**Лабораторная работа №7. Презентация проекта и составление документации.**

**Цель работы:**

Научиться эффективно представлять результаты разработки программного продукта, а также создавать качественную техническую документацию.

**Теоретическая часть:**

**Подготовка презентации проекта**

1. **Формат презентации**:
   * Презентация должна быть выполнена в формате PDF или PowerPoint.
   * Длительность выступления — 5–7 минут.
   * Количество слайдов — 8–10.

**Структура презентации:**

**Слайд 1:** Титульный слайд

* Название проекта.
* ФИО исполнителя и необходимые данные. (ФИО студента, номер группы).

**Слайд 2**: Общее описание проекта

* Основные задачи проекта.
* Целевая аудитория.
* Краткое описание функционала.
* Сравнение с аналогами.

**Слайд 3:** Архитектура системы

* Схема микросервисной архитектуры.
* Перечисление микросервисов (например: каталог товаров, корзина покупок, система заказов, уведомления).
* Объяснение взаимодействия между сервисами через RabbitMQ и Kafka.

**Слайд 4:** Базовые технологии

* Используемые технологии и инструменты (.NET, RabbitMQ, Kafka, Entity Framework, SQL/NoSQL).
* Описание основных библиотек и фреймворков.

**Слайд 5**: Пример реализации

* Показать демонстрацию работы одного из микросервисов (например, создание заказа, отправка уведомления).
* Кодовые примеры для ключевых функций.

**Слайд 6**: Тестирование и мониторинг

* Методы тестирования (юнит-тесты, интеграционные тесты).
* Инструменты мониторинга производительности (например, использование метрик Kafka).

**Слайд 7:** Результаты работы

* Основные достижения проекта.
* Примеры решенных проблем.

**Слайд 8:** Заключение и планы по доработке (план развития)

* Оценка успешности проекта.
* Возможные направления развития (например, добавление новых функций, оптимизация производительности).

**Требования к презентации:**

* Четкая структура и логика.
* Минимум текста на слайдах, максимум визуализации (схемы, диаграммы, скриншоты).
* Нумерация слайдов.
* Читаемый текст (минимум 18 пт., контрастные цвета (черные буквы на белом фоне))

**Техническая документация**

Техническая документация — это набор материалов, описывающих архитектуру, функциональность, взаимодействие компонентов и способы использования программного продукта. Она является важным инструментом для разработчиков, тестировщиков, администраторов и пользователей, обеспечивая понимание системы и упрощая процесс поддержки, масштабирования и доработки.

**Виды технической документации**

**2.1. Архитектурная документация**

* Описывает структуру и организацию системы.
* Включает:
  + Диаграммы архитектуры (UML, ER-диаграммы).
  + Описание микросервисов или модулей.
  + Схемы взаимодействия между компонентами.
  + Требования к базам данных и внешним системам.

**2.2. Разработка и кодирование**

* Документирует реализацию конкретных функций.
* Включает:
  + Примеры кода.
  + Описание API и интерфейсов.
  + Пояснения по работе с библиотеками и фреймворками.

**2.3. Инструкции по установке и настройке**

* Объясняет, как установить и настроить систему.
* Включает:
  + Требования к аппаратному и программному обеспечению.
  + Шаги по развертыванию приложения.
  + Настройку зависимостей (например, баз данных, брокеров сообщений).

**2.4. Инструкции по эксплуатации**

* Предназначены для конечных пользователей или администраторов.
* Включают:
  + Руководства пользователя.
  + Инструкции по управлению системой.
  + Процедуры восстановления после сбоев.

**2.5. Тестирование**

* Описывает методы проверки корректности работы системы.
* Включает:
  + Юнит-тесты.
  + Интеграционные тесты.
  + Методы нагрузочного тестирования.

**2.6. Мониторинг и логирование**

* Описывает способы отслеживания производительности и диагностики проблем.
* Включает:
  + Конфигурацию мониторинга.
  + Анализ логов.
  + Алгоритмы обработки ошибок.

**3. Способы написания технической документации**

**3.1. Структурированный подход**

Документация должна быть четко структурирована для удобства восприятия. Используйте следующие элементы:

* **Заголовки и подзаголовки**: Для разделения информации на логические блоки.
* **Списки**: Для представления перечислений или шагов.
* **Таблицы**: Для сравнения параметров или характеристик.
* **Схемы и диаграммы**: Для визуализации сложных процессов.

**3.2. Использование шаблонов**

Шаблоны помогают стандартизировать формат документации. Например:

* **Шаблон архитектурной документации**:
  + Цель проекта.
  + Архитектурная схема.
  + Описание ключевых компонентов.
* **Шаблон инструкции по установке**:
  + Системные требования.
  + Пошаговая инструкция.
  + Частые проблемы и их решения.

**3.3. Поддержание актуальности**

Документация должна регулярно обновляться:

* После внедрения новых функций.
* При изменении архитектуры.
* При исправлении ошибок.

**4. Основные принципы написания качественной документации (!)**

**4.1. Ясность и точность**

Используйте простой язык, избегайте сложных терминов без объяснений. Убедитесь, что каждое утверждение имеет конкретную цель.

**4.2. Комплетность**

Укажите все необходимые детали, чтобы пользователь или разработчик могли выполнить задачу без дополнительных вопросов.

**4.3. Актуальность**

Регулярно проверяйте документацию на соответствие текущему состоянию системы.

**4.4. Доступность**

Храните документацию в удобном месте (например, в репозитории GitHub или в виде PDF-файла), чтобы она была доступна всем заинтересованным сторонам.

**5. Пример структуры документации**

|  |
| --- |
| **1. Введение**   * Цели проекта. * Краткое описание функциональности.   **2. Архитектура**   * Общая схема системы. * Описание микросервисов. * Взаимодействие между сервисами.   **3. Реализация**   * Примеры кода. * Описание ключевых функций.   **4. Установка и настройка**   * Системные требования. * Инструкция по развертыванию.   **5. Тестирование**   * Методы тестирования. * Примеры тестов.   **6. Мониторинг**   * Сбор метрик. * Логирование.   **7. Заключение**   * Оценка успешности проекта. * Возможные направления развития. |

**Примеры инструментов для создания документации**

**Офисные приложения**

* **Microsoft Office**
* **Google Docs**

**Специализированные инструменты**

* **Confluence**: Корпоративное решение для хранения и управления документацией.
* **Notion**: Многофункциональный инструмент для создания и организации контента.
* **Read the Docs**: Хостинг для документации на основе Sphinx.

**Инструменты для создания диаграмм**

* **Lucidchart**: Создание UML-диаграмм и схем.
* **Draw.io**: Простой инструмент для рисования схем.
* **Visio**: Профессиональный инструмент для создания технических схем.

**Автоматизация создания документации**

* **Swagger/OpenAPI**
* **Doxygen**: Для генерации документации из комментариев в коде.
* **Markdown**: Для создания простых текстовых документов.
* **PlantUML**: Для создания UML-диаграмм.