Software Development Group Project

Mandriva

This software was developed by students at Queen Mary University of London as part of the MSc Bioinformatics course 2023/24

Group members: Tom Bennett, Melika Abdi, Niya Louis, Adam Brachtl

Repository link:

https://github.com/TomBennett2202/Mandriva

Table of Contents

Introduction

Population genetics is a field crucial for understanding genetic diversity and evolutionary dynamics within and between populations. As advances in genomics accelerate, population genetics data is rapidly becoming more available. Therefore, the need for specialised software to analyse genetic data has become increasingly vital. With a focus on a simple user-friendly interface, Mandriva provides the user options to analyse genetic relationships and extract meaningful insights from genomic datasets. The software is designed with a primary emphasis on unravelling patterns of genetic variation, thereby facilitating the exploration of complexities inherent in population genetics, enhancing our comprehension of diverse aspects such as migration patterns, human ancestry, and health within individual populations and broader demographic groups. Mandriva provides a range of features, such as clustering analysis, admixture analysis, and the quantification of genetic differentiation between populations. Additionally, it offers additional relevant insights into the characteristics of chosen single nucleotide polymorphisms (SNPs). The software is tailored to handle Variant Call Format (VCF) data, leveraging it to generate comprehensive visualizations, statistical summaries, and reports.

Clustering analysis

Principal Component Analysis (PCA) stands as a pivotal statistical method chosen for its effectiveness in highlighting variance and unveiling intricate patterns within complex genetic datasets. PCA independently identifies the main sources of variance in a dataset, known as principal components (PCs). The first component captures the most significant variation, and the second component captures the next largest, and so forth until all variance is accounted for. In genetic data, PCA reduces the main dimensions associated with allele frequencies and provides individual positions within these dimensions (Liu *et al.*, 2020). Thus, allowing for visualization of their population structure. However, PCA is not without its limitations. The reduction of these dimensions can lead to the loss of potentially important genetic information. Additionally, PCA assumes linearity and orthogonality of principal components, which might not always be held in complex biological data.

Admixture analysis

ADMIXTURE is a software tool used in population genetics for analysing and inferring the ancestry of individuals from multilocus SNP genotype datasets. It is particularly useful for understanding the genetic structure of populations and for identifying the proportions of individual ancestries from a set of predefined populations, which are often referred to as the source or ancestral populations. ADMIXTURE employs a maximum likelihood estimation (MLE) method as its underlying algorithm to infer ancestry fractions. The selection of ADMIXTURE over alternative admixture analysis tools, including sNMF, was motivated by its distinguished accuracy in ancestry inference. While it is acknowledged that ADMIXTURE is significantly slower than sNMF, the emphasis on precision and reliability took precedence over speed for the specific requirements of this project (Wang, 2022). K is a crucial user-defined parameter while using ADMIXTURE, indicating the hypothesized number of ancestral populations. Occasionally, the choice of K in genetic analysis is guided by biological factors. However, more often, it involves exploring a range of potential K-values using various likelihood-based methods. The selection of the optimal K value is crucial and is determined by employing cross-validation techniques. This approach ensures the robustness and accuracy of the ancestry estimation process. This involves partitioning the dataset into five subsets of equal size for each potential value of K, the number of ancestral populations assumed in the model. The cross-validation process is executed iteratively: in each iteration, one of the five subsets is temporarily "masked" or held out, while the model is trained on the remaining four subsets. During this training phase, the model employs the unmasked data to calculate allele frequencies within each assumed ancestral population and to assign ancestry proportions to the individuals based on these frequencies. Once the model has been trained, it then applies the derived allele frequency and ancestry information to predict the genotypic values of the individuals in the masked subset. This predictive step is critical, as it evaluates the model's ability to accurately infer genetic information based on the learned parameters from the other subsets. By repeating this process for each of the five subsets (each time masking a different subset and training the model on the rest), ADMIXTURE comprehensively assesses the model's predictive performance across the entire dataset. This cross-validation approach serves multiple purposes. (University of Vienna, no date). Primarily, it provides a measure of the model's predictive accuracy, offering insights into how well the chosen value of K reflects the underlying genetic structure of the population. Additionally, by comparing the cross-validation errors for different K values, researchers can identify the most appropriate number of ancestral populations that best fit the genetic data, thereby minimizing the risk of overfitting.

Allele and genotype frequencies and pairwise FST differentiation

Allele and genotype frequencies serve as crucial metrics in population genetics. Genotype frequency reflects the prevalence of a specific genotype within a population, while allele frequency indicates the occurrence of a particular allele in that population. Both metrics are essential for comprehending inheritance patterns, studying evolutionary processes, and assessing genetic diversity within and among populations (Kim et al., 2011, Klug & Cummings, 2002).

In the realm of population genetics, pairwise FST emerges as a valuable tool. This metric gauges the degree of genetic differentiation between populations. Calculations often rely on allele frequencies, comparing the genetic diversity within populations to the overall genetic diversity across multiple populations. FST values span from 0 to 1, where 0 signifies no genetic differentiation, indicating gene flow among populations. Conversely, a value of 1 signifies complete genetic differentiation, suggesting no gene flow and complete isolation between populations (Hedrick, 2005). An individual FST value is calculated by first computing the total count (TC), achieved by summing the counts of homozygous reference (HR), heterozygous (H), and homozygous alternate (HA) genotypes for each SNP: TC=HR+H+HA. Next, the frequency of the reference allele (p) is calculated using the formula $p=\frac{2\cdot HR+H}{2\cdot TC}$ with the condition $TC>0$ to avoid division by zero. Subsequently, the frequency of the alternate allele (q) is obtained by $q=1-p$. Finally, the FST value for each SNP is computed as $FST=1-(p^2+q^2)$, again with the condition $TC>0$ to ensure meaningful calculations. To compare genetic differentiation between different populations or groups, the pairwise FST is computed as the mean absolute difference between the FST values of the two populations.

Technologies

The Mandriva platform leveraged a combination of technologies to facilitate its functionalities. This included the Flask framework, which was utilized for web development, alongside the Python programming language. For data curation and processing, tools such as BCFtools, SnpSift, Plink (utilized both on RStudio and terminal environments), and Ensembl Variant Effect Predictor (VEP) played integral roles. The cyvcf2 library was employed for efficient VCF file parsing and manipulation. SQLite was utilized to manage and query the database. For data visualization and analysis, Plotly, seaborn, and matplotlib were utilized to create interactive plots and visualizations.

HTML, CSS, JavaScript, and Bootstrap were employed for front-end development. HTML was chosen as it is a relatively simple language, that is easy to learn. HTML also allows for the integration of images and visual material like photos and videos to enhance user experience. Additionally, HTML is SEO (Search Engine Optimization) – friendly. CSS enables the definition of styles for classes, promoting consistency and reusability across the web application. This facilitates the creation of cleaner, more readable codebases for better maintenance. JavaScript was chosen for its versatility and ability to enhance user interactivity on the web application. As a client-side scripting language, it allows for dynamic updates to the content without requiring a page reload. This leads to a smoother and more engaging user experience. JavaScript's compatibility with web browsers makes it a reliable choice for implementing various functionalities, such as form validation, real-time updates, and interactive features. Its extensive ecosystem of libraries and frameworks further supports efficient development. By incorporating JavaScript into the project, the goal was to create a more dynamic and responsive web application that meets the requirements of modern user expectations. Bootstrap was chosen for its robust framework that facilitates responsive and mobile-friendly web development. As a front-end framework, Bootstrap provides a collection of pre-designed components, styles, and utilities, streamlining the process of creating a visually appealing and consistent user interface. Its grid system ensures a responsive layout that adapts to different screen sizes, enhancing the accessibility of the web application across various devices.

Data sources

The genetic data utilized in the analyses comes from two primary sources. Firstly, whole-genome data from various human populations was extracted from the 1000 Genomes project database. Additionally, an additional 726 human samples from an undisclosed location in Siberia were included and merged with the 1000 Genomes data. The data file was provided in a compressed VCF file. In total, genetic data for nearly 4000 individuals were accessible, representing 27 populations (refer to table 1). Additionally, for external database integration, the latest ClinVar and dbSNP VCF files were obtained from their corresponding FTP repositories and employed for annotation purposes.

Table 1. Populations and their acronyms, along with the number of samples per population.

Acronym	Population	Number of samples
ACB	African Caribbean in Barbados	116
ASW	African ancestry in Southwest US	74
ESN	Esan in Nigeria	149
GWD	Gambia in Western Division, The Gambia	178
LWK	Luhya in Webuye, Kenia	99
MSL	Mende in Sierra Leone	99
YRI	Yoruba in Ibadan, Nigeria	178
CLM	Colombia in Medellin, Colombia	132
MXL	Mexican ancestry in LA, California	97
PEL	Peruvian in Lima, Peru	122
PUR	Puerto Rican in Puerto Rico	139
CDX	Chinese Dai in Xishuangbanna, China	93
CHB	Han Chinese in Beijing, China	103
CHS	Southern Han Chinese, China	163
KHV	Kinh in Ho Chi Minh City, Vietnam	122
JPT	Japanese in Tokyo, Japan	104
CEU	Utah residents with Northern and Western European ancestry	179
FIN	Finnish in Finland	99
GBR	British in England and South Scotland	91
IBS	Iberian Population in Spain	157
TSI	Toscani in Italy	107
BEB	Bengali in Bangladesh	131
GIH	Gujarati Indian in Houston Texas, TX	103
ITU	Indian Telugu in the UK	107
PJL	Punjabi in Lahore, Pakistan	146
STU	Sri Lankan Tamil in the UK	114
SIB	Siberian in Siberia, Russia	726

Design

Use cases

Use case 1
Users should have the option to initiate a clustering analysis, allowing them to specify which populations to include. Once the selections are made, the system should redirect the user to a results page with a visual representation of the clustering analysis, tailored to the chosen populations.

Use case 2
Users should have the option to initiate an admixture analysis, allowing them to specify which populations to include. Once the selections are made, the system should redirect the user to a results page with a visual representation of the admixture analysis, tailored to the chosen populations.

Use case 3
Users should be able to request detailed information on SNPs by specifying genomic coordinates (chromosome, start, and end positions), gene names, or RS IDs. Additionally, users should be able to select specific populations for analysis. After making these selections, users will be redirected to a dedicated page featuring a comprehensive table presenting genotype and allele frequencies, along with clinical relevance data for the specified SNPs and populations.

Use case 4
If multiple populations were selected in use case 3 the user should be redirected to the results page but with an additional generated matrix of pairwise population genetic differentiation that the user can download, alongside a suitable visual representation of it.

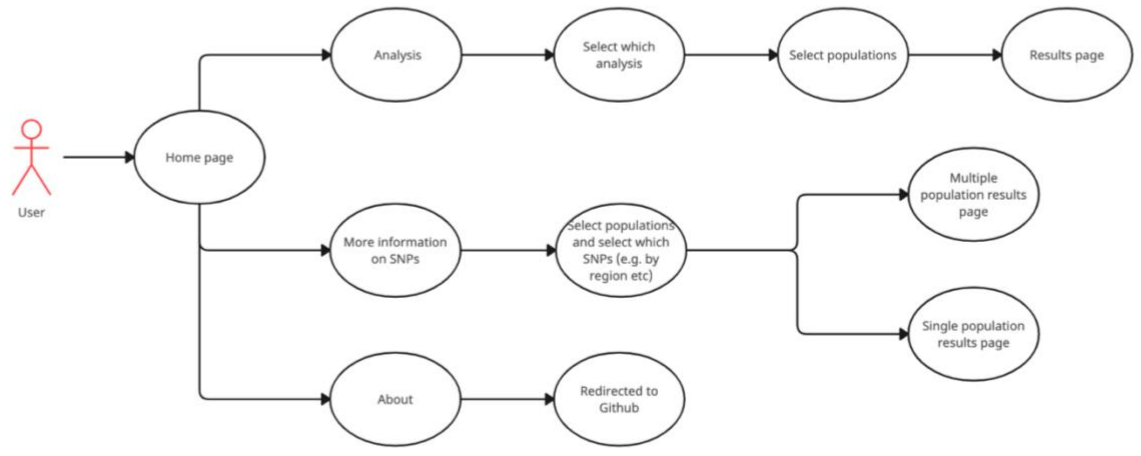

User
Home page
Analysis
Select which analysis
Select populations
Results page
More information on SNPs
Select populations and select which SNPs (e.g. by region etc)
Multiple population results page
Single population results page
About
Redirected to Github

Figure 1. Flowchart Illustrating User Interactions and Workflows.

Initial web design

The web design prioritized creating a user-friendly interface with simple components to enhance user interaction. A clean and minimalistic design approach was adopted for clarity, maintaining consistent layout, colour scheme, and font across all pages. HTML played a key role in structuring and presenting content, offering standardized annotations for headings, lists, forms, paragraphs, and links. This ensures a visually pleasing and orderly interface, promoting seamless navigation and a clear browsing experience. HTML's form elements, including input fields and buttons, enable users to insert data, submit, and interact with various functions of the application.

On the homepage (see figure 2), users encounter the prominent display of the website name "Mandriva" at the top centre. Two main buttons, labelled "Analysis" and "SNP", facilitate navigation to specific pages for cluster and admixture analysis and the SNP query page, respectively. Additionally, a navigation bar, including "Home" and "About" links, remains accessible from any page. The "Home" link provides easy access to return to the initial homepage, enabling users to select their preferred type of data analysis tool. Meanwhile, the "About" link directs users to the project's GitHub page for further information.

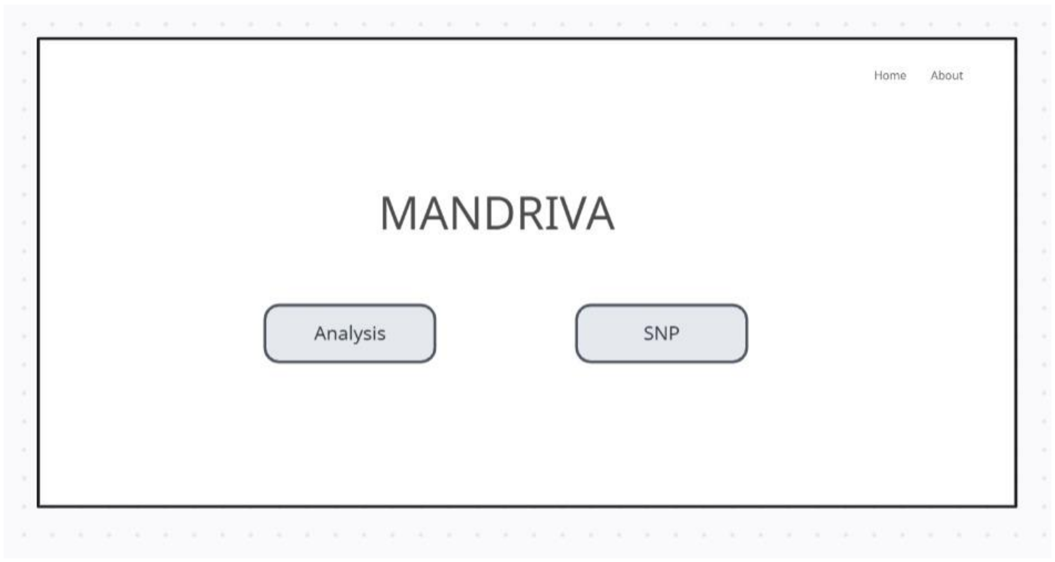


Figure 2. Initial design of the home page of the website

Navigating through the analysis page (see figure 3), users encounter two different buttons for “Cluster analysis” and “Admixture analysis”.

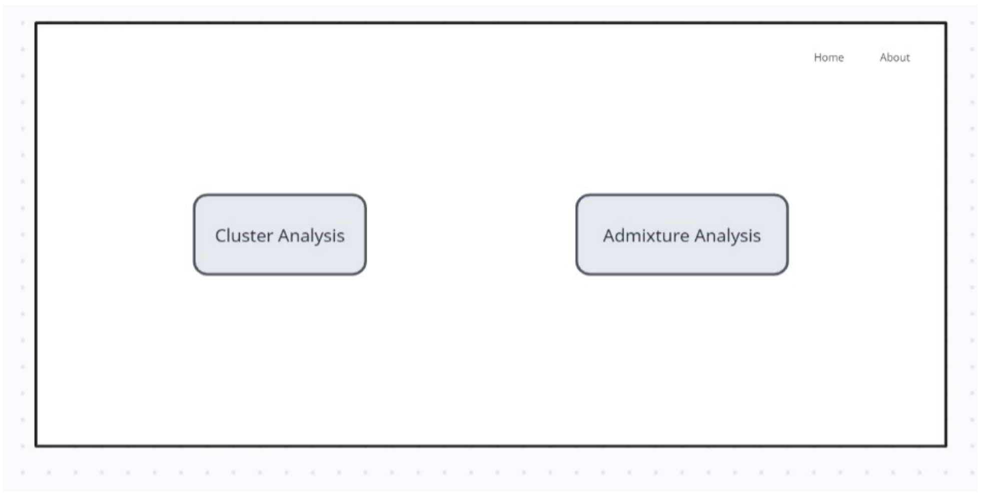


Figure 2. Initial design of the analysis page of the website

Clicking the "Cluster analysis" or "Admixture analysis", the user will be redirected to pages where there are options to choose whichever superpopulation and population that the users would like to include for the genetic analysis and is asked to click confirm button in the end (see figure 4).

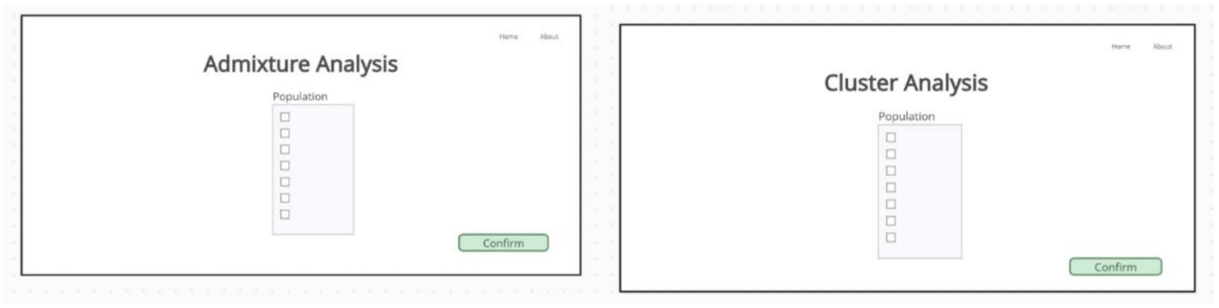


Figure 4. Initial design of the population selection pages for admixture and cluster analysis.

The results will be shown in a new page with plots, along with the download option to save it for later (see figure 5).

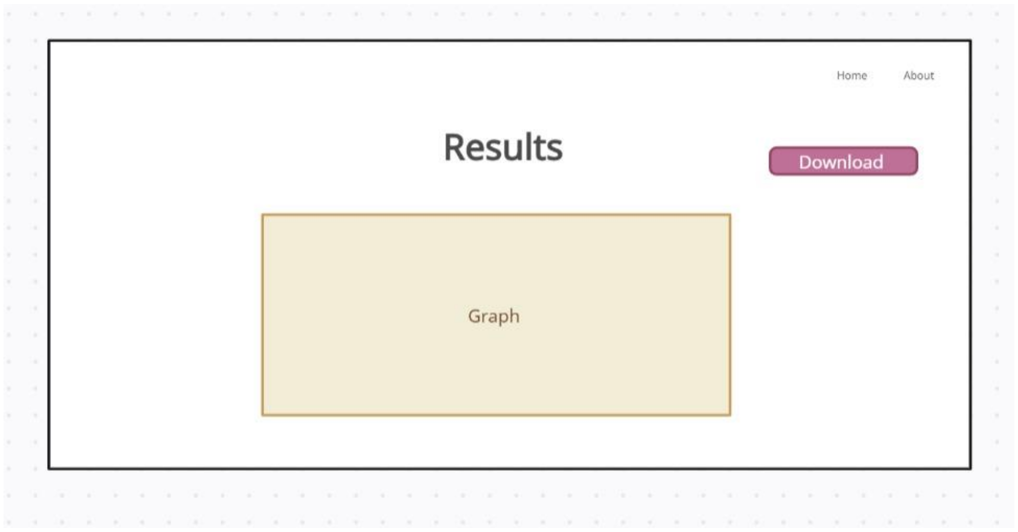


Figure 5. Initial design of the results page either displaying admixture results or cluster results.

If the user selects the other option in the home page called “SNP”, it navigates to a page where it requires the users to select superpopulations or populations and which SNPs the user wants to retrieve more information about (see figure 6). This is achieved by text boxes the user can type in for gene names, RS IDs, chromosome, start position and end position. After selecting the required fields, a “confirm” button can be selected to complete the process.

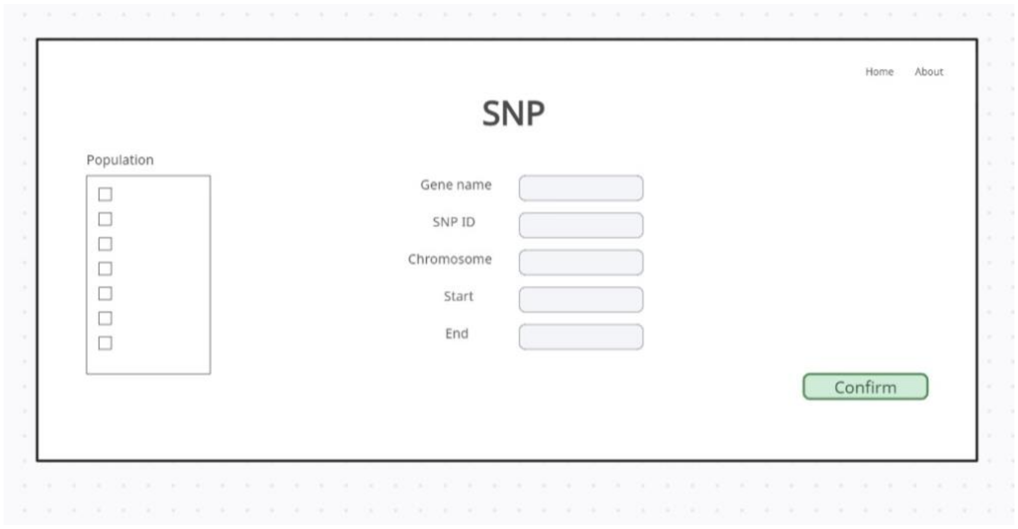


Figure 6. Initial design for the SNP selection form page.

Results are be displayed to the users with an option to download it on the results page (see figure 7).

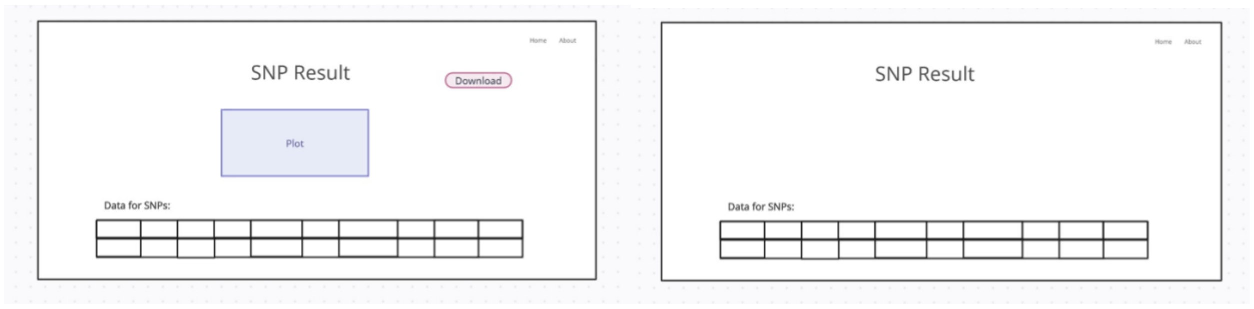
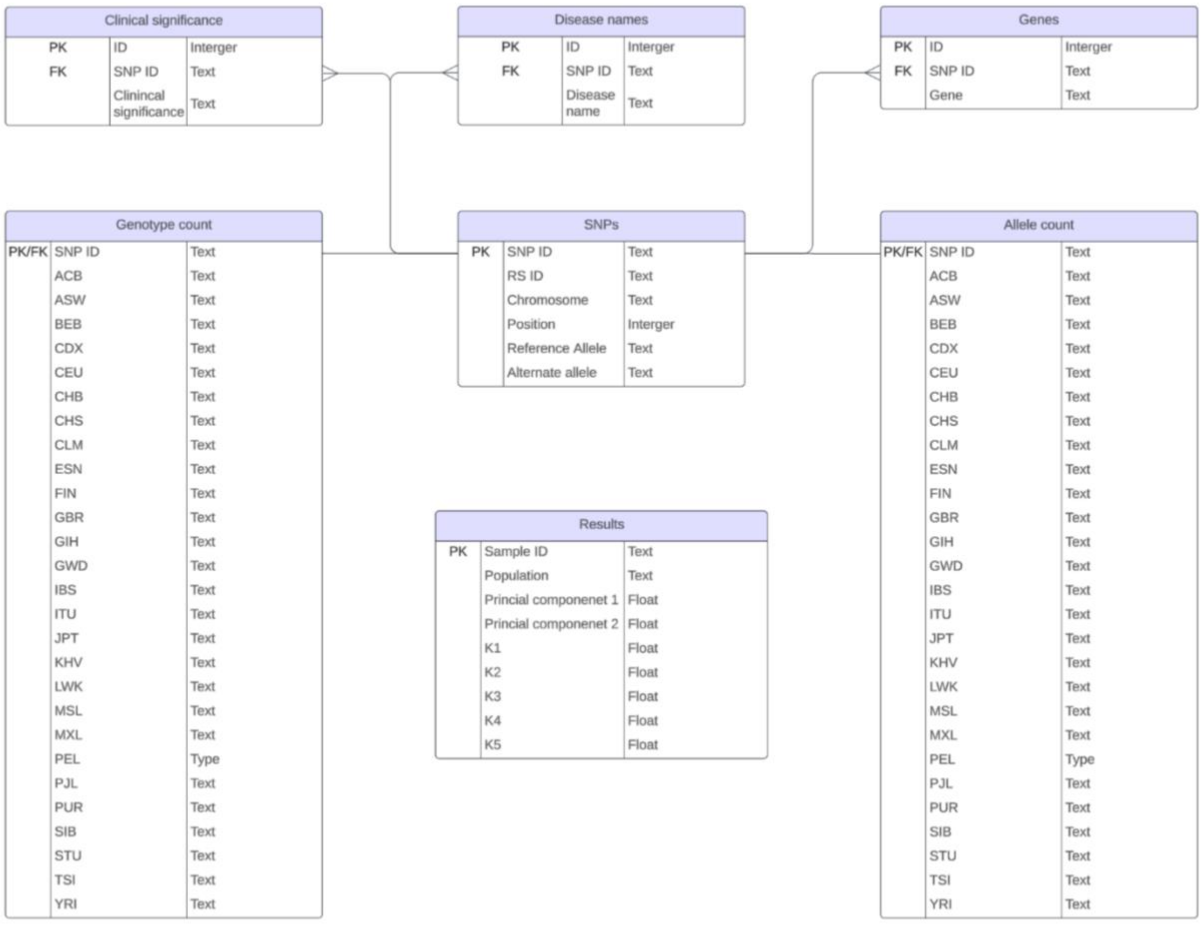


Figure 7. Initial design of the SNP results page (left is if multiple are selected and right is only one population is selected).

Database Design and SQL Implementation

SQLite was selected as the data storage solution for its lightweight nature, making it well-suited for the relatively modest scale of the project. Its ease of use further contributed to its appropriateness for the task at hand. Additionally, its flexibility allows for adaptability based on the evolving needs and scale of the project.

The database comprises seven tables (see figure 8), each serving a specific purpose. The SNPs table contains fundamental information about the SNPs, such as SNP ID, RS ID, chromosome, base position, reference allele, and alternate allele. The genes table, associates genes with specific SNP IDs, featuring an additional autoincremented ID. The clinical significance table incorporates clinical significance data from the ClinVar database, linked to SNP IDs and accompanied by an autoincremented ID. Similarly, the clinical disease names table integrates clinical disease names from ClinVar, correlating with SNP IDs and incorporating an autoincremented ID. The genotype count table records genotype counts for each population corresponding to a given SNP, while the allele count table mirrors this structure but focuses on allele counts. Finally, the results table encapsulates crucial data, including values for the first two principal components and values for K1 to K5, derived from the admixture analysis ancestry proportions. These values are per sample and are further annotated in the table based on their respective populations.



Clinical significance		
PK	ID	Interger
FK	SNP ID	Text
	Clinincal significance	Text

Disease names		
PK	ID	Interger
FK	SNP ID	Text
	Disease name	Text

Genes		
PK	ID	Interger
FK	SNP ID	Text
	Gene	Text

SNPs		
PK	SNP ID	Text
	RS ID	Text
	Chromosome	Text
	Position	Interger
	Reference Allele	Text
	Alternate allele	Text

Results		
PK	Sample ID	Text
	Population	Text
	Princial componenet 1	Float
	Princial componenet 2	Float
	K1	Float
	K2	Float
	K3	Float
	K4	Float
	K5	Float

Genotype count		
PK/FK	SNP ID	Text
	ACB	Text
	ASW	Text
	BEB	Text
	CDX	Text
	CEU	Text
	CHB	Text
	CHS	Text
	CLM	Text
	ESN	Text
	FIN	Text
	GBR	Text
	GIH	Text
	GWD	Text
	IBS	Text
	ITU	Text
	JPT	Text
	KHV	Text
	LWK	Text
	MSL	Text
	MXL	Text
	PEL	Type
	PJL	Text
	PUR	Text
	SIB	Text
	STU	Text
	TSI	Text
	YRI	Text

Allele count		
PK/FK	SNP ID	Text
	ACB	Text
	ASW	Text
	BEB	Text
	CDX	Text
	CEU	Text
	CHB	Text
	CHS	Text
	CLM	Text
	ESN	Text
	FIN	Text
	GBR	Text
	GIH	Text
	GWD	Text
	IBS	Text
	ITU	Text
	JPT	Text
	KHV	Text
	LWK	Text
	MSL	Text
	MXL	Text
	PEL	Type
	PJL	Text
	PUR	Text
	SIB	Text
	STU	Text
	TSI	Text
	YRI	Text

Figure 8. SQL database schema

The central table in the database is the SNPs table, forming a one-to-many relationship with the genes, clinical significance, and disease names tables, where the SNP ID serves as the foreign key. Additionally, the SNPs table establishes a one-to-one relationship with the genotype counts and allele counts tables, again utilizing the SNP ID as the foreign key. This relational structure ensures a cohesive linkage between the primary SNPs table and its associated data in the secondary tables, facilitating comprehensive and efficient data retrieval.

Implementation

Cluster

The raw VCF file was manipulated with the following plink command in R: plink --vcf chr1.vcf.gz --double-id --indep-pairwise 50 10 0.1 --make-bed --out chr1_pruned. The inclusion of the "--double-id" parameter in the command serves to duplicate the IDs present in the VCF file. This step is inherent to the Plink software, which typically expects a structure with one family and one individual ID in the original file. However, after further research this duplication was unnecessary, as the dataset only comprised of a single type of ID. Additionally, the "--indep-pairwise 50 10 0.1" parameter is necessary in plink software for calculating Linkage disequilibrium (LD). LD is a population-wide parameter denoting the non-random association of alleles at two or more loci (Slatkin, 2008). This provides insights into the extent to which information is conveyed when observing an allele at a specific locus. Notably, SNPs exhibiting high linkage disequilibrium harbour redundant information (Liu *et al*., 2020). As it shows in the command, a chromosomal window of 50 SNPs was considered at a time, the second argument is the window step size, which means after calculating the linkage, the window is shifted by 10 bp, and finally, the last argument is the r2 threshold that for any pair whose association threshold is larger than 0.1, the SNP will be removed from the pair. Finally, the "--make-bed" makes a .bim, .bed and a .bam file from the pruned VCF file. Plink calculated LD data, yielding two files: chr1_pruned.prune.in and chr1_pruned.prune.out. The first file with the "in" suffix contained the sites that should remain and the file with the "out" suffix contained the sites that should be removed.

Using the plink-generated files, an additional Plink command was ran to perform the PCA analysis. The command was: plink --bfile chr1_pruned --pca --out chr1_pca_result. The –bfile parameter is the prefix of the .bed, .bim, and .fam files (excluding the file extension), which in this case, was chr1_pruned. This generated two output files chr1_pca_result.eigenval and chr1_pca_result.eigenvec files. The .eigenval file contains the eigenvalues, representing the variance captured by each principal component and the .eigenvec file contains the eigenvectors, which are the principal components themselves (the transformed genotype data). Typically, the first or second columns of the .eigenvec file have the sample IDs (depending on the command you used with PLINK), and the other columns are the PCs. The percentage of variance captured by each PC was extracted from the .eigenval file. Subsequently, the decision to focus on the first two PCs was made, as they demonstrated the most substantial variance compared to the others. Specifically, PC 1 accounted for 52% of the variance, while PC 2 contributed 16%.

Admixture

For the project, ADMIXTURE analysis was conducted on "Apocrita", the High-Performance Computing (HPC) cluster at Queen Mary University of London. HPC environments prove to be optimal for computationally intensive tasks, providing the essential resources for efficient large-scale analyses. The added advantage lies in the capability to run these tasks in the background, allowing time to be allocated to other tasks while the computations proceed uninterrupted.

With PLINK, the raw VCF file underwent conversion into a binary format (BED) tailored for ADMIXTURE analysis. A scripted loop was implemented to execute ADMIXTURE with cross-validation, iterating through a range of K values (from 1 to 5). The cross-validation error at each K value was analysed to pinpoint the model that offered the optimal fit to the data without succumbing to overfitting. As a result, a K value of 5 was chosen as it exhibited the lowest cross-validation error, thus indicating the most probable number of ancestral populations within the dataset, reflecting the model’s effectiveness in capturing the underlying structure of the data at that specific K value. The ADMIXTURE analysis generated a .Q file comprising tab-delimited columns that depict the proportions of each ancestral population for every sample.

SNP IDs

The original VCF file presented inconsistent IDs, including variations like "1:10397:C:A" and others like "rs1557426847." To ensure uniformity and uniqueness, the IDs were annotated according to the Human Genome Variation Society (HGVS) nomenclature. Employing BCFtools, the IDs were transformed to a standardized format, such as "1:10397:C:A," where it represents chromosome 1, base position 10397, reference allele C, and alternate allele A. The modified data was then saved as a new VCF file.

RS IDs

For RS ID annotation of the original VCF file, an annotation file was initially downloaded from the latest SNP VCF file on dbsnp FTP (as of 2022-11-16). This downloaded file served as the database for annotating RS IDs. Using BCFtools, the first chromosome was extracted out of the database VCF and the name of the first chromosome was changed from NC_000001.11 to match the original VCF which was chr1. To annotate, BCFtools was first used, however, whilst using BCFtools, the annotation process only captured the first occurrence of a SNP with the same RS ID, neglecting subsequent instances when the SNP reappeared with a different alternative allele. In contrast, SnpSift exhibited the desired behaviour by consistently annotating multiple occurrences of the same RS ID, even when they presented with varying alternative alleles. This distinction made SnpSift the preferred tool for ensuring comprehensive and accurate SNP annotation in the dataset. First the IDs were removed from the original VCF using BCFtools as SnpSift cannot replace IDs, unlike BCFtools, and then was annotated using SnpSift with the renamed database VCF. Not all entries were successfully annotated, and upon further investigation on the dbSNP website, some were identified with matching RS IDs. This discrepancy could arise from the FTP latest release VCF not being as up to date as their web database.

Genes

To establish the connection between SNPs and their corresponding genes, the SNPs from the new SNP-annotated VCF file were linked with genes using VEP, employing the “--database” parameter. This process generated a tab-delimited text file containing columns for SNP IDs and their corresponding genes.

Clinical Relevance

To annotate each SNP to their given clinical relevance, an annotation file was initially downloaded from the latest SNP VCF file on ClinVar FTP (as of 2024-02-22). The first chromosome in the ClinVar VCF was changed from NC_000001.11 to match the original VCF which was chr1. The SNP ID-annotated VCF was processed using VEP with the following command: "vep -i output.vcf.gz -o vep_output.txt --custom output_renamed.vcf.gz,ClinVar,vcf,exact,0,CLNSIG,CLNREVSTAT,CLNDN". This command utilized the ClinVar VCF as a local custom database, generating a tab-delimited text file. The file included uploaded SNP IDs and an "extra" column containing clinical significance (CLNSIG) and clinical disease name (CLNDN).

Allele and Genotype frequencies

In preparation for retrieving allele and genotype counts using BCFtools, subsets of VCF files were created, each specific to one of the 27 populations under study. This resulted in the generation of 27 distinct sub-VCF files, one for each population.

Subsequently, for genotype counts, a loop iterated through these population-specific sub-VCF files, employing PLINK to compute genotype counts using the “--geno-counts" parameter. This process produced individual population-specific .gcount files, structured with columns for SNP IDs and counts of homozygous reference alleles, heterozygotes, and homozygous alternate alleles. Following this, another loop amalgamated these counts into a single column for each population, consolidating the counts into a unified format (e.g., "116,0,0"). Only this consolidated column was retained and stored in new text files. Next, these individual text files were merged, appending a column of SNP IDs as the first column, thereby creating a comprehensive file with SNP IDs in the first column and genotype counts for all 27 populations in subsequent columns.

For allele counts, a similar approach was employed. Another loop traversed the population-specific sub-VCF files, executing PLINK to compute allele counts. Analogous to the process for genotype counts, this generated population-specific .acount files, containing SNP IDs and counts of reference and alternate alleles. Subsequent consolidation combined these allele counts into a single column for each population, representing reference and alternate allele counts in a unified format (e.g., "72,0"). Following the same procedure, only this consolidated column was retained and stored in new text files. Finally, these individual text files were merged, appending a column of SNP IDs as the first column, resulting in a comprehensive file with SNP IDs in the first column and allele counts for all 27 populations in subsequent columns.

Database Population

To populate the database, a multi-step process was executed using Python and SQLite. In the initial phase, the process commenced with parsing data from the VEP output file that contains sections related to clinical significance and clinical disease names. This was achieved by utilizing regular expressions. This data was then organized into the distinct tables: clinical significance and disease names.

Subsequently, the SNP IDs and their corresponding genes columns were extracted from the VEP file and were stored into the genes table.

The process of loading data into the SNPs table was initiated by leveraging the CyVCF2 library to read and extract information from two distinct VCF files: one containing SNP IDs and the other RS IDs. The RS IDs were specifically extracted from the latter file and stored for reference. Following this, the iteration through the SNP IDed file facilitated the extraction of SNP ID, chromosome, position, reference allele, and alternate allele. Subsequently, this extracted information, along with the associated RS IDs, was loaded into the SNPs table.

Loading the genotype and allele count text files was a straightforward process. By employing the SQLite command ".mode tabs", the content of the text files was imported into their respective tables. This simplified approach ensured an efficient and seamless data transfer into the database.

To populate the results table, the initial step involved extracting the first two principal components from the .eigenvec file. These components were then saved into a new text file, which included columns for sample IDs and their associated populations. Subsequently, the .Q file generated from the ADMIXTURE analysis was appended to this text file. Finally, utilizing the ".mode tabs" command, the content of this text file was imported into the results table.

Flask

The @app.route('/') decorator signifies the route for the home page, where the "home.html" template is rendered, presenting users with a straightforward interface (see figure 9). This page boasts a navigation bar housing links to both the home page and the project's GitHub repository, enhancing accessibility. The central header displays the project's name, "Mandriva." Two distinct buttons, labelled "Analysis" and "SNP," offer clear pathways for users to explore further. Leveraging the url_for function, these buttons redirect users to either the analysis or SNP web pages based on their selection.

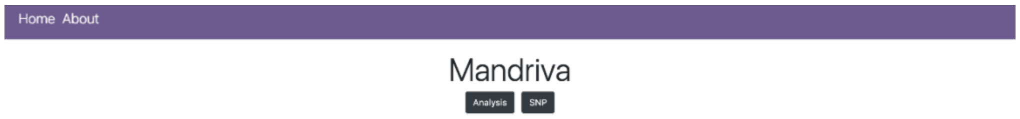


Figure 9. Final design of the home page.

Moving on to @app.route('/analysis'), this decorator defines the route for the analysis page. Upon rendering the "analysis.html" template (see figure 10), users encounter an interface featuring a navigation bar with links to the home page and the project's GitHub repository. The central header bears the title "Analysis," accompanied by two buttons labelled "Cluster Analysis" and "Admixture Analysis," providing users with straightforward options. These buttons, when selected, redirect users to either the cluster or admixture web pages using url_for.

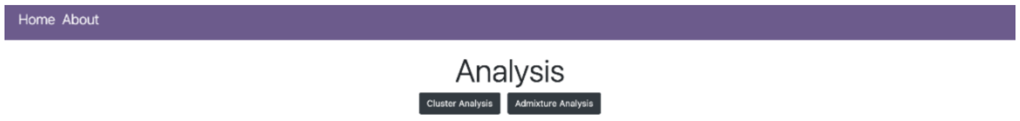
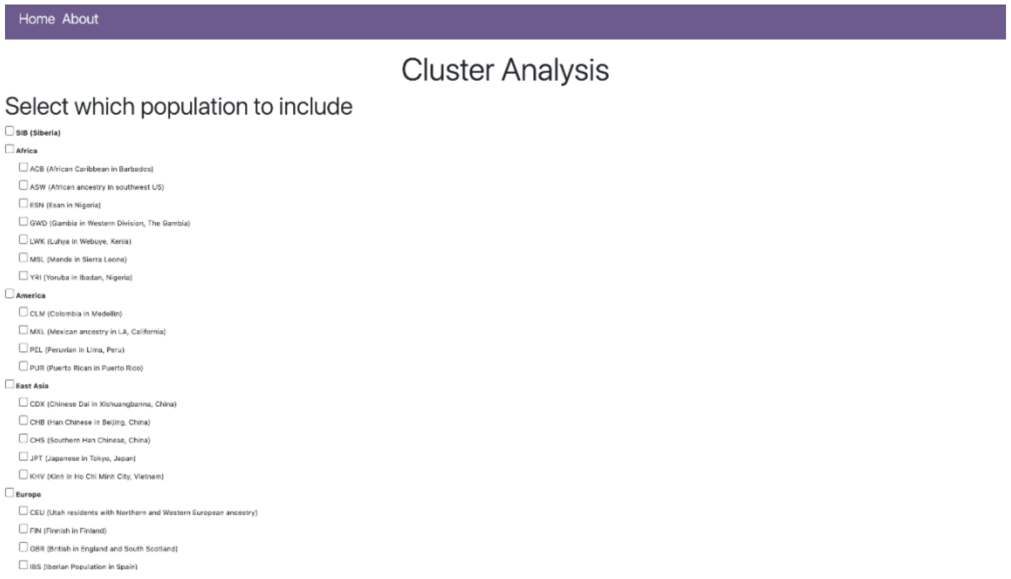


Figure 10. Final design of the analysis selection page.

The @app.route('/cluster', methods=['GET', 'POST']) decorator defines the route for the cluster page, rendering the "cluster.html" template (see figure 11). This template includes a navigation bar with links to the home page and the project's GitHub repository. Checkboxes for selecting populations are provided. Upon submission, a SQL database query retrieves principal component 1 and 2 data for the chosen populations, displayed in the "plot.html" template (see figure 12). This template employs Plotly to present the data in a scatter plot, enabling users to zoom, view samples, and explore population details.



Cluster Analysis

Select which population to include

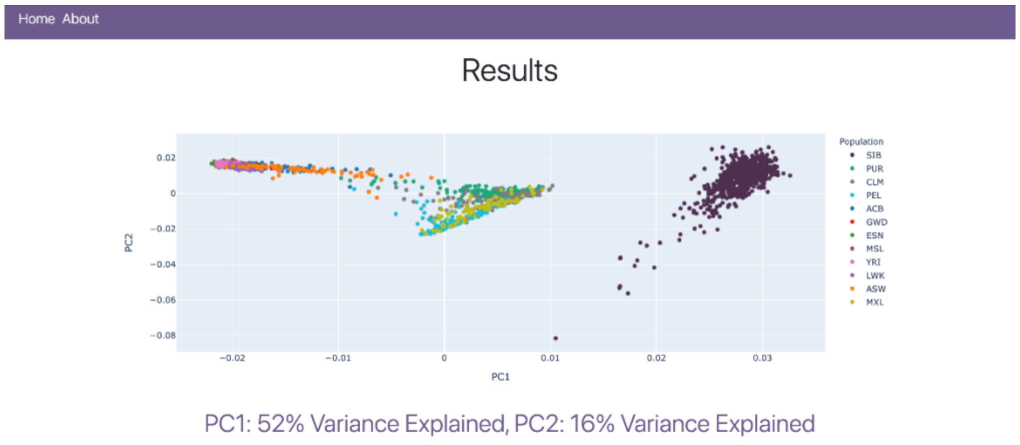
☐ **SIB (Siberia)**
☐ **Africa**
- ☐ ACB (African Caribbean in Barbados)
- ☐ ASW (African ancestry in southwest US)
- ☐ ESN (Esan in Nigeria)
- ☐ GWD (Gambia in Western Division, The Gambia)
- ☐ LWK (Luhya in Webuye, Kenia)
- ☐ MSL (Mende in Sierra Leone)
- ☐ YRI (Yoruba in Ibadan, Nigeria)

☐ **America**
- ☐ CLM (Colombia in Medellin)
- ☐ MXL (Mexican ancestry in LA, California)
- ☐ PEL (Peruvian in Lima, Peru)
- ☐ PUR (Puerto Rican in Puerto Rico)

☐ **East Asia**
- ☐ CDX (Chinese Dai in Xishuangbanna, China)
- ☐ CHB (Han Chinese in Beijing, China)
- ☐ CHS (Southern Han Chinese, China)
- ☐ JPT (Japanese in Tokyo, Japan)
- ☐ KHV (Kinh in Ho Chi Minh City, Vietnam)

☐ **Europe**
- ☐ CEU (Utah residents with Northern and Western European ancestry)
- ☐ FIN (Finnish in Finland)
- ☐ GBR (British in England and South Scotland)
- ☐ IBS (Iberian Population in Spain)

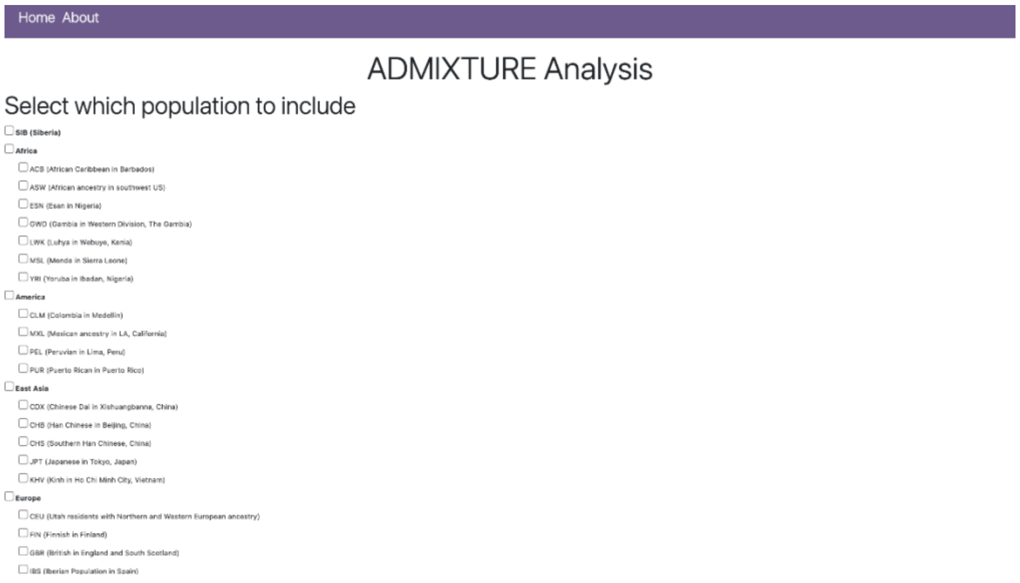
Figure 11. Final design of the cluster population selection page.



PC1: 52% Variance Explained, PC2: 16% Variance Explained

Figure 12. Final design of the cluster results page.

Moving to @app.route('/admixture', methods=['GET', 'POST']), this decorator defines the admixture page, rendering the "admixture.html" template (see figure 13). The template includes navigation bar links to the home page and the project's GitHub repository, along with checkboxes for selecting populations. Upon submission, a SQL database query retrieves proportions of each sample in K1 to K5 ancestry populations, displayed in the "admixture_results.html" template (see figure 14). This template utilizes matplotlib to present the data in a stacked bar plot.



ADMIXTURE Analysis

Select which population to include

- ☐ **SIB (Siberia)**
- ☐ **Africa**
  - ☐ ACB (African Caribbean in Barbados)
  - ☐ ASW (African ancestry in southwest US)
  - ☐ ESN (Esan in Nigeria)
  - ☐ GWD (Gambia in Western Division, The Gambia)
  - ☐ LWK (Luhya in Webuye, Kenia)
  - ☐ MSL (Menda in Sierra Leone)
  - ☐ YRI (Yoruba in Ibadan, Nigeria)
- ☐ **America**
  - ☐ CLM (Colombia in Medellin)
  - ☐ MXL (Mexican ancestry in LA, California)
  - ☐ PEL (Peruvian in Lima, Peru)
  - ☐ PUR (Puerto Rican in Puerto Rico)
- ☐ **East Asia**
  - ☐ CDX (Chinese Dai in Xishuangbanna, China)
  - ☐ CHB (Han Chinese in Beijing, China)
  - ☐ CHS (Southern Han Chinese, China)
  - ☐ JPT (Japanese in Tokyo, Japan)
  - ☐ KHV (Kinh in Ho Chi Minh City, Vietnam)
- ☐ **Europe**
  - ☐ CEU (Utah residents with Northern and Western European ancestry)
  - ☐ FIN (Finnish in Finland)
  - ☐ GBR (British in England and South Scotland)
  - ☐ IBS (Iberian Population in Spain)

Figure 13. Final design of the analysis population selection page.

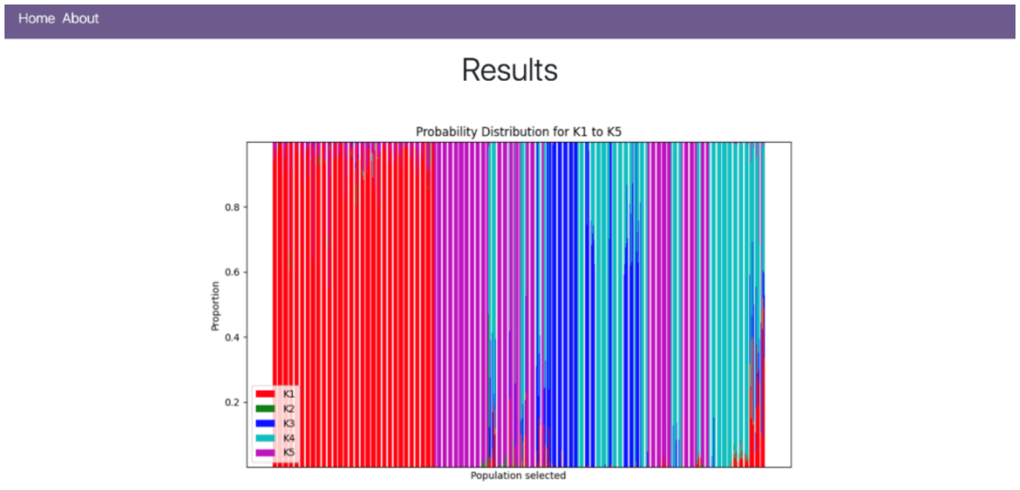
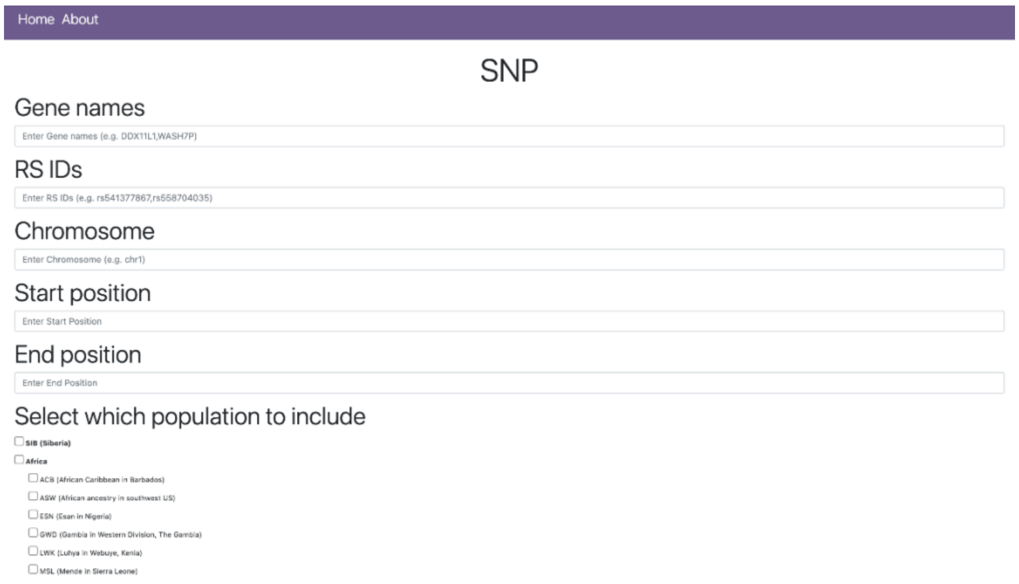


Figure 14. Final design of the admixture results page.

Finally, the @app.route('/snp', methods=['GET', 'POST']) decorator defines the SNP page, rendering the "snp.html" template (see figure 15). This template features a navigation bar with links to the home page and the project's GitHub repository. Additionally, it contains a form with text boxes for gene names (e.g. “DDX11L1,WASH7P”), RS IDs (e.g. “rs541377867,rs558704035”), chromosome, start position, and end position, along with checkboxes for populations. A series of backend processes are triggered upon submission, including SQL database queries, allele and genotype frequency calculations, clinical information fetching. These results are presented in the "result_snp.html" template (see figure 16), complete with a searchable table containing SNPs. Additionally, if multiple populations are selected, a pairwise FST matrix is generated along with a heatmap visualising it. Additionally, a download button to download the pairwise matrix is also generated (see figure 17).



SNP

Gene names

Enter Gene names (e.g. DDX11L1,WASH7P)

RS IDs

Enter RS IDs (e.g. rs541377867,rs558704035)

Chromosome

Enter Chromosome (e.g. chr1)

Start position

Enter Start Position

End position

Enter End Position

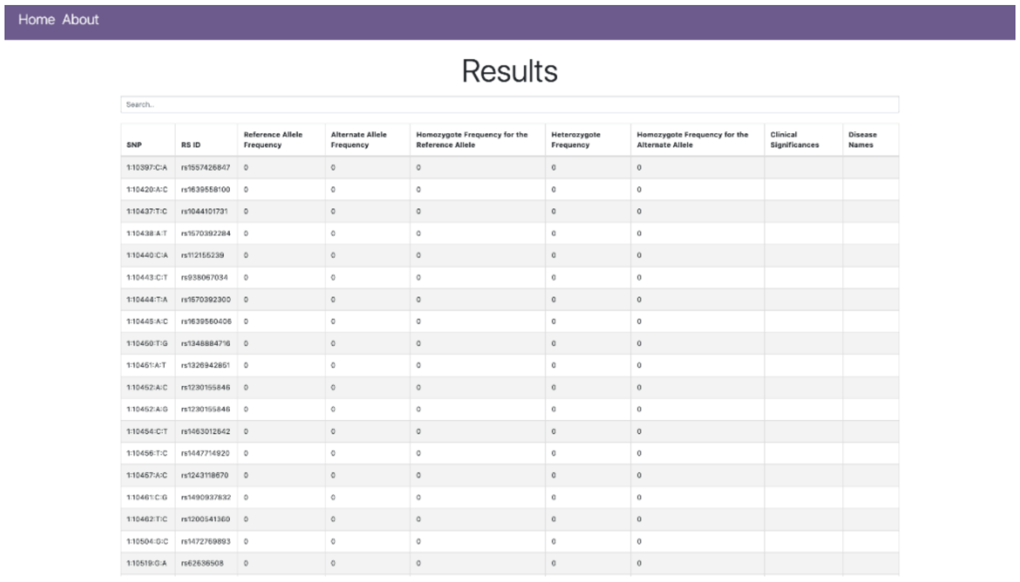
Select which population to include

☐ **SIB (Siberia)**

☐ **Africa**

- ☐ ACB (African Caribbean in Barbados)
- ☐ ASW (African ancestry in southwest US)
- ☐ ESN (Esan in Nigeria)
- ☐ GWD (Gambia in Western Division, The Gambia)
- ☐ LWK (Luhya in Webuye, Kenia)
- ☐ MSL (Mende in Sierra Leone)

Figure 15. Final design of SNP selection form page.



Results

Search..

SNP	RS ID	Reference Allele Frequency	Alternate Allele Frequency	Homozygote Frequency for the Reference Allele	Heterozygote Frequency	Homozygote Frequency for the Alternate Allele	Clinical Significances	Disease Names
1:10397:C:A	rs1557426847	0	0	0	0	0		
1:10420:A:C	rs1639558100	0	0	0	0	0		
1:10437:T:C	rs1044101731	0	0	0	0	0		
1:10438:A:T	rs1570392284	0	0	0	0	0		
1:10440:C:A	rs112155239	0	0	0	0	0		
1:10443:C:T	rs938067034	0	0	0	0	0		
1:10444:T:A	rs1570392300	0	0	0	0	0		
1:10445:A:C	rs1639560406	0	0	0	0	0		
1:10450:T:G	rs1348884716	0	0	0	0	0		
1:10451:A:T	rs1326942861	0	0	0	0	0		
1:10452:A:C	rs1230155846	0	0	0	0	0		
1:10452:A:G	rs1230155846	0	0	0	0	0		
1:10454:C:T	rs1463012642	0	0	0	0	0		
1:10456:T:C	rs1447714920	0	0	0	0	0		
1:10457:A:C	rs1243118670	0	0	0	0	0		
1:10461:C:G	rs1490937832	0	0	0	0	0		
1:10462:T:C	rs1200541360	0	0	0	0	0		
1:10504:G:C	rs1472769893	0	0	0	0	0		
1:10519:G:A	rs62636508	0	0	0	0	0		

Figure 16. Final design of the SNP results page when only one population is queried.

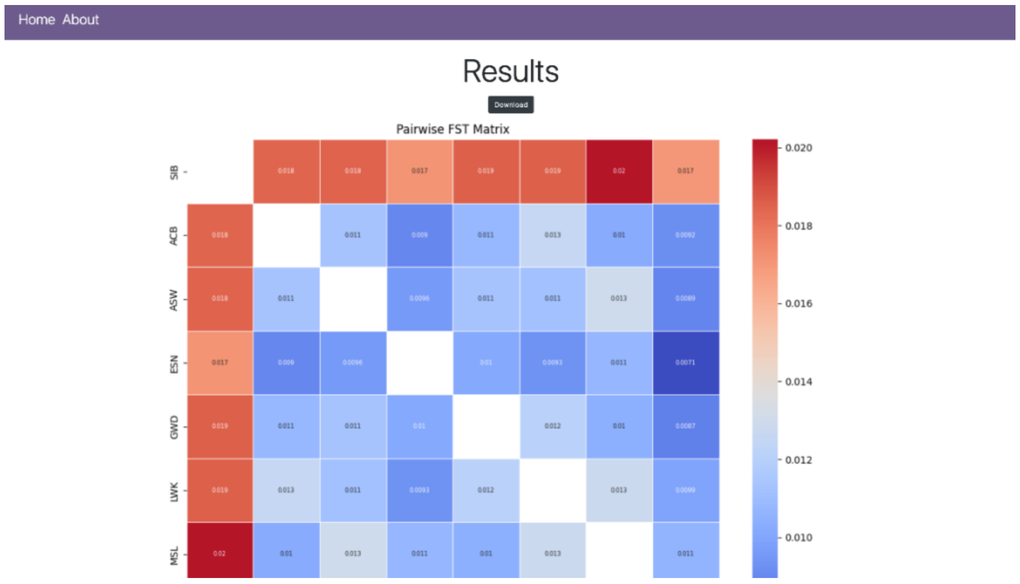
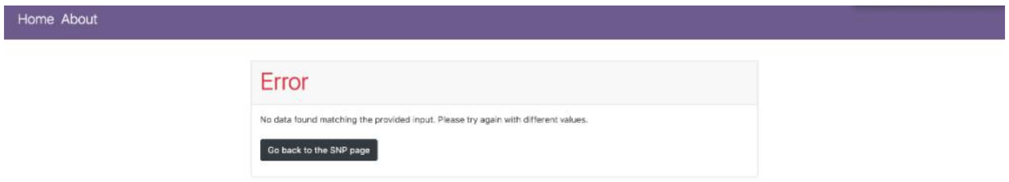


Figure 17. Final design of the SNP results page when multiple populations are queried.

Error Handling

In the web application, robust error-handling mechanisms ensure a seamless and user-friendly experience, even in scenarios where unexpected inputs or issues arise. For instance, when users input data do not present in the SQL database or provide invalid inputs, such as incorrect gene names or RS IDs, the error handling system steps in to guide them. In such cases, the application renders the "error_page.html" template, presenting a clear and concise message informing the user that "No data found matching the provided input (see figure 18). Please try again with different values", along with a button redirecting the user back to the SNP page. This helps users understand the issue and encourages them to revise their inputs for successful data retrieval.



Error

No data found matching the provided input. Please try again with different values.

Figure 18. Error Page - No Data Found Matching Queried Data.

Additionally, specific validations to handle potential errors related to genomic positions have been implemented. If a user inputs too wide of a range for the start and end positions, exceeding the maximum allowed range of 1.5 million, the application detects this discrepancy. In response, it displays a prompt notifying the user that "The range between Start and End positions should not exceed 1.5 million." This proactive validation prevents SQL errors from occurring due to an excessive number of queried variables, maintaining the stability and reliability of the application. Additionally, users are prompted with the error message "Start position must be less than or equal to the End position" if they attempt to input an end position that is lower than the start position.

Furthermore, when users input genomic positions in the chromosome text box, the system ensures that only "chr1" can be inputted as this project only used SNPs from chromosome 1. If the user inputs anything else a prompt is displayed notifying the user that "Chromosome must be 'chr1'". This restriction helps prevent unnecessary database queries and ensures relevant data retrieval.

Finally, the access to different text boxes is dynamically controlled based on user input. For example, if a user enters gene names in the designated text box, the application restricts access to other text boxes, such as RS IDs and genomic positions. Similarly, once users select superpopulations, the application automatically updates the selection status of other text boxes within that superpopulation, streamlining the input process and ensuring coherence in data selection.

By proactively addressing potential errors and providing clear guidance to users, the error handling system ensures a smooth and frustration-free experience, enhancing user satisfaction and overall usability.

Limitations and Future Developments

When conducting the ADMIXTURE analysis, the loop scripted in the HPC cluster for K values 1 to 5 took a week to complete. Although five ancestral populations had the lowest cross-validation value, it did not exhibit an increase, prompting the addition of another loop later for calculating K6 to K10. The second loop revealed that six ancestral populations had the lowest cross-validation score. Unfortunately, due to time constraints, its implementation was not feasible. Future improvements should consider incorporating six ancestral populations for a more accurate analysis.

Additionally, an issue arose when gathering information from America populations and plotting them alongside Africa and Siberia simultaneously. This bug was more apparent when plotting with Plotly, thus the choice to plot with matplotlib was chosen. This anomaly could be attributed to thick borders surrounding the bars, potentially covering other bars. Subsequent development efforts should focus on resolving this issue for a more precise visual representation of the ADMIXTURE results. In addition, when visualizing ADMIXTURE results, there was a challenge in labelling chosen populations on the x-axis. It is important for future updates to tackle this issue. Alternatively, resolving the bug with Plotly could be a solution, given its feature of interactive graphs allowing users to hover and view population details. Furthermore, the inclusion of a download button for obtaining the ADMIXTURE plot would enhance user functionality.

PCA was selected as the clustering technique due to its pattern recognition and dimensionality reduction advantages. While the current choice provides effective results, future iterations of the application could explore the integration of additional clustering techniques like Uniform Manifold Approximation (UMAP) and Projection and Multidimensional Scaling (MDS). This approach ensures user flexibility and the potential for enhanced clustering performance as the application evolves.

Using SQLite for the project was suitable given its lightweight nature and simplicity, making it convenient for initial development and prototyping. However, as the project expands in scale and complexity, especially with the inclusion of more data beyond the first chromosome, it becomes essential to reassess the choice of database manager. Considerations for alternative database managers such as MySQL, SQLAlchemy, or even non-SQL databases like MongoDB arise due to several factors. Firstly, SQLite may struggle with larger datasets, which can lead to performance issues. MySQL, being a robust relational database management system (RDBMS), offers better scalability and performance optimizations suited for handling larger volumes of data. Similarly, SQLAlchemy, a powerful SQL toolkit and Object-Relational Mapping (ORM) library for Python, provides flexibility and abstraction layers that facilitate seamless integration with different database backends. Additionally, non-SQL databases like MongoDB offer advantages such as schema flexibility and horizontal scalability, which may be beneficial for projects requiring rapid iteration and handling diverse data types.

Due to time constraints, thorough testing was not feasible. These testing procedures are essential steps in identifying and rectifying potential vulnerabilities, enhancing the overall quality and reliability of the software. Particularly, unit testing ensures that individual components or modules of the system function correctly in isolation, contributing to the overall stability. End-to-end testing assesses the system's performance, ensuring seamless integration and functionality across different modules. User testing is imperative to validate that the software meets user expectations and requirements, enhancing user experience. Additionally, load testing is essential for evaluating the system's performance under various levels of stress and demand, especially as the website is scaled up. Ensuring thorough testing is crucial for strengthening the software and preventing unexpected issues. Therefore, allocating resources and focusing on enhancing testing procedures in the future is essential to guaranteeing the overall success and quality of the software system.

In future development, it is important to prioritize and implement defensive coding practices against SQL injection. This involves a meticulous approach to handling user input, particularly when entered into the text boxes. By adopting a strategy of double-checking and validating user inputs thoroughly, future developers can mitigate the risk of SQL injection attacks. To achieve this, employing techniques such as parameterized queries or prepared statements is essential. Additionally, incorporating stringent input validation rules becomes paramount. This involves setting clear criteria for the type and format of acceptable input, using tools like regular expressions can filter out any potentially harmful SQL code. These methods ensure that user input is treated as data rather than executable code, minimizing the potential for unauthorized access or manipulation of the database.

In the future, it is beneficial to consider setting up a job scheduler. Instead of waiting on the website, a job scheduler, like a cron job, can handle and execute job requests independently. This becomes especially valuable as the website scales up, ensuring users do not have to wait for their analyses to be completed on the site. This feature becomes particularly useful if users want to choose their own K values for ADMIXTURE analysis. Users can input their desired K value, and the website will run the analysis for the necessary duration independently and without any interruption. Once completed, the results are sent directly to the user, providing a more efficient and user-friendly experience.

Exploring cloud hosting for the website presents several advantages. By opting for cloud hosting, the website can leverage the scalable and reliable infrastructure provided by cloud service providers like AWS (Amazon Web Services) or others. Hosting the website on the cloud offers scalability, meaning that as the website's demand grows, additional resources can be easily allocated to accommodate increased traffic without compromising performance. Additionally, cloud hosting providers like AWS offer a range of services for managing servers, databases, storage, and more. This eliminates the need for manual server management, allowing developers to focus on building and enhancing the website's features and functionality.

References

Hedrick, P.W. (2005) ‘A standardized genetic differentiation measure’, *Evolution*, 59(8).

Kim, S.Y. *et al.* (2011a) ‘Estimation of allele frequency and association mapping using next-generation sequencing data’, *BMC Bioinformatics*, 12(1).

Klug, W.S. and Cummings, M.R. (2002) *Essentials of genetics*. Upper Saddle River, NJ: Prentice Hall.

Liu, C.C., Shringarpure, S., Lange, K. and Novembre, J., 2020. Exploring population structure with admixture models and principal component analysis. Methods Mol Biol, 2090, pp.67-86.

Patterson, N., Price, A. L., & Reich, D. (2006). Population structure and eigenanalysis. PLoS genetics, 2(12)

PLINK 1.90 beta (no date) PLINK 1.9. Available at: https://www.cog-genomics.org/plink/1.9/ (Accessed: 27 February 2024).

Population structure: PCA (no date) Speciation & Population Genomics: a how-to-guide. Available at: https://speciationgenomics.github.io/pca/ (Accessed: 27 February 2024).

Purcell, S., et al. (2007). PLINK: A toolset for whole-genome association and population-based linkage analysis. American Journal of Human Genetics, 81(3)

Slatkin, M., 2008. Linkage disequilibrium—understanding the evolutionary past and mapping the medical future. Nature Reviews Genetics, 9(6), pp.477-485.

University of Vienna (no date) Investigating population structure with ADMIXTURE Available at: https://plantgenomics.univie.ac.at/radseq/admixture/ (Accessed: 28 February 2024).

Wang, J., 2022. Fast and accurate population admixture inference from genotype data from a few microsatellites to millions of SNPs. Heredity, 129(2), pp.79-92.