**SOFT APP PROJECT OVERVIEW**

The aim of the project was to make a website which allow the user to select different operations to be done on two datasets about diseases and genes and show the results on screen.

The project is divided in 3 **main** parts ad some subparts.

* Main.py: Is the file which starts the whole website.
* **Website.py**: Contains all functions related to the website. It manages flask, retrieve the user inputs, ask “mediator.py” to compute the results, render the webpages and pass to them the results.
* **Mediator.py**: When asked by “website.py“ it asks “functions.py” to compute the results and convert them to a format readable by “website.py” and pass them to it.
* **Functions.py**: Contains all the functions needed to analyze the datasets.
* Settings.py: Contains project-wide variables and settings, for example the location of the datasets.

**Dependencies:**

* Flask:

to manage the website.

* Flask-paginate:

to render the pagination in some webpages.

* Flask-caching:

to make use of cache files.

* Pandas:

to execute operations on the datasets.

**Main.py**

It does not have any methods, at the start it calls “run()” from “website.py”.

**Website.py**

It’s the front-end of the website and it relies on Flask to render the webpages.

Global variables:

* “app”:

Is the instantiation of Flask

* “cache”:

Is and instantiation of Cache

Functions:

* Homepage():

Returns the homepage of the website

* diseaseTable():

Allows to go through the disease\_evidences.tsv file

* geseTable():

Allows to go through the gene\_evidences.tsv file

* diseaseDf():

Allows to download the disease\_evidences.tsv file

* geneDf():

Allows to download the gene\_evidences.tsv file

* documentation():

Returns an interactive documentation of the project

* about():

Returns a webpage with the names of the components of the group

* functions():

Returns a webpage with all the operations doable with the data tables

Operations:

* + - info():

Returns a webpage with informations of the datatables and their preview (heads and tails)

* + - distinctGenes():

Returns a webpage with the unique genes in gene\_evidences.tsv

* + - geneEvidences ():

Returns a webpage where the user can input the gene\_symbol or the geneName and then it returns all the evidences of the gene in the sentences

* + - distinctDiseases ():

Returns a webpage with the unique genes in disease\_evidences.tsv

* + - diseaseEvidences ():

Returns a webpage where the user can input the diseaseID or the diseaseName and then it returns all the evidences of the disease in the sentences

* + - correlation10 ():

Returns a webpage with the 10 top most frequent correlations between genes and diseases

* + - correlationAll ():

Returns a webpage with all the correlations between genes and diseases

* + - diseasesRelatedToGene ():

Returns a webpage where the user can input the gene\_symbol or the geneName and then it returns all the diseases related to the gene in the sentences of both data tables

* + - genesRelatedToDisease ():

Returns a webpage where the user can input the diseaseID or the diseaseName and then it returns all the genes related to the disease in the sentences of both data tables

Html documentation:

Instead of repeating the same code for the navigation bar in every page, we created a base file from which all html templates inherit from. Then each page adds its own title, style, head and body through the use of the block statement {% block title %}{% endblock} used by jinja2 engine in each page.

To have a nice looking webpage we used bootstrap to get the navigation bar and many buttons and forms.