# ПОСТАНОВКА ЦЕЛЕЙ РАБОТЫ

Основной целью разрабатываемого приложения "Заметки" является предоставление пользователям удобного и простого инструмента для создания, хранения и управления своими заметками. Приложение должно позволять пользователям создавать заметки, организовывать их поиск посредством поисковой строки, а также редактировать и удалять уже созданные заметки.

Основной задачей приложения является облегчение жизни пользователей, позволяя им быстро и удобно создавать и хранить заметки, которые могут использоваться пользователями для просмотра занесенной информации.

Основными функциями данного приложения являются создание, редактирование и удаление заметки. К второстепенным функциям можно отнести поиск заметки по заголовку, внесение фотографии в заметку посредством камеры или галереи устройства.

# ОПРЕДЕЛЕНИЕ ЗАДАЧ РАБОТЫ

Исходя из предыдущего раздела, можно поставить следующие задачи работы:

* Разработка базы данных заметок;
* Создание CRUD операций для написания базового функционала взаимодействия с БД заметок (создание, редактирование, удаление и просмотр);
* Разработка пользовательского интерфейса приложения;
* Создание второстепенных функций взаимодействия с заметками.

# ОБОСНОВАНИЕ ВЫБОРА СТЕКА ТЕХНОЛОГИЙ

Стек технологий, который будет задействован для реализации приложения "Заметки", включает в себя язык программирования Dart и фреймворк Flutter.

Язык Dart был выбран для разработки приложения из-за его высокой производительности, возможности использования как для клиентской, так и для серверной разработки, а также поддержки объектно-ориентированного программирования. Dart также имеет хорошую документацию и большое сообщество разработчиков, что облегчает процесс разработки. В данном проекте будет использоваться Dart версии 3.0.1. Это означает, что будет использоваться null-safety или null-безопасность, которая позволит избежать ошибок в коде, связанных с использованием null-значений. Важно также уточнить, что все последующие используемые зависимости в нашем проекте будут иметь версии, поддерживающие null-безопасность.

Вот несколько причин, почему стоит использовать Flutter для разработки приложения Заметки:

1. Быстрое развертывание: благодаря горячей перезагрузке в Flutter можно быстро просматривать изменения в режиме реального времени, что ускоряет процесс разработки.

2. Кроссплатформенность: приложение, созданное на Flutter, может работать как на iOS, так и на Android, что позволяет сэкономить время и средства на разработку отдельных приложений для каждой платформы.

3. Высокая производительность: благодаря использованию компиляции в нативный код и минимизации использования промежуточных слоев, Flutter обеспечивает высокую производительность и быстродействие.

4. Большое сообщество: Flutter имеет активное сообщество разработчиков, которые делятся своим опытом и создают полезные инструменты и библиотеки.

5. Удобная работа с UI: в Flutter есть множество готовых виджетов и инструментов для создания красивого и удобного интерфейса приложения.

Для хранения заметок в приложении будет использоваться база данных SQLite, которая реализуется посредством зависимости sqflite. SQLite была выбрана из-за своей легковесности, быстродействия и широкой поддержки во многих языках программирования. Sqflite предоставляет удобный интерфейс для работы с базой данных SQLite в приложении на Flutter.

Для создания CRUD операций для базы данных заметок будет использоваться язык Dart и фреймворк Flutter. Flutter предоставляет удобные инструменты для работы с базами данных, такие как пакет sqflite, который обеспечивает простой и удобный интерфейс для выполнения операций CRUD.

Для разработки пользовательского интерфейса приложения будет использоваться фреймворк Flutter. Flutter предоставляет множество виджетов и инструментов для создания красивых и удобных пользовательских интерфейсов. В приложении будут использоваться различные виджеты, такие как текстовые поля, кнопки, списки и др.

Для создания второстепенных функций взаимодействия с заметками будут использоваться различные зависимости, такие как image\_picker для выбора изображений, intl для работы с локализацией, path для работы с путями файловой системы, path\_provider для получения доступа к файловой системе устройства, provider для управления состоянием приложения и google\_fonts для выбора шрифтов.

В целом, выбранный стек технологий обеспечивает быстрое и удобное создание приложения "Заметки" с использованием современных технологий и инструментов. Он позволяет создавать красивый и удобный пользовательский интерфейс, эффективно работать с базой данных и реализовывать различные функции взаимодействия с заметками.

# ОПИСАНИЕ ПРОЦЕССА РАЗРАБОТКИ

Процесс разработки я разделил на несколько этапов:

* Создание набросков интерфейса будущего приложения;
* Разбиение проекта на подчасти, папки, dart-файлы;
* Подбор зависимостей проекта в соответствии с версией Dart;
* Создание класса Заметок, содержащий методы и свойства объекта заметки;
* Создание базы данных заметок в отдельном dart-файле посредством импорта зависимости sqflite;
* Создание провайдера заметок в отдельном dart-файле, реализующий функции добавления, удаления и обновления заметок;
* Создание файла, содержащий шаблоны шрифтов для дальнейшей реализации интерфейса приложения;
* Создание страницы в отдельном dart-файле, которая будет отображать интерфейс начальной страницы приложения, а также кнопки добавления заметки на данной странице;
* Промежуточное тестирование (тестирование отображения начальной страницы);
* Создание страницы редактирования заметки в отдельном dart-файле, которая будет отображаться при нажатии нажатия на кнопку создания заметки;
* Создание функции сохранения и удаления заметки на странице редактирования, а также кнопки возвращения на главную страницу;
* Занесение части второстепенного функционала на страницу редактирования (вставка картинки при помощи камеры или галереи устройства);
* Промежуточное тестирование имеющегося функционала;
* Создание класса в отдельном dart-файле, используемый для отображения созданных заметок в виде контейнеров на главной странице;
* Промежуточное тестирование;
* Создание страницы в отдельном dart-файле, которая будет отображать содержимое заметки при нажатии на контейнер уже созданной заметки;
* Занесение оставшегося второстепенного функционала – поисковая строк
* Конечное тестирование приложения как на эмуляторе, так и на реальном устройстве.

Основной функционал приложения, описанный в разделе 2, был реализован при помощи нескольких взаимосвязанных классов и виджетов.

Изначально, был создан класс Заметки, представляющий собой модель заметки в приложении. В него также были внесены некоторые методы, для отображения даты создания заметки и отображения пустой заметки.

Далее необходимо было создать класс для реализации взаимодействия приложения с базой данных SQLite. В классе был реализован метод создания базы данных заметок. Также в данном классе были отображены функции для работы с БД, такие как добавление записи в таблицу, получение списка записей из таблицы и удаление записей из таблицы.

Затем был создан класс-провайдер, для работы с заметками приложения. Он использует класс, описанный выше, для работы с базой данных SQLite и имеет методы для получения списка заметок, добавления/обновления заметки и удаления заметки.

После создания бэкенда приложения, нужно было написать интерфейс, который бы и позволил реализовать базовый функционал в графической оболочке. Для этого использовались различные виджеты Flutter и дополнительные функции переходы между страницами.

Соответственно, каждый этап разработки интерфейса программы сопровождался промежуточным тестированием и отладкой, для внесения корректировок в логику работы программы.

Таким образом, был реализован основной функционал приложения.

В процессе разработки, были трудности, которые замедляли работу. К таким проблемам можно причислить:

* Выборка походящих версий зависимостей.
* Частые ошибки в логике реализации null-safety переменных, особенно, если они в дальнейшем использовались как параметры для методов.
* Сложность реализации некоторых функций, из-за отсутствий профессиональных знаний владения имеющимся стеком технологий.

Столкнувшись с вышеуказанными проблемами, необходимо было уделить часть времени на их устранение.

Так, для выборки подходящих зависимостей, хорошим решением стало изучение версий этих пакетов, а также их совместимости с версий Dart SDK на сайте <https://pub.dev/packages/>, где можно было выбрать наименование пакета, а затем просмотреть доступные версии этой зависимости.

Частые ошибки, возникшие при реализации null-safety переменных, удалось решить благодаря подчерпнутым знаниям из статьи <https://skondratev.com/null-safety-v-dart/>, здесь подробно говорилось о различных методах, используемые для реализации null-безопасности в языке Dart. Также было полезно просмотреть русскоязычное видео на ютуб-платформе <https://youtu.be/RFOOdTN4TxU>, позволяющее ясно понять, “когда” и “где” нужно использовать тот или иной метод null-safety.

Возникшие сложности при реализации некоторых функция из-за недостатка знаний в области, были решены благодаря прочтению документации к языку Dart (<https://dart.dev/guides>) и фреймворку Flutter (<https://docs.flutter.dev>).

## Схема переходов приложения

