**Filtering Reads Made Easy**

**<NAMES>**

1 [The Shmunis School of Biomedicine and Cancer Research](https://en-lifesci.tau.ac.il/lp-en-mcbb), George S. Wise Faculty of Life Sciences, Tel Aviv University, Tel Aviv 69978, Israel.

† To whom correspondence should be addressed:

Tal Pupko, Tel: +972 3 640 7693; Fax: +972 3 642 2046; E-mail: talp@tauex.tau.ac.il

Keywords: sequencing, webserver, DNA contamination, interactive webpage

# Abstract

Recent advances in sequencing and informatics technologies have led to a deluge of sequencings. This has dramatically changed our understanding of the genome and the understating of biology. While it is now relatively easy to sequence, extracting only meaningful reads is still a challenge. Here we introduce a new webserver to facilitate the filtering process. Three of the main features implemented in <Name> are: (1) automate updating of the latest bacteria database; (2) fast comparing the input reads to the database; (3) a simple interactive web page to further analyze the results.

# Introduction

Over the past decades, the price of sequencing has been decreasing rapidly (Check Hayden 2014). This allowed research groups all over the world to sequence different species and doubled the size of the GenBank every few months (Benson et al. 2018). The data provided by those sequencing is a vital resource in biology and clarified many unknowns.

With the increase of DNA sequencing, confounding contaminant DNA takes place as one of the main challenges. Instead of finding the genome of the required species, some of the reads are contaminated and originate in different species (Steinegger and Salzberg 2020). This contamination can have a significant effect on the later analysis including false positive SNP’s identification (Goig et al. 2020), incorrect labels on sequences in metagenomic studies (Kirstahler et al. 2018).

The filtering process requires the use of a search engine and downloading required genomes databases. Some research laboratories began implementing their in-house analysis pipelines, and later, different search engines emerge (Wood, Lu, and Langmead 2019). These applications require specific working environments (i.e., operating systems), computation power (multicore machines), more than basic technological skills (e.g., installation and running), and many parameters to configure. Other laboratories do not filter the reads and hence find false discoveries while wasting money and time.

Here we present <Name> web server. The web server was developed to easily filter genomics reads and make it more accessible for the scientific community. This is done by updating the latest database and searching each of the reads in it. No installation and no other prerequisites are needed. A simple and interactive graphical user interface (i.e., GUI) personalizes the user output. Visual and textual results that are ready for publication or further analysis.

# Materials and Methods

## Input

The <Name> web server requires a fasta or fastqc file containing the reads to be filtered. In the analysis stage, we ask the user for the species list he would like and for a k-mer percentage threshold to classify as contamination. Upon completion of the submission, a link to the results is sent to the user if they choose to provide their email address. The results remain available on the webserver for at least 3 months.

## Database

Each month, we download the full bacteria database from <NCBI?>. This is an automated process to verify our bacteria’s genomes are up to date. When the download is completed, the data is preprocessed based on the search engine.

## Search Engine

To conduct a fast search, we installed the Kraken 2 search engine (Wood, Lu, and Langmead 2019). With a given read file, each read is split into k-mers and then searched in the bacteria’s database. <DEFAULT PARAMETERS ARE USED?>. The output of the Kraken 2 search engine is a csv file containing all the reads with a list of species with the number of k-mers found in this read.

## Preprocess Kraken Output

For each read, we sum the amount of k-mers found for each of the species. Then, we save the species with the max k-mers, and the percentage of similarity to the species. This is done, by dividing the k-mers of the max species by the length of the read. To display the data in graphs fast, we transform the list of the reads into a matrix. The rows of the matrix contain the k-mers percentage (this value represents the similarity of the read to one of the species) and the columns of the matrix are the species names. The cells are the number of reads that fits the specific row and column <INSERT FIGURE>.

## User Interactive Analysis

First, the matrix described above is loaded. The sum of each row represents the number of reads with this similarity <INSERT FIGURE>. While the sum of each column represents the number of reads associated with this species. One could find its optimal threshold of k-mer percentage by controlling the slider. In addition, one could control the species list that will be classified as contamination DNA. Those two parameters will immediately change the graphs thus, one could see the effect of the contamination. When the user is ready to export the results, the threshold for the k-mer percentage and the species list will be delivered to classify each of the reads. When the classification process is finished, the user will get a file containing all the unclassified reads.

### Implementation

<NAME> is implemented in Python using Flask framework. The source code is available at: [<GITHUB](https://github.com/orenavram/microbializer) LOCATION>. The web server jobs are processed on ProLiant XL170r Gen9 servers, equipped with 128 GB RAM and 28 CPU cores per node. The Gallery, Overview, and Frequently Asked Questions (FAQ) sections of the web server should help users get the most out of the web server. A running example (different from the case studies analyzed in the Gallery) is also provided.

# Test Case

<ADD ONE TEST CASE>

# Technical Description

## Overview

Implemented in flask, write about the Apache, html, JavaScript at the client? Front / back webserver architecture?

## Front

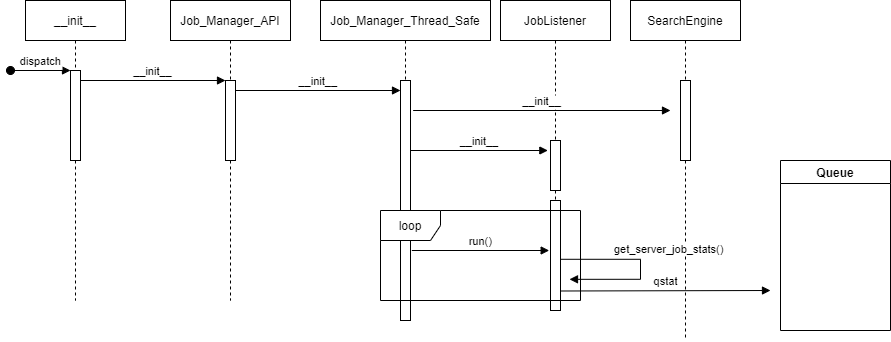
CSS, html, javascript, where can we find each of the source code. Introduce the \_\_init\_\_ object with the endpoints.

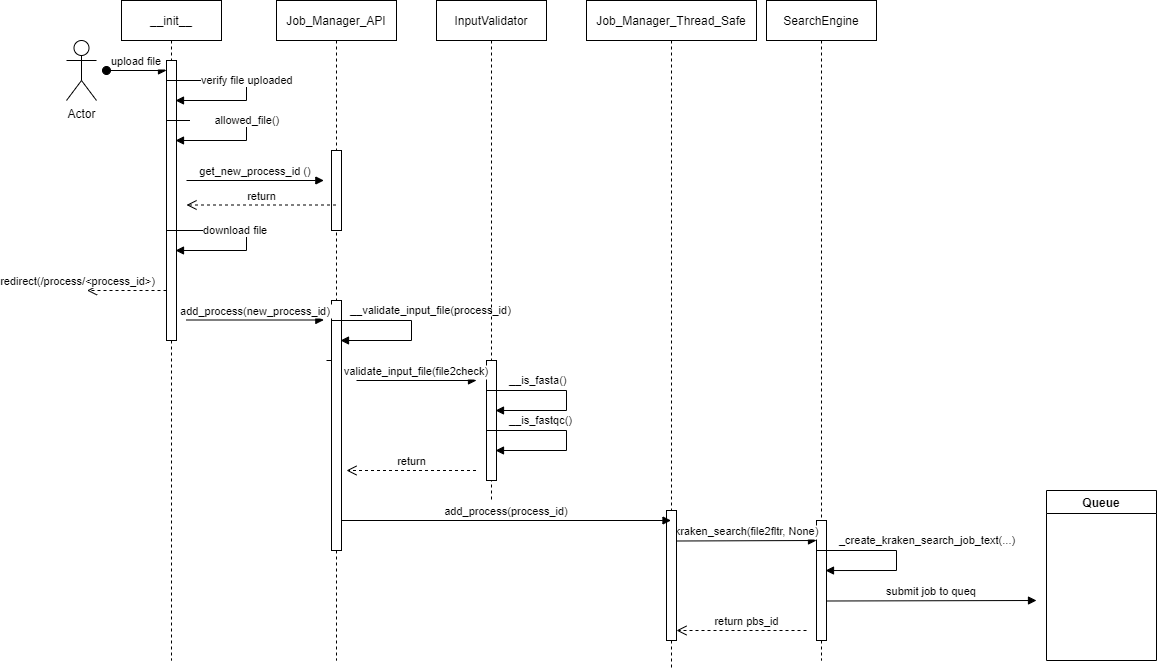
## Back

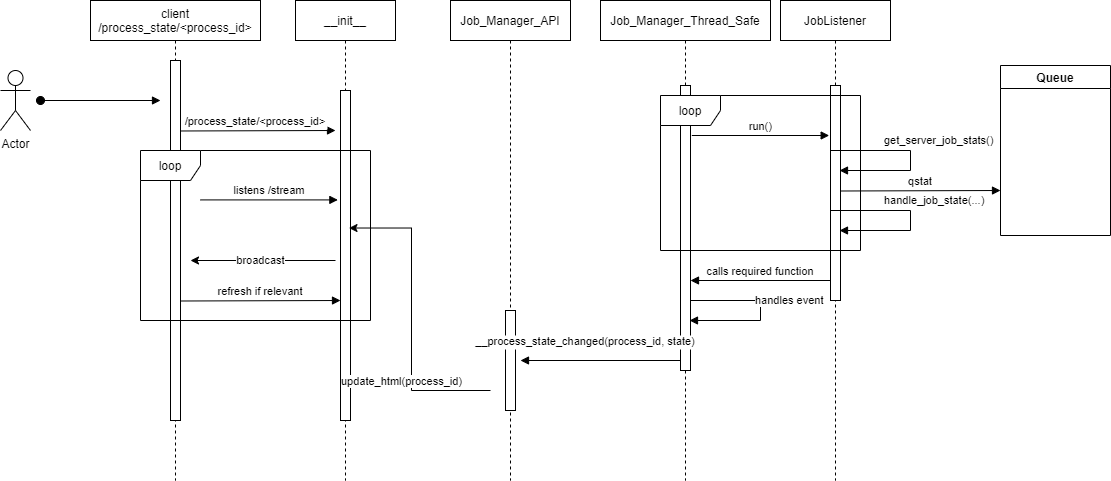
1. UML diagram with all objects

2. A short paragraph on important object: \_\_init\_\_ job\_manager\_API, job\_manager\_thread\_safe, KrakenSearch, PBSListener, OutputProcessor…

3. sequence diagrams for important processes: upload file -> creating a process, process change state -> update UI, kraken results -> user interactive webpage, init:







## Important Parameters to Configure

Probably in shared const.

## Creating another webserver

Stages for creating another webserver.

**References**

Benson, Dennis A, Mark Cavanaugh, Karen Clark, Ilene Karsch-Mizrachi, James Ostell, Kim D Pruitt, and Eric W Sayers. 2018. “GenBank.” *Nucleic Acids Research* 46 (D1): D41–47. https://doi.org/10.1093/nar/gkx1094.

Check Hayden, Erika. 2014. “Technology: The $1,000 Genome.” *Nature* 507 (7492): 294–95. https://doi.org/10.1038/507294a.

Goig, Galo A., Silvia Blanco, Alberto L. Garcia-Basteiro, and Iñaki Comas. 2020. “Contaminant DNA in Bacterial Sequencing Experiments Is a Major Source of False Genetic Variability.” *BMC Biology* 18 (1): 24. https://doi.org/10.1186/s12915-020-0748-z.

Kirstahler, Philipp, Søren Solborg Bjerrum, Alice Friis-Møller, Morten la Cour, Frank M. Aarestrup, Henrik Westh, and Sünje Johanna Pamp. 2018. “Genomics-Based Identification of Microorganisms in Human Ocular Body Fluid.” *Scientific Reports* 8 (1): 4126. https://doi.org/10.1038/s41598-018-22416-4.

Steinegger, Martin, and Steven L. Salzberg. 2020. “Terminating Contamination: Large-Scale Search Identifies More than 2,000,000 Contaminated Entries in GenBank.” *Genome Biology* 21 (1): 115. https://doi.org/10.1186/s13059-020-02023-1.

Wood, Derrick E., Jennifer Lu, and Ben Langmead. 2019. “Improved Metagenomic Analysis with Kraken 2.” *Genome Biology* 20 (1): 257. https://doi.org/10.1186/s13059-019-1891-0.