Topic	Flask Mockup 1	
Class Description	**Students brainstorm the app they are aiming to build on Movie Recommendation and then build the Flask API for the first screen of it.**	
Class	**C141**	
Class time	**45 mins**	
Goal	• Brainstorm on the app between the student and the teacher • Build Flask API for the first screen of the App	

Resources Required	• Teacher Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	• Student Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen	

Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **15 min** **20 min** **5 min**

CONTEXT

- **Review the concepts learned in the earlier classes**

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	Hi <Student Name>! As of now, you must have built a lot of great apps and games using different technologies! You must have experienced getting recommendations on various	

	platforms on the internet, but what if we could build a dedicated app that can provide users with the recommendations on what to watch next based on what they have already watched? We are going to build exactly that!	
	What do you think, how should this app be like in terms of User Experience and User Interface? *<Get the student to think about the app>*	**ESR:** varied
	Great! Now let's brainstorm and start building it!	
Teacher Initiates Screen Share		
CHALLENGE • **Brainstorm the app together and arrive with a set of features that the app might require!**		
Step 2: Teacher-led Activity (15 min)	*<This is a guided project. The student is free to add their own creativity to the app, but some basic templates needs to be followed>* *<The app will consist of 2 screens. The first one would be to take user's input on the type of movie that the user likes and the second one would be to provide recommendations to the user>*	

	<We encourage the student to think and come up with a UI/UX design of their own, however the functionality should be more or less the same> *<For reference purposes, we will be providing with a sample app that we are going to create and the code explanations would be mentioned in the document>*	
	<Student can either build a similar app or add their own unique touch to the application> *<Let the student lead the coding part and help them wherever required.>* *<Teacher can refer to the sample code and explanation to help student on how a problem can be solved>*	
	Alright! Now let's think about the App first! *Brainstorm about the app with students.* The app, to begin with, will have 2 screens. The first screen would be to get the user input and the second one will be to show the recommendations to the user.	*Student brainstorms.*
	Let's take this screen by screen and only focus on the backend APIs for the first screen in this class! The first	**ESR:** The first page should include -

	screen would be where we will be taking input from the user. *Student thinks about the features* What do you think should be the features of the first page?	~ User's ability to see a movie ~ User can mark the movie as liked ~ User can mark the movie as not liked ~ User can say that they did not watch a movie
	<Note - The backend's functionality would be the same, however the student can come up with their own logic for it> Great! Now, we can simply build 4 APIs for these 4 functionalities. The first API should be a GET request, where we get the details about a movie. All the other APIs (To mark a movie as liked, not liked or did not watch) would be POST requests.	
	Also, how do we store all this data?	*The student thinks.*
	We can create 4 different lists. • List of all the movies • List of movies liked by the user • List of movies the user did not like • List of movies that the user did not watch	

	As the student marks the movie as liked, not liked or did not watch, we can remove that movie from the list of all the movies and add it to the designated list. Can you tell me why we need to do that?	**ESR:** So that we do not show the same movie again and again to the user and have duplicates in our data.
	Yes! We will keep only one entry of the movie in any of the 4 storages to avoid having any duplicates. This means that if a user likes a movie, we will first have to remove that movie from our list of all the movies and then add that movie in our list of movies that the user likes. Okay, now we are ready to start coding. First, we need to download the data from the Google Colab. We did a lot of processing on our data and then created the DataFrame. We do not want to lose all that! For that, we will first export the DataFrame into a new CSV and then download that CSV from Google Colab.	

```
from google.colab import files
df2.to_csv('movies.csv')
files.download('movies.csv')
```

Here, we are saving our DataFrame into a CSV **movies.csv** and we are then downloading the files with the Google Colab command.

Pretty similar to how we upload a file!

Note - In the CSV that is downloaded, there needs to be added an extra "id" in the beginning of the headers before the first comma, since exporting a DataFrame into a CSV misses out the name of the first column.

```
1  id,index,budget,genres,
2  0,0,237000000,"['action
```

Teacher Stops Screen Share

Now it's your turn. Please share your screen with me.

- **Ask Student to press ESC key to come back to panel**
- **Guide Student to start Screen Share**
- **Teacher gets into Fullscreen**

ACTIVITY

- **Student codes to build the APIs in Flask and uses the data that we just downloaded**

Step 3: Student-Led Activity (15 min)	*Help the student set up a basic Flask Project inside virtual environment*	*The student sets up a Flask Project.*
	Help the student in reading the data from the CSV file into a python list. *This will be the list of all the movies.* *Also create the other 3 empty lists to store liked, not liked, and did not watch movies. The code up until now would look something like below -*	*Student codes.*

```
1  from flask import Flask, jsonify, request
2  import csv
3
4  all_movies = []
5
6  with open('movies.csv') as f:
7      reader = csv.reader(f)
8      data = list(reader)
9      all_movies = data[1:]
10
11 liked_movies = []
12 not_liked_movies = []
13 did_not_watch = []
14
15 app = Flask(__name__)
```

Here, we are just importing important modules from flask and also the csv module. We are then reading our movies.csv and saving it's contents into a list called **all_movies**. We are then finally creating 3 more lists - **liked_movies, not_liked_movies and did_not_watch**.

Also, do not forget the following lines of code at the end of this file so as you can run your

flask server
 	
`if __name__ == "__main__":`
`  app.run()`		

	Help the student write the first API, where they will be sending the data of a movie as a JSON response from the list of movies. *The code would look something like below -*	*Student codes.*

`@app.route("/get-movie")`
`def get_movie():`
`  return jsonify({`
`  "data": all_movies[0],`
`  "status": "success"`
`  })`
 	
Here, we are creating a route on "/get-movie" and this will be a GET request since we have not specified a method. We are then defining the function **get_movie()** and inside this function, we are returning the first movie from our **all_movies** list as a json using the **jsonify()** method.		

	Make the student write the second API, when the user has liked a movie. Here, we have to remove the movie's entry from our list of movies and then add this entry into the list of liked movies. *The code would look something like below -*	*Student codes.*

```
@app.route("/liked-movie", methods=["POST"])
def liked_movie():
    movie = all_movies[0]
    all_movies = all_movies[1:]
    liked_movies.append(movie)
    return jsonify({
        "status": "success"
    }), 201
```

Here, we are creating a **POST** method on route "/liked-movie". We are then defining the function **liked_movie()**. Now since we sent the first movie in the previous API, it is obvious for us that the user wants to mark that movie as liked. Therefore, we are selecting the first movie out of **all_movies** list, we are then removing this movie from the list of all the movies and adding this movie into **liked_movies**. We are finally returning a success response to the user.

	Make the student write the third API, when the user has not liked the movie. Here, we have to again remove the movie's entry from our list of movies and then add this entry into the list of movies that were not liked. *The code would look something like below -*	*Student codes.*

```
@app.route("/unliked-movie", methods=["POST"])
def unliked_movie():
    movie = all_movies[0]
    all_movies = all_movies[1:]
    not_liked_movies.append(movie)
    return jsonify({
        "status": "success"
    }), 201
```

Same as earlier, but this time the user is marking the movie as unliked so we are moving the movie to **not_liked** movies unlike earlier.

	Finally, make the 4th API where the user has not watched the movie. Again, we have to remove the movie from our list of movies and then add this entry into the list of movies that the user has not watched. *The code would look something like below -*	*Student codes.*

```
42  @app.route("/did-not-watch", methods=["POST"])
43  def did_not_watch():
44      movie = all_movies[0]
45      all_movies = all_movies[1:]
46      did_not_watch.append(movie)
47      return jsonify({
48          "status": "success"
49      }), 201
```

Same as earlier, but this time the user is marking the movie as did_not_watch so we are moving the movie to **did_not_watch** movies unlike earlier.

	<Optional> *Test your APIs using postman if there's time.*	
Teacher Guides Student to Stop Screen Share		
FEEDBACK • **Appreciate the student for their efforts** • **Identify 2 strengths and 1 area of progress for the student**		
Step 4: Wrap-Up (5 min)	Great! We built the APIs for our first screen.	

	Next class, we will be working on the APIs of the second screen and we will also start coding our mobile app!	
Teacher Clicks ✖ End Class		

Activity	Activity Name	Links
Teacher Activity 1	Google Colab	https://colab.research.google.com/gist/shubhamwhj/0d19a873633e8bd6c66bd05691be0e25/pro-c141-reference-code.ipynb
Teacher Activity 2	Solution Sample Flask Code	https://github.com/whitehatjr/c-141