**Developers:** Mokgadi Moitsi, Luyanda Mazibuko, Precious Pheeha  
**Language:** Kotlin   
**Platform:** Android Studio   
**API:** OMDb (Open Movie Database)   
**Repository:** [Insert GitHub Repository Link]   
**Version:** 2.0 (Final Release)

Shape

**1. Introduction**

The **Cinematic App** is a mobile application designed for movie lovers who want to explore, discover, and learn more about their favorite films. Built using **Kotlin** and integrated with the **OMDb API**, the app retrieves up-to-date movie data, including posters, genres, release dates, ratings, and summaries.

The app demonstrates the practical implementation of modern Android development concepts such as **API integration**, **user authentication**, **RecyclerView layouts**, and **intuitive UI design**. It was also developed as part of a coursework project focusing on Android app development, API consumption, and GitHub version control.

Cinematic offers a user-friendly experience for both casual movie fans and developers who wish to explore real-world examples of Android-based applications connected to live databases.

Shape

**2. Purpose of the Application**

The primary purpose of Cinematic is to provide users with an easy-to-use platform to:

* Search for any movie by title.
* Access movie details such as cast, director, runtime, and ratings.
* View high-quality posters and visuals.
* Manage personal accounts through login and registration pages.
* Navigate seamlessly between pages using clearly defined buttons and menus.

Cinematic aims to replicate the experience of browsing a digital movie catalogue without the clutter or complexity of large streaming services. It bridges functionality and simplicity, emphasizing accessibility, design consistency, and usability.

Shape

**3. Target Users**

The app is designed for:

* 📽️ **Movie Enthusiasts:** Individuals who enjoy discovering new films and reading synopses.
* 💻 **Students & Developers:** Those who want to understand Android app architecture, Kotlin coding, and RESTful API integration.
* 🎞️ **Casual Users:** People looking for a quick, lightweight way to find movie information without using large databases like IMDb or TMDb.

Shape

**4. Design Considerations**

**4.1 Interface and Layout**

The interface was created with the goal of simplicity and functionality.

* The **Home Screen** showcases trending and popular movie titles in a clean layout using a **RecyclerView**.
* The **Search Page** includes an editable text field and a button, allowing users to find movies by title.
* The **Details Page** displays selected movie information retrieved via the OMDb API.
* The **Login and Register Pages** ensure that user accounts are verified before accessing the main features.

**4.2 Color Scheme and Visual Style**

* **Primary Color:** Black and dark gray – to evoke a cinema atmosphere.
* **Accent Color:** Red – symbolizing the excitement and energy of movies.
* **Typography:** Sans-serif fonts for readability and modern style.
* **Icons:** Material Design icons for consistency with Android standards.

**4.3 User Experience (UX)**

The UX design focuses on minimal clicks and smooth navigation.

* Pages are linked using Intent actions for efficient routing.
* Scrollable lists and search bars make browsing easy.
* Visual feedback (e.g., toast messages) confirms user actions like login, search results, or errors.

**4.4 Accessibility**

Accessibility was integrated into every part of the design:

* Labels for all images and buttons.
* Large font sizes and good contrast.
* Compatibility with screen readers.
* Follows the **Web Content Accessibility Guidelines (WCAG 2.0)** standards for visual and auditory accessibility.

Shape

**5. System Architecture**

The app follows the **Model-View-Controller (MVC)** pattern:

* **Model:** Handles data fetching from the OMDb API.
* **View:** XML layout files control what the user sees on each page.
* **Controller:** Kotlin activities manage the interaction between the user interface and data model.

**Key Components**

* MainActivity.kt – entry point for the app.
* MovieAdapter.kt – manages RecyclerView layouts for displaying multiple movie results.
* ApiService.kt – handles all API requests using Retrofit.
* LoginActivity.kt and RegisterActivity.kt – manage user authentication screens.
* MovieDetailsActivity.kt – displays full information about selected movies.

Shape

**6. GitHub and GitHub Actions**

**6.1 Repository Management**

GitHub was used for **version control**, **collaboration**, and **documentation**.

* The project repository was initialized with a .gitignore file to exclude build and system files.
* Regular commits were made to track progress.
* Branches were created for experimenting with new layouts before merging them to the main branch.
* The **README.md** file serves as the main documentation source for the project.

**6.2 GitHub Actions and CI/CD**

To maintain build stability and automate testing:

* A GitHub Actions workflow file .github/workflows/build.yml was implemented.
* This workflow automatically checks the Gradle build every time a new commit is pushed.
* Ensures that the project compiles correctly and no new code breaks existing functionality.
* The use of CI/CD pipelines improves software reliability before deployment.

**Reference:**   
[Automated Build with GitHub Action](https://github.com/marketplace/actions/automated-build-androidapp-with-github-action)

Shape

**7. Features**

**7.1 Core Features**

* 🔍 **Movie Search:** Users can type a movie title and retrieve real-time information.
* 🎞️ **Movie Details:** Each movie displays a poster, genre, director, cast, runtime, and IMDB rating.
* 🧾 **Login and Registration:** Secure access system for different users.
* 🧠 **User Dashboard:** Displays curated movie lists.
* ⚙️ **Admin Dashboard:** Enables management and updates of content.

**7.2 Technical Features**

* **API Integration:** Connects to the OMDb database via Retrofit.
* **RecyclerView:** Dynamic presentation of movie lists.
* **Intents and Navigation:** Smooth transitions between screens.
* **Error Handling:** Proper responses for invalid searches or empty results.