**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ АМУРСКОЙ ОБЛАСТИ**

**ГОСУДАРСТВЕННОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**АВТОНОМНОЕ УЧРЕЖДЕНИЕ АМУРСКОЙ ОБЛАСТИ**

**«АМУРСКИЙ АГРАРНЫЙ КОЛЛЕДЖ**»  
  
Информационные системы и программирование  
  
  
**КУРСОВАЯ РАБОТА**   
на тему:   
**РАЗРАБОТКА ЛОКАЛЬНОЙ БАЗЫ ДАННЫХ SQLITE ДЛЯ УЧЕТА ДАННЫХ ПОЛЬЗОВАТЕЛЕЙ**

**Автор работы:**   
студент группы ИП31 очной  
формы обучения Янченко К.В.  
Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
**Руководитель работы**:  
Преподаватель: Храпов А.Б.

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г.

Благовещенск 2024

Содержание

[ВВЕДЕНИЕ 2](#_Toc1)

[1. Актуальность темы 2](#_Toc2)

[2. Определение целей и задач работы 3](#_Toc3)

[3. Описание структуры работы 3](#_Toc4)

[ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАЗРАБОТКИ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ 4](#_Toc5)

[4. Обзор технологий и инструментов разработки мобильных приложений в Intelij IDEA 4](#_Toc6)

[5. Обзор подходов к управлению данными в мобильных приложениях 5](#_Toc7)

[6. Обзор системы управления базами данных SQLite в мобильном приложении на Java. 7](#_Toc8)

[ОБЛАЧНЫЕ ТЕХНОЛОГИИ В РАЗРАБОТКЕ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ 9](#_Toc9)

[7. Обзор облачных сервисов Firebase 9](#_Toc10)

[8. Описание возможностей Firebase для хранения и управления данными в мобильном приложении 11](#_Toc11)

[РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ МЕНЕДЖМЕНТА ТАБЛИЦ EXCEL 12](#_Toc12)

[9. Постановка задачи и описание функциональности приложения 12](#_Toc13)

[10. Описание архитектуры приложения 14](#_Toc14)

[11. Описание процесса разработки и тестирования 14](#_Toc15)

[ЭКОНОМИЧЕСКАЯ ЭФФЕКТИВНОСТЬ ИСПОЛЬЗОВАНИЯ РАЗРАБОТАННОГО ПРИЛОЖЕНИЯ 17](#_Toc16)

[ПРИЛОЖЕНИЯ 20](#_Toc17)

### ВВЕДЕНИЕ

#### Актуальность темы

В нашем современном мире на предприятиях существует проблема:  
Проблема захламленности Excel таблиц является распространенной проблемой в современном бизнесе. Она возникает из-за того, что многие компании используют Excel для хранения и обработки больших объемов данных, но часто делают это неправильно. Например, они могут использовать слишком много формул и функций, что затрудняет чтение и понимание данных. Кроме того, они могут сохранять данные в разных местах и не иметь единой системы управления этими данными.

Это приводит к тому, что сотрудники тратят много времени на поиск нужной информации и исправление ошибок в таблицах. Также это может привести к потере важных данных или неправильной интерпретации результатов анализа.

От этой проблемы страдают различные отрасли бизнеса: финансы, здравоохранение, образование и другие. Они нуждаются в эффективном инструменте для управления своими данными и предотвращения потерь информации.

#### Определение целей и задач работы

Цель данной курсовой работы: Решение проблемы захламленности Excel таблиц на предприятиях. Улучшение эффективности работы сотрудников работающих непосредственно с Excel таблицами. Снижение времени, затраченного сотрудниками на работу с Excel таблицами.

Задачи по решению проблемы захламленности Excel таблиц на предприятиях: Разработать мобильное приложение для эффективного использования Excel таблиц внутри предприятий, которое должно снижать затраты по времени на хранение и перенос Excel таблиц внутри предприятий.

#### Описание структуры работы

Разработать непосредственно мобильное приложение с использованием общего для всех пользователей облачного хранилища и локальной базы данных SQLite для резервного копирования Excel таблиц в формате csv используя среду для разработки Intelij IDEA.

### ТЕОРЕТИЧЕСКИЕ ОСНОВЫ РАЗРАБОТКИ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

#### Обзор технологий и инструментов разработки мобильных приложений в Intelij IDEA

IntelliJ IDEA - это мощная интегрированная среда разработки (IDE), предлагающая широкий спектр инструментов и технологий для создания мобильных приложений. Вот краткий обзор некоторых ключевых технологий и инструментов, доступных в IntelliJ IDEA для разработки мобильных приложений:

1. Android Studio: IntelliJ IDEA тесно интегрирована с Android Studio, официальной IDE для разработки Android-приложений. Это позволяет разработчикам использовать все преимущества IntelliJ IDEA, такие как умное автодополнение кода, рефакторинг, интеграция с системами контроля версий и многое другое, вместе с расширенными возможностями Android Studio, такими как Android SDK Manager, эмулятор Android и инструменты для тестирования.
2. Kotlin: Kotlin - это современный язык программирования, который официально поддерживается Google для разработки Android-приложений. IntelliJ IDEA предлагает полную поддержку Kotlin, включая встроенную компиляцию, отладку и интеграцию с другими языками, такими как Java.
3. Swift: Для разработки iOS-приложений IntelliJ IDEA предлагает поддержку Swift через плагин, такой как JetBrains' AppCode. Это позволяет разработчикам использовать IntelliJ IDEA для создания и отладки приложений на Swift.
4. Xamarin: Xamarin - это платформа для разработки кроссплатформенных мобильных приложений с использованием C# и .NET. IntelliJ IDEA поддерживает Xamarin через плагин, позволяющий разработчикам создавать приложения для Android, iOS и других платформ.
5. Flutter: Flutter - это кроссплатформенный фреймворк от Google для создания мобильных приложений с использованием Dart. IntelliJ IDEA поддерживает Flutter через плагин, предоставляя разработчикам возможность создавать приложения для Android и iOS с единой кодовой базой.
6. Java: Хотя Java больше не является основным языком для разработки Android-приложений, она все еще широко используется в различных аспектах разработки, таких как создание серверных API и сервисов. IntelliJ IDEA предлагает полную поддержку Java, включая расширенные возможности редактирования кода и отладки.
7. Gradle: Gradle - это популярный инструмент сборки для Android-разработки. IntelliJ IDEA глубоко интегрирована с Gradle, что позволяет разработчикам легко управлять зависимостями, конфигурацией сборки и процессом сборки проектов.

#### Обзор подходов к управлению данными в мобильных приложениях

1. Локальное хранилище: Данные могут храниться локально на устройстве пользователя. Это может быть сделано с помощью встроенных механизмов, таких как SQLite для Android и Core Data для iOS. Этот подход обеспечивает быстрый доступ к данным без необходимости подключения к интернету и сохраняет данные даже после закрытия приложения. Однако локальное хранилище ограничено объемом памяти устройства и не подходит для обмена данными между несколькими устройствами.
2. Серверное хранилище: Данные могут храниться на удаленном сервере, к которому приложение обращается через интернет. Это позволяет легко обмениваться данными между различными устройствами и обеспечивает централизованное управление данными. Однако этот подход требует постоянного подключения к интернету и может быть медленным, если соединение слабое.
3. Кроссплатформенные базы данных: Некоторые фреймворки, такие как Realm, предлагают кроссплатформенное решение для управления данными, которое работает как локально, так и через сервер. Это позволяет разработчикам использовать единый API для работы с данными независимо от платформы.
4. NoSQL базы данных: NoSQL базы данных предоставляют гибкие схемы хранения данных, которые хорошо подходят для мобильных приложений, где данные могут быть неструктурированными или полуструктурированными. Они могут использоваться как локально, так и на сервере.
5. Cloud Firestore: Cloud Firestore - это база данных, предлагаемая компанией Google, которая позволяет хранить данные в реальном времени на серверах Google Cloud Platform. Она предлагает гибкую модель документации и транзакций, а также масштабируемость и надежность.
6. Sync Framework: Sync Framework - это технология, которая позволяет синхронизировать данные между локальным хранилищем и сервером. Это позволяет приложениям работать оффлайн, а затем синхронизировать изменения с сервером при восстановлении соединения.
7. Firebase Storage - это облачное хранилище от компании Google, предназначенное для хранения файлов, связанных с мобильными приложениями. Является мощным инструментом для управления файлами в мобильных приложениях и может быть эффективно использован для решения задач, связанных с загрузкой, скачиванием и управлением файлами.

#### Обзор системы управления базами данных SQLite в мобильном приложении на Java.

SQLite - это встраиваемая система управления базами данных, которая может быть полностью размещена в небольшой исполняемый файл. Она не требует отдельного сервера и может быть включена непосредственно в приложение. SQLite является популярным выбором для мобильных приложений, поскольку она легковесна, проста в использовании и не требует сложной настройки.

Особенности SQLite в контексте мобильных приложений на Java:

Простота использования: SQLite предоставляет простой SQL-подобный интерфейс для работы с базой данных. Это означает, что разработчики, знакомые с SQL, могут легко начать работу с SQLite.

Не требуется настройка: Поскольку SQLite является встраиваемой системой, она не требует установки дополнительного программного обеспечения или сервера. Это упрощает процесс разработки и развертывания.

Легковесность: SQLite занимает очень мало места и может быть легко включена в любое приложение. Это делает ее идеальной для мобильных устройств с ограниченными ресурсами.

Автоматическое управление памятью: SQLite управляет всеми ресурсами, связанными с базой данных, что освобождает разработчика от необходимости вручную управлять памятью.

Поддержка транзакций: SQLite поддерживает транзакции, что позволяет выполнять группы операций как единое целое или откатывать их в случае ошибки.

Поддержка ACID: SQLite обеспечивает атомарность, согласованность, изоляцию и долговечность (ACID), что гарантирует надежность и целостность данных.

Поддержка множества языков программирования: SQLite поддерживает множество языков программирования, включая Java, что позволяет разработчикам выбирать наиболее подходящий для их проекта.

Работа в оффлайн-режиме: Поскольку SQLite хранит данные локально на устройстве, приложения могут работать в оффлайн-режиме, а затем синхронизировать данные с сервером при наличии подключения к интернету.

Для работы с SQLite в Java обычно используются библиотеки, такие как JDBC или ORM-фреймворки, такие как Hibernate. Они предоставляют удобный способ взаимодействия с базой данных и позволяют разработчикам использовать стандартные SQL-команды для выполнения операций с данными.

### ОБЛАЧНЫЕ ТЕХНОЛОГИИ В РАЗРАБОТКЕ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

#### Обзор облачных сервисов Firebase

1. Firebase Analytics: Firebase Analytics предоставляет инструменты для отслеживания и анализа поведения пользователей в вашем приложении. Вы можете увидеть, как пользователи взаимодействуют с вашим приложением, узнать, какие функции наиболее популярны, и определить области для улучшения.
2. Firebase Authentication: Firebase Authentication позволяет легко добавить аутентификацию в ваше приложение. Сервис поддерживает множество методов аутентификации, включая Google, Facebook, Twitter, GitHub и Email/Password.
3. Firebase Realtime Database: Firebase Realtime Database - это облачная база данных, которая позволяет быстро и легко хранить и синхронизировать данные в реальном времени между клиентами и серверами.
4. Firebase Cloud Firestore: Cloud Firestore - это документ-ориентированная база данных, которая предлагает более сложные схемы и индексы, чем Realtime Database. Она лучше подходит для приложений, требующих сложных запросов и транзакций.
5. Firebase Storage: Firebase Storage предоставляет безопасное облачное хранилище для файлов, связанных с вашими приложениями. Вы можете загружать, скачивать и управлять файлами любого типа.
6. Firebase Hosting: Firebase Hosting - это сервис для хостинга статических веб-сайтов. Он автоматически оптимизирует ваш сайт для быстрой загрузки и поддерживает SSL-шифрование.
7. Firebase Messaging: Firebase Messaging позволяет отправлять push-уведомления пользователям ваших приложений. Вы можете отправлять уведомления в реальном времени, чтобы информировать пользователей о событиях, обновлениях и специальных предложениях.
8. Firebase Crashlytics: Firebase Crashlytics помогает обнаруживать и устранять сбои в вашем приложении. Вы получаете информацию о сбоях в реальном времени, что позволяет быстро реагировать и улучшать качество приложения.
9. Firebase Test Lab: Firebase Test Lab предоставляет инструменты для автоматизированного тестирования вашего приложения на различных устройствах и конфигурациях.
10. Firebase Performance Monitoring: Firebase Performance Monitoring помогает отслеживать производительность вашего приложения и находить узкие места.
11. Firebase Remote Config: Firebase Remote Config позволяет динамически изменять определенные параметры вашего приложения без необходимости выпуска новой версии.
12. Firebase Dynamic Links: Firebase Dynamic Links помогают направлять пользователей на соответствующие страницы в вашем приложении или на вашем сайте.
13. Firebase ML Kit: Firebase ML Kit предоставляет инструменты машинного обучения для распознавания объектов, лиц, текста и других задач.
14. Firebase Cloud Functions: Firebase Cloud Functions позволяет запускать код в ответ на определенные события в вашем приложении или базе данных.
15. Firebase App Indexing: Firebase App Indexing помогает улучшить видимость вашего приложения в результатах поиска Google.

#### Описание возможностей Firebase для хранения и управления данными в мобильном приложении

Firebase Realtime Database: Это облачная база данных, которая позволяет хранить данные в формате JSON и обеспечивает их синхронизацию в реальном времени между клиентами и серверами. Все изменения, сделанные на одном устройстве, мгновенно становятся видны всем подключенным клиентам.

Firebase Cloud Firestore: Это документ-ориентированная база данных, которая предлагает более сложные схемы и индексы, чем Realtime Database. Она лучше подходит для приложений, требующих сложных запросов и транзакций.

Firebase Storage: Это облачное хранилище для файлов, связанных с вашими приложениями. Вы можете загружать, скачивать и управлять файлами любого типа.

Firebase Cloud Functions: Это сервис, который позволяет запускать код в ответ на определенные события в вашем приложении или базе данных. Вы можете использовать его для автоматизации процессов, связанных с обработкой и управлением данными.

Firebase Remote Config: Это инструмент, который позволяет динамически изменять определенные параметры вашего приложения без необходимости выпуска новой версии. Вы можете использовать его для A/B тестирования, персонализации и других задач, связанных с управлением данными.

### РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ МЕНЕДЖМЕНТА ТАБЛИЦ EXCEL

#### Постановка задачи и описание функциональности приложения

Задание: Написать мобильное приложение для оптимизации хранения Excel таблиц в формате csv, с использованием SQLite для локального хранения данных в интегрированной среде разработки Intelij IDEA .

Функционал приложения: Запуск приложения – открытие окна с полями для входа в аккаунт.

Окно для входа в аккаунт пользователя: Ввод регистрационных данных в поля «Email» и «Password» с последующим нажатием на кнопку «Log in» – вход пользователя в систему – переход в окно личного кабинета пользователя. При нажатии на поле «Register if you didn’t registered!» – переход к окну с регистрацией пользователя в систему.

Окно регистрации аккаунта пользователя: Ввод регистрационных данных в поля «Email» и «Password» с последующим нажатием на кнопку «Register» – регистрация пользователя в систему – переход в окно личного кабинета пользователя.

Окно личного кабинета пользователя: При нажатии на поле «Password» – вместо звездочек появляется введенный пользователем пароль при регистрации/входе. При нажатии на кнопку «Show user metadata» – появление диалогового окна с метаданными пользователя. При нажатии на кнопку «Go to tables list» – переход к окну с списком загруженных таблиц в Firebase Storage.

Диалоговое окно: При нажатии на кнопку «Close» диалоговое окно закрывается.

Окно с списком таблиц: При нажатии на кнопку «download table» – переход к окну с списком таблиц для загрузки c Firebase Storage на устройсвто. При нажатии на кнопку «upload table» – переход к окну для загрузки таблиц на Firebase Storage . При нажатии на кнопку «delete table» – переход к окну с списком таблиц для удаления таблиц.

Окно для загрузки таблиц на Firebase Storage: При нажатии на кнопку «Choose file path» – открытие нативного окна для выбора файла – после выбора файла, он появится в поле «Here will be your table name» с полным путем до него . При нажатии на кнопку «Choose files path» – открытие нативного окна для выбора файлов – после выбора файлов, их пути появятся в поле «Here will be your table name». При нажатии на кнопку «Upload table to the server» – загрузка файлов в Firebase Storage – на время загрузки , пропадает кнопка «Back to tables list» . При нажатии на кнопку «Back to tables list» – переход к окну с списком загруженных таблиц в Firebase Storage.

Окно для загрузки таблиц с Firebase Storage на устройство: При предварительном выборе файлов из списка путем нажатия на чекбокс с последующим нажатием на кнопку «download selected tabs» – открытие нативного окна выбора файла – после выбора файла, папка в которой был выбран файл начнется скачивание выбранных файлов с Firebase Storage, в время скачивания кнопка «back to tables list» недоступна. При нажатии на кнопку «back to tables list» – переход к окну с списком загруженных таблиц в Firebase Storage.

Окно для удаления таблиц с Firebase Storage: При предварительном выборе файлов из списка путем нажатия на чекбокс с последующим нажатием на кнопку «delete selected tabs» – начинается процесс удаления выбранных файлов с Firebase Storage, в время удаления, кнопка «back to tables list» недоступна. При нажатии на кнопку «back to tables list» – переход к окну с списком загруженных таблиц в Firebase Storage.

#### Описание архитектуры приложения

Приложение разрабатывалось для операционных систем Android версий API c 26 по 34. Использованная интегрированная среда для разработки Intelij IDEA с плагинами Android и Android design tools от Google и JetBrains.

Использованные библиотеки:

1. Firebase Storage – Облачное хранилище от гугла.
2. Firebase Authentication – Облачный учет аккаунтов от гугла.
3. Apache commons IO – Библиотека от Apache , упрощающая работу с файлами и их именами.
4. Android SDK - Комплекс библиотек для разработки на Android(Включая SQLite).

Система сборки приложения – Gradle.

#### Описание процесса разработки и тестирования

Создаем новый Android проект c шаблоном указанным в Приложение 1 на Рисунок 1.

После нажатия на кнопку «Next», указываем имя проекта на свое усмотрение, устанавливаем имя пакета на свое усмотрение, выбираем язык программирования Java, минимальную версию андроид берем 8, язык системы сборки выбираем Groovy DSL как в Приложение 1 показано на Рисунок 2.

В обозревателе проекта идем по пути app/src/main/res/layout. ПКМ нажимаем по папке layout , далее создаем xml файл как показно в Приложение 1 на Рисунок 3. Для указания имени xml файла в поле как указано в Приложение 1 на Рисунок 4 вводим латинскими строчными буквами, пробелы заменить на нижнее подчеркивание. Повторяем процедуру до тех пор, пока не получим xml файлы, как показано в Приложение 1 на Рисунок 5

Руководствуясь учебными материалами в интернет источнике <https://developer.android.com/design/ui/mobile>1 создаем xml дизайн в файлах activity\_del\_tabs.xml, activity\_download.xml, activity\_list\_of\_avaible\_tabs.xml, activity\_main.xml, activity\_register.xml, activity\_up\_tab.xml, activity\_user\_panel.xml , check\_boxes\_item.xml, csv\_name\_item.xml, fragment\_user\_metadata.xml, fragment\_user\_metadata\_list.xml соответственно как в Приложение 2 на Рисунок 6,7,8,9,10, 11,12,13,14,15,16.

Далее, создаем пакет с названием placeholder c помощью пкм по имени вашего пакета для вызова контекстного меню в пути app/src/main/java/имя вашего пакета как показано в Приложение 3 Рисунок 17. Вызываем контекстное меню нажатием ПКМ, создаем класс как показано в Приложение 3 Рисунок 18. В появившемся окошке указываем имя класса как показано в Приложение 3 Рисунок 19. Повторяем процедуру до тех пор пока не будет создано классы как показано в Приложение 3 Рисунок 20. Аналогично в имени вашего пакета повторяем процедуру до тех пор, пока не получим результат как показано в Приложении 3 Рисунок 21.

В файле build.gradle внутри пути src/app в разделе dependencies прописываем зависимости как указано в Приложении 3 Рисунок 22.

Руководствуясь учебными материалами в: <https://developer.android.com/reference>2, <https://commons.apache.org/proper/commons-io/description.html>3, <https://firebase.google.com/docs/>4 , прописываем код на java который будет соответствовать описанному выше функционалу приложения.

Тестирование приложения: Подключаем наше андроид устройство с включенным режимом отладки по Wi-Fi. Подключаем наше андроид устройство к IntelIj IDEA через сканирование QR кода как показно в приложении 4 Рисунок 23 и 24.

### ЭКОНОМИЧЕСКАЯ ЭФФЕКТИВНОСТЬ ИСПОЛЬЗОВАНИЯ РАЗРАБОТАННОГО ПРИЛОЖЕНИЯ

#### Анализ затрат на разработку и поддержку приложения

На разработку данного мобильного приложения ушло 7 бессонных календарных дней, потрачено миллиарды нервных клеток, а также потрачено 0 Российских рублей. Поддержка приложения будет осуществляться согласно требованиям заказчика.

#### Оценка экономии времени и ресурсов при использовании приложения

В долгосрочной перспективе сотрудники предприятия использующее данное приложение по назначению сэкономят много важного времени, из за решения проблемы захламленности Excel таблиц. В таком случае у предприятия купившее данное приложение появится временное окно для сотрудников предприятия.

### ЗАКЛЮЧЕНИЕ

#### Подведение итогов работы

Данная курсовая нацелена на решение проблемы захламленности Excel таблиц в формате CSV на предприятиях , путем установки централизованной системы управления таблиц. Данная цель по мнению автора курсовой работы достигнута с помощью мобильного приложения.

#### Предложения по дальнейшему развитию приложения

В будущем планируется добавить намного больше функционала чем описанное выше, например: разграничить права пользователей, видимость таблиц для определенных групп пользователей, добавить просмотр метаданных выбранных таблиц.

### СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Сборник учебных материалов, пособий, методик разработки графического интерфейса для операционной системы Android : официальный сайт. – Google – URL: <https://developer.android.com/design/ui/mobile> (дата обращения: 08.05.2024). – Текст: электронный.
2. Сборник учебных материалов, пособий, методик разработки программного обеспечения с использованием Android API для операционной системы Android : официальный сайт. – Google – URL: <https://developer.android.com/reference> (дата обращения: 08.05.2024). – Текст: электронный.
3. Техническое руководство по использованию библиотеки Apache Commons IO  – Apache Commons – URL: <https://commons.apache.org/proper/commons-io/description.html> (дата обращения: 08.05.2024). – Текст: электронный.
4. Сборник учебных материалов, пособий, методик разработки мобильного приложения с помощью Firebase для операционной системы Android : официальный сайт. – Google – URL: <https://firebase.google.com/docs/> (дата обращения: 08.05.2024). – Текст: электронный.

### ПРИЛОЖЕНИЯ

Приложение 1

Рисунок 1

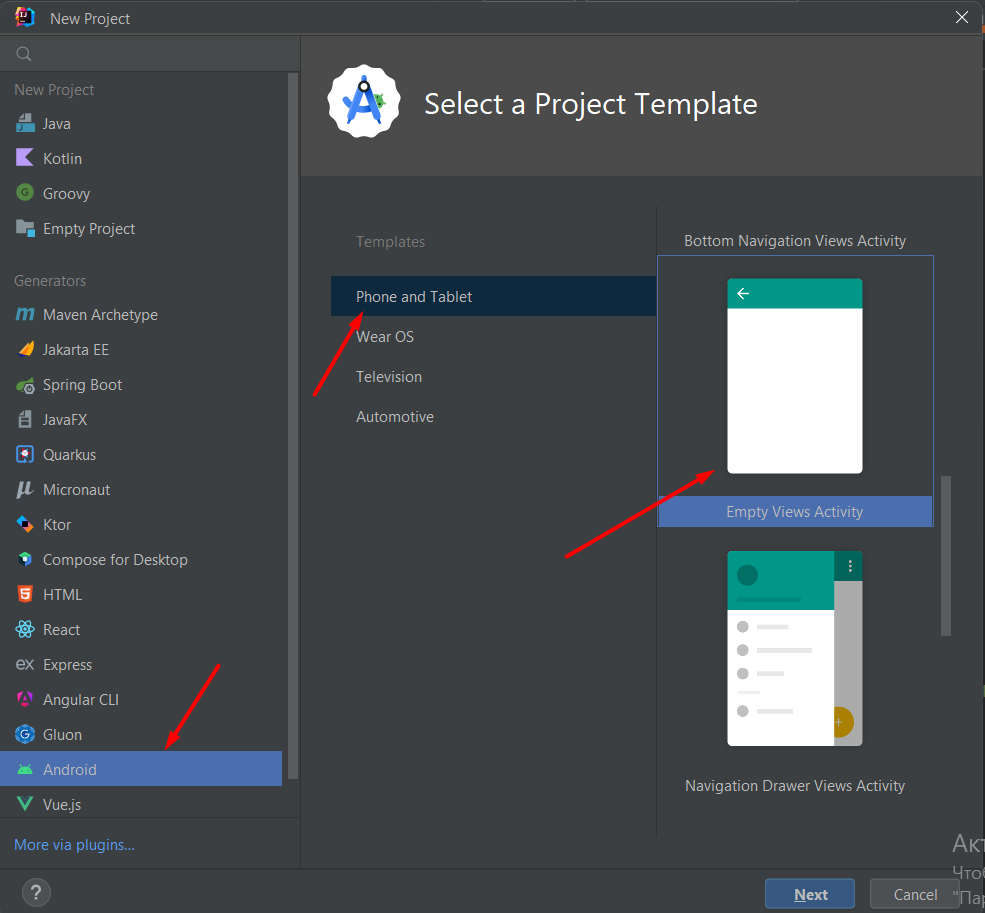


Рисунок 2

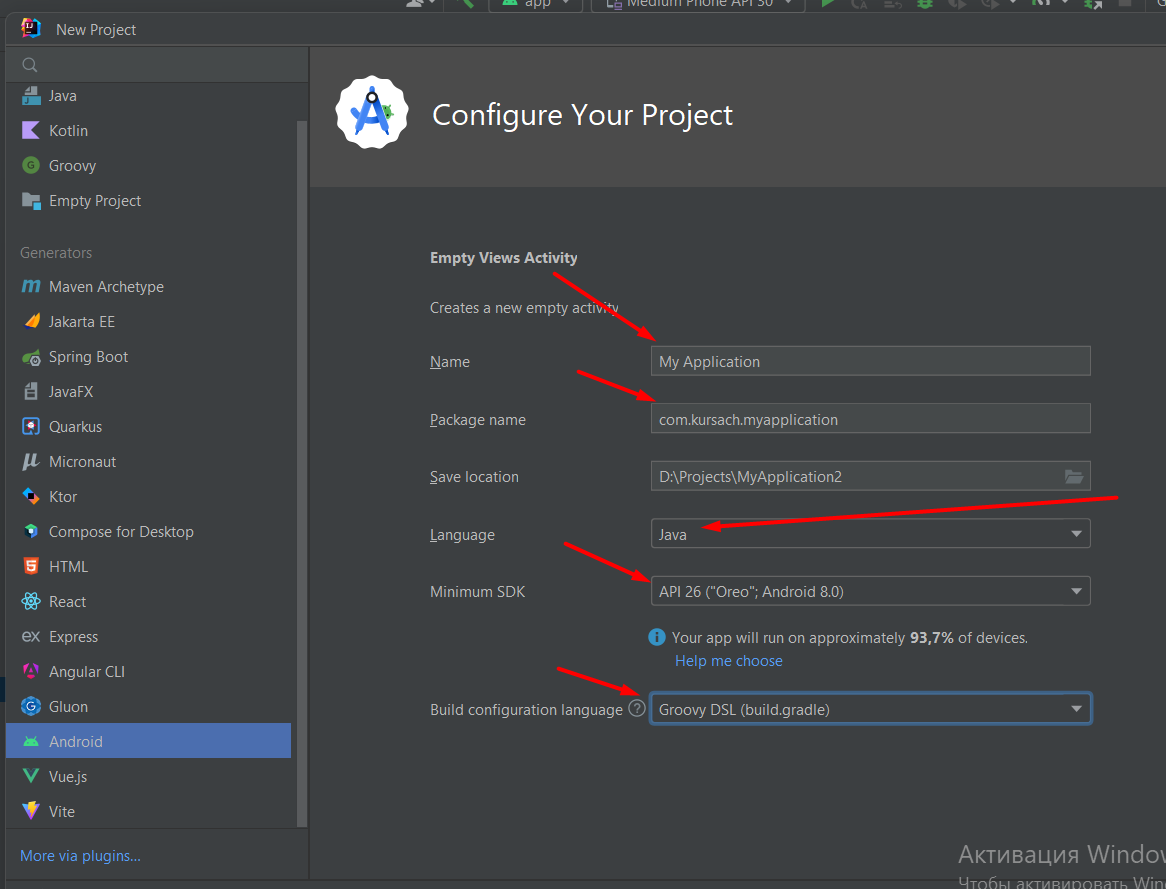


Рисунок 3

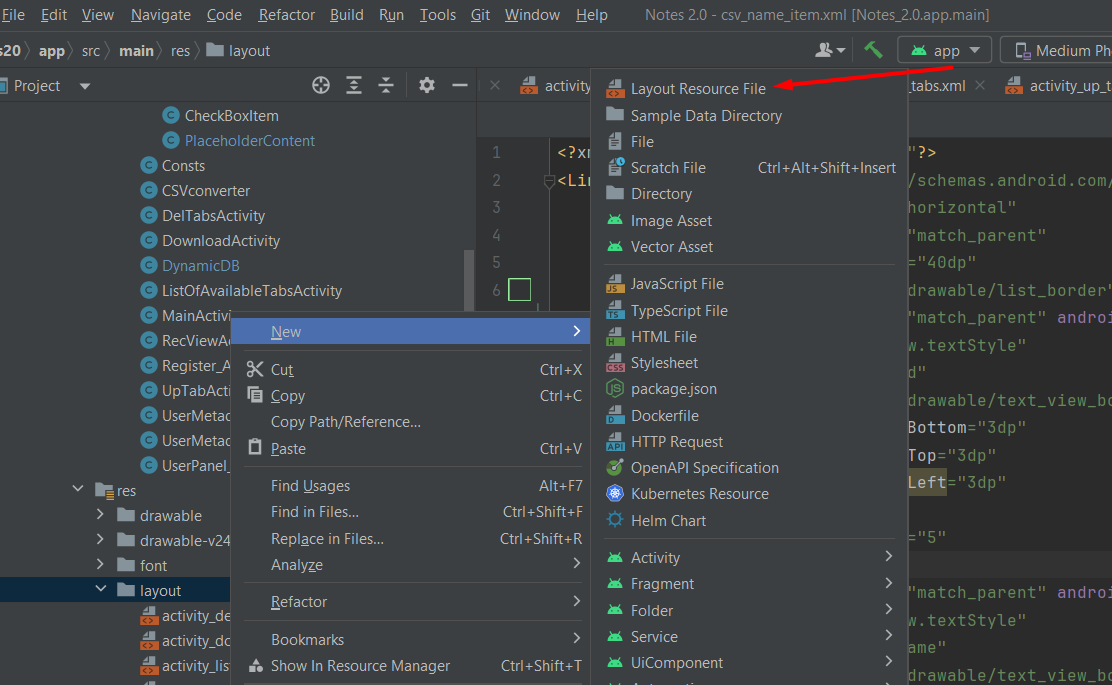


Рисунок 4

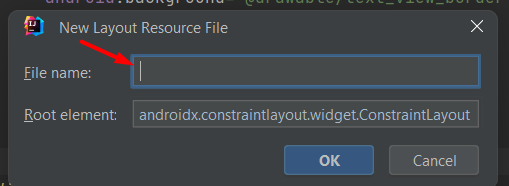
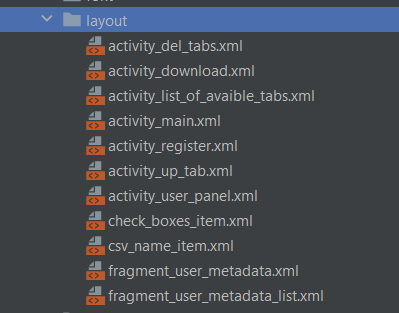


Рисунок 5



Приложение 2

Рисунок 6

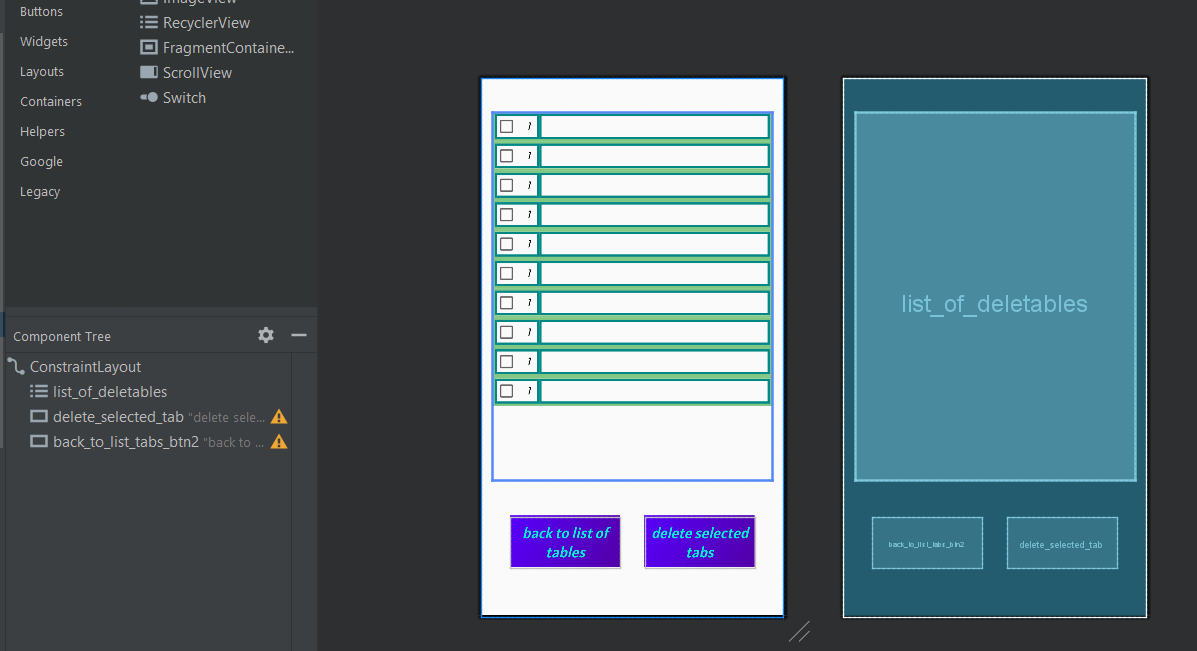


Рисунок 7

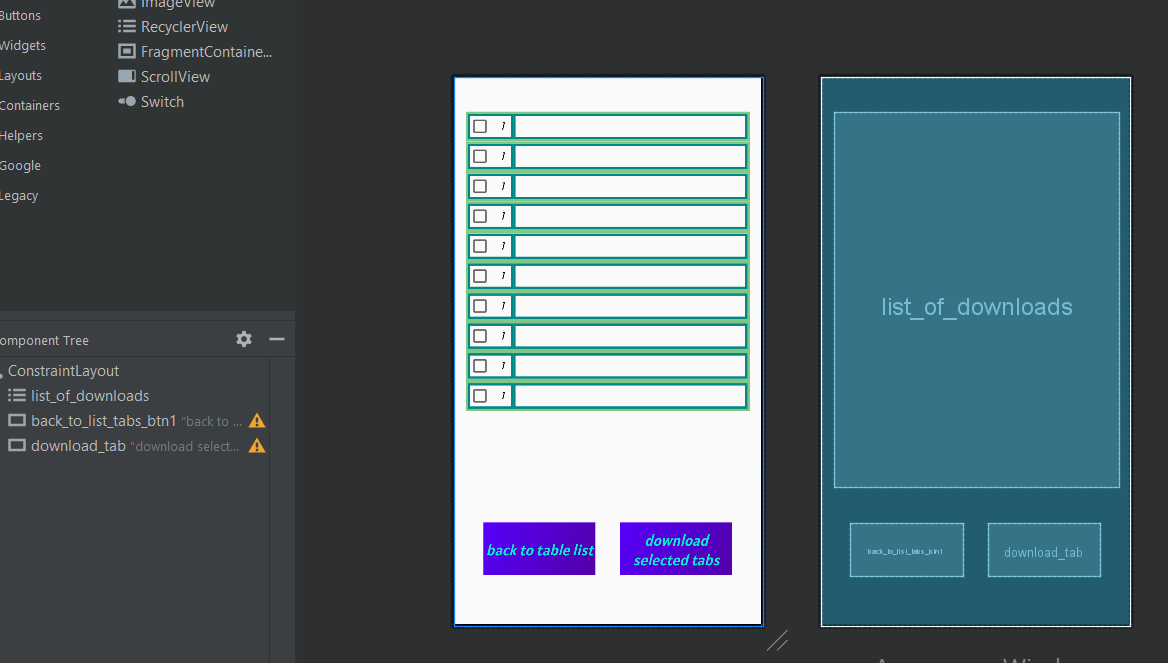


Рисунок 8

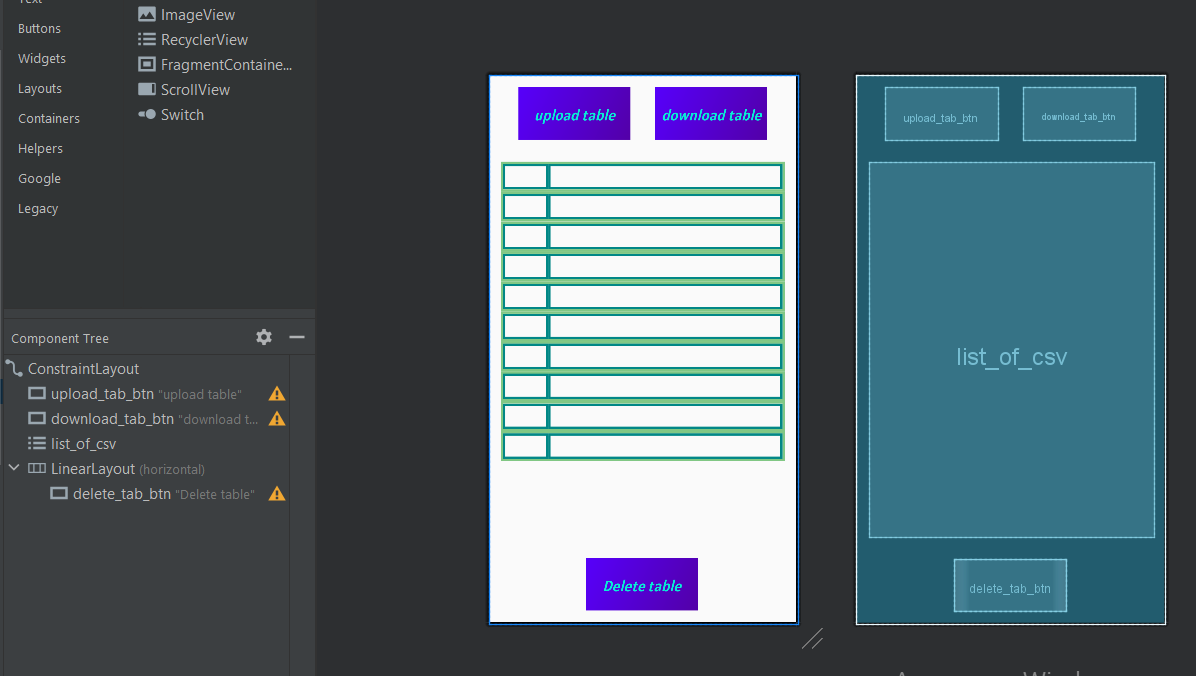


Рисунок 9

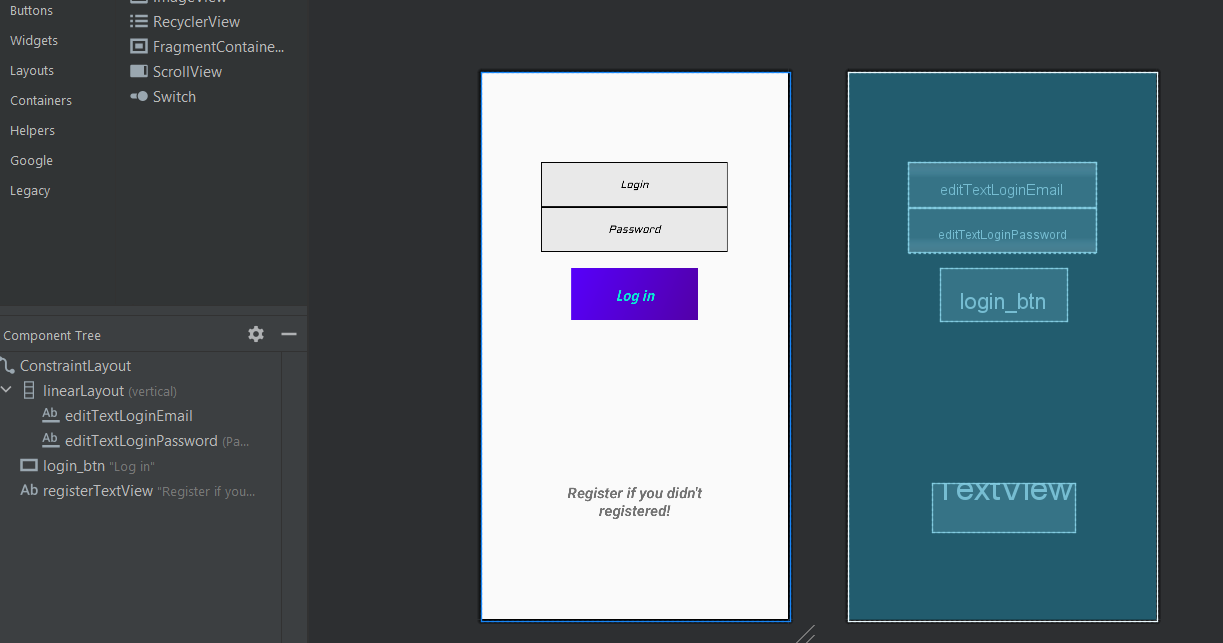


Рисунок 10

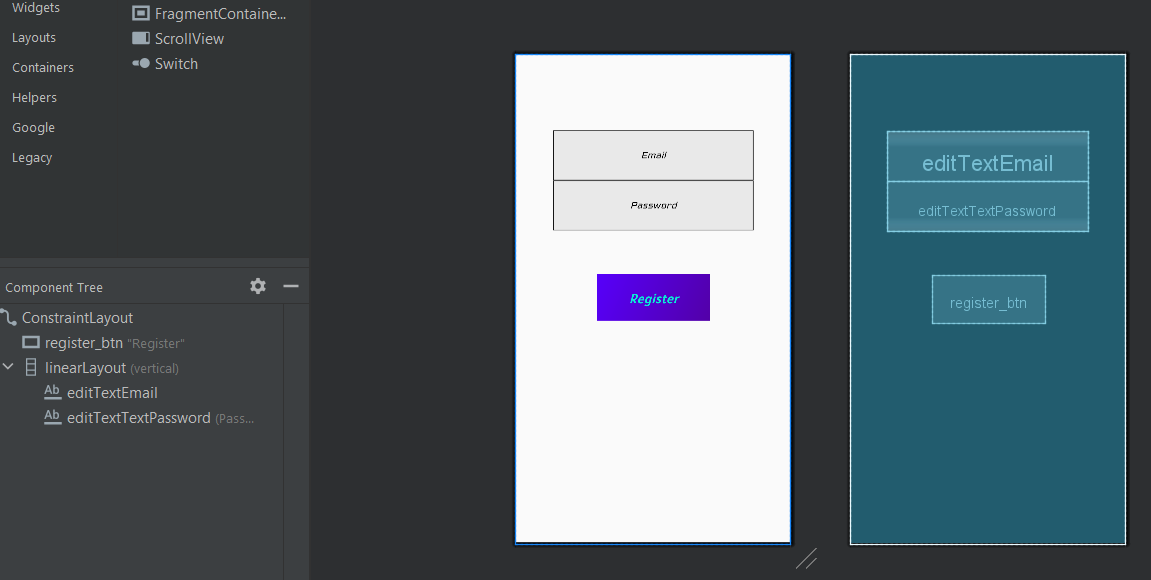


Рисунок 11

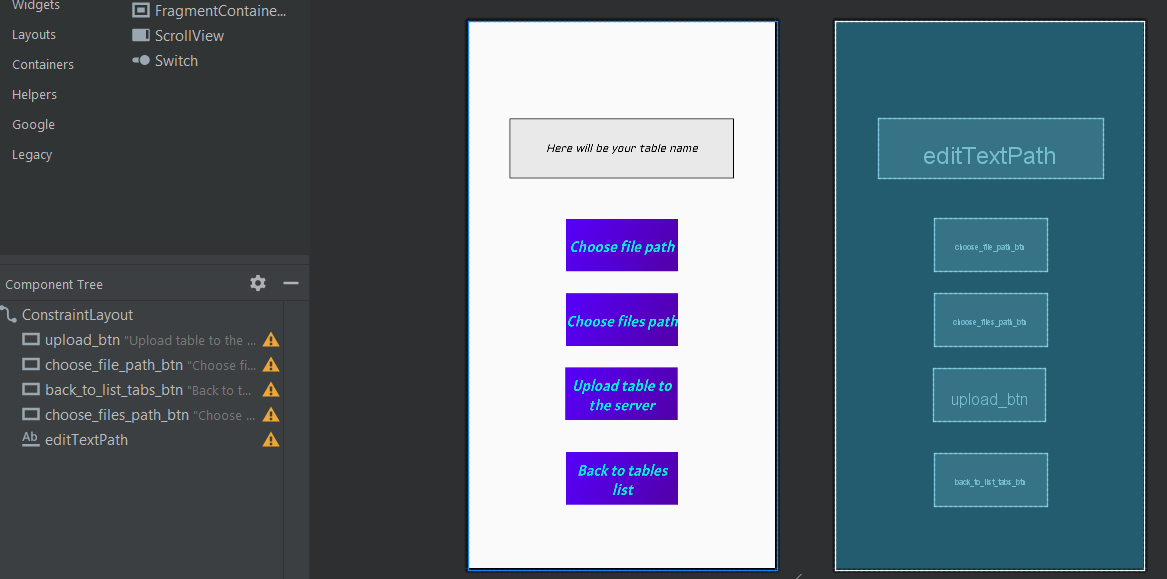


Рисунок 12

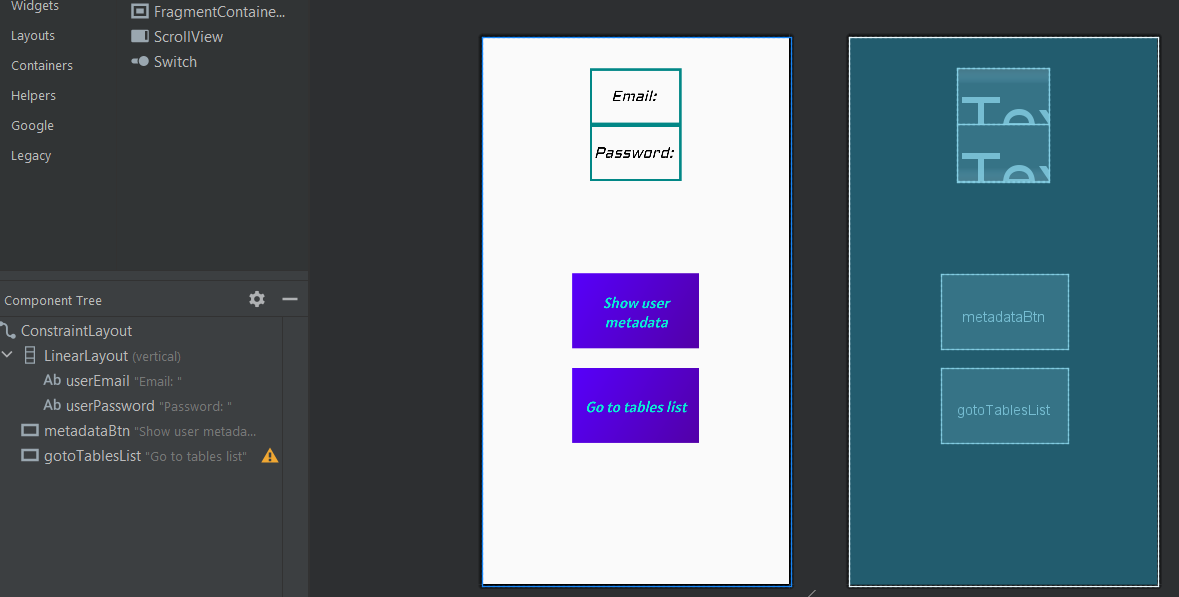


Рисунок 13

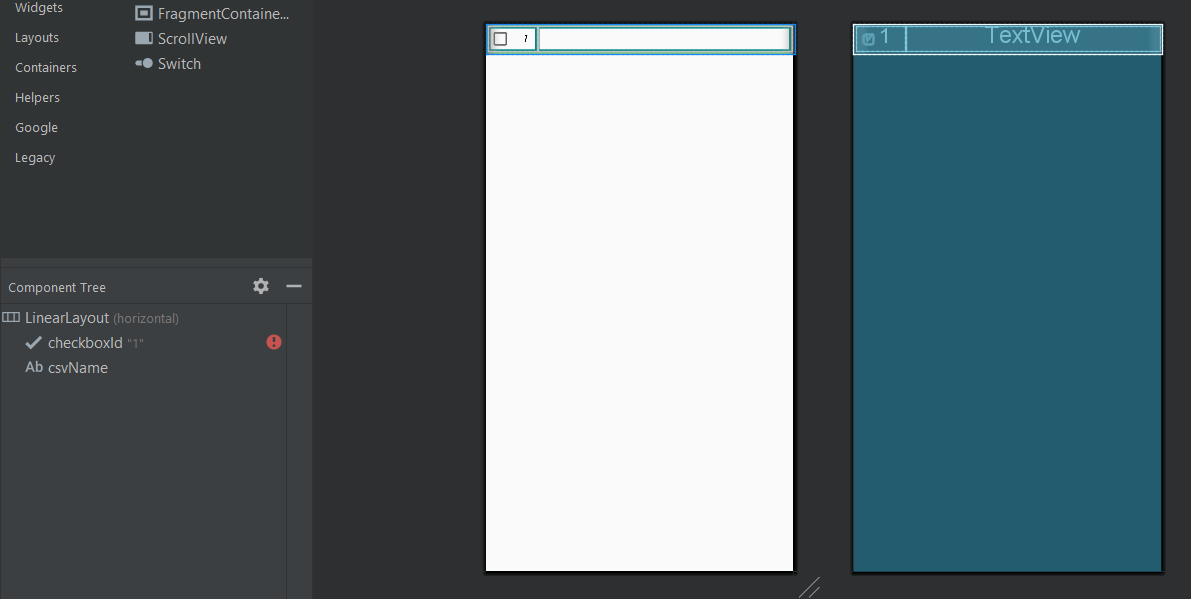


Рисунок 14

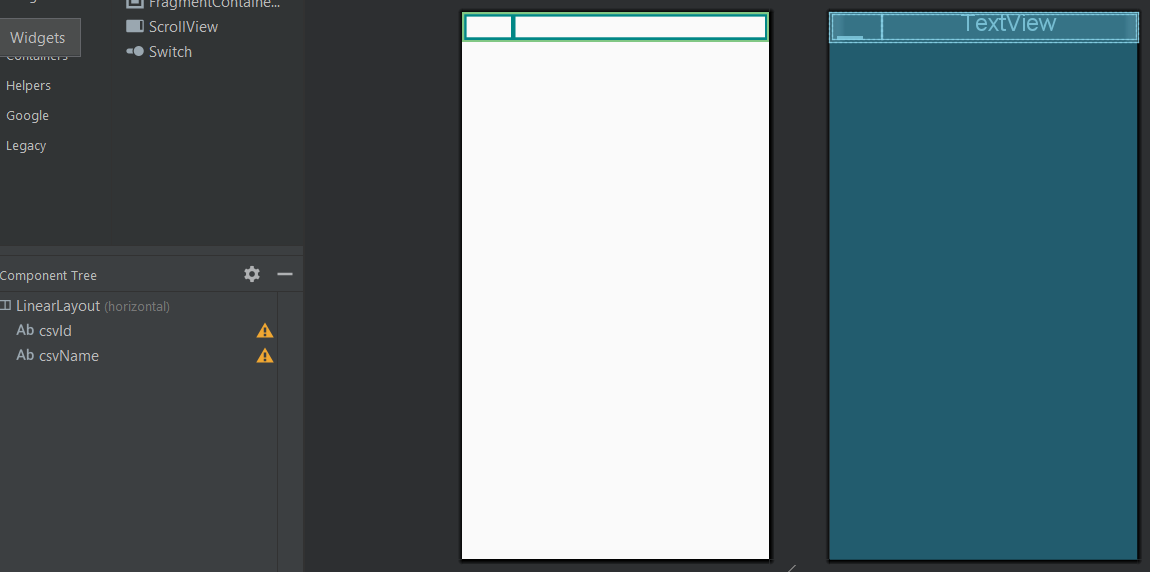


Рисунок 15

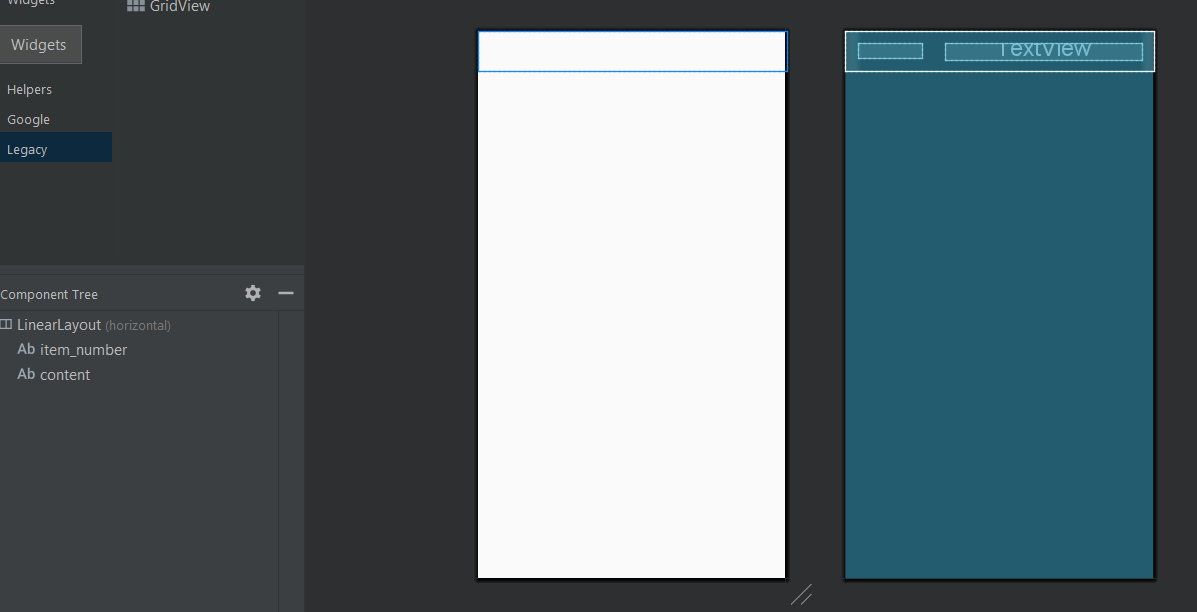
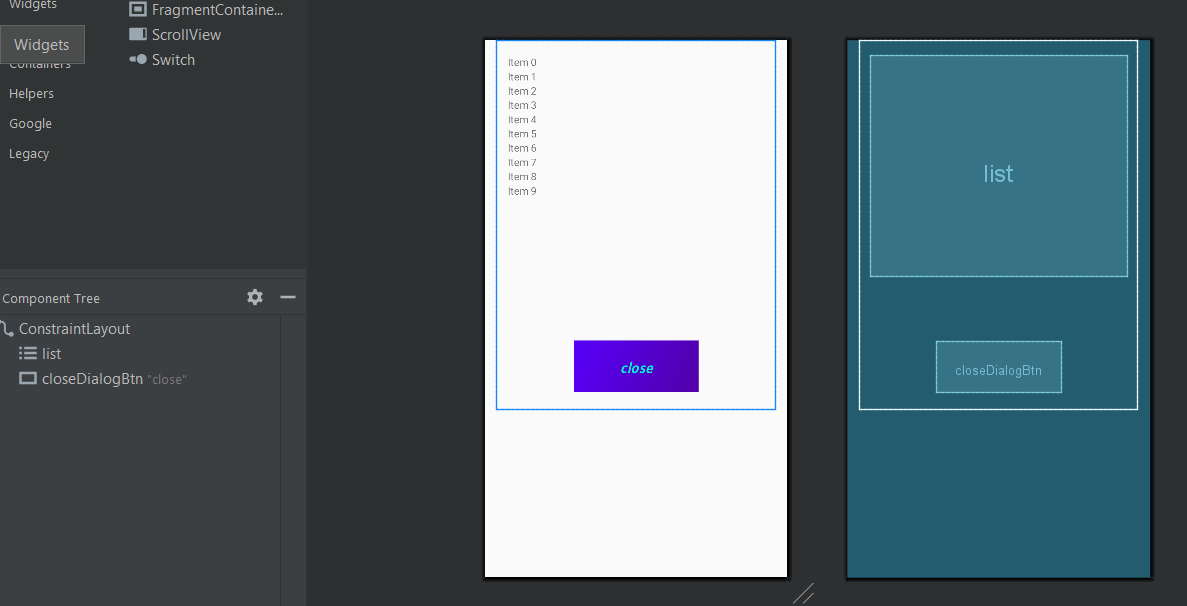


Рисунок 16



Приложение 3

Рисунок 17

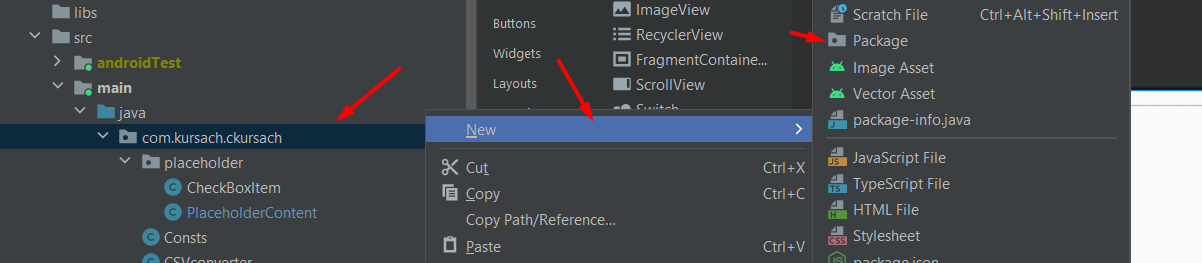


Рисунок 18

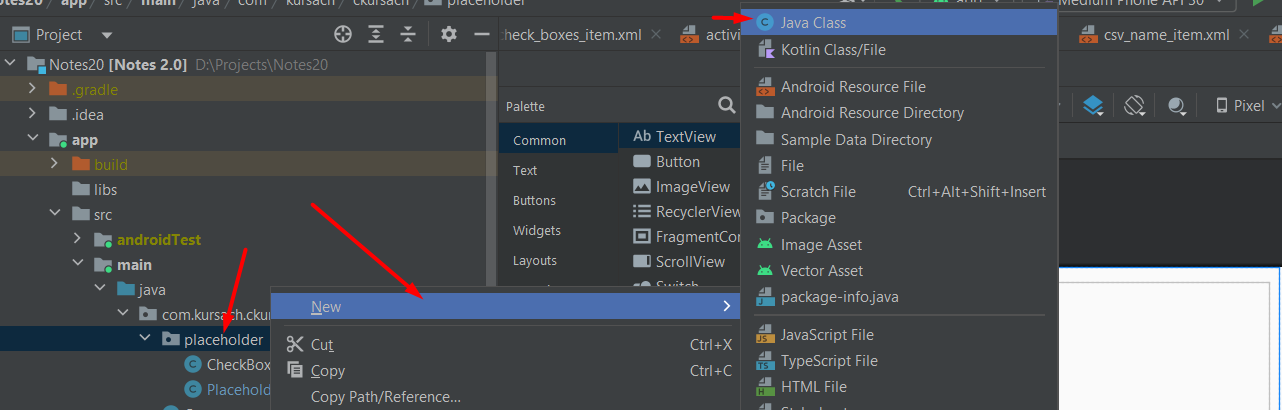


Рисунок 19

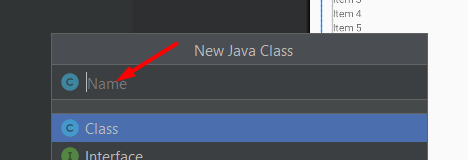


Рисунок 20



Рисунок 21

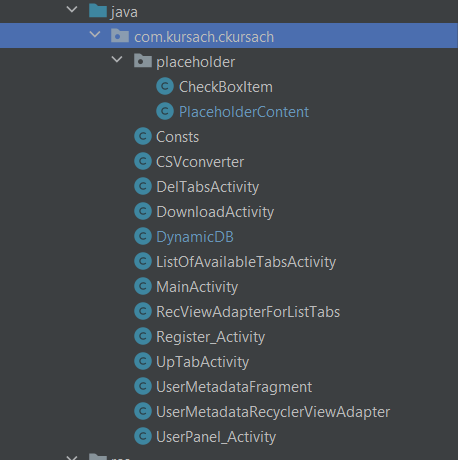
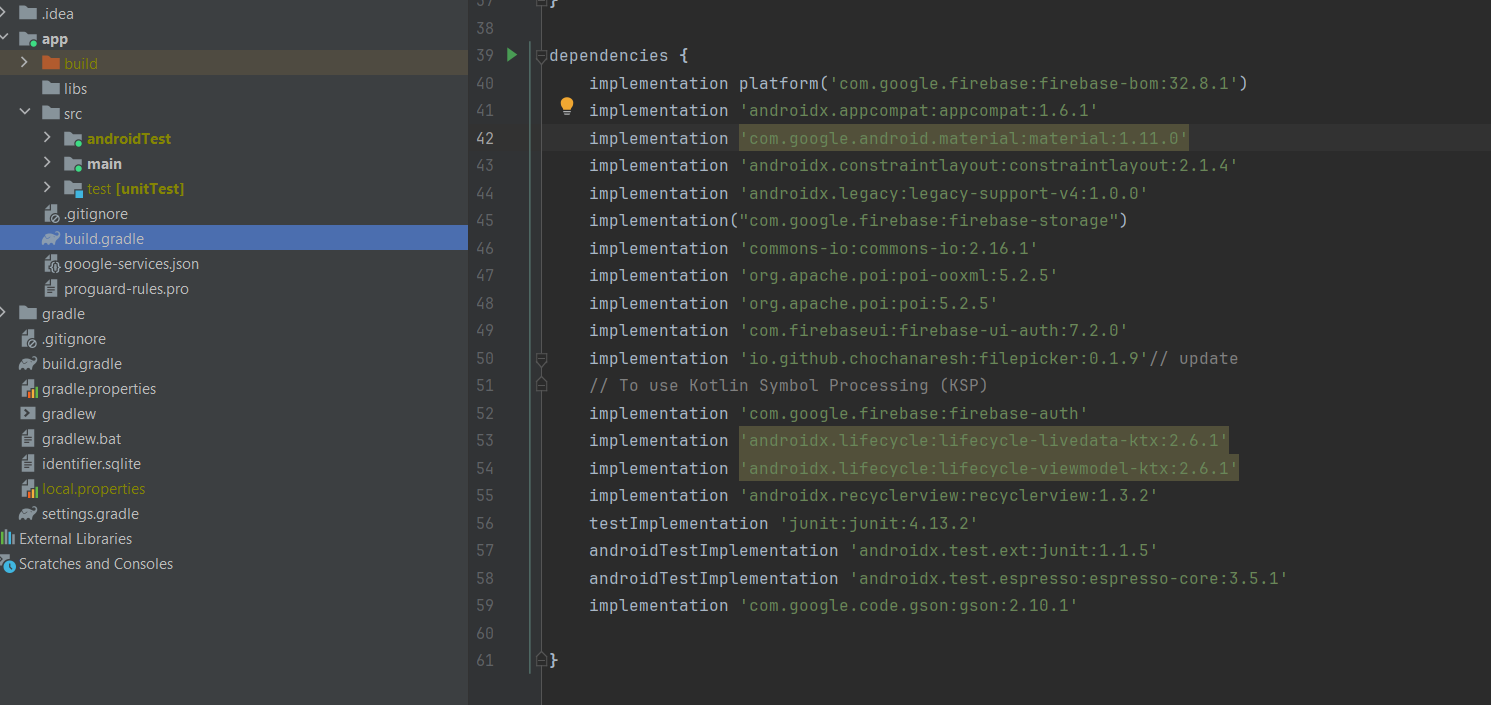


Рисунок 22



Приложение 4

Рисунок 23

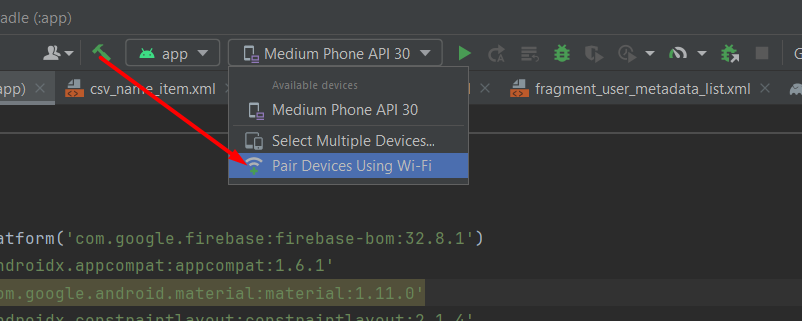


Рисунок 24

