**Mini Project Report on**



INVENTORY MANAGEMENT SYSTEM [IMS]



**Submitted in partial fulfilment of the requirement for the award of the degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted by:**

**Divya 2019477**

**Vinayak**

**Arnav Dobriyal**

***Under the Mentorship of***

**Mr. Jyotir Moy Chatterjee**

**Assistant Professor**

A logo of a graphic era

Description automatically generated

**Department of Computer Science and Engineering**

**Graphic Era (Deemed to be University)**

**Dehradun, Uttarakhand**

**June-2024**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Description** | **Page No.** |
| Chapter 1 | Introduction. | **3-4** |
| Chapter 2 | SRS. | **5-23** |
| Chapter 3 | Result. | **24-27** |
| Chapter 4 | Conclusion. | **28** |
| Chapter 5 | References. | **28** |

**1. INTRODUCTION**

Inventory management is one of the crucial supply chain components in retail store. Every day or by weekly, the store needs to update the stock that is coming in or out. Most of the retail store must at least have a warehouse to store their products. To remain customer satisfaction whenever the shop does not have the product, retail company provide delivery service through customer order for product that comes from warehouse inventory. Having inventory management system within an organization is important because the business can monitor and control their product stock and business revenue that is going on within the organization. At the same time, it is also to determine the suitable product quantity to restock according to customer and market demand which will reduce business loss of overstocking (Plinere & Borisov, 2016). Without having good inventory management system within organization, it can cause many businesses risk especially for retail store such as out of stock and product that is not sold due to market demand which will bring a dissatisfaction to customer and business lost (Patil & Divekar, 2014). Moreover, inventory management in an organization are done manually such as updating and checking inventory stocks in an excel and or logbook in some company nowadays and in the old days.

* 1. **Motivation of Work**

The development of an inventory management system is motivated by the need for businesses to optimize their inventory-related processes. Manual inventory management can be time-consuming and prone to errors, leading to inefficiencies and increased costs. By implementing an automated system, businesses can streamline inventory tracking, reduce human error, and improve overall efficiency. This automation allows for better control over inventory levels, ensuring that products are available when needed while minimizing excess inventory and stockouts.

Additionally, an inventory management system provides businesses with valuable insights through data analysis. By tracking inventory movements and sales trends, businesses can identify patterns and make informed decisions regarding purchasing, stocking, and pricing strategies. This data-driven approach enables businesses to optimize their inventory management practices, leading to cost savings and improved profitability. Furthermore, the system can help businesses forecast demand more accurately, ensuring that they can meet customer needs efficiently.

**1.2 Objective of the project**

This study aims to develop an Inventory Management Systems (IMS) that can provide better control and handling of product stock, customer order, customer service and order delivery that relates to company inventory information. The target user is the owner and employee in Small and Medium Enterprise (SME) retail store that stills manages inventory manually. IMS helps retail store to track down the next arrival of product stocks and record customer order for reservation for the product in the store inventory. In this study, the developer used python for backend system development and HTML, CSS, JavaScript for frontend system development. This study also applies software methodology that emphasize on iterative development process. Even though the inventory management system has been fully developed by the us, there are still limitations found and future enhancement that can be made towards the system.

The objectives are:

• To reduce manual tracking and recording of incoming and outgoing stock from company inventory.

• To provide summary of sales and product report for manager to view business situation.

• To provide a better customer service to handle feedback based on their order quality and problem.

• To produce effective computerized system to be used by all parties who involved directly in inventory management.

* 1. **Summary**

An Inventory Management System is a critical tool for businesses to manage their inventory effectively, optimize inventory levels, and improve operational efficiency. By implementing an IMS, businesses can enhance customer satisfaction, reduce costs, and gain a competitive advantage in the market.

**2.Software Requirement Specification for Inventory Management System**

# 2.1 Introduction

## **2.1.1 Purpose**

The purpose of this document is to provide a comprehensive overview of the inventory management software, outlining its functionalities for tracking inventory, replenishing stock, optimizing warehouse layouts, detailing product quality, monitoring expiration dates, and managing authorization levels for both administrators and customers.

## **Document Conventions**

This document adheres to the IEEE template for System Requirement Specification Documents.

## **Intended Audience and Reading Suggestions**

* Retailers and Business Owners: Those involved in using inventory and warehouse operations as a customer.
* Warehouse Managers and Staff: Individuals responsible for day-to-day inventory management tasks.
* Software Developers and Engineers: Those interested in further developing or customizing the inventory management system.

## **2.1.4 Product Scope**

The inventory management software is designed to streamline inventory tracking, stock replenishment, warehouse layout optimization, product quality management, expiration date monitoring, and user authorization. Users can efficiently organize and manage their inventory data, analyze trends, and ensure product quality and compliance.

## **References**

* <https://gephi.org/>
* <https://github.com/gephi>
* <https://goo.gl/nsUFwy>
* <http://www.gnu.org/licenses/gpl.html>
* <https://opensource.org/licenses/CDDL-1.0>
  1. **Overall Description**

## **2.2.1 Product Perspective**

Gephi was developed for everyone who is interested in graphs and wants either to just experiment with them so that he can understand them or wants to use them as a means of analyzing data.  It can handle various graph formats (GEXF, GDF, Pajek, DOT, GraphML, UCINET, CSV…) and supports directed, undirected and mixed graphs.It is an open source project and it has a very active developer team to support it and provide feedback to users. It was developed to run on Windows, Mac OS X and Linux.

## **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.

## **User Classes and Characteristics**

* Typical Users, such as students, who want to use Gephi for analyzing networks (Social networks, Social Media networks, Semantic networks etc.)
* Advanced/Professional Users, such as engineers or researchers, who want to use Gephi for more demanding graph analysis.
* Programmers who are interested in working on the project by further developing it or fix existing bugs

## **Operating Environment**

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

## **Design and Implementation Constraints**

Gephi is developed in Java, it uses OpenGL for its visualization engine and has been

built on top of the NetBeans Platform. It uses a modular design where every feature is wrapped into a separate module and the modules depend on each other through well written APIs. There are several APIs available to make plugin development easy.

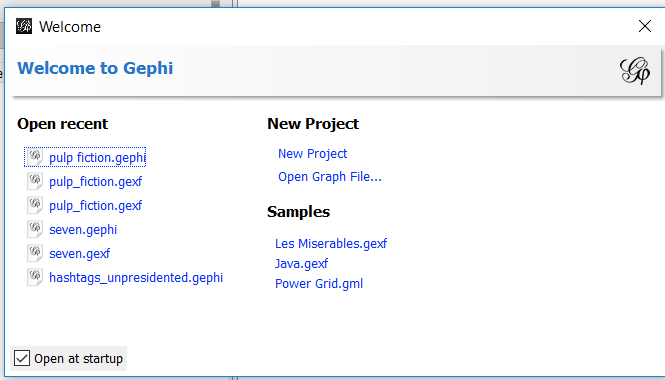
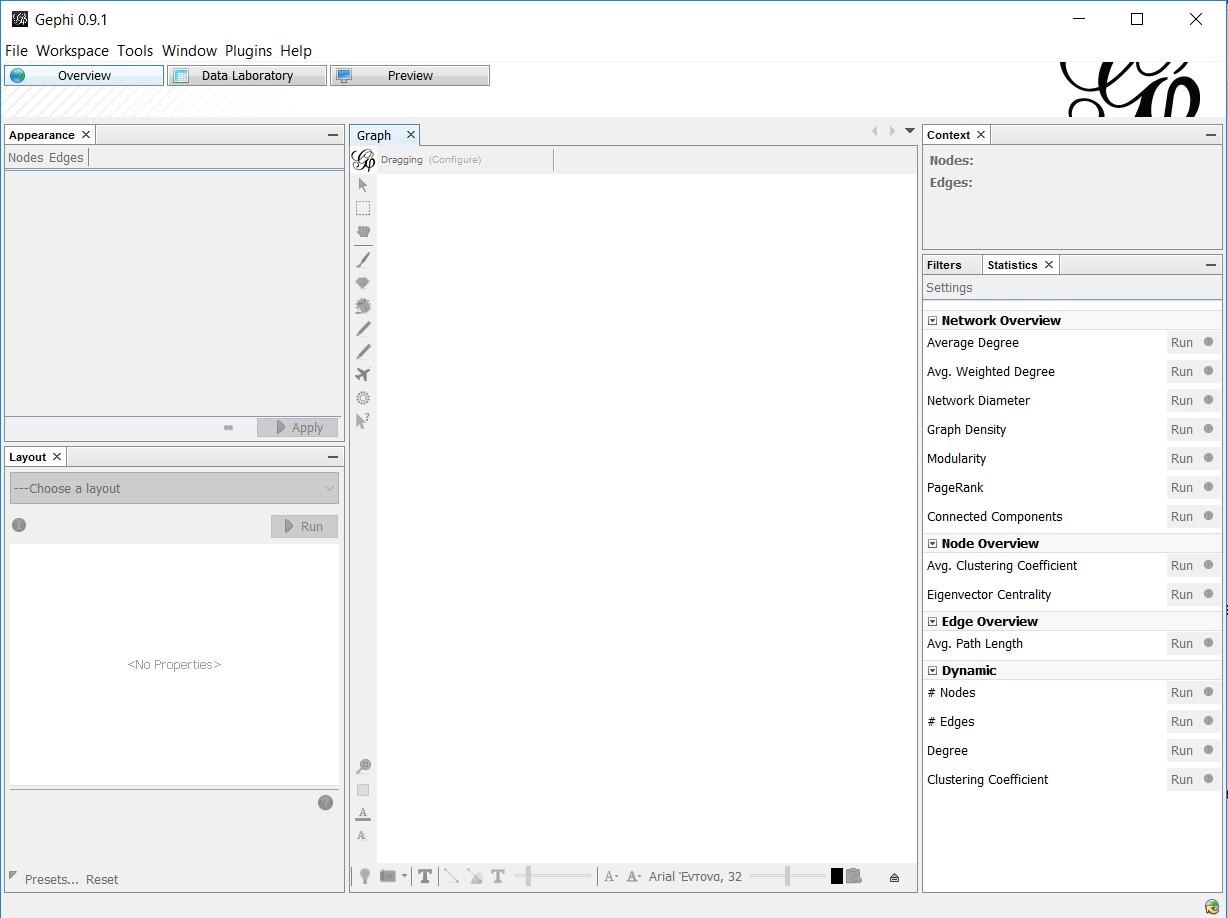
## **Assumptions and Dependencies**

Gephi is developed in Java and therefore requires Java to be installed on the user’s system. The latest stable version of Gephi requires Java version 7 or higher. This applies to Windows and Linux users. On Mac OS X, Java is bundles with the application.

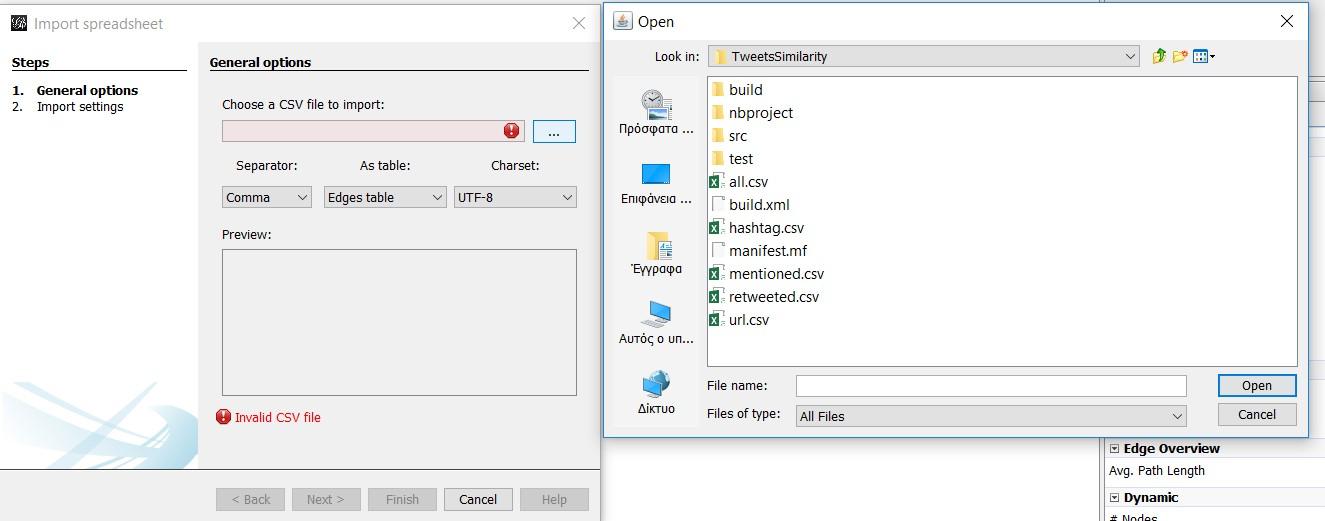
# External Interface Requirements

**2..2.7.1 User Interfaces**

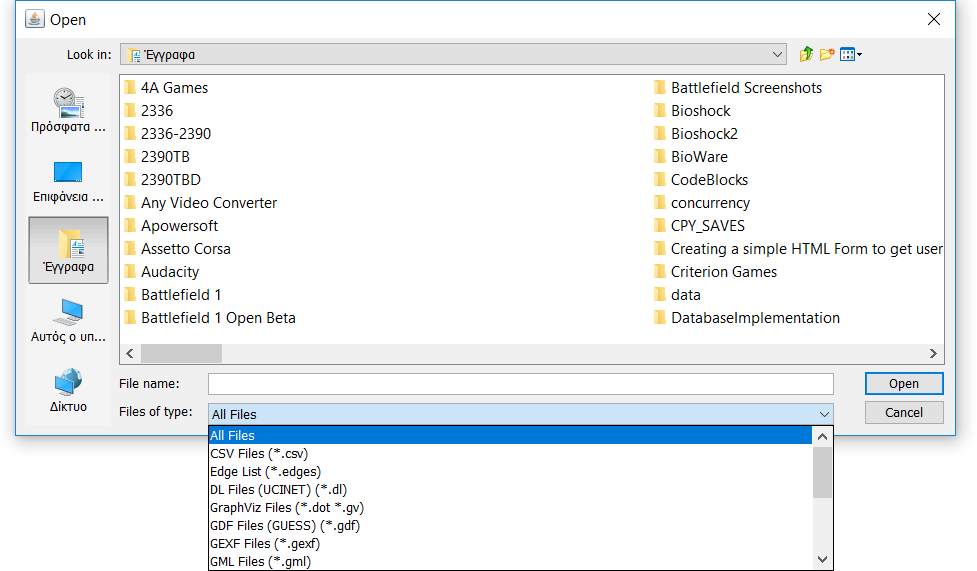
* Gephi’s Welcome Screen:

  
 **[Fig 1]**  
  
 **[ Fig 2]**

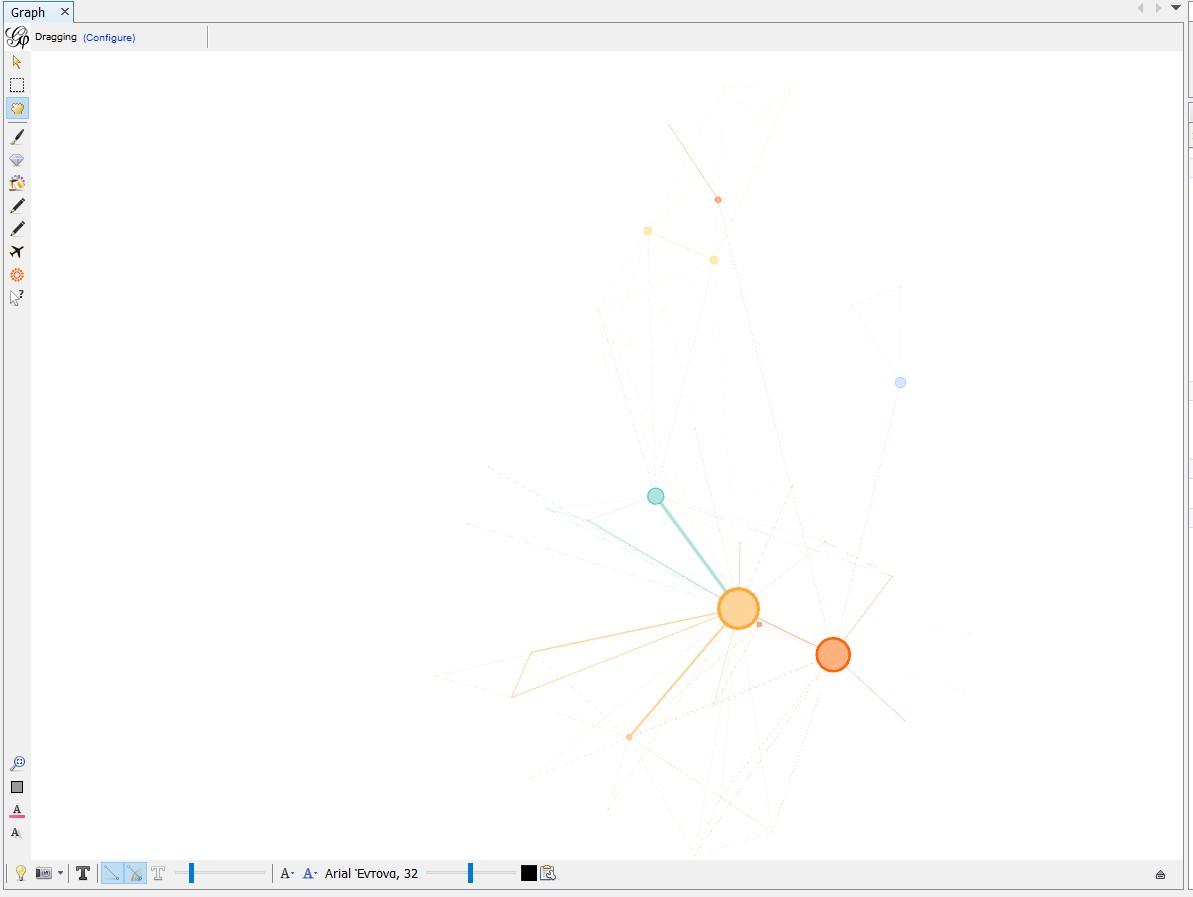
* Import Graph (CSV file)

  
**[Fig 3]**

* Open Graph (all supported file formats)

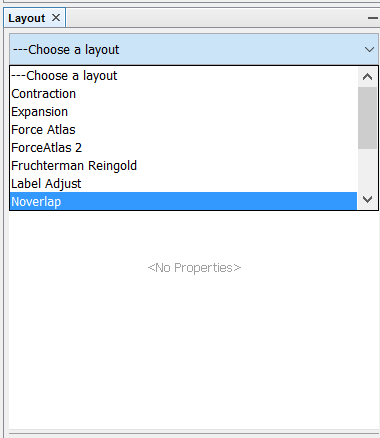
  
 **[Fig 4]**

* Overview of the Graph

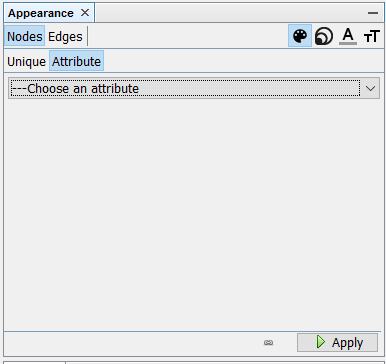


**[Fig 5]**

* Change Graph Layout

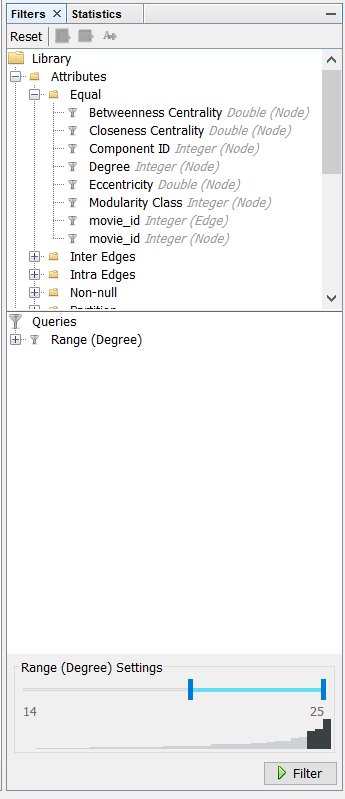
  
**[Fig 6]**

* Change Node/Edge Colour (Unique applies the exact same colour to all nodes/edges. Attribute changes the colour of each node/edge according to its value of the selected attribute)



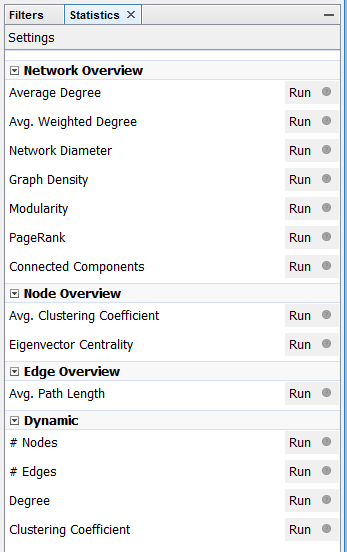
**[Fig 7]**

* Apply Filters Create filters that can hide nodes and edges ( that do not comply with the filter’s settings) To apply a filter drag it from the “Filters” module to the “Queries” module and press the “Filter” button on the bottom right side. Filter settings can be modified.

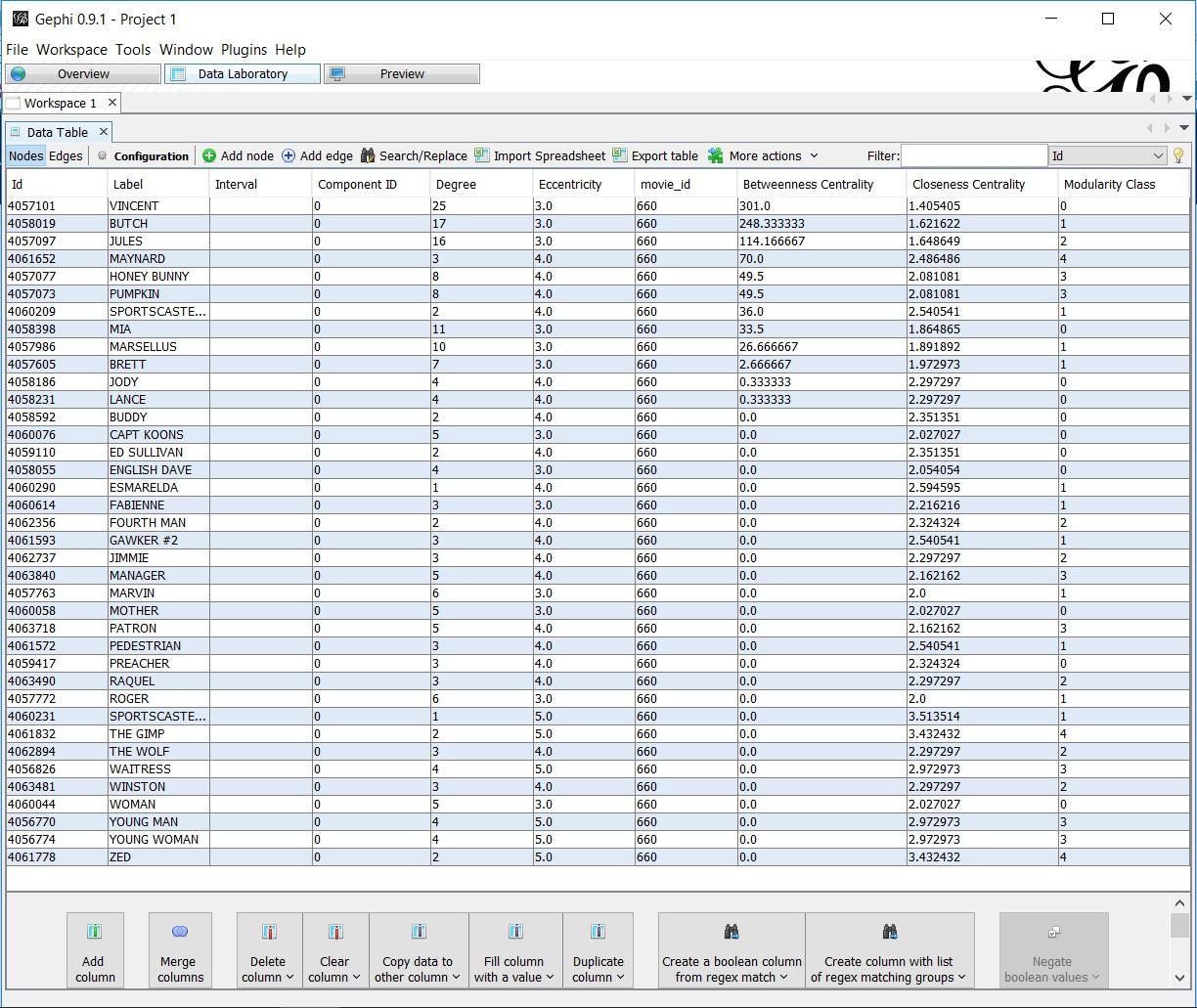


**[Fig 8]**

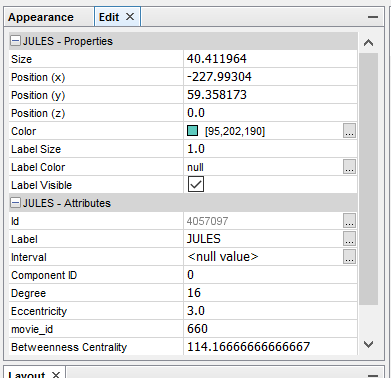
* Calculate Graph Metrics

  
**[Fig 9]**

* Data Laboratory

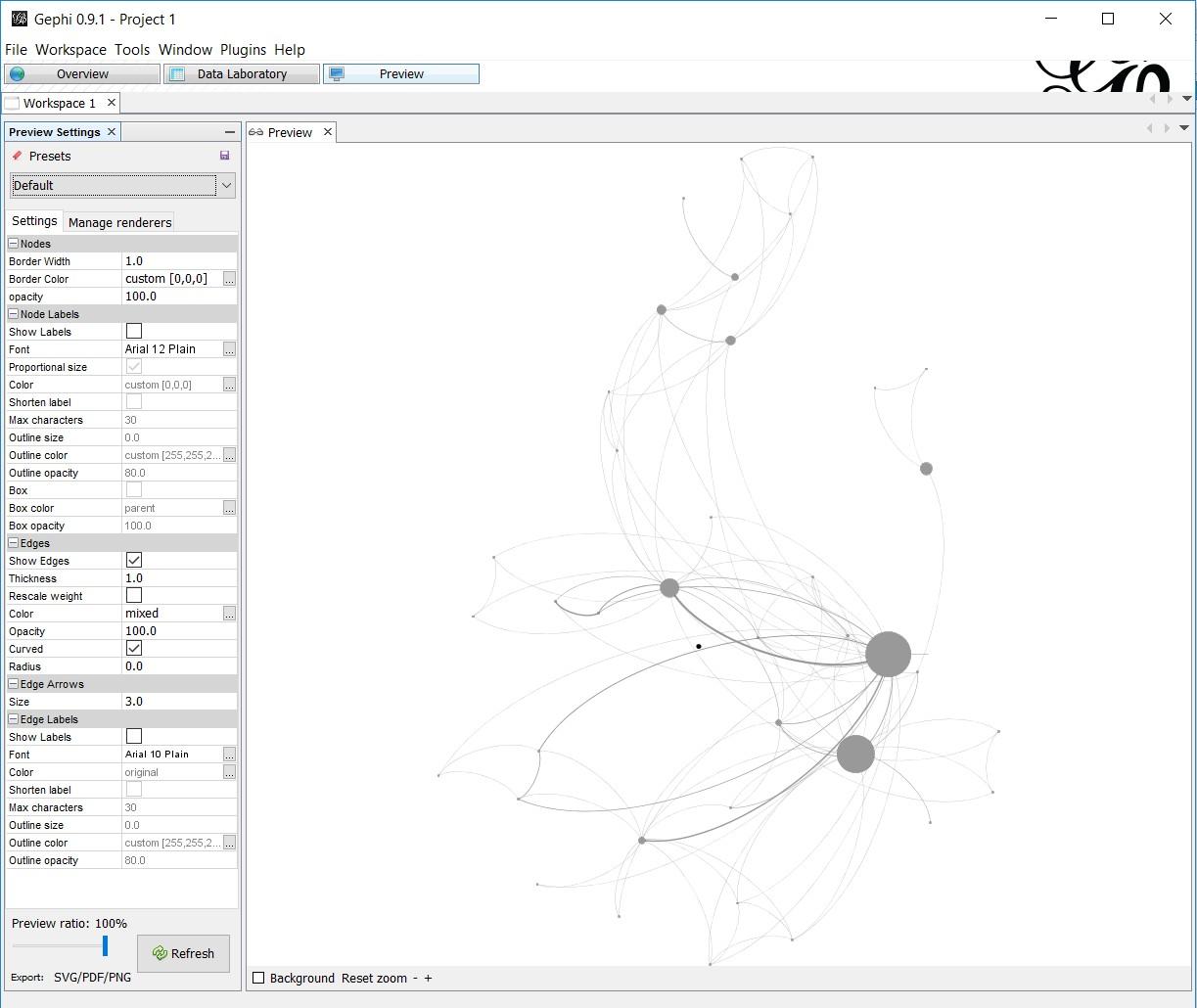
  
**[Fig 10]**

* Edit Node (either use the “Edit” button and click on the Node in the Overview screen or left click on the Node in the Data Laboratory screen)

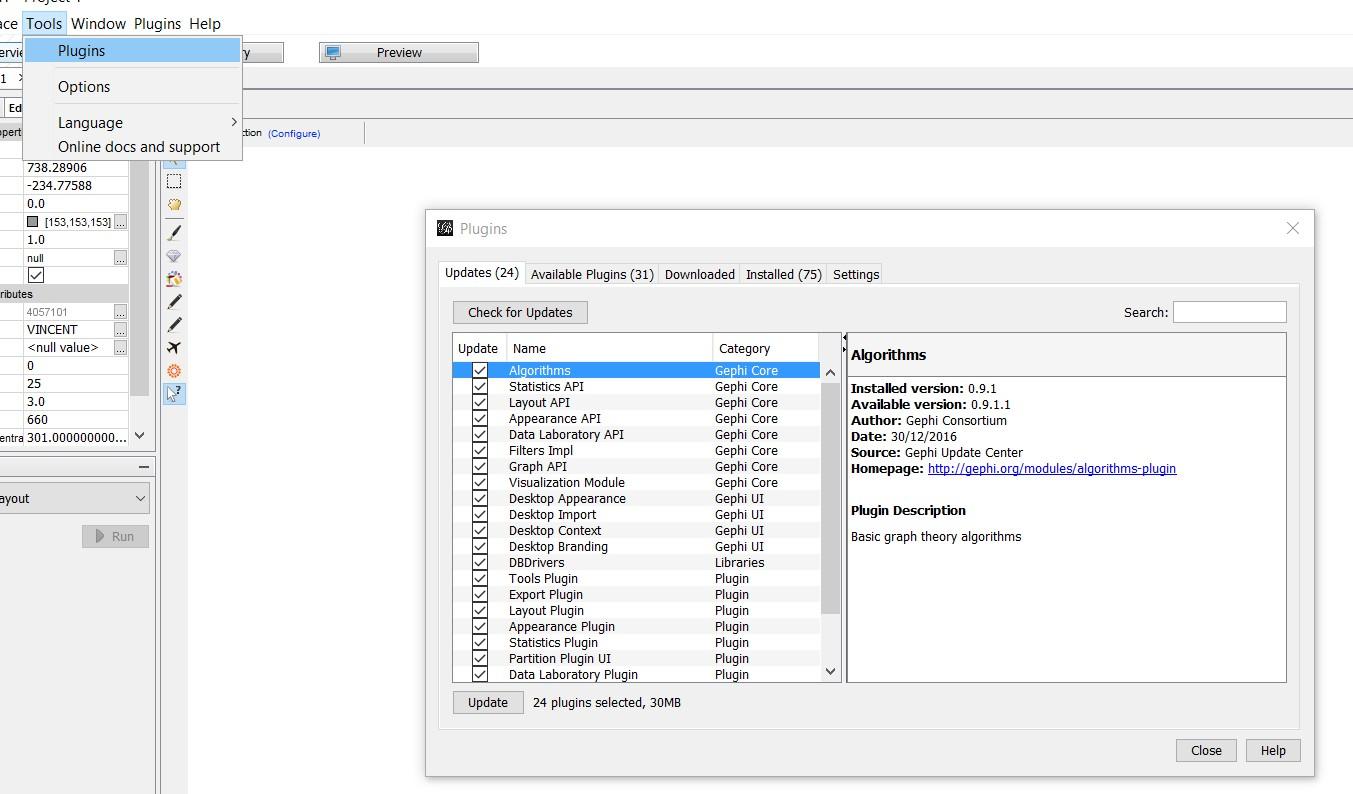


**[Fig 11]**

* Preview Graph

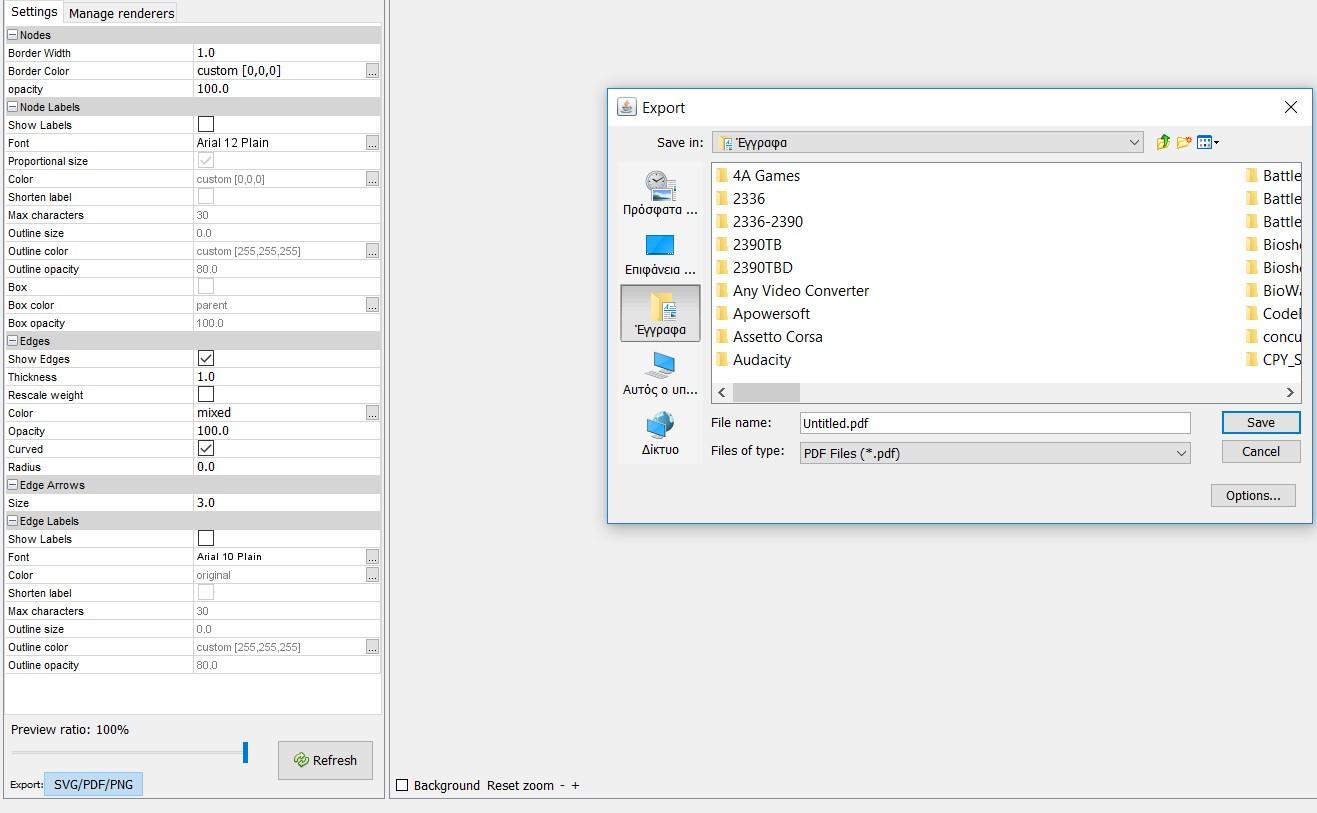
  
**[Fig 12]**

* Manage Plugins



**[Fig 13]**

* Export Graph

  
  
**[Fig 14]**  
  
**2.3 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/>).

**2.4 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.

## **Communications Interfaces**

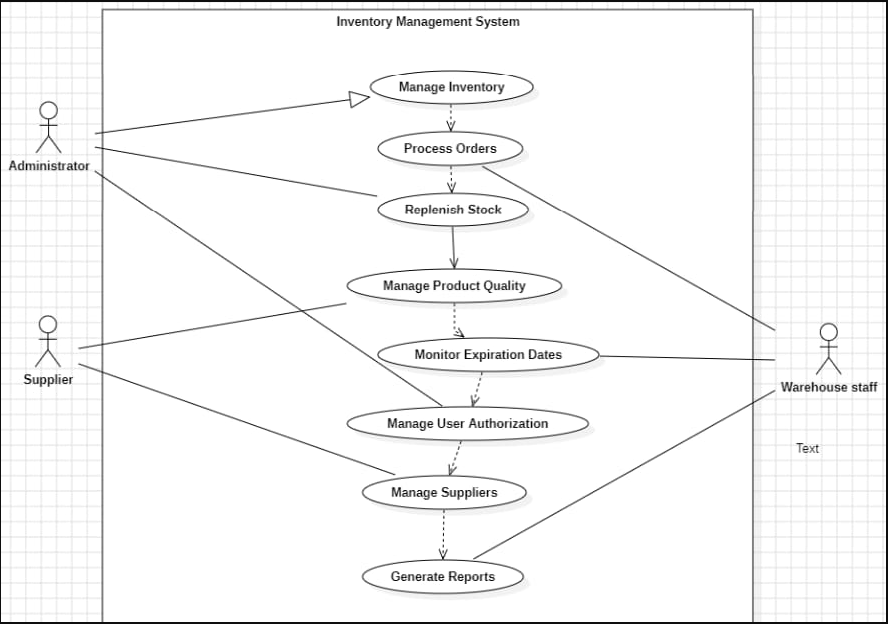
Gephi requires an internet connection to install new plugins, update already installed ones and update some of its components (APIs, modules etc.).

# 2.6 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.

#### **2.6.1 Use Case Diagram**

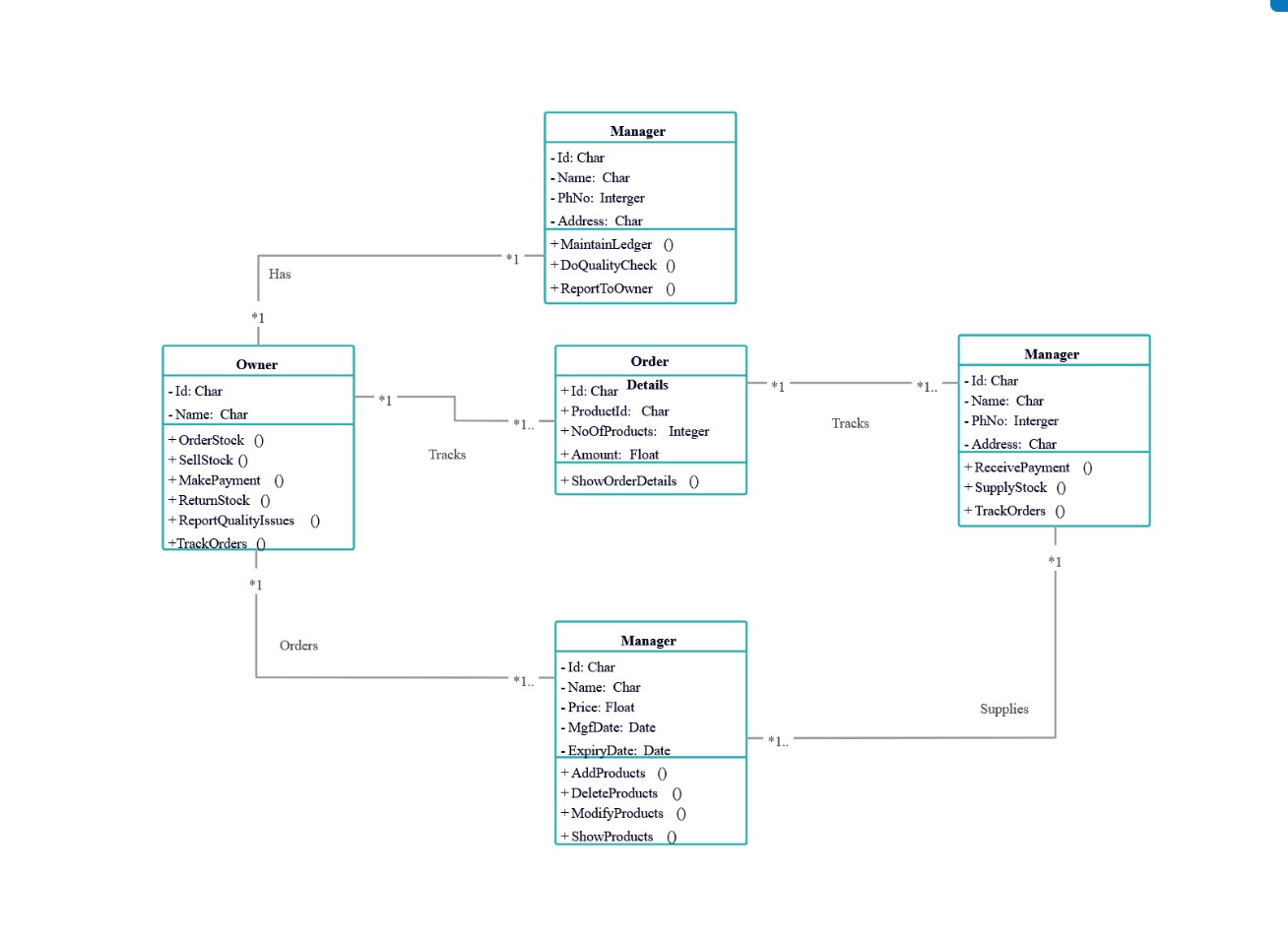
* **Actors**: Administrator, Warehouse Manager
* **Description**: This use case allows administrators and warehouse managers to add, edit, or remove inventory items, update quantities, and optimize warehouse layouts.
* **Extensions:** If validation fails at any step, the system prompts the user to correct the errors and resubmit the form.
* **Special Requirements:**
* The system must maintain data integrity and consistency when updating inventory information.
* The user interface should be intuitive and user-friendly to facilitate efficient inventory management.
* **Assumptions:**
* The user has received appropriate training on how to use the inventory management functionality.
* The system is capable of handling concurrent inventory updates without conflicts.



**[fig 15]**

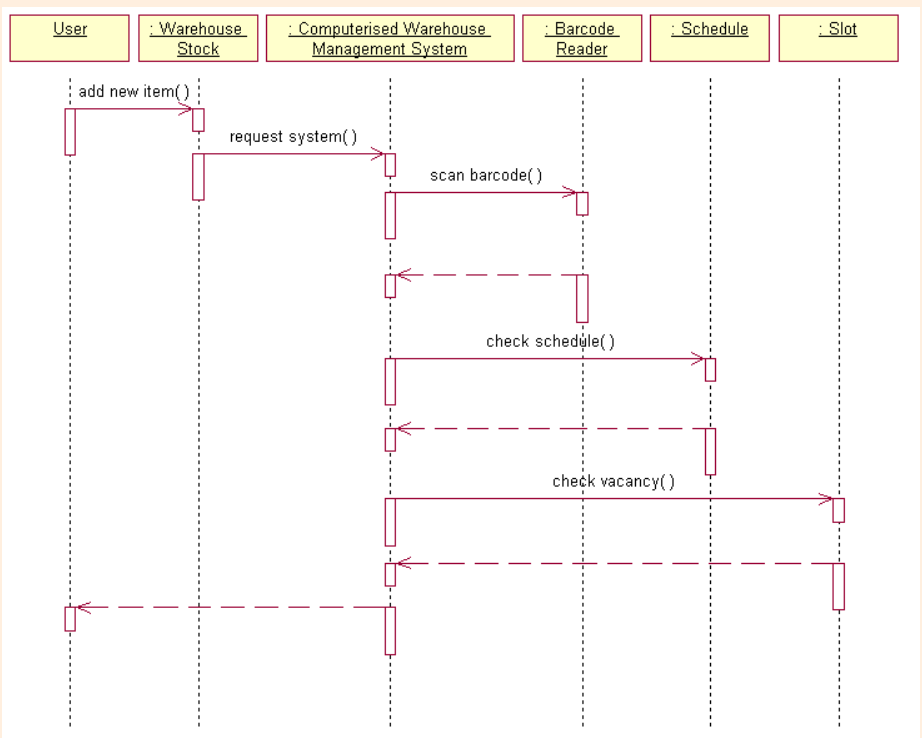
**2.6.2 Class Diagram :**

**Classes : Manger , Order , Owner**



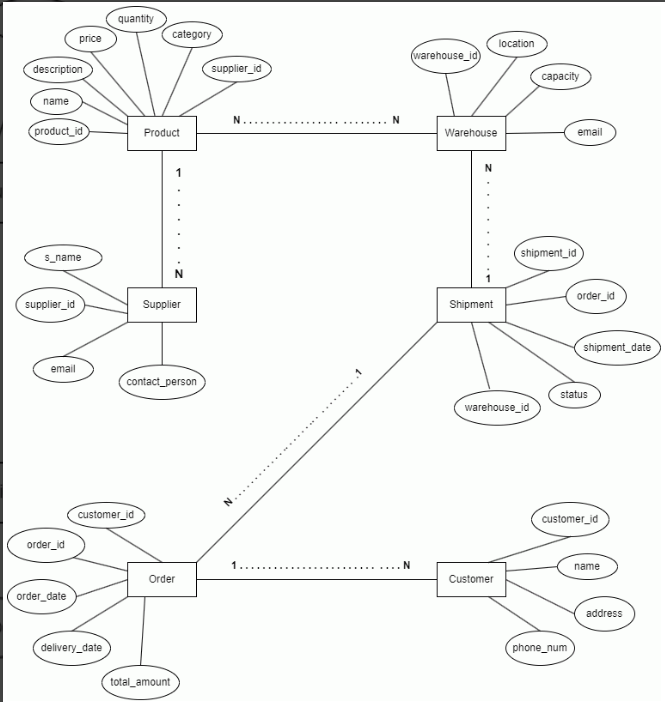
**[fig 16]**

**2.6.3 Sequence Diagram**



**Fig 18**

**2.6.4 E -R Diagram**



**Fig 19**

# Other Nonfunctional Requirements

## **2.7.1 Performance Requirements**

The inventory management software is optimized to function seamlessly across different systems, ensuring efficient inventory management without performance hiccups. It requires a minimum of a 500 MHz CPU, 128 MB RAM, and an OpenGL 1.2 compatible graphics card to support essential tasks such as tracking inventory, replenishing stock, optimizing warehouse layouts, managing product quality, monitoring expiration dates, and authorizing users. However, performance may fluctuate depending on database size and task complexity. Users handling larger inventories or complex operations might need hardware upgrades for better performance.

## **Safety Requirements**

To safeguard inventory data, the software undergoes regular updates and employs a comprehensive bug tracking system. Users are encouraged to report any encountered bugs or issues, which the development team addresses in subsequent updates. This proactive bug management approach minimizes the risk of data loss and ensures software reliability and security.

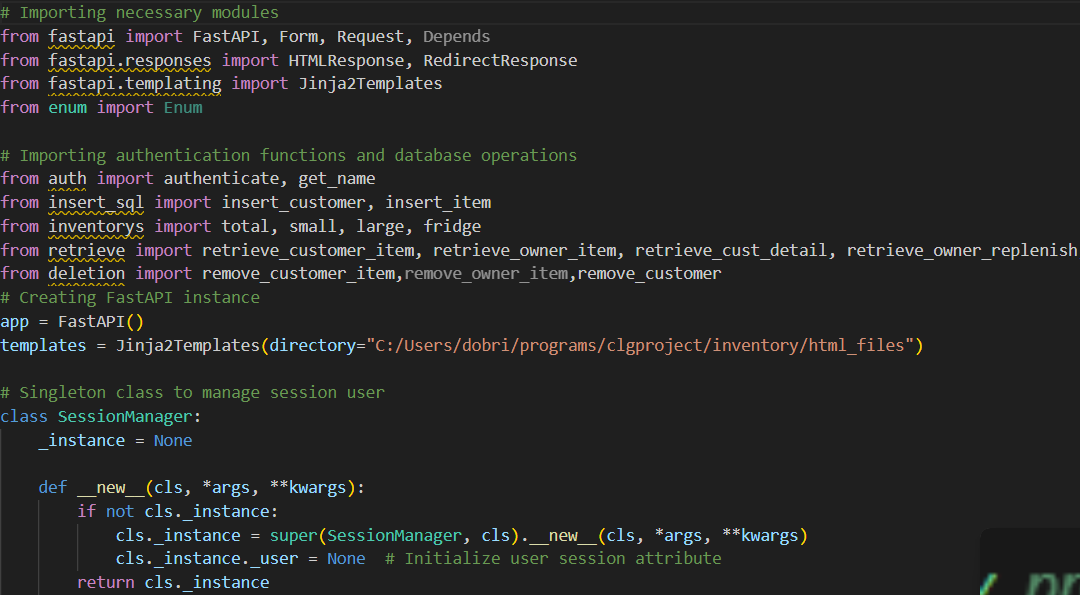
## **Security Requirements**

The inventory management software ensures security through role-based access control (RBAC), which regulates user permissions and limits access to authorized individuals. Users are granted specific authorization levels to carry out tasks like managing permissions and accessing detailed product quality information.

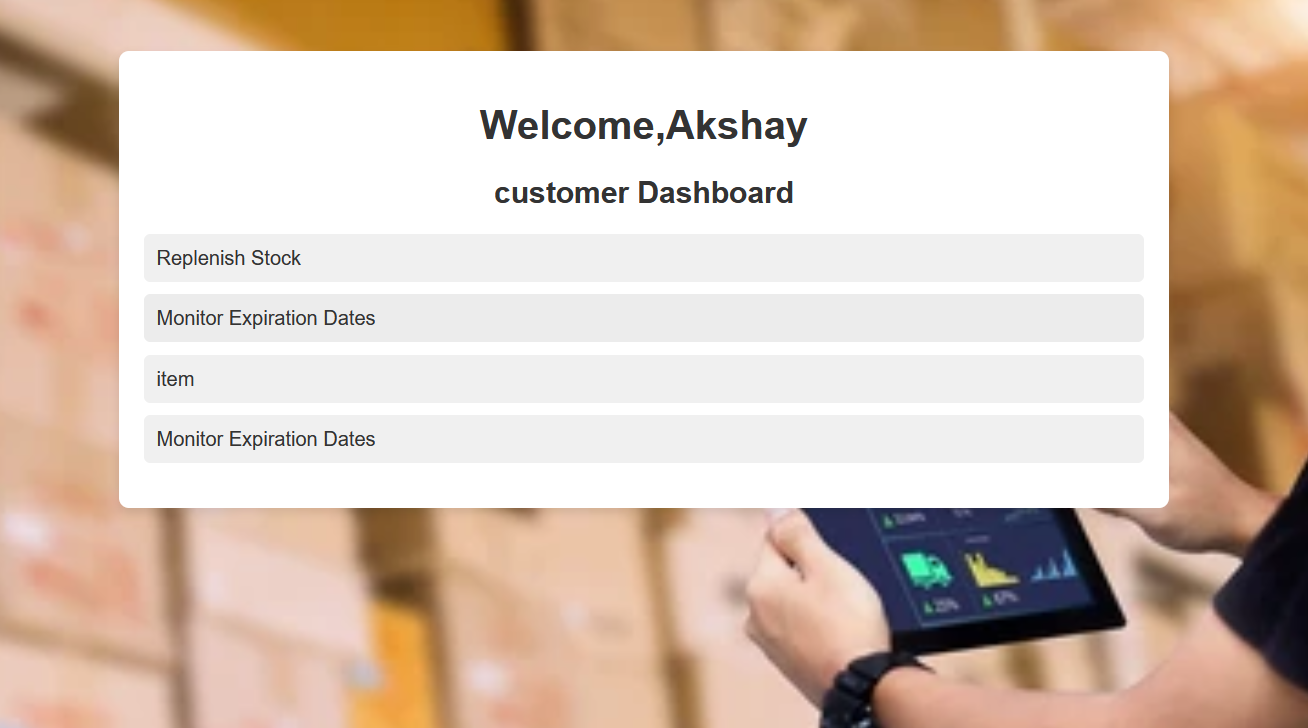
## **Software Quality Attributes**

The inventory management software prioritizes user-friendliness, catering to various user groups, including retailers, business owners, warehouse managers, staff, software developers, and engineers. It offers a range of features from basic inventory tracking to advanced warehouse layout optimization, ensuring usability regardless of technical expertise. The software boasts an intuitive interface for easy navigation and utilization of functionalities. However, users are advised to familiarize themselves with inventory management basics and software features to maximize its capabilities.

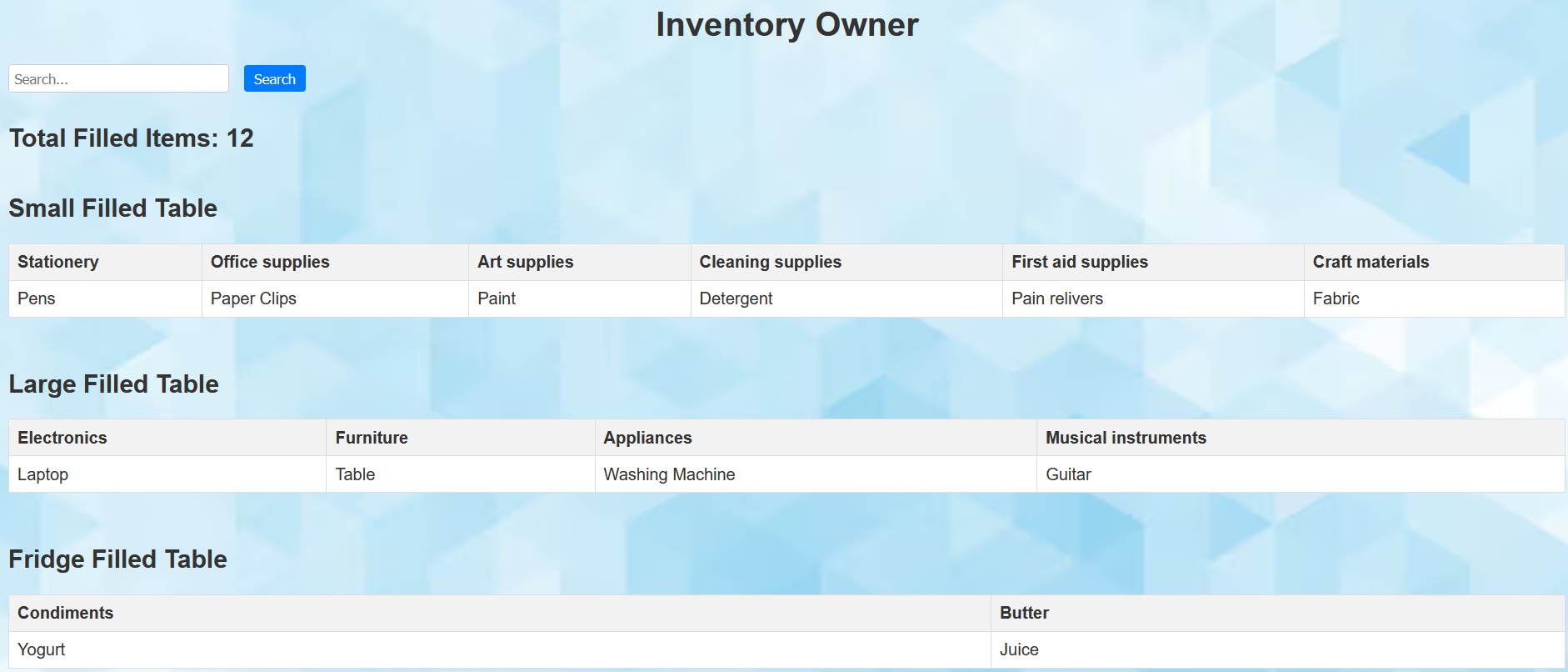
**3. RESULT**



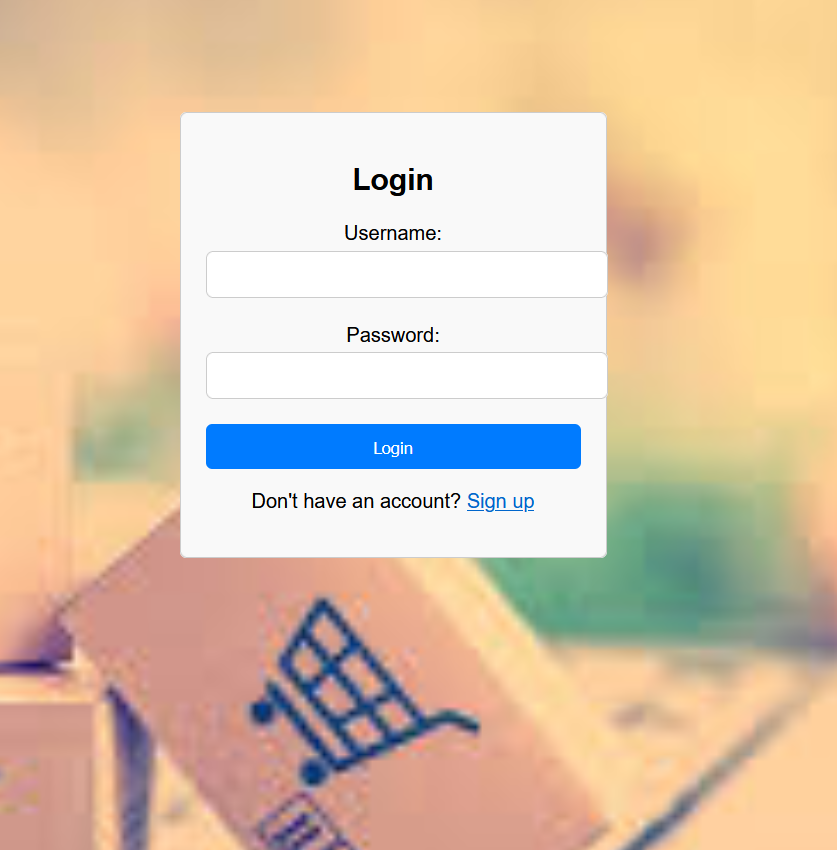
**Backend code Fig 3.1**



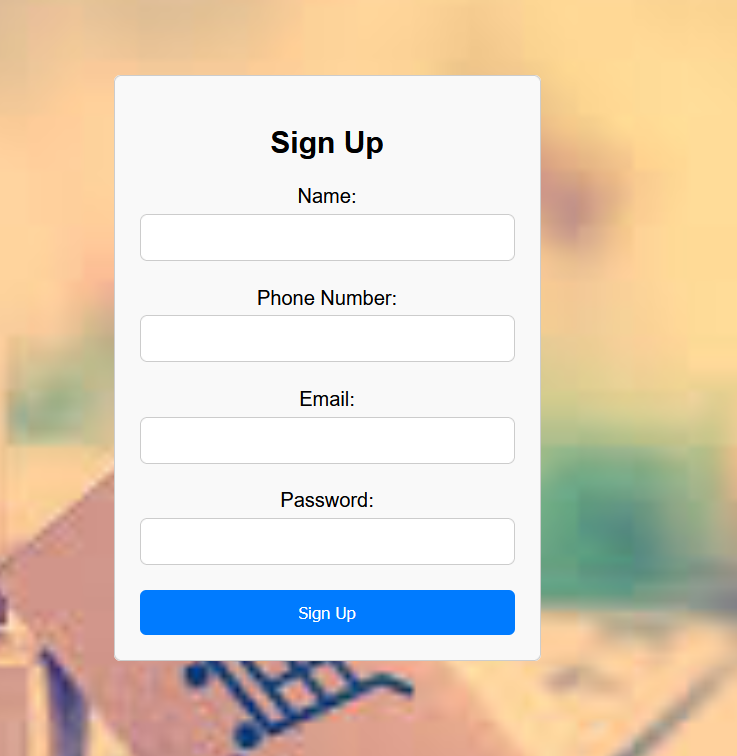
**Fig3.2**



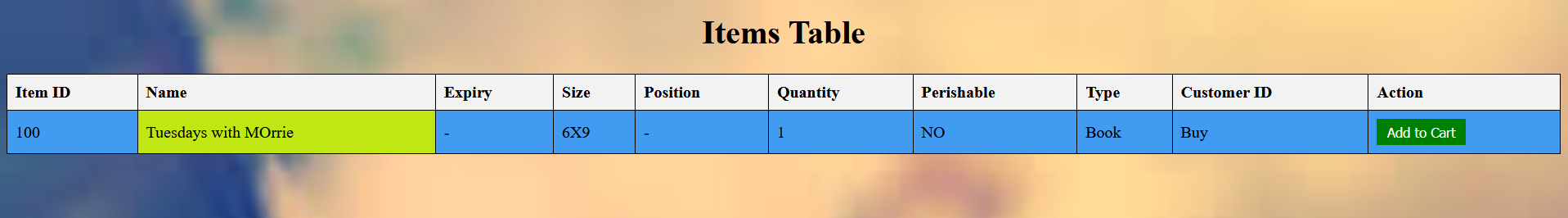
**Fig3.3**



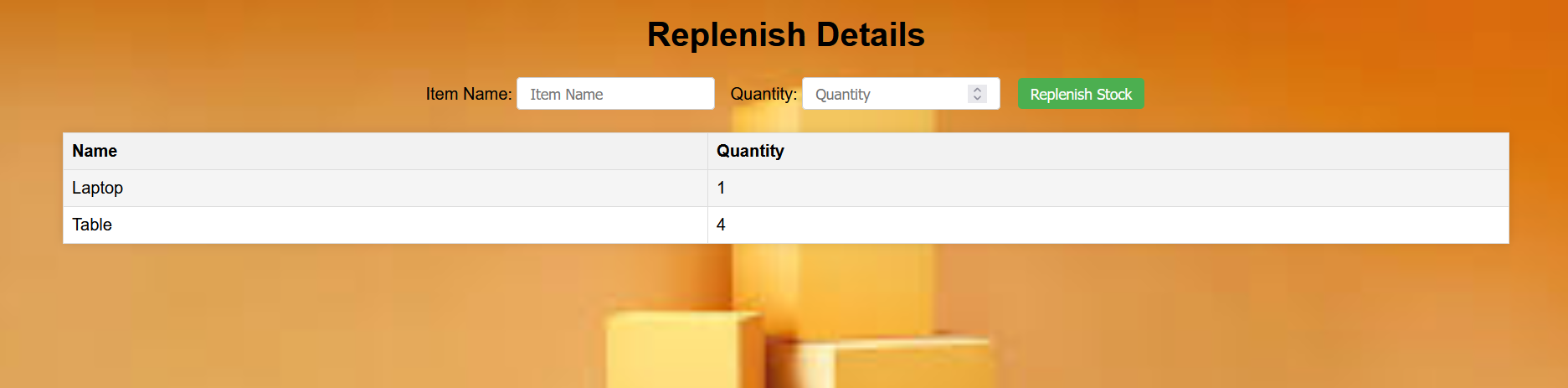
**Fig 3.4**



**Fig3.5**

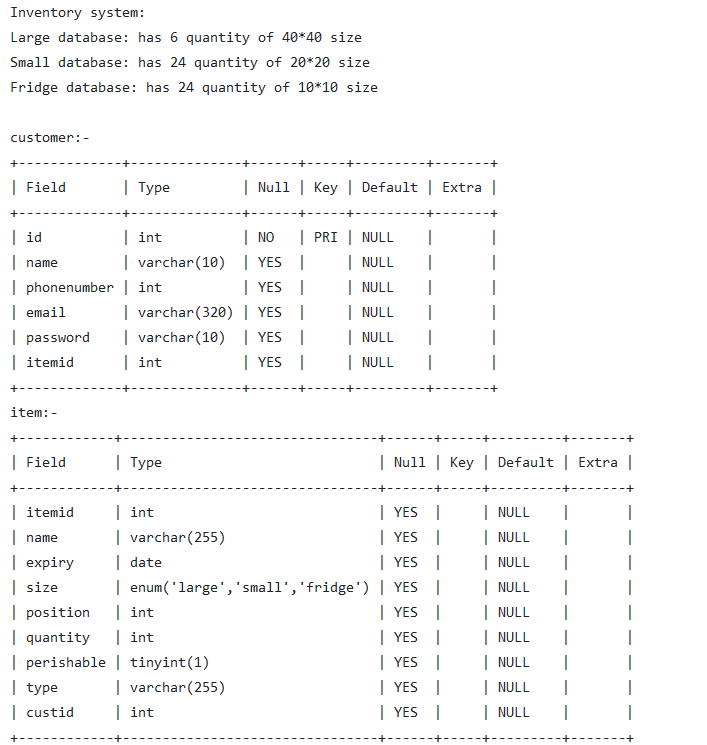


**Fig3.6**



**Fig 3.7**

**DATABASE :**



**Fig 3.8**

**4. CONCLUSION**

In conclusion, the development and implementation of the inventory management system have been successfully completed. The system offers several key features that improve the efficiency and accuracy of inventory management processes.

The system allows users to easily add, update, and delete inventory items, as well as track inventory levels and generate reports. The use of a centralized database ensures that inventory data is always up-to-date and accessible to authorized users.

Overall, the inventory management system provides a comprehensive solution for managing inventory and can help businesses reduce costs, minimize stockouts, and improve customer satisfaction. Future enhancements to the system could include integration with barcode scanners for faster data entry and implementation of predictive analytics for better inventory forecasting.

Top of Form

Bottom of Form

**5.REFERENCES**

[1] <https://en.wikipedia.org/wiki/Inventory_management_software>

[2] <https://www.researchgate.net/publication/372653862_Inventory_Management_Systems_IMS>

[3] <https://www.emerald.com/insight/content/doi/10.1108/RAMJ-09-2022-0145/full/html>

[4] <https://www.sciencedirect.com/science/article/pii/S2212827115012019>