Text

Description automatically generatedSoftware Requirements Specification

FOR

Blood Bank Management System

Submitted to Manipal University, Jaipur

Towards the partial fulfillment for the Award of the Degree of

**BACHELORS OF TECHNOLOGY**

In Computers Science and Engineering

# 2019-2023

By

Kartik Modi (199301158)

Yash Saraf (199301178)

Miraj Bhuptani (199301196)

**Department of Computer Science and Engineering**

**School of Computing and Information Technology**

**Manipal University Jaipur**

**Jaipur, Rajasthan**

### Table of Contents

[Table of Contents .......................................................................................................... ii Revision History .......................................................................................................... iiii](#_Toc23897)

[1. Introduction 1](#_Toc23898)

[1.1 Purpose 1](#_Toc23899)

[1.2 Document Conventions 1](#_Toc23900)

[1.3 Intended Audience and Reading Suggestions 1](#_Toc23901)

[1.4 Product Scope 1](#_Toc23902)

[1.5 References 2](#_Toc23903)

[2. Overall Description 3](#_Toc23904)

[2.1 Product Perspective 3](#_Toc23905)

[2.2 Product Functions 3](#_Toc23906)

[2.3 User Classes and Characteristics 7](#_Toc23907)

[2.4 Operating Environment 8](#_Toc23908)

[2.5 Design and Implementation Constraints 8](#_Toc23909)

[2.6 User Documentation 8](#_Toc23910)

[2.7 Assumptions and Dependencies 9](#_Toc23911)

[3. External Interface Requirements 10](#_Toc23912)

[3.1 User Interfaces 10](#_Toc23913)

[3.2 Hardware Interfaces 18](#_Toc23914)

[3.3 Software Interfaces 19](#_Toc23915)

[3.4 Communications Interfaces 19](#_Toc23916)

[4. System Features 19](#_Toc23917)

[4.1 Graph Visualization 19](#_Toc23918)

[4.2 Graph Layout 22](#_Toc23919)

[4.3 Graph Metrics 23](#_Toc23920)

[4.4 Filters 24](#_Toc23921)

[4.5 Data Table 25](#_Toc23922)

[4.6 Dynamic Graphs 28](#_Toc23923)

[4.7 Graph Export 29](#_Toc23924)

[5. Other Nonfunctional Requirements 30](#_Toc23925)

[5.1 Performance Requirements 30](#_Toc23926)

[5.2 Availablity Requirements 30](#_Toc23927)

[5.3 Security Requirements 30](#_Toc23928)

[5.4 Reliabilty Requirements 30](#_Toc23929)

[Glossary 31](#_Toc23930)

## 

## 1. Introduction

# 1.1 Purpose

The DEVELOPERS aimed to develop and implement an online blood bank management system. This web-based application allows hospitals to make inventories of their blood bags online, thus, allowing each hospital to check the availability of blood bags anytime. Likewise, proper accounting of blood donors ensures that the expected blood transfusion services will be safe and secured.

# 1.2 Document Conventions

This Document was created based on the IEEE template for System Requirement Specification Documents.

# 1.3 Intended Audience:

* **Donors**: People who are willing to donate their blood.
* **Receivers**: People want the blood
* **Administration**: The system administration.

# 1.4 Product Scope:

The system functions and features of our system will include the following:

• **Registration**

This function allows the donor and administrator to register as a user to interact with the system. The system requires the user to login before viewing and editing

any information.

• **View and edit information online**

Donors are allowed to view their blood donation records online by their given

account. They can also edit their personal information through the system.

• **Data is input by the Administrators**

The donor’s information and donation records can be sent from the hospital to the administrator by calling or e-mail. The administrator is responsible for keying the

received data into the system.

• **Recording donation records**

The system is able to record data of whole blood which is sent from the hospital.

• **Manage blood inventory**

The system uses a First-In-First-Out stock management, where the blood stock that is checked-in to the system first will be the first one given to the hospital when requested. When the blood stock is expired, the administrator is responsible for

removing the stock from the inventory and updating the system.

• **Blood requests**

The hospital can request blood via e-mail and by calling to the blood bank.

• **Notify by E-mail**

The donor’s account and generated password will be sent via e-mail, following by their blood result of the previous donation sent in a separated e-mail. Hospitals can also receive e-mail responding to their requested blood whether it is available in

our stock or not.

• **Summary report**

The system is able to generate a report to summarize all records including blood donation, blood requests and blood stock for the administrator.

.

# 1.5 References

* [https://www.blood.co.uk/the-donation-process/further-information/tests-we-carryout/](https://www.blood.co.uk/the-donation-process/further-information/tests-we-carry-out/)
* <http://www.redcrossblood.org/donating-blood/donation-faqs>
* <http://www.redcrossblood.org/learn-about-blood/blood-types>
* <http://www.nhs.uk/Conditions/Blood-groups/Pages/Introduction.aspx>
* <http://www.webmd.com/a-to-z-guides/blood-type-test#1>
* <https://en.wikipedia.org/wiki/Blood_type>
* <https://en.wikipedia.org/wiki/Rh_blood_group_system>
* [http://www.mayoclinic.org/tests-procedures/rh-factor/basics/definition/ prc-20013476](http://www.mayoclinic.org/tests-procedures/rh-factor/basics/definition/prc-20013476)
* <https://goo.gl/nsUFwy>

## 2. Overall Description

# 2.1 Product Perspective

The persons who like to donate blood registers in my site as well as he can modify the details if necessary, giving the Login Id and Password. The persons in need of blood searches for the persons having the same blood group and with in the city. If he found a donor in his city then he gets the total details of the donor, if he doesn’t find any donor then he is given the contact numbers and addresses of the Life Saving Contact Persons for major cities. If he doesn’t have any chance to contact them then he will be provided with Mobilink Paging Services in order to get the blood.

# 2.2 Product Functions

File:

* New Project: Creation of a new project
* Open: Loads an existing project or a file of one of the supported graph formats  Open Recent: Loads one of the displayed, recently opened files.
* Close Project: Closes the currently open project.
* Properties: Displays some properties of the project (such as the title) which can be edited.
* Import Spreadsheet: Loads a CSV file (edges table or nodes table).
* Import Database -> Edges List: Loads the edges of a graph from a Database (MySQL, SQLite, PostgreSQL)
* Generate -> Dynamic Graph Example: Creates a Dynamic Graph from scratch.
* Generate -> Multi-Graph Example: Creates a Multi-Graph from scratch.
* Generate -> Random Graph: Creates a Random type of graph from scratch.
* Save: Saves the project without changing its name or directory.

Save as: Saves the project and gives the user the ability to change its name or directory.

Export -> Graph file: Exports the current project’s graph in one of the supported file formats

* Export -> SVG/PNG/PDF file: Exports a screenshot of the current project’s graph in one of the available formats (SVG, PNG or PDF).
* Exit: Gephi shuts down

Workspace:

* New: Creates a new workspace.
* Delete: Deletes the current workspace.
* Rename: Renames the current workspace

Tools:

* Plugins: Here the user can change his plugin settings (update already installed plugins, install new plugins etc.).
* Options: Here the user can change some of Gephi’s settings (key bindings etc.).  Language: Changes the language to one of the languages displayed.
* Online docs and support: Redirects the user to <https://gephi.org/users/support/>

Window:

* Context
* Data Table: Displays the Data Table tab.
* Filters: Displays the Filters tab.
* Graph: Displays the Graph tab.
* Layout: Displays the Layout tab.
* Output: Displays the Output-Log tab.
* Preview: Displays the Preview tab.
* Preview Settings: Displays the Preview Settings tab.
* Appearance: Displays the Appearance tab.

Welcome: Displays the Welcome window.

* Statistics: Displays the statistics tab.
* Timeline: Displays the timeline tab.
* Configure Window: Here the user can select one of the available display options for the currently selected tab.
* Close Window: Closes the currently selected tab
* Document Group -> New Document Group: Creates a new Document Group.
* Document Group -> Manage: Selects or removes one of the existing Document Groups.

Help

* Check for Updates: Displays the plugins that can be updated to newer versions
* About: Displays the logo of Gephi, which licenses are being used, the product version and other info.

Main Pages:

* Overview: Displays the Overview page, which by default includes the tabs:

Appearance, Layout, Graph, Context, Filters, Statistics and Queries.

* Data Laboratory: Displays the Data Laboratory page, which by default includes the tab Data Table.
* Preview: Displays the Preview page, which by default includes the tabs: Preview and Preview Settings.

Overview:

* Appearance: Here the user can change the color of the edges/nodes. He can choose one color for all the edges/nodes or choose multiple colors, according to the values of a certain attribute.
* Layout: Here the user can change the layout of the graph by using one of the available layouts.

Graph: Here the user can interact directly with the graph itself, by dragging nodes, zooming in or out etc.

Context: Displays the number of nodes and edges and the type of the graph.

* Filters: Here the user can choose which filters to apply on the graph. (the filters are being applied by drag and drop on to the “Queries” tab).
* Statistics: Here the user can choose which of the graph statistics will be calculated

(by pressing on the button “Run” of the desired statistic).

* Queries: This is the module where filters are being placed.

Data Laboratory:

* Nodes: Displays the Nodes table
* Edges: Displays the Edges table
* Configuration: Preferences about how the data is presented.
* Add node: Creates a new node and adds it to the graph.
* Add edge: Creates a new edge between two existing nodes of the graph  Search/Replace: Stand Search/Replace functionality.
* Import Spreadsheet: Opens a dialog to load a csv file (edges or nodes table)
* Export Table: Exports the table as a csv file, the user can choose which attributes will be included in the file.
* More Actions -> Clear Graph: All nodes and edges of the graph will be deleted.
* More Actions -> Clear Edges: All edges of the graph will be deleted.
* More Actions -> Detect and merge node duplicates: Removes node duplicates by merging them into one node.
* More Actions -> Manage Dynamic Column Estimators:
* Filter: Used to find all the rows of the table that contain a specific value at an attribute selected by the user.
* Add column: Adds a new attribute to the table (edges/nodes)
* Merge columns: Combines the selected columns into one. The merge strategy must be selected by the user.

Delete column: Removes all the values of a column and the column itself from the table.

Clear column: For every row sets the selected column empty.

* Copy data to other column: For every row the value of the target column becomes identical to the value of the source column.
* Fill column with a value: For every row the value of the selected column is set to the specified one.
* Duplicate a column: Creates a duplicate of the selected column.
* Create a boolean column from regex match: Creates a new boolean column, the value of which depends on whether the value of the selected existing column matches a regular expression or not. The regular expression must be specified by the user.
* Create column with list of regex matching groups: Creates a new column. For every row the value of this column is a list about how the value of the selected existing column matches with the user’s regular expression.
* Convert column to dynamic: Converts the selected static column into a dynamic one.

Preview

* Preview Settings: Here the user can configure rendering settings.
* Preview Ratio: When set under 100% a partial graph is displayed.
* Refresh: Applies changes made by the user.
* Preview: After changes have been applied, the updated graph will be shown here.

# 2.3 User Classes and Characteristics

There are two internal users involved in this system. The user requirements are considered as follows:

1. **Donor**

* To be able to view their donation records, including where and when they made donations, and the blood results for each, to learn of their donated blood quality and schedule their next donations. (Solving P3, P6)
* To be able to view and update their personal information, including name, contact address, and phone number, to keep their donor’s information record up-to-date with the blood bank. (Solving P1)
* To be notified of the blood results of their previous donation by e-mail, to know the success of their donation. (Solving P4)

1. **Administrator**

* To be able to create, update, delete, and query donor’s records in order to

manage donor information.

* To be able to create, update, delete, and retrieve donation records to manage

information about donations made.

* To be able to deposit donated blood into inventory when donations are made.
* To be able to withdraw blood from the inventory and keep a record of blood stocks to always keep count of the blood bags. (Solving P5)
* To be able to create, update, delete, and retrieve request records from hospitals to manage hospital requests for blood.
* To be able to create, update, delete, and query hospital’s records in order to

manage hospital information.

* To be able to send e-mails to donors for their user account and blood results

through the system. (Solving P6)

* To be able to send e-mail responding to hospitals for their blood requests through the system. (Solving P6)

# 2.4 Operating Environment

* Windows 2000  Windows XP
* Windows Vista
* Windows 7
* Windows 8
* Windows 10
* Mac OS X
* Linux

# 2.5 Design and Implementation Constraints

This research study does not cover the actual blood collection activity, and actual blood transfusion operation. Blood donors and patients or recipients of blood donation are not system users, their registration or information will be encoded by the blood bank receptionists

# 2.6 User Documentation

* <https://en.wikipedia.org/wiki/Blood_type>
* <https://en.wikipedia.org/wiki/Rh_blood_group_system>
* [http://www.mayoclinic.org/tests-procedures/rh-factor/basics/definition/ prc-20013476](http://www.mayoclinic.org/tests-procedures/rh-factor/basics/definition/prc-20013476)
* <http://anthro.palomar.edu/blood/ABO_system.htm>
* <http://www.redcrossblood.org/learn-about-blood/blood-testing>
* <http://www.donateblood.com.au/eligibility/blood-testing-and-safety>

# 2.7 Assumptions and Dependencies

The researchers assume the following assumptions:

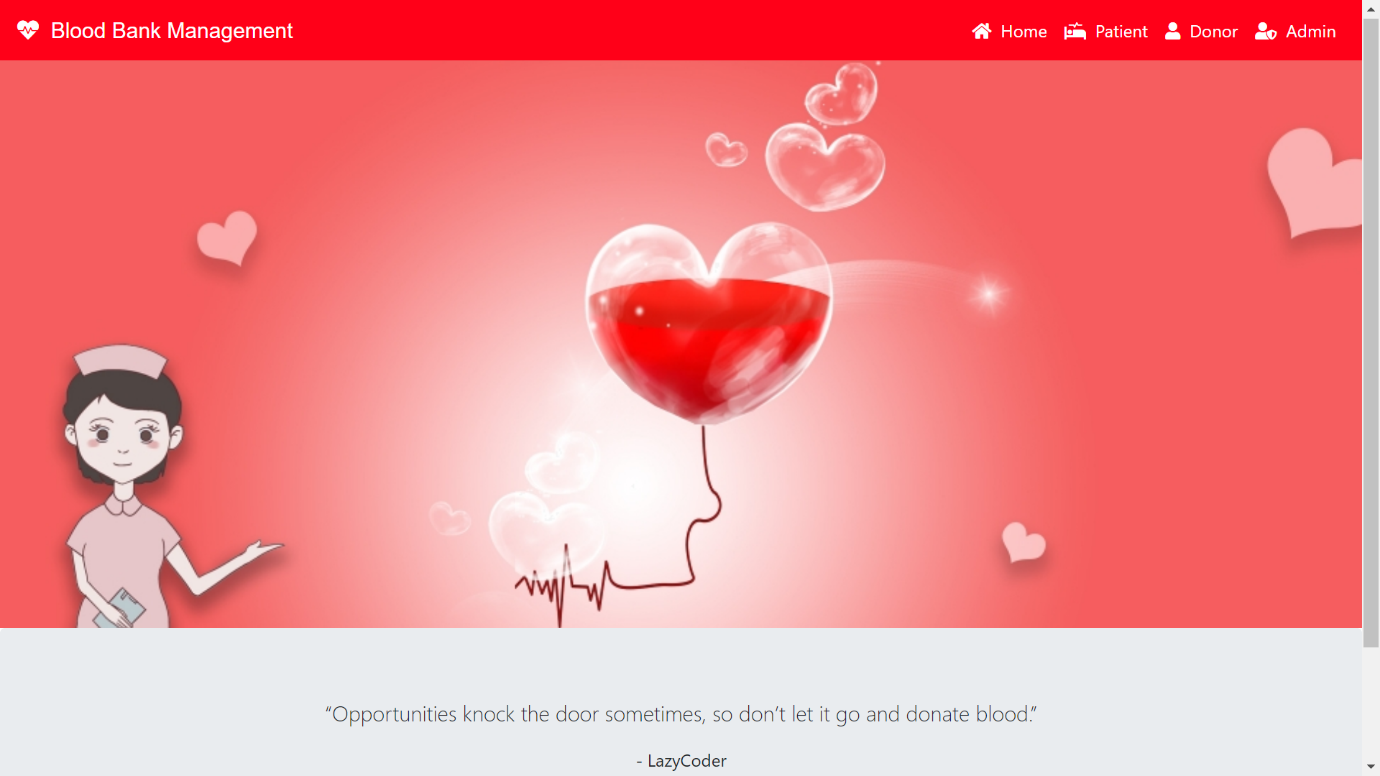
1. Internet connectivity is needed for the online blood management system. Internet speed may affect the perception of the systems users with regards to the system effectiveness and efficiency.

2. Blood transfusion should be performed by medical or professional doctors only. The overall safety depends on the success of the medical operation

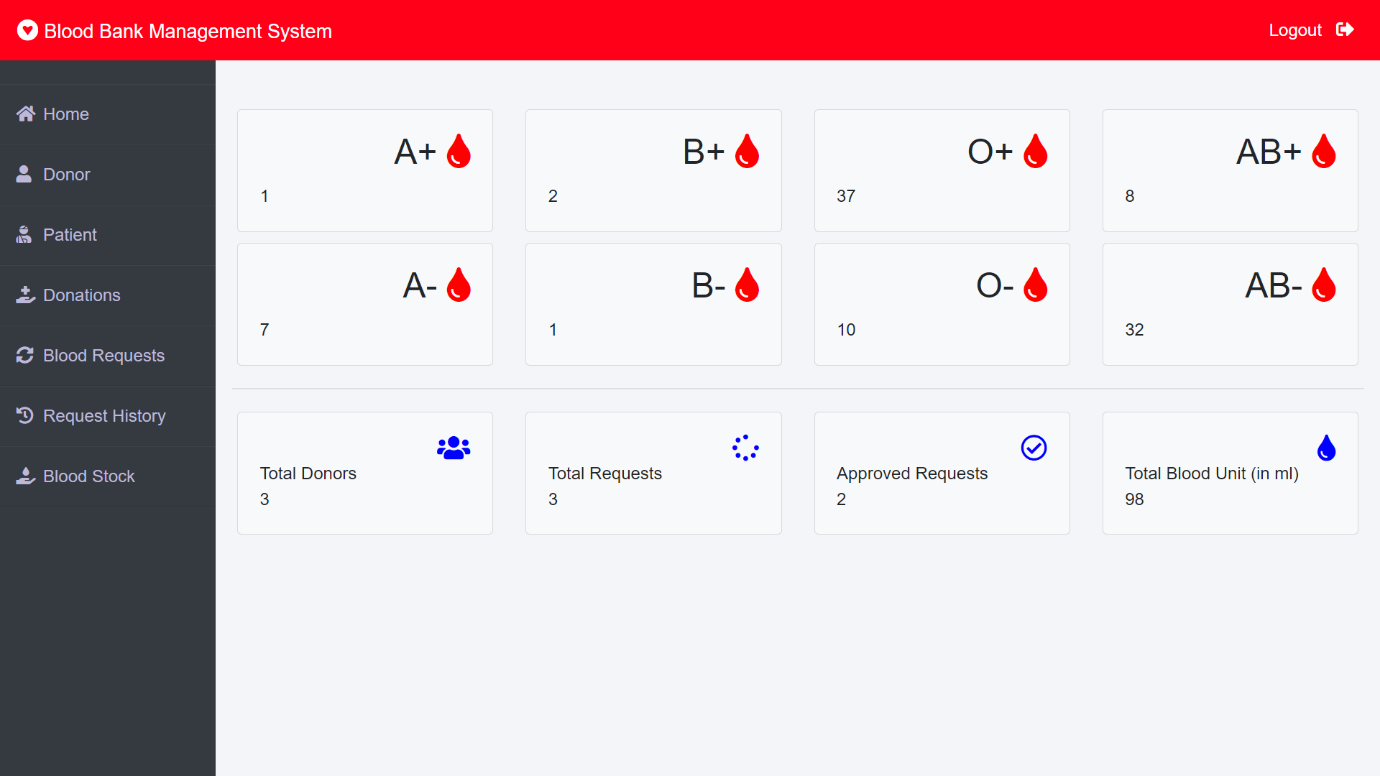
## 3. External Interface Requirements

# 3.1 User Interfaces

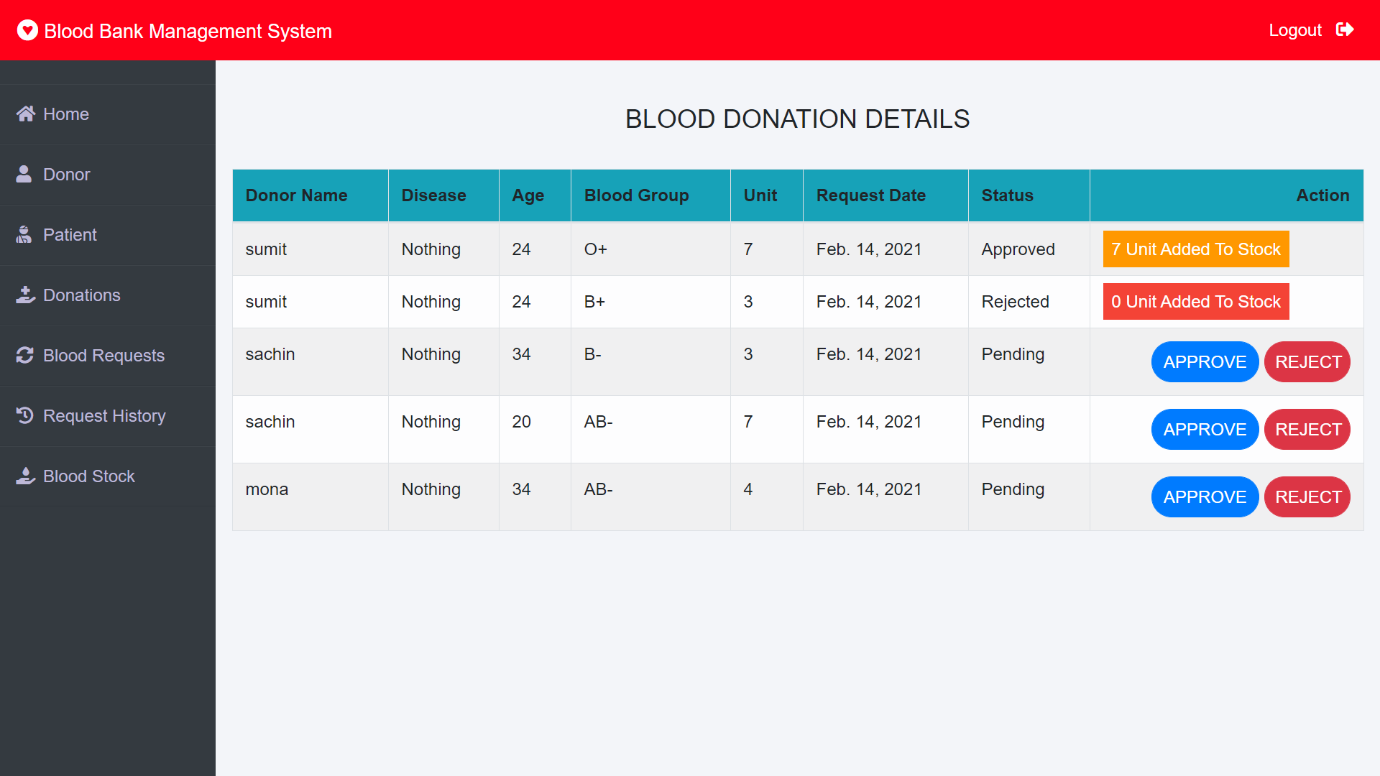
**Homepage**

[](https://github.com/sumitkumar1503/bloodbankmanagement/blob/master/static/screenshot/homepage.png?raw=true)

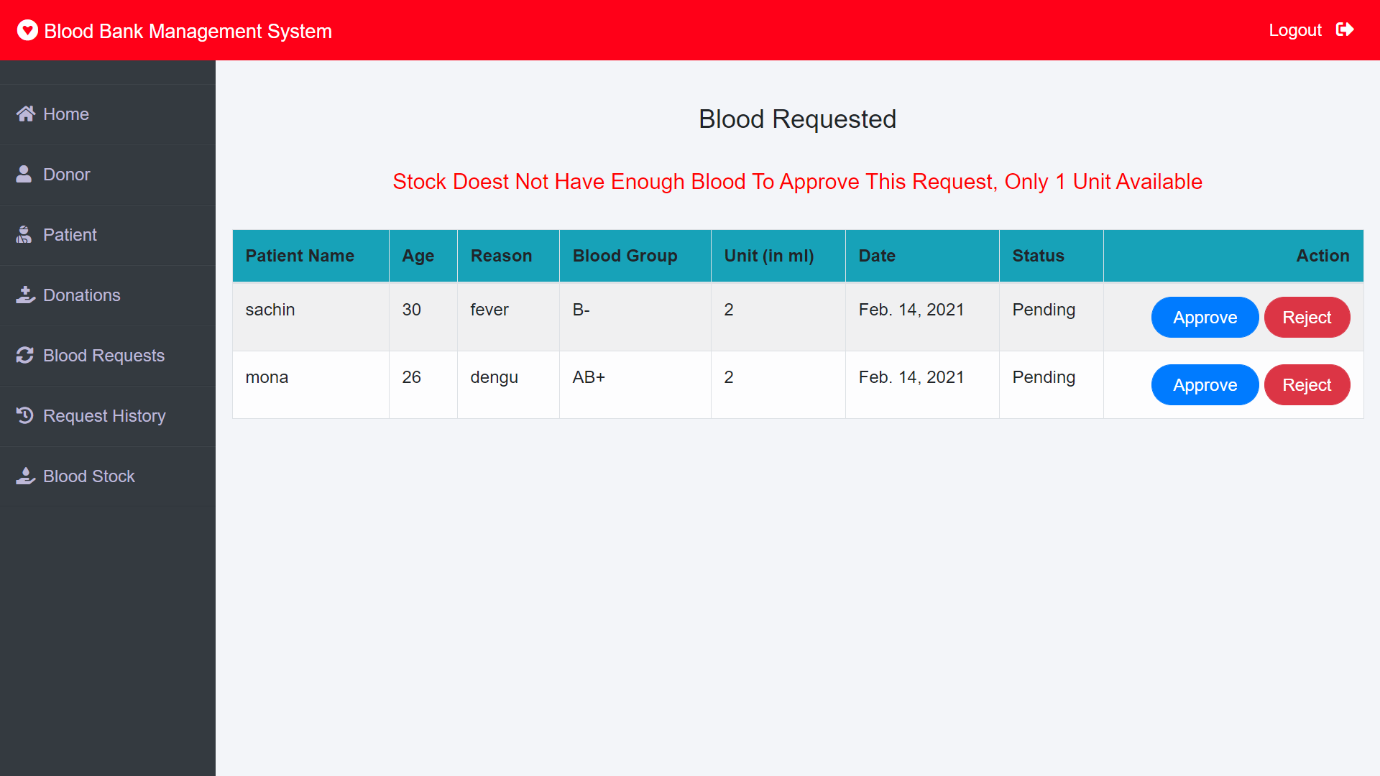
**Admin Dashboard**

[](https://github.com/sumitkumar1503/bloodbankmanagement/blob/master/static/screenshot/admindashboard.png?raw=true)

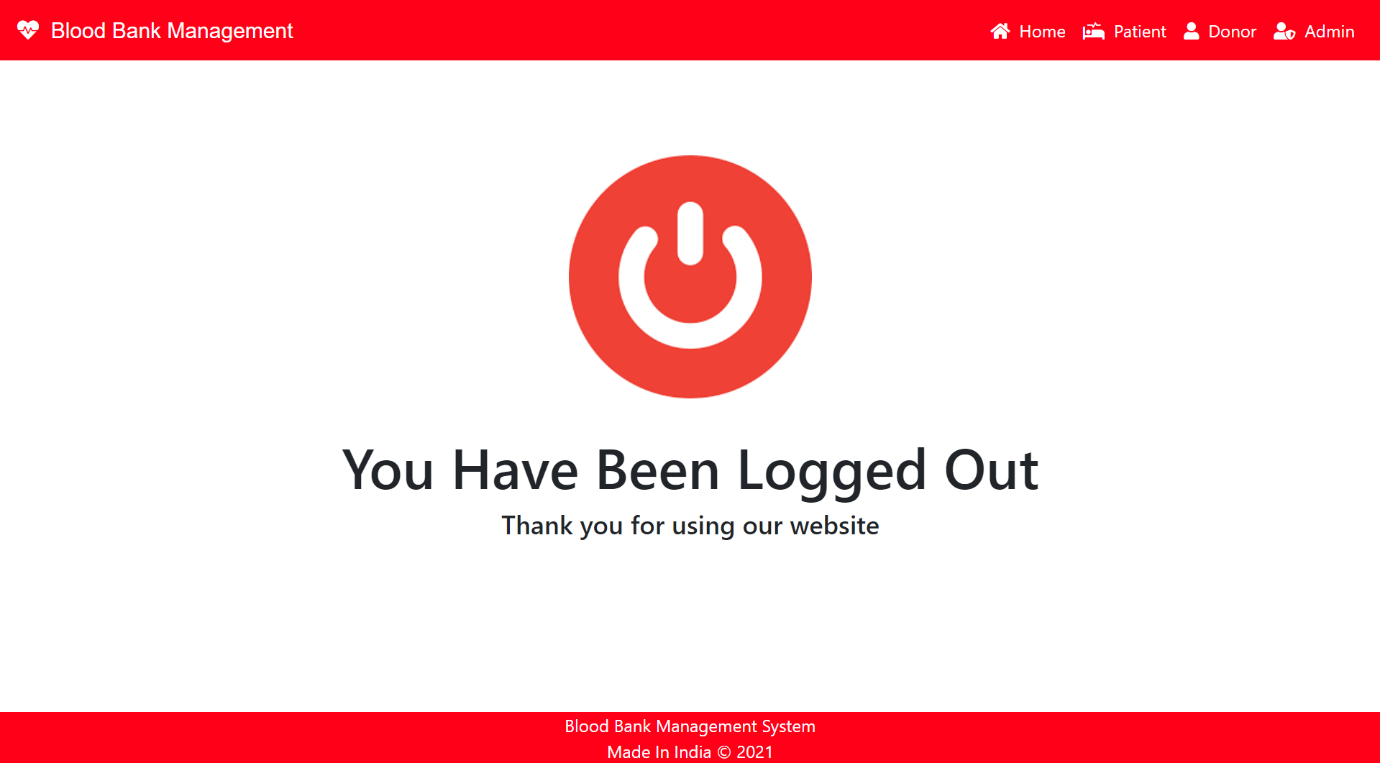
**Blood Donation**

[](https://github.com/sumitkumar1503/bloodbankmanagement/blob/master/static/screenshot/blooddonation.png?raw=true)

**Blood Request**

[](https://github.com/sumitkumar1503/bloodbankmanagement/blob/master/static/screenshot/bloodrequest.png?raw=true)

**Logout**

[](https://github.com/sumitkumar1503/bloodbankmanagement/blob/master/static/screenshot/logout.png?raw=true)

# 3.2 Hardware Interfaces

The minimum hardware requirements of Gephi are a 500 Megahertz CPU and 128 megabytes of RAM. Also, because Gephi uses an OpenGL 3D engine to speed up graph visualization, a compatible graphics card is required. A system with these specifications can handle a Network of approximately 1000 edges and nodes. For bigger networks, additional memory is required ([https://gephi.org/users/requirements/)](https://gephi.org/users/requirements/).

# 3.3 Software Interfaces

Gephi requires Java to be installed on the system, more specifically Java version 7 or 8 for its latest release. Additional information can be found on section 2.7 of this document.

Gephi can be connected with a MySQL, SQLite or PostgreSQL database to import a graph edge list.

# 3.4 Communications Interfaces

Gephi requires an internet connection to install new plugins, update already installed ones and update some of its components (APIs, modules etc.).

## 4. System Features

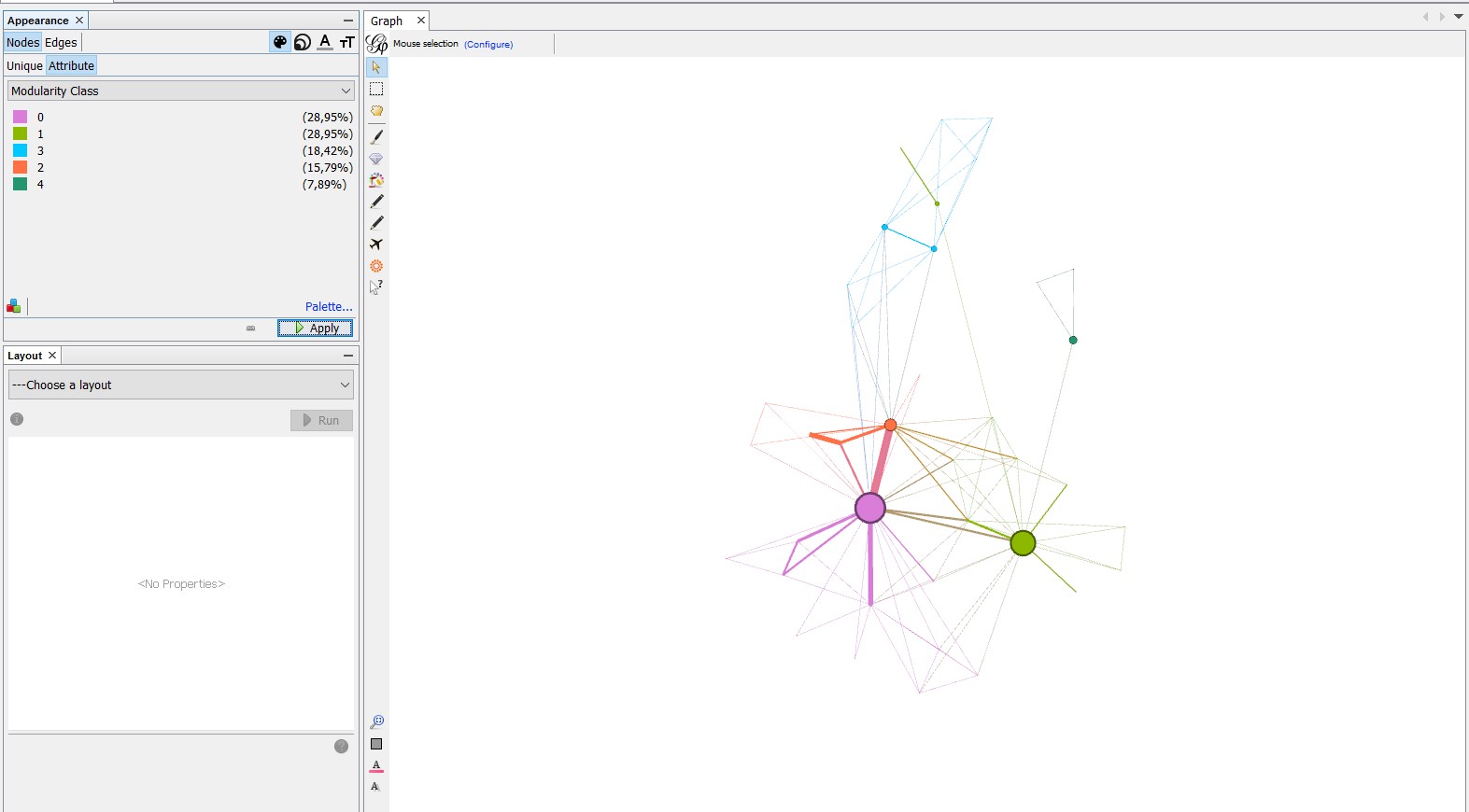
This section demonstrates Gephi’s most prominent features and explains how they can be used and the results they will give back to the user.

# 4.1 Graph Visualization

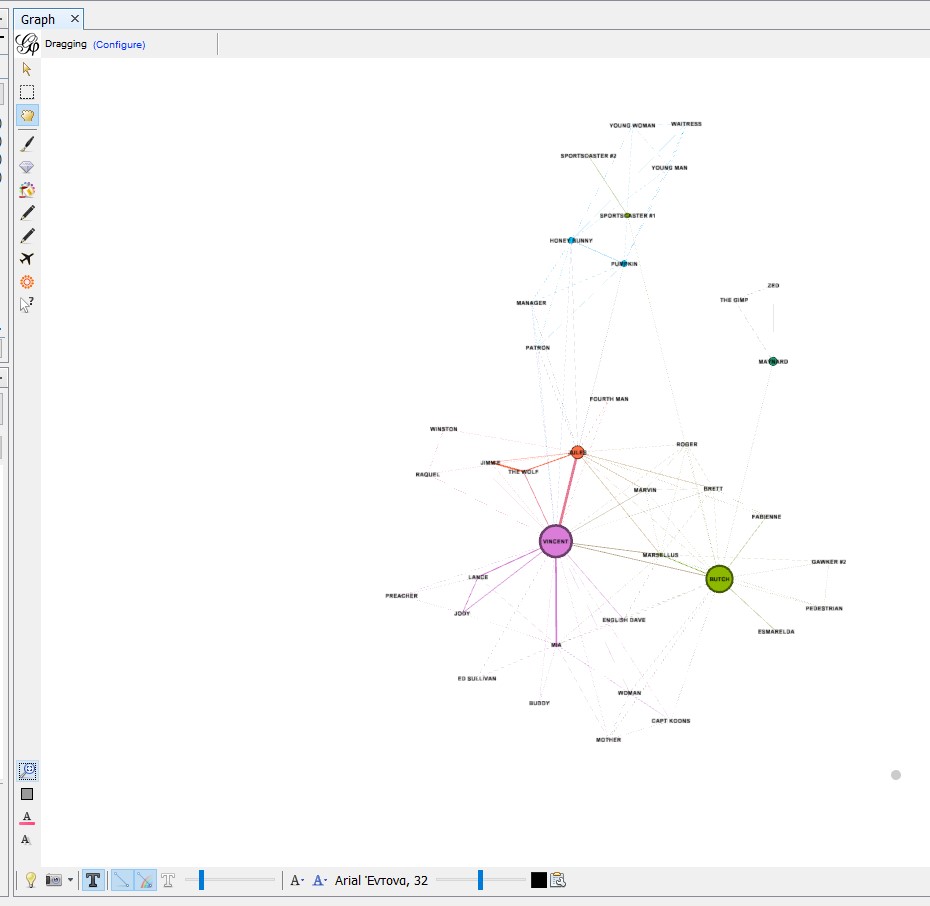
Users can directly interact with the graph by clicking on nodes to drag them, edit their attributes or color, add new nodes and edges, display shortest paths between nodes, make node labels visible/invisible etc.

Users can also use the appearance module to change the color of nodes. They can use the same color for the entire graph or use the values of a specific attribute as a reference point.

Example of changing the color of nodes based on one of their attributes (modularity class):

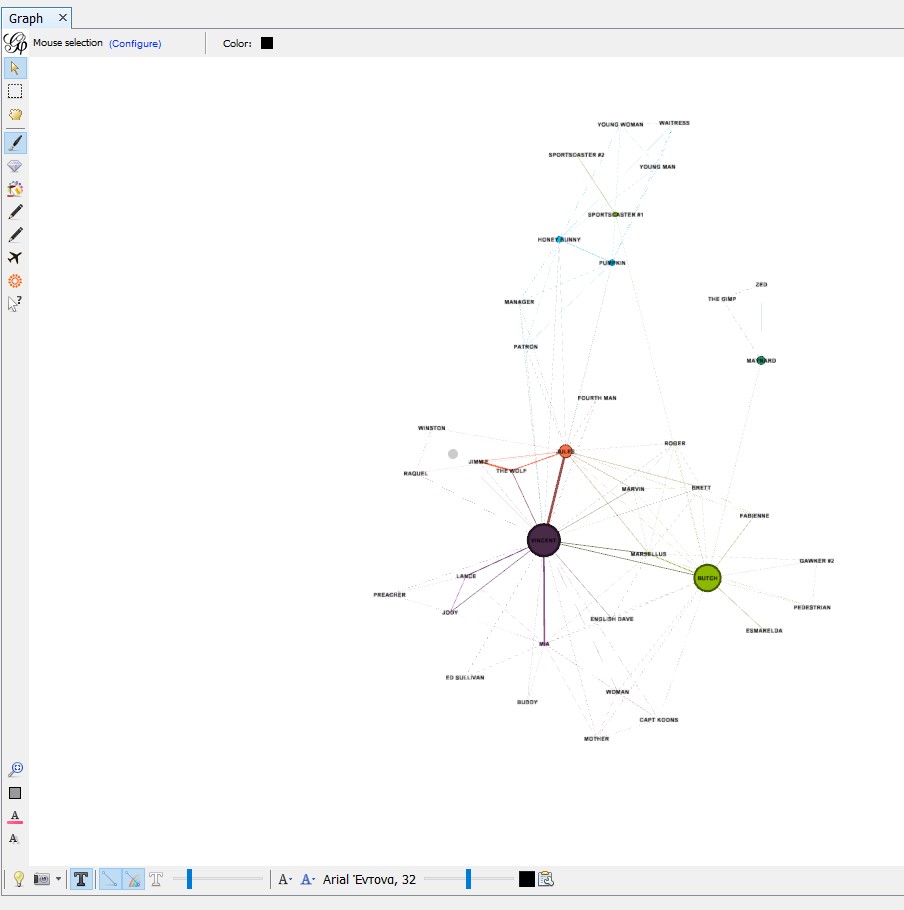
m

Example of making node labels visible (using the Show Node Labels – T button ):

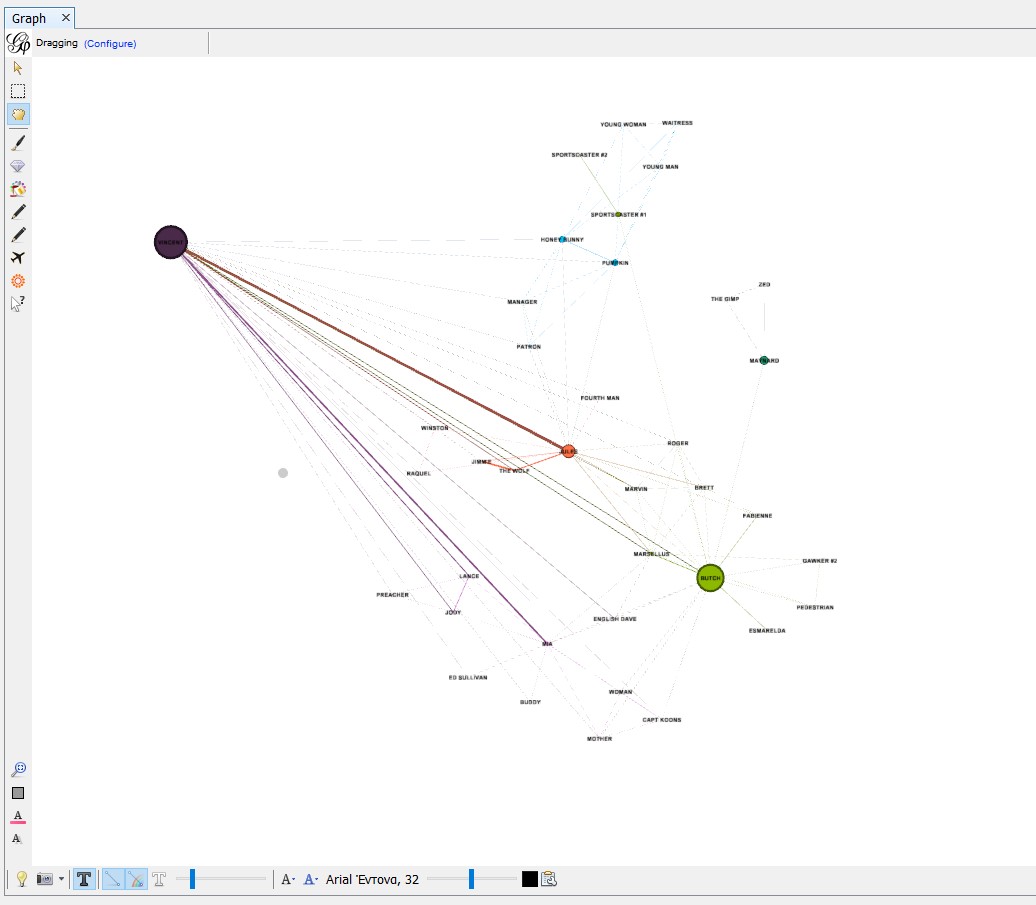


Example of chancing the color of only one node( Vincent) using mouse selection:

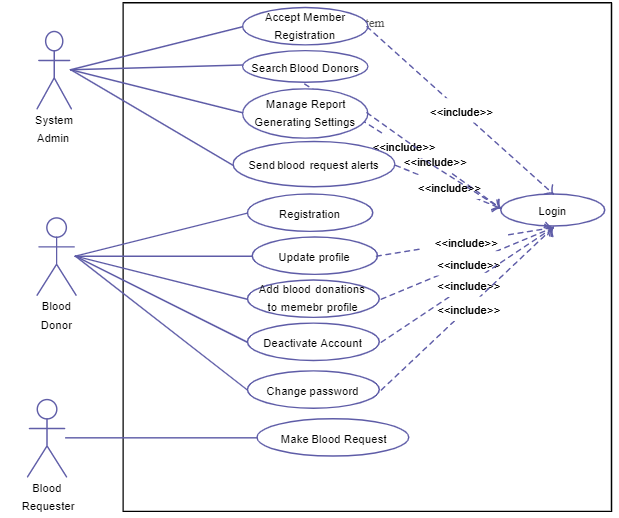
(The user has to press the “Painter” button first, on the left side of the screen, then choose color, on the top of the screen, and finally click on the desired node)



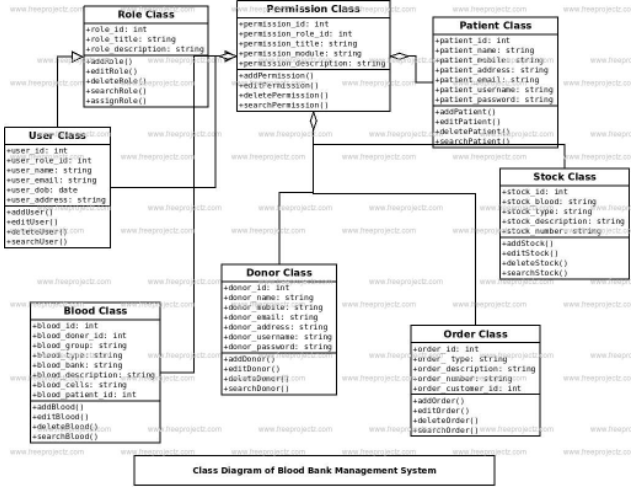
Example of Node Dragging (by pressing the ”Drag” button on the left side of the screen) :

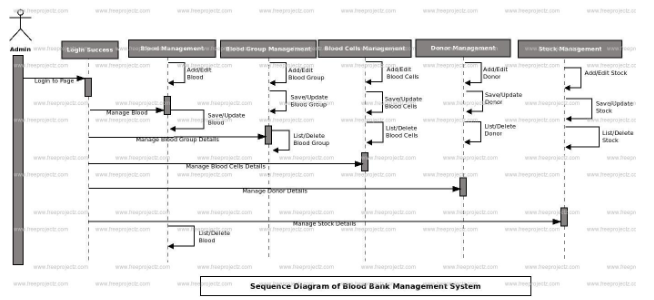


UML diagram:



Class digram:



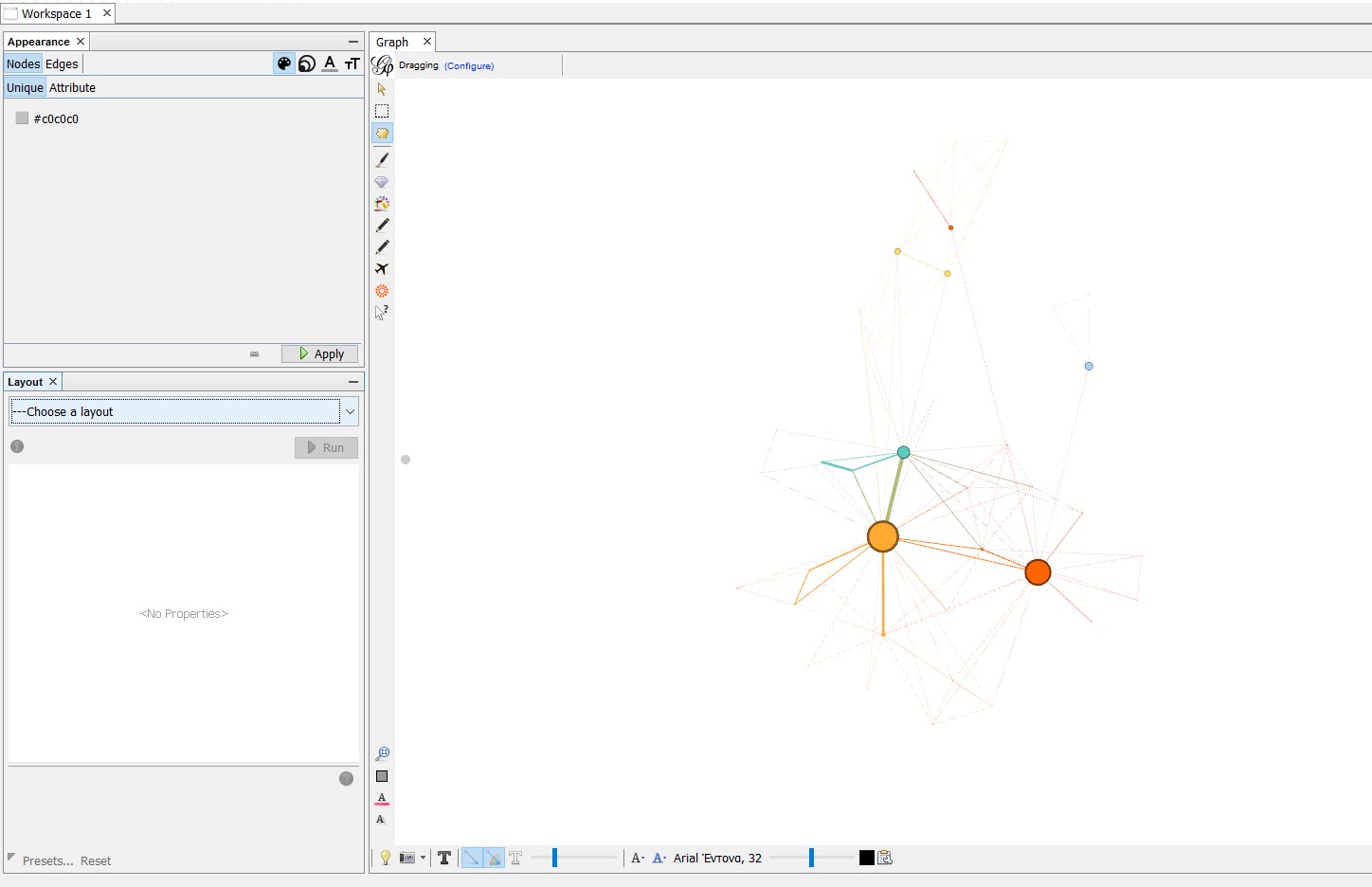


# 4.2 Graph Layout

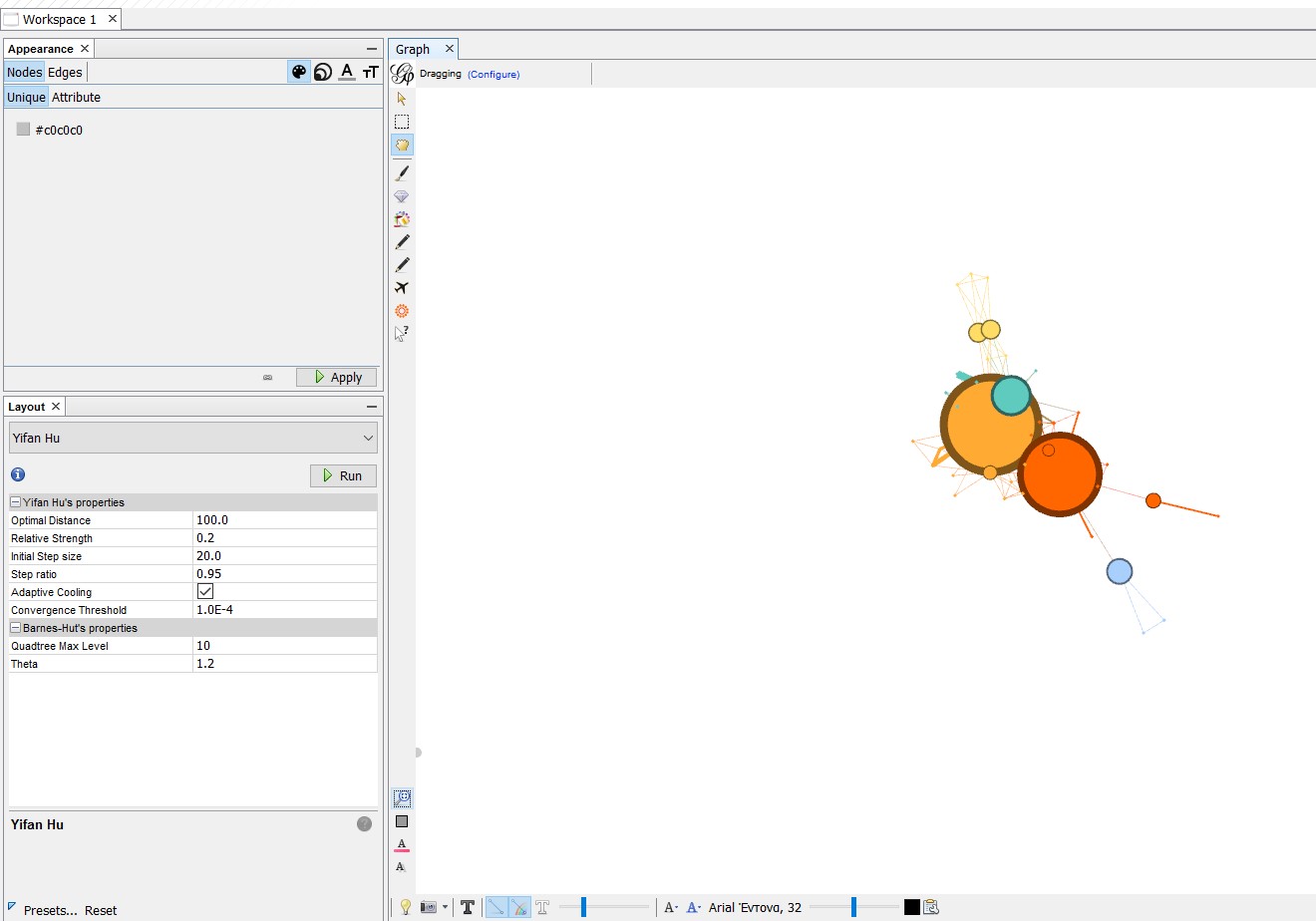
Gephi provides the following layout algorithms: Contraction, Expansion, Force Atlas, Force Atlas 2, Fruchterman Reingold, Label Adjust, OpenOrd, Random Layout, Rotate, Yifan Hu and Yifan Hu Proportional.

Example of applying a Layout algorithm to a graph:

The graph, before changing the Layout:



The graph, after applying the layout “Yifan Hu”:



# 4.3 Graph Metrics

Gephi provides the following Graph metrics:

Network Overview: Average Degree, Average Weighted Degree, Network Diameter,

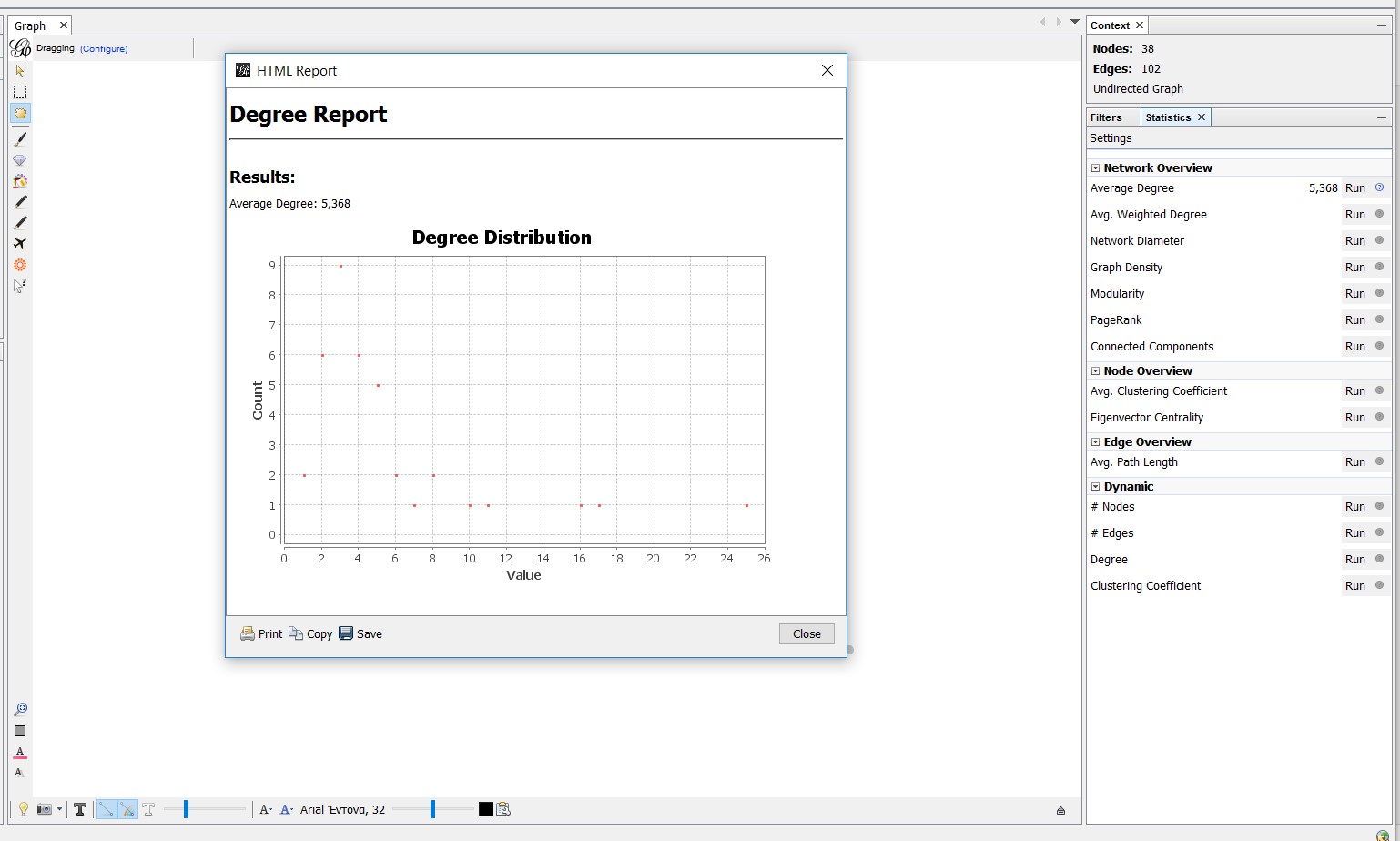
Graph Density, Modularity, PageRank, Connected Components

Node Overview: Average Clustering Coefficient, Eigenvector Centrality

Edge Overview: Average Path Length

Dynamic: #Nodes, #Edges, Degree, Clustering Coefficient

Example of calculating the “Average Degree” of the graph:



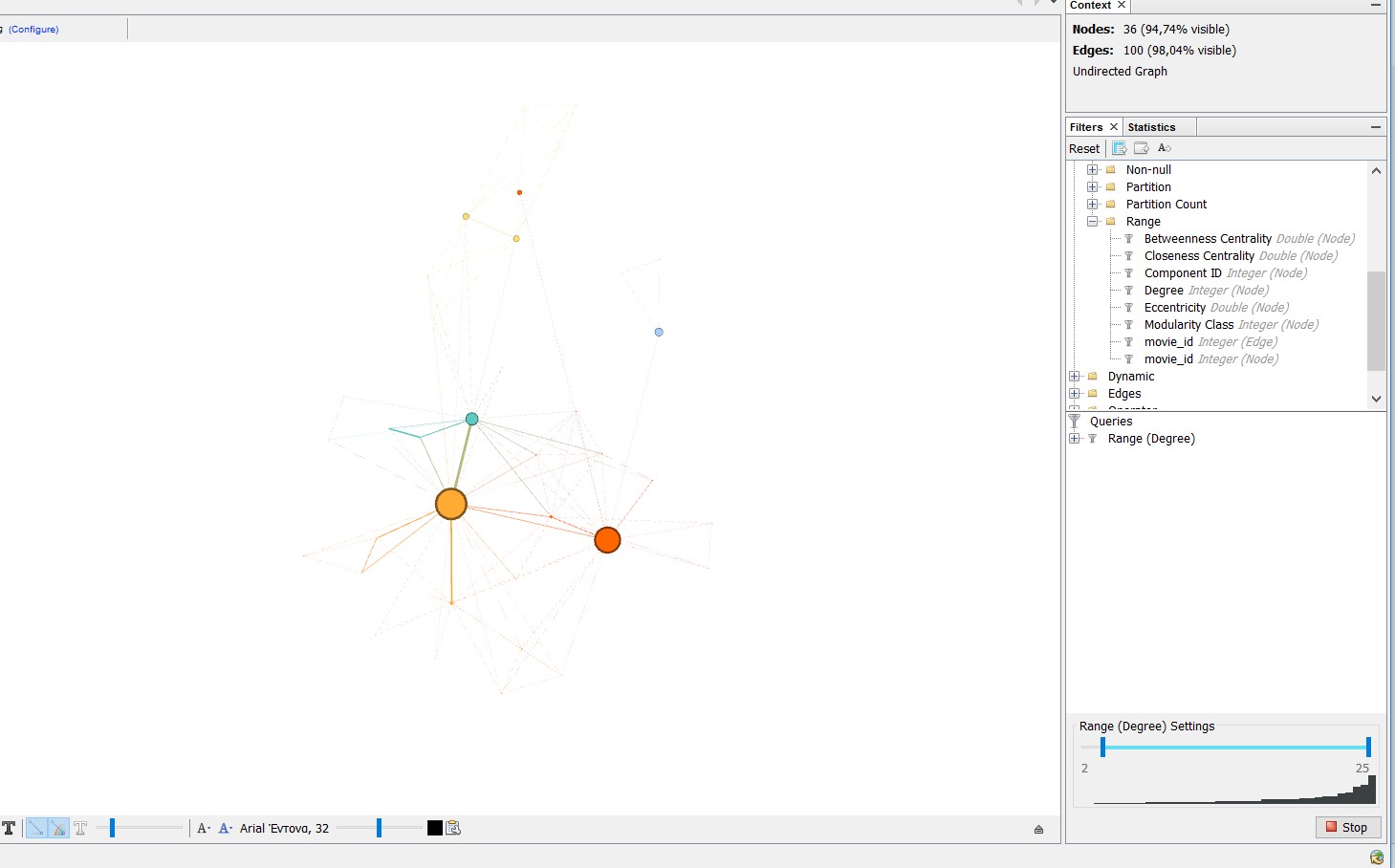
The graph which represents the distribution of the Average Degree can be printed,copied or saved.

# 4.4 Filters

Το apply a filter the user must select it and drag it to the “Queries” module.

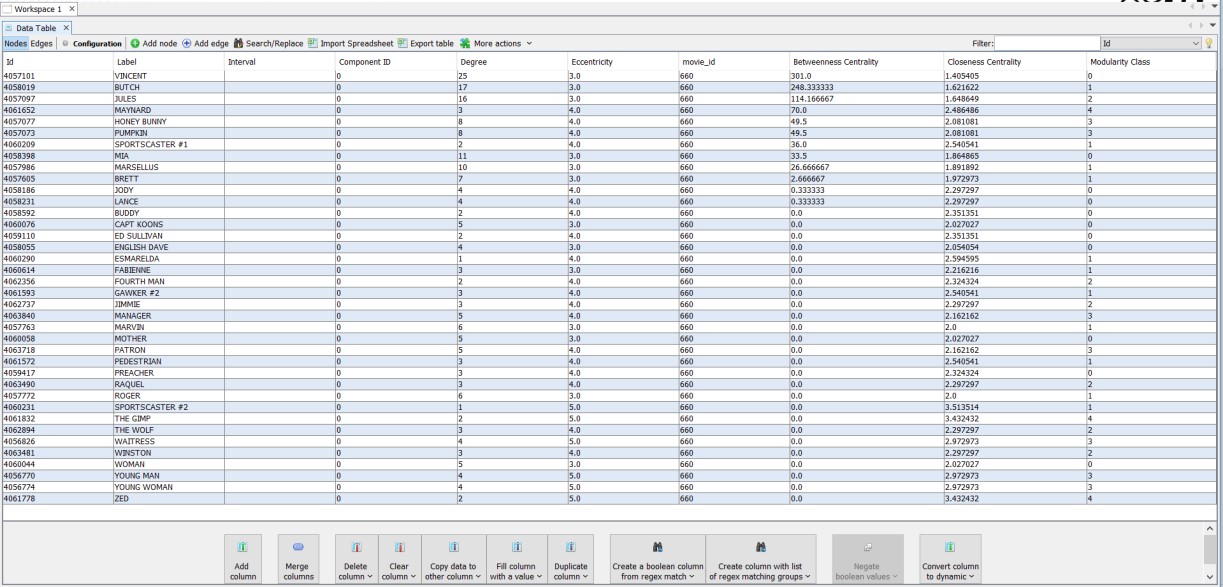
Example of applying a filter which removes every node that has a degree outside of the range [2,25], in other words it removes every leaf node.

On the bottom right side of the screen there is the Stop/Filter button which removes/applies the selected Filters. In the context module, we can see the number and percentage of the nodes and edges that did not get removed by the filter.



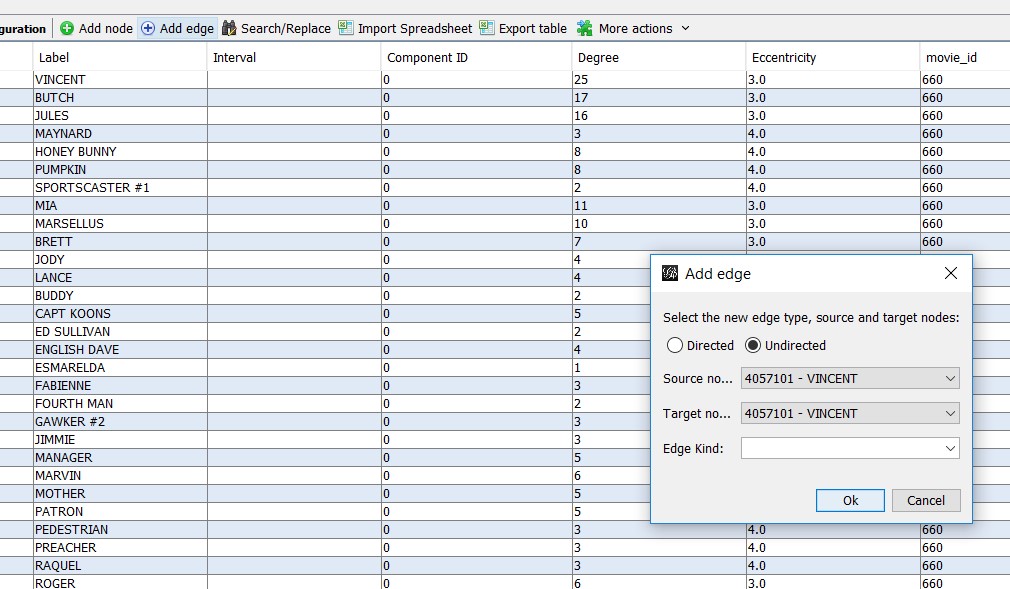
# 4.5 Data Table

Here the data of the graph is presented in two tables, one for nodes and one for edges and can be edited by the user. This is the Nodes table, the user can change to

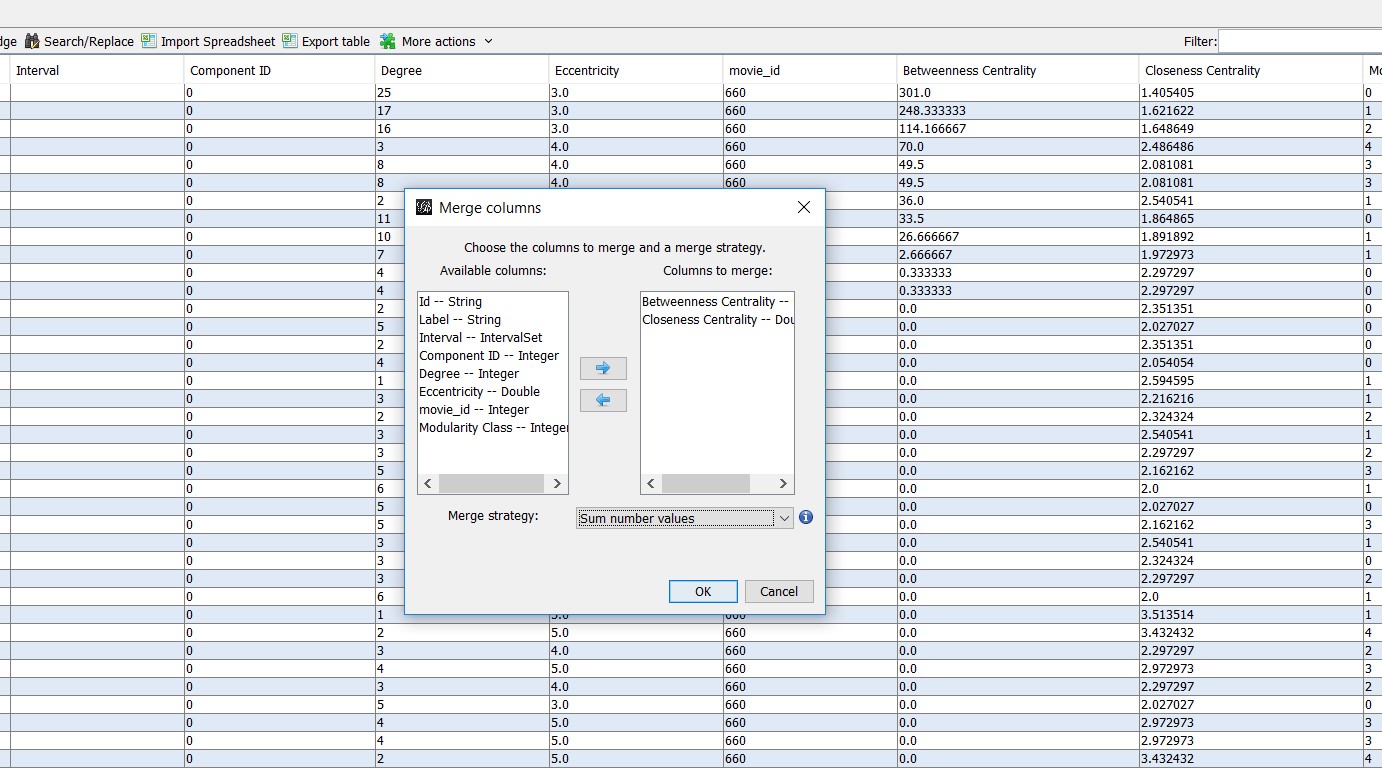


the edges table by pressing the “Edges” button on the top left hand side of the screen. The user can add new nodes/edges, edit existing ones, merge columns, create duplicates of columns etc.

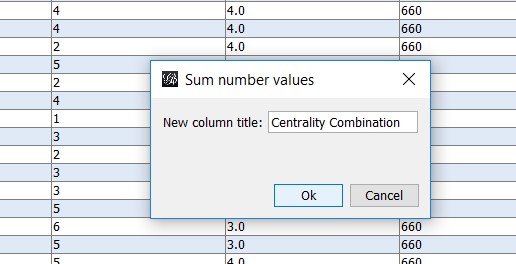
Example of adding a new Edge to the graph (by pressing the “Add edge” button):



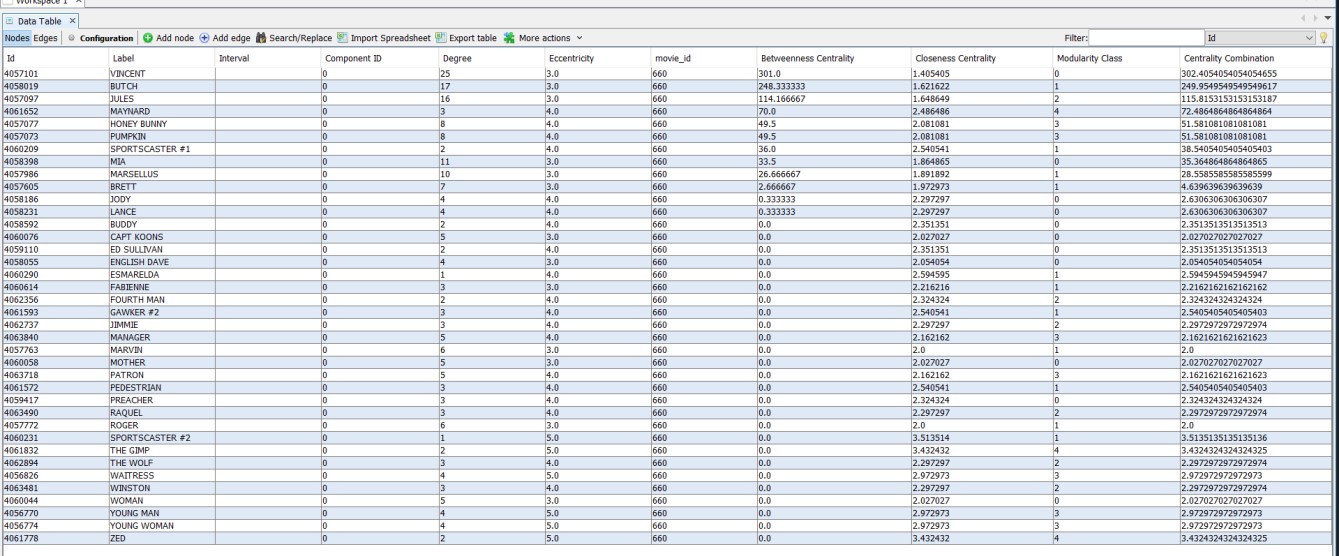
Example of merging two columns into one:



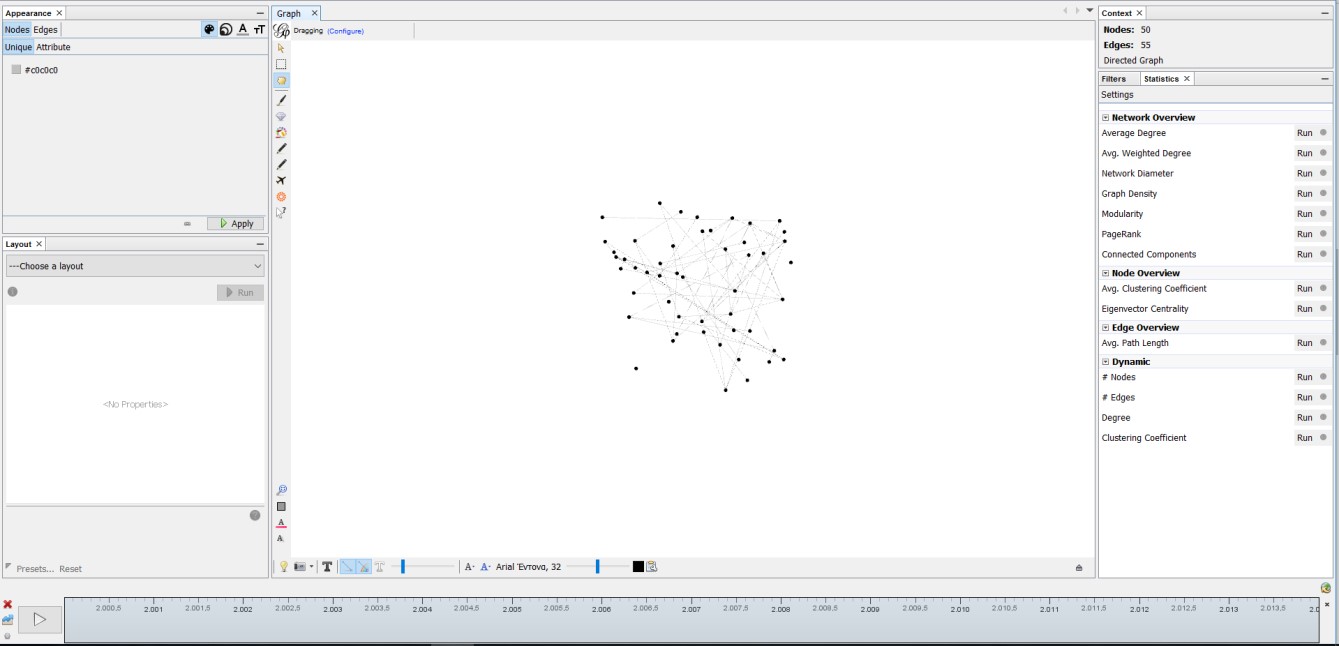
Choosing the new column’s title:



The new column (last one from the left) in the table:

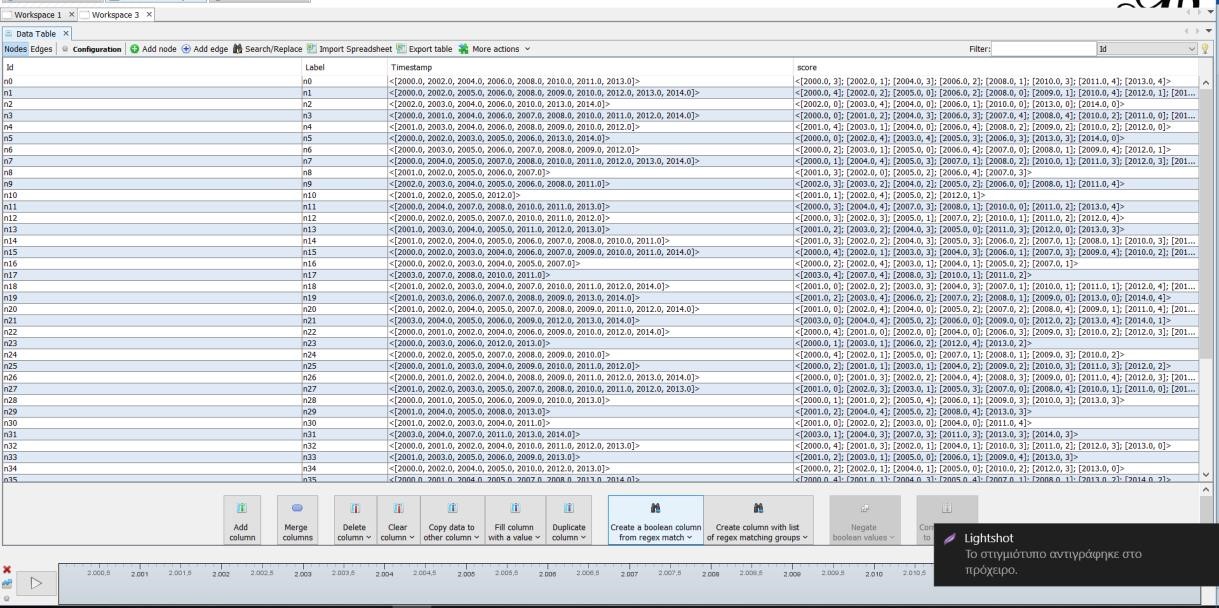


# 4.6 Dynamic Graphs



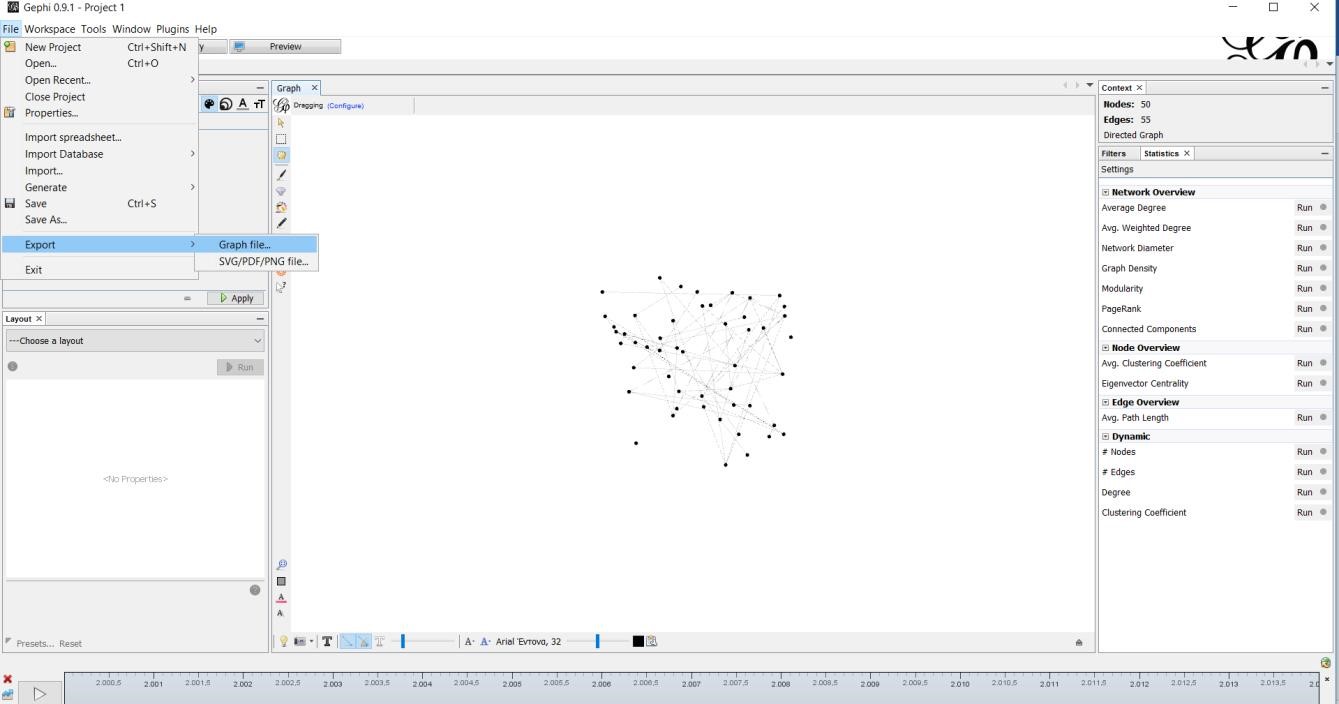
On the bottom side of the page we have the timeline bar, which can be used to demonstrate how the graph has changed at each timestamp. The user can select custom time intervals.

We can see in the data laboratory the appearance of the attribute Timestamp, which represents the time, and also the different values of the attribute score for every timestamp. Gephi also provides special metrics for dynamic graphs (for more information refer to section 4.3)

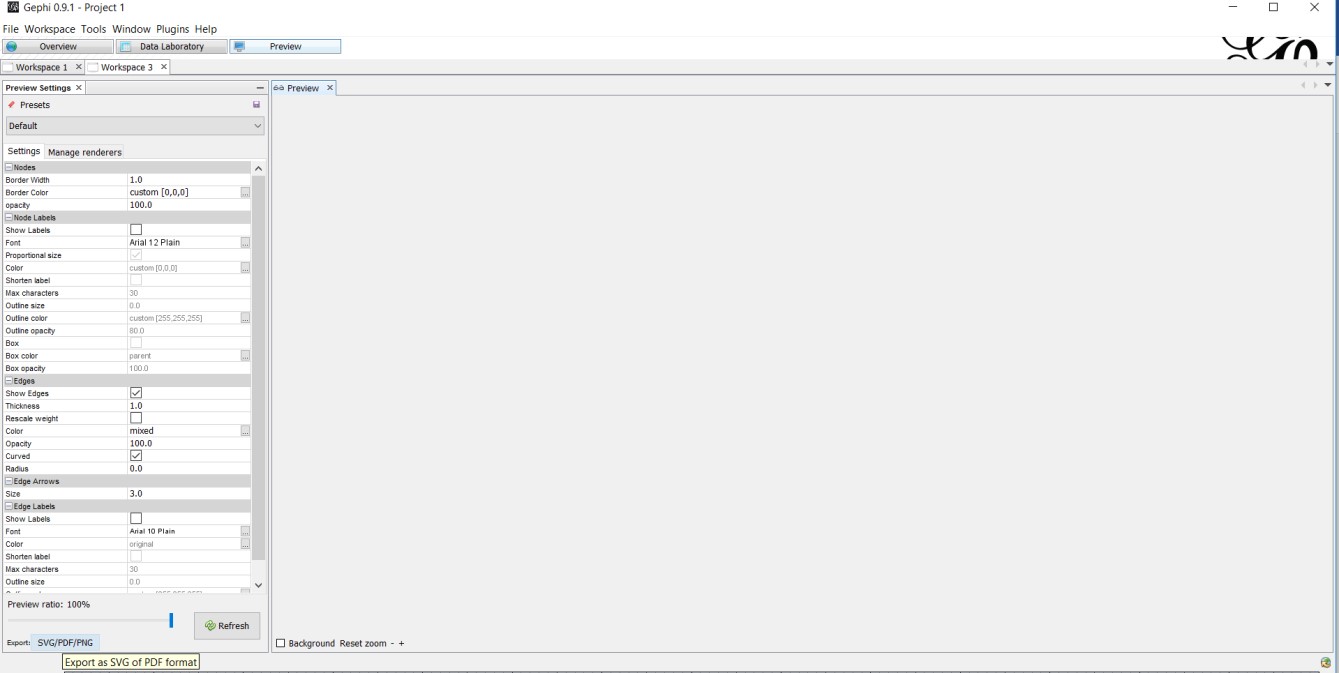


# 4.7 Graph Export

The user can export the graph, either via the main toolbar



Or via the preview page:



The available output file formats are: SVG PDF and PNG (just for the graph it self) and also every supported graph format (such as gexf , which also includes the graph data)

## 5. Other Non-functional Requirements

# 5.1 Performance Requirements

The system is interactive, and the delays involved are less.

For the Software to perform efficiently the Blood bank Management System must be working in latest operating system environments like windows 7, windows 8, windows 10 and on Linux.

# 5.2 Availability Requirements -

•The system should be always available, meaning the user can access it using application.

•In case of a of a hardware failure or database corruption, a replacement page will be shown. Also, in case of a hardware failure or database corruption, backups of the database should be retrieved from the application data folder and saved by the administrator.

•It means our system is 24 x 7 available.

# 5.3 Security Requirements

-The software must remain resilient in the face of attacks. The behaviour of the software must be correct and predictable. The software is available and behave reliably even under DOS attacks. The software must ensure the integrity of the customer account information. -Our system will automatically logout all customers after a period of inactivity.

# 5.4 Reliability Requirements

•As the system provide the right tools for problem solving itis made in such a way that the system is reliable in its operations and for securing the sensitive details.

•The system has the ability to work all the times without failures apart from network failure. A donor can have the faith on the system. The authorities will keeps the privacy of all donors in a proper manner. When the doctors found any disease in the testing stage after providing relevant details to the donor the system keeps the secretively of the donor

## Glossary

References: [**https://en.wikipedia.org/wiki/Main\_Page**](https://en.wikipedia.org/wiki/Main_Page)

* Graph: a system of nodes connected in pairs by edges. Often subdivided into [directed graphs](https://en.wikipedia.org/wiki/Directed_graph) or [undirected graphs](https://en.wikipedia.org/wiki/Undirected_graph) according to whether the edges have an orientation or not. [Mixed graphs](https://en.wikipedia.org/wiki/Mixed_graph) include both types of edges.
* Node: A node is (together with edges) one of the two basic units out of which graphs are constructed. Nodes of graphs are often considered to be atomic objects, with no internal structure.
* Edge: An edge is (together with vertices) one of the two basic units out of which graphs are constructed. Each edge has two (or in hypergraphs, more) vertices to which it is attached, called its endpoints. Edges may be directed or undirected; undirected edges are also called lines and directed edges are also called arcs or arrows. In an undirected [simple graph,](https://en.wikipedia.org/wiki/Simple_graph) an edge may be represented as the set of its vertices, and in a directed simple graph it may be represented as an ordered pair of its vertices. An edge that connects vertices *x* and *y* is sometimes written *xy*.
* Degree: the degree of a node of a [graph](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)) is the number of [edges](https://en.wikipedia.org/wiki/Edge_(graph_theory)) [incident](https://en.wikipedia.org/wiki/Incidence_(graph_theory)) to the node.
* Weight: A graph structure can be extended by assigning a weight to each edge of the graph. Graphs with weights, or [weighted graphs,](https://en.wikipedia.org/wiki/Weighted_graph) are used to represent structures in which pairwise connections have some numerical values. For example, if a graph represents a road network, the weights could represent the length of each road.
* Network(graph) Diameter: The diameter of a graph is the length of the shortest path between the most distanced nodes.
* Graph Density: graph density is the ratio of the number of edges and the number of possible edges.
* Modularity: Modularity is one measure of the structure of [networks](https://en.wikipedia.org/wiki/Complex_network) or [graphs.](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)) It was designed to measure the strength of division of a network into modules (also called groups, clusters or communities).
* PageRank is an [algorithm](https://en.wikipedia.org/wiki/Algorithm) used by [Google Search](https://en.wikipedia.org/wiki/Google_Search) to rank websites in their search engine results. Because the web is represented as a graph when PageRank is applied to it, it can be applied to every graph.
* Connected component: a connected component (or just component) of an [undirected graph](https://en.wikipedia.org/wiki/Undirected_graph) is a [subgraph](https://en.wikipedia.org/wiki/Subgraph_(graph_theory)) in which any two [vertices](https://en.wikipedia.org/wiki/Vertex_(graph_theory)) are [connected](https://en.wikipedia.org/wiki/Connected_graph) to each other by [paths,](https://en.wikipedia.org/wiki/Path_(graph_theory)) and which is connected to no additional vertices in the supergraph.
* Clustering coefficient: clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together.
* Centrality: In [graph theory](https://en.wikipedia.org/wiki/Graph_theory) and [network analysis,](https://en.wikipedia.org/wiki/Network_theory) indicators of centrality identify the most important [vertices](https://en.wikipedia.org/wiki/Vertex_(graph_theory)) within a graph.
* Closeness Centrality: In a [connected](https://en.wikipedia.org/wiki/Connected_component_(graph_theory)) [graph,](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)) the closeness centrality (or closeness) of a node is a measure of [centrality](https://en.wikipedia.org/wiki/Centrality) in a [network,](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)) calculated as the sum of the length of the [shortest paths](https://en.wikipedia.org/wiki/Shortest_path_problem) between the node and all other nodes in the graph. Thus, the more central a node is, the closer it is to all other nodes.
* Betweenness Centrality: In [graph theory,](https://en.wikipedia.org/wiki/Graph_theory) betweenness centrality is a measure of [centrality](https://en.wikipedia.org/wiki/Centrality) in a [graph](https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)) based on [shortest paths.](https://en.wikipedia.org/wiki/Shortest_path_problem) For every pair of vertices in a graph, there exists a shortest [path](https://en.wikipedia.org/wiki/Vertex_(graph_theory)) between the vertices such that either the number of edges that the path passes through (for [undirected graphs)](https://en.wikipedia.org/wiki/Undirected_graph) or the sum of the weights of the edges (for [directed graphs)](https://en.wikipedia.org/wiki/Directed_graph) is minimized. The betweenness centrality for each [vertex](https://en.wikipedia.org/wiki/Vertex_(graph_theory)) is the number of these shortest paths that pass through the vertex.
* Dynamic Graph: An update on a graph is an operation that inserts or deletes edges or vertices of the graph or changes attributes associated with edges or vertices, such as cost or color. A dynamic graph is a graph that is subject to a sequence of updates.