**Самостоятельная работа**

Тема: “Описание и анализ требований, выбор решений для разработки приложений”

Цель: Укрепить знания о классификации требований, методах их анализа, а также навыки выбора и применения технологических решений для разработки приложений

**Часть 1: Теория и анализ требований**

В функциональных и нефункциональных требованиях много различий, основным является то, что к нефункциональным относится сама работоспособность системы. В то время как к функциональным относятся требования к тому, что должна делать система.

К примерам функциональных требований можно отнести:

1. Авторизация и аутентификация
2. Отправка уведомлений
3. Поддержка мобильных устройств

Примеры нефункциональных требований:

1. Производительность
2. Надежность
3. Безопасность
4. Удобство пользования

Правильное оформление требований безусловно очень важно для успешной разработки проекта. Во-первых, хочется отметить, что это помогает легче определить конечные цели проекта. Во-вторых, помогает избежать недопонимая в компании и сокращает количество переделок, что в свою очередь экономит время и ресурсы. В-третьих, помогает создать качественный продукт, который будет соответствовать ожиданиям и потребностям всех заинтересованных сторон.

В качестве бизнес-требований для приложения с функцией бронировки столиков в ресторанах можно привести в пример возможность безналичной оплаты сразу внутри приложения и в будущем создание сайта с аналогичными выполняемыми функциями.

User Story для мобильного приложения по бронированию отелей:

1. Кто будет выполнять действия? Люди, желающие поехать в отпуск и забронировать отель в одном удобном для них приложении.
2. Какое действие будет выполняться? Клиент выбирает срок своего отпуска и в пару кликов может забронировать любой понравившийся ему отель.
3. Зачем это нужно? Главная цель приложения – удобство для обычных пользователей. Не нужно будет делать все эти действия в разных местах, когда есть комфортная альтернатива другим сервисам.

**Часть 2: Практическая работа с требованиями**

Функциональные требования для приложения планировщика задач:

1. Списки задач. Это неотъемлемая часть любого подобного приложения.
2. Напоминания. Данная функция поможет человеку не забывать о важных делах.
3. Заметки. Эта возможность подарит пользователю более подробно прописывать все мелочи поставленных перед ним задач и не забывать о них.
4. Календарь. В приложения можно встроить календарь и возможность редактировать его.
5. Экспорт и импорт данных. Данная функция облегчит жизнь тем пользователям, которые захотят перейти от приложения-конкурента именно к нам!

Нефункциональные требования для приложения планировщика задач:

1. Производительность. Эта составляющая присуща каждому приложению направленного на клиентоориентированность.
2. Безопасность. Каждому человеку важно, чтобы его личные данные не имели доступа к третьим лицам, наш планировщик данных точно не будет промышлять ничем подобным.
3. Надежность. Мы гарантируем отсутствие багов и сбоев работы нашего сервиса.

Бизнес требования для приложения планировщика задач:

1. Повышение доходности посредством платного контента внутри приложения на 15% в течение следующего года.
2. Увеличить скорость работы приложения на 10% в течении полугода.

Требования к разработке приложения планировщика задач:

1. Функциональные

* Пользователь должен иметь возможность зарегистрироваться по номеру телефона, электронной почте и через авторизацию VK.
* Приложение должно иметь возможность импортировать и экспортировать данные из сторонних источников.
* Пользователь должен иметь возможность приобрести Premium подписку на планировщик задач посредством безналичной оплаты в приложении.

1. Нефункциональные

* Сервис должен запрашивать минимально необходимые данные и разрешения для корректной работы приложения.
* Сервис должен иметь функцию удаления информации по запросу клиента, так как могут быть слиты персональные данные о пользователях.
* Сервис при регистрации должен запрашивать подтверждение на согласие о политики конфиденциальности.

1. Бизнес требования

* Необходимо поднять продажи на 15% в течении следующего года.
* Необходимо улучшить работоспособность приложения при большом количестве одновременных пользователей.

**Часть 3: Приоритизация требований в диаграммы**

1. Приоритизация с использованием метода MOSCOW:

* M (must) – просмотр меню и выбора блюд.
* S (should) – возможность оплаты через интернет, поддержка кредитных карт и PayPal.
* C (could) – уведомления о статусе заказа.
* W (would) – отслеживание доставки.

1. Создание диаграмм:

* Класс “Пользователь” имеет айди, логин, пароль, email, адрес, номер телефона.
* Класс “Заказ” содержит информацию о том кто сделал заказ, где сделан заказ, список блюд в заказе, общая стоимость, статус заказа, статус платежа.
* Класс “Ресторан” содержит информацию о названии, адресе, меню. айди
* Класс “Блюдо” содержит информацию о названии, описании, цене.
* Класс “Платеж” содержит информацию о заказе, сумме и статусе платежа.

1. Приоритеты для требований планировщика задач:

* M (must) – списки задач, календарь, заметки
* S (should) – регистрация, подтверждение согласия о политике конфиденциальности
* C (could) – напоминания, импорт и экспорт данных, Premium подписка
* W (would) – авторизация через OK, VK, ICQ, WhatsApp, Viber, Skype.

**Часть 4: Анализ рисков и выбор технологического стека**

**Анализ рисков.** Основными рисками является недостаточный бюджет, из этого вытекает дешевая оплата труда низкоквалифицированным сотрудникам. В следствии чего может возникнуть ситуация плохой работоспособности планировщика задач. Из-за каких-то мелочей может провалиться целый бизнес, основанный на сети приложений, связанных с повышением продуктивности обычного люда. Без такого рода приложений людям будет сложнее визуализировать собственное время. Как мы все прекрасно знаем, время – ценнейший ресурс, ибо он ограничен. В связи с отсутствием планировщика задач люди могут стать рассеяннее, забывчивее, что в свою очередь скажется на благосостоянии всей страны.

**Выбор технического стека.** Безусловно для фронтенда и кросс-платформенной разработки будет выбран React Native. Этот выбор не случаен, мной был проведен тщательный анализ всех существующих фреймворков, которые бы смогли в полной мере помочь с разработкой планировщика задач. Как не удивительно именно React Native способен подарить быструю работоспособность и хорошую производительность, что в нашей ситуации – отличный выбор. Для бэкенда будет выбран Django + REST API, с помощью которых можно со всем удобством управлять данными. Так же будет использован PostgreSQL для хранения информации о пользователях и заказах.