Mobile App

Continuous Assessment 2

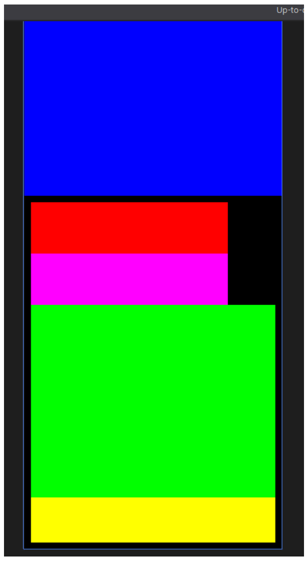
Sarah Tsangou

20 March 2024

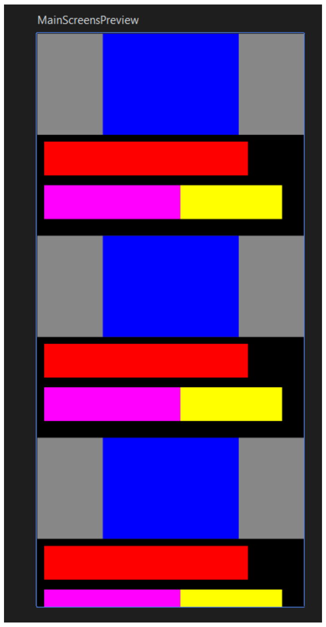
To-Do list:

- [ ] Movie screen:
    - [x] ~~Create a Movie class with the structure specified in movie.json~~
    - [x] ~~Obtain and fill data (minimum of 4 movies)~~
        - [x] ~~Obtain relevant movie data from (https://www.imdb.com/) credits at the bottom of the app~~
        - [x] ~~Generate a random number between 0 and 15 for each movie and assign to seats_remaining~~
        - [x] ~~Start with an initial default seats_selected value of 0 for all movies~~
        - [x] ~~You shall fill random URLs for images from pixabay or other free image providers to begin with(https://unsplash.com/)~~
    - [x] ~~If any seats are selected, show how many seats are selected and hide remaining seats~~
- [x] ~~Seat selection feature:~~
    - [x] ~~Clicking any item (anywhere on the item) on the movie screen should open a new MovieActivity, refer to movie_activity_*.jpg~~
    - [x] ~~Add plus and minus icons, show seats_selected in the middle~~
    - [x] ~~On click plus/minus, update both seats_selected and seats_remaining for that movie~~
    - [x] ~~Add validation, when 0 seats selected minus is disabled, when 0 seats remaining plus is disabled~~
    - [x] ~~When back is pressed, the selected seats are retained and reflected in the screen. (Hint: If you don't see any updates, call adapter notifyItemChanged as soon as you return to the screen activity)~~ ~~added a forward button to do that~~
- [ ] Bonus:
    - [ ] Add "filling fast" badge if less than 3 seats remaining
    - [ ] Use "Roboto Condensed" font to replicate same style
    - [ ] Use original movie images from myvue.com or your favourite provider (Hint: check get_movie_image_url.gif)

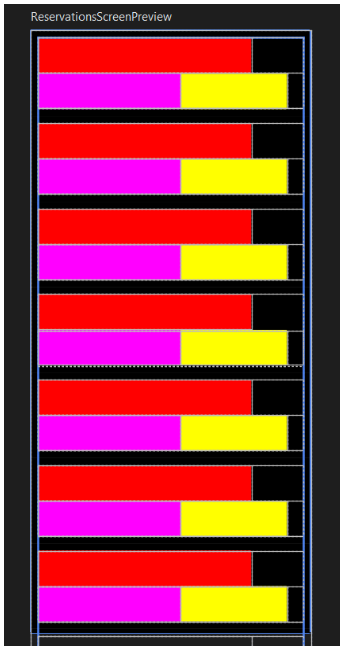
I started by decomposing the different screens from the views given



Movie Details Page Blocked Out

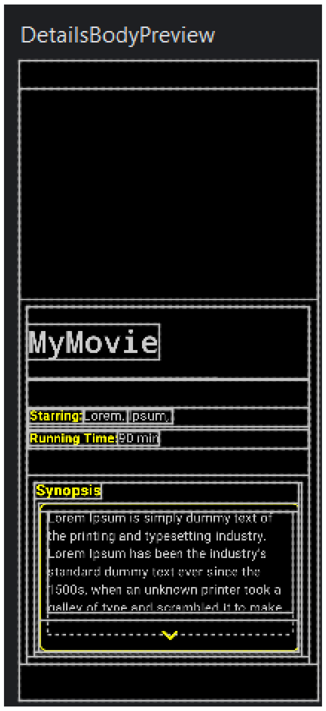


MainScreen Page blocked Out

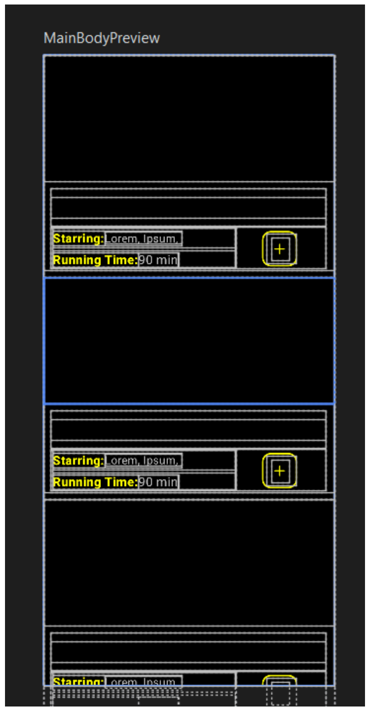


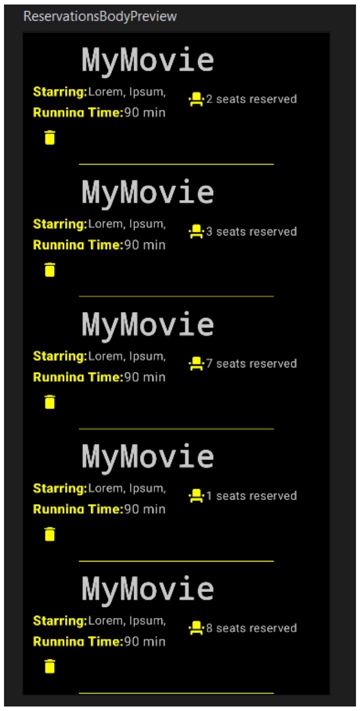
ReservationsPage blocked out

using the preview screen of Android Studio i then adjusted the layout of each screens



Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make





once satisfied i edited the layout of my screens i then for each screen created a route object to make navigation easier

Movie Class then i had the idea of using a movie api so that i can use a broader amount a movies after looking online i found the following movie api movie api: https://rapidapi.com/amrelrafie/api/movies-tv-shows-database

I then started looking for tutorials on how to use api's on android studio and followed:(Pt1: https://medium.com/@kathankraithatha/how-to-use-api-in-jetpack-compose-10d11b8f166f

Pt2: https://medium.com/@kathankraithatha/how-to-use-api-in-jetpackcompose-part-2-e1fb3e60a320)

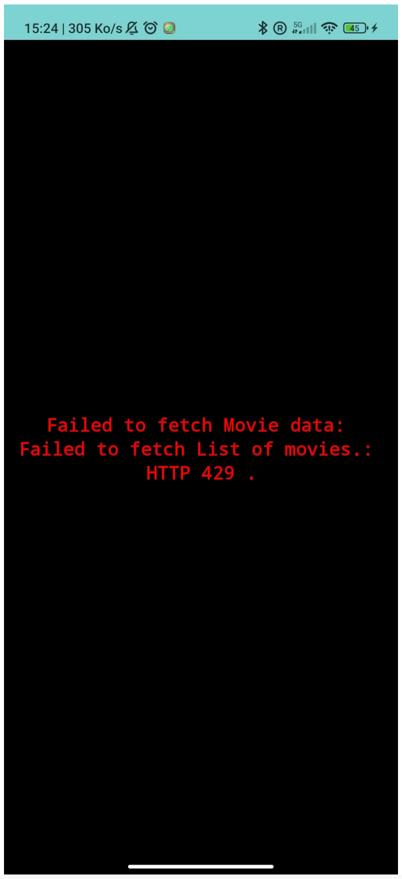
Once i had my database i had to look up how to load images from internet using a url and followed this tutorial:

https://www.delasign.com/blog/android-studio-jetpack-compose-kotlin-image-from-url/

I then modified the movie class to fit the api’s data.

The first problem i ran into while implementing an api was an infinite loading screen when testing the app on a photo spent a big amount of time trying to fix this issue the problem was passing state values to functions instead of view models.

But once i had fixed this issue I was faced with an error screen



```
Failed to fetch Movie data:
Failed to fetch List of movies.:
HTTP 429 .
```

I was now faced with an error http 403

after looking online what the error was i understood that the api i was using had limited the maout of requests that i could make per months and while fixing errors i had exceeded said limit

Thats when i decided to change my project to use a firebase api

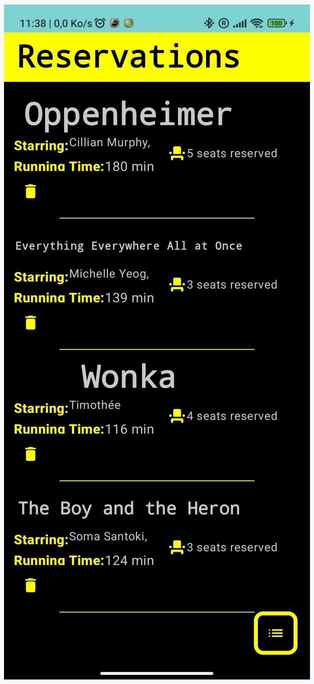
I first had to learn how to use firebase and implement it into my project

followed this tutorial: https://firebase.google.com/docs/firestore/query-data/get-data?hl=fr&authuser=0#before_you_begin

This part took a lot of time as well because i made the mistake of refusing to star over so i spent most of the time adapting my project structure to support firebase

once i was accustomed with firebase is imply had fun correcting and adjusting the screens

Here are the final screens as seen on the app



11:38 | 0,0 Ko/s

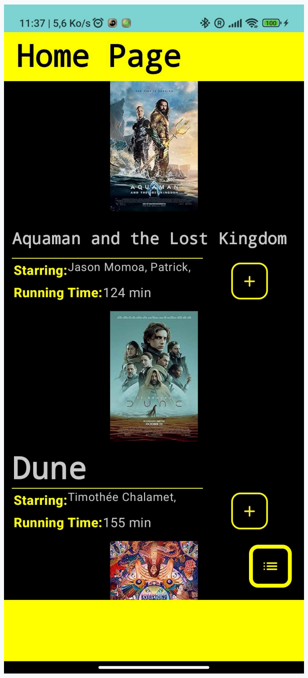
Reservations

Oppenheimer
Starring: Cillian Murphy,
Running Time: 180 min
5 seats reserved

Everything Everywhere All at Once
Starring: Michelle Yeog,
Running Time: 139 min
3 seats reserved

Wonka
Starring: Timothée
Running Time: 116 min
4 seats reserved

The Boy and the Heron
Starring: Soma Santoki,
Running Time: 124 min
3 seats reserved

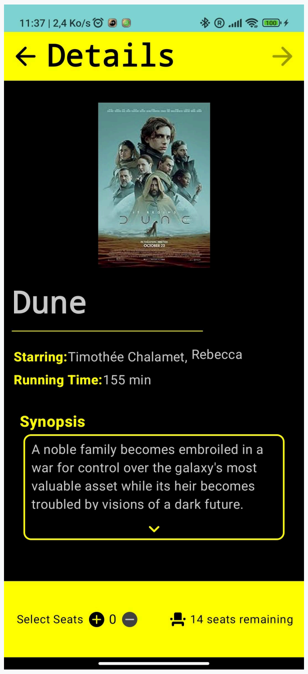


11:37 | 5,6 Ko/s

Home Page

Aquaman and the Lost Kingdom
Starring: Jason Momoa, Patrick,
Running Time: 124 min

Dune
Starring: Timothée Chalamet,
Running Time: 155 min



11:37 | 2,4 Ko/s

Details

Dune
Starring: Timothée Chalamet, Rebecca
Running Time: 155 min

Synopsis
A noble family becomes embroiled in a war for control over the galaxy's most valuable asset while its heir becomes troubled by visions of a dark future.

Select Seats 0
14 seats remaining