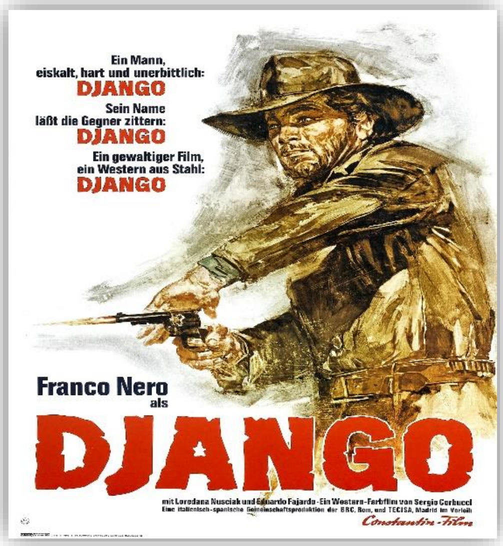
Unicode Django Tasks 2024

Task 1 - Django Unchained: Start of a Movie Database

Concepts: Setup + APIs

Install and setup Django. Create a project named “Moviezz”.

Create a view to show movie details:. Use the OMDb API to fetch and display details of a movie by its title. Ensure to include the API key in your requests. You can obtain an API key by signing up at OMDb API. Note: You should implement the above function using Django views and give the output using HttpResponse (or JsonResponse) or render using Django templates.


Ein Mann, eiskalt, hart und unerbittlich: DJANGO
Sein Name läßt die Gegner zittern: DJANGO
Ein gewaltiger Film, ein Western aus Stahl: DJANGO
Franco Nero als
DJANGO
mit Loredana Nusciak und Eduardo Fajardo · Ein Western-Farbfilm von Sergio Corbucci
Constantin-Film

To help you get started, use the following base API endpoint to fetch a list of movies for populating the home page:

https://omdbapi.com/?s={favorite movie}&apikey={API key}&page=1

This will return a response with about 10 items,so if you want can get further responses changing the page value which you can use to display movie details on the home page.

Note: Use requests module for Python 3 for API requests.

Bonus

You can also try displaying other features and/or movie names in all types present in the same format, error handling will be appreciated.

Reference links:

- Using the Requests Library in Python
- Django's Views - Python Django Tutorials
- Django's Templates - Python Django Tutorials
- API Key

Task 2 - The Search Awakens: Show Me Movies

Concepts: Dynamic URLS + Search

Next, you will now fetch all the Movie for a random single genre (To fetch movies of a random single genre, you will first need to fetch all movies and then filter the results by genre). You can supply parameters using dynamic URLS/ by making a form and query(i.e search) an API using the form values for the type.

Use HTML form, and or Django form for this task will fetch extra points for the same.

Bonus

Now, fetch a type of Movie from the user using forms. You can use the Django Forms for this purpose, but it will fetch you extra points if you can manage to handle a custom form built completely in the HTML template using the form tag and the Django {% url %} template tag.

Try to display the images of the Movies from the API, try finding an alternative route for the image and displaying it simultaneously will fetch additional scores for the same.

Reference links:

- [Django's Forms - Python Django Tutorials]()
- [URL dispatcher]()

Task 3 – Guardians of the Database: Save the Movie

Concepts: Save API output to Database

Design a model to store movies searched using the form from the previous task. Each movie should store details such as title, year, genre, and poster. Implement a system that saves a movie when it is searched for the first time and updates its search count each time it is searched again. Create a different page to display all the Movies searched, along with all their data (from the API + search)

YO DAWG, WE HEARD YOU LIKE APIS
SO WE PUT AN API IN YOUR API SO YOU CAN API WHILE YOU API
DIYLOL.COM

The model must have more than 3 different Movie fields, some of them being date released,rating,genre,plot,rating(any 1 from IMDB,Rotten Tomatoes / all if you're feeling confident) etc. Storing the Movie poster in the database will fetch additional points.

Reference links:

- [Django Models - Python Django Tutorials]()

Task 4 – Endgame: Movie Showdown!

Concepts: Gotta DB 'Em All!

Task 4 is free flow! Create a simple game using what you have done up till now. (Our focus is not on the frontend or UI but on the logic).

Here's our suggestion:

- Using the Movies you have searched, create a screen in which you can choose any 2 Movies and compare them out!
- The Movie with a higher searched value wins or else it's a draw.

Additional Suggestions:

- If you want to make the game more interesting, you can also consider the genre of the Movie (action, suspense, thriller etc.) while choosing the winner.

Submission Details -

1. **Keep the tasks ready during your interview:** Bring these tasks and the projects you might have done during the Unicode interviews and we will review them. Make sure that servers (if used) are set up and ready before you join the call.
2. **Upload your code to a GitHub Repository:** Make a **private** repository on Github and upload the code there before the deadline. Create a **README.MD** file and write a small overview of what you have made along with the screenshots. Refer to this link for how to make a good README.md: Make a README
3. **Make a Single Repository:** Your repository should contain the final code. There should not be different repositories for different tasks. There should be a single repository for them as each task builds on the previous task

---

Important Notes -

- Please use appropriate class/ function and variable names so that it will be easier for us to review. Also **put comments wherever necessary.**
- Now apart from this if you don't want to attempt these tasks you can also showcase the projects that you have built based on Django but the project must be at par with this assignment. (Must use views, urls, models and forms) Try requests module in a separate python file if you don't have any use of it in your project.
- We have provided you with the reference links within the tasks. If you have any further doubts about the implementation, you can refer to **StackOverflow** or just do a quick **Google** search for the same. If you are still confused about the problem statements or how to go about the tasks, refer to the contact list below.
- You have been given enough time so we expect you to complete at least the **first three tasks**. No problem if you can't complete them all but then you should understand that your chances of being selected decrease if you have completed a lesser number of tasks.
- You can also showcase your projects along with the **compulsory** tasks.
- You can also find useful resources related to **Django** on this repository: https://github.com/djunicode/resources.
- After you complete each task don't forget to push your code to GitHub using Git. Note that we should see what you'll have done throughout so don't push everything together (commit history). At a minimum, each task should be in its own commit i.e

we would like to see a **minimum of 4 commits** and more well-named commits are appreciated. Guide to Git: https://rogerdudler.github.io/git-guide/

For any technical doubts related to the assignment, try to search the problem (a quick google always helps). If nothing works feel free to contact us.
(A good way to reach us would be messaging on Whatsapp and for any error-related queries sending a screenshot would be helpful)

Monil Mehta - +91 9082228927
Mehek Mehta - +91 7021084483
Kaumudi Soman - +91 9594881297
Neha Tendulkar - +91 9833006939
Kshitij Poojary - +91 9820601371

ALL THE BEST!