Государственное бюджетное профессиональное образовательное учреждение

«Нижегородский Губернский колледж»

**Выбор паттерна и программного обеспечения для разработки**

Информационной системы: «Make Yourself»

Разработчик\_\_\_\_\_\_\_\_\_Токарева Э.О. 02.10.2024

Заказчик\_\_\_\_\_\_\_\_\_\_\_\_Мамшева Ю. С. 02.10.2024

Нижний Новгород

2024

# ВЫБОРА ПАТТЕРНА ПРОЕКТИРОВАНИЯ

## Сравнение паттернов

### Model View Controller (MVC)

**Преимущества:**

* разделяет бизнес-логику в модели;
* поддерживает асинхронные методы;
* модификация не затрагивает всю модель;
* более быстрый процесс разработки.

**Недостатки:**

* из-за большого количества кода в контроллере он может быть захламлен;
* мешает модульному тестированию.

### Model-View-Presenter (MVP)

**Преимущества:**

* делает view абстрактной, так что бы ее можно было легко поменять;
* переиспользование View и Presenter;
* код более читабельный и удобный в сопровождении;
* простое тестирование, так как бизнес-логика отделена от пользовательского интерфейса.

**Недостатки:**

* тесная связь между View и Presenter;
* огромное количество интерфейсов для взаимодействия между слоями;
* размер кода довольно избыточен.

### Model-View-ViewModel (MVVM)

**Преимущества:**

* Разделение логики приложения и представления
* Гибкость и переиспользование
* Связывание данных
* Упрощение тестирования

**Недостатки:**

* Увеличение объема кода

## Выбор паттерна

Для разработки приложения выбран паттерн проектирования – MVVM, который в дальнейшем будет реализован с помощью библиотеки Hilt.

MVVM, рекомендованный Google для разработки Android-приложений, предлагает наилучшие возможности для двусторонней привязки данных, модульного тестирования и разделения логики.

Обоснование выбора:

* Паттерн обеспечивает чёткое разделение интерфейса, бизнес логики и хранения данных, тем самым предавая упорядоченность структуре проекта и упрощая разработку;
* Разделение ответственности компонентов системы упрощает их тестирование по отдельности;
* Разделение компонентов делает приложение более гибким и поддерживаемым.

# ВЫБОР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ РЕАЛИЗАЦИИ

## Мобильное приложение

Для реализации мобильного приложения было выбрано следующее ПО:

1. Среда разработки – Android Studio

* Android Studio полностью интегрирован с Android SDK, что обеспечивает быстрый доступ к инструментам и библиотекам для разработки Android-приложений;
* Поддержка большого количества плагинов;
* Возможность почти неограниченной работы с устройством, что сводит на нет проблему невозможности написания определённого функционала на Android Studio;
* Широкий набор интегрированных инструментов для отладки, возможность сборки проекта в реальном времени без перезапуска эмулятора;
* Обширное сообщество разработчиков с множеством обучающих материалов и плагинов;
* Android Studio поддерживается Google и имеет большую экосистему плагинов и документации, что облегчает разработку;

2. Фреймворк – Jetpack Compose

* Декларативный подход: Jetpack Compose позволяет создавать пользовательские интерфейсы декларативно, описывая структуру UI с помощью функций, что делает код более читаемым и легко изменяемым.
* Kotlin-only: Больше не нужно переключаться между классами и xml-файлами – вся работа с UI происходит в одном Kotlin-файле.
* Композитный подход: Наследованию – нет, композиции – да. Каждый UI-компонент представляет собой обычную composable-функцию, отвечающую только за ограниченный функционал, т.е. без лишней логики.
* Создание современных UI: Jetpack Compose предназначен для создания современных и динамичных UI с поддержкой анимаций и переходов.
* Простой в изучении: Jetpack Compose относительно прост в изучении, что сокращает время разработки и позволяет быстро приступить к работе.

## Серверная часть

Для реализации серверной части был выбран сервис Supabase:

* Сервис предоставляет полноценную платформу, включая базу данных (PostgreSQL), аутентификацию, хранилище файлов и API.
* Создание масштабируемых баз данных на PostgreSQL. Причем БД представлена в двух интерфейсах: в виде таблицы и в виде кода на SQL. Supabase берет на себя CRUD — функции создания, чтения, модификации и удаления элементов, предлагает API и больше 40 предустановленных расширений Postgres.
* Аутентификация. Система административного управления пользователями включена в каждый проект. Это позволяет пользователям мгновенно логиниться с использованием практически всех популярных соцсетей. Фича полностью интегрирована, то есть подключать сторонние ресурсы не надо, что повышает безопасность приложения.
* Хранение данных. Supabase предлагает масштабируемое хранилище для любых типов данных. Управлять можно через API, все интегрировано с другими сервисами платформы, а значит запросы проходят быстро.
* Простой в использовании интерфейс и хорошая документация, что сокращает время на настройку и разработку серверной части.
* Supabase имеет бесплатный план, что делает его доступным для небольших и учебных проектов.