Индивидуальный проект по предмету
“Управление IT-проектами”

“Разработка интеллектуальной системы прогнозирования спроса на основе больших данных и машинного обучения”

Преподаватель
Архипова Татьяна Александровна

Выполнил

Патаев Арслан Зольванович ББИ221

Оглавление

Введение

Проект: Разработка интеллектуальной системы прогнозирования спроса на основе больших данных и машинного обучения.

В условиях современной экономики, особенно в сфере e-commerce, точное прогнозирование спроса является важным аспектом успешного управления запасами и оптимизации логистических процессов. Изменчивость спроса, влияющая на эффективность бизнеса, подчеркивает необходимость использования передовых технологий для анализа данных и предсказания потребительского поведения.

Цель проекта: Создание автоматизированной системы, которая позволяет прогнозировать спрос на товары в сфере e-commerce. Это поможет управлять запасами, минимизировать дефицит и избыток товаров, а также оптимизировать маркетинговые кампании.

Описание системы: Система анализирует исторические данные о продажах с помощью временного анализа и методов машинного обучения, в частности модели SARIMA, для создания прогноза на будущие периоды. Особое внимание уделяется учету сезонных колебаний и других факторов, влияющих на спрос.

Основные функциональные возможности:

- Автоматический сбор и предобработка данных каждый день.
- Прогнозирование спроса на 30 дней вперед с учетом сезонных изменений.
- Мониторинг точности модели по метрикам MAE (Mean Absolute Error) и RMSE (Root Mean Square Error).
- Настройка уведомлений на случай ухудшения производительности модели.

Ценность для бизнеса:

- Позволяет эффективно управлять запасами, снижая потери от излишков или нехватки товаров.
- Обеспечивает возможность адаптации маркетинговых стратегий к изменениям спроса.
- Автоматизирует обновление данных и переобучение модели, сокращая ручной труд.

Архитектура системы:

1. **Модель прогнозирования (SARIMA)** — анализирует временной ряд и учитывает сезонные колебания.
2. **Apache Airflow** — управляет автоматизацией процессов: обновление данных, предобработка, переобучение модели и прогнозирование.
3. **Docker** — обеспечивает изоляцию и удобство развертывания.
4. **Prometheus и Grafana** — используются для мониторинга и визуализации производительности модели.
5. **Alertmanager** — отправляет уведомления при выходе метрик за допустимые пределы.

Технологии:

- Язык программирования: Python (для обработки данных и построения модели).
- Библиотеки: pandas, numpy, statsmodels, scikit-learn.
- Инструменты автоматизации: Apache Airflow.
- Среда развертывания: Docker.
- Средства мониторинга: Prometheus и Grafana, поддержка уведомлений через Alertmanager.

I - Исследование реализуемости проекта

Используем метод **6 шагов** для проведения исследования реализуемости проекта по разработке системы прогнозирования спроса на основе больших данных и машинного обучения.

Метод 6 шагов

Шаг 1: Формирование стратегии исследования

Цель: Разработать интеллектуальную систему прогнозирования спроса для e-commerce, которая позволит эффективно управлять запасами, минимизировать дефицит и избыток товаров, оптимизировать маркетинговые кампании.

Контекст: Система будет использовать методы анализа временных рядов и машинного обучения, такие как модель SARIMA, для учета сезонных и других факторов. Система должна автоматизировать сбор, обработку и анализ данных.

План: Определить технические, финансовые и временные аспекты проекта для проверки его жизнеспособности и эффективности, а также выявить возможные риски и ограничения.

Шаг 2: Идентификация и ранжирование источников информации

Приоритетные источники:

1. **Исторические данные продаж** — основной источник для построения прогноза спроса, включающий данные по товарам, сезонности, а также маркетинговым и внешним факторам.
2. **Экспертные мнения** — консультации с аналитиками и специалистами по данным помогут уточнить модель, определить ключевые переменные и учесть отраслевые нюансы.
3. **Технологические источники** — официальные публикации и документация по инструментам (Apache Airflow, Docker, Prometheus, Grafana) и моделям (SARIMA).

Шаг 3: Делегирование полномочий

Распределение задач:

- **Сбор данных** — ответственные аналитики и инженеры данных для подготовки и предобработки данных.

- **Разработка модели** — специалисты по машинному обучению для построения и настройки модели SARIMA.
- **Автоматизация процессов** — DevOps инженеры для настройки Docker, Apache Airflow и систем мониторинга.
- **Мониторинг и уведомления** — команда DevOps также будет отвечать за настройку и поддержку Prometheus, Grafana и Alertmanager.
- **Управление проектом** — проектный менеджер для контроля сроков, бюджетов и координации команды.

Шаг 4: Выполнение исследования

Сбор данных: Получены и обработаны исторические данные о продажах, включая сезонные и внешние факторы. Данные будут дополнительно очищены и подготовлены для построения модели.

Технический анализ:

- **Модель SARIMA** — подтверждено, что она подходит для временного анализа с учетом сезонности, что критично для прогнозирования спроса.
- **Автоматизация с Apache Airflow** — обеспечивает автоматический сбор и обработку данных, регулярное обновление прогноза и интеграцию с системой оповещений.
- **Docker** — подтверждена возможность создания изолированной и легко развертываемой среды для всех компонентов системы.

Финансовый анализ: Оценены затраты на разработку модели, развертывание системы и привлечение специалистов, а также на поддержку и обслуживание.

Временной анализ: Разработка, настройка и тестирование модели и системы займут 7 месяцев с учетом всех этапов и возможных задержек.

Шаг 5: Систематизация выводов

Техническая реализуемость: Выбранные технологии и инструменты подходят для выполнения задач проекта. Требуется квалифицированная команда для работы с большими данными и настройки автоматизации.

Финансовая реализуемость: Затраты на реализацию системы оправданы ожидаемой экономией на управлении запасами и

повышением эффективности. Инвестиции окупятся за счет сокращения убытков от неэффективного управления запасами.

Временная реализуемость: Проект реалистично завершить за 7 месяцев при соблюдении сроков и эффективном управлении рисками.

Шаг 6: Заключение и рекомендации

Рекомендации:

- Приступить к этапу детального планирования и составить план задач с учетом рисков и временных резервов.
- Назначить ответственных лиц для каждого этапа и поддерживать связь с экспертами для консультаций по специфике данных и моделирования.
- Обеспечить резервный бюджет на случай необходимости привлечения дополнительных ресурсов для устранения возможных сложностей.

Заключение: Проект по разработке системы прогнозирования спроса на основе машинного обучения и анализа больших данных признан технически, финансово и временно осуществимым. Система принесет бизнесу ценность, автоматизируя управление запасами и улучшая точность прогнозирования, что будет способствовать снижению издержек и повышению конкурентоспособности.

План реализуемости проекта

1. Анализ текущего состояния и определение требований

- **Срок выполнения**: 1 месяц
- **Задачи**:
  - Сбор и анализ исторических данных продаж и внешних факторов, влияющих на спрос.
  - Определение требований к прогнозированию, включая точность, периодичность и основные метрики (например, MAE, RMSE).
  - Оценка доступных ресурсов (финансовых, временных, человеческих).
- **Ответственные лица**: Аналитики данных, проектный менеджер.
- **Результаты**: Полный набор данных для анализа и технические требования к системе.

2. Формирование проектной команды и распределение ролей

- **Срок выполнения**: 1 неделя
- **Задачи**:
    - Назначение ответственных лиц за каждую задачу в проекте.
    - Подготовка расписания для всех участников проекта.
- **Ответственные лица**: Проектный менеджер.
- **Результаты**: Организационная структура команды, утвержденный график работ.

3. Разработка и тестирование модели

- **Срок выполнения**: 2 месяца
- **Задачи**:
    - Подготовка данных и обучение модели SARIMA для анализа временных рядов с учетом сезонности.
    - Тестирование точности модели и калибровка параметров для достижения требуемой точности.
    - Мониторинг производительности модели и оценка необходимости дальнейшего улучшения.
- **Ответственные лица**: Специалисты по машинному обучению, аналитики данных.
- **Результаты**: Рабочая модель SARIMA, протестированная и готовая к интеграции.

4. Интеграция Apache Airflow для автоматизации

- **Срок выполнения**: 1 месяц
- **Задачи**:
    - Настройка Apache Airflow для автоматизации всех этапов процесса: сбор данных, предобработка, прогнозирование и переобучение модели.
    - Тестирование автоматизации на корректность выполнения процессов и мониторинг за возможными ошибками.
- **Ответственные лица**: DevOps-инженеры.
- **Результаты**: Полностью автоматизированный процесс прогнозирования.

5. Настройка инфраструктуры и развертывание Docker

- **Срок выполнения**: 1 месяц
- **Задачи**:
    - Настройка Docker для развертывания замёрдженной версии системы.

    - Обеспечение изоляции среды, что упрощает переносимость и поддержку системы.
- **Ответственные лица**: DevOps-инженеры.
- **Результаты**: Готовая к развертыванию инфраструктура, изолированная и готовая к тестированию.

6. Настройка мониторинга и уведомлений

- **Срок выполнения**: 1 месяц
- **Задачи**:
    - Внедрение Prometheus для мониторинга ключевых метрик системы (точность модели, производительность).
    - Настройка Grafana для визуализации метрик и создания дашбордов.
    - Настройка Alertmanager для уведомлений о критических изменениях в производительности модели.
- **Ответственные лица**: DevOps-инженеры, аналитики данных.
- **Результаты**: Система мониторинга и уведомлений, отслеживающая состояние системы и оповещающая о критических изменениях.

7. Финальное тестирование и оптимизация системы

- **Срок выполнения**: 1 месяц
- **Задачи**:
    - Полный цикл тестирования системы на производительность и устойчивость.
    - Финальная оптимизация модели и процессов на основе результатов тестирования.
    - Подготовка документации для пользователей и команды поддержки.
- **Ответственные лица**: Проектный менеджер, аналитики данных, DevOps-инженеры.
- **Результаты**: Завершенное тестирование, готовая к использованию система с полной документацией.

8. План пост-поддержки и обновлений

- **Срок выполнения**: Постоянно после запуска
- **Задачи**:
    - Регулярное обновление данных для поддержания актуальности прогнозов.
    - Периодическое переобучение модели и обновление инфраструктуры для повышения точности и производительности.

    - Поддержка и оптимизация системы на основе изменений в спросе и отзывов пользователей.
- **Ответственные лица**: Аналитики данных, DevOps-инженеры, специалисты по поддержке.
- **Результаты**: Устойчивая и обновляемая система, способная адаптироваться к изменениям.

Резюме для руководства

Проект: Разработка интеллектуальной системы прогнозирования спроса на основе больших данных и машинного обучения.

Цель проекта: Создание системы, способной точно прогнозировать спрос на товары в сфере e-commerce, что позволит эффективно управлять запасами, минимизировать дефицит и избыток товаров и оптимизировать маркетинговые кампании.

Основные этапы проекта:

1. **Анализ данных и сбор требований** — 1 месяц.
2. **Формирование команды и планирование** — 1 неделя.
3. **Разработка модели прогнозирования (SARIMA)** — 2 месяца.
4. **Интеграция автоматизации с Apache Airflow** — 1 месяц.
5. **Настройка инфраструктуры (Docker)** — 1 месяц.
6. **Мониторинг и уведомления (Prometheus, Grafana)** — 1 месяц.
7. **Финальное тестирование и оптимизация** — 1 месяц.
8. **Постоянная поддержка и обновления** — по мере необходимости.

Срок реализации: 7 месяцев.

Основные выгоды:

- **Управление запасами**: Снижение расходов за счет уменьшения избыточных запасов и предотвращения дефицита.
- **Оптимизация бизнес-процессов**: Повышение эффективности маркетинговых кампаний, адаптация к сезонным изменениям спроса.
- **Автоматизация**: Система автоматически обновляется и пересчитывает прогнозы, что снижает необходимость ручного вмешательства.

Требуемые ресурсы:

- **Финансовые**: Инвестиции в разработку и развертывание системы, оплату специалистов, а также средства на поддержку и периодическое переобучение модели.
- **Технические**: Необходимы квалифицированные специалисты в области данных и DevOps, а также оборудование для обработки больших объемов данных.

Риски и меры по их снижению:

- **Технические риски**: Использование проверенных инструментов (SARIMA, Docker, Apache Airflow) минимизирует вероятность сбоев.
- **Финансовые риски**: Предусмотрен резервный бюджет, который позволит избежать задержек при возникновении непредвиденных расходов.
- **Временные риски**: Этапы включают временные резервы, что позволит уложиться в срок при возможных задержках.

Заключение: Проект технически, финансово и временно реализуем и имеет высокую ценность для компании. Система поможет оптимизировать управление запасами, улучшит точность прогнозирования и повысит конкурентоспособность.

Описание продукта (Product)

1. Содержание продукта (Описание технологии):
Система прогнозирования спроса построена на основе методов машинного обучения и анализа временных рядов, в частности, модели **SARIMA**. Она предназначена для автоматического прогнозирования спроса с учетом сезонных колебаний. Архитектура системы включает в себя инструменты для автоматизации процессов (Apache Airflow), контейнеризацию (Docker) и мониторинг производительности модели (Prometheus и Grafana).

2. Назначение продукта:
Система разработана для бизнеса в сфере e-commerce и предназначена для точного прогнозирования спроса на товары. Основное назначение — оптимизация управления запасами, предотвращение дефицита и избытка товаров, что способствует снижению операционных расходов и повышению удовлетворенности клиентов.

3. Преимущества IT в обеспечении целей проекта:

- **Автоматизация**: Благодаря Apache Airflow процесс сбора данных, предобработки, прогнозирования и переобучения модели полностью автоматизирован.
- **Гибкость и масштабируемость**: С помощью Docker система легко разворачивается и адаптируется к изменениям в масштабах данных и требований.
- **Непрерывный мониторинг**: Prometheus и Grafana обеспечивают постоянное отслеживание производительности системы и позволяют оперативно реагировать на изменения.
- **Точность прогнозов**: Использование модели SARIMA позволяет учитывать сезонные факторы, что улучшает точность прогнозов и снижает риски ошибок.

4. Сравнительная характеристика аналогичных IT и причины выбора данной системы:

- **IT1 (Рекомендуемая система)**: Оценка 7 — система включает все обязательные и желательные функции, имеет высокие показатели точности и гибкости, полностью соответствует требованиям проекта.
- **IT2**: Оценка 4 — система удовлетворяет большинству обязательных требований, но недостаточна по гибкости и автоматизации.
- **IT3**: Оценка 5 — система соответствует минимальным требованиям, но не обеспечивает достаточного уровня мониторинга и адаптивности.

Рекомендация менеджеру проекта:

На основании сравнительного анализа IT-систем наиболее подходящей для проекта является **IT1**, так как она полностью соответствует всем обязательным и желательным требованиям, обеспечивая высокую точность, гибкость и устойчивость к изменяющимся условиям.

Воздействие на пользователей

- **Цель внедрения**: Минимизировать затраты на внедрение и избежать убытков, которые могут возникнуть в процессе интеграции системы.
- **Группы пользователей**: Система повлияет на отделы, ответственные за управление запасами, маркетинг и аналитические подразделения.

- **Время простоя**: Оценка времени, которое потребуется для настройки и интеграции системы, позволит минимизировать перерывы в работе пользователей.
- **Обучение персонала**: Сотрудникам необходимо будет пройти обучение для использования системы. Это потребует организации учебных классов и приобретения соответствующего оборудования.
- **Изменения в обязанностях**: Система оптимизирует работу сотрудников, автоматизируя ряд процессов и улучшая анализ спроса, что повлияет на их обязанности.
- **Апгрейд технологии**: Планируется регулярное обновление системы, которое также потребует дополнительных учебных мероприятий и изменений в процессе работы.

Финансовые обязательства

1. **Прямые затраты**:
    - **Цена продукта**: Стоимость разработки, тестирования и внедрения системы.
    - **Лицензии**: Стоимость лицензий на инструменты и технологии, использующиеся в системе (Apache Airflow, Docker, Prometheus и др.).
    - **Обучение**: Затраты на обучение сотрудников, создание учебных классов и приобретение необходимого оборудования.
    - **Техническая поддержка**: Обеспечение технической поддержки от специалистов и разработчиков.
2. **Регулярные расходы**:
    - **Операционные затраты**: Регулярное обслуживание, обновления и переобучение модели.
    - **Привлечение сторонних специалистов**: Возможные затраты на внешние ресурсы для проведения обновлений или устранения непредвиденных проблем.
3. **Ожидаемая прибыль от инвестиций**:
    - **ROI** (Return on Investment): Предполагается, что система повысит производительность за счет оптимизации управления запасами и улучшения точности прогнозов спроса.
    - **Период окупаемости**: Система должна выйти на точку безубыточности и стать выгодной в течение первых нескольких лет использования.

Рекомендуемые действия - Запуск

Определить конкретную дату запуска проекта и подготовить команду и инфраструктуру к началу работ. Это действие подразумевает, что проект полностью готов к реализации, имеются необходимые ресурсы и план внедрения. Запуск проекта позволит своевременно начать внедрение системы, минимизировать задержки и обеспечить достижение целей, связанных с оптимизацией прогнозирования спроса и управления запасами.

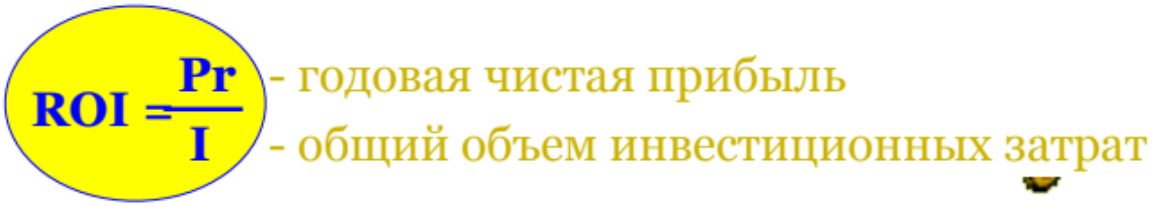
II - Определение экономической эффективности IT-проекта на этапе исследования реализуемости

Цель: Оценить потенциальную экономическую выгоду от реализации проекта по разработке системы прогнозирования спроса. Это поможет обосновать инвестиции, понять сроки окупаемости и спрогнозировать, как проект повлияет на финансовую устойчивость и эффективность компании.

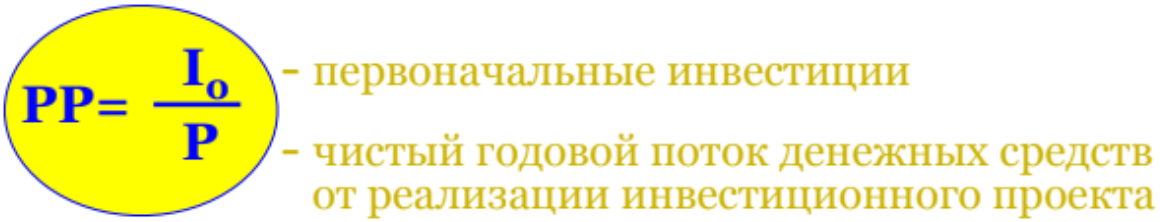
1. Основные показатели для оценки экономической эффективности

Для определения экономической эффективности проекта используются следующие ключевые показатели:

- **ROI (Return on Investment)** — показатель рентабельности инвестиций, который показывает отношение прибыли к вложенным средствам. Формула расчета:



- **PP (Payback Period)** — срок окупаемости проекта, показывающий, за какой период проект начнет приносить прибыль и компенсирует вложенные средства. Формула расчета:



- **NPV (Net Present Value)** — чистая текущая стоимость, которая учитывает временную стоимость денег и показывает ожидаемую прибыль или убыток от проекта с учетом дисконтирования.
- **IRR (Internal Rate of Return)** — внутренняя норма доходности, представляющая собой процентную ставку, при которой NPV проекта равна нулю.

2. Финансовые преимущества от внедрения системы

- **Снижение затрат на управление запасами**: Прогнозирование спроса позволит более точно планировать запасы, что приведет к снижению издержек, связанных с хранением и излишками.
- **Уменьшение потерь от дефицита**: Точные прогнозы помогут избежать ситуаций, когда товар заканчивается в момент высокого спроса, что снижает упущенную прибыль.
- **Оптимизация маркетинговых затрат**: Система поможет лучше понимать сезонные колебания спроса и адаптировать маркетинговые кампании, что повысит эффективность рекламных расходов.
- **Снижение ручного труда**: Автоматизация процессов сбора данных, анализа и отчетности снизит потребность в ручном вмешательстве, что приведет к экономии на трудозатратах.

3. Расчет экономической эффективности

Примерный расчет (используем ориентировочные данные):

- **Инвестиции**: 5 млн рублей
- **Ожидаемый годовой чистый доход от проекта**: 2 млн рублей (в результате оптимизации запасов, повышения точности прогнозов, улучшения маркетинговых решений)

ROI = 2 млн / 5 млн * 100% = 40%

Это означает, что проект будет приносить 40% прибыли на вложенные средства ежегодно.

PP = 5 млн / 2 млн = 2.5

Проект окупится примерно через 2.5 года.

NPV и IRR:

Для расчета NPV и IRR необходимо провести дисконтирование, учитывая ставку дисконтирования и прогнозируемый период получения доходов.

4. Выводы

Экономическая эффективность проекта подтверждается высокими показателями ROI и разумным сроком окупаемости (2.5 года).

Система прогнозирования спроса не только окупит вложения, но и будет способствовать снижению издержек и повышению прибыли компании. Реализация проекта будет выгодной и принесет значительные финансовые преимущества, улучшая управление запасами и оптимизируя маркетинговую активность.

III - Инициация проекта и фиксация его основных параметров

Устав проекта

Определение проекта	
Название проекта:	Разработка интеллектуальной системы прогнозирования спроса на основе больших данных и машинного обучения
Организаторы/ менеджер проекта:	Патаев Арслан Зольванович - менеджер проекта
Срок проекта и основные вехи:	Срок выполнения: 7 месяцев Основные вехи: 1. Анализ данных и сбор требований — 1 месяц 2. Формирование команды и планирование — 1 неделя 3. Разработка модели прогнозирования (SARIMA) — 2 месяца 4. Интеграция автоматизации с Apache Airflow — 1 месяц 5. Настройка инфраструктуры (Docker) — 1 месяц 6. Мониторинг и уведомления (Prometheus, Grafana) — 1 месяц 7. Финальное тестирование и оптимизация — 1 месяц

Содержание и обоснование проекта	
Цель проекта	Разработать и внедрить интеллектуальную систему прогнозирования спроса в сфере e-commerce с использованием модели SARIMA и инструментов автоматизации, которая обеспечит точность прогноза (MAE ≤ 10%, RMSE ≤ 15%) и позволит оптимизировать управление запасами. Система должна быть готова к запуску в течение 7 месяцев.
Задачи проекта	Сбор и анализ исторических данных о продажах, подготовка данных для модели. Разработка и обучение модели SARIMA для прогнозирования спроса с учетом сезонных факторов. Настройка автоматизации с помощью Apache Airflow для обновления данных и предсказаний. Развертывание контейнеризированной среды для масштабируемости и удобства управления.

	Внедрение системы мониторинга производительности и уведомлений для поддержания точности прогноза.
Обоснование инициации проекта (внутренние и/или внешние предпосылки)	Внутренние предпосылки: Необходимость повышения эффективности управления запасами и минимизация затрат на хранение и перемещение товаров. Внешние предпосылки: Конкурентное преимущество на рынке за счет лучшего понимания потребностей клиентов и возможности адаптации к сезонным изменениям спроса.
Ожидаемые результаты проекта	Снижение операционных расходов за счет улучшенного управления запасами. Увеличение доходов за счет оптимизации продаж и более точных прогнозов. Повышение удовлетворенности клиентов благодаря постоянному наличию товаров.
Описание продукта проекта и его структуры (укрупненно)	Прогнозирующая система, включающая модель SARIMA для анализа временных рядов, автоматизацию процессов через Apache Airflow, контейнеризацию с Docker и мониторинг через Prometheus и Grafana.
Факторы риска и успеха	
Ключевые риски проекта	Ошибки в данных или недостаточный объем данных для обучения модели. Возможные задержки в обучении команды и внедрении технологий. Технические трудности при интеграции системы мониторинга и автоматизации.
Ограничения и допущения	Временные ограничения: Завершение в течение 7 месяцев. Бюджетные ограничения: Предусмотренные финансовые лимиты на разработку и обслуживание. Технологические допущения: Использование Python, Apache Airflow, Docker и других инструментов. Географические ограничения отсутствуют, так как проект не зависит от местоположения.
Критерии оценки успешности проекта	Достижение целевых показателей точности прогнозов (например, MAE, RMSE). Окупаемость проекта в установленный срок (2.5 года). Успешное развертывание системы и её стабильная работа. Удовлетворенность пользователей системой, снижение издержек на управление запасами и увеличение доходов.

KPI - Ключевые индикаторы проекта

Для оценки успешности проекта по разработке системы прогнозирования спроса выделены следующие ключевые показатели эффективности (KPI):

1. **Точность прогнозирования**:
   - **MAE (Mean Absolute Error)** — средняя абсолютная ошибка. Целевое значение: не более 10%.
   - **RMSE (Root Mean Square Error)** — среднеквадратическая ошибка. Целевое значение: не более 15%.
   - **MAPE (Mean Absolute Percentage Error)** — средняя абсолютная процентная ошибка. Целевое значение: не более 10%.
2. **Время окупаемости (Payback Period)**:
   - Проект должен окупиться в течение **2.5 лет** с момента внедрения.
3. **Снижение операционных затрат**:
   - Уменьшение расходов на управление запасами на **15%** за первый год работы системы за счет более точного прогнозирования спроса и оптимизации запасов.
4. **Удовлетворенность пользователей**:
   - Оценка удовлетворенности пользователей системы (например, отделов закупок и маркетинга) по итогам опроса. Целевое значение: **удовлетворенность не менее 80%**.
5. **Автоматизация процессов**:
   - Доля автоматизированных процессов в цепочке сбора, предобработки данных и прогнозирования. Целевое значение: **100% автоматизация** всех этапов предсказания спроса с помощью Apache Airflow.
6. **Надежность и устойчивость системы**:
   - **Доступность системы (uptime)** — система должна быть доступна не менее **99%** времени.
   - **Частота ошибок в прогнозировании** — количество сбоев или критических ошибок в работе системы. Целевое значение: не более **5 ошибок в месяц**.
7. **Регулярное обновление модели**:
   - **Частота переобучения модели** — система должна переобучаться на новых данных не реже одного раза в месяц для поддержания точности прогнозов.
8. **Время реакции на уведомления и предупреждения**:
   - **Среднее время реакции** на критические оповещения системы мониторинга (Prometheus, Grafana) не должно превышать **30 минут**.

Обоснование проектной модели

Для разработки системы прогнозирования спроса выбрана **проектная модель на основе машинного обучения** с использованием модели **SARIMA** для анализа временных рядов и учета сезонных факторов. Основные причины выбора данной модели и подхода представлены ниже:

1. Соответствие специфике задачи

Прогнозирование спроса в сфере e-commerce требует учета сезонных колебаний, временных трендов и других факторов, влияющих на продажи. Модель **SARIMA** (Seasonal Autoregressive Integrated Moving Average) хорошо подходит для обработки временных рядов, позволяя учитывать сезонные и временные зависимости. Это делает её эффективным выбором для прогноза спроса, особенно для товаров, подверженных сезонным изменениям спроса.

2. Точность прогнозирования

Модель SARIMA является проверенным инструментом для прогнозирования временных рядов и обеспечивает высокую точность в рамках краткосрочных и среднесрочных прогнозов. Использование модели SARIMA позволяет достичь целевых показателей точности (MAE ≤ 10%, RMSE ≤ 15%), что снижает риски неправильного планирования запасов и увеличивает удовлетворенность потребителей.

3. Автоматизация и поддержка масштабируемости

Для обработки больших объемов данных и автоматизации всех этапов, включая сбор данных, предобработку и прогнозирование, система интегрируется с **Apache Airflow**. Это позволяет:

- Полностью автоматизировать процесс сбора и обработки данных.
- Регулярно обновлять модель и прогнозы без ручного вмешательства.
- Масштабировать систему при увеличении объемов данных или изменении требований.

4. Гибкость и переносимость через Docker

Контейнеризация с использованием **Docker** позволяет создать изолированную и легко разворачиваемую среду для модели и системы автоматизации. Это упрощает процесс тестирования, развертывания и масштабирования системы, обеспечивая её гибкость и возможность работы в различных окружениях.

5. Постоянный мониторинг и надежность системы

Использование **Prometheus** и **Grafana** для мониторинга производительности и точности модели позволяет отслеживать ключевые метрики и получать уведомления в случае отклонений. Это обеспечивает надежность системы и быструю реакцию на возможные проблемы, что снижает риск ошибок в прогнозировании.

6. Экономическая обоснованность

Выбранная проектная модель требует начальных затрат на разработку и внедрение, однако ожидается, что она быстро окупится за счет:

- Сокращения издержек на хранение и управление запасами.
- Минимизации убытков, связанных с дефицитом или излишками товаров.
- Оптимизации маркетинговых расходов, основанных на точных прогнозах спроса.

IV - Управление содержанием проекта

WBS (Work Breakdown Structure) — это иерархическая структура декомпозиции работы, которая используется для разбиения проекта на более мелкие, управляемые задачи. Цель **WBS** — структурировать весь объем работы проекта, чтобы сделать его более понятным и управляемым, облегчить планирование, распределение ресурсов и контроль выполнения. **WBS** часто используется в управлении проектами для обеспечения того, что все элементы проекта учтены и ничего не упущено.

Применение правила 8/80 в текущем WBS

В моей структуре задач я старался:

1. Разбить проект на задачи, которые занимают от 1 дня до 2 недель (8–80 часов), чтобы избежать чрезмерного дробления или излишнего обобщения.
2. Уровень детализации в задачах соответствует масштабу проекта. Например:
    - **1.1. Сбор данных о продажах**: Это задача верхнего уровня, которая делится на подзадачи, каждая из которых (например, определение источников данных) соответствует усилиям в рамках 8–80 часов.
    - **2.1. Разработка модели прогнозирования**: Каждая подзадача, такая как обучение модели или подбор параметров, тоже укладывается в диапазон времени от 1 до 10 рабочих дней.

Work Breakdown Structure (WBS)

Уровень 1: Проект

1. Сбор данных
2. Анализ требований
3. Разработка системы
4. Интеграция и тестирование
5. Внедрение и поддержка

Уровень 2: Основные задачи

1. Сбор данных

1.1. Определение источников данных
1.2. Получение доступа к данным
1.3. Извлечение данных
1.4. Очистка и предобработка данных

2. Анализ требований

2.1. Выявление ключевых пользовательских требований
2.2. Анализ бизнес-процессов
2.3. Документирование требований
2.4. Определение метрик оценки (MAE, RMSE, MAPE)

3. Разработка системы

3.1. Разработка модели прогнозирования (SARIMA)
3.2. Настройка автоматизации процессов (Apache Airflow)
3.3. Развертывание контейнеров (Docker)

3.4. Разработка интерфейса для вывода прогноза
3.5. Подготовка технической документации

4. Интеграция и тестирование

4.1. Интеграция всех компонентов системы
4.2. Настройка системы мониторинга (Prometheus, Grafana)
4.3. Тестирование производительности модели
4.4. Оптимизация параметров прогнозирования
4.5. Финальная проверка системы

5. Внедрение и поддержка

5.1. Подготовка учебных материалов для пользователей
5.2. Обучение сотрудников
5.3. Развертывание системы в рабочей среде
5.4. Организация мониторинга и поддержки
5.5. Регулярное обновление модели и системы

Уровень 3: Детализация задач

1. Сбор данных

1.1. Определение источников данных

- 1.1.1. CRM-системы
- 1.1.2. ERP-системы
- 1.1.3. Базы данных продаж

1.2. Получение доступа к данным

- 1.2.1. Запрос доступа у владельцев данных
- 1.2.2. Настройка прав доступа
- 1.2.3. Обеспечение безопасности данных

1.3. Извлечение данных

- 1.3.1. Выборка необходимых данных
- 1.3.2. Экспорт данных в удобном формате
- 1.3.3. Сохранение данных для последующей обработки

1.4. Очистка и предобработка данных

- 1.4.1. Удаление дубликатов и пропущенных значений
- 1.4.2. Обработка выбросов

- 1.4.3. Нормализация данных

2. Анализ требований

2.1. Выявление ключевых пользовательских требований

- 2.1.1. Проведение интервью с заинтересованными сторонами
- 2.1.2. Сбор пользовательских историй
- 2.1.3. Анализ потребностей бизнеса

2.2. Анализ бизнес-процессов

- 2.2.1. Изучение текущих процессов управления запасами
- 2.2.2. Выявление проблем и узких мест
- 2.2.3. Определение возможностей для оптимизации

2.3. Документирование требований

- 2.3.1. Создание спецификации функциональных требований
- 2.3.2. Определение нефункциональных требований (производительность, безопасность)
- 2.3.3. Утверждение требований с заинтересованными сторонами

2.4. Определение метрик оценки (MAE, RMSE, MAPE)

- 2.4.1. Исследование применимых метрик для оценки модели
- 2.4.2. Выбор оптимальных метрик для проекта
- 2.4.3. Установление целевых значений метрик

3. Разработка системы

3.1. Разработка модели прогнозирования (SARIMA)

- 3.1.1. Подбор гиперпараметров модели
- 3.1.2. Обучение модели на подготовленных данных
- 3.1.3. Тестирование модели на точность прогнозов

3.2. Настройка автоматизации процессов (Apache Airflow)

- 3.2.1. Разработка DAG (Directed Acyclic Graph) для процессов
- 3.2.2. Интеграция задач сбора, обработки данных и обучения модели
- 3.2.3. Настройка расписания выполнения задач

3.3. Развертывание контейнеров (Docker)

- 3.3.1. Создание Dockerfile для каждого компонента системы
- 3.3.2. Сборка и тестирование контейнеров
- 3.3.3. Настройка Docker Compose для оркестрации контейнеров

3.4. Разработка интерфейса для вывода прогноза

- 3.4.1. Проектирование пользовательского интерфейса
- 3.4.2. Реализация фронтенда (например, с использованием React)
- 3.4.3. Разработка API для взаимодействия с моделью

3.5. Подготовка технической документации

- 3.5.1. Документирование архитектуры системы
- 3.5.2. Создание инструкций по развертыванию
- 3.5.3. Описание API и интеграционных точек

4. Интеграция и тестирование

4.1. Интеграция всех компонентов системы

- 4.1.1. Соединение модели с интерфейсом
- 4.1.2. Интеграция базы данных и системы автоматизации
- 4.1.3. Тестирование взаимодействия компонентов

4.2. Настройка системы мониторинга (Prometheus, Grafana)

- 4.2.1. Установка и настройка Prometheus для сбора метрик
- 4.2.2. Настройка Grafana для визуализации метрик
- 4.2.3. Создание дашбордов для отслеживания состояния системы

4.3. Тестирование производительности модели

- 4.3.1. Запуск тестов на тестовом окружении
- 4.3.2. Анализ метрик производительности (MAE, RMSE, время расчета прогноза)
- 4.3.3. Идентификация и устранение узких мест

4.4. Оптимизация параметров прогнозирования

- 4.4.1. Тонкая настройка гиперпараметров модели
- 4.4.2. Проведение дополнительных экспериментов с данными
- 4.4.3. Повторное обучение модели

4.5. Финальная проверка системы

- 4.5.1. Проведение приемочных тестов

- 4.5.2. Утверждение системы с заинтересованными сторонами
- 4.5.3. Подготовка к развертыванию в рабочей среде

5. Внедрение и поддержка

5.1. Подготовка учебных материалов для пользователей

- 5.1.1. Создание руководств пользователя
- 5.1.2. Разработка видеоуроков и презентаций
- 5.1.3. Подготовка FAQ и базы знаний

5.2. Обучение сотрудников

- 5.2.1. Проведение тренингов и семинаров
- 5.2.2. Ответы на вопросы и поддержка в процессе обучения
- 5.2.3. Оценка уровня освоения системы сотрудниками

5.3. Развертывание системы в рабочей среде

- 5.3.1. Подготовка инфраструктуры
- 5.3.2. Развертывание контейнеров на сервере
- 5.3.3. Тестирование системы в боевых условиях

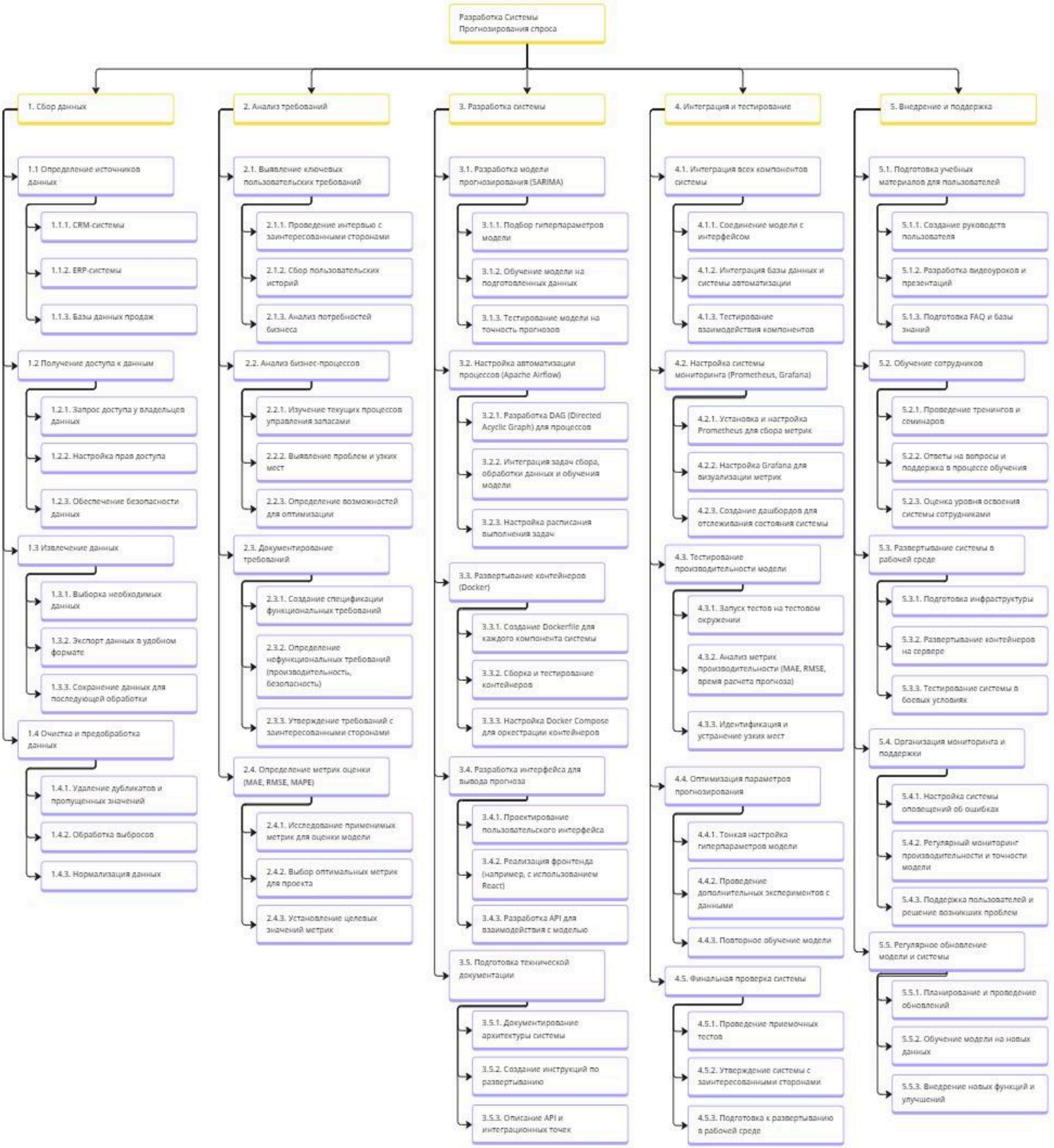
5.4. Организация мониторинга и поддержки

- 5.4.1. Настройка системы оповещений об ошибках
- 5.4.2. Регулярный мониторинг производительности и точности модели
- 5.4.3. Поддержка пользователей и решение возникших проблем

5.5. Регулярное обновление модели и системы

- 5.5.1. Планирование и проведение обновлений
- 5.5.2. Обучение модели на новых данных
- 5.5.3. Внедрение новых функций и улучшений

Визуализация WBS



User stories для проекта

Пользовательские истории представляют собой простые и понятные описания требований к системе с точки зрения её пользователей. Они помогают определить функциональность, которая принесет ценность конечным пользователям и заинтересованным сторонам.

Формат пользовательской истории:

Как [роль пользователя], я хочу [цель/действие], чтобы [ожидаемая выгода]

User Story Mapping — это техника, которая помогает визуализировать пользовательские истории, разбивая их на этапы пользовательского пути (сценарии) и группируя по приоритетам. Вот как мы можем построить **User Story Mapping** для проекта разработки интеллектуальной системы прогнозирования спроса.

User stories для каждого этапа:

- **Сбор данных:**
  - Как аналитик, я хочу собирать данные из CRM и ERP-систем, чтобы получить полную информацию о продажах.
  - Как оператор базы данных, я хочу видеть уведомления об успешной загрузке данных, чтобы убедиться, что данные обновлены.
  - Как сотрудник склада, я хочу вручную добавлять данные, если автоматическая загрузка недоступна.
- **Анализ данных:**
  - Как аналитик данных, я хочу очищать данные от ошибок, чтобы прогнозы были точными.
  - Как руководитель проекта, я хочу видеть прогресс подготовки данных в системе.
  - Как владелец продукта, я хочу видеть отчеты о качестве исходных данных для оценки необходимости их улучшения.
- **Прогнозирование спроса:**
  - Как специалист по данным, я хочу видеть прогнозы спроса на основе трендов и сезонности, чтобы планировать объемы поставок.
  - Как маркетолог, я хочу видеть прогнозы спроса по категориям товаров для планирования акций.
  - Как менеджер по продажам, я хочу сравнивать фактические данные с прогнозами, чтобы оценить точность системы.

- **Использование прогнозов:**
  - Как менеджер по закупкам, я хочу получать рекомендации по объемам поставок, чтобы избегать излишков или дефицита.
  - Как сотрудник склада, я хочу видеть прогнозы спроса на мобильном устройстве, чтобы удобнее работать.
  - Как покупатель, я хочу получать уведомления о наличии популярных товаров в магазине.
- **Обучение и поддержка:**
  - Как пользователь системы, я хочу пройти обучение работе с системой, чтобы эффективно её использовать.
  - Как администратор, я хочу получать уведомления о сбоях в системе для быстрого устранения проблем.
  - Как пользователь системы, я хочу иметь доступ к базе знаний для решения типичных вопросов самостоятельно.

Backlog проекта

Эпик 1: Сбор данных

1. **Название задачи:** Определение источников данных
   - **Приоритет:** Высокий
   - **Описание:** Определить ключевые источники данных, включая CRM, ERP-системы и базы данных продаж.
   - **Критерии приемки:**
     - Все источники данных согласованы и утверждены.
     - Полный список источников документирован.
   - **Оценка сложности:** 3 Story Points
   - **Связанные задачи:** Получение доступа к данным.
2. **Название задачи:** Получение доступа к данным
   - **Приоритет:** Высокий
   - **Описание:** Настроить доступ к CRM и ERP-системам, обеспечив безопасность данных.
   - **Критерии приемки:**
     - Доступ ко всем источникам настроен.
     - Все протоколы безопасности соблюдены.
   - **Оценка сложности:** 5 Story Points
   - **Связанные задачи:** Извлечение данных.
3. **Название задачи:** Извлечение данных
   - **Приоритет:** Высокий
   - **Описание:** Выгрузить данные из всех источников в удобном формате.
   - **Критерии приемки:**

        - Данные успешно выгружены из всех источников.
        - Данные структурированы в заданном формате.
    - **Оценка сложности:** 8 Story Points
    - **Связанные задачи:** Очистка данных.
4. **Название задачи:** Очистка данных от ошибок
    - **Приоритет:** Высокий
    - **Описание:** Удалить дубликаты, пропущенные значения и выбросы.
    - **Критерии приемки:**
        - Дубликаты и выбросы удалены.
        - Пропущенные значения обработаны.
    - **Оценка сложности:** 13 Story Points
    - **Связанные задачи:** Анализ требований.
5. **Название задачи:** Автоматизация сбора данных
    - **Приоритет:** Средний
    - **Описание:** Настроить автоматический процесс сбора данных через API.
    - **Критерии приемки:**
        - Автоматический процесс работает без ошибок.
        - Данные обновляются регулярно.
    - **Оценка сложности:** 13 Story Points
    - **Связанные задачи:** Извлечение данных.

Эпик 2: Анализ требований

1. **Название задачи:** Выявление ключевых требований пользователей
    - **Приоритет:** Средний
    - **Описание:** Провести интервью и собрать ключевые пользовательские требования.
    - **Критерии приемки:**
        - Проведено минимум 5 интервью.
        - Требования согласованы и утверждены.
    - **Оценка сложности:** 8 Story Points
    - **Связанные задачи:** Документирование требований.
2. **Название задачи:** Анализ бизнес-процессов
    - **Приоритет:** Средний
    - **Описание:** Изучить текущие процессы и определить узкие места.
    - **Критерии приемки:**
        - Все проблемы в процессах идентифицированы.

        - Оптимизации процессов описаны.
    - **Оценка сложности:** 8 Story Points
    - **Связанные задачи:** Документирование требований.
3. **Название задачи:** Документирование требований
    - **Приоритет:** Средний
    - **Описание:** Создать спецификацию функциональных и нефункциональных требований.
    - **Критерии приемки:**
        - Все требования документированы.
        - Спецификация утверждена заинтересованными сторонами.
    - **Оценка сложности:** 5 Story Points
    - **Связанные задачи:** Разработка системы.
4. **Название задачи:** Определение метрик для оценки точности (MAE, RMSE, MAPE)
    - **Приоритет:** Средний
    - **Описание:** Настроить расчеты метрик точности для оценки модели.
    - **Критерии приемки:**
        - Метрики рассчитываются корректно.
        - Метрики соответствуют заданным требованиям.
    - **Оценка сложности:** 5 Story Points
    - **Связанные задачи:** Тестирование производительности.

Эпик 3: Разработка системы

1. **Название задачи:** Разработка модели прогнозирования (SARIMA)
    - **Приоритет:** Высокий
    - **Описание:** Построить и обучить модель на подготовленных данных.
    - **Критерии приемки:**
        - Модель обучена и протестирована.
        - RMSE ≤ 5%.
    - **Оценка сложности:** 13 Story Points
    - **Связанные задачи:** Интеграция модели с интерфейсом.
2. **Название задачи:** Настройка автоматизации процессов
    - **Приоритет:** Средний
    - **Описание:** Автоматизировать процессы сбора и обработки данных с использованием Apache Airflow.
    - **Критерии приемки:**
        - Процессы работают без ошибок.

        - Автоматизация покрывает ключевые задачи.
    - **Оценка сложности:** 13 Story Points
    - **Связанные задачи:** Разработка модели.
3. **Название задачи:** Разработка интерфейса для отображения прогнозов
    - **Приоритет:** Высокий
    - **Описание:** Реализовать графический интерфейс для пользователей.
    - **Критерии приемки:**
        - Интерфейс работает без ошибок.
        - Данные экспортируются в PDF.
    - **Оценка сложности:** 8 Story Points
    - **Связанные задачи:** Интеграция компонентов системы.

Эпик 4: Интеграция и тестирование

1. **Название задачи:** Интеграция компонентов системы
    - **Приоритет:** Высокий
    - **Описание:** Объединить модель, интерфейс и автоматизацию в единую систему.
    - **Критерии приемки:**
        - Все компоненты интегрированы и работают без ошибок.
        - Проведено тестирование в тестовой среде.
    - **Оценка сложности:** 8 Story Points
    - **Связанные задачи:** Тестирование производительности.
2. **Название задачи:** Тестирование производительности
    - **Приоритет:** Средний
    - **Описание:** Проверить производительность модели на тестовом окружении.
    - **Критерии приемки:**
        - Метрики производительности соответствуют требованиям.
        - Все ошибки устранены.
    - **Оценка сложности:** 13 Story Points
    - **Связанные задачи:** Финальная проверка.

Эпик 5: Внедрение и поддержка

1. **Название задачи:** Обучение пользователей
    - **Приоритет:** Средний

- **Описание:** Провести обучение пользователей работе с системой.
- **Критерии приемки:**
  - Обучено не менее 10 пользователей.
  - Все пользователи освоили основные функции.
- **Оценка сложности:** 3 Story Points
- **Связанные задачи:** Создание учебных материалов.

2. **Название задачи:** Развертывание системы
- **Приоритет:** Высокий
- **Описание:** Развернуть систему в рабочей среде и обеспечить её стабильную работу.
- **Критерии приемки:**
  - Система успешно развернута и работает в боевых условиях.
  - Проведены проверочные тесты.
- **Оценка сложности:** 8 Story Points
- **Связанные задачи:** Поддержка и обновление.

Структура бэклога

1. **Эпики (Epics)** - Главные этапы работы над проектом, разбивающие его на крупные части

2. **Пользовательские истории (User Stories)** - Описывают функциональность системы с точки зрения конечного пользователя.

3. **Задачи (Tasks)** - Каждая пользовательская история разбита на технические задачи, описывающие, как она будет реализована.

4. Элементы каждой задачи

- **Название**: Описание цели задачи (например, "Очистка данных от ошибок").
- **Приоритет**: Высокий, средний, низкий (например, задачи сбора данных — высокий приоритет).
- **Описание**: Что нужно сделать и какие требования учесть.
- **Критерии приемки**: Что должно быть выполнено, чтобы задача считалась завершенной.
- **Оценка сложности**: В Story Points или часах (например, 5 SP).
- **Связанные задачи**: Линки на зависимости или связанные истории.

V - Управление командой проекта

Цели управления командой:

- Создать эффективную и продуктивную проектную команду.
- Обеспечить четкое распределение ролей и обязанностей.
- Установить прозрачную систему взаимодействия внутри команды.
- Обеспечить своевременную коммуникацию и устранение конфликтов.
- Поддерживать мотивацию и вовлеченность членов команды.

OBS - Организационная структура команды (Organizational Breakdown Structure)

- **OBS** определяет, как команда структурирована и как распределяются роли и обязанности.

0. Руководитель проекта - Отвечает за общее управление проектом, контроль сроков, ресурсов и взаимодействие с командой.

1. Сбор данных

- **1.1. Определение источников данных**
  - **Ответственный**: Бизнес-аналитик.
  - **Участвующие**: Data Scientist, DevOps-инженер.
- **1.2. Получение доступа к данным**
  - **Ответственный**: DevOps-инженер.
  - **Участвующие**: Бизнес-аналитик.
- **1.3. Извлечение данных**
  - **Ответственный**: Data Scientist.
  - **Участвующие**: DevOps-инженер.
- **1.4. Очистка и предобработка данных**
  - **Ответственный**: Data Scientist.
  - **Участвующие**: Бизнес-аналитик.

2. Анализ требований

- **2.1. Выявление ключевых пользовательских требований**
  - **Ответственный**: Бизнес-аналитик.
  - **Участвующие**: Руководитель проекта, Data Scientist.
- **2.2. Анализ бизнес-процессов**

- **Ответственный**: Бизнес-аналитик.
- **Участвующие**: Руководитель проекта.
- **2.3. Документирование требований**
- **Ответственный**: Бизнес-аналитик.
- **Участвующие**: Технический писатель.
- **2.4. Определение метрик оценки (MAE, RMSE, MAPE)**
- **Ответственный**: Data Scientist.
- **Участвующие**: Бизнес-аналитик.

3. Разработка системы

- **3.1. Разработка модели прогнозирования (SARIMA)**
- **Ответственный**: Data Scientist.
- **Участвующие**: DevOps-инженер.
- **3.2. Настройка автоматизации процессов (Apache Airflow)**
- **Ответственный**: DevOps-инженер.
- **Участвующие**: Data Scientist.
- **3.3. Развертывание контейнеров (Docker)**
- **Ответственный**: DevOps-инженер.
- **Участвующие**: Руководитель проекта.
- **3.4. Разработка интерфейса для вывода прогноза**
- **Ответственный**: Frontend-разработчик.
- **Участвующие**: Data Scientist, Технический писатель.
- **3.5. Подготовка технической документации**
- **Ответственный**: Технический писатель.
- **Участвующие**: Руководитель проекта.

4. Интеграция и тестирование

- **4.1. Интеграция всех компонентов системы**
- **Ответственный**: DevOps-инженер.
- **Участвующие**: Data Scientist, Frontend-разработчик.
- **4.2. Настройка системы мониторинга (Prometheus, Grafana)**
- **Ответственный**: DevOps-инженер.
- **Участвующие**: Руководитель проекта.
- **4.3. Тестирование производительности модели**
- **Ответственный**: Тестировщик.
- **Участвующие**: Data Scientist.
- **4.4. Оптимизация параметров прогнозирования**
- **Ответственный**: Data Scientist.
- **Участвующие**: Тестировщик.
- **4.5. Финальная проверка системы**
- **Ответственный**: Тестировщик.

- **Участвующие**: Руководитель проекта.

5. Внедрение и поддержка

- **5.1. Подготовка учебных материалов для пользователей**
- **Ответственный**: Технический писатель.
- **Участвующие**: Бизнес-аналитик.
- **5.2. Обучение сотрудников**
- **Ответственный**: Бизнес-аналитик.
- **Участвующие**: Руководитель проекта.
- **5.3. Развертывание системы в рабочей среде**
- **Ответственный**: DevOps-инженер.
- **Участвующие**: Руководитель проекта.
- **5.4. Организация мониторинга и поддержки**
- **Ответственный**: DevOps-инженер.
- **Участвующие**: Тестировщик.
- **5.5. Регулярное обновление модели и системы**
- **Ответственный**: Data Scientist.
- **Участвующие**: DevOps-инженер.

Связь WBS и OBS

- **WBS** задает, что нужно сделать: каждая задача из WBS назначается конкретным ответственным в OBS.
- **OBS** показывает, кто отвечает за выполнение задач, что упрощает контроль за выполнением работы и улучшает коммуникацию.

Матрица RACI

RACI — это инструмент для четкого определения ответственности участников проекта.

- **Responsible (R)**: Выполняет задачу.
- **Accountable (A)**: Несёт ответственность за результат.
- **Consulted (C)**: Предоставляет консультации.
- **Informed (I)**: Получает информацию о ходе выполнения задачи.

Задачи	Руководитель проекта	Бизнес-аналитик	Data Scientist	DevOps-инженер	Frontend-разработчик	Тестировщик	Технический писатель
Сбор данных	A	R	C	C	I	I	I

Определение источников данных	A	R	C	I	I	I	I
Получение доступа к данным	A	C	I	R	I	I	I
Извлечение данных	A	C	R	C	I	I	I
Очистка и предобработка данных	A	C	R	I	I	I	I
Анализ требований	A	R	C	I	I	I	I
Выявление требований	A	R	C	I	I	I	I
Документирование требований	A	R	C	I	I	I	R
Определение метрик оценки	A	C	R	I	I	I	I
Разработка системы	A	C	R	C	R	I	R
Разработка модели прогнозирования (SARIMA)	A	C	R	I	I	I	I
Настройка автоматизации процессов	A	C	C	R	I	I	I
Развертывание контейнеров (Docker)	A	C	I	R	I	I	I
Разработка интерфейса для вывода прогноза	A	I	C	I	R	I	I
Подготовка технической документации	A	I	C	I	I	I	R
Интеграция и тестирование	A	I	R	R	I	R	I
Интеграция всех компонентов системы	A	I	R	R	I	I	I

Настройка мониторинга	A	I	I	R	I	I	I
Тестирование производительности модели	A	I	R	I	I	R	I
Финальная проверка системы	A	I	I	I	I	R	I
Внедрение и поддержка	A	R	R	R	I	I	R
Подготовка учебных материалов	A	C	I	I	I	I	R
Обучение сотрудников	A	R	I	I	I	I	I
Развертывание системы в рабочей среде	A	I	C	R	I	I	I
Постоянная поддержка и обновление модели	A	I	R	R	I	I	I

Инструменты управления командой

Мы выбрали инструменты, которые помогают эффективно организовать взаимодействие внутри команды и управлять задачами:

1. **Инструменты для коммуникации**:
   - **Slack**, **Microsoft Teams** — для оперативного общения и обсуждения задач.
   - **Zoom**, **Google Meet** — для проведения встреч и видеоконференций.
2. **Системы управления задачами**:
   - **Jira**, **Trello**, **Asana** — для планирования и отслеживания выполнения задач.
3. **Документация и совместная работа**:
   - **Confluence**, **Google Docs** — для хранения и совместного редактирования документации.
   - **Miro**, **Figma** — для визуализации идей и проведения мозговых штурмов.

4. **Инструменты для отчетности и анализа**:
   - **Power BI**, **Google Data Studio** — для создания отчетов и визуализации прогресса.
   - **Grafana** — для мониторинга производительности проекта в реальном времени.

Ключевые аспекты управления

Эффективное управление командой основывается на нескольких аспектах:

1. **Мотивация команды**:
   - Регулярное признание заслуг и достижений.
   - Поддержка профессионального развития (тренинги, обучение).
   - Гибкий график работы и поддержка work-life balance.
2. **Управление конфликтами**:
   - Своевременное выявление конфликтов.
   - Прозрачные процессы взаимодействия и распределения обязанностей.
   - Личное вмешательство руководителя проекта в сложные ситуации.
3. **Коммуникация**:
   - Ежедневные/еженедельные собрания для обсуждения прогресса.
   - Использование прозрачных каналов связи (Slack, Teams).
4. **Мониторинг прогресса**:
   - Регулярное обновление статусов задач в Jira/Trello.
   - Использование KPI для измерения эффективности работы команды.
5. **Распределение ролей и обязанностей**:
   - Создание и использование **RACI-матрицы**.
   - Четкое распределение ответственности между членами команды.

KPI для управления командой

Мы определили основные показатели эффективности (KPI), которые позволят оценить, насколько успешно работает команда:

1. **Скорость выполнения задач**:
   - Процент выполненных задач в срок.
   - Среднее время завершения задачи.
2. **Качество работы**:
   - Количество доработок после тестирования.
   - Уровень удовлетворенности заинтересованных сторон.
3. **Эффективность взаимодействия**:

- Количество конфликтов и их время разрешения.
- Частота коммуникаций внутри команды.

4. **Удовлетворенность команды**:
- Регулярные опросы о комфорте работы в проекте.
- Уровень выгорания сотрудников.

5. **Обучение и развитие**:
- Количество завершенных тренингов.
- Рост компетенций членов команды.

VI Календарное и ресурсное планирование проекта

Описание сетевого графика

1. Сбор данных

1. **1.1. Определение источников данных**
   - **Описание**: Установление списка систем и баз данных, необходимых для сбора данных.
   - **Story Points**: 3
   - **Тип зависимости**: — (начальная задача).
2. **1.2. Получение доступа к данным**
   - **Описание**: Настройка прав доступа и обеспечение безопасности данных.
   - **Story Points**: 5
   - **Тип зависимости**: — (начальная задача).
3. **1.3. Извлечение данных**
   - **Описание**: Выгрузка необходимых данных из всех источников.
   - **Story Points**: 8
   - **Тип зависимости**: FS (зависит от задач 1.1 и 1.2).
4. **1.4. Очистка и предобработка данных**
   - **Описание**: Удаление ошибок, пропусков, дубликатов и нормализация данных.
   - **Story Points**: 13
   - **Тип зависимости**: FS (зависит от задачи 1.3).

2. Анализ требований

5. **2.1. Выявление ключевых пользовательских требований**
   - **Описание**: Сбор пользовательских историй через интервью и анализ бизнес-процессов.
   - **Story Points**: 8
   - **Тип зависимости**: FS (зависит от задачи 1.4).
6. **2.2. Анализ бизнес-процессов**
   - **Описание**: Выявление текущих процессов, проблем и возможностей для оптимизации.
   - **Story Points**: 8
   - **Тип зависимости**: SS (может выполняться параллельно с задачей 2.1).
7. **2.3. Документирование требований**

- **Описание**: Создание спецификации функциональных и нефункциональных требований.
- **Story Points**: 5
- **Тип зависимости**: FS (зависит от задач 2.1 и 2.2).

8. **2.4. Определение метрик оценки (MAE, RMSE, MAPE)**
- **Описание**: Настройка и выбор метрик для оценки точности модели.
- **Story Points**: 5
- **Тип зависимости**: FS (зависит от задачи 2.1).

3. Разработка системы

9. **3.1. Разработка модели прогнозирования (SARIMA)**
- **Описание**: Построение модели прогнозирования и настройка её параметров.
- **Story Points**: 13
- **Тип зависимости**: FS (зависит от задач 2.3 и 2.4).

10. **3.2. Настройка автоматизации процессов (Apache Airflow)**
- **Описание**: Автоматизация сбора, обработки данных и запуска модели.
- **Story Points**: 13
- **Тип зависимости**: FS (зависит от задачи 3.1).

11. **3.3. Развертывание контейнеров (Docker)**
- **Описание**: Создание среды для стабильного функционирования компонентов системы.
- **Story Points**: 8
- **Тип зависимости**: FS (зависит от задачи 3.2).

12. **3.4. Разработка интерфейса для вывода прогноза**
- **Описание**: Реализация пользовательского интерфейса для вывода прогнозов.
- **Story Points**: 8
- **Тип зависимости**: FS (зависит от задачи 3.1).

13. **3.5. Подготовка технической документации**
- **Описание**: Документирование архитектуры системы и инструкций по развертыванию.
- **Story Points**: 8
- **Тип зависимости**: FS (зависит от задачи 3.3).

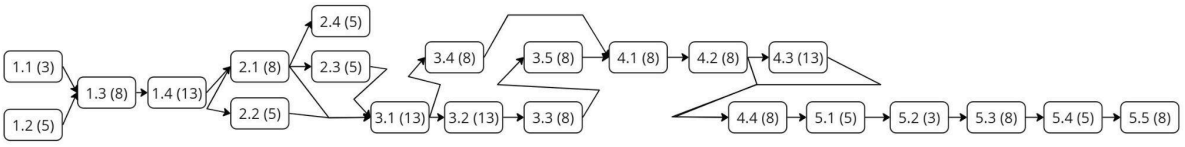
4. Интеграция и тестирование

14. **4.1. Интеграция всех компонентов системы**
    - **Описание**: Объединение модели, интерфейса и автоматизации в одну систему.
    - **Story Points**: 8
    - **Тип зависимости**: FS (зависит от задач 3.4 и 3.5).
15. **4.2. Настройка системы мониторинга (Prometheus, Grafana)**
    - **Описание**: Настройка мониторинга производительности системы.
    - **Story Points**: 8
    - **Тип зависимости**: FS (зависит от задачи 4.1).
16. **4.3. Тестирование производительности модели**
    - **Описание**: Проверка производительности и точности модели на тестовом окружении.
    - **Story Points**: 13
    - **Тип зависимости**: FS (зависит от задачи 4.1).
17. **4.4. Финальная проверка системы**
    - **Описание**: Полное тестирование системы перед внедрением.
    - **Story Points**: 8
    - **Тип зависимости**: FS (зависит от задач 4.2 и 4.3).

5. Внедрение и поддержка

18. **5.1. Подготовка учебных материалов для пользователей**
    - **Описание**: Создание руководств пользователя, FAQ и презентаций.
    - **Story Points**: 5
    - **Тип зависимости**: FS (зависит от задачи 4.4).
19. **5.2. Обучение сотрудников**
    - **Описание**: Проведение тренингов и оценка освоения системы.
    - **Story Points**: 3
    - **Тип зависимости**: FS (зависит от задачи 5.1).
20. **5.3. Развертывание системы в рабочей среде**
    - **Описание**: Внедрение системы и тестирование её работоспособности.
    - **Story Points**: 8
    - **Тип зависимости**: FS (зависит от задачи 5.2).
21. **5.4. Организация мониторинга и поддержки**
    - **Описание**: Настройка мониторинга, решение проблем и поддержка пользователей.
    - **Story Points**: 5
    - **Тип зависимости**: FS (зависит от задачи 5.3).
22. **5.5. Регулярное обновление модели и системы**

- **Описание**: Обновление модели на основе новых данных и добавление улучшений.
- **Story Points**: 8
- **Тип зависимости**: FS (зависит от задачи 5.4).



Итоговая информация

- **Общий вес проекта (Story Points): 152**.
- **Критический путь: 1.1 → 1.3 → 1.4 → 2.1 → 3.1 → 3.2 → 3.3 → 4.1 → 4.3 → 4.4 → 5.3 → 5.5**
- **Длительность критического пути: 86 Story Points**

Типы зависимостей

- **FS (Finish-to-Start):** Одна задача начинается после завершения другой.
- **SS (Start-to-Start):** Задачи выполняются параллельно.

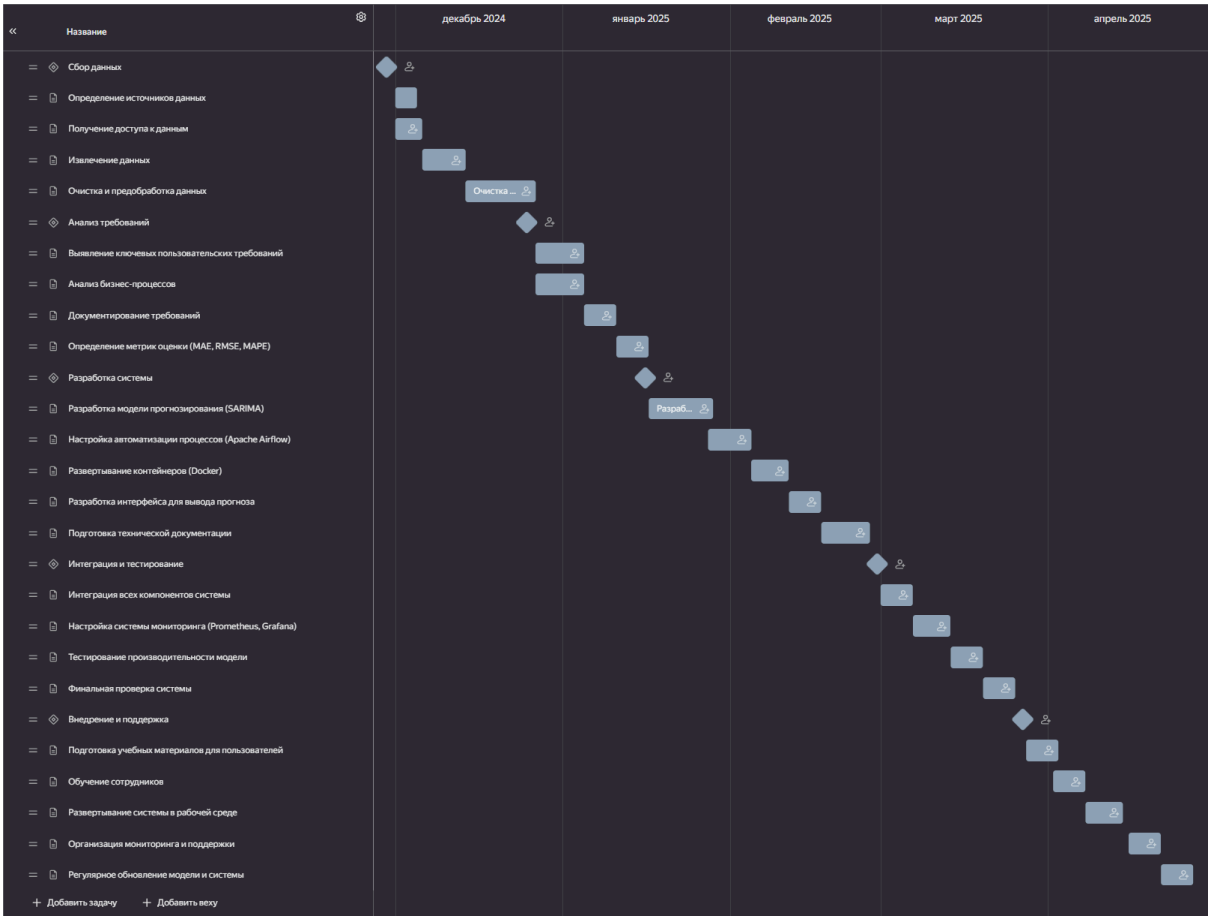
Расписание проекта

1. **Рабочая неделя:** 5 дней (40 часов).
2. **Story Points:** Переводятся в дни на основе сложности задач (1 SP = 1 день работы).
3. **Начальная дата:** Условно принимаем **01 декабря 2024 года**.
4. **Ресурсы:** Задачи распределяются между членами команды с учетом их загрузки.

Длительность

- **Общая длительность:** 144 рабочих дней.
- **Начало проекта:** 01 декабря 2024 года.
- **Завершение проекта:** 25 апреля 2025 года.

Диаграмма Ганта



VII - Управление стоимостью проекта

Вид оценки

Для составления бюджета и сметы использовалась **сметная (аналитическая) оценка**.

Особенности вида оценки:

- **Подробность:** Оценка основана на анализе каждого этапа проекта и задач из **WBS** и **бэклога**.
- **Точность:** Каждая статья затрат детализирована по категориям: заработная плата, оборудование, лицензии, административные расходы и резервы.
- **Прогнозируемость:** Дает четкое представление о полной стоимости проекта, включая прямые, непрямые, и резервные затраты.

Метод оценки

Использовался **метод параметрического расчета**, дополненный элементами **аналога**.

Описание метода:

1. **Параметрический расчет:**
   - Базируется на средней стоимости Story Point (10,000 руб.) и трудозатратах в часах/месяцах.
   - Применяется для оценки задач, основанных на масштабе проекта (например, объем работы команды).
2. **Аналоговый расчет:**
   - Учитывает данные по предыдущим схожим проектам.
   - Используется для оценки амортизации, резервов, и непрямых затрат.

Пример применения:

- Зарплаты сотрудников рассчитаны на основе средней ставки на рынке.
- Затраты на ПО (лицензии) сопоставлены с аналогами в других IT-проектах.

Смета

1. Прямые затраты

Прямые затраты включают заработную плату сотрудников, расходы на оборудование и программное обеспечение, которые связаны с выполнением конкретных задач проекта.

Затраты на персонал:

Роль	Ставка (руб./мес.)	Месяцы работы	Колич ство специалистов	Итого (руб.)
Проектный менеджер	150,000	7	1	1,050,000
Data Scientist	200,000	7	2	2,800,000

DevOps-инженер	180,000	7	1	1,260,000
Бизнес-аналитик	140,000	7	1	980,000
Технический писатель	100,000	3	1	300,000
Frontend-разработчик	160,000	4	1	640,000
Тестировщик	120,000	3	1	360,000

Итого на персонал: 7,390,000 руб.

Затраты на оборудование и программное обеспечение:

Категория	Количество единиц	Стоимость за единицу (руб.)	Итого (руб.)
Серверное оборудование	1	500,000	500,000
Лицензии и ПО	-		
- Apache Airflow	1	150,000	150,000
- Prometheus и Grafana	1	100,000	100,000
- Docker	1	120,000	120,000
Прочие библиотеки и инструменты	1	200,000	200,000

Итого на оборудование и ПО: 1,070,000 руб.

Общая сумма прямых затрат: 8,460,000 руб.

2. Непрямые затраты

Непрямые затраты связаны с поддержкой административных процессов и общих условий работы проекта.

Категория	Описание	Итого (руб.)
Аренда офиса	Помещение для команды	150,000
Коммунальные платежи	Электричество, интернет и пр.	50,000

ИТ-поддержка	Техническая поддержка инфраструктуры	100,000
Обучение сотрудников	Тренинги, материалы	200,000

Общая сумма непрямых затрат: 500,000 руб.

3. Резерв на непредвиденные расходы

Дополнительные средства для покрытия рисков, связанных с задержками или дополнительными затратами.

- **Резерв на непредвиденные расходы:**
  (Прямые + Непрямые затраты) × 10% = (8,460,000 + 500,000) × 10% = 896,000 руб.

4. Управленческий резерв

Резерв на изменения в проекте, связанные с изменением требований или масштабов работ.

- **Управленческий резерв:**
  Прямые затраты × 5% = 8,460,000 × 5% = 423,000 руб.

5. Амортизация

Износ оборудования, используемого для проекта, рассчитанный на 5 лет.

- **Амортизация серверного оборудования:**
  500,000 ÷ 60 месяцев × 7 месяцев = 58,333 руб.

6. Косвенные затраты

Затраты на поддержку общей ИТ-инфраструктуры и административные расходы.

Категория	Описание	Итого (руб.)
ИТ-инфраструктура	Обеспечение стабильной работы	200,000
Содержание офиса	Канцелярия, мелкие расходы	100,000

Общая сумма косвенных затрат: 300,000 руб.

Общая сумма проекта: 10,637,333 руб.

Бюджет проекта

1. Общая структура бюджета

Категория затрат	Сумма (руб.)	Описание
Прямые затраты	8,460,000	Заработная плата сотрудников, оборудование, лицензии и инструменты.
Непрямые затраты	500,000	Аренда офиса, коммунальные услуги, обучение.
Резерв на непредвиденные расходы	896,000	Дополнительные средства для компенсации рисков.
Управленческий резерв	423,000	Фонд для крупных изменений в проекте.
Амортизация	58,333	Стоимость износа серверного оборудования за 7 месяцев.
Косвенные затраты	300,000	Затраты на поддержку ИТ-инфраструктуры и офисные расходы.

Итоговая сумма бюджета: 10,637,333 руб.

2. Распределение бюджета по этапам

Этап 1: Сбор данных

Категория затрат	Сумма (руб.)
Затраты на персонал	1,440,000
Оборудование и лицензии	300,000
Непрямые затраты	100,000
Итого:	1,840,000 руб.

Этап 2: Анализ требований

Категория затрат	Сумма (руб.)
Затраты на персонал	780,000
Программные инструменты	100,000
Непрямые затраты	50,000
Итого:	930,000 руб.

Этап 3: Разработка системы

Категория затрат	Сумма (руб.)
Затраты на персонал	2,480,000
Лицензии и ПО	400,000
Непрямые затраты	100,000
Амортизация	20,000
Итого:	3,000,000 руб.

Этап 4: Интеграция и тестирование

Категория затрат	Сумма (руб.)
Затраты на персонал	520,000
Мониторинг и тестирование	150,000
Непрямые затраты	50,000
Итого:	720,000 руб.

Этап 5: Внедрение и поддержка

Категория затрат	Сумма (руб.)
Затраты на персонал	990,000
Инфраструктура и поддержка	200,000
Обучение пользователей	200,000
Непрямые затраты	50,000
Резерв	896,000
Управленческий резерв	423,000

Итого: **2,759,000 руб.**

3. Общий бюджет с учетом резервов

Категория затрат	Сумма (руб.)
Сбор данных	1,840,000
Анализ требований	930,000
Разработка системы	3,000,000
Интеграция и тестирование	720,000
Внедрение и поддержка	2,759,000
Резерв на непредвиденные расходы	896,000
Управленческий резерв	423,000
Амортизация	58,333
Косвенные затраты	300,000

Итоговая сумма бюджета: 10,637,333 руб.

Показатели для анализа и оценки

ROI (Return on Investment)

Показывает рентабельность проекта, отражает, насколько прибыльным будет проект относительно вложенных средств.

- Формула:
  ROI = (Годовой чистый доход / Инвестиции) × 100%
- Расчет:
  Инвестиции = 5,000,000 руб.
  Годовой чистый доход = 2,000,000 руб.
  ROI = (2,000,000 / 5,000,000) × 100% = 40%
- Вывод:
  Проект приносит 40% дохода ежегодно, что свидетельствует о высокой экономической эффективности.

PP (Payback Period)

Срок окупаемости проекта, показывает, за сколько лет вложения вернутся.

- Формула:
  PP = Инвестиции / Годовой доход
- Расчет:
  PP = 5,000,000 / 2,000,000 = 2.5 года
- Вывод:
  Проект окупится через 2.5 года, что является разумным сроком.

NPV (Net Present Value)

Чистая приведенная стоимость показывает разницу между приведенными доходами и инвестициями с учетом дисконтирования.

- Формула:
  NPV = Σ (Чистый доход / (1 + r)^t) - Инвестиции
  где r — ставка дисконтирования, t — номер года.
- Пример расчета (ставка дисконтирования 10%, период — 5 лет):
  - Год 1: 2,000,000 / (1 + 0.1)^1 = 1,818,181 руб.
  - Год 2: 2,000,000 / (1 + 0.1)^2 = 1,652,893 руб.
  - Год 3: 2,000,000 / (1 + 0.1)^3 = 1,502,630 руб.
  - Год 4: 2,000,000 / (1 + 0.1)^4 = 1,366,027 руб.
  - Год 5: 2,000,000 / (1 + 0.1)^5 = 1,241,842 руб.
- NPV = (1,818,181 + 1,652,893 + 1,502,630 + 1,366,027 + 1,241,842) - 5,000,000 = 2,581,573 руб.
- Вывод:
  Положительное значение NPV подтверждает экономическую целесообразность проекта.

IRR (Internal Rate of Return)

Внутренняя норма доходности — ставка дисконтирования, при которой NPV = 0.

- Определение IRR:
  Для IRR необходимо решить уравнение, при котором:
  NPV = Σ (Чистый доход / (1 + IRR)^t) - Инвестиции = 0.
- Результат (приблизительно):
  IRR ≈ 18%, что выше средней ставки дисконтирования (10%).
- Вывод:
  IRR показывает, что проект генерирует доходность выше стандартных финансовых инвестиций.

Итоговая стоимость проекта

Итоговая стоимость проекта (**10,637,333 руб.**) — это полный бюджет, который включает:

- Прямые затраты (заработные платы, оборудование, лицензии).
- Непрямые затраты (аренда офиса, коммунальные услуги, обучение).
- Резервы (на непредвиденные расходы, управленческий резерв).
- Амортизацию и косвенные затраты.

Эта сумма представляет общий финансовый объем, который потребуется для выполнения проекта.

Инвестиции (5 млн руб.) — это стартовый капитал, необходимый для запуска и выполнения критических этапов.

Разница между этими двумя суммами покрывается за счет операционных доходов и дополнительных источников финансирования, которые проект начинает генерировать после запуска.

Метод оценки: Параметрический расчет обеспечил точную детализацию бюджета на основе рыночных данных. Аналоговый метод помог учесть резервы и амортизацию.

Показатели: Высокие ROI (40%), положительное NPV (2,581,573 руб.) и IRR (18%) подтверждают эффективность проекта.

Срок окупаемости: 2.5 года — разумный период для IT-проекта такого масштаба.

VIII - Управление коммуникациями проекта

Цель управления коммуникациями — обеспечить своевременный обмен информацией между всеми заинтересованными сторонами для поддержания эффективности работы команды и достижения целей проекта.

План управления коммуникациями

1. Идентификация заинтересованных сторон:
   - Участники: команда проекта, спонсоры, заказчики, внешние консультанты.

- Уровень вовлеченности: от высокой (команда) до минимальной (заказчик, внешние стороны).

2. Каналы коммуникации:
- Внутренние: ежедневные стендапы, командные чаты (Slack, MS Teams).
- Внешние: отчеты заказчикам, презентации прогресса, согласования.

3. Частота коммуникации:
- Ежедневно: оперативные задачи (стендапы, синхронизация).
- Еженедельно: отчеты о выполнении задач и анализ рисков.
- Ежемесячно: контрольные точки и обновления для спонсоров/заказчиков.

4. Форматы коммуникации:
- Очные встречи.
- Онлайн-видеоконференции (Zoom, Google Meet).
- Электронные письма (отчеты, документы).

5. Системы управления коммуникациями:
- Jira (отслеживание задач, статусов).
- Confluence (документация, планирование).
- Google Workspace (презентации, отчеты).

Таблица коммуникаций

Тип коммуникации	Целевая аудитория	Цель коммуникации	Частота	Канал/Формат	Ответственный
Ежедневный стендап	Команда проекта	Синхронизация задач, выявление блокеров	Ежедневно	Slack, MS Teams, Zoom	Проектный менеджер
Еженедельный отчет	Спонсоры, команда	Обзор прогресса, выявление проблем	Еженедельно	Confluence, Email	Проектный менеджер
Контрольная точка	Руководство, заказчики	Согласование ключевых решений и рисков	Ежемесячно	Zoom, Google Meet	Проектный менеджер

Финальная презентация	Спонсоры, заказчики	Подведение итогов, демонстрация результатов	В конце проекта	Презентация, Zoom	Проектный менеджер
Обсуждение требований	Бизнес-аналитики, заказчики	Сбор и согласование ключевых требований	На этапе анализа	Интервью, Email	Бизнес-аналитик
Отчет по тестированию	Команда тестирования, руководство	Результаты тестирования, выявленные проблемы	На этапе тестирования	Confluence, Email	Тестировщик
Тренинг для пользователей	Пользователи, руководство	Обучение использованию системы	На этапе внедрения	Очные встречи, презентация	Бизнес-аналитик
Обратная связь	Команда, руководство, заказчики	Уточнение улучшений и пожеланий	На всех этапах	Анкеты, обсуждения	Проектный менеджер
Координация задач	Вся команда проекта	Распределение и контроль выполнения задач	Регулярно (по задаче)	Jira, MS Teams	Проектный менеджер
Мониторинг метрик	DevOps, руководство	Обзор производительности системы	Еженедельно	Grafana, Prometheus	DevOps-инженер

Примечания к таблице:

1. **Частота коммуникаций:** В зависимости от стадии проекта, частота может варьироваться.
2. **Канал/Формат:** Каналы выбираются в зависимости от аудитории (онлайн-инструменты, очные встречи).
3. **Ответственный:** Определенный человек или роль отвечает за своевременность и точность коммуникации.

Управление коммуникациями на этапах проекта

Этап 1: Сбор данных

Цель: Согласовать источники данных, обеспечить доступ и коммуникацию между технической командой и владельцами данных.

Ключевые коммуникации	**Целевая аудитория**	**Формат/ Канал**	**Частота**	**Ответственный**
Согласование источников данных	Заказчики, бизнес-аналитики	Очные встречи, Email	Однократно	Бизнес-аналитик
Запрос доступа к системам	Технические специалисты, заказчики	Email, Jira	По мере необходимости	DevOps-инженер
Обсуждение форматов выгрузки данных	Data Scientist, владельцы данных	Онлайн-встречи, Slack	На старте этапа	Проектный менеджер

Этап 2: Анализ требований

Цель: Уточнить бизнес-процессы, выявить ключевые требования, согласовать метрики и нефункциональные требования.

Ключевые коммуникации	**Целевая аудитория**	**Формат/Канал**	**Частота**	**Ответственный**
Интервью с ключевыми заинтересованными сторонами	Бизнес-аналитики, заказчики	Очные встречи, Zoom	Еженедельно	Бизнес-аналитик
Обсуждение бизнес-процессов	Команда аналитиков	Внутренние встречи	По мере необходимости	Бизнес-аналитик
Утверждение пользовательских требований	Руководство, заказчики	Презентации, Email	По завершении анализа	Проектный менеджер

Этап 3: Разработка системы

Цель: Обеспечить синхронизацию между командами разработки, документировать архитектуру и API.

Ключевые коммуникации	**Целевая аудитория**	**Формат/Канал**	**Частота**	**Ответственный**
Синхронизация разработчиков	Data Scientist, DevOps, Frontend	Ежедневные стендапы	Ежедневно	Проектный менеджер
Обсуждение технической архитектуры	Команда разработки	Внутренние встречи, Confluence	По мере необходимости	DevOps-инженер
Утверждение промежуточных результатов	Заказчики, руководство	Презентации, Email	По завершении задач	Проектный менеджер

Этап 4: Интеграция и тестирование

Цель: Обеспечить взаимодействие между компонентами системы, согласовать тестовые сценарии и их результаты.

Ключевые коммуникации	**Целевая аудитория**	**Формат/Канал**	**Частота**	**Ответственный**
Согласование интеграции компонентов	DevOps, тестировщики	Встречи, Jira	По мере необходимости	DevOps-инженер
Обсуждение тестовых сценариев	Тестировщики, разработчики	Внутренние встречи	На старте тестирования	Тестировщик
Отчет о тестировании	Руководство, заказчики	Confluence, Email	Еженедельно	Тестировщик

Этап 5: Внедрение и поддержка

Цель: Обучить пользователей системе, передать систему в эксплуатацию и организовать мониторинг.

Ключевые коммуникации	Целевая аудитория	Формат/Канал	Частота	Ответственный
Обучение пользователей	Конечные пользователи	Очные тренинги, презентации	Однократно	Бизнес-аналитик
Передача системы в эксплуатацию	Руководство, DevOps	Презентация, документация	По завершении разработки	DevOps-инженер
Обсуждение результатов мониторинга	Руководство, пользователи	Дашборды (Grafana), Email	Еженедельно	Проектный менеджер

Показатели эффективности коммуникаций (KPI)

Для успешного управления коммуникациями в проекте важно измерять их эффективность. Вот основные показатели и способы их расчета:

1. **Уровень вовлеченности (Engagement Rate)**

Цель: Оценить, насколько активно участники проекта вовлечены в коммуникации.

- **Метрика:** Доля участников, которые активно участвуют в обсуждениях, стендапах и других форматах.
- **Пример:** Из 10 членов команды 8 активно участвуют в обсуждениях.

Уровень вовлеченности=(8/10)×100=80%

Идеальное значение: 80% и выше.

2. **Скорость ответа (Response Time)**

Цель: Измерить, как быстро команда реагирует на запросы и вопросы.

- **Метрика:** Среднее время ответа на запрос.
- **Пример:** На 5 запросов было потрачено 10 часов. Скорость ответа=10/5=2 часа
- **Идеальное значение:** ≤ 4 часа.

3. Соответствие форматов (Compliance Rate)

Цель: Оценить, насколько эффективно команда соблюдает установленные форматы коммуникаций (отчеты, встречи, стендапы).

- **Метрика:** Доля проведенных мероприятий или созданных отчетов, соответствующих формату.
- **Пример:** Из 10 запланированных стендапов 9 проведены. Соответствие форматов=(9/10)×100=90%
- **Идеальное значение:** 90% и выше.

4. Количество конфликтов (Conflict Rate)

Цель: Измерить частоту конфликтов, возникающих из-за ошибок в коммуникации.

- **Метрика:** Число конфликтов за определенный период.
- **Пример:** За неделю было 50 коммуникаций и 2 конфликта. Уровень конфликтов=(2/50)×100=4%
- **Идеальное значение:** ≤ 5%.

5. Уровень обратной связи (Feedback Rate)

Цель: Оценить качество коммуникации через сбор обратной связи от участников.

- **Метрика:** Средняя оценка коммуникации по шкале от 1 до 5.
- **Метод:** Опрос участников с вопросами, например:
  - Насколько вы довольны регулярностью коммуникаций?
  - Насколько понятно подаются материалы?
- **Пример:** Средняя оценка обратной связи: 4.2 из 5.
- **Идеальное значение:** ≥ 4 из 5.

6. Доля нерешенных вопросов (Unresolved Issues)

Цель: Оценить, насколько быстро решаются вопросы, поднятые в коммуникациях.

- **Метрика:** Число нерешенных вопросов по сравнению с общим числом вопросов.
- **Пример:** Из 20 вопросов 2 остаются нерешенными. Доля нерешенных вопросов=(2/20)×100=10%
- **Идеальное значение:** ≤ 10%.

7. Своевременность отчетов (Report Timeliness)

Цель: Оценить, насколько регулярно и вовремя создаются отчеты.

- **Метрика:** Доля отчетов, подготовленных вовремя.
- **Пример:** Из 15 отчетов 14 подготовлены вовремя.
  Своевременность отчетов=(14/15)×100=93%
- **Идеальное значение:** 90% и выше.

Риски управления коммуникациями и стратегии их минимизации

Основные риски

Риск	**Описание**
1.1. Несвоевременный обмен информацией	Задержки в передаче данных между участниками проекта могут привести к срыву сроков выполнения задач.
1.2. Потеря информации	Отсутствие четкой системы для хранения и структурирования данных может привести к потере ключевой информации.
1.3. Недопонимание между участниками	Неясные формулировки или отсутствие единых стандартов коммуникации могут вызвать конфликты и разночтения.
1.4. Перегрузка информацией	Чрезмерный объем коммуникаций, особенно нерелевантных, отвлекает команду и снижает продуктивность.
1.5. Проблемы с доступностью участников	Участники проекта могут быть недоступны в нужный момент из-за графика, разницы часовых поясов или других факторов.
1.6. Технические сбои	Проблемы с инструментами коммуникации (например, сбои в работе Slack, Zoom или Jira) могут затруднить оперативное взаимодействие.
1.7. Отсутствие обратной связи	Недостаток регулярной обратной связи от участников или руководства может затруднить корректировку процесса.
1.8. Конфликты в команде	Разные точки зрения или личные конфликты между участниками могут тормозить принятие решений.

Стратегии минимизации рисков

2.1. Для риска: Несвоевременный обмен информацией

- **Стратегия:**

1. Внедрить регулярные встречи (ежедневные стендапы, еженедельные отчеты).
2. Использовать системы уведомлений (Slack, MS Teams) для срочных сообщений.
3. Назначить ответственных за своевременное предоставление информации.

- **Инструменты:** Jira для отслеживания задач, Slack для оперативного общения.

2.2. Для риска: Потеря информации

- **Стратегия:**

1. Организовать централизованное хранилище данных (например, Confluence или Google Drive).
2. Создать четкие правила ведения документации.
3. Регулярно резервировать данные.

- **Инструменты:** Confluence для документирования, Google Drive для резервного копирования.

2.3. Для риска: Недопонимание между участниками

- **Стратегия:**

1. Внедрить единый формат для отчетов и коммуникаций.
2. Уточнять ключевые задачи и формулировки через чек-листы.
3. Проводить регулярные уточняющие сессии.

- **Инструменты:** Структурированные отчеты в Excel, презентации для наглядности.

2.4. Для риска: Перегрузка информацией

- **Стратегия:**

1. Фильтровать информацию перед передачей (только релевантные данные).
2. Установить ограничения на объем информации для коммуникаций.
3. Использовать дашборды для визуализации прогресса.

- **Инструменты:** Grafana для визуализации метрик, Slack для тематических каналов.

2.5. Для риска: Проблемы с доступностью участников

- **Стратегия:**

1. Составить график доступности участников с учетом часовых поясов.
2. Назначить резервных представителей для срочных вопросов.
3. Использовать асинхронные форматы общения (например, записи встреч).

- **Инструменты:** Google Calendar для планирования, Zoom для записи встреч.

2.6. Для риска: Технические сбои

- **Стратегия:**

1. Внедрить резервные каналы связи (например, Telegram или WhatsApp).
2. Регулярно обновлять и проверять используемые инструменты.
3. Назначить ответственного за мониторинг работоспособности систем.

- **Инструменты:** Резервный Slack-канал, альтернативные платформы (Discord, Microsoft Teams).

2.7. Для риска: Отсутствие обратной связи

- **Стратегия:**

1. Регулярно собирать обратную связь через анкеты и опросы.
2. Анализировать обратную связь и внедрять корректировки.
3. Назначить контактное лицо для вопросов и предложений.

- **Инструменты:** Google Forms для опросов, Confluence для хранения обратной связи.

2.8. Для риска: Конфликты в команде

- **Стратегия:**

1. Проводить командные тренинги для улучшения взаимодействия.
2. Внедрить четкий процесс разрешения конфликтов (например, привлечение третьей стороны).
3. Обеспечить прозрачность решений.

- **Инструменты:** Очные встречи с модератором, специальные сессии по разрешению конфликтов.

Эффективное управление коммуникациями включает:

1. Применение правильных инструментов для систематизации и контроля.
2. Установление четких регламентов взаимодействия.
3. Регулярное оценивание качества и корректировка процессов.

IX - Управление рисками проекта

Управление рисками проекта — это процесс выявления, анализа, планирования и контроля факторов, которые могут повлиять на выполнение задач или достижение целей проекта. Для успешного управления рисками необходимо соблюдать системный подход.

Процесс управления рисками

Этапы управления рисками:

1. **Идентификация рисков**
   - Определение всех возможных рисков, которые могут повлиять на проект.
   - Используются такие инструменты, как мозговой штурм, анализ предыдущих проектов, интервью с экспертами.
2. **Анализ рисков**
   - **Качественный анализ:** Оценка вероятности возникновения риска и его влияния.
   - **Количественный анализ:** Определение потенциальных финансовых или временных последствий рисков.
3. **Планирование реагирования**
   - Определение стратегий для минимизации последствий рисков.
   - Подготовка резервных планов.
4. **Мониторинг и контроль**
   - Отслеживание идентифицированных рисков.
   - Оценка эффективности стратегий реагирования.
   - Выявление новых рисков.

Классификация рисков

По характеру влияния

- **Технические риски:**
  - Отказ оборудования.

- Ошибки в программном обеспечении.
- Проблемы интеграции.
- **Организационные риски:**
- Низкая производительность команды.
- Недостаточная квалификация сотрудников.
- Конфликты внутри команды.
- **Экономические риски:**
- Превышение бюджета.
- Рост стоимости ресурсов.
- Недостаточное финансирование.
- **Внешние риски:**
- Изменения законодательства.
- Экономическая нестабильность.
- Проблемы с поставщиками.

Основные инструменты и методы управления рисками

1. **Матричный анализ вероятности и влияния (Probability and Impact Matrix)**
- Используется для качественного анализа рисков, позволяет классифицировать риски по их вероятности и влиянию. Риски с высокой вероятностью и значительным влиянием требуют особого внимания и более детального планирования.
2. **Метод Монте-Карло (Monte Carlo Simulation)**
- Применяется для количественного анализа рисков. Позволяет провести анализ на основе моделирования различных сценариев, оценивая влияние рисков на бюджет и сроки проекта.
3. **Деревья решений (Decision Tree Analysis)**
- Метод для анализа решений в условиях неопределенности, который помогает оценить различные варианты действий и их последствия, учитывая вероятность каждого из них.
4. **SWOT-анализ (SWOT Analysis)**
- Позволяет выявить сильные и слабые стороны проекта, а также возможности и угрозы. Этот анализ помогает лучше понять, как риски могут повлиять на проект и где можно воспользоваться возможностями.
5. **Анализ сценариев (What-if Scenario Analysis)**
- Анализ различных сценариев для прогнозирования влияния рисков на проект и оценка последствий различных вариантов развития событий.
6. **Реестр рисков (Risk Register)**

- Документ, в котором фиксируются все идентифицированные риски, их вероятности, влияние, стратегии реагирования и текущий статус. Реестр рисков помогает отслеживать риски на протяжении всего проекта.

Для нашего проекта оптимальным набором инструментов и методов управления рисками являются:

1. **SWOT-анализ** для начальной оценки.
2. **Реестр рисков** для централизованного учета.
3. **Количественный анализ** для финансового планирования.
4. **Мониторинг метрик (Prometheus, Grafana)** для контроля текущих рисков.
5. **Jira и Confluence** для отслеживания и документирования рисков.

Риски проекта

Риск 1: Отказ серверного оборудования

- **Нежелательное событие:** Отказ серверного оборудования во время обучения модели или её тестирования.
- **Все его последствия:**
  - Задержка сроков обучения модели.
  - Потеря данных из-за недостаточной резервной копии.
  - Увеличение затрат на восстановление работоспособности системы.
- **Степень серьезности влияния:** Высокая.
- **Вероятность события:** Средняя.
- **Время, когда вероятное событие может произойти:** Средний этап (обучение и тестирование модели).
- **Взаимосвязь с другими частями проекта:** Задержка обучения модели приведёт к нарушению сроков интеграции и тестирования системы.

Риск 2: Ошибки прогнозной модели

- **Нежелательное событие:** Модель прогнозирования показывает низкую точность или некорректные прогнозы.
- **Все его последствия:**
  - Неправильные бизнес-решения на основе неточных прогнозов.
  - Снижение доверия пользователей к системе.

- Дополнительные затраты на доработку модели.
- **Степень серьезности влияния:** Средняя.
- **Вероятность события:** Средняя.
- **Время, когда вероятное событие может произойти:** Средний этап (разработка и тестирование модели).
- **Взаимосвязь с другими частями проекта:** Ошибки в модели могут потребовать переработки данных и затянуть сроки разработки.

Риск 3: Задержка доступа к данным

- **Нежелательное событие:** Заказчики или владельцы данных не предоставляют доступ к данным вовремя.
- **Все его последствия:**
- Отсрочка начала работы с данными.
- Задержка разработки модели прогнозирования.
- Увеличение затрат на проект из-за простоя команды.
- **Степень серьезности влияния:** Высокая.
- **Вероятность события:** Средняя.
- **Время, когда вероятное событие может произойти:** Ранний этап (сбор данных).
- **Взаимосвязь с другими частями проекта:** Задержка доступа к данным затормозит все последующие этапы, включая анализ требований и разработку системы.

Риск 4: Превышение бюджета

- **Нежелательное событие:** Фактические затраты превышают запланированный бюджет проекта.
- **Все его последствия:**
- Угроза завершения проекта из-за нехватки средств.
- Снижение качества системы из-за необходимости урезания затрат.
- Увеличение времени на реализацию.
- **Степень серьезности влияния:** Высокая.
- **Вероятность события:** Средняя.
- **Время, когда вероятное событие может произойти:** Поздний этап (внедрение и поддержка).
- **Взаимосвязь с другими частями проекта:** Превышение бюджета может привести к недофинансированию тестирования или поддержки.

Риск 5: Конфликты в команде

- **Нежелательное событие:** Конфликты между участниками команды приводят к снижению продуктивности.
- **Все его последствия:**
  - Задержка выполнения задач.
  - Ухудшение морального духа команды.
  - Увеличение времени на координацию задач.
- **Степень серьезности влияния:** Средняя.
- **Вероятность события:** Низкая.
- **Время, когда вероятное событие может произойти:** Средний этап (разработка системы).
- **Взаимосвязь с другими частями проекта:** Конфликты могут нарушить согласованность работы всех участников, особенно при интеграции компонентов.

Риск 6: Изменение требований заказчика

- **Нежелательное событие:** Заказчики меняют требования на поздних этапах проекта.
- **Все его последствия:**
  - Дополнительные затраты на переработку системы.
  - Увеличение сроков завершения проекта.
  - Нарушение стабильности уже разработанных модулей.
- **Степень серьезности влияния:** Средняя.
- **Вероятность события:** Низкая.
- **Время, когда вероятное событие может произойти:** Поздний этап (внедрение и поддержка).
- **Взаимосвязь с другими частями проекта:** Изменение требований может привести к необходимости пересмотра функционала системы и дополнительным нагрузкам на команду.

Риск 7: Непредвиденные ошибки в данных

- **Нежелательное событие:** Обнаружение значительного количества ошибок или пропусков в данных.
- **Все его последствия:**
  - Увеличение времени на очистку данных.
  - Снижение точности модели прогнозирования.
  - Задержка этапов анализа требований и разработки.
- **Степень серьезности влияния:** Средняя.
- **Вероятность события:** Высокая.
- **Время, когда вероятное событие может произойти:** Ранний этап (сбор данных).

- **Взаимосвязь с другими частями проекта:** Ошибки в данных могут повлиять на обучение модели и качество прогнозов.

Риск 8: Технические сбои инструментов

- **Нежелательное событие:** Сбой в работе используемых инструментов (например, Apache Airflow, Docker).
- **Все его последствия:**
  - Увеличение времени на устранение ошибок.
  - Возможность потери настроек или данных.
  - Замедление процессов автоматизации.
- **Степень серьезности влияния:** Средняя.
- **Вероятность события:** Средняя.
- **Время, когда вероятное событие может произойти:** Средний этап (разработка системы).
- **Взаимосвязь с другими частями проекта:** Технические сбои могут затормозить интеграцию и автоматизацию процессов.

Таблица оценки рисков

Событие	Вероятность	Серьезность	Трудность обнаружения	Время реализации	Уровень угрозы	Рекомендации
Отказ серверного оборудования	Высокая (>60%)	Сильное	Средняя сложность	Средний этап	Критическая	Заключить договор с поставщиком на срочную замену оборудования.
Ошибка прогнозной модели	Средняя (40%)	Среднее	Легко обнаружимый	Средний этап	Высокая	Провести тестирование на тестовом наборе данных.
Задержка доступа к данным	Средняя (50%)	Среднее	Трудно обнаружимый	Ранний этап	Высокая	Утвердить график предоставления доступа у всех сторон на этапе планирования.
Низкая квалификация сотрудников	Низкая (15%)	Среднее	Трудно обнаружимый	Средний этап	Средняя	Организовать обучение и дополнительные тренинги для команды.

Превышение бюджета	Средняя (30%)	Сильное	Средняя сложность	Поздний этап	Высокая	Закладывать резерв на непредвиденные расходы (10–15%).
Изменение требований заказчика	Низкая (20%)	Слабое	Легко обнаружимый	Поздний этап	Средняя	Постоянное согласование изменений с заказчиком, использование управленческого резерва.

Методы реагирования на риски

Стратегии для негативных рисков

- **Избежание:** Устранение причин риска. Например, использование проверенного оборудования.
- **Снижение:** Уменьшение вероятности или влияния риска. Например, тестирование модели перед запуском.
- **Передача:** Перенос ответственности на третью сторону. Например, заключение SLA с поставщиками.
- **Принятие:** Признание риска без дополнительных действий. Например, учет возможных потерь.

Стратегии для позитивных рисков

- **Использование:** Активное применение выгод. Например, внедрение новой технологии для повышения эффективности.
- **Повышение:** Увеличение вероятности наступления позитивного риска.
- **Совместное использование:** Распределение выгоды от позитивного риска между несколькими сторонами.

Мониторинг и контроль рисков

Методы мониторинга

- Использование дашбордов в Grafana для отслеживания ключевых метрик (например, точности прогнозов).
- Регулярные встречи команды для пересмотра текущих рисков.
- Проверка состояния задач, связанных с рисками, через Jira.

Метрики для мониторинга

- **Уровень вовлеченности:** Доля участников, вовлеченных в управление рисками.
- **Доля реализованных стратегий:** Процент рисков, для которых реализованы стратегии реагирования.
- **Количество новых рисков:** Риски, выявленные за последний период.
- **Скорость устранения рисков:** Среднее время устранения риска с момента выявления.

Резервы на риски

Финансовые резервы

- **Цель:** Покрытие расходов на устранение рисков, которые реализуются.
- **Размер:** 10–15% от общего бюджета проекта.

Резервы времени

- **Цель:** Компенсация возможных задержек из-за реализации рисков.
- **Размер:** 5–10% от общего времени проекта.

Ответственные за управление рисками

- **Проектный менеджер:** Общая координация управления рисками.
- **DevOps-инженер:** Управление техническими рисками.
- **Бизнес-аналитик:** Идентификация и анализ бизнес-рисков.
- **Финансовый аналитик:** Оценка экономических последствий рисков.

Пример реализации плана для одного риска

Этап	Действия	Ответственный
Идентификация	Выявление риска отказа серверного оборудования.	DevOps-инженер
Анализ	Оценка вероятности (Высокая) и влияния (Высокая).	Проектный менеджер
Планирование реагирования	Заключение SLA с поставщиком, настройка резервного сервера.	DevOps-инженер

Мониторинг	Настройка дашбордов для контроля состояния серверов.	DevOps-инженер
Контроль	Еженедельная проверка работоспособности оборудования.	DevOps-инженер

Заключение

В рамках подготовки проекта по разработке интеллектуальной системы прогнозирования спроса был проведен всесторонний анализ, включающий исследование его реализуемости, определение экономической эффективности, фиксацию ключевых параметров и разработку детализированных планов управления. Отчет содержит результаты анализа, предложения по реализации и описание подходов к управлению содержанием, командой, рисками и коммуникациями.

Основные выводы

1. **Исследование реализуемости проекта:**
   - Проект доказал свою реализуемость с точки зрения технических и организационных возможностей.
   - Определены ключевые требования к системе, включая обработку больших данных, автоматизацию процессов и точность прогнозов.
2. **Экономическая эффективность:**
   - Рассчитанные показатели эффективности (ROI, PP, NPV) демонстрируют финансовую обоснованность проекта.
   - Система имеет потенциал для значительного снижения издержек и повышения прибыльности компании.
3. **Управление проектом:**
   - Разработаны детализированные WBS и OBS, позволяющие структурировать задачи и распределить роли.
   - Созданы планы управления рисками и коммуникациями, минимизирующие потенциальные угрозы.

Основные риски

1. **Отказ серверного оборудования.**
2. **Ошибки прогнозной модели.**
3. **Задержка доступа к данным.**
4. **Превышение бюджета.**

Для каждого риска предложены стратегии управления, включая снижение вероятности, резервирование времени и бюджета, а также создание резервных планов.

Рекомендации

1. **Дополнительные этапы согласования:** Утвердить бюджет и график реализации с руководством.
2. **Начало пилотного этапа:** Провести ограниченное тестирование системы для проверки гипотез.
3. **Регулярное обновление плана:** Корректировать планы на основе новых данных и анализа рынка.
4. **Расширение команды:** Увеличить численность команды, если проектная нагрузка превысит ожидаемую.

Ожидаемые эффекты от внедрения системы прогнозирования спроса

Внедрение интеллектуальной системы прогнозирования спроса обеспечит ряд положительных эффектов, которые будут способствовать оптимизации бизнес-процессов, снижению издержек и повышению прибыли компании.

Экономические эффекты

1. **Снижение издержек на управление запасами:**
   - Уменьшение избыточных запасов благодаря точному прогнозированию спроса.
   - Сокращение затрат на хранение и логистику.
2. **Повышение точности маркетинговых решений:**
   - Оптимизация акций и скидок на основе прогнозов потребительского поведения.
   - Увеличение эффективности рекламных кампаний.
3. **Увеличение прибыли:**
   - Прогнозы помогут лучше адаптироваться к изменению спроса, что приведёт к росту продаж.
   - Повышение удовлетворённости клиентов за счёт наличия нужных товаров в нужное время.

Операционные эффекты

1. **Ускорение процессов планирования:**
   - Автоматизация процессов сбора и обработки данных сокращает время на подготовку прогнозов.

2. **Улучшение управления цепочкой поставок:**
   - Прогнозирование спроса позволяет точнее планировать заказы, минимизируя задержки поставок.
3. **Сокращение ошибок в данных:**
   - Очистка и нормализация данных повысит качество аналитики и прогнозов.

Технологические эффекты

1. **Увеличение производительности:**
   - Система автоматизирует рутинные задачи, что позволяет команде фокусироваться на стратегических задачах.
2. **Масштабируемость системы:**
   - Возможность использования модели для новых категорий товаров или регионов.
3. **Интеграция с существующими системами:**
   - Система может быть легко интегрирована с текущими ERP- и CRM-системами компании.

Организационные эффекты

1. **Улучшение взаимодействия между подразделениями:**
   - Централизованная система прогнозирования способствует выравниванию целей между отделами закупок, маркетинга и логистики.
2. **Повышение квалификации сотрудников:**
   - Обучение сотрудников работе с новой системой развивает их аналитические навыки.
3. **Увеличение прозрачности процессов:**
   - Регулярное отслеживание данных и прогнозов улучшает контроль и принятие решений.

Социальные эффекты

1. **Повышение удовлетворённости клиентов:**
   - Прогнозирование спроса помогает обеспечивать наличие товаров на полках, минимизируя дефицит или излишки.
2. **Улучшение условий работы сотрудников:**
   - Автоматизация рутинных задач снижает нагрузку на сотрудников.

Эффекты управления рисками

1. **Снижение вероятности финансовых потерь:**

- Точные прогнозы минимизируют риск дефицита товаров или излишков на складах.

2. **Управление изменениями на рынке:**
   - Возможность быстро адаптироваться к изменениям спроса, вызванным сезонностью или внешними факторами.

Вывод

Данный отчет служит основой для последующего принятия решений о реализации проекта. Он включает детализированные расчеты, планы и стратегии управления, обеспечивая готовность к успешному выполнению задач. При наличии утверждения проекта предлагается переходить к следующему этапу — началу работ по внедрению системы.

Источники

1. Официальный сайт Apache Airflow. Доступ: https://airflow.apache.org (дата обращения: 19.11.2024).
2. Руководство по SARIMA-модели на сайте StatQuest. Доступ: https://statquest.org/statistics/sarima (дата обращения: 19.11.2024).
3. Документация по Prometheus. Доступ: https://prometheus.io/docs/introduction/overview (дата обращения: 19.11.2024).
4. Руководство по интеграции Docker. Доступ: https://docs.docker.com (дата обращения: 19.11.2024).
5. Официальный сайт Google Cloud Platform. Доступ: https://cloud.google.com (дата обращения: 19.11.2024).
6. Статья о моделировании временных рядов. Доступ: https://towardsdatascience.com/time-series-forecasting (дата обращения: 19.11.2024).
7. Документация по графикам Gantt в Jira. Доступ: https://jira.atlassian.com (дата обращения: 19.11.2024).
8. Официальный блог Grafana Labs. Доступ: https://grafana.com/blog (дата обращения: 19.11.2024).
9. Ресурс по метрикам прогнозирования (MAE, RMSE). Доступ: https://machinelearningmastery.com/metrics (дата обращения: 19.11.2024).
10. Руководство по управлению проектами на сайте PMI. Доступ: https://www.pmi.org (дата обращения: 19.11.2024).
11. Официальный сайт Python. Доступ: https://www.python.org (дата обращения: 19.11.2024).
12. Руководство по работе с API Google. Доступ: https://developers.google.com/apis (дата обращения: 19.11.2024).
13. Статья о рисках в IT-проектах. Доступ: https://scrum.org/blog/risk-management (дата обращения: 19.11.2024).
14. Сайт по управлению рисками в проектах. Доступ: https://www.riskmanagementinsights.com (дата обращения: 19.11.2024).
15. Руководство по планированию WBS. Доступ: https://www.workbreakdownstructure.com (дата обращения: 19.11.2024).
16. Руководство по KPI для команд разработки. Доступ: https://www.kpi.org (дата обращения: 19.11.2024).
17. Руководство по Agile. Доступ: https://www.agilealliance.org (дата обращения: 19.11.2024).

18. Материалы по прогнозированию спроса. Доступ: https://supplychainforecasting.com (дата обращения: 19.11.2024).
19. Официальная документация Microsoft Azure. Доступ: https://azure.microsoft.com (дата обращения: 19.11.2024).
20. Информация по методологии Scrum. Доступ: https://scrumguides.org (дата обращения: 19.11.2024).
21. Руководство по работе с Google Sheets API. Доступ: https://developers.google.com/sheets (дата обращения: 19.11.2024).
22. Материалы по построению пользовательских историй. Доступ: https://www.atlassian.com/software/jira/guides (дата обращения: 19.11.2024).
23. Официальный сайт RStudio. Доступ: https://www.rstudio.com (дата обращения: 19.11.2024).
24. Описание метода критического пути. Доступ: https://projectmanagement.com/critical-path-method (дата обращения: 19.11.2024).
25. Статья по управлению ресурсами в проектах. Доступ: https://resourceplanning.com (дата обращения: 19.11.2024).
26. Информация о стандартах ISO для управления проектами. Доступ: https://www.iso.org/standard/project-management (дата обращения: 19.11.2024).
27. Руководство по расчёту NPV и IRR. Доступ: https://corporatefinanceinstitute.com/resources (дата обращения: 19.11.2024).
28. Блог по автоматизации процессов. Доступ: https://automation.com/blog (дата обращения: 19.11.2024).
29. Руководство по Docker-Compose. Доступ: https://docs.docker.com/compose (дата обращения: 19.11.2024).
30. Информация о машинном обучении. Доступ: https://scikit-learn.org/stable (дата обращения: 19.11.2024).

Приложения

Документация для системы прогнозирования спроса

Оглавление

Описание проекта

Название: Система прогнозирования спроса для e-commerce

Цель проекта: Создать автоматизированную систему, которая прогнозирует спрос на товары в e-commerce, помогая управлять запасами, минимизировать дефицит и излишки, а также оптимизировать маркетинговые кампании.

Описание работы системы: Система анализирует исторические данные о продажах и использует методы временного анализа и прогнозирования (SARIMA) для создания прогноза спроса на будущие периоды. Основной акцент сделан на учете сезонных колебаний спроса и его зависимости от различных факторов.

Основные функциональные возможности:

- Ежедневный автоматический сбор и предобработка данных.
- Прогнозирование спроса на 30 дней вперед с учетом сезонных факторов.
- Мониторинг точности модели с помощью метрик MAE и RMSE.
- Настройка оповещений в случае ухудшения производительности модели.

Результаты и ценность для бизнеса:

- Прогнозы позволяют компании планировать запасы и минимизировать убытки от излишков или нехватки товаров.
- Возможность адаптации маркетинговых усилий под ожидаемые изменения спроса.
- Уменьшение ручного труда за счет автоматизации процессов обновления данных и переобучения модели.

Архитектура системы

- **Модель прогнозирования (SARIMA)**: Выполняет анализ временного ряда и генерирует прогноз на основе недельной сезонности.
- **Apache Airflow**: Управляет автоматизацией процессов, включая обновление данных, предобработку, переобучение модели и генерацию прогноза.
- **Docker**: Обеспечивает изоляцию окружения и удобство развертывания.
- **Prometheus и Grafana**: Мониторинг метрик модели и визуализация производительности.
- **Alertmanager**: Настроен на оповещения, если метрики (например, MAE, RMSE) выходят за допустимые пороги.

Основные компоненты архитектуры:

1. **Модель прогнозирования спроса (SARIMA)**:

- Основной алгоритм временного анализа, который предсказывает объем продаж, учитывая недельные сезонные колебания и тренды.
- Параметры модели настроены для достижения высокой точности прогноза (используется методика Grid Search для оптимизации).

2. **Apache Airflow**:
- Управляет автоматизацией процессов в системе, таких как обновление данных, предобработка, переобучение модели и генерация прогнозов.
- Запланированные DAG (Directed Acyclic Graphs) в Airflow выполняют задачи по заданному расписанию (например, ежедневно).

3. **Docker**:
- Контейнеризирует приложение, упрощая его развертывание и изоляцию всех необходимых зависимостей.
- Используется для упрощения развертывания как в локальной, так и в облачной среде, обеспечивая воспроизводимость.

4. **Prometheus и Grafana**:
- **Prometheus**: Система мониторинга, собирает метрики точности модели (например, MAE, RMSE) и другие показатели производительности.
- **Grafana**: Визуализирует метрики производительности модели, позволяет отслеживать их в режиме реального времени.

5. **Alertmanager**:
- Система оповещений в экосистеме Prometheus, отправляет уведомления при достижении заданных порогов метрик (например, если ошибка прогноза превышает допустимое значение).

Поток данных:

1. **Сбор данных → Предобработка → Обучение модели → Прогнозирование → Мониторинг и оповещения**.

Используемые технологии

Для реализации системы были выбраны следующие технологии и инструменты:

1. **Язык программирования: Python**
- Используется для предобработки данных, построения модели прогнозирования, и автоматизации процессов.
- Поддерживает большое количество библиотек для анализа данных и временных рядов, что делает его подходящим для проекта.

2. **Библиотеки для анализа данных и построения модели**:
- **pandas**: Обработка и манипуляция данными (например, очистка данных, агрегирование).
- **numpy**: Быстрые вычисления и математические операции.
- **statsmodels**: Реализация ARIMA и SARIMA для временного анализа и прогнозирования.

- **scikit-learn**: Для расчета метрик модели, таких как MAE и RMSE, и для подготовки данных.
3. **Apache Airflow**:
- Автоматизация процессов ETL и моделирования. Airflow управляет DAG (Directed Acyclic Graph), который контролирует порядок выполнения задач: от извлечения и обработки данных до прогноза.
4. **Docker**:
- Контейнеризация приложения для удобного развертывания в локальной и облачной средах.
- Гарантирует, что все зависимости и конфигурации системы останутся неизменными, что упрощает переносимость и масштабируемость.
5. **Prometheus и Grafana**:
- **Prometheus**: Система мониторинга, собирает метрики производительности модели (MAE, RMSE) и других важных параметров.
- **Grafana**: Позволяет визуализировать метрики и создавать наглядные дашборды для контроля точности модели и состояния системы в реальном времени.
6. **Alertmanager**:
- Настроен для отправки уведомлений при достижении порогов метрик (например, ухудшение точности модели). Работает в связке с Prometheus и может отправлять уведомления по email или в мессенджеры.

Развертывание системы

Развертывание системы включает несколько шагов: настройку Docker для контейнеризации приложения, установку Apache Airflow для управления задачами и настройку Prometheus и Grafana для мониторинга.

Шаг 1: Подготовка Docker-контейнера

Создайте файл Dockerfile в корневой директории проекта. Этот файл содержит инструкции по сборке контейнера.

Dockerfile

```
# Базовый образ Python
FROM python:3.8

# Копируем файл с зависимостями и устанавливаем их
COPY requirements.txt .
RUN pip install -r requirements.txt
```

```
# Копируем файлы приложения
COPY . /app
WORKDIR /app

# Запускаем основной скрипт приложения
CMD ["python", "app.py"]
```

Создайте файл requirements.txt для указания всех необходимых библиотек:

plaintext

```
pandas
numpy
statsmodels
scikit-learn
prometheus-client
airflow
```

Сборка Docker-контейнера:

bash

```
docker build -t sales_forecasting_app .
```

Запуск Docker-контейнера:

bash

```
docker run -p 5000:5000 sales_forecasting_app
```

Теперь приложение будет доступно по адресу **http://localhost:5000**.

Шаг 2: Установка и настройка Apache Airflow

Установите Apache Airflow (если не установлен):

bash

```
pip install apache-airflow
```

Создайте DAG (Directed Acyclic Graph) для автоматизации процесса обновления и обработки данных. DAG должен включать задачи по:

- Извлечению данных.
- Предобработке данных.
- Переобучению модели.
- Генерации прогноза.

Запустите Airflow:

bash

```
airflow standalone
```

Откройте интерфейс Airflow в браузере по адресу http://localhost:8080. Запустите DAG и убедитесь, что все задачи выполняются корректно.

Шаг 3: Настройка Prometheus и Grafana

1. **Установите Prometheus и Grafana**:
   - Инструкции по установке Prometheus: Prometheus Installation Guide
   - Инструкции по установке Grafana: Grafana Installation Guide
2. **Настройка Prometheus**:
   - Добавьте Prometheus в качестве системы мониторинга, установив файл конфигурации prometheus.yml с нужными настройками.

Пример файла prometheus.yml:

yaml

```
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'sales_forecasting_app'
    static_configs:
      - targets: ['localhost:8000']  # Порт, на котором экспонируются
метрики
```

3. **Настройка Grafana**:
   - Запустите Grafana и добавьте Prometheus как источник данных.
   - Создайте дашборд для визуализации метрик производительности, таких как MAE и RMSE.

Запуск сервера Prometheus для сбора метрик:

bash

```
prometheus --config.file=prometheus.yml
```

4. **Запуск Grafana**:
   - После установки Grafana запустите её и откройте веб-интерфейс по адресу http://localhost:3000.

После завершения всех этапов развертывания, ваша система будет готова к работе. Приложение будет автоматически обновлять данные, переобучать модель и генерировать прогнозы, а Prometheus и Grafana помогут отслеживать производительность модели в реальном времени.Автоматизация и управление задачами (Airflow)

Для автоматизации процессов в системе используется **Apache Airflow**, который позволяет управлять задачами по обновлению данных, предобработке, переобучению модели и генерации прогноза. Airflow использует DAG (Directed Acyclic Graph) для определения последовательности задач.

Шаги по настройке Airflow

Установка Airflow:
Если Airflow еще не установлен, выполните команду:

bash

```
pip install apache-airflow
```

Создание DAG для автоматизации процессов
Создайте новый Python файл, например `sales_forecasting_dag.py`, в папке `dags` Airflow. В этом файле создайте DAG, который будет управлять процессами, связанными с обновлением данных, предобработкой и прогнозированием.

Пример структуры DAG:

python

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from datetime import datetime, timedelta

# Определение функций для задач пайплайна
def extract_data():
    # Код для извлечения и обновления данных
    pass

def preprocess_data():
    # Код для предобработки данных
    pass

def retrain_model():
    # Код для переобучения модели на новых данных
    pass

def generate_forecast():
    # Код для генерации прогноза на основе обновленной модели
    pass

# Параметры DAG
default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': datetime(2023, 1, 1),
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
}

# Создание DAG
dag = DAG(
    'sales_forecasting_pipeline',
    default_args=default_args,
    description='Pipeline for automated sales forecasting',
    schedule_interval=timedelta(days=1),
)

# Определение задач
t1 = PythonOperator(
    task_id='extract_data',
    python_callable=extract_data,
    dag=dag,
)

t2 = PythonOperator(
```

```
    task_id='preprocess_data',
    python_callable=preprocess_data,
    dag=dag,
)

t3 = PythonOperator(
    task_id='retrain_model',
    python_callable=retrain_model,
    dag=dag,
)

t4 = PythonOperator(
    task_id='generate_forecast',
    python_callable=generate_forecast,
    dag=dag,
)

# Определение последовательности выполнения задач
t1 >> t2 >> t3 >> t4
```

Описание задач в DAG:

- `extract_data`: Эта задача отвечает за извлечение данных из источника (например, базы данных или файла) и их обновление для последующей обработки.
- `preprocess_data`: Включает все этапы предобработки данных, такие как очистка, фильтрация и подготовка данных для модели.
- `retrain_model`: Переобучает модель на обновленных данных, чтобы прогнозы оставались актуальными.
- `generate_forecast`: Создает прогноз на заданный период (например, на 30 дней), используя обновленную модель.

Запуск Airflow:
Запустите Airflow, чтобы DAG начал выполняться по расписанию:

bash

```
airflow standalone
```

Проверка и запуск DAG в веб-интерфейсе:

- Откройте Airflow UI по адресу `http://localhost:8080`.
- В разделе DAGs найдите `sales_forecasting_pipeline` и вручную запустите его или дождитесь автоматического запуска согласно расписанию.

Мониторинг задач в Airflow:

- Airflow позволяет отслеживать статус выполнения каждой задачи, что помогает вовремя обнаружить ошибки и устранить их.
- Можно задать параметры ретрая для задач, чтобы при ошибке они автоматически повторялись.

Мониторинг и оповещения

Для обеспечения стабильной работы системы и своевременного реагирования на изменения в производительности используются **Prometheus** для мониторинга метрик и **Grafana** для визуализации. **Alertmanager** интегрируется с Prometheus и позволяет отправлять уведомления при достижении критических порогов.

Шаги по настройке мониторинга и оповещений

1. Установка Prometheus и Grafana

1. **Установка Prometheus**:
   - Загрузите и установите Prometheus, следуя документации Prometheus.
   - Создайте файл конфигурации `prometheus.yml` для настройки источников метрик.
2. **Установка Grafana**:
   - Загрузите и установите Grafana, следуя документации Grafana.
   - После установки запустите Grafana и откройте веб-интерфейс по адресу `http://localhost:3000`.

2. Настройка Prometheus для сбора метрик

В файле `prometheus.yml` добавьте конфигурацию для сбора метрик из вашего приложения, например:

yaml

```yaml
global:
  scrape_interval: 15s

scrape_configs:
  - job_name: 'sales_forecasting_app'
    static_configs:
      - targets: ['localhost:8000']  # Порт, на котором экспонируются метрики
```

Запустите Prometheus с новой конфигурацией:

bash

```
prometheus --config.file=prometheus.yml
```

Настройте приложение для экспонирования метрик с помощью библиотеки `prometheus-client`. Например:

python

```
from prometheus_client import start_http_server, Gauge
import time

# Определяем метрики
mae_metric = Gauge('sales_forecast_mae', 'Mean Absolute Error of Sales Forecast')
rmse_metric = Gauge('sales_forecast_rmse', 'Root Mean Square Error of Sales Forecast')

def update_metrics(mae, rmse):
    mae_metric.set(mae)
    rmse_metric.set(rmse)

# Запуск HTTP-сервера для экспонирования метрик на порту 8000
if __name__ == '__main__':
    start_http_server(8000)
    while True:
        mae, rmse = 150000, 200000  # Пример значений метрик
        update_metrics(mae, rmse)
        time.sleep(3600)  # Обновление метрик каждый час
```

3. Настройка Grafana для визуализации метрик

1. **Добавьте Prometheus как источник данных**:
   - В интерфейсе Grafana перейдите в `Configuration > Data Sources`.
   - Выберите Prometheus и введите URL (например, `http://localhost:9090`), затем сохраните.
2. **Создайте дашборд**:
   - Создайте новый дашборд в Grafana и добавьте панели (panels) для метрик, таких как MAE и RMSE.
   - Настройте графики и диаграммы для наглядного отображения производительности модели и других метрик.

4. Настройка Alertmanager для оповещений

1. Установите и настройте Alertmanager, чтобы отправлять уведомления, когда метрики достигают заданных порогов.

Добавьте правила оповещений в конфигурацию Prometheus (`prometheus.yml`):

yaml

```yaml
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          - 'localhost:9093'  # Адрес Alertmanager

rule_files:
  - 'alert_rules.yml'  # Файл с правилами оповещений
```

Создайте файл `alert_rules.yml` с определением правил. Например:

yaml

```yaml
groups:
- name: sales_forecast_alerts
  rules:
  - alert: HighMAE
    expr: sales_forecast_mae > 200000  # Порог для метрики MAE
    for: 5m
    labels:
      severity: warning
    annotations:
      summary: "Высокая MAE для прогнозов продаж"
      description: "Средняя абсолютная ошибка превысила 200000 в
течение 5 минут."

  - alert: HighRMSE
    expr: sales_forecast_rmse > 250000  # Порог для метрики RMSE
    for: 5m
    labels:
      severity: critical
    annotations:
      summary: "Высокая RMSE для прогнозов продаж"
      description: "Среднеквадратичная ошибка превысила 250000 в течение
```

```
5 минут."
```

2. **Запуск Alertmanager**:
   - Запустите Alertmanager и настройте его для отправки оповещений по email или в мессенджеры, такие как Slack или Telegram.

Результат

После настройки Prometheus и Grafana вы сможете контролировать точность и производительность модели в реальном времени. При превышении допустимых порогов метрик (например, MAE или RMSE) система отправит оповещение через Alertmanager, позволяя оперативно реагировать на изменения.

Использование и эксплуатация

После развертывания и настройки системы вы можете начать использовать её для регулярного прогнозирования спроса и анализа точности модели. Ниже приведены основные инструкции по работе с системой и использованию её функционала.

---

1. Запуск DAG в Airflow

1. **Откройте Airflow**:
   - Перейдите в веб-интерфейс Airflow по адресу `http://localhost:8080`.
2. **Запуск DAG**:
   - Найдите DAG `sales_forecasting_pipeline` в списке и включите его, чтобы DAG выполнялся по расписанию.
   - Вы можете запустить DAG вручную, нажав на иконку "play" рядом с его названием.
3. **Мониторинг задач**:
   - Airflow отображает статус выполнения каждой задачи в DAG, что позволяет вам отслеживать успешность выполнения этапов (извлечение данных, предобработка, переобучение, прогноз).
   - В случае ошибки Airflow сохранит логи, что поможет быстро выявить и устранить проблему.

---

2. Просмотр прогнозов

1. **Генерация прогноза**:

- Прогнозы автоматически обновляются DAG’ом в Airflow после успешного выполнения всех задач.

2. **Доступ к прогнозам**:
- Прогнозы на следующие 30 дней сохраняются в файлах или базе данных (в зависимости от настроек).
- Вы можете настроить экспорт прогнозов в удобный формат, например CSV или Excel, для дальнейшего анализа или интеграции с другими системами.

---

3. Мониторинг точности модели

1. **Проверка метрик в Grafana**:
- Откройте Grafana по адресу `http://localhost:3000`.
- В Grafana создайте дашборд с панелями для отслеживания метрик точности (MAE, RMSE), а также любых других параметров, таких как количество заказов и доход.
2. **Анализ изменений точности**:
- Постоянно отслеживайте MAE и RMSE на графиках, чтобы выявить возможные ухудшения точности.
- Регулярный мониторинг позволит оперативно реагировать на изменения в данных и производительности модели.

---

4. Оповещения и реагирование

1. **Получение оповещений**:
- Система автоматически отправляет оповещения через Alertmanager при достижении порогов для метрик (например, при высокой MAE или RMSE).
- Оповещения можно получать по email, в мессенджерах или через Slack в зависимости от настроек Alertmanager.
2. **Реагирование на оповещения**:
- При получении оповещения о высокой ошибке модели зайдите в Airflow и проверьте статус задач DAG. Ошибка может быть связана с данными или параметрами модели.
- Проверьте логи в Airflow и Prometheus, чтобы выяснить источник проблемы и предпринять корректирующие меры, такие как изменение гиперпараметров или проверка данных.

Полезные команды

Перезапуск контейнера Docker (в случае обновления кода или зависимостей):

bash

```
docker restart sales_forecasting_app
```

Перезапуск Prometheus и Alertmanager (для применения новых правил или конфигураций):

bash

```
prometheus --config.file=prometheus.yml
alertmanager --config.file=alertmanager.yml
```

Запуск Airflow DAG вручную:

- В интерфейсе Airflow нажмите на DAG и выберите "Trigger DAG" для ручного запуска.

Результат эксплуатации

После настройки всех компонентов вы сможете:

- Автоматически генерировать прогнозы спроса и получать обновления каждые 24 часа.
- Получать уведомления при ухудшении метрик модели для быстрой реакции.
- Визуализировать метрики точности и производительности модели в реальном времени.
- Использовать прогнозы для планирования запасов и улучшения маркетинговых стратегий.

Обновление и поддержка системы

Чтобы поддерживать систему в актуальном состоянии и гарантировать её стабильную работу, предусмотрены автоматическое переобучение модели, обновление данных, мониторинг метрик и обработка ошибок. Вот основные шаги и рекомендации для поддержки системы.

---

1. Переобучение модели

1. **Регулярное переобучение**:
   - Модель автоматически переобучается по расписанию, заданному в Airflow DAG (например, ежедневно или еженедельно).
   - Переобучение происходит на актуальных данных, что позволяет модели адаптироваться к изменениям в спросе и поддерживать высокую точность.
2. **Проверка качества модели после переобучения**:
   - После каждого переобучения отслеживайте метрики (MAE, RMSE) в Grafana.
   - Если качество модели ухудшилось, проверьте, не вызвано ли это аномальными данными или изменением поведения спроса.

2. Обновление кода и зависимостей

1. **Обновление кода приложения**:
   - При необходимости обновления кода (например, улучшения алгоритма или добавления новых признаков), внесите изменения и пересоберите Docker-контейнер.

Для пересборки и перезапуска контейнера используйте команды:

bash

```
docker build -t sales_forecasting_app .
docker restart sales_forecasting_app
```

2. **Обновление зависимостей**:
   - Если вам нужно обновить зависимости (например, библиотеки Python), обновите файл `requirements.txt` и пересоберите Docker-контейнер.
   - Убедитесь, что все новые зависимости протестированы перед применением в производственной среде.

3. Мониторинг и обработка ошибок

1. **Постоянный мониторинг метрик**:
   - Продолжайте отслеживать метрики модели в Grafana и проверяйте оповещения от Alertmanager.
   - Оповещения помогают своевременно обнаружить проблемы с производительностью модели или точностью прогноза.

2. **Анализ ошибок**:
   - Если метрики ухудшились, проверьте логи в Airflow и Prometheus, чтобы определить причину.
   - Возможные причины ухудшения производительности: изменения в данных, новые тренды или неполадки с системой.
3. **Ретрай задач в Airflow**:
   - В случае ошибок в задачах DAG, Airflow автоматически выполнит ретрай по заданному расписанию.
   - Проверьте логи задач в Airflow для детальной информации об ошибках и внесите корректировки при необходимости.

4. Обратная связь и улучшение модели

1. **Получение обратной связи от пользователей**:
   - Регулярно обсуждайте результаты прогнозов с ключевыми пользователями (например, отделом продаж или маркетинга), чтобы убедиться, что модель соответствует их ожиданиям.
   - Полученная обратная связь может помочь в улучшении модели и добавлении новых признаков.
2. **Добавление новых признаков**:
   - Если модель можно улучшить, добавив новые признаки (например, макроэкономические факторы или данные о погоде), интегрируйте их в процесс предобработки и переобучите модель.

Полезные рекомендации

1. **Резервное копирование данных**: Регулярно создавайте резервные копии данных и модели, чтобы избежать потерь информации в случае непредвиденных сбоев.
2. **Оптимизация параметров**: Периодически проверяйте и оптимизируйте гиперпараметры модели, так как изменения в данных могут повлиять на оптимальные значения.
3. **Регулярное тестирование**: Проводите регулярное тестирование системы, особенно после обновления кода или зависимостей, чтобы убедиться в стабильной работе всех компонентов.

Заключение

Эти шаги помогут поддерживать систему в актуальном состоянии и гарантировать её стабильную работу. Постоянное переобучение и мониторинг, а также своевременная обратная связь от пользователей обеспечат высокую точность прогнозов и удовлетворенность бизнес-команды.

Эта документация завершает проект и включает все ключевые аспекты использования, эксплуатации и поддержки системы прогнозирования спроса.

Итог

Данная документация описывает создание, развертывание, использование и поддержку системы прогнозирования спроса для e-commerce. Система разработана для автоматического обновления и предсказания объемов продаж на основе исторических данных, с учетом сезонных и трендовых факторов. Это позволяет бизнесу более эффективно управлять запасами и адаптировать стратегию продаж под ожидаемые изменения спроса.

Основные преимущества системы:

- **Автоматизация**: Все этапы процесса прогнозирования, включая обновление данных, предобработку, переобучение модели и генерацию прогнозов, выполняются автоматически с помощью Apache Airflow.
- **Точность и адаптивность**: Система использует модель SARIMA для учета сезонных колебаний и регулярно переобучается на новых данных, обеспечивая актуальность прогноза.
- **Мониторинг и реагирование**: Метрики точности прогнозов отслеживаются в режиме реального времени с использованием Prometheus и Grafana, а система оповещений Alertmanager информирует о любых изменениях в производительности.
- **Масштабируемость и гибкость**: Контейнеризация с помощью Docker и использование распространенных инструментов мониторинга позволяют легко развертывать и масштабировать систему в разных средах.

Система является эффективным инструментом для прогнозирования спроса, помогая бизнесу принимать обоснованные решения, снижать потери от излишков или нехватки товаров и повышать удовлетворенность клиентов.

Эта документация будет полезна для понимания архитектуры системы, её развертывания и эксплуатационных процедур, а также для обеспечения её стабильной работы в долгосрочной перспективе.