**Самостоятельная работа**

**"Описание и анализ требований, выбор решений для разработки приложений"**

*Часть 1:*

1. *Различия требований* заключаются с в том, что функциональные определяют, что должна делать система, то нефункциональные описывают, как она будет это делать.

Примеры:

* + Функциональные – в онлайн магазине размер скидки зависит от кол-ва покупок; при оформлении заказа в сообщения приходит код с подтверждением.
  + Нефункциональные – система должна быть защищена и безопасна; проста в обслуживании и обновлениях.

1. *Правильное оформление* требований важно для успешной разработки проекта, тк благодаря этому будет четкое понимание работы, сразу будут определены все этапы и риски, что поспособствует быстрой реализации и практически безошибочному исполнению. Влияние на команду – масштабное, в связи с заранее прописанным четким планом действий. Т.е. люди будут делать все в одном «стиле». А конечный результат будет успешным, ведь чем лучше вы подготовлены к предстоящему проекту, тем меньше вероятность того, что вы столкнетесь с проектными рисками.
2. *Первое* – приложение должно увеличить кол-во забронированных столиков на 15%. *Второе* – приложение должно поддерживать оповещения клиентов в виде СМС за 24 часа.
3. Как *путешественник*, я хочу быстро искать и фильтровать отели по цене, удобствам и расположению, чтобы легко выбрать подходящее место для проживания, соответствующее моим требованиям и бюджету.

*Часть 2:*

**Приложение: Планировщик задач**

## *Функциональные требования:*

* Создание задач: Пользователь должен иметь возможность создавать задачи с указанием названия, описания, срока выполнения и приоритета.
* Редактирование задач: Пользователь должен иметь возможность редактировать существующие задачи, изменяя их детали, такие как срок, описание и приоритет.
* Установка напоминаний: Приложение должно позволять пользователю устанавливать напоминания для каждой задачи через push-уведомления.
* Категоризация задач: Пользователь должен иметь возможность группировать задачи по категориям для удобства просмотра и организации (например, работа, личное, учеба).
* Просмотр расписания: Приложение должно предоставлять различные форматы просмотра задач.

## *Нефункциональные требования:*

* Приложение должно обеспечивать время отклика не более 2 секунд при загрузке задач, чтобы пользователи не испытывали задержек в работе.
* Приложение должно быть доступно на основных мобильных платформах (iOS и Android), обеспечивая одинаковый функционал и пользовательский опыт на обеих системах.
* Приложение должно обеспечивать шифрование пользовательских данных и соответствовать современным стандартам безопасности для защиты личной информации.

## *Бизнес-требования:*

* Приложение должно увеличить кол-во молодежи на платформе на 30%.
* В первый год приложение должно привести доход в 100 000 долл.

*Часть 3:*

1. *Приоритизация с использованием метода MOSCOW*

* Просмотр меню и выбор блюд
* Поддержка кредитных карт и PayPal.
* Возможность оплаты через интернет.
* Уведомления о статусе заказа.
* Отслеживание доставки.

1. *Создание диаграмм*

* Класс «Пользователь» содержит атрибуты: ФИО, номер, СМС код.
* Класс «Заказ» содержит пункты информации о цене товара, наличии в определенном магазине и уровне сбора.
* Класс «Ресторан» содержит общее меню, контактный телефон, сборник отзывов и предложений.

*Часть 4:*

1. *Анализ рисков:*

* Неполнота требований: Требования могут изменяться по мере разработки, что увеличит время разработки и потребует пересмотра архитектуры.
* Технологические ограничения: Выбор неподходящих технологий может усложнить процесс интеграции и повлиять на производительность.
* Ошибки в коде: Непредвиденные ошибки могут удлинить срок разработки и снизить качество.
* Низкий интерес пользователей: Если приложение не удовлетворит потребности пользователей, это может привести к низкому уровню популярности.

1. *Выбор технологического стека:*

* Фронтенд: Vue.js - для создания пользовательского интерфейса.
* Бэкенд: Node.js - позволяет создавать серверную часть приложения на JavaScript, что облегчает взаимодействие с фронтендом.
* База данных: MongoDB - документо-ориентированная база данных, хорошо подходящая для работы с JSON-структурами и простая в масштабировании.

***Выполнила: Таран Настя, 319гр***