бюджетное профессиональное образовательное учреждение Вологодской области
«Череповецкий лесомеханический техникум им. В.П. Чкалова»

Специальность **09.02.07** «Информационные системы и программирование»

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

ПП по ПМ.05 Проектирование и разработка информационных систем

Выполнил студент 3 курса группы ИС-32
Аксёнова Ксения Максимовна
подпись _____
место практики

_____
наименование юридического лица, ФИО ИП

Период прохождения:
с «25» мая 2025 г.

по «07» июня 2025 г.

Руководитель практики от предприятия
должность_____

_____

подпись_____

МП
Руководитель практики от техникума:
Материкова А.А.
Оценка:_____
«___» _____2025 года

г. Череповец 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

- Цель и задачи практики.

Производственная практика направлена на формирование у обучающего ся общих и профессиональных компетенций, приобретение практического опыта и реализуется в рамках модуля ПМ.05 специальности 09.02.07 Информационные системы и программирование.

Задачи:

1. Формирование общих компетенций.
2. Формирование профессиональных компетенций: проектирование и разработка информационных систем, выполнять анализ предметной области, основные модели построения информационных систем, платформы для создания, исполнения и управления информационной системой.
3. Приобретение практического опыта в: управлении процессом разработки приложений с использованием инструментальных средств, обеспечении сбора данных для анализа использования и функционирования информационной системы, программировании в соответствии с требованиями технического задания, определения состава оборудования и программных средств разработки информационной системы, разработке документации по эксплуатации информационной системы.

• Краткое описание организации, где проходила практика

Малленом Системс — это одна из ведущих компаний в России, занимающаяся разработкой и внедрением инновационных систем компьютерного зрения, а также промышленной видеоаналитики и интеллектуальной обработки данных. В центре её решений лежат современные технологии машинного зрения и искусственного интеллекта, включая алгоритмы машинного обучения и глубокие нейронные сети.раммировании в соответствии с требованиями технического задания, определения состава оборудования и программных средств разработки

информационной системы, разработке документации по эксплуатации информационной системы.

• Сроки и место прохождения.

Срок прохождения практики с 25.05.25 по 7.06.25, ООО “Малленом Системс”, практика проходилась дистанционно.

1 ОБЩАЯ ХАРАКТЕРИСТИКА ПРЕДПРИЯТИЯ (ОРГАНИЗАЦИИ)

Общая информация (сфера деятельности)

Компания "Малленом Системс" специализируется на разработке и внедрении интеллектуальных систем обработки и анализа изображений, решений в области промышленного зрения, технического зрения и автоматизации. Основные направления деятельности включают:

• разработку программного и аппаратного обеспечения для систем машинного зрения;

• автоматизацию процессов контроля качества на производственных предприятиях;

• создание интеллектуальных систем распознавания (например, номеров автомобилей, объектов на конвейерах и др.);

• интеграцию решений в промышленные комплексы.

Роль информационных систем (ИС) в работе организации

Информационные системы играют ключевую роль в деятельности компании, поскольку она напрямую связана с разработкой и внедрением программных решений. Основные аспекты использования ИС:

• Автоматизация процессов разработки — системы управления версиями, среды разработки и трекеры задач помогают командам эффективно взаимодействовать.

• Разработка программного обеспечения — ИС используются на всех этапах: от сбора требований и проектирования до тестирования и внедрения.

• Анализ изображений и видео — критически важный элемент продуктов компании, реализуемый средствами ИИ и компьютерного зрения.

• Хранение и обработка данных — информационные системы обеспечивают надёжную работу с большими объёмами данных, получаемых от промышленных камер и сенсоров.

• Контроль и мониторинг проектов — с помощью ИС ведётся управление проектами, задачами и версиями ПО.

Таким образом, ИС являются не вспомогательным, а центральным инструментом в деятельности предприятия.

Основные используемые технологии

Языки программирования:

• C# — основной язык для разработки десктопных и встроенных решений.

• Python — применяется для задач машинного обучения, прототипирования и анализа данных.

СУБД:

• PostgreSQL, SQLite, MySQL — используются в зависимости от конкретного проекта.

• Также применяются файловые хранилища и специализированные форматы данных (например, для хранения изображений, логов, конфигураций).

• Системы контроля версий:

• Git (используется повсеместно).

• Репозитории размещаются в GitLab и GitHub.

• Фреймворки и библиотеки:

• .NET / .NET Core — основа серверных и клиентских приложений на C#.

• OpenCV — для компьютерного зрения.

• TensorFlow / PyTorch — в проектах, связанных с нейросетями и ИИ.

• Flask / FastAPI — для создания API на Python.

Инструменты проектирования:

• Draw.io, Lucidchart, Enterprise Architect — для создания диаграмм UML и ER-моделей.

• PlantUML — для генерации схем на основе текстового описания.

• Методологии разработки:

• Agile / Scrum / Kanban — гибкие методологии управления проектами.

• Часто используются итеративные подходы и CI/CD-практики.

Системы управления задачами:

• Jira, Redmine, Trello, YouTrack — применяются для постановки задач, контроля прогресса и ведения документации.

• Также возможна интеграция с GitLab Issues.

2 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННЫХ СИСТЕМ

2.1. Анализ требований к информационным системам

Анализ требований — критический начальный этап жизненного цикла информационной системы, определяющий успех разработки. Он включает сбор функциональных (что должна делать система) и нефункциональных (как она должна работать: надежность, производительность, безопасность) требований, а также выявление и устранение конфликтов между участниками проекта.

Для формализации и визуализации требований используется моделирование, помогающее понять архитектуру системы и выявить недостатки до кодирования. Применяются знаковые модели: диаграммы потоков данных (DFD), UML-диаграммы (вариантов использования, классов, последовательностей) и ER-модели. Могут также разрабатываться интерактивные прототипы UI.

Например, для системы учёта сотрудников ER-модель покажет сущности («Сотрудник», «Должность», «Отдел») и их связи, а DFD — процессы приёма и увольнения.

2.2. Проектирование информационных систем

Этап проектирования включает выбор архитектурного подхода и построение логической структуры системы. Разработчики могут использовать функционально-ориентированную модель, при которой система разбивается на иерархию функций, или объектно-ориентированную, где основное внимание уделяется данным и их взаимосвязям (объектам и классам).

Ключевое значение имеет выбор архитектуры. Примеры:

• Клиент-серверная архитектура — клиентские приложения обращаются к серверу за обработкой данных, что снижает требования к мощности клиентского оборудования.

• Многоуровневая архитектура (например, трёхзвенная) — включает уровни представления, бизнес-логики и данных, что обеспечивает более высокий уровень безопасности, удобство масштабирования и управления кодом.

• Подбор технологий происходит с учётом требований к системе:

• Для работы со структурированными данными обычно применяются реляционные СУБД (PostgreSQL, MySQL);

• При необходимости обрабатывать неструктурированные или гибкие форматы данных (например, JSON-документы) выбираются NoSQL-решения (MongoDB, Couchbase).

Дополнительно учитываются особенности среды исполнения — например, использование микросервисной архитектуры в облачных решениях.

2.3. Разработка информационных систем

В процессе разработки необходимо определить методологию управления проектом. Для проектов с чётко установленными требованиями применяется каскадная модель (Waterfall), где этапы (анализ, проектирование, кодирование, тестирование, внедрение) выполняются последовательно. Однако этот подход слабо адаптируется к изменяющимся условиям.

Современные практики предпочитают гибкие методологии (Agile, Scrum, Kanban), при которых продукт создаётся итерационно с постоянной обратной связью от пользователей. Это позволяет своевременно вносить корректировки и ускоряет выход промежуточных версий.

Инструменты разработки включают:

• Системы контроля версий — например, Git, обеспечивающий совместную работу над кодом и отслеживание изменений;

• Среды разработки (IDE) — такие как Visual Studio, PyCharm, VS Code;

• Фреймворки — для веб-разработки популярны Django (Python), ASP.NET (C#), React, Angular, которые ускоряют создание интерфейсов и бизнес-логики.

Крайне важна техническая и пользовательская документация: спецификации API, примеры использования, инструкции для администраторов и конечных пользователей. Она облегчает ввод новых разработчиков в проект, поддержку и дальнейшее развитие системы.

2.4. Тестирование информационных систем

Цель тестирования — подтвердить соответствие разработанной системы требованиям и выявить возможные дефекты. Тестирование проводится на нескольких уровнях:

• Модульное — проверка работы отдельных функций или классов.

• Интеграционное — оценка взаимодействия между модулями.

• Системное и приёмочное — проверка всей системы на соответствие бизнес-целям.

Нагрузочное тестирование особенно важно для систем с высокой посещаемостью. Оно позволяет определить, сколько пользователей система выдержит одновременно, и при каких условиях начинаются сбои.

Безопасность тестируется с использованием специализированных инструментов — например, OWASP ZAP или Burp Suite, позволяющих находить уязвимости, такие как SQL-инъекции, межсайтовое выполнение скриптов (XSS) и др.

Для повышения эффективности применяются средства автоматизированного тестирования и непрерывной интеграции/доставки (CI/CD), например, Jenkins, GitLab CI, GitHub Actions. Эти инструменты автоматически запускают тесты при каждом обновлении кода, что помогает оперативно выявлять ошибки.

2.5. Внедрение, эксплуатация и сопровождение информационных систем

Фаза внедрения предполагает установку и конфигурацию системы, обучение пользователей и технического персонала. Один из подходов — параллельный запуск, при котором старая и новая системы работают одновременно, снижая риск ошибок при переходе.

Во время эксплуатации проводится мониторинг работы системы, сбор логов и метрик, таких как загрузка процессора, объём памяти, отклик API. Популярные

инструменты мониторинга — Prometheus, Grafana, Zabbix.

Сопровождение включает:

• устранение найденных ошибок;

• выпуск обновлений;

• оптимизацию производительности;

• обеспечение масштабируемости по мере роста нагрузки;

• регулярное резервное копирование данных и реализацию плана аварийного восстановления (Disaster Recovery).

Это обеспечивает стабильную работу ИС в долгосрочной перспективе и минимизирует возможные потери данных при сбоях.

3 ВЫПОЛНЯЕМЫЕ ЗАДАНИЯ

Задание 1

1) Анализ и освоение методологии управления проектами на основе Kanban.

Методология Kanban берет свое начало в японской автомобильной промышленности конца 1950-х годов. Изначально она применялась на производственных линиях: мастера на каждом этапе работы использовали карточки, размещаемые на общей доске, чтобы отслеживать и координировать рабочий процесс. Это позволяло добиться высокой прозрачности выполнения задач и оперативно реагировать на изменения в загрузке.

Со временем Kanban эволюционировала в популярную гибкую методологию управления задачами в самых разных областях, особенно в сфере информационных технологий. Сегодня она активно применяется в разработке программного обеспечения, технической поддержке, управлении контентом, маркетинговых и исследовательских командах.

В рамках изучения этой методологии был проведен обзор нескольких современных инструментов, реализующих принципы Kanban:

Trello – интуитивно понятный визуальный инструмент, идеально подходящий для небольших коллективов и персонального планирования. Позволяет группировать задачи по стадиям выполнения и легко перемещать карточки между колонками.

Jira – мощная корпоративная платформа, широко применяемая в больших компаниях. Обладает гибкой настройкой бизнес-процессов, системой приоритетов, поддержкой Agile и возможностью интеграции с другими сервисами.

GitLab – многофункциональный сервис с открытым исходным кодом, включающий в себя возможности ведения репозиториев, CI/CD, трекинга задач и документации через встроенный Wiki. Удобен для полного цикла разработки.

Azure DevOps – комплексное решение от Microsoft, обеспечивающее

поддержку всех этапов разработки: от планирования и контроля версий до автоматизации сборок и развёртывания.

2) Выбор подходящей системы для управления проектной работой.

Из проанализированных решений предпочтение было отдано GitLab. Основной причиной стало наличие полной интеграции с GitHub, что облегчает командную работу над проектом и упрощает управление версиями. Кроме того, GitLab предоставляет удобный интерфейс для работы с задачами, визуализацию прогресса и возможность создания отдельных веток под каждую задачу.

3) Создание репозитория и размещение проектных материалов.

После выбора инструмента управления проектом был инициализирован репозиторий, содержащий исходный код и сопутствующие материалы по разработке. Этот репозиторий служит центральным хранилищем для организации всей проектной документации, реализации, и контроля задач.

На Рисунке 1 представлен внешний вид репозитория проекта tirecontrol в системе GitLab.

Задание 2

1) Формулировка направления разработки.

В качестве темы проекта была выбрана разработка подсистемы хранения данных для мониторинга состояния подшипников. Данная подсистема предназначена для работы с техническими данными о подшипниках: она позволяет собирать, обрабатывать, хранить и управлять такими параметрами, как вибрация, температура, износ и другие метрики.

Сервис, входящий в подсистему, должен обеспечивать базовые функции: добавление новых записей, удаление устаревших данных, обновление информации и предоставление доступа к ним другим компонентам системы.

Пример: в случае установки на промышленное оборудование, подсистема будет собирать данные с датчиков, определять отклонения от нормы и передавать их в модуль визуализации для дальнейшего анализа специалистами.

2) Планирование работ и организация задач с использованием Kanban-доски.

Для систематизации всех этапов разработки был составлен подробный план с указанием задач, сроков выполнения и последовательности их реализации. Каждой задаче назначены конкретные исполнители и определена степень приоритета.

Все задачи занесены на доску Kanban, созданную в GitLab (см. Задание 1). На доске визуально отражены этапы реализации: от постановки задачи до её завершения, с возможностью отслеживания текущего статуса.

Примерные категории колонок:

To Do (К выполнению) — задачи, которые нужно реализовать.

In Progress (В процессе) — задачи, над которыми ведётся работа.

Review (На проверке) — задачи, требующие оценки/тестирования.

Done (Выполнено) — завершённые задачи.

1) Составление документации для выбранной подсистемы:

1. Техническое задание.

Техническое задание

1. Общая информация

Название проекта: Подсистема хранения данных для мониторинга состояния подшипников

2. Цель и задачи проекта

Цель: Создание надёжной и масштабируемой подсистемы для хранения и анализа параметров состояния подшипников на производстве.

Задачи:

- Проектирование структуры базы данных
- Реализация REST API
- CRUD-операции с данными
- Фильтрация и сортировка по параметрам
- Обеспечение безопасности и валидации

2. Функциональные требования

2.1. Параметры хранения:

Каждая запись включает: ID, название, вес, размер, тип, скорость, трение, ресурс, зазор, температуру, вибрацию, давление (опц.), износ, статус и др.

2.2. REST API:

- GET /bearings — все записи
- GET /bearings/{id} — конкретная запись
- POST /bearings — добавление
- PUT /bearings/{id} — редактирование
- DELETE /bearings/{id} — удаление
- GET /bearings/filter?status=... — фильтрация по статусу

2.3. Роли пользователей:

- Администратор — полный доступ
- Пользователь — просмотр и фильтрация

3. Нефункциональные требования

- Язык: Python 3.10+
- Фреймворк: FastAPI (или Flask)
- БД: PostgreSQL 13+
- Формат: JSON
- Отклик API: ≤500 мс
- Доступность: ≥99.5%
- Безопасность: аутентификация, валидация, защита от SQL-инъекций
- Документация: OpenAPI (Swagger)

4. Сценарии использования

- Добавление: оператор отправляет POST-запрос

• Мониторинг: отображение и фильтрация по статусу
• Обновление: редактирование параметров
• Удаление: администратор удаляет записи

5. Этапы и сроки реализации представлены в таблице 1.

Этап	Задача	Сроки
Этап	Задача	Сроки
1	Анализ и проектирование БД	25.05.2025 – 26.05.2025
2	Настройка проекта	27.05.2025
3	Реализация API	28.05.2025 – 30.05.2025
4	Тестирование и отладка	31.05.2025 – 02.06.2025
5	Документация (Swagger, README)	03.06.2025
6	Финализация и отчёт	04..06.2025 – 06.06.2025

6. Критерии приёмки

- API работает без ошибок
- Эндпоинты соответствуют требованиям
- Приложение развёрнуто (локально или в Docker)
- README и документация присутствуют
- CRUD-операции выполняются корректно

7. Среда развертывания

- ОС: Ubuntu / Windows
- ЯП: C++
- Фреймворк: FastAPI
- БД: PostgreSQL 13+
- Контейнеризация: Docker (опц.)
- Контроль версий: Git (GitLab/GitHub)

8. Документация

- Техническое задание
- Руководство пользователя и администратора
- ER и UML-диаграммы (варианты использования, последовательности, компоненты, пакеты, деятельность)

Руководство пользователя и администратора

Подсистема хранения данных для мониторинга состояния подшипников

1. Назначение: Руководство предназначено для пользователей и администраторов подсистемы, обеспечивающей сбор, хранение и анализ данных о состоянии подшипников (вибрация, температура и т.п.).
2. Часть 1. Руководство пользователя
3. Возможности подсистемы
   - Сбор и хранение данных от датчиков.
   - Просмотр текущих и архивных данных.
   - Фильтрация, поиск и визуализация информации.
   - Экспорт в форматы CSV, Excel, PDF.
4. Работа с интерфейсом

Вход: Перейдите по адресу, введите логин и пароль. Навигация:

- Главная: Сводка по подшипникам
- История: Архивные данные
- Отчёты: Генерация по параметрам

Настройки: Персонализация Просмотр данных:

Выберите подшипник, интервал и нажмите "Показать".

Экспорт: После отображения данных выберите формат и сохраните файл. Часть 2. Руководство администратора

5. Установка и настройка Требования:
   - ОС: Windows Server 2016+
   - ОЗУ: ≥16 ГБ
   - Диск: ≥100 ГБ
   - СУБД: SQL Server 2017+ Установка:

Следуйте шагам установщика, укажите СУБД, завершите установку. Настройка:

- Подключите датчики
- Задайте расписание сбора
- Назначьте роли и доступ

6. Управление пользователями
   - Добавление: Введите данные, назначьте роль
   - Редактирование: Обновите информацию
   - Удаление: Подтвердите удаление
7. Мониторинг и обслуживание

◦ Проверяйте журналы событий

◦ Настройте резервное копирование

◦ Устанавливайте обновления своевременно

8. Безопасность

◦ Используйте сложные пароли и 2FA

◦ Включите шифрование данных

◦ Минимизируйте права доступа пользователей

◦ Составление диаграмм и их описание

1. ER-диаграмма: Визуализирует взаимосвязи данных (сущностей), их характеристики и взаимодействия в базе данных.

2. Диаграмма вариантов использования: Описывает взаимодействие пользователей с системой и её основные функции с их точки зрения.

3. Диаграмма последовательностей: Детализирует временное взаимодействие объектов в системе для выполнения задачи, показывая хронологию обмена сообщениями.

4. Диаграмма компонентов: Отображает физическую структуру системы как совокупность развёртываемых модулей (компонентов), их взаимосвязи и зависимости.

5. Диаграмма пакетов: Организует элементы системы в логические группы (пакеты), демонстрируя их иерархию и зависимости для структурирования больших проектов.

6. Диаграмма деятельности: Моделирует последовательность действий в бизнес- процессах, включая шаги, точки принятия решений, параллельные процессы и условия перехода.

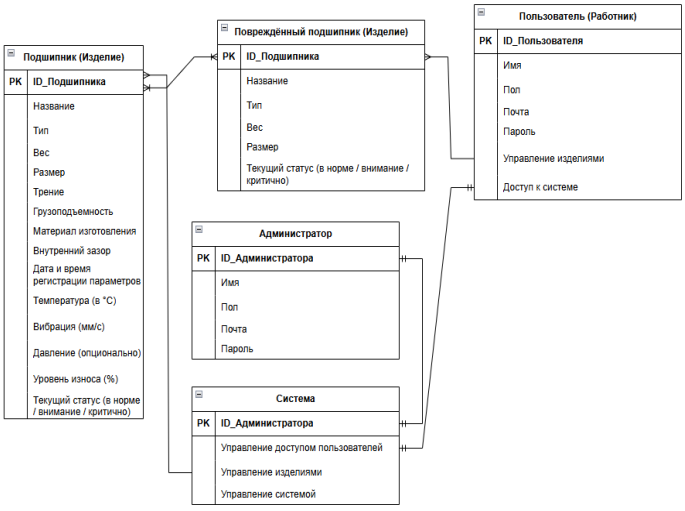

Рисунок 1 – ER диаграмма

1. Диаграмма вариантов использования.

Внешний вид диаграммы вариантов использования рисунок 6.

2. Диаграмма последовательностей.

Внешний вид диаграммы последовательности рисунок 7.

3. Диаграмма компонентов.

Рисунок 2 – Диаграмма компонентов

3. Диаграмма пакетов.

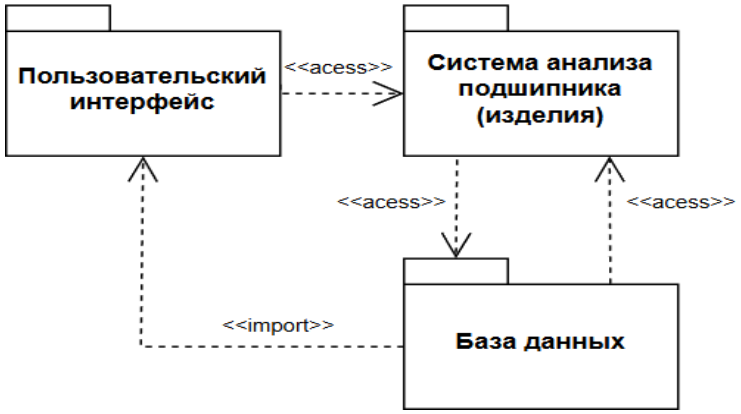

Рисунок 3 – Диаграмма пакетов

4. Диаграмма деятельности.

Внешний вид диаграммы деятельности рисунок 8.

ЗАКЛЮЧЕНИЕ

Прохождение практики в компании ООО «Малленом Системс» стало для меня ценным этапом профессионального развития и дало возможность значительно расширить спектр прикладных знаний в области проектирования и создания информационных систем. В рамках практики я не только углубила теоретические знания, полученные в учебном процессе, но и на практике освоила работу с различными инструментами проектирования, научилась строить ER-диаграммы, разрабатывать техническую документацию и структурировать задачи в канбан-системах.

Особенно важным для меня стало знакомство с корпоративными подходами к управлению проектами. Использование Kanban-досок позволило лучше понять, как внутри компаний выстраиваются процессы планирования и реализации задач, как организуется командная работа и каким образом осуществляется контроль сроков. Благодаря этому опыту я научилась самостоятельно ставить задачи, приоритизировать их, следить за сроками выполнения и оперативно вносить коррективы в план действий.

Практика дала мне возможность погрузиться в разработку реального подсистемного решения — системы хранения данных для мониторинга состояния подшипников. Я научилась проектировать базу данных на основе требований к системе, составлять грамотное техническое задание, разрабатывать структурные схемы и документировать процессы. Такой опыт особенно ценен, поскольку он приближает к условиям работы в ИТ-индустрии, где от специалиста требуется не только знание языков программирования, но и понимание архитектурных принципов, стандартов документации и умение работать в команде.

Полученные знания и навыки стали важным шагом на пути к моей профессиональной самореализации. Я уверена, что опыт, приобретенный в ходе практики, послужит прочной основой для моей будущей карьеры в области информационных технологий и поможет успешно решать реальные производственные задачи.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. 1. Методы анализа требований [Электронный ресурс]/ - Режим доступа: https://logrocon.ru/news/requirements_analysis2

2. 2. Графический редактор диаграмм [Электронный ресурс]/ Режим доступа: https://www.drawio.com/

3. 3. Методы моделирования и модели разработки ИС [Электронный ресурс]/ - Режим доступа: https://inftis.narod.ru/pis/pis-p3-1.htm

4. 4. Основные этапы проектирования ИС [Электронный ресурс]/ - Режим доступа: https://scilead.ru/article/714-osnovnie-etapi-proektirovaniya-informatsionnik

5. 5. Тестирование информационных систем [Электронный ресурс]/ - Режим доступа: https://at-consulting.ru/testirovanie-informacionnyh-sistem

6. 6. Разработка и внедрение информационных систем [Электронный ресурс]/ - Режим доступа: https://infocom-s.ru/sozdanie-informacionnyh-sistem

7. 7. Примеры UML диаграмм [Электронный ресурс]/ - режим доступа: https://practicum.yandex.ru/blog/uml-diagrammy/?

8. 8. Разработка требований к ИС [Электронный ресурс]/ - Режим доступа: https://searchinform.ru/services/outsource-ib/zaschita-informatsii/razrabotka-trebovanij-po-ib/razrabotka-trebovanij-k-informatsionnoj-sisteme/

9. 9. Методы проектирования ИС [Электронный ресурс]/ - Режим доступа: https://studfile.net/preview/9057964/page:87/

10. 10. Система управления проектами [Электронный ресурс]/ - Режим доступа: https://shtab.app/

ПРИЛОЖЕНИЕ

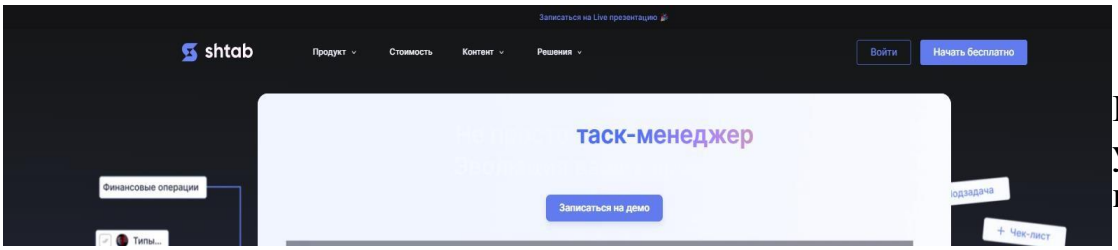

Рисунок 4 – Скриншот сайта выбранной системы управления проектами

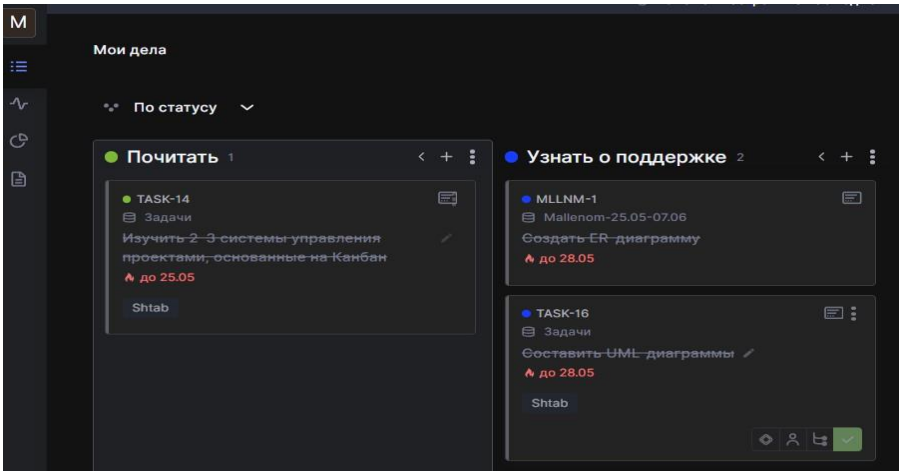

Рисунок 5 – Доска канбан

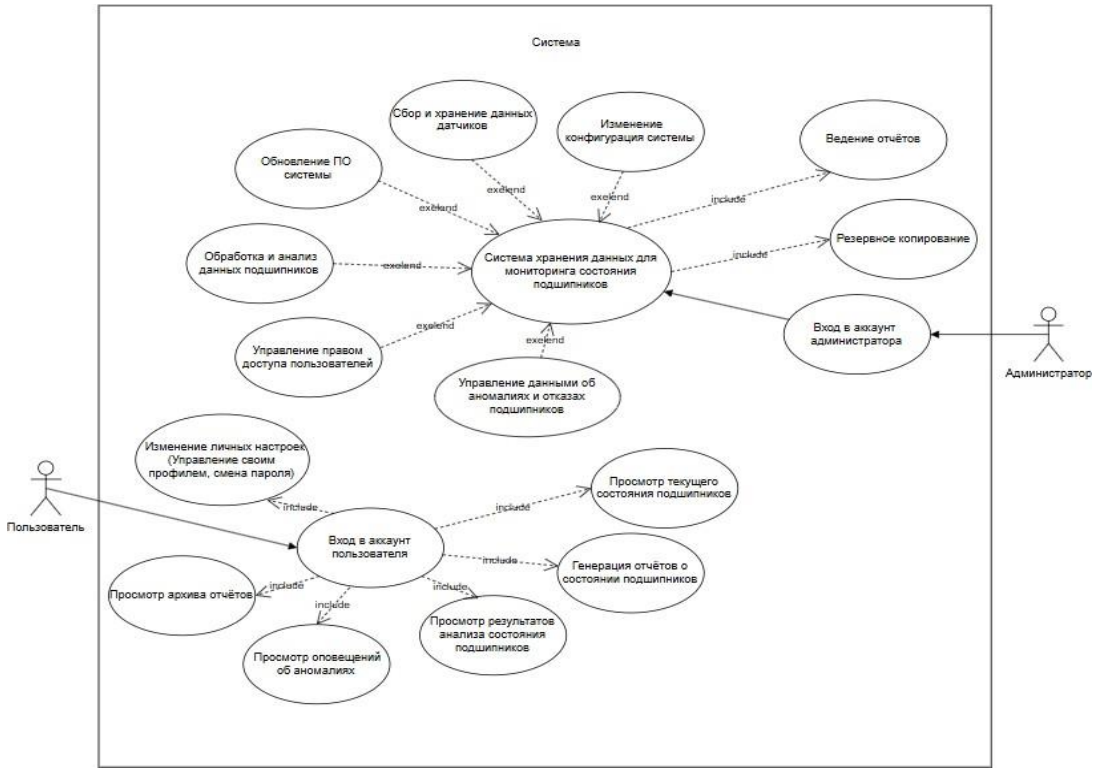

Рисунок 6 – Диаграмма вариантов использования

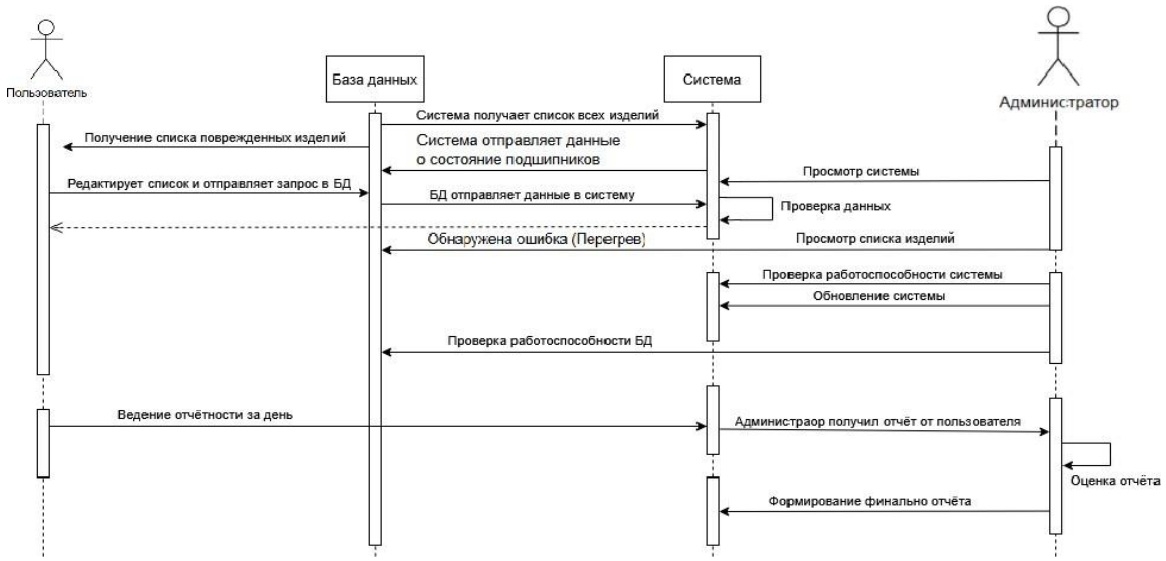

Рисунок 7 – Диаграмма последовательности