

**Capstone Documentation (5/15/2025)**  
**Movie Recommendation Model**  
**Tom Pennell**

## **Setup**

1. Install python (3.11.7) and PostGreSQL. Python was downgraded for spacy compatibility, so a virtual environment is preferred.
2. Configure movie database in PostGreSQL and install pgvector extension  
Setup link (pgvector):  
<https://github.com/pgvector/pgvector?tab=readme-ov-file#installation>  
Info: <https://www.timescale.com/learn/postgresql-extensions-pgvector>
3. Train custom spacy model, following procedure on CapstoneTrainNER.ipynb. Make sure that config.cfg and base\_config.cfg (for the base spacy model) are located in the active directory. Also make sure that the training data is in the active directory (CombinedTrainUpdated.json).
4. For this step, switch to a traditional editor like vsCode. Install flask library and create a FlaskPage folder in the active directory (entire folder is included in code submission). The FlaskPage folder includes app.py, which hosts the flask page. It also includes [ModelScript.py](#) which is the backend file for the model computation, occurring in python. NamesList.csv is a file containing all of the names in the database for fuzzy matching. The templates subfolder in FlaskPage contains the 2 html pages for the website (home.html for the homepage and results.html for the results page). If using a virtual environment, ensure that [app.py](#) is running within it.

## **User Manual**

1. Run [app.py](#) in vsCode which will open the flask page locally
2. Enter prompt or click on a suggested prompt to see movie results

## Career Plans:

I intend to pivot towards data science. Now that I have these capstone projects under my belt, I am beginning to apply for jobs. These projects are hopefully valuable in helping me cross the bridge from analytics to science. I am not picky about the domain I will work in, but given the wide range this field covers, I can likely find a domain of personal interest, like financial investment.

## Resources:

<https://developer.themoviedb.org/reference/discover-movie>

Setup link (pgvector): <https://github.com/pgvector/pgvector?tab=readme-ov-file#installation>

Info: <https://www.timescale.com/learn/postgresql-extensions-pgvector>

Spacy Links:

<https://blog.futuresmart.ai/building-a-custom-ner-model-with-spacy-a-step-by-step-guide>

<https://spacy.io/usage/spacy-101>

<https://spacy.io/api/doc>

<https://stackoverflow.com/questions/71984818/what-does-i-in-token-i1-mean-when-using-a-token-returned-by-spacys-language>

PostgreSQL Python connection resource:

<https://stackoverflow.com/questions/26496388/how-to-connect-python-to-postgresql>

Dynamic Query Builder Resources:

<https://www.postgresql.org/docs/9.1/functions-array.html>

<https://www.geeksforgeeks.org/python-string-join-method/>

<https://gist.github.com/VankatPetr/5a2d6ea582b2072b6926dc3271e2c189>

<https://www.geeksforgeeks.org/python-psycopg-cursor-class/>

SBERT and Vector Querying Resources:

<https://adityagoel123.medium.com/vector-similarity-using-sbert-based-method-d66c77b44163>

[https://sbert.net/docs/sentence\\_transformer/usage/semantic\\_textual\\_similarity.html](https://sbert.net/docs/sentence_transformer/usage/semantic_textual_similarity.html)

Casting: <https://neon.tech/postgresql/postgresql-tutorial/postgresql-cast>

<https://pypi.org/project/pgvector/0.1.6/>

[https://sbert.net/docs/package\\_reference/util.html](https://sbert.net/docs/package_reference/util.html)

<https://www.vennify.ai/semantic-similarity-sentence-transformers/>

UI Resources:

<https://www.geeksforgeeks.org/flask-creating-first-simple-application/>

<https://www.digitalocean.com/community/tutorials/how-to-make-a-web-application-using-flask-in-python-3>  
<https://uiverse.io/>  
<https://jinja.palletsprojects.com/en/stable/templates/>  
[https://www.w3schools.com/css/css3\\_flexbox.asp](https://www.w3schools.com/css/css3_flexbox.asp)  
[https://developer.mozilla.org/en-US/docs/Learn\\_web\\_development/Core/Structuring\\_content/HTML\\_images](https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Structuring_content/HTML_images)

Name Matching Resource:

<https://pypi.org/project/RapidFuzz/>  
<https://rapidfuzz.github.io/RapidFuzz/Usage/process.html#rapidfuzz.process.extractOne>

Multiple Actors Display:

[https://python-web.teclado.com/section07/lectures/06\\_jinja2\\_conditional\\_statements/](https://python-web.teclado.com/section07/lectures/06_jinja2_conditional_statements/)  
<https://jinja.palletsprojects.com/en/stable/templates/>  
<https://stackoverflow.com/questions/14884523/can-a-div-have-multiple-classes-twitter-bootstrap>