Software Design Document(SDD)

Project Name:-MovieSwiper

Project Overview:

MovieSwiper is a mobile application that gives us the overview of movies and helps users to save time. They can read the movie information such as Trailer Release date, Rating, Description etc.
MovieSwiper is developed using ReactJS and web APIs(Restful) are implemented. MovieSwiper application is based on the data available in TMdb [The Movie DB API].
The basic idea is to keep it simple and provide a user-friendly experience to the users. Overall.

Scope of project:

With MovieSwiper, the user is able to

- View the list of movies and their posters.
- View all types of movies on a different list.
- View the top-rated movies first in this application.
- Users can search for movies via names.
- Users can navigate between different tabs and view movies based on different categories like Popular, Browse, Library etc..
- This application helps users to find the most popular movies and save and skip them.
- New movies added weekly

On clicking the movie poster, users can able to view movie details below.

- Overview
- Release Date
- Rating
- Reviews

Tools:-

● ***VScode***	**Visual Studio Code** is a source-**code** editor developed by Microsoft for Windows, Linux and macOS. It includes embedded Git and support for debugging, syntax highlighting, intelligent **code** completion, snippets, and **code** refactoring.
● ***Android Studio***	**Android Studio** is the official Integrated Development Environment (IDE) for Android app development.
● ***Xcode***	**Xcode** is Apple's integrated development environment (IDE) that you use to build apps for Apple products including the iPad, iPhone, Apple Watch, and Mac.

Component Diagram:-

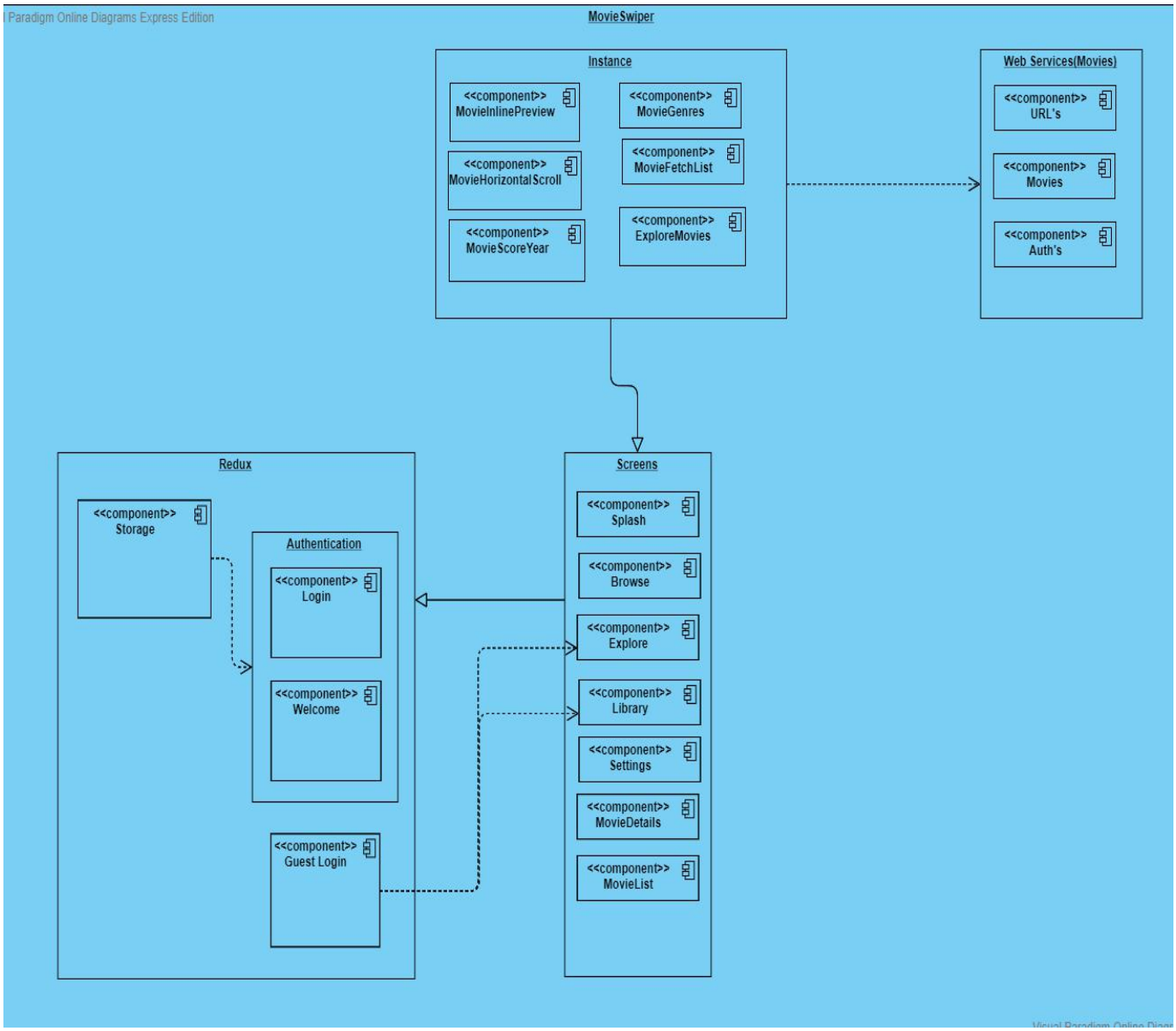
A component diagram describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development

Major Components of the Component Diagram

- **Screens:-** All the given screen list is a UI element that describes our major Application how it look alike.

- **Instance:-** These are our connected, reusable i.e from Screen to API resources of the Application that are designed for logical interaction.

- **WebService:-** The Movies are fetched from the Web based RESTful API's .

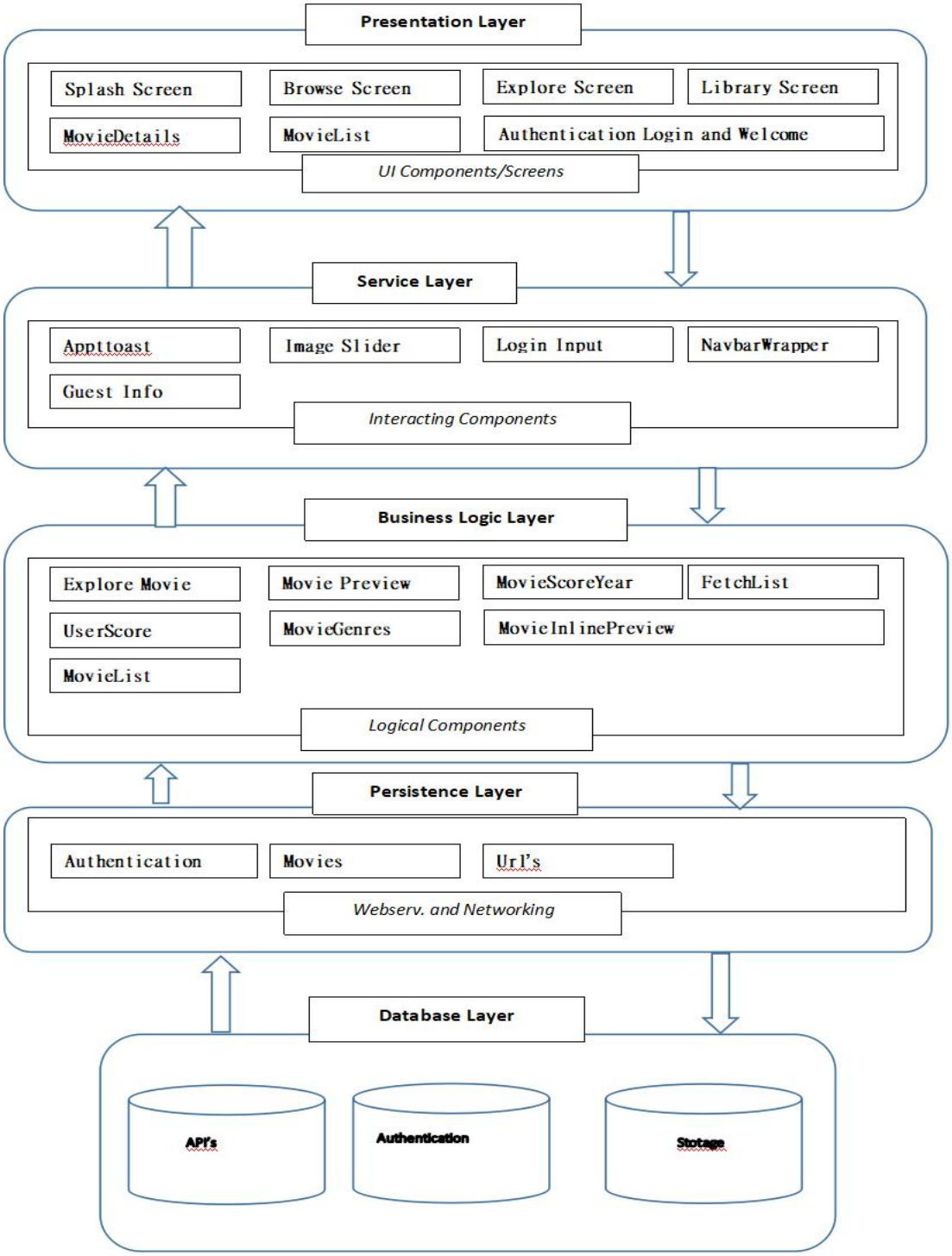


Architecture Diagram:-

A Diagram that details how each of the components are grouped as a layer and interacts with each other. In the Architecture Diagram the first layer is the Presentation layer which contains Views and Controllers, In Views we have different types of Screens from which the user interacts and we have controllers which are responsible for all the activities happening in view.

Layers of the Architecture Diagram

- **Presentation layer:-** All our UI interacting elements comes under the presentation layer i.e. the major presenatble components.
- **Service Layer:-** The activities of the user that how a user is interacting with the logic of the application.
- **Business Logic Layer:-** All our logical outcomes and components of the application comes under the Business Logical Layer
- **Persistence Layer:-**The persistence layer is the set of code to manipulate the database. In the Application the three components doing the tasks are Authentication, URL and Movie.
- **Database:-** The database is set of related data and we are fetching the data from that database.



Sequence Diagram:-

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.
Sequence diagrams are sometimes called event diagrams or event scenarios.
A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner