**Московский университет имени С.Ю. Витте**

Презентация на тему “Этапы разработки программного обеспечения”

Работу выполнили: Шавыкин А.,

Гайда К.,

Искалиев Р.

# **СОДЕРЖАНИЕ**

[**СОДЕРЖАНИЕ** 2](#_Toc180081009)

[**Анализ требований** 3](#_Toc180081010)

[**1.1.** **Сбор требований** 3](#_Toc180081011)

[**1.2.** **Определение целей проекта** 5](#_Toc180081012)

[**1.3.** **Оценка рисков и возможностей** 5](#_Toc180081013)

[**Проектирование системы** 6](#_Toc180081014)

[**2.1. Архитектурное проектирование** 6](#_Toc180081015)

[**2.2. Создание технической документации** 6](#_Toc180081016)

[**2.3. Прототипирование** 7](#_Toc180081017)

# **Анализ требований**

* 1. **Сбор требований**

*Сбор требований* — это важный этап в разработке программного обеспечения для прогнозирования погоды, который включает в себя определение того, что именно должно быть реализовано в системе. Это может быть разделено на функциональные и нефункциональные требования.

**Функциональные требования**

Прогнозирование погоды:

Программное обеспечение должно предоставлять точные прогнозы погоды на различные временные интервалы (например, на 1, 3, 7 и 14 дней).

Возможность отображения различных метеорологических параметров, таких как температура, влажность, скорость ветра, осадки и атмосферное давление.

Сбор данных:

Система должна интегрироваться с различными источниками данных, такими как метеостанции, спутники и другие метеорологические службы, для получения актуальной информации.

Визуализация данных:

Программное обеспечение должно обеспечивать визуализацию данных в виде графиков, диаграмм и интерактивных карт, чтобы пользователи могли легко воспринимать информацию.

Уведомления и оповещения:

Возможность отправки уведомлений пользователям о значительных изменениях в погодных условиях, таких как штормы, сильные дожди или другие опасные явления.

Персонализация:

Пользователи должны иметь возможность настраивать интерфейс и выбирать интересующие их регионы и параметры прогноза.

Исторические данные:

Доступ к архивным метеорологическим данным для анализа и сравнения с текущими прогнозами.

**Нефункциональные требования**

Производительность:

Программное обеспечение должно обеспечивать быструю обработку данных и генерацию прогнозов, чтобы пользователи могли получать актуальную информацию без задержек.

Безопасность:

Необходимо обеспечить защиту данных пользователей и метеорологических данных от несанкционированного доступа и утечек. Это включает в себя шифрование данных и аутентификацию пользователей.

Совместимость:

Программное обеспечение должно быть совместимо с различными операционными системами и устройствами, включая мобильные телефоны, планшеты и настольные компьютеры.

Масштабируемость:

Система должна быть способна обрабатывать увеличивающееся количество пользователей и данных без потери производительности.

Удобство использования:

Интерфейс должен быть интуитивно понятным и простым в навигации, чтобы пользователи могли легко находить необходимую информацию.

Поддержка и обновления:

Необходимо обеспечить регулярные обновления программного обеспечения для улучшения функциональности и исправления ошибок, а также предоставление технической поддержки пользователям.

* 1. **Определение целей проекта**

**Определение ключевых целей и задач:**

Создание надежного и точного программного обеспечения для прогнозирования погоды, которое будет удовлетворять потребности пользователей.

Обеспечение доступности актуальной метеорологической информации в удобном и понятном формате.

Разработка системы, способной интегрироваться с различными источниками данных и обеспечивать высокую производительность.

Повышение уровня информированности пользователей о погодных условиях через уведомления и персонализированные настройки.

* 1. **Оценка рисков и возможностей**

**Оценка возможных рисков:**

Технические риски: Возможные проблемы с интеграцией данных из различных источников, ошибки в алгоритмах прогнозирования.

Организационные риски: Недостаток квалифицированных специалистов для разработки и поддержки системы.

Финансовые риски: Превышение бюджета на разработку и поддержку программного обеспечения.

Выявление возможностей для минимизации рисков:

Проведение тщательного тестирования и валидации системы на ранних этапах разработки.

Обучение и привлечение квалифицированных специалистов для работы над проектом.

Оптимизация бюджета и планирование ресурсов на основе анализа рисков.

Формирование стратегии для преодоления возможных препятствий:

Разработка плана управления рисками, который включает в себя регулярный мониторинг и оценку рисков на протяжении всего проекта.

Создание резервного плана на случай возникновения критических проблем, включая возможность привлечения дополнительных ресурсов или изменения подхода к разработке.

**Проектирование системы**

**2.1. Архитектурное проектирование**

Разработка архитектуры программного обеспечения:

Выбор технологического стека, включая языки программирования (например, Python, JavaScript), фреймворки (например, Django, React) и базы данных (например, PostgreSQL, MongoDB).

Определение структуры приложения, включая клиентскую и серверную части, а также взаимодействие с внешними API для получения метеорологических данных.

Определение компонентов системы, таких как модули для обработки данных, визуализации и уведомлений.

**2.2. Создание технической документации**

Формирование технической документации:

Описание архитектуры системы, включая схемы компонентов и их взаимодействие.

Определение интерфейсов, API, форматов данных и протоколов передачи данных, которые будут использоваться для интеграции с внешними источниками и клиентскими приложениями.

**2.3. Прототипирование**

Разработка прототипов или макетов интерфейсов:

Создание интерактивных прототипов пользовательского интерфейса для визуализации будущей системы, что позволит получить обратную связь от пользователей на ранних этапах разработки.

Использование инструментов для прототипирования (например, Figma, Adobe XD) для создания макетов, которые демонстрируют основные функции и взаимодействия в приложении.