**Отчет по самостоятельной работе №**2

**по дисциплине МДК 01.02 “Инструментальные средства разработки программного обеспечения”.**

Выполнил: студент

группы 319

Бакеев Рамзан Русланович

Дата 26.11.2024

**Цель работы**

Цель данной работы – научиться разрабатывать перечень артефактов и протоколов для успешного управления проектом и документирования его ключевых этапов.

**Основная структура задания**

Анализ проекта

Описание проекта: Разработка веб-приложения для управления задачами (Task Manager) с функционалом для создания, редактирования и отслеживания задач.

Цели проекта:

* Создать интуитивно понятное веб-приложение для пользователей.
* Обеспечить возможность совместной работы над задачами.
* Реализовать функционал уведомлений и напоминаний.

Задачи проекта:

* Сбор требований от пользователей.
* Проектирование архитектуры приложения.
* Разработка интерфейса и функционала.
* Тестирование приложения.
* Запуск и поддержка.

Основные этапы выполнения:

1. Сбор требований
2. Проектирование
3. Разработка
4. Тестирование
5. Запуск

Участники проекта и их роли:

* Менеджер проекта: координация работы команды, контроль сроков.
* Аналитик: сбор и анализ требований.
* Разработчики: реализация функционала приложения.
* Дизайнер: разработка пользовательского интерфейса.
* Тестировщик: проверка качества приложения.

Определение артефактов проекта

Классификация артефактов по назначению:

1. Документы требований:

* Техническое задание (ТЗ)
* Протоколы встреч с пользователями

1. Проектные документы:

* Дизайн-документ
* Архитектурная схема

1. Документы разработки:

* Код приложения
* Документация по API

1. Документы тестирования:

* План тестирования
* Отчеты о тестировании

1. Отчетные документы:

* Отчет о ходе выполнения проекта
* Финальный отчет

Разработка перечня артефактов

Техническое задание на разработку веб-приложения для управления задачами (Task Manager)

1. Введение

Данное техническое задание определяет требования к разработке веб-приложения для управления задачами, которое позволит пользователям создавать, редактировать и отслеживать задачи, а также обеспечит возможность совместной работы над ними.

1. Наименование и область применения

Разработчик:

Закрытое акционерное общество «ТехноМенеджмент»

Адрес: 123456, Москва, ул. Менеджеров, д. 5

Тел.: +7 (495) 123-45-67, факс: +7 (495) 765-43-21

Банковские реквизиты: ЗАО «ТехноМенеджмент», ИНН 1234567890, р/сч № 40702810100000000000 в ОАО «Сбербанк России», БИК 044525225, корр. счет № 30101810400000000225

Заказчик:

Компания «Продуктивность»

Адрес: 654321, Санкт-Петербург, ул. Рабочая, д. 10

Тел.: +7 (812) 987-65-43, факс: +7 (812) 345-67-89

Банковские реквизиты: ООО «Продуктивность», ИНН 9876543210, р/сч № 40802810000000000000 в ОАО «Альфа-Банк», БИК 044525593, корр. счет № 30101810400000000225

1. Основание для разработки

Разработка приложения основана на необходимости улучшения организации рабочего процесса и повышения продуктивности пользователей. Актуальность разработки подтверждается растущими требованиями к цифровым инструментам для управления задачами в условиях удаленной работы и командной работы.

1. Назначение разработки

Приложение предназначено для:

* Создания и редактирования задач с возможностью установки сроков и приоритетов.
* Совместной работы над задачами с возможностью назначения исполнителей и комментирования.
* Реализации функционала уведомлений и напоминаний о предстоящих сроках выполнения задач.
* Обеспечения интуитивно понятного интерфейса для пользователей разного уровня подготовки.

1. Технические требования к программе или программному изделию

* Наименование: Веб-приложение для управления задачами (Task Manager).
* Область применения: Приложение предназначено для использования индивидуальными пользователями и командами для организации работы, планирования задач и повышения продуктивности.
* Платформа: Веб-приложение должно быть доступно через современные браузеры (Chrome, Firefox, Safari).
* Язык программирования: Разработка может быть выполнена на JavaScript (React/Vue) для фронтенда и Node.js/Python для бэкенда.
* База данных: Использование реляционной базы данных (например, PostgreSQL или MySQL) для хранения информации о задачах и пользователях.
* Интерфейс: Должен быть адаптивным и обеспечивать удобный доступ с разных устройств.
* Безопасность: Необходимо обеспечить защиту данных пользователей и конфиденциальность информации о задачах.
* Интеграция: Возможность интеграции с другими системами (например, Google Calendar, Slack).

1. Технико-экономические показатели

* Увеличение производительности пользователей за счет улучшения организации задач.
* Снижение времени на управление проектами на 20% благодаря автоматизации процессов.
* Повышение удовлетворенности пользователей за счет удобного интерфейса и функционала.

1. Стадии и этапы разработки
2. Сбор требований: Сбор и анализ требований от пользователей, формирование функциональных спецификаций.
3. Проектирование: Разработка архитектуры приложения и дизайна пользовательского интерфейса.
4. Разработка: Программирование функционала приложения.
5. Тестирование: Проведение тестирования приложения на различных этапах разработки для выявления и устранения ошибок.
6. Запуск: Запуск приложения в эксплуатацию и обучение пользователей.
7. Порядок контроля и приемки

После завершения каждого этапа разработки Заказчик имеет право провести контрольный аудит выполненных работ. Приемка будет осуществляться по следующим критериям:

* Соответствие выполненных работ требованиям технического задания.
* Функциональность приложения согласно спецификациям.
* Отсутствие критических ошибок после тестирования.

|  |  |
| --- | --- |
| Представитель заказчика | Представитель разработчика |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Иванов И.И/ | \_\_\_\_\_\_\_\_\_\_\_\_/Петров П.П./ |
| подпись | подпись |

**Дизайн-документ для веб-приложения управления задачами (Task Manager)**

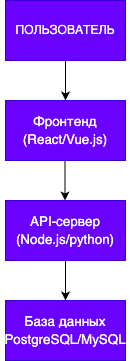
**1. Введение**

Данный документ описывает архитектуру и дизайн веб-приложения для управления задачами. Он предназначен для команды разработчиков, архитекторов и других заинтересованных сторон, чтобы обеспечить понимание структуры системы, ее компонентов и взаимодействия между ними.

**2. Архитектура системы**

**2.1. Общая архитектура**

Приложение будет построено по архитектурной модели клиент-сервер, где клиентская часть (фронтенд) будет взаимодействовать с серверной частью (бэкенд) через RESTful API.



2.2. Компоненты системы

2.2.1. Фронтенд

* Технологии: React или Vue.js для создания динамического пользовательского интерфейса.

Структура:

* + Главная страница (Dashboard)
  + Страница создания/редактирования задачи
  + Страница просмотра задач
  + Страница профиля пользователя
  + Состояние приложения: Использование Redux или Vuex для управления состоянием приложения.

2.2.2. Бэкенд

* + Технологии: Node.js с Express или Python с Flask/Django.
  + RESTful API: Реализация следующих конечных точек:
  + GET /tasks— получение списка задач.
  + POST /tasks— создание новой задачи.
  + GET /tasks/:id— получение информации о конкретной задаче.
  + PUT /tasks/:id— обновление существующей задачи.
  + DELETE /tasks/:id— удаление задачи.
  + POST /users/login— аутентификация пользователя.
  + POST /users/register— регистрация нового пользователя.

2.2.3. База данных

* + СУБД: PostgreSQL или MySQL.
  + Структура базы данных:
  + Таблица users: хранит информацию о пользователях (id, имя, email, пароль).
  + Таблица tasks: хранит информацию о задачах (id, название, описание, срок выполнения, приоритет, статус, id исполнителя).
  + Таблица comments: хранит комментарии к задачам (id, текст комментария, id задачи, id пользователя).

3. Пользовательский интерфейс

3.1. Дизайн интерфейса

* + Простота и удобство: Интерфейс должен быть интуитивно понятным и простым в использовании.
  + Адаптивность: Дизайн должен адаптироваться под различные размеры экранов (мобильные устройства, планшеты, десктопы).
  + Цветовая палитра: Использование нейтральных тонов с акцентами на кнопках и важных элементах.

3.2. Макеты страниц

* + Главная страница (Dashboard):
  + Список задач с фильтрацией и сортировкой.
  + Кнопка для создания новой задачи.
  + Страница задачи:
  + Полная информация о задаче.
  + Возможность редактирования и удаления задачи.
  + Секция комментариев.
  + Страница профиля:
  + Информация о пользователе.
  + Возможность изменения пароля.

4. Безопасность

4.1. Аутентификация и авторизация

* + Использование JWT (JSON Web Tokens) для аутентификации пользователей.
  + Защита маршрутов API с помощью middleware для проверки токенов.

4.2. Защита данных

* + Шифрование паролей пользователей с использованием bcrypt.
  + Регулярные резервные копии базы данных.

5. Тестирование

5.1. Типы тестирования

* + Юнит-тестирование:Тестирование отдельных компонентов фронтенда и бэкенда.
  + Интеграционное тестирование:Проверка взаимодействия между компонентами системы.
  + Энд-то-энд тестирование: Полное тестирование функциональности приложения с помощью инструментов вроде Cypress или Selenium.

5.2. План тестирования

* + Разработка тест-кейсов на основе функциональных требований.
  + Автоматизация тестирования для повышения эффективности.

Протоколы взаимодействия в проекте

Пример протокола совещания

Протокол совещания №1

Дата: 01.01.2023

Время: 10:00

Место: Zoom

Участники:

• Менеджер проекта: Иванов И.И.

• Аналитик: Петров П.П.

• Дизайнер: Сидорова С.С.

• Разработчик: Кузнецов А.А.

Повестка дня:

1. Обсуждение требований к приложению.

2. Определение сроков выполнения этапов.

Решения

1. Утвердить список функциональных требований.

2. Назначить сроки для завершения этапа сбора требований до 15.01.2023.

Следующее совещание:15.01.2023

**Пример протокола внесения изменений**

**Протокол внесения изменений в ТЗ**

**Дата:** 10.01.2023

**Изменение №:** 1

**Описание изменения:** Добавление нового функционала — возможность прикрепления файлов к задачам.

**Причина изменения:** Запрос от пользователей для улучшения функционала.

**Ответственные:**

* + Аналитик: Петров П.П.
  + Разработчик: Кузнецов А.А.

**Сроки выполнения:** Изменения должны быть внесены до 20.01.2023.

Оформление примеров артефактов и протоколов

ТЗ и дизайн документа приведены выше.

Заключение

Артефакты и протоколы обеспечивают контроль за выполнением проекта и помогают его успешной реализации, поскольку:

Артефакты фиксируют требования, проектные решения и результаты, что позволяет отслеживать прогресс и избегать недоразумений.

Протоколы формализуют процессы взаимодействия и принятия решений, что способствует прозрачности и эффективному управлению командой.

Использование данных инструментов позволяет минимизировать риски и повышать качество конечного продукта, что в свою очередь способствует удовлетворенности клиентов и успешному завершению проекта.

**Выводы работы**

Мы научились разрабатывать перечень артефактов и протоколов для успешного управления проектом и документирования его ключевых этапов.