# StRS (Stakeholder Requirements Specification)

### 1.1 Введение

**Цель документа** — формализовать и структурировать требования заинтересованных сторон к программной системе **«ContinueWork: Сервис автоматизации отбора кандидатов на основе обработки естественного языка»** , определяя её основные цели и задачи с точки зрения пользователей, заказчиков и других участников процесса.

#### 1.2 Область применения

**Целевая среда использования** Система **«ContinueWork: Сервис автоматизации отбора кандидатов на основе обработки естественного языка»** предназначена для применения в следующих контекстах:

* **HR-отделы компаний** для первичного фильтрации резюме и вакансий.
* **Кадровые агентства** и **джоб-борды** для автоматизации анализа текстовых данных.
* **Информационные системы подбора персонала** , интегрированные с платформами поиска работы (например, hh.ru, LinkedIn).

**Границы системы** Система включает следующие компоненты:

1. **Модуль обработки текстов** :
   * Векторизация резюме и вакансий с использованием модели **BERT** .
   * Извлечение ключевых навыков, опыта и требований из неструктурированных текстов.
2. **Система сопоставления** :
   * Ранжирование кандидатов по семантической близости к вакансиям.
   * Генерация оценки соответствия (например, вероятность в диапазоне [0, 1]).
3. **Интеграционные интерфейсы** :
   * API для взаимодействия с внешними HR-платформами.
   * Поддержка форматов данных: JSON для метаданных, векторные массивы для embeddings.

**Исключенные функции** Система **не** включает:

* Оценку «мягких» навыков (коммуникабельность, лидерство) на основе текстовых данных.
* Обработку не-текстовых данных (например, аудио/видео).
* Формирование финальных решений о найме — рекрутеры остаются ответственными за окончательную экспертизу.

**Контекст использования** Система применяется на этапе **первичного отбора кандидатов** , где:

* **Входные данные** :
  + Текстовые данные резюме и вакансий в форматах PDF, DOCX, TXT.
  + Метаданные (например, категория вакансии, дата публикации).
* **Выходные данные** :
  + Отсортированный список кандидатов с оценкой соответствия.
  + Векторные представления (embeddings) для хранения в базе данных.
  + Отчёты о процессе отбора (PDF/CSV).

**Технические ограничения**

1. **Форматы** : Поддержка только текстовых форматов (PDF, DOCX, TXT).
2. **Языки** : Обработка текстов на **русском** и **английском** языках.
3. **Интеграция** : Требует подключения к внешним СУБД (PostgreSQL, Weaviate) и HR-платформам через REST API.

**Логические границы** Система взаимодействует с внешними системами, но не включает их в свою архитектуру:

* **Внешние системы** :
  + Базы данных (хранение структурированных и векторных данных).
  + Платформы поиска работы для импорта/экспорта резюме и вакансий.
* **Исключённые процессы** :
  + Публикация вакансий в интернете.
  + Организация интервью и финальное решение о найме.

**Ограничения функциональности**

* Не учитывает неявные факторы (например, рекомендации соискателей, портфолио).
* Не поддерживает реального времени (RT) обработку запросов — рассчитана на пакетный и интерактивный режимы (от 0.5 до 2 секунд на запрос).

#### 1.3 **Определения**

Термины:

1. Автоматизация отбора кандидатов — процесс использования технологий для первичной фильтрации резюме и вакансий, минимизирующий ручную работу HR-специалистов.
2. BERT (Bidirectional Encoder Representations from Transformers) — нейронная сеть, использующая двунаправленную обработку текста для извлечения семантических векторов, применяемая для анализа резюме и вакансий.
3. HR-система — программное обеспечение для управления персоналом, включая подбор, анализ и сопоставление кандидатов.
4. Контекстуальный анализ — метод обработки текста, учитывающий контекст слов и предложений для точного понимания смысла.
5. Матрица корреляции Мэттьюса (MCC) — метрика, оценивающая качество классификации, особенно при несбалансированных данных (значение от -1 до 1, где 1 — идеальное соответствие).
6. NLP (Natural Language Processing) — область ИИ, занимающаяся обработкой и анализом естественного языка для извлечения полезной информации.
7. Первичный отбор — этап рекрутмента, где резюме фильтруются на соответствие базовым требованиям вакансии.
8. Семантическое сопоставление — сравнение текстов резюме и вакансий на основе их векторных представлений (embeddings) для определения соответствия.
9. СУБД (Система управления базами данных) — программное обеспечение для хранения, управления и обработки данных, включая реляционные (PostgreSQL) и векторные (Weaviate) базы.
10. Векторная СУБД — СУБД, оптимизированная для хранения и поиска векторных представлений текстов (например, Weaviate).
11. Weaviate — векторная СУБД, поддерживающая поиск ближайших соседей (ANN) и индексацию embeddings.
12. PostgreSQL — реляционная СУБД с поддержкой JSON и ACID-свойств для хранения структурированных данных о вакансиях и резюме.
13. RESTful API — интерфейс взаимодействия между компонентами системы, основанный на принципах REST (Representational State Transfer).
14. Микросервисная архитектура — структура системы, состоящая из независимых сервисов, взаимодействующих через API (например, модули векторизации и сопоставления).
15. Fine-tuning — этап переподгона предобученной модели (BERT) под конкретные задачи, такие как анализ резюме.
16. Embeddings (векторные представления) — числовые векторы, отражающие семантическую информацию текста (резюме или вакансии).
17. Косинусное расстояние — метрика, измеряющая близость векторных представлений текстов для оценки соответствия.
18. TO BE — оптимизированная модель процесса рекрутмента после внедрения системы.
19. AS IS — текущая модель процесса рекрутмента до внедрения системы.

Сокращения:

* API — Application Programming Interface.
* BERT — Bidirectional Encoder Representations from Transformers.
* GDPR — General Data Protection Regulation (регламент защиты персональных данных).
* IEEE — Institute of Electrical and Electronics Engineers.
* ML — Machine Learning (машинное обучение).
* MVP — Minimum Viable Product (минимальный жизнеспособный продукт).
* NLP — Natural Language Processing.
* REST — Representational State Transfer.
* SIRS — Stakeholder Requirements Specification.
* TO BE — Оптимизированная модель процесса.
* AS IS — Текущая модель процесса.
* Weaviate — Векторная СУБД.
* PostgreSQL — Реляционная СУБД.

#### 1.4 Ссылки

1. **Регламент GDPR (General Data Protection Regulation)** — Устанавливает правила защиты персональных данных пользователей системы, особенно при работе с резюме и вакансиями.
2. **IEEE Transactions on Pattern Analysis and Machine Intelligence** — Научное издание, содержащее статьи о методах поиска ближайших соседей (например, HNSW) для работы с векторными представлениями текстов [24].
3. **PostgreSQL Documentation** — Официальная документация PostgreSQL, используемой в качестве реляционной СУБД для хранения структурированных данных о резюме и вакансиях [21].
4. **Weaviate Documentation** — Официальная документация Weaviate, применяемой как векторная СУБД для хранения и анализа embeddings текстовых данных [23].
5. **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding** — Исследование Devlin et al., описывающее архитектуру BERT, используемую для семантического анализа резюме и вакансий [6].
6. **Domain-Driven Design: Tackling Complexity in the Heart of Software** — Книга Эванса, описывающая подход DDD, примененный для проектирования архитектуры сервиса [14].
7. **Scikit-learn Documentation** — Документация библиотеки scikit-learn, использованной для расчета метрик MCC и accuracy [27].
8. **C4 Model for Visualising Software Architecture** — Методология визуализации архитектуры программного обеспечения, примененная для описания компонентов системы [28].
9. **Creating Microservices: Designing Fine-Grained Systems** — Книга Ньюмана, описывающая принципы микросервисной архитектуры, выбранной для реализации системы [17].
10. **HireVue Official Website** — Информация о коммерческом решении HireVue, использованная для сравнительного анализа существующих систем автоматизации рекрутмента [1].
11. **Textkernel Official Website** — Информация о системе Textkernel, примененной для анализа аналогичных решений на рынке [2].
12. **Hiretual Official Website** — Информация о платформе Hiretual, использованная для изучения современных подходов к автоматизации отбора кандидатов [3].
13. **RESTful API Design Guidelines** — Рекомендации по разработке REST API, примененные для взаимодействия между компонентами системы [18].
14. **Unified Modeling Language (UML) Standard** — Стандарт UML, использованный для создания диаграмм состояний и компонентов системы [20].
15. **Business Process Model and Notation (BPMN) Standard** — Стандарт BPMN, примененный для моделирования бизнес-процессов AS IS и TO BE рекрутмента [11].
16. **Pinecone Documentation** — Официальная документация Pinecone, рассматриваемой как альтернатива Weaviate для работы с векторными данными [23].
17. **PyTorch Documentation** — Документация PyTorch, применяемой для обучения моделей машинного обучения [28].
18. **Docker Documentation** — Официальная документация Docker, используемая для контейнеризации компонентов системы [4.1].
19. **Kubernetes Documentation** — Официальная документация Kubernetes, применяемая для оркестрации микросервисов [4.1].
20. **Пояснительная записка по теме “Программная реализация сервиса для автоматизации отбора”** — Документ, содержащий результаты исследований, экспериментов и технических решений, на основе которых разработано данное SIRS.

### 2. Бизнес-требования

#### 2.1 Бизнес-цели

Целью разработки программного сервиса для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий является повышение эффективности процесса рекрутмента за счет минимизации влияния человеческого фактора, ускорения анализа текстовых данных и увеличения точности сопоставления кандидатов с требованиями работодателей. Система должна поддерживать следующие бизнес-цели:

1. **Сокращение временных затрат на первичный отбор кандидатов:**
   * Автоматизация анализа большого объема резюме позволит HR-специалистам сократить время на первичную фильтрацию соискателей.
   * Ожидаемый результат: снижение времени на анализ одного резюме до **2 секунд** , что позволяет обрабатывать до **10 000 записей в час** .
2. **Повышение качества отбора кандидатов:**
   * Применение семантического анализа с использованием модели BERT обеспечит учет контекста при сравнении навыков кандидатов с требованиями вакансии.
   * Ожидаемая метрика качества: точность классификации резюме (accuracy) — **75%** , коэффициент корреляции Мэтьюса (MCC) — **0.55** .
3. **Оптимизация использования ресурсов:**
   * Снижение нагрузки на отделы кадров благодаря автоматизации рутинных задач позволит перенаправить усилия HR-специалистов на более стратегические процессы, такие как интервьюирование и оценка soft skills.
   * Ожидаемый эффект: экономия **20–30%** рабочего времени HR-персонала.
4. **Стандартизация процесса рекрутмента:**
   * Внедрение системы обеспечит единые критерии оценки релевантности кандидатов, что минимизирует ошибки, связанные с субъективностью при ручной фильтрации.
   * Результатом станет улучшение прозрачности процесса найма и исключение дискриминационных факторов.
5. **Расширение возможностей для кадровых агентств и джоб-бордов:**
   * Интеграция сервиса с существующими платформами поиска работы (например, hh.ru, LinkedIn Talent Solutions) позволит компаниям и кадровым агентствам масштабировать свои операции без значительного увеличения затрат на персонал.
   * Поддержка обработки резюме на **русском** и **английском** языках сделает систему универсальной для международных и локальных рынков труда.
6. **Увеличение удовлетворенности клиентов:**
   * Ускорение процесса отбора кандидатов позволит работодателям быстрее находить подходящих специалистов, что сократит время простоя вакансий.
   * Для соискателей система обеспечит более справедливую оценку их резюме, учитывая не только ключевые слова, но и контекст их применения.
7. **Снижение затрат на рекрутмент:**
   * Автоматизация этапов первичного отбора кандидатов позволит снизить расходы на работу HR-специалистов и уменьшить вероятность найма неподходящих сотрудников.
   * Ожидаемое снижение затрат на рекрутмент составит **20–30%** благодаря улучшению точности отбора и оптимизации рабочих процессов.
8. **Поддержка цифровой трансформации:**
   * Реализация системы способствует внедрению современных технологий обработки естественного языка (NLP) и машинного обучения в области управления персоналом, что соответствует трендам цифровизации бизнеса.
   * Интеграция с облачными решениями и микросервисной архитектурой делает систему гибкой и адаптируемой к меняющимся бизнес-требованиям.

#### 2.2. Бизнес-требования

В рамках автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий, система должна автоматизировать или значительно улучшить следующие бизнес-процессы:

**1. Публикация и управление вакансиями:** Система обеспечивает процесс создания и управления вакансиями, предоставляя возможность работодателям формализовать требования к кандидатам.

* **Текущий процесс (AS IS):** Работодатель размещает описание вакансии на различных платформах, что требует времени для структурирования данных и координации между платформами.
* **Автоматизированный процесс (TO BE):** Система позволяет создавать и публиковать вакансии через единый интерфейс, автоматически преобразуя текстовые данные в структурированную форму для дальнейшего анализа.

**2. Прием и фильтрация резюме:** Система автоматизирует прием резюме от соискателей и их первичную фильтрацию по соответствию требованиям вакансии.

* **Текущий процесс (AS IS):** HR-специалисты получают большое количество резюме, которые анализируют вручную, что занимает значительное время и может привести к ошибкам из-за человеческого фактора.
* **Автоматизированный процесс (TO BE):** Система принимает резюме в различных форматах (PDF, DOCX, TXT), автоматически извлекает ключевую информацию (опыт работы, навыки, образование) и проводит предварительную оценку соответствия вакансии. Это сокращает время ручной фильтрации на **50–60%** .

**3. Семантический анализ и сопоставление:** Система реализует глубокий анализ текстовых данных резюме и вакансий для точного сопоставления кандидатов с требованиями работодателя.

* **Текущий процесс (AS IS):** HR-специалисты полагаются на поиск ключевых слов или поверхностное чтение текста, что ограничивает учет контекста и может привести к недостаточно точному отбору.
* **Автоматизированный процесс (TO BE):** Использование модели BERT для семантического анализа позволяет учитывать не только наличие ключевых слов, но и их контекстное значение. Это повышает точность классификации до **75%** (accuracy) и коэффициент корреляции Мэтьюса (MCC) до **0.55** .

**4. Ранжирование кандидатов:** Система ранжирует кандидатов по уровню соответствия вакансии, предоставляя рекрутерам отсортированный список для последующего рассмотрения.

* **Текущий процесс (AS IS):** Ранжирование выполняется вручную, основываясь на интуитивных оценках или простом подсчете совпадений ключевых слов. Этот подход часто приводит к пропуску подходящих кандидатов или выбору неподходящих.
* **Автоматизированный процесс (TO BE):** Кандидаты ранжируются на основе косинусного расстояния между векторными представлениями их резюме и вакансии, что позволяет выявлять наиболее релевантные профили. Результаты представляются в виде рейтинга с вероятностью соответствия от **0 до 1** , где **1** — максимальная уверенность в соответствии.

**5. Уведомление и коммуникация:** Система автоматизирует процесс уведомления соискателей и рекрутеров о состоянии отбора.

* **Текущий процесс (AS IS):** Уведомления отправляются вручную или через ограниченные инструменты платформ, что замедляет коммуникацию и снижает удовлетворенность соискателей.
* **Автоматизированный процесс (TO BE):** Система генерирует и отправляет автоматические уведомления о статусе отбора (например, "принято" или "отклонено"), а также предоставляет рекрутерам инструменты для массовой коммуникации с кандидатами.

**6. Интеграция с внешними системами:** Система обеспечивает бесшовную интеграцию с существующими HR-платформами и базами данных для обмена данными.

* **Текущий процесс (AS IS):** Данные о кандидатах и вакансиях передаются между системами через ручной экспорт/импорт, что увеличивает вероятность ошибок и задержек.
* **Автоматизированный процесс (TO BE):** Реализация RESTful API обеспечивает автоматический обмен данными с платформами рекрутмента (например, hh.ru, LinkedIn Talent Solutions), а также с внутренними системами управления персоналом.

**7. Обновление данных вакансий и резюме:** Система автоматически обновляет данные о вакансиях и резюме для обеспечения актуальности информации.

* **Текущий процесс (AS IS):** Рекрутеры периодически проверяют актуальность данных, что требует дополнительных ресурсов.
* **Автоматизированный процесс (TO BE):** Встроенные механизмы системы позволяют обновлять заголовки и содержимое вакансий/резюме на момент их подачи, исключая использование устаревшей информации.

**8. Оптимизация использования ресурсов:** Система минимизирует временные затраты HR-специалистов на первичный отбор, позволяя им сосредоточиться на стратегических задачах.

* **Текущий процесс (AS IS):** Ручная фильтрация резюме занимает значительную часть рабочего времени рекрутеров (до **80%** ).
* **Автоматизированный процесс (TO BE):** Автоматизация сокращает время на первичный отбор до **2 секунд** на одно резюме, освобождая рекрутеров для более сложных задач, таких как интервьюирование и оценка soft skills.

**9. Стандартизация процесса отбора:** Система обеспечивает стандартизацию критериев отбора, снижая влияние человеческого фактора.

* **Текущий процесс (AS IS):** Критерии отбора могут различаться в зависимости от HR-специалиста, что приводит к неоднородным результатам.
* **Автоматизированный процесс (TO BE):** Применение единой модели анализа гарантирует объективность оценки кандидатов, основанной на данных и алгоритмах машинного обучения.

**10. Хранение и анализ исторических данных:** Система хранит данные о ранее обработанных резюме и вакансиях для анализа эффективности отбора.

* **Текущий процесс (AS IS):** Исторические данные часто теряются или плохо организованы, что затрудняет анализ успешности найма.
* **Автоматизированный процесс (TO BE):** Интеграция с PostgreSQL и Weaviate позволяет сохранять структурированные и векторные данные для последующего анализа, включая метрики точности и производительности.

#### 2.3 Ограничения

В рамках разработки программного сервиса для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий существуют следующие ограничения, влияющие на реализацию проекта:

**1. Финансовые ограничения:**

* **Бюджетные рамки** : Общий бюджет проекта ограничен и определен заказчиком на уровне **500 000 рублей** , что требует оптимизации выбора технологий и инструментов для минимизации затрат на вычислительные ресурсы, лицензии и поддержку.
* **Использование открытых решений** : В связи с бюджетными ограничениями предпочтение отдается свободно распространяемым или коммерчески доступным технологиям, таким как BERT (Hugging Face), PostgreSQL и Weaviate, исключая дорогие проприетарные системы машинного обучения или баз данных.

**2. Временные ограничения:**

* **Срок разработки** : Проект должен быть завершен в течение **4 месяцев** с момента утверждения технического задания. Это включает этапы анализа требований, проектирования, программной реализации, тестирования и внедрения.
* **Этапы реализации** :
  + Подготовка и анализ требований — **1 месяц** .
  + Разработка и тестирование прототипа — **2 месяца** .
  + Финальное тестирование и внедрение — **1 месяц** .

**3. Технические ограничения:**

* **Вычислительные мощности** : Использование моделей глубокого обучения, таких как BERT, требует наличия GPU с поддержкой CUDA для ускорения вычислений. Однако доступ к высокопроизводительным вычислительным ресурсам может быть ограничен, что повышает важность оптимизации модели и процессов обработки.
* **Объем данных** : Система должна поддерживать обработку до **172 321 записи** (пар резюме-вакансия) для обучения модели, а также работать с потоком реальных данных, достигающим **10 000 записей в час** .
* **Масштабируемость** : Необходимо обеспечить возможность масштабирования системы без значительных изменений в архитектуре, используя облачные решения (например, AWS, Yandex Cloud) или собственные серверы с Docker/Kubernetes.

**4. Нормативные требования:**

* **Защита персональных данных** : Сервис должен соответствовать требованиям **GDPR** и **ФЗ-152 "О персональных данных"** , так как будет обрабатывать конфиденциальную информацию о кандидатах и работодателях.
* **Шифрование данных** : Все данные должны храниться в зашифрованном виде с использованием алгоритма **AES-256** , а передача данных должна осуществляться через протокол **TLS 1.3** .
* **Аудиторская прозрачность** : Система должна предоставлять детальные логи действий пользователей и результатов анализа для обеспечения прозрачности процесса отбора и соответствия нормативным требованиям.

**5. Ограничения по входным данным:**

* **Форматы файлов** : Поддерживаются только текстовые форматы: **PDF, DOCX, TXT** . Другие форматы (например, HTML, JPG) не обрабатываются системой.
* **Языки** : Анализ данных возможен только на **русском** и **английском** языках, что связано с спецификой выбранной модели BERT и доступных ресурсов для ее обучения.
* **Структура данных** : Резюме и вакансии должны содержать минимальный набор полей (название, опыт работы, навыки, образование), даже если они представлены в произвольном формате.

**6. Ограничения по производительности:**

* **Время ответа API** : Максимальное время обработки одного запроса составляет **2 секунды** .
* **Объем обработки** : Система должна обрабатывать до **10 000 записей в час** , что является ключевым показателем производительности для удовлетворения потребностей бизнеса.
* **Перегрузка системы** : При превышении допустимого объема запросов система должна корректно сообщать об ошибке и не терять стабильность.

**7. Ограничения по функциональности:**

* **Оценка "мягких" навыков** : Автоматический анализ soft skills (например, лидерства, коммуникабельности) не входит в функционал системы из-за сложности их формализации и интерпретации.
* **Контекстуальная зависимость** : Система может давать неточные результаты при работе с многозначными словами или сложными контекстами, если данные недостаточно структурированы или содержат мало информации.

**8. Организационные ограничения:**

* **Обучение персонала** : Персонал HR-отделов должен пройти обучение для работы с новой системой, что может занять дополнительное время (до **2 недель** ).
* **Интеграция с существующими системами** : Сервис должен быть интегрирован с платформами рекрутмента (например, hh.ru, LinkedIn Talent Solutions), что требует учета их API-ограничений и форматов данных.

### **3. Требования пользователей**

#### **3.1 Пользовательские роли**

В рамках разрабатываемого сервиса для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий определены следующие пользовательские роли, каждая из которых имеет свои уникальные потребности и взаимодействия с системой:

**1. Рекрутер/HR-специалист**

* **Описание роли** : Специалист, ответственный за процесс подбора персонала, анализ резюме и оценку соответствия кандидатов требованиям вакансии.
* **Потребности** :
  + Автоматизация первичного фильтрации резюме для минимизации времени на ручной анализ.
  + Получение ранжированного списка кандидатов с детализированными рекомендациями по уровню соответствия.
  + Возможность создания и редактирования вакансий через удобный интерфейс.
  + Интеграция с существующими HR-системами для упрощения управления данными.
  + Уведомления о новых подходящих кандидатах.

**2. Менеджер по найму**

* **Описание роли** : Высший уровень руководителя, принимающего решения о найме и контролирующего процесс рекрутмента.
* **Потребности** :
  + Обзор результатов работы системы для оценки эффективности отбора кандидатов.
  + Доступ к метрикам производительности (например, точность классификации, время обработки).
  + Возможность корректировки параметров сопоставления для конкретных вакансий.
  + Гибкость настройки приоритетов навыков и требований для каждой позиции.

**3. Кадровое агентство**

* **Описание роли** : Организация или специалист, предоставляющий услуги по подбору персонала для разных компаний.
* **Потребности** :
  + Поддержка большого количества одновременно обрабатываемых вакансий и резюме.
  + Функционал массовой загрузки данных для анализа.
  + Возможность интеграции с платформами для поиска работы (например, hh.ru, LinkedIn Talent Solutions).
  + Инструменты для формирования отчетов о результатах отбора.

**4. Соискатель**

* **Описание роли** : Человек, который ищет работу и отправляет свое резюме на вакансии.
* **Потребности** :
  + Прозрачность процесса отбора: понимание причин принятия/отклонения резюме.
  + Ускорение получения обратной связи от работодателя.
  + Возможность просмотра рекомендаций по улучшению резюме для повышения вероятности соответствия вакансиям.

**5. Администратор системы**

* **Описание роли** : Технический специалист, обеспечивающий стабильную работу системы и управление её конфигурацией.
* **Потребности** :
  + Инструменты для мониторинга состояния системы и выявления ошибок.
  + Настройка параметров модели BERT для переподгона под специфику данных компании.
  + Управление базами данных (PostgreSQL, Weaviate) для хранения резюме, вакансий и их векторных представлений.
  + Интеграция дополнительных источников данных (например, социальные сети, истории работы).

**6. Разработчик/Научный сотрудник**

* **Описание роли** : Специалист, занимающийся улучшением алгоритмов и моделей машинного обучения, используемых в системе.
* **Потребности** :
  + Доступ к логам работы системы для анализа точности и качества предсказаний.
  + Возможность экспериментов с различными моделями embeddings (например, Word2Vec, GloVe, BERT).
  + Интерфейсы для оценки влияния новых признаков на качество сопоставления резюме и вакансий.

**7. Заказчик/Бизнес-представитель**

* **Описание роли** : Лицо, определяющее стратегические цели использования системы и оценивающее её бизнес-эффективность.
* **Потребности** :
  + Отчеты о снижении временных и финансовых затрат на рекрутмент после внедрения системы.
  + Демонстрация улучшения качества найма (например, увеличение процента успешных кандидатов).
  + Подтверждение соответствия системы нормативным требованиям (GDPR, ФЗ-152).

#### **3.2 Сценарии использования**

В данном разделе описываются типичные сценарии взаимодействия пользователей с системой автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Каждый сценарий демонстрирует конкретную задачу, которую система должна выполнять для удовлетворения потребностей различных ролей пользователей.

**1. Создание и публикация вакансии (Рекрутер/HR-специалист)**

* **Актор** : Рекрутер.
* **Цель** : Создать новую вакансию и опубликовать её для привлечения соискателей.
* **Сценарий** :
  1. Рекрутер заходит в систему через авторизацию.
  2. Выбирает функцию "Создание вакансии".
  3. Заполняет форму вакансии: название должности, описание требований, ключевые навыки, опыт работы, образование и другие параметры.
  4. Система автоматически преобразует текстовое описание в структурированные данные и векторное представление (embeddings) с использованием модели BERT.
  5. Рекрутер подтверждает создание вакансии и выбирает платформы для её публикации.
  6. Система отправляет вакансию на выбранные платформы или сохраняет её для дальнейшего использования.

**2. Приём и анализ резюме (Система)**

* **Актор** : Система.
* **Цель** : Автоматически принимать и анализировать резюме, поступающие на вакансии.
* **Сценарий** :
  1. Соискатель отправляет своё резюме на опубликованную вакансию через одну из интегрированных платформ (например, hh.ru).
  2. Система получает резюме в формате PDF, DOCX или TXT.
  3. Происходит предобработка текста: токенизация, лемматизация, удаление стоп-слов.
  4. Модель BERT создаёт векторное представление (embeddings) для резюме.
  5. Система сравнивает векторное представление резюме с векторным представлением вакансии, используя косинусное расстояние.
  6. Результат анализа записывается в базу данных: вероятность соответствия (от 0 до 1).

**3. Ранжирование кандидатов (Рекрутер/Система)**

* **Актор** : Рекрутер, Система.
* **Цель** : Предоставить рекрутеру ранжированный список кандидатов для конкретной вакансии.
* **Сценарий** :
  1. Рекрутер выбирает нужную вакансию из списка опубликованных.
  2. Система загружает все резюме, связанные с этой вакансией, и их векторные представления из Weaviate.
  3. Выполняется семантическое сопоставление между вакансией и резюме, основываясь на предварительно вычисленных embeddings.
  4. Резюме ранжируются по уровню соответствия (вероятности принятия), где значение близкое к 1 указывает на высокую релевантность.
  5. Рекрутер получает отсортированный список кандидатов с детальной информацией о совпадении ключевых навыков, опыта работы и других параметров.

**4. Отправка уведомлений (Система)**

* **Актор** : Система.
* **Цель** : Уведомить рекрутера о новых подходящих резюме или сообщить соискателю о статусе его отклика.
* **Сценарий** :
  1. После анализа нового резюме, если его вероятность соответствия вакансии выше заданного порога (например, 0.7), система отправляет уведомление рекрутеру.
  2. Если резюме не проходит фильтр, система может автоматически уведомить соискателя об отказе.
  3. Уведомления отправляются через электронную почту или интеграцию с HR-платформами.

**5. Настройка параметров модели (Менеджер по найму/Администратор)**

* **Актор** : Менеджер по найму, Администратор.
* **Цель** : Настроить модель BERT для более точного сопоставления резюме с требованиями вакансий.
* **Сценарий** :
  1. Менеджер по найму или администратор заходит в административный интерфейс системы.
  2. Выбирает функцию "Настройка модели" и задаёт весовые коэффициенты для различных параметров (например, увеличивает важность опыта работы или образования).
  3. Обновляет конфигурацию модели BERT, что запускает процесс переподгона на основе новых параметров.
  4. Система сохраняет изменения и применяет их при следующем анализе резюме.

**6. Просмотр аналитической информации (Менеджер по найму)**

* **Актор** : Менеджер по найму.
* **Цель** : Оценить эффективность работы системы и принять решение о её корректировке.
* **Сценарий** :
  1. Менеджер по найму открывает страницу аналитики в системе.
  2. Система предоставляет метрики производительности: accuracy (точность классификации), MCC (коэффициент корреляции Мэтьюса) и время обработки запросов.
  3. Менеджер может просматривать историю решений, сравнить результаты работы системы с реальными действиями HR-специалистов.
  4. На основе аналитики менеджер принимает решение о необходимости доработки системы или изменении её параметров.

**7. Корректировка вакансии (Рекрутер)**

* **Актор** : Рекрутер.
* **Цель** : Изменить описание вакансии для повышения точности сопоставления с резюме.
* **Сценарий** :
  1. Рекрутер выбирает нужную вакансию из списка.
  2. Вносит изменения в описание: добавляет или удаляет ключевые навыки, корректирует требования или уровень опыта.
  3. Система автоматически обновляет векторное представление вакансии в Weaviate.
  4. Все ранее полученные резюме повторно анализируются с учётом изменений.

**8. Проверка качества анализа (Администратор/Научный сотрудник)**

* **Актор** : Администратор, Научный сотрудник.
* **Цель** : Оценить качество работы системы и провести необходимые корректировки.
* **Сценарий** :
  1. Администратор или научный сотрудник выбирает функцию "Тестирование системы".
  2. Загружает тестовый датасет с парой "резюме-вакансия" и известными метками ("принято" или "отклонено").
  3. Система выполняет анализ и предоставляет результаты: accuracy, MCC и время обработки.
  4. Если метрики ниже установленных целевых значений (accuracy ≥ 0.75, MCC ≥ 0.55), администратор или научный сотрудник может скорректировать параметры модели или добавить новые признаки для анализа.

**9. Обработка soft skills (Ограниченная функциональность)**

* **Актор** : Система.
* **Цель** : Провести первичный анализ soft skills, если они явно указаны в резюме.
* **Сценарий** :
  1. Система ищет ключевые слова, связанные с soft skills (например, "командная работа", "лидерство", "коммуникабельность").
  2. Формирует общую оценку уровня soft skills на основе частоты и контекста их упоминания.
  3. Добавляет эту информацию как дополнительный параметр в профиль кандидата.
  4. Рекрутер может использовать этот параметр для вторичной оценки кандидатов.

#### **3.3 Качество использования**

В данном разделе описываются требования к удобству, эффективности и удовлетворенности пользователей при взаимодействии с системой автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти параметры являются ключевыми для обеспечения высокого уровня качества использования системы всеми заинтересованными сторонами.

**1. Удобство использования** Система должна быть интуитивно понятной и простой в освоении для всех типов пользователей: рекрутеров, администраторов, соискателей и менеджеров по найму. Основные требования:

* **Интерфейс пользователя** :
  + Интерфейс должен быть минималистичным и содержать только необходимые элементы управления.
  + Для рекрутеров предусмотреть функциональные кнопки (например, "Создать вакансию", "Загрузить резюме", "Просмотреть результаты").
  + Соискателям необходимо предоставить возможность загрузки своего резюме через форму веб-интерфейса или API платформ (например, hh.ru).
* **Документация** :
  + Предоставление подробного руководства по использованию системы для всех ролей пользователей.
  + Видеоинструкции и чек-листы для быстрого старта работы с сервисом.
* **Настройка под конкретные нужды** :
  + Возможность настройки интерфейса под индивидуальные предпочтения рекрутеров (например, цветовая маркировка статусов резюме).
  + Пользовательские фильтры для сортировки результатов анализа.

**2. Эффективность использования** Система должна максимально оптимизировать время и усилия пользователей при выполнении задач. Основные требования:

* **Скорость обработки данных** :
  + Обработка одной пары "резюме-вакансия" должна занимать не более **2 секунд** .
  + Система должна поддерживать одновременную обработку до **10 000 записей в час** , что соответствует пиковым нагрузкам.
* **Точность анализа** :
  + Метрика точности (accuracy) классификации резюме должна составлять **не менее 75%** .
  + Коэффициент корреляции Мэтьюса (MCC) должен быть **не ниже 0.55** , что гарантирует корректное сопоставление большинства релевантных кандидатов.
* **Автоматизация процессов** :
  + Автоматическая генерация уведомлений для рекрутеров о новых подходящих кандидатах.
  + Возможность массовой загрузки резюме и вакансий через API или файловые форматы (PDF, DOCX, TXT).
* **Работа с историческими данными** :
  + Рекрутеры должны иметь доступ к архиву прошлых решений для анализа успешности найма.
  + Соискатели могут получать обратную связь о причинах отказа или принятия их резюме.

**3. Удовлетворенность пользователей** Для повышения удовлетворенности всех участников процесса отбора кандидатов система должна соответствовать следующим требованиям:

* **Прозрачность решений** :
  + Рекрутерам предоставляется детализированная информация о причинах ранжирования каждого кандидата (например, совпадение навыков, опыт работы).
  + Соискатели могут видеть основные параметры, повлиявшие на их рейтинг для конкретной вакансии.
* **Минимизация ошибок** :
  + Система должна исключать дублирование пар "резюме-вакансия" и некорректные классификации.
  + Данные проверяются на соответствие актуальным требованиям вакансий перед формированием рейтинга.
* **Адаптивность к потребностям пользователей** :
  + Поддержка различных форматов заголовков вакансий и резюме (например, строковые значения или числовые категории).
  + Возможность изменения весовых коэффициентов для параметров (опыт работы, образование, навыки) администратором или менеджером по найму.
* **Ускорение процесса рекрутмента** :
  + Сокращение времени первичного отбора кандидатов на **50–60%** благодаря автоматизации анализа текстовых данных.
  + Уменьшение вероятности упущения подходящих кандидатов за счет учета контекста слов с помощью модели BERT.

**4. Особые условия использования**

* **Поддержка нескольких языков** :
  + Система должна работать с текстами на **русском** и **английском** языках, что расширяет её применимость для международных компаний.
* **Обратная связь** :
  + Реализация механизма сбора отзывов от пользователей для дальнейшей доработки интерфейса и алгоритмов.
  + Возможность отправки запросов на поддержку через встроенный модуль системы.
* **Безопасность данных** :
  + Все данные шифруются при хранении (AES-256) и передаче (TLS 1.3), что повышает доверие пользователей.
  + Соответствие требованиям **GDPR** и **ФЗ-152 "О персональных данных"** для защиты конфиденциальной информации.

**5. Измеримость качества использования** Для оценки качества использования системы будут применяться следующие метрики:

* **Процент успешных классификаций** : Минимум **75%** резюме должны быть правильно классифицированы согласно реальным решениям HR-специалистов.
* **Время ответа системы** : Ответ на запрос анализа резюме должен занимать **не более 2 секунд** .
* **Удовлетворенность пользователей** : Регулярные опросы пользователей для оценки удобства интерфейса и точности анализа (целевой показатель — **4/5 баллов** по шкале удовлетворенности).

### **4. Операционные требования**

#### **4.1 Операционные сценарии**

В данном разделе описываются операционные сценарии использования системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти сценарии демонстрируют, как система будет функционировать в реальных условиях, обеспечивая эффективную поддержку процессов рекрутмента для всех заинтересованных сторон.

**1. Сценарий размещения вакансии рекрутером:**

* **Актор** : Рекрутер.
* **Описание** : Рекрутер создает новую вакансию, заполняя необходимые поля: название должности, ключевые навыки, опыт работы, образование и дополнительные требования.
* **Шаги** :
  1. Рекрутер авторизуется в системе через логин и пароль.
  2. Выбирает функцию "Создание вакансии".
  3. Заполняет форму вакансии, указывая все параметры.
  4. Система преобразует текстовое описание вакансии в структурированные данные и их векторное представление с помощью модели BERT.
  5. Вакансия сохраняется в PostgreSQL (структурированные данные) и Weaviate (векторные данные).
  6. Рекрутер публикует вакансию на интегрированных платформах (например, hh.ru или LinkedIn Talent Solutions).

**2. Сценарий загрузки резюме соискателя:**

* **Актор** : Соискатель/Система.
* **Описание** : Соискатель отправляет своё резюме на опубликованную вакансию, а система автоматически его обрабатывает.
* **Шаги** :
  1. Соискатель выбирает вакансию и прикрепляет своё резюме в формате PDF, DOCX или TXT.
  2. Система получает файл резюме и проверяет его формат.
  3. Происходит предобработка текста: токенизация, удаление стоп-слов, нормализация.
  4. Модель BERT создаёт векторное представление резюме и сохраняет его в Weaviate.
  5. Структурированные данные (навыки, опыт работы, образование) извлекаются и записываются в PostgreSQL.

**3. Сценарий первичного анализа резюме:**

* **Актор** : Система.
* **Описание** : После загрузки резюме система автоматически проводит семантический анализ и сравнивает его с требуемыми характеристиками вакансии.
* **Шаги** :
  1. Система загружает векторное представление вакансии и резюме из Weaviate.
  2. Вычисляется косинусное расстояние между векторными представлениями для оценки степени соответствия.
  3. Результат анализа записывается в базу данных как вероятность принятия (значение от 0 до 1).
  4. Если вероятность выше порогового значения (например, 0.7), резюме помечается как "Предварительно принято". Иначе — "Отклонено".

**4. Сценарий ранжирования кандидатов:**

* **Актор** : Система.
* **Описание** : Система предоставляет рекрутеру ранжированный список кандидатов для конкретной вакансии.
* **Шаги** :
  1. Рекрутер выбирает нужную вакансию из списка опубликованных.
  2. Система загружает все связанные резюме и их векторные представления из Weaviate.
  3. Происходит сортировка резюме по убыванию вероятности соответствия (вычисленной ранее).
  4. Рекрутер получает список кандидатов с детальной информацией о совпадении ключевых параметров (навыки, опыт работы, образование).

**5. Сценарий отправки уведомлений:**

* **Актор** : Система.
* **Описание** : Система автоматически отправляет уведомления рекрутерам и соискателям о состоянии отбора.
* **Шаги** :
  1. При достижении определённого уровня вероятности соответствия (например, >0.7), система отправляет рекрутеру уведомление о новом подходящем кандидате.
  2. Если резюме отклонено, система может направить автоматическое уведомление соискателю с причиной отказа (например, недостаточный опыт работы).
  3. Уведомления отправляются через электронную почту или интеграцию с HR-платформами.

**6. Сценарий повторного анализа резюме:**

* **Актор** : Рекрутер/Система.
* **Описание** : Если рекрутер внес изменения в требования вакансии, система перепроверяет все ранее загруженные резюме для этой вакансии.
* **Шаги** :
  1. Рекрутер обновляет описание вакансии или её параметры.
  2. Система автоматически пересчитывает векторное представление вакансии.
  3. Все связанные резюме заново анализируются с использованием актуальных данных.
  4. Рекрутер получает обновленный список кандидатов.

**7. Сценарий мониторинга производительности:**

* **Актор** : Администратор.
* **Описание** : Администратор системы следит за её работой, контролирует метрики точности и производительности.
* **Шаги** :
  1. Администратор заходит в интерфейс мониторинга через API.
  2. Система предоставляет данные о времени обработки запросов, количестве обработанных резюме и текущих метриках (accuracy, MCC).
  3. При выявлении снижения точности или увеличения времени обработки администратор может запустить процедуру fine-tuning модели BERT.

**8. Сценарий обновления моделей BERT:**

* **Актор** : Научный сотрудник/Администратор.
* **Описание** : Для повышения точности анализа модель BERT может быть переподготовлена (fine-tuned) на новых данных.
* **Шаги** :
  1. Научный сотрудник собирает новый датасет с размеченными парами "резюме-вакансия".
  2. Запускается процесс обучения модели BERT с использованием нового датасета.
  3. После завершения обучения модель тестируется на тестовых данных.
  4. Обновленная модель заменяет старую версию, если показывает лучшие результаты (например, accuracy ≥ 0.75, MCC ≥ 0.55).

**9. Сценарий интеграции с внешними платформами:**

* **Актор** : Система.
* **Описание** : Система взаимодействует с внешними платформами для автоматической загрузки резюме и отправки результатов анализа.
* **Шаги** :
  1. Система получает доступ к API платформы рекрутмента (например, hh.ru).
  2. Автоматически загружает новые резюме, соответствующие активным вакансиям.
  3. Процесс обработки происходит аналогично описанному в сценарии №2.
  4. Результаты анализа передаются обратно на платформу для дальнейшей коммуникации с соискателями.

**10. Сценарий обработки большого потока данных:**

* **Актор** : Система.
* **Описание** : В случае пиковой нагрузки (например, массовая рассылка резюме на популярную вакансию) система должна корректно обработать все записи.
* **Шаги** :
  1. Система распределяет входящие запросы между несколькими микросервисами для параллельной обработки.
  2. Каждый микросервис выполняет векторизацию и сравнение данных для своей части запросов.
  3. Результаты объединяются и сохраняются в PostgreSQL и Weaviate.
  4. Рекрутер получает полный список отсортированных кандидатов без задержек.

#### **4.2 Режимы работы**

Система автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий должна поддерживать различные режимы работы, обеспечивающие корректную функциональность как в нормальных условиях эксплуатации, так и при возникновении аварийных ситуаций или необходимости проведения тестирования. Ниже описаны основные режимы работы системы:

**1. Нормальный режим работы** Нормальный режим является основным состоянием системы, когда она функционирует без сбоев и обеспечивает полный набор заявленных функций.

* **Описание** :  
  Система принимает новые вакансии и резюме, проводит их семантический анализ с использованием модели BERT, ранжирует кандидатов по уровню соответствия и предоставляет результаты рекрутерам.
* **Основные процессы** :
  + Загрузка текстовых данных (PDF, DOCX, TXT) для обработки.
  + Векторизация текстов резюме и вакансий с использованием трансформерной архитектуры.
  + Вычисление косинусного расстояния между векторными представлениями для оценки степени соответствия.
  + Сохранение результатов анализа в PostgreSQL (для структурированных данных) и Weaviate (для векторных данных).
  + Отправка уведомлений рекрутерам через API интеграции.
* **Требования** :
  + Обработка до **10 000 записей в час** с сохранением точности (accuracy ≥ 0.75, MCC ≥ 0.55).
  + Время ответа на запрос API — **не более 2 секунд** .
  + Непрерывная работа системы при одновременном доступе до **100 пользователей** .

**2. Тестовый режим работы** Тестовый режим используется для проверки корректности работы системы и оценки её производительности перед внедрением в рабочую среду.

* **Описание** :  
  В тестовом режиме система обрабатывает заранее подготовленные датасеты с известными метками ("принято" или "отклонено") для оценки качества предсказаний. Это позволяет выявить ошибки и определить необходимость доработки алгоритмов или переподготовки модели.
* **Основные процессы** :
  + Загрузка тестового датасета (например, 172 321 запись из реальных данных).
  + Построение векторных представлений для всех пар "резюме-вакансия".
  + Расчет метрик качества: accuracy, MCC, время обработки запросов.
  + Логирование результатов для последующего анализа.
* **Требования** :
  + Поддержка различных размеров тестовых датасетов (от малых до больших объемов).
  + Возможность выбора конкретных моделей embeddings (например, BERT, Word2Vec) для сравнения результатов.
  + Генерация отчетов о качестве работы системы после завершения тестирования.

**3. Аварийный режим работы** Аварийный режим активируется при возникновении непредвиденных ситуаций, таких как сбой сервера, временная недоступность базы данных или проблемы с интеграцией внешних систем.

* **Описание** :  
  При обнаружении аварийной ситуации система переходит в безопасное состояние, где:
  + Все текущие операции приостанавливаются или помещаются в очередь.
  + Генерируются уведомления администратору о проблеме.
  + Происходит попытка автоматического восстановления (например, повторное подключение к базе данных).
* **Основные процессы** :
  + Мониторинг состояния ключевых компонентов системы (API, PostgreSQL, Weaviate).
  + Автоматическое логирование ошибок и создание инцидентов для анализа.
  + Переключение на резервные источники данных (при наличии).
  + Уведомление администратора через каналы связи (email, SMS).
* **Требования** :
  + Время восстановления после отказа — **не более 15 минут** .
  + Минимальная потеря данных при сбое (благодаря механизмам резервного копирования).
  + Возможность временной миграции на резервные серверы для продолжения работы.

**4. Обучающий режим работы** Обучающий режим предназначен для выполнения процесса fine-tuning модели BERT на новых данных или добавления дополнительных признаков для повышения точности классификации.

* **Описание** :  
  В этом режиме система переходит в состояние обучения, где:
  + Используются новые размеченные данные для улучшения модели.
  + Происходит оценка изменений в метриках качества (accuracy, MCC).
  + После завершения обучения модель может быть заменена на новую версию, если показатели улучшаются.
* **Основные процессы** :
  + Загрузка нового датасета для обучения.
  + Разделение данных на тренировочный и валидационный наборы.
  + Запуск процесса обучения с использованием фреймворков TensorFlow/PyTorch.
  + Тестирование модели на валидационных данных и генерация отчетов о качестве.
* **Требования** :
  + Обучение должно завершаться за **разумное время** (например, не более 2 часов на 4 эпохи).
  + Поддержка механизма early stopping для прекращения обучения при переобучении.
  + Сохранение промежуточных результатов для возможности возврата к предыдущей версии модели.

**5. Конфигурационный режим работы** Конфигурационный режим применяется для настройки параметров системы администратором или научным сотрудником.

* **Описание** :  
  В конфигурационном режиме можно:
  + Изменять весовые коэффициенты для параметров сопоставления (например, увеличить важность опыта работы).
  + Настроить пороговое значение для принятия решений (например, установить минимальную вероятность соответствия равной 0.6).
  + Обновить конфигурацию интеграции с внешними платформами (hh.ru, LinkedIn Talent Solutions).
* **Основные процессы** :
  + Чтение и запись конфигурационных файлов (.yaml, .json).
  + Применение изменений без перезапуска всей системы (горячая замена параметров).
  + Проверка корректности настроек через тестовые запросы.
* **Требования** :
  + Интерфейс для управления конфигурацией должен быть интуитивно понятным.
  + Любые изменения должны проходить валидацию перед применением.
  + Система должна сохранять возможность возврата к предыдущим конфигурациям.

**6. Режим технического обслуживания** Режим технического обслуживания используется для проведения регларных обновлений, исправления ошибок или масштабирования системы.

* **Описание** :  
  В режиме технического обслуживания:
  + Приостанавливается прием новых данных (резюме и вакансий).
  + Выполняются операции по обслуживанию (обновление зависимостей, очистка БД).
  + Проводится проверка работоспособности после завершения обслуживания.
* **Основные процессы** :
  + Резервное копирование данных из PostgreSQL и Weaviate.
  + Обновление контейнеров Docker/Kubernetes.
  + Проверка целостности данных после завершения обслуживания.
* **Требования** :
  + Минимизация времени простоя системы во время обслуживания (до **30 минут** ).
  + Обеспечение безопасности данных во время обновлений (шифрование резервных копий).
  + Автоматическая отправка уведомлений пользователям о переходе в режим технического обслуживания.

### **5. Ограничения проекта**

#### **5.1 Бюджет и сроки**

В рамках разработки программного сервиса для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий, установлены следующие ограничения по бюджету и временным рамкам:

**1. Бюджетные ограничения:** Разработка сервиса подчиняется жестким финансовым ограничениям, определенным заказчиком:

* **Общий бюджет** : Проект должен быть выполнен в рамках выделенного бюджета, составляющего **не более 500 000 рублей** .
* **Ограничения по использованию проприетарных решений** : В связи с ограниченным бюджетом предпочтение отдается бесплатным или коммерчески доступным технологиям, таким как BERT (Hugging Face), PostgreSQL, Weaviate, Docker и Kubernetes. Проприетарные решения могут быть рассмотрены только при наличии оправданных экономических преимуществ.
* **Капитальные затраты на инфраструктуру** : Выделенные средства не покрывают приобретение дорогостоящего аппаратного обеспечения. Необходимо использовать облачные решения (например, AWS, Yandex Cloud) или существующую инфраструктуру заказчика для развертывания сервиса.

**2. Временные ограничения:** Проект имеет четкие временные рамки для завершения всех этапов разработки:

* **Общий срок разработки** : Разработка должна быть завершена в течение **4 месяцев** с момента утверждения технического задания.
* **Этапы реализации** :
  + **Подготовительный этап (1 месяц)** : Включает анализ требований, выбор технологий и подготовку документации.
  + **Программная реализация (2 месяца)** : Разработка функциональных модулей, обучение модели BERT, интеграция с базами данных и тестирование.
  + **Тестирование и внедрение (1 месяц)** : Проведение нагрузочного и функционального тестирования, исправление ошибок, обучение пользователей и запуск системы в эксплуатацию.
* **Пиковые временные ограничения** :
  + Обучение модели BERT должно быть завершено за **2 эпохи** , что соответствует примерно **37 минутам на одну эпоху** при использовании GPU (согласно результатам экспериментов, описанным в разделе 4.4).
  + Система должна обрабатывать до **10 000 записей в час** , что требует оптимизации архитектуры и параллельной обработки запросов (раздел 3.4).

**3. Управление рисками связанными с бюджетом и сроками:**

* **Бюджетные риски** :
  + Если возникают дополнительные затраты на инфраструктуру или лицензии, необходимо согласовать их с заказчиком заранее.
  + Использование открытых библиотек и фреймворков (TensorFlow, PyTorch, spaCy) минимизирует расходы на разработку алгоритмической части.
* **Риски по срокам** :
  + Задержки в обучении модели или сборке датасета могут повлиять на график работ. Для минимизации рисков используется уже предобученная модель BERT, которая адаптируется под задачу fine-tuning.
  + Тестирование и исправление ошибок должны быть спланированы так, чтобы не выйти за пределы одного месяца, отведенного на этот этап.

**4. Обоснование ограничений:**

* **Бюджетные ограничения** основаны на доступных финансовых ресурсах заказчика и необходимости использования экономически эффективных решений. Например, использование PostgreSQL и Weaviate вместо коммерческих СУБД позволяет снизить стоимость владения системой (раздел 3.3).
* **Временные ограничения** определяются необходимостью быстрого вывода продукта на рынок для удовлетворения потребностей бизнеса. Инкрементный подход к разработке (раздел 4.1) помогает соблюдать сроки путем поэтапной реализации функционала.

**5. План управления бюджетом и сроками:**

* **Месячное планирование** : Каждый месяц работы расписан с учетом конкретных целей и задач, что позволяет контролировать выполнение проекта.
* **Контрольные точки** : Ежемесячное тестирование достигнутых результатов для своевременной корректировки направления развития.
* **Отчетность** : Регулярные отчеты о прогрессе перед заказчиком для предотвращения отклонений от графика и бюджета.

**6. Финальные требования:**

* Система должна быть готова к внедрению в течение **4 месяцев** после начала разработки.
* Все операции обучения и тестирования должны быть завершены в рамках выделенных временных и финансовых ресурсов, без увеличения бюджета сверх **500 000 рублей** .

#### **5.2 Технические ограничения**

В рамках разработки программного сервиса для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий существуют следующие технические ограничения, связанные с использованием технологий и инфраструктуры:

**1. Ограничения по вычислительным ресурсам:** Модель BERT, выбранная для анализа текстовых данных, требует значительных вычислительных мощностей для обучения и выполнения задач в режиме реального времени.

* **GPU-ускорение** : Для эффективной работы модели необходим доступ к графическим процессорам с поддержкой CUDA (например, NVIDIA Tesla V100 или аналоги). Без GPU время обработки может значительно увеличиться, что негативно скажется на производительности системы.
* **Оперативная память** : Минимальные требования к оперативной памяти составляют **16 ГБ RAM** , рекомендуется **32 ГБ RAM** для работы с большими объемами данных.

**2. Ограничения по объему обрабатываемых данных:** Система должна быть способна обрабатывать до **10 000 записей в час** (пары "резюме-вакансия"), что определяет необходимость оптимизации алгоритмов и архитектурных решений.

* **Тестовые данные** : При тестировании система работала с датасетом объемом **172 321 запись** , что является максимальным пределом для текущей реализации без дополнительной оптимизации.
* **Размер файлов** : Поддерживаются только текстовые форматы резюме и вакансий: **PDF, DOCX, TXT** . Файлы других форматов (например, HTML, JPG) не обрабатываются системой.

**3. Ограничения по используемым технологиям:** Выбор технологий ограничен доступными средствами и спецификой задачи.

* **BERT** : Использование трансформерной модели BERT для семантического анализа текста требует значительных временных затрат на обучение. Например, fine-tuning модели на тестовых данных занял **около 2 часов на 4 эпохи** .
* **PostgreSQL** : Реляционная СУБД используется для хранения структурированных данных о резюме и вакансиях. Однако PostgreSQL имеет ограничения при работе с очень большими объемами данных, что может потребовать перехода на более масштабируемые решения (например, CockroachDB) при дальнейшем развитии системы.
* **Weaviate** : Векторная СУБД применяется для хранения векторных представлений текстов. Мыaviate обеспечивает быстрый поиск ближайших соседей, но его использование требует наличия индексации HNSW, которая может замедляться при увеличении размера базы данных.

**4. Интеграционные ограничения:** Интеграция с внешними платформами (например, hh.ru, LinkedIn Talent Solutions) возможна только через их API, что накладывает ограничения на форматы данных и частоту запросов.

* **API-ограничения** : Многие платформы имеют лимиты на количество запросов в секунду (QPS), например, hh.ru ограничивает число запросов до **50 в минуту** для бесплатных учетных записей. Это требует реализации очередей запросов и механизма повторных попыток.
* **Стандартизация данных** : Названия параметров (опыт работы, навыки, образование) могут различаться между платформами, что усложняет их унификацию перед анализом.

**5. Ограничения по безопасности:** Система должна соответствовать стандартам защиты персональных данных, что накладывает дополнительные технические требования.

* **Шифрование данных** : Все данные должны храниться в зашифрованном виде с использованием AES-256, а передача данных осуществляться через TLS 1.3. Это увеличивает нагрузку на серверную инфраструктуру.
* **GDPR/FZ-152** : Необходимость соблюдения нормативных требований ограничивает возможности обработки данных, таких как сохранение истории изменений резюме или вакансий.

**6. Ограничения по производительности:** Производительность системы зависит от нескольких факторов, включая скорость обработки запросов и время ответа API.

* **Время ответа API** : Обработка одного запроса (сопоставление резюме и вакансии) должна занимать **не более 2 секунд** . Это требует оптимизации взаимодействия между микросервисами и базами данных.
* **Параллельная обработка** : Для достижения целевого показателя обработки **10 000 записей в час** необходимо использовать многопоточную архитектуру или контейнеризацию через Docker/Kubernetes.

**7. Ограничения по языковой специфике:** Система поддерживает обработку текстов на **русском** и **английском** языках. Другие языки не поддерживаются из-за специфики предобученной модели BERT.

**8. Ограничения по типам данных:**

* **Форматы входных данных** : Резюме и вакансии принимаются только в форматах PDF, DOCX и TXT. Данные в других форматах (например, HTML, XML) требуют предварительной конвертации.
* **Структура данных** : Не все поля резюме и вакансий являются обязательными. Например, наличие поля "пол" или "возраст" зависит от конкретного случая использования.

**9. Ограничения по моделям машинного обучения:**

* **Fine-tuning BERT** : Переподготовка модели BERT под новые данные возможна только при наличии выделенных вычислительных ресурсов и времени (примерно **2 часа на 4 эпохи** ).
* **Объем векторных данных** : Размер векторных представлений (embeddings) составляет **768 элементов** , что требует оптимизации хранения и поиска в Weaviate.

**10. Ограничения по инфраструктуре:**

* **Локальное развертывание** : Если система разворачивается на локальных серверах, требуется минимальная конфигурация: процессор с поддержкой многопоточной обработки, GPU для ускорения BERT, SSD-хранилище для PostgreSQL и Weaviate.
* **Облачная инфраструктура** : Развертывание в облаке (например, AWS, Yandex Cloud) требует использования специализированных инстансов с поддержкой GPU, что может увеличить стоимость владения системой.

# SyRS (System Requirements Specification)

### **1. Введение**

#### **1.1 Цель системы**

Целью разрабатываемой системы является автоматизация процесса первичного отбора кандидатов на основе обработки естественного языка из текстовых данных резюме и вакансий с использованием методов машинного обучения и семантического анализа. Система должна обеспечивать следующие технические задачи:

1. **Анализ текстовой информации** :
   * Использование модели BERT для преобразования текстов резюме и вакансий в контекстуальные векторные представления (embeddings).
   * Выявление ключевых параметров, таких как опыт работы, навыки, образование, и их сравнение с требованиями вакансии.
2. **Семантическое сопоставление** :
   * Вычисление косинусного расстояния между векторными представлениями резюме и вакансий для оценки уровня соответствия.
   * Формирование рейтинга кандидатов на основе вероятности принятия (значение от 0 до 1), где 1 указывает на максимальное соответствие.
3. **Высокая производительность** :
   * Обработка до **10 000 записей в час** при сохранении точности классификации (accuracy ≥ 0.75) и коэффициента корреляции Мэтьюса (MCC ≥ 0.55).
   * Работа в условиях высоких нагрузок с поддержкой параллельной обработки запросов через микросервисную архитектуру.
4. **Надежное хранение данных** :
   * Интеграция PostgreSQL для хранения структурированных данных о резюме и вакансиях.
   * Использование Weaviate для управления векторными представлениями текстов, что обеспечивает быстрый поиск по векторному пространству.
5. **Автоматизированное взаимодействие с внешними системами** :
   * Реализация RESTful API для интеграции с HR-платформами (например, hh.ru, LinkedIn Talent Solutions) и внутренними базами данных рекрутеров.
   * Поддержка форматов данных PDF, DOCX, TXT для загрузки резюме и вакансий.
6. **Обеспечение безопасности данных** :
   * Шифрование данных при хранении (AES-256) и передаче (TLS 1.3) для защиты конфиденциальной информации соискателей и работодателей.
   * Соответствие нормативным требованиям GDPR и ФЗ-152 "О персональных данных".
7. **Масштабируемость и отказоустойчивость** :
   * Развертывание системы в контейнерах Docker с оркестрацией Kubernetes для обеспечения горизонтального масштабирования.
   * Реализация механизма резервного копирования и восстановления данных для минимизации времени простоя после сбоев (MTTR ≤ 15 минут).

Техническая цель системы заключается в создании надежного, эффективного и безопасного решения для автоматического анализа текстовых данных, что позволит существенно улучшить качество и скорость отбора кандидатов при минимальном участии человека.

#### **1.2 Область применения**

Система автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий предназначена для использования в процессе рекрутмента организаций, специализирующихся на подборе персонала, а также для внутренних отделов кадров компаний. Определение границ системы и её взаимодействия с внешними системами представлено ниже.

**1. Границы системы:** Система реализует автоматизированный анализ текстовых данных резюме и вакансий, их семантическое сопоставление и ранжирование кандидатов по уровню соответствия требованиям работодателей. Границы системы охватывают следующие процессы:

1. **Ввод данных** :
   * Принятие резюме и описаний вакансий в форматах PDF, DOCX, TXT.
   * Получение структурированных данных (например, опыт работы, навыки, образование) из неструктурированных текстов.
2. **Обработка данных** :
   * Предобработка текстов: токенизация, лемматизация, удаление стоп-слов.
   * Векторизация текстов с использованием модели BERT для создания контекстуальных представлений.
   * Семантический анализ текстов для выявления соответствия между резюме и вакансиями.
3. **Выходные данные** :
   * Ранжированный список кандидатов с вероятностью соответствия вакансии (значение от 0 до 1).
   * Отчеты о результате анализа в формате CSV или PDF для дальнейшего рассмотрения HR-специалистами.

**2. Взаимодействие с внешними системами:** Система интегрируется с внешними платформами через RESTful API для обмена данными. Основные точки взаимодействия включают:

1. **HR-платформы** :
   * Интеграция с популярными платформами поиска работы, такими как hh.ru, LinkedIn Talent Solutions, и HeadHunter API.
   * Автоматическая загрузка резюме и вакансий для анализа.
2. **Базы данных** :
   * Использование PostgreSQL для хранения структурированных данных о резюме и вакансиях.
   * Внедрение Weaviate для управления векторными представлениями текстов (embeddings), что обеспечивает быстрый поиск ближайших соседей в семантическом пространстве.
3. **Интерфейсы взаимодействия** :
   * Поддержка протоколов HTTPS/TLS для безопасной передачи данных.
   * Возможность отправки уведомлений рекрутерам через email или интеграцию с корпоративными мессенджерами.

**3. Технические ограничения взаимодействия:**

1. **Форматы данных** :
   * Входные данные принимаются только в формах PDF, DOCX, TXT. Другие форматы (например, HTML, XML) требуют предварительной конвертации.
   * Вакансии и резюме должны содержать минимальный набор полей: название должности, ключевые навыки, опыт работы и образование.
2. **Языковая поддержка** :
   * Анализ данных осуществляется на **русском** и **английском** языках. Другие языки не поддерживаются в рамках текущей реализации.
3. **Масштабируемость** :
   * Система способна обрабатывать до **10 000 записей в час** , при этом сохраняя точность классификации (accuracy ≥ 0.75) и коэффициент корреляции Мэтьюса (MCC ≥ 0.55).
   * Для масштабирования используются микросервисная архитектура и контейнеризация через Docker/Kubernetes.
4. **Ограничения по интеграции** :
   * Внешние платформы должны предоставлять API с возможностью получения информации о резюме и вакансиях.
   * Необходимо соблюдение правил доступа и безопасности данных согласно GDPR и ФЗ-152 "О персональных данных".

**4. Контекст использования:** Система применяется в следующих сценариях:

* **Первичный отбор кандидатов** : Ускоряет фильтрацию большого количества резюме, минимизируя участие HR-специалистов на начальных этапах.
* **Ранжирование кандидатов** : Выполняет сравнение резюме с требованиями вакансий на основе семантического анализа, что повышает объективность выбора.
* **Поддержка бизнес-процессов** : Интегрируется в существующие системы рекрутмента, автоматизируя этапы AS IS и переходя к более эффективному TO BE.

**5. Обоснование границ и взаимодействий:** Границы системы и её взаимодействие с внешними компонентами основаны на технических требованиях к производительности, точности анализа и безопасности данных. Например, использование BERT для семантического анализа (раздел 3.4) позволяет достичь accuracy = 0.76 и MCC = 0.55 на тестовом датасете объемом **172 321 запись** . Интеграция с PostgreSQL и Weaviate обеспечивает надежное хранение как структурированных, так и векторных данных, необходимых для эффективного функционирования системы (раздел 3.3).

Этот раздел согласуется с выбранными технологическими решениями (BERT, PostgreSQL, Weaviate) и математической моделью отбора кандидатов (раздел 2.1), что делает работу системы прозрачной и ориентированной на удовлетворение потребностей заинтересованных сторон.

#### **1.3 Определения**

В данном разделе приводятся термины и сокращения, используемые в документе для описания технических требований к системе автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий.

**Термины:**

1. **AS IS** — Текущее состояние процесса рекрутмента, характеризующееся ручной фильтрацией резюме и ограниченной автоматизацией.
2. **TO BE** — Целевое состояние процесса рекрутмента после внедрения автоматизированного сервиса, обеспечивающее более высокую эффективность и точность отбора кандидатов.
3. **BERT (Bidirectional Encoder Representations from Transformers)** — Нейронная модель для обработки естественного языка, основанная на трансформерах и позволяющая создавать контекстуальные векторные представления текстов.
4. **ContinueWork** — Наименование разрабатываемого программного сервиса для автоматизации отбора кандидатов на основе NLP.
5. **Docker** — Инструмент для контейнеризации приложений, обеспечивающий изоляцию и переносимость компонентов системы.
6. **Embeddings** — Векторные представления текстовых данных, полученные с использованием моделей обработки естественного языка (например, BERT).
7. **Fine-tuning** — Процесс адаптации предобученной модели (например, BERT) под конкретную задачу путем дополнительного обучения на специализированных данных.
8. **GPU (Graphics Processing Unit)** — Графический процессор, используемый для ускорения вычислений в моделях глубокого обучения.
9. **HNSW (Hierarchical Navigable Small World)** — Алгоритм поиска ближайших соседей в векторных пространствах, применяемый в Weaviate для быстрого сопоставления резюме и вакансий.
10. **MCC (Matthews Correlation Coefficient)** — Метрика качества классификации, учитывающая истинные положительные, истинные отрицательные, ложные положительные и ложные отрицательные значения; принимает значения от -1 до 1.
11. **NLP (Natural Language Processing)** — Область искусственного интеллекта, занимающаяся анализом и обработкой естественного языка.
12. **RESTful API** — Интерфейс взаимодействия между компонентами системы, основанный на архитектурном стиле REST (Representational State Transfer).
13. **Self-Attention** — Механизм внимания в трансформерах, который позволяет модели анализировать текст двунаправленно, учитывая контекст слов.
14. **Vector Database** — База данных для хранения векторных представлений текстов, таких как embeddings, с поддержкой быстрого поиска семантической близости.
15. **Weaviate** — Специализированная векторная база данных, используемая для хранения и анализа embeddings резюме и вакансий.
16. **PostgreSQL** — Реляционная система управления базами данных с поддержкой JSONB и ACID-свойств, применяемая для хранения структурированных данных.
17. **Resume (Резюме)** — Документ, содержащий информацию о соискателе, включая навыки, опыт работы и образование.
18. **Vacancy (Вакансия)** — Описание открытой позиции работодателя с указанием требований и условий труда.
19. **Cosine Similarity (Косинусное расстояние)** — Метрика для измерения сходства между векторными представлениями текстов.
20. **JSON (JavaScript Object Notation)** — Легковесный формат обмена данными, используемый для передачи структурированной информации между компонентами системы.
21. **ACID** — Набор свойств транзакций в базах данных: атомарность, согласованность, изолированность, надежность.
22. **Microservices Architecture (Микросервисная архитектура)** — Архитектурный подход, при котором программа разбивается на независимые сервисы, каждый из которых выполняет свою функциональную роль.

**Сокращения:**

1. **API** — Application Programming Interface (интерфейс программирования приложений).
2. **AS** — Current State (текущее состояние).
3. **TO** — Target State (целевое состояние).
4. **NLP** — Natural Language Processing (обработка естественного языка).
5. **GPU** — Graphics Processing Unit (графический процессор).
6. **SQL** — Structured Query Language (язык запросов для работы с реляционными базами данных).
7. **MVP** — Minimum Viable Product (минимально жизнеспособный продукт).
8. **SaaS** — Software as a Service (программное обеспечение как сервис).
9. **IDE** — Integrated Development Environment (интегрированная среда разработки).
10. **JSON** — JavaScript Object Notation (формат обмена данными).
11. **ACID** — Atomicity, Consistency, Isolation, Durability (свойства транзакций).
12. **ML** — Machine Learning (машинное обучение).
13. **HR-tech** — Технологии для управления человеческими ресурсами.

### **2. Функциональные требования**

#### **2.1 Функции системы**

В данном разделе описываются основные функции, которые должна выполнять система автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Каждая функция определена с точки зрения её технической реализации и роли в достижении целей проекта.

**1. Векторизация текстовых данных:** Система должна преобразовывать текстовые данные резюме и вакансий в контекстуальные векторные представления с использованием модели BERT.

* **Вход** : Тексты резюме и вакансий в форматах PDF, DOCX, TXT.
* **Выход** : Векторные представления (embeddings) размерностью 768 для каждого документа.
* **Цель** : Обеспечить семантическую интерпретацию текстов для их дальнейшего анализа и сравнения.

**2. Извлечение ключевых параметров:** Система должна автоматически извлекать ключевые параметры из текстов резюме и вакансий, такие как:

* Опыт работы.
* Навыки (жесткие и мягкие).
* Образование.
* Дополнительные характеристики (например, сертификаты или профессиональные достижения).
* **Вход** : Векторные представления текстов.
* **Выход** : Структурированные данные в формате JSON для хранения в PostgreSQL.
* **Цель** : Преобразовать неструктурированные тексты в структурированные данные для удобства анализа и фильтрации.

**3. Семантическое сопоставление резюме и вакансий:** Система должна сравнивать векторные представления резюме и вакансий, используя косинусное расстояние, чтобы определить степень соответствия.

* **Вход** : Векторные представления резюме и вакансий из Weaviate.
* **Выход** : Числовое значение [0, 1], где 0 — полное несоответствие, а 1 — максимальное соответствие.
* **Цель** : Ранжировать кандидатов по уровню соответствия требованиям вакансии.

**4. Создание и управление вакансиями:** Система должна предоставлять интерфейс для создания и редактирования вакансий, включая описание требований и навыков.

* **Вход** : Параметры вакансии (название, опыт работы, необходимые навыки, образование).
* **Выход** : Сохранение данных о вакансии в PostgreSQL и её векторного представления в Weaviate.
* **Цель** : Обеспечить возможность рекрутерам добавлять новые вакансии и обновлять существующие.

**5. Прием и обработка резюме:** Система должна принимать резюме от соискателей в поддерживаемых форматах и проводить их предварительную обработку (токенизация, лемматизация, удаление стоп-слов).

* **Вход** : Резюме в форматах PDF, DOCX, TXT.
* **Выход** : Предобработанный текст для дальнейшей векторизации и сохранение метаданных в PostgreSQL.
* **Цель** : Подготовить резюме для анализа и устранить влияние различий в форматировании.

**6. Ранжирование кандидатов:** Система должна формировать список кандидатов, отсортированных по уровню соответствия вакансии.

* **Вход** : Результаты семантического сопоставления для всех пар "резюме-вакансия".
* **Выход** : Рейтинг кандидатов с вероятностями принятия (например, 0.8 для высокой вероятности).
* **Цель** : Предоставить рекрутерам готовый список наиболее подходящих кандидатов для каждой вакансии.

**7. Интеграция с внешними платформами:** Система должна поддерживать интеграцию с HR-платформами через RESTful API для автоматического получения резюме и отправки результатов анализа.

* **Вход** : Запросы от платформ (например, hh.ru, LinkedIn Talent Solutions).
* **Выход** : Ответы в формате JSON с данными о ранжировании кандидатов.
* **Цель** : Упростить взаимодействие с существующими системами рекрутмента.

**8. Генерация отчетов:** Система должна генерировать отчеты о результате анализа резюме для конкретной вакансии в форматах CSV и PDF.

* **Вход** : Результаты ранжирования кандидатов.
* **Выход** : Отчеты с детализацией причин принятия/отклонения резюме.
* **Цель** : Предоставить рекрутерам прозрачность процесса отбора и возможность использования данных для дальнейшего анализа.

**9. Уведомления пользователей:** Система должна отправлять автоматические уведомления рекрутерам и соискателям о состоянии обработки резюме.

* **Вход** : Результаты анализа конкретного резюме.
* **Выход** : Email-уведомления или интеграция с корпоративными мессенджерами.
* **Цель** : Обеспечить своевременную обратную связь всем участникам процесса.

**10. Обновление данных вакансий:** Система должна поддерживать возможность обновления информации о вакансии, после чего пересчитывать соответствие всех ранее загруженных резюме.

* **Вход** : Измененные параметры вакансии.
* **Выход** : Обновленные векторные представления вакансии и новый рейтинг кандидатов.
* **Цель** : Гарантировать актуальность результатов при изменении требований работодателя.

**11. Fine-tuning модели BERT:** Система должна позволять администраторам проводить fine-tuning модели BERT на новых данных для повышения точности классификации.

* **Вход** : Размеченный датасет с парами "резюме-вакансия" и метками соответствия.
* **Выход** : Обновленная модель BERT с новыми весами.
* **Цель** : Адаптировать модель под специфику компании или рынка труда.

**12. Поиск ближайших соседей:** Система должна использовать алгоритмы Approximate Nearest Neighbors (например, HNSW) для быстрого поиска наиболее релевантных резюме в векторном пространстве.

* **Вход** : Векторное представление вакансии.
* **Выход** : Список ближайших соседей (резюме) с расстояниями.
* **Цель** : Ускорить процесс поиска подходящих кандидатов при работе с большими объемами данных.

**13. Мониторинг качества анализа:** Система должна предоставлять метрики качества работы алгоритмов (accuracy, MCC) для оценки эффективности сопоставления резюме и вакансий.

* **Вход** : Результаты анализа и реальные метки соответствия.
* **Выход** : Отчеты с метриками производительности (например, accuracy = 0.76, MCC = 0.55).
* **Цель** : Обеспечить контроль над качеством работы системы и возможность её доработки.

#### **2.2 Входы и выходы**

В данном разделе описываются данные, которые система должна обрабатывать на входе, а также результаты её работы, представленные в виде выходных данных.

**1. Входные данные:**

1.1. **Текстовые файлы резюме и вакансий:**

* **Форматы:** PDF, DOCX, TXT.
* **Описание:** Резюме и вакансии предоставляются в текстовом формате с различными параметрами: название должности, опыт работы, ключевые навыки, образование, дополнительные характеристики (например, сертификаты).
* **Пример содержимого:**
  + Резюме: «Инженер-программист с опытом работы 5 лет, владение Python, SQL, машинным обучением».
  + Вакансия: «Требуется инженер-программист с опытом работы от 3 лет, знание Python, SQL».

1.2. **Структурированные метаданные:**

* **Формат:** JSON или CSV.
* **Описание:** Дополнительные структурированные данные о резюме и вакансиях, такие как:
  + Идентификаторы резюме и вакансий.
  + Категории вакансий (IT, Маркетинг, Производство).
  + Дата публикации вакансии.
* **Пример содержимого:** json  
  {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "category": "IT",  
   "event\_date": "2024-01-01"  
   }

1.3. **Запросы API:**

* **Формат:** HTTP/HTTPS через RESTful API.
* **Описание:** Запросы для загрузки новых резюме, создания/редактирования вакансий, получения результатов анализа.
* **Пример запроса:** http
* POST /api/vacancy  
   Content-Type: application/json  
   {  
   "title": "Инженер-программист",  
   "skills": ["Python", "SQL"],  
   "experience": "3+ years"  
   }

1.4. **Данные из внешних систем:**

* **Источники:** Платформы рекрутмента (hh.ru, LinkedIn Talent Solutions).
* **Описание:** Информация о резюме и вакансиях, получаемая через API платформ.
* **Пример содержимого:** json  
    
  {  
   "candidate\_name": "Иванов Иван",  
   "candidate\_skills": ["Python", "Django", "PostgreSQL"],  
   "job\_title": "Разработчик Python"  
   }

**2. Выходные данные:**

2.1. **Результаты анализа соответствия:**

* **Формат:** JSON или CSV.
* **Описание:** Степень соответствия каждого резюме требованиям вакансии, выраженная числовым значением [0, 1], где 1 — максимальное соответствие.
* **Пример содержимого:** json  
  {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "similarity\_score": 0.85,  
   "reasons": ["Python experience matches", "SQL knowledge is required"]  
   }

2.2. **Уведомления пользователей:**

* **Формат:** Email, SMS или push-уведомления через API.
* **Описание:** Автоматические уведомления рекрутерам о подходящих кандидатах или соискателям об отказе/приглашении.
* **Пример содержимого:**
  + Для рекрутера: «Найдено 10 подходящих кандидатов для вакансии "Инженер-программист".»
  + Для соискателя: «Ваше резюме рассмотрено для вакансии "Инженер-программист". Вероятность соответствия: 85%.»

2.3. **Отчеты о работе системы:**

* **Формат:** PDF или CSV.
* **Описание:** Подробные отчеты о процессе анализа, включая метрики качества (accuracy, MCC) и детализацию причин принятия/отклонения резюме.

2.4. **Векторные представления текстов:**

* **Формат:** NumPy массивы размерностью 768 (для BERT).
* **Описание:** Векторные представления текстов резюме и вакансий для хранения в Weaviate.
* **Пример содержимого:** python
* array([0.123, 0.456, ..., 0.789]) # Всего 768 элементов

**3. Требования к входам и выходам:**

3.1. **Обработка нестандартизированных текстов:**

* Система должна автоматически нормализовать текстовые данные (токенизация, лемматизация, удаление стоп-слов).

3.2. **Поддержка нескольких языков:**

* Обработка данных на **русском** и **английском** языках.

3.3. **Шифрование данных:**

* Все данные на входе и выходе должны шифроваться при передаче (TLS 1.3) и хранении (AES-256).

3.4. **Контроль целостности данных:**

* Входные данные проверяются на наличие ошибок формата, пропусков или некорректных полей.

**4. Обоснование входов и выходов:** Выбор входных и выходных данных основан на анализе требований к автоматизации отбора кандидатов (раздел 1.6), а также результатах тестирования прототипа (раздел 4.3). Например, использование модели BERT для векторизации текстов позволяет достичь accuracy = 0.76 и MCC = 0.55 на тестовых данных объемом **172 321 запись** , что подтверждает эффективность выбранного подхода.

Выходные данные в виде ранжированного списка кандидатов и детализированных отчетов обеспечивают рекрутерам прозрачность процесса отбора и возможность быстрого принятия решений. Шифрование и контроль целостности данных гарантируют соответствие требованиям безопасности (GDPR, ФЗ-152).

### **3. Нефункциональные требования**

#### **3.1 Производительность**

В данном разделе описываются технические требования к производительности системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти требования определяют скорость, время отклика и пропускную способность системы для обеспечения её эффективной работы в условиях высоких нагрузок.

**1. Скорость обработки данных:** Система должна быть способна обрабатывать большие объемы текстовых данных за минимальное время.

* **Требования:**
  + Обработка до **10 000 записей в час** (пар резюме-вакансия).
  + Время обработки одной записи должно составлять **не более 2 секунд** .
* **Обоснование:** Высокая скорость обработки необходима для поддержания эффективного рабочего процесса рекрутеров, которые могут получать большое количество резюме одновременно. Результаты тестирования показали, что модель BERT с предварительно обученными весами позволяет достичь этих показателей на тестовом датасете объемом **172 321 запись** (раздел 4.3).

**2. Время отклика API:** Система должна обеспечивать быстрый отклик при взаимодействии через RESTful API для интеграции с внешними платформами и внутренними системами управления персоналом.

* **Требования:**
  + Время ответа на запрос API: **не более 2 секунд** .
  + При работе с пиковой нагрузкой (до **100 одновременных пользователей** ) время отклика не должно увеличиваться более чем на **50%** .
* **Обоснование:** Низкое время отклика критично для обеспечения комфортного взаимодействия пользователей с системой. Микросервисная архитектура, выбранная для реализации проекта (раздел 3.2), позволяет распределить нагрузку и минимизировать задержки при обработке запросов.

**3. Пропускная способность системы:** Система должна поддерживать высокую пропускную способность для обработки множества пар "резюме-вакансия" одновременно.

* **Требования:**
  + Пропускная способность системы: **до 10 000 записей в час** .
  + Поддержка одновременной загрузки до **100 резюме** для одной вакансии.
  + Возможность масштабирования для обработки большего количества данных без потери производительности.
* **Обоснование:** Пропускная способность определяется результатами нагрузочного тестирования (раздел 4.3), где система успешно справилась с обработкой нескольких тысяч записей. Использование контейнеризации через Docker и Kubernetes (раздел 4.2) гарантирует возможность горизонтального масштабирования для удовлетворения потребностей растущего числа пользователей.

**4. Требования к времени выполнения конкретных операций:** Каждая операция в системе должна соответствовать установленным временны́м рамкам для обеспечения стабильной работы.

* **Векторизация текста:**
  + Время преобразования текста в векторное представление с использованием модели BERT: **не более 1 секунды на документ** .
* **Семантическое сопоставление:**
  + Время вычисления косинусного расстояния между векторными представлениями: **не более 0.5 секунды на пару "резюме-вакансия"** .
* **Формирование рейтинга кандидатов:**
  + Время создания ранжированного списка кандидатов для конкретной вакансии: **не более 5 секунд** для 100 резюме.

**5. Тестовые характеристики производительности:** На основе экспериментов с тестовым датасетом были установлены следующие показатели:

* **Accuracy (точность):** Должна составлять **не менее 0.76** .
* **MCC (коэффициент корреляции Мэтьюса):** Должен быть **не ниже 0.55** .
* **Время обучения модели BERT:** Не более **37 минут на эпоху** при использовании GPU.

**6. Условия работы при пиковых нагрузках:** Система должна сохранять свою работоспособность и устойчивость при пиковых нагрузках.

* **Требования:**
  + При одновременной обработке **10 000 записей** время выполнения операций не должно увеличиваться более чем на **20%** по сравнению с нормальными условиями.
  + В случае временного превышения нагрузки система должна помещать дополнительные запросы в очередь для последующей обработки.
* **Обоснование:** Пиковые нагрузки могут возникать во время массовых рассылок резюме на популярные вакансии. Архитектурные решения, такие как использование Weaviate для хранения векторных данных и PostgreSQL для структурированных данных (раздел 3.3), обеспечивают необходимую производительность даже при высоких нагрузках.

**7. Интеграция с внешними системами:** При взаимодействии с внешними HR-платформами (например, hh.ru, LinkedIn Talent Solutions) система должна обеспечивать непрерывный обмен данными без значительных задержек.

* **Требования:**
  + Время импорта данных через API: **не более 10 минут** для 10 000 записей.
  + Время экспорта результатов анализа: **не более 5 минут** для 10 000 записей.
* **Обоснование:** Эффективная интеграция с внешними платформами снижает риск потери данных и обеспечивает своевременную передачу результатов рекрутерам. RESTful API (раздел 3.2) используется для обеспечения гибкого взаимодействия с различными источниками данных.

**8. Обработка больших объемов данных:** Система должна быть готова к работе с миллионами записей без существенного снижения производительности.

* **Требования:**
  + При работе с объемом данных **1 000 000 записей** время обработки одной записи должно составлять **не более 2.5 секунд** .
  + Время полной обработки всех данных: **не более 24 часов** .
* **Обоснование:** Для крупных компаний или кадровых агентств, которые получают значительное количество резюме, требуется высокая производительность системы. Использование векторных баз данных (Weaviate) и механизма Approximate Nearest Neighbors (ANN) (раздел 3.3) позволяет достигать требуемых показателей.

#### **3.2 Надежность**

В данном разделе описываются требования к устойчивости системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти требования направлены на обеспечение стабильной работы системы при различных условиях эксплуатации, минимизацию времени простоя и защиту данных от потери или повреждения.

**1. Требования к отказоустойчивости:** Система должна быть способна восстанавливаться после сбоев без значительных потерь данных и функциональности.

* **Механизмы аварийного восстановления** :
  + Время восстановления после отказа должно составлять **не более 15 минут** для всех ключевых компонентов (BERT-модуль, PostgreSQL, Weaviate).
  + Использование резервных копий данных для быстрого восстановления базы резюме и вакансий.
* **Обработка ошибок взаимодействия между сервисами** :
  + При временной недоступности одного из микросервисов система должна сохранять запросы в очереди и повторно отправлять их после восстановления доступности.
  + Реализация механизма Circuit Breaker для предотвращения перегрузки неисправных компонентов.

**2. Требования к защите от перегрузок:** Система должна поддерживать высокие нагрузки и обеспечивать стабильную работу даже при пиковых значениях входящих данных.

* **Масштабируемость** :
  + Архитектура микросервисов с использованием Kubernetes должна позволять динамически увеличивать количество экземпляров сервисов при необходимости.
  + В случае перегрузки API-шлюза должен быть реализован механизм Rate Limiting для ограничения количества одновременных запросов.
* **Управление пиковыми нагрузками** :
  + Система должна успешно обрабатывать до **10 000 записей в час** , сохраняя время ответа на запрос **не более 2 секунд** .
  + Если объем запросов превышает допустимые пределы, новые запросы помещаются в очередь для последующей обработки.

**3. Резервное копирование данных:** Для защиты информации о резюме, вакансиях и их аналитических данных необходимо реализовать регулярное резервное копирование.

* **Частота резервного копирования** :
  + Ежедневные резервные копии всех данных в PostgreSQL и Weaviate должны создаваться автоматически.
  + Для наиболее важных данных (например, результаты анализа) предусмотрены дополнительные еженедельные архивы.
* **Хранение резервных копий** :
  + Все резервные копии шифруются с использованием AES-256 и хранятся в географически распределенных центрах данных для защиты от локальных сбоев.

**4. Мониторинг состояния системы:** Для своевременного выявления проблем и предотвращения сбоев необходимо внедрить систему мониторинга.

* **Контроль метрик производительности** :
  + Постоянный сбор данных о загрузке процессора, оперативной памяти, GPU и времени выполнения запросов через инструменты мониторинга (например, Prometheus, Grafana).
* **Логирование действий** :
  + Все действия пользователей и события внутри системы должны быть детально задокументированы в логах. Логи хранятся в зашифрованном виде и доступны администраторам через интерфейс ELK Stack.

**5. Защита от некорректных данных:** Система должна быть устойчива к ошибкам в формате входных данных и корректно обрабатывать такие ситуации.

* **Проверка целостности данных** :
  + При загрузке резюме и вакансий проверяется соответствие форматов (PDF, DOCX, TXT) и наличие обязательных полей (название должности, опыт работы, навыки).
  + Если данные не проходят проверку, система должна предоставлять соответствующее уведомление и исключать их из анализа.
* **Обработка исключений** :
  + В случае возникновения ошибок при работе с моделями BERT или базами данных, система должна генерировать информативные сообщения об ошибке и предлагать пути её решения.

**6. Обеспечение надежности векторных данных:** Векторная СУБД (Weaviate) должна обеспечивать высокую степень надежности для хранения и поиска векторных представлений текстов.

* **Индексация данных** :
  + Мыaviate должна использовать алгоритмы Approximate Nearest Neighbors (например, HNSW), чтобы гарантировать быстрый и точный поиск ближайших соседей даже при больших объемах данных.
* **Репликация данных** :
  + Данные в Weaviate должны быть реплицированы для предотвращения потери информации при сбое основного хранилища.

**7. Требования к безопасности и конфиденциальности:** Надежность системы также зависит от уровня её безопасности, особенно при работе с конфиденциальной информацией.

* **Шифрование данных** :
  + Все данные должны шифроваться при передаче (TLS 1.3) и хранении (AES-256).
  + Соблюдение требований GDPR и ФЗ-152 "О персональных данных".
* **Защита от атак** :
  + Реализация механизмов защиты от SQL-инъекций, XSS-атак и несанкционированного доступа.
  + Правильная настройка прав доступа через ролевую модель управления (RBAC).

**8. Обоснование требований к надежности:** Требования к надежности основаны на результатах тестирования прототипа (раздел 4.3), где была продемонстрирована способность системы работать с большими объемами данных (до **172 321 записи** ) и сохранять стабильность при пиковых нагрузках. Например, использование модели BERT показало высокую точность (accuracy = 0.76, MCC = 0.55), но требует надежного хранения и обработки векторных данных, что достигается через интеграцию PostgreSQL и Weaviate.

Кроме того, микросервисная архитектура (раздел 3.2) позволяет минимизировать влияние отказа одного компонента на всю систему, а контейнеризация через Docker и Kubernetes обеспечивает масштабируемость и отказоустойчивость. Это гарантирует, что система будет работать стабильно даже при изменении условий эксплуатации или увеличении объема данных.

Таким образом, требования к надежности системы полностью соответствуют задачам автоматизации отбора кандидатов и обеспечивают её готовность к реальным условиям использования.

#### **3.2 Безопасность**

В данном разделе описываются технические требования к защите данных и системы от несанкционированного доступа для обеспечения безопасности при работе с конфиденциальной информацией резюме, вакансий и других данных, обрабатываемых сервисом автоматизации отбора кандидатов на основе обработки естественного языка.

**1. Требования к защите данных:**

1.1. **Шифрование данных при передаче:**

* Все данные, передаваемые между компонентами системы или внешними платформами, должны быть защищены протоколом **TLS 1.3** .
* Поддержка безопасных каналов взаимодействия через HTTPS для API-запросов.

1.2. **Шифрование данных в покое:**

* Данные, хранящиеся в PostgreSQL и Weaviate, должны шифроваться алгоритмом **AES-256** для защиты от несанкционированного доступа.
* Резервные копии также должны быть зашифрованы тем же алгоритмом перед сохранением.

1.3. **Соблюдение нормативных требований:**

* Система должна соответствовать требованиям **GDPR** и **ФЗ-152 "О персональных данных"** , что гарантирует законную обработку персональной информации соискателей и работодателей.
* Ограничение использования данных только для целей, указанных пользователем при согласии на обработку.

1.4. **Контроль доступа к данным:**

* Реализация ролевой модели управления доступом (**RBAC** ) для всех типов пользователей: рекрутеры, администраторы, научные сотрудники.
* Установка строгих прав доступа к базам данных, где только авторизованные пользователи могут выполнять операции чтения/записи.

1.5. **Логирование действий:**

* Все действия пользователей, связанные с загрузкой, обработкой и анализом данных, должны быть детально задокументированы в логах.
* Логи должны содержать информацию о времени выполнения операции, типе операции, ID пользователя и статусе выполнения.

**2. Требования к защите системы:**

2.1. **Защита от атак:**

* Реализация механизмов защиты от SQL-инъекций, XSS-атак и других распространенных угроз.
* Валидация входных данных на всех уровнях системы (API, веб-интерфейсы, файловые загрузки).

2.2. **Управление аутентификацией и авторизацией:**

* Использование многофакторной аутентификации (**MFA** ) для повышения безопасности учетных записей администраторов и рекрутеров.
* Поддержка современных методов аутентификации, таких как OAuth 2.0 и JWT, для интеграции с внешними системами.

2.3. **Разделение привилегий:**

* Разработка системы управления правами доступа, которая позволяет назначать роли и разрешения на уровне конкретных функций (например, доступ только к просмотру результатов анализа или к изменению параметров вакансии).

2.4. **Мониторинг состояния системы:**

* Непрерывный мониторинг системы с использованием инструментов, таких как ELK Stack или Prometheus, для выявления подозрительной активности.
* Автоматическая генерация оповещений при попытках несанкционированного доступа или других инцидентах безопасности.

**3. Обеспечение конфиденциальности данных:**

3.1. **Анонимизация данных:**

* При необходимости система должна предоставлять возможность анонимизации данных, чтобы исключить возможность идентификации пользователей из обработанных текстов резюме и вакансий.

3.2. **Ограничение обработки персональных данных:**

* Персональные данные (например, имя, контактная информация) должны обрабатываться только при явном согласии пользователя.
* Предоставление функции удаления персональных данных по запросу пользователя для соблюдения GDPR.

**4. Требования к восстановлению после инцидентов:**

4.1. **Резервное копирование:**

* Регулярное создание резервных копий данных (ежедневно для текущих данных, еженедельно для архивных данных).
* Хранение резервных копий в географически распределенных центрах данных для предотвращения потери данных при сбоях основного хранилища.

4.2. **Механизмы восстановления:**

* Время восстановления данных после отказа должно составлять **не более 15 минут** .
* Возможность быстрого переключения на резервные серверы или контейнеры для обеспечения непрерывности работы.

**5. Обоснование требований к безопасности:** Требования к безопасности основаны на необходимости защиты конфиденциальной информации, такой как персональные данные соискателей и внутренние требования работодателей. Например, модель BERT обрабатывает текстовые данные резюме и вакансий, которые могут содержать чувствительную информацию. Шифрование данных (AES-256) и использование TLS 1.3 обеспечивают защиту от несанкционированного доступа во время их передачи и хранения.

Соблюдение требований GDPR и ФЗ-152 гарантирует законность обработки персональных данных, что особенно важно для HR-систем. Кроме того, внедрение RBAC и MFA минимизирует риск утечки данных из-за человеческого фактора или злоумышленников.

Результаты тестирования (раздел 4.3) показывают, что реализованные механизмы безопасности позволяют достичь высокой степени защиты данных, даже при работе с большими объемами информации (до **172 321 записи** ). Это делает систему готовой к эксплуатации в условиях реальных бизнес-процессов.

#### **3.4 Удобство использования**

В данном разделе описываются технические требования к интерфейсу и простоте использования системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти требования направлены на обеспечение интуитивно понятного взаимодействия с системой для всех заинтересованных сторон, таких как рекрутеры, соискатели и администраторы.

**1. Требования к пользовательскому интерфейсу:**

1.1. **Единый язык взаимодействия:**

* Интерфейс должен быть локализован на **русский** и **английский** языки для поддержки международных и локальных пользователей.
* Использование единой терминологии, соответствующей предметной области HR-tech (например, "вакансии", "резюме", "соответствие").

1.2. **Простота навигации:**

* Главная страница должна предоставлять быстрый доступ к основным функциям: созданию/редактированию вакансий, загрузке резюме и просмотру результатов анализа.
* Меню должно быть структурировано по категориям (например, "Управление вакансиями", "Анализ резюме", "Настройки").

1.3. **Поддержка различных форматов данных:**

* Система должна принимать резюме и вакансии в форматах **PDF, DOCX, TXT** , а также предоставлять возможность экспорта результатов в **CSV** или **PDF** .
* Должна быть реализована форма для загрузки данных через drag-and-drop или кнопку "Обзор".

1.4. **Интерфейс управления вакансиями:**

* Рекрутерам должна быть предоставлена удобная форма для создания и редактирования вакансий, включающая поля: название должности, ключевые навыки, опыт работы, образование и дополнительные характеристики.
* Все поля должны иметь подсказки и примеры заполнения для минимизации ошибок при вводе данных.

1.5. **Интерфейс просмотра результатов:**

* Результаты анализа должны быть представлены в виде ранжированного списка с вероятностью соответствия (значение от 0 до 1).
* Для каждого кандидата система должна показывать детальную информацию о совпадении ключевых параметров (например, "Python experience matches", "SQL knowledge required").

**2. Требования к простоте использования:**

2.1. **Минимальное время обучения:**

* Интерфейс должен быть интуитивно понятным, чтобы пользователи могли начать работу с системой без необходимости длительного обучения.
* Предусмотреть наличие справочной документации и видеоинструкций для быстрого освоения.

2.2. **Автоматизация рутинных задач:**

* Система должна автоматически предобрабатывать текстовые данные (удаление стоп-слов, токенизация) и сохранять результаты для дальнейшего анализа.
* Реализовать механизмы автоматической отправки уведомлений рекрутерам о подходящих кандидатах и соискателям об их статусе.

2.3. **Гибкость настроек:**

* Администраторы должны иметь возможность изменять параметры анализа (например, веса для опыта работы, образования и навыков).
* Пользователи могут выбрать формат вывода результатов (например, CSV или PDF) в зависимости от своих потребностей.

2.4. **Оптимизация производительности для конечных пользователей:**

* Время ответа системы на запрос не должно превышать **2 секунды** для обеспечения комфортного взаимодействия.
* При работе с большими объемами данных (до **10 000 записей** ) интерфейс должен оставаться отзывчивым и удобным для использования.

**3. Обоснование требований к удобству использования:** Требования к удобству использования основаны на анализе текущих процессов рекрутмента (раздел 2.3) и тестировании прототипа (раздел 4.3). Например, использование модели BERT для семантического анализа позволяет достичь высокой точности (accuracy = 0.76, MCC = 0.55), но требует интуитивно понятного интерфейса для настройки параметров анализа.

Разработка удобного интерфейса является критически важной для успешного внедрения системы, так как это гарантирует её принятие пользователями и повышает эффективность работы. Микросервисная архитектура (раздел 3.2) и поддержка RESTful API обеспечивают гибкость интеграции с внешними платформами, что упрощает взаимодействие рекрутеров с системой.

Кроме того, использование современных методов обработки естественного языка (NLP) делает процесс анализа более сложным с технической точки зрения, но требования к удобству использования помогают скрыть эту сложность от конечных пользователей, предоставляя им простой и функциональный интерфейс.

### **4. Интерфейсы**

#### **4.1 Внешние интерфейсы**

В данном разделе описываются способы взаимодействия системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий с внешними системами, а также требования к этим интерфейсам для обеспечения корректного обмена данными.

**1. Взаимодействие с HR-платформами:**

Система должна поддерживать интеграцию с популярными платформами рекрутмента через RESTful API для автоматического получения резюме и отправки результатов анализа.

* **Поддерживаемые платформы:** hh.ru, LinkedIn Talent Solutions, HeadHunter API и другие.
* **Требования к API:**
  + Использование протокола HTTPS/TLS для безопасной передачи данных.
  + Поддержка формата JSON для запросов и ответов.
  + Возможность авторизации через OAuth 2.0 или JWT для доступа к данным пользователей.
* **Основные операции:**
  + Загрузка резюме и вакансий в текстовых форматах (PDF, DOCX, TXT).
  + Отправка результатов анализа в виде рейтинга кандидатов с вероятностью соответствия [0, 1].

**2. Взаимодействие с базами данных:**

Система будет использовать внешние базы данных для хранения структурированных и векторных данных.

* **Реляционная СУБД (PostgreSQL):**
  + Хранение метаданных о резюме и вакансиях: ID, название должности, опыт работы, навыки, образование.
  + Интеграция через SQL-запросы для добавления, обновления и чтения данных.
  + Обеспечение ACID-свойств транзакций для надежности данных.
* **Векторная СУБД (Weaviate):**
  + Хранение векторных представлений (embeddings) текстов резюме и вакансий.
  + Взаимодействие через REST API для выполнения операций поиска ближайших соседей и записи новых векторов.
  + Поддержка индексации HNSW для ускорения семантического поиска.

**3. Взаимодействие с файловыми системами:**

Система должна поддерживать загрузку файлов резюме и вакансий различных форматов для их предварительной обработки.

* **Поддерживаемые форматы:** PDF, DOCX, TXT.
* **Требования к интерфейсу:**
  + Реализация механизма drag-and-drop или кнопки "Обзор" для удобства загрузки файлов.
  + Автоматическая проверка формата файла перед обработкой.
  + Возможность конвертации файлов в текстовый формат для последующего анализа.

**4. Взаимодействие с системами безопасности:**

Для обеспечения защиты персональных данных система должна интегрироваться с современными средствами шифрования и управления доступом.

* **Шифрование данных при передаче:**
  + Использование TLS 1.3 для всех каналов связи между компонентами системы и внешними сервисами.
* **Шифрование данных в покое:**
  + Защита хранимых данных с использованием AES-256.
* **Механизмы аутентификации и авторизации:**
  + Поддержка многофакторной аутентификации (MFA) для администраторов и рекрутеров.
  + Использование ролевой модели управления доступом (RBAC) для ограничения прав пользователей.

**5. Взаимодействие с мессенджерами и почтовыми сервисами:**

Система должна предоставлять возможность отправки уведомлений рекрутерам и соискателям через email или корпоративные мессенджеры.

* **Поддерживаемые сервисы:** SMTP для email, Slack или Telegram для корпоративных сообщений.
* **Требования к интерфейсу:**
  + Шаблонизация уведомлений для разных типов событий (например, "Принято", "Отклонено").
  + Возможность настройки получателей уведомлений через административный интерфейс.

**6. Взаимодействие с облачными платформами:**

Для масштабирования и оптимизации вычислительных ресурсов система может быть развернута на облачных платформах.

* **Поддерживаемые платформы:** AWS, Yandex Cloud, Google Cloud Platform.
* **Требования к интерфейсу:**
  + Интеграция через Docker/Kubernetes для контейнеризации и оркестрации микросервисов.
  + Возможность использования GPU-ресурсов облачных платформ для ускорения обучения модели BERT.

**7. Обоснование требований к внешним интерфейсам:**

Требования к внешним интерфейсам основаны на необходимости обеспечить эффективное взаимодействие с различными источниками данных и пользователями системы. Например, использование RESTful API для интеграции с HR-платформами позволяет легко подключать новые источники резюме и вакансий, что повышает гибкость системы. Модель BERT, применяемая для анализа текстов (раздел 3.4), требует значительных вычислительных мощностей, поэтому интеграция с облачными платформами становится критически важной для поддержания производительности при работе с большими объемами данных.

Использование PostgreSQL и Weaviate для хранения данных гарантирует надежное управление как структурированными, так и векторными представлениями текстов, что необходимо для точного сопоставления резюме и вакансий. Тестирование показало, что такая архитектура обеспечивает высокую точность (accuracy = 0.76, MCC = 0.55) даже при анализе сложных текстов (раздел 4.3).

Кроме того, реализация механизмов безопасности и шифрования данных (TLS 1.3, AES-256) гарантирует соответствие требованиям GDPR и ФЗ-152 "О персональных данных", что особенно важно при работе с конфиденциальной информацией.

#### **4.2 Внутренние интерфейсы**

В данном разделе описываются внутренние интерфейсы системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти интерфейсы обеспечивают взаимодействие между компонентами системы, что гарантирует её корректную работу и модульность.

**1. Взаимодействие между сервисами управления резюме и вакансиями:**

* **Описание** : Сервис управления резюме и сервис управления вакансиями должны обмениваться метаданными для последующего анализа.
* **Формат данных** : JSON для структурированных данных.
* **Пример данных** :  
   json
* {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "skills": ["Python", "SQL"],  
   "experience\_years": 5,  
   "education": "Bachelor's degree in Computer Science"  
   }
* **Методы взаимодействия** : RESTful API через HTTP-запросы.

**2. Взаимодействие с модулем векторизации текстов (BERT):**

* **Описание** : Модуль векторизации преобразует текстовые данные резюме и вакансий в контекстуальные векторные представления для дальнейшего анализа.
* **Формат входных данных** : Текстовые строки (например, описание опыта работы или требований).
* **Формат выходных данных** : Векторы размерностью 768 элементов в формате NumPy массивов.
* **Пример взаимодействия** :
  + Вход: "Опыт работы с Python и SQL"
  + Выход: [0.123, 0.456, ..., 0.789] (всего 768 элементов)
* **Методы взаимодействия** : Использование микросервисной архитектуры с вызовом функций через очередь сообщений (например, RabbitMQ).

**3. Взаимодействие с PostgreSQL для хранения структурированных данных:**

* **Описание** : Сервис управления резюме и вакансиями записывает структурированные данные в PostgreSQL, а также считывает их при необходимости.
* **Тип данных** : Структурированные данные, такие как ID резюме/вакансии, название должности, опыт работы, образование.
* **Пример записи** :  
   sql
* SELECT \* FROM vacancies WHERE id = '67890';
* **Методы взаимодействия** : SQL-запросы через ORM (например, SQLAlchemy) для упрощения работы с базой данных.

**4. Взаимодействие с Weaviate для хранения векторных данных:**

* **Описание** : Модуль векторизации передает полученные embeddings в Weaviate для их хранения и быстрого поиска ближайших соседей.
* **Формат данных** : JSON для записи векторов и метаданных.
* **Пример записи** :  
   json
* POST /weaviate/v1/graphql  
   Content-Type: application/json  
   {  
   "query": "{ Get { Resume(where: {path: ["id"], valueText: "12345"}) { vector } } }"  
   }
* **Методы взаимодействия** : REST API Weaviate для выполнения операций записи, обновления и поиска векторов.

**5. Взаимодействие между модулем анализа и модулем ранжирования:**

* **Описание** : После получения векторных представлений из Weaviate и структурированных данных из PostgreSQL, модуль анализа выполняет семантическое сопоставление резюме и вакансий. Результат передается модулю ранжирования для создания рейтинга кандидатов.
* **Формат данных** : Числовые значения вероятности соответствия [0, 1].
* **Пример взаимодействия** :
  + Вход: {"resume\_vector": [0.123, 0.456, ...], "vacancy\_vector": [0.234, 0.567, ...]}
  + Выход: {"similarity\_score": 0.85}
* **Методы взаимодействия** : Обмен данными через внутреннюю очередь сообщений (например, Kafka или Redis).

**6. Взаимодействие с модулем пользовательского интерфейса:**

* **Описание** : Модуль пользовательского интерфейса получает результаты анализа и ранжирования для отображения рекрутерам. Также он отправляет запросы на создание/редактирование вакансий и загрузку резюме.
* **Формат данных** : JSON для передачи данных о рейтингах кандидатов и параметрах вакансий.
* **Пример запроса** :  
   http
* [  
   {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "similarity\_score": 0.85,  
   "reasons": ["Python matches", "SQL required"]  
   },  
   {  
   "resume\_id": "54321",  
   "vacancy\_id": "67890",  
   "similarity\_score": 0.78,  
   "reasons": ["Python matches"]  
   }  
   ]
* **Методы взаимодействия** : RESTful API для обмена данными между фронтендом и бэкендом.

**7. Взаимодействие с модулем безопасности:**

* **Описание** : Все компоненты системы должны использовать модуль безопасности для шифрования данных и аутентификации пользователей.
* **Формат данных** : Шифрованные данные (AES-256) и токены авторизации (JWT).
* **Пример взаимодействия** :
  + Вход: {"username": "recruiter123", "password": "hashed\_password"}
  + Выход: {"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."}
* **Методы взаимодействия** : Протокол HTTPS/TLS для защиты каналов связи, а также использование JWT для авторизации запросов.

**8. Взаимодействие с модулем логирования:**

* **Описание** : Все события внутри системы (загрузка резюме, анализ, ошибки) должны быть задокументированы через модуль логирования.
* **Формат данных** : Структурированные логи в формате JSON.
* **Пример лога** :  
   json
* {  
   "timestamp": "2024-01-01T12:00:00Z",  
   "event": "Resume processed",  
   "details": {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "status": "success",  
   "similarity\_score": 0.85  
   }  
   }
* **Методы взаимодействия** : Интеграция с ELK Stack или аналогичными системами для сбора и анализа логов.

**9. Взаимодействие с модулем интеграции внешних платформ:**

* **Описание** : Модуль интеграции обеспечивает взаимодействие с HR-платформами (например, hh.ru, LinkedIn Talent Solutions) для получения резюме и отправки результатов анализа.
* **Формат данных** : JSON для обмена информацией.
* **Пример запроса** :  
   http
* POST /api/integration/hh.ru  
   Content-Type: application/json  
   {  
   "action": "upload\_resume",  
   "data": {  
   "file\_url": "https://example.com/resume.pdf",  
   "vacancy\_id": "67890"  
   }  
   }
* **Методы взаимодействия** : REST API для обмена данными между внутренними и внешними компонентами.

**10. Обоснование внутренних интерфейсов:**

Выбор внутренних интерфейсов основан на следующих факторах:

* **Масштабируемость** : Использование микросервисной архитектуры и очередей сообщений (например, Kafka) позволяет легко масштабировать систему при увеличении нагрузки.
* **Производительность** : RESTful API обеспечивает быстрый обмен данными между компонентами, особенно при работе с большими объемами информации (до **10 000 записей в час** ).
* **Гибкость** : Формат JSON для передачи данных делает взаимодействие между компонентами универсальным и понятным.
* **Безопасность** : Протокол TLS 1.3 и использование JWT для авторизации гарантируют защиту данных при их передаче между сервисами.

Результаты тестирования показали, что выбранные интерфейсы обеспечивают высокую производительность и надежность работы системы (accuracy = 0.76, MCC = 0.55). Это подтверждает эффективность выбранного подхода к проектированию внутренних интерфейсов.

**5. Ограничения**

#### **5.1 Технические ограничения**

В данном разделе описываются технические ограничения, связанные с аппаратным обеспечением, программным обеспечением и инфраструктурой системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти ограничения определяют рамки разработки и эксплуатации системы.

**1. Аппаратные ограничения:**

1.1. **Вычислительные мощности для работы с моделью BERT:**

* Для корректной работы модели BERT требуется доступ к графическим процессорам (GPU) с поддержкой CUDA. Это связано с высокими вычислительными затратами при обработке семантических данных.
* Минимальные требования к серверу: многопоточный процессор (не менее 4 ядер), оперативная память объемом **16 ГБ** , SSD-диск для хранения данных и моделей.

1.2. **Ограничения по времени обучения модели:**

* Процесс fine-tuning модели BERT занимает значительное время, особенно при работе с большими датасетами. На тестовых данных объемом **172 321 запись** обучение одной эпохи заняло около **37 минут** (раздел 4.4). Поэтому количество эпох ограничивается **4 эпохами** , чтобы минимизировать временные затраты.

1.3. **Масштабируемость оборудования:**

* При использовании облачных решений (например, AWS, Yandex Cloud) необходимо учитывать стоимость и доступность GPU-ресурсов. В локальной инфраструктуре система должна быть способна масштабироваться через добавление новых серверов или увеличение мощности существующих.

**2. Программные ограничения:**

2.1. **Поддерживаемые форматы данных:**

* Система принимает только текстовые файлы в форматах **PDF, DOCX, TXT** для загрузки резюме и вакансий. Другие форматы (например, HTML, JPG) требуют предварительной конвертации.

2.2. **Языковая специфика:**

* Обработка данных осуществляется на **русском** и **английском** языках. Модель BERT, выбранная для реализации, не поддерживает другие языки без дополнительного обучения.

2.3. **Требования к операционной системе:**

* Поддерживаемые операционные системы: **Linux (Ubuntu 20.04+), Windows Server 2019+, macOS Server 10.15+** . Использование других ОС может потребовать доработки компонентов системы.

2.4. **Зависимости от библиотек и фреймворков:**

* Система основана на следующих технологиях:
  + Python (версии 3.8–3.11).
  + Библиотеки для NLP: Hugging Face Transformers, spaCy.
  + Фреймворки для машинного обучения: TensorFlow/PyTorch.
* Использование альтернативных библиотек или фреймворков может привести к снижению производительности или необходимости переписывания кода.

**3. Инфраструктурные ограничения:**

3.1. **Объем хранимых данных:**

* СУБД PostgreSQL используется для хранения структурированных данных о резюме и вакансиях. Однако при работе с миллионами записей возможны проблемы с производительностью, что потребует перехода к более масштабируемым решениям (например, CockroachDB).

3.2. **Ограничения векторной СУБД Weaviate:**

* Размер векторных представлений (embeddings) составляет **768 элементов** для каждой записи. При увеличении количества хранимых данных до нескольких миллионов записей может возникнуть необходимость в оптимизации алгоритмов индексации (например, использование HNSW).

3.3. **Интеграция с внешними платформами:**

* API платформ рекрутмента (например, hh.ru, LinkedIn Talent Solutions) имеют ограничения по количеству запросов в секунду (QPS). Например, hh.ru ограничивает число запросов до **50 в минуту** для бесплатных учетных записей. Это требует реализации очередей запросов для соблюдения лимитов.

3.4. **Ограничения по шифрованию данных:**

* Все данные должны храниться в зашифрованном виде с использованием AES-256, а передача данных осуществляться через TLS 1.3. Это накладывает дополнительную нагрузку на серверную инфраструктуру, особенно при работе с большими объемами данных.

**4. Обоснование технических ограничений:**

Выбор технических ограничений основан на анализе реальных данных (раздел 4.3) и архитектурных решений (раздел 3.2). Например, использование модели BERT для векторизации текстов позволяет достичь accuracy = 0.76 и MCC = 0.55, но требует наличия GPU для ускорения вычислений. Кроме того, ограничения по форматам входных данных (PDF, DOCX, TXT) связаны с необходимостью стандартизации обработки текста для достижения высокой точности анализа.

Использование PostgreSQL и Weaviate для хранения данных гарантирует эффективную работу системы при текущих объемах данных (до **10 000 записей в час** ), но при значительном росте нагрузки могут потребоваться дополнительные оптимизации или переход к другим решениям.

#### **5.2 Ограничения по времени**

В данном разделе описываются требования к времени выполнения операций в системе автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти ограничения направлены на обеспечение высокой производительности системы при работе с большими объемами данных, а также на поддержание комфортного взаимодействия пользователей.

**1. Время обработки одного резюме:** Система должна быть способна обрабатывать одно резюме за **не более 2 секунды** , включая все этапы: предобработку текста, векторизацию с помощью модели BERT и семантическое сопоставление с требованиями вакансии.

* **Обоснование:** На тестовых данных объемом **172 321 запись** достигнутые метрики точности (accuracy = 0.76) и коэффициента корреляции Мэтьюса (MCC = 0.55) демонстрируют возможность соблюдения данного ограничения при использовании оптимизированных алгоритмов и вычислительных ресурсов (раздел 4.3).

**2. Общее время обработки больших объемов данных:** Система должна обрабатывать до **10 000 записей в час** (пар "резюме-вакансия"), сохраняя стабильное время выполнения для каждой записи.

* **Обоснование:** Исходя из результатов нагрузочного тестирования, система успешно справилась с обработкой нескольких тысяч записей одновременно, что позволяет установить данное требование как достижимое для реальных условий эксплуатации (раздел 4.3).

**3. Время ответа API:** Все запросы через RESTful API должны обрабатываться за **не более 2 секунды** , независимо от типа операции (загрузка резюме, создание вакансии, получение результатов анализа).

* **Примеры:**
  + Запрос на загрузку резюме:  
     http
  + GET /api/ranking?vacancy\_id=67890  
     Accept: application/json  
     Результат должен быть сформирован и отправлен клиенту за **менее 2 секунд** .
* **Обоснование:** Быстрый отклик API необходим для обеспечения удобства использования системы рекрутерами и соискателями, особенно при работе с внешними платформами, такими как hh.ru или LinkedIn Talent Solutions (раздел 3.4).

**4. Время обучения модели BERT:** Процесс fine-tuning модели BERT на новых данных не должен занимать более **37 минут на одну эпоху** при использовании GPU. Обучение должно завершаться за **не более 2 часа** для четырех эпох.

* **Обоснование:** Экспериментальные данные показывают, что обучение модели BERT на тестовом датасете объемом **172 321 запись** заняло около 37 минут на эпоху, что соответствует установленному ограничению (Приложение 1).

**5. Время восстановления после отказа:** При возникновении сбоев система должна быть восстановлена в рабочее состояние за **не более 15 минут** . Это включает:

* Восстановление баз данных PostgreSQL и Weaviate.
* Перезапуск микросервисов.
* Возобновление работы API.
* **Обоснование:** Низкое время восстановления (MTTR) является критически важным для обеспечения надежности системы, особенно при её интеграции с бизнес-процессами компаний (раздел 3.4).

**6. Время формирования отчетов:** Система должна генерировать отчеты о результате анализа резюме в форматах CSV или PDF за **не более 5 минут** для набора из **1000 записей** .

* **Обоснование:** Скорость формирования отчетов важна для HR-специалистов, которые могут требовать быстрого предоставления результатов для принятия решений (раздел 2.4).

**7. Время обновления данных вакансий:** При изменении параметров вакансии система должна пересчитать соответствие всех связанных резюме за **не более 5 минут** для набора из **100 записей** .

* **Обоснование:** Данное ограничение гарантирует актуальность результатов анализа даже при частых изменениях требований работодателя (раздел 2.3).

**8. Время загрузки данных из внешних систем:** Загрузка данных о резюме и вакансиях через API внешних платформ (например, hh.ru) должна занимать **не более 10 минут** для **10 000 записей** .

* **Обоснование:** Интеграция с внешними системами требует учета ограничений их API, таких как количество запросов в секунду (QPS), что делает это требование реалистичным и достижимым (раздел 2.3).

**9. Время резервного копирования данных:** Резервное копирование данных из PostgreSQL и Weaviate должно выполняться ежедневно за **не более 30 минут** для текущего объема данных (до **100 000 записей** ).

* **Обоснование:** Регулярное резервное копирование необходимо для защиты данных от потери, при этом время выполнения должно быть минимальным, чтобы не влиять на работу системы (раздел 3.3).

**10. Время мониторинга и логирования:** Система должна собирать и записывать логи событий в реальном времени, без задержек более **1 секунды** .

* **Обоснование:** Непрерывный мониторинг состояния системы и детальное логирование помогают быстро выявлять и исправлять ошибки, что повышает общую надежность решения (раздел 4.3).

**11. Время первичной загрузки системы:** Система должна запускаться и становиться доступной для пользователей за **не более 2 минут** после старта контейнеров в Docker/Kubernetes.

* **Обоснование:** Быстрая загрузка системы важна для минимизации простоя при перезапусках или масштабировании компонентов (раздел 4.1).

**12. Время обработки пиковых нагрузок:** При пиковой нагрузке (до **100 одновременных пользователей** ) время выполнения операций не должно увеличиваться более чем на **50%** по сравнению с нормальными условиями.

* **Обоснование:** Микросервисная архитектура с использованием Kubernetes обеспечивает горизонтальное масштабирование, что позволяет справляться с пиковыми нагрузками без значительного снижения производительности (раздел 3.2).  
  POST /api/resume  
   Content-Type: application/json  
    
    
   Ответ должен быть предоставлен за **менее 2 секунд** после получения файла.

# SRS (Software Requirements Specification)

### **1. Введение**

#### **1.1 Цель ПО**

Целью разрабатываемого программного обеспечения является автоматизация процесса первичного отбора кандидатов на основе обработки естественного языка из текстовых данных резюме и вакансий. Программное обеспечение представляет собой модуль системы автоматизации рекрутмента, направленный на минимизацию влияния человеческого фактора, улучшение точности сопоставления навыков кандидатов с требованиями вакансий и повышение эффективности работы HR-специалистов.

Основные цели программного обеспечения:

1. **Семантический анализ текстов** :
   * Использование модели BERT для преобразования текстов резюме и вакансий в контекстуальные векторные представления (embeddings), что позволяет учитывать смысл слов и их взаимосвязи в предложении.
   * Выявление ключевых параметров (опыт работы, образование, навыки) из неструктурированных текстовых данных.
2. **Сопоставление кандидатов и вакансий** :
   * Реализация механизма сравнения векторных представлений резюме и вакансий через вычисление косинусного расстояния для оценки степени соответствия.
   * Формирование рейтинга кандидатов с вероятностью принятия от 0 до 1, где 1 означает максимальное соответствие вакансии.
3. **Ускорение процесса рекрутмента** :
   * Сокращение времени анализа одного резюме до **не более 2 секунд** , что позволяет обрабатывать до **10 000 записей в час** .
   * Автоматическая фильтрация резюме, исключающая необходимость ручного просмотра большого количества документов.
4. **Обеспечение прозрачности решений** :
   * Предоставление детальной информации о причинах ранжирования каждого кандидата (например, совпадение опыта работы или конкретных навыков).
   * Логирование действий системы для последующего анализа и контроля качества работы алгоритмов.
5. **Поддержка интеграции с внешними системами** :
   * Интеграция с платформами поиска работы (например, hh.ru, LinkedIn Talent Solutions) через RESTful API для автоматического получения резюме и отправки результатов анализа.
   * Возможность работы с различными форматами данных (PDF, DOCX, TXT) для обеспечения широкой применимости решения.
6. **Соблюдение требований безопасности** :
   * Защита персональных данных пользователей путем шифрования при передаче (TLS 1.3) и хранении (AES-256).
   * Соответствие нормативным требованиям GDPR и ФЗ-152 "О персональных данных".
7. **Гибкость и адаптивность** :
   * Возможность настройки весов для различных параметров (опыт работы, образование, навыки) под специфику компании или вакансии.
   * Поддержка fine-tuning модели BERT для повышения точности классификации на основе новых данных.

Разрабатываемое программное обеспечение предназначено для замены ручной фильтрации резюме, которая является наиболее трудоемким этапом процесса рекрутмента. Благодаря использованию современных технологий обработки естественного языка (NLP), таких как трансформеры и методы векторизации текста, программа способна достичь высокой точности (accuracy = 0.76) и коэффициента корреляции Мэтьюса (MCC = 0.55) на тестовых данных объемом **172 321 запись** .

Таким образом, программное обеспечение обеспечивает комплексную автоматизацию процесса отбора кандидатов, снижая временные затраты HR-специалистов и повышая объективность решений за счет учета контекста текстов. Это делает его важной частью целевой системы автоматизации рекрутмента.

#### **1.2 Область применения**

В данном разделе определяются границы программного обеспечения (ПО), разрабатываемого для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий, а также описывается его взаимодействие с другими компонентами системы.

**1. Границы программного обеспечения:**

Программное обеспечение является частью более крупной системы автоматизации рекрутмента и фокусируется на решении задачи первичного анализа и сопоставления текстовых данных резюме и вакансий. Основные границы ПО включают:

1.1. **Обработка входных данных:**

* Программное обеспечение принимает текстовые файлы резюме и вакансий в форматах **PDF, DOCX, TXT** .
* Данные предварительно проверяются на соответствие форматам и наличие обязательных полей (например, название должности, ключевые навыки).

1.2. **Внутренние процессы:**

* Система преобразует текстовые данные в структурированные представления через механизмы токенизации, лемматизации и удаления стоп-слов.
* Используется модель **BERT** для создания контекстуальных векторных представлений (embeddings) текстов резюме и вакансий.
* Выполняется семантическое сопоставление между векторными представлениями для оценки соответствия кандидатов требованиям вакансии.

1.3. **Выходные данные:**

* Результатом работы ПО является ранжированный список кандидатов с вероятностью их соответствия конкретной вакансии (числовое значение от 0 до 1).
* Детализированная информация о причинах ранжирования (например, совпадение навыков, уровня образования или опыта работы).

1.4. **Ограничения функциональности:**

* Программное обеспечение не осуществляет анализ "мягких" навыков (например, лидерства, коммуникабельности), если они явно не указаны в тексте.
* Не поддерживается обработка графических файлов (например, JPG, PNG) или других форматов, не связанных с текстом.

**2. Взаимодействие с другими компонентами системы:**

Разрабатываемое ПО интегрируется в существующую экосистему рекрутмента и взаимодействует с следующими компонентами:

2.1. **Реляционная база данных (PostgreSQL):**

* Хранение структурированных данных о резюме и вакансиях, таких как ID, название должности, опыт работы, образование.
* Программное обеспечение выполняет запись и чтение данных из PostgreSQL через SQL-запросы.

2.2. **Векторная база данных (Weaviate):**

* Хранение векторных представлений текстов резюме и вакансий для быстрого поиска ближайших соседей.
* Программное обеспечение передает embeddings в Weaviate и получает результаты сравнения через REST API.

2.3. **API внешних платформ:**

* Взаимодействие с HR-платформами (например, hh.ru, LinkedIn Talent Solutions) для получения резюме и отправки результатов анализа.
* Использование протокола HTTPS/TLS для безопасной передачи данных через RESTful API.

2.4. **Микросервисная архитектура:**

* Программное обеспечение разворачивается как один из микросервисов системы, взаимодействуя с другими сервисами через API-шлюз.
* Сервис управления вакансиями передает информацию о новой вакансии для последующего анализа.
* Сервис пользователей предоставляет данные о правах доступа и авторизации.

2.5. **Фронтенд интерфейс:**

* Программное обеспечение взаимодействует с пользовательским интерфейсом, предоставляя результаты анализа для отображения рекрутерам.
* Поддерживает форматы ответов **JSON** и **CSV** для интеграции с различными клиентскими приложениями.

**3. Обоснование границ и взаимодействия:**

Выбор границ программного обеспечения и его взаимодействия с другими компонентами системы основан на необходимости эффективной обработки больших объемов текстовых данных и минимизации влияния человеческого фактора при отборе кандидатов. Например, использование модели BERT для семантического анализа позволяет достичь высокой точности (accuracy = 0.76) и коэффициента корреляции Мэтьюса (MCC = 0.55) на тестовых данных объемом **172 321 запись** (раздел 4.3).

Интеграция с PostgreSQL и Weaviate обеспечивает надежное хранение как структурированных, так и векторных данных, что гарантирует корректную работу системы даже при увеличении нагрузки. Микросервисная архитектура (раздел 3.2) позволяет легко масштабировать и обновлять функциональность без перерыва в работе всей системы.

Таким образом, данное программное обеспечение четко ограничивается задачами анализа текстовой информации и ее сопоставления с требованиями вакансий, предоставляя качественные результаты для дальнейшей обработки и использования другими компонентами системы.

#### **1.3 Определения**

В данном разделе приводятся термины и сокращения, используемые в документе для описания требований к программному обеспечению, разрабатываемому как часть системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий.

### **Термины:**

1. **AS IS** — Текущее состояние процесса рекрутмента, характеризующееся ручной фильтрацией резюме и ограниченной автоматизацией.
2. **TO BE** — Целевое состояние процесса рекрутмента после внедрения автоматизированного сервиса, обеспечивающее более высокую эффективность и точность отбора кандидатов.
3. **BERT (Bidirectional Encoder Representations from Transformers)** — Нейронная модель для обработки естественного языка, основанная на трансформерах и позволяющая создавать контекстуальные векторные представления текстов.
4. **ContinueWork** — Наименование разрабатываемого программного обеспечения для автоматизации отбора кандидатов на основе NLP.
5. **Docker** — Инструмент для контейнеризации приложений, обеспечивающий их изоляцию и переносимость между средами выполнения.
6. **Embeddings** — Векторные представления текстовых данных, полученные с использованием моделей обработки естественного языка (например, BERT).
7. **Fine-tuning** — Процесс адаптации предобученной модели (например, BERT) под конкретную задачу путем дополнительного обучения на специализированных данных.
8. **GPU (Graphics Processing Unit)** — Графический процессор, используемый для ускорения вычислений в моделях глубокого обучения.
9. **HNSW (Hierarchical Navigable Small World Graphs)** — Алгоритм поиска ближайших соседей в векторных пространствах, применяемый в Weaviate для быстрого сопоставления резюме и вакансий.
10. **MCC (Matthews Correlation Coefficient)** — Метрика качества классификации, учитывающая истинные положительные, истинные отрицательные, ложные положительные и ложные отрицательные значения; принимает значения от -1 до 1.
11. **NLP (Natural Language Processing)** — Область искусственного интеллекта, занимающаяся анализом и обработкой естественного языка.
12. **RESTful API** — Интерфейс взаимодействия между компонентами системы, основанный на архитектурном стиле REST (Representational State Transfer).
13. **Resume (Резюме)** — Документ, содержащий информацию о соискателе, включая навыки, опыт работы и образование.
14. **Self-Attention** — Механизм внимания в трансформерах, который позволяет модели анализировать текст двунаправленно, учитывая контекст слов.
15. **Validation Loss** — Потери функции ошибки на валидационном наборе данных во время обучения модели машинного обучения.
16. **Vacancy (Вакансия)** — Описание открытой позиции работодателя с указанием требований и условий труда.
17. **Vector Database** — База данных для хранения векторных представлений текстов, таких как embeddings, с поддержкой быстрого поиска семантической близости.
18. **Weaviate** — Специализированная векторная база данных, используемая для хранения и анализа embeddings резюме и вакансий.
19. **PostgreSQL** — Реляционная система управления базами данных с поддержкой JSONB и ACID-свойств для хранения структурированных данных.
20. **JSON (JavaScript Object Notation)** — Легковесный формат обмена данными, используемый для передачи структурированной информации между компонентами системы.
21. **ACID** — Набор свойств транзакций в базах данных: атомарность, согласованность, изолированность, надежность.
22. **Microservices Architecture (Микросервисная архитектура)** — Архитектурный подход, при котором программа разбивается на независимые сервисы, каждый из которых выполняет свою функциональную роль.
23. **Cosine Similarity (Косинусное расстояние)** — Метрика для измерения сходства между векторными представлениями текстов.
24. **UBML (Unified Business Modeling Language)** — Стандарт для моделирования бизнес-процессов, используемый для создания диаграмм AS IS и TO BE.
25. **UML (Unified Modeling Language)** — Формальный стандарт для визуализации архитектуры программного обеспечения, включая диаграммы состояний и компонентов.
26. **SQL (Structured Query Language)** — Язык запросов для работы с реляционными базами данных.
27. **API (Application Programming Interface)** — Интерфейс программирования приложений, используемый для взаимодействия между компонентами системы и внешними платформами.
28. **GDPR (General Data Protection Regulation)** — Регламент защиты персональных данных Европейского Союза.
29. **ФЗ-152 "О персональных данных"** — Российский закон о защите персональных данных пользователей.
30. **C4 Model** — Методология визуализации архитектуры программного обеспечения, используемая для описания структуры системы на разных уровнях детализации.

### **Сокращения:**

1. **API** — Application Programming Interface (интерфейс программирования приложений).
2. **AS** — Current State (текущее состояние).
3. **TO** — Target State (целевое состояние).
4. **NLP** — Natural Language Processing (обработка естественного языка).
5. **GPU** — Graphics Processing Unit (графический процессор).
6. **SQL** — Structured Query Language (язык запросов для реляционных баз данных).
7. **MVP** — Minimum Viable Product (минимально жизнеспособный продукт).
8. **IDE** — Integrated Development Environment (интегрированная среда разработки).
9. **JSON** — JavaScript Object Notation (формат обмена данными).
10. **ACID** — Atomicity, Consistency, Isolation, Durability (свойства транзакций).
11. **ML** — Machine Learning (машинное обучение).
12. **HR-tech** — Технологии для управления человеческими ресурсами.

### **2. Функциональные требования**

#### **2.1 Функции ПО**

В данном разделе описываются основные функции программного обеспечения, разрабатываемого для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти функции обеспечивают выполнение ключевых задач системы.

**1. Векторизация текстовых данных:** Программное обеспечение должно преобразовывать тексты резюме и вакансий в контекстуальные векторные представления с использованием модели BERT.

* **Вход** : Тексты резюме и вакансий в форматах PDF, DOCX, TXT.
* **Выход** : Векторные представления размерностью 768 для каждого документа.
* **Цель** : Обеспечить семантическое понимание текстов для их дальнейшего анализа.

**2. Извлечение ключевых параметров:** Программное обеспечение должно автоматически извлекать важную информацию из текстов резюме и вакансий, такую как:

* Опыт работы.
* Ключевые навыки (жесткие и мягкие).
* Образование.
* Дополнительные характеристики (например, сертификаты или профессиональные достижения).
* **Вход** : Векторные представления текстов.
* **Выход** : Структурированные данные в формате JSON для хранения в PostgreSQL.
* **Цель** : Преобразовать неструктурированные тексты в структурированные данные для удобства анализа.

**3. Семантическое сопоставление резюме и вакансий:** Программное обеспечение должно сравнивать векторные представления резюме и вакансий, используя косинусное расстояние, чтобы определить степень соответствия.

* **Вход** : Векторные представления резюме и вакансий из Weaviate.
* **Выход** : Числовое значение [0, 1], где 0 — полное несоответствие, а 1 — максимальное соответствие.
* **Цель** : Ранжировать кандидатов по уровню соответствия требованиям вакансии.

**4. Создание и управление вакансиями:** Программное обеспечение должно предоставлять интерфейс для создания и редактирования вакансий, включая описание требований и необходимых навыков.

* **Вход** : Параметры вакансии (название должности, опыт работы, ключевые навыки, образование).
* **Выход** : Сохранение данных о вакансии в PostgreSQL и её векторного представления в Weaviate.
* **Цель** : Обеспечить возможность рекрутерам добавлять новые вакансии и обновлять существующие.

**5. Прием и предварительная обработка резюме:** Программное обеспечение должно принимать резюме от соискателей в поддерживаемых форматах и проводить их предобработку (токенизация, лемматизация, удаление стоп-слов).

* **Вход** : Файлы резюме в форматах PDF, DOCX, TXT.
* **Выход** : Предобработанный текст для последующей векторизации и метаданные для сохранения в PostgreSQL.
* **Цель** : Подготовить резюме для анализа и устранить влияние различий в форматировании.

**6. Ранжирование кандидатов:** Программное обеспечение должно формировать список кандидатов, отсортированных по уровню соответствия требованиям вакансии.

* **Вход** : Результаты семантического сопоставления для всех пар "резюме-вакансия".
* **Выход** : Рейтинг кандидатов с вероятностями принятия (например, 0.8 для высокой вероятности).
* **Цель** : Предоставить рекрутерам готовый список наиболее подходящих кандидатов для каждой вакансии.

**7. Интеграция с внешними платформами:** Программное обеспечение должно поддерживать интеграцию с HR-платформами через RESTful API для автоматического получения резюме и отправки результатов анализа.

* **Вход** : Запросы от платформ (например, hh.ru, LinkedIn Talent Solutions).
* **Выход** : Ответы в формате JSON с данными о ранжировании кандидатов.
* **Цель** : Упростить взаимодействие с существующими системами рекрутмента.

**8. Генерация отчетов:** Программное обеспечение должно генерировать отчеты о результате анализа резюме для конкретной вакансии в форматах CSV и PDF.

* **Вход** : Результаты анализа соответствия кандидатов.
* **Выход** : Отчеты с детализацией причин принятия/отклонения резюме.
* **Цель** : Предоставить рекрутерам прозрачность процесса отбора и возможность использования данных для дальнейшего анализа.

**9. Уведомления пользователей:** Программное обеспечение должно отправлять автоматические уведомления рекрутерам и соискателям о состоянии обработки резюме.

* **Вход** : Результаты анализа конкретного резюме.
* **Выход** : Email-уведомления или сообщения через корпоративные мессенджеры.
* **Цель** : Обеспечить своевременную обратную связь всем участникам процесса.

**10. Обновление данных вакансий:** Программное обеспечение должно поддерживать возможность обновления информации о вакансии, после чего пересчитывать соответствие всех ранее загруженных резюме.

* **Вход** : Измененные параметры вакансии.
* **Выход** : Обновленные векторные представления вакансии и новый рейтинг кандидатов.
* **Цель** : Гарантировать актуальность результатов при изменении требований работодателя.

**11. Fine-tuning модели BERT:** Программное обеспечение должно позволять администраторам проводить fine-tuning модели BERT на новых данных для повышения точности классификации.

* **Вход** : Размеченный датасет с парами "резюме-вакансия" и метками соответствия.
* **Выход** : Обновленная модель BERT с новыми весами.
* **Цель** : Адаптировать модель под специфику компании или рынка труда.

**12. Поиск ближайших соседей:** Программное обеспечение должно использовать алгоритмы Approximate Nearest Neighbors (например, HNSW) для быстрого поиска наиболее релевантных резюме в векторном пространстве.

* **Вход** : Векторное представление вакансии.
* **Выход** : Список ближайших соседей (резюме) с расстояниями.
* **Цель** : Ускорить процесс поиска подходящих кандидатов при работе с большими объемами данных.

**13. Мониторинг качества анализа:** Программное обеспечение должно предоставлять метрики качества работы алгоритмов (accuracy, MCC) для оценки эффективности сопоставления резюме и вакансий.

* **Вход** : Результаты анализа и реальные метки соответствия.
* **Выход** : Отчеты с метриками производительности (например, accuracy = 0.76, MCC = 0.55).
* **Цель** : Обеспечить контроль над качеством работы системы и возможность её доработки.

#### **2.2 Входы и выходы**

В данном разделе описываются данные, которые программное обеспечение (ПО) должно обрабатывать на входе, а также результаты его работы, представленные в виде выходных данных.

**1. Входные данные:**

1.1. **Текстовые файлы резюме и вакансий:**

* **Форматы:** PDF, DOCX, TXT.
* **Содержимое:** Описания опыта работы, навыков, образования, достижений и других характеристик кандидата или требований работодателя.
* **Пример:** Резюме может содержать фразы: "Опыт работы с Python более 5 лет", "Управление проектами", "Высшее техническое образование".

1.2. **Структурированные метаданные:**

* **Формат:** JSON или CSV.
* **Содержимое:** Дополнительная информация о резюме и вакансиях, такая как:
  + ID документа.
  + Категория вакансии (IT, Маркетинг, Производство).
  + Дата публикации вакансии.
  + Статус обработки резюме (новое, обработанное, ранжированное).
* **Пример:** json
* {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "category": "IT",  
   "event\_date": "2024-01-01"  
   }

1.3. **Запросы через API:**

* **Формат:** HTTP/HTTPS через RESTful API.
* **Содержимое:** Запросы на загрузку новых резюме, создание/редактирование вакансий, получение результатов анализа.
* **Пример запроса:** http  
  POST /api/resume  
   Content-Type: application/json  
   {  
   "file\_url": "https://example.com/resume.pdf",  
   "vacancy\_id": "67890"  
   }

**2. Выходные данные:**

2.1. **Результаты семантического анализа:**

* **Формат:** JSON или CSV.
* **Содержимое:** Числовые значения [0, 1], представляющие вероятность соответствия резюме требованиям вакансии, где 0 — полное несоответствие, а 1 — максимальное соответствие.
* **Пример:** json
* {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "similarity\_score": 0.85,  
   "reasons": ["Python matches", "SQL required"]  
   }

2.2. **Ранжированный список кандидатов:**

* **Формат:** CSV или PDF.
* **Содержимое:** Отсортированный список кандидатов по уровню соответствия для конкретной вакансии.

2.3. **Уведомления пользователей:**

* **Формат:** Email, SMS или push-уведомления через API.
* **Содержимое:** Информация о статусе обработки резюме (например, "Принято" или "Отклонено").
* **Пример уведомления рекрутеру:** "Найдено 10 подходящих кандидатов для вакансии 'Инженер-программист'."
* **Пример уведомления соискателю:** "Ваше резюме рассмотрено для вакансии 'Инженер-программист'. Вероятность соответствия: 85%."

2.4. **Логи событий:**

* **Формат:** JSON или текстовый формат.
* **Содержимое:** Информация о времени выполнения операций, типах событий, ID резюме/вакансии, статусах обработки.
* **Пример:** json  
  {  
   "timestamp": "2024-01-01T12:00:00Z",  
   "event": "Resume processed",  
   "details": {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "status": "success",  
   "similarity\_score": 0.85  
   }

2.5. **Векторные представления текстов:**

* **Формат:** NumPy массивы размерностью 768 для каждого документа.
* **Содержимое:** Векторные представления (embeddings) текстов резюме и вакансий, полученные с использованием модели BERT.
* **Пример:** [0.123, 0.456, ..., 0.789] (всего 768 элементов).

**3. Требования к входным и выходным данным:**

3.1. **Поддерживаемые форматы данных:**

* Программное обеспечение должно принимать только текстовые файлы в форматах PDF, DOCX, TXT для резюме и вакансий. Другие форматы (например, HTML, JPG) требуют предварительной конвертации.

3.2. **Объем данных:**

* ПО должно быть способно обрабатывать большие объемы данных: до **172 321 записи** для обучения моделей и до **10 000 записей в час** для реальной эксплуатации.

3.3. **Метрики качества:**

* На выходе должны быть предоставлены метрики точности анализа: accuracy ≥ 0.76 и MCC ≥ 0.55, что подтверждает высокую эффективность системы.

3.4. **Шифрование данных:**

* Все данные на входе и выходе должны шифроваться при передаче (TLS 1.3) и хранении (AES-256) для обеспечения безопасности персональной информации.

**4. Обоснование входов и выходов:**

Выбор входных и выходных данных основан на задачах автоматизации отбора кандидатов и особенностях их обработки. Например, использование модели BERT для векторизации текстов позволяет достичь высокой точности (accuracy = 0.76, MCC = 0.55) на тестовых данных объемом **172 321 запись** (раздел 4.3). Это гарантирует корректный анализ даже при работе с разнообразными текстовыми форматами резюме и вакансий.

RESTful API обеспечивает удобное взаимодействие между компонентами системы и внешними платформами, такими как hh.ru или LinkedIn Talent Solutions. Шифрование данных (TLS 1.3, AES-256) необходимо для соблюдения требований GDPR и ФЗ-152 "О персональных данных".

Генерация ранжированных списков кандидатов в формате CSV или PDF помогает рекрутерам быстро оценить наиболее подходящих специалистов для каждой вакансии, что существенно ускоряет процесс отбора.

### **3. Нефункциональные требования**

#### **3.1 Производительность**

В данном разделе описываются требования к скорости и эффективности программного обеспечения, разрабатываемого для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти требования обеспечивают корректную работу системы при больших нагрузках и гарантируют высокую производительность.

**1. Требования к скорости обработки:**

1.1. **Обработка одного резюме или вакансии:**

* Программное обеспечение должно быть способно обработать одно резюме или вакансию за **не более 2 секунды** , включая предобработку текста, его векторизацию с использованием модели BERT и семантическое сопоставление.

1.2. **Массовая обработка данных:**

* Система должна обрабатывать до **10 000 записей в час** (пары "резюме-вакансия"), сохраняя стабильное время выполнения операций для каждой записи.
* При тестировании прототипа на датасете объемом **172 321 запись** достигнутые метрики точности (accuracy = 0.76) и коэффициента корреляции Мэтьюса (MCC = 0.55) подтверждают возможность соблюдения данного ограничения.

1.3. **Размер обрабатываемых данных:**

* ПО должно поддерживать загрузку и анализ файлов резюме и вакансий размером до **5 МБ** для каждого документа.

**2. Требования к эффективности:**

2.1. **Точность анализа:**

* Программное обеспечение должно демонстрировать высокую точность классификации: метрика accuracy должна составлять **не менее 0.76** , а MCC — **не менее 0.55** .

2.2. **Эффективность поиска ближайших соседей:**

* Использование алгоритмов Approximate Nearest Neighbors (например, HNSW) должно обеспечивать поиск подходящих резюме среди миллионов записей за **считанные миллисекунды** .

2.3. **Интеграция с внешними платформами:**

* Время получения данных через API платформ рекрутмента (например, hh.ru) и их загрузка в систему не должно превышать **10 минут** для набора из **10 000 записей** .

**3. Обоснование требований к производительности:**

Выбор требований к скорости и эффективности ПО основан на анализе реальных данных (раздел 4.3) и особенностях работы модели BERT (раздел 1.4). Например, модель BERT показала способность преобразовывать тексты в контекстуальные векторные представления и проводить их сопоставление с требованиями вакансий за приемлемое время при работе с большими объемами данных.

Микросервисная архитектура (раздел 3.2) и использование распределенной инфраструктуры позволяют масштабировать систему и минимизировать задержки при пиковых нагрузках. RESTful API обеспечивает быстрый обмен данными между компонентами системы и внешними сервисами, что гарантирует стабильную производительность даже при одновременной обработке множества запросов.

#### **3.2 Надежность**

В данном разделе описываются требования к устойчивости программного обеспечения (ПО), разрабатываемого для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти требования направлены на обеспечение стабильной работы системы при различных сбоях, нагрузках и условиях эксплуатации.

**1. Требования к восстановлению после сбоев:**

1.1. **Время восстановления (MTTR):**

* ПО должно быть способно восстанавливаться после отказа любой компоненты системы за **не более 15 минут** . Это включает:
  + Восстановление данных из резервных копий PostgreSQL и Weaviate.
  + Перезапуск микросервисов через Docker/Kubernetes.
  + Возобновление работы API.

1.2. **Механизмы аварийного восстановления:**

* Регулярное создание резервных копий данных:
  + Ежедневные полные бэкапы PostgreSQL и Weaviate.
  + Еженедельные дифференциальные бэкапы для оптимизации места хранения.
* Все резервные копии должны шифроваться алгоритмом AES-256 перед сохранением.

1.3. **Обработка временных сбоев:**

* При временном недоступности внешних платформ (например, hh.ru) или внутренних сервисов система должна помещать запросы в очередь для последующей обработки.
* Использование инструментов, таких как RabbitMQ или Kafka, для управления очередями запросов.

**2. Защита от перегрузок:**

2.1. **Управление пиковыми нагрузками:**

* ПО должно поддерживать горизонтальное масштабирование через Kubernetes для распределения нагрузки между экземплярами сервисов.
* Максимальная обработка данных: до **10 000 записей в час** , без потери производительности или увеличения времени ответа более чем на **20%** .

2.2. **Лимитирование запросов (Rate Limiting):**

* Для предотвращения перегрузки API необходимо ограничить количество одновременных запросов от одного пользователя до **100 в минуту** .
* При превышении лимита система должна возвращать код ошибки HTTP 429 Too Many Requests с указанием времени ожидания повторной попытки.

**3. Обеспечение отказоустойчивости:**

3.1. **Репликация данных:**

* PostgreSQL должна использовать механизм репликации для создания нескольких копий данных, что гарантирует доступность информации даже при отказе основного сервера.
* Weaviate должен поддерживать репликацию векторных данных через индексацию HNSW для обеспечения быстрого поиска соседей в случае сбоя основного хранилища.

3.2. **Мониторинг состояния системы:**

* Система должна предоставлять детальную информацию о состоянии всех компонентов через мониторинговые инструменты, такие как Prometheus и Grafana.
* Автоматическая генерация оповещений администраторам при возникновении проблем, таких как:
  + Отказ микросервисов.
  + Превышение пороговых значений загрузки процессора, GPU или оперативной памяти.
  + Проблемы с доступностью внешних платформ (API hh.ru, LinkedIn Talent Solutions).

3.3. **Mechanism Circuit Breaker:**

* Реализация механизма Circuit Breaker для защиты от множественных неудачных попыток взаимодействия с внешними сервисами. Если API платформы рекрутмента становится недоступным, система должна временно прекратить отправку запросов и повторить их после восстановления доступности.

**4. Устойчивость к ошибкам входных данных:**

4.1. **Проверка форматов файлов:**

* ПО должно проверять форматы входных данных (PDF, DOCX, TXT) перед началом обработки. Если формат неверный, система должна возвращать соответствующее уведомление пользователю с указанием причины ошибки.

4.2. **Обработка некорректных текстов:**

* При обнаружении поврежденных или некорректных текстов в резюме или вакансиях ПО должно исключать их из анализа и регистрировать в логах для дальнейшего рассмотрения администратором.

**5. Обоснование требований к надежности:**

Требования к надежности ПО основаны на необходимости обеспечить непрерывность работы системы при больших объемах данных и высоких нагрузках. Например, тестирование показало, что система успешно обрабатывает до **172 321 записи** тестового датасета, но при этом важно предусмотреть механизмы восстановления и защиты от сбоев для реальной эксплуатации.

Использование PostgreSQL для хранения структурированных данных и Weaviate для векторных представлений (раздел 3.3) гарантирует отказоустойчивость благодаря репликации данных и возможности масштабирования. Микросервисная архитектура (раздел 3.2) позволяет легко заменять или восстанавливать конкретные компоненты системы без остановки всей инфраструктуры.

Кроме того, механизм шифрования данных (TLS 1.3 для передачи, AES-256 для хранения) обеспечивает безопасность при возможных сбоях или атаках, а регулярное логирование действий помогает выявлять причины отказов и предотвращать их повторение.

#### **3.3 Безопасность**

В данном разделе описываются требования к защите данных и программного обеспечения (ПО) от несанкционированного доступа, а также механизмы обеспечения безопасности при обработке конфиденциальной информации из резюме и вакансий.

**1. Защита данных:**

1.1. **Шифрование данных при передаче:**

* Все данные, передаваемые между компонентами системы или внешними платформами (например, hh.ru), должны быть защищены с использованием протокола **TLS 1.3** для обеспечения безопасной передачи через сети.
* Взаимодействие с базами данных (PostgreSQL, Weaviate) должно осуществляться через шифрованные соединения.

1.2. **Шифрование данных в покое:**

* Хранение данных в PostgreSQL и Weaviate должно производиться с использованием алгоритма **AES-256** , гарантирующего защиту от несанкционированного доступа.
* Резервные копии данных также должны быть зашифрованы тем же алгоритмом перед сохранением.

1.3. **Контроль доступа к данным:**

* Реализация ролевой модели управления доступом (**RBAC** ) для всех типов пользователей: рекрутеры, администраторы, научные сотрудники.
* Ограничение прав на чтение, запись и изменение данных в зависимости от роли пользователя. Например:
  + Рекрутеры могут просматривать и редактировать только свои вакансии и связанные с ними резюме.
  + Администраторы имеют права на управление системой и доступ ко всем данным для административных целей.

1.4. **Соблюдение нормативных требований:**

* ПО должно соответствовать стандартам защиты персональных данных: **GDPR** и **ФЗ-152 "О персональных данных"** .
* Конфиденциальная информация, такая как контактные данные кандидатов, должна быть обрабатываться исключительно в рамках согласия пользователя.

1.5. **Логирование действий:**

* Все операции с данными, включая загрузку резюме, создание/редактирование вакансий и запросы API, должны быть детально задокументированы в логах.
* Логи должны содержать следующую информацию: время события, тип операции, ID пользователя, статус выполнения.
* Логи должны быть защищены от несанкционированного доступа и храниться в зашифрованном виде.

**2. Защита программного обеспечения:**

2.1. **Защита от атак:**

* Реализация механизмов защиты от распространенных угроз, таких как SQL-инъекции, XSS-атаки и CSRF.
* Валидация входных данных для предотвращения ввода недопустимых значений.

2.2. **Авторизация и аутентификация:**

* Использование современных методов аутентификации, таких как OAuth 2.0 или JWT, для обеспечения безопасного доступа к API.
* Поддержка многофакторной аутентификации (**MFA** ) для пользователей с повышенными привилегиями (например, администраторов).

2.3. **Обновление безопасности:**

* Регулярное обновление зависимостей и библиотек для устранения известных уязвимостей.
* Проведение периодических проверок безопасности системы через инструменты статического анализа кода (например, SonarQube) и динамического тестирования (например, OWASP ZAP).

**3. Обоснование требований к безопасности:**

Требования к безопасности основаны на необходимости защищать конфиденциальную информацию кандидатов и работодателей, а также на соблюдении законодательных норм (раздел 2.4). Например, использование модели BERT для семантического анализа текстов резюме и вакансий требует надежного хранения и передачи данных, так как они могут содержать чувствительную информацию. Шифрование TLS 1.3 и AES-256 обеспечивает защиту от перехвата и несанкционированного доступа.

Реализация RBAC и MFA минимизирует риск утечки данных через человеческий фактор. Кроме того, регулярные проверки безопасности и обновления системы гарантируют её готовность к противодействию новым угрозам. Тестирование показало, что такие меры позволяют достичь высоких показателей безопасности при работе с большими объемами данных (до **172 321 записи** , раздел 4.3).

#### **3.4 Удобство использования**

В данном разделе описываются требования к интерфейсу и простоте использования программного обеспечения (ПО), разрабатываемого для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти требования направлены на обеспечение комфортного взаимодействия пользователей с системой, минимизацию временных затрат на обучение и повышение эффективности работы.

**1. Требования к пользовательскому интерфейсу:**

1.1. **Интуитивная навигация:**

* Интерфейс должен быть простым и понятным для всех типов пользователей, включая рекрутеров, администраторов и соискателей.
* Главное меню должно содержать основные функции: "Создание/редактирование вакансии", "Загрузка резюме", "Ранжирование кандидатов", "Настройки системы".

1.2. **Минимальное время обучения:**

* Пользователи должны иметь возможность начать работу с системой без необходимости длительного обучения.
* Предусмотреть подробную справочную документацию и видеоинструкции для быстрого освоения ключевых функций.

1.3. **Поддержка различных форматов данных:**

* Интерфейс должен поддерживать загрузку резюме и вакансий в форматах **PDF, DOCX, TXT** , а также предоставлять возможность экспорта результатов анализа в форматах **CSV** или **PDF** .
* Реализовать механизм drag-and-drop для загрузки файлов через графический интерфейс или кнопку "Обзор" для удобства использования.

1.4. **Гибкость настроек:**

* Администраторам должна быть предоставлена возможность изменять параметры анализа (например, веса для опыта работы, образования и навыков) через административный интерфейс.
* Рекрутеры могут фильтровать и сортировать результаты анализа по различным критериям (например, уровню соответствия, географическому положению).

1.5. **Детализированные уведомления:**

* Система должна отправлять уведомления рекрутерам о новых подходящих кандидатах и соискателям об их статусе (принято/отклонено) через email или интеграцию с корпоративными мессенджерами.
* Уведомления должны быть понятными и содержать детальную информацию о причинах принятия или отказа.

**2. Требования к простоте использования:**

2.1. **Автоматизация рутинных задач:**

* Все этапы первичной обработки резюме (токенизация, лемматизация, удаление стоп-слов) должны выполняться автоматически без участия пользователя.
* Автоматическая генерация рейтинга кандидатов с вероятностью соответствия вакансии [0, 1], где 1 означает максимальное соответствие.

2.2. **Прозрачность решений:**

* Для каждого кандидата система должна показывать конкретные причины ранжирования (например, совпадение навыков, опыт работы, образование).
* Информация должна быть представлена в виде списка или таблицы, что позволит рекрутерам быстро оценить результаты анализа.

2.3. **Производительность при больших объемах данных:**

* Время ответа системы на запрос не должно превышать **2 секунды** , даже при работе с пиковой нагрузкой (до **10 000 записей в час** ).
* Интерфейс должен оставаться отзывчивым и удобным для использования при массовой загрузке данных.

2.4. **Локализация интерфейса:**

* ПО должно поддерживать два языка интерфейса: **русский** и **английский** , чтобы удовлетворить потребности как российских, так и международных пользователей.

**3. Обоснование требований к удобству использования:**

Требования к удобству использования основаны на необходимости минимизировать участие человека в процессе первичного отбора кандидатов и обеспечить комфортное взаимодействие с системой для всех заинтересованных сторон. Например, использование модели BERT для семантического анализа текстов (раздел 1.4) требует сложной предобработки данных, но интерфейс скрывает эту техническую сложность, предоставляя рекрутерам готовый рейтинг кандидатов с детальной информацией о причинах ранжирования.

Выбор форматов данных (PDF, DOCX, TXT) и возможность экспорта результатов в CSV/PDF делает систему совместимой с существующими бизнес-процессами HR-отделов. Механизмы автоматической отправки уведомлений (раздел 2.3) снижают нагрузку на рекрутеров и повышают удовлетворенность соискателей благодаря своевременной обратной связи.

### **4. Интерфейсы**

#### **4.1 Пользовательские интерфейсы**

В данном разделе описываются пользовательские интерфейсы программного обеспечения, разрабатываемого для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти интерфейсы предназначены для взаимодействия с пользователями системы: рекрутерами, администраторами и соискателями.

**1. Интерфейс создания и управления вакансиями:**

* **Описание:** Интерфейс позволяет рекрутерам создавать, редактировать и удалять вакансии, указывая необходимые параметры, такие как название должности, ключевые навыки, опыт работы и образование.
* **Формат данных:** JSON или форма ввода через графический интерфейс.
* **Пример содержимого:** json  
  {  
   "title": "Инженер-программист",  
   "skills": ["Python", "SQL", "Machine Learning"],  
   "experience\_years": "3+",  
   "education": "Высшее техническое"  
   }
* **Цель:** Обеспечить удобное управление вакансиями для рекрутеров без необходимости прямого взаимодействия с базой данных.

**2. Интерфейс загрузки и анализа резюме:**

* **Описание:** Интерфейс позволяет соискателям загружать свои резюме в поддерживаемых форматах (PDF, DOCX, TXT) для дальнейшего анализа системой. Также рекрутеры могут импортировать резюме напрямую через API интеграции с HR-платформами.
* **Механизмы загрузки:** Drag-and-drop или кнопка "Обзор".
* **Поддерживаемые форматы:** PDF, DOCX, TXT.
* **Пример запроса через API:** http
* POST /api/resume/upload  
   Content-Type: application/json  
   {  
   "file\_url": "https://example.com/resume.pdf",  
   "vacancy\_id": "67890"  
   }
* **Цель:** Упростить процесс загрузки резюме для последующей обработки и минимизировать участие человека в первичной фильтрации данных.

**3. Интерфейс просмотра результатов анализа:**

* **Описание:** Интерфейс предоставляет рекрутерам ранжированный список кандидатов с вероятностью их соответствия требованиям вакансии. Для каждого кандидата показываются причины ранжирования (например, совпадение навыков, уровень опыта).
* **Формат представления:** Таблица или список с детализацией причин.
* **Цель:** Предоставить рекрутерам прозрачность процесса отбора и возможность быстрого принятия решений.

**4. Интерфейс настройки параметров модели:**

* **Описание:** Администраторы могут использовать этот интерфейс для изменения весовых коэффициентов параметров анализа (опыт работы, образование, навыки) и запуска fine-tuning модели BERT.
* **Формат данных:** Форма ввода числовых значений или JSON.
* **Пример содержимого:** json
* {  
   "parameters": {  
   "skills\_weight": 0.6,  
   "experience\_weight": 0.3,  
   "education\_weight": 0.1  
   }  
   }
* **Цель:** Обеспечить гибкость настройки модели под специфику компании или конкретной вакансии.

**5. Интерфейс отправки уведомлений:**

* **Описание:** Система автоматически отправляет уведомления рекрутерам и соискателям о статусе обработки резюме.
* **Типы уведомлений:** Email, SMS или push-уведомления через корпоративные мессенджеры.
* **Пример уведомления рекрутеру:** "Найдено 10 подходящих кандидатов для вакансии 'Инженер-программист'."
* **Пример уведомления соискателю:** "Ваше резюме рассмотрено для вакансии 'Инженер-программист'. Вероятность соответствия: 85%."
* **Цель:** Улучшить коммуникацию между рекрутерами и соискателями, обеспечив своевременную обратную связь.

**6. Интерфейс экспорта данных:**

* **Описание:** Интерфейс позволяет экспортировать результаты анализа резюме в форматах CSV или PDF для дальнейшего использования.
* **Форматы экспорта:** CSV для анализа в Excel, PDF для печати отчетов.
* **Пример содержимого CSV:**  
  resume\_id,vacancy\_id,similarity\_score,reasons\_for\_match  
   12345,67890,0.85,"Python, SQL"
* **Пример содержимого PDF:** Отчет содержит таблицу с рейтингом кандидатов и дополнительными деталями для каждой пары "резюме-вакансия".
* **Цель:** Обеспечить удобство использования результатов анализа в различных бизнес-процессах.

**7. Интерфейс логирования и мониторинга:**

* **Описание:** Интерфейс предназначен для администраторов и предоставляет информацию о событиях системы, таких как время обработки резюме, ошибки и успешные операции.
* **Формат данных:** JSON или текстовые логи.
* **Пример записи логов:** json
* {  
   "timestamp": "2024-01-01T12:00:00Z",  
   "event": "Resume processed",  
   "details": {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "status": "success",  
   "similarity\_score": 0.85  
   }  
   }
* **Цель:** Обеспечить контроль над работой системы и возможность анализа её производительности и точности.

**8. Локализация интерфейса:**

* **Описание:** Интерфейс должен быть доступен на **русском** и **английском** языках для удовлетворения потребностей как российских, так и международных пользователей.
* **Механизм реализации:** Использование файлов локализации (например, .po или .json) для хранения переводов интерфейса.
* **Цель:** Обеспечить комфортное использование системы для пользователей из разных регионов.

**9. Обоснование пользовательских интерфейсов:**

Выбор и функциональность пользовательских интерфейсов основаны на необходимости обеспечить простоту и эффективность взаимодействия всех заинтересованных сторон с системой. Например, интерфейс создания вакансий позволяет рекрутерам легко добавлять новые позиции, а механизм ранжирования кандидатов предоставляет им готовый список наиболее подходящих специалистов.

Результаты тестирования прототипа показывают, что интерфейс успешно справился с обработкой **172 321 записи** , демонстрируя высокую точность (accuracy = 0.76) и коэффициент корреляции Мэтьюса (MCC = 0.55). Это подтверждает корректность выбранного подхода к проектированию пользовательских интерфейсов.

Использование современных технологий, таких как RESTful API для интеграции с внешними платформами (hh.ru, LinkedIn Talent Solutions), и микросервисная архитектура гарантируют масштабируемость и удобство работы с большими объемами данных.

#### **4.2 Аппаратные интерфейсы**

В данном разделе описываются аппаратные интерфейсы, необходимые для взаимодействия программного обеспечения (ПО), разрабатываемого для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий, с аппаратным обеспечением. Эти требования направлены на обеспечение корректной работы системы в различных вычислительных средах.

**1. Взаимодействие с графическими процессорами (GPU):**

1.1. **Требования:**

* ПО должно быть способно взаимодействовать с графическими процессорами через CUDA или аналогичные API для ускорения вычислений моделей глубокого обучения, таких как BERT.
* GPU должен поддерживать архитектуру NVIDIA Compute Capability версии **не ниже 6.0** , что необходимо для эффективной работы с библиотеками машинного обучения (например, PyTorch, TensorFlow).

1.2. **Пример конфигурации:**

* Рекомендуемый GPU: NVIDIA Tesla V100 или GeForce RTX 3090.
* Минимальная память GPU: **16 ГБ VRAM** для обработки больших объемов данных (например, датасет объемом **172 321 запись** ).

1.3. **Цель:**

* Обеспечить быстрое выполнение операций векторизации текста и обучения модели BERT, снижая время обработки одного резюме до **нескольких секунд** .

**2. Взаимодействие с центральным процессором (CPU):**

2.1. **Требования:**

* CPU должен поддерживать многопоточность для выполнения параллельных задач предобработки данных и управления запросами.
* Минимальное количество ядер: **4** , рекомендуется **8+** для достижения оптимальной производительности при высоких нагрузках.

2.2. **Пример конфигурации:**

* Процессор: Intel Xeon E5-2680 v4 или AMD EPYC 7402.
* Тактовая частота: **2.4 ГГц и выше** для обеспечения стабильной работы микросервисов.

2.3. **Цель:**

* Поддержать работу системы при низкой загрузке GPU, например, при предобработке текстовых данных или управлении базами данных.

**3. Взаимодействие с оперативной памятью (RAM):**

3.1. **Требования:**

* Система должна иметь доступ к **минимум 16 ГБ RAM** , рекомендуется **32 ГБ RAM** для работы с большими объемами данных и поддержания нескольких экземпляров микросервисов.
* Оперативная память должна быть достаточной для хранения временных данных и параметров модели BERT во время выполнения операций.

3.2. **Обоснование:**

* При тестировании прототипа на датасете объемом **172 321 запись** была достигнута точность классификации accuracy = 0.76 и MCC = 0.55, что потребовало значительных ресурсов оперативной памяти.

3.3. **Цель:**

* Обеспечить достаточный запас памяти для бесперебойной работы системы при больших нагрузках.

**4. Взаимодействие с твердотельными накопителями (SSD):**

4.1. **Требования:**

* Для хранения предварительно обученных моделей BERT, временных данных и логов система требует SSD-накопитель с минимальным объемом **200 ГБ** .
* Желательный объем: **500 ГБ** для поддержания резервных копий и расширения хранилища данных.

4.2. **Пример конфигурации:**

* NVMe SSD с высокой скоростью чтения/записи для минимизации задержек при работе с PostgreSQL и Weaviate.

4.3. **Цель:**

* Обеспечить быстрый доступ к данным и ускорить процессы записи/чтения информации, особенно при работе с векторными представлениями текстов.

**5. Взаимодействие с сетевым оборудованием:**

5.1. **Требования:**

* Поддержка стабильного подключения к сети с минимальной скоростью **100 Мбит/с** для передачи данных между компонентами системы и внешними платформами (например, hh.ru).
* Использование протоколов шифрования TLS 1.3 для защиты каналов связи.

5.2. **Пример конфигурации:**

* Сервер с гигабитным сетевым адаптером для обеспечения быстрой передачи данных в случае использования облачной инфраструктуры.

5.3. **Цель:**

* Обеспечить надежную и безопасную передачу данных между серверами и клиентскими устройствами.

**6. Взаимодействие с системой охлаждения:**

6.1. **Требования:**

* Если система разворачивается на физических серверах, необходимо предусмотреть эффективную систему охлаждения для предотвращения перегрева оборудования при длительной работе.

6.2. **Пример конфигурации:**

* Использование серверов с жидкостным охлаждением или высокопроизводительными вентиляторами.

6.3. **Цель:**

* Обеспечить стабильную температуру оборудования в диапазоне **+15°C до +30°C** , что гарантирует долговременную работоспособность сервера.

**7. Обоснование аппаратных интерфейсов:**

Выбор аппаратных интерфейсов основан на необходимости обеспечить высокую производительность и отказоустойчивость системы. Например, использование GPU через CUDA позволяет значительно ускорить обработку векторных представлений текста, что необходимо для достижения целевого времени обработки одного резюме — **не более 2 секунд** .

Микросервисная архитектура (раздел 3.2) требует наличия достаточного количества оперативной памяти для поддержания нескольких экземпляров сервисов одновременно. SSD-накопители обеспечивают быстрый доступ к базам данных PostgreSQL и Weaviate, что важно для сохранения высокой производительности системы даже при больших объемах данных.

Сетевое оборудование с поддержкой TLS 1.3 гарантирует безопасность передачи данных, а также стабильность взаимодействия с внешними HR-платформами. Это делает систему готовой к реальной эксплуатации, где важны скорость, безопасность и масштабируемость.

#### **4.3 Программные интерфейсы**

В данном разделе описываются программные интерфейсы, обеспечивающие взаимодействие разрабатываемого программного обеспечения для автоматизации отбора кандидатов на основе обработки естественного языка с другими программными компонентами системы.

**1. Взаимодействие с векторной СУБД Weaviate:**

* **Описание:** Программное обеспечение использует API Weaviate для хранения и поиска векторных представлений текстовых данных резюме и вакансий.
* **Формат данных:** JSON для записи/чтения векторов и метаданных.
* **Пример запроса на запись:** http
* POST /weaviate/v1/objects  
   Content-Type: application/json  
   {  
   "class": "Resume",  
   "properties": {  
   "id": "12345",  
   "vector": [0.123, 0.456, ..., 0.789]  
   }  
   }
* **Цель:** Обеспечить эффективное хранение и поиск ближайших соседей в семантическом пространстве для точного сопоставления резюме с требованиями вакансий.

**2. Взаимодействие с реляционной СУБД PostgreSQL:**

* **Описание:** ПО осуществляет запросы к PostgreSQL для управления структурированными данными, такими как ID резюме, название должности, опыт работы, навыки и образование.
* **Формат данных:** SQL-запросы для записи/чтения данных.
* **Пример записи данных:** sql
* SELECT id, title, skills FROM vacancies WHERE id = '67890';
* **Цель:** Предоставить надежное хранение и быстрый доступ к структурированным данным для анализа и управления.

**3. Взаимодействие с моделями машинного обучения (BERT):**

* **Описание:** Программное обеспечение интегрируется с предобученной моделью BERT через соответствующие библиотеки (например, Hugging Face Transformers). Модель используется для преобразования текстов резюме и вакансий в контекстуальные векторные представления.
* **Формат данных:** Тексты в виде строк для входа, векторы размерностью 768 для выхода.
* **Пример использования модели:** python
* from transformers import BertTokenizer, BertModel  
    
   tokenizer = BertTokenizer.from\_pretrained('bert-base-multilingual-cased')  
   model = BertModel.from\_pretrained('bert-base-multilingual-cased')  
    
   input\_text = "Опыт работы с Python более 5 лет"  
   tokens = tokenizer(input\_text, return\_tensors="pt")  
   embeddings = model(\*\*tokens).last\_hidden\_state.mean(dim=1).detach().numpy()
* **Цель:** Обеспечить семантический анализ текстов резюме и вакансий для их дальнейшего сравнения.

**4. Взаимодействие с HR-платформами через RESTful API:**

* **Описание:** Программное обеспечение интегрируется с внешними платформами рекрутмента (например, hh.ru, LinkedIn Talent Solutions) через RESTful API для получения данных о резюме и отправки результатов анализа.
* **Формат данных:** JSON для обмена информацией.
* **Пример получения данных из платформы:** http
* POST /api/recruitment/results  
   Content-Type: application/json  
   {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "similarity\_score": 0.85,  
   "reasons": ["Python matches", "SQL required"]  
   }
* **Цель:** Упростить интеграцию с существующими системами рекрутмента и автоматизировать процесс обмена данными.

**5. Взаимодействие с механизмом логирования (ELK Stack):**

* **Описание:** ПО предоставляет информацию о событиях системы для сбора и анализа через ELK Stack (Elasticsearch, Logstash, Kibana). Это включает данные о времени выполнения операций, типах событий и статусах обработки.
* **Формат данных:** JSON для записи логов.
* **Пример записи лога:** json
* {  
   "timestamp": "2024-01-01T12:00:00Z",  
   "event": "Resume processed",  
   "details": {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890",  
   "status": "success",  
   "similarity\_score": 0.85  
   }  
   }
* **Цель:** Обеспечить контроль и мониторинг работы системы для выявления ошибок и оптимизации производительности.

**6. Взаимодействие с микросервисами через API-шлюз:**

* **Описание:** Система состоит из нескольких микросервисов, каждый из которых выполняет свою функциональную роль (обработка резюме, управление вакансиями, ранжирование кандидатов). Все сервисы взаимодействуют через API-шлюз, который маршрутизирует запросы между компонентами.
* **Формат данных:** JSON для передачи данных между сервисами.
* **Пример запроса к API-шлюзу:** POST /api/vacancy/match  
   Content-Type: application/json  
   {  
   "resume\_id": "12345",  
   "vacancy\_id": "67890"  
   }
* **Цель:** Обеспечить гибкость и масштабируемость архитектуры системы, позволяя каждому сервису развиваться независимо.

**7. Взаимодействие с системой безопасности (OAuth 2.0/JWT):**

* **Описание:** Для обеспечения безопасного доступа к API и защите данных используются протоколы OAuth 2.0 или JWT (JSON Web Token). Это позволяет управлять авторизацией пользователей и ограничивать доступ к конфиденциальной информации.
* **Формат данных:** JSON Web Token для аутентификации.
* **Пример токена:** text
* eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...
* **Цель:** Защитить данные пользователей и обеспечить соответствие требованиям GDPR и ФЗ-152 "О персональных данных".

**8. Взаимодействие с очередями сообщений (RabbitMQ/Kafka):**

* **Описание:** Для управления пиковыми нагрузками и обработки больших объемов данных используются очереди сообщений, такие как RabbitMQ или Kafka. Это позволяет временно помещать запросы в очередь при высокой загрузке системы.
* **Формат данных:** Сообщения в формате JSON.
* **Пример сообщения в очереди:** {  
   "action": "process\_resume",  
   "data": {  
   "file\_url": "https://example.com/resume.pdf",  
   "vacancy\_id": "67890"  
   }  
   }
* **Цель:** Обеспечить отказоустойчивость и стабильность работы системы при больших нагрузках.

**9. Взаимодействие с библиотеками NLP:**

* **Описание:** Программное обеспечение использует библиотеки для обработки естественного языка, такие как spaCy и NLTK, для предобработки текстовых данных (токенизация, удаление стоп-слов, нормализация).
* **Формат данных:** Тексты в виде строк для входа, обработанные тексты или токены для выхода.
* **Пример использования spaCy для токенизации:** python
* import spacy  
    
   nlp = spacy.load("ru\_core\_news\_sm")  
   doc = nlp("Опыт работы с Python более 5 лет")  
   tokens = [token.text for token in doc if not token.is\_stop]
* **Цель:** Подготовить текстовые данные для последующей векторизации и анализа, минимизируя влияние шума и нестандартизированных форматов.

**10. Взаимодействие с библиотеками машинного обучения (PyTorch/TensorFlow):**

* **Описание:** Для реализации алгоритмов машинного обучения используются библиотеки PyTorch или TensorFlow. Эти инструменты применяются для fine-tuning модели BERT и выполнения других задач анализа данных.
* **Формат данных:** Тензоры для входа, числовые значения или векторы для выхода.
* **Пример использования PyTorch для обучения модели:** python
* import torch  
   from torch.optim import AdamW  
    
   optimizer = AdamW(model.parameters(), lr=2e-5)  
   loss\_fn = torch.nn.CrossEntropyLoss()  
    
   outputs = model(inputs)  
   loss = loss\_fn(outputs, labels)  
   loss.backward()  
   optimizer.step()
* **Цель:** Обеспечить высокую точность анализа текстов резюме и вакансий через обучение моделей машинного обучения.

**11. Взаимодействие с системой уведомлений:**

* **Описание:** ПО поддерживает отправку уведомлений рекрутерам и соискателям через email или корпоративные мессенджеры.
* **Формат данных:** JSON для передачи содержимого уведомления.
* **Пример отправки уведомления:**{  
   "to": "recruiter@example.com",  
   "subject": "Найдено подходящих кандидатов",  
   "body": "Для вакансии 'Python Developer' найдено 10 подходящих кандидатов."  
   }
* **Цель:** Обеспечить своевременную обратную связь всем участникам процесса рекрутмента.

**12. Обоснование программных интерфейсов:**

Выбор программных интерфейсов основан на необходимости обеспечить эффективное взаимодействие всех компонентов системы для достижения целей автоматизации отбора кандидатов. Например, использование RESTful API для интеграции с HR-платформами гарантирует возможность работы с различными источниками данных, такими как hh.ru или LinkedIn Talent Solutions.

Микросервисная архитектура (раздел 3.2) и интеграция с Weaviate для хранения векторных данных позволяют обрабатывать большие объемы данных (до **172 321 записи** ) с высокой точностью (accuracy = 0.76, MCC = 0.55). Библиотеки spaCy и NLTK обеспечивают качественную предобработку текста, что важно для последующего анализа моделями типа BERT.

Использование ELK Stack для логирования действий помогает администраторам контролировать работу системы, а механизмы очередей сообщений (RabbitMQ/Kafka) гарантируют стабильность при пиковых нагрузках.

### **5. Ограничения**

#### **5.1 Технические ограничения**

В данном разделе описываются технические ограничения, связанные с используемыми технологиями и инфраструктурой программного обеспечения для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий.

**1. Ограничения по вычислительным мощностям:**

1.1. **Требования к GPU:**

* Для корректной работы модели BERT необходим доступ к графическим процессорам (GPU) с поддержкой CUDA. Это связано с высокими вычислительными затратами при обработке семантических данных больших объемов. Минимальные требования: NVIDIA GPU с архитектурой Compute Capability версии **6.0 и выше** .

1.2. **Процессор (CPU):**

* Система должна работать на многопоточных процессорах с минимальным количеством ядер **4** , рекомендуется **8+** для параллельной обработки запросов.

1.3. **Оперативная память (RAM):**

* Объем оперативной памяти должен составлять **не менее 16 ГБ** , рекомендуется **32 ГБ** для работы с большими датасетами (например, объемом **172 321 запись** ).

**2. Ограничения по использованию моделей машинного обучения:**

2.1. **BERT:**

* Модель BERT была выбрана для векторизации текстовых данных, однако её использование требует значительных вычислительных ресурсов и времени на обучение. Процесс fine-tuning может занять до **37 минут на одну эпоху** при работе с GPU, что накладывает ограничения на частоту переподготовки модели.

2.2. **Fine-tuning модели:**

* Количество эпох обучения ограничивается **4 эпохами** , так как дальнейшее обучение может привести к переобучению модели, как показали эксперименты (раздел 4.4).

**3. Ограничения по базам данных:**

3.1. **PostgreSQL:**

* PostgreSQL используется для хранения структурированных данных о резюме и вакансиях. Однако его производительность может снижаться при работе с миллионами записей, что потребует оптимизации или перехода к более масштабируемым решениям, таким как CockroachDB.

3.2. **Weaviate:**

* Weaviate применяется для хранения векторных представлений текстов (embeddings). Размер векторов составляет **768 элементов** , что ограничивает возможности использования других размерностей векторов без дополнительной настройки.

3.3. **MongoDB:**

* MongoDB используется для хранения полуструктурированных JSON-данных. Однако её применение ограничено задачами, где требуется гибкость схемы данных, и не рекомендуется для анализа векторных данных.

**4. Ограничения по форматам данных:**

4.1. **Поддерживаемые форматы:**

* Программное обеспечение принимает только текстовые файлы в форматах **PDF, DOCX, TXT** для загрузки резюме и вакансий. Другие форматы (например, HTML, JPG) требуют предварительной конвертации.

4.2. **Лимиты по размеру файла:**

* Максимальный размер загружаемого файла составляет **5 МБ** , чтобы минимизировать время обработки и нагрузку на систему.

**5. Ограничения по API-взаимодействию:**

5.1. **Внешние платформы:**

* Интеграция с внешними HR-платформами (например, hh.ru) осуществляется через их API, которые имеют ограничения по количеству запросов в секунду (QPS). Например, hh.ru ограничивает число запросов до **50 в минуту** для бесплатных учетных записей.

5.2. **RESTful API:**

* Все внутренние компоненты системы взаимодействуют через RESTful API, что требует стандартизации форматов данных (JSON) и соблюдения принципов масштабирования.

**6. Ограничения по шифрованию и безопасности:**

6.1. **Шифрование данных:**

* Все данные должны передаваться через защищенные каналы связи с использованием протокола **TLS 1.3** , а храниться — с применением алгоритма **AES-256** . Это увеличивает нагрузку на серверную инфраструктуру, особенно при работе с больших объемах данных.

6.2. **Соблюдение нормативных требований:**

* ПО должно соответствовать стандартам защиты персональных данных: **GDPR** и **ФЗ-152 "О персональных данных"** . Это ограничивает способы использования и хранения данных, таких как логирование детальной информации о пользователях.

**7. Ограничения по языковой специфике:**

7.1. **Поддерживаемые языки:**

* Модель BERT поддерживает анализ текстов на **русском** и **английском** языках. Использование других языков потребует дополнительного обучения модели или выбора другой предобученной версии.

**8. Ограничения по интеграции микросервисов:**

8.1. **Контейнеризация:**

* Микросервисы разворачиваются в контейнерах Docker с управлением через Kubernetes. Это требует наличия соответствующей инфраструктуры для оркестрации.

8.2. **Межсервисное взаимодействие:**

* Взаимодействие между микросервисами происходит через очередь сообщений (например, RabbitMQ или Kafka). При высоких нагрузках возможны задержки в обработке сообщений, что необходимо учитывать при проектировании.

**9. Обоснование технических ограничений:**

Выбор и установка технических ограничений основаны на необходимости обеспечить высокую производительность, безопасность и совместимость системы с реальными условиями эксплуатации. Например, модель BERT требует значительных вычислительных ресурсов для обработки больших объемов данных, но демонстрирует высокую точность (accuracy = 0.76, MCC = 0.55) на тестовом датасете объемом **172 321 запись** (раздел 4.3).

Использование PostgreSQL для структурированных данных и Weaviate для векторных представлений позволяет эффективно решать задачи анализа, но требует оптимизации при работе с миллионами записей. RESTful API обеспечивает удобство интеграции с внешними платформами, хотя и накладывает ограничения на форматы данных и методы взаимодействия.

Кроме того, соблюдение требований GDPR и ФЗ-152 гарантирует защиту персональных данных, что особенно важно при работе с резюме и вакансиями.

#### **5.2 Ограничения по времени**

В данном разделе описываются требования к времени выполнения операций программного обеспечения для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти ограничения направлены на обеспечение стабильной работы системы при больших нагрузках, а также на повышение удобства использования для всех заинтересованных сторон.

**1. Время обработки одного резюме:** Программное обеспечение должно обрабатывать одно резюме за **не более 2 секунды** , включая все этапы: предобработку текста, векторизацию с использованием модели BERT и семантическое сопоставление с требованиями вакансии.

* **Обоснование:** На тестовых данных объемом **172 321 запись** достигнутые метрики точности (accuracy = 0.76) и коэффициента корреляции Мэтьюса (MCC = 0.55) показывают возможность соблюдения данного ограничения. Это время является приемлемым для пользователей и позволяет эффективно работать с большими объемами данных.

**2. Общее время обработки пакетов данных:** Система должна быть способна обработать до **10 000 записей в час** , сохраняя стабильность времени выполнения каждой операции.

* **Обоснование:** Для HR-специалистов важно получать результаты анализа в максимально короткие сроки, особенно при работе с массовыми рассылками резюме. Экспериментальные данные подтверждают, что система может справиться с такими объемами данных, используя микросервисную архитектуру и параллельную обработку запросов через RESTful API.

**3. Время обучения модели BERT:** Процесс fine-tuning модели BERT не должен занимать более **37 минут на одну эпоху** при использовании GPU. При необходимости переподготовки модели общее время обучения должно составлять **не более 2 часов** для четырех эпох.

* **Обоснование:** Результаты экспериментов с моделью BERT показывают, что обучение одной эпохи занимает около **37 минут** (раздел 4.4). Установленное ограничение по времени обучения гарантирует возможность своевременной адаптации модели под новые данные без значительных задержек в работе системы.

**4. Время восстановления после отказа:** При возникновении сбоев или отказов компонентов система должна быть восстановлена в рабочее состояние за **не более 15 минут** .

* **Обоснование:** Быстрое восстановление критично для непрерывной работы сервиса, особенно при его интеграции с внешними платформами рекрутмента. Использование механизмов резервного копирования баз данных (PostgreSQL, Weaviate) и контейнеризации через Docker/Kubernetes позволяет минимизировать время простоя.

**5. Время ответа API:** Все запросы через RESTful API должны обрабатываться системой за **не более 2 секунды** , независимо от типа операции (загрузка резюме, создание/редактирование вакансии, получение результатов анализа).

* **Пример запроса:** http
* POST /api/resume/upload  
   Content-Type: application/json  
   {  
   "file\_url": "https://example.com/resume.pdf",  
   "vacancy\_id": "67890"  
   }
* **Обоснование:** Скорость ответа API важна для комфортного взаимодействия пользователей и интеграции с внешними платформами, такими как hh.ru. Тестирование показало, что система успешно справляется с этим требованием даже при высоких нагрузках.

**6. Время загрузки данных из внешних систем:** Загрузка данных о резюме и вакансиях через API внешних платформ (например, hh.ru) должна занимать **не более 10 минут** для набора из **10 000 записей** .

* **Обоснование:** Ограничения по количеству запросов в секунду (QPS) у внешних платформ (например, hh.ru ограничивает количество запросов до **50 в минуту** ) требуют оптимизации процесса загрузки данных. Установленное ограничение гарантирует своевременную доставку информации в систему.

**7. Время генерации отчетов:** Система должна формировать отчеты о результате анализа резюме в форматах CSV или PDF за **не более 5 минут** для набора из **1 000 записей** .

* **Обоснование:** HR-специалисты часто требуют быстрого предоставления результатов анализа для принятия решений. Тестирование показывает, что система способна сформировать такие отчеты в указанные временные рамки.

**8. Время мониторинга и логирования:** Логирование действий системы и сбор метрик производительности через ELK Stack или аналогичные инструменты должны выполняться в реальном времени, без задержек более чем **1 секунда** на каждое событие.

* **Обоснование:** Непрерывный мониторинг необходим для выявления проблем и предотвращения сбоев. Короткие временные интервалы между записями логов обеспечивают оперативное реагирование на инциденты.

**9. Время создания резервных копий:** Резервное копирование данных из PostgreSQL и Weaviate должно выполняться ежедневно за **не более 30 минут** для текущего объема данных (до **172 321 записи** ).

* **Обоснование:** Регулярное резервное копирование гарантирует защиту данных от потери. Установленное ограничение по времени позволяет выполнять эту операцию без значительного влияния на работу основной системы.

**10. Время первичной загрузки системы:** При запуске или перезапуске компонентов системы через Docker/Kubernetes время их становления в рабочее состояние должно составлять **не более 2 минут** .

* **Обоснование:** Быстрая загрузка компонентов критически важна для обеспечения минимального простоя системы при техническом обслуживании или восстановлении после сбоев.

**11. Время обновления данных вакансий:** При изменении параметров вакансии система должна пересчитать соответствие всех связанных резюме за **не более 5 минут** для набора из **100 записей** .

* **Обоснование:** Частые изменения требований к вакансиям требуют оперативного обновления рейтингов кандидатов. Установленное ограничение по времени гарантирует, что рекрутеры всегда имеют актуальные данные.

**12. Обоснование ограничений по времени:**

Требования к времени выполнения операций основаны на анализе реальных данных (раздел 4.3) и выбранной архитектуре системы (раздел 3.2). Например, использование модели BERT для векторизации текстов требует значительных вычислительных ресурсов, но тестирование показало, что при оптимизации она способна обрабатывать один документ за **2 секунды** .

Микросервисная архитектура и распределенная инфраструктура позволяют масштабировать систему для обработки больших объемов данных (до **10 000 записей в час** ) без увеличения времени ответа более чем на **20%** . Интеграция с внешними платформами через RESTful API гарантирует стабильную передачу данных даже при ограничениях QPS (например, hh.ru — **50 запросов в минуту** ).

Кроме того, регулярное резервное копирование данных (ежедневно для PostgreSQL, еженедельно для Weaviate) и шифрование каналов связи (TLS 1.3) обеспечивают безопасность и надежность системы, не влияя на её производительность.

# OpsCon (Operation Concept)

### **1. Введение**

#### **1.1 Цель документа**

Целью данного документа является описание того, как система должна использоваться в реальных условиях, с точки зрения пользователей и операторов.

#### **1.2 Область применения**

Система должна работать в процессе рекрутмента для автоматизации этапов первичного отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Определены следние границы системы и её роль в операционной среде:

1. Система должна взаимодействовать с текстовыми данными резюме и вакансий, представленными в форматах PDF, DOCX, TXT.
2. Система должна поддерживать анализ данных только на русском и английском языках, используя модель BERT для семантического анализа текста.
3. Система должна интегрироваться с внешними платформами рекрутмента через RESTful API для получения данных о резюме и отправки результатов их анализа.
4. Система должна использовать PostgreSQL для хранения структурированных данных о резюме и вакансиях, таких как ID, название должности, ключевые навыки, опыт работы и образование.
5. Система должна применять Weaviate для управления векторными представлениями текстовых данных резюме и вакансий, что позволит эффективно находить ближайших соседей в семантическом пространстве.
6. Система должна предоставлять рекрутерам ранжированный список кандидатов с вероятностью соответствия [0, 1], где значение близкое к 1 указывает на высокую релевантность.
7. Система должна автоматически анализировать большие объемы данных, минимизируя участие человека в процессе фильтрации резюме.
8. Система должна обеспечивать прозрачность решений, объясняя причины ранжирования каждого кандидата.
9. Система должна быть готова к работе с миллионами записей без потери производительности, сохраняя время обработки одного резюме не более 2 секунд.

Таким образом, система должна занимать ключевое место в процессе рекрутмента, автоматизируя трудоемкие задачи по фильтрации и анализу резюме, а также предоставляя рекрутерам качественные и объективные результаты. Это позволит значительно повысить эффективность процесса найма и сократить временные затраты на первичный отбор кандидатов.

#### **1.3 Определения**

В данном разделе приводятся термины и сокращения, используемые в документе для описания того, как система должна использоваться в реальных условиях с точки зрения пользователей и операторов.

**Термины:**

1. **AS IS** — Система должна отражать текущее состояние процесса рекрутмента, характеризующееся ручной фильтрацией резюме и ограниченной автоматизацией.
2. **TO BE** — Система должна обеспечивать целевое состояние процесса рекрутмента, включающее автоматизацию первичного анализа резюме и вакансий на основе обработки естественного языка.
3. **BERT (Bidirectional Encoder Representations from Transformers)** — Система должна использовать модель BERT для преобразования текстовых данных резюме и вакансий в контекстуальные векторные представления.
4. **ContinueWork** — Система должна называться ContinueWork и предоставлять функционал автоматизации отбора кандидатов.
5. **Docker** — Система должна разворачиваться в изолированной среде через Docker для обеспечения стабильности работы.
6. **Embeddings** — Система должна преобразовывать текстовые данные в векторные представления с помощью моделей обработки естественного языка.
7. **Fine-tuning** — Система должна выполнять адаптацию предобученной модели BERT под задачу классификации резюме и вакансий.
8. **GPU (Graphics Processing Unit)** — Для ускорения вычислений система должна использовать графические процессоры при работе с алгоритмами машинного обучения.
9. **HNSW (Hierarchical Navigable Small World Graphs)** — Система должна применять алгоритмы HNSW для поиска ближайших соседей в векторном пространстве при анализе соответствия резюме и вакансий.
10. **MCC (Matthews Correlation Coefficient)** — Система должна рассчитывать коэффициент корреляции Мэтьюса для оценки качества классификации резюме.
11. **NLP (Natural Language Processing)** — Система должна использовать технологии обработки естественного языка для анализа текстовой информации из резюме и вакансий.
12. **PostgreSQL** — Система должна хранить структурированные данные о резюме и вакансиях в PostgreSQL для обеспечения надежности и производительности.
13. **Resume (Резюме)** — Система должна принимать файлы резюме в форматах PDF, DOCX, TXT для дальнейшего анализа.
14. **Self-Attention** — Система должна использовать механизм само-внимания (Self-Attention) для анализа контекста слов в текстах резюме и вакансий.
15. **Validation Loss** — Система должна рассчитывать потери функции ошибки на валидационном наборе данных во время обучения модели.
16. **Vacancy (Вакансия)** — Система должна обрабатывать описания вакансий, предоставленные в формате текста, для их сравнения с резюме.
17. **Vector Database** — Система должна использовать векторную базу данных для хранения и анализа векторных представлений текстов.
18. **Weaviate** — Система должна интегрироваться с Weaviate для управления векторными данными резюме и вакансий.

**Сокращения:**

1. **API** — Application Programming Interface (интерфейс программирования приложений).
2. **AS** — Current State (текущее состояние).
3. **BERT** — Bidirectional Encoder Representations from Transformers.
4. **CSV** — Comma-Separated Values (формат файла для хранения табличных данных).
5. **GPU** — Graphics Processing Unit (графический процессор).
6. **JSON** — JavaScript Object Notation (формат обмена данными).
7. **MVP** — Minimum Viable Product (минимально жизнеспособный продукт).
8. **NLP** — Natural Language Processing (обработка естественного языка).
9. **REST** — Representational State Transfer (архитектурный стиль взаимодействия между компонентами системы).
10. **SQL** — Structured Query Language (язык запросов для работы с базами данных).
11. **TLS** — Transport Layer Security (протокол защиты передачи данных).
12. **TO** — Target State (целевое состояние).

### **2. Обзор системы**

#### **2.1 Цель системы**

Система должна создаваться для автоматизации этапов первичного отбора кандидатов в процессе рекрутмента, основываясь на обработке естественного языка из текстовых данных резюме и вакансий. Система должна решать следующие задачи:

1. Система должна сокращать время фильтрации большого объема резюме, предоставляя рекрутерам ранжированный список кандидатов по уровню их соответствия требованиям вакансии.
2. Система должна минимизировать влияние человеческого фактора при анализе резюме за счет использования точных методов семантического анализа, таких как модель BERT.
3. Система должна обеспечивать высокую точность сопоставления навыков кандидатов с требованиями вакансий, достигая метрики accuracy не менее 0.76 и коэффициента корреляции Мэтьюса (MCC) не менее 0.55.
4. Система должна поддерживать интеграцию с внешними HR-платформами через RESTful API, что позволит автоматически загружать данные о резюме и вакансиях.
5. Система должна предоставлять детализированную информацию о причинах ранжирования каждого кандидата, чтобы рекрутер мог объективно оценить результаты анализа.
6. Система должна учитывать контекст слов в текстах резюме и вакансий, что позволит различать значения многозначных терминов и синонимов для более качественного анализа.
7. Система должна отправлять уведомления рекрутерам о подходящих кандидатах и соискателям об их статусе через email или интеграцию с мессенджерами.
8. Система должна работать с данными только на русском и английском языках, используя предобученные модели NLP для этих языков.

#### **2.2 Контекст системы**

Система должна функционировать в определенном окружении, взаимодействуя с внешними системами используя доступ к её функционалу через удобные интерфейсы. Ниже описан контекст использования системы, включая внешние компоненты и их взаимосвязь с разрабатываемым решением.

Внешние системы, с которыми система должна взаимодействовать:

* **HR-платформы** : Платформы для публикации вакансий и сбора резюме от соискателей.
* **Базы данных** : PostgreSQL для структурированных данных и Weaviate для векторных представлений.
* **Механизмы безопасности** : Инструменты для шифрования данных и управления доступом пользователей.
* **Каналы связи** : Email-серверы или корпоративные мессенджеры для отправки уведомлений пользователям.

#### **2.3 Основные функции**

Система должна предоставлять следующие ключевые функции для автоматизации процесса отбора кандидатов на основе обработки естественного языка из резюме и вакансий:

1. Система должна автоматически загружать данные о резюме и вакансиях через интеграцию с внешними платформами рекрутмента, такими как hh.ru или LinkedIn Talent Solutions.
2. Система должна преобразовывать текстовые данные резюме и вакансий в структурированные параметры, такие как опыт работы, образование и ключевые навыки, используя методы предобработки текста (токенизация, удаление стоп-слов, лемматизация).
3. Система должна создавать контекстуальные векторные представления текстовых данных с помощью модели BERT для их дальнейшего анализа.
4. Система должна сравнивать векторные представления резюме и вакансий, вычисляя их семантическое соответствие с использованием метрики косинусного расстояния.
5. Система должна формировать ранжированный список кандидатов по уровню соответствия требованиям вакансии, представляя результаты в виде вероятности [0, 1].
6. Система должна отправлять уведомления рекрутерам о подходящих кандидатах и соискателям об их статусе через email или интеграцию с мессенджерами.
7. Система должна обеспечивать возможность обновления данных вакансий и пересчета соответствия ранее загруженных резюме при изменении требований.
8. Система должна сохранять исторические данные о прошлых откликах и решениях для анализа эффективности отбора кандидатов.
9. Система должна предоставлять детальную информацию о причинах ранжирования каждого кандидата, чтобы пользователи могли объективно оценить результаты анализа.

#### **2.4 Пользователи и операторы**

Система должна поддерживать взаимодействие с различными ролями пользователей и операторов, каждый из которых играет ключевую роль в процессе автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Ниже описаны основные роли и их взаимодействие с системой.

**1. Рекрутер:** Рекрутер является главным пользователем системы и отвечает за создание, управление и анализ вакансий.

* Система должна предоставлять рекрутеру возможность создания и редактирования вакансий через удобный интерфейс.
* Система должна автоматически загружать резюме соискателей через интеграцию с внешними платформами рекрутмента (например, hh.ru).
* Система должна формировать ранжированный список кандидатов для каждой вакансии и представлять его рекрутеру с вероятностью соответствия [0, 1].
* Система должна объяснять причины ранжирования каждого кандидата, чтобы рекрутер мог принять объективное решение.

**2. Соискатель:** Соискатель представляет собой конечного пользователя системы, который отправляет свое резюме на вакансии.

* Система должна принимать файлы резюме в форматах **PDF** , **DOCX** , **TXT** через удобный интерфейс или API.
* Система должна уведомлять соискателя о статусе его резюме (принято/отклонено) через email или интеграцию с мессенджерами.
* Система должна обеспечивать прозрачность процесса отбора, предоставляя соискателю информацию о причинах принятия или отклонения его резюме.

**3. Администратор:** Администратор отвечает за настройку и поддержание работы системы, а также за обеспечение безопасности данных.

* Система должна предоставлять администратору возможность управления параметрами модели BERT (например, fine-tuning для учета специфики компании).
* Система должна позволять администратору контролировать доступ к данным через механизм ролевой модели управления доступом (RBAC).
* Система должна обеспечивать регулярное резервное копирование данных в PostgreSQL и Weaviate, а также восстановление этих данных при необходимости.
* Система должна использовать шифрование TLS 1.3 для передачи данных и AES-256 для их хранения, чтобы администратор мог гарантировать безопасность персональных данных пользователей.

**4. Научный сотрудник/разработчик:** Научный сотрудник или разработчик отвечает за доработку алгоритмов и тестирование системы для повышения её точности.

* Система должна предоставлять научному сотруднику доступ к логам событий для анализа производительности и качества работы алгоритмов.
* Система должна позволять научному сотруднику проводить эксперименты с различными моделями embeddings (например, Word2Vec, GloVe) для сравнения результатов.
* Система должна генерировать отчеты о метриках классификации (accuracy, MCC) для научного сотрудника после каждого этапа обучения или тестирования.

**5. Оператор технической поддержки:** Оператор технической поддержки должен обеспечивать работу системы в реальных условиях и решать возникающие проблемы.

* Система должна информировать оператора о сбоях или нештатных ситуациях через детальные логи и уведомления.
* Система должна предоставлять оператору возможность перезапуска микросервисов через Docker/Kubernetes для быстрого восстановления работоспособности.
* Система должна иметь механизм Circuit Breaker для защиты от множественных неудачных попыток взаимодействия с внешними платформами, чтобы оператор мог легко устранять такие проблемы.

**Взаимодействие между ролями:**

* Рекрутер и соискатель должны взаимодействовать через графический интерфейс или API, где рекрутер создает вакансии, а соискатель загружает свои резюме.
* Администратор и научный сотрудник должны иметь доступ к внутренним компонентам системы для настройки и анализа работы.
* Оператор технической поддержки должен координировать действия по устранению сбоев и оптимизации производительности системы, взаимодействуя с администратором и рекрутером для минимизации времени простоя.

### 3. Операционные сценарии

#### **3.1 Нормальные сценарии**

В данном разделе описываются типичные сценарии использования системы в штатных условиях, с точки зрения пользователей и операторов. Эти сценарии демонстрируют основные процессы взаимодействия с системой для достижения её целей.

1. **Создание и публикация вакансии:** Система должна предоставлять рекрутеру возможность создавать новую вакансию через интерфейс или API. После создания система должна преобразовать текстовое описание вакансии в структурированные данные (название должности, ключевые навыки, опыт работы) и их векторное представление с помощью модели BERT. Затем система должна сохранить эти данные в PostgreSQL для структурированных параметров и в Weaviate для векторного представления.
2. **Загрузка резюме:** Система должна поддерживать загрузку файлов резюме в форматах PDF, DOCX, TXT. После загрузки система должна автоматически извлекать текстовые данные из файла, проводить предобработку (токенизацию, удаление стоп-слов, нормализацию) и преобразовывать текст в структурированные данные (опыт работы, навыки, образование) и векторное представление. Эти данные должны быть сохранены в соответствующих базах данных: PostgreSQL для структурированных данных и Weaviate для векторных представлений.
3. **Анализ соответствия кандидатов:** Для каждой загруженной пары "резюме-вакансия" система должна использовать алгоритм косинусного расстояния для вычисления степени семантического соответствия между векторными представлениями резюме и вакансии. Результат анализа должен быть представлен числовым значением от 0 до 1, где значение близкое к 1 указывает на высокое соответствие.
4. **Ранжирование кандидатов:** Система должна формировать ранжированный список кандидатов для конкретной вакансии на основе результатов анализа. Этот список должен быть упорядочен по убыванию вероятности соответствия. Для каждого кандидата система должна предоставлять детальную информацию о причинах ранжирования (например, совпадение опыта работы, образование, ключевые навыки).
5. **Отправка уведомлений рекрутерам:** При обнаружении подходящих кандидатов система должна отправлять уведомления рекрутерам через email или интеграцию с корпоративными мессенджерами. Уведомления должны содержать ссылку на подробный анализ кандидата и его вероятность соответствия вакансии.
6. **Обновление данных вакансий:** Если рекрутер изменяет требования к вакансии, система должна автоматически пересчитать степень соответствия всех ранее загруженных резюме для этой вакансии. При этом система должна использовать актуальное описание вакансии для генерации новых векторных представлений и их сравнения.
7. **Интеграция с внешними платформами:** Система должна автоматически получать новые резюме через интеграцию с HR-платформами (например, hh.ru, LinkedIn Talent Solutions). При этом система должна соблюдать ограничения API платформы по количеству запросов в секунду (QPS). Полученные резюме должны быть обработаны в соответствии с вышеописанными сценариями.
8. **Экспорт результатов анализа:** По запросу рекрутера система должна экспортировать результаты анализа в формате CSV или PDF. Экспорт должен содержать ID резюме, ID вакансии, вероятность соответствия и детали анализа (например, совпадение навыков, уровень образования).
9. **Предоставление обратной связи соискателям:** Для каждого резюме, которое было проанализировано, система должна отправлять соискателю уведомление о результате обработки (принято/отклонено) через email или интеграцию с платформами размещения резюме. В случае отказа система должна объяснить причину (например, недостаточный опыт работы или несоответствие ключевых навыков).
10. **Мониторинг состояния системы:** Система должна обеспечивать постоянный мониторинг своих процессов, включая время обработки запросов, точность анализа и количество обработанных резюме. Администратор должен иметь доступ к детальным логам событий для контроля работоспособности системы.

#### **3.2 Аварийные сценарии**

В данном разделе описываются действия системы в нештатных ситуациях, таких как сбои компонентов, отказ оборудования или проблемы с интеграцией внешних систем. Система должна быть подготовлена к обработке аварийных ситуаций для обеспечения минимального влияния на рабочие процессы.

**1. Отказ GPU при обработке данных:** Система должна переключаться на использование CPU для выполнения операций анализа текстовых данных и вычисления эмбеддингов, если доступ к GPU временно отсутствует. Это гарантирует продолжение работы сервиса без полной остановки процессов.

**2. Недоступность внешних HR-платформ:** Если внешние платформы рекрутмента (например, hh.ru) становятся недоступными, система должна сохранять загруженные ранее резюме и вакансии в локальных базах данных PostgreSQL и Weaviate. После восстановления доступности платформ система должна автоматически повторить попытки получения новых данных через REST API.

**3. Превышение ограничений по времени обработки запросов:** При превышении установленного времени обработки одного запроса (не более 2 секунд), система должна помещать запрос в очередь для последующего выполнения. Очередь должна быть реализована с использованием инструментов, таких как RabbitMQ или Kafka, чтобы обеспечить отказоустойчивость при пиковых нагрузках.

**4. Ошибка загрузки файла резюме:** Если пользователь загружает файл резюме в формате, который система не поддерживает (например, HTML, JPG), система должна уведомлять пользователя об ошибке через интерфейс или email. Для поддерживаемых форматов (PDF, DOCX, TXT) система должна проверять целостность данных и сигнализировать об ошибках, связанных с поврежденным содержимым файла.

**5. Переобучение модели BERT:** При проведении fine-tuning модели BERT, если происходит переобучение (validation loss начинает расти), система должна автоматически прекратить обучение после второй эпохи. Это позволит избежать снижения точности модели и гарантировать её работоспособность.

**6. Сбой микросервиса:** Если один из микросервисов системы становится недоступным, система должна использовать механизм Circuit Breaker для защиты от множественных неудачных запросов к этому сервису. Кроме того, система должна автоматически перенаправлять запросы на резервные экземпляры микросервисов через Kubernetes, что обеспечит стабильную работу системы даже при частичном отказе компонентов.

**7. Потеря связи с Weaviate:** Если возникает проблема с доступностью векторной базы данных Weaviate, система должна временно хранить данные о векторных представлениях в локальной памяти или использовать резервное хранилище MongoDB для сохранения контекстуальной информации до восстановления связи с Weaviate.

**8. Ошибки шифрования данных:** При возникновении ошибок шифрования данных (TLS 1.3 для передачи или AES-256 для хранения), система должна немедленно блокировать доступ к данным и отправлять уведомление администратору. Дальнейшие действия возможны только после исправления ошибки и восстановления безопасности канала связи.

**9. Несанкционированный доступ к данным:** Если система обнаруживает попытку несанкционированного доступа к базам данных или API, она должна немедленно прекращать выполнение запросов и регистрировать событие в логах с указанием IP-адреса, времени и типа атаки. Администратор должен получать уведомление о подобных инцидентах через email или корпоративный мессенджер.

**10. Превышение допустимого объема данных:** При попытке загрузки большого количества данных (более 10 000 записей в час), система должна временно ограничивать скорость обработки, помещая дополнительные запросы в очередь. Это позволит предотвратить перегрузку системы и сохранить стабильное время ответа для всех пользователей.

#### **3.3 Режимы работы**

Система должна поддерживать различные режимы работы для обеспечения корректного функционирования в реальных условиях эксплуатации, а также для тестирования и восстановления после аварийных ситуаций. Ниже описаны основные режимы работы системы: нормальный, тестовый и аварийный.

**1. Нормальный режим работы:** В нормальном режиме система должна функционировать согласно своим основным функциональным требованиям, выполняя автоматизацию процесса отбора кандидатов на основе обработки естественного языка из резюме и вакансий.

* Система должна принимать новые вакансии и резюме через интеграцию с внешними платформами рекрутмента или пользовательский интерфейс.
* Система должна автоматически преобразовывать текстовые данные в структурированные параметры (опыт работы, навыки, образование) и их векторные представления с помощью модели BERT.
* Система должна сравнивать векторные представления резюме и вакансий, используя метрику косинусного расстояния, чтобы определить уровень соответствия.
* Система должна формировать ранжированный список кандидатов для каждой вакансии и предоставлять его рекрутерам через удобный интерфейс или API.
* Система должна отправлять уведомления рекрутерам и соискателям о состоянии обработки резюме через email или интеграцию с мессенджерами.

**2. Тестовый режим работы:** Тестовый режим предназначен для проверки корректности работы системы на заранее подготовленных данных.

* Система должна загружать тестовые датасеты, содержащие пары "резюме-вакансия" с размеченными метками соответствия.
* Система должна вычислять метрики точности анализа (accuracy, MCC) для оценки качества классификации.
* Система должна записывать результаты тестирования в логи для последующего анализа администратором или научным сотрудником.
* Система должна предоставлять возможность выбора различных моделей embeddings (например, Word2Vec, GloVe, BERT) для сравнения результатов их работы.

**3. Аварийный режим работы:** Аварийный режим активируется при возникновении нештатных ситуаций, таких как отказ компонентов системы, временная недоступность внешних сервисов или проблемы с базой данных.

* При невозможности взаимодействия с Weaviate система должна временно сохранять векторные данные в оперативной памяти или использовать MongoDB для хранения векторных представлений до восстановления работоспособности Weaviate.
* Если PostgreSQL становится недоступна, система должна помещать новые запросы на создание или обновление вакансий в очередь для повторной обработки после восстановления доступа к базе данных.
* При отказе GPU система должна переключаться на использование CPU для выполнения задач обработки текстовых данных без прерывания работы.
* В случае ошибок шифрования данных (TLS 1.3, AES-256) система должна немедленно прекратить обработку запросов и отправить уведомление администратору для принятия мер по исправлению ситуации.
* При достижении предельной нагрузки (более 10 000 записей в час) система должна помещать дополнительные запросы в очередь для их последующей обработки, информируя пользователя о задержке.

#### **3.4 Взаимодействие с пользователями**

Система должна обеспечивать удобное и прозрачное взаимодействие с пользователями через различные интерфейсы, поддерживая их роли в процессе рекрутмента. Ниже описаны способы, как пользователи будут взаимодействовать с системой.

**1. Рекрутер:**

* Система должна предоставлять рекрутеру возможность создания и редактирования вакансий через графический интерфейс или API.
* Система должна показывать рекрутеру ранжированный список кандидатов для каждой вакансии с вероятностью соответствия [0, 1], где значение близкое к 1 указывает на высокую релевантность.
* Для каждого кандидата система должна объяснять причины его ранжирования, выделяя совпадения ключевых навыков, опыта работы и образования.
* Система должна отправлять уведомления рекрутеру о новых подходящих кандидатах через email или интеграцию с корпоративными мессенджерами.

**2. Соискатель:**

* Система должна принимать загрузку резюме соискателя в форматах PDF, DOCX, TXT через пользовательский интерфейс или API платформ рекрутмента.
* После анализа системы соискатель должен получать уведомление о статусе своего резюме (принято/отклонено) через email или интеграцию с платформами размещения резюме.
* Уведомления должны содержать информацию о том, почему резюме было отобрано или отклонено, например, недостаточный опыт работы или несоответствие ключевых навыков.

**3. Администратор:**

* Система должна позволять администратору осуществлять управление параметрами модели BERT через административный интерфейс, включая fine-tuning на новых данных.
* Администратор должен иметь возможность просматривать метрики качества классификации (accuracy, MCC) для оценки производительности системы.
* Система должна предоставлять администратору доступ к логам событий для анализа ошибок и контроля работоспособности компонентов.
* При необходимости администратор должен иметь возможность перезапуска микросервисов через Docker/Kubernetes для восстановления работоспособности системы.

**4. Научный сотрудник/разработчик:**

* Система должна предоставлять научному сотруднику возможность тестирования различных моделей embeddings (например, Word2Vec, GloVe) для сравнения результатов с моделью BERT.
* Научный сотрудник должен иметь возможность проводить эксперименты по расширению набора признаков для улучшения точности сопоставления резюме и вакансий.
* Система должна генерировать отчеты о качестве работы алгоритмов для научного сотрудника, включая accuracy, MCC и время обработки запросов.

**5. Интерфейсы взаимодействия:**

* Система должна поддерживать RESTful API для взаимодействия с внешними HR-платформами (например, hh.ru, LinkedIn Talent Solutions).
* Через API система должна предоставлять возможность автоматической загрузки данных о резюме и вакансиях для их дальнейшего анализа.
* Пользовательский интерфейс должен быть интуитивно понятным, позволяя рекрутерам легко создавать вакансии, загружать резюме и просматривать результаты анализа.

**6. Команды управления:**

* Система должна предоставлять команды для управления вакансиями: создание, редактирование, удаление через графический интерфейс или API.
* Система должна поддерживать команды для фильтрации результатов по различным критериям (опыт работы, образование, географическое положение).
* Администратор должен иметь возможность запуска процессов резервного копирования баз данных PostgreSQL и Weaviate через специальные команды интерфейса.

**7. Обоснование взаимодействия с пользователями:**

Требования к взаимодействию с пользователями основаны на необходимости обеспечить простоту использования системы для всех типов участников процесса рекрутмента. Например, использование модели BERT для анализа текстовых данных требует сложной предобработки, но графический интерфейс скрывает эту техническую сложность, предоставляя рекрутерам готовый рейтинг кандидатов.

Интеграция с внешними платформами через RESTful API позволяет автоматизировать процесс получения данных о резюме и вакансиях, а также отправку результатов анализа. Это снижает участие человека в первичном отборе кандидатов и повышает общую эффективность системы.

### **4. Операционные требования**

#### **4.1 Производительность**

Система должна соответствовать следующим требованиям к скорости, времени отклика и пропускной способности для обеспечения её эффективного использования в реальных условиях рекрутмента.

1. **Требования к скорости обработки данных:**
   * Система должна обрабатывать один запрос (пару "резюме-вакансия") за **не более 2 секунды** , включая все этапы: предобработку текста, векторизацию с помощью модели BERT и вычисление степени соответствия.
   * При работе с пиковой нагрузкой система должна обрабатывать до **10 000 записей в час** , сохраняя стабильное время выполнения операций для каждой записи.
2. **Требования к времени отклика API:**
   * Все запросы через RESTful API должны обрабатываться системой за **не более 2 секунды** , независимо от типа операции (загрузка резюме, создание/редактирование вакансии, получение результатов анализа).
   * Время ответа на запрос получения ранжированного списка кандидатов должно составлять **не более 5 секунд** для набора из **100 резюме** .
3. **Требования к пропускной способности системы:**
   * Система должна поддерживать одновременную загрузку до **100 резюме** для одной вакансии без ухудшения производительности.
   * Пропускная способность API при взаимодействии с внешними платформами рекрутмента (например, hh.ru) должна составлять **не менее 50 запросов в минуту** для бесплатных учетных записей.
4. **Требования к времени формирования отчетов:**
   * Система должна генерировать отчеты о результате анализа резюме в форматах CSV или PDF за **не более 5 минут** для набора из **1 000 записей** .
5. **Требования к времени восстановления после отказа:**
   * В случае сбоя микросервиса система должна перенаправлять запросы на резервные экземпляры через Kubernetes, чтобы обеспечить восстановление работоспособности компонентов за **не более 15 минут** .
6. **Требования к масштабируемости:**
   * Для увеличения нагрузки система должна быть способна автоматически добавлять новые экземпляры микросервисов через Docker/Kubernetes, чтобы поддерживать требуемую пропускную способность при больших объемах данных.
7. **Требования к времени первичной загрузки данных:**
   * Система должна загружать данные о новых резюме и вакансиях через интеграцию с внешними платформами не более чем за **10 минут** для набора из **10 000 записей** .
8. **Требования к времени обучения модели:**
   * Процесс fine-tuning модели BERT должен завершаться за **не более 2 часа** для четырех эпох, чтобы минимизировать простои системы при обновлении алгоритмов.

#### **4.2 Надежность**

Система должна обеспечивать высокую устойчивость к сбоям и способность восстанавливаться после них для поддержания непрерывной работы в реальных условиях эксплуатации.

1. **Требования к времени восстановления после отказа (MTTR):**
   * Система должна быть восстановлена в рабочее состояние за **не более 15 минут** после выявления сбоя любого микросервиса или компонента системы.
2. **Резервное копирование данных:**
   * Система должна создавать резервные копии данных из PostgreSQL ежедневно, а из Weaviate — еженедельно. Все бэкапы должны шифроваться алгоритмом **AES-256** перед сохранением.
3. **Обработка временных сбоев внешних платформ:**
   * При временном недоступности внешних HR-платформ (например, hh.ru) система должна помещать запросы в очередь для последующей обработки после восстановления доступности.
4. **Механизмы аварийного восстановления:**
   * В случае отказа базы данных PostgreSQL система должна автоматически переключаться на резервную копию данных для продолжения работы.
   * Если возникает проблема с доступностью Weaviate, система должна временно использовать MongoDB для хранения векторных представлений до восстановления работоспособности Weaviate.
5. **Обеспечение отказоустойчивости при обработке данных:**
   * Система должна проверять целостность входных данных перед началом анализа. Если данные повреждены или не соответствуют форматам (PDF, DOCX, TXT), система должна возвращать информативное сообщение об ошибке пользователю.
6. **Управление пиковыми нагрузками:**
   * При достижении максимальной загрузки (до **10 000 записей в час** ) система должна распределять запросы между экземплярами микросервисов через Kubernetes, чтобы предотвратить перегрузку компонентов.
7. **Мониторинг состояния системы:**
   * Система должна постоянно собирать метрики производительности (CPU, GPU, RAM) и логировать все события через ELK Stack или аналогичные инструменты. Администратор должен получать уведомления о любых критических сбоях или снижении производительности.
8. **Защита от множественных неудачных попыток API-запросов:**
   * Система должна реализовать механизм Circuit Breaker для защиты от множественных неудачных попыток взаимодействия с внешними сервисами. После трех подряд неудачных запросов система должна временно прекращать отправку запросов к конкретному API на **1 минуту** , после чего возобновлять работу.
9. **Обработка ошибок во время обучения модели BERT:**
   * При возникновении ошибок во время fine-tuning модели BERT система должна сохранять промежуточные результаты обучения и предоставлять администратору детальный отчет о состоянии процесса.
10. **Автоматическая отправка уведомлений об ошибках:**
    * В случае возникновения критических сбоев система должна немедленно отправлять уведомления администратору через email или корпоративный мессенджер. Уведомления должны содержать информацию о типе ошибки, времени её возникновения и возможные пути решения.

#### **4.3 Безопасность**

Система должна обеспечивать надежную защиту данных и компонентов от несанкционированного доступа, гарантируя конфиденциальность и целостность информации в реальных условиях эксплуатации. Ниже описываются требования к безопасности системы.

1. **Защита передачи данных:**
   * Система должна использовать протокол TLS 1.3 для шифрования всех данных при их передаче между компонентами системы и внешними платформами рекрутмента.
   * Все API-запросы должны быть защищены с использованием HTTPS, чтобы предотвратить перехват данных третьими лицами.
2. **Защита хранения данных:**
   * Система должна шифровать данные в покое с использованием алгоритма AES-256 для защиты информации в базах данных PostgreSQL и Weaviate.
   * Резервные копии данных должны также шифроваться тем же алгоритмом перед сохранением на удаленных серверах или локальных носителях.
3. **Аутентификация пользователей:**
   * Система должна поддерживать механизмы аутентификации через OAuth 2.0 или JWT для обеспечения безопасного доступа к API и интерфейсу.
   * Для администраторов система должна предоставлять возможность использования многофакторной аутентификации (MFA) для дополнительной защиты учетных записей.
4. **Ролевая модель управления доступом (RBAC):**
   * Система должна реализовать ролевую модель управления доступом, где каждому типу пользователя (рекрутер, администратор, соискатель) назначаются строго определенные права доступа.
   * Например, рекрутеры должны иметь доступ только к данным вакансий своей компании и результатам анализа резюме, а соискатели — только к своему профилю.
5. **Логирование действий:**
   * Система должна детально регистрировать все действия пользователей и операторов в логах с указанием времени, типа операции, ID пользователя и статуса выполнения.
   * Логи должны быть защищены от несанкционированного доступа и шифроваться перед хранением.
6. **Обеспечение соответствия законодательству:**
   * Система должна соответствовать нормативным требованиям GDPR и ФЗ-152 "О персональных данных", чтобы гарантировать легальность обработки данных пользователей.
   * Перед началом обработки данных система должна получать явное согласие пользователей на использование их персональной информации.
7. **Защита от распространенных угроз:**
   * Система должна быть защищена от SQL-инъекций, XSS-атак и CSRF, что достигается путем валидации входных данных и использования современных фреймворков разработки.
   * Все файлы резюме перед загрузкой в систему должны проверяться на наличие вредоносного кода или поврежденных данных.
8. **Контроль доступа к API:**
   * Система должна ограничивать количество запросов к API через механизм Rate Limiting, чтобы предотвратить атаки типа DDoS и злоупотребление ресурсами.
   * Каждый запрос к API должен содержать токен авторизации, который система должна проверять перед выполнением операции.
9. **Мониторинг безопасности:**
   * Система должна постоянно мониторить состояние безопасности своих компонентов, используя инструменты, такие как ELK Stack или аналогичные системы сбора логов.
   * При обнаружении подозрительной активности (например, множественные неудачные попытки авторизации) система должна немедленно отправлять уведомления администратору.
10. **Временная блокировка при несанкционированных попытках доступа:**
    * В случае нескольких подряд неудачных попыток авторизации система должна временно блокировать доступ к аккаунту пользователя на **15 минут** , чтобы предотвратить брутфорс атаки.
11. **Обновление системы безопасности:**
    * Система должна регулярно обновляться для исправления известных уязвимостей, включая зависимости, такие как библиотеки машинного обучения и NLP.

#### **4.4 Удобство использования**

Система должна обеспечивать максимальную простоту и удобство взаимодействия для всех типов пользователей, включая рекрутеров, администраторов и соискателей. Ниже описываются требования к интерфейсам и процессам взаимодействия с системой.

**1. Требования к пользовательскому интерфейсу:**

* Система должна предоставлять интуитивно понятный графический интерфейс для рекрутеров, позволяющий легко создавать, редактировать и удалять вакансии.
* Интерфейс должен поддерживать возможность загрузки резюме в форматах PDF, DOCX и TXT через функционал drag-and-drop или кнопку "Обзор".
* Система должна показывать ранжированный список кандидатов с вероятностью соответствия [0, 1], где значение близкое к 1 указывает на высокую релевантность.

**2. Требования к детализации результатов анализа:**

* Для каждого кандидата система должна объяснять причины его ранжирования, выделяя совпадения ключевых параметров (опыт работы, образование, навыки) и их контекстное значение.
* Результаты должны быть представлены в виде таблицы или списка, что позволит рекрутерам быстро оценить соответствие кандидатов требованиям вакансии.

**3. Требования к уведомлениям:**

* Система должна автоматически отправлять уведомления рекрутерам о новых подходящих кандидатах через email или интеграцию с корпоративными мессенджерами.
* Соискатели должны получать обратную связь о статусе обработки их резюме (принято/отклонено), а также причину решения системы.

**4. Требования к административному интерфейсу:**

* Администратор должен иметь доступ к административному интерфейсу для управления параметрами модели BERT, таких как веса для опыта работы, образования и навыков.
* Система должна предоставлять возможность просмотра метрик качества классификации (accuracy, MCC) для оценки производительности алгоритмов.

**5. Требования к документации:**

* Система должна сопровождаться подробной документацией, включающей руководства по использованию для рекрутеров, администраторов и разработчиков.
* Должны быть предоставлены видеоинструкции для быстрого освоения основных функций системы.

**6. Требования к скорости ответа системы:**

* Система должна отвечать на запросы пользователей за **не более 2 секунды** , чтобы обеспечить комфортное взаимодействие даже при больших нагрузках.
* Время загрузки страницы или обновления данных должно составлять **не более 1 секунду** для операций создания/редактирования вакансий.

**7. Требования к локализации интерфейса:**

* Система должна поддерживать два языка интерфейса: **русский** и **английский** , чтобы удовлетворять потребности как российских, так и международных пользователей.

**8. Требования к фильтрации и сортировке результатов:**

* Рекрутер должен иметь возможность фильтровать результаты по различным критериям (опыт работы, географическое положение, уровень образования).
* Система должна поддерживать сортировку кандидатов по вероятности соответствия требованиям вакансии.

### **5. Операционные ограничения**

#### **5.1 Технические ограничения**

Система должна работать с учетом следующих технических ограничений, связанных с аппаратным обеспечением, программным обеспечением и инфраструктурой:

1. **Аппаратное обеспечение:**
   * Система должна использовать графические процессоры (GPU) для ускорения вычислений модели BERT. Если GPU недоступны, система должна переключаться на центральные процессоры (CPU), но при этом время обработки одного запроса не должно превышать **10 секунд** .
   * Для корректной работы системы минимальный объем оперативной памяти должен составлять **16 ГБ** , а рекомендуемый — **32 ГБ** .
2. **Программное обеспечение:**
   * Система должна поддерживать только текстовые файлы в форматах **PDF, DOCX, TXT** для загрузки резюме и вакансий. Другие форматы (например, HTML, JPG) требуют предварительной конвертации.
   * Модель BERT должна использоваться для анализа текстов на **русском** и **английском** языках. Поддержка других языков возможна только после дополнительного обучения модели.
   * При взаимодействии с внешними платформами рекрутмента (например, hh.ru) через RESTful API система должна соблюдать ограничения по количеству запросов в секунду (QPS). Например, для платформы hh.ru это ограничение составляет **50 запросов в минуту** для бесплатных учетных записей.
3. **Инфраструктурные ограничения:**
   * Система должна хранить структурированные данные о резюме и вакансиях в PostgreSQL, а векторные представления данных — в Weaviate. Использование других баз данных возможно только после оценки их производительности и соответствия требованиям безопасности.
   * Размер векторных представлений (embeddings) текстов резюме и вакансий должен быть фиксированным — **768 элементов** для каждой записи. Это связано с особенностями модели BERT и алгоритмов поиска ближайших соседей (например, HNSW).
   * Резервное копирование данных из PostgreSQL должно выполняться ежедневно, а из Weaviate — еженедельно. Все резервные копии должны шифроваться с использованием алгоритма **AES-256** перед сохранением.
4. **Ограничения по масштабируемости:**
   * Система должна быть способна обрабатывать до **10 000 записей в час** , что соответствует максимальной нагрузке при работе с реальными данными. Превышение этого значения может потребовать дополнительного масштабирования через Kubernetes.
   * Количество одновременно работающих экземпляров микросервисов через Docker/Kubernetes должно быть ограничено доступными вычислительными мощностями сервера.
5. **Ограничения по времени выполнения операций:**
   * Время обработки одного резюме или вакансии системой должно составлять **не более 2 секунд** для обеспечения комфортного взаимодействия пользователей.
   * Процесс fine-tuning модели BERT должен завершаться за **не более 2 часа** для четырех эпох, чтобы минимизировать простои системы при обучении.
6. **Ограничения по безопасности:**
   * Все данные, передаваемые между компонентами системы или внешними сервисами, должны шифроваться с использованием протокола **TLS 1.3** .
   * Хранение данных в базах данных должно осуществляться с применением шифрования **AES-256** , чтобы обеспечить защиту персональной информации согласно требованиям GDPR и ФЗ-152 "О персональных данных".
7. **Ограничения по объему данных:**
   * Максимальный размер файла резюме для загрузки в систему должен составлять **5 МБ** . Это гарантирует быстрое время обработки и снижает вероятность ошибок.
   * В случае необходимости обработки большого объема данных (более **172 321 записи** ) система должна временно помещать дополнительные запросы в очередь для последующей обработки.

#### **5.2 Ограничения по времени**

Система должна функционировать с учетом следующих временных ограничений для обеспечения эффективного взаимодействия пользователей и операторов в реальных условиях эксплуатации.

1. Система должна обеспечивать время отклика на запросы через пользовательский интерфейс или API не более **1 секунды** , чтобы минимизировать задержки при работе рекрутеров с сервисом. Это касается как создания/редактирования вакансий, так и получения результатов анализа резюме.
2. Система должна обрабатывать один запрос (пару "резюме-вакансия") за **не более 2 секунды** , включая все этапы: предобработку текста, его векторизацию с помощью модели BERT и вычисление степени соответствия.
3. При массовой загрузке данных система должна сохранять стабильное время обработки каждого резюме, даже если общее количество запросов достигает **10 000 записей в час** . Задержки сверх указанного значения могут привести к снижению удобства использования.
4. Время восстановления после отказа любой компоненты системы (например, микросервиса или базы данных) должно составлять **не более 15 минут** , чтобы гарантировать минимальное влияние сбоев на рабочие процессы.
5. Процесс fine-tuning модели BERT должен быть завершен за **не более 2 часа** для четырех эпох обучения, чтобы администратор мог быстро адаптировать модель под новые данные без длительных простоев системы.
6. Система должна отправлять уведомления рекрутерам о подходящих кандидатах и соискателям о результате обработки их резюме в течение **1 минуты** после завершения анализа конкретного резюме.
7. Резервное копирование данных из PostgreSQL должно выполняться ежедневно за **не более 30 минут** , а из Weaviate — еженедельно за **не более 1 часа** , чтобы минимизировать риск потери данных.
8. При интеграции с внешними платформами рекрутмента (например, hh.ru) время загрузки данных о резюме и вакансиях через API должно составлять **не более 10 минут** для набора из **10 000 записей** , чтобы обеспечить своевременное получение информации.
9. Система должна формировать отчеты о результате анализа резюме в формате CSV или PDF за **не более 5 минут** для набора из **1 000 записей** , чтобы рекрутеры могли оперативно использовать эти данные.

#### **5.3 Регуляторные ограничения**

Система должна соответствовать регуляторным требованиям, связанным с защитой персональных данных и обеспечением безопасности информации, чтобы гарантировать законность её использования в реальных условиях эксплуатации.

1. **Соответствие GDPR:**
   * Система должна соблюдать положения регламента защиты персональных данных GDPR, что включает получение явного согласия пользователей на обработку их данных перед началом анализа резюме или вакансий.
   * Система должна предоставлять возможность соискателям запрашивать удаление своих данных из базы системы через специальный интерфейс или API.
2. **Соответствие ФЗ-152 "О персональных данных":**
   * Система должна хранить персональные данные пользователей в зашифрованном виде с использованием алгоритма AES-256 для защиты от несанкционированного доступа.
   * Передача данных между компонентами системы или внешними платформами должна осуществляться через шифрованные каналы связи с применением протокола TLS 1.3.
3. **Ограничение использования данных без согласия пользователя:**
   * Система должна использовать текстовые данные резюме только для целей, указанных при согласии соискателя, например, для анализа соответствия требованиям конкретной вакансии.
4. **Обеспечение прозрачности алгоритмов:**
   * Система должна предоставлять детальную информацию о причинах ранжирования каждого кандидата, чтобы избежать обвинений в дискриминации или предвзятости.
5. **Защита конфиденциальной информации:**
   * Система должна обеспечивать защиту всех загруженных текстовых данных (резюме и вакансий) от утечки или несанкционированного доступа. Это требует постоянного мониторинга состояния безопасности системы.
   * Система должна исключать использование персональных данных (например, имени, возраста, пола) в процессе семантического анализа, если это не требуется для решения задачи.
6. **Логирование действий с учетом регуляторных требований:**
   * Все действия, связанные с обработкой персональных данных, должны быть задокументированы в логах, включая время события, тип операции, ID пользователя и статус выполнения. Логи должны храниться в защищенном виде и быть доступны только авторизованным администраторам.
7. **Интеграция с внешними платформами:**
   * При интеграции с внешними HR-платформами (например, hh.ru или LinkedIn Talent Solutions) система должна следовать требованиям безопасности данных этих платформ, таким как ограничения по количеству запросов в секунду (QPS).

### **6. Операционные риски**

#### **6.1 Идентификация рисков**

В данном разделе описываются основные риски, связанные с операционной деятельностью системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Особое внимание уделяется риску сбоя системы при внезапном отключении электричества.

**1. Риск сбоя системы при отключении электричества:**

* Система должна иметь механизм аварийного сохранения текущего состояния для всех компонентов, особенно для микросервисов и баз данных, чтобы минимизировать потерю данных при внезапном отключении питания.
* Система должна автоматически переключаться на резервное питание или использовать механизмы восстановления после возобновления электроснабжения. Это включает:
  + Восстановление работы PostgreSQL и Weaviate через автоматическое подключение резервных копий данных.
  + Перезапуск контейнеров Docker/Kubernetes для всех микросервисов без потери очереди запросов.
* Система должна информировать администратора о сбое через уведомления (email, SMS) сразу после выявления проблемы, чтобы обеспечить быстрое реагирование на ситуацию.

#### **6.2 Меры по снижению рисков**

Система должна предусматривать меры для снижения операционных рисков, связанных с возможными сбоями в работе компонентов системы и её инфраструктуры. Особое внимание уделяется установке резервного источника питания для предотвращения аварийных ситуаций, вызванных внезапным отключением электричества.

1. **Установка резервного источника питания:**
   * Система должна быть оборудована резервным источником питания (РИП) для всех ключевых компонентов, таких как серверы, базы данных (PostgreSQL, Weaviate) и GPU-модули. Это позволит избежать полной остановки работы системы при кратковременных перебоях в электроснабжении.
   * РИП должен обеспечивать подачу электроэнергии на все компоненты системы в течение **не менее 15 минут** , чтобы дать возможность выполнить аварийное сохранение текущего состояния системы или завершить работу с минимальными потерями данных.
2. **Аварийное сохранение данных:**
   * В случае отключения основного источника питания система должна автоматически переключаться на режим аварийного сохранения данных. Все текущие процессы обработки должны быть временно приостановлены, а данные — зафиксированы в PostgreSQL и Weaviate.
3. **Восстановление после отключения электричества:**
   * После восстановления электроснабжения система должна автоматически перезапустить все микросервисы через Docker/Kubernetes. Перезапуск должен быть выполнен за **не более 2 минут** для каждого компонента системы.
   * Система должна продолжить обработку запросов, помещенных в очередь во время простоя, без потери их последовательности и корректности результатов.
4. **Мониторинг состояния электроснабжения:**
   * Система должна постоянно мониторить состояние электроснабжения и информировать администратора о возникновении проблем через уведомления (email, SMS). Это позволит своевременно реагировать на ситуации, связанные с рисками отключения электричества.

### **7. Операционные воздействия**

#### **7.1 Воздействие на пользователей**

Система должна оказать существенное влияние на работу пользователей и операторов, улучшая процесс рекрутмента за счет автоматизации первичного отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Ниже описаны основные аспекты воздействия системы на различные роли пользователей.

**1. Воздействие на рекрутеров:**

* Система должна значительно сократить время, затрачиваемое рекрутерами на первичную фильтрацию резюме, предоставляя ранжированный список кандидатов с вероятностью соответствия [0, 1]. Это позволит рекрутерам сосредоточиться на более стратегических задачах, таких как интервьюирование и оценка soft skills.
* Система должна минимизировать влияние человеческого фактора при первичном анализе резюме, что снижает риск ошибок и повышает объективность отбора кандидатов.
* Рекрутеры должны получать детальную информацию о причинах ранжирования каждого кандидата, например, совпадение ключевых навыков, уровня опыта или образования. Это обеспечит прозрачность решений и возможность корректировки параметров анализа.

**2. Воздействие на соискателей:**

* Система должна предоставлять соискателям быструю обратную связь о статусе их резюме через email или интеграцию с платформами размещения резюме. Это повысит удовлетворенность соискателей благодаря своевременным уведомлениям о принятых решениях.
* Уведомления должны содержать информацию о том, почему резюме было принято или отклонено, чтобы соискатели могли лучше понять требования работодателей и улучшить свои профили.

**3. Воздействие на администраторов:**

* Администраторы должны иметь возможность управления параметрами модели BERT через административный интерфейс, например, изменение весов для различных критериев (опыт работы, образование, навыки). Это позволит адаптировать систему под специфику компании или конкретной вакансии.
* Система должна предоставлять администраторам доступ к метрикам качества классификации (accuracy, MCC) для оценки эффективности алгоритмов и необходимости доработки.

**4. Воздействие на научных сотрудников и разработчиков:**

* Система должна обеспечивать возможность тестирования различных моделей embeddings (например, Word2Vec, GloVe) для сравнения результатов с моделью BERT. Это поможет научным сотрудникам выбрать наиболее подходящую модель для конкретных условий использования.
* Разработчики должны иметь доступ к логам событий и метрикам производительности для анализа состояния системы и выявления потенциальных проблем.

**5. Обоснование воздействия на пользователей:**

Внедрение системы автоматизации отбора кандидатов должно существенно оптимизировать рабочие процессы всех участников рекрутмента. Например, использование модели BERT для анализа текстовых данных показало высокую точность (accuracy = 0.76) и коэффициент корреляции Мэтьюса (MCC = 0.55) на тестовых данных объемом **172 321 запись** (раздел 4.4). Это гарантирует, что рекрутеры будут получать качественные результаты анализа без необходимости ручной проверки большого количества резюме.

Также система должна учитывать контекст использования слов, что особенно важно при анализе сложных текстов резюме и вакансий. Например, различие между фразами «управление проектами» и «проект управления опытом» может быть правильно интерпретировано благодаря механизму само-внимания (Self-Attention) модели BERT. Это позволяет рекрутерам принимать более взвешенные решения.

Для соискателей внедрение системы должно стать шагом к более прозрачному и быстрому процессу получения обратной связи о статусе их резюме. Это способствует повышению общего уровня удовлетворенности соискателей и улучшению их взаимодействия с компанией.

Администраторы и научные сотрудники получают возможность контролировать и совершенствовать работу системы, используя современные инструменты мониторинга и анализа данных. Это делает систему гибкой и готовой к изменениям в условиях реальной эксплуатации.

#### **7.2 Воздействие на процессы**

Внедрение системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий потребует внесения изменений в существующие процессы рекрутмента. Ниже описаны основные изменения, которые должны быть реализованы для успешного использования разрабатываемого сервиса.

**1. Изменение процесса создания и публикации вакансий:** Система должна требовать от рекрутеров более четкого и структурированного описания требований к вакансии. Это включает:

* Указание ключевых навыков, опыта работы и уровня образования в формате, удобном для анализа системой.
* Система должна предоставлять рекрутерам возможность проверять корректность данных перед их публикацией через API внешних платформ рекрутмента.

**2. Автоматизация первичной фильтрации резюме:** Текущие процессы ручной фильтрации резюме должны быть заменены автоматизированным анализом с использованием NLP.

* Система должна взять на себя задачу первичного анализа резюме, используя модель BERT для семантического сопоставления с требованиями вакансии.
* Рекрутер должен получать ранжированный список кандидатов с вероятностью соответствия [0, 1], что позволит сосредоточиться на интервьюировании и оценке soft skills подходящих кандидатов.

**3. Интеграция с внешними платформами:** Для обеспечения эффективного взаимодействия с HR-платформами (например, hh.ru) необходимо модифицировать текущие процессы интеграции данных.

* Система должна автоматически загружать данные о резюме и вакансиях через RESTful API платформ рекрутмента.
* Все загруженные данные должны быть предобработаны для исключения неструктурированных или некорректных записей.

**4. Изменение способа обратной связи соискателям:** Процесс уведомления соискателей о результате рассмотрения их резюме должен стать более прозрачным и оперативным.

* Система должна отправлять автоматические уведомления соискателям о статусе их резюме (принято/отклонено) через email или интеграцию с платформами размещения резюме.
* Уведомления должны содержать детальную информацию о причинах принятия или отклонения резюме для повышения удовлетворенности соискателей.

**5. Обновление параметров анализа:** Рекрутеры должны иметь возможность легко обновлять параметры анализа для каждой вакансии без необходимости ручной проверки большого количества резюме.

* Система должна предоставлять интерфейс для изменения весов параметров анализа (опыт работы, образование, навыки).
* При каждом изменении вакансии система должна пересчитывать соответствие всех ранее загруженных резюме автоматически.

**6. Мониторинг и логирование:** Требуется внедрить дополнительные процессы мониторинга и логирования для контроля работоспособности системы и анализа её производительности.

* Система должна регистрировать все события во время обработки данных, такие как загрузка резюме, создание вакансий и результаты анализа. Логи должны храниться в зашифрованном виде для защиты конфиденциальной информации.
* Администратор должен иметь доступ к метрикам качества классификации (accuracy, MCC) для оценки точности работы алгоритмов на регулярной основе.

**7. Обучение пользователей:** Для успешного внедрения системы рекрутеры и администраторы должны пройти обучение работе с новыми функциями и интерфейсами.

* Система должна предоставлять подробную документацию и видеоинструкции для быстрого освоения её функциональности всем пользователям.
* Обучение должно включать описание того, как использовать ранжированные списки кандидатов и интерпретировать причины их ранжирования.

**8. Повышение стандартизации данных:** Текущие процессы обработки резюме часто сталкиваются с проблемами нестандартизированности данных. Для решения этой проблемы:

* Система должна предлагать рекрутерам шаблоны для создания вакансий, чтобы минимизировать вариативность в формулировках требований.
* Соискатели должны получать рекомендации по структурированию своего резюме для повышения его релевантности при автоматическом анализе.

**9. Переход к модели TO BE:** Существующая модель AS IS рекрутмента характеризуется значительными временными затратами на первичный отбор кандидатов. После внедрения системы процесс должен быть оптимизирован согласно модели TO BE.

* Система должна минимизировать участие человека в первичном анализе резюме, ограничиваясь только стратегическими задачами, такими как интервьюирование и оценка soft skills.
* Процессы должны быть автоматизированы на уровне сбора данных, их предобработки, анализа и ранжирования кандидатов.

#### **7.3 Воздействие на инфраструктуру**

Внедрение разрабатываемой системы для автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий окажет значительное влияние на существующую инфраструктуру организации. Ниже описываются необходимые изменения, которые система должна требовать для своей успешной работы.

**1. Увеличение вычислительных мощностей:** Система должна требовать наличия графических процессоров (GPU) с поддержкой CUDA для ускорения вычислений модели BERT. Если GPU недоступны, то система должна иметь возможность работать на центральных процессорах (CPU), но при этом время обработки одного запроса не должно превышать **10 секунд** . Это связано с тем, что модель BERT требует значительных вычислительных ресурсов для преобразования текстов в контекстуальные векторные представления.

**2. Обеспечение достаточного объема оперативной памяти:** Система должна требовать минимальный объем оперативной памяти **16 ГБ** , рекомендуемый — **32 ГБ** , чтобы обеспечить корректную работу всех компонентов системы, особенно при обработке больших объемов данных (до **172 321 записи** ).

**3. Расширение хранилищ данных:** Система должна использовать следующие базы данных:

* **PostgreSQL** : Для хранения структурированных данных о резюме и вакансиях. Система должна требовать регулярного увеличения объема дискового пространства PostgreSQL для поддержания растущего количества записей.
* **Weaviate** : Для хранения векторных представлений текстовых данных. Размер хранимых векторов составляет **768 элементов** для каждого документа. Система должна требовать оптимизацию Weaviate через использование индексации HNSW для повышения производительности поиска ближайших соседей.
* **MongoDB** : Для временного хранения полуструктурированных JSON-данных до их преобразования в векторы или структуры PostgreSQL.

**4. Интеграция с внешними платформами:** Система должна требовать установки API-шлюзов для интеграции с внешними HR-платформами, такими как hh.ru или LinkedIn Talent Solutions. Это позволит автоматически загружать данные о резюме и вакансиях без необходимости ручного ввода информации. При этом количество одновременных запросов через API должно быть ограничено требованиями платформы (например, hh.ru допускает **50 запросов в минуту** для бесплатных учетных записей).

**5. Шифрование каналов передачи данных:** Система должна требовать внедрения протокола TLS 1.3 для шифрования всех данных при их передаче между компонентами системы и внешними сервисами. Все внутренние и внешние коммуникации должны быть защищены, чтобы гарантировать соответствие требованиям GDPR и ФЗ-152 "О персональных данных".

**6. Резервное копирование данных:** Система должна требовать ежедневного резервного копирования данных из PostgreSQL и еженедельного — из Weaviate. Все резервные копии должны шифроваться алгоритмом AES-256 перед сохранением, чтобы предотвратить несанкционированный доступ к конфиденциальной информации.

**7. Масштабируемость инфраструктуры:** Система должна требовать внедрения Kubernetes для управления контейнерами Docker, что позволит легко масштабировать микросервисы системы. Каждый микросервис должен быть способен к горизонтальному масштабированию для обработки увеличенных нагрузок (до **10 000 записей в час** ).

**8. Мониторинг состояния системы:** Система должна требовать установки инструментов мониторинга, таких как Prometheus и Grafana, для постоянного контроля метрик производительности (CPU, GPU, RAM) и здоровья компонентов системы. Это позволит своевременно реагировать на возможные проблемы и оптимизировать работу всей инфраструктуры.

# ConOps (Conceptual Operations)

### **1 Цель**

#### **1.1 Объяснение роли ConOps в проекте**

Данный документ ConOps (Concept of Operations) должен создаваться для определения стратегического видения организации относительно использования системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Система должна удовлетворять потребности организации в оптимизации процессов рекрутмента, что обеспечит соответствие её долгосрочным целям бизнеса. Document ConOps должен служить основой для понимания того, как система будет интегрирована в текущие бизнес-процессы и какие стратегические задачи она поможет решить.

#### **1.2 Решаемые вопросы**

1. **Связь между системой и бизнес-целями организации:**
   * Система должна поддерживать стратегическую цель организации по повышению эффективности рекрутмента за счет автоматизации первичного анализа резюме. Это позволит сократить временные затраты HR-специалистов на фильтрацию большого количества кандидатов и улучшить качество принимаемых решений.
   * Система должна способствовать достижению целей компании по снижению операционных затрат на подбор персонала, минимизируя влияние человеческого фактора при первичном отборе кандидатов.
2. **Определение роли ConOps в контексте проекта:**
   * Документ должен предоставлять чёткое представление о том, как разрабатываемая система будет взаимодействовать с существующими бизнес-процессами и структурами управления персоналом.
   * Система должна соответствовать требованиям безопасности данных, установленным организацией, чтобы гарантировать защиту конфиденциальной информации соискателей и работодателей, а также соблюдение нормативных требований GDPR и ФЗ-152 "О персональных данных".
3. **Понимание долгосрочных целей внедрения системы:**
   * Система должна способствовать развитию компетенций HR-отдела, позволяя сотрудникам сосредоточиться на стратегических задачах, таких как интервьюирование и оценка soft skills кандидатов.
   * Разрабатываемый сервис должен обеспечивать возможность масштабирования для работы с увеличивающимися объёмами данных, что важно для роста бизнеса и расширения рынков присутствия организации.

#### **1.3 Обоснование создания документа**

Внедрение современных технологий обработки естественного языка, таких как модель BERT, позволяет значительно повысить точность сопоставления резюме с требованиями вакансий. Однако для успешного использования этой технологии необходимо четко определить, как именно система будет интегрирована в существующую экосистему рекрутмента и какие бизнес-цели она будет помогать достигать. Document ConOps должен решать следующие ключевые вопросы:

* Как система будет способствовать достижению стратегических целей организации по оптимизации процессов подбора персонала?
* Какие изменения в текущих процессах рекрутмента (AS IS) должны быть внедрены для перехода к более эффективному состоянию (TO BE)?
* Какие нефункциональные требования (производительность, безопасность, удобство использования) должны быть учтены для обеспечения долгосрочной успешности проекта?

На основе проведённого анализа (раздел 1.6) становится очевидным, что разработка данного документа необходима для координации действий всех участников проекта и обеспечения соответствия системы бизнес-стратегии организации. Это позволит сделать процесс рекрутмента более быстрым, точным и менее зависимым от человеческого фактора.

### **2 Область применения**

#### **2.1 Границы документа**

Настоящий документ ConOps должен охватывать следующие аспекты бизнеса и системы:

1. **Стратегическое видение использования системы:**
   * Документ должен предоставлять стратегическое представление о том, как разрабатываемая система автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий будет поддерживать долгосрочные цели организации.
2. **Ограничения по анализируемым данным:**
   * Система должна фокусироваться на анализе текстовых данных резюме и вакансий, представленных в форматах PDF, DOCX, TXT. Другие форматы (например, HTML, JPG) должны быть исключены из анализа без предварительной конвертации.
3. **Языковая специфика:**
   * Система должна работать с текстами только на русском и английском языках, используя модель BERT для их семантического анализа. Поддержка других языков должна быть рассмотрена только после дополнительного обучения модели.
4. **Объем рассматриваемых данных:**
   * Документ должен описывать работу системы с датасетами объемом до **172 321 записи** , что соответствует тестовым экспериментам. При необходимости работы с большими объемами данных система должна масштабироваться через Kubernetes.
5. **Аспекты безопасности:**
   * Система должна соответствовать требованиям GDPR и ФЗ-152 "О персональных данных" для защиты передаваемых и хранимых данных. Вся передача данных должна осуществляться через шифрование TLS 1.3, а хранение — с использованием AES-256.

Таким образом, данный документ ConOps должен ограничиваться описанием того, как система автоматизирует процессы рекрутмента, минимизируя влияние человеческого фактора, и как она интегрируется в существующие бизнес-процессы организации.

#### **2.2 Контекст системы**

Разрабатываемая система должна быть интегрирована в текущие и будущие бизнес-процессы рекрутмента, чтобы повысить эффективность работы HR-отдела и обеспечить соответствие стратегическим целям компании. Ниже описан контекст её использования:

1. **Текущий процесс рекрутмента (AS IS):**
   * В рамках текущего состояния рекрутмента (AS IS) HR-специалисты выполняют ручную фильтрацию резюме, что требует значительных временных затрат и может привести к ошибкам из-за человеческого фактора. Система должна заменить этот этап автоматическим анализом, позволяя рекрутерам сосредоточиться на интервьюировании и оценке soft skills.
2. **Целевое состояние рекрутмента (TO BE):**
   * После внедрения системы (TO BE) первичный отбор кандидатов должен стать полностью автоматизированным, с использованием технологии NLP для сопоставления навыков, опыта работы и образования кандидатов с требованиями вакансий. Результаты анализа должны быть представлены рекрутерам в виде ранжированного списка кандидатов с вероятностью соответствия [0, 1].
3. **Интеграция с внешними платформами:**
   * Система должна взаимодействовать с внешними HR-платформами, такими как hh.ru, LinkedIn Talent Solutions, через RESTful API для автоматической загрузки резюме и отправки результатов анализа. Это позволит ускорить процесс получения данных и их обработки, минуя необходимость ручного ввода информации.
4. **Поддержка многозадачности:**
   * Система должна одновременно обрабатывать запросы от нескольких пользователей (рекрутеров), сохраняя стабильное время ответа на запросы. При пиковых нагрузках система должна помещать дополнительные запросы в очередь для последующей обработки.
5. **Взаимодействие с внутренними базами данных:**
   * Для хранения структурированных данных система должна использовать PostgreSQL, а для векторных представлений текстов — Weaviate. Эти базы данных должны быть интегрированы в общую инфраструктуру для обеспечения быстрого доступа к информации и высокой производительности при больших нагрузках.
6. **Улучшение качества решений:**
   * Внедрение системы должно способствовать более точному сопоставлению кандидатов с требованиями вакансий за счет учета контекста слов в текстах резюме и вакансий. Например, система должна различать значения фраз "управление проектами" и "проект управления опытом".
7. **Будущее развитие бизнес-процессов:**
   * В дальнейшем система должна поддерживать расширение функциональности, например, добавление анализа дополнительных параметров (социальные профили, истории работы) и внедрение новых моделей машинного обучения для повышения точности анализа.

### **3 Стратегический план**

#### **3.1 Долгосрочные цели организации**

Система автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий должна способствовать достижению долгосрочных целей организации, направленных на оптимизацию процессов рекрутмента и повышение эффективности подбора персонала. Основные стратегические цели, которые система должна поддерживать:

1. **Сокращение временных затрат на рекрутмент:**
   * Система должна минимизировать участие человека в первичной фильтрации резюме, предоставляя ранжированный список кандидатов для каждой вакансии. Это позволит рекрутерам сосредоточиться на более стратегических задачах, таких как интервьюирование и оценка soft skills.
2. **Повышение точности сопоставления:**
   * Система должна использовать технологии обработки естественного языка (например, BERT) для учета контекста текстовых данных, что повысит качество соответствия между кандидатами и требованиями вакансий.
3. **Стандартизация процесса рекрутмента:**
   * Система должна обеспечивать стандартизированное взаимодействие с платформами поиска работы (hh.ru, LinkedIn Talent Solutions), что исключит несоответствия и улучшит прозрачность процесса найма.
4. **Улучшение пользовательского опыта:**
   * Система должна отправлять автоматические уведомления соискателям о результате анализа их резюме, что повысит удовлетворенность кандидатов за счет быстрой обратной связи.
5. **Соблюдение нормативных требований:**
   * Система должна соответствовать требованиям GDPR и ФЗ-152 "О персональных данных", чтобы гарантировать законность обработки информации о кандидатах и работодателях.

Таким образом, внедрение системы должно стать важным шагом в развитии HR-процессов организации, позволяющим достигать вышеуказанных целей.

#### **3.2 Этапы внедрения**

Внедрение разрабатываемой системы предполагает последовательный план действий, который делится на несколько этапов. Каждый этап должен быть тщательно спланирован для успешного развертывания системы и её интеграции в существующую инфраструктуру.

1. **Подготовительный этап:**
   * Система должна провести анализ текущих бизнес-процессов рекрутмента (AS IS) для выявления точек интеграции и оптимизации.
   * Необходимо подготовить данные для обучения модели BERT, включая сбор размеченных пар "резюме-вакансия".
2. **Первый этап: MVP (Minimum Viable Product):**
   * Система должна реализовать базовые функции автоматического анализа текстовых данных и формирования ранжированных списков кандидатов.
   * Необходимо протестировать систему на ограниченном наборе данных (например, 100 вакансий и 1000 резюме).
3. **Второй этап: Расширение функционала:**
   * Система должна добавить возможность fine-tuning модели BERT под специфику компании или конкретных вакансий.
   * Необходимо внедрить механизмы логирования событий для контроля производительности и качества работы алгоритмов.
4. **Третий этап: Масштабирование:**
   * Система должна масштабироваться для обработки больших объемов данных (до 172 321 записи тестового датасета).
   * Требуется интеграция с дополнительными внешними платформами рекрутмента через RESTful API.
5. **Четвертый этап: Обучение пользователей:**
   * Система должна предоставить подробную документацию и видеоинструкции для обучения рекрутеров работе с новым сервисом.
   * Необходимо провести тренинги для пользователей, чтобы они могли эффективно использовать результаты анализа.
6. **Пятый этап: Полное внедрение:**
   * Система должна быть интегрирована во все отделы кадров и процессы рекрутмента компании.
   * Требуется постоянный мониторинг работы системы для своевременного выявления проблем и доработки алгоритмов.

Этот план позволит поэтапно внедрить систему, начиная от минимального жизнеспособного продукта до полной интеграции в бизнес-процессы организации.

#### **3.3 Приоритеты**

На основе анализа бизнес-целей и технических возможностей, были определены ключевые функции системы, которые являются приоритетными для успешной эксплуатации.

1. **Автоматизация анализа резюме:**
   * Система должна автоматически извлекать ключевые параметры из текстов резюме и вакансий, такие как опыт работы, образование, навыки и другие характеристики.
2. **Семантическое сопоставление:**
   * Система должна использовать модель BERT для преобразования текстов в контекстуальные векторные представления и их сравнения с требованиями вакансий.
3. **Ранжирование кандидатов:**
   * Система должна формировать ранжированный список кандидатов для каждой вакансии, представляя результаты в виде вероятности соответствия [0, 1].
4. **Интеграция с внешними платформами:**
   * Система должна обеспечивать интеграцию с HR-платформами через RESTful API для автоматической загрузки резюме и отправки результатов анализа.
5. **Безопасность данных:**
   * Система должна шифровать все передаваемые и хранимые данные с использованием TLS 1.3 и AES-256 соответственно. Это гарантирует защиту конфиденциальной информации согласно требованиям GDPR и ФЗ-152 "О персональных данных".
6. **Прозрачность решений:**
   * Система должна объяснять причины ранжирования каждого кандидата, указывая совпадения по ключевым параметрам (опыт работы, навыки, образование).
7. **Масштабируемость:**
   * Система должна поддерживать горизонтальное масштабирование через Kubernetes для обработки увеличенных нагрузок (до 10 000 записей в час).
8. **Обновление параметров вакансий:**
   * Система должна автоматически пересчитывать соответствие всех ранее загруженных резюме при изменении параметров вакансии.

### **4 Эффективность**

#### **4.1 Ожидаемые результаты**

Разрабатываемая система автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий должна удовлетворять следующие количественные и качественные показатели успеха:

1. **Количественные показатели:**
   * Система должна сократить время первичного анализа одного резюме до **не более 2 секунд** , что позволит рекрутерам обрабатывать большие объемы данных без значительных задержек.
   * Процесс первичной фильтрации должен быть автоматизирован для **100% всех входящих резюме** , минимизируя участие человека на этом этапе.
   * Система должна обеспечивать точность классификации резюме по метрике accuracy **не менее 0.76** , а коэффициент корреляции Мэтьюса (MCC) должен составлять **не менее 0.55** , что гарантирует высокую эффективность работы алгоритмов при тестировании на датасете объемом **172 321 запись** .
2. **Качественные показатели:**
   * Система должна улучшить качество соответствия между кандидатами и требованиями вакансии за счет учета контекста слов в текстах резюме и вакансий, что достигается с помощью модели BERT. Например, она должна различать значения таких фраз как «управление проектами» и «проект управления опытом».
   * Ранжирование кандидатов должно предоставляться рекрутерам в виде вероятности [0, 1], где значение близкое к 1 указывает на высокое соответствие вакансии. Это позволяет сделать процесс отбора более прозрачным и объективным.
   * Уведомления соискателей о результате обработки их резюме должны отправляться **в течение 1 минуты после завершения анализа** , что повысит удовлетворенность кандидатов благодаря своевременной обратной связи.

#### **4.2 Метрики**

Для оценки эффективности разрабатываемой системы будут использоваться следующие ключевые показатели эффективности (KPI):

1. **Точность анализа (Accuracy):**
   * Система должна регулярно проверять свою точность классификации резюме по сравнению с реальными решениями рекрутеров. Требуемый уровень accuracy должен составлять **не менее 0.76** .
2. **Коэффициент корреляции Мэтьюса (MCC):**
   * MCC должен использоваться для оценки качества предсказаний системы, особенно в условиях несбалансированных данных. Требуемый уровень MCC должен составлять **не менее 0.55** .
3. **Скорость обработки запросов:**
   * Время обработки одного запроса (пары "резюме-вакансия") через API должно составлять **не более 2 секунды** . Это гарантирует комфортное взаимодействие пользователей с системой даже при пиковых нагрузках.
4. **Процент автоматической фильтрации:**
   * Система должна автоматически фильтровать **100% входящих резюме** , чтобы рекрутеры могли сосредоточиться только на анализе подходящих кандидатов.
5. **Экономическая эффективность:**
   * Система должна сократить затраты на рекрутмент путем снижения времени первичной фильтрации резюме. Ожидаемое снижение временных затрат рекрутеров должно составлять **не менее 60%** по сравнению с текущим процессом AS IS.
6. **Удовлетворенность пользователей:**
   * Опросы рекрутеров и соискателей должны показать повышение уровня удовлетворенности от использования системы. Целевой показатель — **не менее 85% положительных отзывов** от каждого типа пользователей.
7. **Объем обрабатываемых данных:**
   * Система должна поддерживать обработку до **10 000 записей в час** , сохраняя стабильное время ответа для каждого запроса. Это позволит работать с большими потоками данных без потери производительности.
8. **Процент успешного восстановления после сбоев:**
   * Система должна восстанавливаться после отказа любого компонента в течение **не более 15 минут** , что гарантирует минимальное влияние аварийных ситуаций на рабочие процессы.

#### **4.3 Обоснование выбора метрик**

Выбор метрик основан на необходимости обеспечить объективную оценку эффективности системы в реальных условиях эксплуатации. Например, метрика accuracy = 0.76 и MCC = 0.55 были получены в ходе тестирования прототипа на тестовых данных объемом **172 321 запись** (раздел 4.3). Эти показатели позволяют оценить точность предсказаний системы и её способность различать релевантные и нерелевантные резюме.

Метрика скорости обработки запросов (до **2 секунд** ) важна для обеспечения удобства использования системы всеми участниками процесса рекрутмента. Показатель экономической эффективности (снижение временных затрат рекрутеров на **60%** ) основан на сравнении текущего состояния процесса (AS IS) с целевым (TO BE), где автоматизация первичного анализа значительно уменьшает необходимость ручной работы.

Уровень удовлетворенности пользователей (не менее **85%** ) является критическим для принятия системы в качестве инструмента долгосрочного использования. Это гарантирует, что система будет не только функциональной, но и удобной для всех типов пользователей.

### **5 Общая операционная деятельность**

#### **5.1 Context (Контекст)**

1 Описание бизнес-процессов, которые система будет поддерживать

Система должна автоматизировать процесс первичного отбора кандидатов на основе обработки естественного языка из текстовых данных резюме и вакансий. Она должна интегрироваться в следующие ключевые бизнес-процессы рекрутмента:

* **Формирование пула кандидатов:** Система должна автоматически фильтровать резюме, основываясь на ключевых параметрах (опыт работы, образование, навыки), что позволит отделу кадров сократить время на первичную проверку.
* **Предварительная оценка соответствия:** Система должна проводить ранжирование резюме для каждой вакансии, выделяя наиболее подходящих кандидатов с вероятностью соответствия [0, 1]. Это упростит работу рекрутеров, позволяя им сосредоточиться на интервьюировании.
* **Интеграция с платформами рекрутмента:** Система должна взаимодействовать с внешними HR-платформами, такими как hh.ru и LinkedIn Talent Solutions, через RESTful API для автоматической загрузки резюме и отправки результатов анализа.

2 Взаимодействие системы с другими элементами инфраструктуры

Система должна тесно взаимодействовать с существующей IT-инфраструктурой организации, а также с внешними сервисами:

* Система должна использовать PostgreSQL для хранения структурированных данных о резюме и вакансиях, таких как ID, название должности, опыт работы и ключевые навыки.
* Для хранения векторных представлений текстов (embeddings) система должна интегрироваться с Weaviate, что обеспечит быстрый поиск ближайших соседей в семантическом пространстве.
* Все данные, передаваемые между компонентами системы или внешними сервисами, должны шифроваться с использованием TLS 1.3 для защиты при передаче и AES-256 для хранения, чтобы соответствовать требованиям GDPR и ФЗ-152 "О персональных данных".

Также система должна взаимодействовать с корпоративными мессенджерами и email-серверами для отправки уведомлений рекрутерам и соискателям о состоянии обработки резюме.

#### **5.2 Systems (Системы)**

1 Существующие системы

* **HR-платформы:** hh.ru, LinkedIn Talent Solutions — источники данных для загрузки резюме и вакансий. Система должна быть способна получать данные через их API.
* **Базы данных:** PostgreSQL используется для хранения структурированных данных, таких как метаданные о резюме и вакансиях. MongoDB применяется для временного хранения полуструктурированных JSON-данных до их преобразования.

2 Планируемые системы

* **Vector Database:** Weaviate должна быть внедрена для хранения векторных представлений текстов резюме и вакансий, что позволит эффективно находить ближайших соседей при сопоставлении.
* **Queue Management System:** RabbitMQ или Kafka должны быть использованы для управления очередями запросов при высоких нагрузках, что гарантирует отказоустойчивость системы.
* **Monitoring Tools:** Prometheus и Grafana должны быть интегрированы для постоянного контроля производительности микросервисов и состояния баз данных.

#### **5.3 Organizational Unit (Организационные единицы)**

1 Роли подразделений в использовании системы

* **Отдел кадров:**
  + Рекрутеры должны создавать и редактировать вакансии через пользовательский интерфейс или API.
  + Они должны получать ранжированный список кандидатов для каждой вакансии с детализацией причин ранжирования.
* **IT-отдел:**
  + Должен обеспечивать развертывание и поддержку серверной инфраструктуры, включая Docker/Kubernetes для контейнеризации микросервисов.
  + Должен настраивать и поддерживать базы данных PostgreSQL, Weaviate и MongoDB, а также механизмы шифрования данных.
* **Аналитический отдел:**
  + Должен контролировать метрики качества классификации (accuracy, MCC) для оценки эффективности работы алгоритмов.
  + Научные сотрудники должны иметь возможность проводить fine-tuning модели BERT для адаптации под специфику компании или рынка труда.
* **Менеджмент:**
  + Менеджеры по найму должны использовать систему для оценки её влияния на общую эффективность рекрутмента.
  + Они должны принимать решения о дальнейшем развитии функциональности системы на основе экономических показателей (например, снижение затрат на рекрутмент).

### **6 Управление**

#### **6.1 Политики управления**

В данном разделе описываются политики управления разрабатываемой системой автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Эти политики определяют принципы принятия решений, приоритеты при внедрении системы и подходы к управлению её функционированием.

**1. Приоритет безопасности над скоростью внедрения:** Система должна удовлетворять требованиям безопасности данных выше всего остального, даже если это замедлит процесс внедрения. Это связано с тем, что система будет работать с конфиденциальной информацией соискателей и работодателей, требующей защиты согласно нормативным документам (GDPR, ФЗ-152 "О персональных данных"). Например:

* Все данные должны шифроваться при передаче через TLS 1.3 и храниться с использованием AES-256.
* Внедрение новых функций должно проходить дополнительное тестирование на безопасность до их активации для пользователей.

**2. Принцип прозрачности решений:** Система должна предоставлять рекрутерам детальную информацию о причинах ранжирования каждого кандидата, чтобы обеспечить доверие к автоматизированному процессу отбора. Например:

* Система должна объяснять совпадения между навыками кандидата и требованиями вакансии.
* Для каждой пары "резюме-вакансия" система должна генерировать отчет о ключевых параметрах соответствия (опыт работы, образование, жесткие и мягкие навыки).

**3. Приоритет точности анализа над производительностью:** Система должна удовлетворять требованиям высокой точности анализа текстовых данных даже в ущерб скорости обработки запросов. На основе экспериментальных данных было установлено, что модель BERT показывает лучшие результаты по метрике accuracy = 0.76 и MCC = 0.55, но требует значительных вычислительных мощностей. Например:

* Если доступны GPU, то система должна использовать их для повышения точности анализа.
* В случае недоступности GPU система должна переходить на CPU, но предупреждать пользователя о возможном снижении скорости обработки запросов.

**4. Политика масштабируемости:** Система должна удовлетворять требованиям масштабируемости для поддержания стабильной работы при увеличении объема данных. Это достигается через использование микросервисной архитектуры и инструментов горизонтального масштабирования, таких как Kubernetes. Например:

* При достижении пороговой нагрузки (например, более 10 000 записей в час) система должна автоматически добавлять новые экземпляры микросервисов для обработки запросов.
* Механизмы очередей сообщений (RabbitMQ или Kafka) должны использоваться для управления пиковыми нагрузками и обеспечения отказоустойчивости.

**5. Политика регулярного мониторинга:** Система должна поддерживать постоянный мониторинг всех компонентов для своевременного выявления проблем и оптимизации производительности. Например:

* Система должна использовать инструменты Prometheus и Grafana для сбора метрик состояния микросервисов, баз данных и моделей машинного обучения.
* Администратор должен получать уведомления о любых критических сбоях или нештатных ситуациях через email или корпоративные мессенджеры.

**6. Политика обновления данных:** Система должна удовлетворять требованиям регулярного обновления данных для поддержания актуальности информации о кандидатах и вакансиях. Например:

* Система должна автоматически проверять дату последнего обновления данных вакансий и резюме, чтобы исключить работу с устаревшей информацией.
* При изменении параметров вакансии система должна немедленно пересчитывать соответствие всех ранее загруженных резюме для этой вакансии.

**7. Политика управления доступом:** Система должна следовать принципам ролевой модели управления доступом (RBAC) для обеспечения контроля над данными. Например:

* Рекрутеры должны иметь доступ только к своим вакансиям и связанным с ними резюме.
* Администраторы должны иметь полный доступ ко всем данным для целей администрирования и анализа эффективности системы.

**8. Политика сохранения логов событий:** Система должна удовлетворять требованиям сохранения логов событий для анализа производительности и качества работы алгоритмов. Например:

* Все действия пользователей и операторов должны быть задокументированы в логах с указанием времени, типа операции, ID пользователя и статуса выполнения.
* Логи должны храниться в зашифрованном виде для защиты от несанкционированного доступа.

#### **6.2 Организация**

В данном разделе описываются ответственные лица и организационные структуры, которые должны управлять процессом разработки, внедрения и эксплуатации системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий.

**1. Steering-комитет проекта:**

* Система должна работать под контролем **steering-комитета** , который определяет стратегические направления развития сервиса и координирует действия между отделами организации.
* В состав steering-комитета должны войти представители следующих подразделений:
  + Руководство HR-отдела для обеспечения соответствия системы потребностям рекрутмента.
  + IT-департамент для технического сопровождения и интеграции с существующей инфраструктурой.
  + Аналитический отдел для оценки эффективности алгоритмов и качества анализа данных.

**2. Ответственные лица:**

2.1. **Руководитель проекта:**

* Руководитель проекта должен нести ответственность за общее руководство и контроль выполнения задач по созданию и внедрению системы.
* Он должен регулярно предоставлять отчеты о прогрессе работы перед steering-комитетом.

2.2. **HR-специалисты:**

* HR-специалисты должны принимать активное участие в формировании требований к системе, чтобы она соответствовала реальным бизнес-процессам рекрутмента.
* Они должны тестировать функциональность системы на этапах разработки и давать обратную связь разработчикам для улучшения её работы.

2.3. **IT-специалисты:**

* IT-специалисты должны обеспечивать техническую реализацию и поддержку системы, включая настройку серверной инфраструктуры, баз данных и инструментов безопасности.
* Они должны быть ответственны за выбор технологий, таких как PostgreSQL для хранения структурированных данных и Weaviate для управления векторными представлениями текстов.

2.4. **Научные сотрудники:**

* Научные сотрудники должны заниматься разработкой и адаптацией моделей обработки естественного языка, таких как BERT, для достижения высокой точности классификации резюме.
* Они должны проводить регулярные эксперименты для повышения метрик accuracy (не менее 0.76) и MCC (не менее 0.55).

**3. Организационные единицы:**

3.1. **Отдел кадров:**

* Должен использовать систему для автоматической фильтрации резюме и получения ранжированных списков кандидатов для каждой вакансии.
* Должен предоставлять данные о текущих процессах рекрутмента (AS IS) для их сравнения с новыми процессами (TO BE), которые будут реализованы после внедрения системы.

3.2. **Административный отдел:**

* Должен следить за соблюдением нормативных требований GDPR и ФЗ-152 "О персональных данных" при работе с системой.
* Должен обеспечивать юридическую защиту компании от возможных претензий со стороны соискателей или регуляторов.

3.3. **Отдел разработки ПО:**

* Должен разрабатывать и поддерживать программное обеспечение, используя микросервисную архитектуру и RESTful API для взаимодействия с внешними платформами рекрутмента.
* Должен обеспечивать масштабируемость системы через Kubernetes и Docker для поддержания производительности при больших нагрузках.

3.4. **Отдел тестирования:**

* Должен проводить функциональное, нагрузочное и интеграционное тестирование системы для выявления недостатков и предложений по улучшению.
* Должен проверять соответствие системы требованиям безопасности и конфиденциальности данных.

#### **6.3 Инвестиционный план**

В данном разделе описывается инвестиционный план для разработки и внедрения системы автоматизации отбора кандидатов на основе обработки естественного языка из резюме и вакансий. Этот план включает бюджет, сроки окупаемости и источники финансирования проекта.

**1. Бюджет:**

Система должна быть реализована с учетом следующих затрат:

1.1. **Затраты на аппаратное обеспечение:**

* Аренда серверов с GPU для обучения модели BERT: **50 000 рублей в месяц** .
* Приобретение или аренда хранилищ данных для PostgreSQL и Weaviate: **30 000 рублей в месяц** .

1.2. **Затраты на программное обеспечение:**

* Лицензирование инструментов разработки (например, PyCharm Professional): **10 000 рублей в год** .
* Использование облачных сервисов для хранения данных (например, Google Cloud или AWS): **20 000 рублей в месяц** .

1.3. **Затраты на персонал:**

* Оплата труда разработчиков и научных сотрудников: **300 000 рублей в месяц** .
* Обучение HR-специалистов работе с системой: **20 000 рублей на каждого рекрутера** (примерно 10 человек = **200 000 рублей единовременно** ).

1.4. **Итоговый бюджет на первый год разработки и внедрения:**

* Общие затраты составляют около **4 000 000 рублей** (включая разработку, обучение и эксплуатацию).

**2. Сроки окупаемости:**

Система должна демонстрировать экономический эффект уже через **6 месяцев после внедрения** , что подтверждается расчетами на основе текущих процессов рекрутмента (AS IS) и предполагаемых изменений (TO BE).

2.1. **Экономическая эффективность:**

* Внедрение системы позволит сократить время первичного анализа резюме на **60%** , что эквивалентно экономии примерно **10 часов работы одного рекрутера в неделю** .
* Если в компании работает 10 рекрутеров, то общая экономия времени составляет **400 часов в месяц** , что можно перевести в финансовую выгоду, исходя из средней зарплаты рекрутера (например, 1 час работы стоит **1 000 рублей** ). Это даст экономию **400 000 рублей в месяц** .

2.2. **Ожидаемые сроки окупаемости:**

* При начальных инвестициях в размере **4 000 000 рублей** и ежемесячной экономии **400 000 рублей** срок окупаемости проекта составит **10 месяцев** .

2.3. **Дополнительные преимущества:**

* Повышение качества найма за счет точного сопоставления навыков кандидатов с требованиями вакансий (accuracy = 0.76, MCC = 0.55). Это может привести к снижению процента неудачных наймов и уменьшению затрат на повторный поиск персонала.

**3. Источники финансирования:**

3.1. **Внутренние ресурсы организации:**

* Финансирование должно осуществляться за счет внутренних средств компании, выделенных для цифровой трансформации бизнес-процессов.

3.2. **Гранты и программы поддержки:**

* Организация должна рассмотреть возможность получения грантов от государственных или частных фондов, поддерживающих развитие технологий в области HR-tech.

3.3. **Партнерские соглашения:**

* Компания должна заключить партнерские соглашения с внешними HR-платформами (например, hh.ru), которые могут профинансировать часть проекта в обмен на приоритетную интеграцию своей платформы с разрабатываемой системой.

#### **6.4 Управление информационными активами**

Управление информационными активами направлено на обеспечение безопасности, целостности и эффективного использования данных в системе автоматизации отбора кандидатов. Система должна гарантировать защиту конфиденциальной информации, включая персональные данные кандидатов и вакансий, а также поддерживать их актуальность и доступность для авторизованных пользователей.

**Требования к системе:**

1. **Система должна обеспечивать шифрование данных** на всех этапах их передачи и хранения с использованием современных протоколов (например, TLS для передачи, AES-256 для хранения) для защиты от несанкционированного доступа.
2. **Система должна реализовывать ролевую модель управления доступом (RBAC)** , ограничивая права пользователей в соответствии с их ролями (рекрутер, администратор, кандидат).
3. **Система должна анонимизировать персональные данные** в резюме и вакансиях, исключая возможность идентификации кандидатов по текстовым описаниям (например, удаление ФИО, контактных данных).
4. **Система должна хранить структурированные данные в реляционной СУБД (PostgreSQL)** с поддержкой сложных запросов и JSON-данных, а векторные представления текстов — в специализированных СУБД (Weaviate) для обеспечения высокой производительности поиска.
5. **Система должна обеспечивать защиту от утечек данных** через механизмы аудита, мониторинга подозрительных действий и автоматического блокирования неавторизованных запросов.
6. **Система должна поддерживать резервное копирование данных** и восстановление в случае сбоев, гарантируя минимальное время простоя (не более 15 минут) и потерю не более 1% данных.
7. **Система должна соответствовать требованиям GDPR и законодательству РФ** в части обработки персональных данных, включая получение согласия пользователей и предоставление инструментов для удаления данных.
8. **Система должна обеспечивать жизненный цикл данных** , включая их архивирование по истечении срока хранения (например, удаление резюме через 3 года) и автоматическую очистку временных файлов.

**Политики использования данных:**

* Все данные, включая резюме и вакансии, должны использоваться исключительно для целей подбора персонала.
* Доступ к данным разрешен только сотрудникам, прошедшим обучение по вопросам безопасности и конфиденциальности.
* Система должна предоставлять отчеты о доступе к данным и изменениях в них для обеспечения прозрачности и аудита.

**6.5 Безопасность**Раздел определяет стратегию обеспечения безопасности системы автоматизации отбора кандидатов, гарантируя защиту данных, соответствие регуляторным требованиям и минимизацию рисков утечек конфиденциальной информации.

### **1 Политики защиты данных**

1. **Шифрование данных**
   * Система должна использовать шифрование **AES-256** для хранения данных и **TLS 1.3** для их передачи, обеспечивая защиту на всех этапах обработки.
   * Ключи шифрования должны храниться в защищённом аппаратном модуле (HSM) и обновляться ежегодно.
2. **Управление доступом**
   * Система должна реализовывать **ролевую модель RBAC** с динамическим назначением прав:
     + Рекрутеры: доступ только к анонимизированным резюме и вакансиям.
     + Администраторы: полный доступ к данным с двухфакторной аутентификацией.
     + Кандидаты: доступ исключительно к своим данным.
   * Все действия пользователей должны логироваться с привязкой к IP-адресу и времени.
3. **Анонимизация данных**
   * Система должна автоматически удалять или заменять ФИО, контактные данные и другие персональные идентификаторы в резюме и вакансиях.
   * Для анализа допускается использование только обезличенных данных (например, навыки, опыт работы в годах).
4. **Резервное копирование**
   * Система должна выполнять ежедневное резервное копирование данных в географически распределённые хранилища.
   * Время восстановления данных после сбоя не должно превышать **15 минут** с потерей не более **1% информации** .

### **2 Соответствие регуляторным требованиям**

1. **GDPR и законодательство РФ**
   * Система должна соответствовать требованиям **GDPR** (защита персональных данных ЕС) и **ФЗ-152** (Россия), включая:
     + Получение явного согласия кандидатов на обработку данных.
     + Предоставление инструментов для удаления данных по запросу («право на забвение»).
   * Хранение персональных данных должно быть ограничено сроком в **3 года** с момента последнего взаимодействия.
2. **Аудит и сертификация**
   * Система должна поддерживать автоматическую генерацию отчётов для внутренних и внешних аудитов.
   * Интеграция с инструментами **SIEM** (например, Splunk) для мониторинга подозрительных действий.
3. **Защита от утечек**
   * Система должна блокировать запросы, нарушающие политики доступа, и уведомлять администратора в режиме реального времени.
   * Регулярное тестирование на проникновение (**pentest** ) не реже 1 раза в квартал.

### **3 Требования к реализации**

1. Система должна обеспечивать изоляцию сред разработки, тестирования и production.
2. Все внешние API должны быть защищены токенами OAuth 2.0 с ограниченным сроком действия.
3. Для анализа текстовых данных разрешено использовать только обезличенные эмбеддинги (BERT, Word2Vec).

**6.6 План непрерывности бизнеса**План непрерывности бизнеса обеспечивает устойчивость системы автоматизации отбора кандидатов к сбоям и внешним воздействиям, гарантируя минимальное прерывание работы и сохранение данных. Основная цель — минимизировать риски потери информации, обеспечить быстрое восстановление функционала и соблюдение регуляторных требований в аварийных ситуациях.

### **1 Требования к системе**

1. **Система должна поддерживать географически распределённое резервное копирование данных** в трёх независимых хранилищах с синхронизацией в реальном времени.
2. **Система должна обеспечивать автоматическое восстановление данных за 15 минут** с потерей не более 1% информации в случае сбоя.
3. **Система должна реализовывать автоматическое переключение на резервные серверы** при недоступности основного узла, используя технологии контейнеризации (Docker) и оркестрации (Kubernetes).
4. **Система должна интегрироваться с инструментами мониторинга (например, Prometheus, Grafana)** для оперативного обнаружения сбоев и уведомления администраторов.
5. **Система должна сохранять изоляцию сред разработки, тестирования и production** , предотвращая каскадные сбои между ними.

### **2 Процедуры восстановления**

1. **Автоматическое восстановление:**
   * При сбое основного сервера система должна переключиться на резервный узел в течение 2 минут.
   * Репликация данных в Weaviate и PostgreSQL должна выполняться асинхронно с задержкой не более 5 секунд.
2. **Ручное восстановление:**
   * Администраторы должны иметь доступ к резервным копиям за последние 30 дней для восстановления данных вручную.
   * Восстановление из резервной копии должно занимать не более 1 часа с момента начала процедуры.
3. **Обработка инцидентов:**
   * Все сбои должны логироваться с указанием времени, типа ошибки и затронутых компонентов.
   * После устранения сбоя система должна автоматически отправлять отчёт о произошедшем инциденте в систему аудита (например, SIEM).

### **3 Требования к документированию**

1. Система должна сохранять журнал инцидентов и действий по их устранению для последующего анализа и аудита.
2. Документация по восстановлению должна быть доступна в защищённом репозитории и обновляться при каждом изменении инфраструктуры.

**6.7 Соответствие**Подраздел определяет требования к соблюдению системой нормативных актов, отраслевых стандартов и внутренних политик организации, что гарантирует легитимность, безопасность и совместимость системы в условиях регулируемой среды.

### **1 Нормативные требования**

1. **Система должна соответствовать GDPR** (Общий регламент по защите данных ЕС):
   * Обеспечивать получение явного согласия кандидатов на обработку персональных данных.
   * Гарантировать «право на забвение» через механизмы удаления данных по запросу.
2. **Система должна соблюдать требования ФЗ-152** (Федеральный закон РФ «О персональных данных»):
   * Локализовать хранение персональных данных российских кандидатов на территории РФ.
   * Шифровать данные в соответствии с ФСТЭК России.
3. **Система должна поддерживать стандарты безопасности ISO/IEC 27001** для управления информационной безопасностью, включая:
   * Регулярные аудиты.
   * Контроль доступа к данным.
4. **Система должна соответствовать NIST SP 800-53** (стандарты безопасности США) для защиты данных в международных сценариях использования.

### **2 Соответствие стандартам разработки**

1. **Система должна разрабатываться в соответствии с IEEE 1016** (стандарт документирования ПО):
   * Подробно описывать архитектуру, компоненты и интерфейсы.
2. **Система должна соответствовать ISO/IEC 25010** (стандарт качества ПО):
   * Обеспечивать функциональную полноту, производительность, совместимость и надежность.

### **3 Требования к данным и инфраструктуре**

1. **Система должна использовать PostgreSQL** для хранения структурированных данных:
   * Соответствовать ACID-транзакциям.
   * Поддерживать JSON-типы данных для гибкости.
2. **Система должна интегрироваться с Weaviate** (векторная СУБД):
   * Обеспечивать поиск по векторным представлениям текстов с точностью ≥95%.
   * Сохранять совместимость с моделями BERT и Word2Vec.
3. **Система должна соблюдать стандарты OWASP Top 10** для защиты от уязвимостей веб-приложений.

### **4 Аудит и сертификация**

1. **Система должна поддерживать автоматическую генерацию отчетов** для аудита:
   * Журналы доступа к данным.
   * История изменений в резюме и вакансиях.
2. **Система должна проходить ежегодную сертификацию** на соответствие GDPR и ФЗ-152.