**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 2](#_Toc197527421)

[ОСНОВНАЯ ЧАСТЬ 4](#_Toc197527422)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#_Toc197527423)

[1.1. Анализ отрасли 4](#_Toc197527424)

[1.2. Анализ имеющихся решений. 6](#_Toc197527425)

[1.3. Обоснование необходимости разработки и юридической правомерности. 9](#_Toc197527426)

[2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ 10](#_Toc197527427)

[2.1. Требования к системе. 10](#_Toc197527428)

[2.2. Функциональная модель системы (диаграмма прецедентов) 13](#_Toc197527429)

[2.3. Архитектура системы (клиент сервер) 13](#_Toc197527430)

[2.4. Моделирование данных (классы, данные) 13](#_Toc197527431)

[2.5. Моделирование логики работы (диаграмма состояний, компонентов, последовательности) 13](#_Toc197527432)

[3. РАЗРАБОТКА СИСТЕМЫ 13](#_Toc197527433)

[3.1. Этапы разработки (Последовательность выполнения действий для разработки системы). 13](#_Toc197527434)

[3.2. Средства разработки (стек технологий и аппаратное обеспечение). 13](#_Toc197527435)

[3.3. Описание системы 13](#_Toc197527436)

[3.3.1. Интерфейс пользователя (прототипы в Фигме) 13](#_Toc197527437)

[3.3.2. Дизайн (красивое/не красивое) 13](#_Toc197527438)

[3.3.3. Функциональные блоки (Список страниц отличных по наполнению и их описание) 13](#_Toc197527439)

[3.3.4. Серверная часть системы (потоки http запросов от клиента к серверу и обратно) 13](#_Toc197527440)

[3.3.5. Описание контроллеров (конченых точек и их описание) 13](#_Toc197527441)

[3.3.6. Описание моделей (как и где тащим данные из базы или загружаем их) 13](#_Toc197527442)

[3.3. Алгоритмы обработки данных (как мы html в текст превращали, где и как это сделать) 13](#_Toc197527443)

[3.3.1. Входные данные (параметры для веб-скрапинга) 13](#_Toc197527444)

[3.3.2. Обработка данных (Определяем, что за тип файла, определяем структуру, отсекаем ненужное, тащим нужное) 13](#_Toc197527445)

[3.3.3. Выходные данные (Описание того, что вытащили) 13](#_Toc197527446)

[3.4. Тестирование системы (зачем тестируем, что тестируем) 13](#_Toc197527447)

[3.4.1. Методика тестирования системы (интеграционное, функциональное, модульное) 13](#_Toc197527448)

[3.4.2. Результаты тестов (тестировали то то получили то то) 13](#_Toc197527449)

[4. ПЕРСПЕКТИВА РАЗВИТИЯ СИСТЕМЫ 13](#_Toc197527450)

[ЗАКЛЮЧЕНИЕ 14](#_Toc197527451)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 15](#_Toc197527452)

# ВВЕДЕНИЕ

В современном мире строительная отрасль активно использует цифровые технологии для оптимизации процессов проектирования, закупок и выполнения строительных работ. Информация о строительных материалах, их свойствах, производителях и ценах является ключевым ресурсом для принятия обоснованных решений. Однако проблема заключается в том, что данные об этих материалах зачастую разрознены, хранятся в различных форматах и на множестве платформ.

Создание программных решений для автоматизации сбора и обработки таких данных становится актуальной задачей. Автоматизированный сбор данных позволяет не только сократить временные затраты, но и повысить точность анализа, обеспечивая доступ к актуальной информации о строительных материалах в режиме реального времени.

С развитием онлайн-коммерции строительные материалы становятся все более доступными через интернет-магазины и специализированные платформы. Однако поиск и систематизация данных о них требует значительных усилий, особенно если речь идет о сравнении цен, изучении характеристик и отзывов. Компании и частные клиенты, принимающие решения на основе неполной или устаревшей информации, рискуют столкнуться с перерасходом бюджета или получением некачественного продукта.

Программное обеспечение, разработанное для автоматизированного сбора данных, предоставляет решение этой проблемы. Оно может осуществлять сбор информации с веб-ресурсов, таких как каталоги товаров, маркетплейсы и форумы, обрабатывать полученные данные и предоставлять пользователю структурированные результаты в удобном формате.

Кроме того, такие системы находят применение в мониторинге рынка, анализе конкурентов и оптимизации логистики. Для малых и средних предприятий внедрение подобных решений особенно важно, так как оно позволяет снизить затраты на ручную обработку информации и повышает конкурентоспособность.

Целью данной работы является разработка программного обеспечения для автоматизированного сбора данных о строительных материалах. Программа должна обеспечивать сбор данных (характеристики, цены) с различных веб-ресурсов и предоставлять их в удобной для анализа форме.

Для достижения данной цели необходимо решить следующие задачи:

1. Проанализировать существующие подходы и инструменты для сбора данных с веб-ресурсов, включая веб-скрейпинг и обработку данных.
2. Определить ключевые требования к программному обеспечению, включая целевые источники данных и формат их представления.
3. Разработать архитектуру программы, включающую модули сбора, обработки и хранения данных.
4. Реализовать основные алгоритмы веб-скрейпинга и обработки текстовой информации.
5. Провести тестирование программы на примере сбора данных с реальных веб-ресурсов.
6. Оценить эффективность программы и её практическую применимость.

Разработка программы для сбора данных о строительных материалах позволит автоматизировать трудоемкие процессы мониторинга и анализа информации. Компании смогут получать актуальные данные о товарах и ценах, анализировать характеристики материалов и изучать отзывы клиентов, что приведет к повышению качества принимаемых решений.

Кроме того, созданное программное решение может быть адаптировано для других отраслей, где требуется сбор и обработка данных из различных источников. Это делает проект универсальным инструментом для работы с информацией.

# ОСНОВНАЯ ЧАСТЬ

# АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

# Анализ отрасли

Строительная отрасль играет ключевую роль в экономике, формируя инфраструктуру, жильё, промышленные объекты и общественные сооружения. Это одна из самых динамично развивающихся сфер, которая тесно связана с рядом смежных отраслей, таких как производство строительных материалов, транспорт, логистика и архитектурное проектирование.

Особенности строительной отрасли:

1. Высокий уровень конкуренции среди компаний, предлагающих строительные материалы и услуги.

2. Зависимость от сезонности: пик активности приходится на тёплые месяцы года.

3. Постоянная потребность в оптимизации затрат на материалы, труд и логистику.

4. Широкий спектр используемых строительных материалов, который варьируется от стандартных (бетон, кирпич) до инновационных (композитные материалы, эко-решения).

Строительные материалы являются основой любой стройки, определяя качество и долговечность объектов. Рынок этих материалов характеризуется большим разнообразием и включает продукцию различных производителей: от крупных международных брендов до локальных компаний.

Основные категории строительных материалов:

1. Основные строительные материалы:

1.1. Бетон, цемент, песок, гравий.

1.2. Кирпич (керамический, силикатный).

1.3. Металлоконструкции.

2. Отделочные материалы:

2.1. Штукатурки, шпаклёвки, краски.

2.2. Плитка (керамическая, мраморная), ламинат, паркет.

2.3. Обои, декоративные панели.

3. Изоляционные материалы:

3.1. Теплоизоляция (минеральная вата, пенопласт).

3.2. Звукоизоляция.

3.3. Гидроизоляционные плёнки и мембраны.

4. Инженерные материалы:

4.1. Трубы и фитинги.

4.2. Электрические кабели, системы водоснабжения и вентиляции.

5. Инновационные и экологичные материалы:

5.1. Эко-материалы (солома, переработанный пластик, бамбук).

5.2. Самовосстанавливающийся бетон, «умные» стекла.

Факторы, определяющие выбор строительных материалов:

1. Качество и долговечность.

2. Стоимость. Это особенно важно для крупных проектов с жёстким бюджетом.

3. Эстетика и соответствие современным стандартам.

4. Экологичность и энергоэффективность.

Современный строительный бизнес ориентирован на внедрение новых технологий и материалов, что позволяет сокращать затраты и повышать производительность.

Основные тенденции:

1. Инновационные материалы.

1.1. Использование композитных материалов с высокой прочностью и низким весом.

1.2. Внедрение 3D-печати для создания строительных конструкций.

1.3. Применение «зелёных» материалов, способствующих снижению углеродного следа.

2. Цифровизация отрасли.

2.1. BIM-технологии (Building Information Modeling) для проектирования и управления строительными объектами.

2.2. Использование программного обеспечения для оценки смет и управления поставками строительных материалов.

3. Автоматизация процессов.

3.1. Использование роботов и дронов для выполнения строительных задач и мониторинга.

3.2. Внедрение умных систем на объектах строительства (контроль за качеством материалов, автоматическое управление климатом).

4. Устойчивое строительство.

4.1. Переход на возобновляемые источники энергии в строительных процессах.

4.2. Разработка зданий с нулевым потреблением энергии (zero-energy buildings).

Проблемы и вызовы строительной отрасли:

1. Колебания цен на строительные материалы. Рынок материалов подвержен влиянию внешних факторов, таких как экономическая нестабильность, инфляция, логистические проблемы.

2. Качество материалов. Риск покупки несертифицированной или контрафактной продукции, что может негативно сказаться на долговечности строительства.

3. Дефицит инновационных материалов. В некоторых регионах существует ограниченный доступ к современным материалам, что замедляет внедрение инноваций.

4. Управление информацией. Из-за обилия поставщиков и материалов застройщики часто сталкиваются с трудностями в анализе и сравнении предложений.

5. Экологические стандарты. Ужесточение экологических требований вынуждает компании искать более экологичные материалы и технологии.

Важность автоматизации сбора данных:

Автоматизация процессов, связанных с выбором и покупкой строительных материалов, становится ключевым фактором успеха. Компании и индивидуальные застройщики нуждаются в инструменте, который:

1. Позволяет быстро находить и анализировать предложения поставщиков.

2. Обеспечивает доступ к актуальной информации о ценах, характеристиках и доступности материалов.

3. Снижает риск человеческой ошибки при работе с большими объёмами данных.

Создание программы для автоматизированного сбора данных даст следующие преимущества:

1. Снижение издержек. Экономия времени и ресурсов за счёт автоматизации анализа рынка строительных материалов.

2. Повышение конкурентоспособности. Быстрая реакция на изменения цен и условий поставки.

3. Улучшение качества строительства. Использование только проверенных и подходящих материалов.

Таким образом, строительный бизнес нуждается в эффективных цифровых решениях для обработки данных, что делает тему дипломной работы актуальной и практически значимой.

# Анализ имеющихся решений.

В современном строительном бизнесе широко используются цифровые инструменты для поиска, анализа и закупки строительных материалов. Эти решения позволяют автоматизировать рутинные процессы и принимать более обоснованные решения, основываясь на актуальных данных. На рынке представлено несколько типов программного обеспечения, которые могут быть разделены на три основные категории:

1. Каталоги строительных материалов (онлайн-платформы).

2. Программы для управления строительными проектами (строительный ERP).

3. Универсальные парсеры данных и платформы сбора информации.

Каталоги строительных материалов. Это специализированные онлайн-платформы, предоставляющие информацию о продукции, ценах и наличии. Основная цель таких решений – упростить процесс поиска материалов для строителей и проектировщиков.

Примеры решений:

1. "Стройплощадка.ру", "Пульс Цен", "Маркет строительных материалов".

Основной функционал: поиск материалов по категориям, сравнение цен от разных поставщиков, контакты для заказа напрямую у производителей.

Преимущества: простота использования, доступность информации о продуктах, возможность сравнения поставщиков.

Ограничения: нет автоматического сбора информации с сайтов поставщиков, часто требуется ручной поиск и выбор.

2. "AliExpress для стройматериалов" (международные площадки).

Преимущества: широкий выбор продукции, включая инновационные материалы.

Ограничения: сложность работы с большими объёмами данных, отсутствие локальных поставщиков.

ERP-системы интегрируют управление строительными процессами, включая закупки материалов, расчёты смет и контроль затрат.

Примеры решений:

1. 1С: Управление строительной организацией.

Возможности: управление проектами, автоматизация смет, интеграция с базами данных, отслеживание поставок строительных материалов.

Преимущества: высокая надёжность, глубокая интеграция с бухгалтерскими и складскими процессами.

Недостатки: сложность настройки, не подходит для сбора данных с внешних источников (нужен ручной ввод или интеграция с другими программами).

2. PlanRadar, Procore.

Это системы управления строительными проектами, включая закупки материалов.

Ограничения: акцент на управлении проектами, а не на сборе данных с внешних источников.

Универсальные парсеры данных и платформы. Для автоматизации сбора информации многие компании используют универсальные инструменты веб-скрейпинга, которые можно адаптировать под сбор данных о строительных материалах.

Примеры решений:

1. BeautifulSoup, Scrapy (Python-библиотеки).

Используются для веб-скрейпинга HTML-страниц и извлечения неструктурированных данных.

Преимущества: гибкость и возможность адаптации, подходит для сбора данных с любых сайтов.

Ограничения: требуется высокая квалификация для настройки и поддержки, отсутствие встроенного анализа данных.

2. Octoparse, ParseHub. Платформы с графическим интерфейсом для веб-скрейпинга данных.

Преимущества: простота настройки для пользователей без навыков программирования. визуализация результатов.

Недостатки: ограниченные возможности обработки больших массивов данных. платная подписка.

Результаты сравнения указаны в таблице 1.

Таблица 1 – Результаты сравнения аналогов

|  |  |  |
| --- | --- | --- |
| Категория решений | Преимущества | Недостатки |
| Каталоги строительных материалов | Удобный интерфейс, информация о продуктах доступна сразу. | Ограничение данными только определённых поставщиков. |
| Строительный *ERP* | Интеграция всех процессов в компании, точность данных. | Сложность настройки, нет сбора данных с внешних сайтов. |
| Универсальные парсеры | Гибкость, возможность автоматического сбора больших объёмов данных. | Требуют навыков программирования или платных подписок. |

Категория решений Преимущества Недостатки

Каталоги строительных материалов Удобный интерфейс, информация о продуктах доступна сразу. Ограничение данными только определённых поставщиков.

Строительный ERP Интеграция всех процессов в компании, точность данных. Сложность настройки, нет сбора данных с внешних сайтов.

Универсальные парсеры Гибкость, возможность автоматического сбора больших объёмов данных. Требуют навыков программирования или платных подписок.

Проблемы существующих решений.

1. Ограниченность функционала каталогов. Каталоги предоставляют только уже готовую информацию, отсутствует гибкость в сборе данных из разных источников.

2. Сложность настройки универсальных парсеров. Самостоятельная настройка веб-скрейпинга требует знаний в программировании и создания сложных алгоритмов.

3. Отсутствие интеграции между системами. Например, ERP-системы не позволяют автоматически анализировать актуальные данные с сайтов поставщиков.

4. Неполная информация. В каталогах и ERP часто отсутствуют данные об уникальных или локальных поставщиках, что ограничивает возможности пользователей.

Потенциал разрабатываемого решения. На основании анализа можно выделить ключевые особенности, которые должны присутствовать в эффективной системе сбора данных о строительных материалах:

1. Автоматизация сбора информации. Интеграция данных из HTML-страниц (неструктурированные данные) и API поставщиков (структурированные данные).

2. Фильтрация и структурирование информации. Преобразование неструктурированных данных в удобный для анализа формат.

3. Удобство использования. Простота настройки и возможность быстрого анализа данных.

4. Гибкость. Возможность адаптации под потребности различных пользователей: строительных компаний, проектировщиков, частных застройщиков.

Предлагаемое решение будет эффективно сочетать возможности универсальных парсеров с пользовательской простотой, характерной для каталогов, и интеграционным потенциалом ERP. Это создаст новый уровень эффективности и конкурентоспособности для участников строительного рынка.

# Обоснование необходимости разработки и юридической правомерности.

В условиях современного строительного рынка, характеризующегося высокой волатильностью цен и постоянным обновлением ассортимента материалов, проблема оперативного получения достоверной информации приобретает особую актуальность. Традиционные методы сбора данных через ручной мониторинг сайтов поставщиков или работу с бумажными каталогами демонстрируют свою неэффективность, приводя к существенным временным и финансовым издержкам.

* + - 1. Основная проблема заключается в том, что при ручном сборе информации:
      2. Существенно возрастает вероятность ошибок ввода данных;
      3. Теряется оперативность получения информации;
      4. Отсутствует возможность комплексного анализа большого массива данных;
      5. Затруднено сравнение предложений различных поставщиков.

Разрабатываемое программное решение призвано устранить эти недостатки путем автоматизации процесса сбора и обработки информации. Его внедрение позволит строительным компаниям существенно сократить временные затраты на анализ рынка, повысить точность принимаемых решений и, как следствие, оптимизировать бюджет строительных проектов. Особенно важным представляется возможность оперативного реагирования на изменения рыночной конъюнктуры, что в современных экономических условиях является критически важным фактором конкурентоспособности.

С юридической точки зрения разработка подобных систем требует особого внимания к вопросам правомерности сбора и использования данных. В данном проекте предусмотрены следующие меры для обеспечения соответствия законодательным требованиям:

Во-первых, система будет работать только с общедоступной информацией, не требующей авторизации или специальных разрешений для доступа. Это исключает нарушение положений о защите персональных данных и коммерческой тайне. При разработке алгоритмов сбора данных особое внимание уделяется соблюдению условий пользовательских соглашений веб-ресурсов.

Во-вторых, частота запросов к источникам информации будет ограничена таким образом, чтобы не создавать избыточной нагрузки на серверы поставщиков. Это соответствует принципам добросовестного использования интернет-ресурсов и исключает возможность квалификации таких действий как несанкционированный доступ.

В-третьих, в алгоритмы обработки данных закладываются механизмы фильтрации и исключения любой информации, которая может быть отнесена к персональным данным или коммерческой тайне. Это обеспечивает соответствие требованиям Федерального закона "О персональных данных" и аналогичных международных нормативных актов.

Важно отметить, что разрабатываемая система не предусматривает хранение или обработку данных, защищенных авторским правом. Все собираемые сведения относятся исключительно к техническим характеристикам товаров и их стоимости, что соответствует практике добросовестной конкуренции.

Таким образом, предлагаемое решение не только отвечает насущным потребностям строительной отрасли в автоматизации процессов сбора и анализа информации, но и полностью соответствует действующим правовым нормам, что делает возможным его практическое применение в реальных бизнес-процессах строительных компаний.

# ПРОЕКТИРОВАНИЕ СИСТЕМЫ

# Требования к системе.

**Функциональные требования.**

Функциональные требования определяют, что система должна **делать**, то есть её основные возможности и поведение с точки зрения пользователя и разработчика.

1. **Парсинг данных с внешних сайтов**  
   Система должна автоматически собирать информацию о товарах (название, цена, описание, единица измерения и т.д.) со сторонних веб-ресурсов, специализирующихся на продаже строительных материалов. Источники должны быть настраиваемыми, а сам процесс парсинга — управляемым.
2. **Сохранение данных в базу данных**  
   Все полученные данные после парсинга должны сохраняться в централизованную реляционную базу данных с возможностью последующей обработки и визуализации. Должна обеспечиваться структурированная схема хранения информации, поддерживающая связи между товарами, категориями и источниками.
3. **Категоризация товаров**  
   Система должна классифицировать строительные материалы по категориям (например, цемент, кирпич, утеплители, пиломатериалы и т.д.). Классификация может быть выполнена автоматически (по ключевым словам) либо вручную через интерфейс администратора.
4. **Просмотр данных через веб-интерфейс.**  
   Пользователи должны иметь доступ к информации о товарах через интуитивно понятный веб-интерфейс. Интерфейс должен предоставлять базовую информацию о товаре, цену, дату последнего обновления, а также ссылку на оригинальный источник.
5. **Поиск и фильтрация.**  
   Интерфейс должен поддерживать возможность поиска товаров по ключевым словам, фильтрации по категориям, диапазону цен, датам изменения и другим параметрам.
6. **Загрузка данных**  
   Пользователи должны иметь возможность выгружать данные в форматах CSV и JSON для дальнейшего анализа или использования в сторонних системах.
7. **Построение графика динамики цен**  
   Для каждого товара должна быть возможность построения диаграммы изменения цены во времени за заданный период. Это позволит анализировать рыночные тенденции и сравнивать цены от разных поставщиков.

**Нефункциональные требования.**

Нефункциональные требования описывают **качество работы системы**, её надёжность, производительность, удобство использования и другие технические аспекты.

1. **Надёжность и сохранность данных**  
   Система должна обеспечивать устойчивое хранение и защиту информации. Все данные должны сохраняться в отказоустойчивой базе данных с регулярным резервным копированием. В случае сбоев не должно происходить потери данных.
2. **Регистрация и аутентификация пользователей**  
   Должна быть реализована система регистрации и входа с поддержкой безопасного хранения паролей (например, хеширование с солью).
3. **Многопоточность и параллельная обработка**  
   Система должна использовать многопоточность или асинхронную обработку при парсинге, чтобы одновременно обрабатывать данные с нескольких сайтов без снижения производительности.
4. **Отказоустойчивость**  
   В случае ошибок при подключении к источникам данных, сбоев в работе парсера или базы данных, система должна корректно логировать ошибки, сохранять текущее состояние и перезапускать процессы при необходимости.
5. **Производительность**  
   Интерфейс пользователя должен работать без заметных задержек, парсинг — происходить в разумные сроки (например, обработка 100 страниц за несколько минут), а построение графиков — выполняться мгновенно при наличии кэширования.
6. **Масштабируемость**  
   Архитектура системы должна предусматривать возможность расширения (например, добавление новых сайтов для парсинга, интеграция дополнительных аналитических модулей) без необходимости кардинальной переработки кода.
7. **Безопасность**  
   Система должна быть защищена от типовых веб-уязвимостей (SQL-инъекции, XSS, CSRF). Также следует ограничить количество попыток входа для предотвращения атак перебором.
8. **Интерфейс пользователя (UI/UX)**  
   Веб-интерфейс должен быть удобным, адаптивным под разные устройства (десктоп, планшет, смартфон), с поддержкой тёмной темы, если это предусмотрено.
9. **Кроссбраузерность**  
   Система должна корректно работать в популярных браузерах (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari).
10. **Логирование и мониторинг**  
    Все ключевые действия системы (например, успешный парсинг, ошибки, входы в систему, скачивание файлов) должны логироваться. Это поможет в отладке, аудите и выявлении проблем.
11. **Локализация и язык интерфейса**  
    Интерфейс системы должен быть реализован минимум на русском языке с возможностью добавления других языков в будущем.
12. **Обновляемость**  
    Код и компоненты системы должны быть разработаны с учётом возможности обновления — добавление новых функций, исправление багов и обновление зависимостей не должно требовать полной переработки.

# Функциональная модель системы

**Функциональная модель системы** – это абстрактное описание поведения системы с точки зрения её взаимодействия с внешними сущностями (акторами) и выполняемых функций (прецедентов). В контексте UML (Unified Modeling Language) такая модель чаще всего представляется в виде **диаграммы прецедентов (use case diagram)**, которая визуализирует функциональные требования к системе.

Ключевые элементы функциональной модели на основе диаграммы прецедентов:

1. **Система (System)** – границы анализируемой системы, внутри которой находятся прецеденты (функциональные возможности).
2. **Акторы (Actors)** – внешние сущности (пользователи, другие системы или устройства), взаимодействующие с системой.
   1. Первичные акторы – инициируют взаимодействие (например, пользователь).
   2. Вторичные акторы – участвуют в процессе, но не являются инициаторами (например, внешний сервис).
3. **Прецеденты (Use Cases)** – отдельные функции или сценарии использования системы, описывающие её поведение при взаимодействии с актором.
   1. Примеры: «Аутентификация пользователя», «Оформление заказа», «Формирование отчёта».
4. **Связи (Relationships)** – взаимодействия между акторами и прецедентами:
   1. Ассоциация – связь актора с прецедентом.
   2. Включение (Include) – обязательное использование одного прецедента внутри другого.
   3. Расширение (Extend) – условное добавление функциональности.
   4. Обобщение (Generalization) – наследование ролей.

Назначение функциональной модели:

1. Определение **границ системы** и её ключевых функций.
2. Формализация **требований** к системе на ранних этапах проектирования.
3. Упрощение коммуникации между заинтересованными сторонами (аналитиками, разработчиками, заказчиками).

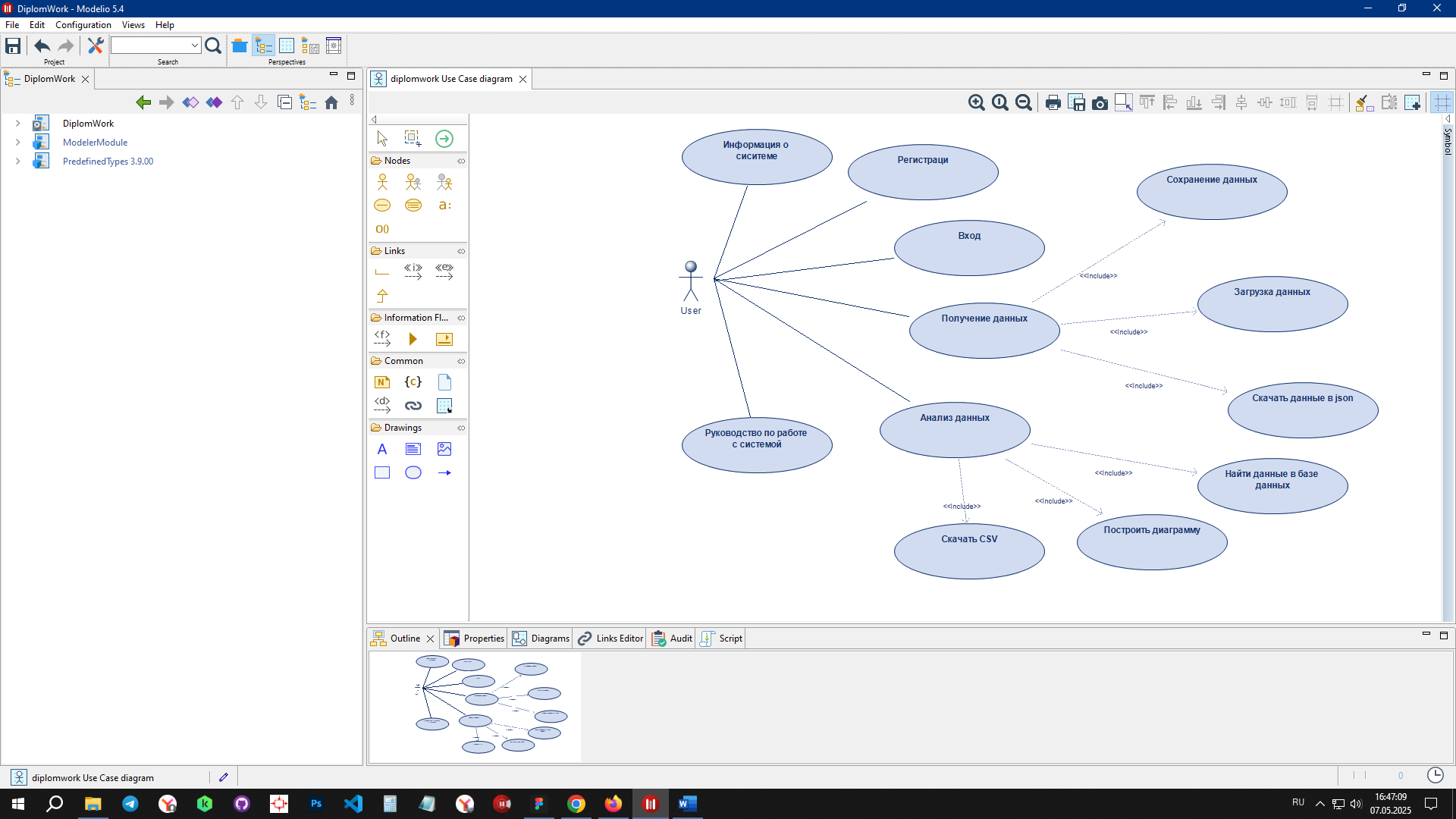


Рисунок 1 – диаграмма прецедентов

Данная диаграмма вариантов использования демонстрирует взаимодействие пользователя с системой сбора и анализа данных о строительных материалах.

Актор:

User (Пользователь) — основной актор, взаимодействующий с системой.

Основные варианты использования:

1. Информация о системе — пользователь получает общее описание функционала и возможностей платформы.
2. Регистрация / Вход / Выход — базовые функции аутентификации, позволяющие пользователю создать аккаунт, войти в систему и завершить сессию.
3. Получение данных — основной модуль, включающий в себя:
   1. Сохранение данных
   2. Загрузка данных
   3. Скачивание данных в JSON
   4. Поиск данных в базе  
      Эти функции обеспечивают взаимодействие с системой сбора и хранения информации: автоматический парсинг, добавление в БД, выгрузка и поиск.
4. Анализ данных — включает функции анализа собранной информации:
   1. Скачать CSV — экспорт результатов анализа.
   2. Построить диаграмму — визуализация изменения цены товара за выбранный период.
5. Руководство по работе с системой — помогает пользователю освоить функционал платформы, предоставляя справочную информацию.

# Архитектура системы

**Архитектура системы в контексте диаграммы развертывания** — это описание физической структуры программного обеспечения, включающее распределение компонентов системы по аппаратным узлам, связи между ними и инфраструктурные элементы, необходимые для функционирования системы.

**Ключевые аспекты архитектуры на диаграмме развертывания**

1. **Узлы (Nodes)**
   1. Представляют физические или виртуальные устройства (серверы, компьютеры, облачные инстансы, IoT-устройства).
   2. Примеры: Веб-сервер, Сервер БД, Клиентское устройство.
2. **Артефакты (Artifacts)**
   1. Исполняемые файлы, библиотеки, контейнеры (Docker), виртуальные машины.
   2. Примеры: backend.jar, nginx:latest, database.dump.
3. **Коммуникационные пути (Communication Paths)**
   1. Соединения между узлами (сети, протоколы).
   2. Примеры: HTTP/TCP, gRPC, WebSocket.
4. **Зависимости и связи**
   1. Показывают, какие компоненты развернуты на каких узлах и как они взаимодействуют.

**Цель создания:**

* 1. Понять, **как развертывается система** (локально, в облаке, гибридно).
  2. Определить **узкие места** (например, нагрузку на сервер БД).
  3. Спланировать **масштабирование** (репликация, балансировка).

Диаграмма развертывания в UML помогает визуализировать **"где что работает"** и **"как компоненты связаны физически"**. Это особенно важно для DevOps, архитекторов и администраторов. Диаграмма развертывания представлена на рисунке 2.

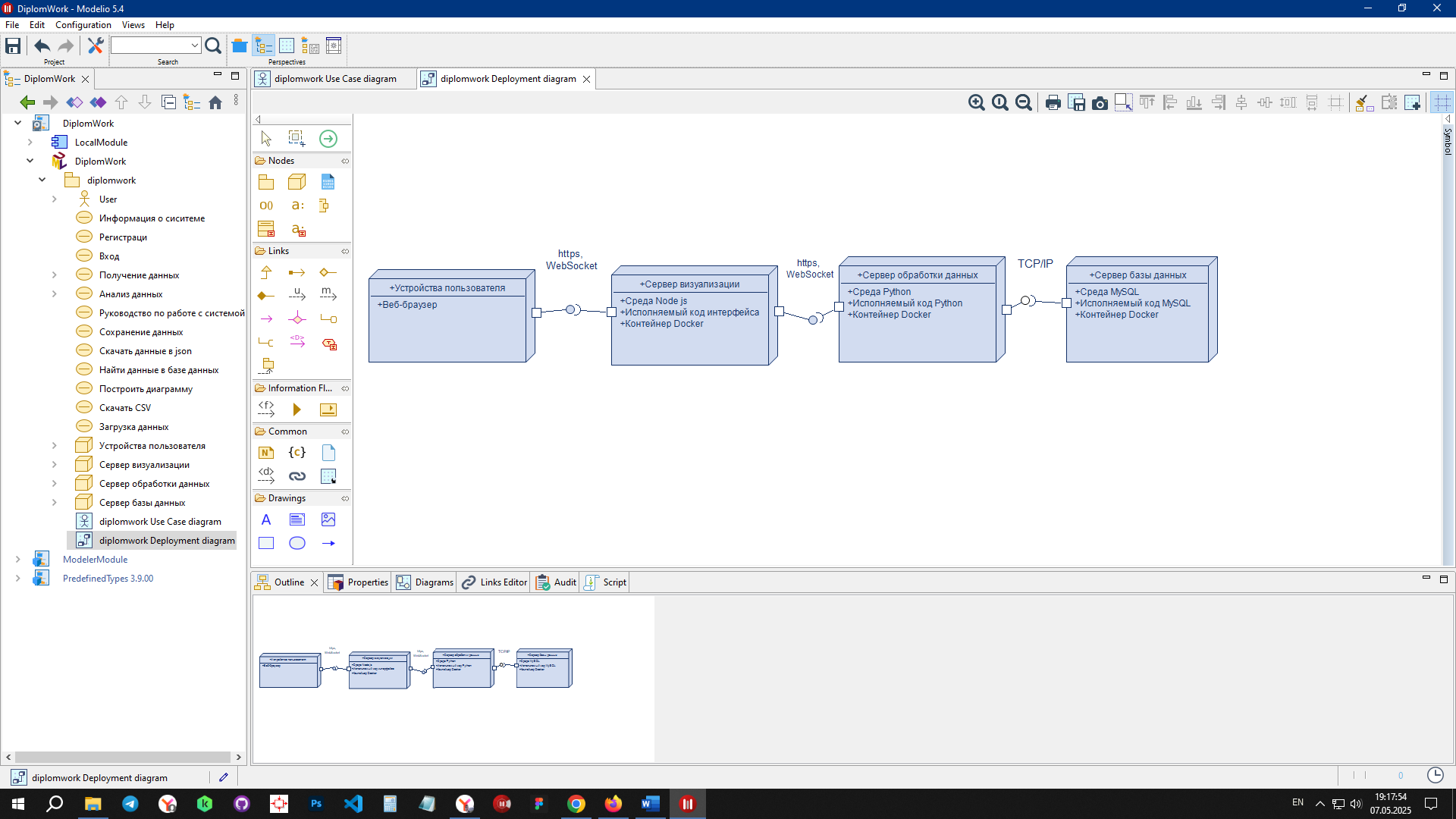


Рисунок 2 – диаграмма развертывания.

На диаграмме представлена архитектура системы, включающая взаимодействие пользовательских устройств с серверными компонентами через различные сетевые протоколы.

1. **Устройства пользователя**
   1. **ВКС-бюджер** – клиентское устройство (веб-браузер), которое взаимодействует с серверными компонентами.
2. **Серверные компоненты**
   1. **Сервер визуализации**
      1. **Протокол:** WebSocket (двустороннее взаимодействие в реальном времени) и TCP/IP.
      2. **Функции:**
         1. Обработка и отображение данных.
         2. Использует код интерфейса (фронтенд-логику).
      3. **Среда развертывания:** Docker (контейнеризация).
   2. **Сервер обработки данных**
      1. **Протокол:** WebSocket.
      2. **Функции:**
         1. Анализ и преобразование данных.
         2. Использует Python (бэкенд-логику и ML-модели).
      3. **Среда развертывания:** Docker.
   3. **Сервер файла данных (БД)**
      1. **Протокол:** TCP/IP (стандартный сетевой протокол для надежной передачи).
      2. **Функции:**
         1. Хранение и управление данными через MySQL.
         2. **Среда развертывания:** Docker.
3. **Взаимодействие между компонентами**
   1. **ВКС-бюджер** подключается к **серверу визуализации** через WebSocket.
   2. **Сервер визуализации** может запрашивать данные у **сервера обработки данных** (также через WebSocket).
   3. **Сервер обработки данных** взаимодействует с **сервером БД (MySQL)** по TCP/IP для чтения/записи данных.
4. Система использует микросервисную архитектуру с контейнеризацией (Docker), где:
   1. Визуализация и обработка данных работают через WebSocket (низкие задержки).
   2. Хранение данных обеспечивается классической SQL-БД (MySQL).
   3. Все компоненты изолированы и масштабируемы благодаря Docker.
      1. Такой подход подходит для систем реального времени (например, дашборды аналитики, IoT-платформы).

# Моделирование данных.

Для моделирования данных проектируемой системы использованы диаграмма классов и ER диаграмма.

**Диаграмма классов** — это один из типов диаграмм в языке моделирования UML (Unified Modeling Language), предназначенный для отображения структуры системы на уровне классов.  
Она показывает:

1. Классы системы
2. Их атрибуты (поля)
3. Методы (функции, поведение)
4. Связи между классами: ассоциации, наследование, агрегация, композиция

Диаграмма классов помогает разработчику и аналитикам понять архитектуру проекта, структуру данных и логические взаимосвязи объектов.

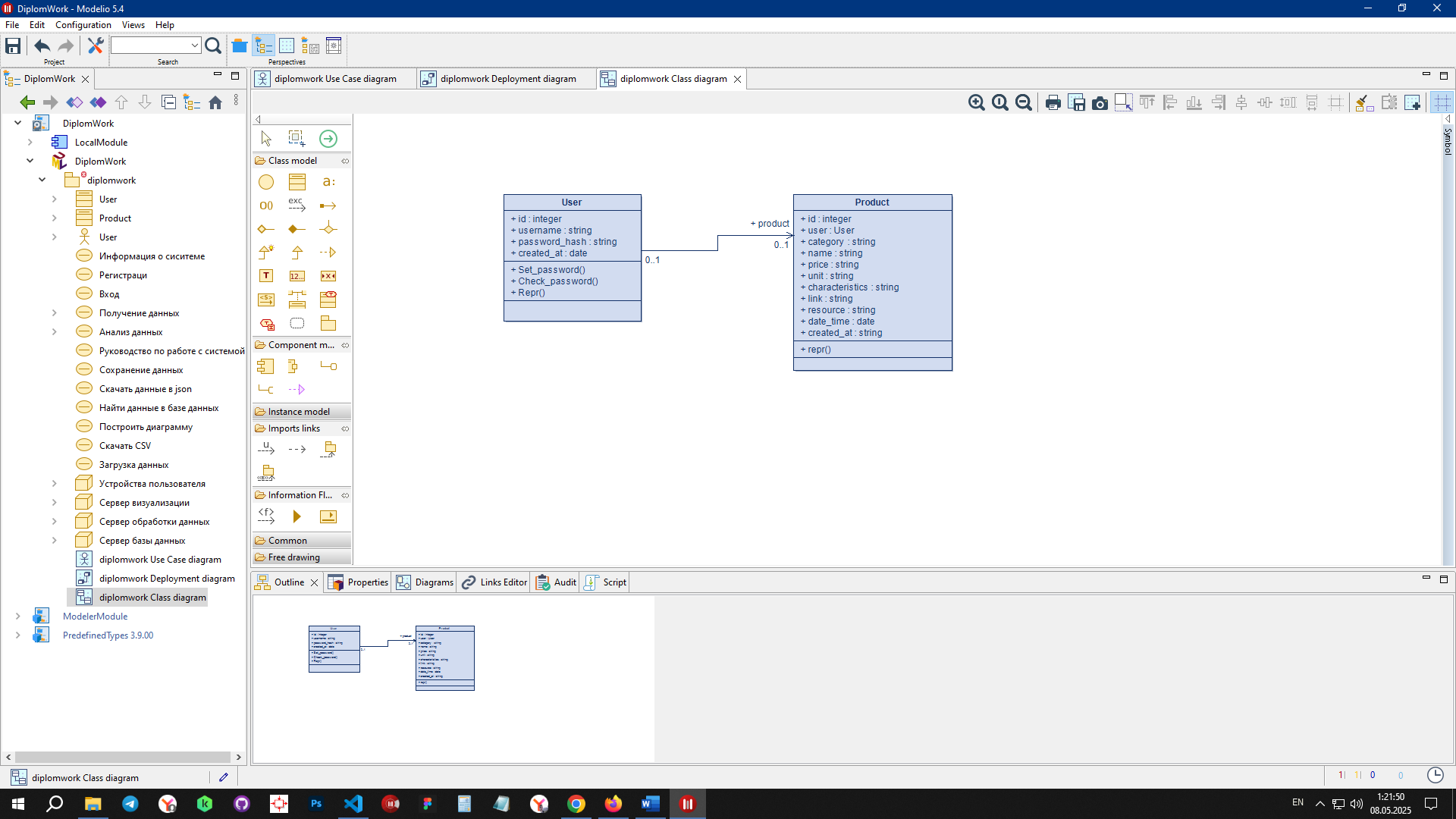


Рисунок 3 – Диаграмма классов.

Диаграмма отражает структуру двух основных классов: User и Product.  
Она реализует функциональность, где **один пользователь** может иметь **несколько загруженных продуктов** (парсинг данных).

Класс: User

**Сущность пользователя, который взаимодействует с системой.**

**Атрибуты:**

1. id : integer — уникальный идентификатор пользователя (первичный ключ)
2. username : string — имя пользователя
3. password\_hash : string — хэшированный пароль
4. created\_at : date — дата регистрации пользователя

**Методы:**

1. Set\_password() — установка пароля с последующим хэшированием
2. Check\_password() — проверка соответствия пароля
3. Repr() — строковое представление объекта (служебный метод)

Класс: Product

**Представляет продукт (товар со строительного сайта), загруженный пользователем.**

**Атрибуты:**

1. id : integer — уникальный идентификатор продукта
2. user : User — ссылка на пользователя, загрузившего продукт (внешний ключ)
3. category : string — категория продукта (например, цемент, кирпич и т.п.)
4. name : string — наименование товара
5. price : string — цена (возможно строкой для поддержки разных форматов)
6. unit : string — единица измерения (например, кг, м²)
7. characteristics : string — описание или характеристики товара
8. link : string — URL на страницу товара
9. resource : string — источник (например, название сайта)
10. date\_time : date — дата получения/обновления данных
11. created\_at : string — дата и время создания записи (в строковом формате)

**Методы:**

1. repr() — строковое представление объекта (служебный метод)

Тип связи между классами

1. Между User и Product установлена **ассоциация один-ко-многим**:
   1. Один User может быть связан с **множеством** объектов Product.
   2. У каждого Product может быть только **один** владелец (User).

**ER-диаграмма** — это визуальная модель данных, которая описывает:

1. **Сущности** (таблицы в БД) и их **атрибуты** (поля).
2. **Связи** между сущностями (один-ко-одному, один-ко-многим, многие-ко-многим).

**Ключевые элементы:**

1. **Сущность** — объект (например, Пользователь, Заказ).
2. **Атрибут** — свойство сущности (например, email, дата создания).
3. **Первичный ключ (PK)** — уникальный идентификатор (id).
4. **Внешний ключ (FK)** — ссылка на первичный ключ другой сущности (id пользователя).

**Типы связей:**

1. **1:1** (Один к одному) — например, паспорт и человек.
2. **1:N** (Один ко многим) — например, пользователь и его заказы.
3. **M:N** (Многие ко многим) — например, студенты и курсы (с промежуточной таблицей).

**Применение:**

1. Проектирование структуры базы данных.
2. Документирование требований к данным.
3. Обсуждение с командой перед реализацией.

ER диаграмма проектируемой системы представлена на рисунке 4.



Рисунок 4 – ER диаграмма.

На диаграмме представлена **модель данных** для проектируемой системы.

**Сущности (Entities)**

1. **Пользователь**
   1. Атрибуты:
      1. id (уникальный идентификатор)
      2. Логин
      3. Пароль
      4. Дата регистрации
2. **Данные**
   1. Атрибуты:
      1. id(уникальный идентификатор)
      2. Цена
      3. Дата сбора
      4. Учетная единица (например, штуки, литры)
      5. Ссылка (возможно, URL или путь к файлу)
      6. Описание
      7. Ресурс (основная информация)
      8. id пользователя (внешний ключ)

**Связи (Relationships)**

1. **Пользователь** → **Данные**
   1. Связь: «1 ко многим» (один пользователь может иметь множество записей данных).
   2. Обозначение: «Имеет» (стрелка или линия с пометкой).

# Моделирование логики работы

Диаграмма процесса работы системы в нотации BPMN (Business Process Model and Notation) представляет собой инструмент визуального моделирования, предназначенный для описания последовательности операций, принимаемых решений, взаимодействий между участниками процесса и внешними системами. Основное назначение BPMN-диаграммы — обеспечить понятное и стандартизированное отображение логики функционирования системы или процесса, которое будет одинаково доступно для анализа как техническими специалистами, так и пользователями.

Моделирование процессов с использованием BPMN позволяет формализовать поведение системы на различных этапах, выделить роли участников, показать порядок выполнения задач, условия переходов и варианты развития сценариев. Это особенно важно при проектировании программных решений, так как способствует выявлению узких мест, потенциальных ошибок в логике и неочевидных зависимостей между компонентами. Диаграмма способствует унификации понимания требований и процессов между всеми участниками разработки.

BPMN поддерживает использование таких элементов, как события (начало, завершение, промежуточные сигналы), задачи (автоматические или выполняемые вручную), шлюзы (для описания условий и ветвлений) и потоки управления. Благодаря своей гибкости и четкой структуре, нотация BPMN позволяет как моделировать простые пользовательские сценарии, так и описывать комплексную архитектуру взаимодействия системных компонентов.

Таким образом, построение BPMN-диаграммы является важным этапом анализа и проектирования логики функционирования информационной системы. Она способствует более глубокому пониманию процессов, облегчает сопровождение и развитие системы, а также может быть использована как часть документации при защите проекта или передаче его в дальнейшую разработку.

Диаграмма процесса взаимодействия пользователя и проектируемой системы в нотации BPMN представлена на рисунке 5.

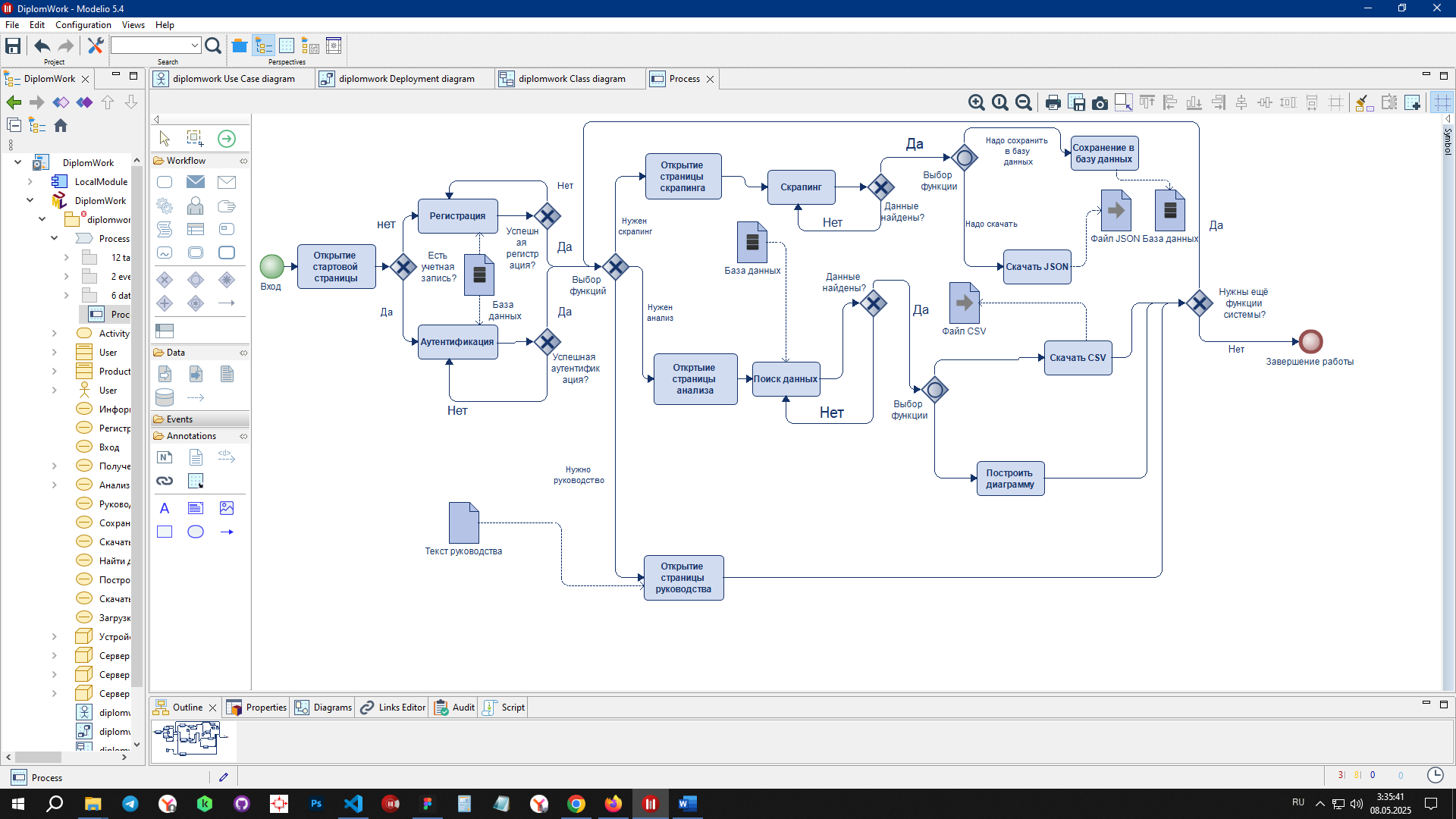


Рисунок 5 - Диаграмма процесса взаимодействия пользователя и проектируемой системы

# РАЗРАБОТКА СИСТЕМЫ

# Этапы разработки (Последовательность выполнения действий для разработки системы).

# Средства разработки (стек технологий и аппаратное обеспечение).

# Описание системы

# Интерфейс пользователя (прототипы в Фигме)

# Дизайн (красивое/не красивое)

# Функциональные блоки (Список страниц отличных по наполнению и их описание)

# Серверная часть системы (потоки http запросов от клиента к серверу и обратно)

# Описание контроллеров (конченых точек и их описание)

# Описание моделей (как и где тащим данные из базы или загружаем их, схема бд)

# Алгоритмы обработки данных (как мы html в текст превращали, где и как это сделать)

# Входные данные (параметры для веб-скрапинга)

# Обработка данных (Определяем, что за тип файла, определяем структуру, отсекаем ненужное, тащим нужное)

# Выходные данные (Описание того, что вытащили)

# Тестирование системы (зачем тестируем, что тестируем)

# Методика тестирования системы (интеграционное, функциональное, модульное)

# Результаты тестов (тестировали то то, получили то то)

# ПЕРСПЕКТИВА РАЗВИТИЯ СИСТЕМЫ

# ЗАКЛЮЧЕНИЕ

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ