Mauritius Research Council

INNOVATION FOR TECHNOLOGY

COMPUTATIONAL ANALYSIS FOR UNDERSTANDING EVOLUTION OF INFECTIOUS DISEASES

Final Report

February 2019

Mauritius Research Council

Address:

Level 6, Ebene Heights
34, Cybercity
Ebene

Telephone: (230) 465 1235
Fax: (230) 465 1239
e-mail: mrc@intnet.mu
Website: www.mrc.org.mu

This report is based on work supported by the Mauritius Research Council under award number MRC/RUN-1626. Any opinions, findings, recommendations and conclusions expressed herein are the author's and do not necessarily reflect those of the Council.

PI Name and Address

Dr. Shakuntala Baichoo
Department of Digital Technologies
FoICDT
University of Mauritius

MAURITIUS RESEARCH COUNCIL FINAL REPORT

PART I-PROJECT IDENTIFICATION INFORMATION
1. **Type the name of the MRC Scheme under which grant is made:** Unsolicited Research and Innovation Grant Scheme (URIGS)
2. **Award Dates** (MM/YYYY) **From**: 08/2016 **To**: November 11/2018
3. **Organisation and Address**: University of Mauritius, Reduit
4. **Award Number:** MRC/RUN/1626
5. **Project Title:** Computational Analysis for Understanding Evolution of Infectious Diseases

18 February 2019

Mauritius Research Council

Computational Analysis for Understanding Evolution of Infectious Diseases

TECHNICAL REPORT

Final Report

MRC Reference: MRC/RUN/1626 (URIGS)

Investigators

PI: Dr. Shakuntala Baichoo

Co-Investigators:

Prof. Yasmina Jaufeerally-Fakim

Mrs. Zahra Mungloo-Dilmohamud

Abstract

Infectious diseases are one of the main causes of death around the world each and every year. Microorganisms present in the air, soil and water affect human, plants and animals in Africa and this impedes the development of the whole continent. Several diseases have emerged due to the cohabitation of animals and human beings. Sometimes, the pathogen evolve, evade the host defenses by varying their antigenic molecules and this renders the host vulnerable.

The aim of this project is to develop an Infectious Disease Evolutionary Analysis System (IDEAS) which provides appropriate **tools** for the researchers in the area of microbial genomics, to perform an intensive computational and evolutionary analysis of existing and newly-sequenced infectious-causing pathogens. A web-based front-end application is developed for interacting with the data warehouse. Several facilities will be provided for interfacing with the sequences. Some of them include: similarity searches using BLAST, calculate the evolutionary rate of genes, text-mining component to search for specific genes, identification of polymorphic genes and highly-conserved genes, geographical visualization of the evolution of infectious diseases on the African continent.

This project is a comprehensive and in-depth study in the area of infectious diseases that will have a positive impact on all the researchers in the African continent that are actively involved in bioinformatics.

Table of Contents

Chapter 1 – Introduction

The term disease refers to conditions that impair normal tissue function. An infectious disease can be defined as an illness caused by another living agent, or its products, that can be spread from one organism to another. Infectious diseases (also known as communicable diseases) are one of the primary causes of death worldwide each year and are caused by microorganisms found in the air, soil and water (https://www.ncbi.nlm.nih.gov/books/NBK20370/). Emerging and reemerging diseases, and drug resistant pathogens have further contributed to the seriousness of the problem. In an emergency situation, they can raise the death rate up to 60 times in comparison to other causes (Ameli, 2015). Pathogenic microorganisms, such as bacteria, viruses, parasites or fungi, can cause infectious diseases. Such diseases can be spread, directly or indirectly, from one person to another. Infectious diseases, whether they affect humans, animals or plants, continue to be a fundamental hurdle to both economic development and human health in Africa. Until this challenge is met, the development of the continent will continue to be severely retarded. Due to the cohabitation of animals and human beings, there have been several incidences of infectious diseases being transmitted from animals (usually vertebrates) to humans and these are known as zoonoses. In fact, many emerging diseases such as the Ebola virus disease and influenza are zoonoses which can be caused by any of the germs mentioned above. Zoonoses can be transmitted directly from animals to humans or through media such as air (influenza) or through bites and saliva. Zoonoses transmission can also occur via an intermediate species (known as a vector), which carries the disease pathogen without showing symptoms.

Due to the rapid increases in the generation of genomic and clinical data related to infectious diseases, a new field namely the "infectious diseases informatics" has emerged around 10 years back (Zeng, et al. 2005). This has lead to a combination of experimental and informatics evidence that has fuelled the expectations of better controlling the outbreak of infectious diseases. The objectives of infectious disease informatics are spanning from the development of antimicrobials and more effective vaccines, through the identification of biomarkers for transmissibility to a better understanding of host-pathogen interactions. With the emergence of new informatics methods and integrated databases the objectives are gradually being realized.

The main aim of this project is to set up an Infectious Diseases Evolutionary Analysis System (IDEAS) that will allow researchers from the local community and the African continent to perform an intensive computational and evolutionary analysis of existing or newly-sequenced infection-causing pathogens, more specifically bacteria and viruses.

1.1 Objectives of the proposed project

The proposed project aims at developing an application named ***Infectious Diseases Evolutionary Analysis System*** (IDEAS) for performing analyses on whole genomes and gene sequences of infectious diseases. More specifically, IDEAS will:

(a) Prepopulate a data warehouse with gene and whole genome sequences of known bacteria and viruses causing infectious diseases, along with the host-specificity of each microorganism

(b) Include vector genome sequences and gene expression data

(c) Provide a facility to perform literature search through text-mining for specific organisms or genes or classes of genes

(d) Develop facilities to update relevant data automatically into the data warehouse, from public databases

(e) Provide analytical tools at the front end for comparison of sequence data and visualization of the results. The tools to be implemented will:

- Allow users to browse through already-loaded sequences
- Upload new sequences and associated data pertaining to bacteria or virus causing an infectious disease
- Search for known features from the set of genomes found in the data warehouse
- Integrate a local BLAST component to perform similarity searches
- Examine sequences for the presence of known protein motifs
- Identify highly-conserved genes which will allow design of primers for diagnostic purposes
- Determine the evolutionary dynamics of polymorphic genes whose products can be targeted for drugs and vaccine development
- Calculate the evolutionary rate of all genes for a given specie
- Determining horizontal gene transfer of specific strains
- Generate tailor-made reports for specific analyses

(f) Investigate into the possibility of integrating Geographical Information System (GIS) with the resulting software so as to map host pathogen interaction during infectious diseases and antibiotic resistance to geographical locations. This can help to proactively anticipate trends in disease and resistance as well as prioritize emerging infectious threats to human health, especially on the African continent.

1.2 Issues to be addressed by IDEAS

In the past two decades there has been an explosion in the amount of biological sequence data becoming available, due to the very rapid progress of genome sequencing projects. Clearly, we have reached a point where computers are essential for the storage, retrieval, and analysis of biological sequence data. However, we need to take a disciplined approach in analyzing the data found at different sources and in different formats. There is also a need to localize related data from different sources at one place so that researchers do not waste time to move data from several places before proceeding to the actual analysis.

The proposed solution is expected to provide a one-stop-shop to african/local researchers working in the area of infectious diseases. Users can simply login to

the new service and perform a number of analyses relating to infectious diseases and get the results in publication-ready formats. The service will also structure the data into an appropriate format which will be easy to manipulate. The results from the system can be used for further analysis using other existing tools if necessary.

1.3 Scope of work

The system will be limited to the analysis of bacteria and viruses causing infectious diseases, and their vectors of transmission that are of direct relevance to Africa.

1.4 Methodology

The following steps will be used to achieve the objectives mentioned in section 1.1:

(1) A data warehouse will be created with existing genome and gene sequences for bacteria and viruses from public databases, namely NCBI, EMBL and DDBJ. Information from ABSA (American Biological Safety Association) will also be loaded with respect to host specificity for specific infections.

(2) In parallel, relevant publications associated with the genomes populated in the data warehouse will also be loaded in the data warehouse.

(3) Appropriate text-mining tool will be integrated in the system.

(4) A facility for updating data relevant to the loaded sequences on a regular basis will be developed.

(5) A web-based front-end application for interfacing with the data warehouse, with the following facilities will be developed:

- Browsing through genome and gene sequences
- Uploading new sequences and associated data pertaining to the bacteria or viruses
- Facility to search for known features from the set of genomes found in the data warehouse
- Performing local BLAST for determining the similarity between given sequences
- Examining sequences for the presence of known DNA/protein motifs
- Facility to determine a list of highly-conserved genes which may be used for designing primers for diagnostic purposes
- Identification of polymorphic genes whose products can be targeted for drugs and vaccine development
- Determining the evolutionary rate of all genes for a given specie
- Determining horizontal gene transfer of chosen genomes using a locally-developed tool
- Generation of tailor-made reports for specific analyses

Chapter 2 – Genomic Data Warehouses

Infectious diseases are complex systems, involving multiple organisms (e.g., pathogens and hosts) interacting across different environments and time scales. Much of the data that we have related to infectious disease is multi-dimensional, incomplete, and likely to be biased in ways we do not fully understand. In addition, these data are often not integrated nor interoperable making it difficult for researchers from different disciplines to communicate.

Bioinformatics research in Infectious disease depends on the advances in microbial genomics, the sequencing and comparative study of the genomes of pathogens, and proteomics or the identification and characterization of their protein related properties and reconstruction of metabolic and regulatory pathways (Bansal, 2005). The speed of genome sequencing, especially for microbial species, has been steadily accelerating since the introduction of modern DNA sequencing methods more than forty years ago (Sanger, et al. 1977). Microbial genomes are thousands or millions of base pairs in length, requiring both a global view of the genome and the ability to zoom in on details for the purpose of analysis and annotation. Annotation is the extraction of biological knowledge from raw nucleotide sequences (Médigue and Moszer 2007). Such decoding of the genomes allows the prediction of protein-coding genes and therefore, the proteins the organism is able to produce.

The first step in analyzing the evolution of infectious diseases is to create a data warehouse of infectious diseases causing organisms by localizing data from different sources.

The second step in proceeding with the study of evolution of genes in related species is to compare the genomes species of interest.

2.1 Genomic data warehousing systems

The progress in the area of genomic research in the recent years has led to a variety of different databases. Molecular biology deals with complex biological problems and a huge amount of resourceful data are being produced by high-throughput sequencing techniques. Hence, there is a continuous increase in the total number of databases, as well as the data itself and thus leading to a rise in the heterogeneity and distribution of the data. Unfortunately, there are a number of challenges associated with biological data, from the lack of standard formats to data inconsistencies resulting from experimental data variations.

The importance of database integration in this context is very important. Hence, there is a need for a paradigm shift, from single organism databases to working with more complex data warehousing systems that can accommodate the wealth of genomics data from numerous organisms and provide effective mining tools for comparative genomics and system-wide queries. To facilitate the integration and querying of genomics data, a number of data warehousing tools and frameworks have been developed, each differing in its design and capabilities, as well as the intended users. In this section we provide a broad review of those genomic data warehousing tools and frameworks.

2.1.1 Atlas

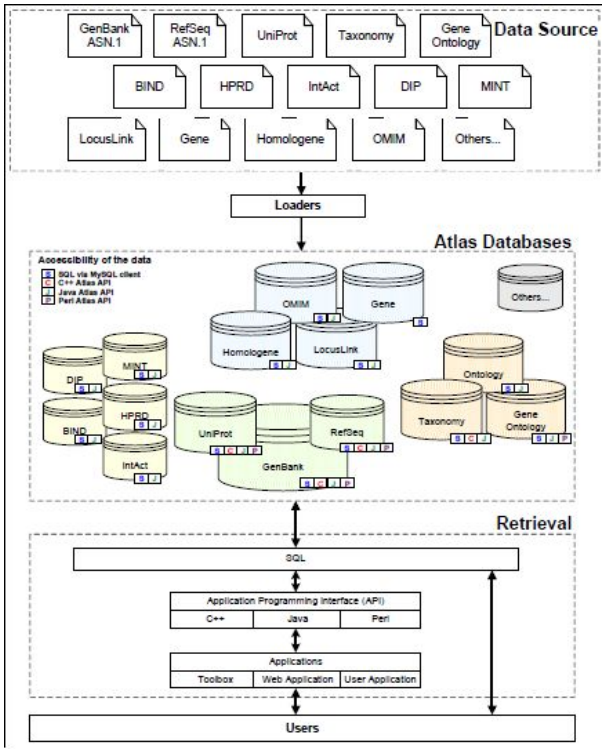


Figure 1 The architecture of Atlas together with the data sources, integral database and its retrieval systems (Shah, et al. 2005)

Atlas is a data warehouse for integrative bioinformatics (Shah, et al. 2005). Atlas integrates biological sequences, homology information, biological ontologies, molecular interactions and functional annotation of genes under the same platform for research and development purposes. The main aim of Atlas is to facilitate integration of data from disparate sources and provide a bioinformatics workbench (Shah, et al. 2005) for inferring biological relationships between the entities stored. The architecture of Atlas is depicted in Figure 1.

2.1.2 BioDWH

BioDWH is a data warehouse toolbox for the integration of life science data (Töpel, et al. 2008). BioDWH retrieves life science information from different public repositories and performs a standard integration of the data into a local database management system. This data warehouse is mainly used in the field of integrative bioinformatics for the purpose of research and development (Töpel, et al. 2008). The architecture of BioDWH is illustrated in Figure 2.

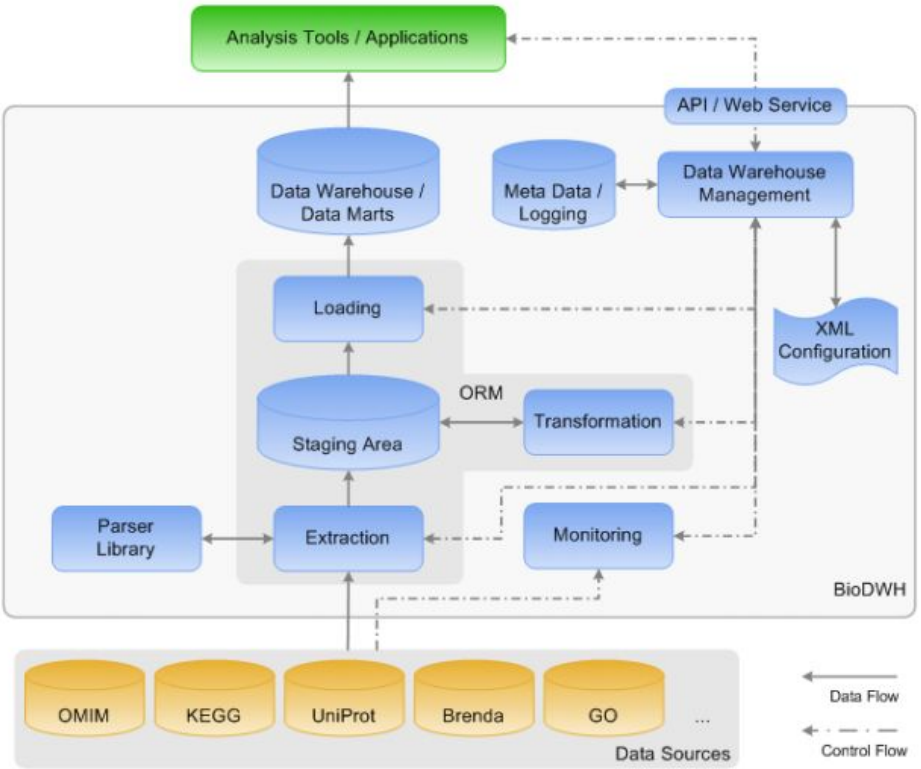


Figure 2 The System architecture of BioDWH (Töpel, et al. 2008)

2.1.3 BioMart

BioMart is a data federation framework (Zhang, et al. 2011) that provides a unified user interface to multiple data sources that may be distributed worldwide (Smedley, et al. 2015). One of the main benefits of BioMart is its ability to integrate any existing data source. The data sources are presented to the user through a unified set of graphical and programmatic interfaces so that they appear to be a single integrated database. The software package also includes ***MartConfigurator***, a user-friendly tool that facilitates the configuration of the web user interface and the definition of the relationships between data sources. It also provides REST/SOAP and Java APIs, as well as SPARQL for semantic queries. Given a user's input, the BioMart distributes parts of the query to individual data sources, collects the data and presents the user with the unified result set. BioMart has been successfully used by numerous laboratories and consortia to build integrated portals for cancer-related, microarray and gene expression data (Triplet T., and Butler G., 2013).

2.1.4 BioXRT

BioXRT was designed as a desktop application to allow biologists to publish their data on the Internet with only minimal knowledge of database design and usage. More specifically, BioXRT is known to be an excellent choice for small and medium sized laboratories, with a need to publish their results and correlate them to data from other public sources. The system's simple setup allows the researchers to bring their database online quickly, and its flexible database

schema can manage database expansion of unforeseen complex data without the need for database or software redevelopment.
BioXRT was implemented in Perl and allows researchers to convert spreadsheets to the internal XRT data schema into the underlying MySQL (http://www.mysql.com) database. The XRT schema consists of four (4) CrossReferenced Tables, which describe data, their structure and their relationships. The Cross-Referenced Tables (XRT) model is highly flexible and may be expanded as needed to accommodate unforeseen data complexities.
BioXRT has been used in a number of projects, ranging from the annotation of the Human Chromosome 7 to the study of structural variation of chromosomes in autism spectrum disorder, thereby demonstrating the versatility and flexibility of the framework (Triplet T. and Butler G., 2013).

2.1.5 InterMine

InterMine has been implemented as an open-source data warehouse system for facilitating the building of databases with complex data integration requirements and a need for a fast customizable query facility. It relies on a traditional ETL (Extract, Transform and Load) architecture and provides a core data model and a collection of parsers to load data from 28 typical data sources such as the Gene Ontology (Ashburner, et al. 2000), the Kyoto Encyclopedia of Genes and Genomes (Kanehisa, et al. 2012) or the Protein Data Bank (Rose, et al. 2013).
Using InterMine, large biological databases can be created from a range of heterogeneous data sources, and the extensible data model allows for easy integration of new data types. The analysis tools include a flexible query builder, genomic region search and a library of 'widgets' performing various statistical analyses. It provides the facility to export the results in many commonly used formats. InterMine is developed as a fully extensible framework where developers can add new tools and functionality. The default web-based interface of InterMine can be customized and enhanced with widgets and plugins such as the genome browser GBrowse (Donlin, et al. 2009), the interaction graph viewer Cytoscape (Cline, et al. 2007) and gene expression heat maps.
InterMine is mainly implemented in Java. It translates the data model from a data source into a normalized database schema and loads the relevant data into an underlying PostgreSQL (http://www.postgresql. org) database with optional pre/post-processing steps. InterMine also caters for Java, Perl, Python, Ruby and RESTful APIs for programmatic access to the data and the implementation of automated workflows. It has been successfully leveraged to build warehouses describing omics data from numerous organisms (Balakrishnan, et al. 2012, Chen, et al. 2011, Lyne, et al. 2007).

2.1.6 Prokaryotic Genome Analysis Tool (PGAT)

PGAT is a web enabled database application that is mainly responsible for multi strain analysis of genomes (Brittnacher, et al. 2011). The main features of PGAT include gene comparison at the sequence level, the computation of pan-genome of the selected strains by the end user, the identification of Single-Nucleotide Polymorphism (SNPs) within a set of orthologs, the determination of the absence or presence of a set of user defined genes in some metabolic pathways as well as

their comparison (Brittnacher, et al. 2011). PGAT also supports manual community annotation.
The back end of PGAT is implemented using a relational database that runs on a PostgreSQL server while the front end is implemented using Perl CGI scripts. These scripts run on Apache Web Server (Brittnacher, et al. 2011).
The output of one feature of PGAT is illustrated in Figure 3.

Search Results [New Search]

Present in Genomes	Absent in Genomes	Options
Acinetobacter baumannii 1656-2 Acinetobacter baumannii AB0057 Acinetobacter baumannii AB307-0294	Acinetobacter baumannii ACICU	Only display genes present in **all** selected **Present** genomes

Text Report | FASTA aa file | FASTA nt file

☐ Show all reference genes

62 Genes found (first 20 displayed only)

** indicates pseudogene*
 # indicates CDS overlapping contig end(s)

	Poson number	Locus tag	Genbank accession	Gene name	Description	Acinetobacter baumannii 1656-2 (64)	Acinetobacter baumannii AB0057 (64)	Acinetobacter baumannii AB307-0294 (62)	Acinetobacter baumannii ACICU
1	pCP001172_037793	ABBFA_003320	ACJ58522.1	-	hypothetical protein	pCP001921_002862	pCP001182_002894	ABBFA_003320	
2	pCP001182_000082	AB57_0009	ACJ39441.1	ampC	beta-lactamase ADC7	ABK1_2681	AB57_0009 AB57_2796	ABBFA_001076	ACICU_02564*
3	pCP001182_002480	AB57_0207	ACJ39638.1	-	hypothetical protein	ABK1_0201	AB57_0207	ABBFA_003351	ACICU_00193*
4	pCP001182_002874	AB57_0237	ACJ39664.1	-	outer membrane protein OprE3	ABK1_0228	AB57_0237	ABBFA_003321	ACICU_00219*
5	pCP001182_002897	AB57_0239	ACJ39665.1	-	hypothetical protein	ABK1_0229	AB57_0239	ABBFA_003319	
6	pCP001182_002904	AB57_0240	ACJ39666.1	-	putative dihydrodipicolinate synthase	ABK1_0230	AB57_0240	ABBFA_003318	
7	pCP001182_002913	AB57_0241	ACJ39667.1	-	class II aldolase/adducin domain protein	ABK1_0231	AB57_0241	ABBFA_003317	
8	pCP001182_002923	AB57_0242	ACJ39668.1	-	transcriptional regulator, GntR family	ABK1_0232	AB57_0242	ABBFA_003316	
9	pCP001182_004503	AB57_0362	ACJ39788.1	tuf (tufB,tufA)	translation elongation factor Tu	ABK1_0856 ABK1_0323	AB57_0362 AB57_0914	ABBFA_003256	ACICU_00296* ACICU_00818*
10	pCP001182_005571	AB57_0440	ACJ39864.1	-	magnesium Mg(2+)/cobalt Co(2+) transport protein	ABK1_0401	AB57_0440	ABBFA_003174	ACICU_00374*
11	pCP001182_007465	AB57_0607	ACJ40028.1	yfgL	outer membrane assembly lipoprotein YfgL	ABK1_0546	AB57_0607	ABBFA_003030	ACICU_00515*
12	pCP001182_008942	AB57_0738	ACJ40537.1	pabB	P-aminobenzoate synthetase	ABK1_0671	AB57_0738	ABBFA_002927	ACICU_00634*
13	pCP001182_009991	AB57_0819	ACJ40617.1	-	hypothetical protein	ABK1_0766	AB57_0819	ABBFA_002843	pCP000863_009142*
14	pCP001182_011793	AB57_0982	ACJ40774.1	-	major facilitator superfamily MFS_1	ABK1_0903	AB57_0982	ABBFA_002694	ACICU_00870*
15	pCP001182_012296	AB57_1021	ACJ40813.1	-	rieske (2Fe-2S) protein	pCP001921_011428	AB57_1021	pCP001172_030025	pCP000863_011275*

Figure 3 This feature shows the presence and absence of genes in a set of genomes

2.1.7 Integrated Microbial Genomes (IMG)

IMG is a data warehouse implemented for the purpose of microbial genome analysis. The first version of IMG was released in 2005 (Markowitz, et al. 2006) and since then, new analysis tools have been implemented for the comparison of genomes from the three domains of life (Markowitz, et al. 2013). The main aim of IMG is to compare publicly available genomes, genomes submitted on IMG by sequencing centres, draft genomes, genomes of viruses and plasmids to answer biological queries. IMG provides researchers with a wide variety of analysis tools for comparing genomes, genes and gene functions (Markowitz, et al. 2013). Moreover, IMG integrates third-party platforms such as VISTA (Mayor, et al. 2000), Dot Plot (Grigoriev, et al. 2011) and Artemis ACT (Carver, et al. 2005) to enhance the viewing of the genomes in a comparative context. Multiple web based platforms are also integrated in IMG to improve the analysis of genomes.
The architectural and data model developed for IMG is not available. However, IMG provides detailed documentation for the usage of its tools. An analysis tool of IMG is illustrated in Figure 4.

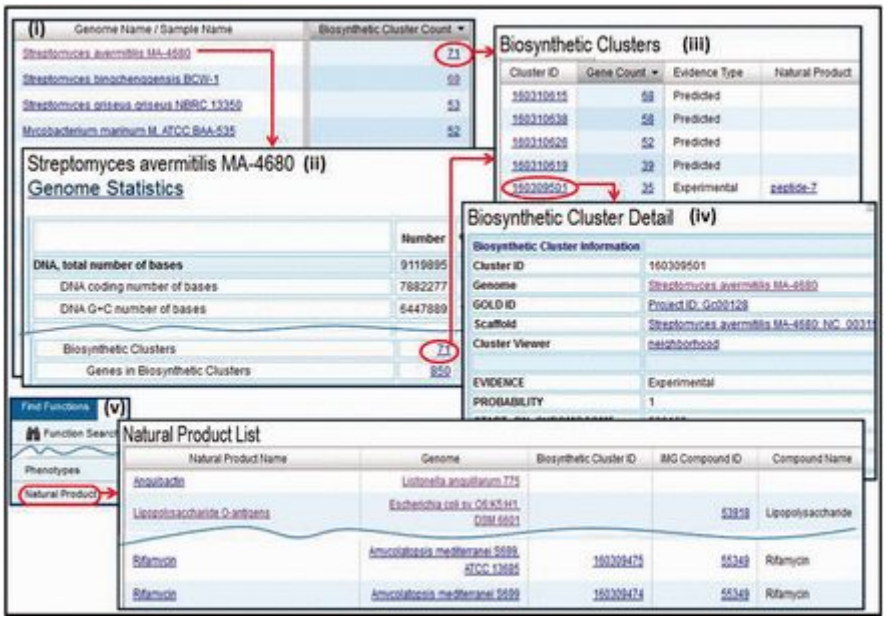


Figure 4 Biosynthetic clusters generated by IMG (Markowitz, et al. 2013)

Chapter 3 – Design

This chapter describes the design structure of the proposed data warehouse. It follows the general schema of a data warehouse whereby information has been taken from disparate sources and integrated into one single repository for ease of analysis and retrieval of biological data. The architecture of proposed system is similar to a typical data warehouse. The schematic illustration is shown in Figure 5.

3.1 Data Sources

This section describes the data sources used in building the data warehouse and the reasons for using these specific data sources.

3.1.1 GenBank National Center for Biotechnology Information (NCBI) FTP service

GenBank stores bacterial genomes from different sequencing centres and make them publicly available via FTP service (Wheeler, et al. 2007). It is also a central repository which integrates genomes from the EMBL data library (Kanz, et al. 2005) and DNA databank of Japan (DDBJ) (Miyazaki, et al. 2004), thus ensuring a uniform distribution of genomes across all the databases.
GenBank was mainly chosen because the FTP service provided by GenBank makes it easier to download specific bacterial genomes programmatically as compared to other biological databases. Moreover, as GenBank already integrates data from the main biological databases like EMBL and DDBJ; it therefore contains a wide variety of bacterial genomes and also ensures that its data is reliable. Furthermore, the finished genomes found in GenBank were stored in the flat file format (GBK format) which made it easier to parse and retrieve the required information for populating the database with the biological data of infectious bacteria.

3.1.2 American Biological Safety Association (ABSA)

The American Biological Safety Association is mainly responsible for addressing the needs of biosafety professionals (https://my.absa.org/tiki-index.php?page=Riskgroups). ABSA provides a Risk Group database which contains the list of all infectious bacteria classified by their genus. The database also provides the pathogenicity and host range for the infectious bacteria.
ABSA was chosen as a data source because it provides the list of infectious bacteria and their corresponding host specificity in a user friendly way. This information is available from published literature and would have required a text-mining approach. Moreover, the list of infectious bacteria and their corresponding host specificity was easily accessible programmatically so as to integrate the information in the data warehouse. The list of infectious bacteria

obtained from ABSA was used as a point of reference to target the download of the corresponding bacterial genomes from the NCBI FTP site.

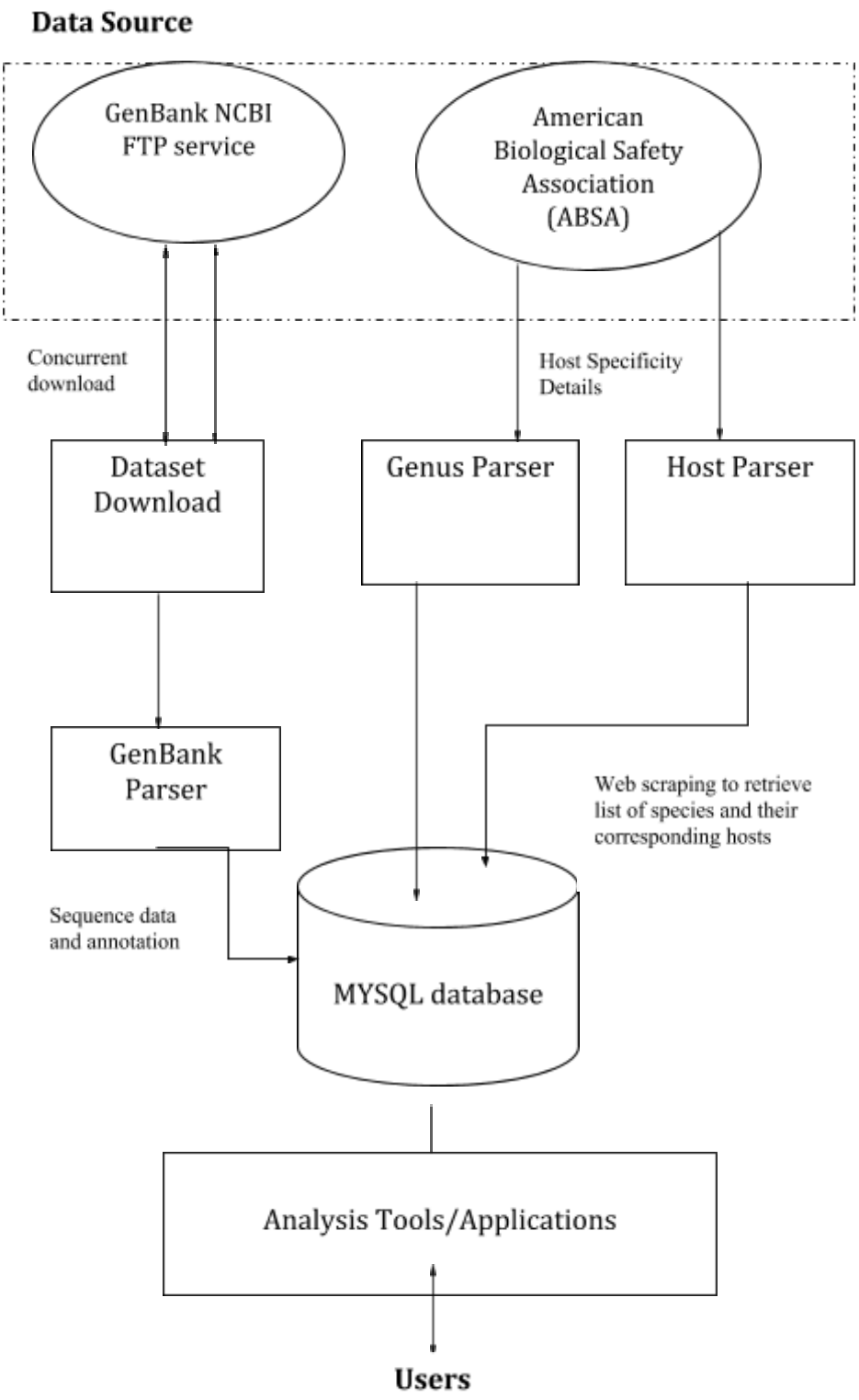


Figure 5 The architecture of proposed system

The data integrated in the data warehouse is clearly shown in the data source panel. The square boxes denote the modules implemented to clean, parse and retrieve the appropriate data which was then loaded in the MySql database. The analysis tools and applications are used to retrieve and process the biological data so as to provide correct information to users of the data warehouse.

3.2 Modules Design

This section describes the purpose and structure of the modules that form the backend of the data warehouse and how these modules were used to retrieve the relevant information in order to build the complete data warehouse.

3.2.1 Genus Parser

The Genus Parser module handles the extraction of the list of genera from ABSA, filters any duplicate or incorrect information and finally inserts the appropriate list of genus in the database (figure 6). The list of genus for pathogenic bacteria is already available on the html page of ABSA and the data was pulled and filtered directly into MYSQL database. This module ensures a dynamic population of the list of genus for infectious diseases in the data warehouse.

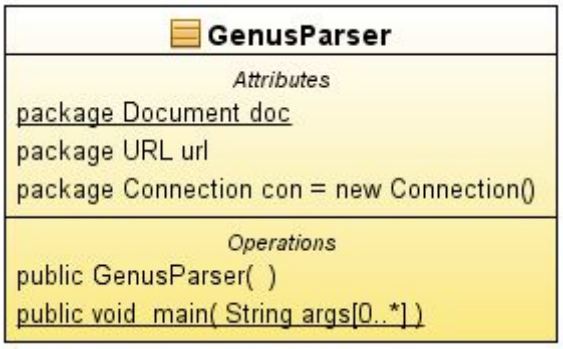


Figure 6 Genus Parser Module Class Diagram

3.2.2 Host Parser

The Host Parser module is mainly responsible for identifying the complete set of pathogenic bacteria and their respective hosts (figure 7). It is dependent on the Genus Parser module as it uses the list of genera extracted from Genus Parser to automatically mine the list of species and the corresponding hosts (human, animal, plant) they infect. The filtered information is thereby populated in the respective database.

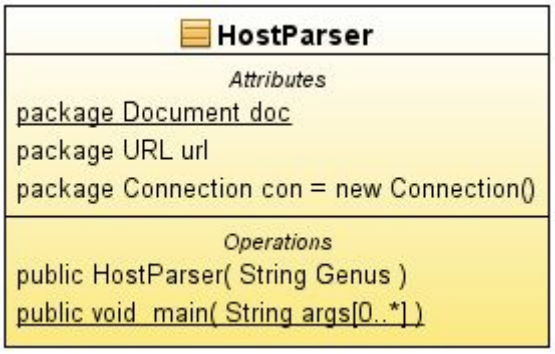


Figure 7 Host Parser Module Class Diagram

3.2.3 Dataset Download

The Dataset Download module handles the automated downloading of GenBank files for infectious bacteria from the NCBI FTP site (figure 8). This module caters for the complexity of connecting, searching and downloading of GenBank files for each genus and specie that fall into the category of infectious diseases. This module uses the list of genus and specie retrieved from ABSA to search for the corresponding GenBank files and download them concurrently in the specified directory. The concurrent download is achieved by the use of multi-threading.

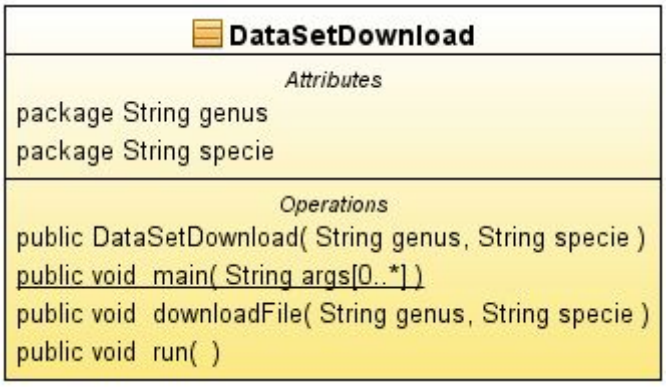


Figure 8 Dataset Download Module Class Diagram

3.2.4 GenBank Parser

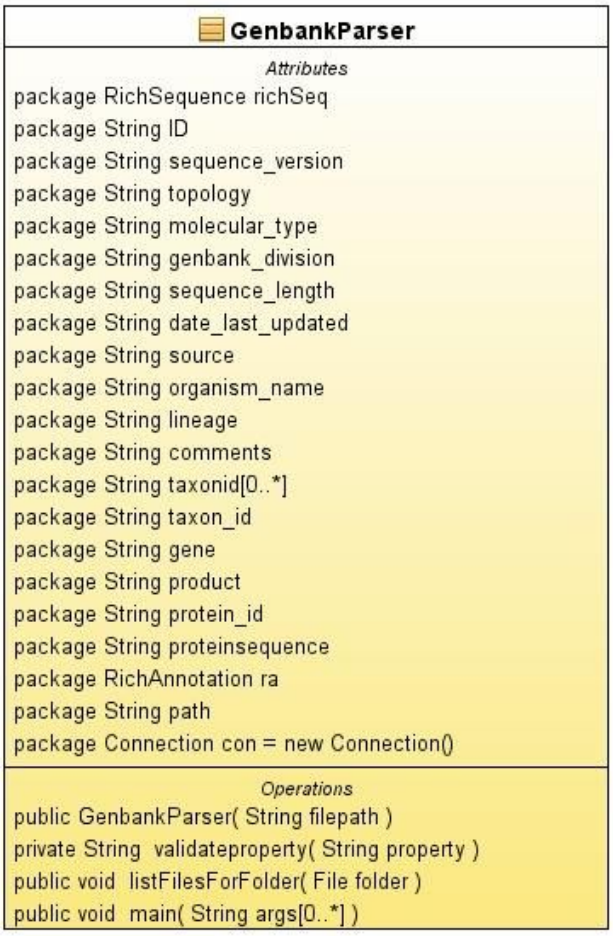


Figure 9 GenBank Parser Module Class Diagram

The GenBank Parser module parses GenBank files and extracts relevant information such as organism name, source information, sequence length; CDS feature tags including the gene names, locus tags, corresponding protein sequences and gene references amongst others (figure 9). After the extraction of the specific data, the module performs a sequential insertion of the features for each specie in the respective tables of the database. The insertion is performed sequentially to avoid any type of foreign key constraints conflict and to ensure that the complete set of features for a specie has been inserted successfully in the database.

3.3 Database Design

3.3.1 Database schema

The database schema for the data warehouse consists of six main tables, which have been normalised accordingly to facilitate the searching and analysis of the biological data stored. The main table describes the organism and stores the primary information such as organism name, GenBank ID, sequence length amongst others from the GenBank files for specific strains downloaded from

NCBI. The organism table is linked to the reference table for publications or references stored in one GenBank file. The same principle applies to the cds table which stores the coding sequences and their respective features for each strain found in the organism table. Since, each coding sequence (CDS) has at least three external database references, the gene_ref table stores the references for each CDS found in the cds table. The genus and species_host tables have been parsed from ABSA and contain information about the infectivity of all species whose genomes are in the data warehouse. The information stored in these tables are processed programmatically to perform sequence analysis. Figure 10 gives a detailed outline of the database design of the data warehouse.

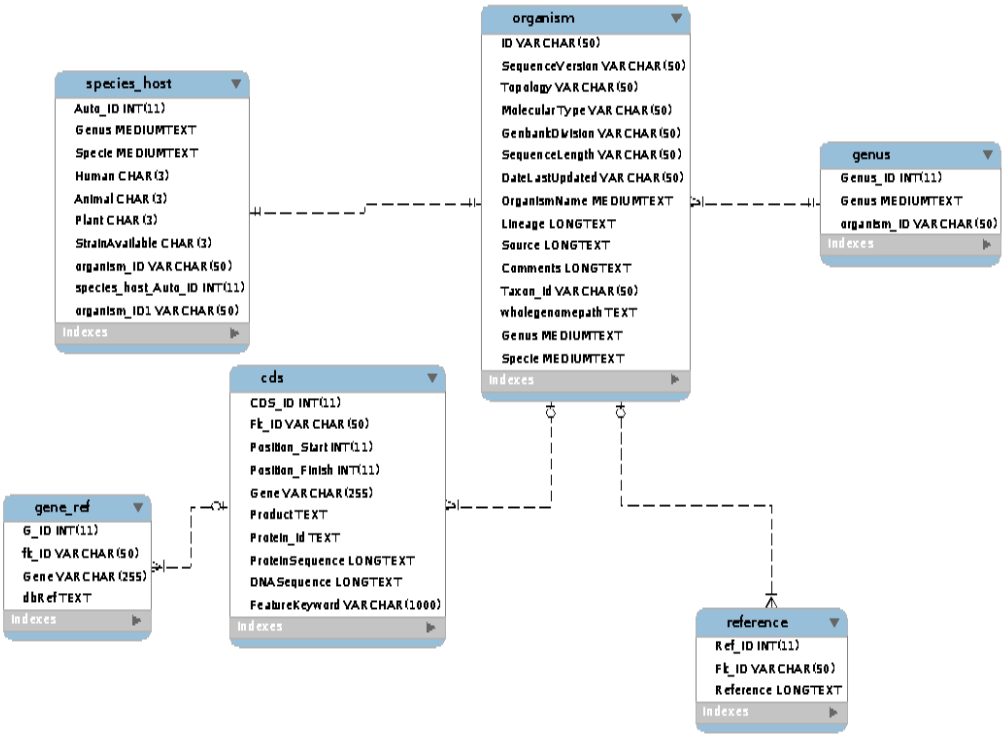


Figure 10 Database schema for IDEAS data warehouse

3.3.2 Database Connection

All the database connections are handled by the ***Connection*** class (figure 11). The Connection class contains all the Connection objects, parameters to connect to the database and methods used by the analysis tools for programmatically solving the biological problems. Objects of the connection class are used by the analysis tools for connecting and retrieving specific information from the database.

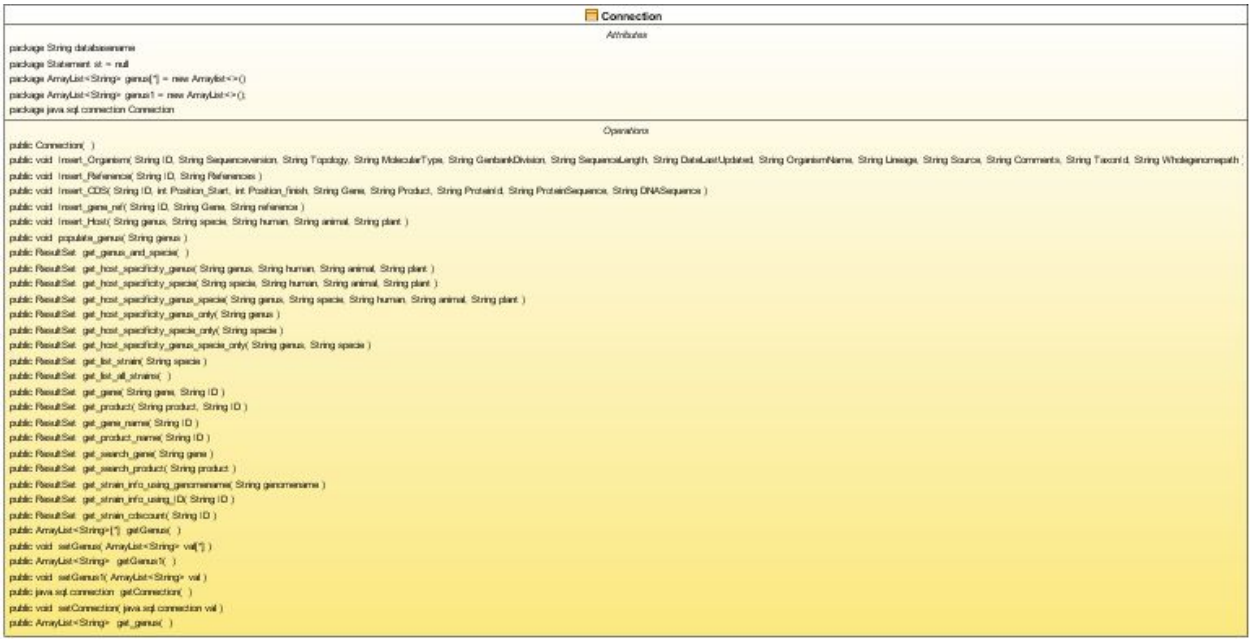


Figure 11 Class Diagram for Connection Class

3.4 Genome Comparison

Genome comparison involves the comparison of sequenced genomes, particularly for the identification of insertions, deletions and variation in syntenic regions. Comparison can be of types within-genome or between-genome. Within-genome comparisons focus on the genome of a single species, including variations in base composition, k-tuple frequency, gene density, numbers and kinds of transposable elements and segmental duplications. Between-genome comparisons use closely related species for identifying conserved genes, gene structure and organization and control elements. More distantly related species are used for phylogenetic profiling.

When using annotated sequences, genomes can be compared using the gene names, product names and sequence comparison. For comparison based on gene names, the genomes are searched for features that have common gene names. For comparison based on product names, the genomes are compared using product names.

Since all genomes are not annotated in a consistent manner, using gene names and product names may miss a number of similarities between genomes being compared. Hence, sequence comparison can lead to a better comparison between genomes. The local blast software can be used to perform comparison between features of genomes.

3.4.1 NCBI local blast application

BLAST(Basic **L**ocal **A**lignment **S**earch **T**ool) is an algorithm for comparing primary biological sequence information, such as the amino-acid sequences of different proteins or the nucleotides of DNA sequences. NCBI Local blast is developed in C language. The features of local blast include the following:

- It allows users to enter a specific sequence or to upload a genomic file (GENBANK or FASTA format) for searching against target databases.

- Searches can either be done on protein sequences or on nucleotide sequences.
- There are different variations of BLAST such as blastp and blastn amongst others for comparing query. ***blastp*** is used for comparing protein sequences and ***blastn*** is used to compare nucleotide sequences.
- It uses the command line interface.
- ***BLAST*** also calculates statistical significance of query matches.
- It uses the e-value (expectation value) to demonstrate the similarity between a query sequence and a target sequence)

3.4.2 Comparison of genomes based on NCBI local blast

In order to perform comparison of sequences using local blast, the features from chosen genomes are used to build a blast database. Then the feature from each genome is queried onto the blast database using specific cutoff parameters for query coverage and e-value. Hence for each feature from each genome, a list of friends are listed (based on cut-off parameters specified) and this can be used to identify the common genes/features between chosen genomes. Figure 12 below demonstrates the steps performed in this comparison.

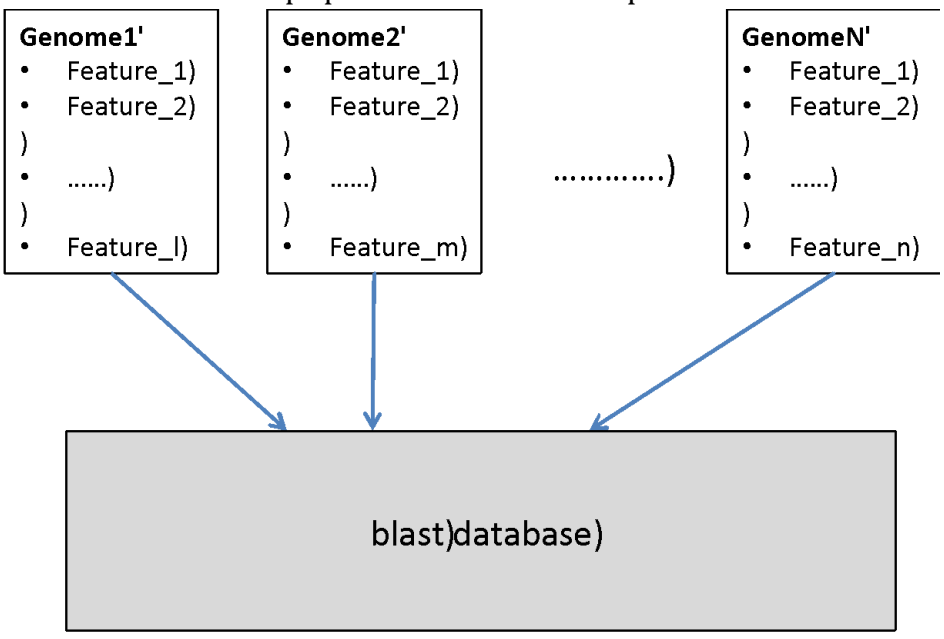


figure 12(a) – create the blast database

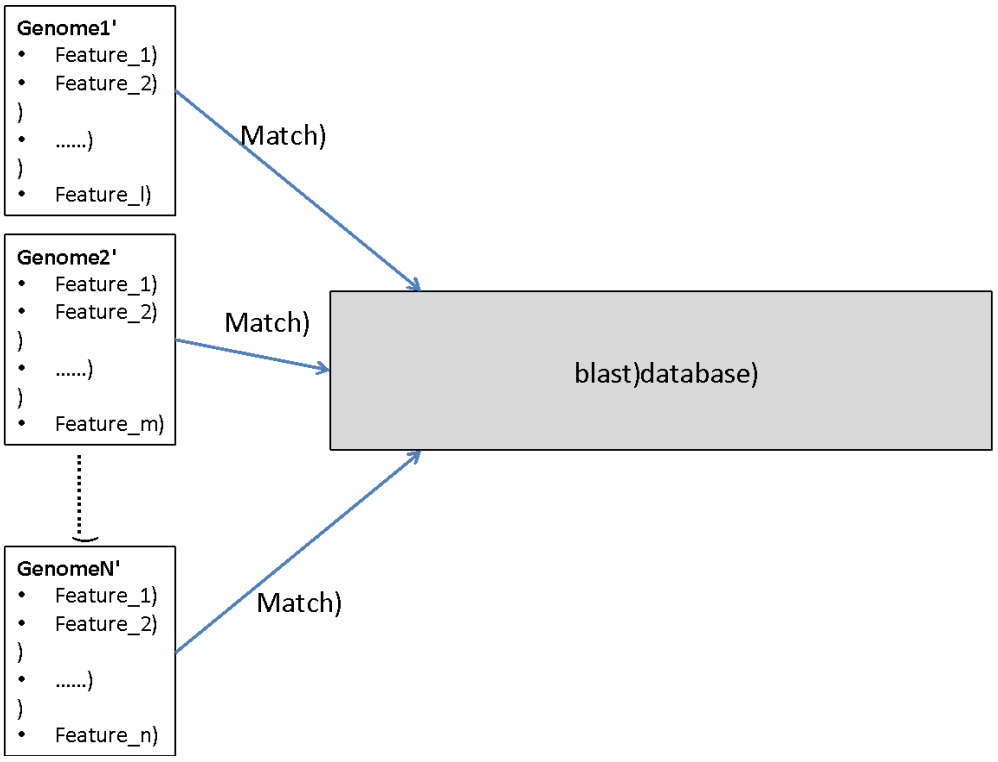


figure 12(b) – search for common features

3.5 Analysis Tools

Once the bacterial strains have been integrated in the data warehouse, a comparison of chosen strains is performed in order to extract the common features between related strains of chosen organisms, using the local-blast method. Then using the common features, a number of analyses can be performed using existing tools.

This section describes the architectural model, interface design and main schema of the analysis tools developed until now.

3.5.1 Architecture

A brief description of the architecture of the application tools is provided hereunder. Figure 13 gives a schematic representation of the flow of information and the relationship of the Analysis Tools with different entities of the system.

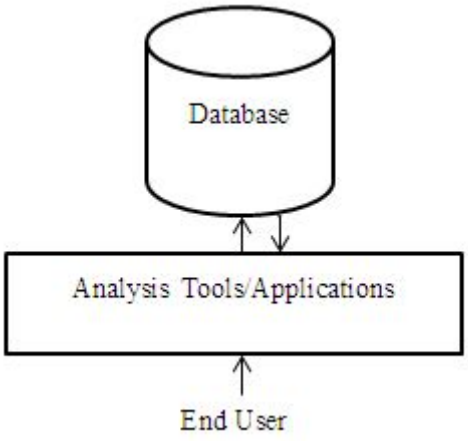


Figure 13 Schematic Representation of the Analysis Tools

3.5.2 Interface Design

This section illustrates the graphical user interfaces (GUIs) of the IDEAS web-based application. Each screen and functionality is described.

Main screen

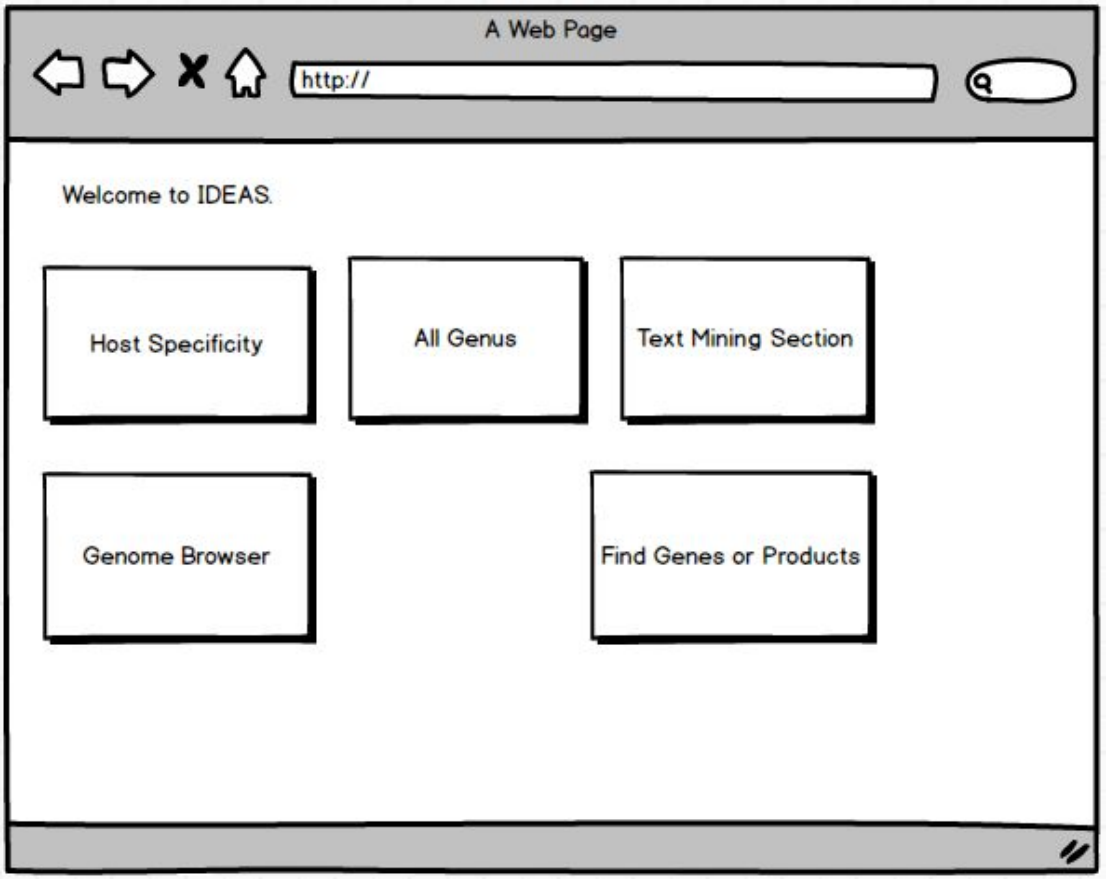


Figure 14 Main Screen

Upon startup, a user has the option to access the application based on:

- **All Genus:** whole genome set available in the data warehouse or
- **host specificity** i.e. data based on bacteria affecting human, animal or plant or more than one host.
- **Text Mining Section:** retrieve important from research papers using their pubmed ids

Choosing strains from whole genome set

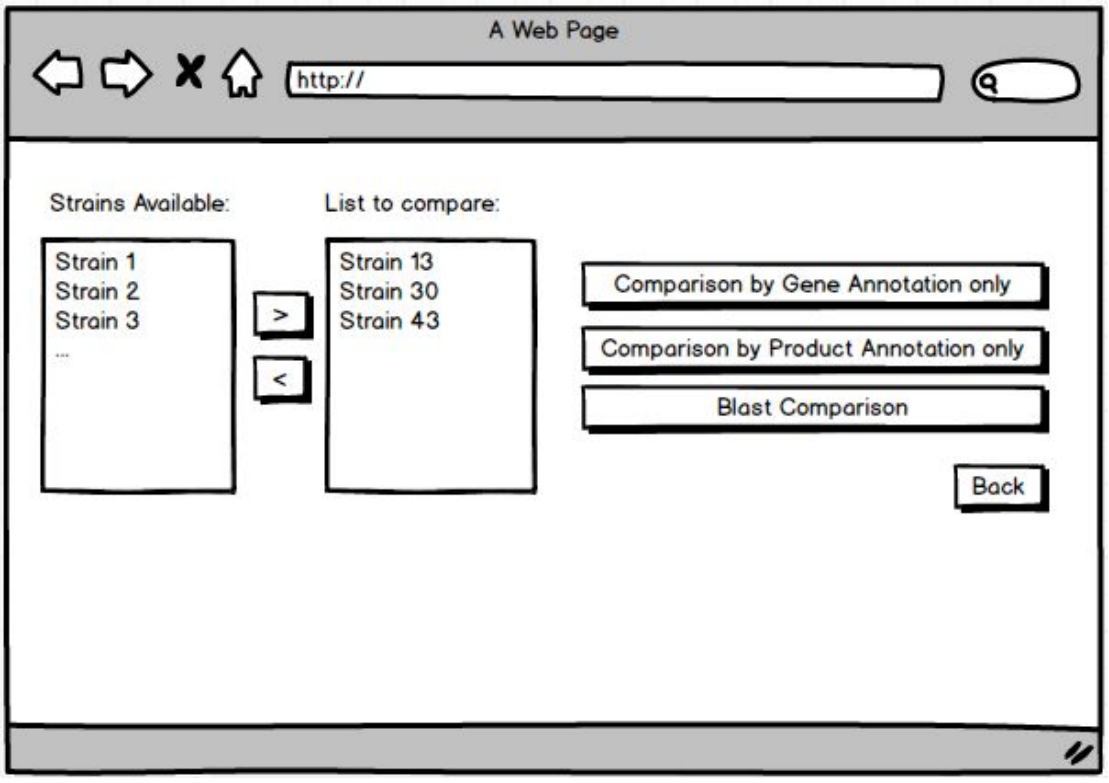


Figure 15 Whole Genome Set Analysis screen

This option provides the user with all the available strains in the data warehouse so that user can choose strains of interest to him/her and proceed with further analysis (Figure 15). More specifically, it has the following components:

1. **Forward Button** : This button adds the selected strain from the list of strains available to the list being chosen for comparison when clicked by the user
2. **Backward Button**: This button adds the selected strain from the list available for comparison back to the initial list of strains chosen.
3. **Comparison by Gene Annotation only**: This button is used to compare the selected strains using their gene annotation.
4. **Comparison by Product Annotation only**: This button is used to compare the selected strains using their product annotation.

5. **Blast Comparison**: This button is used to choose the ncbi local blast program to find the common genes between chosen strains and opens the common genes screen.
6. **Back**: This button allows user to go back to the main menu.

Display common genes screen

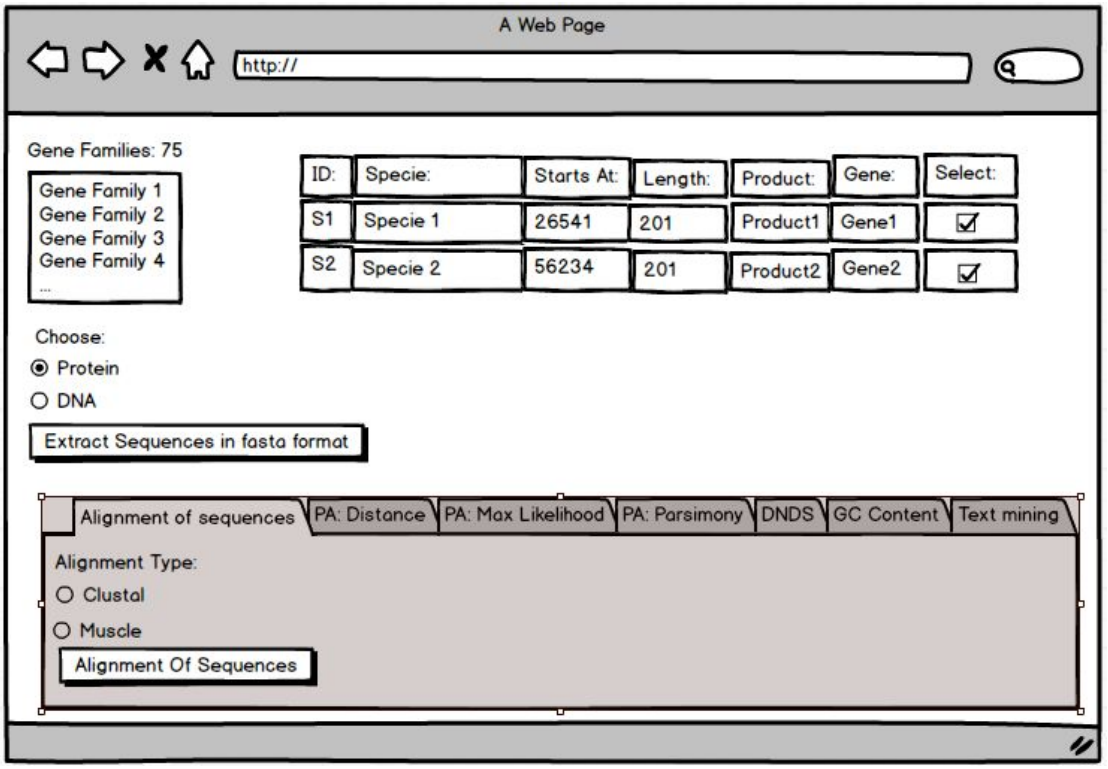


Figure 16 Display common genes screen

After selecting strains of interest, user can perform a comparison of the features between these strains, using gene annotation (figure 16). The resulting screen has the following components:

1. **Extract sequences as Fasta Format button**: This button extracts selected Protein or DNA sequences in Fasta format and displays the sequences in a text editor. The user can also download the sequences to a fasta file.
2. **Alignment of sequences button**: This button aligns the selected Protein or DNA sequences using clustalW from Biojava or muscle software and displays the alignment graphically.
3. **Phylogenetic Analysis: distance**: This button builds the phylogenetic tree using distance based methods from the selected aligned gene sequences and displays the tree using the javascript library jsPhyloSVG.
4. **Phylogenetic Analysis: Maximum Likelihood**: This button builds the phylogenetic tree using the maximum likelihood method from the selected aligned gene sequences and displays the tree using the javascript library jsPhyloSVG.

5. **Phylogenetic Analysis: Parsimony**: This button builds the phylogenetic tree using the Parsimony method from the selected aligned gene sequences and displays the tree using the javascript library jsPhyloSVG.
6. **GC content button**: This button calculates and displays the GC content of the selected sequences in a graphical view.
7. **dN/dS Analysis**: This button allows the user to perform dN/dS analysis of the selected sequences using the JCoda application.
8. **Protein or DNA radio button**: This button allows the user to choose whether protein or DNA sequences should be extracted in Fasta format as well as which sequences should be aligned.

Host specificity screen

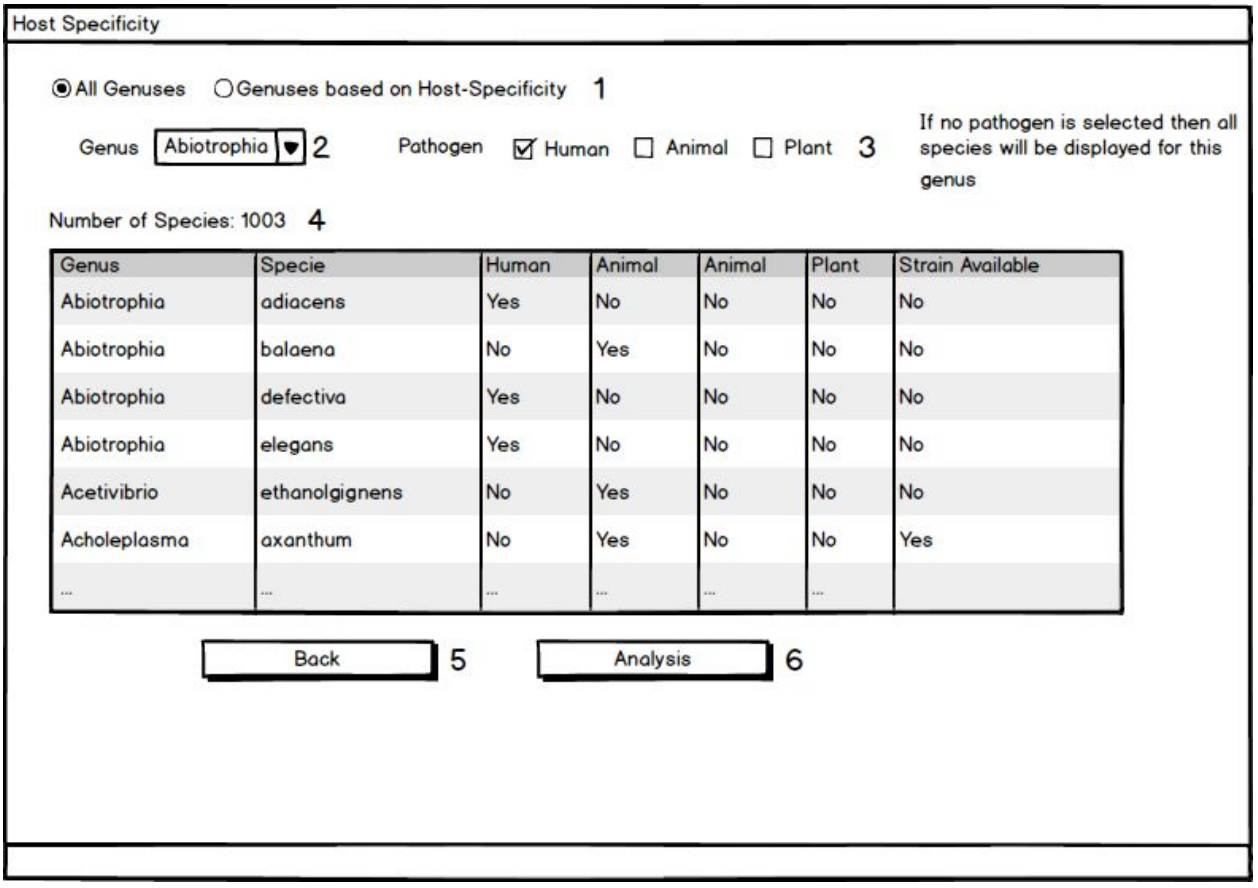


Figure 17 Host Specificity Screen

At the time of startup, if a user decides to perform analysis based on host-specificity, s/he can do so using the above screen which has the following features.

1. **All Genera/Genera based on Host-Specificity** radio button: User can select ***All Genera***, or Genera that affect Human, Animal or Plant.
2. **Genus Drop-down List**: This list becomes available if user selects "Genera **based on Host-Specificity**" in 1.This drop-down is available if

the user wishes to select a particular genus from the available list e.g. *Mycobacterium*.

3. **Pathogen Check box**: This check box is available so that the user can filter the hosts for a particular search term. For a chosen genus, user can select the hosts that it can infect i.e human, animal and plant. For instance, if the user wants to display species that infect both human and animal, the respective check boxes must be selected.
4. **Number of species label**: On loading this screen, this label gives the number of species found in the database and their respective hosts. The number of this label corresponds to the number of rows in the table. If the user performs a search for genus and specie, the table is reloaded and the value of the label changes accordingly.
5. **Back**: This button allows user to go back to the main menu.
6. **Analysis Button**: User can select the rows for which strains are available in the data warehouse and proceed for further analysis using this button. Following this option user will be provided with a screen similar Figure 17 and thereafter it follows in the same manner as choosing strains from whole set.

Text Mining Section

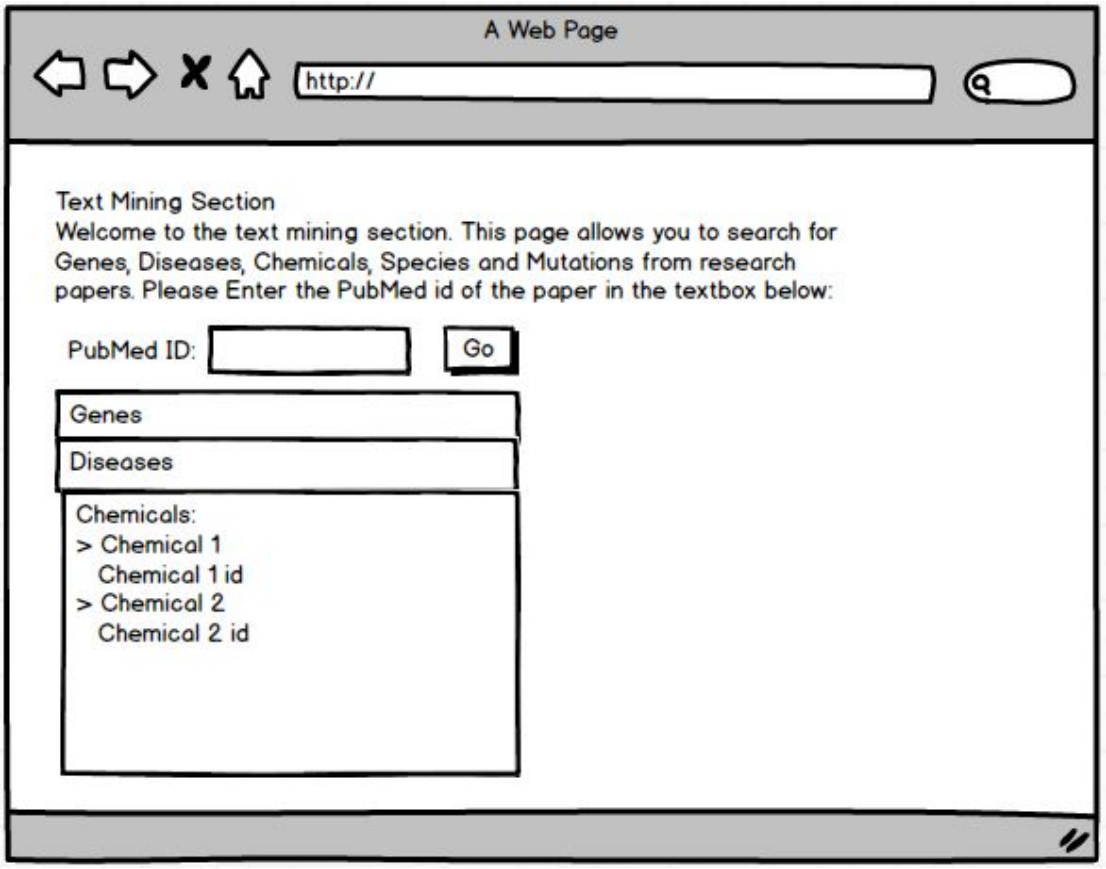


Figure 18 Text Mining Screen

If the user decides to go to the text mining section from the main menu, the above screen is displayed. The user has to type the pubmed id of the research paper in the textbox and press GO. Information about the genes, diseases, chemicals, species, and mutations are displayed in the vertical tab below.

Chapter 4 – Implementation

This chapter provides an overview of the technologies used to build the data warehouse and describes the implementation of the different modules and analysis tools.

4.1 Development Tools and Environment

This section gives a brief description of the tools and environment used to support the development of data warehouse.

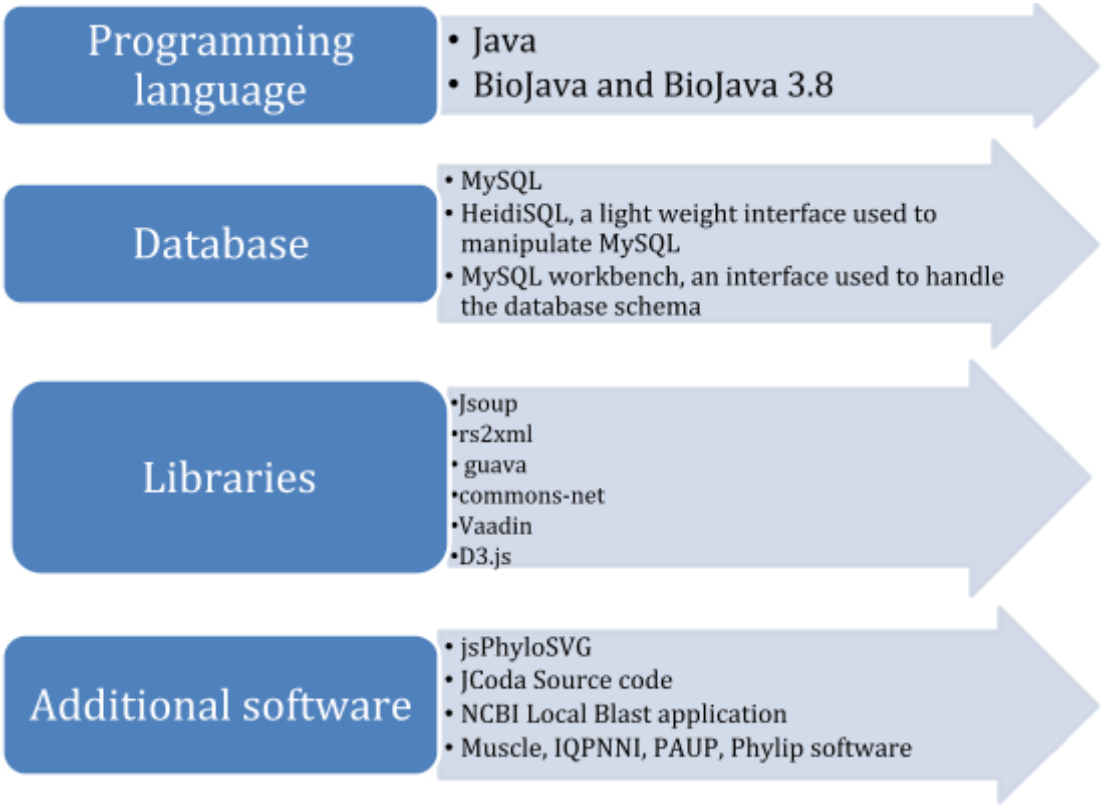


Figure 19 Tools and Environment used for the proposed system

4.2 Programming Language

Netbeans IDE 8.2 was used as programming environment together with Java Development Toolkit (JDK) 1.7. For the current standalone version Netbeans IDE was chosen because it already provides built-in components to facilitate the implementation of user interfaces. Therefore, Java programming language was the optimal choice for building the data warehouse. Moreover, BioJava libraries have multiple functionalities to ease the development process and solve multiple biological problems. All these libraries are well supported and are under review continuously to provide better solutions each time. Adequate support is available for them and they are easily integrated in Netbeans. In this way, the environment chosen ensures that newer releases of libraries or JDK will not make the developed software obsolete.

4.3 Database

MYSQL database was mainly chosen because it is a widely used open source database and it provides a lot of documentation concerning technical problems that users can encounter. Moreover, it can handle massive amount of information for querying and processing which would suit the purpose of the data warehouse. It is also easily available to users who want to download and use the data warehouse. Furthermore, it is known to be very robust and reliable. HeidiSQL 8.3.0 was used as an IDE to process and query the biological data more effectively. MySQL workbench was mainly used to generate the schema of the data warehouse since this feature is not available in HeidiSQL.

4.1.3 Libraries

Jsoup (Hedley 2010) is a library in Java for handling html pages. Jsoup has been used to parse the html pages from ABSA and retrieve the necessary information in an appropriate format to insert in the database.

Rs2xml is a library that contains functionalities for mapping database queries to JTable models. In this project, JTable was used quite often for displaying the results of multiple analyses. Hence, rs2xml was used to set the models of the table in accordance with the database queries.

Guava (Google, 2017) is a set of core libraries for Java for handling I/O, collections, String processing amongst others. In this project, Guava's set operations methods were mostly used to ease the comparison of bacterial genomes at annotation level.

The **Apache Commons Net** library provides the interface for the client side of many basic internet protocols. The library handled the FTP connection and GenBank file transfer from the NCBI FTP site to the application software in the data warehouse.

D3.js is a javascript library that can be used to display graph or perform powerful visualisation of data. Since D3.js is a javascript library and the web application is written using java language, a javascript connector was used to handle the communication between the D3.js javascript library and the web application. D3.js visualisation charts were then implemented as abstract javascript components in the web application. Values can be passed as parameters from the java web application to the javascript component via the javascript connector.

jsPhyloSVG is a simple javascript library that can be used to display phylogenetic trees. It takes as input a string in newick format and generates the

html code using javascript. It is implemented as an abstract javascript component similar to D3.js.

Vaadin is a web framework used to develop rich internet applications. It uses java as the programming language for creating web content. Google Web Toolkit is then used to render the web page from the java code. Ajax technology runs on the browser side to create an interactive user experience.

4.4 Additional software

JCoda (Steinway et al., 2010) is a Java-based open source, user-friendly visualization tool for performing evolutionary analysis on homologous coding sequences. JCoDA can be used to rapidly screen for genes and regions of genes under selection using PAML. JCoda has been integrated with the web application to perform dN/dS analysis of selected features from strains of interest.
These software tools were chosen because they provide additional functionalities on top of the analysis performed by the web application.
Phylip – Phylip is a package of programs for phylogenetic inference. It contains several executables that can be used to analyse molecular data and construct phylogenetic trees. Phylogenetic trees are constructed from genetic distances by using specific substitution models (pairwise distance methods). Some executables that were used are listed as follows:

1. **DNADist.exe** – dnadist.exe was used to calculate the genetic distances using nucleotide distance matrices: F84, Kimura, Jukes Cantor, Logdet.
2. **ProtDist.exe** – protdist.exe was used to calculate the genetic distances using amino acid distance matrices: Jones Taylor Thornton matrix, Henikoff Tillier Pbm matrix, Dayhoff Pam matrix, Kimura Formula, Categories Model.
3. **Neighbor.exe** – neighbour.exe was used to calculate the Neighbour Joining and UPGMA tree from the genetic distances.
4. **Fitch.exe** – fitch.exe was used to calculate the Fitch Margoliash and Minimum Evolution tree from the genetic distances.
5. **Seqboot.exe** – seqboot.exe was used to generate bootstrap replicates of the aligned DNA or protein sequences.
6. **Consense.exe** – consense.exe was used to generate the consensus tree from bootstrap replicates.

IQPNNI – IQPNNI is an efficient tree reconstruction algorithm that can be used to build a maximum likelihood tree. It was implemented with bootstrap, rate Heterogeneity, Nucleotides substitution models and amino acid substitution models.
Nucleotides substitution models that have been implemented are as follows:

- HKY85 (Hasegawa et al. 1985)
- TN93 (Tamura-Nei 1993)
- General Time Reversible

Amino acids substitution models that have been implemented as follows:

- WAG (Whelan-Goldman 2000)

- JTT (Jones et al. 1992)
- mtREV24 (Adachi-Hasegawa 1996)
- rtREV (Dimmic et al. 2001)
- BLOSUM62 (Henikoff- Henikoff 1992)
- Dayhoff (Dayhoff et al. 1978)

MUSCLE – Muscle is a standalone software that can be used to compute an iteratively refined alignment. In other words, muscle computes the alignment of sequences using an iterative scheme that gradually diverge towards the optimal alignment. It takes in as input sequences in fasta format and returns the result in aligned fasta format. The result is then parsed and displayed in a user friendly format.

Apache Tomcat web server – Apache tomcat is a servlet container. It is used to deploy the web application. Since the web application is developed using vaadin framework, it contains servlet classes. Therefore it must be compiled and deployed as a java web application using apache tomcat. Once deployed the web application can be accessed from a browser.

4.5 Web Services

NCBI text mining web service - NCBI provides a text mining web service that searches for research papers using their PubMed ids. Research papers are processed and tagged on their web application according to 5 different BioConcepts: Gene, Disease, Chemical, Species and Mutation. A list of PubMed ids, the BioConcept and response format is used as input and the tags will be retrieved from the server. A short description of the paper and a list of genes, species, chemicals, diseases and mutations are retrieved from the paper. The data retrieved are in the following format:

- Genes: Gene IDs e.g: 246759
- Chemical: MeSH unique id e.g: CHEBI:53063, D014302
- Species: NCBI Taxonomy e.g: 10116
- Disease: MEDIC Disease vocabulary e.g: D005355
- Mutation: (unknown for now)

The web service returns the response in BioC which is an xml format. The BioC format is slower to parse but has all the relevant information required. The REST API can be implemented in Java by using a HttpURLConnection object to make the network call and the response can be received by a BufferedReader.

UniProt Web Service - UniProt provides a web services to convert database identifiers. It supports several databases including: UniProt, Sequences Databases, Protein-Protein Interaction databases, Chemistry, Genome annotation databases, etc. It stores the identifier mappings between the databases. A web service can be written in Java to convert the identifier from one format to another.
For example: identifier can be converted from **UniProtKB AC** from the UniProt database to **RefSeq Protein** from the Sequence database.
UniProt also provides a web service to search for entries from their database. The data set name and an identifier is required as input.

The result is can be returned in xml, txt, rdf, fasta, gff format. The fasta format returns only the sequence of the data queried. The xml and txt format returns all the relevant information (full entries) from the database. The rdf file is the Resource Description Framework of the data queried. The gff file is the gene finding format (tab separated file). The xml is the most appropriate format to parse the result because it contains most information about the protein queried. Some information that are returned by a protein queried on UniProt is listed below:

1. Accession
2. Recommended name of the protein
3. Alternative names of the protein
4. Lineage (Taxon)
5. Taxonomy and taxonomy id
6. General comments about the protein activities
7. Database references
8. GO terms and GO ids
9. Feature Type
10. Sequence information
11. Relevant research papers

The database contains all the GeneIDs from the genome files. The Gene IDs is converted from P_ENTREZGENEID (GeneID) to its corresponding UniProtKB ID using the UniProt web service.
The UniProtKB ID is then used on the second webservice to retrieve the full entry. The full entry contains all the information about the gene, including the relevant research papers about the gene.
The response is in xml format which contains all the information of the Gene. An xml parser is then used to read the response and extract the important parts of the xml and display them in the web application.

Chapter 5 – Project Progress

In the last eight months, the stand-alone version of the data warehouse that was developed in the first phase has been partly converted into a web application using Java Vaadin framework. The web application can access the pre-populated data warehouse which has 854 genomes of infection-causing bacterial species, along with information on their infectivity. The existing java code has been used to access the database and a new user interface has been created using Vaadin framework while keeping the layout similar to the previous stand-alone application.

An intensive literature review has been carried out on nucleotide and protein phylogenetic analysis (Salemi, M., Vandamme, A.-M., and Lemey, P., 2009). Several software packages have been tested for pairwise distance, maximum likelihood and parsimony phylogenetic inference. The most appropriate software package was implemented for the each type of phylogenetic inference in the web application. A data mining component has also been implemented. The data mining (Chih-Hsuan W., Robert L., Zhiyong L., 2016) feature from NCBI is accessed as a web service and the results are displayed in the web application.

5.1 Initial Screen of the IDEAS Web Application

Users have the options of choosing genomes/strains:

- from the whole genome set loaded into the data warehouse (total 854 strains currently), listed using the genus and species names
- based on infectivity of species i.e. whether they infect humans, animals and/or plants.

Figures 20(a), 20(b) and 20(c) demonstrate the above options.

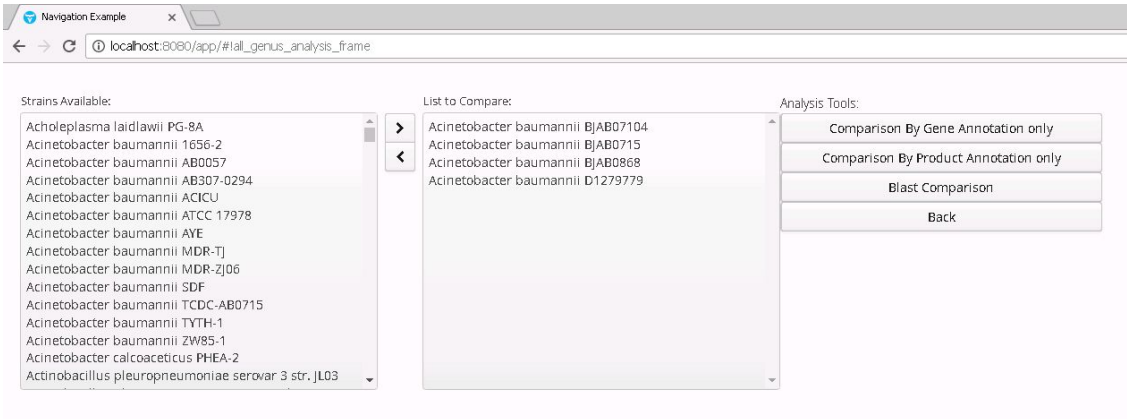


Figure 20(a) Choosing strains from whole genome set.

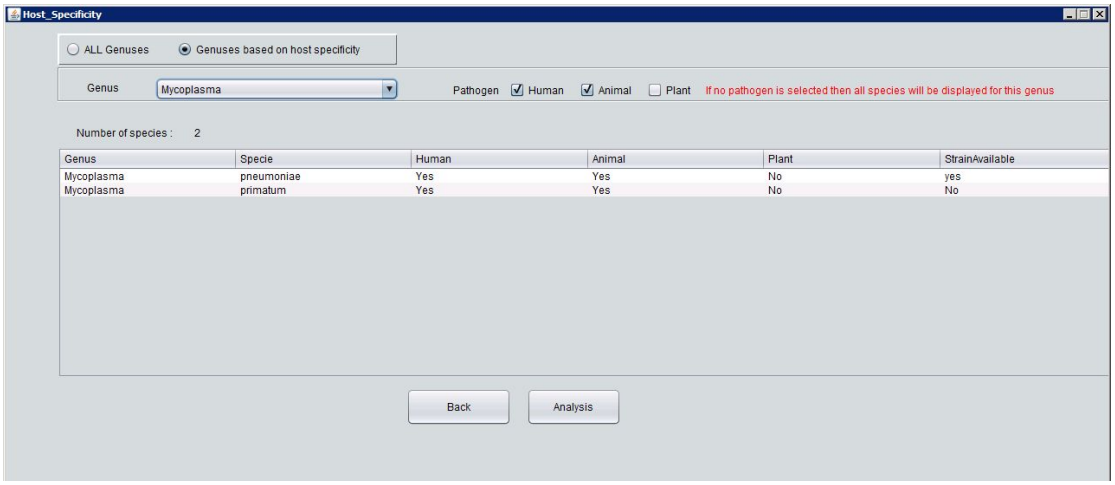


Figure 20(b) Choosing genomes based on the infectivity of strains, e.g. choosing the genus Mycoplasma that infects humans and animals.

Suppose the user chooses Pneumoniae from the Mycoplasma genus that infects both human and animal, because strains are available in the data warehouse for this specie, four strains are displayed as shown in figure 20(c).

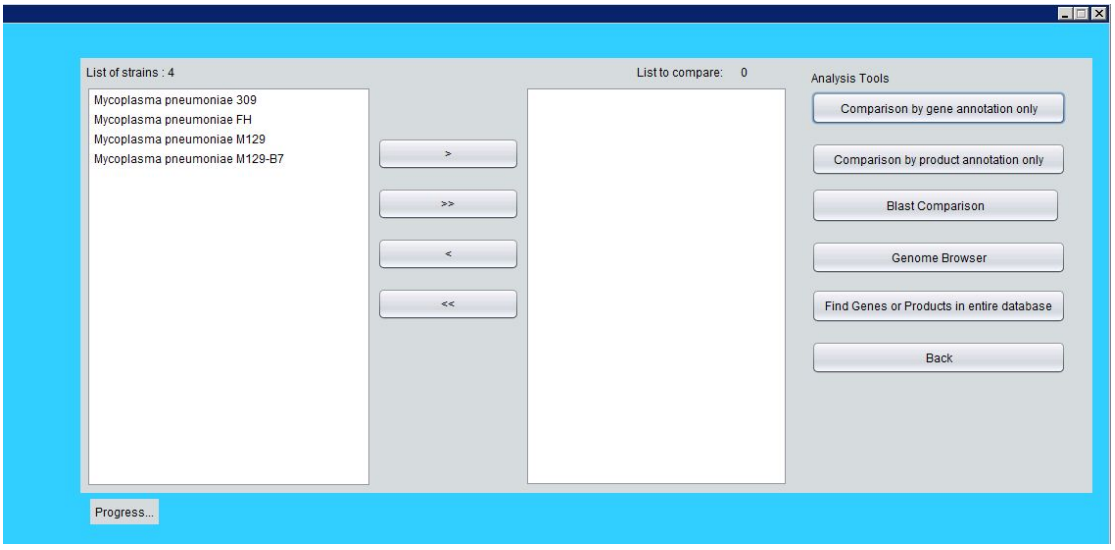


Figure 20(c) A list of strains from Mycoplasma Pneumoniae is displayed for user to choose from.

5.2 Search for common genes

Once a user has chosen the genomes/strains that s/he wishes to proceed for further analysis, the first step consists of comparing these strains. We have currently provided 3 methods of comparison, namely comparison based on gene name, comparison based on product name and comparison based on protein sequences using local blast application.

It is important to mention that the number of coding sequences (equivalent to expressed proteins) in the chosen four strains are as follows:

- Mycoplasma Pneumoniae 309: 707
- Mycoplasma Pneumoniae FH: 629
- Mycoplasma Pneumoniae M129: 648

- Mycoplasma Pneumoniae M129-B7: 612

Figures 21(a), 21(b) and 21(c) demonstrate the results of choosing each method of comparison.

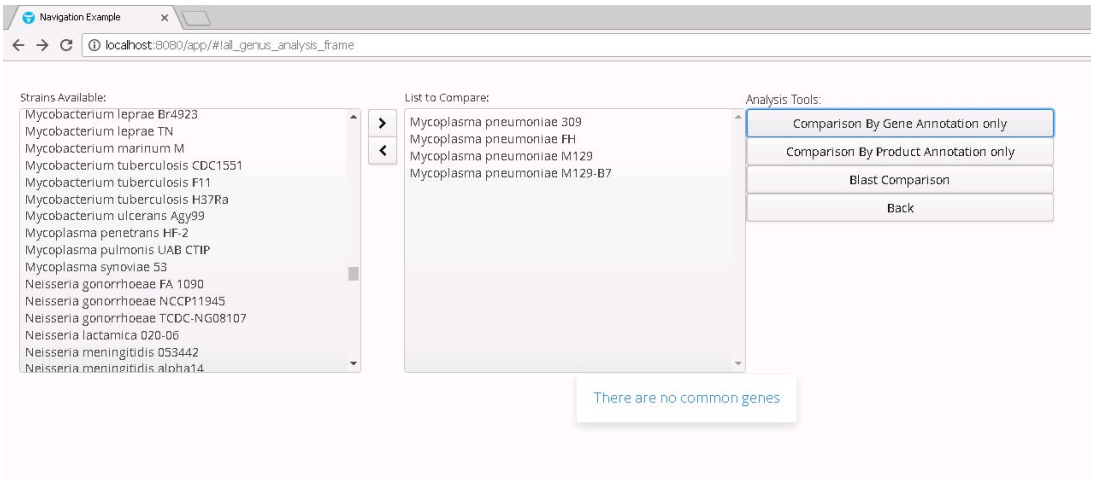


Figure 21(a) Comparison based on gene name

When comparison is performed using gene name, no common feature is found since all the strains are not sequenced from the same place and thus are not annotated using the same gene names.

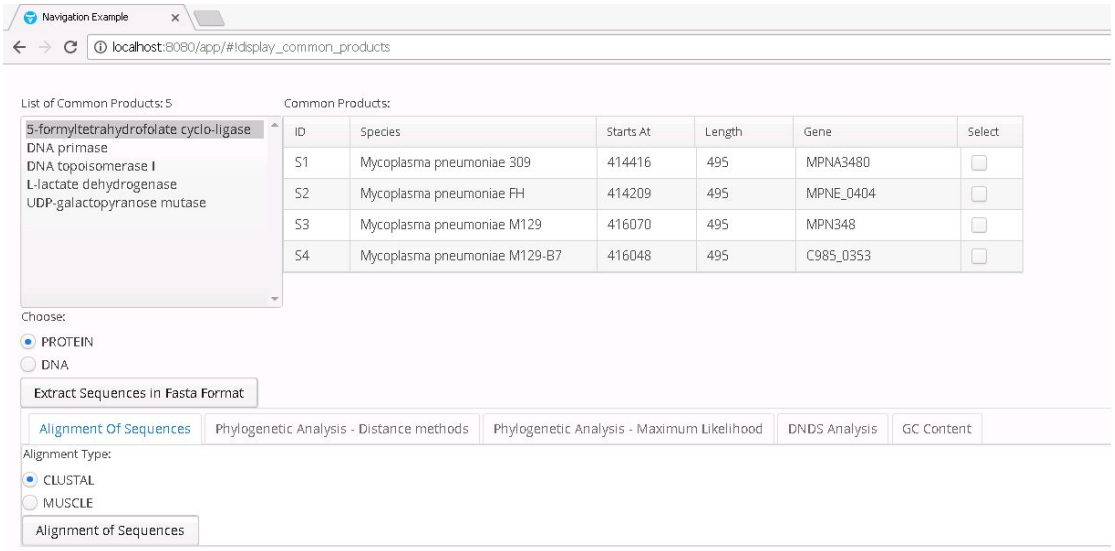


Figure 21(b) Comparison based on product name

When comparison is performed using product name, five (5) common features are found, which is still a very poor result as all the strains are from the same species. Again the product names may not have been annotated consistently.

For the comparison based on protein sequences using local blast application, user can use default parameters provided for blast or can change them to new values (figure 21(c) (i)).

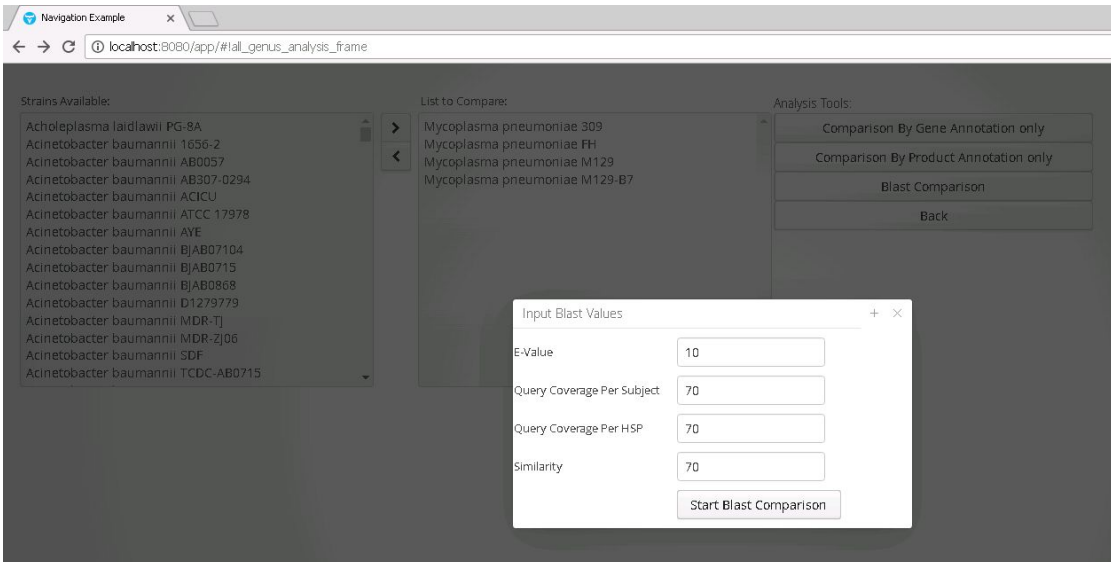


Figure 21(c) (i) choosing the comparison based on sequences using local blast

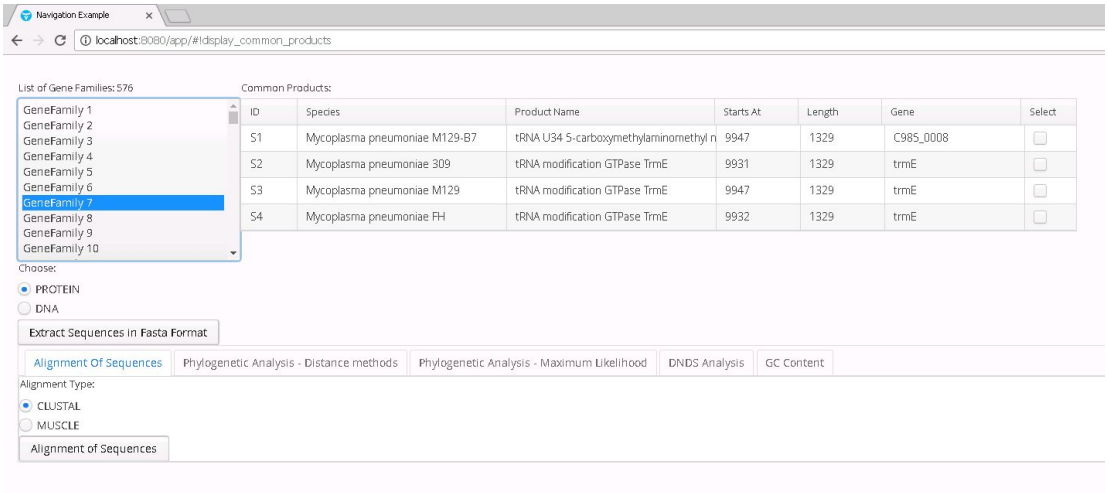


Figure 21(c) (ii) Comparison based on protein sequences using local blast

When comparison was performed using blast (on protein sequences) with default parameters, 576 features were found to be common among the four (4) chosen strains of *Mycoplasma Pneumoniae*. Since the number of coding sequences range from 612 to 707 for the four (4) chosen strains, this is definitely a much better result compared to the previous two (2) methods of comparison as all four strains belong to the same genus and specie, though sequenced from different places.

5.3 Sample Analysis on one specific gene family (gene family 7) common in all four (4) chosen genomes

5.3.1 Multiple Sequence Alignment of the sequences of gene family 7

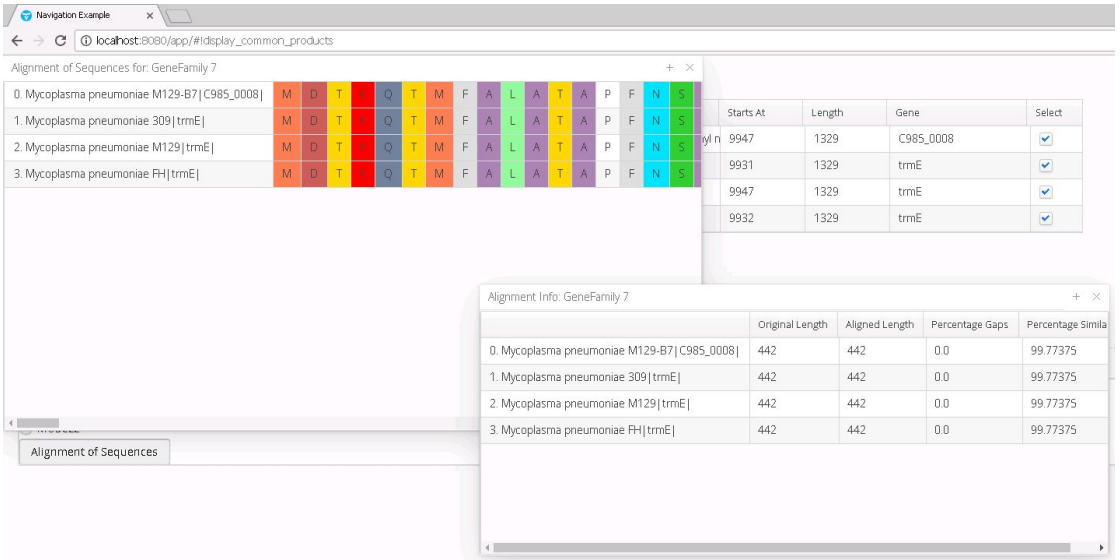


Figure 22 Results of Clustal Multiple Sequence Alignment using protein sequences of gene family 7

When the clustal multiple sequence alignment is performed using the protein sequences of the gene family 7 (one from each strain), they seem to be very similar (above 99%) (figure 22), though they are not named similarly in all strains.

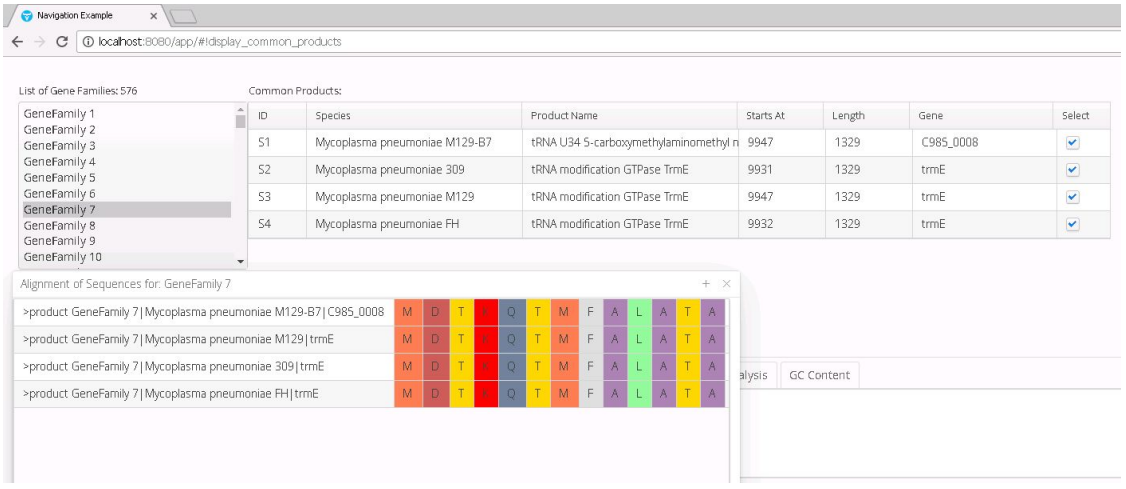


Figure 23 Results of Muscle Multiple Sequence Alignment using protein sequences of gene family 7

The user has the option to run the muscle multiple sequence alignment which is an iteratively refined alignment. Muscle alignment (figure 23) may take more time to process but it is more accurate than clustal alignment.

5.3.2 Phylogenetic analysis of sequences from gene family 7

5.3.2.1 Pairwise distance method without bootstrap

Distance analysis compares two aligned sequences at a time and builds a matrix of all possible sequence pairs. The Matrix provides an idea of how similar or different each sequence is from the other. For each pairwise comparison, the number of base insertion, deletion and substitutions are counted and presented as a proportion of the overall sequence length. The sequences are then arranged to a tree according to their distances. In the web application, pairwise distance phylogenetic analysis can be performed using different substitution models and different tree building algorithms. Each algorithm may create a different phylogenetic tree. The list of parameters available in the web application is as follows:

DNA substitution models:

1. F84
2. Kimura
3. Jukes Cantor
4. LogDet

Protein substitution models:

1. Jones Taylor Thornton matrix
2. Henikoff Tillier PBM matrix
3. Dayhoff PAN matrix
4. Kimura Formula
5. Categories Model

Tree building algorithms:

1. Neighbor Joining
2. UPGMA
3. Fitch Margoliash
4. Minimum Evolution

A UPGMA tree is built using Jukes Cantor distance matrix without bootstrap:

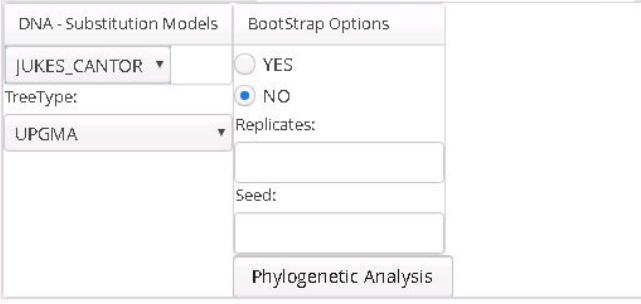


Figure 24 Input parameters for building UPGMA tree

The resulting tree is shown below:

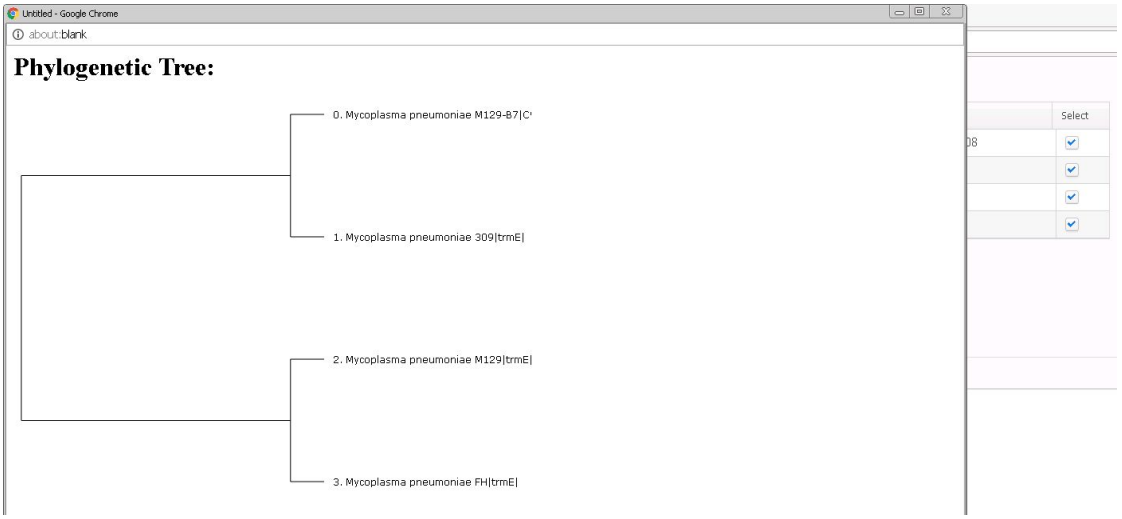


Figure 25 UPGMA tree using Jukes Cantor substitution model of the gene family 7

The phylogenetic analysis of the dna sequences of the gene family 7 for the four (4) strains of Mycoplasma pneumoniae show that the strains M129 and FH are the close and strains M129-B7 and 309 are close (figure 25).

5.3.2.2 Pairwise distance method with bootstrap

A Minimum Evolution tree can be built using Jones Taylor Thornton distance matrix with 200 bootstrap replicates, as shown in figure 26:

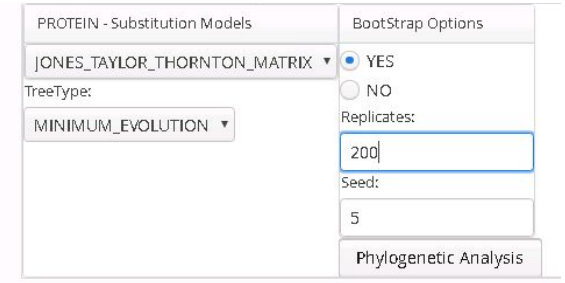


Figure 26 Input parameters for building Minimum evolution tree with bootstrap

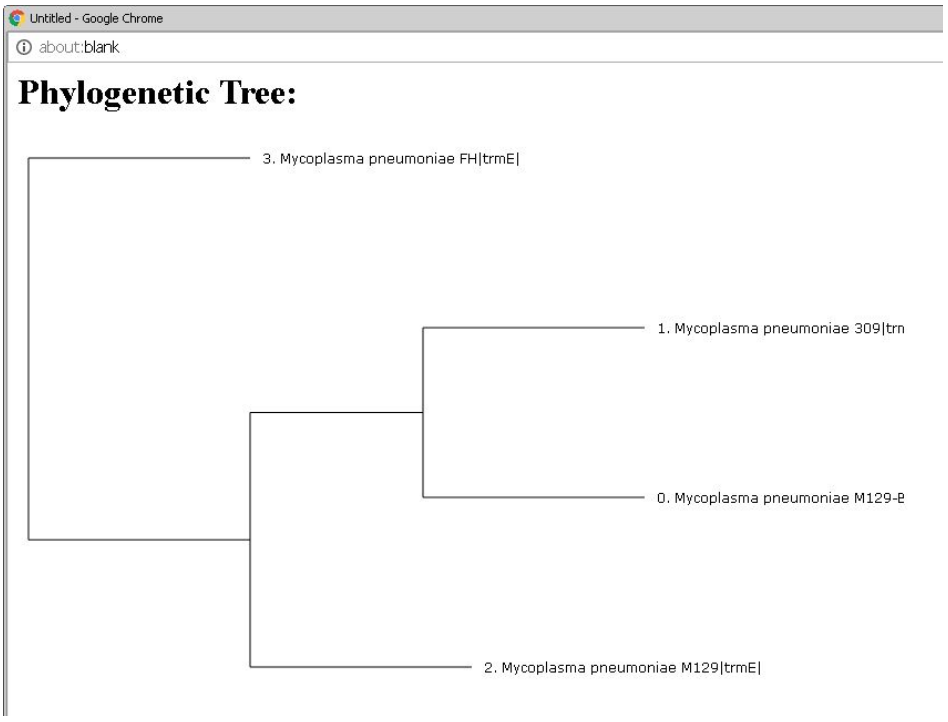


Figure 27 Minimum evolution tree using Jones Taylor Thornton substitution model with 200 bootstrap replicates

The phylogenetic analysis of the protein sequences of the gene family 7 for the four (4) strains of Mycoplasma pneumoniae show that the strains 309 and M129-B7 are the closest (figure 27).

5.3.2.3 Maximum-Likelihood method

The Maximum Likelihood method provides probabilities of the sequences using a model of their evolution on a particular tree. The tree having the highest probability value (likelihood) given a specific model of evolution is preferred. This method is computationally intense because all possible trees are considered. In the web application, the Maximum-Likelihood tree can be built using different substitution models and rate heterogeneity. Each algorithm may create a different phylogenetic tree. The parameters available is listed as follows:
Amino Substitution models:

1. WAG (Whelan-Goldman 2000)
2. JTT (Jones et al. 1992)
3. VT (Mueller-Vingon 2000)
4. mtREV24 (Adachi-Hasegawa 1996)
5. rtREV (Dimmic et al. 2001)
6. BLOSUM62 (Henikoff-Henikoff 1992)
7. Dayhoff (Dayhoff et al. 1978)

DNA Substitution models:

1. HKY85 (Hasegawa et al. 1985)
2. TN93 (Tamura-Nei 1993)
3. General Time Reversible

Rate Heterogeneity:

1. Uniform Rate
2. Gamma Distributed Rates
3. Site Specific Substitution Rates

A maximum likelihood tree can be built with the four protein sequences using the WAG (Whelan-Goldman 2000) substitution model and uniform rate heterogeneity.

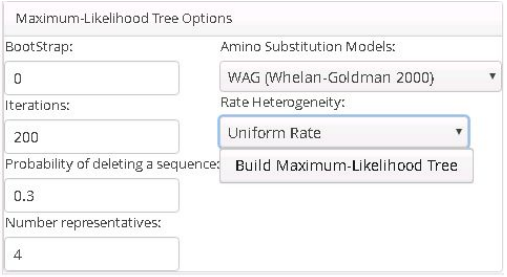


Figure 28 Input parameters for building maximum likelihood tree

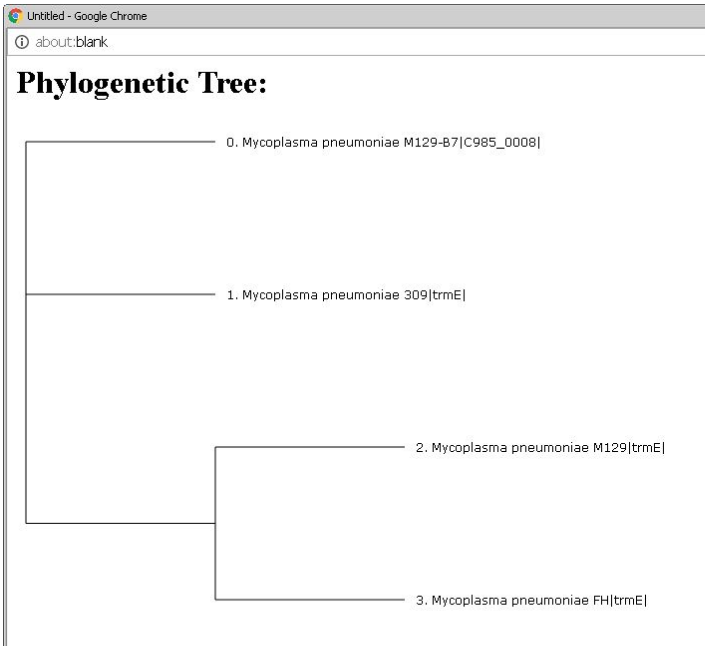


Figure 29 Maximum-Likelihood tree

The phylogenetic analysis of the protein sequences of the gene family 7 for the four (4) strains of Mycoplasma pneumoniae show that the strains FH and M129 are the closest (figure 29).

5.3.2.4 Parsimony Analysis to estimate phylogenetic trees

Phylogenetic analysis can also be performed using the parsimony principle to study the evolution of the sequences. The most parsimonious tree is the phylogeny that requires the fewest necessary changes to explain the difference

among the sequences. Phylogenetic analysis can be performed using the following parameters:

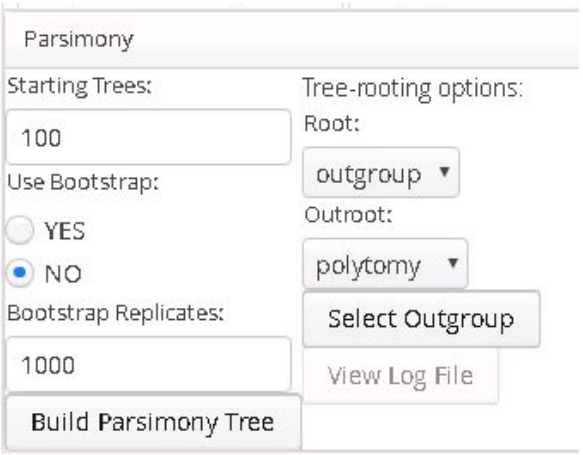


Figure 30 Parsimony parameters

Tree rooting options:

Root: outgroup, midpoint, lundberg

Outroot: polytomy, paraphyl, monophyl

The user also has the option to select an outgroup that will be used as a reference to root the tree when "outgroup" is selected as the root method. The following window is displayed to allow the user to select the outgroup:

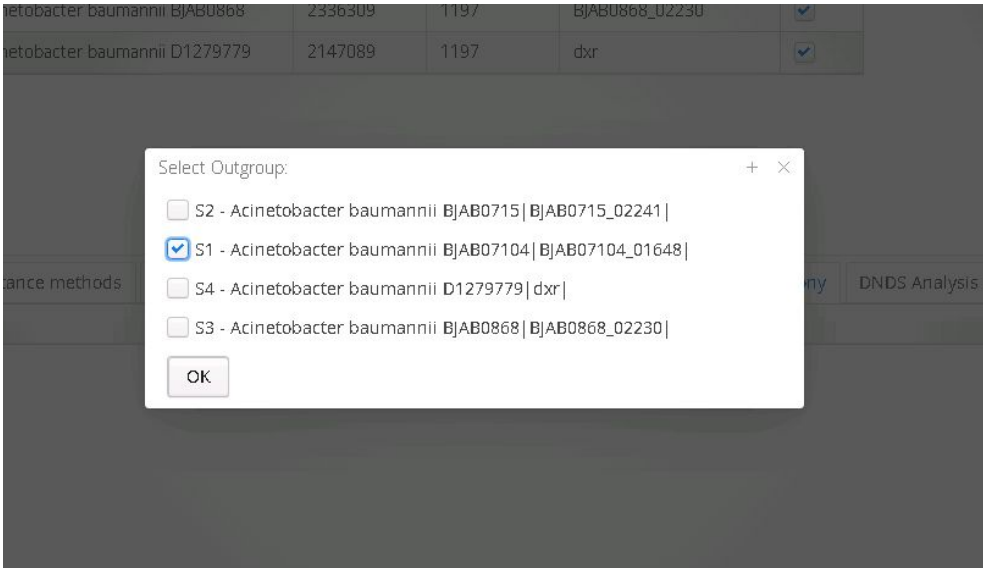


Figure 31 Selecting outgroup for parsimony analysis

The phylogenetic tree is then constructed with the four above sequences, using sequence S1 as the outgroup, 100 starting trees without bootstrap:

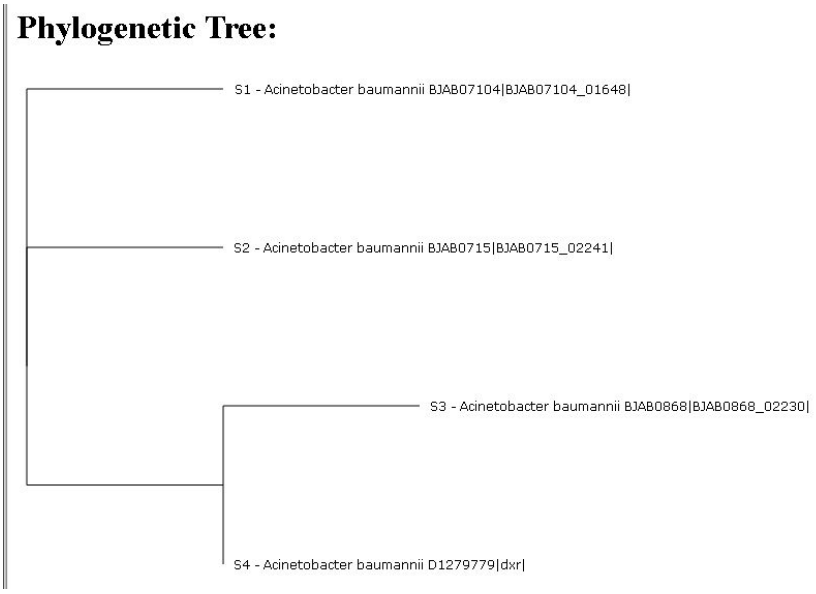


Figure 32 Parsimony tree without bootstrap

The phylogenetic tree is then constructed using the above 4 sequences, sequence S1 as the outgroup, 100 starting trees with 1000 bootstrap replicates:

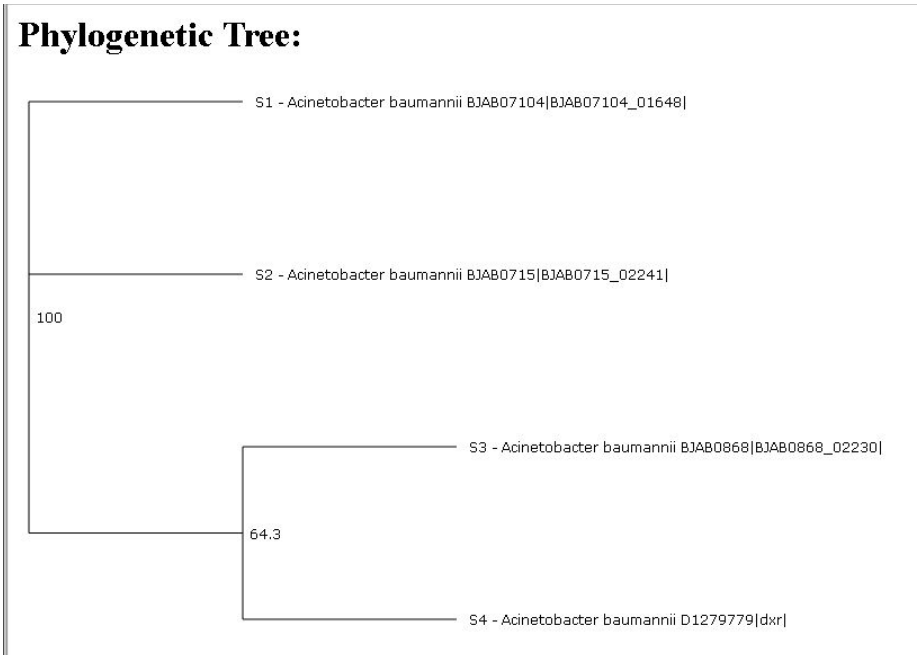


Figure 33 Parsimony tree with bootstrap

One additional feature of the web application is that it allows the user to consult the log file for some more extensive details of the analysis.

```
Log output:
 Heuristic search settings:
   Optimality criterion = parsimony
     Character-status summary:
       Of 398 total characters:
         All characters are of type 'unord'
         All characters have equal weight
         394 characters are constant
         3 variable characters are parsimony-uninformative
         Number of parsimony-informative characters = 1
     Gaps are treated as "missing"
   Starting tree(s) obtained via stepwise addition
     Addition sequence: random
     Number of replicates = 100
     Starting seed = generated automatically
     Number of trees held at each step = 1
   Branch-swapping algorithm: tree-bisection-reconnection (TBR) with reconnection limit = 8
     Steepest descent option not in effect
   Initial 'Maxtrees' setting = 100 (will be auto-increased by 100)
   Branches collapsed (creating polytomies) if maximum branch length is zero
   'MulTrees' option in effect
   No topological constraints in effect
   Trees are unrooted

 Heuristic search completed
   Total number of rearrangements tried = 2
   Score of best tree(s) found = 4
   Number of trees retained = 1
   Time used = 0.00 sec (CPU time = 0.00 sec)

 Tree-island profile:
                      First      Last                        First    Times
 Island      Size      tree      tree           Score    replicate      hit
 ---------------------------------------------------------------------------
      1         1         1         1               4            1      100
```

Figure 34 parsimony log file

5.3.3 dN/dS analysis of sequences from gene family 7

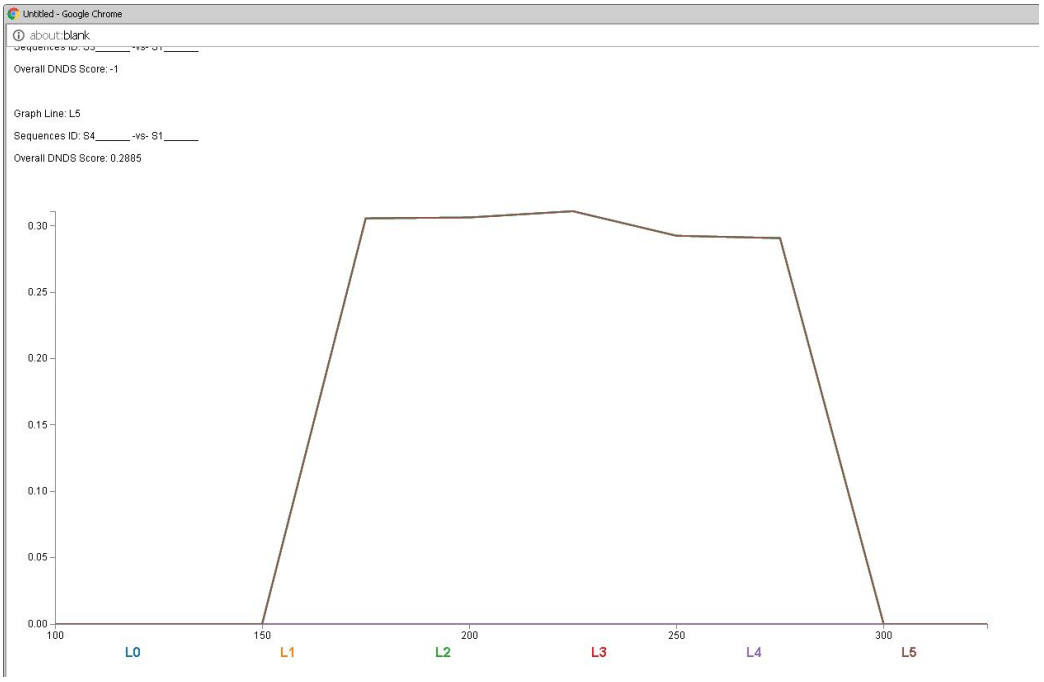


Figure 35 dN/dS analysis of the sequences of gene family 7

The dN/dS analysis of the sequences from the gene family 7 shows the ratio of the non-synonymous to the synonymous mutations in the form of an evolution graph (figure 29).

5.5.4 GC count variation of sequences from gene family 7

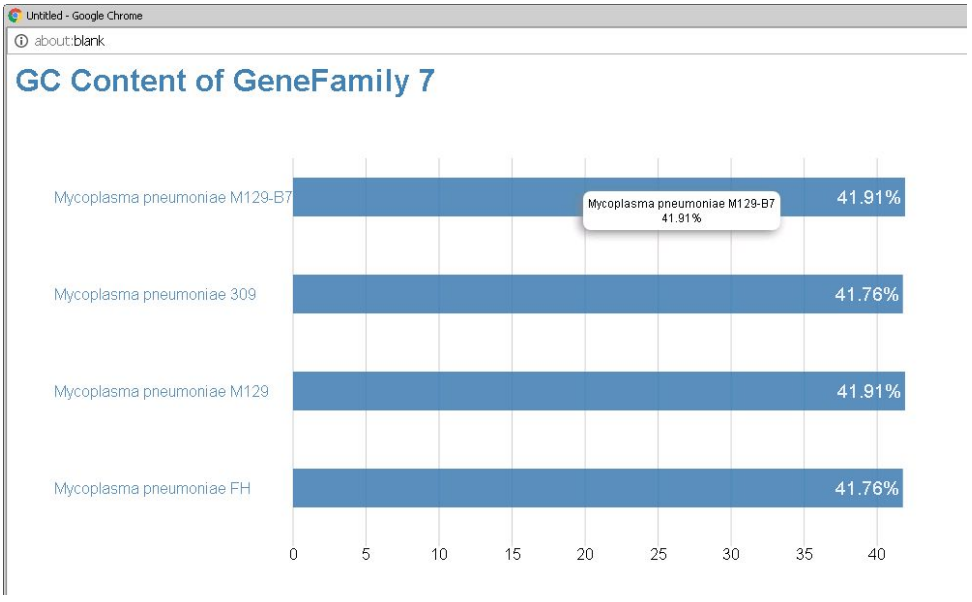


Figure 36 GC count distribution of sequences from gene family 7

The GC content distribution of the four (4) strains is as follows (figure 30):

- Mycoplasma Pneumoniae 309: 41.76%
- Mycoplasma Pneumoniae FH: 41.76%

- Mycoplasma Pneumoniae M129: 41.91%
- Mycoplasma Pneumoniae M129-B7: 41.91%

The results are as expected, i.e. all four strains have almost the same GC content since all the four strains belong to same genus and specie.

6. Text Mining

6.1 Extract important information from research papers

When the PubMed id 23819905 is entered, the genes, diseases, chemicals, species and mutations referenced in the research paper are displayed:

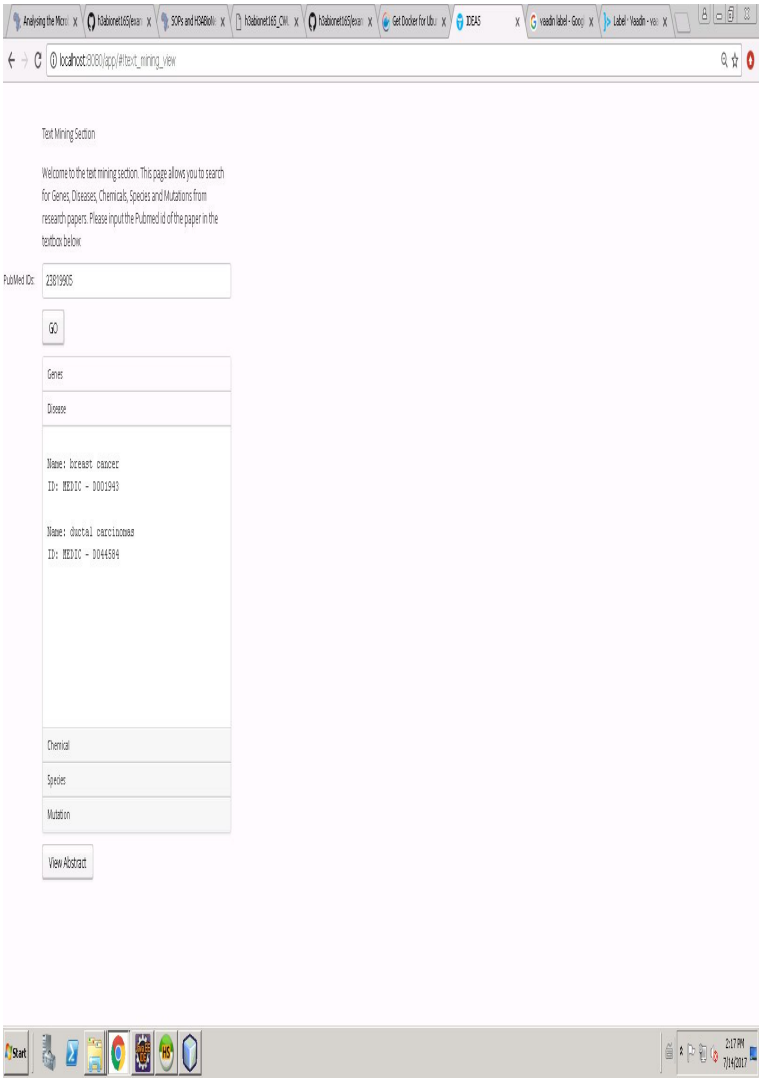


Figure 37 Diseases referenced in the research paper with PubMed id 23819905

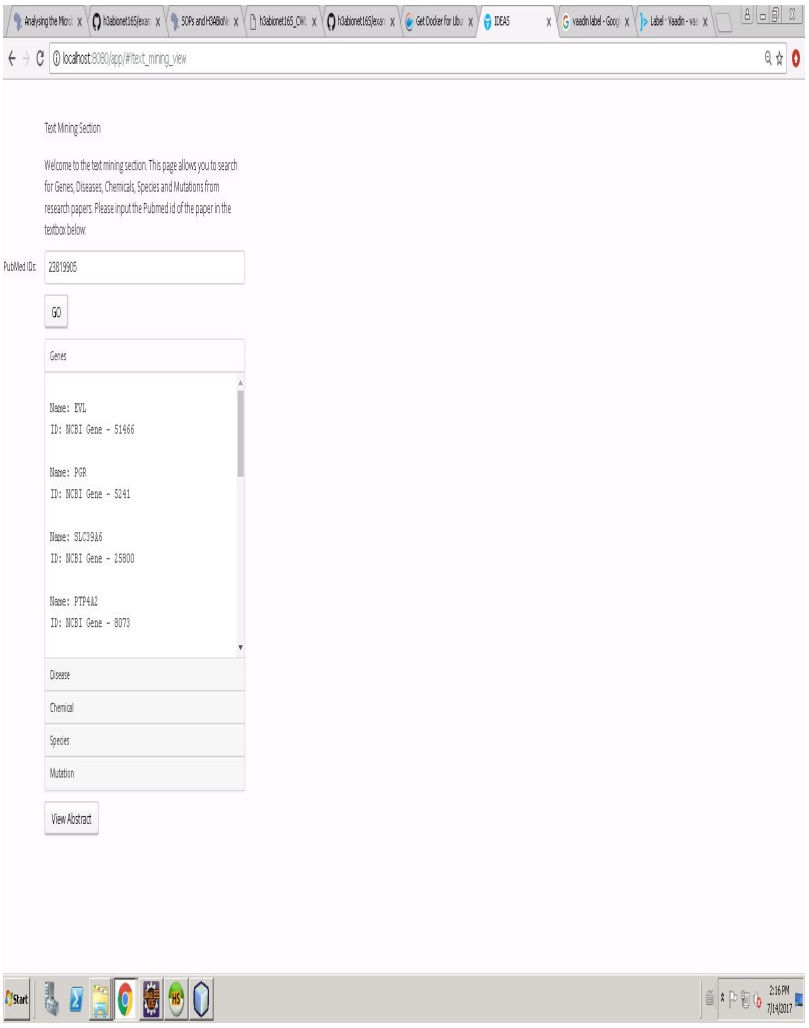


Figure 38 Genes referenced in the research paper with PubMed id 23819905

The name and id of the gene is displayed. The user can also read the abstract of the research paper.

6.2 Search for research papers

One gene from Mycoplasma pneumonia M129 and one gene from Mycoplasma pneumonia M129-B7 have been selected. The information about the relevant research papers are displayed in a new window:

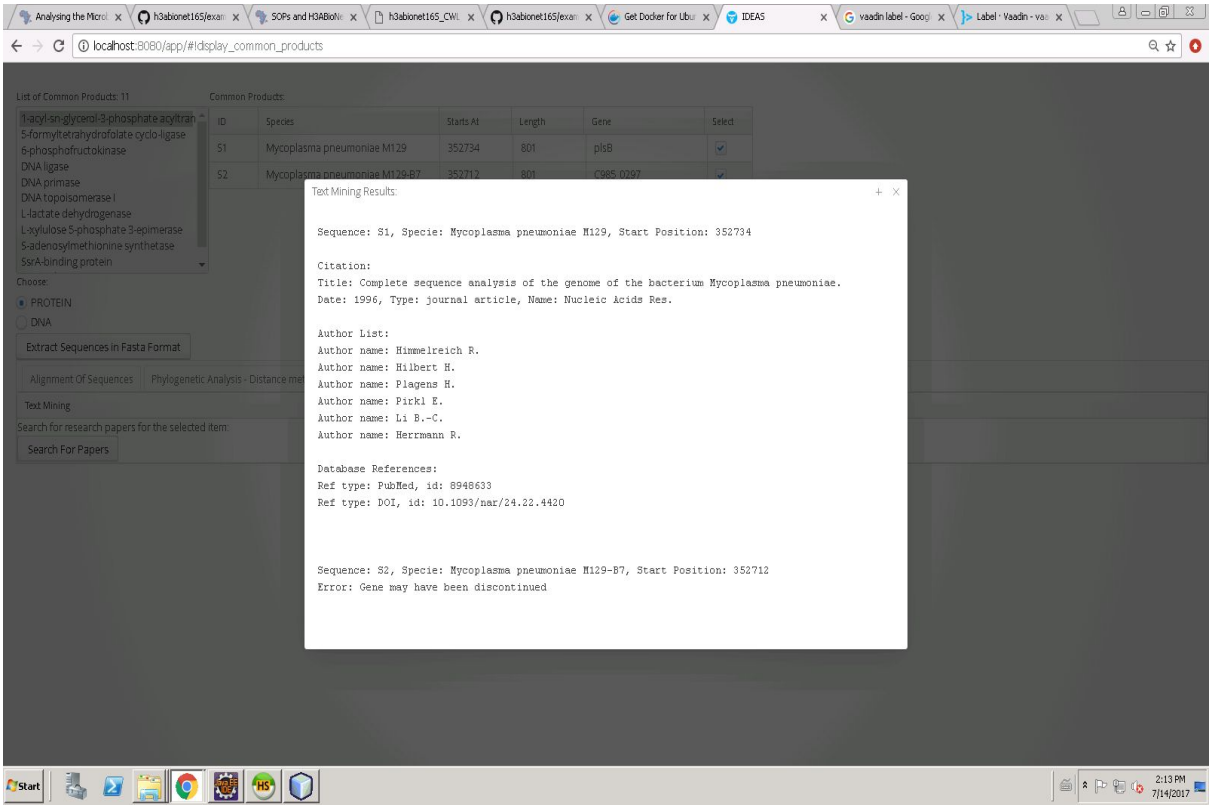


Figure 39 Research papers that cited the genes selected

The title of the research paper, the author list and database references are displayed. The user can then use the PubMed id to search for the research paper.

7. Hosting the web application on the University Intranet

A separate Tomcat Server had to be installed on the virtual machine to host the web application on the university intranet. This will allow academics who have access to the university network to access the web application using a browser. There is already one instance of tomcat that runs on the server. That instance is located in the "C:\apache-tomcat-9.0.0.M17" folder and is used to test the web application during the development phase.

Setting Up the Tomcat Server on the Windows Virtual Machine

The Tomcat Server v9.0 installation was downloaded from the link: **https://tomcat.apache.org/download-90.cgi** and installed in the "TC9-Server" folder in Libraries\Documents. The rest of the server set up is described as follows:

Settings:

1. The server is set to listen to port “9090” in the server.xml file in the “conf” folder. Port 8080 is already in use for testing in the eclipse IDE. Windows firewall is also set to allow Port 9090.

```
-->
<Connector connectionTimeout="20000" port="9090" protocol="HTTP/1.1" redirectPort="8443" />
<!-- A "Connector" using the shared thread pool-->
<!--
```

2. The username and password are also set in the tomcat-users.xml file. (username and password are both admin for the tomcat server)

```
<role rolename="manager-status"/>
<role rolename="manager-script"/>
<role rolename="manager-gui"/>
<user username="admin" password="admin" roles="manager-status,manager-script,manager-gui"/>
```

3. reloadable=true for context for testing/debugging purposes in context.xml file

```
--><!-- The contents of this file will be loaded for each web application --><Context reloadable="true">

    <!-- Default set of monitored resources. If one of these changes, the    -->
    <!-- web application will be reloaded.                                   -->
    <WatchedResource>WEB-INF/web.xml</WatchedResource>
    <WatchedResource>${catalina.base}/conf/web.xml</WatchedResource>

    <!-- Uncomment this to disable session persistence across Tomcat restarts -->
    <!--
    <Manager pathname="" />
    -->
</Context>
```

4. set listing to true in web.xml for testing/debugging purposes

```
<servlet>
    <servlet-name>default</servlet-name>
    <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
    <init-param>
        <param-name>debug</param-name>
        <param-value>0</param-value>
    </init-param>
    <init-param>
        <param-name>listings</param-name>
        <param-value>true</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
</servlet>
```

After the server has been set up:

1. The project was exported as a war file from the eclipse IDE. (optimized for Tomcat 9)
2. Copy the war file to the webapps folder in the “TC9-Server” folder

Open a command prompt (See Figure 40). Navigate to bin folder of Tomcat. Execute startup.

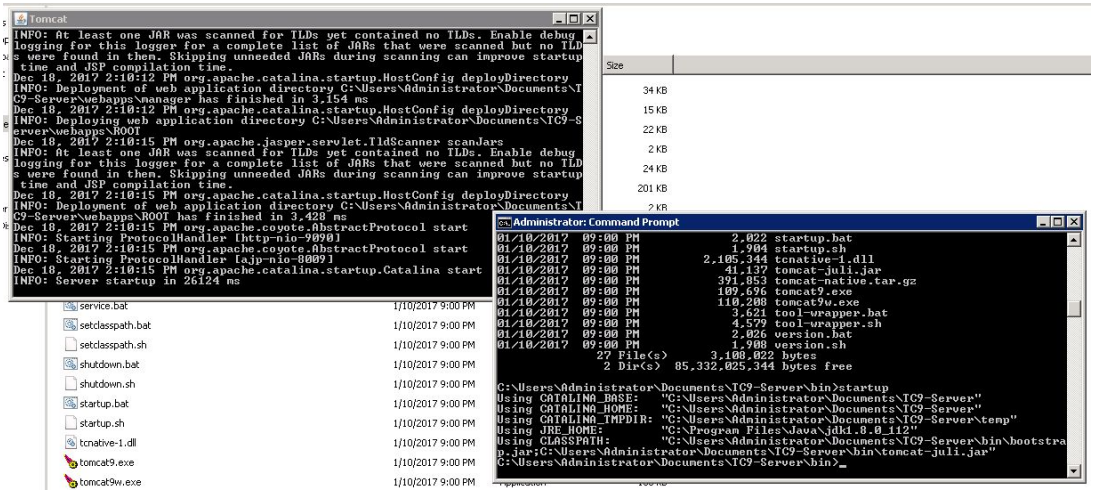


Figure 40 – Snapshot of Command Prompt for setting up Tomcat Server

3. The **IDEAS** web app was then deployed from the webapps folder.
4. The application can now be accessed with the ip "172.22.8.244:9090/ideas" (Figure 40)

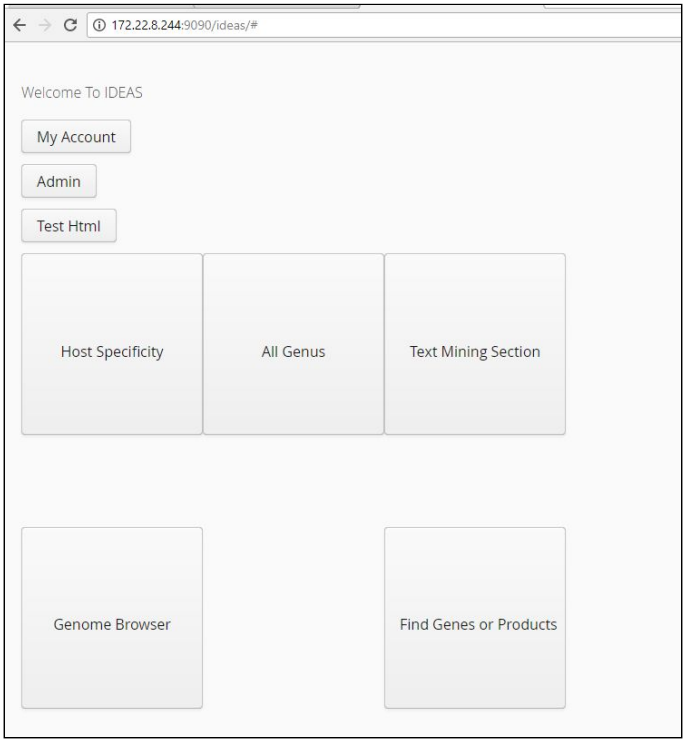


Figure 41 – Web access of IDEAS application

8. Updating genomes in the IDEAS database

The data warehouse contains genome files that are being constantly updated. If outdated genome files are used for analysis, they may generate errors indicating that the genome has been discontinued. A dashboard for

an Administrator has been created to monitor the Genome files in the data warehouse and update them accordingly. The administrator will log in to the system using a username and password. This will navigate him directly to the database management section.

New genome files are copied to the folder: C:\Users\Administrator\Desktop\genFile. The genome files are read from this folder and displayed in the database management section. The genomes that are already in the database are also displayed in that table. The Administrator can check the date of the genome files from the table and choose to update or delete a genome from the database.

The Administrator will be provided with a dashboard as shown in Figure 42.

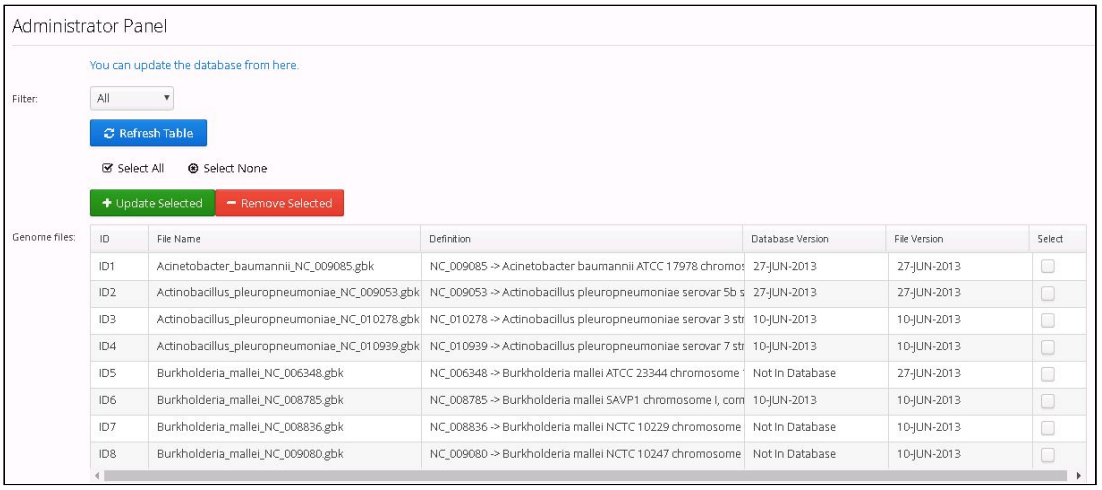


Figure 42 – Dashboard for genome update

Genome files can be selected for update by checking the combo box provided for each genome, in the last column (Figure 43).

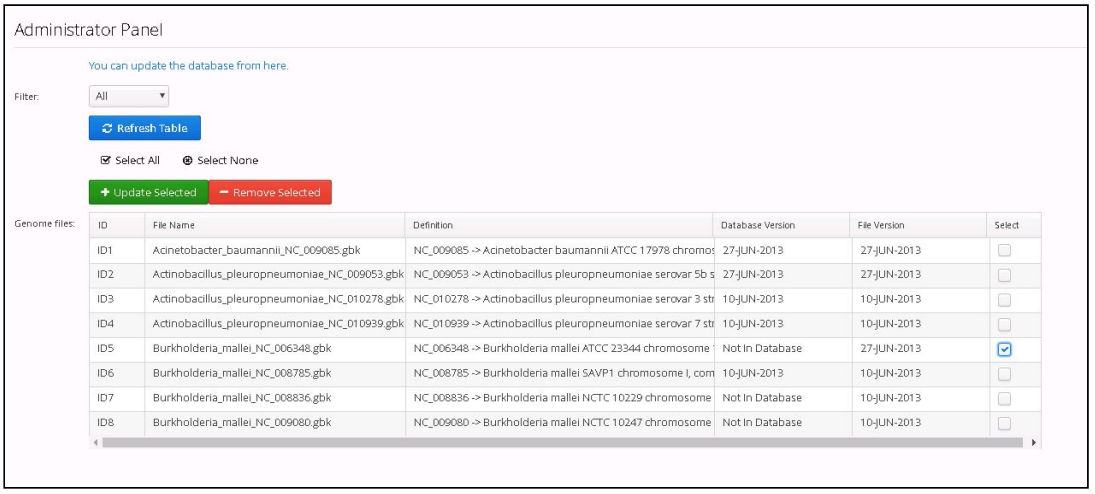


Figure 43 – Choosing a genome to update

Once the update is complete a notification is shown (see Figure 44).

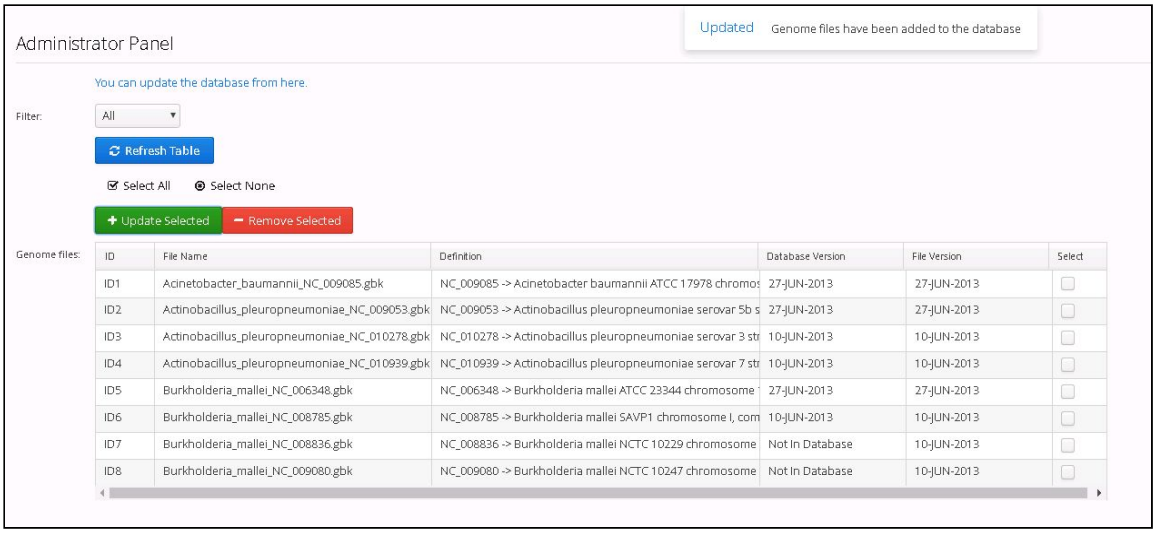


Figure 44 – Notification of a successful genome update

9. Creation of User Accounts for accessing the IDEAS application online

Login

The facilities for a user to log in IDEAS has been implemented. Users are asked to log in the first time they access the system. The users have to use their email address and a password (Figure 45).

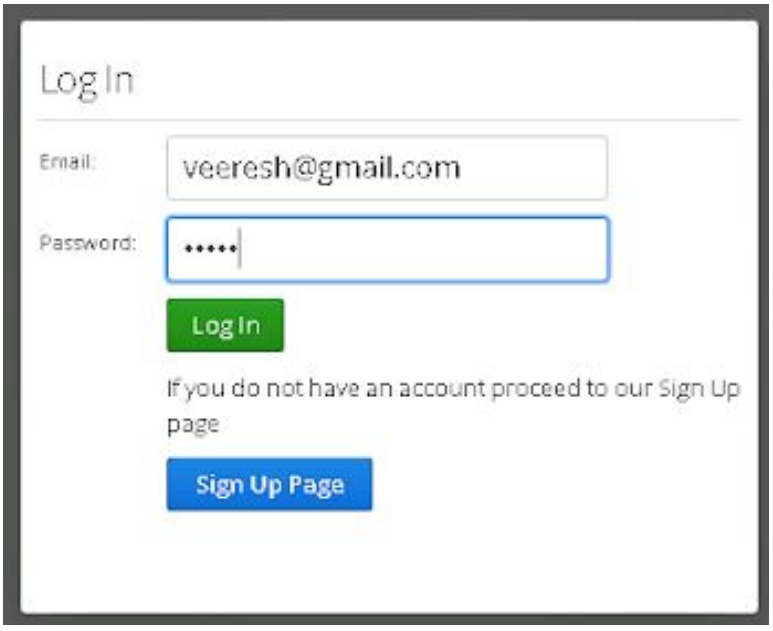


Figure 45 – Login to IDEAS application

Sign Up

If a user is not registered on the system, a new account can be created. This will require the username, email address, and password of the user (Figure 46).

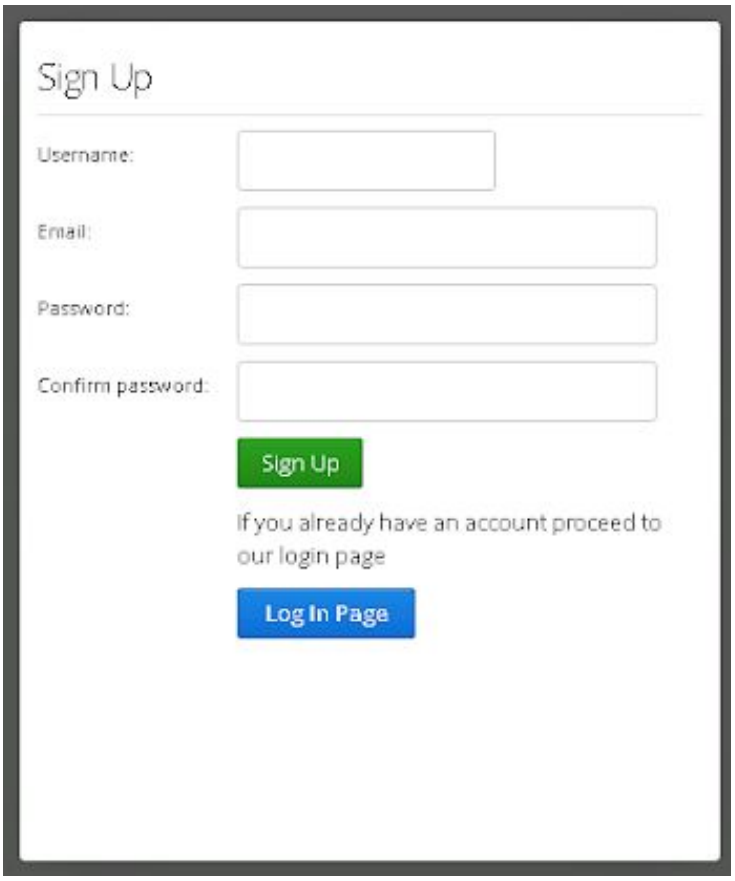


Figure 46 – Sign Up to IDEAS application

Database

The information about users is stored securely in a database. The ID is generated automatically and it is used to identify each user on the system. Directories are created on the server for each user using their corresponding user id. The user password is hashed using the SHA256 function. The date and time the user account has been created is stored in the system together with the last date and time the user was online.

The table **tbl_user** has been created with the following attributes: *ID, username, email, password, date_time_joined, date_time_last_online* (Figure 47).

ID	username	email	password	date_time_joined	date_time_last_online
3	Veeresh	veeresh@gmail.com	$2a$10$oSYZE4lvL1ST94dGIhCx2uW0GnmKw4NG/e1yq...	2018-03-19 11:04:21	2018-03-19 11:04:21

Figure 47 –Schema for tbl_user

Once the user has logged into the system, he can access all the features of the web application. Some analyses in the web application take a lot of time to run. Separate threads have been implemented to run these lengthy processes in the background so that the user doesn't have to wait for the analysis to complete on just one screen.

One example of such lengthy process is the blast comparison. The screenshots in Figures 48 - 51 demonstrate how the background process is functional in the web application. Once the sequences have been selected for the blast comparison, the "Start Blast Comparison" button can be selected to start the background thread. This will run the blast comparison in the background (on the server) and the user will be directed to the user dashboard.

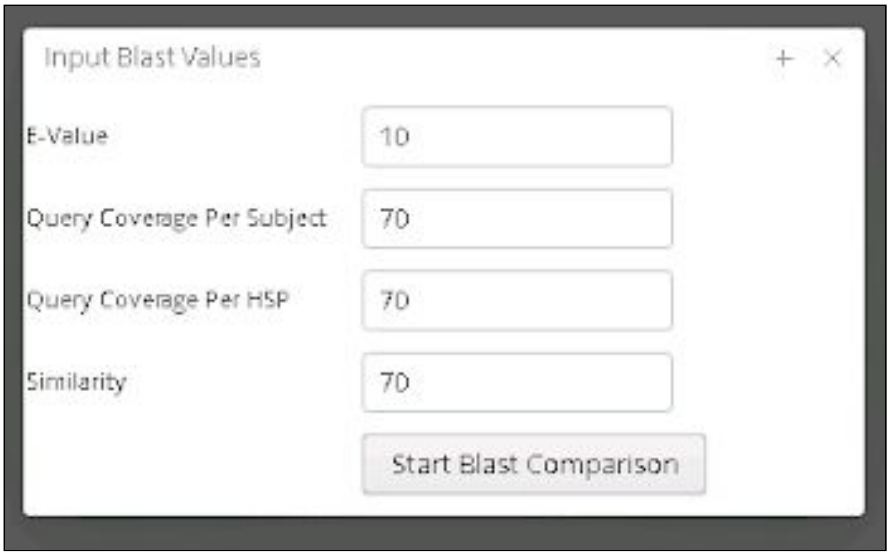


Figure 48 –Blast Operation

Thereafter the user is provided with a dashboard as in Figure 49. The status of the background process is set to "Running". The user can refresh this status or go back to the main menu and access other parts of the web application.

Figure 49 –Dashboard for Status of Blast operation

The user can navigate to the main menu and access other parts of the application:

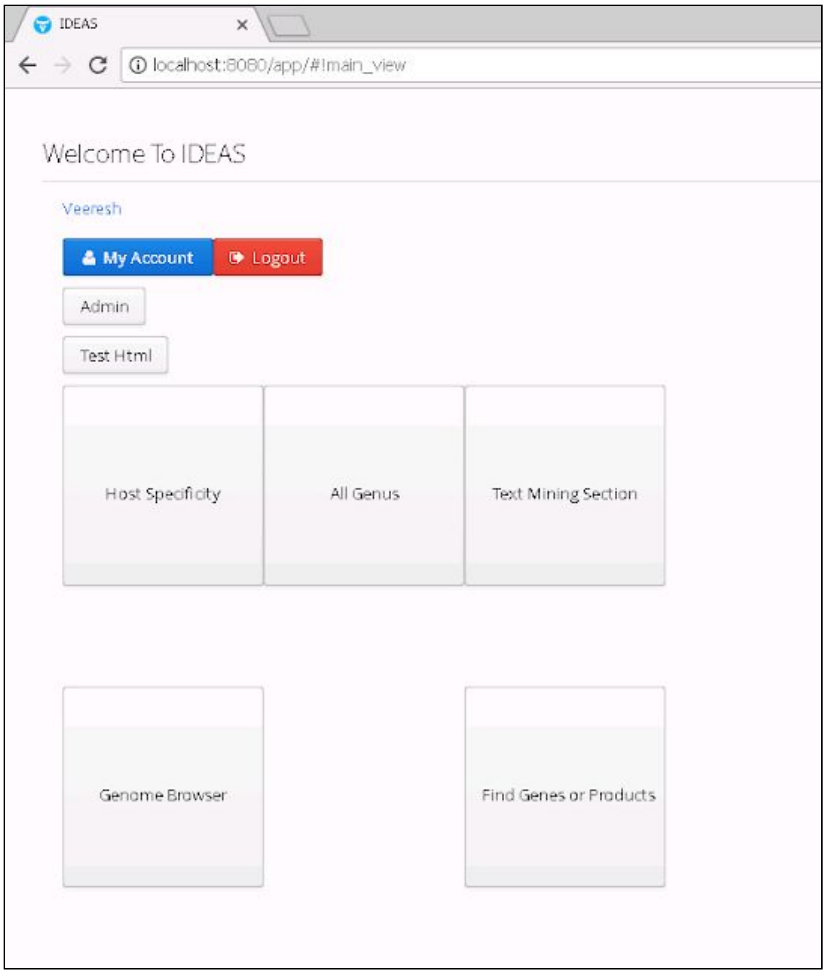


Figure 50 –Back to Main Menu while Blast operation is still running.

Once the analysis is complete, the status of the background thread is set to "Complete" in the user dashboard section and a notification is displayed on

the top of the screen, regardless of what page of the web application the user is on, to alert the user that the analysis is complete. The user can then click on the "Check Results" link displayed below to access the analysis results.

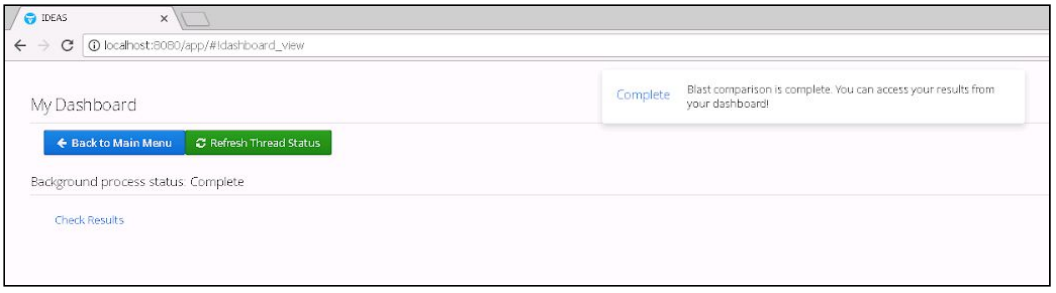


Figure 51 –Notification that Blast is complete.

The "**Check Result**" link redirects the user to the analysis section of the web app with all the results from the blast comparison (Figure 52).

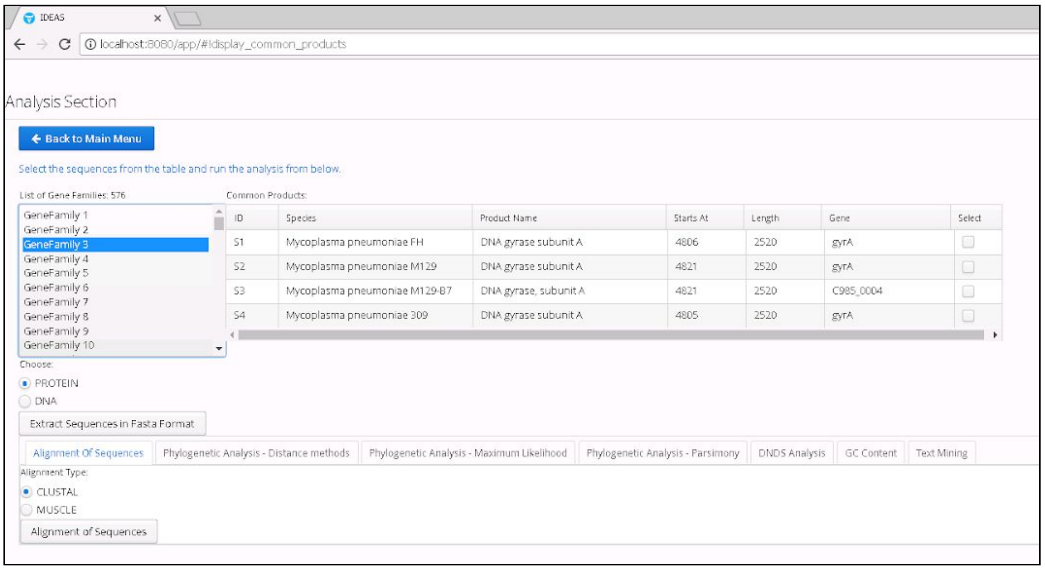


Figure 52 - Results of Blast Operation.

10. Advanced validation options for phylogenetic analysis

The complete set of parameters and options need to be implemented to make sure that the web application reflects all the functionalities of the phylogenetic packages used. These include additional parameters that weren't implemented during the previous milestone and some additional validations and user-friendly labels to make the application more robust and easier to use.

Tooltips have been implemented for the different parameters (Figure 53). Whenever the user hovers the mouse pointer over a textbox, a tooltip is shown to indicate the values that the text box accepts. This makes it easier for novice users who are not well versed in phylogenetic analysis.

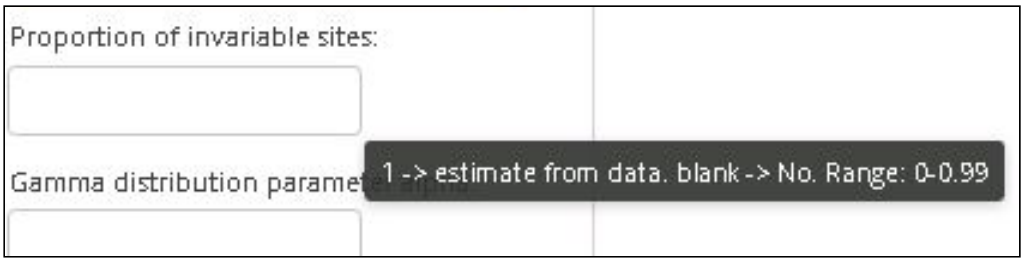


Figure 53 – Tooltips for phylogenetic analysis parameter options

When running a maximum likelihood analysis, experienced users have the ability to choose some advanced options when gamma distributed rates is selected. These parameters were not implemented during the last milestone. These additional parameters together with their corresponding validations have been implemented as follows:

- Three extra parameters are visible only when "Gamma Distributed Rates" is selected from the Rate Heterogeneity combo box (Figure 54).

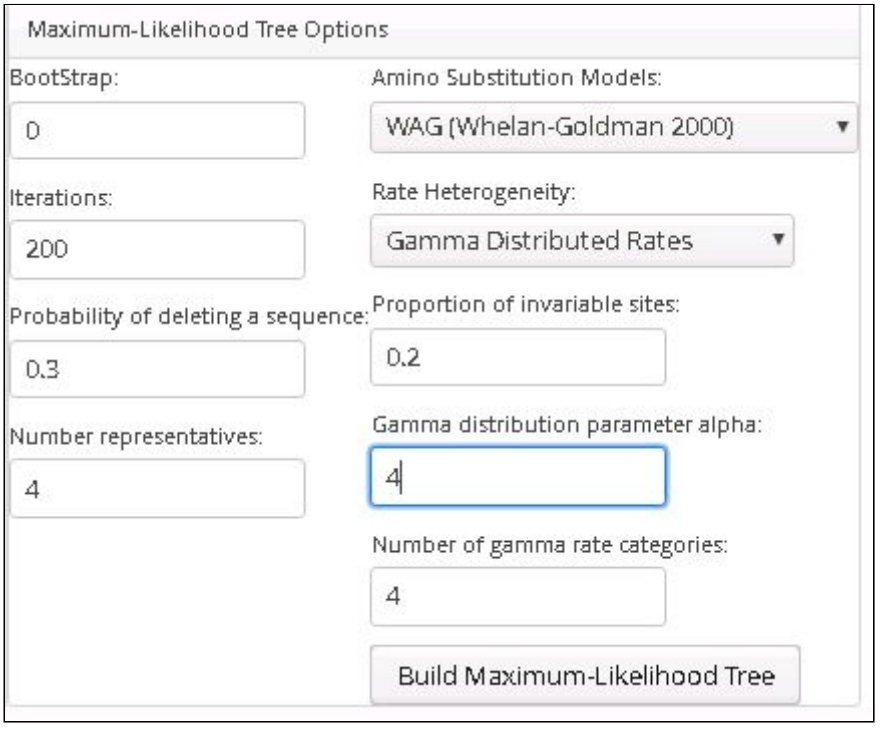


Figure 54 – Extra parameters for Gamma distributed rates

- Result after running a maximum-likelihood analysis with these parameters are shown in Figure 55.

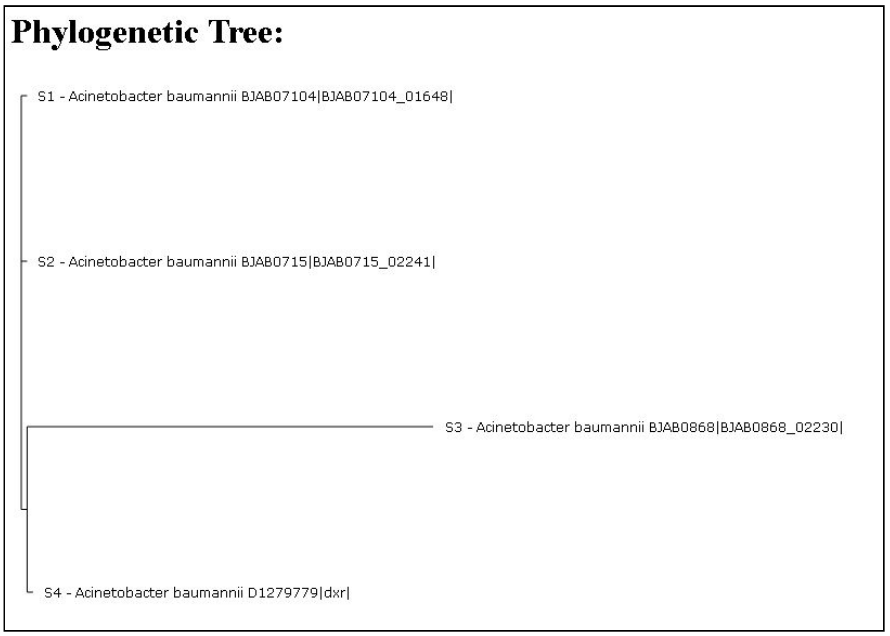


Figure 55 – Results after providing extra parameters for Gamma distributed rates

- A notification is shown when the user has entered a wrong value and the validation has failed. The user is also informed about acceptable values for that parameter (Figure 56).

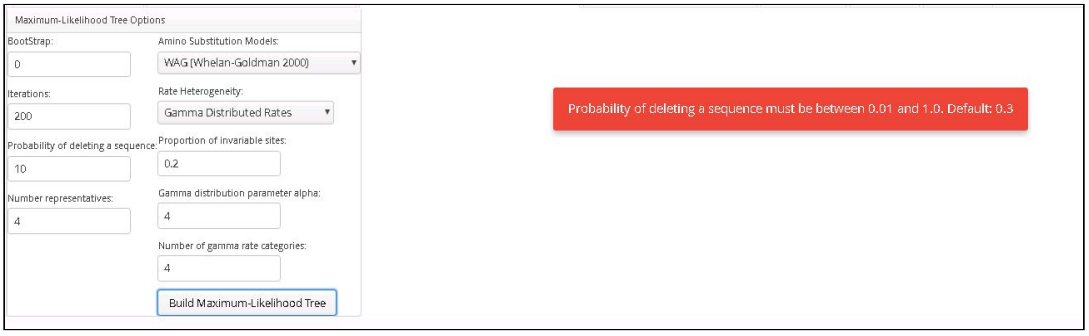


Figure 56 - Notification for Wrong parameter values

11. Implementation of Genomic Island Detection Component

Bacteria are very diverse and versatile, and exist in most habitats that we can think of, including extreme conditions like high temperatures and acidic regions. They adapt very easily and rapidly to physical challenges and to environmental changes. Due to their easy adaptation nature, bacteria have increased their resistance to antibiotics, gained the ability to degrade artificially synthesized substances, mutated for survival when attached to new surfaces and have escaped our medical efforts to eliminate their pathogenic species.

Apart from mutation, bacteria also experience changes in their behavior due to the introduction of blocks of genes from unrelated individuals via horizontal gene transfer (HGT). As a result, bacteria may experience very rapid and

dramatic changes in ecological abilities after acquiring genes, which allow for the degradation of new food sources, or the synthesis of new metabolites, or the attachment to and invasion of host tissues.

Over the past decade, researchers have discovered that apart from the fundamental genes encoding essential metabolic functions, bacterial genomes also contain a variable amount of accessory genes acquired by HGTs that encode adaptive traits, which might be beneficial for the species under certain growth or environmental conditions. This has lead to new challenges in the medical as well as the agricultural sector and for this reason, the analysis of bacterial genomes and HGTs has become a major research area in the bioinformatics field. A significant part of HGT is or has been assisted by genomic islands (GIs), which normally refer to syntenic blocks formed by many accessory genes. GIs are generally recognized as discrete DNA segments containing a group of tens to hundreds of genes whose products may cooperate to confer complex functions to the recipient cells.

Identifying horizontally-transferred genes remains a challenging task despite a number of works done in this area in the last decade, mainly because of the large spectrum of variability found in the compositional properties of both native and acquired genes.

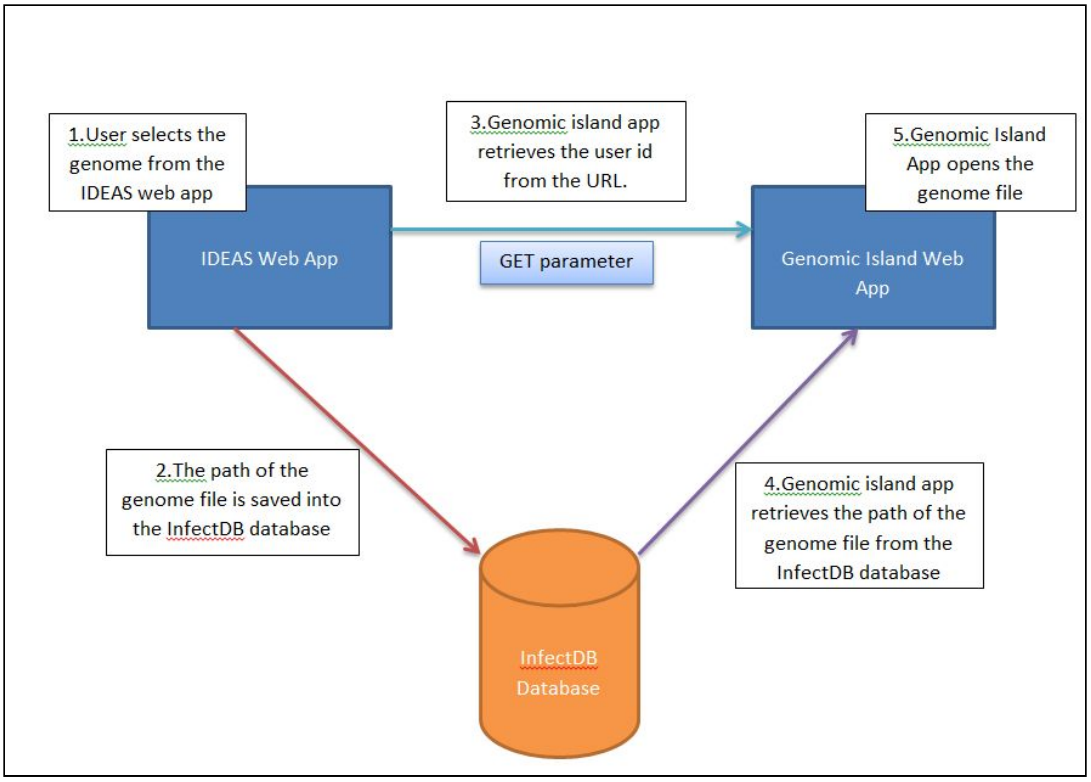
GIs have many specific features. They are often inserted at tRNA genes and are flanked by 16-20 bp perfect direct repeats (DR). They contain mobility genes such as integrases and transposases and unusual guanine and Cytosine (% G+C) content. They are normally large (10-200kb) with small genomic islets (<10kb). Moreover, GIs may be predicted by nucleotide statistics that generally differ from the rest of the genome.

Using these specific features, GI regions can be predicted effectively. The most common GI identification methods are the diversities in sequences between the GI and the host DNA, including codon usage, Guanine-Cytosine (GC) content, k-mer signature analysis and the frequency of specific di-nucleotides and tetra-nucleotides.

An option to detect genomic islands from whole bacterial genomes has also been developed as a web application using the Java Vaadin 7 framework. The Genomic Island (GI) web application is hosted on the same web server as the IDEAS web application. Both web applications also share the same InfectDB database.

The user has to select a genome from the IDEAS web application; the absolute path of the corresponding genome file on the web server is saved in the InfectDB database. The user is then redirected to the GI web application by

changing the url in the web browser and passing the user id as a GET parameter. The GI web application retrieves the genome file by getting the absolute path from the database. The whole process is summarized in Figure



57.

Figure 576 - Moving to GI Detection from the IDEAS Web Application

Genomic islands can be detected as follows:

1. The User has to select a genome from the list and click the Genomic Island button (Figure 58).

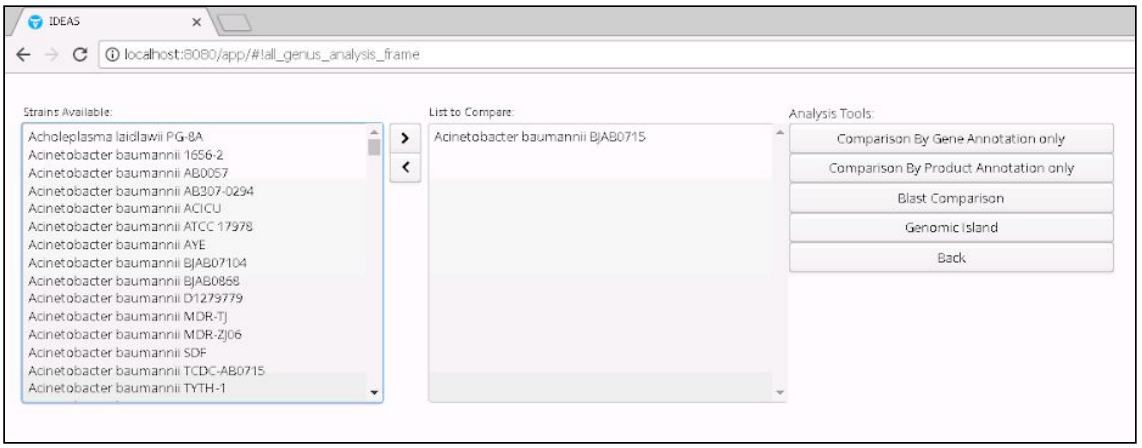


Figure 58 - Choosing a genome to find its GIs

2. The user is then redirected to the main screen of the GI web application (Figure 59). Note that the user id is stored as the query part of the URL. The web application extracts the user id from the URL and loads the file that is stored in the database for that user.

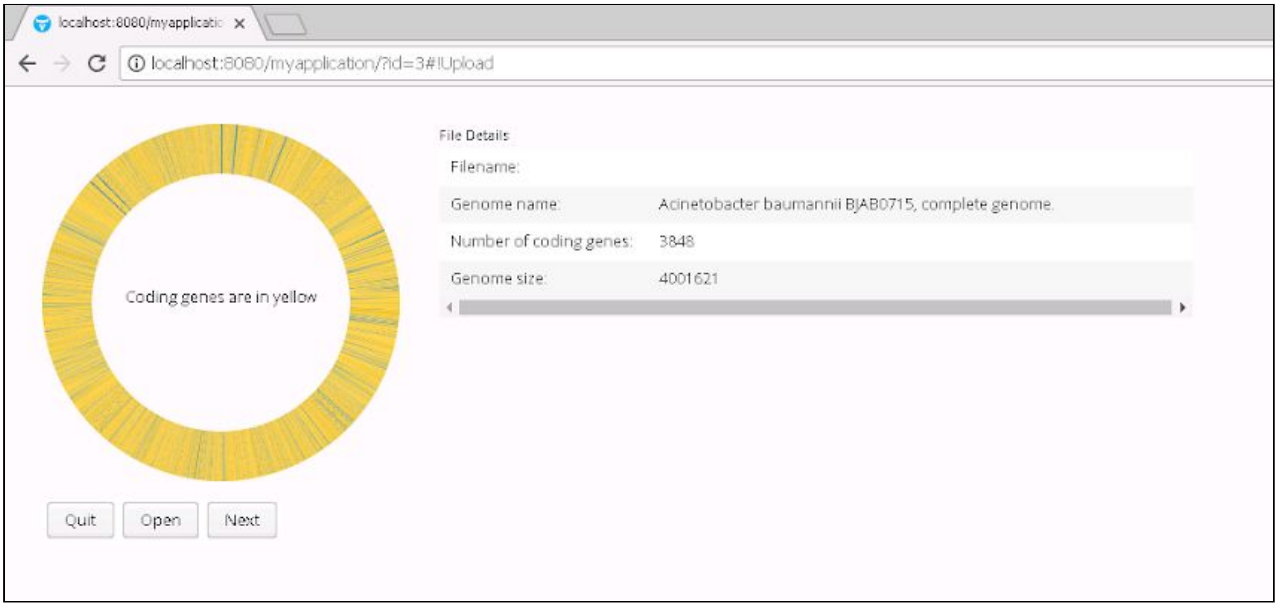


Figure 59 - GI Detection application

The user table stores the path of the genome file (Figure 60)

#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default	Comment
1	ID	INT	10	☑	☐	☐	AUTO_INCREM...	
2	username	VARCHAR	30	☐	☐	☐	0	
3	email	VARCHAR	60	☐	☐	☐	0	
4	password	VARCHAR	60	☐	☐	☐	0	
5	date_time_joined	DATETIME		☐	☐	☐	No default	
6	date_time_last_online	DATETIME		☐	☐	☐	No default	
7	valid	BIT	1	☐	☐	☐	0	
8	genbank	LONGBLOB		☐	☑	☐	No default	genbank file that is us...
9	genbankPath	VARCHAR	256	☐	☑	☐	NULL	

Figure 60 - Table stores genome file path

3. The application next provides a screen that allows the user to select the algorithms that will be used to detect the genomic islands (Figure 61).

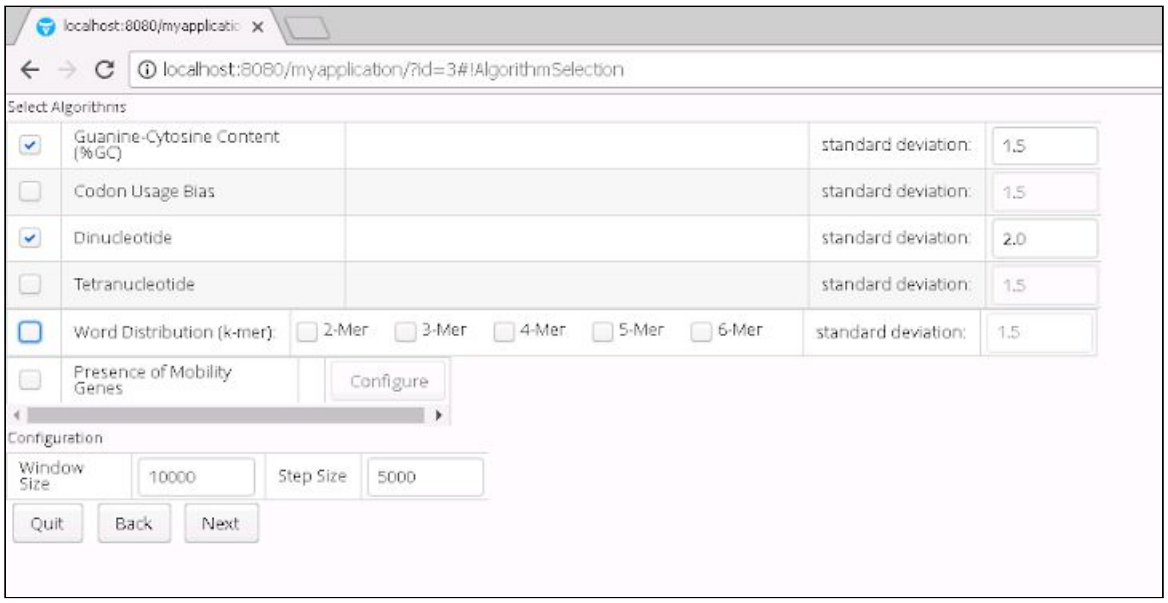
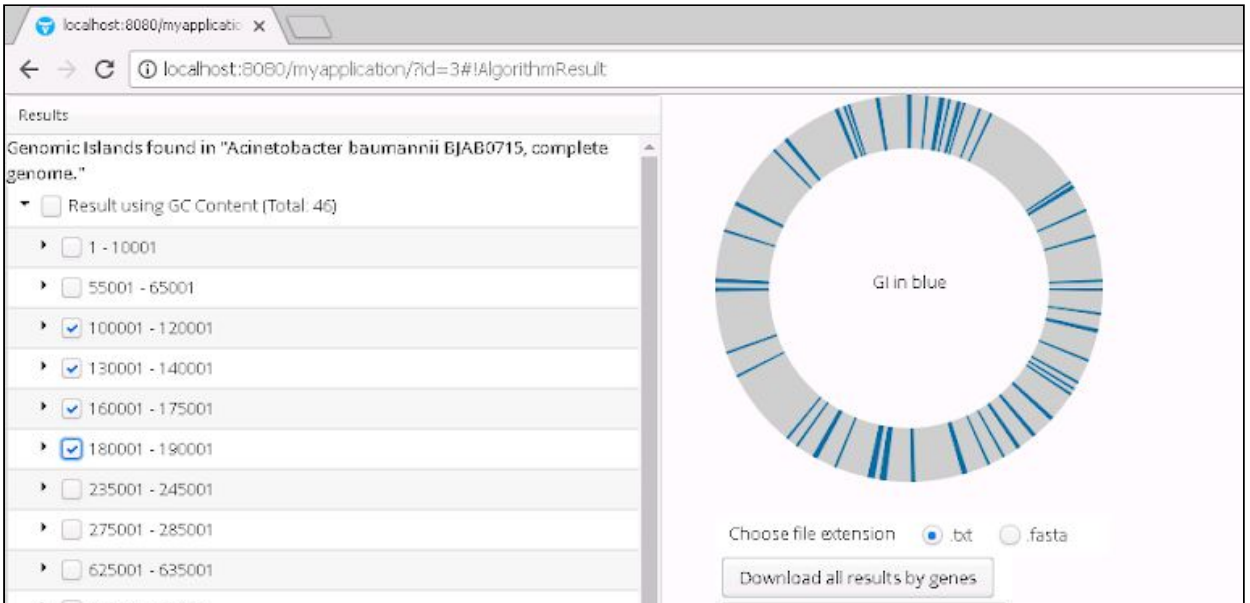


Figure 61 – Choosing algorithms to perform GI detection

4. The genomic islands found using the chosen algorithms are displayed visually and a tree-view format as well (by regions) as shown in Figure 62. The user can select the regions that will be used for analysis.

Figure 62 - GI Results



5. The application also provides an option for choosing a region detected as plausible GI region and sending the same to NCBI Blast portal (Figure 63) to conduct a blast analysis of the selected sequences and download the results:

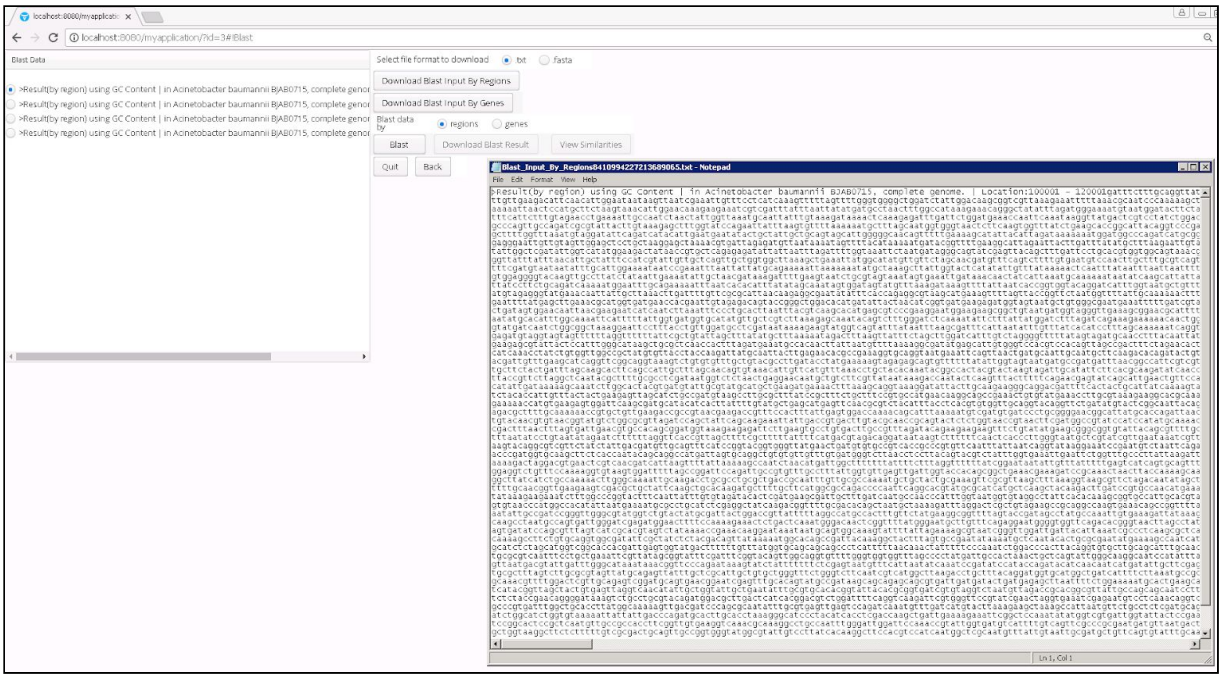


Figure 63 – Blast option

12. Revamping the user interface

The layout of the user interface for the main screen has been modified to be more user-friendly. The organisms that are present in the database are displayed at the top. The functionalities offered by the web application were split into 3 different sections:

Search:-

1. Text Mining
2. Genome Browser

This section allows the user to search for genes, products, mutations, and so on, from our database and other online database (for example NCBI).

Analysis:-

1. Comparison by Gene Annotation
2. Comparison by Product Annotation
3. Comparison by Sequence

Once the sequences have been selected the user can proceed to other analysis options: Multiple sequence alignment, Phylogenetic Analysis, DNDS Analysis, GC Content.

Other:-

1. Genomic Island

This option allows the user to select genomes from the IDEAS data-warehouse and detect genomic islands from those whole bacterial genomes.

13. Genome Browser

This section allows the user to browse through all the organisms and their coding sequences from the IDEAS data-warehouse and select a set of coding-sequences to perform some further analysis.

The database can be searched for genomes as shown in figure 64.

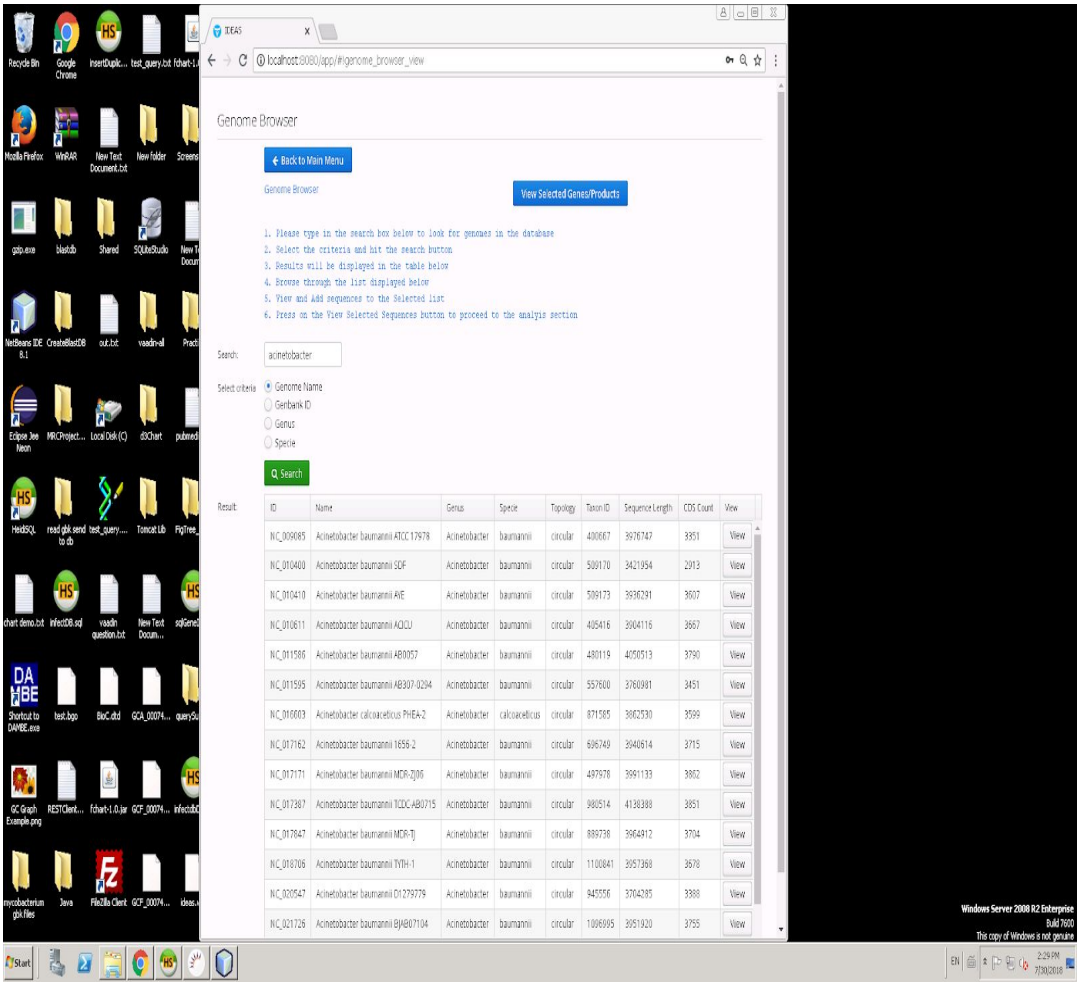


Figure 64 – Search genomes

Genome can be browsed for coding sequences. Specific coding sequences can be selected for analysis as shown in figure 65.

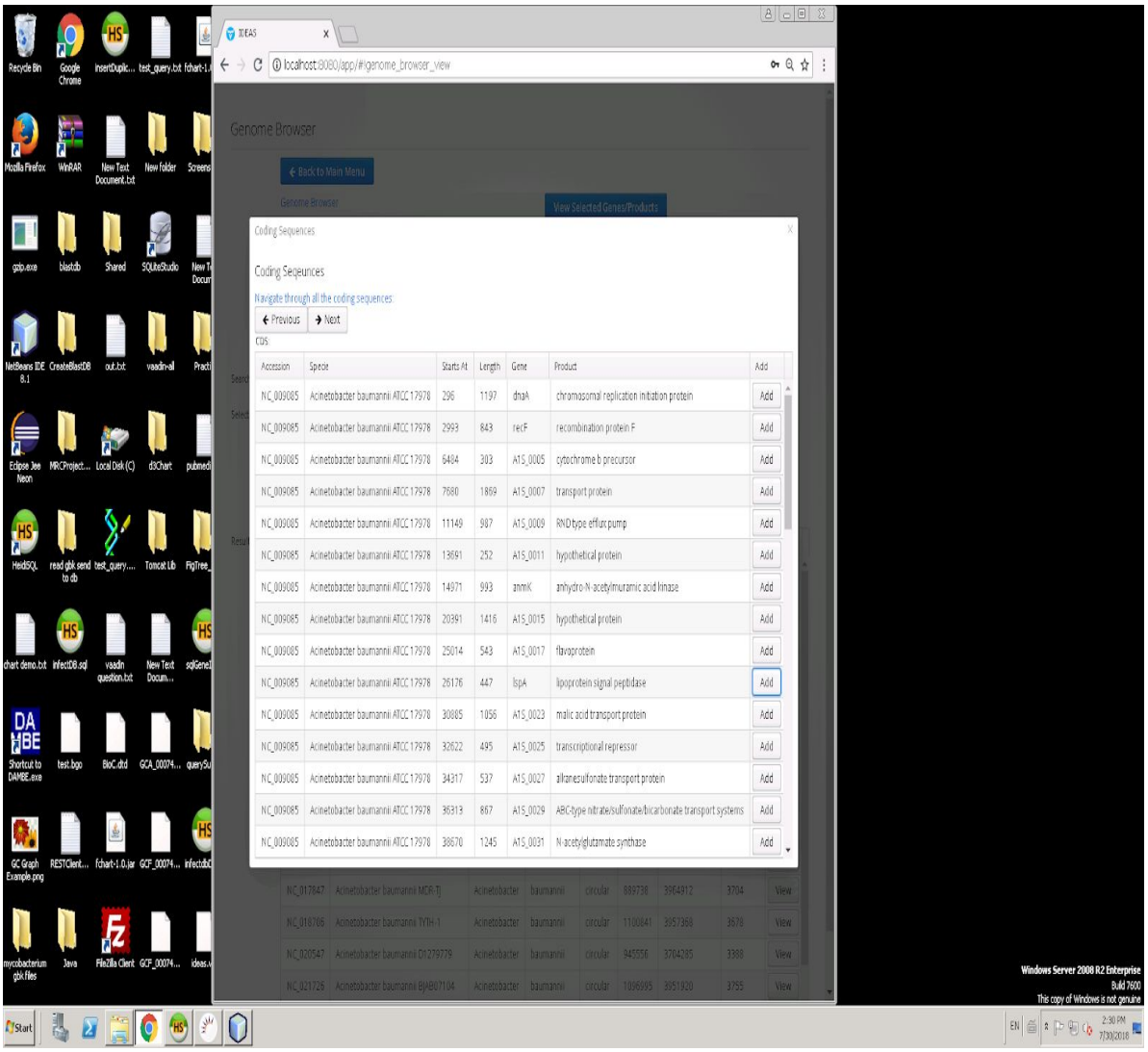


Figure 65 – browse through genome

The selected sequences can then be sent to the analysis section.

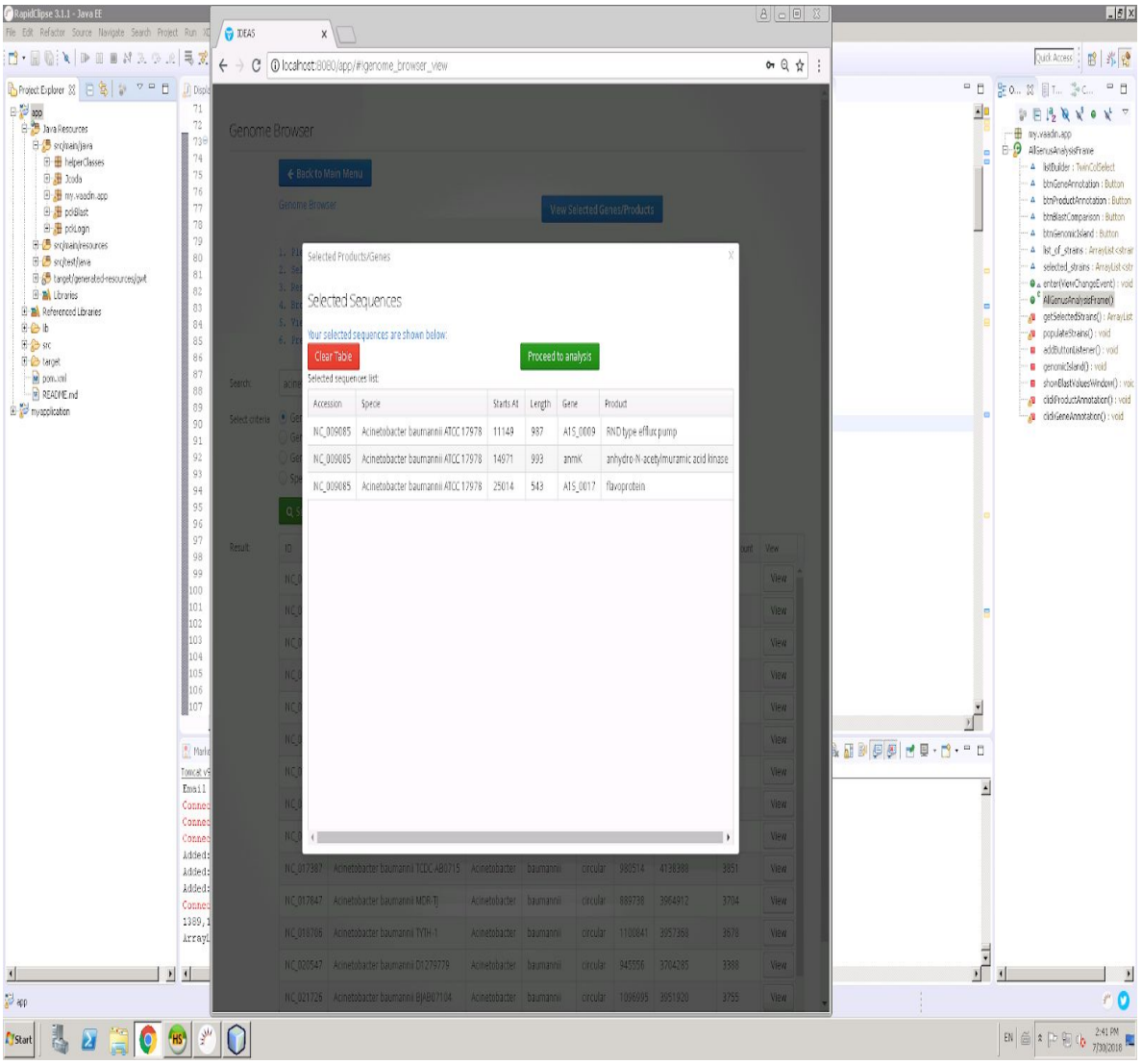


Figure 66 – proceed to analysis

14. Host Specificity

The host specificity section shows the genus in the database and its corresponding host specificity (human, plant or animal).

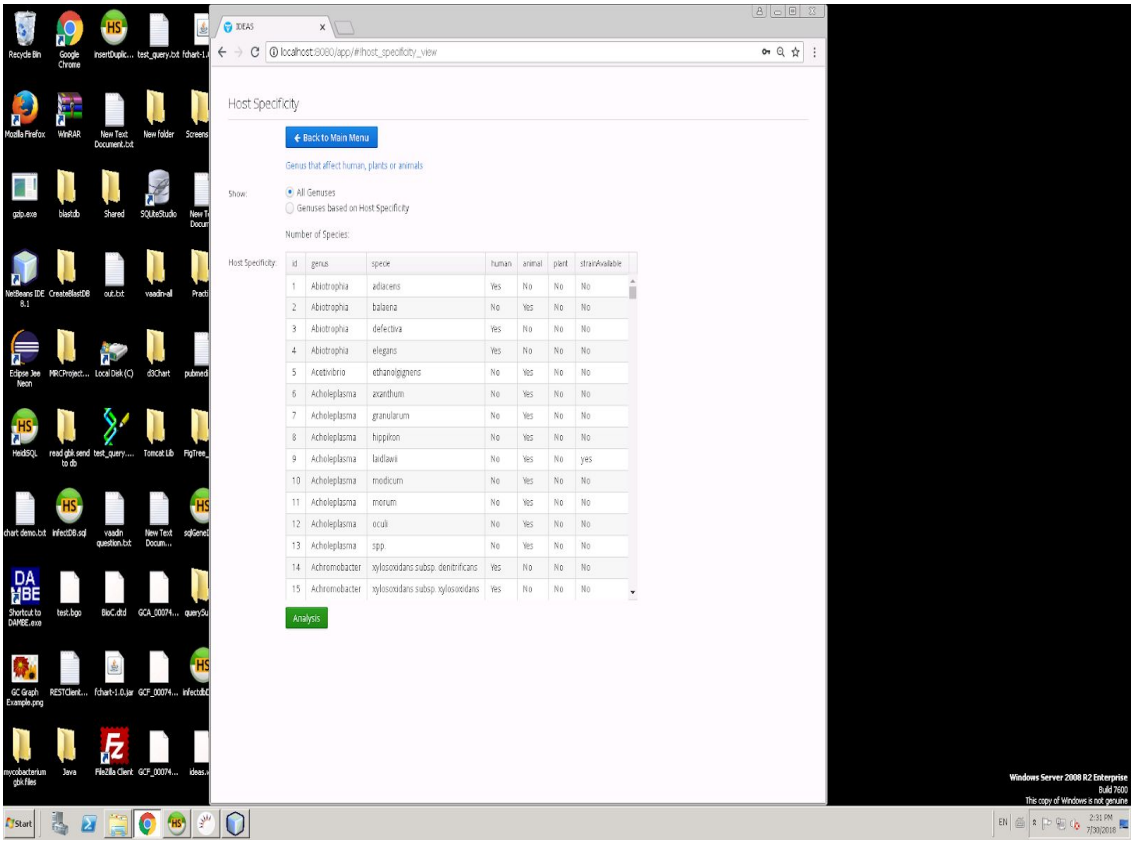


Figure 67 – Host Specificity Main Screen

Genus in the database can be further filtered to show only the ones that affect human, plants or animals.

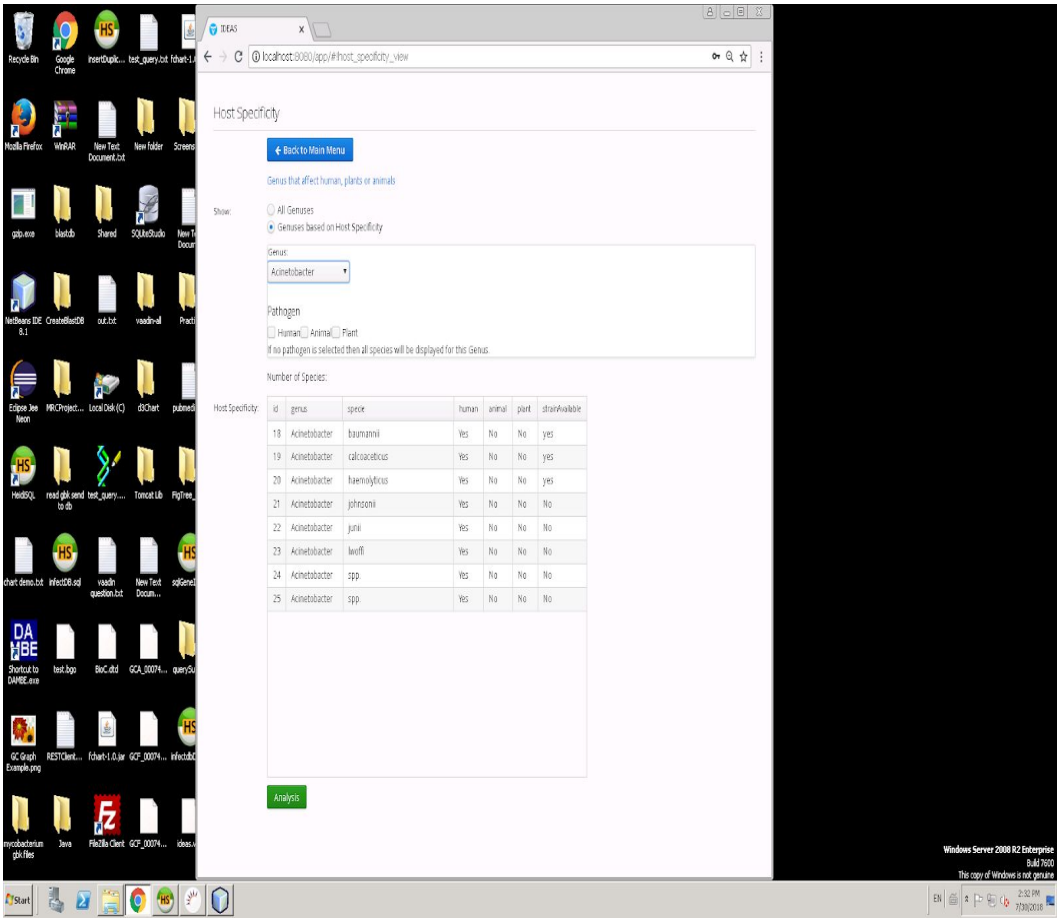


Figure 68 – Filter Genus

The genus can then be selected from the list for analysis as shown in figure 69.

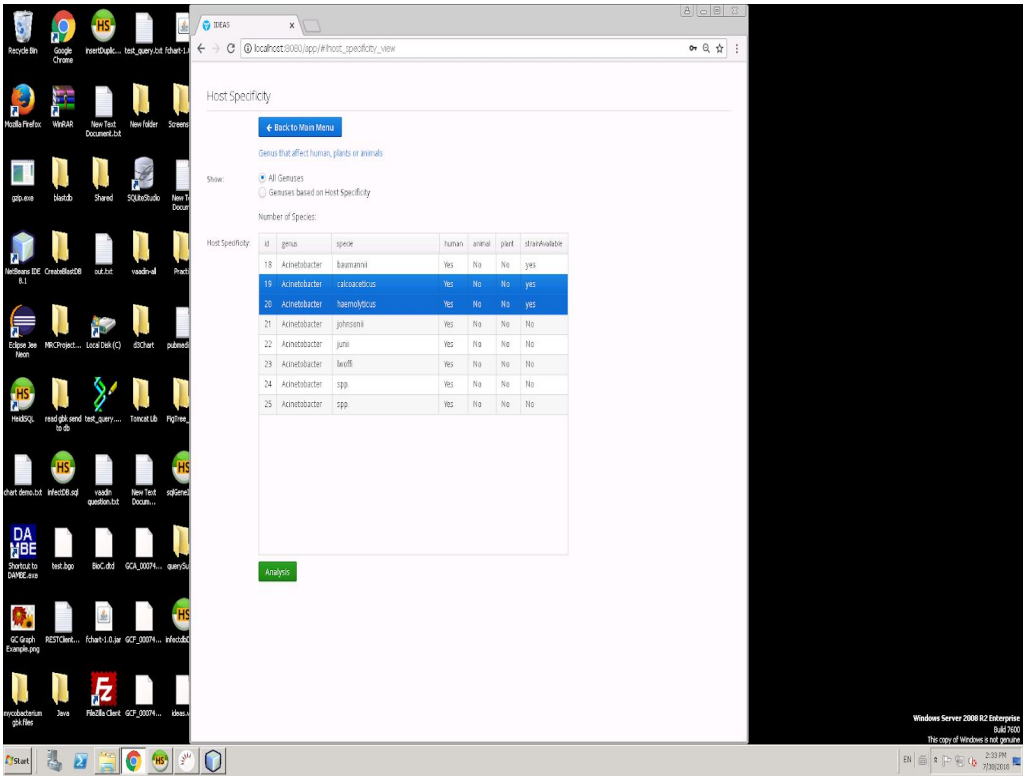


Figure 69 – Select Genus

Further analysis can then be performed on the Genomes as shown in figure 70.

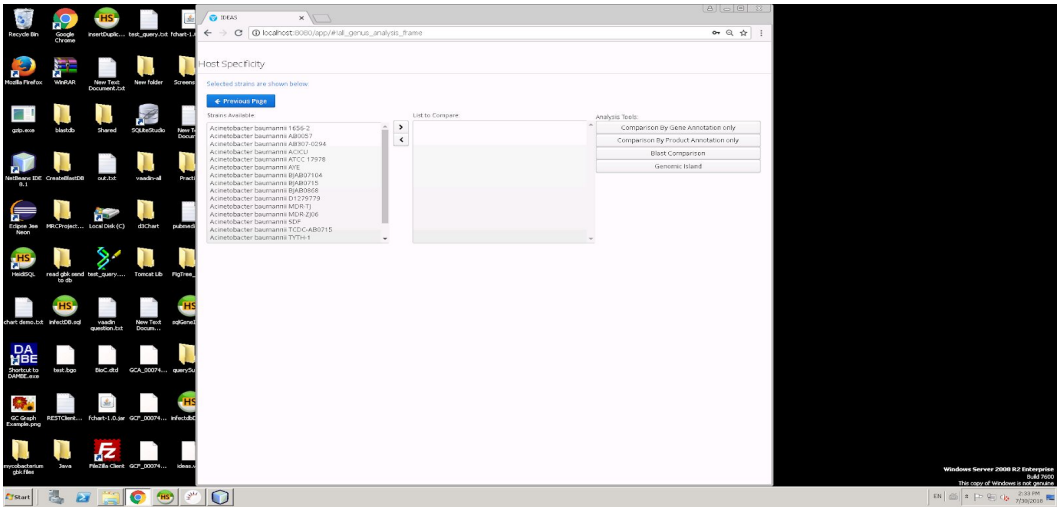


Figure 70 – Proceed to analysis

Chapter 6 - Conclusion

This section gives a brief description of the achievements of the project, difficulties encountered while carrying out the implementation of certain functionalities and the future work that can be carried out to enhance the resulting web application.

6.1 Achievements

The main achievement of IDEAS include:

(i) A web-based application that can be deployed so that researchers from various places can access the system to perform microbial analyses.

(ii) A local-blast based application to compare strains from same organism or different organisms

(iii) A text-mining component that can allow users to search for related literature for a specific gene or organism.

(iv) A component to perform phylogenetic analysis using various algorithms.

(v) An application to perform the dN/dS analysis of chosen genes of interest from various strains/organisms.

(vi) A locally-developed tool to extract genomic islands from a given strain.

Since the application is based on infection-causing bacterial species, users can also perform analyses based on host-specificity of bacterial species.

(vii) The resulting application can be easily extended to include facilities for other microbial analyses e.g. pan-genome analyses, e.g. Roary (https://sanger-pathogens.github.io/Roary/).

In general, most of the core requirements that were set at the beginning of the project were met successfully, except for the GIS component.

6.2 Difficulties Encountered

Prior to starting the development of the resulting application, we already had a pre-populated database of bacterial strains. These can be updated on a regular basis, from the NCBI. Unfortunately we could not automate the update of the data as there are a number of firewalls set at the University of Mauritius network and this does not allow the automatic update. Currently the update can be done manually.

The analyses being performed require a lot of processing power and memory. We have currently used a vm which has limited computing resources. If we manage to get a computer that can crunch data at a faster rate, we can process analyses with more strains.

We do not have a place to host the application. If we get the required resource we can host the application so that it can be accessed externally as well.

6.3 Future Work

We have already implemented a local-blast based algorithm for the comparison of strains but this takes a lot of time. This can be achieved by comparing genes of genomes using their corresponding sequences and aligning them to validate their extent of similarity. The existing algorithm can be improved by the standardisation of all the annotations in terms of gene names or product names. This can be achieved by using a common functional annotation applications like prokka (https://github.com/tseemann/prokka). Then we can use annotations to compare strains which will be much faster.

The DataSet download module can be improved to fetch updated information from NCBI each time a new genome is added to their FTP site and existing genome files is updated, but this will have to circumvent the issue of firewalls on the UoM network.

Additional data sources like KEGG can be used to extract further information about biological pathways or gene functions so as to categorise genes into functional units and perform more efficient analysis on the bacterial genomes.

Lastly, a whole genome comparison tool can be integrated in such a way that it carries whole genome comparisons of existing bacterial genomes and provides an intuitive graphical display of the comparisons to the user.

We can also include visualization tools like IGV (http://software.broadinstitute.org/software/igv/) to compare genomes and display the similarities and differences between them.

References

Altschul, S., Gish, W., Miller, W., Myers, E., Lipman, D. 1990. Basic local alignment search tool. Journal of Molecular Biology. 215 (3): 403–410. doi:10.1016/S0022-2836(05)80360-2. PMID 2231712

Ameli, J., 2015. Communicable Diseases and Outbreak Control. Turkish Journal of Emergency Medicine, 15(Suppl 1), pp.20–26. Available at: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4910139/.

Ashburner M., Ball C. A., Blake JA, et al. 2000. Gene ontology: tool for the unification of biology. Nat Genet;25: 25–29.

Balakrishnan R., Park J., Karra K., et al. 2012. YeastMine–an integrated data warehouse for Saccharomyces cerevisiae data as a multipurpose tool-kit. Database;2012:bar062.

Bansal, A.K., 2005. Bioinformatics in microbial biotechnology -- a mini review. *Microbial Cell Factories*, 4(1), p.19. Available at: http://dx.doi.org/10.1186/1475-2859-4-19.

Berry, M. W. 2003. Survey of Text Mining, Springer, New York, NY, USA.

Brittnacher, M.J, Fong, C., Hayden, H.S, Jacobs, M.A, and Radey, M.; R. 2011. PGAT: a multistrain analysis resource for microbial genomes., *Bioinformatics* 27 (17): 2429-2430.

Carver, T.J., Rutherford K.M., Berriman M., Rajandream M.A., Barrell B.G., and Parkhill, J. 2005. ACT: the Artemis comparison tool. *Bioinformatics* 21 (16): 3422-3423.

Chen Y. A., Tripathi L. P., Mizuguchi K. TargetMine. 2011. An integrated data warehouse for candidate gene prioritisation and target discovery. PLoS ONE;6:e17844.

Chih-Hsuan W., Robert L., Zhiyong L., 2016. Beyond accuracy: creating interoperable and scalable text-mining web services. *Bioinformatics* 2016; 32 (12)

Cline M. S., Smoot M., Cerami E., et al. 2007. Integration of biological networks and gene expression data using Cytoscape. Nat Protoc;2:2366–82.

Donlin M. J. 2009. Using the Generic Genome Browser (GBrowse). Curr Protoc Bioinformatics. John Wiley and Sons, Chapter 9: Unit 9.9.

Google. 2017. *guava-libraries.* Accessed February 5, 2017. https://code.google.com/p/guava-libraries/wiki/GuavaExplained.

Grigoriev, I.V., Nordberg H., Shabalov I., Aerts A., Cantor M., Goodstein D., and

Dubchak, I. 2011. The genome portal of the department of energy joint genome institute. *Nucleic acids research* (gkr947).

Hedley, J. 2010. jsoup: Java html parser.

Kanehisa M., Goto S, Sato Y, et al. 2012. KEGG for integration and interpretation of large-scale molecular data sets. Nucleic Acids Res;40:D109–14.

Kanz, C., Aldebert, P., Althorpe, N., Baker, W., Baldwin, A., Bates, K., and Browne, P. 2005. The EMBL nucleotide sequence database. *Nucleic acids research* 33 (1): D29-D33.

Kemper, B., Matsuzaki, T., Matsuoka, Y., Tsuruoka, Y., Kitano, H., Ananiadou, S., Tsujii, J. 2010. PathText: a text mining integrator for biological pathway visualizations. *Bioinformatics* ; 26 (12): i374-i381. doi: 10.1093/bioinformatics/btq221

Lyne R., Smith R., Rutherford K., et al. 2007. FlyMine: an integrated database for Drosophila and Anopheles genomics. Genome Biol;8:R129.

Markowitz, V.M., Korzeniewski F., Palaniappan K., Szeto E., Werner G., Padki A., and Kyrpides N.C. 2006. The integrated microbial genomes (IMG) system. *Nucleic acids research* 34 (1): D344-D348.

Markowitz, V.M, Chen I.M., Palaniappan K., Chu K., Szeto E., Pillay M., and Kyrpides N.C. 2013. IMG 4 version of the integrated microbial genomes comparative analysis system. *Nucleic acids research* 42 (1): D560-D567.

Mayor, C., Brudno M., Schwartz J.R., Poliakov A., Rubin E.M., Frazer K.A., and Dubchakn, I. 2000. VISTA: visualizing global DNA sequence alignments of arbitrary length. *Bioinformatics* 16 (11): 1046-1047.

Medigue, C., Moszer, I. 2007. Annotation, comparison and databases for hundreds of bacterial genomes. Res Microbiol., 158 (10): 724-736. 10.1016/j.resmic.2007.09.009.

Miyazaki, S., Sugawara, H., Ikeo, K. , Gojobori, T., and Tateno, Y. 2004. DDBJ in the stream of various biological data. *Nucleic acids research* 32 (1): D31-D34.

Page, R.D. 2002. "Page, Roderic DM. "Visualizing phylogenetic trees using TreeView." *Current Protocols in Bioinformatics* 6 (2).

Rose P. W., Bi C, Bluhm W. F., et al. 2013, The RCSB Protein Data Bank: new resources for research and education. Nucleic Acids Res;41:D475–82.

Salemi, M., Vandamme, A.-M., & Lemey, P. (2009). *The phylogenetic handbook: A practical approach to phylogenetic analysis and hypothesis testing*. Cambridge, UK: Cambridge University Press.

Sanger, F., Nicklen, S., and Coulson, A. R. 1977. DNA sequencing with chain-terminating inhibitors, Proceedings of the National Academy of

Sciences of the United States of America, vol. 74, no. 12, pp. 5463–5467, 1977.

Sebastiani, F. 2002. Machine learning in automated text categorization, ACM Computing Surveys, vol. 34, no. 1, pp. 1–47, 2002.

Shah, S.P, Y. Huang, T. Xu, M.M Yuen, J. Ling, and B.F Ouellette. 2005. "Atlas–a data warehouse for integrative bioinformatics." *BMC bioinformatics* 6 (1): 34.

Smedley et.al. (2015), The BioMart community portal: an innovative alternative to large, centralized data repositories. *Nucl Acids Res*; 43 (W1): W589-W598. doi: 10.1093/nar/gkv350

Smith, R.N. et al., 2012. InterMine: a flexible data warehouse system for the integration and analysis of heterogeneous biological data. Bioinformatics, 28(23), pp.3163–3165. Available at: http://dx.doi.org/10.1093/bioinformatics/bts577.

Steinway, S.N., Dannenfelser, R., Laucius, C.D., Hayes, J.E., and Nayak, S. 2010. Software JCoDA: a tool for detecting evolutionary selection, BMC Bioinformatics 2010, 11:284

Töpel, T., B. Kormeier, A. Klassen, and R. Hofestädt. 2008. BioDWH: a data warehouse kit for life science data integration. *Journal of integrative bioinformatics* 5 (2): 93.

Triplet T. and Butler G.; (2013) A review of genomic data warehousing systems, *Brief Bioinformatics*, 15 (4): 471-483. doi: 10.1093/bib/bbt031

Wheeler, D.L., Barrett T., Benson D.A., Bryant S.H., Canese K., Chetvernin V., and Church, D.M. 2007. Database resources of the national center for biotechnology information. *Nucleic acids research* 35 (1): D5-D12.

Zhang,J., Haider,S., Baran,J., Cros,A., Guberman,J.M., Hsu,J., Liang,Y., Yao,L. and Kasprzyk,A. (2011) BioMart: a data federation framework for large collaborative projects. Database, bar038.

Zeng D., Chen H., Lynch C., Eidson M., Gotham I. (2005) Infectious disease informatics and outbreak detection. In: Chen H, Fuller SS, Friedman C, Hersh W (eds) Medical informatics: knowledge management and data mining in biomedicine. Springer, New York