

AN ONLINE TOOL FOR METAGENOMICS ANALYSIS

BY

CHERALYN PHONG JIA ERN

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfilment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Perak Campus)

JAN 2018

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: _____

Academic Session: _____

I _____
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature)

(Supervisor's signature)

Address:

Supervisor's name

Date: _____

Date: _____

AN ONLINE TOOL FOR METAGENOMICS ANALYSIS

BY

CHERALYN PHONG JIA ERN

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONS)

Faculty of Information and Communication Technology

(Perak Campus)

JAN 2018

DECLARATION OF ORIGINALITY

I declare that this report entitled “AN ONLINE TOOL FOR METAGENOMICS ANALYSIS” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _____

Name : CHERALYN PHONG JIA ERN_____

Date : _____

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and appreciation to my supervisor, Dr. Ng Yen Kaow for this opportunity to collaborate with a Hong Kong City University student to get data for this project and for providing invaluable guidance all the way until completion of this project. To my parents for their unconditional love, support and constant encouragement, for without them I would not be where I am today. Finally I would like to extend my sincere thanks to all who have directly or indirectly aided me in the completion of this project.

ABSTRACT

This project aims to produce a web-based online tool for metagenomics analysis. The tool is to be easily accessible to researchers and students alike to visualize metagenomics analyses. At present, there are existing websites that do metagenomics analysis online but regrettably, there isn't one to display whole genome shotgun metagenomics analysis results. Current available tools provide analysis but these websites only provide one aspect of the analysis such as drawing a heat map or building trees. Much consideration should be put into this problem given that without a proper method to sequence long DNA strands, it is near impossible to completely sample and analyse the diversity and abundance of all genetic information in a complex microbial community. It would also be immensely difficult to study uncultivable microorganisms because these microorganisms would not be available for use in laboratories. With metagenomics shotgun sequencing, human diseases can be better understood. Genetic variants in the human population can be identified and correlated to distinguished phenotypes which can be traits leading to a disease. The objective of this project is to provide one such tool whereby complete analysis and visualization of inputted data can be achieved. The visualization of the analysis includes the drawing area, which displays the major graph. Interactions with the graph such as hovering and clicking would further display additional information in suspension boxes and some data processing when required. The final output is an interactive tool made available for metagenomics analysis whereby users can access it online to upload data and visualise as well as interact with the drawing.

TABLE OF CONTENTS:

TABLE OF CONTENTS

TITLE	iv
DECLARATION OF ORIGINALITY	v
ACKNOWLEDGEMENTS	vi
ABSTRACT	vii
TABLE OF CONTENTS	viii
LIST OF FIGURES	x
LIST OF TABLES	xiii
LIST OF ABBREVIATIONS	xiv
CHAPTER 1: INTRODUCTION	15
SECTION 1.1- PROBLEM STATEMENT AND MOTIVATION	15
SECTION 1.2- OBJECTIVES	16
SECTION 1.3- PROJECT SCOPE AND DIRECTION	16
SECTION 1.4- CONTRIBUTION	18
SECTION 1.5- DISSERTATION SUMMARY	19
CHAPTER 2: LITERATURE REVIEW	22
SECTION 2.1- GALAXY/HUTLAB	22
SECTION 2.2- INTERACTIVE TREE OF LIFE	24
SECTION 2.3- ST. JUDE PECAN DATA PORTAL	25
SECTION 2.4- IMG/M	27
SECTION 2.5 - EBI METAGENOMICS	29
SECTION 2.6 – FINAL VERDICT	32
CHAPTER 3: SYSTEM DESIGN	33
SECTION 3.1- DESIGN SPECIFICATIONS	33
3.1.1 Methodology	33
3.1.2 Tools to use	34
3.1.3 General Work Procedures	35

TABLE OF CONTENTS:

SECTION 3.2- SYSTEM DESIGN/OVERVIEW	38
CHAPTER 4: DATA PROCESSING	50
SECTION 4.1- OVERVIEW	50
SECTION 4.2- TAXONOMY TREE	54
CHAPTER 5: DRAWING AND INTERACTION	59
SECTION 5.1- OVERVIEW	59
5.1.1 Heat Map	59
5.1.2 Tree	65
SECTION 5.2- TAXONOMY TREE	68
CHAPTER 6: CONCLUSION	73
BIBLIOGRAPHY	76
POSTER	78
PLAGIARISM CHECK SUMMARY	79

LIST OF FIGURES:

LIST OF FIGURES

Figure Number	Title	Page
Figure 1.2.1	Example of an overview display	18
Figure 1.2.2	Example of a Taxonomy Tree display	18
Figure 1.5.1	Example of extra assets	20
Figure 1.5.2	Overview	21
Figure 1.5.3	Taxonomy tree	23
Figure 2.1.1	Galaxy/Hutlab main page	24
Figure 2.2.1	Interactive Tree of Life sample dataset displaying an interactive phylogenetic tree	26
Figure 2.3.1	St. Jude PeCan Data Portal home page	27
Figure 2.3.2	St. Jude PeCan Data Portal Protein Paint of TP53	28
Figure 2.4.1	Integrated Microbial Genomes and Microbial Samples –Joint Genome Institute Homepage	29
Figure 2.4.2	Radial Tree distribution of <i>Acidianus hospitalis</i> W1, <i>Acidianus manzaensis</i> YN-25, <i>Acidilobus</i> <i>saccharovorans</i> 345-15, <i>Aciduliprofundum boonei</i> T469 and <i>Aciduliprofundum</i> sp. MAR08-339	30
Figure 2.5.1	EBI Metagenomics Home Page	31
Figure 2.5.2	Biological process heat map from Comparative freshwater metagenomics of Swedish and American lakes	32
Figure 3.1.1	Phased Development (Ku, 2017)	35
Figure 3.1.1	Sample heat map display	38
Figure 3.2.1	Flowchart for development process	41
Figure 3.2.2	Framework on localhost	46
Figure 3.2.3	administration page	47
Figure 3.2.4	add analysis page part 1	48

LIST OF FIGURES:

Figure 3.2.5	add analysis page part 2	49
Figure 3.2.6	Dovirus dashboard	49
Figure 3.2.7	Dovirus project page	50
Figure 3.2.8	Dovirus project repository	50
Figure 3.2.9	Dovirus db folder contents	51
Figure 3.2.10	Overview contents	51
Figure 3.2.11	Visualizer.ts	52
Figure 3.2.12	The database model	53
Figure 4.1.1	Data mapping	54
Figure 4.1.2	Data loading	54
Figure 4.1.3	Data handling for heat map (red) and tree (blue)	55
Figure 4.1.4	Data as objects in an array (heat map)	55
Figure 4.1.5	Creating restructured data (heat map)	56
Figure 4.1.6	NewickNode interface	56
Figure 4.1.7	Setup editor	57
Figure 4.1.8	File Description	57
Figure 4.2.1	Data interface	58
Figure 4.2.2	Data load	58
Figure 4.2.3	Data handling 1	59
Figure 4.2.4	Data handling 2	59
Figure 4.2.5	Data handling 3	60
Figure 4.2.6	Data handling visualization 1	60
Figure 4.2.7	Data handling 4	61
Figure 4.2.8	Data handling 5	62
Figure 4.2.9	Data handling 6	62
Figure 4.2.10	Data handling 7	62
Figure 5.1.1	Color Gradient Legend	64
Figure 5.1.2	Horizontal Tree	64
Figure 5.1.3	Vertical Tree	64
Figure 5.1.4	Heat Map	65

LIST OF FIGURES:

Figure 5.1.5	Overview heat map	65
Figure 5.1.6	Rect in the heat map	66
Figure 5.1.7	Color Scale	66
Figure 5.1.8	Drawing part 1	67
Figure 5.1.9	Drawing part 2	68
Figure 5.1.10	Example of Hover function interactivity	69
Figure 5.1.11	Drawing part 3	70
Figure 5.1.12	Drawing part 4	71
Figure 5.1.13	Visualization of SVG Path Mini-Language	72
Figure 5.2.1	Taxonomy Tree	73
Figure 5.2.2	Drawing part 1	73
Figure 5.2.3	Drawing part 2	74
Figure 5.2.4	Drawing part 3	74
Figure 5.2.5	Drawing part 4	75
Figure 5.2.6	Taxonomy Tree interaction	76
Figure 5.2.7	Example Taxonomy Tree interaction	77

LIST OF TABLES:

LIST OF TABLES

Table 2.6.1	Brief comparison	34
Table 3.2.1	Functional Interaction Sidebar aspects and details	39
Table 3.2.2	Input list and how the information should be rendered	40

LIST OF ABBREVIATIONS:

LIST OF ABBREVIATIONS

PCA	Principle Coordinate Analysis
LEfSe	Linear discriminant analysis effect size
HGT	Horizontal Gene Transfer
DNA	Deoxyribonucleic acid
EBI	European BioInformatics
SVG	Scalable Vector Graphics
d3	Data Driven Documents
CSV	Comma Separated Values

CHAPTER 1: INTRODUCTION

CHAPTER 1: INTRODUCTION

SECTION 1.1- PROBLEM STATEMENT AND MOTIVATION

Metagenomics is the study of microorganisms in their natural living habitat, which normally involves the multiplex of microbial communities that coexists along with it (Smith, 2017). Microbes are everywhere in the environment and thus involved in contributing to every process in the ecosystem, playing important if not crucial roles. The microbes that thrive on the surface of the human body alone outnumber human cells by over 10 times, so understanding how this microbial community structure affects the human body may help in providing better diagnosis for deterring, inhibiting and treating diseases (National Research Council (US) Committee on Metagenomics, 2007).

Metagenomics shotgun sequencing is a complicated process of sequencing DNA from a sample to determine correlation between microorganisms and their interactions. Samples taken from the same gut may contain such vast amounts of organisms that each individual sample might not even overlap and may prove hard to determine interactions amongst each other.

Through metagenomics shotgun sequencing, researchers are able to extensively sample all genes in a complex microbial community to analyse bacterial diversity and abundance as well as study unculturable microorganisms that would otherwise be near impossible to study (Shotgun Metagenomic Sequencing, 2017). According to reviews done by Wang & Jia (2016), research using shotgun metagenomic sequencing on metagenomics shows microbial markers that indicate association with diseases such as obesity, type 2 diabetes, liver cirrhosis, colorectal cancer and rheumatoid arthritis were identifiable based on samples collected in the gut microbiome. Development in bioinformatics technology has made studying the human microbiome more approachable through metagenomics shotgun sequencing. Some tools that can do analyse metagenomics exist but none provide whole genome shotgun metagenomics analysis results.

Much consideration should be put into this problem given that without a proper method to sequence long DNA strands, it is near impossible to completely sample and analyse the diversity and abundance of all genetic information in a complex microbial community. It would also be immensely difficult to study

CHAPTER 1: INTRODUCTION

uncultivable microorganisms because these microorganisms would not be available for use in laboratories. With metagenomics shotgun sequencing, human diseases can be better understood. Genetic variants in the human population can be identified and correlated to distinguished phenotypes which can be traits leading to a disease. We can also make data less challenging to analyse as everything is computerised. There are existing websites that do metagenomics analysis online but currently, there isn't one to display whole genome shotgun metagenomics analysis results. These websites only provide one aspect of the analysis such as drawing a heat map or building trees. This project aims to create an online tool to provide researchers and students alike to do whole genome shotgun metagenomics analysis.

SECTION 1.2- OBJECTIVES

1. To aid researchers in visualizing huge datasets more easily.
2. To provide an online tool for whole genome shotgun metagenomics analysis result.
3. To create more interactive and useful graphs that can better explain and display the inputted data set so that key information is highlighted and easily noticed.
4. To come up with design visualizations for new analysis and figures.
5. To create a complete tool which comprises of multiple different aspects that highlights all attributes of the dataset so that users can complete all analysis on a metagenomics dataset just from this single website without needing to input the dataset into multiple tools to generate different graphs.
6. To allow users to display and adjust figures online and then export them with publishable quality to be directly used in articles.

CHAPTER 1: INTRODUCTION

SECTION 1.3- PROJECT SCOPE AND DIRECTION

This project aims to develop an online tool to aid researchers and students in visualizing and displaying metagenomics analysis results more efficiently. Whole genome shotgun metagenomics analysis results shall be provided. Based on backstage pipeline, some new analysis and figures shall be designed so users can display and adjust their figures online and output them with publishable quality to be directly used in articles. The tool shall be able to display every figure necessarily with:

- Drawing area – the major graph is displayed
- Interaction – interacting with the graph. Actions such as hovering and clicking to display links and other detailed information in suspension boxes.
- Data process – might not necessarily apply to every figure but may be required to calculate sum etc.

The interfaces we are going to provide include:

- Overview – the overview of the entire dataset with integrated heat map, clustering, differential bar plots and composition bar plots in one figure.
- Taxonomy Abundance Tree – a breakdown tree showing abundance of reads from one specific organism.

CHAPTER 1: INTRODUCTION

Below are examples of what is to be done but figures included only shows the drawing area. Interactions are not shown.

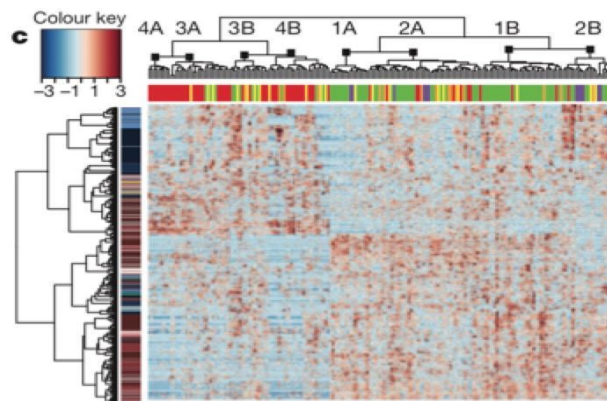


Figure 1.2.1 – Example of an overview display

Figure 1.2.1 shows the overview display which displays a heat map, a tree, information bar, composition bar and compare bar. Each sub-display in this entire display takes in an input to generate the data. The heat map generates information from a matrix of abundance, the tree from a tree file, and the information bar from sample clinic information. Other interactions include displaying detailed information such as detailed values and samples as well as genus information when the mouse hovers over the heat map.

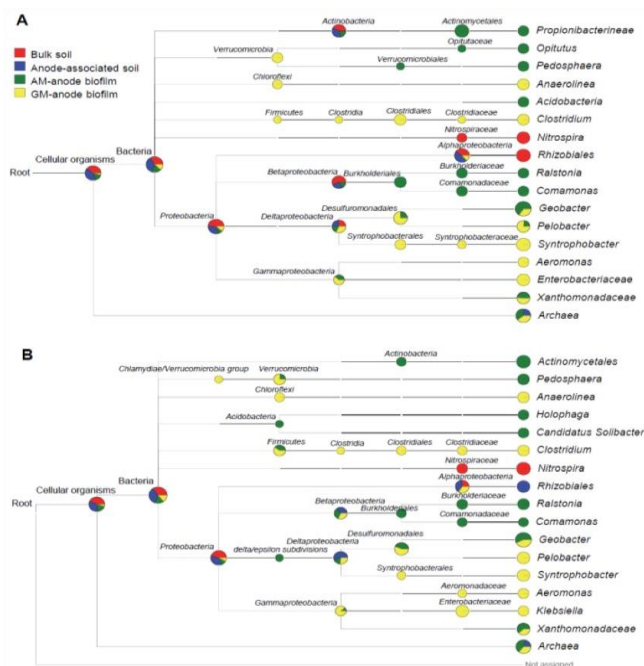


Figure 1.2.2 – Example of a Taxonomy Tree display

CHAPTER 1: INTRODUCTION

Graphs depicting the metagenomics analysis information shall be created using data collected by a metagenomics student researcher. Interactions shall be further developed throughout the entire process.

SECTION 1.4- CONTRIBUTION

Researchers working in labs pioneering the cure to diseases, students in the bioinformatics field and anyone studying in any field related to metagenomics and the study of DNA will encounter this problem and need this solution. With recent advances in this field, it has become tremendously easy for people to get whole shotgun metagenomics sequencing data but yet, many do not know how to analyse the data and get useful insights for clinical problems. With this platform, our interface gives people a chance to conduct analysis online which provides ease of access and is meaningful to their research. The findings of the researchers, through the use of this tool, will not only contribute to helping doctors cure their patients, but also help early detection and prevention of diseases in currently healthy individuals.

This project aims to provide society with an online tool for whole genome shotgun metagenomics analysis results so that the user can make changes to their figures online and export these figures with publishable quality to be used in articles. Visualization of the complex metagenomics is a huge problem therefore this tool not only helps visualize but also saves time. It will also provide users with a complete tool like no other for its users to do all the necessary analysis on their data set without the need for any other tool. As this tool is online, users will have ease of access wherever they go. The figures obtained using this tool are going to provide better visualization and understanding of the complex and diverse microbial communities, help propel research in this field and provide insights into microbial communities. This tool will then later be integrated into <http://www.dovirus.com/> as part of many other tools for various types of other research purposes.

CHAPTER 1: INTRODUCTION

SECTION 1.5- DISSERTATION SUMMARY

The overall aim of this project, to provide tools for visualization of metagenomics analysis, has been achieved. This project is ideally to be done on a Mac operating system although other Linux operating systems can support it. The framework was successfully installed on Ubuntu operating system with all the required dependencies. Dependencies include an object-oriented programming language – Python, a package manager – Node/npm, an object-relational advanced open source database management system – PostgreSQL and a version control system – Git. On success of the set-up, the system has to first be analysed and understood before we can build the metagenomics visualization analysis on it. Before any drawings can be done, we first have to create a project which contains all the analyses, an analysis category to categorize the analysis and then we create the analysis. And additional assets used such as css files or javascript files can be added under the analysis.

Change analysis

Name:

Url name:

Template name:

Extra assets:

```
{
  "css":[
    "app/overview/overview.css"
  ],
  "js":[
    "app/overview/newick.js"
  ]
}
```

Enter valid JSON

Figure 1.5.1 – Example of extra assets

CHAPTER 1: INTRODUCTION

These are the analyses that has been achieved:

Overview

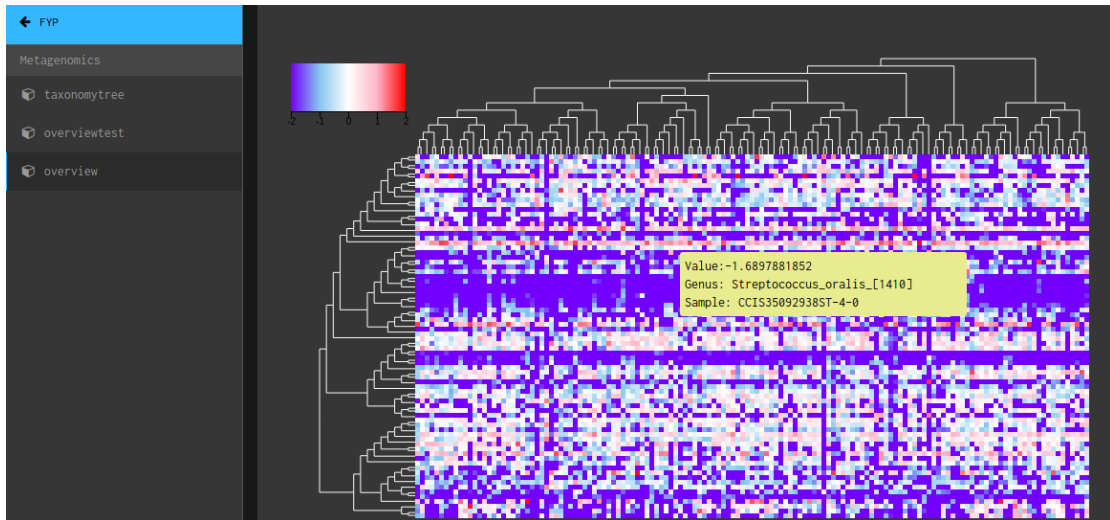


Figure 1.5.2 – Overview

Interactivity – On hover, displays information such as value, genus and sample of that particular unit.

Taxonomy Tree

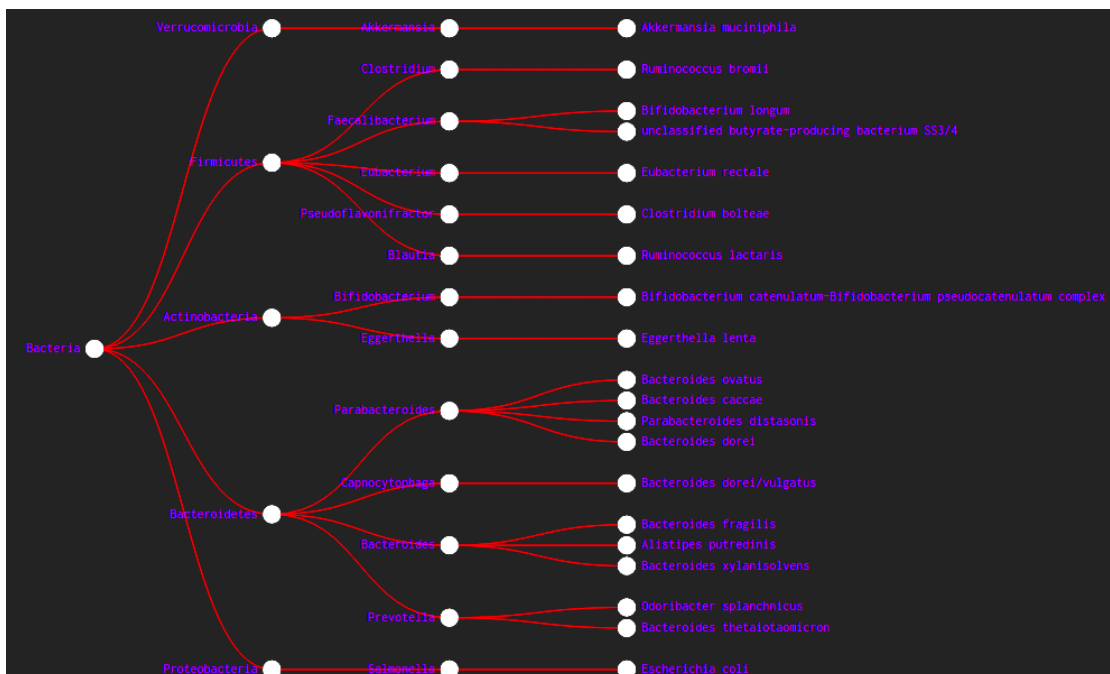


Figure 1.5.3 – Taxonomy tree

CHAPTER 2: LITERATURE REVIEW

CHAPTER 2: LITERATURE REVIEW

SECTION 2.1- Galaxy/Hutlab

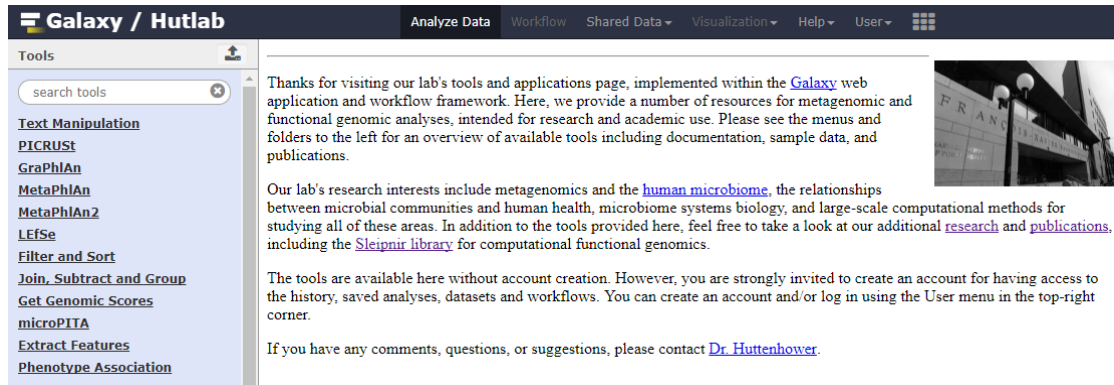


Figure 2.1.1 – Galaxy/Hutlab main page

Galaxy is an online laboratory tool application page to aid research and academic use which provides resources for metagenomic analysis. Dr Curtis Huttenhower, the principal investigator at The Huttenhower Lab created this online tool for the purpose of conducting research on metagenomics and the human microbiome, the correlation between microbial communities and human health, microbiome systems biology and large-scale computational methods for investigation in all of these areas. Some of the tools for composition analysis provided include:

HUMAnN – Profile microbial pathways and genes; stratify by contributing species

MetaPhlAn – Profile microbial clades and their amplitude

PhyloPhlAn – Rebuild microbial phylogenetic trees

ShortBRED – Profile proteins of interest with very elevated specificity

StrainPhlAn – Identify and keep track of microbial strains using SNPs in conserved, distinctive marker genes

MelonnPan – Predict metabolome content from microbial enzyme abundance

CHAPTER 2: LITERATURE REVIEW

Strengths:

- Provides tutorials and walkthroughs on installation and usage.
- Provides sample coding.
- Provides a variety of available external databases to load from.
- Integration of multiple online tools.
- Very detailed result queries.
- Saves history of used data for later access.

Limitations:

- Require login to use workflows.
- No visual representation to aid analysis.
- Tool can be complicated and confusing to use.
- Does not display whole genome shotgun metagenomics analysis results.
- Page goes down occasionally, making access at times unavailable.

Proposed improvements:

- Allow guest users to use workflows.
- Create more interactive visual representations of data.
- Make it more user friendly.

CHAPTER 2: LITERATURE REVIEW

SECTION 2.2- Interactive Tree of Life

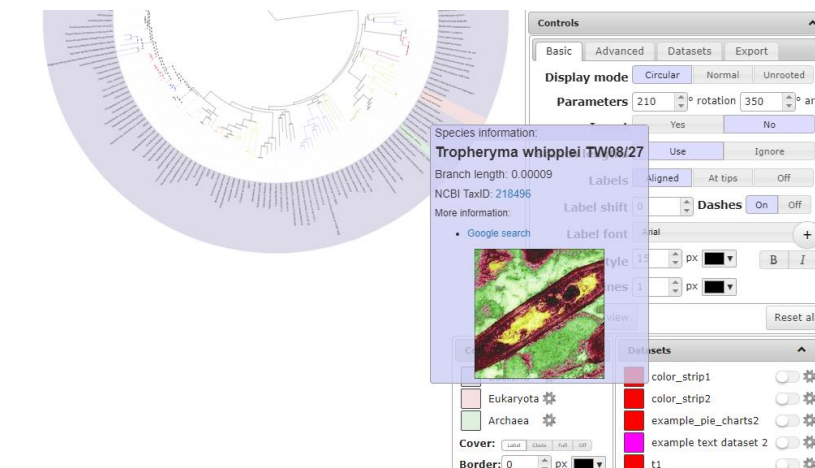


Figure 2.2.1 – Interactive Tree of Life sample dataset displaying an interactive phylogenetic tree

Interactive Tree of Life is a free online interactive tool to display phylogenetic trees and is supported by the latest web browsers. It was released in 2006 and up till today is still running to provide users with an interactive way of viewing phylogenetic trees with external data. This tool supports phylograms, cladograms, rectangular, radial, rooted and unrooted trees besides allowing users to drag to move and click to expand inner nodes to display additional information. A search button is also provided to help speed up searching data. Currently, Interactive Tree of Life supports 14 different dataset types and provide export functions both as vector graphics and as bitmaps.

Strengths:

- Simple, easy to use interface
- Supports up to 100,000 leaves and 5 datasets in a single display and unlimited number of datasets allowed
- Newick and the Nexus file formats supported
- Precise positioning of annotation
- Has an account system that allows you to store your tree online to access from anywhere

CHAPTER 2: LITERATURE REVIEW

- Various colour branches, multiple columns and shapes available to avoid confusion and help visually differentiate between different groups
- Allows drag and drop of dataset onto webpage and automatic visualization

Limitations:

- Does not display whole genome shotgun metagenomics analysis results.
- Data visualization is only in 2 dimension.
- Not suitable for large size datasets.
- Huge display of data makes the displayed words hard to read, requiring user to further zoom in or hover over the part before being able to read them.

Proposed improvements:

- Allow larger dataset sizes by improving algorithm.
- Use 3 dimensional graph implementation to better display large amounts of data

SECTION 2.3- St. Jude PeCan Data Portal

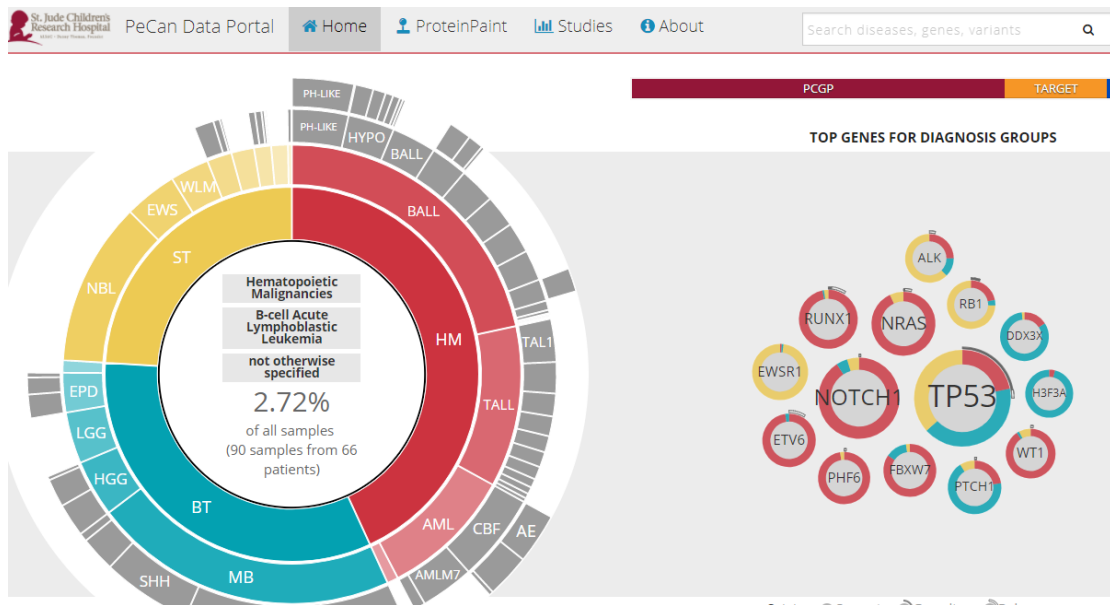


Figure 2.3.1 – St. Jude PeCan Data Portal home page

CHAPTER 2: LITERATURE REVIEW

St. Jude PeCan Data Portal is a Pediatric Cancer Data Portal that provides visualizations of cancer mutation in children in an interactive way. This site as of now has a total of 3310 samples, 3156 patients, 17 diagnoses, 12,520 genes and 35,077 validated somatic mutations collected worldwide with its top contributor from the United States. The aim of this project is to better understand the cause of childhood cancer and through this understanding find solutions to paediatric cancer. This site provides free data analysis, visualization tools and raw sequence data for all published results for nonclinical academic research use in hopes that someone could utilize it to make breakthrough discoveries. The visualization tool by this organisation is Protein Paint.

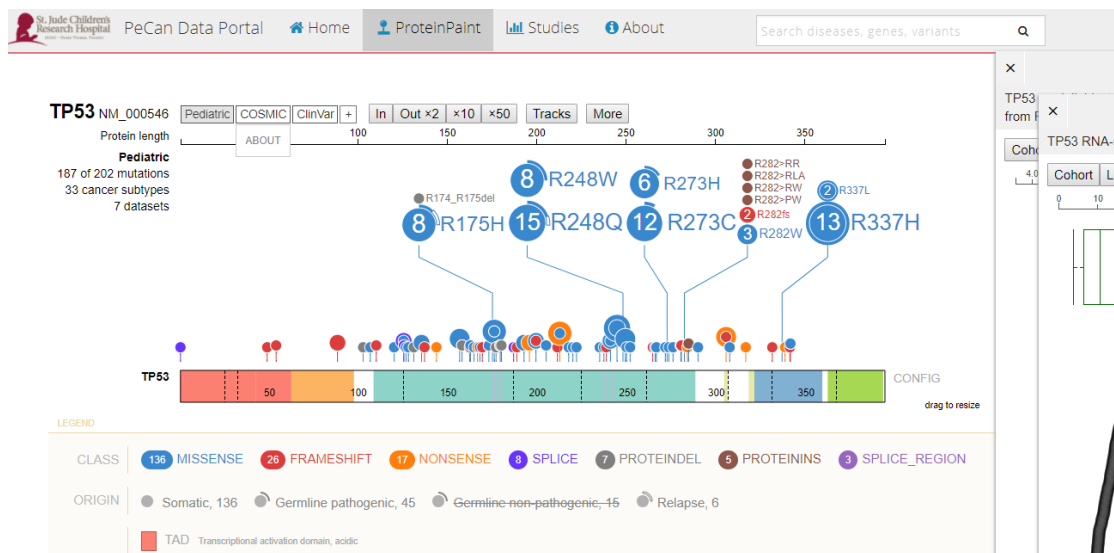


Figure 2.3.2 – St. Jude PeCan Data Portal Protein Paint of TP53

Figure 2.2.2 shows the interactive visualization of TP53. Hovering over the lower smaller circles displays some information about it and clicking on it expands it to the top part with a line linking it to the bottom. Once expanded, we can further click on the circles to display further breakdown of information on the mutations.

Strengths:

- Interactive and easy to use interface.
- Ability to show large amounts of data and still maintain suitable size font readability

CHAPTER 2: LITERATURE REVIEW

- Very easy comparison can be done at a glance
- Distinguishes between germline and pediatric mutations
- Visualizes cancer in children, unlike most tools that only focuses on adult cancer
- Allows comparison between adult cancer genomes and child cancer

Limitations:

- Does not display whole genome shotgun metagenomics analysis results.
- Suspension box sizes do not fit data displayed making it difficult to display 2 or more suspension boxes without overlapping.

Proposed improvements:

- Fit information in suspension boxes to size.
- Allow scale to resize to fit based on number of suspension boxes opened.

SECTION 2.4- IMG/M

The screenshot shows the IMG/M website interface. At the top, there is a navigation bar with the JGI logo and the text 'IMG/M INTEGRATED MICROBIAL GENOMES & MICROBIOME SAMPLES'. A search bar is present with a 'Go' button. Below the navigation bar, there are several menu items: Home, Find Genomes, Find Genes, Find Functions, Compare Genomes, OMICS, My IMG, Data Marts, and Help. The main content area is divided into two columns. The left column, titled 'IMG Content', lists various dataset types with counts for 'JGI' and 'All'. The right column contains a text description of the IMG system, a small image of a microorganism, and a table titled 'IMG Statistics' showing the number of 'Isolates', 'SAGs', and 'MAGs' sequenced at JGI and All. Below the table, there is a note about data sets with GOLD metadata and a section for 'Combined assembly data sets' with a table showing counts for 'Metagenome' and 'Metatranscriptome' across different categories like 'Engineered', 'Environmental', and 'Host-associated'.

Sequenced at:	Isolates		SAGs		MAGs	
	JGI	All	JGI	All	JGI	All
Bacteria	6059	47157	1813	2186	3854	4347
Archaea	206	772	186	282	79	226
Eukarya	76	266	0	0	1	1
Viruses	0	7854	0	44	0	0

CHAPTER 2: LITERATURE REVIEW

Figure 2.4.1 – Integrated Microbial Genomes and Microbial Samples –Joint Genome Institute Homepage

The Integrated Microbial Genomes system is a community resource for analysis of genomic and metagenomic datasets. Raw sequence data is always made available early before publication is done here because the organisation believes early release should help the progress in science. Some tools are also provided for users to analyse the datasets online. Some tools users can use to compare genome include:

- Synteny Viewers
 - VISTA
 - Dot Plots
 - Artemis ACT
- Phylogenetic Distributions
 - Metagenomes vs Genomes
 - Genomes vs Metagenomes
 - Radial Tree

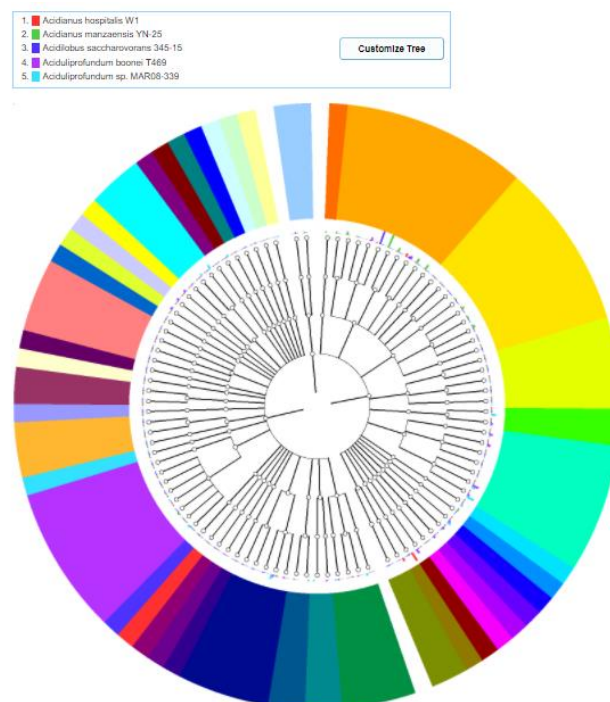


Figure 2.4.2 - Radial Tree distribution of Acidianus hospitalis W1, Acidianus manzaensis YN-25, Acidilobus saccharovorans 345-15, Aciduliprofundum boonei T469 and Aciduliprofundum sp. MAR08-339

CHAPTER 2: LITERATURE REVIEW

Based on Figure 2.4.2, users can select datasets to draw and it can be later edited by clicking the customize tree button but drawn graphs are not very interactive based. No links or additional more detailed information shown and no drag and drop function although export function is still available.

Strengths:

- Huge amount of dataset provided and constantly updated.
- Data stored online is private for 2 years before being publicly accessible.
- Provides user guides and forums for users who have questions.

Limitations:

- Does not display whole genome shotgun metagenomics analysis results.
- 2 account system because it is a joint project between two organisations may initially confuse new users.
- Only keep latest version of datasets while older versions are removed.
- Visualization graphs not very interactive based.

Proposed improvements:

- Allow previous versions of data to be stored.
- Make graphs more interactive.

SECTION 2.5 - EBI Metagenomics

EMBL-EBI Services Research Training About us

EBI Metagenomics Search

Home Search Submit data Projects Samples Comparison tool About Contact Not logged in Login

Submit, analyse, visualize and compare your data. SUBMIT DATA

109541 data sets

77780 amplicons
138 assemblies
1258 metabarcoding
15448 metagenomes
1091 metatranscriptomes

95748 runs
73612 public samples
1217 projects

6120 private runs
5908 private samples
162 private projects

Browse projects

Figure 2.5.1- EBI Metagenomics Home Page

CHAPTER 2: LITERATURE REVIEW

EBI metagenomics is a free site for analysing metagenomic data. It was first developed by the European Molecular Biology Laboratory, European Bioinformatics Institute for metagenomics analysis. Users can upload raw data and the automated pipeline will analyse and archive the metagenomic data for research purposes.

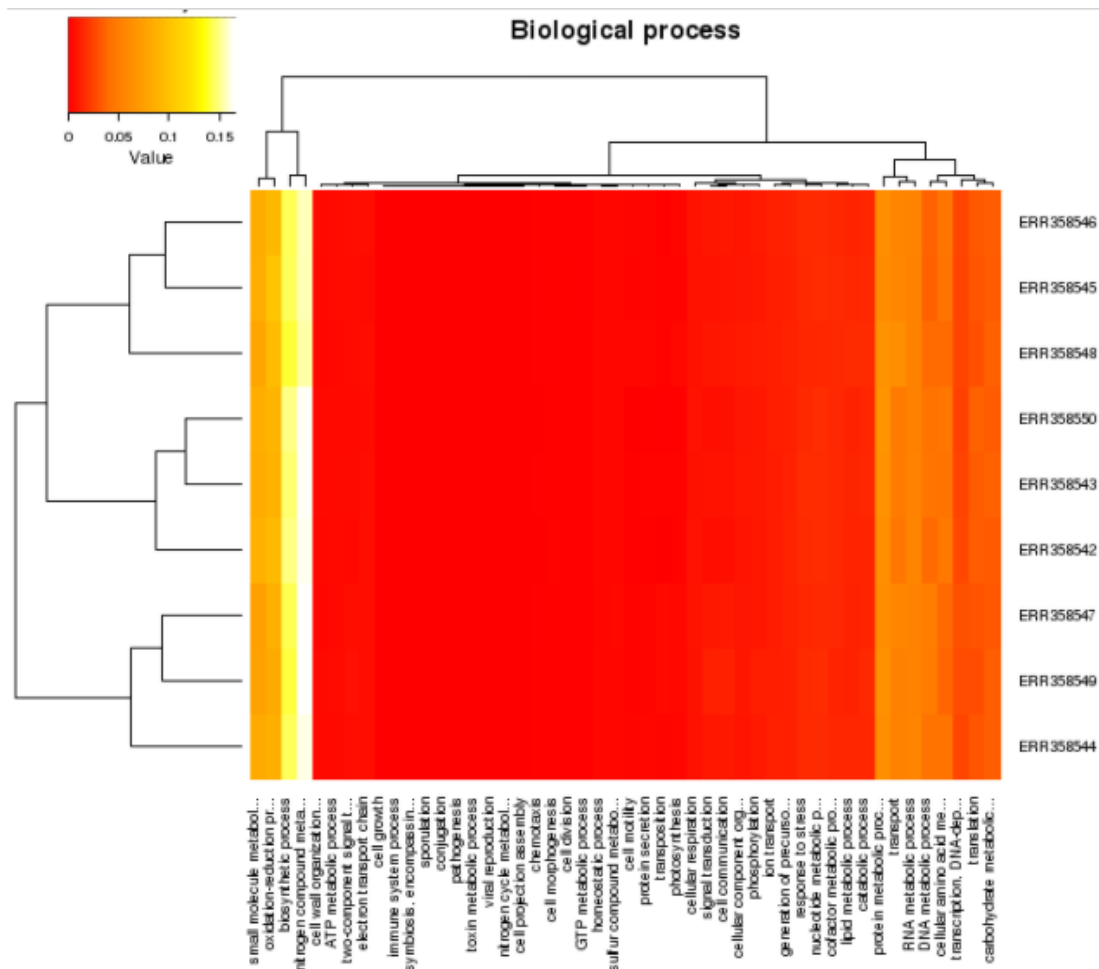


Figure 2.5.2 – Biological process heat map from Comparative freshwater metagenomics of Swedish and American lakes

CHAPTER 2: LITERATURE REVIEW

Based on Figure 2.5.2, users can select which runs in a project to compare and the results are displayed as barcharts, stacked columns, heat maps, principle component analysis and in table form. Figure 2.5.2 only displays the biological process heat map of the project. The molecular function and cellular component heat maps can also be displayed although the outputted images are not interactive at all.

Strengths:

- Huge amount of dataset provided and constantly updated with additional analysis and visualization tools.
- Data stored online is private for 2 years before being publicly accessible.
- Results are easily exported into various formats.
- Website tool easy to navigate and use

Limitations:

- Does not display whole genome shotgun metagenomics analysis results.
- Users must register to upload raw reads.
- Visualization graphs not interactive based at all.

Proposed improvements:

- Make graphs interactive.
- Allow unregistered users to upload.

CHAPTER 2: LITERATURE REVIEW

SECTION 2.6 – Final Verdict

Tools Features	Galaxy	Interactive Tree of Life	St. Jude PeCan Data Portal	IMG/M	EBI Metagenomics
Interactivity	Yes	Yes	Yes	Limited	-
Ease of use	Difficult	Average	Easy	Easy	Easy
Databases	Available (+ External)	Available	Available (Paediatric)	Available	Available
Displays whole genome shotgun metagenomics analysis results?	-	-	-	-	-
Allows exporting graphs?	Yes	Yes	Yes	Yes	Yes

Table 2.6.1 – Brief comparison

None of the reviewed online metagenomics tool provided whole genome shotgun metagenomics analysis results therefore this project is valid.

CHAPTER 3: SYSTEM DESIGN

CHAPTER 3: SYSTEM DESIGN

SECTION 3.1- DESIGN SPECIFICATIONS

3.1.1 Methodology

The methodology that is going to be used in the project is Phased Development. Phased Development is one of the two methods categorized under Rapid Application Development besides prototyping.

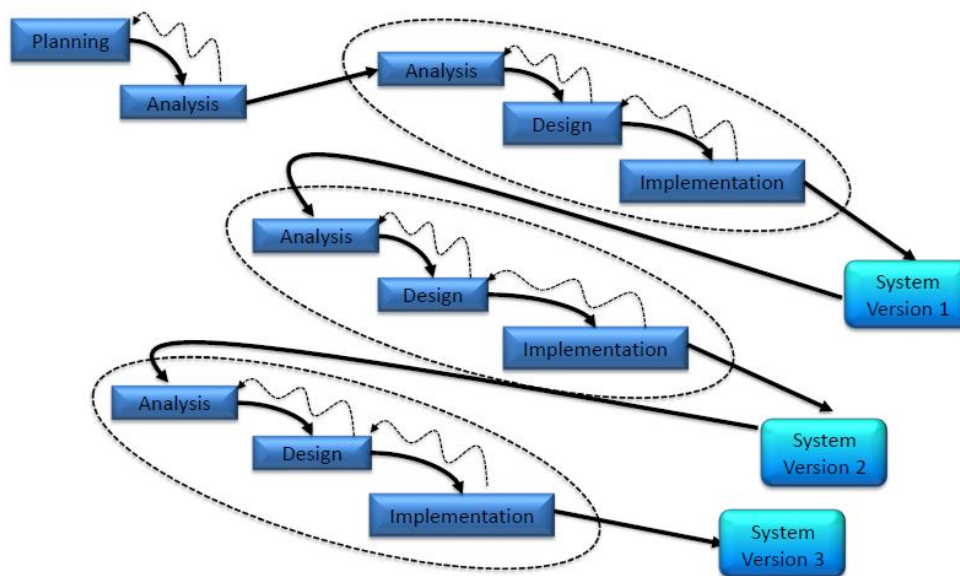


Figure 3.1.1- Phased Development (Ku, 2017)

This development method gets some part of the system quickly done so that system can be better understood to create what the user really needs. The overall system is broken into a series of versions that are developed sequentially and logically. The most important functions are built into the first version. Then later versions improve the first version. The final version is the complete system built.

The Planning phase gathers requirements from previously built online metagenomic tools that are somewhat similar to what this project intends to build. The strengths, limitations and ways to improve this online tool is listed out to understand what was done with similar tools and how we can learn from the available tools to not create the same shortcomings in our project.

CHAPTER 3: SYSTEM DESIGN

Then analysis, design and implementation is interleaved with many different incomplete system versions before the final version is completed.

3.1.2 Tools to use

Software

Visual Studio Code

- Source code editor with intellisense function.

Mozilla Firefox/Google Chrome

- Web browser used for testing and debugging purposes.

Dovirus framework

- Framework that the project is built on. Dependencies that the framework requires include:
 - ✓ Python - Python is an object-oriented, interactive, high-level programming language that is easy to use and integrates systems more effectively. It was created by Guido van Rossum and the source code is available under the GNU General Public License. Python/pip is used to run the project server.
 - ✓ Node/npm - NPM is a packet manager for Node.js packages which contains files needed for modules. Modules are javascript libraries that can be included in projects.
 - ✓ PostgreSQL - PostgreSQL is an object-relational advanced open source database management system developed at Berkeley Computer Science Department, University of California. It requires very minimum effort to maintain because of its stability and some of the features it supports includes user defined types, table inheritance, nested transactions and multi-version concurrency control. Used as the database for this project.
 - ✓ Git – version control system that is used to manage source code. It is used to clone the Dovirus framework.

CHAPTER 3: SYSTEM DESIGN

Hardware

Operating System	Ubuntu 16.04 LTS
CPU	Intel Core i5-4200U CPU @1.60GHz
GPU	NVIDIA GeForce 840M
RAM	4096 MB

3.1.3 General Work Procedures

In order to generate the visualization graphs, users are first required to upload data sets to be used as input. The graphs that we are going to visualize are some of the commonly graphs used in metagenomics publications. Overview reveals the overall results in the entire project. PCA provides dimensionality reduction for further analysis, comparison network clusters elements in order to find key biomarkers, and taxonomy tree reveals species profile relationships. The main purpose of these graphs is to identify the difference between case and control and to find biomarkers (such as some genus) to be further used in medical tests.

To draw the graph, languages that are used include TypeScript, SASS, d3.js and SVG. TypeScript is used because it is more object-oriented and can cope with the complicated data structures in bioinformatics, which would then significantly increase the efficacy of development. SASS is used because it is completely compatible with CSS and makes visualisation more interesting and the graphs more visually attractive and can highlight differences. d3.js is javascript's data visualisation library that can bind documents and keep them updated. It also operates SVG conveniently and provides common objects and components in visualization. SVG is basically scalable vector graphics which deals with XML vectors. Generally, construction of the graph is divided into 4 main parts which is the visualization of the drawing area, the interaction with the drawing area, the functional interaction of the sidebar and the

CHAPTER 3: SYSTEM DESIGN

backstage data processes. Using overview as an example, for the visualization part of overview graph, some aspects include the heat map, clustering tree, group information bar, composition bar and compared bar plot. The obtained information from data sets firstly has to be extracted and visualized as graphs drawn before any further steps can be taken.

Once visualisation is complete, we can then continue to the interaction with the drawing area. Still using the overview as example, we can then hover the mouse over the heat map or the composition bar to show detailed values and sample as well as genus information. Another interactive function includes hovering over the selected tree node and highlighting the entire branch.

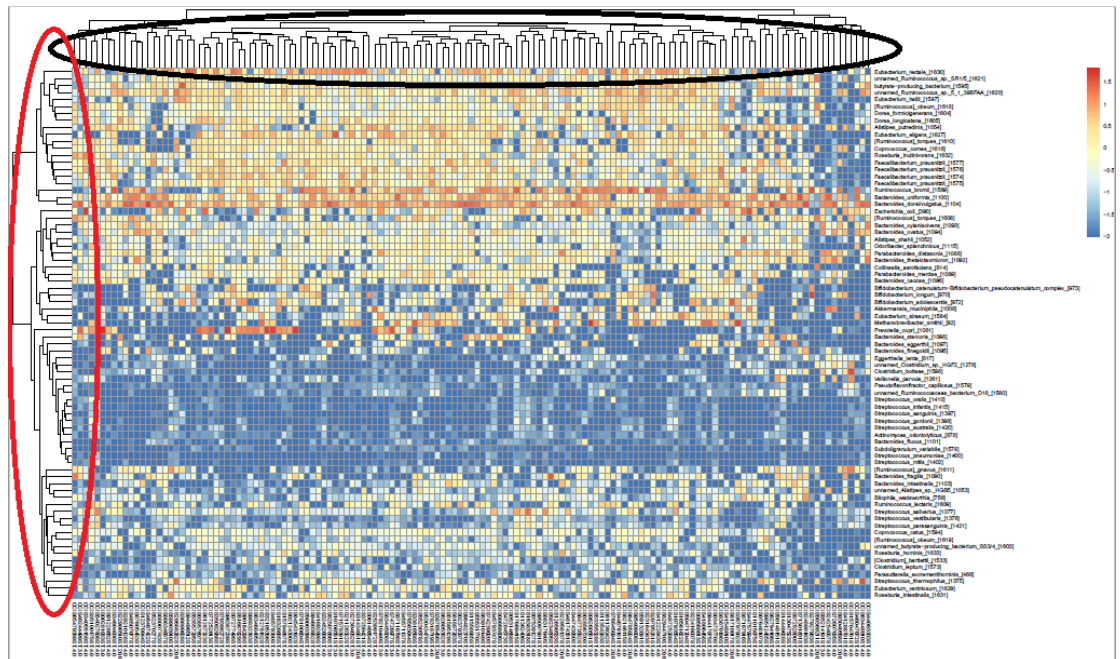


Figure 3.1.1 Sample heat map display

Figure 3.1.1 shows a sample heat map. On the right, which displays the rows are the list of genus whereas the bottom which are the columns are the sample clinic information. On a colour gradient scale of 2 to -2 with 2 being red and -2 being blue signifies the proportion of a genus in a particular sample. Using colour as indicator, we can see that the redder the colour displayed, the higher the proportion of this genus in that particular sample. Vice versa, the bluer the colour the lesser the proportion of this genus in that particular sample. The distance between every two samples/genus is calculated and a tree is constructed to represent their distance relationship as shown

CHAPTER 3: SYSTEM DESIGN

by the black and blue circled areas, the black displaying the tree for the samples and the blue displaying the tree for the genus.

After that is complete we can then move over to the functional interaction part which is the sidebar. From this sidebar we can adjust settings to the displayed graph such as changing the cut-off of the significant p-value which is by default 0.05. A low p-value means that this genus has a significant difference between the case and the control subjects. Table 3.2.1 below shows the Functional Interaction Sidebar for the overview graph denoting the aspects and its details.

Functional Interaction Sidebar	
Aspects	Details
Samples selection	① selection samples to display
Clinic info selection	① if user provide lots of clinic information, let them choose one to display group info bar

Table 3.2.1 – Functional Interaction Sidebar aspects and details

Finally, sometimes some backstage data processing is required to be done. Table 1.2.2 shows some sample data types that is used as input for the overview graph.

Input list:	Details
<u>Matrix of abundance</u>	Rows are genus, columns are samples, and data is relative abundance of every genus in every sample.
<u>Tree file</u>	Trees of samples & genus in Newick tree format file for columns and rows respectively.

Table 3.2.2 – Input list and how the information should be rendered

CHAPTER 3: SYSTEM DESIGN

SECTION 3.2- SYSTEM DESIGN/OVERVIEW

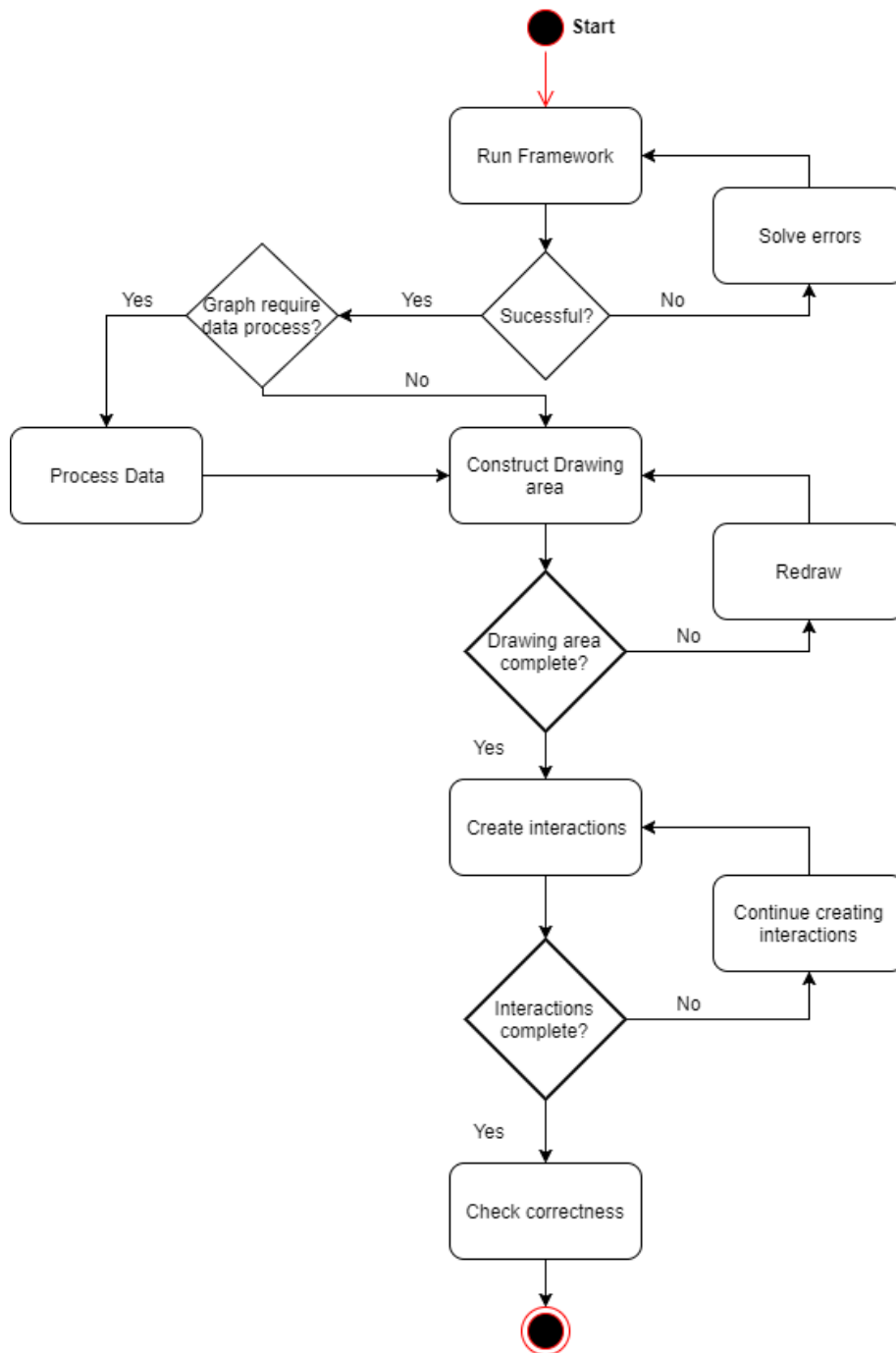


Figure 3.2.1 – Flowchart for development process

CHAPTER 3: SYSTEM DESIGN

Loading the framework

This framework runs on a Linux operating system (Ubuntu). Optionally, an ios operating system such as that on a MacBook can be used. Firstly Ubuntu is dual booted on the laptop alongside Windows. Then, dependencies have to be installed before the project can be cloned. In order to clone the project we have to install git. Git is installed using the terminal with the command:

```
$ sudo apt-get update
$ sudo apt-get install git
$ git --version
```

We check the version of git to ensure that we have downloaded it. Then, tabix is installed:

```
$ sudo apt-get update
$ sudo apt-get install tabix
```

Next, we have to install Python 2.7/pip as this is the version of python that is being used by the framework. Before we install them, we firstly have to install the dependencies that is required.

```
$ sudo apt-get install build-essential checkinstall
$ sudo apt-get install libreadline-gplv2-dev libncursesw5-dev libssl-dev
libsqlite3-dev tk-dev libgdbm-dev libc6-dev libbz2-dev
```

Then we download Python2.7 and extract the file before going over to the directory in which the file is located and install it.

```
$ version=2.7.13
$ cd ~/Downloads/
$ wget https://www.python.org/ftp/python/$version/Python-$version.tgz
$ tar -xvf Python-$version.tgz
$ cd Python-$version
$ ./configure
$ make
$ sudo checkinstall
```

CHAPTER 3: SYSTEM DESIGN

After installing Python2.7, we can then install pip which is a software management system for packages in Python and a virtual environment virtualenv.

```
$ sudo apt-get install python-pip python-dev build-essential
$ sudo pip install --upgrade pip
$ sudo pip install --upgrade virtualenv
```

We also have to install node, which is a server framework and npm which is a package manager for node.js

```
$ sudo apt-get update
$ sudo apt-get install curl
$ sudo apt-get install python-software-properties
$ curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
$ sudo apt-get install nodejs
$ node -v
$ npm -v
```

We type node -v and npm -v to check the versions to ensure they are the latest as needed by the framework. Finally, we need to install PostgreSQL which is the database we are going to be using for the project.

```
$ sudo apt-get update
$ sudo apt-get install postgresql postgresql-contrib
```

Once all these are installed, we first have to generate a ssh key to be sent to the person in charge of the dovirus framework before we can clone it. To generate an ssh key, we first check if a key exists.

```
$ cd ~/.ssh
```

If a key exists we type the following to list out the keys:

```
$ ls id_*
```

CHAPTER 3: SYSTEM DESIGN

Else, to generate a new key we type as below and press enter to set the default file and location as well as not put a passphrase

```
$ ssh-keygen -t rsa -C "your_email@example.com"
```

The key is now generated and to display the key we type:

```
$ $ cat < ~/.ssh/id_rsa.pub
```

Finally the cloning process can be done. Steps are as below.

1. Clone the project

- ✓ **git** clone git@git.lhc.moe:dovirus
- ✓ **cd** dovirus

The above commands clone's the project and moves over to the project directory

2. Install packages

- ✓ **pip** install -r requirements.txt
- ✓ **npm** install

Then the packages are installed using the above commands.

3. Set up database

- ✓ **sudo -u postgres createuser virus --createdb**
- ✓ **sudo -u postgres createdb -O virus virus_dev**
- ✓ **sudo -u postgres psql**
- ✓ **#ALTER USER virus WITH PASSWORD 'virus_test';**
- ✓ **python manage.py migrate**
- ✓ **python manage.py createsuperuser**

After that, the database is set up.

CHAPTER 3: SYSTEM DESIGN

4. Initialize the project

```
✓ echo -e "BVD3_ENABLE_SAMPLES =  
False\nBVD3_INDEX_PACKAGE = 'dovirus'" > bvd3/settings_bvd3.py
```

5. Run server

```
✓ python manage.py runserver  
✓ npm run watch
```

Run both these commands every time before launching the project

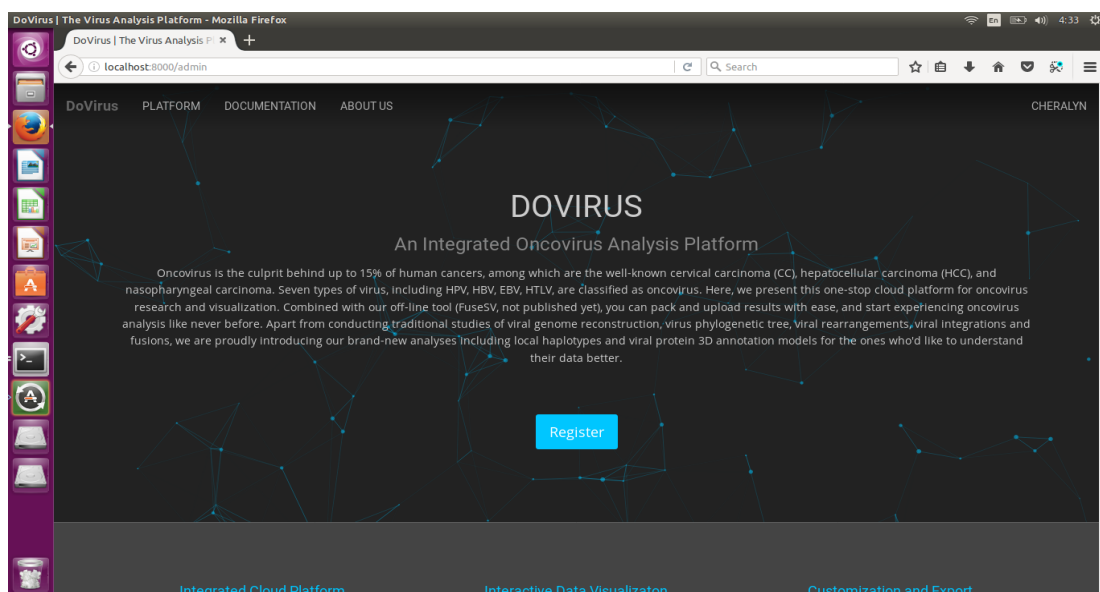


Figure 3.2.2 – Framework on localhost

Once the framework is successfully loaded, the database is managed at <http://localhost:8000/admin/>

CHAPTER 3: SYSTEM DESIGN

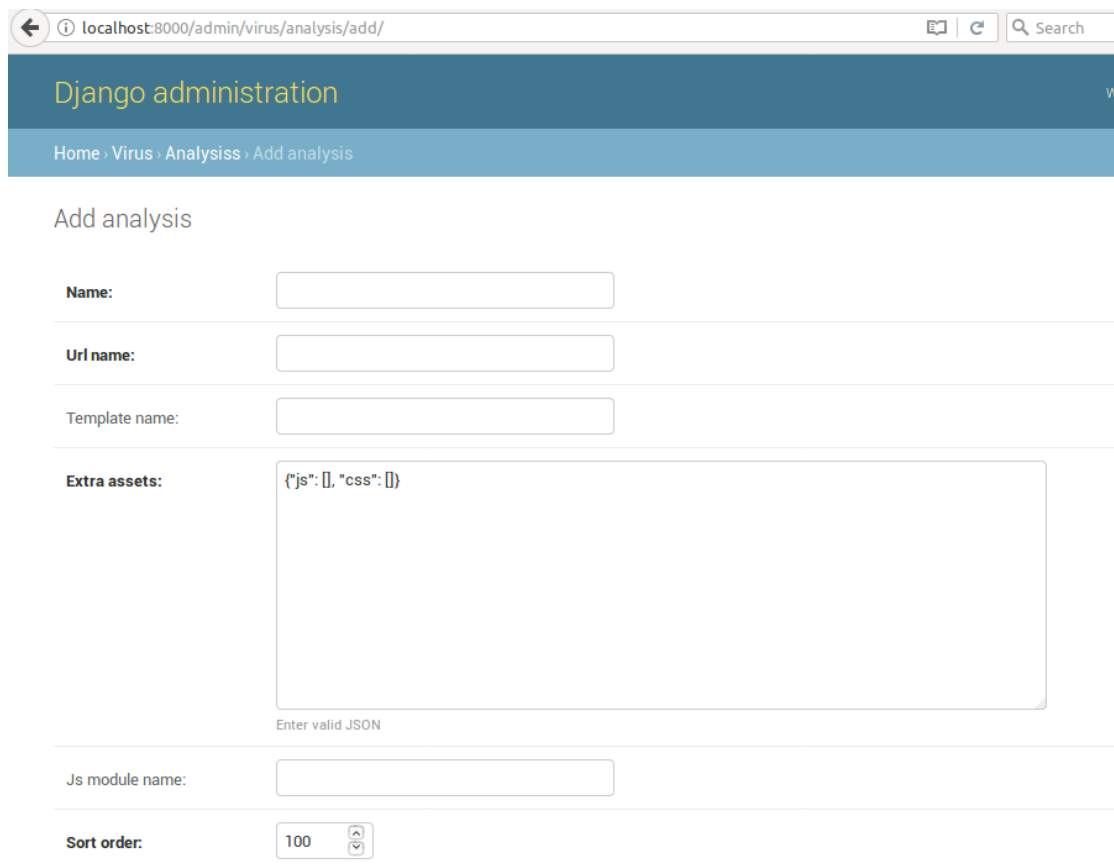
The screenshot shows a web application interface for site administration. At the top, there is a browser address bar with 'localhost:8000/admin/' and a search bar. The main content area is titled 'Site administration' and is divided into two main sections: 'AUTHENTICATION AND AUTHORIZATION' and 'VIRUS'. The 'AUTHENTICATION AND AUTHORIZATION' section includes 'Groups' and 'Users', each with '+ Add' and 'Change' buttons. The 'VIRUS' section includes various analysis categories such as 'Analysis categorys', 'Analysis', 'Fasta apis', 'File mount points', 'File relations', 'File upload keys', 'Memos', 'Notifications', 'Projects', 'Reg codes', 'Sample viruss', 'Samples', and 'Tabix apis', each with '+ Add' and 'Change' buttons. On the right side, there is a 'Recent actions' sidebar with a 'My actions' section. This section lists several actions, including '[Metagenomics] overview Analysis' (marked with a green plus), 'FYP Project' (marked with a green plus), 'Test Project' (marked with a red cross), and '[Metagenomics] overview Analysis' (marked with a red cross).

Figure 3.2.3 - administration page

Before any drawings can be done, we first have to create:

- Create Project
- Create Analysis Category
- Create Analysis

CHAPTER 3: SYSTEM DESIGN



The screenshot shows a web browser at the URL `localhost:8000/admin/virus/analysis/add/`. The page title is "Django administration" and the breadcrumb trail is "Home > Virus > Analysis > Add analysis". The main heading is "Add analysis".

The form contains the following fields:

- Name:** A text input field.
- Url name:** A text input field.
- Template name:** A text input field.
- Extra assets:** A large text area containing the JSON string `{"js": [], "css": []}`. Below the text area is the text "Enter valid JSON".
- Js module name:** A text input field.
- Sort order:** A numeric input field with the value "100" and a small icon for increment/decrement.

Figure 3.2.4 – add analysis page part 1

When creating an analysis, we have to input a name, a url name which just acts as the link name in the url section a template name and the Js module name. The js module name is very important as it has to be similar to the directory which stores all the files needed in the drawing process.

CHAPTER 3: SYSTEM DESIGN

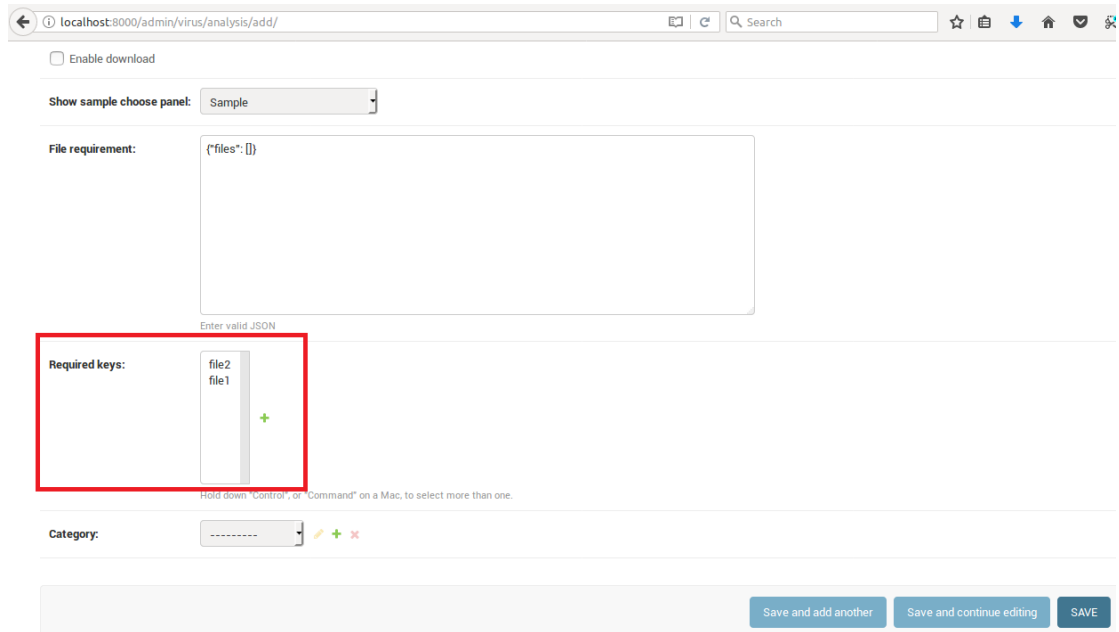


Figure 3.2.5 – add analysis page part 2

This is the continuation of the add analysis page which requires that a category be selected and the required keys selected. These required keys determines the number of data sets which are required to be uploaded before rendering the graphs.

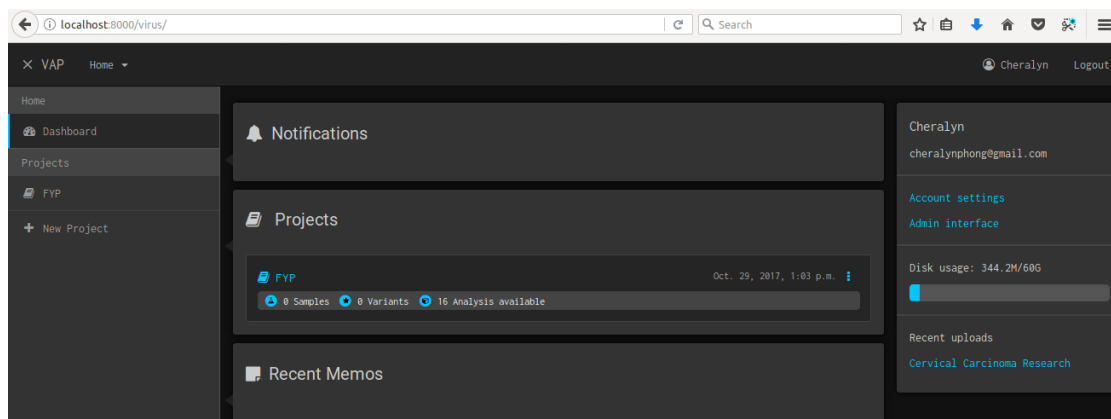


Figure 3.2.6 – Dovirus dashboard

This is the dashboard which shows all the projects a user has.

CHAPTER 3: SYSTEM DESIGN

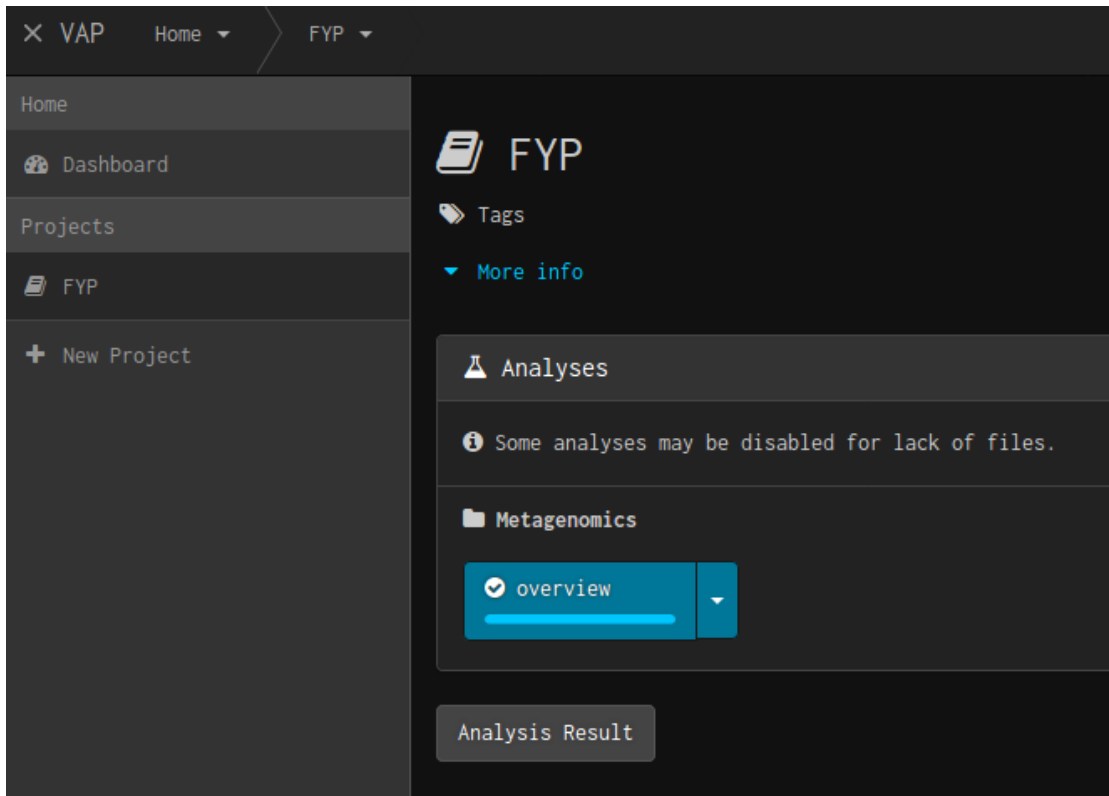


Figure 3.2.7 – Dovirus project page

This page displays the project contents which are the available analyses in a particular project and each analysis category.

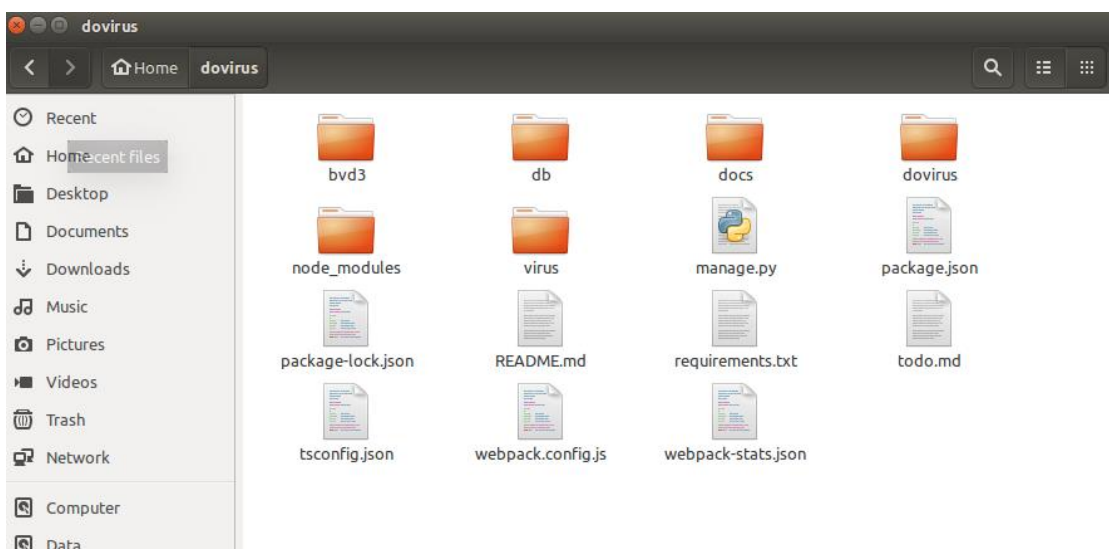


Figure 3.2.8 – Dovirus project repository

CHAPTER 3: SYSTEM DESIGN

From the above we can see that once the dovirus framework has been successfully set up, there will be 6 folders in it. The db folder stores the uploaded datasets of an analysis as shown in Figure 3.2.8 below.

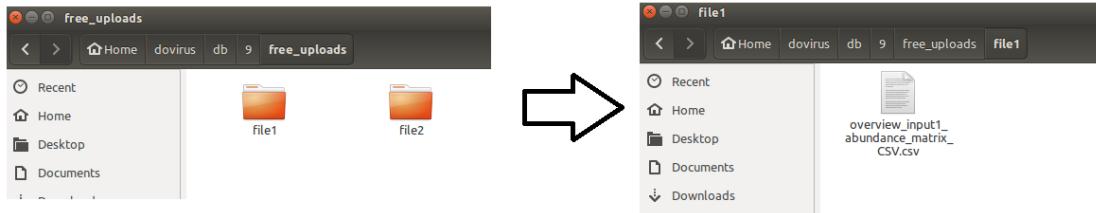


Figure 3.2.9 – Dovirus db folder contents

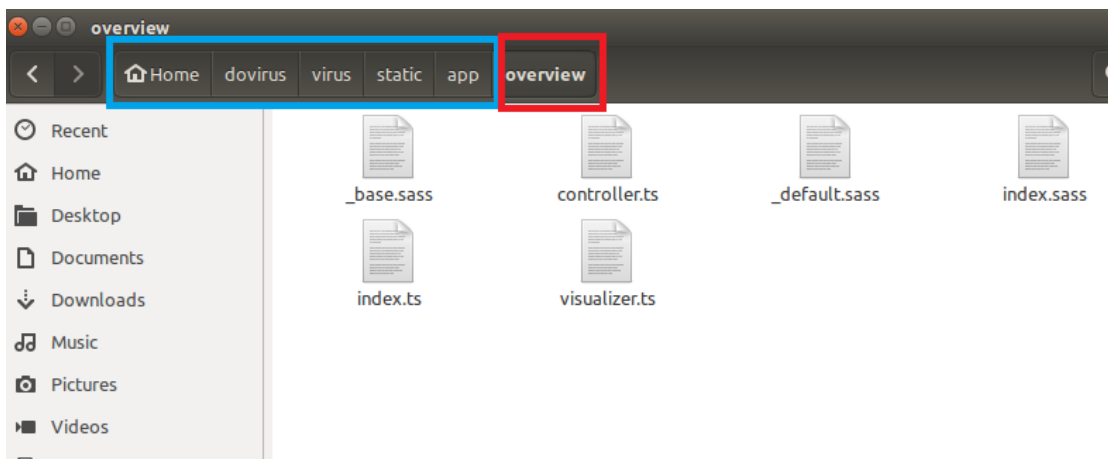


Figure 3.2.10 – Overview contents

The location in which our analysis is stored is in dovirus > virus > static > app.

Inside app, a folder is created which has to be similar in name to the name typed at Js module name as shown in Figure 3.2.4. Inside this folder there are 3 SASS and 3 TypeScript files which are used in the development of the graphs.

CHAPTER 3: SYSTEM DESIGN

```
Visual Studio Code
TS controller.ts TS visualizer.ts x
222     .attr("height", function(d) { return h; });
223     .style("fill", function(d) { return colorScale(d.value); });
224
225     });
226   }
227
228   public run(): void {
229     this.loadCSV("data1", (d) => {
230       this.data.data1 = d
231       console.log(this.data.data1);
232       //d["data"]
233     });
234   }
235   this.loadCSV("data2", (d) => {
236     this.data.data2 = d
237   });
238 }
239
240
241 public redraw(layers: string[] | string) {
242   this.isRedraw = true
243   this.size = new Size(this.option.svgWidth)
244   super.redraw(layers)
245 }
```

Figure 3.2.11 – Visualizer.ts

Inside this visualizer.ts file is where the data is loaded and visualization of the graphs are done. Code is written here to draw the graph based on the loaded datasets.

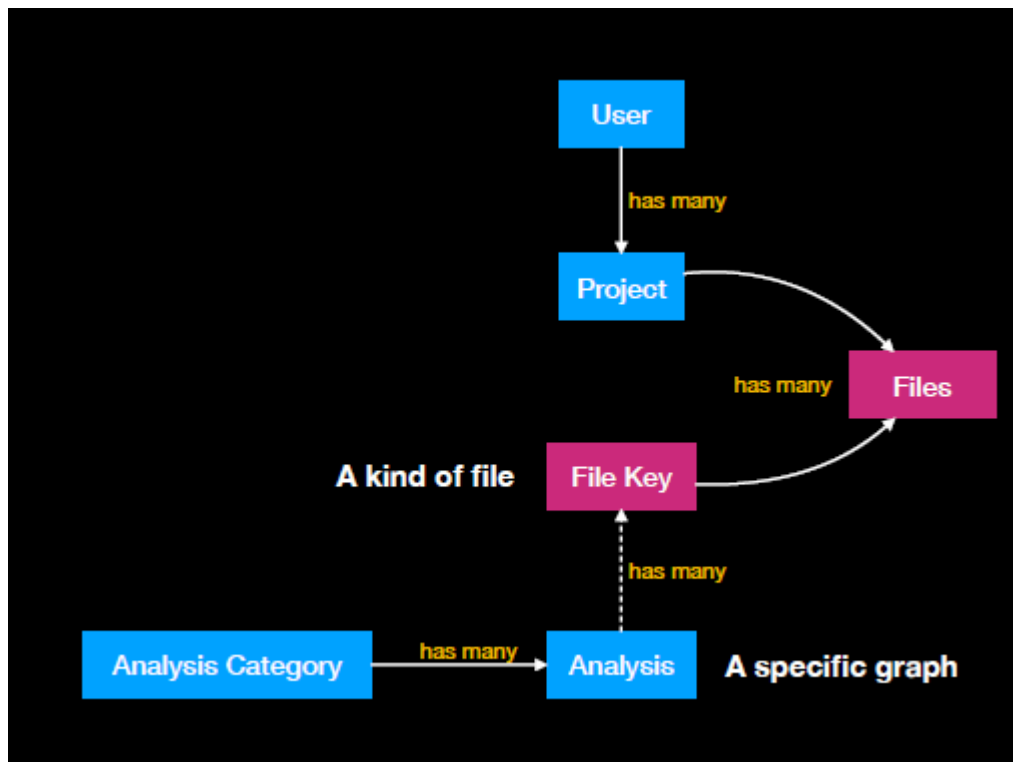


Figure 3.2.12 – The database model

CHAPTER 3: SYSTEM DESIGN

Once the framework is set up, the graphs can be constructed. Firstly we process the data if data process is required. Then we draw the graph as desired with the data given to visualize the large dataset. After the drawing part is complete, further interactions are added on to allow users to hover over a particular part of a graph to further display additional information. Graphs are written in TypeScript with SVG and d3.js

CHAPTER 4: DATA PROCESSING

CHAPTER 4: DATA PROCESSING

SECTION 4.1- OVERVIEW

```
static dataTypes = ["data1", "data2", "data3", "data4", "data5"];
static apiMap = {
    data1: { file_key: "file1" },
    data2: { file_key: "file2" },
    data3: { file_key: "file3" },
    data4: { file_key: "file4" },
    data5: { file_key: "file5" }
};
```

Figure 4.1.1 – Data mapping

Data is mapped from the file key to names created to be used such as data1.

```
this.loadCSV("data1", (d) => {
    this.data.data1 = d
})
this.loadCSV("data2", (d) => {
    this.data.data2 = d
})
this.loadCSV("data3", (d) => {
    this.data.data3 = d
})

this.loadText("data4", (d) => {
    this.data.data4 = Newick.parse(d);
})
this.loadText("data5", (d) => {
    this.data.data5 = Newick.parse(d);
})
```

Figure 4.1.2 – Data loading

Data is loaded. For the heat map, excel sheet where data is stored is converted to CSV which stands for comma separated values and then loaded using loadCSV. For the trees, data is loaded as text and then parsed into Newick format.

CHAPTER 4: DATA PROCESSING

```
interface Data {  
    data1: string[]  
    data2: string  
    data3: string  
    data4: NewickNode  
    data5: NewickNode  
  
    restructuredData1: abundance_matrix[]  
}  
  
class abundance_matrix {  
    genus: any  
    sample: string  
    value: any  
    col: number  
    row: number  
    constructor(genus: any, sample: string, value: any, col: number, row: number) {  
        this.genus = genus  
        this.sample = sample  
        this.value = value  
        this.col = col  
        this.row = row  
    }  
}
```

Figure 4.1.3 – Data handling for heat map (red) and tree (blue)

Then create an abundance matrix class with constructor to structure the object. Data from the string is formed into objects consisting of properties of each individual block in the overall overview heat map such as the row, column, genus, sample and value.

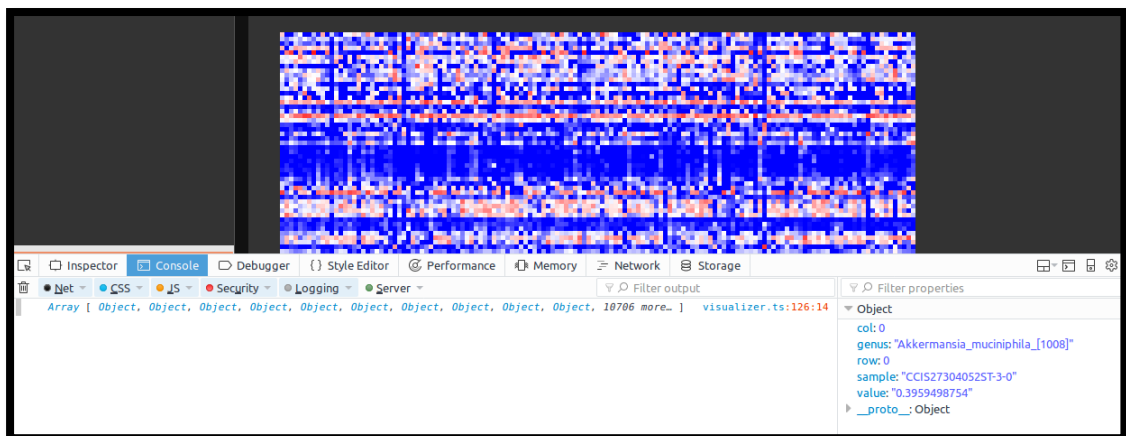


Figure 4.1.4 – Data as objects in an array (heat map)

Figure 4.1.4 shows data as objects with their properties expanded on the right and the array they are all put into.

CHAPTER 4: DATA PROCESSING

```
//heatmap
let names = $data.data1.map(function (i) {
    return Object.keys(i);
});
//array containing the object item values
let values = $data.data1.map(function (i) {
    return Object.keys(i).map(function (key) {
        return i[key];
    });
});

$data.restructuredData1 = new Array<abundance_matrix>();
for (let a = 0; a < names.length; a++) {
    for (let b = 1; b < names[0].length; b++) {
        $data.restructuredData1.push(new abundance_matrix(<string>values[a][0],
            names[0][b], <number>values[a][b], a, b - 1));
    }
}
```

Figure 4.1.5 – Creating restructured data (heat map)

Firstly, we extract data from data1 which was previously loaded as shown in figures above. We then extract these data as 2 separate lists one for the names and one for the values respectively. Then this two lists are used to create the objects which contain properties and inserted into an array of objects.

```
export interface NewickNode {
    name: string;
    pname: number;
    type: string;
    length?: number;
    branchset?: NewickNode[];
    radius?: number;
    radiusLog: number;
    color?: string;
    linkExtensionNode?: any;
    linkNode?: any;
    enabled: boolean;
}
```

Figure 4.1.6 – NewickNode interface

Create an interface for the NewickNode to be used with name, pname, type, length, branchset, radius, radiusLog, color, linkExtansionNode, and enabled.

CHAPTER 4: DATA PROCESSING

```
this.setupEditor({
  {
    title: "File",
    id: "file",
    layout: "tabbed-option-list",
    tabs: [
      {
        name: "Upload",
        items: [
          {
            type: "builtin",
            name: "file-upload",
            file_keys: ["file1", "file2", "file3", "file4", "file5"]
          },
        ]
      },
      {
        name: "Choose File",
        items: [
          {
            type: "builtin",
            name: "file-select",
            file_keys: ["file1", "file2", "file3", "file4", "file5"]
          }
        ]
      }
    ]
  }
},
],
},
```

Figure 4.1.7 – Setup editor

This is how the keys of the upload file are set in the controller.

```
protected fileDescription = {"file1": "Abundance Matrix", "file2": "Metadata",
"file3": "Statistic Result", "file4": "Tree Samples", "file5": "Tree Genus"}
```

Figure 4.1.8 – File Description

This is how the file descriptions are set so that user knows which file to upload to where in the controller.

CHAPTER 4: DATA PROCESSING

SECTION 4.2- TAXONOMY TREE

```
interface Data {  
    data1: string  
    data2: string  
}
```

Figure 4.2.1 – Data interface

Create an interface data with data1 and data2 as type string.

```
public run(): void {  
  
    this.loadText("data1", (d) => {  
        this.data.data1 = d;  
    })  
  
    this.loadText("data2", (d) => {  
        this.data.data2 = d;  
    })  
}
```

Figure 4.2.2– Data load

The csv files are then loaded as text and assigned.

These data are then in the form of csv:

```
Value,value,value  
Value,value,value  
Value,value,value
```

CHAPTER 4: DATA PROCESSING

```
var rows = $data.data1.split("\n");

var array = new Array(rows.length - 1);

var count;
var finalCount = 0;
for (var i = 0; i < rows.length - 1; i++) {
    count = rows[i].split(",");

    if (i == 0) {
        finalCount = count.length;
    }
    if (count.length > finalCount) {
        finalCount = count.length;
    }
}
```

Figure 4.2.3 – Data handling 1

Data is then split by “\n” which splits the data line by line. Then, line by line, data is split again by comma. We then get the largest number of data per row and store it inside final count.

```
for (var i = 0; i < rows.length - 1; i++) {
    var eachRow = rows[i].split(",");
    array[i] = new Array(finalCount);
    for (var j = 0; j < finalCount; j++) {
        array[i][j] = eachRow[j];
    }
}
```

Figure 4.2.4 – Data handling 2

A loop is then created which loops the data line by line. Inside each loop, the line is split by comma and inserted into an array of size finalCount which was created earlier.

```
var check = new Array(array.length * 4);
var num = 0;
for (var i = 0; i < array.length; i++) {
    for (var j = 0; j < array[0].length; j++) {
        check[num] = array[i][j] + " " + array[i][j - 1];
        num++;
    }
}
```

Figure 4.2.5 – Data handling 3

CHAPTER 4: DATA PROCESSING

In a loop, we then create a check array that stores the value of each element and the element in front of it.

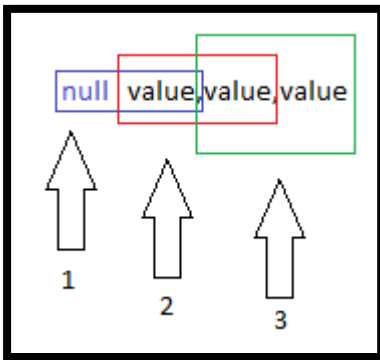


Figure 4.2.6 – Data handling visualization 1

Illustration of how the check array stores the values.

```
var flatData;
var checkCount = 1;
for (var i = 0; i < array.length; i++) {
  for (var j = 0; j < array[0].length; j++) {
    var current = array[i][j] + " " + array[i][j - 1];
    var flag = true;

    for (var k = 0; k < checkCount - 1; k++) {

      if (i == 0 && j == 0) { continue }
      if (current == check[k]) {
        flag = false;
        continue
      }
    }
    if (flag) {
      if (i == 0 && j == 0) {
        flatData = '[\n{"name": "' + array[i][j] + '", "parent": null }';
      }
      else if (j == 0) {
        continue
      } else {
        flatData = flatData + '{"name": "' + array[i][j]
          + '", "parent": "' + array[i][j - 1] + '" }';
      }

      if (i != array.length - 1) {
        flatData = flatData + ",\n";
      }
      if (i == array.length - 1 && j != array[i].length - 1) {
        flatData = flatData + ",\n";
      }
    }
    checkCount++;
  }
}
flatData = flatData + "\n]";
var jsonFlatData = JSON.parse(flatData);
```

Figure 4.2.7 – Data handling 4

CHAPTER 4: DATA PROCESSING

A JSON structure is created using all that was created previously, and because there might be duplicate values in the data, we use the check array and compare to ensure no repetition of data occurs. If there is no repetition then it shall be appended until the very end. Upon completion the JSON data that was created is then parsed.

```
// convert the flat data into a hierarchy
var treeData = d3.stratify()
  .id(function (d) { return d['name']; })
  .parentId(function (d) { return d['parent']; })
  (jsonFlatData);
```

Figure 4.2.8– Data handling 5

The flat data that was created from the JSON parse is then converted into a hierarchy.

```
this.setupEditor([
  {
    title: "File",
    id: "file",
    layout: "tabbed-option-list",
    tabs: [
      {
        name: "Upload",
        items: [
          {
            type: "builtin",
            name: "file-upload",
            file_keys: ["taxtree_tree", "taxtree_species"]
          }
        ]
      },
      {
        name: "Choose File",
        items: [
          {
            type: "builtin",
            name: "file-select",
            file_keys: ["taxtree_tree", "taxtree_species"]
          }
        ]
      }
    ]
  }
],
```

Figure 4.2.9 – Data handling 6

This is how the keys of the upload file are set in the controller.

CHAPTER 4: DATA PROCESSING

```
protected fileDescription =  
{ "taxtree_tree": "Tree Data", "taxtree_species": "Species" }
```

Figure 4.2.10 – Data handling 7

This is how the file descriptions are set so that user knows which file to upload to where in the controller.

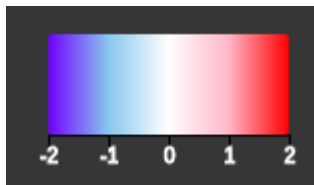
CHAPTER 5: DRAWING AND INTERACTION

CHAPTER 5: DRAWING AND INTERACTION

SECTION 5.1- OVERVIEW

5.1.1 Heat Map

Consists of:



Colour Gradient Legend

Figure 5.1.1 – Color Gradient Legend



Figure 5.1.2 – Horizontal Tree

Horizontal Tree



Vertical Tree

Figure 5.1.3 – Vertical Tree

CHAPTER 5: DRAWING AND INTERACTION

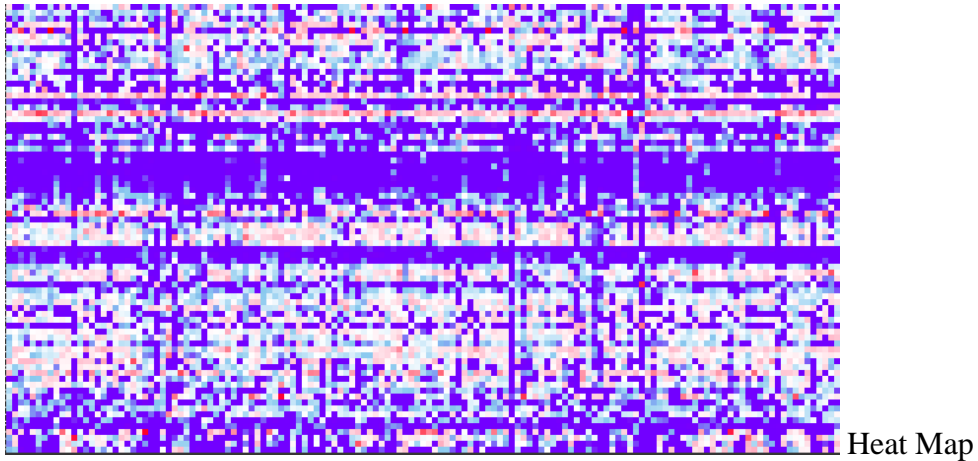


Figure 5.1.4 – Heat Map

Figure 5.1.5 below shows the heat map part of the overview interface drawn using TypeScript and d3.js on the framework.

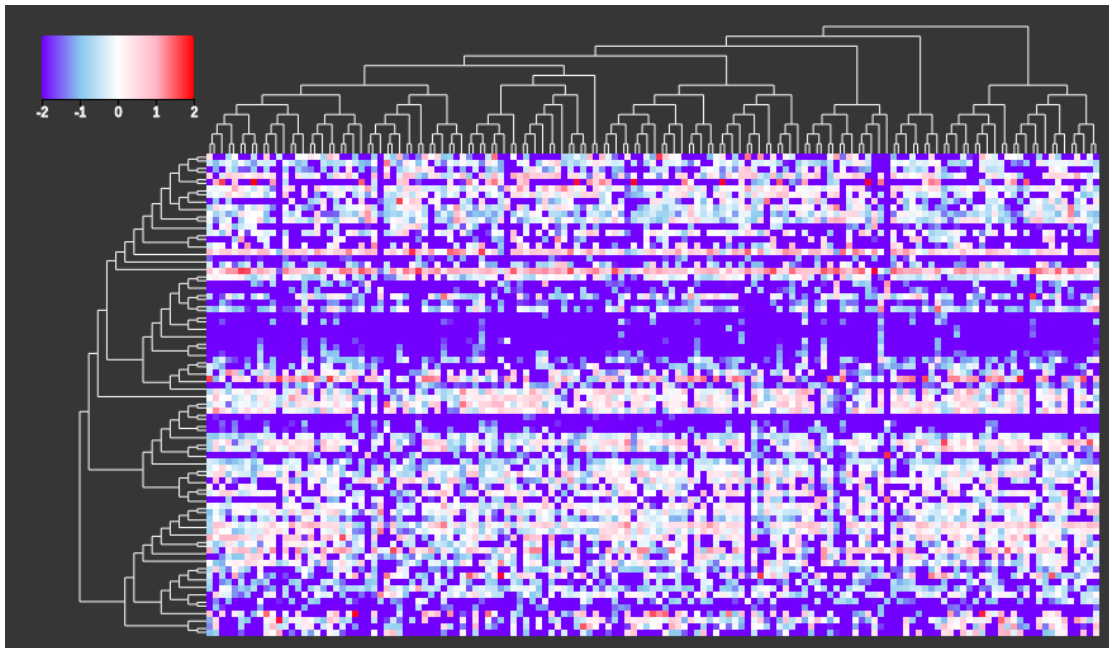


Figure 5.1.5 – Overview heat map

To better understand this heat map, a colour gradient scale of 2 to -2 is used. 2 will be represented by the colour red, approaching 0 the colour used to represent 0 is white and blue is used to signify the value of -2. Using colour as an indicator to better visualize, we can deduce the proportion of a genus in a particular sample. The higher the proportion of a genus in a sample the redder the colour indicated in the heat map.

CHAPTER 5: DRAWING AND INTERACTION

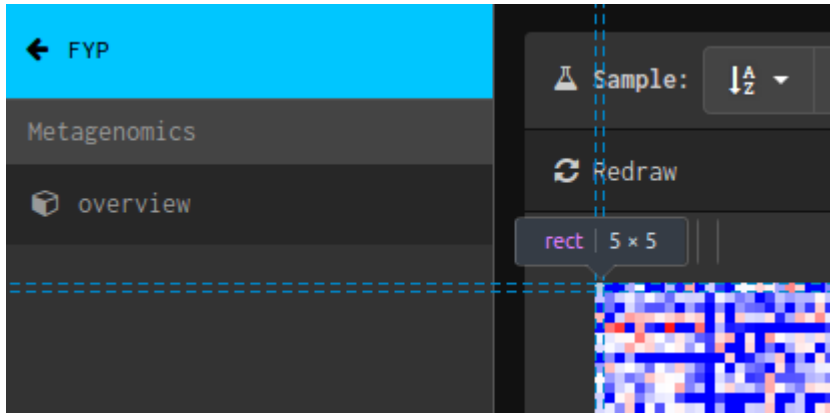


Figure 5.1.6 – Rect in the heat map

Rect is a rectangle which is a type of shape predefined by d3.js. Each rect represents an object that was created and the colour of it is determined by its value in the object.

```
var colorScale = d3.scaleLinear<string>()
  .domain([-2, -1, 0, 1, 2])
  .range(["#0000ff", "#87ceeb", "#ffffff", "#ffc0cb", "#ff0000"]);
```

Figure 5.1.7 – Color Scale

Setting the domain and range of the colour scale. A colour scale is then created for the purpose of differentiating each rect by its respective value. A domain of [-2, -1, 0, 1, 2] is mapped to a range of [blue, light blue white, pink, red]. Grid size, margin and rectPadding values are then set.

CHAPTER 5: DRAWING AND INTERACTION

```
//Append a defs (for definition) element to your SVG
var defs = this.append("defs");

//Append a linearGradient element to the defs and give it a unique id
var linearGradient = defs.append("linearGradient")
    .attr("id", "linear-gradient");

linearGradient.selectAll("stop")
    .data(colorScale.range())
    .enter().append("stop")
    .attr("offset", function (d, i) { return i / (colorScale.range().length - 1); })
    .attr("stop-color", function (d) { return d; });

group.appendElement(Rect).options({
    position: [30, 30],
    width: 120,
    height: 50,
    cornerRadius: 2,
    fill: "url(#linear-gradient)",
}).render()

var x = d3.scaleLinear().domain([-2, 2])
    .range([0, 120]);
var xAxis = d3.axisBottom(x)
    .scale(x)
    .ticks(5);

this.append('g')
    .attr('class', 'x axis')
    .attr('transform', 'translate(-30,' + (55) + ')')
    .call(xAxis)
    .selectAll('text')
    .style('text-anchor', 'middle')
    .attr('dy', '.5em')
```

Figure 5.1.8– Drawing part 1

Creating the legend which is a color gradient scale that shows how the color gradient works on each rect depending on the value of each rect.

CHAPTER 5: DRAWING AND INTERACTION

```
let heatmap = this.selectAll(".heatmap")
    .data($data.restructuredData)
    .enter().append("g").append("svg:rect") //Append by using D3's data/enter step
    .attr("x", function (d) { return d.row * w; })
    .attr("y", function (d) { return d.col * h; })
    .attr("width", function (d) { return w; })
    .attr("height", function (d) { return h; })
    .attr("genus", function (d) { return d.genus; })
    .attr("value", function (d) { return d.value; })
    .attr("sample", function (d) { return d.sample; })
    .style("fill", function (d) { return colorScale(d.value); })
    .attr("borderWidth", function (d) { return "1px" })
    .attr("borderColor", function (d) { return "black" })
    //.style("margin", function (d) { return "1px black" })
    .attr("transform", (d, i) => tool.translate(100, 100))
    .on("mouseover",
        function (d) {
            var tipText = "<text >Value:"
                + d.value + "<br>Genus: " + d.genus
                + "<br>Sample: " + d.sample + "</text>";

            tooltip.html(tipText)
                .style("top", (d3.event.pageY - 200) + "px")
                .style("left", (d3.event.pageX - 250) + "px")
                .style("position", "absolute")
                .style("width", "300px")
                .style("padding", "5px")
                .style("color", "black")
                .style("background", "rgb(240, 223, 151)")
                .style("border", "0px")
                .style("border-radius", "3px")
                .style("z-index", 100)
            return tooltip.style("visibility", "visible");
        })
    .on("mouseout",
        function () {
            return tooltip.style('visibility', 'hidden');
        });
};
```

Figure 5.1.9– Drawing part 2

Svg with the set attributes are then appended to the heat map. After that we append the array of objects to a non-existent heat map class and use d3's data/enter step to inflate the heat map with the objects in the array. We then create a mouseover function which displays the information of the value, genus and sample when mouse is over a rect. The mouseout function hides the information when the mouse leaves the rect.

CHAPTER 5: DRAWING AND INTERACTION

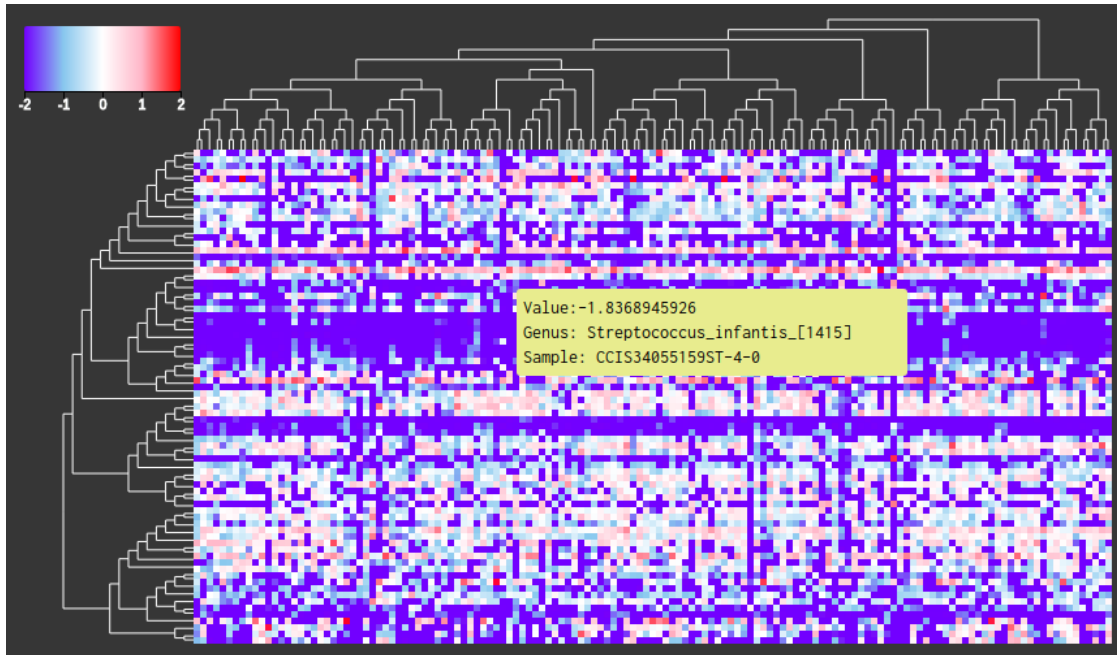


Figure 5.1.10 – Example of Hover function interactivity

CHAPTER 5: DRAWING AND INTERACTION

5.1.2 Tree

To draw the tree, we will use information that has been parsed into Newick format.

```
// declares a tree layout and assigns the size
var treemap = d3.cluster()
  .size([705, 100]);
// assigns the data to a hierarchy using parent-child relationships
var root = d3.hierarchy($data.data4, function (d) { return d.branchset; })
  .sum(function (d) { return d.branchset ? 0 : 1; })

// maps the node data to the tree layout
var nodes = treemap(root);

// append the svg object to the body of the page
// appends a 'group' element to 'svg'
// moves the 'group' element to the top left margin
var g = this.append("g")
  .attr("transform",
    "translate(" + 100 + "," + 0 + ")");
```

Figure 5.1.11 – Drawing part 3

First we declare a tree layout and assign a size to it. Then we assign the data to a hierarchy using parent-child relationship. Hierarchy is d3's built in structure for hierarchy layouts. JSON is an example of a hierarchical structure. We will need a root node to start before the rest of the hierarchical structure can be computed. After data has been assigned, the node data is then mapped to the tree layout.

CHAPTER 5: DRAWING AND INTERACTION

```
// adds the links between the nodes
var link = g.selectAll(".link")
  .data(nodes.descendants().slice(1))
  .enter().append("path")
  .attr("fill", "none")
  .attr("stroke", "white")
  .attr("class", "link")
  .attr("d", elbow);

function elbow(d, i) {
  return "M" + d.x + "," + d.y
    + "V" + d.parent.y + "H" + d.parent.x;
}

// adds each node as a group
var node = g.selectAll(".node")
  .data(nodes.descendants())
  .enter().append("g")
  .attr("class", function (d) {
    return "node" +
      (d["branchset"] ? " node--internal" : " node--leaf");
  })
  .attr("transform", function (d) {
    return "translate(" + d.x + "," + d.y + ")";
  });
```

Figure 5.1.12 – Drawing part 4

After that, we add links between the nodes. To add the links, we need to draw paths between the nodes using SVG Path Mini-Language. To draw the path from one node to another node we first have to start with M (moveto) that if we equate to drawing with pencil and paper would mean to put the pen down at this spot. In our case, we put the pen down at the child node. All paths have to begin with M. We then continue with V (vertical lineto) which draws a vertical line from the child node up to the parent node's y point. From this point we draw a H (horizontal lineto) to the parent node's x point. This is done from leaf nodes all the way up to the root node. Each node is then added as a group and rendered on the screen.

CHAPTER 5: DRAWING AND INTERACTION

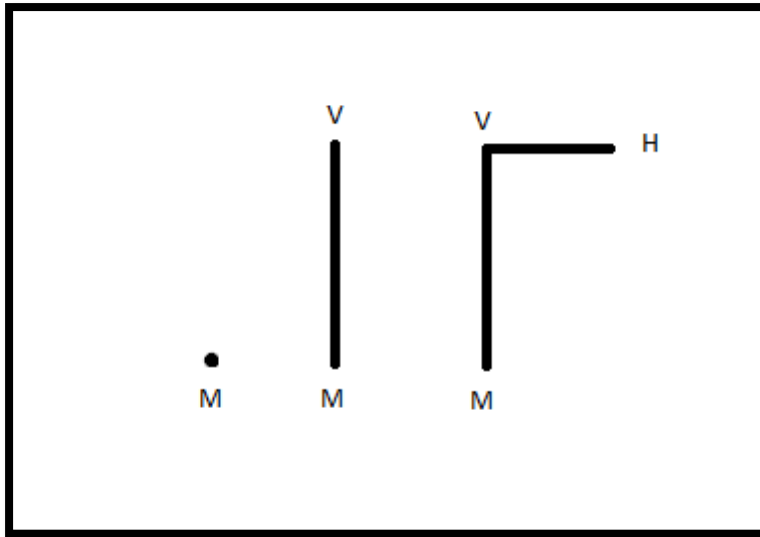


Figure 5.1.13 – Visualization of SVG Path Mini-Language

Visualization of how M, V and H works in path data.

CHAPTER 5: DRAWING AND INTERACTION

SECTION 5.2- TAXONOMY TREE

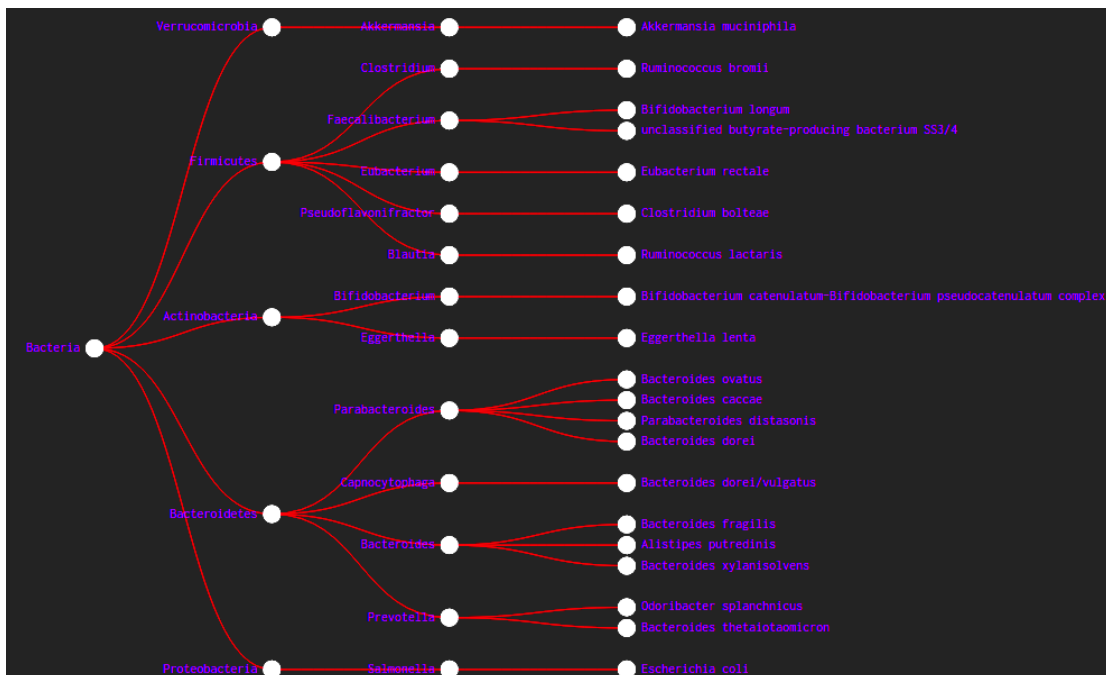


Figure 5.2.1 Taxonomy Tree

```
// assign the name to each node
treeData.each(function (d) {
  .....
  d['name'] = d.id;
});

// set the dimensions and margins of the diagram
var margin = { top: 5, right: 5, bottom: 5, left: 20 },
    width = 500 - margin.left - margin.right,
    .....
    height = 600 - margin.top - margin.bottom;

// declares a tree layout and assigns the size
var treemap = d3.tree()
  .....
  .size([height, width]);

// assigns the data to a hierarchy using parent-child relationships
var nodes = d3.hierarchy(treeData, function (d) {
  .....
  return d.children;
});

// maps the node data to the tree layout
treemap(nodes);
```

Figure 5.2.2 – Drawing part 1

CHAPTER 5: DRAWING AND INTERACTION

To draw the taxonomy tree, first we assign the name to each node. Then set the dimensions and margins of the diagram. Then, like all tree structures, we declare a tree layout and assign the size. Using parent-child relationship, the data is assigned to a hierarchy structure. The hierarchy structure is finally mapped to the tree layout.

```
// append the svg object to the body of the page
// appends a 'group' element to 'svg'
// moves the 'group' element to the top left margin
var svg = d3.select("body .svg-container">//div#canvas
.append("g")
  .attr("width", width + margin.left + margin.right)
  .attr("height", height + margin.top + margin.bottom)
  ,
  g = this.append("g")
  .attr("class", "whole")
  .attr("transform",
    "translate(" + margin.left + "," + margin.top + ")");
```

Figure 5.2.3 – Drawing part 2

We then append the svg object to the body of the page and set the margins, etc.

```
// adds the links between the nodes
var link = g.selectAll(".link")
  .data(nodes.descendants().slice(1))
  .enter().append("path")
  .attr("fill", "none")
  .attr("stroke", "white")
  .attr("class", "link")
  .attr("d", function (d) {
    return "M" + d['y'] + "," + d['x']
      + "C" + ( d['y'] + d.parent['y']) / 2 + "," + d['x']
      + " " + ( d['y'] + d.parent['y']) / 2 + "," + d.parent['x']
      + " " + d.parent['y'] + "," + d.parent['x'];
  });
```

Figure 5.2.4 – Drawing part 3

To draw links between the nodes, path is used using SVG Path Mini-Language. To draw the path from one node to another node we first have to start with M (moveto) that if we equate to drawing with pencil and paper would mean to put the pen down at this spot. In our case, we put the pen down at the child node. All paths have to begin with M. Then we draw a C (curveto) which is a cubic Bézier curve to the parent

CHAPTER 5: DRAWING AND INTERACTION

coordinates using $(d['y'] + d.parent['y']) / 2$ and $d['x']$ as the control point at the start of the curve and $(d['y'] + d.parent['y']) / 2$ and $d.parent['x']$ as the control point towards the end of the curve. This is done from leaf nodes all the way up to the root node.

```
// adds each node as a group
var node = g.selectAll(".node")
    .data(nodes.descendants())
    .enter().append("g")
    .attr("class", function (d) {
        return "node" +
            (d.children ? " node--internal" : " node--leaf");
    })
    .attr("transform", function (d) {
        return "translate(" + d['y'] + "," + d['x'] + ")";
    });

// adds the circle to the node
node.append("circle")
    .attr("r", 10);

// adds the text to the node
node.append("text")
    .attr("dy", ".25em")//.25em
    .attr("x", function (d) { return d.children ? -13 : 13; })
    .attr("font-size", "10px")
    .style("text-anchor", function (d) {
        return d.children ? "end" : "start";
    })
    .text(function (d) { return d.data['name']; });
```

Figure 5.2.5 – Drawing part 4

Each node is then added as a group and a circle is appended to each node to represent the position of the node. Then text is appended to the node to identify each node.

CHAPTER 5: DRAWING AND INTERACTION

```
node.on("mouseover", function (p) {

    // Select all descendants
    node.filter(function (d) {
        while (d = d.parent) {
            if (d === p) return true;
        }
    }).transition().duration(300).style("opacity", 0.2);

    // Select links that are connected to descendants
    link.filter(function (d) {
        for (d['children']; d; d = d["parent"]) {
            if (d === p) {
                return true;
            }
        }
    }).transition().duration(300).style("opacity", 0.2);

    // How to select all parents leading to path to center
    console.log('Well this returns the parent: ', p.parent);

}).on("mouseout", function (p) {

    node.transition().duration(300).style("opacity", 1);
    link.transition().duration(300).style("opacity", 1);

});
```

Figure 5.2.6 – Taxonomy Tree interaction

This is the taxonomy tree interaction whereby the mouseovered node's entire children branch is made slightly less opaque to highlight that these are its descendants.

CHAPTER 5: DRAWING AND INTERACTION

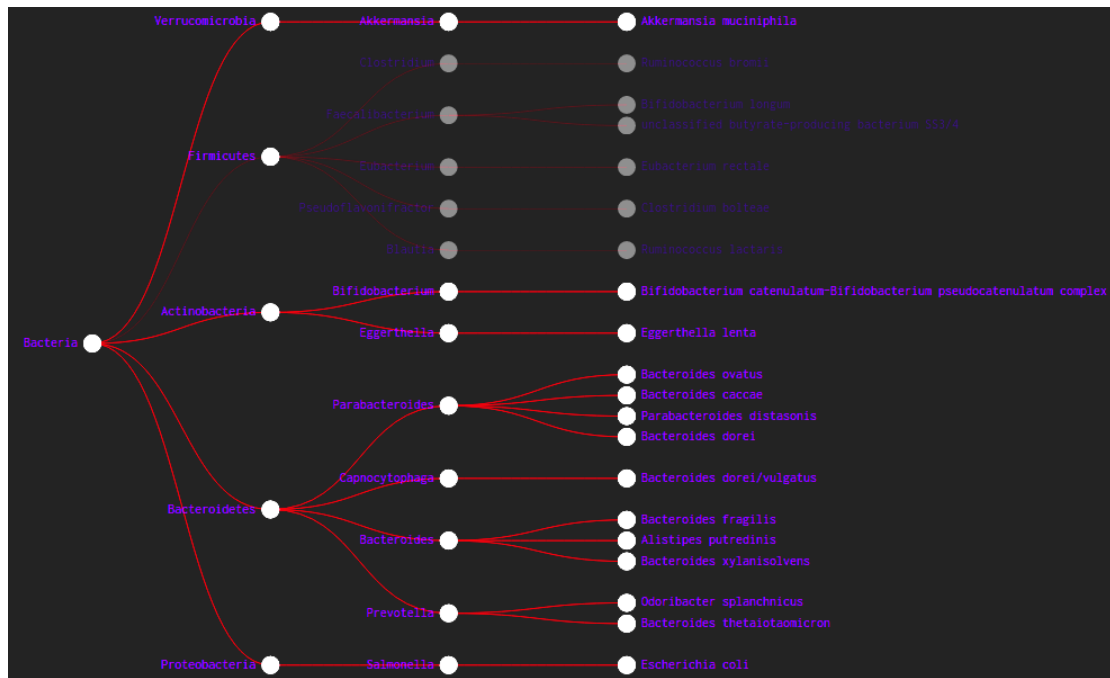


Figure 5.2.7 – Example Taxonomy Tree interaction

The figure above shows how the interactivity works. When the firmicutes node is hovered over, all its descendants are made slightly less opaque to highlight that all these nodes belong under this category and are considered as its descendants.

CHAPTER 6: CONCLUSION

CHAPTER 6: CONCLUSION

In conclusion, this project created an easily accessible tool for researchers and students alike to visualize metagenomics analyses. There are existing websites that do metagenomics analysis online but currently, there isn't one to display whole genome shotgun metagenomics analysis results. The objective of this project is to provide one such tool where complete analysis and visualization of inputted data can be achieved. This tool provides a drawing area, interactions and data processing. As metagenomics will help propel and increase understanding of how microbiomes interact and affect our lives, this project will indirectly help in the future development of healthcare and disease control as well as the medical field.

Some of the implementation issues and challenges faced was that because the framework is a large complex program, precaution had to be taken to carefully load it properly to prevent loading an incomplete, incorrect framework. Loading the framework required dependencies and these dependencies had to be loaded in the correct directories before everything can run smoothly and correctly. Any error in the loading process in turn would later on cause problems.

Another problem faced is the problem of debugging and error checking. Because this project is an online tool, the most viable way to run and check is to use a browser to preview the graphs. Unfortunately, web browsers are not the best debugging tools available as a web browser's main purpose is to render web pages and not for error checking.

There is also the issue of lack of d3 and TypeScript documentation. Javascript documentations are plentiful but TypeScript documentation for reference purposes are hard to come by and the information provided are vague. There are also not much examples online that could serve as references.

Another problem faced is the problem of syntax due to version upgrade. Whatever d3 documentation that is available on the web are in version 3 whereas the version that this system is built on is in version 4. A lot of time is spent on fixing minor bugs that occur due to the version difference. Developer is also unfamiliar and

CHAPTER 6: CONCLUSION

unable to find sufficient documentation as reference and therefore slowed down development time.

While drawing the overview graph, the tree part of the entire graph slowed down the entire process greatly. This is due to the lack of documentations available. This so-called tree is actually called a phylogram or a phylogenetic tree and most phylograms are drawn in a circular manner. The one drawn in the overview graph is a right-angled phylogram and therefore finding a way to draw it in d3 v4 with the given dataset in Newick format was difficult.

Handling the data was not easy either. The data supplied by the researcher was not consistent throughout the process. Each data received had to be processed and inserted into data structures before it can be used.

Although faced with much difficulty the objectives of this project have still been met. There are not many metagenomics analysis tools available as metagenomics is still a growing topic and the world of microorganisms is still uncharted territory with much yet to be discovered by mankind. Researchers and students alike can use this tool to visualize their metagenomics data and it is still better than most other metagenomics tools which only provide one kind of visualization.

From this project I've learnt resilience and to not give up despite the obstacles faced. Reading about metagenomics has taught me a lot in terms of how much mankind still does not know about the microorganisms that could be living inside us. I also understand much better how this framework works and familiarized myself with TypeScript and d3 and how the d3 library and TypeScript language are both still fairly new and constantly being developed.

In the future, more interactivity and higher flexibility in terms of data input can be improved. Other ideas for other metagenomics analysis graphs could also be implemented. Examples include:

- PCA/Enterotype – classifying the organism based on the microbial community in the gut.
- Compared Network – comparison of the relationship between multiple metagenomes.

CHAPTER 6: CONCLUSION

- Lefse – a linear discriminant analysis effect size to discover biomarkers between 2 or more groups based on relative abundance.
- HGT elements – horizontal gene transfer view.
- Pathway between human and microbiota – displays the pathway between humans and microbiota.
- Bacteria & Fungi & Virus interaction – shows interaction between bacteria, fungi and virus.
- Time line change – shows a time line as the microbial community changes.

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Bitbucket, 2017. biobakery / biobakery / wiki / Home — Bitbucket . [ONLINE] Available at: <https://bitbucket.org/biobakery/biobakery/wiki/Home>. [Accessed 10 August 2017].
2. InterPro EMBL-EBI, 2017. EBI metagenomics: archiving, analysis and integration of metagenomics data < EBI metagenomics < EMBL-EBI. [ONLINE] Available at: <https://www.ebi.ac.uk/metagenomics/>. [Accessed 11 August 2017].
3. JGI IMG Home, 2017. JGI IMG Home. [ONLINE] Available at: <https://img.jgi.doe.gov/cgi-bin/m/main.cgi>. [Accessed 9 August 2017].
4. Ku, C. S., 2017. Chapter 1a: Introduction to System Analysis and Design, lecture notes, Object-Oriented System Analysis And Design UCCD2003 Universiti Tunku Abdul Rahman, delivered January 2017. [Accessed 11 August 2017].
5. Letunic, I. and Bork, P., 2016. Interactive tree of life (iTOL) v3: an online tool for the display and annotation of phylogenetic and other trees. Nucleic Acids Research, 44(Web Server issue), W242–W245. [ONLINE] Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4987883/>. [Accessed 9 August 2017].
6. Letunic, I. and Bork, P., 2017. iTOL: tree_of_life.tree. [ONLINE] Available at: <http://itol.embl.de/tree/892651230403971455132871>. [Accessed 11 August 2017].
7. Mitchell, A. et al, 2016. EBI metagenomics in 2016 - an expanding and evolving resource for the analysis and archiving of metagenomic data | Nucleic Acids Research | Oxford Academic. [ONLINE] Available at: <https://academic.oup.com/nar/article/44/D1/D595/2502680/EBI-metagenomics-in-2016-an-expanding-and-evolving>. [Accessed 11 August 2017].
8. National Research Council (US) Committee on Metagenomics, 2007. Why Metagenomics? Challenges and Functional Applications. The New Science of Metagenomics: Revealing the Secrets of Our Microbial Planet. Washington (DC): National Academies Press (US). [ONLINE] Available at:

BIBLIOGRAPHY

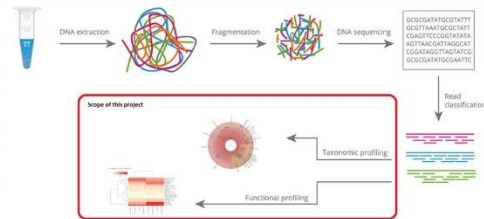
- <<https://www.ncbi.nlm.nih.gov/books/NBK54011/>>. [Accessed 8 August 2017].
9. Research | The Huttenhower Lab, 2017. Research | The Huttenhower Lab. [ONLINE] Available at: <<http://huttenhower.sph.harvard.edu/research>>. [Accessed 6 August 2017].
 10. Shotgun Metagenomic Sequencing, 2017. Shotgun Metagenomic Sequencing. [ONLINE] Available at: <<https://www.illumina.com/areas-of-interest/microbiology/microbial-sequencing-methods/shotgun-metagenomic-sequencing.html>>. [Accessed 8 August 2017].
 11. Smith, Y., 2017. What is Metagenomics? [ONLINE] Available at: <<https://www.news-medical.net/life-sciences/What-is-Metagenomics.aspx>>. [Accessed 8 August 2017].
 12. St. Jude PeCan Data Portal, 2017. St. Jude PeCan Data Portal. [ONLINE] Available at: <<https://pecan.stjude.org/proteinpaint/TP53>>. [Accessed 8 August 2017].
 13. tutorialspoint.com, 2017. Python tutorial. [ONLINE] Available at: <<https://www.tutorialspoint.com/python/>>. [Accessed 11 August 2017].
 14. Wang, J. and Jia, H., 2016. Metagenome-wide association studies: fine-mining the microbiome. Nature Reviews Microbiology, 14(8), pp.508–522. Available at: <<http://www.nature.com/nrmicro/journal/v14/n8/abs/nrmicro.2016.83.html>>. [Accessed 6 August 2017].
 15. Zhou et al., 2015. St. Jude researchers develop powerful interactive tool to mine data from cancer genome [ONLINE] Available at: <<https://www.stjude.org/media-resources/news-releases/2015-medicine-science-news/st-jude-researchers-develop-powerful-interactive-tool-to-mine-data-from-cancer-genome.html>>. [Accessed 8 August 2017].

An Online Tool for Metagenomics Analysis

Cheralyn Phong Jia Ern | Supervisor: Dr Ng Yen Kaow

What is metagenomics?

Metagenomics is the study of microorganisms in their natural living habitat, which normally involves the multiplex of microbial communities that coexists along with it. Microbes are everywhere in the environment and thus is involved in contributing to every process in the ecosystem, playing if not crucial roles. The microbes that thrive on the surface of the human body alone outnumber human cells by over 10 times. Understanding how this microbial community structure affects the human body may help in providing better diagnosis for deterring, inhibiting and treating diseases. DNA from a sample such as the gut is extracted, fragmented, DNA sequencing is done and classified. Data is then collected from a researcher for analysis. The scope of this project only comprises of constructing the analysis for visualization.



Problem Statement

Much consideration should be put into this problem given that without a proper method to sequence long DNA strands, it is near impossible to completely sample and analyze the diversity and abundance of all genetic information in a complex microbial community. It would also be immensely difficult to study uncultivable microorganisms because these microorganisms would not be available for use in laboratories. With metagenomics shotgun sequencing, human diseases can be better understood. Genetic variants in the human population can be identified and correlated to distinguished phenotypes which can be traits leading to a disease. We can also make data less challenging to analyze as everything is computerized. There are existing websites that do metagenomics analysis online but these websites only provide one aspect of the analysis such as drawing a heat map or building trees. There isn't one to display whole genome shotgun metagenomics analysis results. This project aims to create an online tool to provide researchers and students alike to do whole genome shotgun metagenomics analysis.

Methods

The tool shall be able to display every figure necessarily with:

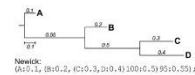
- Drawing area – the major graph is displayed.
- Interaction – interacting with the graph. Actions such as hovering and clicking to display links and other detailed information in suspension boxes.
- Data process – may be required to calculate sum etcetera but might not necessarily apply to every figure.

Written using:

- Sass - an extension of CSS that makes visualization more interesting and visually attractive.
- Typescript - superset of JavaScript which is more object-oriented and can cope with the complicated data structures of bioinformatics.
- d3.js - JavaScript's data visualization library that can bind documents and keep them updated.

Data input format:

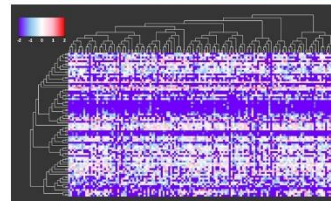
- As excel sheets converted to CSV
- Newick tree format file



Objectives

- ✓ To aid researchers in visualizing huge datasets more easily.
- ✓ To provide an online tool for whole genome shotgun metagenomics analysis result.
- ✓ To create more interactive and useful graphs that can better explain and display the inputted data set so that key information is highlighted and easily noticed.
- ✓ To come up with visualization designs for new analysis and figures.
- ✓ To create a complete tool which comprises of multiple different aspects that highlights all attributes of the dataset so that users can complete all analysis on a metagenomics dataset just from this single website without needing to input the dataset into multiple tools to generate different graphs.
- ✓ To allow users to display and adjust figures online and then export them with publishable quality to be directly used in articles.

Sample

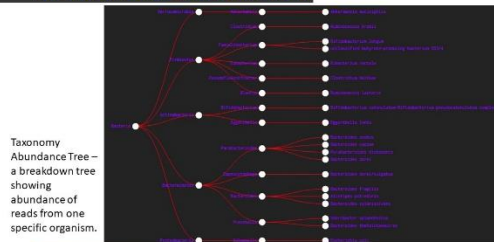


Overview – the overview of the entire dataset with integrated heat map and clustering in one figure.

User Input into Visualization

Sample	CC544093303ST-4-0	CC563448910ST-4-0
unlabeled_Alistipes_sp_HMR_12051	-2	-2
Bacteroides_Eggs_12005	-2	-0.21812397
Bacteroides_HIF63161_11303	-1.71329478	-2
Streptococcus_thermophilus_13152	1.222291509	1.319392647
Streptococcus_thermophilus_13176	-2	-0.80585062
Streptococcus_sakaiensis_11377	-0.961226214	-0.72060253
[Bacteroides_bacterium_12416]	-2	-0.05046622
Clostridium_Therap_11378	1.08866277	-2
Corynebacterium_sah_11394	-2	-2
unlabeled_Bacteroides_pseudocatalis_bacterium_	0.085290382	0.28981767
Kuminococcus_bactarum_13609	-2	-2
[Bacteroides_grow_11611]	-1.462314566	0.057564375
[Bacteroides_grow_11612]	-2	-2
Rubrobacterium_ventriosum_13129	-2	-2
Roseburia_thermofila_11411	-2	-2
Roseburia_homob_11633	-2	-2
Parabacteroides_anaerofaciens_bacterium_0481	-2	1.310203044
Roseburia_ustulata_bacterium_1252	-1.350766472	-2

The above table on the left shows a sample input while the image on the right shows the heat map output based on the table. On a colour gradient scale of 2 to -2 with 2 being red and -2 being blue signifies the proportion of a genus in a particular sample. Using colour as indicator, we can see that the redder the colour displayed, the higher the proportion of this genus in that particular sample. When the following data is inputted into the overview, we can see that for patient CC544093303ST-4-0 most of its values are approaching -2 hence the blue color for each block with the exception for unnamed_butyrate-producing_bacterium_ at 0.08 producing a different colour. Patient CC563448910ST-4-0 on the other hand has high proportions of Streptococcus_thermophilus_13175 at 1.57 hence the colour displayed is red. Visualizations as such makes it easier to display huge amounts of data which are easy to interpret.



Taxonomy Abundance Tree – a breakdown tree showing abundance of reads from one specific organism.

Conclusion

This project aims to create an easily accessible tool for researchers and students alike to visualize metagenomics analyses. As there are no existing websites doing whole genome shotgun metagenomics analysis results, the objective of this project is to provide one such tool whereby complete analysis and visualization of inputted data can be achieved. This tool will then provide as necessary a drawing area, interactions and data processing. As metagenomics will help propel and increase understanding of how microbiomes interact and affect our lives, this project is a good cause and will indirectly help in the future development of healthcare and disease control in the medical field.

PLAGIARISM CHECK SUMMARY

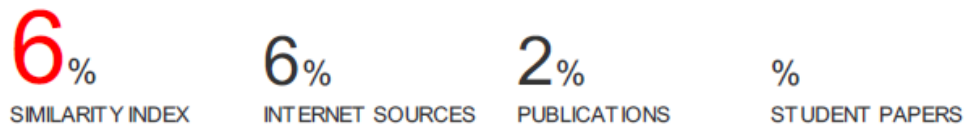
PLAGIARISM CHECK SUMMARY

The screenshot shows the 'feedback studio' interface for 'Cheralyn Phong' using 'An Online Tool For Metagenomics Analysis'. The document content includes a 'DECLARATION OF ORIGINALITY' and a signature line. A 'Match Overview' sidebar on the right displays a total similarity of 6% and lists seven matches with their respective percentages.

Match Number	Source	Percentage
1	eprints.utar.edu.my (Internet Source)	1%
2	www.alexandria.unisg... (Internet Source)	1%
3	www.tecreader.com (Internet Source)	<1%
4	www.uday.net (Internet Source)	<1%
5	repositories.lib.utexas... (Internet Source)	<1%
6	www.net4geeks.com (Internet Source)	<1%
7	www.buttelafco.org (Internet Source)	<1%

An Online Tool For Metagenomics Analysis

ORIGINALITY REPORT



PRIMARY SOURCES

1	eprints.utar.edu.my Internet Source	1%
2	www.alexandria.unisg.ch Internet Source	1%
3	www.tecreader.com Internet Source	<1%
4	www.uday.net Internet Source	<1%
5	repositories.lib.utexas.edu Internet Source	<1%

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	CHERALYN PHONG JIA ERN
ID Number(s)	14ACB02581
Programme / Course	BACHELOR OF COMPUTER SCIENCE (HONS)
Title of Final Year Project	AN ONLINE TOOL FOR METAGENOMICS ANALYSIS

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: _____ % Similarity by source Internet Sources: _____ % Publications: _____ % Student Papers: _____ %	
Number of individual sources listed of more than 3% similarity: _____	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Signature of Co-Supervisor

Name: _____

Name: _____

Date: _____

Date: _____

UNIVERSITI TUNKU ABDUL RAHMAN
FACULTY OF INFORMATION & COMMUNICATION
TECHNOLOGY (PERAK CAMPUS)

CHECKLIST FOR FYP2THESIS SUBMISSION

Student Id	1402581
Student Name	CHERALYN PHONG JIA ERN
Supervisor Name	DR. NG YEN KAOW

TICK (✓)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Title Page
	Signed form of the Declaration of Originality
	Abstract
	Table of Contents
	List of Figures (if applicable)
	List of Tables (if applicable)
	List of Symbols (if applicable)
	List of Abbreviations (if applicable)
	Chapters / Content
	Bibliography (or References)
	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
	Poster
	Signed Turnitin Report (Plagiarism Check Result – Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

<p>I, the author, have checked and confirmed all the items listed in the table are included in my report.</p> <p>_____</p> <p>(Signature of Student)</p> <p>Date:</p>	<p>Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction.</p> <p>_____</p> <p>(Signature of Supervisor)</p> <p>Date:</p>
--	--