SCHOOL OF
COMPUTER SCIENCE

**Technical Design Document**

**CIS4250**

A.Blankenstein, H.Mostafa, J.Subang, K.Steeves

# Table of Contents

**Table of Contents** **1**

**CinemaSwipe** **2**

**Part 1: App Technology, Platform, Tools, and Resources** **2**

**Part 2: App Plan (milestones, roles, and responsibilities)** **3**

Project description 3

Detailed requirements/functionality 4

Milestones 11

Final Release Changes 12

**Part 3: Architecture (Diagrams)** **13**

Use case diagram 13

UI mockups 14

Component / Class diagram 16

Entity / Relationship diagram (or equivalent) 18

# CinemaSwipe

## Part 1: App Technology, Platform, Tools, and Resources

**Target Platform**

Web Application

**Development Platform**

MongoDB, Express, React, Node.js

**Where the source code will be hosted**

Heroku

**List any Third party SDKs or APIs**

- API 1: https://developer.imdb.com/
- API 2: The Movie Database (TMDB) (themoviedb.org)
- API 3: JustWatch - The Streaming Guide

**Identify any existing software that we plan to start with**

None

**List any other Assets**

Images

- Possibly a logo for main page
- User images or avatars
- Movie cover posters will be supplied by the movie database API listed above.

Sounds

- None (possible swiping animation sound if we find time).

**Database and/or virtual servers used to host the project data:**

MongoDB, Express.js

# Part 2: App Plan (milestones, roles, and responsibilities)

## 1. Project description

A web application for choosing movies to watch with friends and family. After creating an account, users can send friend requests to their friends and family to become ‘friends’ on the CinemaSwipe platform. The main page of the site showcases different movies from sources such as IMDB and stream watching guides, allowing users to indicate whether they are interested in watching a given movie or not. Showcasing movies will include presenting details of a movie, such as the box office cover, description, cast, genre and so on. To initiate the movie matching process, users intending to choose a movie to watch should join a ‘room’. If a user ‘likes’ the same movie as a friend (in the same room), both users will be notified of the movie match. Users will have access to their friend list, their liked movie list, and a list of movies that they have *liked* in common with each friend.

**Algorithm Description**

1. User swipes on a given movie within the movie carousel.
2. Upon user liking or disliking a given movie, the algorithm calls appropriate functions for a like or dislike action, respectively.
3. Function creates a new ‘vote’ record in the database containing the user ID, movie ID, and vote (like or dislike).
4. A query will now be run in the Vote database for another record that contains a ‘like’ for the same movie ID, and a different user ID.
    a. If a match is found, create a match record
    b. Call notification function to send a match notification to all user IDs

## 2. Detailed requirements/functionality

Each requirement is broken down into smaller tasks. In the next table, the following requirements are assigned across all team members, and each member is responsible for completing the tasks that are corresponding to their assigned requirement.

| | **Requirements/ Features** | **Description** | **Tasks** | **Priority** | **Dependency** | **Alpha** |
|---|---|---|---|---|---|---|
| **0** | Web Application *Milestone 1* | Main web application accessed as a website by users, allowing them to use the movie matching service. | - Create CSS stylesheet matching UI<br>- Create homepage with app title<br>- Setup main file structure<br>- Design and add app logo | Must have | N/A | Yes |
| **1** | Carousel with movies *Milestone 2* | Carousel that cycles through movies. Users can swipe on each item to indicate whether they want to watch the movie or not. | - Import React Carousel library<br>- Style to match UI<br>- Display movie data correctly<br>- Add like/dislike buttons | Must have | 0,25 | Yes |
| **2** | User profiles and Authentication *Milestone 1* | Users will be prompted to create and sign into a user profile. This will be used to share with friends and to store user data in reference to what movies they have selected to watch. (“swipe right”) | - Import authentication library<br>- Ensure database is correctly setup to save data<br>- Test to ensure login/logout is setup correctly<br>- Setup login/logout | Must have | 7 | Yes |
| **3** | Ability to add, view and match with friends and family accounts *Milestone 2* | Users can send friend requests to each other. If the request is confirmed, the users will become “friends”. This allows the app to determine if a friend wants to watch the same movie | - Setup user IDs in database & authentication<br>- Setup user IDs in database & authentication<br>- Build UI search bar for accounts<br>- Create algorithm to process friend matches | Must have | 2 | Yes |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | as the user. | | | | |
| **4** | Ability to like a movie<br>*Milestone 2* | When displayed a movie, a user can (swipe to the right) to "Like" a movie to indicate to the other friends that they would like to watch that particular movie | - Add like button to UI<br>- Configure like button to add movie to users' liked movie list onclick<br>- Flag movie as seen by user | Must Have | 1,7 | Yes |
| **5** | Ability to dislike a movie<br>*Milestone 2* | When displayed a movie, , a user can (swipe to the left) to "dislike" a movie to indicate to the other friends that they do not want to watch that particular movie | - Add dislike button<br>- Configure dislike button to skip disliked movie suggestions<br>- Add disliked movie to seen list | Must Have | 1,7 | Yes |
| **6** | Ability to “must watch” a movie<br>Milestone 5 | When displayed a movie, a user can (swipe upwards) to "must watch" a movie to indicate to the other friends that they REALLY want to watch that particular movie, doing this will put that movie on the top of the queue of movies to show the other users. | - Add must watch button<br>- Configure must watch algorithm<br>- Add movie to users' must watch movie list | Nice to have | 1,7 | No |
| **7** | Database storage<br>*Milestone 1* | User data and movie preferences will be saved to a firebase storage. This will allow remote sync as well as the ability for users to see when they matched on a movie with their friends. | - Create database schema<br>- Install MongoDB npm library<br>- Create MongoDB Atlas Cluster<br>- Connect to database from Node backend | Must have | N/A | Yes |
| **8** | Platform | In user creation/settings, users | - Ensure there is a "platform" column in the | Nice to | 2 | No |

| | | | | | | |
|---|---|---|---|---|---|---|
| | preference *Milestone 4* | can indicate which streaming platforms they have. This will make the app only show them movies that they have access to | User database record<br>- Allow users to edit their account settings in the app<br>- Have a section in account settings for checking off which streaming platforms are available to the user<br>- Allow users to check off one or more streaming platforms<br>- Save these changes to the user's record in the database | have | | |
| **9** | Have a dynamic landing page with updated movies *Milestone 6* | When a user logs into our program, they should be greeted with the latest popular and new released movies on the front page. | - call the API to deliver the top movies.<br>- pull the movie title, poster, and release date.<br>- display all content on the main page in a pleasing manner. | Nice to Have | 1 | No |
| **10** | Ability to filter movies by genre *Milestone 5* | ex. If one of the users really does not want to watch a horror movie, all movies of the horror genre will be excluded from being displayed in feature #1 | - Have a function that handles populating the movie carousel<br>- Movie carousel function takes in a single argument of parameters<br>- From the UI, have an option for custom carousel settings which has a checklist of genres to allow, (default set to all genres) | Nice to Have | 1 | No |
| **11** | Receiving notifications of movie matches *Milestone 3* | A notification bubble will appear at the top of the web page that will notify the user that a connection has been made and both you and a friend have “liked” the same movie. | - On a single swipe have a function do a database query for other vote records of the same movieID and a userID that is in the swiper's friends list<br>- If there is a match create a new match record in the database containing the movieID and userIDs<br>- For all users that matched, update their | Should Have | 14 | Yes |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | matches column in their User record to include the newly created matchID<br>- Web app will display a popup component indicating that there was a match | | | |
| **12** | Integrating movie trailers in movie description, allowing users to view trailer within the app *Milestone 4* | In the movie description, there will be a trailer for the movie. | - Allow users to tap on a movie in the carousel to "view more details" and swipe through additional detail cards to see the movie trailer card<br>- TMDB api call to get the youtube key using the movie id<br>- Have a function that will append the youtube key to "https://www.youtube.com/watch?v=" + youtube key<br>- Movie trailer card will have a movie trailer react component that will display an embedded youtube video with the youtube link as an argument | Nice to have | 1,21 | No |
| **13** | Movie cast *Milestone 4* | In the movie description, there will be a list of the cast members | - When making an API call for movie descriptions.<br>- Gather data on Cast list.<br>- Save this cast list from API to a new list saved under the movie object.<br>- Add Carousel component to the movie description area.<br>- Display Cast List object within newly created Carousel in description portion. | Nice to have | 21 | No |
| **14** | Friends movie watch list *Milestone 6* | View a list of movies that your friends have selected as “must watch”. | - Create Button to view all of the User's friends and display them in a list.<br>- Make each friend element within the list, clickable.<br>- Create function that searches through a | Nice to have | 3, 4 | No |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | Friend's "must Watch" list.<br>- Display function result as a list to UI. | | | |
| **15** | Personal movie “matched with” list.<br>*Milestone 3* | Users can view a list of movies that they have “liked” and see which of their friends have also “liked” that same movie. | - Add "Matched With" button to Menu<br>- Display new Webapp Page when button is clicked.<br>- Create function that compares User's and Friends "Liked" movies.<br>- Save the result of the compare function to a list or array.<br>- Display list on new webapp Page<br>- (optional) have a subheading on each element that lists friends who "matched" on this movie element. | Must Have | 3, 4 | Yes |
| **16** | Users should be able to access application on any device with a browser.<br>*Milestone 2* | Application should be hosted publicly so no downloading is required as well as having it run on all platforms that have a web browser. | - Host site on hosting server<br>- Develop Desktop Friendly UI (Desktop Mode)<br>- Develop Mobile Friendly UI (Mobile/Tablet Mode)<br>- Ensure UI works on the entire Width Range. (automatically adjusts content based of width) | Must Have | 0 | Yes |
| **17** | Have popular and highly rated movies at the top of the list<br>*Milestone 2* | There are lots of lots in the database to choose from so it would make sense to see popular movies and highly rated movies first | - Make an API call that gathers movies based on date (new release) and popularity or ratings.<br>- Update movie carousel based on API call. | Should Have | 1 | Yes |
| **18** | Ability to filter movies by decade released<br>*Milestone 5* | View and “swipe” through movies that have been filtered by a particular decade that the film was released in. | - Add search bar and button to UI.<br>- Call API to provide a movie's release date.<br>- Add a function that uses User input and matches it to all release dates for each movie. Save movies that match to a new list. | Nice to Have | 1 | No |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | - Display new movie list in carousel. | | | |
| **19** | Ability to filter movies by language *Milestone 5* | Only show movies in a certain language | - Add Search bar and button to UI.<br>- Call API for list of movies and their language settings.<br>- Create a function that iterates through movies and matches the language to that of the user's input. Any matches go into a new list.<br>- Display the newly created movie list in the carousel. | Nice to Have | 1 | No |
| **20** | Notify user of any pending friend requests on sign-on *Milestone 6* | As a user, I want to be able to know if I have any pending friend requests when I first log into my existing account | - Trigger a notification on user sign on if their list of pending friend requests is not empty. | Nice to Have | 3 | No |
| **21** | View details of a movie by tapping or clicking on it. *Milestone 3* | Users can click or tap on a movie while in the swiping system. This will flip over the movie card and display info based on it. | - When movie posters are tapped/clicked, change the poster to a new box that will contain movie details.<br>- Fill the box will movie details from the API.<br>- When the details box is tapped/clicked, change the details box back to the movie poster. | Should Have | 1 | Yes |
| **22** | Change user avatar to any uploaded jpeg or png *Milestone 6* | Allow users to upload a personalized user avatar to represent themselves. | - Create an option in settings that allows the user to change their avatar.<br>- When selected, prompt the user to upload a new image.<br>- Confirm change.<br>- On confirm: overwrite the user's avatar in the database. | Nice to Have | 2 | No |

| | | | | | | |
|---|---|---|---|---|---|---|
| **23** | Add ability to use keyboard arrow keys to “like, dislike, Must watch” a movie. *Milestone 6* | Users can use the left, right and up arrow keys to swip movies that come up. This will allow users on a laptop or desktop to easily “like, dislike, and Must watch” a movie. | - Add functionality so that when the right arrow is pressed, a movie is liked.<br>- Add functionality so that when the left arrow is pressed, a movie is disliked.<br>- Add functionality so that when the up arrow is pressed, a movie is must watched. | Nice to Have | 1 | No |
| **24** | Ability to join & leave a movie room/group. *Milestone 3* | A room where friends’ movie votes are processed to find possible matches. | - Add a settings option to join a movie room/group.<br>- Enter the group's name/code to confirm.<br>- Display the group in the user's group list. | Should Have | 3,5 | Yes |
| **25** | Connect application to API *Milestone 1* | Connect application to Movie Database API and be able to manipulate movie data. | - Connect to the API in the backend.<br>- Store info received from API in a data structure that is easy to use/navigate. | Must Have | 0 | Yes |

## 3. Milestones

**Alpha Release**
All **must have** requirements and **should have** requirements will be completed in milestones 1, 2 and 3 denoted by blue in the table below.

**Final Release**
All **nice to have** requirements are part of milestones 4, 5 and 6 denoted by green in the table below.

| Milestone | Deadline | Description | Tasks / Requirements | Team Member(s) Leading Task |
|---|---|---|---|---|
| 0 | Feb 7 | Create Technical Design Document | Class Diagram | Miles |
| | | | Use Case Diagram | Alex |
| | | | UI Mockups | Kaitlyn |
| | | | E/R Diagram | Habiba |
| 1 | Feb 18 | Complete React Native App starter app & Database/API Setup | Req.7. Database storage | Miles |
| | | | Req.25. Connect Application to API | Alex |
| | | | Req.0. Web Application | Habiba |
| | | | Req.2. User profiles and Authentication | Kaitlyn |
| 2 | Feb 25 | Finalize Movie, Friends, Likes and Dislikes | Req.16. Access and Use Application on any Device | Alex |
| | | | Req.17.Have Popular and New Release Movies appear first in the movie carousel. | Alex |
| | | | Req.1. Carousel with movies | Kaitlyn |
| | | | Req.3. Ability to add, view and match with friends and family accounts | Miles |
| | | | Req.4. Ability to like a movie | Habiba |
| | | | Req.5. Ability to dislike a movie | Habiba |
| 3 | Mar 4 | MVP (Alpha) Complete Main Matching | Req.15. Create user's “Matched With” list. | Miles |
| | | | Req.11. Receiving notifications of movie matches | Kaitlyn |

| | | | | |
|---|---|---|---|---|
| | | Algorithm | Req.24. Ability to join a movie room/group | Habiba, Miles |
| | | | Req.21. Show movie details by tapping/clicking on it | Alex |
| 4 | Mar 18 | Polish MVP | Req.12. Integrating movie trailers in movie description | Habiba, Alex |
| | | | Req.8. Platform preference filter | Miles |
| | | | Req.13. Add Movie Cast to Description. | Kaitlyn |
| 5 | Mar 25 | Apply Advanced Movie Filtering | Req.18. Filter movies by ~~Actor/Actress~~ Year | Alex |
| | | | Req.6. Ability to “must watch” a movie | Miles |
| | | | Req.10. Ability to filter movies by genre | Kaitlyn |
| | | | Req.19. Filter movies by Language | Habiba |
| 6 | Apr 9 | Final Release | Req.9. ~~Swiping animation and component to movie selector~~ Make home-page dynamic, displaying movie cards with Title & Release Date | Alex |
| | | | Req.20. ~~Send movie suggestion to a friend~~ Notify user of any pending friend requests on sign-on | |
| | | | Req.22. Change User Avatar | Alex, Miles |
| | | | Req.23. Add ability to use keyboard arrow keys to “like”, “dislike” or “must watch” a movie | Kaitlyn |
| | | | Req.14. Add ability to view Friend’s “must watch” | Habiba, Miles |

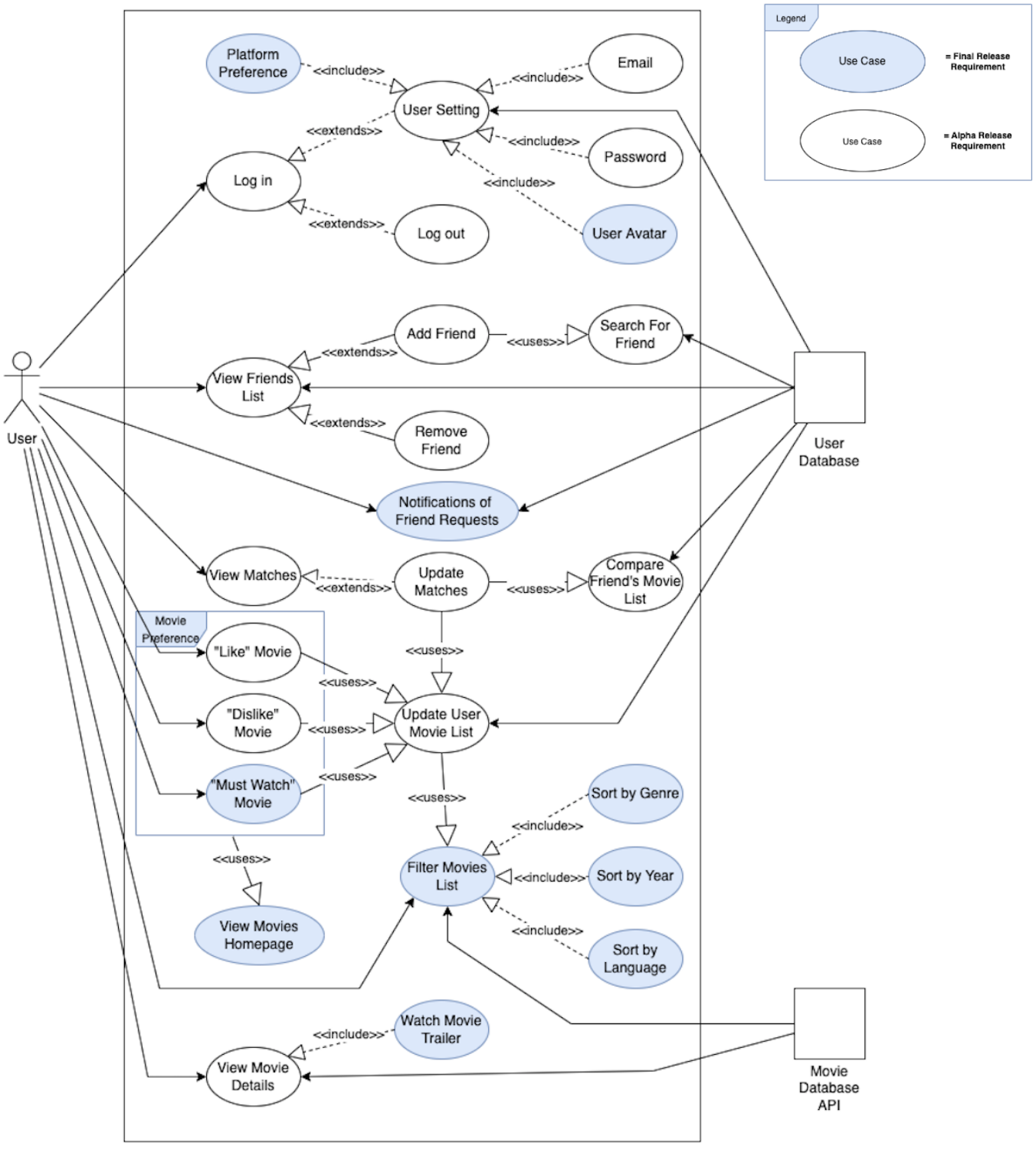
## Final Release Changes

Requirements that have been stroked off have undergone changes for the final release. As we revisited the 3 requirements above in milestones 5 and 6, we analyzed our user criteria, assessed our overall user experience and concluded that the requirements we had previously do not add as much value to our current product. Therefore, we replaced them with the requirements above to enhance our user experience and achieve our revisited goals. Requirements 9, 18 and 20 are the only ones that have undergone any changes, and were updated accordingly along with the Use Case Diagram.

## Part 3: Architecture (Diagrams)

1. Use case diagram

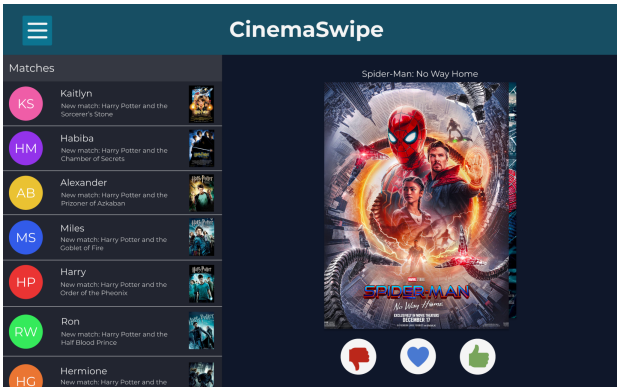
**Alpha** functionality = Transparent/White-filled Use Cases
**Final Release** functionality = Blue-filled Use Cases



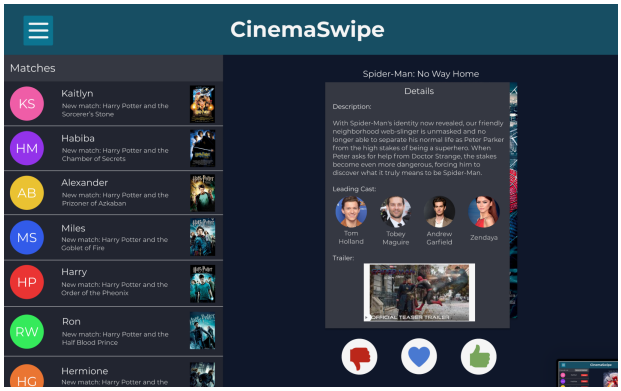
## 2. UI mockups

**Desktop**

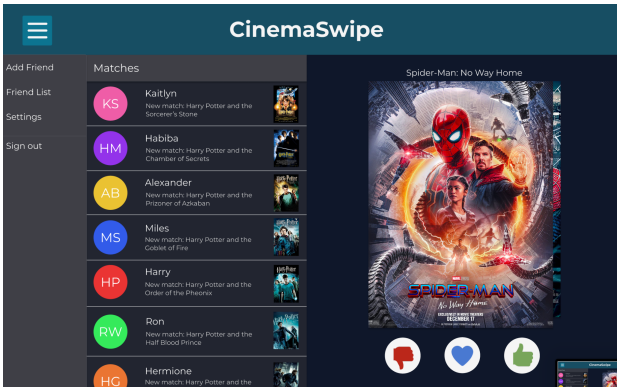
Main Screen & Matches



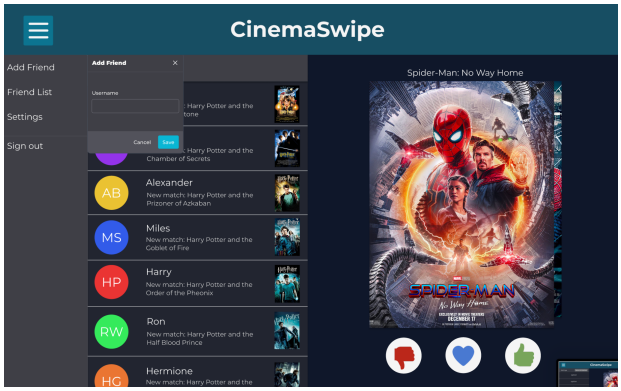

Movie Info



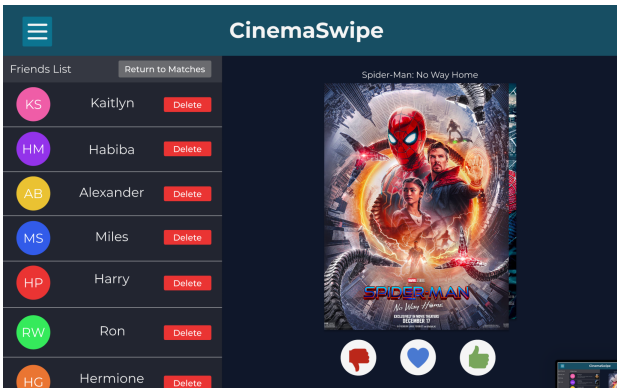

Menu



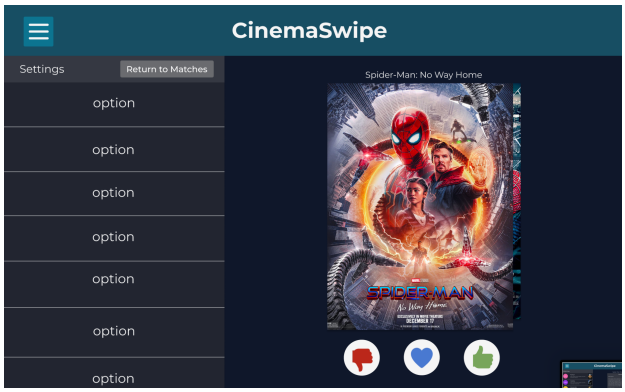

Add Friend




Friends List



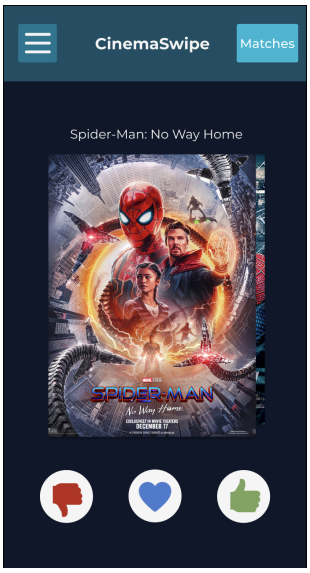

Settings



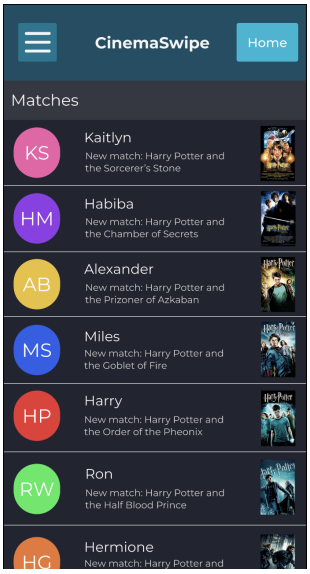
## Mobile

Our main design is for desktop, hence our web application and choice of tech stack. However, if time allows post Alpha release, we will look into developing a mobile version and have put together a mobile mockup below.

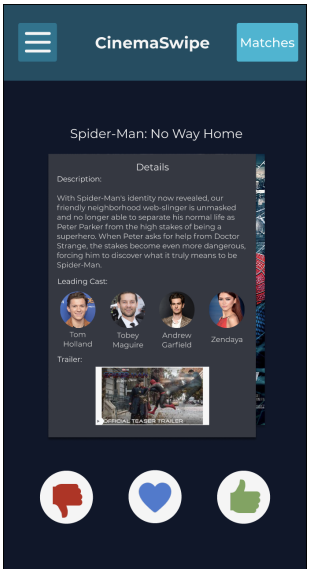
Main Screen



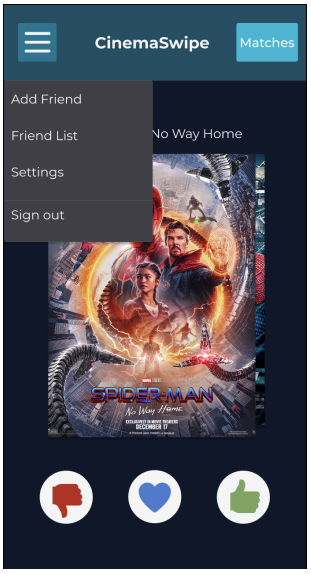

Matches



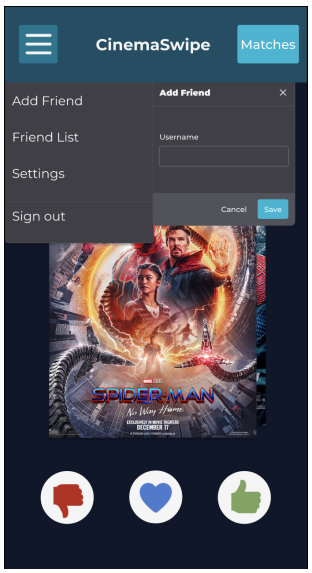

Movie Info



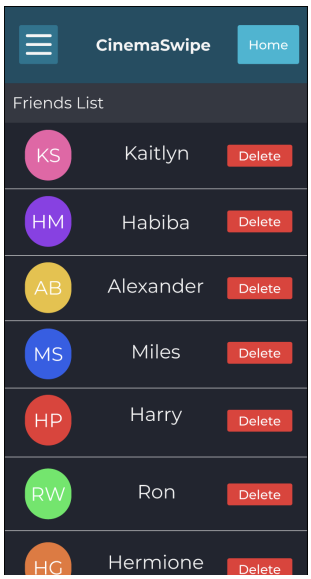

Menu



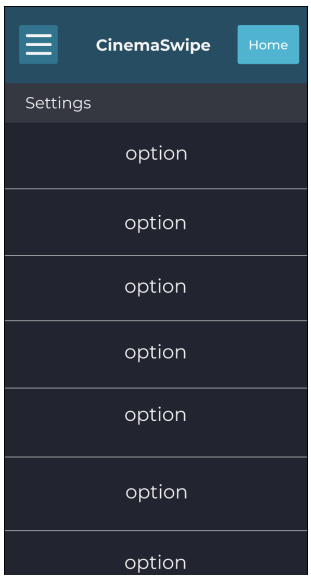

Add Friend




Friends List

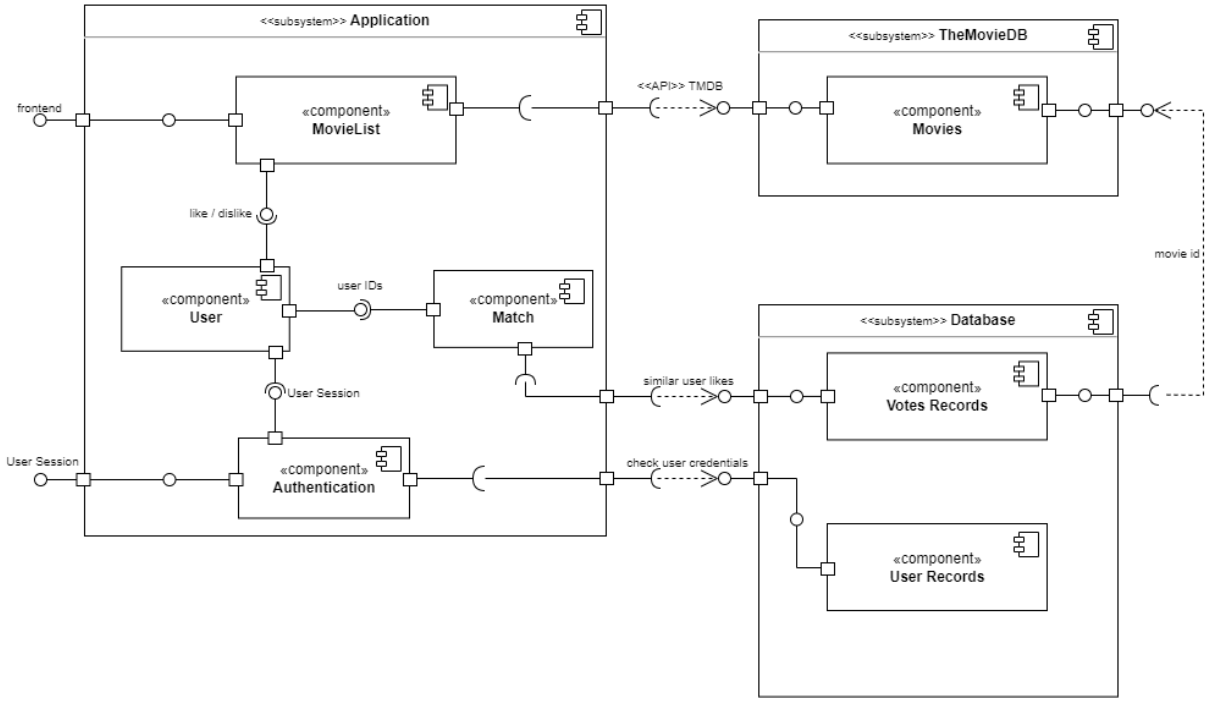



Settings

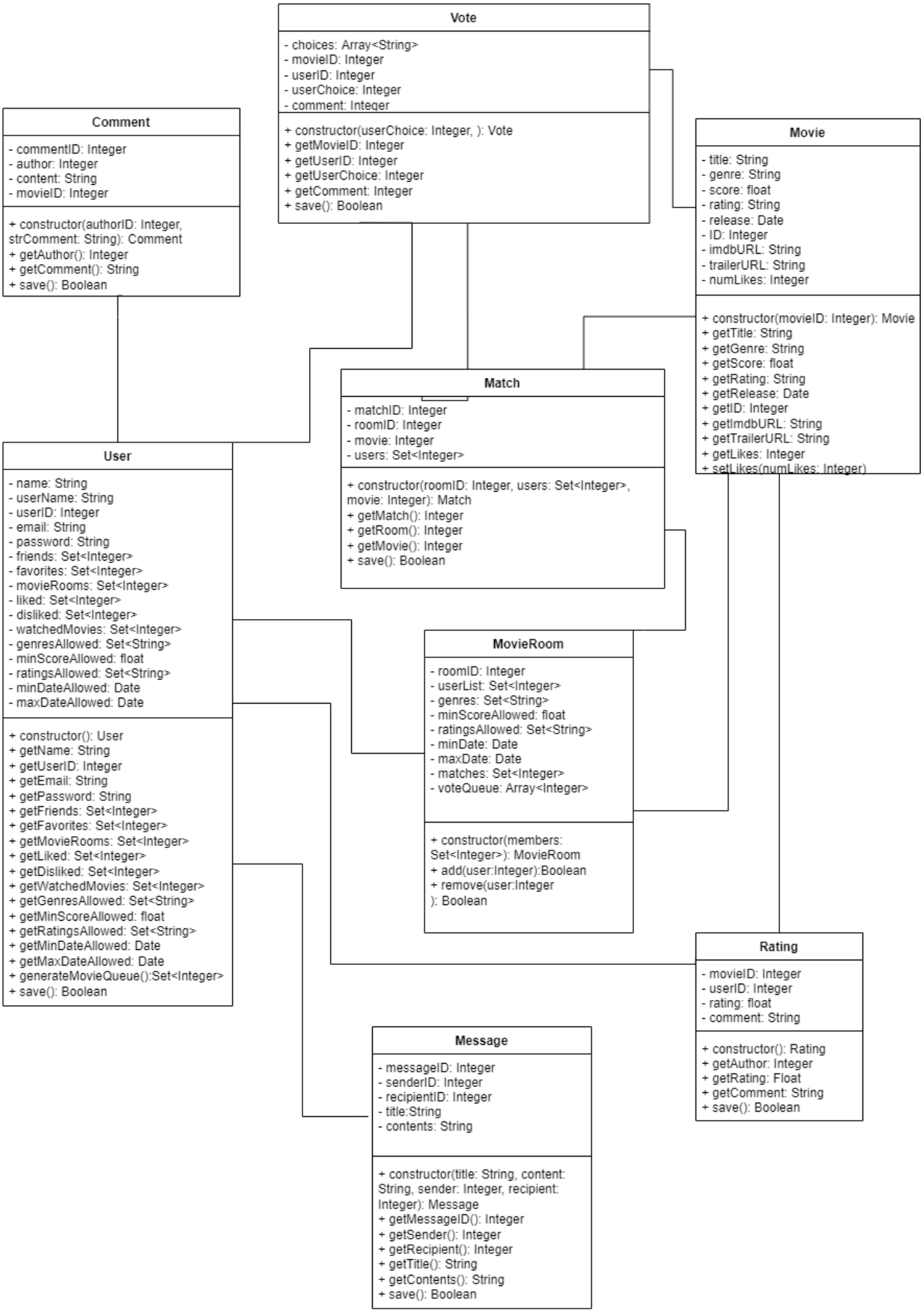


*Link to Figma Mockups*
https://www.figma.com/proto/TzFnr7zXsZ2r9Ol7RT21Nl/Application-Design?node-id=0%3A1&scaling=min-zoom&page-id=0%3A1&starting-point-node-id=2%3A13619&show-proto-sidebar=1

## 3. Component / Class diagram



# Class Diagram



4. Entity / Relationship diagram (or equivalent)