Пермский филиал федерального государственного автономного
образовательного учреждения высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

*Факультет экономики, менеджмента и бизнес-информатики*

Соломатин Роман Игоревич

# РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ ПОИСКА ИСПОЛНИТЕЛЕЙ ПО ТЕХНИЧЕСКОМУ ЗАДАНИЮ ПРИКЛАДНОГО ПРОЕКТА

*Курсовая работа*

студента образовательной программы «Программная инженерия»
по направлению подготовки *09.03.04 Программная инженерия*

Руководитель
к.т.н., доцент кафедры Информационных технологий в бизнесе НИУ ВШЭ-Пермь

А. В. Бузмаков

Пермь, 2021 год

# Аннотация

# Оглавление

# Введение

Каждый день выкладывается несколько десятков тендеров, сроки участия в которых очень сжаты, и потенциальным исполнителям надо быстро определиться смогут они выполнить проект или нет. Для принятия надо ознакомиться с проектом, собрать команду профессионалов для участия и подготовить заявку. Это все очень сложно успеть за месяц. В «Высшей школе экономики» много людей с разными компетенциями, и надо по текстовому описанию задачи понять, кто его сможет сделать. Для этого нужно определить, в чем каждый из работников университета компетентен.

В рамках ВШЭ преподаватели пишут в основном 2 вида научных работ – статьи в научные журналы и выпускные курсовые работы, которые пишут студенты под их руководством. Анализируя эти тексты можно выделить сферы интересов преподавателей. Для этого были выбраны ВКР, потому что для доступа ко многим научным журналам требуется платная подписка и все работы находятся на многих разных сайтах, с которыми не удобно работать, также статьи часто пишутся на разных языках, что тоже затрудняет работу. А выпускные курсовые работы похожи на статьи, находятся в свободном доступе и содержат много текста для получения компетенций. Таким образом, можно получить сферу компетенций преподавателя, по которой в дальнейшем искать соответствие между пришедшем текстовым описанием задачи и профилем преподавателя. Таким образом, есть проблемы поиска исполнителей на проект.

В этой работе будет проверятся гипотеза можно ли из выпускных курсовых работ студентов получить сферу компетенций преподавателя по которой в дальнейшем искать соответствие с текстовым описанием задачи.

Объект исследования - процесс поиска исполнителей по текстовому описанию задачи.

Предмет исследования - автоматизация процесса из объекта.

Цель работы – создать информационную систему для поиска исполнителя по текстовому описанию задачи.

Для достижения поставленной цели нужно сделать:

В первой главе постановка задачи.

Во второй главе описание проектирования системы.

В третьей главе пример работы приложения.

# Глава 1. Анализ предметной сферы

## 1.1. Обзор существующих решений

Не существует программных продуктов, которые решают данную задачу в явном виде. Потому что это очень особенная ситуация, когда есть много исполнителей с разными компетенциями и надо для них подбирать задания. Также редко в каких организациях по текстам, которые пишут исполнители можно представить их компетенции. Сейчас это проблема решается вручную.

Приходит в ВШЭ много технических заданий, из отбирает человек, который знает компетенции многих исполнителей. После этого если он думает, что компетенции исполнителя подходят для проекта, то пишет ему. Сотрудник отвечает готов или не готов. Потом за короткий промежуток времени (обычно месяц) нужно подготовить заявку на проект, задать уточняющие вопросы организатору, найти недостающих исполнителей. Данный процесс занимает много времени. Эта система имеет недостатки:

- Много проектов теряется, так как человек не знает все компетенции всех исполнителей
- Сложно искать проекты
- Тяжело масштабировать, потому что необходимо знать много про разных людей
- Тратится много времени

Иногда применятся способ массовой рассылки требующихся исполнителей для проекта. У данного подхода недостатки:

- Массовую рассылку не все читают
- Не дает полную информацию

Разрабатываемая система поможет автоматизировать процессы:

- Определения компетенций сотрудника
- Поиск сотрудников для выполнения задания

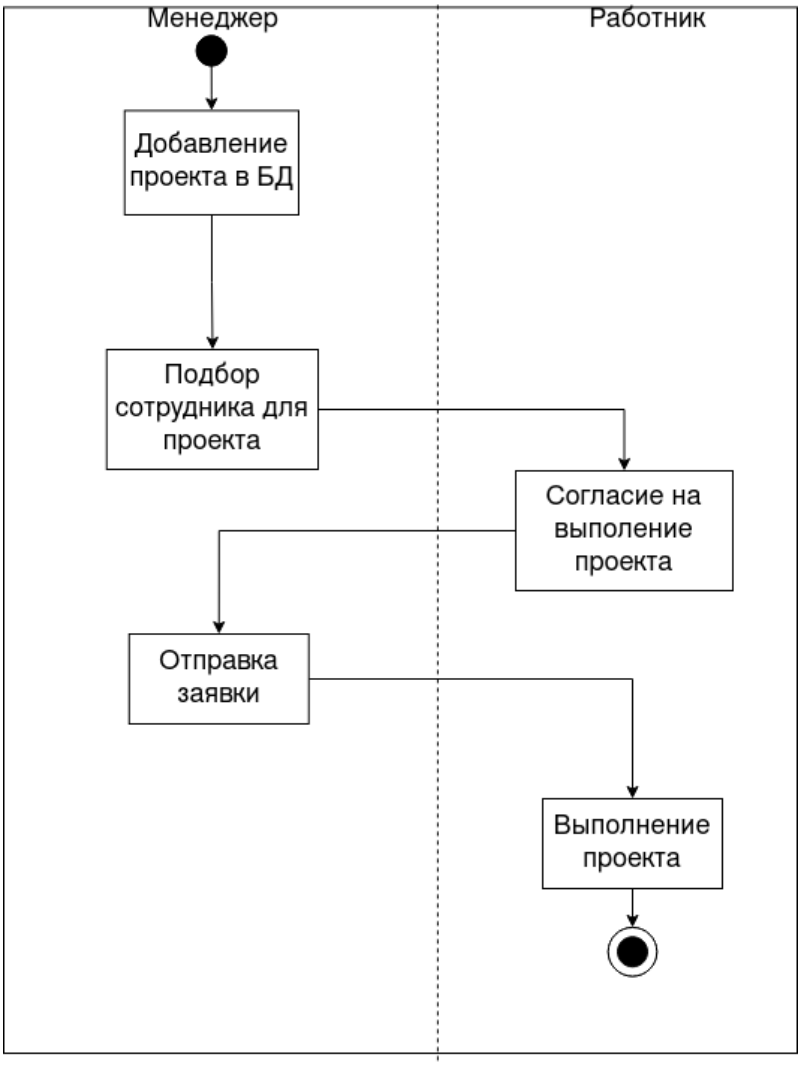


**Рис. 1.1. Автоматизация бизнес-процесса "Подбор сотрудника для проекта"**

Варианты использования реализуемой системы:

- Просмотр активных проектов
- Поиск компетенций исполнителя
- Подбор сотрудника для проекта
- Редактирование информации о исполнителя

Так как в данной работе нужно проверить гипотезу, будет разрабатываться прототип программы. Диаграмма вариантов использования конечного программного продукта 1.2.

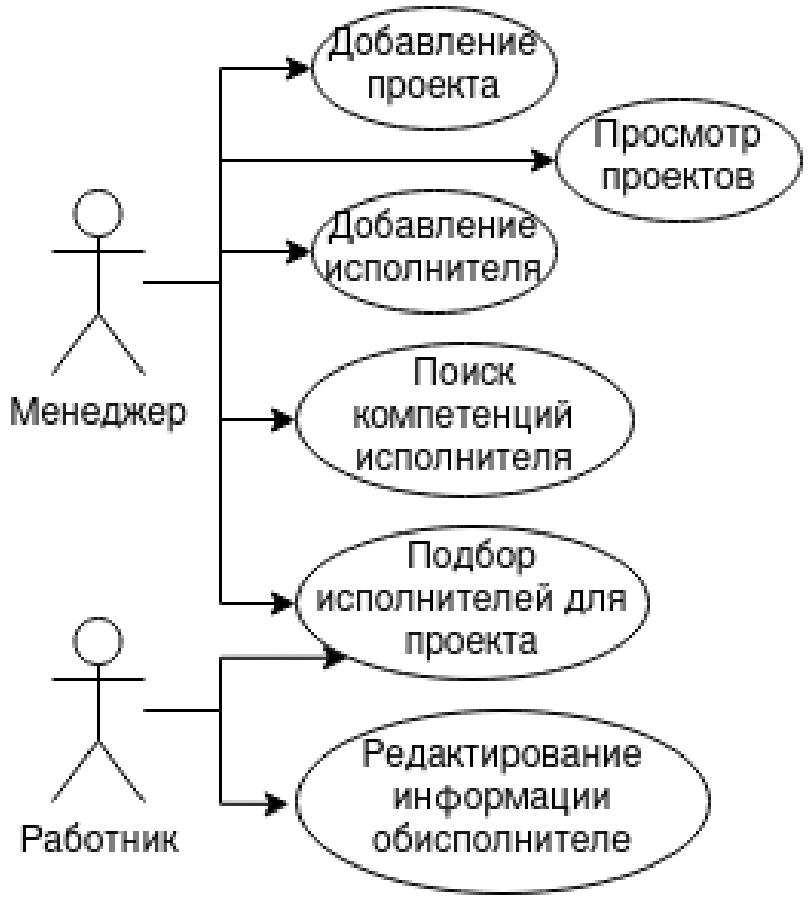


**Рис. 1.2. Диаграмма вариантов использования**

#### 1.1.1. Описание прецедента «Добавление проекта»

Предоставляется текстовое описание проекта и потом для него подбираются исполнители из базы данных.

#### 1.1.2. Описание прецедента «Просмотр проектов»

Выдаётся список проектов, на которые были отправлены заявки об участии в проекте.

#### 1.1.3. Описание прецедента «Добавление исполнителя»

Добавление исполнителя в базу данных, и запись данных о нем.

#### 1.1.4. Описание прецедента «Поиск компетенций исполнителя»

Создание компетенций для заданного человека. Для этого надо обработать тексты ВКР с его профиля на сайте «Высшей школы экономики», которые хранятся в формате docx, doc или pdf.

1.1.5. Описание прецедента «Подбор исполнителя для проекта»

Подбор исполнителя под текстовое описание проекта.

1.1.6. Описание прецедента «Редактирование информации об исполнителе»

Редактирование данных исполнителя.

1.1.7. Описание прототипа

Основной целью данной работы является проверка гипотезы. Поэтому будет разрабатываться прототип, который будет включать в себя только следующие функции:

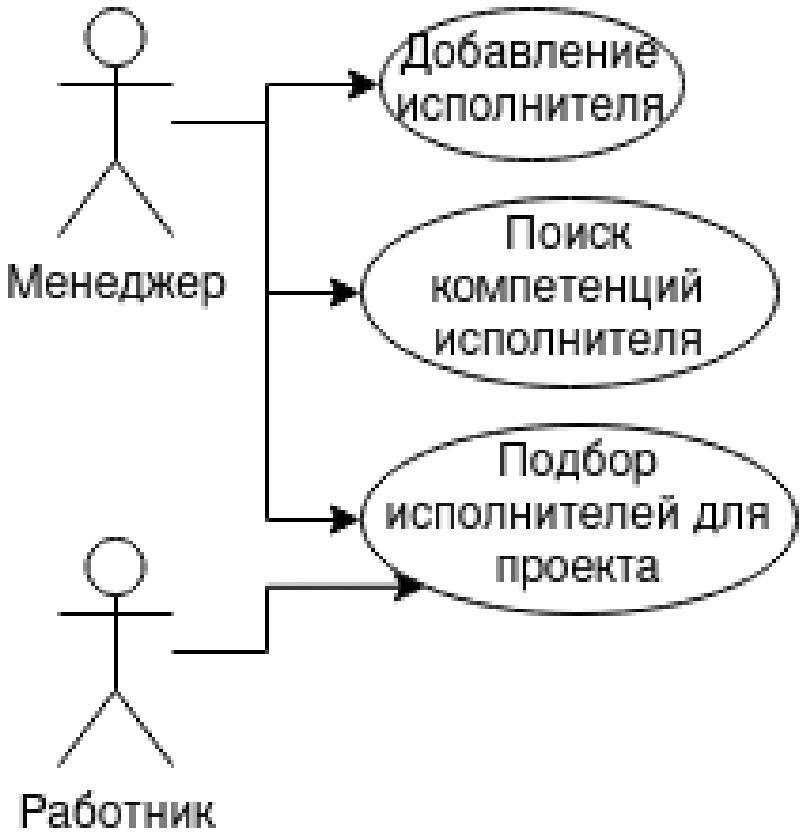


**Рис. 1.3. Диаграмма вариантов использования**

- Добавление исполнителей
- Поиск компетенций исполнителей

## 1.2. Выбор языка программирования

Для прототипирования подходят многие языки программирования, вот некоторые из них: Python, C#, Java, JavaScript. Определимся с основными критериями для выбора языка:

- Библиотеки для обучения моделей. Для работы с моделями машинного обучения.
- Предобученные модели. Так как для обучения модели надо иметь большой объем данных и много вычислительных мощностей.
- Библиотеки для обработки сайта.
- Библиотека для обработки docx и pdf файлов и обработки сайтов.
- Интерактивный режим. Позволяет просмотреть данные, преобразовать их, вмешаться в исполнение кода. Выполнить в произвольном порядке.

Таблица 1.1. Таблица сравнения языков программирования

| Язык программирования | Библиотека для МО | Предобученные модели | Работа с сайтом | Работа с файлами | Интерактивный режим |
|---|---|---|---|---|---|
| Python | ++ | ++ | + | + | + |
| C# | + | + | + | + | - |
| Java | + | + | + | + | - |
| JavaScript | + | + | + | + | ? |

Под эти критерии подходит Python. Он простой для написание и отладки, так как это интерпретируемый язык программирования, у него есть интерактивный режим. Также есть фреймворк PyThorch и библиотека с предобученными моделями HuggingFace Transformers, в частности RuBERT [1], а также есть библиотека Bert Extractive Summarizer [2] для удобной работы с моделей.

## 1.3. Выбор СУБД

Для Python существуют библиотеки для работы с любыми системами управления базами данных. Для курсовой работы был выбран SQLite потому

что его легко настраивать, нет необходимости устанавливать ничего дополнительно, что достаточно для прототипирования с небольшим количеством данных. 300 преподавателей и 1700 выпускных квалификационных работ.

# Глава 2. Проектирование системы

## 2.1. Проектирование базы данных

Для разработки системы необходимо хранить информации о преподавателях и выпускных квалификационных работах, где они были руководителями была разработана база данных. Необходимо было хранить:

**Код преподавателя** – уникальный код преподавателя для его идентификации в базе данных

**ФИО преподавателя** – фамилия имя и отчество преподавателя

**Ссылка на профиль преподавателя** – ссылка на профиль преподавателя на сайте Высшей школы экономики, для последующей проверки в ручную при подборе на проект

**Код статуса** – код статуса для последующей связи в базе данных

**Код кафедры** – код кафедры для последующей связи в базе данных

**Компетенции** – компетенции преподавателя полученные путём автоматического анализа текстов ВКР в текстовом виде

**Эмбеддинги** – компетенции преподавателя полученные путём автоматического анализа текстов ВКР в векторном пространстве для дальнейшего подбора исполнителя для проекта

**Код статуса** – уникальный код статуса для идентификации в базе данных

**Наименования статуса** – доцент, старший научный сотрудник и тд. Для представление информации о преподавателе.

**Код кафедры** – уникальный код кафедры для идентификации в базе данных

**Наименование кафедры** – название кафедры, где работает преподаватель. Для представление информации о преподавателе.

**Код кампуса** – уникальный код для идентификации в базе данных

**Наименования кампуса** – название кампуса. Для представление информации о преподавателе.

**Код ВКР** – уникальный код выпускной квалификационной работы для идентификации в базе данных

**Название ВКР** – название выпускной квалификационной работы. Необходимо для дальнейшего соотнесения в преподвателя ручном режиме

**Научный руководитель** – код преподавателя для последующей связи в базе данных

**Ссылка на ВКР** – ссылка на выпускную квалификационную работу на сайте ВШЭ для проверки корректности сбора информации

**Ссылка на полный текст ВКР** – ссылка на файл для загрузки с полным текстом ВКР

**ФИО студента** – фамилия имя и отчество студента написавшего ВКР

**Код ОП студента** – код образовательной программы студента для последующей связи в базе данных

**Код кампуса** – код кампуса для последующей связи в базе данных

**Код образовательной программы** – уникальный код для идентификации в базе данных

**Код факультета** – код факультета для последующей связи в базе данных

**Наименование ОП** – название образовательной программы, где обучается студент

**Код факультета** – уникальный код факультета для идентификации в базе данных

**Наименование факультета** – название факультета

Описание данных для проектирования БД 2.1.

Таблица 2.1. Таблица атрибутов

| Имя атрибута | Тип данных | Значение по умолчанию | Формат ввода | Ограничение на значения |
|---|---|---|---|---|
| Код преподавателя | Число | Нет | Нет | Нет |
| ФИО преподавателя | Строка | Нет | Нет | Нет |
| Ссылка на профиль преподавателя | Строка | Нет | Нет | Нет |
| Компетенции | Строка | Нет | Нет | Нет |
| Эмбеддинги | Число | Нет | Нет | Нет |
| Код ВКР | Число | Нет | Нет | Нет |
| Название ВКР | Строка | Нет | Нет | Нет |
| Ссылка на ВКР | Строка | Нет | Нет | Нет |
| Ссылка на полный текст ВКР | Строка | Нет | Нет | Нет |
| ФИО студента | Строка | Нет | Нет | Нет |
| Код ОП студента | Число | Нет | 00.00.00 | Нет |
| Код статуса | Число | Нет | Нет | Нет |
| Наименования статуса | Строка | Нет | Нет | Нет |
| Код кафедры | Число | Нет | Нет | Нет |
| Наименование кафедры | Строка | Нет | Нет | Нет |
| Код кампуса | Число | Нет | Нет | Нет |
| Наименования кампуса | Строка | Нет | Нет | Нет |
| Код образовательной программы | Число | Нет | 00.00.00 | Нет |
| Код факультета | Число | Нет | Нет | Нет |
| Наименование ОП | Строка | Нет | Нет | Нет |
| Код факультета | Число | Нет | Нет | Нет |
| Наименование факультета | Строка | Нет | Нет | Нет |

### 2.1.1. Приведение к 1НФ

Отношение находится в первой нормальной форме, если выполнены все свойства реляционных отношений, в частности все атрибуты отношения принимают простые значения (атомарные или неделимые), не являющиеся множеством или кортежем из более элементарных составляющих, все кортежи уникальны (отсутствуют дубли).

1. Код преподавателя
2. ФИО преподавателя
3. Ссылка на профиль преподавателя
4. Компетенции
5. Эмбеддинги
6. Код статуса преподавателя
7. Код кафедры преподавателя
8. Код ВКР
9. Название ВКР
10. Код преподавателя
11. Ссылка на ВКР
12. Ссылка на полный текст ВКР
13. ФИО студента
14. Код кампуса
15. Код ОП студента
16. Код статуса
17. Наименование статуса
18. Код кафедры

19. Наименование кафедры
20. Код кампуса
21. Наименования кампуса
22. Код образовательной программы
23. Код факультета
24. Наименование ОП
25. Код факультета
26. Наименование факультета

Данные атрибуты находятся в 1НФ.

### 2.1.2. Приведение к 2НФ

Отношение находится во второй нормальной форме, если оно находится в первой нормальной форме и каждый неключевой атрибут функционально полно зависит от всего ключа в целом, то есть отсутствует частичная функциональная зависимость не ключевых атрибутов от ключа. В соответствии с описанными выше зависимостями можно сделать вывод, что в описанном отношении имеются следующие частичные зависимости:

1. Код преподавателя определяет:
    - ФИО преподавателя
    - Ссылка на профиль преподавателя
    - Компетенции
    - Эмбеддинги
    - Код статуса преподавателя
    - Код кафедры преподавателя
2. Код ВКР определяет:
    - Название ВКР
    - Код преподавателя

- Ссылка на ВКР
- Ссылка на полный текст ВКР
- ФИО студента
- Код кампуса
- Код ОП студента

3. Код статуса преподавателя определяет:
   - Наименование статуса
4. Код факультета определяет:
   - Наименование факультета
5. Код кампуса определяет:
   - Наименование кампуса
6. Код кафедры определяет:
   - Наименование кафедры
7. Код образовательной программы определяет:
   - Название образовательной программы

#### 2.1.3. Приведение к 3НФ

Отношение находится в третьей нормальной форме, если оно находится во второй нормальной форме, и каждый неключевой атрибут не является транзитивно зависимым от первичного ключа.

## 2.2. Сбор данных

Для работы программы необходимо собрать данные о преподавателях и ВКР, которые у них писались. Для этого надо было обрабатывать информация находящуюся на сайте ВШЭ. Такая обработка возможна с помощью библиотеки BeautifulSoap, который позволяет получать html любого сайта.

В первую очередь надо было получить список преподавателей и ссылки на их страницы. Для этого обрабатывался сайт со списком преподавателей, но там можно получить преподавателей, фамилии которых начинаются на 1

букву. Что бы решить эту проблему с сайты получались ссылки на буквы. А потом с каждой такой ссылки собирался список преподавателей и ссылки на их страницы. Для этого брались все элементы страницы с атрибутом "a" и классом "link" , в которых значение атрибута "href" было "/org/persons/" или "/staff/". В базу данных записывались ФИО преподавателя и ссылку на их страницу.

Далее надо было получить список ВКР для каждого преподавателя. Для этого брались все элементы страницы с атрибутом "a" и классом "link" , в которых значение атрибута "href" было "/edu/vkr/". Итого получилось собрать информацию о 321 преподавателе и 1708 ВКР.

Потом надо было получить информацию о ВКР, но их данные ВКР, в отличие от данных преподавателей, подгружались после загрузки основой страницы. Чтобы решить эту проблему пришлось эмулировать работу браузера с помощью библиотеки Selenium и драйвера Gecko (движок браузера FireFox). Библиотека Selenium не использовалась раньше, так как для ее работы надо больше времени и ресурсов, поэтому данная библиотека использовалась только на данном этапе. Для полной загрузки страницы ставилась задержка в 2 секунды. И бралась информация атрибута "p" с классом "vkr-card__item". Также бралась информация о научном руководителе, он искался в базе данных, и потом в базу данных уже записывался Код преподавателя.

Чтобы получить тексты из файлов, они обрабатывались с помощью textract, которая позволяет получить информацию из pdf, doc и docx файлов, и помещались в базу данных.

## 2.3. Создание профиля человека

Для создание профиля человека рассматривались TFiDF и суммаризация текста с помощью BERT[3] (Bidirectional Encoder Representations from Transformers).

Подход TFiDF заключается в том что бы для каждого слова посчитать его отношение числа вхождений некоторого слова к общему числу слов документа (TF) и частоты, с которой некоторое слово встречается в документах коллекции (DF). Данный подход часто используется для кластеризации текстов. Но данный метод плохо подходит для данной задачи, потому что он не рассматривает контекст используемых слов. Это значит у него проблемы с омонимами и синонимами.

BERT – это нейросетевая языковая модель, которая рассматривает контекст, в котором находится слово и кодирует данное слово в своём векторном пространстве. Для этого модель обрабатывает предложения и закрывает 15% слов маской

Для получение компетенций преподавателя использовались тексты ВКР, которые у него писали студенты. Для этого обрабатывался сайты Высшей школы экономики и скачивались тексты ВКР. Потом эти тексты группировались по преподавателю и обрабатывались с помощью BERT Extractive summarizer, в качестве модели для обучения использовался предобученный RuBERT от DeepPavlovAI.

BERT Extractive summarizer [2] – использует предпредпоследний слой модели в качестве данных. Потом использует алгоритм K-Means для кластеризации текстов.

# Глава 3. Применение системы

# Заключение

# Библиографический список


1. *Kuratov Y.*, *Arkhipov M.* Adaptation of Deep Bidirectional Multilingual Transformers for Russian Language. — 2019.
2. *Miller D.* Leveraging BERT for Extractive Text Summarization on Lectures. — 2019.
3. *Devlin J.*, *Chang M.-W.*, *Lee K.*, *Toutanova K.* BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. — 2019.

# Приложения