|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА** – **Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**КУРСОВАЯ РАБОТА**

по дисциплине «Архитектура операционных систем мобильных устройств»

**Тема курсовой работы «**»

|  |  |
| --- | --- |
| **Студент группы** ИКБО-12-22 | Солобай Александр Петрович |
|  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись студента) |

|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

Утверждаю

Заведующий кафедрой МОСИТ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Головин С.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_ 2023 г.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 4](#_Toc164791547)

[ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ 6](#_Toc164791548)

[ПРОЕКТНАЯ ЧАСТЬ 13](#_Toc164791549)

[ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 16](#_Toc164791550)

[ЗАКЛЮЧЕНИЕ 31](#_Toc164791551)

[СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 32](#_Toc164791552)

# ВВЕДЕНИЕ

В современном информационном обществе, где цифровые технологии прочно вошли в повседневную жизнь, создание инновационных программных продуктов становится неотъемлемой частью развития. В контексте этой динамичной эпохи, домашняя медиатека, представляющая собой средство организации и управления мультимедийным контентом, приобретает особую актуальность.

Цель настоящей работы заключается в разработке и реализации программного комплекса, ориентированного на создание интуитивно понятного пользовательского интерфейса и обеспечение широкого спектра функциональных возможностей. Предлагаемый прототип мобильного приложения на платформе Android предполагает создание интерактивной системы, способной эффективно управлять медиаконтентом в домашних условиях.

Задачи работы включают в себя:

* Исследование предметной области разрабатываемой программной системы, анализ требований и формулирование технического задания в соответствии с ГОСТ 19.201-78.
* Проектирование и реализация ролевой модели безопасности для обеспечения конфиденциальности и целостности данных.
* Изучение жизненного цикла мобильных программ и компонентов, а также выбор и применение языков программирования высокого уровня, соответствующих целям и тематике курсовой работы.
* Разработка и внедрение визуальных элементов, сервисов и методов хранения данных в создаваемом программном комплексе.
* Тестирование и оптимизация разработанного приложения для обеспечения высокой производительности и удовлетворения потребностей пользователей.

Общая структура работы представляет собой последовательное выполнение данных задач, что позволит достичь поставленных целей и создать эффективный инструмент управления мультимедийным контентом в домашних условиях.

# ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

Таблица 1 — Аналоги разрабатываемого приложения “Домашняя медиатека”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Приложение | OC | Цена | Основной  функционал | Плюсы | Минусы |
| Plex | Windows  Mac Linux  iOS Android | Бесплатно/  Платно | Организация и потоковая передача медиафайлов. | Поддерживает множество форматов медиафайлов. Возможность потоковой передачи на разные устройства  Создание персонального сервера для доступа к медиа из любой точки мира. | Некоторые функции требуют подписки Plex Pass.  Не всегда стабильная работа на некоторых устройствах.  Для доступа к некоторым продвинутым функциям может потребоваться дополнительная настройка. |
| Kodi | Windows Mac Linux  Android | Бесплатно | Мультимедийный центр с открытым исходным кодом. | Поддержка плагинов и различных тем оформления. Мощные настройки проигрывателя и библиотеки медиафайлов. Гибкость настройки и расширяемость функционала. | Не имеет встроенной функциональности для потоковой передачи медиафайлов.  Некоторые пользователи могут считать интерфейс сложным для начинающих.  Требует наличия собственной библиотеки медиафайлов на устройстве, где установлен Kodi. |
| Emby | Windows Mac Linux  iOS Android | Бесплатно/  Платно | Организация и потоковая передача медиафайлов. | Синхронизация между устройствами, медиа-сервер и клиентские приложения. Может использоваться для доступа к медиа из дома или удаленно.  Поддержка различных форматов медиафайлов. | Некоторые продвинутые функции доступны только в платной версии.  Не всегда стабильная работа на некоторых устройствах требуется некоторое время на настройку и запуск медиа-сервера. |
| Jellyfin | Windows Mac Linux  iOS Android | Бесплатно | Организация и потоковая передача медиафайлов. | Проект с открытым исходным кодом.  Поддержка плагинов и широкий спектр функций.  Возможность создания персонального медиа-сервера. Синхронизация и транскодирование медиафайлов. | - Интерфейс и функциональность могут показаться менее полными по сравнению с коммерческими аналогами. Некоторые функции могут работать не столь стабильно или требуют дополнительной настройки.  Поддержка может быть ограничена в сравнении с более популярными альтернативами. |
| Infuse | iOS  tvOS | Бесплатно/  Платно | Проигрыватель медиафайлов с поддержкой сетевых хранилищ. | Интуитивный интерфейс и поддержка множества форматов медиафайлов.Поддержка сетевых хранилищ, включая облачные сервисы.  Богатые функции, такие как автоматическая метаданные и трейлеры.  Синхронизация просмотра через iCloud. | Ограниченная поддержка платформ (только iOS и tvOS).  Некоторые продвинутые функции доступны только в платной версии.  Не подходит для пользователей, которые хотят централизованную библиотеку медиафайлов на сервере. |
| VLC | Windows Mac Linux  iOS Android | Бесплатно | Медиаплеер с открытым исходным кодом. | Поддержка широкого спектра форматов медиафайлов и потоковое воспроизведение.  Простой в использовании и не требует настройки сервера.  Бесплатное и открытое ПО. | Интерфейс не всегда настолько интуитивен, как у других приложений.  Ограниченные функции потоковой передачи и организации библиотеки медиафайлов по сравнению с специализированными медиа-серверами.  Не поддерживает функции синхронизации между устройствами. |
| Serviio | Windows Mac  Linux | Бесплатно/  Платно | Медиа-сервер для потоковой передачи медиафайлов. | Поддержка множества устройств и форматов медиафайлов.Транскодирование видео и аудио в реальном времени.  Прост в настройке и поддерживает потоковую передачу видео, аудио и изображений на различные устройства в сети. | Некоторые продвинутые функции доступны только в платной версии.  Не так много расширений и дополнительных функций, как у некоторых других медиа-серверов.  Интерфейс может показаться не таким интуитивным для новичков. |
| Universal Media Server | Windows Mac  Linux | Бесплатно | Медиа-сервер с открытым исходным кодом. | Поддержка большого количества устройств и форматов медиафайлов.Прост в настройке и поддерживает потоковую передачу видео, аудио и изображений на различные устройства в сети.  Бесплатное и открытое ПО. | Не так много продвинутых функций и настроек, как у некоторых других медиа-серверов.  Интерфейс может показаться не таким интуитивным для новичков.  Не всегда самое активное обновление и поддержка сообщества в сравнении с другими проектами. |

Аналоги Android Studio для разработки мобильных приложений:

1. Изображение выглядит как Графика, логотип, круг, символ

   Автоматически созданное описаниеIntelliJ IDEA: это интегрированная среда разработки (IDE), разработанная компанией JetBrains. Она предоставляет широкий спектр функций для разработки Android-приложений, включая умный редактор кода, поддержку языков программирования Kotlin и Java, а также инструменты для тестирования и отладки.
2. Изображение выглядит как круг, Графика, логотип, символ

   Автоматически созданное описаниеEclipse: это одна из самых популярных IDE для разработки программного обеспечения, включая приложения для Android. Eclipse предлагает широкий набор плагинов и инструментов для разработки, включая поддержку для языков Java и Kotlin, систему управления версиями, отладчик и многие другие.
3. Изображение выглядит как Графика, снимок экрана, символ, логотип

   Автоматически созданное описаниеVisual Studio Code (VS Code): хотя изначально разработан для веб-разработки, VS Code также имеет расширения для разработки приложений Android. Он обладает высокой настраиваемостью, легкостью использования и богатым функционалом, включая редактор кода с подсветкой синтаксиса, отладчик, систему управления версиями и многое другое.
4. Изображение выглядит как Красочность, дизайн

   Автоматически созданное описаниеNetBeans: еще одна мощная и гибкая IDE, которая поддерживает разработку Android-приложений с использованием языков Java и Kotlin. NetBeans предлагает широкий набор инструментов для разработки, включая поддержку множества плагинов и расширений.
5. Изображение выглядит как инструмент, молоток, ручной инструмент

   Автоматически созданное описаниеXcode: это интегрированная среда разработки (IDE), разработанная Apple для создания приложений для их операционных систем, таких как iOS, macOS, watchOS и tvOS. Хотя Xcode преимущественно ориентирован на разработку приложений для устройств Apple, таких как iPhone и iPad, его можно использовать и для создания приложений для Android через некоторые дополнительные инструменты и плагины. Xcode предоставляет широкий набор инструментов, включая редактор кода, интерфейс для проектирования пользовательского интерфейса, отладчик, профилировщик и многое другое.

# ПРОЕКТНАЯ ЧАСТЬ

Для разработки мобильного приложения по теме курсовой работы была выбрана Android Studio.

Android Studio — это официальная интегрированная среда разработки (IDE) для создания приложений под операционную систему Android. Ее выбор для разработки мобильного приложения обосновывается несколькими преимуществами:

* Официальная поддержка: Android Studio разрабатывается и поддерживается напрямую командой Google, что обеспечивает соответствие последним требованиям и стандартам платформы Android.
* Интеграция с Android SDK: Android Studio включает в себя все необходимые инструменты для разработки под Android, включая компиляторы, отладчики, эмуляторы и т. д.
* Удобство разработки интерфейса: С помощью Android Studio удобно создавать пользовательские интерфейсы благодаря интуитивно понятному редактору макетов.
* Поддержка языков программирования: Android Studio поддерживает различные языки программирования, включая Java, Kotlin и C++. Это дает разработчикам возможность выбирать наиболее удобный для них язык.
* Мощный отладчик: Инструменты отладки в Android Studio позволяют быстро находить и исправлять ошибки в приложении.
* Поддержка Google Play Services: Android Studio легко интегрируется с сервисами Google Play, такими как Google Maps, Firebase, аналитика и другие.
* Активное сообщество и обширная документация: Использование Android Studio поддерживается обширным сообществом разработчиков Android, что облегчает получение помощи и решение проблем.
* Бесплатность и открытость: Android Studio является бесплатным и с открытым исходным кодом, что позволяет разработчикам использовать его без дополнительных затрат и иметь возможность вносить изменения в инструмент при необходимости.

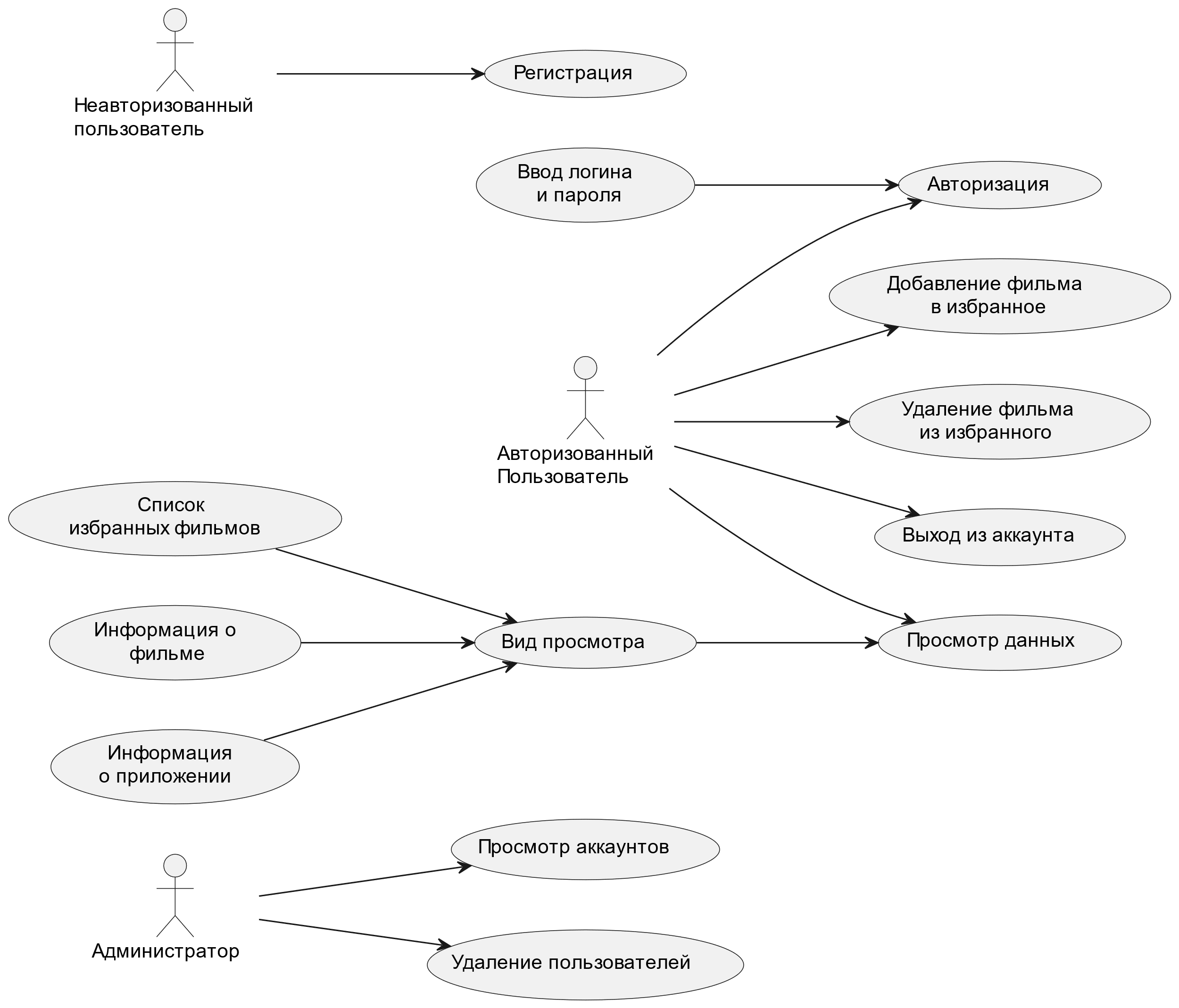


Рисунок 1 – диаграмма вариантов использования для разрабатываемого приложения

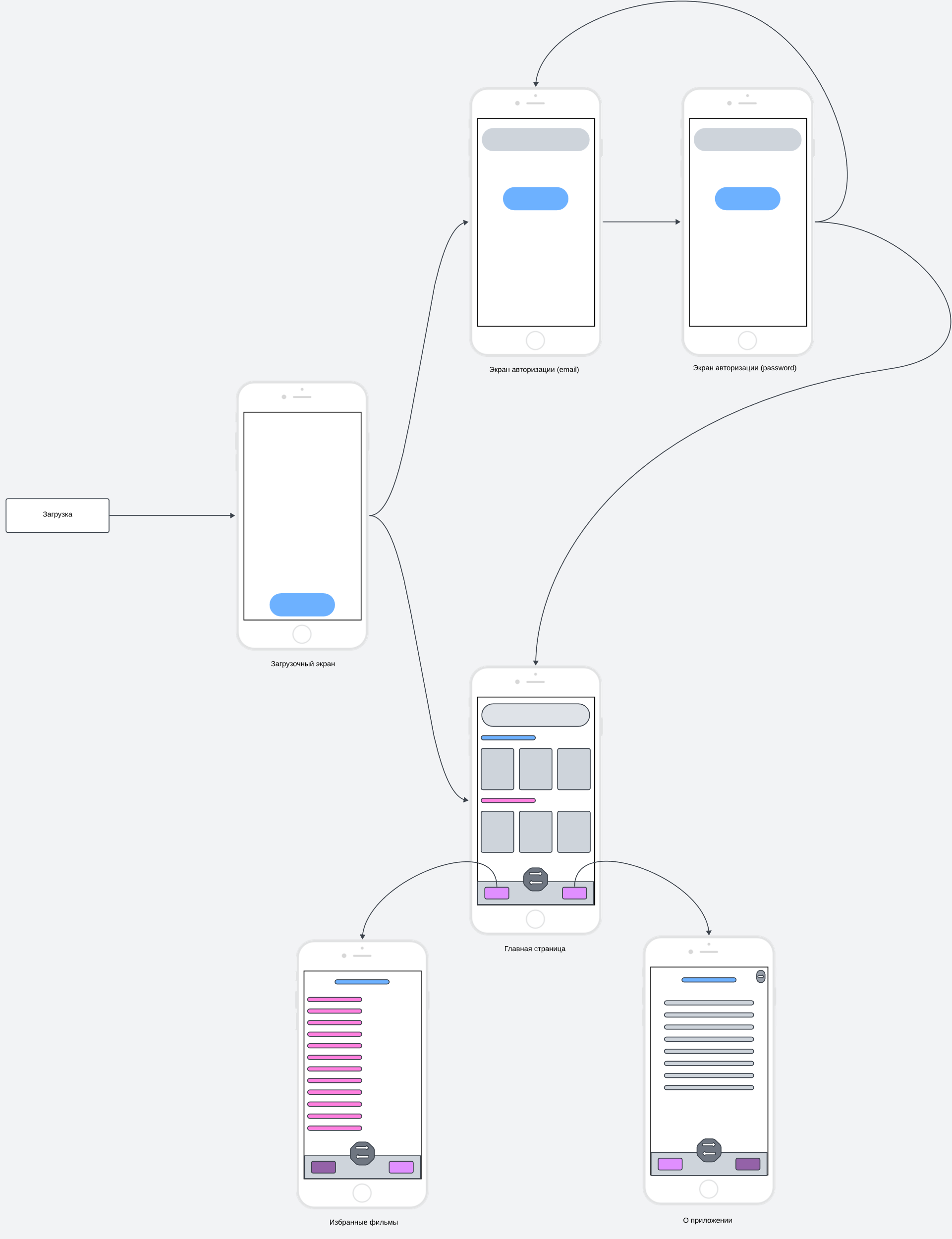


Рисунок 2 – проектирование пользовательского интерфейса

# ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

Для работы с главным экраном нужно сгенерировать Java-объекты из JSON.

Изображение выглядит как текст, снимок экрана, число, программное обеспечение

Автоматически созданное описание

Рисунок 3 – JSON с конкретным фильмом

Изображение выглядит как текст, снимок экрана, компьютер, программное обеспечение

Автоматически созданное описание

Рисунок 4 – полученный Java-объект

Те же действия нужно проделать со списком фильмов.

Напишем IntroFragment, который будем показываться неавторизованным пользователям при первом запуске нашего приложения.

*Листинг 1 – IntroFragment.java*

|  |
| --- |
| *package* com.example.coursach.Fragments;  *import* android.annotation.SuppressLint; *import* android.content.Context; *import* android.content.SharedPreferences; *import* android.net.ConnectivityManager; *import* android.net.Network; *import* android.net.NetworkCapabilities; *import* android.os.Bundle; *import* android.view.LayoutInflater; *import* android.view.View; *import* android.view.ViewGroup; *import* android.widget.Button; *import* android.widget.TextView;  *import* androidx.annotation.NonNull; *import* androidx.annotation.Nullable; *import* androidx.fragment.app.Fragment; *import* androidx.navigation.Navigation;  *import* com.example.coursach.R; *import* com.google.android.material.snackbar.Snackbar; *import* com.google.firebase.auth.FirebaseAuth; *import* com.google.firebase.auth.FirebaseUser;  *public class* IntroFragment *extends* Fragment {   @SuppressLint("SetTextI18n")  *private void* showCustomSnackbar() {  Snackbar **snackbar** = Snackbar.*make*(requireView(), "", Snackbar.LENGTH\_LONG);  @SuppressLint("RestrictedApi") Snackbar.SnackbarLayout **layout** = (Snackbar.SnackbarLayout) **snackbar**.getView();   LayoutInflater **inflater** = LayoutInflater.*from*(requireContext());  @SuppressLint("InflateParams") View **customView** = **inflater**.inflate(R.layout.custom\_snackbar, *null*);   TextView **textView** = **customView**.findViewById(R.id.snackbar\_text);  **textView**.setText("Internet connection is required");   **layout**.setPadding(0, 0, 0, 0);  **layout**.addView(**customView**, 0);   **snackbar**.show();  }   @Nullable  @Override  *public* View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {  *return* inflater.inflate(R.layout.fragment\_intro, container, *false*);  }  @Override  *public void* onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {  *super*.onViewCreated(view, savedInstanceState);   *if* (checkUserStatus()) {  Navigation.*findNavController*(view).navigate(R.id.action\_introFragment\_to\_mainFragment);  } *else* {  setupStartButton(view);  }  }   *private void* setupStartButton(View view) {  Button **getStartBtn** = view.findViewById(R.id.getStart);   **getStartBtn**.setOnClickListener(v -> {  *if* (isNetworkAvailable()) {  Navigation.*findNavController*(requireView()).navigate(R.id.action\_introFragment\_to\_loginFragment);  } *else* {  showCustomSnackbar();  }  });  }   *private boolean* checkUserStatus() {  FirebaseUser **currentUser** = FirebaseAuth.*getInstance*().getCurrentUser();  *return* **currentUser** != *null* && **currentUser**.isEmailVerified() && isUserLoggedIn();  }   *private boolean* isUserLoggedIn() {  SharedPreferences **sharedPreferences** = requireContext().getSharedPreferences("user\_auth", Context.MODE\_PRIVATE);  *return* **sharedPreferences**.contains("userId");  }   *private boolean* isNetworkAvailable() {  ConnectivityManager **connectivityManager** = (ConnectivityManager) requireActivity().getSystemService(Context.CONNECTIVITY\_SERVICE);  *if* (**connectivityManager** != *null*) {  Network **network** = **connectivityManager**.getActiveNetwork();  *if* (**network** != *null*) {  NetworkCapabilities **networkCapabilities** = **connectivityManager**.getNetworkCapabilities(**network**);  *return* **networkCapabilities** != *null* && (**networkCapabilities**.hasTransport(NetworkCapabilities.TRANSPORT\_WIFI) ||  **networkCapabilities**.hasTransport(NetworkCapabilities.TRANSPORT\_CELLULAR) ||  **networkCapabilities**.hasTransport(NetworkCapabilities.TRANSPORT\_ETHERNET));  }  }  *return false*;  } } |

Теперь пользователей будет встречать стартовый экран с кнопкой “Начать”.

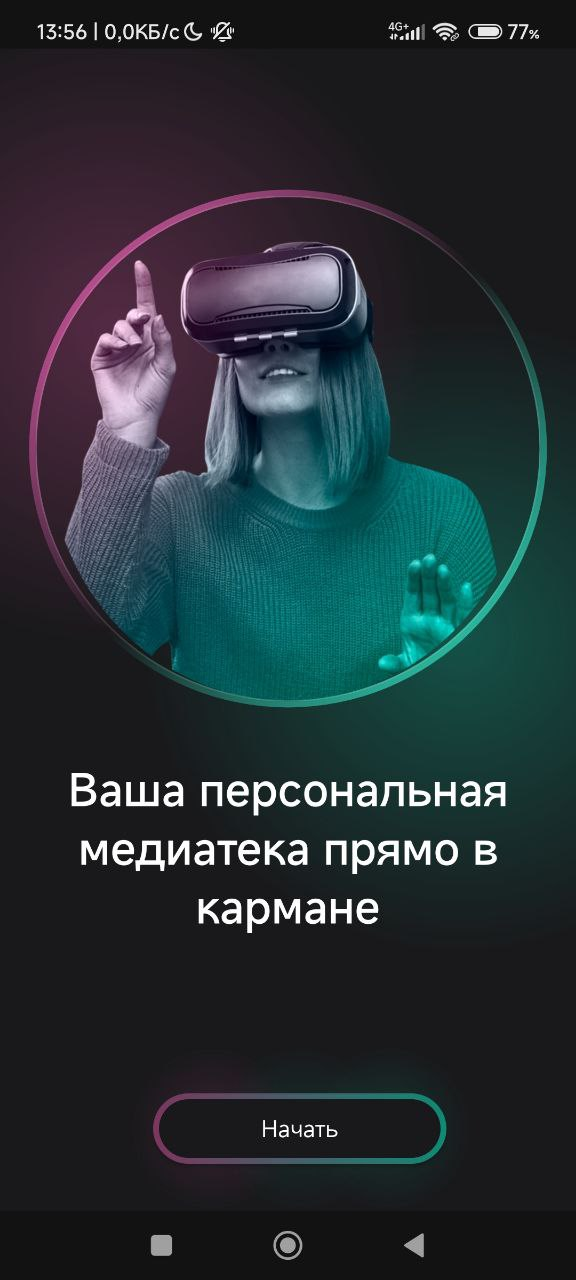


Рисунок 5 – стартовый экран

Теперь нужно написать авторизацию пользователей в приложении с помощью ссылки с подтверждением, которая будет отправляться на введенную почту.

В курсовой работе будет использоваться NoSQL база данных FireBase, которая предоставляет возможность авторизации через почту.

*Листинг 2 – фрагмент кода файла LoginFragment.java с проверкой на корректность почты и переходом к экрану ввода пароля*

|  |
| --- |
| *private boolean* isEmailValid(String email) {  String **emailPattern** = "[a**-**zA**-**Z0**-**9.\_-]+@[a**-**z]+\\.+[a**-**z]+";  *return* email.matches(**emailPattern**); }  *private void* attemptLogin() {  String **email** = editTextEmail.getText().toString().trim();   *if* (**email**.isEmpty()) {  showCustomSnackbar("Пожалуйста, введите адрес электронной почты.");  *return*;  }   *if* (!isEmailValid(**email**)) {  showCustomSnackbar("Некорректный адрес электронной почты.");  *return*;  }   mAuth.fetchSignInMethodsForEmail(**email**).addOnCompleteListener(task -> {  *boolean* **isNewUser** = Objects.*requireNonNull*(task.getResult().getSignInMethods()).isEmpty();  *if* (!**isNewUser**) {  Navigation.*findNavController*(requireView()).navigate(R.id.action\_loginFragment\_to\_mainFragment);  } *else* {  LoginFragmentDirections.ActionLoginFragmentToPassFragment **direction** =  LoginFragmentDirections.*actionLoginFragmentToPassFragment*(email);  Navigation.*findNavController*(requireView()).navigate(**direction**);  }  }).addOnFailureListener(e -> showCustomSnackbar("Произошла ошибка при попытке входа. Попробуйте снова.")); } |

Изображение выглядит как электроника, текст, снимок экрана, Электронное устройство

Автоматически созданное описание

Рисунок 6 – экран ввода почты

*Листинг 3 – фрагмент кода файла PassFragment.java с отправкой письма для подтверждения почты и регистрацией пользователя*

|  |
| --- |
| *private void* registerUser() {  String **password** = passEdt.getText().toString().trim();   *if* (**password**.isEmpty()) {  showCustomSnackbar("Пожалуйста, введите пароль.");  *return*;  }   mAuth.signInWithEmailAndPassword(email, **password**).addOnCompleteListener(task -> {  *if* (task.isSuccessful()) {  FirebaseUser **user** = mAuth.getCurrentUser();  *if* (**user** != *null*) {  saveUserAuthState(**user**.getUid());  }  Navigation.*findNavController*(requireView()).navigate(R.id.action\_passFragment\_to\_mainFragment);  } *else* {  mAuth.createUserWithEmailAndPassword(email, password).addOnCompleteListener(registerTask -> {  *if* (registerTask.isSuccessful()) {  FirebaseUser **newUser** = mAuth.getCurrentUser();  *if* (**newUser** != *null*) {  saveUserAuthState(**newUser**.getUid());  sendEmailVerification();  }  } *else* {  showCustomSnackbar("Ошибка при регистрации. Попробуйте снова.");  }  }).addOnFailureListener(e -> showCustomSnackbar("Ошибка при регистрации. Такой пользователь уже существует. Возможно вы ввели неверный пароль"));  }  }); }  *private void* sendEmailVerification() {  FirebaseUser **user** = mAuth.getCurrentUser();  *if* (**user** != *null*) {  **user**.sendEmailVerification().addOnCompleteListener(task -> {  *if* (task.isSuccessful()) {  showCustomSnackbar("Письмо с подтверждением отправлено на вашу почту.");  Navigation.*findNavController*(requireView()).navigate(R.id.action\_passFragment\_to\_mainFragment);  } *else* {  showCustomSnackbar("Не удалось отправить письмо с подтверждением. Попробуйте снова.");  }  });  } *else* {  showCustomSnackbar("Ошибка: пользователь не найден.");  } } |

Изображение выглядит как электроника, текст, снимок экрана, Электронное устройство

Автоматически созданное описание

Рисунок 7 – экран ввода пароля

Изображение выглядит как текст, снимок экрана, дизайн

Автоматически созданное описание

Рисунок 8 – письмо с подтверждением почты

После успешной авторизации данные пользователя сохранятся и при последующих открытиях приложения он сразу будет попадать на главный экран с фильмами.

Изображение выглядит как текст, Человеческое лицо, мультимедиа, снимок экрана

Автоматически созданное описание

Рисунок 9 – главный экран

*Листинг 4 – фрагмент кода файла MainFragment.java с функцией поиска фильмов.*

|  |
| --- |
| *private void* searchMovies(String query) {  String **searchUrl** = "https://moviesapi.ir/api/v1/movies?q=" + query;  loading1.setVisibility(View.VISIBLE);  StringRequest **mStringRequest** = *new* StringRequest(Request.Method.GET, **searchUrl**, response -> {  Gson **gson** = *new* Gson();  loading1.setVisibility(View.GONE);   ListFilm **items** = **gson**.fromJson(response, ListFilm.*class*);  NavController **navController** = Navigation.*findNavController*(requireActivity(), R.id.nav\_host\_fragment);  FilmListAdapter **adapter** = *new* FilmListAdapter(**items**, **navController**);  recyclerViewNewMovies.setAdapter(**adapter**);  }, error -> loading1.setVisibility(View.GONE));  mRequestQueue.add(**mStringRequest**); } |

С главного экрана пользователь может перейти к списку избранных фильмов или посмотреть информацию о приложении, а также выполнить выход из аккаунта. При нажатии на постер с фильмом пользователь может посмотреть детальную информацию о нем.

Изображение выглядит как текст, снимок экрана, программное обеспечение, мультимедиа

Автоматически созданное описание

Рисунок 10 – избранные фильмы

При нажатии на элемент из списка пользователя будет перекидывать на экран с детальной информацией об избранном фильме.

*Листинг 5 – фрагмент кода файла DetailFragment.java с методами добавления/удаления фильма из избранного*

|  |
| --- |
| *private void* toggleFavorite(FilmItem film) {  *if* (film == *null*) {  showCustomSnackbar("Информация о фильме недоступна!");  *return*;  }   FirebaseUser **user** = FirebaseAuth.*getInstance*().getCurrentUser();  *if* (**user** == *null*) {  Log.*d*("Authentication", "Пользователь не аутентифицирован.");  *return*;  }   String **userId** = **user**.getUid();  DatabaseReference **mDatabase** = FirebaseDatabase.*getInstance*(getResources().getString(R.string.url)).getReference().child("users").child(**userId**).child("favorites").child(String.*valueOf*(film.getId()));  SharedPreferences **prefs** = requireActivity().getSharedPreferences("Favorites", Context.MODE\_PRIVATE);  *boolean* **isCurrentlyFavorite** = **prefs**.getBoolean("Film\_" + film.getId(), *false*);   *if* (**isCurrentlyFavorite**) {  **mDatabase**.removeValue().addOnSuccessListener(aVoid -> {  showCustomSnackbar("Удалено из избранного.");  saveFavoriteStatus(film.getId(), *false*);  updateFavoriteButtonBackground(*false*);  }).addOnFailureListener(e -> showCustomSnackbar("Ошибка при удалении из избранного."));  } *else* {  **mDatabase**.setValue(film).addOnSuccessListener(aVoid -> {  showCustomSnackbar("Добавлено в избранное.");  saveFavoriteStatus(film.getId(), *true*);  updateFavoriteButtonBackground(*true*);  }).addOnFailureListener(e -> showCustomSnackbar("Не удалось добавить в избранное."));  } }   *private void* saveFavoriteStatus(*int* filmId, *boolean* isFavorite) {  SharedPreferences **prefs** = requireActivity().getSharedPreferences("Favorites", Context.MODE\_PRIVATE);  SharedPreferences.Editor **editor** = **prefs**.edit();  **editor**.putBoolean("Film\_" + filmId, isFavorite);  **editor**.apply(); }   *private void* updateFavoriteButtonBackground(*boolean* isFavorite) {  *if* (isFavorite) {  favoriteImg.setBackgroundResource(R.drawable.bg\_circle\_pink);  } *else* {  favoriteImg.setBackgroundResource(R.drawable.bg\_cirlcle\_dark);  } }  *private void* checkFavoriteStatus() {  SharedPreferences **prefs** = requireActivity().getSharedPreferences("Favorites", Context.MODE\_PRIVATE);  *boolean* **isFavorite** = **prefs**.getBoolean("Film\_" + idFilm, *false*);  updateFavoriteButtonBackground(**isFavorite**); } |

Изображение выглядит как текст, снимок экрана, коллекция

Автоматически созданное описание

Рисунок 11 – детальная информация о фильме

Изображение выглядит как текст, электроника, снимок экрана, программное обеспечение

Автоматически созданное описание

Рисунок 12 – раздел с информацией о приложении и кнопкой выхода из аккаунта

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы был проведен полный цикл разработки программного комплекса – домашней медиатеки для мобильной платформы Android. Актуальность данного проекта подтверждается постоянным ростом интереса к мультимедийным контентам и необходимостью их организации и управления в современных домашних условиях.

В рамках работы были успешно реализованы все поставленные цели и задачи, включая анализ предметной области, проектирование безопасной ролевой модели, выбор и применение соответствующих языков программирования, а также разработку и оптимизацию пользовательского интерфейса и функциональности приложения.

Важным результатом является создание интуитивно понятного и функционального программного комплекса, способного удовлетворить потребности пользователей в организации и управлении мультимедийным контентом в домашних условиях.

Дальнейшее развитие проекта может включать в себя расширение функциональности, улучшение пользовательского опыта, а также адаптацию приложения для работы на других платформах и устройствах.

Таким образом, разработанный программный комплекс представляет собой важный шаг в области создания инновационных решений для организации и управления мультимедийным контентом в домашних условиях, и может быть успешно применен в практических задачах повседневной жизни.

# СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Разработка мобильных приложений (КР/КП). [Электронный ресурс]. URL: https://online-edu.mirea.ru/course/view.php?id=7400 (дата обращения 02.03.2024).
2. Разработка мобильных приложений (часть 1/1). [Электронный ресурс]. URL: https://online-edu.mirea.ru/course/view.php?id=6592 (дата обращения 05.03.2024).
3. Navigation. — Текст: электронный // Developers: [сайт]. — URL: https://developer.android.com/guide/navigation (дата обращения: 30.04.2024).
4. Fragments. — Текст: электронный // Developers: [сайт]. — URL: https://developer.android.com/guide/fragments (дата обращения: 30.04.2024).
5. How to make gradient background in android. — Текст: электронный // stackoverflow: [сайт]. — URL: https://stackoverflow.com/questions/13929877/how-to-make-gradient-background-in-android (дата обращения: 30.04.2024).
6. Add a floating action button. — Текст: электронный // Developers: [сайт]. — URL: https://developer.android.com/develop/ui/views/components/floating-action-button (дата обращения: 30.04.2024).
7. Generate Plain Old Java Objects from JSON or JSON-Schema.. — Текст : электронный // jsonschema2pojo : [сайт]. — URL: https://www.jsonschema2pojo.org/ (дата обращения: 06.05.2024).