Лабораторная работа №2

**Организационные меры по внедрению программного обеспечения**

**Модель жизненого цикла ПП «Спиральная модель»**

**Задание**

**План работ по созданию программного продукта (ПП):**

1. **Определение требований:**

Провести встречи с заказчиком для выяснения функциональных и технических требований.

Документировать полученные требования.

1. **Анализ рынка и конкурентов:**

Исследовать существующие аналоги ПП.

Определить особенности, преимущества и недостатки конкурентов.

1. **Проектирование:**

Разработать архитектуру ПП.

Создать дизайн интерфейса пользователя.

Определить технологический стек и инструменты разработки.

1. **Разработка:**

Написать код основной функциональности.

Провести тестирование промежуточных версий.

1. **Тестирование:**

Провести функциональное тестирование для проверки соответствия требованиям.

Выявить и устранить ошибки.

1. **Доработка и оптимизация:**

Улучшить производительность и стабильность программы.

Внедрить корректировки и дополнения, если необходимо.

1. **Документирование:**

Подготовить техническую документацию по ПП.

Создать руководство пользователя.

1. **Тестирование завершенной версии:**

Провести финальное тестирование перед выпуском.

1. **Релиз и внедрение:**

Подготовить программу к выпуску.

Разработать стратегию маркетинга и внедрения.

1. **Поддержка и обновление:**

Обеспечить техническую поддержку пользователям.

Регулярно выпускать обновления для улучшения и дополнения функционала.

Контрольные задания:

1. Перечислите модели ЖЦПО

Модель жизненного цикла программного обеспечения (ЖЦПО) представляет собой процесс разработки и управления программным продуктом от начальной концепции до завершения. Существует несколько моделей ЖЦПО, каждая из которых описывает этот процесс по-своему. Вот некоторые из наиболее распространенных моделей ЖЦПО:

**Водопадная модель (Waterfall Model):** Линейная модель, где каждая фаза разработки следует за предыдущей, начиная с анализа и проходя через проектирование, реализацию, тестирование и поддержку.

**Итеративная модель (Iterative Model):** Разработка программы выполняется в циклах, называемых итерациями. Каждая итерация добавляет новый функционал и проходит через все этапы разработки.

**Инкрементальная модель (Incremental Model):** Программа разрабатывается как набор небольших инкрементов или модулей, которые добавляются по мере развития проекта.

**Спиральная модель (Spiral Model):** Модель, объединяющая элементы водопадной модели и итеративной разработки. Процесс построен на циклах, с каждым циклом добавляются новые функции.

**Модель быстрой разработки приложений (RAD - Rapid Application Development):** Уделяет большое внимание быстрому прототипированию и вовлечению заказчика в процесс разработки.

**Модель спецификаций на языке формальных методов (Formal Methods Specification Model):** Основывается на математических методах для формальной верификации и спецификации программ.

**DevOps и CI/CD (Continuous Integration/Continuous Deployment):** Эти модели сосредотачиваются на автоматизации процесса разработки, тестирования и развертывания ПО, обеспечивая непрерывное улучшение и доставку.

Каждая из этих моделей имеет свои преимущества и недостатки, и выбор модели зависит от конкретных требований и характеристик проекта.нтрольные вопросы:

1. Дайте сравнительную характеристику каждому виду.

Стравнительная характеристика различных видов моделей жизненного цикла программного обеспечения (ЖЦПО):

**Водопадная модель (Waterfall Model):**

**Особенности:** Линейный подход, последовательные фазы.

**Преимущества:** Простота управления, хорошо подходит для небольших проектов с четкими требованиями.

**Недостатки:** Не подходит для проектов с изменяющимися требованиями, риски высоки при отсутствии реальных результатов до завершения.

**Итеративная модель (Iterative Model):**

**Особенности:** Разработка в итерациях, повторное проектирование и улучшение на каждой итерации.

**Преимущества**: Возможность быстрого реагирования на изменения требований, улучшение качества на каждой итерации.

**Недостатки**: Может потребовать дополнительного времени и ресурсов.

**Инкрементальная модель (Incremental Model):**

**Особенности**: Разработка по частям, поочередное добавление инкрементов.

**Преимущества**: Возможность быстрого выпуска базового функционала, удобно для расширения.

**Недостатки**: Требует внимательного планирования и управления версиями.

**Спиральная модель (Spiral Model):**

**Особенности**: Комбинирует элементы водопада и итераций с учетом рисков.

**Преимущества**: Учитывает риски, позволяет быстро реагировать на изменения.

**Недостатки**: Сложность управления, может потребовать больше времени.

**Модель быстрой разработки приложений (RAD - Rapid Application Development):**

**Особенности**: Основана на быстром прототипировании и вовлечении заказчика.

**Преимущества**: Сокращение времени разработки, высокая степень участия заказчика.

**Недостатки**: Не всегда подходит для сложных проектов, требует активного участия заказчика.

**Модель спецификаций на языке формальных методов (Formal Methods Specification Model):**

**Особенности**: Использует математические методы для спецификации и верификации.

**Преимущества**: Высокая точность и надежность.

**Недостатки**: Сложность в реализации, не всегда подходит для всех типов ПО.

**DevOps и CI/CD (Continuous Integration/Continuous Deployment):**

**Особенности**: Автоматизированный процесс интеграции, тестирования и развертывания.

**Преимущества**: Быстрая поставка, непрерывное улучшение.

**Недостатки**: Требует инфраструктуры и автоматизации.

Техническое задание (ТЗ) для разработки программного обеспечения (ПО) является важным документом, который описывает требования и спецификации проекта. Вот общий шаблон технического задания для разработки ПО:

**Техническое задание на разработку нашего программного обеспечения**

1. Введение

1.1 Цель проекта

Описать основную цель создания программного продукта и ожидаемый результат от его разработки.

1.2 Контекст проекта

Указать общую информацию о контексте, в котором будет использоваться ПО.

2. Требования к функциональности

2.1 Основные функции

Перечислить основные функции и возможности, которые должны быть реализованы в ПО.

2.2 Дополнительные функции

Описать дополнительные функции, которые могут быть включены в ПО (по желанию).

3. Требования к интерфейсу

3.1 Графический интерфейс

Описать требования к пользовательскому интерфейсу, включая дизайн, цветовую схему, компоненты и элементы управления.

3.2 Интерфейсы с другими системами

Если ПО должно взаимодействовать с другими системами или сторонними сервисами, описать эти интерфейсы.

4. Требования к производительности

4.1 Производительность

Установить требования к скорости работы, времени отклика и другим параметрам производительности.

4.2 Нагрузочное тестирование (по желанию)

Если необходимо проводить нагрузочное тестирование, указать требования к нему.

5. Требования к безопасности

5.1 Защита данных

5.2 Защита от угроз

Указать требования к защите от внутренних и внешних угроз безопасности.

6. Требования к масштабируемости

Описать требования к масштабируемости ПО, если оно должно быть способным масштабироваться под рост нагрузки.

7. Требования к поддержке и обслуживанию

Указать требования к поддержке, обновлениям и обслуживанию ПО после его разработки и внедрения.

8. Требования к документации

Указать, какие документы должны быть предоставлены в рамках проекта, включая техническую документацию, руководства пользователя и прочие.

9. Требования к срокам и бюджету

9.1 Сроки

Установить желаемые сроки завершения проекта и важные этапы.

9.2 Бюджет

Определить бюджет проекта, включая расходы на разработку, оборудование и другие затраты.

10. Риски и ограничения

10.1 Риски проекта

Идентифицируйте возможные риски, которые могут повлиять на проект, и опишите меры по их снижению.

10.2 Ограничения

Указать ограничения, которые могут ограничивать процесс разработки ПО.

11. Заключение

11.1 Согласование

Указать ответственных за утверждение и согласование данного технического задания.

11.2 Подписи

Подписать документ и указать дату утверждения.

**Техническое задание служит основой для разработки ПО и является ключевым документом, определяющим требования и ожидания заказчика.**

**Контрольные задания:**

1. Назовите возможные цели и задачи внедрения ПО.

Внедрение программного обеспечения (ПО) может иметь разнообразные цели и задачи в зависимости от конкретного контекста и потребностей организации или пользователя. Ниже приведены некоторые возможные цели и задачи внедрения ПО:

**Цели:**

Автоматизация бизнес-процессов: Улучшение эффективности и производительности организации через автоматизацию рутинных задач и процессов.

Улучшение качества продукта или услуги: Повышение качества и надежности продукции или услуги с помощью ПО для контроля и управления производством или предоставлением услуг.

Улучшение взаимодействия с клиентами: Создание приложений для улучшения взаимодействия с клиентами, предоставления онлайн-сервисов и усовершенствования клиентского опыта.

Улучшение безопасности: Защита систем и данных организации от киберугроз и угроз безопасности через внедрение специализированных ПО и мер безопасности.

Сокращение затрат: Оптимизация бизнес-процессов и снижение операционных расходов с использованием эффективного ПО.

**Задачи:**

Выбор ПО: Идентификация подходящего ПО с учетом потребностей организации, бюджета и функциональных требований.

Установка и настройка: Установка программного обеспечения на компьютеры и серверы, а также его настройка в соответствии с требованиями.

Обучение персонала: Подготовка сотрудников к использованию нового ПО путем обучения и обеспечения необходимых ресурсов и материалов.

Тестирование: Проведение тестирования для обеспечения стабильной работы ПО и выявления возможных проблем.

Миграция данных: Перенос данных из старых систем в новое ПО, если необходимо.

Сопровождение и обновление: Поддержка и обновление ПО для обеспечения его надежной и актуальной работы.

Мониторинг и анализ: Оценка производительности и эффективности внедренного ПО, а также анализ его влияния на бизнес-процессы.

Цели и задачи внедрения ПО могут сильно различаться в зависимости от конкретной ситуации, и они должны быть четко определены перед началом проекта внедрения.

1. Назовите основные рабочие группы в команде разработчиков.

команде разработчиков обычно выделяют следующие основные рабочие группы:

**Front-end разработчики:** Отвечают за создание пользовательского интерфейса, который видит и взаимодействует с ними конечные пользователи.

**Back-end разработчики:** Занимаются разработкой серверной части приложения, управлением базами данных, и обеспечивают взаимодействие между фронт- и бэк-эндом.

**QA (Quality Assurance) или тестировщики**: Отвечают за тестирование приложения на ошибки и баги, а также за обеспечение его качества и соответствия требованиям.

**Дизайнеры**: Разрабатывают дизайн интерфейса, графику и визуальное оформление продукта.

**Продуктовые менеджеры:** Занимаются формированием требований к продукту, определением его функциональности и управлением процессом разработки.

**DevOps-инженеры:** Занимаются настройкой и поддержкой инфраструктуры, автоматизацией процессов разработки и развертывания приложения.

**Аналитики данных:** Если приложение связано с анализом данных, то аналитики занимаются сбором, анализом и визуализацией данных.

Системные администраторы: Занимаются управлением серверами и инфраструктурой, обеспечивая их надежную работу.

Эти группы работают совместно для успешной разработки и поддержки программных продуктов.

1. Дайте функциональную характеристику каждой группе

функциональная характеристика каждой из основных рабочих групп в команде разработчиков:

**Front-end разработчики:**

Отвечают за создание пользовательского интерфейса (UI) приложения.

Разрабатывают клиентскую часть, с которой взаимодействуют пользователи.

Занимаются версткой, стилизацией и созданием интерактивных элементов на веб-сайтах или в мобильных приложениях.

**Back-end разработчики:**

Разрабатывают серверную часть приложения, обеспечивая его функциональность и безопасность.

Управляют базами данных и обеспечивают взаимодействие фронт- и бэк-энда через API.

Решают задачи, связанные с хранением и обработкой данных, а также управлением серверами.

**QA (Quality Assurance) или тестировщики:**

Проводят тестирование приложения на ошибки, баги и несоответствие требованиям.

Отвечают за обеспечение качества разрабатываемого продукта.

Создают тестовые сценарии и отчеты о найденных проблемах.

**Дизайнеры:**

Разрабатывают дизайн интерфейса, определяя его визуальное оформление и пользовательскую эргономику.

Создают макеты, иллюстрации и графические элементы приложения.

Сотрудничают с разработчиками для внедрения дизайна в приложение.

**Продуктовые менеджеры:**

Формулируют требования к продукту, определяют его стратегию развития и функциональность.

Управляют процессом разработки, учитывая потребности пользователей и бизнес-цели.

Принимают решения о приоритетах и выпуске новых версий продукта.

**DevOps-инженеры:**

Обеспечивают автоматизацию процессов разработки, тестирования и развертывания приложения.

Управляют инфраструктурой и серверами, обеспечивая их надежную работу.

Разрабатывают и внедряют системы мониторинга и управления конфигурациями.

**Аналитики данных:**

Занимаются сбором, анализом и визуализацией данных, если приложение связано с аналитикой.

Помогают принимать бизнес-решения на основе данных и метрик.

Разрабатывают отчеты и дашборды для анализа производительности приложения.