

Terminal App T1-A3

FilmSpot: a terminal program allowing users to search, select, get recommendations and be tested on the top 1000 greatest movies ever made

Video walkthrough - <https://vimeo.com/753217247>

Logic Design

Class "App_dataframe"

Class variable: IMBD database stored as a class variable

Staticmethods (I.e. class functions)

1. Main menu function (access all staticmethods in program)
2. Database Search function (accesses and manipulate the class variable)
3. Trivia game function (accesses class variable)

Class "User"

IMBD database inherited from parent

Instance variables: `self.username`, `self.fav_movies`

Staticmethods (I.e. sub-class functions)

1. Create/store user instance (+access all class functions)
2. Recommendation function + Store recommended movies into User instance
3. Display user watchlist

Search/filter function checklist (trello card)

Build out a user filter/search system

in list [Done](#)

Description

Edit

search function that queries the database, using input w/ pandas loc.[]

Checklist

Delete

0%

initialise a static method within the dataframe class.

build input function that accepts only a 'titled' string value

write initialiser prompt that asks the user how they wish to search the primary Dataframe: title, cast member, year, director

make if/else block within an infinite while that returns a dataframe with rows that match the users input

Ensure errors are caught either with else or try blocks

ensure user can return to main menu when entering [back]

ensure while loop is infinite and user can keep searching if they want to

Add an item

Add to card

Members

Labels

Checklist

Dates

Attachment

Cover

Custom Fields

Add dropdowns, text fields, dates, and more to your cards.

Start free trial

Power-Ups

+ Add Power-Ups

Automation

+ Add button

Trivia function checklist (trello card)

Build out trivia component

in list [Done](#) 🚩

Members

SG

+

Description

Edit

0%

✓ Checklist

Delete

☐

Create staticmethod within primary database class that is accessible in the main menu function

☐

Allow user to decide when to start by inputting 'start'. Start will initiate an infinite while loop that ends when the user enters back to the menu

☐

Once started, fetch a RANDOM row from the database and print only the value string from the 'synopsis' column. User is then prompted to input what they think the title of the movie is.

☐

Part of the trivia game is a hint option: if the user enters 'hint', then print the value within the 'director' and 'year' column of the same row

☐

add a counter component for both the round (how many

Add to card

Members

Labels

Checklist

Dates

Attachment

Cover

Custom Fields

Add dropdowns, text fields, dates, and more to your cards.

Start free trial

Power-Ups

+ Add Power-Ups

Automation

+ Add button

Recommendation
+ account creation
component
checklist (trello
card)

build a 'reccomendation' algo/function

in list [To Do](#)

Description

Edit

The reccomendation function should return a list of 10 movies that are losely similar and genre and identical in classification as a . User then has the option to add these to their 'watchlist'.

Checklist

Delete

0%

☐

Modify any returned dataframes to include only columns:
Name, Genre1, Genre2, Director and Classification.

☐

Write input logic that takes a users input keyword and returns the top 10 results of the app_df that share the same Genre 1, Genre 2 and classification as the users inputted movie title

☐

prompt the user to choose whether they want to add the list to their 'watchlist', user must be able to create an account (instance) which can be stored with one of the attributes being their own movies dataframe

☐

to add to the users watchlist, their existing instance (name, watchlist) is loaded via Pickle, the dataframe is appended with the recommended list, then saved again (overwritten) via the pickle.

Add to card

Members

Labels

Checklist

Dates

Attachment

Cover

Custom Fields

Add dropdowns, text fields, dates, and more to your cards.

Start free trial

Power-Ups

+ Add Power-Ups

Automation

+ Add button

View Watchlist component checklist (trello)

Display user watchlist

in list To Do

Start date

yesterday ▾

Description

Edit

User enters 2 on the main menu will be taken to the function that displays their saved/pickled watchlist. Requires authentication and instance attribute print.

Checklist

Delete

0%

☐

create a static method within the User class that links to option 2 on the main menu function

☐

user prompt (converted to string title) that accepts a user name, build ifelse and Try block to catch any user name that does not exist.

☐

if username exists, unload and print their associated watchlist attribute which is stored/serialised within the program file folder

Add an item

Add to card

Members

Labels

Checklist

Dates

Attachment

Cover

Custom Fields

Add dropdowns, text fields, dates, and more to your cards.

Start free trial

Power-Ups

+ Add Power-Ups

Automation

Main menu checklist

Build a main-menu function/component

in list [To Do](#)

Description

Edit

Checklist

Delete

0%

☐

Build a staticmethod within the apps primary database class.

☐

this method prints (str) four options the user is prompted to chose from (string converted integer).

☐

Using an if else block within a infinite while loop: run each component function depending on the user input - 1. Search function, 2. user watchlist display, 3. reccomendation function, 4. trivia

Add an item

Add

Cancel

Assign

Due date

@

Add to card

Members

Labels

Checklist

Dates

Attachment

Cover

Custom Fields

Add dropdowns, text fields, dates, and more to your cards.

Start free trial

Power-Ups

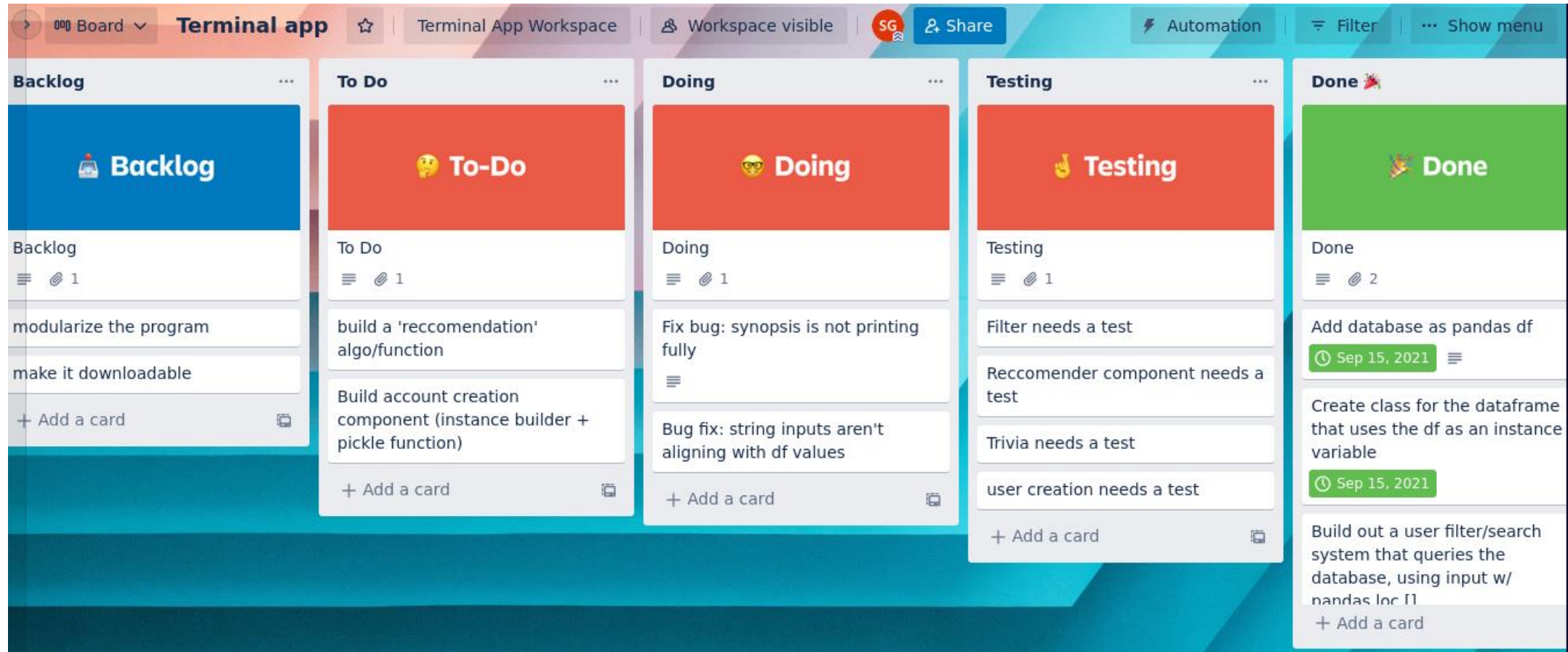
+

Add Power-Ups

Automation

i

Implemented via Kanban on *Trello*



(Early stage kanban board)

Final notes/caveats/challenges...

- Only have implemented 3 Try block error handlers throughout the app. This was due to how primitive the user inputs (e.g. no dealing with integers) and (lack of) 'algorithms' are - i.e. exceptions weren't warranted for each conditional block.
- Fell a bit behind OOP content. Where I usually use challenges to grasp concepts I've had to use this assignment as a learning exercise:
 - This meant I had less time on refactoring and creating modular code as well as implementing more robust tests (e.g. putting my 2 classes in separate modules – it kept causing a circular import error I need to learn more about)
 - **End result** is I have learned a solid foundation on OOP and learned enough to know shortcomings in the code/what to improve for next time