# ***Задание на практику***

# ***ПП.01.01 Разработка мобильных приложений***

# *Цель практики:*

# *Разработать мобильное приложение заданной тематики*

# *Структура задания:*

1. Модуль 1 Разработка технического задания:

* описание функциональных требований;
* проектирование структуры данных;
* создание макета приложения.

1. Модуль 2 Разработка серверной части;
2. Модуль 3 Верстка мобильного приложения;
3. Модуль 4 Разработка функционала мобильного приложения;
4. Модуль 5 Тестирование приложения;
5. Модуль 6 Документирование результатов разработки.

# *Общие требования к приложению:*

1. в работе необходимо использовать систему контроля версий Git, и ежедневно сохранять разработанные элементы проекта;
2. проект приложения должен быть структурирован, то есть исходные файлы должны быть в соответствующей папке, общие классы необходимо поместить в папку common.

Например, при использовании в проекте MVC, файлы проекта должны быть распределены по следующим папкам: view, model, controller, common.

1. в приложении должны корректно обрабатываться запросы к серверу. В случае получения ошибки от сервера или отсутствия соединения с сетью Интернет необходимо отобразить соответствующий текст ошибки в диалоговом окне, которое должно закрываться только пользователем.

# *Этапы выполнения*

Перечисленные далее модули являются обязательными, в независимости от темы индивидуального задания. Внимательно изучите задачи и этапы модулей прежде чем приступать к заданию. Необходимо добавить дополнительные функции, соответствующие теме индивидуального задания. Допускается изменение обязательных функций, с разрешения преподавателя.

## **Модуль 1 – Разработка технического задания**

1. Определение целей и задач приложения:

Цель: Создать удобное и интуитивно понятное мобильное приложение для онлайн-покупки мебели, обеспечивающее пользователям возможность просмотра каталога, выбора товаров, оформления заказов и отслеживания статуса доставки.

Задачи:

* + Предоставить пользователям удобный интерфейс для просмотра каталога мебели с детализированными описаниями, фотографиями и ценами.
  + Обеспечить возможность фильтрации и сортировки товаров по различным параметрам (цена, тип, размер, материал, бренд и т.д.).
  + Реализовать функцию добавления товаров в корзину и оформления заказа с указанием адреса доставки и способа оплаты.
  + Обеспечить безопасную авторизацию пользователей и хранение их данных.
  + Предоставить пользователям возможность отслеживания статуса заказа.
  + Реализовать функцию обратной связи с продавцом.
  + Уведомлять пользователей о статусе заказов, акциях и скидках.

1. Определение функциональных требований:
   * Авторизация/Регистрация:
     1. Регистрация пользователя по email и паролю.
     2. Авторизация пользователя по email и паролю.
     3. Восстановление пароля через email (с использованием OTP).
   * Просмотр каталога:
     1. Отображение списка категорий мебели (диваны, кресла, столы, кровати и т.д.).
     2. Отображение списка товаров в выбранной категории.
     3. Детализированное описание товара (фотографии, описание, характеристики, цена).
     4. Фильтрация товаров по различным параметрам (цена, бренд, материал, размер, стиль).
     5. Сортировка товаров (по цене, популярности, новизне).
     6. Поиск товаров по названию или описанию.
   * Корзина:
     1. Добавление товаров в корзину.
     2. Просмотр содержимого корзины.
     3. Редактирование количества товаров в корзине.
     4. Удаление товаров из корзины.
     5. Расчет общей стоимости заказа.
   * Оформление заказа:
     1. Ввод адреса доставки.
     2. Выбор способа оплаты (онлайн-оплата, оплата при получении).
     3. Подтверждение заказа.
   * Личный кабинет:
     1. Просмотр информации о пользователе (имя, email, адрес доставки).
     2. Редактирование информации о пользователе.
     3. Просмотр истории заказов.
     4. Отслеживание статуса текущих заказов.
   * Прочее:
     1. Push-уведомления о статусе заказа, акциях, скидках.
     2. Обратная связь с продавцом (чат или форма обратной связи).
     3. Отображение информации о доставке и оплате.
     4. Интеграция с платежными системами (если требуется).
2. Проектирование и описание структуры данных:

* Пользователи (Users):
  + id (INT, PRIMARY KEY, AUTOINCREMENT) - Уникальный идентификатор пользователя.
  + email (VARCHAR) - Email пользователя.
  + password (VARCHAR) - Хэшированный пароль пользователя.
  + name (VARCHAR) - Имя пользователя.
  + address (VARCHAR) - Адрес доставки.
  + phone (VARCHAR) - Номер телефона (опционально).
  + registration\_date (TIMESTAMP) - Дата регистрации.
* Категории (Categories):
  + id (INT, PRIMARY KEY, AUTOINCREMENT) - Уникальный идентификатор категории.
  + name (VARCHAR) - Название категории (например, “Диваны”, “Кресла”).
  + description (TEXT) - Описание категории (опционально).
  + image\_url (VARCHAR) - URL изображения категории (опционально).
* Товары (Products):
  + id (INT, PRIMARY KEY, AUTOINCREMENT) - Уникальный идентификатор товара.
  + category\_id (INT, FOREIGN KEY references Categories.id) - ID категории товара.
  + name (VARCHAR) - Название товара.
  + description (TEXT) - Описание товара.
  + price (DECIMAL) - Цена товара.
  + image\_urls (JSONB) - Массив URL изображений товара (или отдельная таблица для изображений).
  + specifications (JSONB) - JSON с характеристиками товара (размеры, материалы и т.д.).
  + brand (VARCHAR) - Бренд товара (опционально).
* Заказы (Orders):
  + id (INT, PRIMARY KEY, AUTOINCREMENT) - Уникальный идентификатор заказа.
  + user\_id (INT, FOREIGN KEY references Users.id) - ID пользователя, сделавшего заказ.
  + order\_date (TIMESTAMP) - Дата и время заказа.
  + delivery\_address (VARCHAR) - Адрес доставки.
  + total\_amount (DECIMAL) - Общая сумма заказа.
  + status (VARCHAR) - Статус заказа (например, “В обработке”, “Отправлен”, “Доставлен”, “Отменен”).
  + payment\_method (VARCHAR) - Способ оплаты.
* Элементы заказа (OrderItems):
  + id (INT, PRIMARY KEY, AUTOINCREMENT) - Уникальный идентификатор элемента заказа.
  + order\_id (INT, FOREIGN KEY references Orders.id) - ID заказа, к которому относится элемент.
  + product\_id (INT, FOREIGN KEY references Products.id) - ID товара.
  + quantity (INT) - Количество товара в заказе.
  + price\_per\_item (DECIMAL) - Цена за единицу товара на момент заказа.
* OTP (One-Time Password - для восстановления пароля):
  + id (INT, PRIMARY KEY, AUTOINCREMENT) - Уникальный идентификатор записи.
  + email (VARCHAR) - Email, для которого сгенерирован OTP.
  + otp\_code (VARCHAR) - Сгенерированный OTP-код (6 символов).
  + expiration\_time (TIMESTAMP) - Время истечения срока действия кода.

4. Разработка дизайн-макета приложения в Figma.

* Дизайн-макет должен включать в себя экраны, описанные в Модуле 3 (см. ниже), с четким представлением интерфейса, навигации, стилей и элементов управления.
* Макет должен быть удобным для пользователя, с учетом принципов юзабилити.
* Необходимо предоставить несколько вариантов дизайна для экранов, чтобы показать различные варианты стилей.

## **Модуль 2 - Разработка серверной части**

Требования при разработке серверной части:

1. Реализованная база данных должна покрывать все задачи приложения: База данных (Supabase) должна содержать все необходимые таблицы, описанные в разделе 3 (Структура данных).
2. Все данные, которые используются в системе, должны храниться в базе данных, за исключением локальных и данных для тестирования: Все данные, требуемые для функционирования приложения, включая данные о пользователях, товарах, категориях, заказах и т.д., должны храниться в базе данных.
3. Для управления доступом к данным должна быть реализована политика безопасности: Настройте политики безопасности в Supabase, чтобы ограничить доступ к данным в соответствии с ролью пользователя (например, только авторизованные пользователи могут видеть свои заказы, только администраторы могут изменять информацию о товарах).
4. В базе данных должно быть реализовано хранение изменяемых файлов и изображений: Используйте хранилище файлов Supabase для хранения изображений товаров и других динамических файлов.
5. Аутентификация пользователя должна быть реализована по почте и паролю: Используйте встроенную систему аутентификации Supabase для аутентификации пользователей по email и паролю.
6. Восстановление пароля пользователя по почте должно быть реализовано с помощью шестизначного OTP-кода: Реализуйте функциональность восстановления пароля с использованием OTP, генерируемого и отправляемого на email пользователя (используйте функции Supabase для этого или реализуйте логику отправки email самостоятельно).
7. База данных должна быть заполнена достаточным количеством данных для проверки работоспособности всех функций приложения: Заполните базу данных тестовыми данными: пользователи, категории, товары, заказы, чтобы проверить работу всех функций приложения.

## **Модуль 3 Верстка мобильного приложения**

Требования при реализации дизайна приложения:

1. Дизайн приложения должен строго следовать разработанному макету: При верстке используйте элементы, цвета, шрифты и стили, соответствующие дизайну, созданному в Figma.
2. Вся верстка должна быть адаптивной (следует учитывать разные размеры экранов): Используйте адаптивный дизайн, чтобы приложение корректно отображалось на устройствах с разными размерами экранов.
3. Приложение должно иметь настроенную иконку в соответствии с макетом: Создайте и настройте иконку приложения в соответствии с дизайном.
4. Приложение должно иметь следующие экраны:

* Загрузочный экран: Отображается при запуске приложения (логотип, анимация загрузки).
* Экран Приветствия (Onboarding), который отображается, если пользователь не авторизован: Серия экранов с кратким описанием возможностей приложения (если есть).
* Экран Входа: Форма для ввода email и пароля.
* Экран Регистрации: Форма для регистрации нового пользователя (email, пароль, имя).
* Экран Восстановление пароля: Форма для ввода email для отправки OTP.
* экран Ввода ПИН-кода.
* экран Входа по ПИН-коду.
* Экран Ввода OTP-кода: Форма для ввода OTP-кода, полученного по email.
* Экран Изменения пароля: Форма для ввода нового пароля.
* Экран Личного профиля: Отображает информацию о пользователе (имя, email, адрес, история заказов).
* Экран Получение множества объектов данных (Каталог): Отображение списка категорий мебели, списка товаров в выбранной категории.
* Экран Получения множества данных с фильтрацией: Отображение списка товаров с возможностью фильтрации по различным параметрам.
* Экран Получение подробностей об одном объекте: Детальное описание товара (изображения, описание, характеристики, цена, кнопка добавления в корзину).
* Экраны Создание нового объекта, Изменения объекта (допускается объединение): Для создания и редактирования информации о пользователе, для изменения данных в корзине.
* Экран Создание сложных объектов (например, корзина, группа пользователей): Отображение корзины с выбранными товарами, возможностью изменения количества и удаления, расчетом итоговой стоимости, оформление заказа.
* Экран Оформления Заказа: Ввод адреса доставки, выбор способа оплаты.
* Экран “История заказов”.
* Экран “Статус заказа”.

## **Модуль 4 Разработка функционала клиентского приложения**

Минимальные требования к функционалу приложения:

1. Аутентификация: Неавторизованные пользователи должны быть ограничены в доступе к данным, за исключением общих разделов (каталог, информация о доставке и оплате)
2. Регистрация пользователя: Для создания нового пользователя необходимо ввести:
   * Имя пользователя (опционально).
   * Адрес электронной почты.
   * Пароль.
   * Подтверждение пароля.
   * Действия:
     + Данные пользователя отправляются на сервер (Supabase) для регистрации.
     + После успешной регистрации, пользователь авторизуется с помощью ПИН-кода.
     + В случае ошибки регистрации (например, неверный формат email, уже существующий email, несовпадение паролей), отображается соответствующее сообщение об ошибке.
3. Авторизация пользователя: Для авторизации пользователя необходимо ввести:
   * Адрес электронной почты.
   * Пароль.
   * Действия:
     + Данные пользователя отправляются на сервер (Supabase) для авторизации.
     + При успешной аутентификации:
       - Переход на главный экран (каталог).
       - Сохранение пользовательской сессии (см. п.4).
     + В случае ошибки авторизации (неверный email/пароль), отображается сообщение об ошибке.
4. Пользовательские сессии:
   * После успешной аутентификации данные пользователя сохраняются в системе для дальнейшего использования: Используйте токены JWT (JSON Web Tokens), предоставляемые Supabase, для аутентификации последующих запросов.
   * Электронная почта и пароль сохраняются в локальное хранилище (Shared Preferences, UserDefaults и т.д.), они используются для повторного входа и удаляются при выходе из системы (logout): Пользователь может выбрать “Запомнить меня” при авторизации. При этом email и пароль (или, лучше, зашифрованный ключ для получения токена) сохраняются в локальном хранилище. При последующих запусках приложения пользователь автоматически авторизуется, если сохранены учетные данные.
   * Данные токена пользователя сохраняются в виде локальной переменной и очищаются при закрытии приложения (или при logout): Токен JWT хранится в памяти приложения и удаляется при logout или закрытии приложения.
5. Валидация вводимых данных:
   * Проверять поля на соответствие типу данных:
     + Email: должен быть строкой.
     + Пароль: должен быть строкой.
   * Проверять поля для ввода на пустоту, на длину (короткий текст и числа: не более 20 символов, пароль: не более 8 символов):
     + Все обязательные поля (email, пароль) не должны быть пустыми.
     + Имя (если используется) - не более 20 символов.
     + Пароль - не более 8 символов (или другая приемлемая длина, с учетом безопасности).
   * Email на корректность (соответствие паттерну “[name@domenname.ru](mailto:name@domenname.ru)”, где имя и доменное имя может состоять только из маленьких букв и цифр): Используйте регулярное выражение для проверки соответствия email-адреса заданному формату.
   * Проверять повторный ввод пароля на соответствие исходному: Пароли, введенные в полях “Пароль” и “Подтвердить пароль” должны совпадать при регистрации и изменении пароля.
   * Проверять даты на корректность: При вводе даты рождения (в профиле пользователя) убедитесь, что дата соответствует реальной дате (например, 31 февраля не допустима).
6. Навигация в приложении:
   * Реализовать навигацию между всеми необходимыми экранами приложения (описанными в Модуле 3).
   * Для реализации навигации рекомендуется использовать фрагменты или библиотеки навигации: (Navigation Component для Android, SwiftUI или UIKit для iOS).
   * Для перехода между основными экранами приложения (например, главный экран, профиль, корзина) рекомендуется использовать нижнюю навигационную панель: Обеспечивает интуитивно понятный интерфейс для быстрого доступа к ключевым разделам.
7. ПИН-код:
   * После аутентификации пользователю открывается экран ПИН-кода, на котором он может ввести свой ПИН-код: После успешной авторизации (или после включения ПИН-кода в настройках профиля) пользователь должен ввести ПИН-код для доступа к приложению.
   * Если пользователь успешно авторизован в системе, при повторном входе в приложение открывается Экран входа по ПИН-коду, в котором пользователь может ввести ПИН-код либо выйти из приложения для повторной аутентификации: Пользователь может либо ввести ПИН-код, либо выйти из приложения.
8. Восстановление пароля:
   * Пользователь может перейти к функции восстановления пароля либо с экрана входа, либо с экрана профиля: В меню авторизации (или в профиле пользователя) должна быть ссылка на “Восстановление пароля”.
   * Пользователь вводит электронную почту, на которую приходит шестизначный OTP-код: В форме восстановления пароля пользователь вводит email, на который будет отправлен OTP-код (генерируется сервером, отправляется через Supabase Email Templates, например).
   * После отправки кода на почту приложение переводит пользователя на экран Ввода OTP-кода, где пользователь должен ввести корректный OTP-код: Пользователь получает email с OTP-кодом, вводит его в соответствующее поле.
   * Если OTP-код корректен, пользователь переходит на экран Изменения пароля, где вводит и подтверждает новый пароль: При успешной проверке OTP, пользователь переходит к изменению пароля.
9. Локальное хранилище:
   * Необходимо хранить данные пользователя, необходимые для автоматической аутентификации, в локальном хранилище приложения: (Shared Preferences для Android, UserDefaults/Keychain для iOS, или другая библиотека для локального хранения данных).
   * Совет: Храните только зашифрованные данные (например, токен аутентификации, но не пароль напрямую).
10. Профиль пользователя:
    * Личные данные пользователя сохраняются в его профиле и отображаются на экране Личного профиля: Отображение имени, email, адреса доставки (если есть), истории заказов (если есть).
    * На экране Личного профиля пользователь может изменить данные о пользователе: имя пользователя, электронную почту, пароль, аватар и т.д.:
      + Реализуйте возможность редактирования профиля.
      + При изменении данных, отправляйте обновления на сервер (Supabase).
    * Пользователь может загрузить аватар из галереи устройства:
      + Реализуйте функциональность загрузки аватара, используя встроенные средства ОС.
      + Загруженные изображения должны быть сохранены в хранилище файлов Supabase.
      + Отображайте загруженное изображение в профиле.
11. Отображение множества объектов данных пользователю (Каталог, списки товаров):
    * Данные должны отображаться в удобном для пользователя формате в виде горизонтального и вертикального списка: Используйте RecyclerView (Android), ListView/TableView (iOS) или другой подходящий компонент для отображения списков.
    * Обновление списка объектов по жесту (Pull-to-refresh): Реализуйте возможность обновления списка путем потягивания вниз (Pull-to-refresh).
    * Детализация элементов списка:
      + Переход на подробное описание элемента по нажатию на элемент: По нажатию на элемент списка (например, товар) пользователь должен перейти на экран с детальной информацией о товаре.
      + Вызов контекстного меню при долгом нажатии на элемент: (Опционально) Долгое нажатие может вызывать контекстное меню с дополнительными действиями (например, добавить в избранное).
      + Действия при нажатии на детали на элементе списка (например, добавление в избранное при нажатии на кнопку на элементе списка): Добавление товара в избранное, добавление в корзину (если функциональность предусмотрена).
    * Экран Получение подробностей об одном объекте (Товар):
      + Отображение фотографий товара (с возможностью увеличения).
      + Подробное описание товара.
      + Характеристики товара (размеры, материалы и т.д.).
      + Цена.
      + Кнопка “Добавить в корзину”.
      + (Опционально) Кнопка “Добавить в избранное”.
12. Фильтрация данных (Каталог):
    * Поиск по названию и/или описанию объекта данных: Реализуйте поле поиска для фильтрации товаров по названию и описанию.
    * Фильтрация по нескольким свойствам объекта данных (например, категориям, кол-ву товара, просмотрам и т.д.) отдельно или одновременно: Реализуйте возможность фильтрации по различным параметрам (цена, бренд, материал, размер, стиль). Предусмотрите интерфейс для выбора фильтров (например, выпадающие списки, чекбоксы, ползунки).
13. Изменение данных об объекте данных:
    * Создание, редактирование и удаление объектов данных: Данная функциональность доступна, вероятно, только администраторам или модераторам.
      + Реализуйте соответствующие экраны и интерфейсы для создания, редактирования и удаления товаров.
    * Система должна корректно аутентифицировать пользователя и позволять управлять только разрешенными объектами (например, только администраторы могут добавлять товары): Реализуйте проверку ролей пользователей (администратор, пользователь) и ограничьте доступ к функциям в соответствии с ролью.
    * Изменение данных должно быть реализовано в удобном для пользователя виде: Используйте интуитивно понятные формы и элементы управления.
14. Работа со сложными объектами:
    * Отображение, фильтрацию, создание, удаление, редактирование сложных объектов (имеющих две и более связи с другими таблицами) или сложных для восприятия (неструктурированных, или связующих таблиц) в удобном человеко-читаемом формате:
      + Пример: Корзина (связь между товарами и заказами). Отображение содержимого корзины (товары, количество, итоговая стоимость), возможность изменения количества, удаления товаров, оформление заказа.
      + Пример: Заказы (связь с пользователями, товарами, способами оплаты, адресами доставки). Отображение истории заказов, статуса заказа.
15. Работа с изображениями:
    * Все изменяемые медиа (изображения товаров, аватары пользователей и т.д.) должны храниться, создаваться и добавляться из сервиса Supabase:
      + При загрузке изображения, используйте хранилище файлов Supabase.
      + Получайте URL загруженного изображения.
      + Отображайте изображение, используя URL, полученный от Supabase.
      + Совет: Оптимизируйте размер загружаемых изображений, чтобы уменьшить трафик и улучшить производительность. Используйте механизмы кэширования изображений (если они не предоставляются библиотекой).
16. Обработка ошибок:
    * Необходимо корректно обрабатывать запросы к базе данных (Supabase API):
      + Перехватывайте ошибки при выполнении запросов к базе данных (например, при получении данных, сохранении данных, аутентификации).
      + Отображайте пользователю понятные сообщения об ошибках.
    * В случае получения ошибки от сервера или отсутствия соединения с сетью Интернет необходимо отобразить соответствующий текст ошибки в диалоговом окне (AlertDialog, Snackbar и т.д.), которое должно закрываться только пользователем (или через таймаут):
      + При отсутствии соединения: отобразите сообщение “Отсутствует подключение к интернету. Проверьте ваше соединение”.
      + При ошибке от сервера: отобразите информативное сообщение, содержащее информацию об ошибке, полученной от Supabase. Например: “Ошибка при загрузке данных. Пожалуйста, попробуйте позже.” или “Неверный email или пароль.”
      + Используйте диалоговые окна, которые требуют от пользователя подтверждения (нажатие кнопки “OK”, “Закрыть”), чтобы предотвратить непреднамеренное закрытие сообщения об ошибке.
      + Совет: Логируйте все ошибки (в консоль разработки, или в специальный лог-файл), чтобы облегчить отладку приложения.
17. Выход:
    * Выход из пользовательской сессии (logout):
      + Удалите токен аутентификации из памяти приложения.
      + Очистите данные, сохраненные в локальном хранилище (если использовалось “Запомнить меня”).
      + Перенаправьте пользователя на экран входа.
    * Удаление данных пользователя из локального хранилища по нажатию соответствующей кнопки:
      + Предоставьте возможность пользователю удалить все данные, связанные с его учетной записью, из локального хранилища. (например, в разделе “Настройки” -> “Очистить данные”). Это должно быть отдельной кнопкой, не относящейся к Logout.
      + При нажатии на эту кнопку, удалите всю информацию, хранящуюся в локальном хранилище.
      + Предупредите пользователя перед удалением данных!
18. Локализация:
    * Использовать ресурсы Android (strings.xml, и т.д.) там, где это возможно:
      + Используйте ресурсы Android для хранения строк (текстов) пользовательского интерфейса. Это упрощает процесс перевода и обновления текста.
    * Приложение должно быть локализовано на русский и английский язык:
      + Создайте отдельные файлы ресурсов для каждого языка (values-ru, values-en).
      + Переведите все строки пользовательского интерфейса на русский и английский языки.
      + Определите локаль устройства и используйте соответствующие ресурсы.
      + Совет: Рассмотрите возможность автоматизированного перевода с помощью API (например, Google Translate API) для экономии времени (с последующей ручной корректировкой).

## **Модуль 5 Документирование результатов разработки**

1. Создание отчета по выполненной работе:
   * Укажите цели и задачи:
     + Кратко опишите цели и задачи проекта (создание мобильного приложения для покупки мебели “Уютный Дом”).
   * Отобразите разработанные вами экраны и опишите их:
     + Предоставьте скриншоты всех экранов приложения (загрузочный экран, экраны авторизации, каталога, корзины, профиля и т.д.).
     + Под каждым скриншотом дайте краткое описание экрана и его основных элементов управления.
   * Укажите разработанные вами функции приложения:
     + Перечислите все реализованные функции, описанные в ТЗ (аутентификация, регистрация, просмотр каталога, фильтрация, сортировка, корзина, оформление заказа, личный кабинет, уведомления и т.д.).
     + Укажите, какие функции были реализованы в полном объеме, а какие частично или не реализованы (и почему).
     + Если были внесены отклонения от ТЗ, укажите их и обоснуйте.
2. Создание презентации. В презентации необходимо рассказать о реализации приложения:
   * Презентация должна быть рассчитана на разработчиков (Вы не должны продавать приложение!): Сфокусируйтесь на технических аспектах разработки.
   * О реализованных задачах:
     + Опишите реализованные функциональные возможности приложения.
     + Покажите основные экраны приложения (с акцентом на структуру, элементы управления, переходы между экранами).
   * Важные / интересные моменты проделанной работы:
     + Расскажите о наиболее сложных и интересных задачах, с которыми вы столкнулись.
     + Поделитесь опытом использования новых технологий, библиотек или инструментов.
     + Укажите, какие проблемы пришлось решать и как вы это сделали.
   * Возникшие трудности:
     + Опишите технические проблемы, с которыми вы столкнулись в процессе разработки.
     + Расскажите о найденных решениях и полученном опыте.
   * Используемые архитектурные решения:
     + Опишите архитектуру приложения (например, MVC, MVVM, или другую подходящую структуру).
     + Объясните, почему был выбран тот или иной подход.
     + Опишите структуру проекта (файловую структуру).
   * Используемые библиотеки и инструменты:
     + Перечислите все использованные библиотеки, фреймворки и инструменты разработки (например, React Native, Flutter, Android Studio, Xcode, Supabase, Figma).
     + Кратко расскажите о назначении каждой библиотеки/инструмента.
     + Укажите версии использованных инструментов.
3. Подготовка файлов для публикации приложения:

* файлы для публикации загружайте в папку «Publication» в основную ветку main;
* создайте установочный файл с названием «[первая буква фамилии латиницей]-Matule»;
* загрузите иконки приложения для магазина с размерами:
  + - 32х32;
    - 64х64;
    - 128х128.
* создайте текстовый документ с кратким описанием приложения;
* добавьте скриншоты экранов приложения в количестве 5 штук.