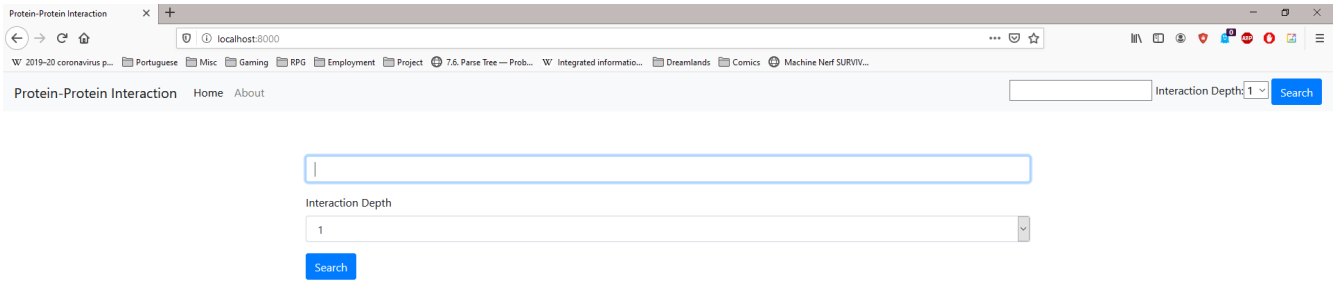
Week 13

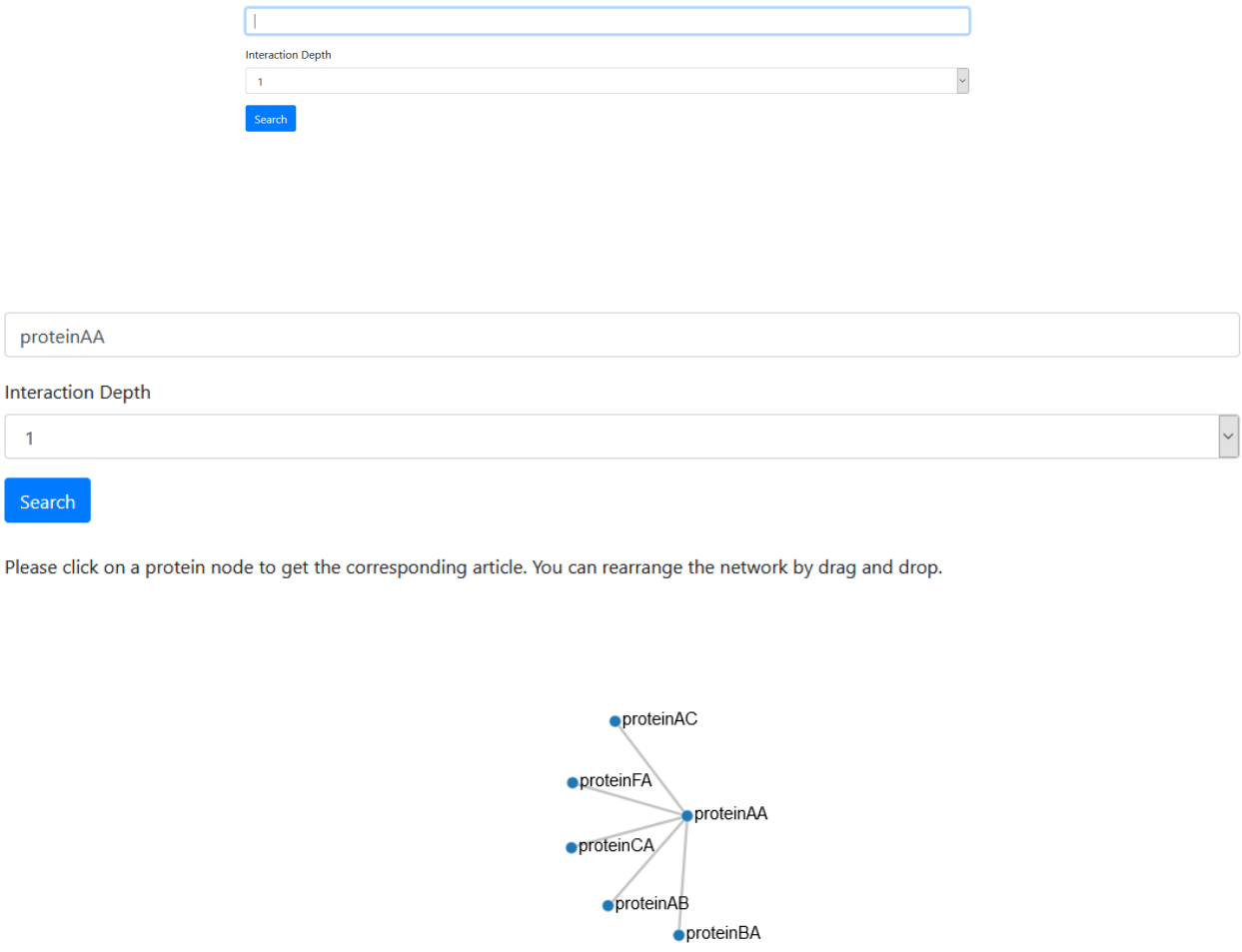
Work done

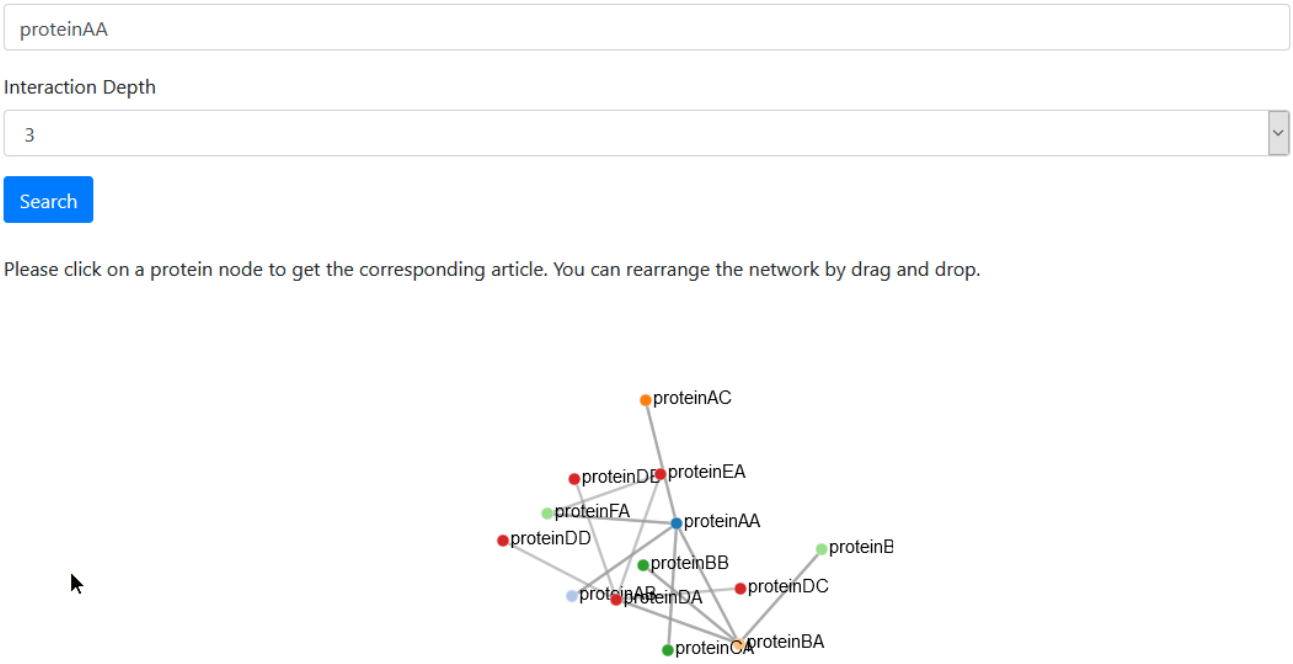
The week has been spent creating the Django webapp, with the protein network representation and database storage.

The researcher went with a simple but encompassing webapp based on a search engine paradigm. The user is encouraged to enter a protein and select a level of depth for the interaction. If this protein exists in our database, based on the level of depth, a protein interaction network is displayed, expanding from the original protein.

By clicking on a node in the network, a new browser tab is opened and the user is taken to the corresponding scientific article on pubmed.

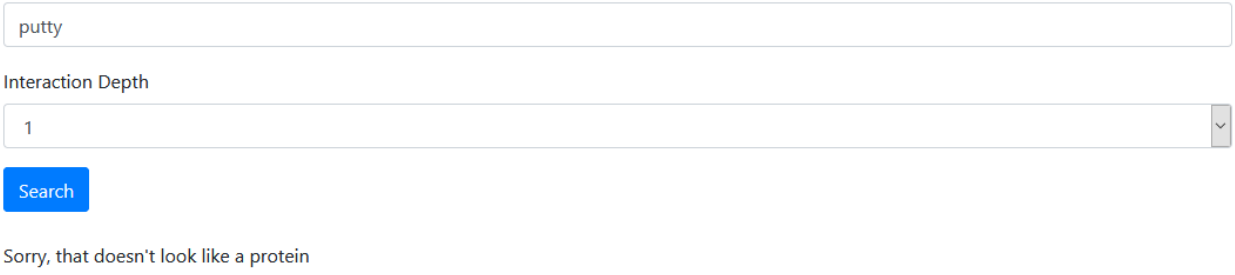






This is achieved by are recursive search of the sql database based on the first protein. The result is compiled as a python dictionary which is then converted to a javascript object. We pass the result to a d3js function for generating an interactive SVG image which displays the protein interaction network.

In case the user enters a protein that isn't in the database, the app checks if the entered word is indeed a protein, then adds it to a To Check list. This list is verified by a crawler thread every ten seconds. If the To Check list is not empty, the webcrawler compares the proteins in the To Check list against those already in the database. Ultimately this will be done by retrieving and checking pubmed articles. If an interaction is found, the protein is added to the database, along with a through table entry detailing the article specifying the interaction.



proteinGA

Interaction Depth

1

Search

We don't have that protein in our database, so we've added it to our checklist. Return later to see results.

proteinGA

Interaction Depth

1

Search

That protein is already in our list to check, try again later.

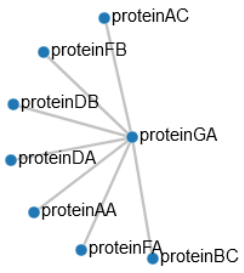
proteinGA

Interaction Depth

1

Search

Please click on a protein node to get the corresponding article. You can rearrange the network by drag and drop.



For beautification and responsiveness purposes, we have used the Bootstrap framework along with the Django Crispy package. The code for the webapp is available in github under

Code/02Website/003 PPI website

Problems

The Django app development was not difficult.

Hosting the final app however, with the dozens of GB of machine learning prerequisites such as keras and spacy, on a GPU capable server is likely beyond the finances of an unpaid undergrad.

To do for next week:

The next week will likely be spent coupling the machine learning elements such as spacy, keras, and the implemented machine learning networks.