Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»
(Университет ИТМО)

Факультет **Прикладной информатики**

Направление подготовки **09.03.03 Прикладная информатика**

Образовательная программа **Мобильные и сетевые технологии**

К У Р С О В О Й П Р О Е К Т

Тема: Организация мероприятий или участие

Обучающийся: Гриценко Ярослав Александрович К3141

Санкт-Петербург 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3

1 Суть проекта 4

2 Процессы работы над проектом 6

3 Поставленные передо мной задачи 9

4 Решение поставленных передо мной задач 10

5 Анализ индивидуальной работы 14

6 Анализ взаимодействия с командой и руководителем проекта 15

7 Оценка работы руководителя 16

ЗАКЛЮЧЕНИЕ 17

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 18

ВВЕДЕНИЕ

В современном мире, где технологии стремительно развиваются, а общественные связи становятся все более важными, организация мероприятий и участие в них приобретают особое значение. С каждым годом количество различных событий — от культурных и образовательных до спортивных и деловых — растет, что создает потребность в удобных и эффективных инструментах для их организации и управления.

В этой связи разработка мобильного приложения для платформ Android и iOS (имея в виду распространенность обеих операционных систем), которое позволит пользователям создавать, редактировать и публиковать мероприятия, а также подавать заявки на участие в них, становится актуальной задачей.

Одной из ключевых особенностей такого приложения должна стать его многофункциональность. Пользователи должны иметь возможность не только организовывать собственные мероприятия, но и находить интересные события в своем регионе или за его пределами.

Это создаст платформу для обмена информацией и идеями, а также позволит людям легче находить единомышленников и участвовать в мероприятиях, которые соответствуют их интересам.

Важно отметить, что приложению следует быть ориентированным как на организаторов, так и на участников, что сделает его универсальным инструментом для всех заинтересованных сторон.

Разработка такого программного продукта - комплексный процесс, требующий вдумчивого поэтапного проектирования, разумного выбора технологий, инструментов разработки для того, чтобы обеспечить легкую расширяемость и возможность интеграции сторонних сервисов (например,

сервисов карт - Яндекс. Карты, Google Карты, 2ГИС и прочих или сервисов прогноза погоды), и самое главное - времени.

Последнее, как известно, всегда в дефиците, поэтому реализовать по-настоящему большую систему в сжатые сроки - задача крайне амбициозная и обреченная на провал.

Понимая это обстоятельство, мы решили разработать приложение, содержащее лишь часть возможного функционала.

1 Суть проекта

В рамках этой работы мы занимались разработкой приложения, которое помогает организовывать мероприятия и систематизировать информацию о них. Приложение предназначено как для обычных пользователей — участников, волонтёров и сотрудников мероприятия, так и для организаторов.

На момент сдачи курсовой работы в проекте были реализованы следующие технические элементы:

1. экраны входа и регистрации;
2. экран списка событий;
3. экран с подробной информацией о событии;
4. экран создания события;
5. экран профиля пользователя;
6. экран редактирования профиля.

К сожалению, у нас не было достаточно времени, чтобы завершить проект, и остались задачи, которые мы хотели бы выполнить и сохранить в портфолио. Среди них — интеграция с серверной частью для хранения аутентификационных данных, информации о мероприятиях и пользователях.

Остальные аспекты, которые должны быть описаны в сути проекта, такие как его актуальность, цель и задачи, уже были подробно рассмотрены во введении.

2 Процессы работы над проектом

Для более структурированной работы, процесс разработки был разделен на несколько блоков, показанных в таблице 1.

Этап	Сроки выполнения
Анализ требований	1.11.2024 – 10.11.2024
Изучение литературы	1.11.2024 – 24.11.2024
Проектирование UI	25.11.2024 – 8.12.2024
Разработка приложения	25.11.2024 – 16.12.2024
Тестирование	16.12.2024 – 18.12.2024

Таблица 1 – Этапы работы над проектом и их временные рамки

Техническое задание было составлено руководителем проекта, однако команда принимала посильное участие в внесении изменений в предоставленные им версии технического задания для улучшения описательных свойств, достижения понятности текста, которые способствовали бы снижению риска коллизий при разработке, а также снижали трудозатраты на объективную оценку соответствия проекта запланированным в начале работы требованиям.

Связь между членами команды и руководителем проекта поддерживалась через мессенджер Telegram.

Техническое задание хранилось на платформе Odoo, позволяющей удобно отслеживать этапы разработки всем заинтересованным лицам (участникам проекта и преподавателям).

Для разработки дизайна была использована Figma.

Для разработки приложения был использован редактор кода Android Studio.

3 Поставленные передо мной задачи

В самом начале, имея в виду имеющийся у меня опыт Android-разработки, было решено, что я буду помогать с координацией работы Android-команды, консультированием по техническим вопросам, связанным с Android-разработкой, составлением списка необходимых учебных материалов для других членов Android-команды, а также выполнением непосредственно проектных задач.

Среди таких задач были:

- Составление списка используемых технологий
- Начальное конфигурирование проекта
- Разработка основных классов и структур данных для повсеместного использования в проекте
- Разработка базовых ui-компонентов
- Разработка экрана логина
- Разработка экрана списка событий

4 Решение поставленных передо мной задач

Итак, начнем по-порядку.

Взаимодействие с Android-командой действительно зачастую происходило с моим участием, поскольку я постоянно имел актуальные сведения о текущих статусах загруженности ребят, так как мы часто обсуждали разные технические детали и тонкости, связанные с выполнением поставленных задач.

Помимо этого, я также составил список полезных для ознакомления материалов по теме Android-разработки и языка программирования Kotlin. Он полностью включен в список источников.

Составление первоначального списка используемых технологий - важный этап разработки программного продукта, так как он влияет на особенности реализации тех или иных элементов системы.

Осознавая краткость предоставленного для работы над времени, я первым делом спросил у других разработчиков с чем им приходилось работать, чтобы выбрать наиболее знакомый для всех вариант технологического стека.

Таким образом, я остановился на следующих технологиях:

- Retrofit 2 - для работы с сетью
- Jetpack Compose - для верстки UI
- Decompose - для навигации между экранами, компонентизации и менеджмента состояний бизнес-логики приложения
- Koin - для реализации инъекции зависимостей
- Kotlin serialization - для преобразования данных в программные объекты и наоборот
- Coil - для загрузки и отображения изображений
- Google Firebase - для взаимодействия с облачными сервисами Firebase от Google

Затем мной было проведено начальное конфигурирование проекта. Для реализации небольших проектов или небольших частей масштабных проектов по моему опыту можно в угоду экономии времени отказаться от всеобщей модуляризации и свести проект всего лишь к трём модулям:

- app - модуль, содержащий входную точку
- core - модуль со вспомогательными инструментами, общими функциями и решениями
- feature - модуль, содержащий все “фичи” проекта

На рисунке 1 можно увидеть как это выглядит из редактора кода Android Studio.

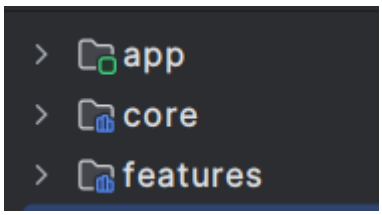

Рисунок 1 - Демонстрация разделения проекта на модули

Также начальное конфигурирование включает в себя описание зависимостей проекта, что также мной было сделано в стандартном виде - в файлах build.gradle.kts.

Еще одна из моих задач - разработка основных классов и структур данных для повсеместного использования в проекте.

В рамках неё я создал корневые сущности бизнес-логики, в чьи задачи входили, например, предоставление информации об авторизованности пользователя, предоставление информации о данных текущего пользователя; механизм, позволяющий ловить всевозможные программные ошибки для дальнейшей возможной их обработки на ui.

Другая задача - разработка базовых ui-компонентов. В рамках нее были созданы базовое поле для ввода текста, гибко настраиваемое извне, базовая кнопка, а также специальный элемент для показа так называемого LCE-

состояния - в зависимости от поступающих на вход данных, он показывает ошибку, загрузку или успешно загруженные данные.

Поскольку мы стараемся придерживаться современных кодовых практик и подходов, в своём проекте мы использовали паттерн Clean Architecture.

Каждая “фича” была разделена на три пространства:

- presentation - всё, что связано с ui
- domain - всё, что связано с бизнес-логикой
- data - всё, что связано с сетью и хранением данных

Следуя такой организации кода, для каждого экрана нужно создать специальные сущности:

- Для хранения общих данных
- Для взаимодействия с сетью
- Для отображения ui
- Для хранения ui-состояния

Итак, зная всё это, я с легкостью решил задачу создания экрана для авторизации пользователя (логин).

Первым делом, я создал абстракции для компонента экрана и репозитория (сущность, которая отвечает за взаимодействие с данными), а также domain-модели.

Затем принялся за реализацию этих интерфейсов.

На определенном этапе разработки мы поняли, что не успеваем добавить интеграцию Firebase, поэтому все реализации репозиториев, которые попали в финальное приложение были моковыми - хранили данные только в рамках одного сеанса использования приложения.

По этой причине я реализовал только моковый репозиторий, однако такой подход позволяет в будущем с легкостью переключиться на реализацию с Firebase.

Также я реализовал интерфейс компонента и сделал верстку ui. На рисунке 2 видно как получился экран логина в стандартном состоянии.

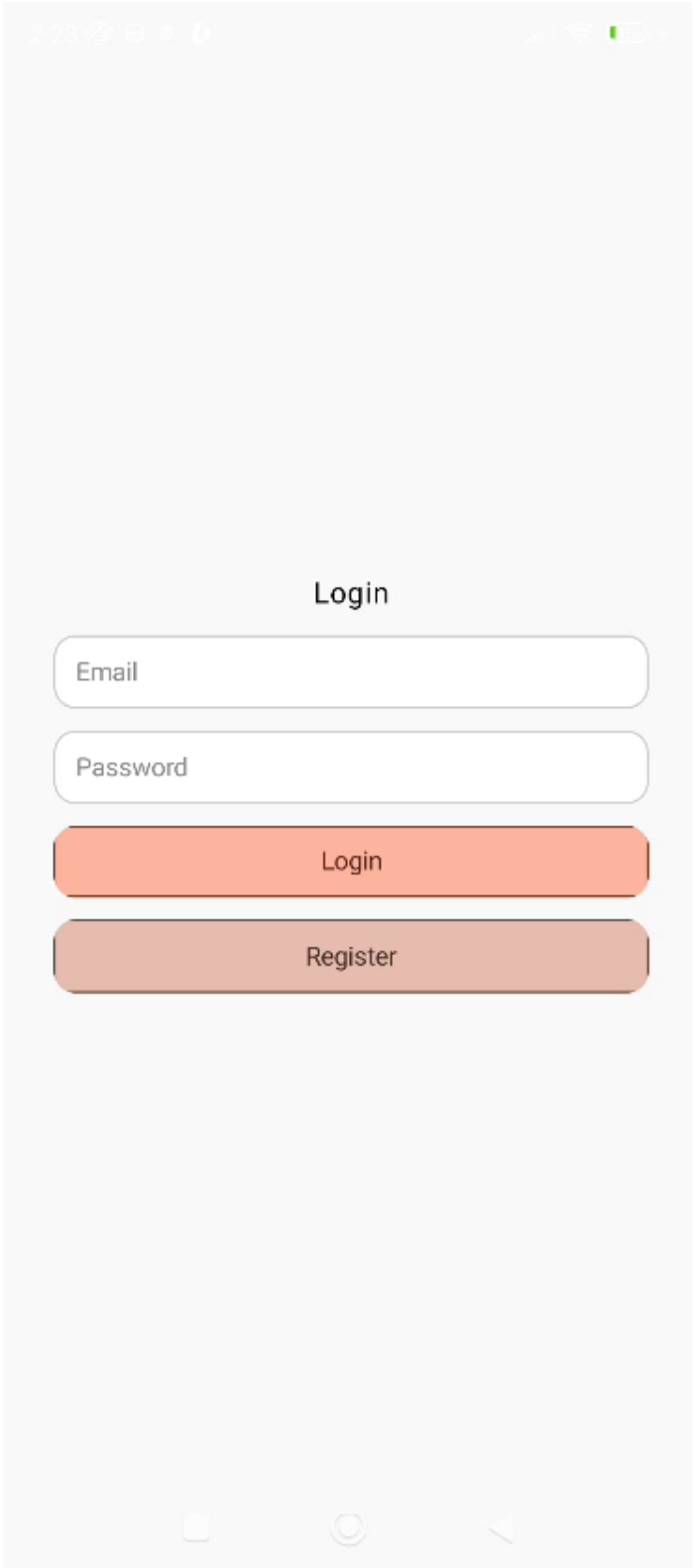

Рисунок 2 - Экран логина

С экрана авторизации можно перейти на регистрацию, нажав кнопку с текстом “Register”.

Также на нём сделана валидация почты - если ввести её в неправильном формате, то кнопка “Login” станет неактивной, а рядом с полем ввода появится ошибка, как показано на рисунке 3.

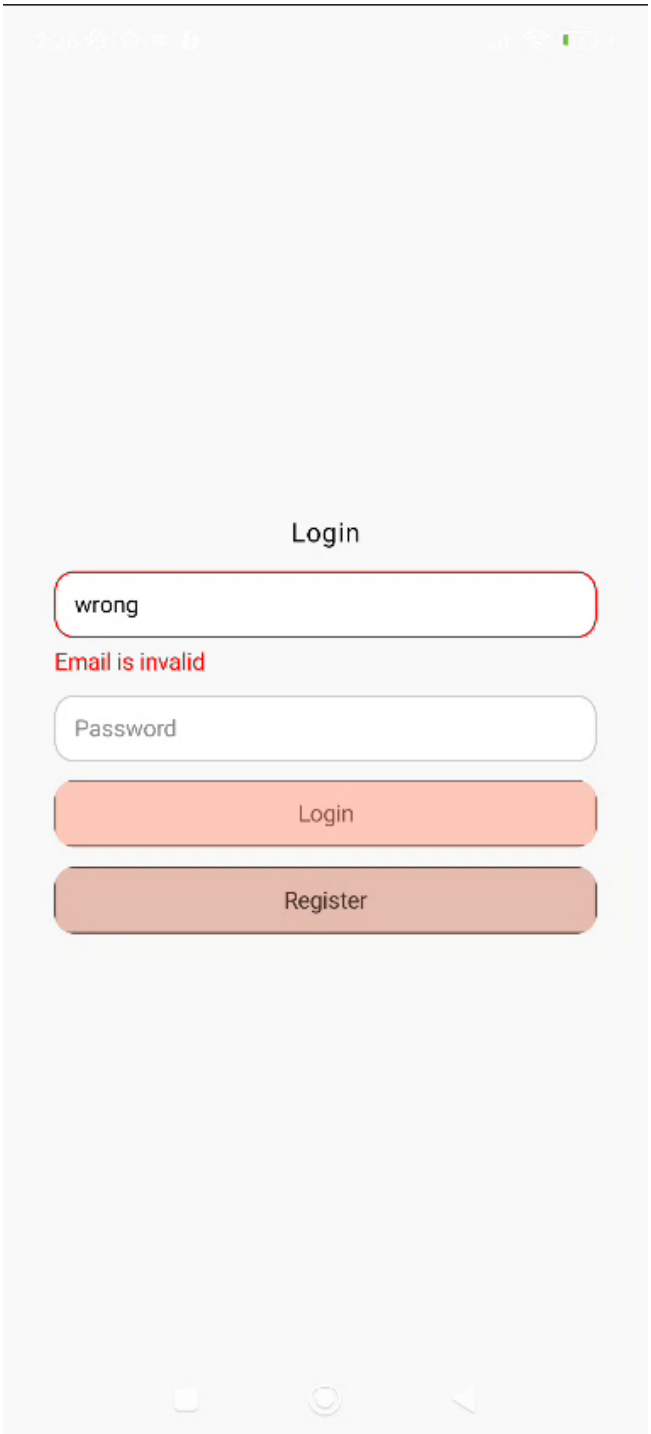

Рисунок 3 - Ошибка валидации поля почты при на экране авторизации

5 Анализ индивидуальной работы

Помимо сжатых сроков особых трудностей при работе над проектом у меня не возникло.

Мне было интересно обсуждать технические вопросы с членами нашей команды и составлять список материалов для подготовки - таким образом я и сам освежил свои знания по Android-разработке, языку программирования Kotlin, компонентному подходу к разработке приложений, архитектуре и многом другом.

Практика же разработки самого программного продукта была несложной и иногда не очень захватывающей - всё же мы разрабатывали достаточно типовой продукт, и все сложные механизмы закладывались как перспективы для дальнейшего развития.

Тем не менее, мне довелось поизучать Firebase, хоть мы и в итоге решили не включать его в конечный продукт.

6 Анализ взаимодействия с командой и руководителем проекта

Мое взаимодействие с командой было достаточно активным. Мы часто разговаривали о проекте и особенностях реализации тех или иных технических механизмов.

Однажды даже созвонились для того, чтобы я ответил на интересующие вопросы и рассказал немного про технологии, незнакомые для других.

Имелось место и личному общению.

С руководителем проекта мы были на связи, он постоянно интересовался о статусе проекта.

7 Оценка работы руководителя

Руководитель нашего проекта достаточно хорошо справился со своей задачей. Он помогал iOS команде с реализацией приложения на их стороне, постоянно спрашивал о статусах задач, удостоверивался, что разработка идёт штатно и без эксцессов.

ЗАКЛЮЧЕНИЕ

Подводя итоги проекта, я могу сказать, что пусть у нас не получилось реализовать всю первоначально заявленную функциональность, тем не менее мы значительно продвинулись на пути к конечному результату.

Работа над проектом была полезна для всех нас - я получил опыт координации действий небольшой команды разработчиков, опыт обучения других и проектирования архитектуры приложения с нуля, а ребята поработали над проектом, который является пусть и приближением, но приближением реального проекта, познакомились с тем, как процесс разработки устроен в командах на настоящих предприятиях, а также “пощупали” технологии, которые являются передовыми решениями в сфере разработки приложения для операционной системы Android.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Лекция по многопоточности в Java - https://www.youtube.com/watch?v=ShzQJUFzq58

2. Продолжение лекции по многопоточности в Java - https://www.youtube.com/watch?v=5YLA29EybMo&list=PLlb7e2G7aSpTCB2OxGlezpgOXwq4xer7Z&index=14

3. Лекция по корутинам в языке программирования Kotlin - https://www.youtube.com/watch?v=ITLe4FIrrTg&list=PLIh9yLdjK2YeRLnD-gJyVWIq_w-7OMv8r&index=7

4. Лекции от Яндекса по Android-разработке - https://www.youtube.com/watch?v=Yz2K1p8OJ1o&list=PLIh9yLdjK2YeRLnD-gJyVWIq_w-7OMv8r

5. Курс по корутинам в языке программирования Kotlin от Кирилла Розова - https://www.youtube.com/watch?v=mD1r9zIwHbs&list=PL0SwNXKJbuNmsKQW9mtTSxNn00oJlYOLA

6. Курс по Android-разработке от Андрея Сумина - https://stepik.org/course/121507/promo

7. Курс по профессиональной Android-разработке от Андрея Сумина - https://stepik.org/course/117314/promo

8. Курс по Jetpack Compose от Андрея Сумина - https://stepik.org/course/125685/promo

9. Курс от Google по Jetpack Compose и Android - https://developer.android.com/courses/android-basics-compose/course

10. Документация языка программирования Kotlin - https://kotlinlang.org/docs/home.html

10. Документация языка программирования Kotlin на русском языке - https://kotlinlang.ru/docs/kotlin-doc.html

11. Статья про Clean Architecture - https://habr.com/ru/companies/mobileup/articles/335382/

12. Статья про MV* паттерны - https://habr.com/ru/companies/mobileup/articles/313538/

13. Документация библиотеки для инъекции зависимостей Koin - https://insert-koin.io/

14. Документация библиотеки Decompose - https://arkivanov.github.io/Decompose/

15. Гитхаб библиотеки Decompose (установка и версии) - https://github.com/arkivanov/Decompose

16. Гитхаб Kotlin serialization (установка и версии) - https://github.com/Kotlin/kotlinx.serialization

17. Документаця Kotlin serialization - https://github.com/Kotlin/kotlinx.serialization/blob/master/docs/serialization-guide.md

18. Документация мультиплатформенной библиотеки для загрузки изображений Coil - https://coil-kt.github.io/coil/

19. Гитхаб мультиплатформенной библиотеки для загрузки изображений Coil - https://github.com/coil-kt/coil

ПРИЛОЖЕНИЕ 1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

1. Наименование проекта: Мобильное приложение для организации мероприятий
2. Цель: Создать прототип мобильного iOS/Android приложения для организации мероприятий и участия в них.
3. Сроки выполнения: 1.11.2024 – 18.12.2024
4. Руководитель проекта: Бацанов Артем Александрович
5. Термины и сокращения:

«ОС» - Операционная система;

«UI» - пользовательский интерфейс;

«Android» – ОС для смартфонов и планшетов от Google;

«iOS» – ОС для смартфонов и планшетов от Apple;

«Firebase» – платформа, набор инструментов и сервисов для разработки аутентификационной компоненты приложения;

«Мероприятие» – совокупность действий, нацеленных на выполнение единой задачи. Дополнительно может подразумевать наличие списка участников и некоторого плана мероприятия;

6. Технические требования

Таблица 1 – Общие требования к разрабатываемой системе

Требование	**Способ реализации**	**Используемые инструменты**	**Потребитель**
Кроссплатформенность	Разработка двух приложений под Android и iOS	Kotlin / Swift	Все пользователи
Возможность входа по электронной	Реализация способа взаимодействи	Firebase	Разработчики и пользователи

почте	я с базой данных		
Планирование и просмотр мероприятий: описание, дата, время, место	Создание соответствующих экранов в приложении	Kotlin / Swift	Все пользователи
Редактирование и просмотр профиля, просмотр списка созданных и посещенных мероприятий	Создание соответствующих экранов в приложении	Kotlin / Swift	Все пользователи

Таблица 2 – Функциональные требования к системе

Требование	**Способ реализации**	**Используемые инструменты**	**Потребитель**
Если пользователь авторизован, то отображается главный экран со списком всех мероприятий и навигационной панелью	Создание зависимостей компонентов	Kotlin / Swift	Все пользователи
Экран создания встречи имеет форму с обязательными полями информации о мероприятии	Создание соответствующих элементов дизайна	Figma	Все пользователи

При клике на мероприятие отображается подробная информация о встрече, список участников	Создание зависимостей компонентов	Figma; Kotlin / Swift;	Все пользователи

7. Содержание работы

Таблица 3 – Этапы разработки

Этап	**Сроки выполнения**
Анализ требований	1.11.2024 – 10.11.2024
Изучение литературы	1.11.2024 – 24.11.2024
Проектирование UI	25.11.2024 – 8.12.2024
Разработка приложения	25.11.2024 – 16.12.2024
Тестирование	16.12.2024 – 18.12.2024

8. Основные результаты работы и формы их представления

Таблица 4 – список результатов работы и способов их представления

Результат работы	**Форма представления**
Техническое задание	Файл технического задания оформленный по ГОСТу в формате “docx” или “pdf”
Проект UI	Макет ключевых экранов приложения
Экран входа/регистрации	Набор данных (package) содержащий ключевые файлы обеспечивающие работу этого фрагмента приложения
Экран основного меню	
Экран создания мероприятия	

Экран информации о мероприятии	
Экран профиля	
Экран редактирования профиля	
Презентация	Подготовленный файл презентации в формате “pptx” или подобном с планом выступления