SW Engineering CSC 648/848 Section 1

Ensight

"Your Fellow Cinephile"

Milestone 2

(10/11/23)

– **App Name:** **ensight.space** (where en-tertainment meet in-sights)

– **Section 01, Team 02**

- Pragati Makani: Team Leader
- Sarah Abusaif: Front-end Lead
- Alexander Del Rio: Back-end Lead
- Mohammad Dahbour: Scrum Master/Backend
- Christian Montalvo: GitHub Master/Frontend

– **History Revisions**

Revision Id	**Revision Date**	**Milestone**
Version 0.1	9/27/23	1
Version 0.2	10/11/23	2

1. Data Definitions V2:

- Revisions from Milestone 1:

Name	Definition	Usage
User	Represents individuals who register and use the application	Users can log in, rate films, write reviews, create and share film lists, and engage in social interactions.
Review	Represents a user's written critique and analysis of a movie.	Reviews are created by users to share their thoughts on movies they've watched.
Comment	Represents a user's comment on other user-written reviews.	Comments are assigned by users to express their opinions on someone's review.
Movie	Represents a film title with associated details, such as title, release date, and genre.	Movies are logged by users when they watch films and are used in reviews, lists, and ratings.
Statistics	Represents aggregated data about a user's viewing habits.	Statistics Seekers can view and analyze their film-watching patterns.

- Data Definitions Continued:

Primary Data	Sub-Data
User	• id • email • password • reviews • comments • watchlist • lists • favorites • following • followers
Review	• user • movie • review text • comments • datetime • likes
Comment	• user • comment text • datetime • likes
Movie	• name • image • release date • synopsis • cast • crew • details • genres • stream providers

	• reviews • review average • lists containing • watchlist count • similar
Statistics	• user • moviesWatched • genreBreakdown • totalRatingsGiven • averageRating • reviewsPosted • listsCreated • followersCount • followingCount

2. Functional Requirements V2:

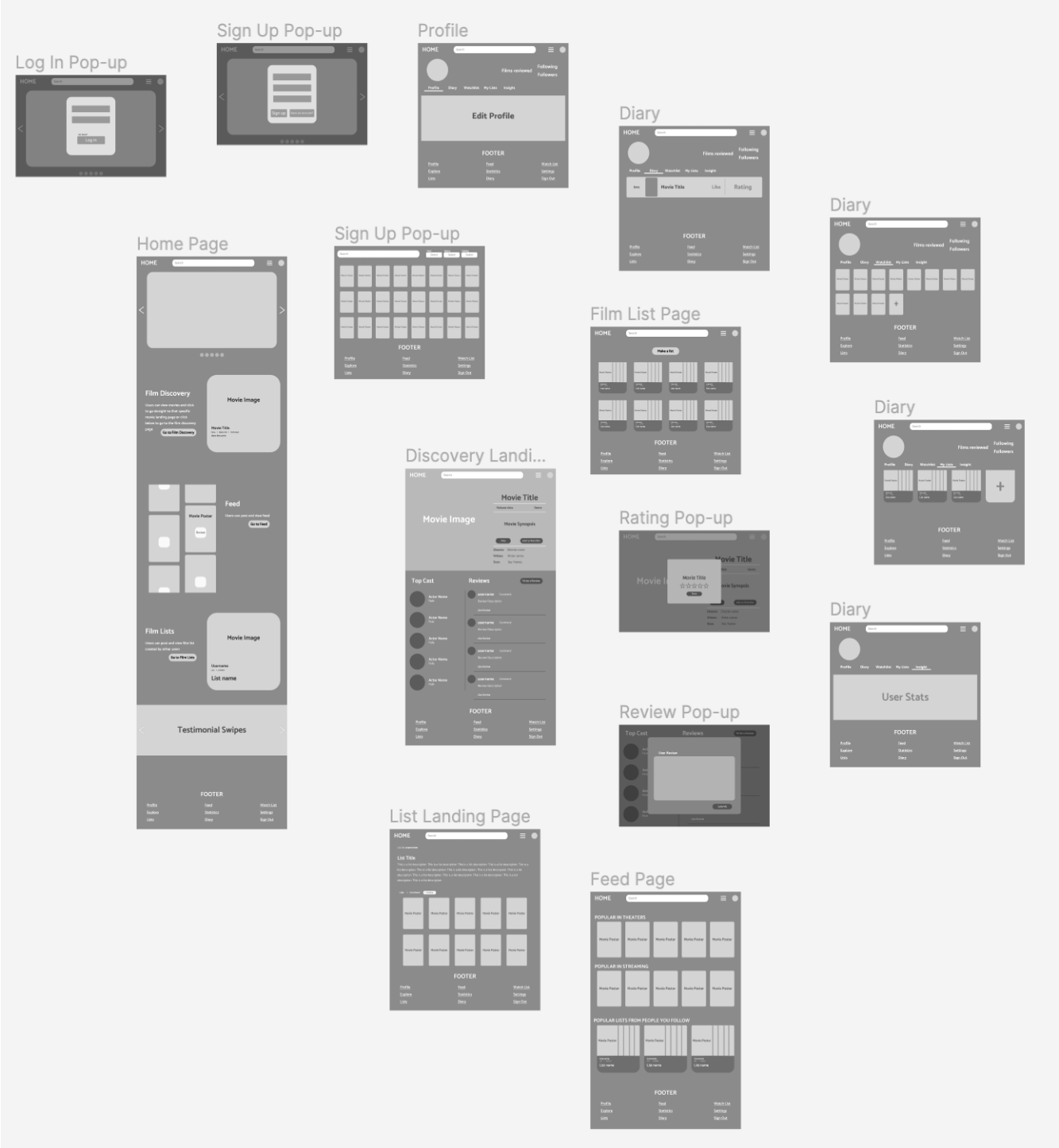
No.	Description	Details	Priority
1.	Users can register and log in securely for their Ensight account	1.1) Users can register for a new account 1.2) Users can login with their existing Ensight account	Must-Have
2.	Users can write reviews with shared ratings	2.1) Users can write a movie review 2.2) Users can rate a movie	Must-Have
3.	Users can create, edit, and share film lists	3.1) Users can create a film list 3.2) Users can edit their existing film list 3.3) Users can share their film lists	Must-Have
4.	Users can like and comment on user reviews and follow other users	4.1) Users can like reviews posted by other users 4.2) Users can comment on reviews posted by other users 4.3) Users can follow other user-generated lists	Must-Have
5.	Users can browse through film directory	5.1) Users can browse film directory and look for films, tv shows and people	Must-Have
6.	Users access detailed statistics on film-watching habits	6.1) Statistics can include the number of movies watched in total by year 6.2) Genre Breakdown of how many movies of each genre they watch 6.3) Average rating given to the	Desired

		user will be counted 6.4) Total lists created by user	
7.	Users maintain a chronological record of watched films	7.1) Users should be able to log movies they have watched by specifying the movie title, date watched, and optional additional details like rating and review. 7.2) Users should have the ability to edit or delete diary entries to correct mistakes or update information. 7.3) Users can choose to keep their diary entries private or share them with their followers and the community. 7.4) Users should be able to filter their diary entries by date, allowing them to see what they watched on a specific day or within a date range. 7.5) Users should have a dedicated section where they can view their diary entries in chronological order.	Desired
8.	Users will have the ability to update their account profile.	8.1) User can reset or change their password 8.2) User can change their name and bio	Opportunistic
9.	Users will receive personalized film recommendations based	9.2) Users will receive personalized film recommendations based on their generated film lists	Opportunistic

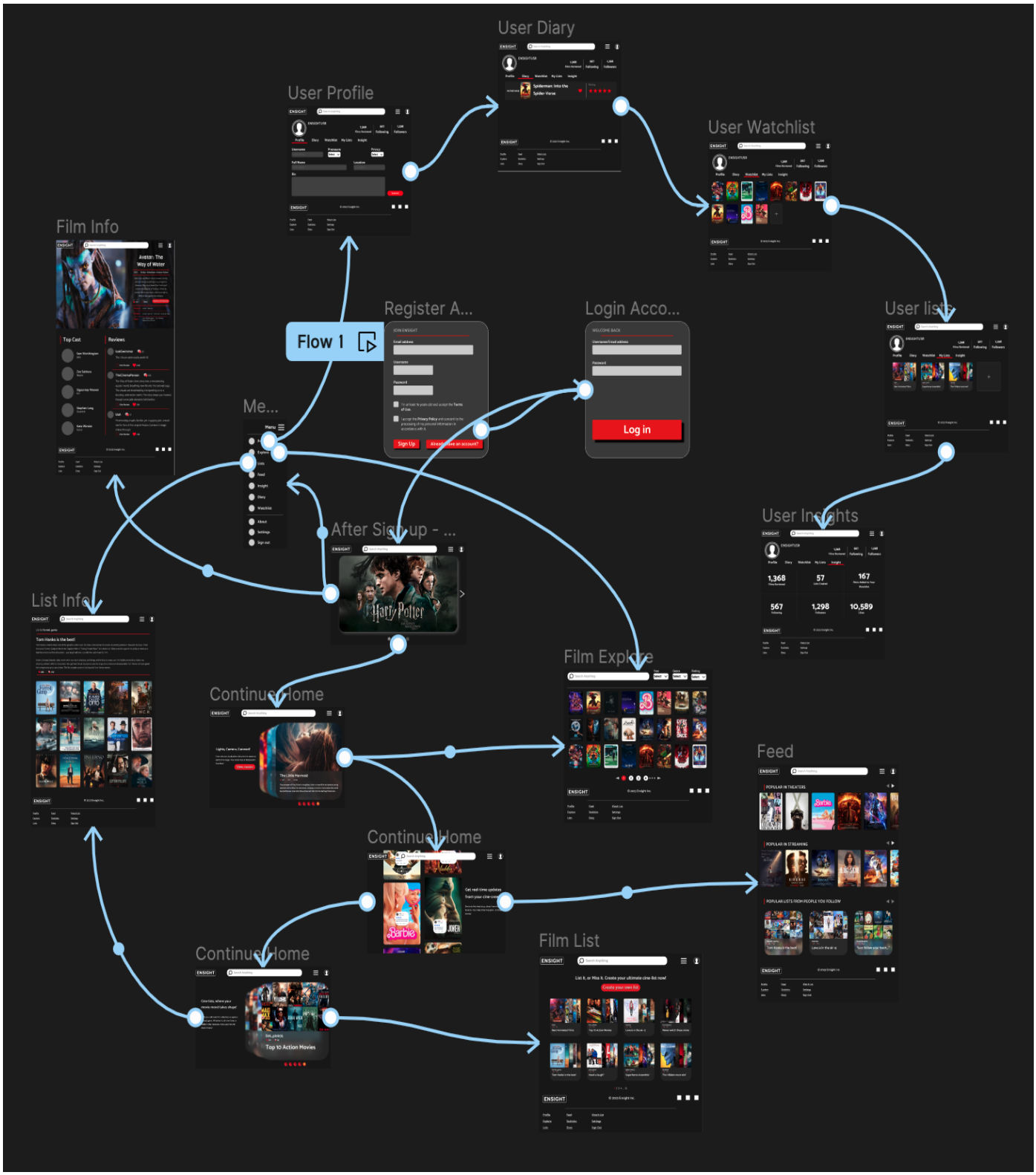
	on their generated film lists	9.3) Recommendation lists can be generated from genre, director, writer, and other films that are commonly found on user generated lists	
10.	Users can log in securely with google single sign-on	10.1) Users can login with Google	Desired

3. UI Mockups and UX Flows:

- Gray and White WireFraming



- [GUI Prototype created by Figma]



4. High level Architecture, Database Organization:

- Database Organization

User

id	int PK
username	varchar(150)
password_hash	varchar(128)
email	varchar(254)
last_login	datetime(6)
date_joined	datetime(6)
image_path	varchar(100)

Movie

id	int PK
title	varchar(256)
release_date	date
description	varchar(1024)
rating_count	int
rating_average	decimal(3,2)
poster_path	varchar(100)

Review

id	int PK
author_id	int FK -> user.id
movie_id	int FK -> movie.id
title	varchar(512)
rating	decimal(2,1)
text	longtext
created_at	datetime(6)

Genre

id	int PK
name	varchar(64)

Movie_genres

id	int PK
movie_id	int FK -> movie.id
genre_id	int FK -> genre.id

MovieList_Movies

id	int PK
movielist_id	int FK -> movielist.id
movie_id	int FK -> movie.id

MovieList

id	int PK
author_id	int FK -> user.id
title	varchar(512)
description	longtext
created_at	datetime(6)

Comment

id	int PK
author_id	int FK -> user.id
review_id	int FK -> review.id
text	longtext
created_at	datetime(6)

Add/Delete/Search Architecture	Functional Requirement
Add/Delete/Search for Users	Users can register accounts
Search/Display for Movies	Users can browse through the film directory
Search/Display for Genres	Users can browse films by genre
Add/Delete/Search/Display for Reviews	Users can write reviews with ratings
Add/Delete/Display for Comments	Users can comment on other users' reviews or lists

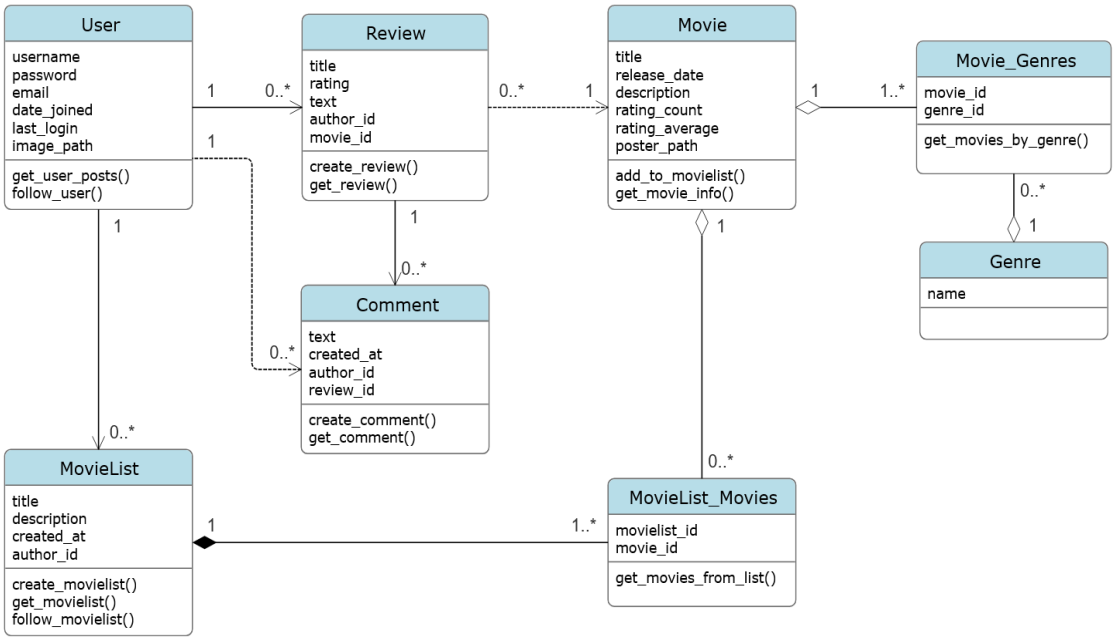
Add/Delete/Search/Display for MovieList	Users can create and edit film lists
None for MovieList_Movies	This is a junction table for a many-to-many relationship between Movies and MovieLists
None for Movie_Genres	This is a junction table for a many-to-many relationship between Movies and Genres

- APIs

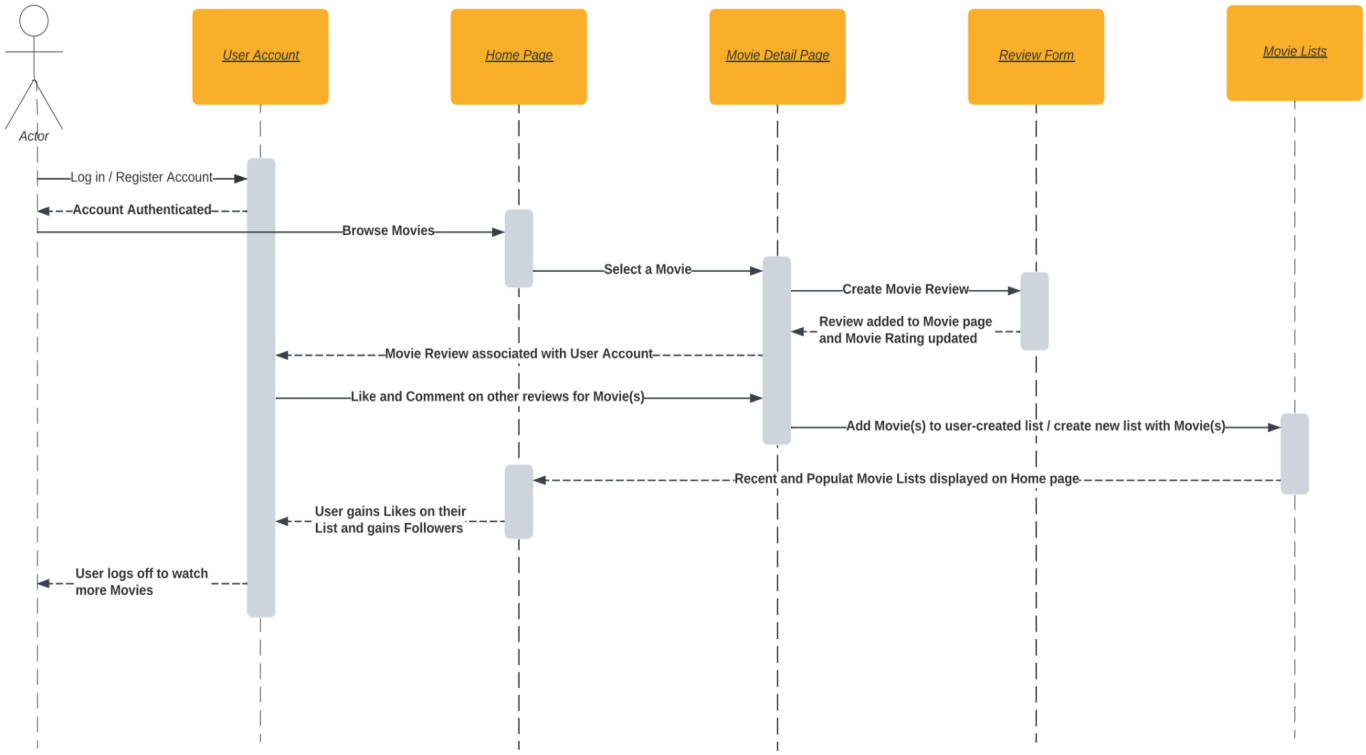
  Our backend leverages Django's powerful built-in features, including its ORM (Object Relational Mapping), to efficiently handle database operations such as adding, searching, and deleting records. Django's ORM streamlines the interaction between our application and the database, making these operations seamless and reducing the complexity of SQL queries. In our architecture, we are planning to utilize the TMDB database (third-party API) to enhance the user experience and enrich our application's functionalities. Notably, we're exploring the depth of integration with TMDB (The Movie Database) API to ensure seamless retrieval of movie data. This integration aims to provide users with up-to-date information on films, enhancing their interaction with the platform.

5. High Level UML Diagrams:

- High-level UML Class Diagram



- High-level Sequence Diagram



6. Current Actual Key Risks:

❖ **Skills risks and mitigation plan**

- ➢ Lack of experience with UI/UX design tools: If team members are unfamiliar with the tools for creating mockups, delays may occur in the milestone.
  - To resolve:
    - Allocate specific training sessions.
    - Pair an experienced member with a novice.
    - Encourage sharing of tutorials and online resources
- ➢ Inadequate knowledge of database design: If data definitions are improperly created, the application's data processing can be flawed.
  - To resolve:
    - Ensure understanding of data relationships.
    - Regularly review database design principles.
    - Use database design tools to visualize and validate.

❖ **Schedule risks**

- ➢ Incomplete mockups delaying next steps: If UI mockups are not completed on time, subsequent steps like integration and testing may be delayed.
  - To resolve:
    - Allocate buffer time after mockup completion.
    - Use tools like Figma for rapid prototyping.
    - Hold regular progress checks.
- ➢ Misalignment in data definitions: If team members have different understandings of data definitions, it can cause confusion and rework.
  - To resolve:
    - Create a shared documentation platform.

- Designate a data definition owner for reviews and clarifications.
- Organize regular sync-up meetings.

❖ **Teamwork risks**

➢ Inconsistent design vision: If team members envision different designs for the application, it can lead to rework and disputes.

- To resolve:
  - Set clear design guidelines.
  - Regularly review mockups as a team.
  - Encourage feedback and iteration.

➢ Disjointed front-end and back-end development: If there's a lack of collaboration between front-end and back-end teams, integration issues might arise.

- To resolve:
  - Encourage cross-functional meetings.
  - Clearly define APIs and integration points.
  - Use version control effectively.

➢ Inefficient team coordination: If team members aren't aware of each other's work, overlapping efforts or misalignment can occur.

- To resolve:
  - Use project management tools [monday.com].
  - Define clear roles and responsibilities.
  - Foster an open communication environment.

❖ **Legal/content risks**

➢ Improper usage of copyrighted movie posters and stills: If copyrighted content is used without permissions, legal repercussions can ensue.

- To resolve:
  - Obtain necessary permissions for content.

        - Use royalty-free alternatives when possible.
        - Educate the team on copyright laws and regulations.
- ➢ User-generated content violations: If users post copyrighted content or inappropriate reviews, the platform can face legal and reputation risks.
    - To resolve:
        - Implement content moderation tools and protocols.
        - Provide clear guidelines to users.
        - Set up a reporting and take-down mechanism.

7. Project Management:

For M2, our team is made up of five members with specialized skills in various areas. Our project management structure ensures that everyone is fully aware of the tasks they are responsible for, and there is transparent sharing of progress at every stage.

Each team member's task is documented and tracked using monday.com, a collaborative tool where tasks are assigned to individual members and are monitored for progress. This tool has been instrumental in ensuring that every member is held accountable for their assigned tasks and that there is clarity on what is expected from each person at every stage of the project.

During our scrum meetings, which we hold bi-weekly, each member gives an update on their progress. We have employed the use of a digital Kanban board on monday.com to make this process more visual and interactive. Every task is represented by a card, which moves across various columns like "Not started", "Working on it", and "Done" as work progresses. This visual aid has helped in ensuring that every team member understands the stage at which every other

person is, promoting transparency and encouraging collaborative effort to overcome challenges.

For instance, two of our developers were responsible for creating UI mockups, and their progress was visible to all team members on monday.com as well as Discord. Every update, challenge encountered, and milestone achieved was recorded on the card associated with that task and posted on discord channel. It made it easier for other team members to chip in with advice, assistance, or resources to ensure that the task was completed efficiently.

In terms of task allocation for M2, two members worked on UI mockups, utilizing their expertise in UX/UI design to create intuitive and visually appealing designs. They used tools like Figma to share their designs transparently with the team. Our backend lead was tasked with data definitions, ensuring that our application's data structure is robust and efficient. Both frontend and backend teams worked together on developing a vertical prototype, laying the groundwork for the application's core functionality.

This organized approach to task allocation and progress tracking ensured that every team member was aware of their responsibilities and the expectations placed upon them. Our transparent communication, aided by the use of monday.com and discord for task management during our scrum meetings, has instilled a sense of collective responsibility and unity in our team, driving us towards successful completion of M2. We plan to iterate and improve upon this approach as we move forward to do even better in the next steps.

—---------- Thank you —----------