**Table of Contents**

Software Architecture – Pedro

**A screenshot of a cell phone

Description automatically generatedFigure 1: Macroscopic software architecture used components used. The black box in the left refers to the front end of the website showing that the base page of the website.**

The main components of our website are shown in Figure 1. Tables were created with all necessary information on the kinases, phosphosites and inhibitors. All this information was integrated into a database file called “kinase.db” using SQLite code. SQLite connects the database with main application Flask. With SQLite queries we can easily and quickly access the necessary data from the “kinase.db” file. Flask incorporates the front-end of the website using Jinja as template engine and Python 3.7. The main Flask files used are: (1) application.py, which is the executable file; (2) DataAccess.py, responsible for the integration of the database; (3) Routes.py, responsible for the different website pages; (4) analysis.py, which is responsible for analysing and creating the results of the file uploaded by the user. All website routes are defined by using HTML language, static files written in CSS were also created to maintain a constant design and also used Javascript language for some visualization aspects. All these files are interlinked with the main application Flask. The interconnection of all these separate parts return the final, fully functional website. We then deployed our website application using Amazon Web Services Elastic Beanstalk making the website available for any user.

Database Schema –

A picture containing screenshot

Description automatically generated

**Figure 2: Kinview Database schema. The figure shows the different tables included in the database and the lines represent the links between the tables. Also, all tables where exists a column with accession\_number can be linked with any other table that has accession\_number in one of its table.**

The database schema was designed having in mind what we needed to show on the website. The tables shown in figure 2 were created individually and trough SQLite the database was created. The links between tables are the Accession\_number. In table Kinase\_targets, K\_accession\_number means the kinase accession number and the T\_accession\_number is the target of the kinase, also as accession number. Having the accession number as link in all tables it allows us to quickly get the information from any table if we have the accession number associated with the protein we are interested. The Inhibitor\_names table shows all the names of a specific inhibitor and its assigned Cnumber. This Cnumber is then used to link this table with the Kinase\_inhibitors table. This table is then linked with the rest of the database by having the accession number of the targets and other targets of each inhibitor. For example, if the user searches for protein X, we search X on the Protein table (Protein\_name, Gene\_name, Gene\_alias or Protein\_alias columns) to get the accession number assigned to that protein. From there we can get the kinases that affect protein X and the proteins that X might target by assessing table Kinase targets. If we would like to know which inhibitors target protein X, we just need to look for the Protein X accession number in the columns Target\_accession\_number and Other\_target\_accession\_number from the table Kinase inhibitors and extract the column Inhibitor when the accession number was found in the target columns. The schema might look that it could have less/joined table but the reason for this is that the table Protein contains all existing human proteins, while other tables, like for example Kinase characteristics only has information for proteins that are actually kinases. Other reason is that, some tables have the accession number as a primary key (each accession number only appear once) and other tables like for example table Domains have several entries for the same accession number. We could also separate the kinase targets into two tables, one with kinases and other with targets/substrates. However, we found that having the tables this way reduces the number and the complexity of the queries necessary to extract both phosphosites and kinases of a determined protein. The retrieving of the data from the database is done by using simple SQLite queries, such as for example: "SELECT kinase\_characteristics.accession\_number FROM kinase\_characteristics WHERE kinase\_characteristics.accession\_number = ?"

Kinase Analysis Tool

The website provides a tool called Kinase activity analysis that allow the user to upload phosphoproteomic data and get as a result a summary of the uploaded data, a volcano plot and the relative human kinase activity. The user must upload a .tsv file with only one inhibitor and the columns should be presented in a specific order: Substrate, Control mean, inhibitor mean, fold change, p-value, control CV, inhibitor CV. The substrate column accepts both protein names and gene names.

After the file is uploaded the data is processed. The data is processed in a series of steps. First, we provide a bar graph summarizing the data present in the file. In the bar plot we show the total number of substrates and the number of substrates where the columns are empty. Furthermore, out of the substrates that were not empty we show how many of them have significant fold changes. From the significant ones, we show the number of substrates that showed increased or decreased phosphorylation.

On the second step we completely clean the data. We remove all columns that are completely empty and remove any row that has at least on value empty. After this a volcano plot is produced. This volcano plot shows all substrates with two different colours. The yellow dots are the substrates that show a low fold change (less than 100 and bigger than 0.1) and the blue dots that are the one with the higher level of fold change. In the volcano plot it is also possible to see a red horizontal dashed line that sets the p-value on 0.05. The x and y axis of the plot show the log10 of both the fold change (x-axis) and the p-value (y-axis).

The third step is to calculate the relative activity of the kinases. For this we used the KSEA algorithm described by Casado et al., 2013. With this algorithm we calculate a kinase z-score as shown in figure 3 below.

A picture containing object

Description automatically generated

Figure 3: KSEA algorithm.  s̄- denotes the mean log2(FC) of known phosphosite substrates of the given kinase, p̄ - represents the mean log2(FC) of all phosphosites in the dataset*, m* denotes the total number of phosphosite substrates identified from the experiment that annotate to the specified kinase, and *δ* denotes the standard deviation of the log2(FC) across all phosphosites in the dataset.

To find the kinases that were affecting the substrate we used the curated Kinase–Substrate annotations from PhosphoSitePlus (Hornbeck P.V et al., 2012), which was also used to retrieve data for one of the tables of our database. The z-score obtained after all the calculations is based exclusively on the phosphorylation status of the substrates. If the z-score is negative means the kinase is being inhibited, if positive, then the kinase is upregulated. Then the p-value is obtained assuming that the z-score follows a normal distribution and that the standard deviation is 1 and the mean 0. The results a bar graph that shows all kinase z-scores and a table below containing all kinases, their z-score and p-value. This bar plot is interactive and allow the user to zoom in and see the exact kinase and z-score associated with it. The number of substrates that didn´t have any kinase matched are also given. The tool also provides the kinases the have a significant p-value (<0.05) of the z-score by showing another interactive bar graph.

, denotes the mean log2(FC) of known phosphosite substrates of the given kinase, p−p- represents the mean log2(FC) of all phosphosites in the dataset*, m* denotes the total number of phosphosite substrates identified from the experiment that annotate to the specified kinase, and *δ* denotes the standard deviation of the log2(FC) across all phosphosites in the dataset.

Data schema limitations

Our database schema runs fast and correctly. However, if we were to increase drastically the amount of data, we have in the database that would probably make our website much slower. In that case the schema should be changed and look up tables should be added to decrease searching time.

Kinase activity limitations

Our kinase activity tool only accepts one inhibitor at a time, which is not ideal. If the user has data with several inhibitors in a “.tsv” table, then will have to separate these. Furthermore, in the actual analysis we use annotations of phosphositePlus for find the kinases. These annotations are in a table that does not belong to the database. Our kinase target table in the database has made having as source the phosphositePlus annotations and the analysis could have used the database table instead of a separate table. Due to late awareness of this, that could not be done. Using our own database would slightly increase the process time.

References

Casado Pet al.  (2013) Kinase–substrate enrichment analysis provides insights into the heterogeneity of signaling pathway activation in leukemia cells. *Sci. Signal*., 6, rs6–rs6.

Hornbeck P.V et al.  (2012) PhosphoSitePlus: a comprehensive resource for investigating the structure and function of experimentally determined post-translational modifications in man and mouse. *Nucleic Acids Res*., 40, D261–D270.