользователям, интеграция заявок пользователей на проекты в приложении с системами LMS и SmartLMS. В конечном итоге, возможна адаптация приложения для взаимодействия с более широкими категориями проектов, использование системы для привлечения к проектной работе школьников, например из Лицей ВШЭ.

Со второй точки зрения, возможна более детальная проработка функциональности системы, добавления новых возможностей, например расширение информации в личном кабинете пользователя, добавление туда заданий со сроками по проектам, системы управления проектами. Другим возможным направлением развития приложения является увеличение социальной стороны платформы, а именно добавление чатов, возможности связи с руководителями проектов и другими студентами прямо в приложении. Также создание командного режима пользования приложением кажется разумным развитием, так как многие проекты выполняются в командах.

Наиболее вероятным является второе направление развития, так как другие группы занимаются адаптацией системы для других пользовательских платформ. Однако, расширение и детализация функций приложения (в частности, существенное усиление внимания обозначению студентом своих интересов, взаимодействию студента с проектами) планируется в рамках ВКР участника проекта Виктора Поволоцкого.

14. Отчеты о взаимодействии с заказчиком

Шаблон 1: <https://drive.google.com/drive/folders/1VAZXGNI8PAo1Jna3TO2U-7RY8Q8yDFcJ>

Шаблон 2:

<https://drive.google.com/file/d/14QAYLAS6qGIp0y4-fpHX6Vwadpf2QbFF/view?usp=sharing>

Шаблон 3: <https://drive.google.com/file/d/1z9ab3nY9xkzeIIzNAYLspWuDFmndsJQc/view?usp=sharing>

Шаблон 4:

<https://drive.google.com/file/d/1XTp08XCMZHOKm8III4KDAfVhBseKmbZV/view?usp=sharing>

Шаблон 5:

<https://drive.google.com/file/d/19niVOLwD8dNY-7SQfKpEcqayEBWcIxxa/view?usp=sharing>

Шаблон 6:

<https://drive.google.com/file/d/1BEOUYCBmwSenMKseD5khD4o-MQQwToRB/view?usp=sharing>

15. Отзыв заказчика

Отзыв

заказчика:

https://drive.google.com/file/d/16tFBKq8kmqER81GhHbVKO5bGtY-tt6oe/view?usp=share_link**16. Руководство пользователя**

Руководство по использованию всей системы и отдельных ее блоков расположено в файле ReadMe в GitHub репозитории проекта по ссылке https://github.com/aparinov/2023_DSA_iOS#readme.

Список используемой литературы

1. Alamofire // Github URL: <https://github.com/Alamofire/Alamofire> (дата обращения: 25.03.2023).
2. Combine // Apple Developer URL: <https://developer.apple.com/documentation/combine> (дата обращения: 25.03.2023).
3. DataGrip // JetBrains URL: <https://www.jetbrains.com/ru-ru/datagrip/> (дата обращения: 25.03.2023).
4. Discord // Discord URL: <https://discord.com/> (дата обращения: 25.03.2023).
5. Django REST Framework // [django-rest-framework](https://www.django-rest-framework.org/) URL: <https://www.django-rest-framework.org/> (дата обращения: 25.03.2023).
6. Foundation // Apple Developer URL: <https://developer.apple.com/documentation/foundation> (дата обращения: 25.03.2023).
7. git // git URL: <https://git-scm.com/> (дата обращения: 25.03.2023).
8. Github // Github URL: <https://github.com/> (дата обращения: 25.03.2023).
9. Google Docs // Google URL: <https://www.google.com/docs/about/> (дата обращения: 25.03.2023).
10. Google Drive // Google URL: https://www.google.com/intl/ru_ru/drive/ (дата обращения: 25.03.2023).
11. HSE App X // Apple Apps URL: <https://apps.apple.com/ru/app/hse-app-x/id1527320487> (дата обращения: 25.03.2023).
12. HTML // Wikipedia URL: <https://en.wikipedia.org/wiki/HTML> (дата обращения: 25.03.2023).
13. Package Manager // Swift URL: <https://www.swift.org/package-manager/> (дата обращения: 25.03.2023).
14. PostgreSQL: The World's Most Advanced Open Source Relational Database // PostgreSQL URL: <https://www.postgresql.org/> (дата обращения: 25.03.2023).
15. Python // Python URL: <https://www.python.org/> (дата обращения: 25.03.2023).
16. SmartLMS // it.hse URL: <https://it.hse.ru/digitalreport/smartlms> (дата обращения: 25.03.2023).
17. SQL // Wikipedia URL: <https://en.wikipedia.org/wiki/SQL> (дата обращения: 25.03.2023).
18. Swift // Apple URL: <https://www.apple.com/ru/swift/> (дата обращения: 25.03.2023).
19. SwiftUI // Apple URL: <https://developer.apple.com/xcode/swiftui/> (дата обращения: 25.03.2023).
20. Telegram Desktop // Telegram URL: <https://desktop.telegram.org/?setln=en> (дата обращения: 25.03.2023).
21. The MoSCoW prioritization method explained // MondayBlog URL: [https://monday.com/blog/project-management/moscow-prioritization-method/#:~:text=What%20are%20the%20categories%20within,have%20\(this%20time\).%E2%80%9D](https://monday.com/blog/project-management/moscow-prioritization-method/#:~:text=What%20are%20the%20categories%20within,have%20(this%20time).%E2%80%9D) (дата обращения: 25.03.2023).
22. LMS // LMS URL: <https://lms.hse.ru/> (дата обращения: 25.03.2023).
23. Miro // Miro URL: <https://miro.com/ru/> (дата обращения: 25.03.2023).
24. MobaXterm // MobaXterm URL: <https://mobaxterm.mobatek.net/> (дата обращения: 25.03.2023).
25. MVVM+Coordinators iOS Architecture Tutorial // Medium URL: <https://medium.com/nerd-for-tech/mvvm-coordinators-ios-architecture-tutorial-fb27eaa36470> (дата обращения: 25.03.2023).
26. Notion for Mac and Windows // Notion URL: <https://www.notion.so/desktop> (дата обращения: 25.03.2023).
27. UIKit // GetUIKit URL: <https://getuikit.com/> (дата обращения: 25.03.2023).
28. Welcome to django-allauth! // django-allauth URL: <https://django-allauth.readthedocs.io/en/latest/> (дата обращения: 25.03.2023).
29. What is Agile? // Atlassian URL: <https://www.atlassian.com/agile> (дата обращения: 25.03.2023).
30. XCode // Apple URL: <https://developer.apple.com/xcode/> (дата обращения: 25.03.2023).
31. Zoom // Zoom URL: <https://zoom.us/> (дата обращения: 25.03.2023).