2018

FetBASE

Documentation

NAYAM CHOWDHURY, DIONYSIOS GRIGORIADIS, MICHAELA MATTHEWS,
DANIEL GRANT

Contents

$Fettucine Retrotransposon database project

Features:

- Family Table
- Distribution of retrotransposons
- Translated products
- Amino Acid relationship viewer
- Peptide Sequence identifier
- Mzid/mzTab peptide analyser
- Expression Atlas

Packages / Systems used:

- Flask
- Python
- MySQL
- JQuery / Javascript

FLASK

Flask is a micro web framework written in python which is also based on the Werkzeug toolkit and uses the Jinja2 template engine which allows for the use of python-like expressions in HTML document.

FLASK was chosen as the main web development toolkit as it was quick to learn and understand under the time constraints that were given.

Flask was also used as it, by default, protects against the use of cross site scripting (XSS) so malicious scripts that are injected into the website does not compromise the website security.

This micro web framework is also compatible with python 3 so that if the website needs to migrate from python 2.7, it can occur generally without too much hassle.

MySQL

MySQL is an open-source database which uses structured query language to be able to access data. The system is usually known to be fast, reliable and stable and works on all major operating systems.

We used MySQL as the database as it is fast and efficient to use and has the ability to query on the fly. It was also very simple to use in conjunction with flask and allowed for multiple different data tables to be created and read easily.

JQuery

JQuery is a JavaScript library designed to simplify the client-side scripting of HTML It was used to be able to add JavaScript functionality into the browser. The features that are available with the use of

JQuery are endless as it allows you to manipulate and visualise the webpage in many different ways such as adding animations and processing events after certain specifications are met.

HTML

All websites require HTML to be able to run. Combined with CSS and Javascript, the possibilities that can occur are limitless. CSS allows for custom styling to be created to provide a more visually appeasing website.

Python Modules and Libraries

The following libraries and modules were used to be able to create the website:

- Flask (flask, send_from_directory, render_template, request, url_for, flash)
- OS (operating system module)
- MySQLdb
- Pandas
- Numpy
- Bio (SeqIO)
- Re (regex)

The Flask library was used as it was the fundamental micro web framework required to run the whole system and connect the individual processes together.

Starting of Program

The application has been setup that it tries to connect to the database first using the try statement (Used for Errors and Exceptions) and failure to connect to a database results in the program still running, but displaying an error message in the console stating that the database has not connected. This would mean that most of the website would not function as intended.

Family Table

Distribution table

Peptide sequence list

Relationship Viewer

Muscle was used to create an alignment followed by emboss consensus to create a consensus sequence. Muscle was then used again to align all the sequences and a tree file was created using megacc. (megaproto used to created configuration file).

Uploading of Files

Allowing users to upload files of their own allows them to be able to search our database and identify family members that the peptide sequence has been translated from.

Peptide Sequence Identifier (Fasta format)

This webpage allows for user input via 2 different methods. The first method allows a user to upload a fasta sequence file to the server in order for it to analyse the file. The second method allows a user to paste a fasta sequence into a text box before pressing search.

The server first waits for a POST request method to engage before doing any further actions. If a GET request method is used, the normal webpage is served to the client with some data (up to 1000 rows) from the database being displayed. This allows the user to be able to navigate part of the database without any input.

Fasta Checker

Due to the possible memory constraints on the host machine, the option was taken to disallow entering in more than one fasta sequence and fasta sequences with headers via the text box. Instead the user is able to upload the file as a fasta file (including headers) where it can be parsed more efficiently using SeqIO module from biopython.

The text field allows the user to enter any length of peptide sequence with spaces and tabs in between and is able to concatenate and parse the data together (for ease of copy / paste use for the client).

If a file is uploaded that contains no data, an error message is relayed back to the user stating the file is empty.

Incorrect Filetypes

The program is able to differentiate between the correct and incorrect filetypes. If an incorrect filetype is uploaded, an error will be produced stating the user to upload a correct file with the correct extension.

Parsing correct files

If the correct requirements have been met, the peptide data is then queried using MYSQL and if a match has been found in the database, it returns the family name in a table.

Mzident / MzTab

This webpage allows a user to upload a file that can be either an mzident or mztab formatted file. A user can also select the tissue type and disease progression type before submitting the file if the information is known about where the data has been sourced from.

This part of the software is able to detect the different filetypes that are uploaded and will only accept files with the correct extension.

Parsing of files

The type of parsing that occurs is dependent on the type of file extension that is uploaded to the server. The server is able to locate the extension and then use regular expressions to extract the information needed. The type of regex used differs between the type of files. The extracted peptide sequence is then queried in the database via MYSQL and the resulting matches are displayed in a table.

The additional information is also saved and is saved to be used for the expression atlas. The server conducts a hash check of the file to see if it has been uploaded before, if it has not been uploaded previously, the additional information is saved which contributes to the count tracker that is available on the expression atlas.