Software Requirements Specification

for

AnGenoV

Version 1.0 approved

Prepared by
Mert DOĞAN
Sevcan DOĞRAMACI
Tuğba Nur KORKMAZ
Süleyman Emre ÇELİK
Mehmet Gürsel ARSLAN

Konya Food and Agriculture University
Computer Engineering Department

20.12.2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
AnGenoV	20.12.2020	First release	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the first version of the open-source software AnGenoV. AnGenoV is the abbreviation for Analysis of Genomic Variants. The document will explain the purpose, features and the interfaces of the software, what the software will do and the constraints under which it must operate. This document is intended for the users of the software along with developers.

1.2 Document Conventions

This document was created based on the IEEE Software Requirements Specification Template. Glossary terms are written in Italic font in their first uses.

1.3 Intended Audience and Reading Suggestions

This document is intended for users who are interested in exploring *genetic variation* for human diseases and population genetics of humans together with software developers that are interested in the *bioinformatics* field.

The rest of the document is organized in the following fashion:

- In the second section, the main perspective of AnGenoV is described by user functions, its constraints.
- In the third section, external requirements of AnGenoV are stated.
- Functional and non-functional requirements are listed in detail in the fourth and fifth sections respectively.
- The other requirements related to legal and ethical issues are expressed in the sixth section.

In short, the main aspects of AnGenoV can be found in the second section. For further details, users should have a look at the subsequent sections.

Furthermore, a glossary is provided at the end of the document including domain specific terminology.

1.4 Product Scope

AnGenoV is a product that researchers and students can use to explore genetic variation of human genomes. Its main purpose is to create a tool which combines *variant calling*, *variant annotation* and *variant visualization* to make researchers' jobs easier. It serves in command line as well as in a user-friendly web interface for easy usage. It consists of three separate modules: variant calling, variant annotation and variant visualization. It is responsible for performing variant calling, variant annotation by getting required files from the users and visualizing the results. Each module can run individually or all can be executed together. AnGenoV can not be used to explore genetic variation of genomes other than human genomes.

1.5 References

- GitHub Page of AnGenoV:
  https://github.com/SevcanDogramaci/AnGenoV

- User Interface (UI) Design Page of AnGenoV:
  https://www.figma.com/proto/08cYhKwxt6g0xsnIo9Qhwm/Graduation-Project?node-id=139%3A677&scaling=contain

- IEEE Software Requirements Specification Template:
  https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc

- Notion Page of AnGenoV:
  https://www.notion.so/Project-Specification-f4bf78bc3cde445bb5b4e5909d6ee11e

2. Overall Description

2.1 Product Perspective

Living beings are encoded with a sequence called DNA or genome. It hides the genetic information of creatures and is transplanted to next generations. Over time, this genetic sequence undergoes several variations because of mutations or other variation sources. This causes two genomes within a population to be different at variation regions.

Since the genome is the source of hereditary and genetic information, these variants induce certain consequences on organisms, leading hereditary diseases, phenotypic abnormalities and more. Therefore, analysis of genome variants is crucial for inspecting possible consequences which can be helpful for diagnosis of diseases as well as identification of genetic structure causing abnormalities.

In the bioinformatics area, variant calling; the identification of variants on the genome data, variant annotation; associating the genome variants with possible consequences and abnormalities and variant visualization; visualizing the variants with interactive graphics to identify and explore them are the top three tasks. However, people utilize different tools to perform each. Therefore, they need to deal with each tool separately and reach the final conclusion by giving the result of one to the other.

This is where AnGenoV comes for action. AnGenoV is designed for researchers and students, who are interested in exploring genetic variation, to do variant calling, annotation and visualization of human genome variants in one single program, thus enabling them to spend less time and effort.

In the current version of AnGenov focuses on human genomes. It is aimed to increase genome types in future versions.

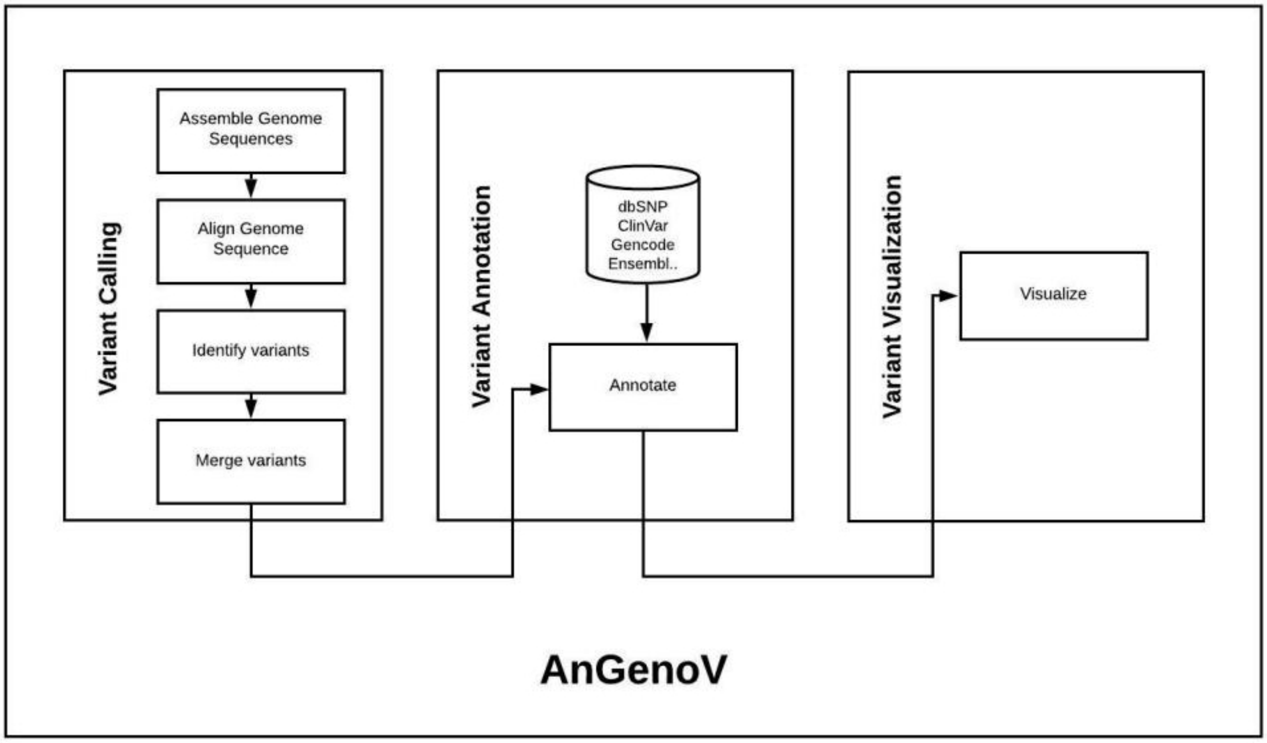


Figure 1. Workflow of AnGenoV

2.2 Product Functions

For AnGenoV, product functions are explained based on its web interface. From the command line all these functions are also available.

Each Window Has:

- Menu bar: Displays name of the product, Home, Variant Calling, Variant Annotation, Docs.
- Home: Opens home page.
- Variant Calling: Opens Variant Calling page.
- Variant Annotation: Opens Variant Annotation page.
- Docs: Opens GitHub documentation of AnGenoV: https://github.com/SevcanDogramaci/AnGenoV
- Context of the related window.
- Close Button: closes the currently selected page.

Main Screens:

1. Home: Displays the homepage.
2. Variant Calling: Displays the Variant Calling Page
3. Variant Annotation: Displays the Variant Annotation Page.

1. Home:

- Displays name of the product, meaning of the name and module options to learn about with buttons named “Learn More”.

- Learn More: Opens the related part of the GitHub documentation of AnGenoV: https://github.com/SevcanDogramaci/AnGenoV

2. Variant Calling: (Steps: Alignment/Assembly, Calling, MergeSV)

2.1 Alignment/Assembly: (Sections: Alignment, Assembly)

- Stepper: Allows users to go to a step (egz. Alignment) they desire.

2.1.1 Alignment:

- Choose Files: Choose a file for alignment one for reference one for sample.
- Align: Creates a bam file as a result of the alignment.
- Save: Saves the output file.
- View: Views the output file.
- Send to Next Step: Sends the generated output file to the Calling step.

2.1.2 Assembly:

- Choose Files: Choose files that have sequences for assembly.
- Assemble: Creates a file by assembling two input files
- Save: Saves the output file.
- View: Views the output file.
- Send to Next Step: Sends the generated output file to Alignment step.

2.2 Calling: (Sections: Input, SNP Calling/ Genotyping/ SV Calling, Outputs)

- Stepper: Allows users to go to a step (egz. Alignment) they desire.

2.2.1 Input:

- Choose File: Choose a file for calling in bam format.
- Choose a Method: Choose a sequencing method (egz. Illumina).
- Choose a Calling Type: Choose a calling type between SNP calling, SV (*Structural Variant*) calling, Genotyping.

2.2.2 SNP Calling/ Genotyping/ SV Calling:

- Choose an Algorithm: Choose an algorithm depending on the chosen calling type (egz. for SV calling: Tardis, Delly etc.)
- Run: Creates vcf files as output.

2.2.3 Outputs:

- Choose Output Files: Choose between created vcf files.
- View: View the vcf files.
- Save: Saves the chosen output files.
- Send to Next Step: Send chosen output vcf files to the MergeSV step.

2.3 MergeSV: (Sections: Merge)

- Stepper: Allows users to go to a step (egz. Alignment) they desire.

2.3.1 Merge:

- Choose Files: Choose files that have SV’s in vcf file format (at least 2 files have to be loaded).
- Choose an Algorithm: Choose an algorithm for merging (egz. Truvari).
- Merge: Merge vcf files.

- Save: Save the output file.
- View: View the output file.

3. **Variant Annotation:** (Sections: Load Input, Results)

3.1 Load Input:

- Choose File: Choose file to load variants in vcf file format.
- Annotate: Annotate the input vcf file.

3.2 Results:

- Table: Display the variants and their annotations.
- Save: Save the output file.

2.3 User Classes and Characteristics

Typical Users, such as students, who want to use AnGenoV for exploring genetic variation in the context of the wealth of genome annotations available for the human genome.

Advanced/Professional Users, such as researchers, who want to use AnGenoV for exploring genetic variation in the context of the wealth of genome annotations available for the human genome and further, for exploring genetic variation for diseases and population genetics.

2.4 Operating Environment

AnGenoV operates on Unix based systems. More detailed information is given in sections 3.2 and 3.3.

2.5 Design and Implementation Constraints

AnGenoV's front end software will be created by using JavaScript and ReactJS. Back end software will be created by using Python and Flask. It will be designed to work from the command line and web interface. It will use SQLite to store data. It will allow users to switch between modules (variant calling, variant annotation, variant visualization). It will be an open source product and will be stored in GitHub.

2.6 User Documentation

- There will be a documentation of AnGenoV on GitHub: https://github.com/SevcanDogramaci/AnGenoV

- Users will be able to reach the documentation both from command line and web interface.

2.7 Assumptions and Dependencies

Third - party Components

- AnGenoV uses annotation sources named 1000 Genomes, dbSNP, dbVar, ClinVar, OMIM, Gencode, Ensembl, COSMIC, ENCODE, HPO to fetch annotations.
- AnGenov uses igv.js for some part of the visualization.

For AnGenoV to work, following requirements must be satisfied on the user's computer:

Hardware Requirements

- There should be enough space on the user's computer memory to download the system.
- To save alignment, calling and annotation files into the user's computer, the user should have enough memory space.

Software Requirements

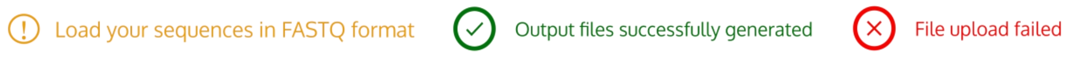
- AnGenoV must run on Unix based systems.
- Python, Node.js and a package manager for Node.js (npm or yarn) must be installed on the operating system.
- Internet connection must exist to fetch data from data sources.

3. External Interface Requirements

3.1 User Interfaces

AnGenoV comes with a web interface for enabling users to operate with a user friendly interface. In the main aspect, it consists of three screens which are home, variant calling and variant annotation pages. All user interface layouts are designed in the Figma prototyping tool before implementation. The prototype screens' reference link can also be found in References.

All screens have a common top navigation bar to get into the modules along with documentations. Furthermore, information messages are shown in orange along with an info icon through all screens, whereas success messages are shown in green along with a tick icon, and error messages are shown in red along with a cross icon. They are illustrated in Figure 2.



Load your sequences in FASTQ format

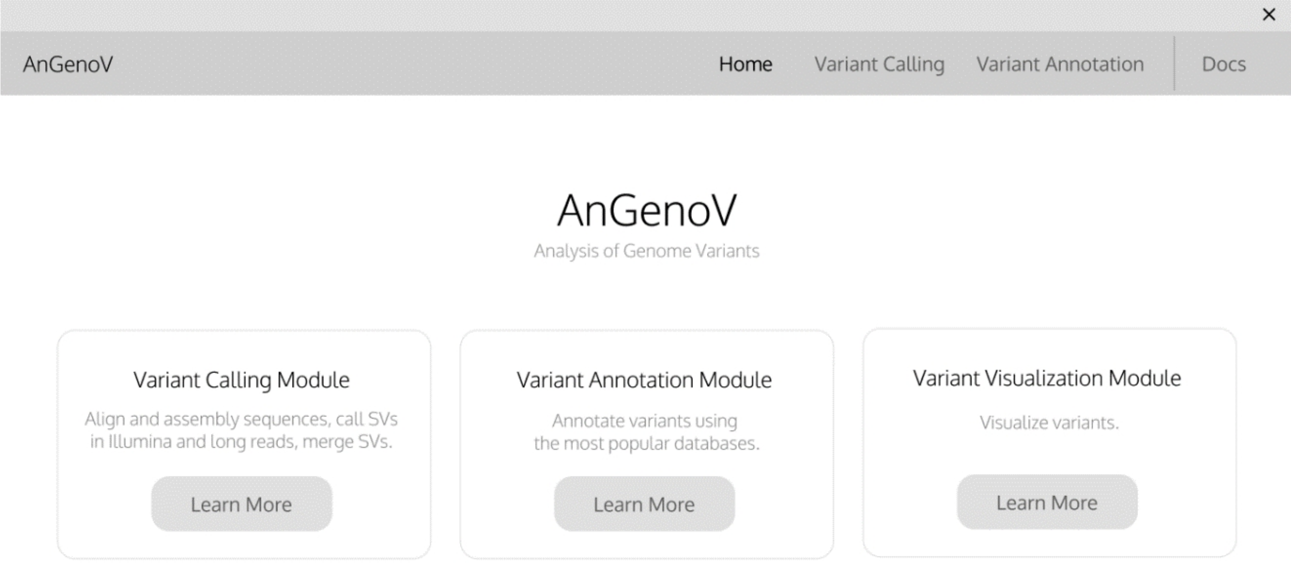
Output files successfully generated

File upload failed

a) b) c)

Figure 2. Information Message (a), Success Message (b), Error Message (c)

3.1.1 Home Screen



AnGenoV

Analysis of Genome Variants

Variant Calling Module

Align and assembly sequences, call SVs in Illumina and long reads, merge SVs.

Variant Annotation Module

Annotate variants using the most popular databases.

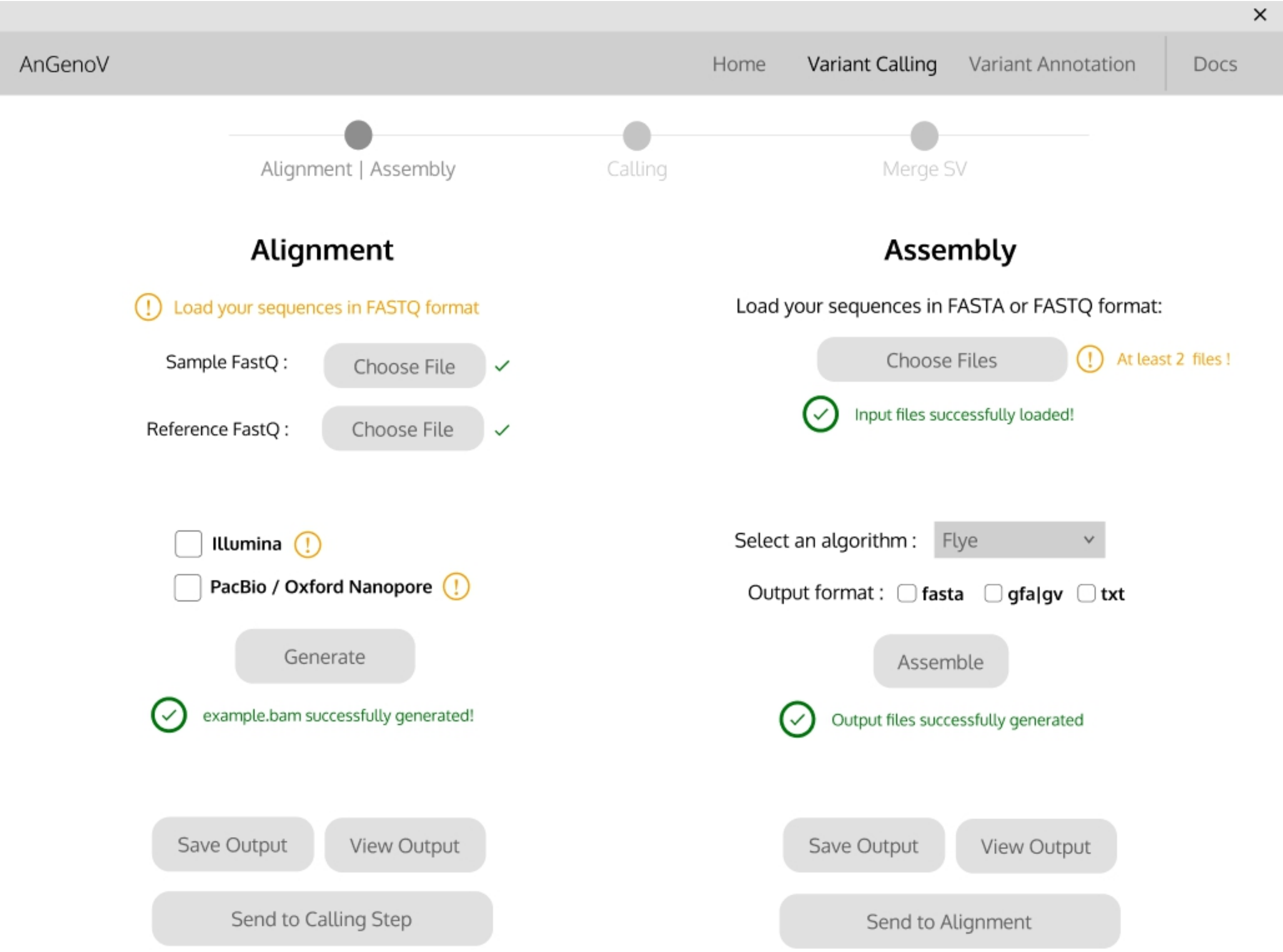
Variant Visualization Module

Visualize variants.

Figure 3. UI Design of AnGenoV's Home Screen

This is the welcoming screen for users. It contains fast links to each module documentation.

3.1.2 Variant Calling - Alignment | Assembly Page



Alignment

Load your sequences in FASTQ format

Sample FastQ : Choose File ✓

Reference FastQ : Choose File ✓

□ Illumina

□ PacBio / Oxford Nanopore

example.bam successfully generated!

Assembly

Load your sequences in FASTA or FASTQ format:

At least 2 files !

Input files successfully loaded!

Select an algorithm : Flye

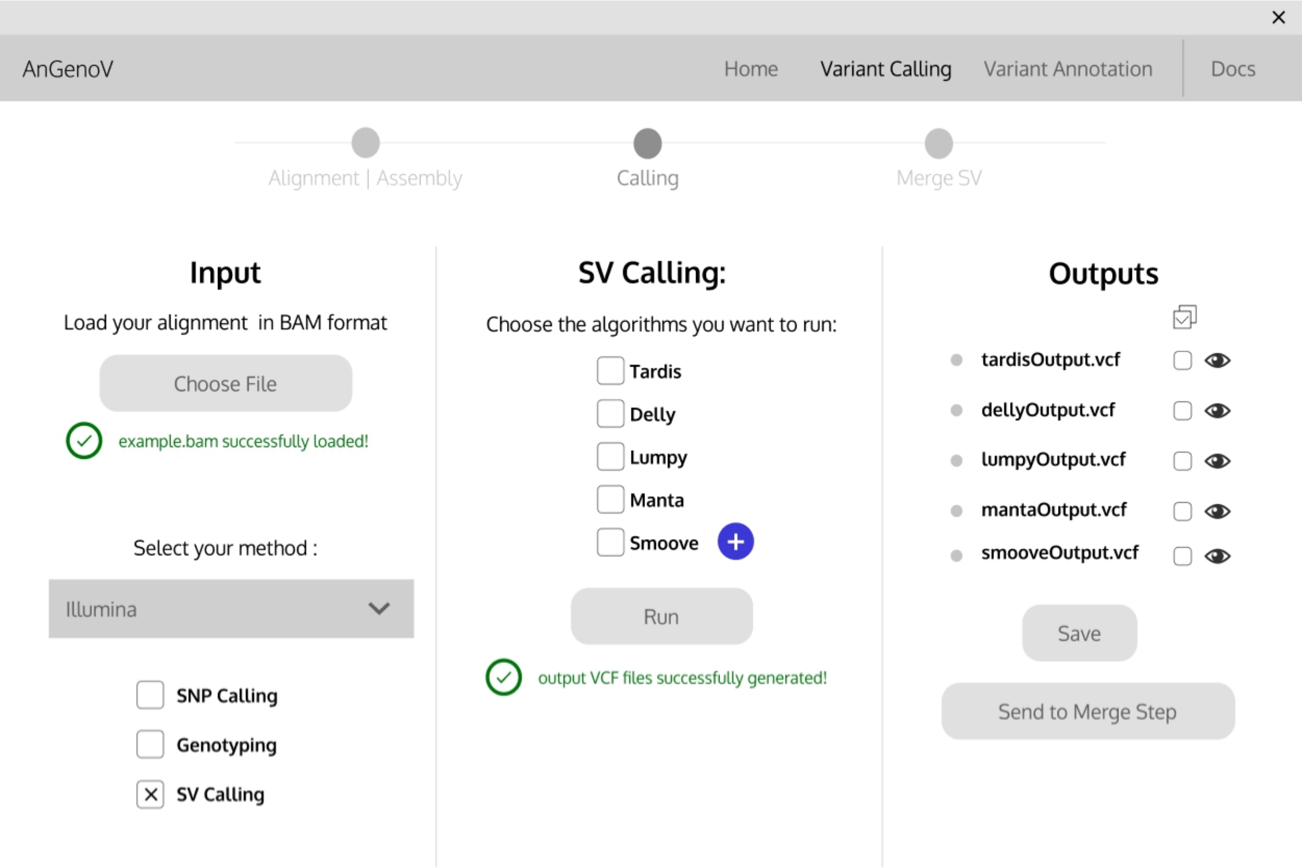
Output format : □ fasta □ gfa|gv □ txt

Output files successfully generated

Figure 4. UI Design of AnGenoV's Variant Calling Screen - 1

When users call for Variant Calling section, this is the first page as shown in Figure 4 they are directed to. It contains instructions to perform Alignment and Assembly steps which are the beginning step for Variant Calling. Users are required to load proper inputs and choose their desired methods on this page. Output manipulations can be done via buttons. The current step of Variant Calling is shown in a stepper at the beginning of the page. Users can navigate through the steps by clicking the step's circle in the stepper, then they are directed to the desired step.

3.1.3 Variant Calling - Calling Page



Load your alignment in BAM format

✓ example.bam successfully loaded!

Select your method : Illumina

□ SNP Calling
□ Genotyping
☒ SV Calling

Choose the algorithms you want to run:

□ Tardis
□ Delly
□ Lumpy
□ Manta
□ Smoove +

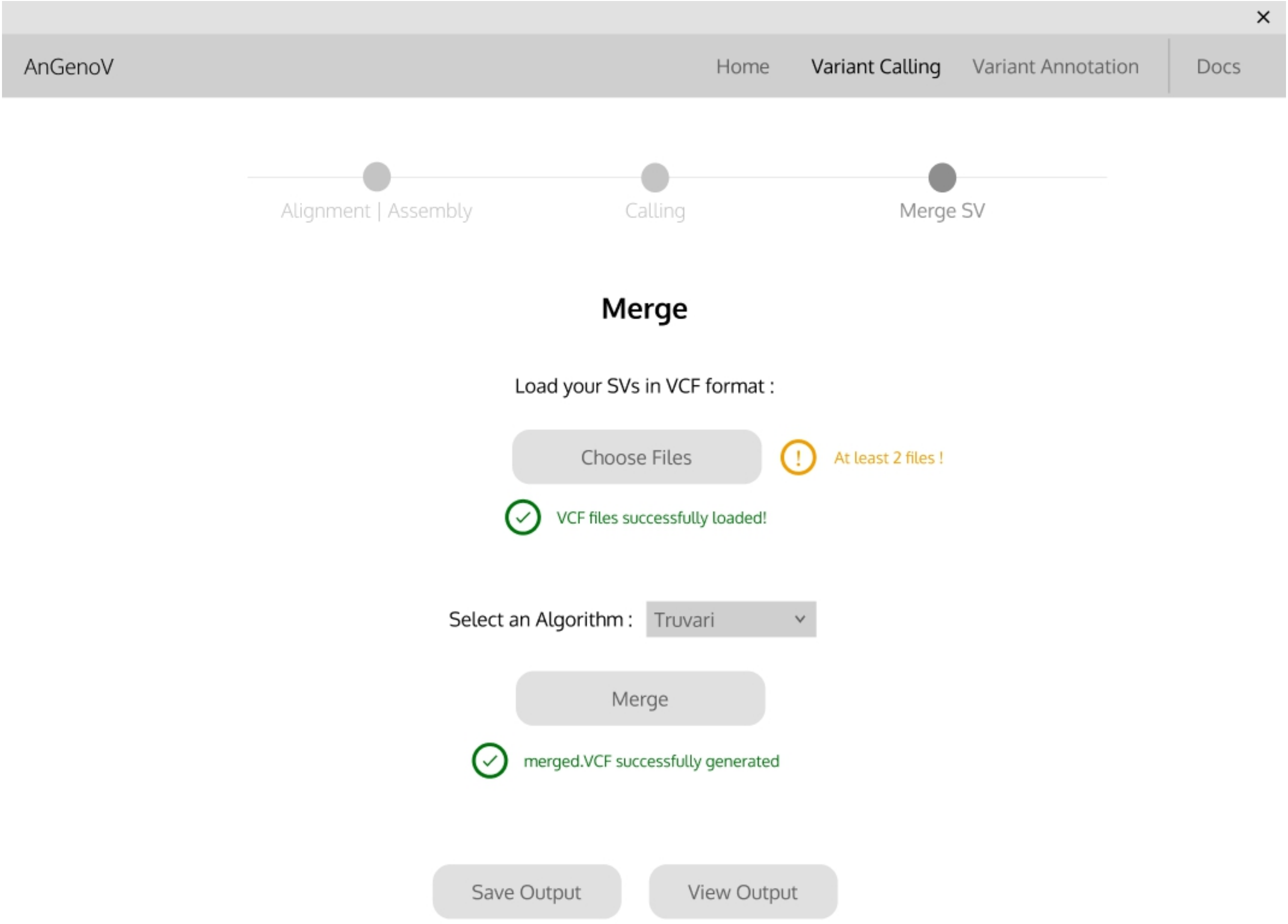
✓ output VCF files successfully generated!

● tardisOutput.vcf □
● dellyOutput.vcf □
● lumpyOutput.vcf □
● mantaOutput.vcf □
● smooveOutput.vcf □

Figure 5. UI Design of AnGenoV's Variant Calling Screen - 2

After completion of the Alignment step, users can click the Calling step. This is the page they are directed to as shown in Figure 5. It contains instructions to perform the Calling step which is separated into three inner phases. Users operate from left to right. In other words, first it is required to supply input files and methods. Second, variant calling by selecting algorithms and output manipulation at the end.

3.1.4 Variant Calling - Merge Page



Load your SVs in VCF format :

At least 2 files !

VCF files successfully loaded!

Select an Algorithm : Truvari

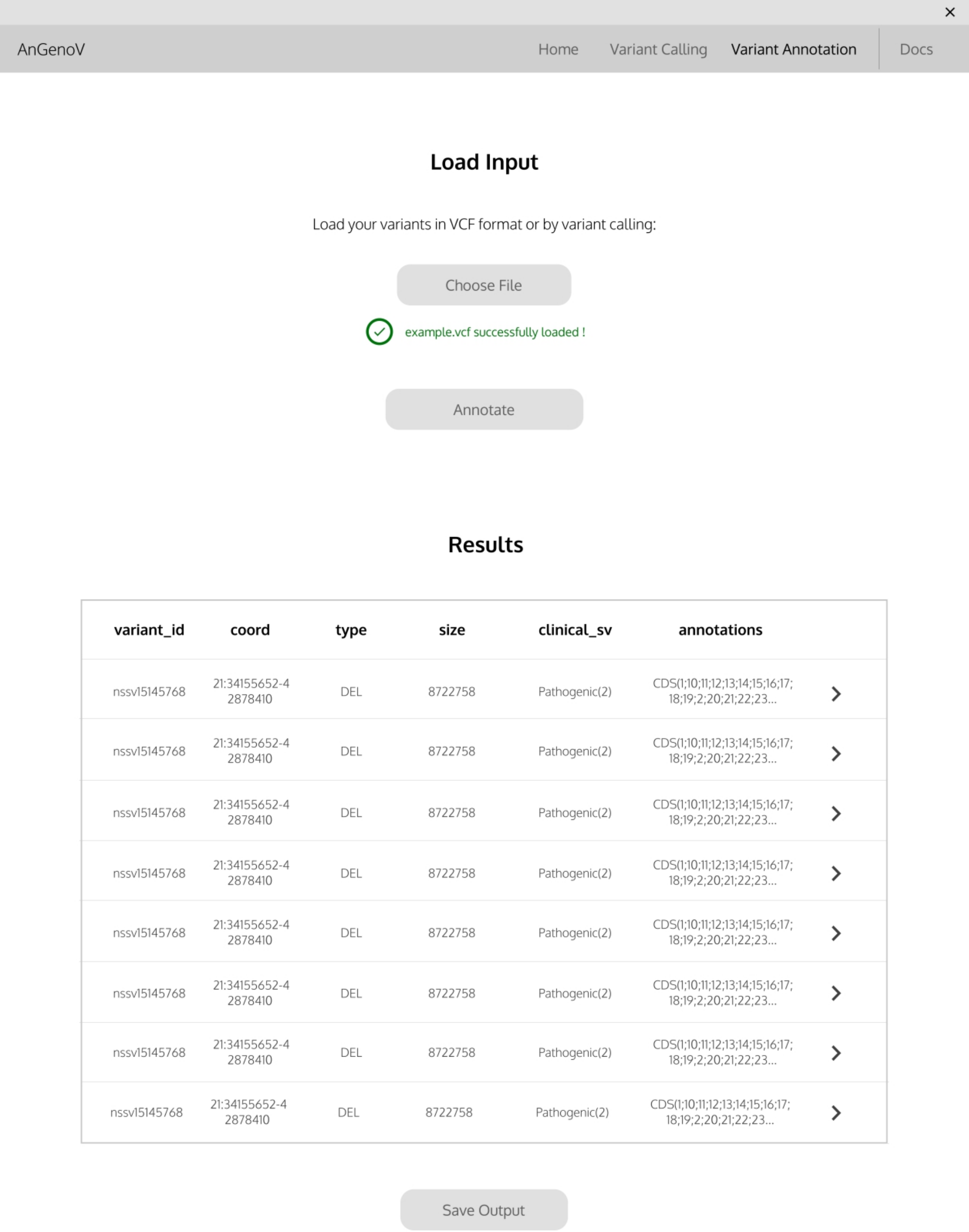
merged.VCF successfully generated

Figure 6. UI Design of AnGenoV's Variant Calling Screen - 3

After completion of the Calling step, users can click the Merge SV step. This is the page they are directed to as shown in Figure 6. It contains instructions to perform the Merge SV step. Users are required to load VCF files containing SVs and choose their method.

3.1.5 Variant Annotation Page

When users call for Variant Annotation section, this is the page that they are directed to, as shown in Figure 7. Users are required to load VCF files to annotate. Annotation results are displayed on a table. Users can move to visualization of a variant by clicking the right arrow in the variant row.



Load Input

Load your variants in VCF format or by variant calling:

✓ example.vcf successfully loaded !

Results

variant_id	coord	type	size	clinical_sv	annotations	
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›
nssv15145768	21:34155652-4 2878410	DEL	8722758	Pathogenic(2)	CDS(1;10;11;12;13;14;15;16;17; 18;19;2;20;21;22;23...	›

Figure 7. UI Design of AnGenoV's Variant Annotation Screen

3.2 Hardware Interfaces

- AnGenoV supports desktop devices with the Unix based systems.
- AnGenoV operates on human genome sequence, alignment and variant data having sizes of GB. Therefore, a sufficient memory and CPU speed must be provided not to affect the overall performance of the user computer while running AnGenoV.

3.3 Software Interfaces

- AnGenoV requires Python with versions higher than 3 to be installed on the user's computer.
- AnGenoV requires Node.js (14.15.2) to be installed on the user's computer and package manager for Node.js. Either npm or yarn package managers can be preferred and installed with their latest versions.
- AnGenoV connects to a SQLite database to store any genome sequence, alignment and variation data.
- AnGenoV uses an Integrative Genomics Viewer (IGV) in the visualization module. To be more specific, igv.js is embedded in the web interface. It will operate upon visualization requests of users both from command line and web interface.
- AnGenoV connects to a public GitHub repository specified by the user to fetch the variant calling algorithm in it.

3.4 Communications Interfaces

- AnGenoV requires an internet connection to get GitHub algorithms in variant calling, annotate variants by connecting annotation databases in variant annotation and view the documentation on GitHub.
- AnGenoV uses REST API calls and FTP to fetch annotation data of variants from annotation databases.
- AnGenoV's fetching time depends on internet connection quality along with the servers' response times.

4. System Features

4.1 Alignment

4.1.1 Description

Alignment is the process of finding out which part of the genome a sequence comes from, relative to the given *reference genome*. It uses the BWA-MEM algorithm for the *Illumina* sequences and the Minimap2 algorithm for the *PacBio* / *Oxford Nanopore* sequences.

4.1.2 Stimulus/Response Sequences

- User uploads sample and reference input files.
- The system checks whether the inputs are appropriate.
- The user selects the sequencing method, Illumina or Oxford Nanopore / PacBio, to determine the algorithm to use.
- The user sends the run command.
- The system runs the appropriate algorithm according to the user's choice.
- After the process is completed and the output file is created, the system displays an informative message.
- The user saves the outputs into the database, views it or directs it to the related step.

4.1.3 Functional Requirements

REQ-1: The system shall allow users to upload only FASTQ or FASTA files from their computer.
REQ-2: The system shall allow users to upload two files as input, one for reference and one for sample.
REQ-3: The system shall display a success message if the files are uploaded successfully, and a failure message if the upload fails.
REQ-4: The system shall allow users to decide Illumina or PacBio / Oxford Nanopore sequencing options to decide the algorithm for alignment.
REQ-5: The system shall display a success message if the output is successfully generated, and an error message if an error occurs while outputting.
REQ-6: The system shall allow users to view, save the output into the database, or send output into the Calling step.

4.2 Assembly

4.2.1 Description

Sequence Assembly is the merging of short segments called DNA *reads* which are read from a longer DNA sequence to reconstruct the original sequence.

4.2.2 Stimulus/Response Sequences

- The user uploads at least two FASTQ or FASTA files as input.
- The system checks whether the inputs are appropriate.
- The user selects an assembly algorithm to use and output formats of the algorithm.
- The user sends the run command.
- The system runs the algorithm selected by the user.
- After the process is completed and the output file is created, the system displays an informative message.
- The user saves the outputs into the database, views it or directs it to the Alignment step.

4.2.3 Functional Requirements

REQ-1: The system shall allow users to upload FASTQ or FASTA files from their computer.
REQ-2: The system shall allow users to upload two or more files as input.

REQ-3: The system shall display a success message if the files are uploaded successfully, and a failure message if the upload fails.
REQ-4: The system shall allow users to select an algorithm for assembly.
REQ-5: The system shall allow users to select output format(s) for assembly.
REQ-6: The system shall display a success message if the output is successfully generated, and an error message if an error occurs while outputting.
REQ-7: The system shall allow users to view, save the output into the database, or send output into the related step.

4.3 Variant Calling

4.3.1 Description

Variant Calling includes SNP calling, Genotyping, and SV calling. In these processes, inputs of BAM file type are generally taken and as a result, outputs of VCF file type are obtained. There are different algorithms customized for each process. Basically, the goal is to find genome locations that are different from what they should be. In the system, the user will be able to choose Illumina or PacBio / Oxford Nanopore methods and access algorithms customized according to these methods.

4.3.2 Stimulus/Response Sequences

- The user uploads a BAM file.
- The user chooses the method they want to use (Illumina or PacBio / Oxford Nanopore)
- The user chooses SNP Calling, Genotyping or SV calling.
- The user selects and runs the algorithms the user wants according to the process the user has selected in the SNP Calling, Genotyping or SV Calling steps.
- The user views, saves the outputs into the database or sends the outputs of the algorithms as input to the Merge step.

4.3.3 Functional Requirements

REQ-1: The system shall allow users to upload a BAM file from their computer.
REQ-2: The system shall allow users to fetch a BAM file from Alignment/Assembly step.
REQ-3: The system shall allow users to decide sequencing methods.
REQ-4: The system shall allow users to decide the calling method.
REQ-5: The system shall allow users to select multiple algorithms of calling methods.
REQ-6: The system shall show an output list of selected algorithms.
REQ-7: The system shall allow users to view each output file.
REQ-8: The system shall allow users to select outputs from the list to save into the database or send into the Merge step.
REQ-9: The system shall display a success message if the file is loaded successfully.
REQ-10: The system shall allow different algorithms to be added via GitHub link.

4.4 Merge

4.4.1 Description

It is the process of merging SVs. Different VCF files can be created as a result of using more than one algorithm in the Variant Calling process. More accurate results can be obtained by merging these results. At least two VCF files are required for this process.

4.4.2 Stimulus/Response Sequences

- The user selects the VCF files from the results of the Calling process or uploads them to the system.
- The user selects the desired algorithm from among the algorithm options.
- The system runs the chosen algorithm.
- The user saves the outputs into the database or views the output.

4.4.3 Functional Requirements

REQ-1: The system shall allow users to upload VCF files from their computer.
REQ-2: The system shall allow users to upload at least two files.
REQ-3: The system shall allow users to fetch VCF files from the Calling stage.
REQ-4: The system shall allow users to choose a merging algorithm.
REQ-5: The system shall allow users to view the output or save it into the database.
REQ-6: The system shall display a success message if the files are loaded successfully.
REQ-7: The system shall display an error message if the loading of the files fails.

4.5 Annotation

4.5.1 Description

Variant annotation is the process of assigning information to DNA variants. There are many different types of information that can be associated with variants, and a first commonly used resource is using databases which contain variants that have previously been described.

4.5.2 Stimulus/Response Sequences

- User uploads an input file in VCF format.
- The system checks whether the input is appropriate.
- The user sends the run command.
- The system annotates the variants.
- After the process is completed, the system shows results in a table format.
- The user can visualize variants or save into the database.

4.5.3 Functional Requirements

REQ-1: System shall allow users to upload a VCF file from their computer.
REQ-2: The system shall display a success message if the file is uploaded successfully, and a failure message if the upload fails.

REQ-3: System shall use annotation sources named 1000 Genomes, dbSNP, dbVar, ClinVar, OMIM, Gencode, Ensembl, COSMIC, ENCODE, HPO to fetch annotations.
REQ-4: System shall connect to the internet to make annotation requests from the annotation sources.
REQ-5: System shall send GET requests to RESTful services of the sources.
REQ-6: System shall show loading while it is fetching the annotation data.
REQ-7: System shall show an error message if it cannot connect to the internet.
REQ-8: System shall show an error message if the sources' services return error responses..
REQ-9: System shall show a success message if annotation fetching is performed successfully.
REQ-10: System shall allow users to save the results of annotation as a text file.
REQ-11: System shall show variant information and annotations for each variant in the VCF file in a table.
REQ-12: System shall allow users to visualize the results of annotation.

4.6 Visualization

4.6.1 Description

It is the visualization part of the annotations resulting from the Variant Annotation process. Integrative Genomics Viewer (IGV) tool is used for visualization. It is displayed through the web interface.

4.6.2 Stimulus/Response Sequences

- The user opens Visualization process from the results of the Variant Annotation process from the interface or from the command line.
- Visualization is displayed through the web interface.

4.6.3 Functional Requirements

REQ-1: The system shall allow users to open Visualization from interface or command line.
REQ-2: The system shall allow users to view Visualization via web interface.

4.7 Documentation

4.7.1 Description

Documentation part is the part where the project is documented and explained how to use it. It will be accessed via the GitHub page.

4.7.2 Stimulus/Response Sequences

- The user goes to the Documentation page from the web interface or command line.
- The system redirects the user to the relevant GitHub page.

4.7.3 Functional Requirements

REQ-1: The system should direct the user to the relevant GitHub page.
REQ-2: The system shall connect to the internet to view the GitHub page.

REQ-3: System shall show an error message if it cannot connect to the internet.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

REQ-1: Downtime of the system should not exceed 10 seconds.
REQ-2: The web interface pages' load time should not be more than one second.
REQ-3: The duration of the process in variant calling should be close to the sum of the specified average running times of the algorithms used in the process.
REQ-4: The duration of annotation fetching should not exceed 60 seconds.

5.2 Safety Requirements

REQ-1: The system shall give correct results so that users are not misinformed.

5.3 Security Requirements

REQ-1: The system shall not share or use users' name, age, gender, nationality, location and other personal information for other purposes.
REQ-2: The system shall not share files uploaded to the system by users and files resulting from the processing of these files or use them for other purposes.

5.4 Software Quality Attributes

REQ-1: The system shall be able to work through the web interface or via the command line.
REQ-2: The web interface has to be user-friendly and easy to use.

5.5 Business Rules

There is no business rule in this system.

6. Other Requirements

REQ-1: Protecting files is the user's responsibility.
REQ-2: The system should have a database to keep variants.
REQ-3: The system should allow downloading the files such as reference genomes or bam files from the database.

Appendix A: Glossary

Term	Definition	Source
Bioinformatics	An interdisciplinary field that develops methods and software tools for understanding biological data.	S1
Genetic Variation	The difference in DNA among individuals or the differences between populations.	S2
Illumina	Short read sequencing of Illumina	S3
Oxford Nanopore	Long read sequencing of Oxford Nanopore Technologies	S4
PacBio	Long read sequencing of Pacific Biosciences.	S5
Read	A raw sequence that comes off a DNA sequencing machine, the segments of DNA.	S6
Reference Genome	A digital database of nucleic acid sequences assembled by scientists that represents the idealized gene set of a species.	S7
Structural Variant	Large genomic alterations typically encompassing at least 50 bp and are classified as deletions, duplications, insertions, inversions, and translocations describing different combinations of DNA gains, losses, or rearrangements.	S8
Variant	A specific region of the genome which differs between two genomes.	S9
Variant Calling	The process by which variants are identified from genome sequence data.	S10
Variant Annotation	The process of assigning information to variants by using variant databases.	S11
Variant Visualization	Visual exploration of genomic variant data.	S12

Appendix B: Analysis Models

Workflow of AnGenoV is shown in Figure 1 in section 2.1.

Appendix C: To Be Determined List

There is no to be determined list in the system.