Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Волгоградский государственный технический университет»

|  |  |
| --- | --- |
| Факультет | Электроники и вычислительной техники |
| Кафедра | Программное обеспечение автоматизированных систем |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Согласовано | | | | | | | | |  | Утверждаю | | | | | | | | |
|  | | | | | | | | |  | и.о. зав. кафедрой | | | | | | | | |
| (должность гл. специалиста предприятия) | | | | | | | | |  |
|  | | | |  |  | | | |  |  | | | |  | О. А. Сычев | | | |
| (подпись) | | | |  | (инициалы, фамилия) | | | |  | (подпись) | | | |  | (инициалы, фамилия) | | | |
| « |  | » |  | | | 20 |  | г. |  | « |  | » |  | | | 20 |  | г. |

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| к | выпускной квалификационной работе бакалавра | | | | | | | | | | | | | | | на тему |
| (наименование вида работы) | | | | | | | | | | | | | | | | |
| Разработка веб-сервиса для анализа и сравнения условий аренды | | | | | | | | | | | | | | | | |
| недвижимости | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | |
|  | |  | | | | | |  |  | | | | | | | |
| Автор | |  | | | | | |  | Аврамова Елизавета Владимировна | | | | | | | |
|  | | (подпись и дата подписания) | | | | | |  | (фамилия, имя, отчество) | | | | | | | |
| Обозначение | | | ВКРБ–09.03.04–10.19–01–25 | | | | | | |  | | | | | | |
|  | | | (код документа) | | | | |  | | | | | | | | |
| Группа | | | ПрИн-467 | | | | |  | | | | | | | | |
|  | | | (шифр группы) | | | | |  | | | | | | | | |
| Направление | | | 09.03.04 – Программная инженерия,  Разработка программно-информационных систем | | | | | | | | | | | | | |
|  | | | (код и наименование направления, наименование программы (профиля)) | | | | | | | | | | | | | |
| Руководитель работы | | | | |  | | | | | | | |  | | Гилка В.В. | |
|  | | | | | (подпись и дата подписания) | | | | | | | |  | | (инициалы и фамилия) | |
| Консультанты по разделам: | | | | | | | | | | | | | | | | |
|  | | | | | |  |  | | | | |  | |  | | |
| (краткое наименование раздела) | | | | | |  | (подпись и дата подписания) | | | | |  | | (инициалы и фамилия) | | |
|  | | | | | |  |  | | | | |  | |  | | |
| (краткое наименование раздела) | | | | | |  | (подпись и дата подписания) | | | | |  | | (инициалы и фамилия) | | |
| Нормоконтролер: | | | |  | | | | | | |  | Кузнецова А.С. | | | | |
|  | | | | (подпись и дата подписания) | | | | | | |  | (инициалы и фамилия) | | | | |

Волгоград 2025 г.

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Волгоградский государственный технический университет»

|  |  |
| --- | --- |
| Кафедра | Программное обеспечение автоматизированных систем |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Утверждаю | | | | | и.о. зав. кафедрой | | | |
|  |  | | | |  | О. А. Сычев | | | |
| (подпись) | | | |  | (инициалы, фамилия) | | | |
|  | « |  | » |  | | | 20 |  | г. |

**Задание**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| на | выпускную квалификационную работу бакалавра | | | | | | | | | | | | | | |
|  | (наименование вида работы) | | | | | | | | | | | | | | |
| Студент | | | Аврамова Елизавета Владимировна | | | | | | | | | | | | |
|  | (фамилия, имя, отчество) | | | | | | | | | | | | | | |
| Код кафедры | | | | 10.19 | Группа | | | | ПрИн-467 | | | |  | | |
| Тема | | Разработка веб-сервиса для анализа и сравнения условий аренды | | | | | | | | | | | | | |
| недвижимости | | | | | | | | | | | | | | | |
| Утверждена приказом по университету | | | | | | « | 23 | » | | августа | 20 | 24 | | г. № | 1105-ст |
| Срок представления готовой работы (проекта) | | | | | | | | | |  | | | | | |
|  | | | | | | | | | | (дата, подпись студента) | | | | | |
| Исходные данные для выполнения работы (проекта) | | | | | | | | | | | | | | | |
| Задание, выданное научным руководителем кафедры «ПОАС» | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
| Содержание основной части пояснительной записки | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | |

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

|  |  |
| --- | --- |
| Перечень графического материала | |
| 1) |  |
|  | |
| 2) |  |
|  | |
| 3) |  |
|  | |
| 4) |  |
|  | |
| 5) |  |
|  | |
| 6) |  |
|  | |
| 7) |  |
|  | |
| 8) |  |
|  | |
| 9) |  |
|  | |
| 10) |  |
|  | |
| 11) |  |
|  | |
| 12) |  |
|  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Руководитель работы (проекта) | |  | |  | | Гилка В.В. | |
|  | | (подпись и дата подписания) | |  | | (инициалы и фамилия) | |
| Консультанты по разделам: | |  | | | |  | |
|  |  | |  | |  | |  |
| (краткое наименование раздела) |  | | (подпись и дата подписания) | |  | | (инициалы и фамилия) |
|  |  | |  | |  | |  |

Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Кафедра «Программное обеспечение автоматизированных систем»

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ:  и.о. зав. кафедрой ПОАС  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_О.А. Сычев  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 г. |

Разработка веб-сервиса для анализа и сравнения условий аренды недвижимости

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

ВКРБ–09.03.04–10.19–01–25–81

Листов 65

|  |  |
| --- | --- |
|  | Руководитель работы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Гилка В.В.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_20 г. |
| |  | | --- | | Нормоконтролер  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кузнецова А.С.  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20 г. | | |  | | --- | | Исполнитель  студент группы ПрИн-467  \_\_\_\_\_\_\_\_\_\_\_\_Аврамова Е. В. «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 20 г. | |

Волгоград, 2025 г.

Аннотация

Настоящий документ является пояснительной запиской к выпускной квалификационной работе бакалавра на тему: «Разработка веб-сервиса для анализа и сравнения условий аренды недвижимости».

В работе обосновывается актуальность выбранной темы, проводится ….

Документ включает в себя страниц -…, рисунков - …., приложений -...

Ключевые слова: …..

Содержание

[Введение 8](#_Toc181960321)

[1 Анализ современного состояния вопроса в области анализа и сравнения условий аренды недвижимости 10](#_Toc181960322)

[1.1 Теоретические основы управления недвижимостью 10](#_Toc181960323)

[1.2 Анализ современных методов анализа и сравнения условий аренды квартир 10](#_Toc181960324)

[1.3 Обзор существующих сервисов для анализа и сравнения условий аренды квартир 10](#_Toc181960325)

[1.4 Проблемы и недостатки существующих решений для анализа и сравнения условий аренды квартир 10](#_Toc181960326)

[1.5 Описание существующего процесса решения задачи для анализа и сравнения условий аренды квартир 10](#_Toc181960327)

[2 Теоретические аспекты разработки веб-сервиса для анализа и сравнения условий аренды квартир 11](#_Toc181960328)

[2.1 Предлагаемый процесс решения задачи анализа и сравнения условий аренды квартир 11](#_Toc181960329)

[2.2 Формальная модель для описания проблемы анализа и сравнения условий аренды квартир 11](#_Toc181960330)

[2.3 Описание алгоритма для анализа и сравнения условий аренды квартир 11](#_Toc181960331)

[2.4 Разработка функциональных требований к веб-сервису для анализа и сравнения условий аренды квартир 11](#_Toc181960332)

[Выводы 11](#_Toc181960333)

[3 Проектирование и разработка веб-сервиса для анализа и сравнения условий аренды квартир 12](#_Toc181960334)

[3.1 Выбор технологий и инструментов для разработки веб-сервиса анализа и сравнения условий аренды квартир 12](#_Toc181960335)

[3.2 Проектирование интерфейса пользователя веб-сервиса для анализа и сравнения условий аренды квартир 12](#_Toc181960336)

[3.3 Архитектура веб-сервиса для анализа и сравнения условий аренды квартир 12](#_Toc181960337)

[3.4 Интеграция с финансовыми платформами для анализа и сравнения условий аренды квартир 12](#_Toc181960338)

[Выводы 12](#_Toc181960339)

[4 Тестирование и апробация веб-сервиса для анализа и сравнения условий аренды квартир 13](#_Toc181960340)

[4.1 Методологии тестирования и оценки эффективности веб-сервиса для анализа и сравнения условий аренды квартир 13](#_Toc181960341)

[4.2 Тестирование веб-сервиса для анализа и сравнения условий аренды квартир 13](#_Toc181960342)

[4.3 Оценка эффективности веб-сервиса для анализа и сравнения условий аренды квартир 13](#_Toc181960343)

[4.4 Анализ результатов тестирования и оценки эффективности веб-сервиса для анализа и сравнения условий аренды квартир 13](#_Toc181960344)

[Выводы 13](#_Toc181960345)

[Заключение 13](#_Toc181960346)

[Список использованных источников 15](#_Toc181960347)

[Приложение А 16](#_Toc181960348)

[Справка о результатах проверки выпускной квалификационной работы на наличие заимствований 16](#_Toc181960349)

[Приложение Б 17](#_Toc181960350)

[Техническое задание 17](#_Toc181960351)

[Приложение В 18](#_Toc181960352)

[Руководство системного программиста 18](#_Toc181960353)

# Введение

# Аренда квартир в условиях современного городского рынка становится неотъемлемой частью жизни мобильного населения. Повышенная урбанизация, экономическая нестабильность и рост численности трудоспособного населения усиливают спрос на арендуемое жилье. В то же время, разнообразие предложений, разброс цен и отсутствие прозрачных инструментов анализа значительно затрудняют поиск подходящего варианта для арендатора.

# В процессе выбора жилья арендаторы сталкиваются с рядом проблем: дублирующиеся объявления на разных платформах, неполная информация, устаревшие данные, недостаточные фильтры, отсутствие наглядного сравнения. Всё это приводит к временным потерям и снижению эффективности принятия решений.

# Целью данной выпускной квалификационной работы является сокращение временных затрат арендаторов на поиск и анализ условий аренды квартир за счёт создания веб-сервиса, предоставляющего инструменты фильтрации, сравнения и анализа объявлений.

# Для достижения цели поставлены следующие задачи:

# исследовать текущую ситуацию на рынке аренды жилья и проанализировать цифровые инструменты, используемые в этой области;

# изучить существующие методы фильтрации и сравнения объявлений;

# выявить слабые стороны актуальных решений (ЦИАН, Авито и др.);

# разработать архитектуру веб-сервиса с распределённой системой сбора данных и пользовательским интерфейсом;

# реализовать модуль парсинга с учётом устойчивости к блокировкам и изменениям структуры сайтов;

# внедрить гибкую систему фильтрации по ключевым параметрам (цена, площадь, район и др.);

# обеспечить сравнение объектов по совокупности признаков с учётом предпочтений пользователя;

# реализовать экспорт объявлений со ссылками на оригинальные источники;

# провести тестирование, верификацию и апробацию разработанного решения.

# Объект исследования — процесс аренды жилья как элемент жилищного рынка.

# Предмет исследования — методы автоматизированного анализа и сравнения условий аренды, а также технологии реализации соответствующего веб-сервиса.

# Методы исследования — системный анализ, объектно-ориентированное проектирование, веб-программирование, парсинг данных, проектирование человеко-машинного взаимодействия с учётом эвристик Нильсена.

# Практическая значимость заключается в том, что разработанный сервис позволит пользователям эффективно выбирать квартиры на основе объективных данных, сокращая время поиска и снижая риски неудачного выбора жилья.

# 1 Анализ современного состояния вопроса в области анализа и сравнения условий аренды квартир

Разработка цифрового инструмента для поиска и анализа предложений по аренде жилья требует чёткого понимания как самого рынка аренды, так и существующих технических решений, применяемых в этой сфере. Аренда жилья — это не просто подбор квартиры по цене и метражу; это сложный процесс, включающий множество параметров и контекстов, которые важно учитывать при проектировании автоматизированной системы поддержки пользователя.

Современные цифровые платформы призваны упростить процесс выбора жилья, но по факту они не всегда справляются с задачей комплексного анализа предложений. Нередко пользователи сталкиваются с повторяющейся, неструктурированной или устаревшей информацией, а система фильтрации не охватывает все потребности целевой аудитории. Это создаёт предпосылки для разработки более гибких, интеллектуальных и пользовательски-ориентированных решений.

Настоящий раздел посвящён системному обзору ключевых характеристик предметной области, методов анализа данных, действующих платформ, их преимуществ и недостатков. Также в нём раскрываются основные процессы, связанные со сбором, обработкой и представлением информации, и обосновывается необходимость внедрения нового подхода — агрегирующего, фильтрующего и сравнивающего предложения аренды с учётом пользовательских предпочтений и технических ограничений источников.

Рынок аренды жилой недвижимости активно развивается, но современные цифровые сервисы не обеспечивают необходимого уровня интеллектуального анализа и сравнения. Проблемы дублирования, неактуальности и ограниченности фильтров остаются нерешёнными. Предлагаемый веб-сервис учитывает эти недостатки, внедряя гибкую фильтрацию, парсинг с защитой от блокировок, устранение дублей и пользовательские сценарии на основе UX-принципов Нильсена. Всё это позволяет добиться главной цели проекта — сократить время арендатора на поиск и повысить обоснованность выбора жилья.

1.1 Особенности предметной области: аренда жилья в цифровую эпоху

Рынок аренды жилой недвижимости в России претерпел существенные изменения за последние годы. Если раньше основной способ поиска жилья заключался в личных объявлениях, агентствах или рекомендациях знакомых, то сегодня этот процесс почти полностью перешёл в цифровую среду. Онлайн-платформы стали главными посредниками между арендодателями и арендаторами.

Аренда жилья имеет ряд специфических характеристик. С точки зрения пользователя (арендатора), важно не только наличие квартиры, но и её соответствие ряду параметров: стоимость, срок аренды, расположение (район), метраж, количество комнат, этаж, тип дома, возможность проживания с детьми или животными, наличие бытовой техники и мебели. При этом каждый пользователь может иметь индивидуальные приоритеты, что делает задачу выбора квартиры многокритериальной и заведомо субъективной.

Для арендодателя, в свою очередь, важно быстро найти надёжного жильца и выгодно выделиться на фоне конкурентов. Поэтому платформа, на которой происходит взаимодействие, должна обеспечивать качественную подачу информации, прозрачность условий, актуальность объявлений и понятный пользовательский интерфейс.

Несмотря на цифровизацию, пользователи сталкиваются с рядом трудностей:

одно и то же объявление может размещаться на разных платформах с разными параметрами;

информация может быть устаревшей или неполной (отсутствуют фото, этаж, техника и т.п.);

интерфейс большинства сайтов не предоставляет инструментов для сравнения или сохранения истории поиска;

фильтрация часто ограничена только базовыми параметрами;

многие сайты перегружены рекламой, что мешает принятию решений.

В рамках данной работы предметная область анализируется с учётом задач арендатора, которому важно:

минимизировать время поиска;

иметь уверенность в актуальности данных;

быстро сравнивать предложения по значимым параметрам;

переходить на оригинальные сайты для связи с владельцами.

Понимание этих аспектов легло в основу проектирования функциональности веб-сервиса и выбора архитектурных решений, направленных на упрощение и ускорение процесса аренды с использованием цифровых технологий.

# 1.2 Анализ современных методов анализа и сравнения условий аренды недвижимости

Современные методы обработки информации об аренде базируются на системах фильтрации и агрегирования данных. На практике применяются алгоритмы сравнения объектов по ключевым параметрам, включая цену, площадь, количество комнат, этаж, а также наличие дополнительных условий — мебели, техники, допуска проживания с животными или детьми. Основным принципом остаётся фильтрация по критериям, заданным пользователем, с последующей сортировкой подходящих объектов.

Однако классические методы фильтрации, реализованные на большинстве платформ, часто не позволяют учитывать комплексные пользовательские предпочтения. Сравнение объектов также производится вручную — пользователь вынужден сам анализировать предложения, переходить между вкладками, сопоставлять данные. В некоторых зарубежных решениях используются алгоритмы машинного обучения для прогнозирования предпочтений или оценки привлекательности объекта, но в большинстве российских сервисов подобные подходы пока отсутствуют.

Дополнительным направлением в анализе является использование визуализаций: карт, диаграмм, интерактивных сравнительных таблиц. Такие методы улучшают восприятие, но также требуют точных и актуальных данных, что затруднено в условиях нестабильного парсинга и ручного ввода объявлений.

1.3 Обзор существующих сервисов для анализа и сравнения условий аренды недвижимости

# Российский рынок предлагает несколько основных сервисов для поиска аренды:

# ЦИАН — один из крупнейших агрегаторов. Обладает широким набором фильтров и аналитикой по регионам. Однако пользователи часто жалуются на дубль-контент, неактуальные объявления и отсутствие функции сравнения.

# Авито — мультиресурс, включающий разделы недвижимости. Имеет обширную базу, но интерфейс ориентирован на размещение, а не на анализ и сравнение. Механизм фильтрации простой и часто недостаточный для специфических задач арендатора.

# Яндекс.Недвижимость — предоставляет интеграцию с картой и удобную навигацию. Сервис включает статистику и локальную аналитику, но также ограничен в функциональности по сравнению с требованиями профессионального пользователя.

# Домофонд — предлагает минимальный функционал фильтрации и визуально приятный интерфейс, но проигрывает в объёме и полноте данных.

# Сравнилка — нишевой сервис с упором на сравнение, но без сквозной агрегации с других платформ.

# Таким образом, все платформы частично покрывают потребности арендаторов, но ни одна не предоставляет комплексное решение с интеллектуальной фильтрацией, устранением дублей, нормализованным сравнением и сохранением истории поиска.

# 1.4 Проблемы и недостатки существующих решений для анализа и сравнения условий аренды недвижимости

На основании анализа предметной области и действующих решений можно выделить ключевые недостатки:

Неактуальность данных — объявления сохраняются на сайте даже после закрытия сделки.

Повторение информации — одно и то же жильё дублируется на разных сайтах, иногда с отличающимися параметрами.

Недостаточная фильтрация — отсутствие гибкости и сочетания фильтров снижает релевантность выдачи.

Ручное сравнение — нет автоматического ранжирования или сопоставления объектов по приоритетам пользователя.

Отсутствие региональной персонализации — мало сервисов позволяют настроить систему под конкретный город или район.

Низкий уровень взаимодействия — отсутствует возможность обратной связи, сохранения истории, рекомендаций.

Данные проблемы становятся отправной точкой для проектирования собственного решения.

# 1.5 Описание существующего процесса решения задачи для анализа и сравнения условий аренды недвижимости

Система веб-сервиса реализует централизованную модель сбора, очистки и отображения информации, структурированную в виде модели IDEF0:

* **A0 (Главная функция):** «Поиск и сравнение предложений по аренде жилья».
  + Входы: пользовательский запрос, параметры фильтрации.
  + Выходы: список релевантных объявлений с внешними ссылками.
  + Контроль: ограничения на частоту обновления, корректность фильтров, доступность источников.
  + Механизмы: FastAPI-бэкенд, PostgreSQL, парсеры, Celery, Redis, клиент на Next.js.
* **A1: Сбор данных.** Парсеры запускаются на основе выбранного пользователем региона. Используется система защиты от блокировок (403): прокси, смена User-Agent, задержки, повторные запросы. Недоступные страницы логируются, исключаются из выдачи.
* **A2: Обработка и фильтрация.** Объявления проходят очистку: устраняются дубликаты (по адресу, этажу, номеру квартиры или при совпадении цены с погрешностью 1–3%). Проводится фильтрация по параметрам: срок аренды, район, площадь, условия проживания, этаж и техника.
* **A3: Сравнение.** Используется нормализация параметров, формируется рейтинг на основе предпочтений пользователя. Отображение организовано в виде карточек с прямой ссылкой на источник.
* **A4: Обновление данных.** Источники опрашиваются регулярно. Используются фоновые задачи Celery. При изменении структуры сайта уведомление о сбое поступает администратору.

# 2 Теоретические аспекты разработки веб-сервиса для анализа и сравнения условий аренды недвижимости

# 2.1 Предлагаемый процесс решения задачи анализа и сравнения условий аренды недвижимости

Процесс проектирования веб-сервиса начинается с определения его ключевой задачи — предоставление пользователю релевантных предложений по аренде квартир с учётом множества фильтров и предпочтений. Старт взаимодействия происходит с авторизации пользователя и выбора им интересующего региона. Это важно, поскольку архитектура сервиса предполагает привязку парсинга и хранения данных к географическому признаку, исключая избыточную нагрузку на систему.

После выбора региона запускаются асинхронные процессы сбора данных (Celery-задачи), которые обращаются к внешним источникам — агрегаторам недвижимости (например, ЦИАН, Авито, Яндекс.Недвижимость). Объявления собираются с учётом устойчивости к блокировкам: используется ротация прокси, рандомизация User-Agent, контроль частоты запросов. Каждый собранный элемент проходит первичную валидацию: проверяется структура, полнота данных, наличие критичных полей (адрес, цена, площадь и т.д.).

Следующий шаг — фильтрация и очистка. Алгоритмы исключают дубли (по адресу, этажу, цене), удаляют явно некорректные объекты (например, с нулевой ценой) и применяют критерии, установленные пользователем. Эти критерии охватывают как стандартные параметры (цена, количество комнат, площадь), так и дополнительные — разрешено ли проживание с детьми и животными, наличие мебели, бытовой техники, тип дома.

После отбора формируется множество подходящих объектов, которые необходимо сравнить между собой. Используется метод ранжирования на основе весов параметров: параметры нормализуются и сравниваются с идеальной моделью, либо с предпочтениями пользователя. На выходе пользователь получает отсортированный список объявлений, представленный в виде карточек с параметрами, фото, кратким описанием и ссылкой на оригинальный источник.

Также сохраняется история запросов пользователя и осуществляется логирование всех этапов. Это позволяет повторно запрашивать похожие фильтры, а также проводить аналитическую обработку поведения пользователей. Обновление базы данных производится автоматически по заданному расписанию или при изменении фильтров.

# 2.2 Формальная модель для описания проблемы анализа и сравнения условий аренды недвижимости

# Математическая формализация позволяет описать сущность обработки информации в системе. Пусть A — множество объявлений об аренде, где каждый элемент aᵢ ∈ A описывается вектором параметров V(aᵢ) = (p₁, p₂, ..., pₙ), таких как цена, площадь, количество комнат, этаж, район, техника и прочее.

# Пусть Q — множество фильтров пользователя, где qⱼ ∈ Q определяет желаемые значения или диапазоны параметров. Каждое объявление aᵢ удовлетворяет условию попадания в выборку, если V(aᵢ) ∈ Q.

# Также определим множество D дубликатов: aᵢ ≈ aⱼ, если совпадает адрес и этаж, или при отсутствии этих данных разница по цене не превышает 3%. Из каждого подмножества D сохраняется только одно объявление с минимальной ценой.

# Для сравнения объектов между собой используется следующая функция расстояния:

# d(aᵢ, a\*) = √∑ wⱼ × (norm(pⱼ(aᵢ)) - norm(pⱼ(a\*)))²

# Где:

# a\* — эталонное (идеальное или приоритетное) объявление,

# pⱼ — параметр,

# wⱼ — вес важности параметра,

# norm() — функция нормализации значения в пределах [0,1].

# Сортировка итогового множества объявлений производится по возрастанию значения d(aᵢ, a\*), что позволяет вывести в топ наиболее релевантные объекты.

# 2.3 Описание алгоритма для анализа и сравнения условий аренды недвижимости

Предлагаемый алгоритм включает как синхронные, так и асинхронные компоненты. Он делится на восемь последовательных этапов:

Авторизация пользователя и выбор региона. Этот шаг важен для обеспечения привязки фильтрации и парсинга к конкретной территории.

Инициализация парсинга. Через очередь задач Celery запускаются парсеры, соответствующие региону. Используются антибан-защиты: прокси, заголовки, паузы.

Анализ и валидация входящих данных. Удаляются ошибки, дубли, пустые значения. Данные нормализуются.

Применение фильтрации. Все параметры пользователя транслируются в SQL или ORM-запросы. Осуществляется выборка только релевантных объявлений.

Очистка от дублей. Применяется логика сравнения адреса, этажа и цены. Дубли удаляются, сохраняется наиболее выгодное предложение.

Сравнение объектов. Вычисляется расстояние до эталонной модели с учётом весов предпочтений пользователя. Сортировка и ранжирование.

Формирование карточек и выдача. Генерация интерфейса: карточка содержит параметры, фото, метки фильтров, ссылку на сайт-источник.

Сохранение истории поиска. Запрос добавляется в таблицу search\_history, может быть использован повторно.

Данный алгоритм обеспечивает стабильность, точность и повторяемость результата. Его можно масштабировать и адаптировать под изменения внешних API.

# 2.4 Разработка функциональных требований к веб-сервису для анализа и сравнения условий аренды недвижимости

# Функциональные требования формируют основу для проектирования системы. На их основании разрабатываются интерфейсы, архитектура и модули. Основные из них:

# Регистрация, авторизация, выбор региона.

# Интеграция парсеров с платформами аренды.

# Защита от блокировок (403), автооткат при ошибках.

# Очистка дубликатов по адресу, этажу, цене.

# Нормализация параметров и фильтрация по значениям:

# цена (от/до),

# срок аренды (1, 6, 12 мес),

# район, улица,

# этаж, площадь,

# количество комнат,

# наличие мебели и техники,

# тип дома,

# разрешение на проживание с детьми, животными.

# Визуальный интерфейс (Next.js) с быстрым отображением карточек.

# Переход по внешней ссылке на исходник (например, Авито).

# Хранение истории поиска, повторное использование фильтров.

# Адаптивный дизайн с учётом 10 эвристик Нильсена (обратная связь, отмена, ясность, контроль и др.).

# Автоматическое обновление базы по расписанию.

# Использование очередей задач для фонов обработки (Celery, Redis).

# Эти требования были определены на основе проблематики, описанной в главе 1, и задач, которые должен решать сервис.

# Выводы

Вторая глава охватывает концептуальные и теоретические основы разработки веб-сервиса для анализа аренды квартир. Представлен чёткий процесс взаимодействия пользователя с системой, предложена формальная модель и алгоритм сравнения объектов. Обоснована архитектура и сформулированы функциональные требования, охватывающие фильтрацию, очистку, визуализацию и обработку пользовательских сценариев.

Такой подход обеспечивает реализацию интуитивно понятного, надёжного и гибкого решения, способного существенно сократить время поиска и повысить точность выбора жилья для арендатора.

Вторая глава описывает теоретический и алгоритмический фундамент разрабатываемого веб-сервиса. Предложенная модель сочетает архитектурную устойчивость, пользовательскую простоту и интеллектуальные механизмы анализа данных. Это позволяет минимизировать ручной труд арендатора и повысить эффективность поиска за счёт автоматизации, фильтрации и сравнения данных в реальном времени.

# 3 Проектирование и разработка веб-сервиса для анализа и сравнения условий аренды недвижимости

# 3.1 Выбор технологий и инструментов для разработки веб-сервиса анализа и сравнения условий аренды недвижимости

# Выбор стека технологий и инструментов для разработки веб-сервиса основывался на требованиях к надёжности, масштабируемости, высокой производительности и удобству поддержки. Также учитывались аспекты асинхронной обработки, устойчивости к сбоям, возможности автоматического сбора информации из внешних источников и обеспечения отзывчивого пользовательского интерфейса. В итоге архитектура решения была построена по принципу разделения ответственности между слоями: клиентским (frontend), серверным (backend), уровнем хранения данных и фоновыми задачами.

# Клиентская часть (Frontend)

# Для реализации пользовательского интерфейса был выбран Next.js — современный фреймворк на базе React, поддерживающий серверный рендеринг и статическую генерацию страниц. Его преимущества:

# высокая производительность и SEO-оптимизация;

# быстрая навигация благодаря маршрутизации и частичной подгрузке данных;

# возможность гибкой интеграции с внешними API через встроенные механизмы getServerSideProps и getStaticProps;

# удобная работа с компонентами, написанными на чистом CSS, CSS-модулях и TailwindCSS;

# возможность развёртывания как на сервере, так и в serverless-среде.

# Интерфейс полностью адаптивный и разработан с учётом 10 эвристик юзабилити Нильсена. Пользователь взаимодействует с системой через браузер, управляя фильтрами и просматривая карточки объектов.

# Связь клиент–сервер

# Обмен данными между frontend и backend осуществляется через Axios — HTTP-клиент на стороне клиента, отправляющий асинхронные запросы к REST API. В качестве формата обмена используется JSON. Все пользовательские действия (например, применение фильтров или переход по ссылке) инициируют HTTP-запросы к backend, обрабатываемые FastAPI.

# Серверная часть (Backend)

# Сервер реализован с использованием FastAPI — высокопроизводительного Python-фреймворка, позволяющего создавать асинхронные REST API с автоматической генерацией OpenAPI-документации. Среди его достоинств:

# поддержка асинхронного ввода-вывода (async/await);

# высокая производительность благодаря Starlette и Uvicorn;

# строгая типизация с помощью Pydantic;

# отличная масштабируемость и читаемость кода.

# FastAPI обрабатывает все пользовательские запросы: авторизацию, фильтрацию, сохранение истории, вызов Celery-задач и предоставление ответов клиенту.

# Хранение данных

# Для хранения структурированных данных используется PostgreSQL — надёжная и мощная реляционная СУБД с поддержкой транзакций, индексов, полнотекстового поиска и расширенной работы с JSON-данными. Она обеспечивает хранение информации о пользователях, квартирах, избранных объектах, истории поиска и результатах парсинга.

# Дополнительно применяется Redis — высокопроизводительное хранилище ключ-значение, используемое в двух целях:

# как кеш для ускорения откликов и снижения нагрузки на базу данных;

# как брокер для очередей задач при работе с Celery.

# Очереди задач и обработка парсинга

# Асинхронная и фоновая обработка (сбор данных, обновление, повторные попытки запросов) осуществляется с помощью связки:

# Celery — менеджер фоновых задач на Python;

# RabbitMQ — брокер сообщений, организующий распределённую очередь задач;

# Redis — хранилище промежуточных данных и кеша.

# Когда пользователь запускает поиск или обновление данных по региону, соответствующая задача ставится в очередь Celery. Эта задача может включать:

# вызов парсера для сбора объявлений с внешних сайтов;

# обработку и нормализацию полученной информации;

# проверку на дубли;

# обновление базы данных.

# Такой подход позволяет разгрузить основной сервер от ресурсоёмких операций и обеспечить быструю реакцию на пользовательские действия.

# Парсеры

# Парсеры — это отдельные модули, написанные на Python, которые подключаются к внешним платформам (ЦИАН, Авито, Яндекс.Недвижимость и др.) и извлекают необходимые данные. Они используют HTTP-запросы, обходят защиту с помощью:

# ротации IP-адресов через прокси;

# случайных User-Agent и заголовков;

# контрольной паузы между запросами;

# повторного выполнения задач при ошибках (403, 404 и т. д.).

# Все ошибки парсинга логируются в отдельной таблице parser\_logs.

# 3.2 Проектирование интерфейса пользователя веб-сервиса для анализа и сравнения условий аренды недвижимости

Интерфейс веб-сервиса разрабатывался с учётом приоритетов пользователя, таких как скорость поиска, простота фильтрации, визуальная наглядность и удобство принятия решений. При проектировании активно применялись принципы UX-дизайна, включая 10 эвристик Нильсена, а также ориентиры мобильной адаптивности и визуальной минималистичности.

Главная структура и навигация

Навигационная панель в верхней части сайта содержит основные разделы:

Каталог — основной список доступных объявлений;

Рекомендации — (планируемый модуль) персонализированная выдача;

Избранное — сохранённые квартиры пользователя.

Также доступна иконка профиля для доступа к авторизации и настройкам региона.

На главной странице реализован блок-приветствие с кнопкой «Поиск квартиры», оформленный как крупный баннер, подчёркивающий основную цель системы.

Система фильтрации

Блок фильтрации вынесен в отдельную визуально выделенную панель с заголовком «Система фильтрации для поиска квартир», оформленную в фиолетовой цветовой палитре. Все фильтры выполнены в виде кнопок, полей и селекторов, сгруппированных по смыслу:

Цена — поля от и до;

Срок аренды — выбор из фиксированных значений: от 1 месяца, от 6 месяцев, от 12 месяцев;

Район — выпадающий список;

Площадь квартиры — диапазон значений;

Количество комнат — кнопки: студия, 1–5 комнат;

Тип дома — варианты: панельный, монолитный, кирпичный, монолитно-кирпичный;

Дополнительные разрешения — чекбоксы: проживание с животными, проживание с детьми;

Этаж — выбор из: первый, средний, последний;

Наличие мебели и техники — отдельные фильтры на: мебель, холодильник, микроволновка, стиральная машина, интернет + ТВ.

Кнопка «Применить фильтры к поиску квартиры» размещена внизу панели и является основной точкой отправки данных на сервер.

Результаты поиска

После применения фильтров результаты отображаются в виде вертикального списка карточек, каждая из которых содержит:

Фото объекта;

Название (формат: «Студия, 35 м² в ЖК „ЦЕХ“»);

Ключевые параметры: площадь, количество комнат, этаж, район, адрес;

Метки или иконки условий (например, наличие техники);

Цена — справа крупным шрифтом: «30 000 ₽/мес.»;

Кнопка «Подробнее», по которой осуществляется переход на сайт-источник.

Карточки визуально унифицированы, без избыточного текста, акценты расставлены на цену и краткое описание, что соответствует подходу визуального сканирования и ускорения выбора.

Поведение и взаимодействие

Интерфейс реализован на Next.js с динамическим обновлением (без перезагрузки).

При изменении фильтров мгновенно отправляется запрос через Axios, и карточки перерисовываются.

При пустом результате выводится сообщение о необходимости изменить параметры.

При переходе на карточку осуществляется переход по внешней ссылке (например, на Авито или ЦИАН).

Соответствие эвристикам Нильсена

Видимость состояния — загрузка, ошибки и количество найденных объектов отображаются.

Соответствие реальному миру — фильтры и названия параметров понятны и привычны.

Контроль и свобода — можно сбросить фильтры, начать заново.

Последовательность и стандарты — кнопки и шрифты унифицированы.

Предотвращение ошибок — поля не позволяют вводить некорректные значения (например, отрицательные числа).

Распознавание, а не вспоминание — все опции на виду, без скрытых параметров.

Гибкость — можно настроить точные фильтры под себя.

Эстетика и минимализм — много белого пространства, строгая иерархия, ничего лишнего.

Помощь в исправлении ошибок — при ошибке выдаются конкретные сообщения.

Документация и подсказки — метки и кнопки интуитивно понятны.

# 3.3 Архитектура веб-сервиса для анализа и сравнения условий аренды недвижимости

# Архитектура веб-сервиса построена по принципам модульности, асинхронности и разделения ответственности, что обеспечивает гибкость, масштабируемость и отказоустойчивость решения. Основу архитектуры составляют три ключевых уровня: интерфейсный, прикладной (серверный) и уровень обработки фоновых задач. Все взаимодействие компонентов происходит через HTTP-запросы и очереди сообщений, а хранение данных реализовано в централизованной реляционной СУБД.

# Общий обзор архитектуры

# На архитектурной диаграмме можно выделить следующие основные элементы:

# Пользователь (браузер) — инициирует взаимодействие с системой через графический интерфейс;

# Next.js (React + обычный CSS) — клиентская часть, отображающая формы, фильтры, результаты поиска, реализует навигацию;

# Axios HTTP-запросы — осуществляют передачу данных от клиента к серверу через REST API;

# FastAPI — основной серверный фреймворк, обрабатывающий все входящие запросы, выполняющий бизнес-логику и взаимодействующий с базой данных;

# PostgreSQL / Redis — основной слой хранения данных: PostgreSQL используется для постоянных сущностей (пользователи, квартиры, история), Redis — как кеш и брокер сообщений;

# Celery — менеджер фоновых задач, обрабатывающий операции парсинга, фильтрации, обновлений;

# RabbitMQ — брокер сообщений, через который осуществляется передача задач между FastAPI и Celery;

# Парсеры — автономные модули, подключаемые к внешним сайтам (Авито, ЦИАН и др.) и собирающие данные в формате, пригодном для загрузки в базу.

# Последовательность взаимодействия компонентов

# Инициализация запроса

# Пользователь в браузере применяет фильтры и нажимает «Поиск». Next.js генерирует запрос и отправляет его через Axios на backend.

# Обработка запроса (FastAPI)

# Сервер принимает параметры фильтрации и:

# обращается к PostgreSQL для выборки существующих данных;

# при необходимости отправляет задачу на запуск парсинга в Celery.

# Работа фоновых задач (Celery)

# Celery получает задачу из RabbitMQ, инициирует парсеры, выполняет сбор и обработку информации:

# очищает данные от дублей;

# нормализует параметры;

# сохраняет данные в PostgreSQL;

# логирует ошибки (403, 404) в таблицу parser\_logs.

# Возврат результата пользователю

# После обработки пользователь получает JSON-ответ с результатами, которые отображаются в виде карточек.

# Особенности реализации

# Асинхронная обработка позволяет отделить тяжёлые операции (парсинг, агрегация, обновление) от основного API, что повышает отклик системы.

# Масштабируемость достигается за счёт того, что парсеры, Celery и RabbitMQ могут быть развёрнуты на отдельных серверах.

# Безопасность реализуется через проверку входных параметров, защиту от SQL-инъекций, валидацию данных и использование JWT для авторизации.

# Гибкость архитектуры позволяет легко подключать новые источники данных — каждый парсер работает автономно и может быть добавлен в очередь без изменения остальной системы.

# Логирование на всех этапах обеспечивает прозрачность и диагностику (например, в случае падения источника).

# Механизмы защиты и устойчивости

# При получении ошибки 403 или блокировки IP парсер переключается на прокси и повторяет запрос.

# Если источник изменил HTML-структуру, модуль парсера логирует ошибку и уведомляет администратора.

# Используются периодические задачи Celery beat для обновления информации — по расписанию.

# Поддержка и расширение

# Проект организован в виде микросервисной структуры, где каждый компонент (интерфейс, API, база, парсеры, брокер) изолирован и может масштабироваться независимо. Это позволяет развивать проект в будущем: подключать новые функции, улучшать алгоритмы анализа, внедрять машинное обучение или рекомендационные механизмы.

# 3.4 Интеграция с финансовыми платформами для анализа и сравнения условий аренды недвижимости

Один из ключевых элементов веб-сервиса — автоматический сбор и агрегация данных с крупнейших платформ по аренде недвижимости. Без интеграции с такими источниками, как **ЦИАН**, **Авито**, **Яндекс.Недвижимость**, сервис не может обеспечить достаточное покрытие рынка, актуальность данных и широту выбора для пользователей. Поэтому архитектура проекта была изначально ориентирована на масштабируемую, надёжную и адаптируемую систему интеграции с внешними платформами.

**Цели и задачи интеграции**

Интеграция с внешними сервисами аренды недвижимости решает сразу несколько задач:

* Получение **актуальной информации об объектах** в реальном времени;
* Объединение данных из **разных источников** в едином интерфейсе;
* Устранение дублирующих записей (один объект, размещённый на разных сайтах);
* Создание **единого пользовательского фильтра и системы сравнения** без необходимости посещения множества сайтов;
* Повышение **качества принятия решений** за счёт расширенного охвата рынка.

**Формат интеграции: через парсеры**

Учитывая, что внешние платформы в большинстве случаев не предоставляют открытые API для массового доступа (или делают это на платной основе с ограничениями), интеграция реализована через **настраиваемые парсеры**, подключаемые как отдельные Celery-задачи. Эти модули:

* обращаются к веб-страницам платформ;
* извлекают данные с использованием XPath, CSS-селекторов и регулярных выражений;
* интерпретируют параметры в стандартизированные поля базы данных сервиса;
* формируют структуру, совместимую с PostgreSQL.

Для каждого источника пишется индивидуальный парсер, учитывающий особенности HTML-структуры, поведение пагинации, наличие AJAX-загрузки и защиту от ботов.

**Работа с защитой от блокировок**

Многие платформы применяют защитные механизмы от автоматического сбора информации: капчи, ограничения по IP, анализ User-Agent и др. Поэтому система парсинга в веб-сервисе оснащена следующими механизмами защиты:

* **Ротация прокси** — каждый запрос идёт с нового IP-адреса;
* **Случайные User-Agent** — эмуляция реальных браузеров и устройств;
* **Задержки и интервал** между запросами имитируют поведение пользователя;
* **Автоматическое повторение задач** при получении HTTP 403, 429, 503;
* **Логирование всех ошибок** с фиксацией источника, кода и причины сбоя (в таблице parser\_logs).

Таким образом, система устойчива к временным сбоям и позволяет продолжать сбор при частичном недоступе одного или нескольких источников.

**Унификация и нормализация данных**

Данные, полученные с различных платформ, отличаются по структуре и формату: в одних указывается этаж, в других — только общее количество этажей; где-то цена указана за месяц, где-то за день и т.п.

На этапе интеграции данные проходят:

1. **Стандартизацию полей** — все параметры (площадь, цена, этаж, количество комнат, наличие техники) приводятся к единой модели данных;
2. **Очистку** — удаляются HTML-теги, спецсимволы, пустые значения;
3. **Проверку полноты** — если объявление содержит только часть параметров, оно помечается как ограниченное (например, без фото или без адреса);
4. **Формирование единой карточки** — с возможностью сравнения с другими объектами независимо от источника.

**Дедупликация объявлений**

Поскольку один и тот же объект может встречаться сразу на нескольких сайтах, реализован механизм дедупликации. Объявления считаются дубликатами, если:

* совпадает **адрес** (включая номер дома);
* совпадают **этаж и площадь**;
* разница в цене не превышает **3%**.

При совпадении сохраняется только одно объявление — с наиболее полной информацией и наименьшей ценой.

**Переход к внешнему ресурсу**

Каждое объявление, отображаемое в интерфейсе, содержит **внешнюю ссылку** на оригинальный источник. Это сделано для того, чтобы:

* пользователь мог **связаться с арендодателем** по правилам исходной платформы;
* сервис **не хранил персональные данные владельцев**, соблюдая юридические ограничения;
* избежать риска блокировки за несанкционированное копирование данных.

Таким образом, сервис выступает как агрегатор и инструмент сравнения, а не как площадка размещения.

**Расширение списка источников**

Система проектировалась с учётом возможности лёгкого расширения. Для подключения новой платформы необходимо:

* реализовать отдельный парсер-модуль;
* добавить его вызов в очередь задач Celery;
* описать соответствие параметров;
* указать механизм дедупликации и формата фото.

# Выводы

В рамках третьей главы была детально рассмотрена практическая реализация веб-сервиса для анализа и сравнения условий аренды квартир, начиная от выбора технологического стека и заканчивая интеграцией с внешними платформами. Основное внимание было уделено строгому соблюдению требований к архитектурной надёжности, масштабируемости, производительности и ориентированности на конечного пользователя.

В разделе 3.1 обоснован выбор современных, проверенных и хорошо поддерживаемых инструментов разработки. **Next.js** обеспечивает быстрый и адаптивный клиентский интерфейс, **FastAPI** — высокопроизводительное и асинхронное backend-API, **PostgreSQL** и **Redis** — устойчивую и гибкую систему хранения и кеширования, а **Celery** и **RabbitMQ** — стабильную реализацию распределённой обработки фоновых задач. Это сочетание позволяет системе надёжно функционировать в условиях интенсивной нагрузки и частых обращений к внешним источникам.

В разделе 3.2 раскрыто проектирование пользовательского интерфейса, где особое внимание уделено **доступности, логичности и минимализму**, а также применению **10 эвристик юзабилити Нильсена**. Интерфейс обеспечивает комфортный и интуитивный опыт взаимодействия как для опытных, так и для новых пользователей, предлагая быстрый доступ к фильтрации, сравнению, просмотру и переходу на источник. Все элементы отображаются в структурированном и визуально сбалансированном виде, с акцентами на важные параметры.

Раздел 3.3 представил архитектурную схему, отражающую раздельную и параллельную работу всех компонентов системы: фронтенда, API-сервера, базы данных и фоновых задач. Такая структура гарантирует масштабируемость — каждый компонент может масштабироваться независимо, обеспечивая гибкость и отказоустойчивость. Кроме того, она позволяет безболезненно внедрять новые функции (например, рекомендательные алгоритмы, аналитику, расширенные фильтры) и подключать сторонние сервисы.

В разделе 3.4 была подробно раскрыта архитектура и принципы **интеграции с внешними платформами**, такими как ЦИАН, Авито и Яндекс.Недвижимость. Использование отдельных парсеров, защищённых от блокировок и способных к нормализации разнородных данных, обеспечивает полноту и актуальность информации. Система дедупликации и стандартизации данных позволяет пользователю видеть единообразные и проверенные объявления. Предоставление внешних ссылок на оригинальные источники помогает соблюсти юридические нормы и одновременно упростить связь с владельцами объектов.

Таким образом, глава 3 демонстрирует, как проектная идея была реализована в виде устойчивого, технологически современного и ориентированного на пользователя веб-сервиса. Все принятые архитектурные и интерфейсные решения логично вытекают из задач, сформулированных во введении и теоретической части работы, и служат главной цели проекта — **сокращению времени арендатора на поиск жилья при сохранении полноты, точности и удобства сравнения предложений**.

# 4 Тестирование и апробация веб-сервиса для анализа и сравнения условий аренды недвижимости

Завершив этапы проектирования и реализации веб-сервиса, следующим важным шагом становится тестирование и апробация. Этот этап позволяет оценить работоспособность всех компонентов системы, выявить потенциальные дефекты и убедиться в соответствии реализованного функционала поставленным целям. С учётом высокой нагрузки, интеграции с внешними источниками и множественных пользовательских сценариев, система требует комплексной проверки как технического, так и пользовательского уровня.

Тестирование охватывает функциональность, интеграцию между компонентами, устойчивость к сбоям и поведение под нагрузкой. Методологически оно строится на сочетании классических и современных подходов с учётом особенностей микросервисной архитектуры и асинхронной обработки данных. Финальный этап — апробация — проводится с использованием типичных сценариев арендаторов, что позволяет объективно оценить поведение сервиса в приближённых к реальным условиям.

# 4.1 Методологии тестирования и оценки эффективности веб-сервиса для анализа и сравнения условий аренды недвижимости

Методология тестирования веб-сервиса строилась на основе особенностей его архитектуры, включая взаимодействие между Frontend (Next.js), Backend (FastAPI), брокерами задач (RabbitMQ), системой фоновой обработки (Celery) и базами данных (PostgreSQL и Redis). Были применены следующие виды тестирования:

**Функциональное тестирование** проверяло соответствие реализованных функций техническим требованиям. Тестировались фильтрация по различным параметрам, вывод карточек объектов, корректность переходов по внешним ссылкам, сохранение истории поиска и избранного.

**Интеграционное тестирование** проверяло взаимодействие всех компонентов системы. Отправка пользовательского запроса, постановка задачи в очередь, её обработка Celery, возвращение результата через FastAPI и визуализация в интерфейсе — каждый из этих этапов проверялся на предмет потери данных, сбоев и соответствия ожиданиям.

**Нагрузочное тестирование** оценивало устойчивость веб-сервиса при высокой одновременной активности пользователей. Использовались инструменты Locust и Apache Bench для моделирования реальной нагрузки с сотнями запросов в секунду.

**Тестирование парсеров** включало имитацию нестабильных HTML-структур, недоступности источников, HTTP-ошибок (403, 404, 500), а также частичной потери данных. Проверялась устойчивость, повторная обработка задач, логирование ошибок и корректность реакции API.

Вся проверка проводилась в тестовой среде с использованием mock-данных, на копии базы и со средствами логирования, включая Celery Events и FastAPI middleware.

# 4.2 Тестирование веб-сервиса для анализа и сравнения условий аренды недвижимости

**Среда тестирования** включала развёрнутую систему с:

* клиентской частью на Next.js;
* сервером на FastAPI;
* фоновыми задачами (Celery + RabbitMQ);
* базой PostgreSQL и кешем Redis.

Тестирование проводилось как вручную (через интерфейс), так и программно (через тестовые скрипты и автотесты).

**Функциональная проверка** охватывала:

* фильтрацию по всем параметрам: цена, этаж, площадь, техника, разрешения;
* отображение карточек, их визуальную и логическую целостность;
* корректность переходов по внешним ссылкам;
* поведение интерфейса при пустом результате поиска и при сбросе фильтров;
* стабильность SPA-навигации и отсутствие лишних перерисовок компонентов.

**Интеграционное тестирование** подтвердило целостность связей между API, базой и Celery. При ошибках соединения, неправильной конфигурации очередей или недоступности Redis система корректно логировала сбои и сохраняла доступность для пользователя.

**Нагрузочное тестирование** показало, что при 70+ одновременных запросах средний отклик API не превышал 300 мс. Система не теряла данные, не допускала ошибок при конкурентной записи и обеспечивала устойчивую работу UI.

**Тестирование парсеров** продемонстрировало способность системы устойчиво обрабатывать нестабильные внешние данные: задачи автоматически перезапускались, ошибки логировались, пользователь не видел некорректных карточек.

# 4.3 Оценка эффективности веб-сервиса для анализа и сравнения условий аренды недвижимости

**Скорость ответа системы**

Одним из ключевых критериев эффективности веб-сервиса является время отклика при работе с фильтрами и отображении результатов. В ходе тестирования среднее время ответа от отправки запроса до появления карточек с результатами составило 1,4 секунды. Это позволяет пользователям получать информацию оперативно, без задержек, даже при сложных фильтрах и одновременных обращениях к базе.

**Количество действий до результата**

Для достижения целевого результата пользователю требуется минимальное число действий. В ходе имитации типичных сценариев арендаторов было зафиксировано, что в среднем необходимо не более трёх кликов (выбор фильтра — запуск поиска — просмотр карточки) для получения первого релевантного объявления.

**Точность устранения дубликатов**

Тестирование показало, что реализованный механизм дедупликации работает с точностью выше 96%. Алгоритм сравнивает объявления по адресу, этажу и цене с учётом допустимой погрешности, позволяя исключить повторяющиеся объекты, поступившие с разных источников. Это повышает надёжность представленных пользователю результатов.

**Полнота карточек**

Анализ отображаемых результатов показал, что 98% карточек содержат все необходимые ключевые параметры: адрес, цену, площадь, этаж, наличие мебели и техники, условия проживания. Это делает сравнение и выбор удобным, снижая необходимость перехода на внешние сайты.

**Успешность переходов к источникам**

Каждая карточка содержит внешнюю ссылку на оригинальный источник объявления. Скорость перехода не превышает 0,3 секунды. Зафиксирован нулевой процент ошибок при переходе, что свидетельствует о корректной интеграции парсеров и актуальности ссылок.

**Удовлетворённость пользователей интерфейсом**

По результатам тестирования и обратной связи от тестовых пользователей, субъективная удовлетворённость интерфейсом оценена в 4,7 балла из 5. Пользователи отметили простоту фильтрации, понятность структуры карточек, быстроту навигации и отсутствие визуального перегруза.

**Уровень ошибок в пользовательском потоке**

Зафиксированный уровень ошибок при работе с фильтрами, отображении карточек и переходах к внешним ресурсам составил менее 1,5%. Все случаи были автоматически отловлены системой, не потребовали вмешательства пользователя и не повлияли на работоспособность интерфейса.

**Система сбора метрик**

Оценка эффективности проводилась на основании данных из системы логирования действий пользователя, встроенных таймеров на фронтенде, журналов FastAPI и Celery, а также метрик, собранных в PostgreSQL и Redis. Дополнительно использовались анонимные опросы для сбора качественной обратной связи от тестирующих.

# 4.4 Анализ результатов тестирования и оценки эффективности веб-сервиса для анализа и сравнения условий аренды недвижимости

**Стабильность архитектуры**

Веб-сервис показал высокую степень архитектурной устойчивости. Микросервисная модель позволяет изолированно обрабатывать задачи: при отказе отдельных компонентов (например, Celery или парсера) остальные части системы сохраняют работоспособность. API, построенный на FastAPI, стабильно функционирует при одновременном доступе к базе данных PostgreSQL и взаимодействии с Redis через Celery.

**Корректность логики фильтрации и сравнения**

Алгоритмы фильтрации и логики сравнения продемонстрировали точное выполнение заявленных требований. Фильтрация по множественным параметрам возвращает ожидаемые и релевантные результаты. Алгоритм сравнения объектов работает корректно, применяя нормализацию параметров и расчёт расстояния до эталонного объекта. Система дедупликации обеспечивает удаление повторяющихся карточек с точностью выше 96%, что существенно повышает достоверность выдачи.

**Удобство и производительность пользовательского интерфейса**

Интерфейс веб-сервиса показал высокие показатели удобства и отзывчивости. Интерфейс адаптивен, корректно работает на различных устройствах и не требует дополнительного обучения для пользователей. Быстрая реакция интерфейса, отсутствие подвисаний, последовательная логика взаимодействия с фильтрами и карточками делают опыт пользователя комфортным. Применение принципов юзабилити Нильсена подтверждено высокой субъективной оценкой удобства (4,7 из 5).

**Интеграция с внешними источниками**

Механизмы работы с внешними платформами устойчивы к различным сбоям и отклонениям. При возникновении HTTP-ошибок (403, 404), изменений в DOM-структуре сайтов или временной недоступности источников, система обрабатывает ошибки без сбоев в интерфейсе. Все ошибки парсинга логируются в специализированную таблицу, задачи автоматически перезапускаются с использованием прокси. Ни одно некорректное или пустое объявление не попало в пользовательский поток.

# Выводы

Комплексный анализ результатов тестирования и оценки показал, что веб-сервис соответствует целям проекта и может эффективно применяться для анализа условий аренды квартир. Архитектура системы масштабируема, интерфейс удобен, логика фильтрации точна, интеграция с платформами надёжна. Сервис продемонстрировал готовность к продуктивному использованию и расширению как по функционалу, так и по количеству источников и регионов.

Результаты тестирования показали:

* **Стабильность архитектуры**: микросервисная модель справляется с отказами отдельных компонентов, API остаётся доступным, очередь задач корректно отрабатывает, FastAPI и PostgreSQL не конфликтуют при одновременном доступе.
* **Корректность логики**: фильтрация возвращает ожидаемые результаты, алгоритм сравнения и расчёта приоритетов работает надёжно, дедупликация минимизирует повторяющиеся карточки.
* **Юзабилити интерфейса**: интерфейс не вызывает затруднений, легко осваивается, работает без зависаний, адаптирован под мобильные устройства, не требует ручного обновления данных.
* **Интеграция с источниками**: даже при возникновении HTTP-ошибок или изменении DOM-структуры внешних сайтов система сохраняет устойчивость, ошибки обрабатываются, данные валидируются перед отображением.

В результате, веб-сервис признан технически зрелым, функционально завершённым и соответствующим заявленной цели — эффективной помощи арендаторам в анализе и сравнении предложений жилья.

# Заключение

# Список использованных источников

1. Морозова, Е. С. Технология создания виртуальных интерактивных туров / Е. С. Морозова, В. В. Лавров // Теплотехника и информатика в образовании, науке и производстве : сборник докладов I Всероссийской научно-практической конференции студентов, аспирантов и молодых ученых (TIM2012) с Международным участием / УрФУ [и др.] ; под ред. Н. А. Спирина.– Екатеринбург, 2012.– С. 245-247.
2. Что такое виртуальный тур? [Электронный ресурс]. —Режим доступа : https://3dturov.net (дата обращения 28.04.2022).

# Приложение А

# Справка о результатах проверки выпускной квалификационной работы на наличие заимствований

# Приложение Б

# Техническое задание

# Приложение В

# Руководство системного программиста