**Государственное бюджетное профессиональное образовательное учреждение**

**«Нижегородский Губернский колледж»**

**ОТЧЕТ**

**По разработке мобильного приложения**

**«Выбор паттерна проектирования и ПО для реализации».**

Руководитель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ю. С. Мамшева  
 02.10.2024

Студент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Е. С. Колиниченко

02.10.2024

Специальность, группа: 09.02.07, 43П

Нижний Новгород

2024г.

1. **ВЫБОР ПАТТЕРНА ПРОЕКТИРОВАНИЯ**

Для мобильного приложения «Meow Mates2 был выбран паттерн проектирования MVVM (Model-View-ViewModel). Этот паттерн позволяет эффективно разделять логику приложения и пользовательский интерфейс, что особенно важно для мобильных приложений, где удобство взаимодействия с пользователем имеет первостепенное значение.

Обоснование выбора MVVM:

1. Четкое разделение ответственности:

* Model отвечает за данные и бизнес-логику приложения (например, информация о питомцах, пользователях и их взаимодействиях).
* View представляет пользовательский интерфейс (UI), который отображает данные и реагирует на действия пользователя.
* ViewModel служит связующим звеном между Model и View, обеспечивая привязку данных и управление состоянием UI.

1. Упрощение тестирования: Паттерн MVVM позволяет легко тестировать бизнес-логику, так как ViewModel может быть протестирован независимо от View. Это упрощает процесс написания юнит-тестов.
2. Поддержка реактивного программирования: Использование LiveData и других реактивных компонентов в Android позволяет View автоматически обновляться при изменении данных в ViewModel, что улучшает пользовательский опыт.
3. Удобство разработки: MVVM хорошо интегрируется с библиотеками Android Jetpack, такими как LiveData и ViewModel, что упрощает разработку и уменьшает количество кода.
4. **ВЫБОР ДЛЯ РЕАЛИЗАЦИИ ПО**

Для реализации мобильного приложения «Meow Mates» были выбраны следующие технологии:

**2.1. Язык программирования: Kotlin**

Обоснование выбора:

* Kotlin является официальным языком для разработки под Android, что обеспечивает хорошую поддержку со стороны Google.
* Он предлагает лаконичный синтаксис и современные возможности, такие как расширения функций и корутины для асинхронного программирования.
* Kotlin совместим с Java, что позволяет использовать существующие библиотеки и фреймворки.

**2.2. Среда разработки: Android Studio**

Обоснование выбора:

* Android Studio предоставляет мощные инструменты для разработки, отладки и тестирования приложений.
* Она включает в себя встроенные средства для работы с Gradle, что упрощает управление зависимостями и сборкой проекта.
* Поддерживает интеграцию с Git для управления версиями кода.

**2.3. Хранилище данных: Supabase**

Обоснование выбора:

* Supabase предоставляет облачное решение для хранения данных с поддержкой PostgreSQL, что обеспечивает надежность и масштабируемость.
* Он предлагает простую интеграцию с приложениями на Kotlin через RESTful API или библиотеку клиентского SDK.
* Supabase включает функции аутентификации и управления пользователями, что упрощает реализацию безопасности в приложении.

**2.4. UI-фреймворк: Jetpack Compose**

Обоснование выбора:

* Jetpack Compose позволяет создавать пользовательские интерфейсы декларативным способом, что упрощает разработку и поддержку UI.
* Он интегрируется с другими компонентами Jetpack, такими как LiveData и ViewModel, что делает его идеальным выбором для использования с паттерном MVVM.
* Compose обеспечивает высокую производительность и возможность кастомизации интерфейса без необходимости написания большого объема кода.