**National College of Ireland**

**Higher Diploma in Computing**

**Software Development**

**2020/2021**

Vanessa de Oliveira Lyra

X19234554

vanelyra@hotmail.com

EasyPub – Stock Management System

Technical Report



**Table of Contents**

[Executive Summary 5](#_Toc80448596)

[1 Introduction 6](#_Toc80448597)

[1.1 Background 6](#_Toc80448598)

[1.2 Aims 7](#_Toc80448599)

[1.3 Technologies 7](#_Toc80448600)

[1.4 Definitions, Acronyms and Abbreviations 8](#_Toc80448601)

[2 System 9](#_Toc80448602)

[2.1 Requirements 9](#_Toc80448603)

[2.1.1 Functional requirements 9](#_Toc80448604)

[2.1.2 Data requirements 9](#_Toc80448605)

[2.1.3 User requirements 9](#_Toc80448606)

[2.1.4 Usability requirements 9](#_Toc80448607)

[2.1.5 Recover requirement 10](#_Toc80448608)

[2.1.6 Security requirement 11](#_Toc80448609)

[2.2 Design and Architecture 12](#_Toc80448610)

[2.2.1 Database 12](#_Toc80448611)

[2.2.2 System architecture 14](#_Toc80448612)

[2.2.3 Communication diagram 15](#_Toc80448613)

[2.2.4 Design 16](#_Toc80448614)

[2.3 Implementation 17](#_Toc80448615)

[2.3.1 Frames: 17](#_Toc80448616)

[2.3.2 Classes 18](#_Toc80448617)

[2.3.3 Code snippets 18](#_Toc80448618)

[2.4 Testing 35](#_Toc80448619)

[2.5 Graphical User Interface (GUI) Layout 36](#_Toc80448620)

[2.5.1 Login 36](#_Toc80448621)

[2.5.2 Forgot Password 36](#_Toc80448622)

[2.5.3 Admin user dashboard 38](#_Toc80448623)

[2.5.4 User dashboard 38](#_Toc80448624)

[2.5.5 Products, Category, Suppliers 40](#_Toc80448625)

[2.5.6 Other features 42](#_Toc80448626)

[2.6 Customer testing 45](#_Toc80448627)

[2.6.1 5 Seconds test 46](#_Toc80448628)

[2.6.2 Usability test and scale 47](#_Toc80448629)

[2.7 Evaluation 48](#_Toc80448630)

[2.7.1 Heuristic Evaluation 48](#_Toc80448631)

[2.7.2 GUI testing: 50](#_Toc80448632)

[3 Conclusions 62](#_Toc80448633)

[3.1 Advantages 62](#_Toc80448634)

[3.2 Disadvantages 62](#_Toc80448635)

[3.3 Opportunities and limits of the project 62](#_Toc80448636)

[4 Further development or research 63](#_Toc80448637)

[5 References 64](#_Toc80448638)

[6 Appendix 65](#_Toc80448639)

[6.1 Project Proposal 65](#_Toc80448640)

[6.1.1 Objectives 65](#_Toc80448641)

[6.1.2 Background 65](#_Toc80448642)

[6.1.3 Technical Approach 66](#_Toc80448643)

[6.1.4 Special resources required 66](#_Toc80448644)

[6.2 Project Plan 67](#_Toc80448645)

[6.2.1 Technical Details 68](#_Toc80448646)

[6.2.2 Evaluation 68](#_Toc80448647)

[6.3 Requirement Specification 68](#_Toc80448648)

[6.3.1 Purpose 68](#_Toc80448649)

[6.3.2 Project Scope 68](#_Toc80448650)

[6.3.3 Definitions, Acronyms, and Abbreviations 69](#_Toc80448651)

[6.3.4 User Requirements Definition 70](#_Toc80448652)

[6.3.5 Requirements Specification 70](#_Toc80448653)

[6.3.6 Requirement 2 <Register user> 73](#_Toc80448655)

[6.3.7 Requirement 3 <Maintain product> 74](#_Toc80448656)

[6.3.8 Requirement 4 <Maintain supplier> 76](#_Toc80448657)

[6.3.9 Requirement 5 <Maintain stock> 77](#_Toc80448658)

[6.3.10 Non-Functional Requirements 79](#_Toc80448659)

[6.3.11 GUI 79](#_Toc80448660)

[6.3.12 System Architecture 83](#_Toc80448661)

[6.3.13 System Evolution 84](#_Toc80448662)

# Executive Summary

The main object of this project is to develop an application aiming independent or small chain pubs. The program allows the users to record sales, print receipts for customers, manage users, manage suppliers and products by adding new, editing, or deleting them, record goods receipt, waste products or returned items. The system also allows the customer to generate sales and stock reports, the reports can be displayed on screen or exported to excel.

Users are divided in two categories: user and admin. The group defined as “user” has access to the point of sales area only and the “admin” group has total access to the application, this feature is determined by the “admin” when creating the new user.

In case someone forgets its password or simply needs to change it, it can be done via the forgo t password link in the login area.

# Introduction

## Background

I have worked in Supply Chain; stock management are for over nine years and three years’ experience working is small independent pubs. I decided to develop this project based on the problem detected constantly when it comes to stock in small business. Stock management consists in perform management of goods, it registers products inputs and outputs enabling its efficient use and minimizing investment.

A rigorous control of this process is the key to boost profitability and minimize costs generated by stock, the main goal is to avoid shortages or overstocking besides optimize processes, organize resources, and maximize profit return.

Overall, inventory Management has a great importance for companies, no matter how small they are, it is one of the keys to a successful business, although it is not an easy task it “can be a complex task for any business to master. Trying to optimise inventory stock levels can seem like an endless battle. Sometimes you end up with too much or too little. This trial-and-error saga continues for businesses who don’t have the right inventory management systems in place. Small businesses often lag when it comes to setting up their inventory management systems properly. Historically, inventory management systems were big investments and reserved for large companies.” ( [www.unleashedsoftware.com](http://www.unleashedsoftware.com)).

Bar managers and supervisors are intitled to do the restock, it is usually controlled manually via spreadsheet, it takes a long time and leads to encountered problems already discussed above, having a stock management software brings many benefits to small companies such as:

* avoiding costs and lost revenue coming from overstocking.
* automates tasks leading to reduced workload and time saving for managers once all stock records are available at the click of a button.
* manual form filling is discarded, reducing human error.
* reduction of shortages.
* better purchases.

The options available on the market are not affordable, overcomplicated and mostly directed to restaurant. The idea was to develop something free, pub-oriented to meet all needs from pubs.

## Aims

Application objectives:

* Allow users to add, edit and delete suppliers, products records.
* Create new user, change password.
* Generate sales and stock reports.
* Record sales in real time.
* Print receipt to customers.
* Store data in an online database in real time.
* Record goods receipt, waste and returned products.

The objective of this project is to deliver a point of sales and stock management together in one application, accessible to small pubs, also a user-friendly program considering pubs usually have a big rotation of staff, it is important to have an application that can be easily and fast learned.

## Technologies

The technologies chosen to develop this project were determined regarding my previous experience in college. The back end, to store all data on this application is MySQL and PHP localhost, to the front end I chose Java and Netbeans 12.3 IDE, libraries:

* JDK11 -java version.
* JCalendar 1.3.2 – calendar features.
* rs2xml- populate JTables.
* MySQL connector 8.0.25 – connect Java to database.
* POI 5.0.0 read and write Microsoft Office files.

## Definitions, Acronyms and Abbreviations

* POS (point of sale): area in the system where all sales are registered containing items, quantity, amount to pay.
* Java libraries: are already written classes created to perform tasks reducing coding.
* GitHub -open-source repository host, allows user to save their projects online and allows project collaboration.
* JFrames: is the screen to build Java GUI.
* GUI: graphical user interface, where the user communicates with the system.
* **Java:** programming language that produces software.
* MariaDB: open-source relational database management system, replacement for MySQL.

# System

## Requirements

### Functional requirements

The functional requirements continued the same in general, the only aspect that changed in a certain way was the create report topic. The user should be able to generate reports from several aspects of the system, due to the short period to complete the project, only two reports are available, stock levels and sales records.

### Data requirements

Data is stored in a localhost in PHP, the user can analyse data through tables available, generate reports and export them to excel for other analysis and documentation. Admin users can manipulate date and interact with the whole system.

### User requirements

The main user requirements to a stock management system are:

* Reliability and security: the system must be secure and deliver real-time information accurately.
* User-friendly: must be easy to use and does not require extensive training
* Agility: provide fast and easy inventory control.
* Scalability: able to allow additional features requiring effortless configuration.
* Traceability: the user needs to be able to track its inventory.
* Analytics: the system must provide analytics features and reports, this is a very important part of Stock Management.

### Usability requirements

The application design was developed based on usability. Pubs usually have a big rotation of staff all year around, so it was important to develop an easy to remember application, that would not require long training time, that could be intuitive and mainly not demanding.

### Recover requirement

**Database**

The Database can be easily exported from PHP, and it is saved on the local server. It allows the user to access data without internet, in case any internet outages, the system will work accordingly it is more secure since it is in the company’s device.

**GitHub**

Every change to the project is uploaded to GitHub, it stores and allows project management in a cloud, also it is possible to keep a version control of the project. Every change in the project was fetched to GitHub using GitHub Desktop, it detects the changes in your repository and the user upload it to the cloud.

Adress: https://github.com/vanelyra/EasyPub

Graphical user interface, text, application, email

Description automatically generated

*Github backup*

### Security requirement

Using a local server, makes it easier for the company to control access to it since the data is not transferred on the internet. The customer will need an antivirus and activate a firewall to ensure security.

## Design and Architecture

### Database

#### Design

Graphical user interface

Description automatically generatedThe user executes the application and its commands on the client side via terminal, the server reads and interprets and executes the commands using the SQL language.

#### Structure and tables

Database and tables

Graphical user interface

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

### Diagram, schematic Description automatically generatedSystem architecture

### Communication diagram

Diagram

Description automatically generated

This diagram presents the user interaction to the application. According to its classification – admin or user, he/she can interact with the whole appliction or only part of it.

Users classified as simply “user”, are allowed to: login, recover password. In case of sucesfull login, he/she will be redirected to a Main Page which will allow acess to the point of sales area.

Users classified as “admin”, after the login, will be redirected to a Main Page which grants acess to all application areas: Point of sales, Category, Others, Product, Stock, Suppliers. For safety reasons, only admins will be responsible for maintain data, except for sales, in the application.

### Design

The biggest objectives regarding design in this application were: keep it user-friendly and clean. The frames have the same design in green and white, there was not enough time to make the application look modern, I only simply changed the background and button colours.

The whole application follows the same design patterns as mentioned before. For user input, Jtextfields were adopted, the visual part of the data is displayed by tables, buttons were used to execute the applications queries, most warnings are displayed using JOptiopane, but some of them, used for validation purposes, are displayed in text fields. Some Jcomboxbox were employed to retrieve data from one table of the database to another or to retrieve data from the database to text fields.

Graphical user interface

Description automatically generated

## Implementation

The application consists in 11 JFrames representing the different sections of the program and two classes that execute parallel tasks to the application correct functioning:

### Frames:

* Login: the user inputs login, password, and user type to have your access granted or denied to the system. It also has access to the “Forgot Password” screen.
* Forgot password: The user needs to input username and answer a security question to recover his password.
* Main: after “admin” user login, this screen is displayed, it grants access to other pages on the application such as: Category, Others, Products, Sale, Stock, Suppliers.
* MainUser: after user type “user” login, this screen is displayed, it grants access the sales page only.
* Category: allow the user to add, edit and delete categories and consult any information regarding this section in the table available.
* Others: this page is divided in two sections, sales and stock reports can be generate in this page, and new users can be created.
* Products: allow the user to add, edit and delete products and consult any information regarding this section in the table available.
* Receipt: displays the data representation of the sales recorded on sale screen.
* Sale: that is the point of sales of the application, the user inputs products and quantities and the system calculate the sale amount.
* Stock: where the user is allowed to update the stock, input any goods receipt, waste or transaction voids.
* Suppliers: allow the user to add, edit and delete suppliers and consult any information regarding this section in the table available.

### Classes

* ConnectDB: this class connects holds the connection to the database.
* TextfieldLength: used for validation purposes, maximum character input in text fields.

### Code snippets

#### ConnectDB class

**Connection to database**

Graphical user interface, text, application

Description automatically generated

Class ConnectDB, connects java to database, the JDBC driver is loaded than the driver manager connects using the database URL, root, and password.

#### TextFieldLength

Graphical user interface, text, application, email

Description automatically generated

This class validates the maximum number of characters allowed in a text Field by extending PlainDocument. The number of characters will be defined in each frame by calling this function.

A screenshot of a computer

Description automatically generated with low confidence

That is the example of this function being called in a Jframe, the number of characters match the ones in the database.

#### Login

**Checking users**

Table

Description automatically generated with medium confidence

Checks if username or password are empty, if yes, a message appears to the user. Using a prepared statement, system verifies with database if user, password, and user type are correct, in affirmative case, the user is redirected to the main area according to your type, if not, “login failed” message is displayed.

#### Forgot password

**Recovering password**

Graphical user interface, text

Description automatically generated

When user presses enter after typing username, the system checks against database if it exists, is yes, the security question is displayed to the user, if not, user not found message is displayed.

#### Category, products, and suppliers

**Constructor**

Graphical user interface, text, application, email

Description automatically generated

Properties defined under constructor:

* Connection with database calling ConnectDB class.
* Frame cannot be resized by the user.
* When closed, frame is only hidden and do not close the system, system is terminated when user closes login or main page.
* Calling method to update Jtable with database data in real time.
* Method called to limit character in Jtextfields

**Custom ID numbers**

Text

Description automatically generated with low confidence

Method creates a customized ID numbers with letter and numbers retrieving the last – or maximum – ID registered in the database.

**Populating JTable with database**

Graphical user interface, text, application, email

Description automatically generated

This method populates the Jtable with database by using data from resultSet with DBUtils library.

**Button clear**

Graphical user interface, text

Description automatically generated

When this button is clicked, it clears all inputs in the JTextFields.

**Button Add**

Text

Description automatically generated with medium confidence

When the button Add is clicked, this method checks if any JtextField is empty, in affirmative case, a message will be displayed asking the user to input the information required, it checks if the ID field is filled, if it is, the new item cannot be added since the ID already exists. The query necessary for the task is insert and the JtextFields are matched to the right position to be added in the database. The method to update the database is called, a message of success in registering is display and the event matching the button clear is called to clear all data filled in the JtextFields.

**Button Edit**

Timeline

Description automatically generated with medium confidence

When button edit is clicked, method checks if the ID field is filled, if not, a message is displayed to the user, if it is, to query is executed updating the database with the text in the fields matched with its columns accordingly. The table is updated, and the button cleat event is called.

**Button delete**

Timeline

Description automatically generated

When button edit is clicked, method checks if the ID field is filled, if not, a message is displayed to the user, if it is filled, the system displays a yes or no warning message, asking if the user really wants to delete data, if the user chooses YES, the system will delete the row matching the ID selected.

**Selecting data from table**

Text

Description automatically generated

The user can select the data to be deleted or edited by clicking the row corresponding to the item. The info from the table will populate the JTextFields according to the mapping inside the method.

**Search field**

Text

Description automatically generated

The user can find data present on Jtables easily by using the search field, every character inputted by the user triggers the search automatically.

**JComboBox**

Graphical user interface, text

Description automatically generated

Method populates JcomboBox with data from database according to the selected table and column.

#### Others

**Button Generate**

Text

Description automatically generated

This button populates the report table, if no dates are chosen, it displays all sales records, if the user selects the dates, it will update the table accordingly.

**Button clear**

Text

Description automatically generated

This button is in the sales report panel and clears info from the JDateChooser fields.

**Button print**

A picture containing table

Description automatically generated

This button exports data present in the Stock Report panel to excel.

**Excel path choice**

Text, letter

Description automatically generated

Method allows user to choose the path to save the excel file generated.

#### Receipt

Graphical user interface

Description automatically generated with medium confidence Table

Description automatically generated

This code snippet shows the template of the sales receipt generated. It prints all data from the Jtable located in the sales Jframe.

#### Sale

**Button Add**

A picture containing text

Description automatically generated

When the button add is clicked, this method is called, it checks if the quantity (mandatory field) is filled, in affirmative case, it populates the Jtable with product, price, quantity it also calculates the total (quantity x price) and fill JtextField and Jtable with the result. This method also checks if there is enough inventory to the item being sold.

**Button Pay**

Timeline

Description automatically generated with low confidence

Graphical user interface, text, application

Description automatically generated

This function is called when button pay is clicked, it updates multiple tables in the database: sale, sale\_item and the column quantity on the product table according to the serving size registered in the database.

**Button print**

Text

Description automatically generated

Method called when button print is clicked, calls the method on the Receipt frame, and make it visible to the user.

**Button remove**

Text

Description automatically generated

Removes a selected row in the Jtable, recalculates total and updated total JTextField.

**ComboBox feature**

Graphical user interface, text, application

Description automatically generated

This combobox is editable, using “AutoCompleteDecorator.decorate(cbProduct);”, the user can find the required product by typing in the comboBox head.

#### Stock

**Button update**

Graphical user interface

Description automatically generated with low confidence

This code snippet shows the calculations to update stock, depending on the comboBox selection.

#### Validations

**Email**

Text

Description automatically generated

This is the email validation chosen. I tested many other options such as Regex, Apache commons validator but none of them worked properly. This one from Java mail does not works properly either, I accept any input if there is a “@” and at a least three digits after. It does not validate domain or dot.

Text

Description automatically generated

Calling validation function when user types on JtextField and alerting user of incorrect input.

Text

Description automatically generated

Since Java Mail validation only alerts the user regarding the incorrect input, I added another function to make sure the wrong input does not go through. When the user selects some other component on the system, if the alert field still shows incorrect input message, the input will be cleared.

Graphical user interface, text

Description automatically generated

**Case-insensitivity**

Text

Description automatically generated

All fields containing letters (except email) were set to uppercase to make the system case insensitive.

**Number validation**

Graphical user interface, text, application

Description automatically generated

If the user tries to type any character that is not a number, the JtextField will be blocked, backspace and delete are also allowed so the user can correct any input mistakes.

**Price validation**

Graphical user interface, text, application, email

Description automatically generated

Only allows the user to input numbers, one dot, backspace and delete. Error message is displayed.

## Testing

In this section I am going to present some of the tests carried on the application GUI.

#### Forgot password

**Recovering password**

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Security question retrieved from database successfully

Graphical user interface, application

Description automatically generated

Successful password recovery.

#### Category/Sales/Suppliers

Graphical user interface, application

Description automatically generated

**Add new category:**

Graphical user interface, application

Description automatically generated

Table updated with new data, category number generated by the system, success warning displayed

Graphical user interface, application

Description automatically generated

Successful edition and warning displayed

Graphical user interface, application

Description automatically generated

New item updated to the database

**Delete item**

Graphical user interface

Description automatically generated

Deletion warning displayed

**Others**

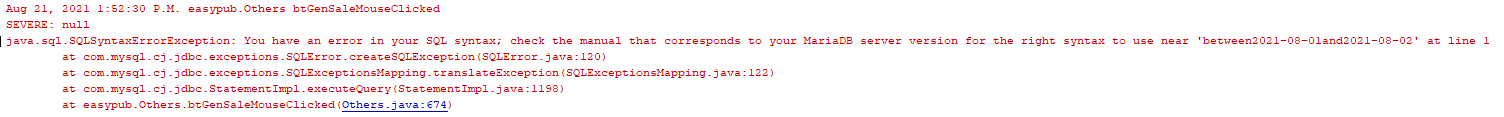
Graphical user interface, application

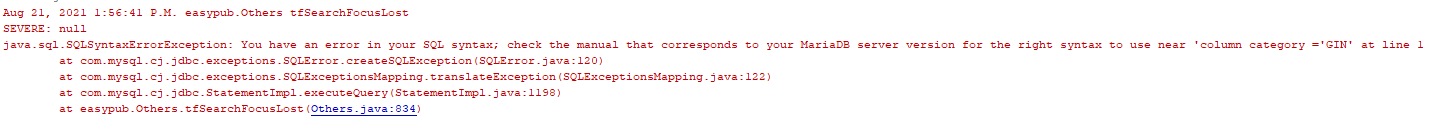
Description automatically generated

Two different validations displayed:

* Name: do not allow user to input incorrect characters
* Email: if email format in incorrect, the input is cleared when the field loses focus.

**Reports generation**





Syntax errors when trying to filter search on the report tables, this area of the application is not currently working.

**Stock**

Graphical user interface

Description automatically generated

**Adding stock**

Graphical user interface

Description automatically generated

Sucessfully added 50 pieces of ginger ale to the stock and message displayed

**Recording product waste**

Graphical user interface

Description automatically generated

Recorded the waste of 35 ml of rum, stock updated and message displayed

A picture containing graphical user interface

Description automatically generated

Transaction added to the database – table stock.

Graphical user interface

Description automatically generated with medium confidence

Stock updated accordingly in the database – table product.

**Point-of-sale area**

Graphical user interface

Description automatically generated

**Inputting sale to the system**

Graphical user interface

Description automatically generated

Price retrieved from the database when item is selected from combobox, number validation, only allow number input.

Graphical user interface

Description automatically generated

Products added to sales table, total quantity calculated.

**Receipt**

Graphical user interface, application, table

Description automatically generated

Example of receipt generated by the application.

A picture containing icon

Description automatically generated

After clicking button PAY, database sale table updated.

Graphical user interface

Description automatically generated with low confidence

Table sale item is also updated

## Graphical User Interface (GUI) Layout

### Login

Graphical user interface

Description automatically generated

The login area contains:

* Two text fields were the user inputs user and password.
* Label “Forgot password”, when clicked it redirects the user to the page with the same name.
* ComboBox where the user chooses its user type.
* Button submit, it checks the user information against the database.

### Forgot Password

Graphical user interface, application

Description automatically generated

After the username input, the user presses enter, if the input is correct the security question will be displayed. After answering the question, the user press “Verify”, if the data is correct, the password will be displayed in the password field.

In case the username or answer fields are empty, or the data does not match the database, the user will be notified via popup dialog box.

### Admin user dashboard

Diagram

Description automatically generated

The buttons redirect the user to they are names from.

### User dashboard

Diagram

Description automatically generated

It has only one button and redirects the user to the point of sales page.

### Products, Category, Suppliers

Graphical user interface, application

Description automatically generated

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated

The ID field is disabled for edition, the system generates an ID automatically when the user wants to create a new object. To delete or edit, the item must be chosen from the table, when a row is clicked, all text fields are filled with its respective data. The comboBoxes are populated from the database.

All text fields are mandatory, in case one is empty, the user is notified via popup alert. The search field works in real time, every character typed trigger a search on the table. Button “clear” erases all input in the text fields. When the user clicks the delete button, a Yes or No popup message is displayed to avoid deletions by mistake.

All validations happen using key typed event and an alert message is displayed in red when the user tries to input any not allowed character.

Types of validation:

* Only numbers: the system blocks the input of any character that is not a number, backspace or delete.
* Pricing format: allows only numbers and one dot.
* Letters only: the system blocks the input of any character that is not a letter, backspace or delete.
* Email requires a “@” and at a least three characters after. It does not validate domain.

### Other features

Graphical user interface, application

Description automatically generated

Table

Description automatically generated

This page is divided in two sections.

* Login

All text fields are mandatory, in case one is empty, the user is notified via popup alert. The fields name and last name are validated against numbers, if the user tries to type a number, a popup alert will be displayed, and the field cleared. Email field is also validated against email format, in case the format is incorrect, a message is displayed, and the input deleted. Button created, create the new user and clear deletes all user input.

* Reports

The user is allowed to create two typed of reports, sales, and stock. They are separated in a Tabbed panel, the sales panel is composed by a table, to date Chooser and three buttons. From the date choose the user defines the period to check the records, the table retrieves data from database according to the users’ needs, the button search generates the report, button print exports data to excel and allows the user to choose the path to save the document, button clear, clear data from date fields.

Stock is composed of a table, one text field and two buttons, within the text field the user can choose the category to be displayed. The table, clear and print have the same functions as described above.

### Point-of-sale

Graphical user interface

Description automatically generated

That is the point of sales area, where the user records the pubs sales. In the Combobox the user can find the product sold, he can look for the item opening the combobox or typing in it. After the selection, the price field will be populated automatically, and the quantity field will be put in focus. The quantity field is mandatory and validated to accept only numbers. The button Add, populates the table with data from the product, price, quantity and total – which is also calculated automatically – the print button generates a receipt, and the pay button updates de database with info from the table.

### Stock

Graphical user interface

Description automatically generated

In this tables, the user updates stock, choosing the item from the table and updating the quantity according to the comboBox selection, the waste selection reduces the stock. The actual quantity displays the quantity presented on the Jtable.

## Customer testing

I asked two people to test my application, both meet the demographic target of the application. One of them has with 17 years’ experience in the business and works as bar manager for the last 10 years, the second has no experience in the area. I chose these two profiles to test the application to be able to have an evaluation in a professional view – who would use the whole system and train new people - and in a beginner view – considering the staff rotativity mentioned before. I will mention the users during the test as user 1 and user 2, number one will be the professional overview.

### 5 Seconds test

That was the first test applied to the user, I quickly showed the application and collected their first impressions with some questions:

What is the purpose of the system? Describe in a few words.

User 1: “Keep stock control, automated control of stock.”

User 2: “to help manage stock in pubs”.

What is the name of the application?

User 1: “I can`t remember, but I remember it was a cool one”.

User 2: “EasyPUB”.

What was your first impression regarding design?

User 1:

User 2: “it has a nice and clean layout which makes accessible for all levels of users.”

Which public does it target?

User 1: “bar managers and supervisors.”

User 2: “pubs, more specifically barman’s and bar managers”

Did you identify the main elements?

User 1:” simple, easy to use, everything is transparent, all features are easy find.”

User 2: “stock control, suppliers, registration, sales.”

Any feedback?

User 1: “The idea of the application really good, easy to understand and would not require too much training. The only problem identified is the sales area, it is overcomplicated, it would not work properly in a busy environment. The better way would be using buttons to the products, and nothing typed.”

User 2: “the EasyPub is very useful software which has many features to offer to both manager and floor staff. The unique access provided for general users and admin also guarantee more safety regarding the information added to the system.”

### Usability test and scale

In this test I showed both users all the detailed features of the system and asked them to perform some simple tasks, the trickier tasks overall were input and record sales and recover the password. Regarding the sales input as user 1 gave a feedback is complicated and takes long if the sale has many items. Regarding the recover area, to retrieve the security question, the user must press enter after input the username, that was not specified so made the task “harder” than it should be.

The user had been given some questions after the test regarding the application, they should give a grade from one to five.

Questions:

1. I would use this app in my business
2. System was overcomplicated
3. The app was easy to use
4. I would not require extensive training to use the app
5. The system was integrated properly
6. The system was inconsistent
7. People could learn to use the app easily
8. The system is very complex
9. I felt confident using this app
10. I need to learn many features before using the app

A screenshot of a computer

Description automatically generated with medium confidence

Average result 77.5, according UIUXTrend, the application got an A grade, considered to be excellent in “effectiveness, efficiency, and overall ease to use”.

## Evaluation

### Heuristic Evaluation

Although the system was not designed based on the heuristic evaluation, I found it to be one of the best ways to drive the evaluation of this application.

**Visibility of System Status**

**The system always keeps the user informed if their actions were successfully updated to the database via popup message.**

**Match between system and real world**

The system was designed to the user without the need of a logical order considering how simple it is to navigate. The main page has only button to redirect the user to the page required, when closing the page after use, the user will see again the main page.

**User control and freedom**

When clicking any delete buttons while working in the app, a popup warning is displayed to the user asking if he really wants to delete data or not. He must click the yes option to confirm the action.

In the sales area, the user can remove any incorrect items before finishing the purchase. In the stock area, the user can add back stock selecting the option void.

**Consistency and standards**

The same design patterns were adopted in the system, data displayed in tables, popup messages maintain the same pattern of text, buttons, and headers with same colour pattern

**Error Prevention:**

**The system was designed to be case insensitive except where the case sensitivity was required (username, password), email, number, name, and letter validations were developed accordingly, fields that require user input have character limitation to match database design. In case the user tries to input something that goes against the validation, the system warns accordingly via popup message or label warning, in some cases it even blocks typing forbidden characters.**

**Recognition rather than recall**

**The system works with a database so the ID`s are considered primary keys so they cannot be repeated. The IDs are generated automatically by the system and the user is not allowed to input data in these fields. Also, if he tries to input some item with the same ID, he will be warned and not allowed.**

**Flexibility and efficiency of use:**

The system was designed in an intuitive way, allowing new users to navigate easily.

**Aesthetic and minimalist design:**

**The system was designed with clean and simple design to meet its user-friendly requirement.**

**Help users recognize, diagnose, and recover from errors**

**As mentioned above in case the user tries to input something incorrect or nor allowed, he will be warned.**

**Help and documentation:**

**Unfortunately, the application does not meet this requirement, there is no section with documentation to help users.**

# Conclusions

## Advantages

* The application is user-friendly and easy to use.
* It helps business to control stocks level.
* It reduces the workload of bar managers.
* Discard the use of spreadsheet
* Makes it easier to check stock situation and sales history

## Disadvantages

* Sales area is not designed properly for busy environments.
* Reports area not working

## Opportunities and limits of the project

The point of sales screen requires improvement. As user 1 mentioned, it should be designed differently to meet busy environment requirements, instead of a ComboBox, the products must be displayed as buttons and the quantity should be typed on screen. I am not particularly happy on how this frame ended up designed but I spent a lot of time trying to design the right way, but I could not find a solution with my level of expertise that would not require big chunks of coding and that still would allow the client to add or edit new data and I was not able to code dynamic buttons.

Due to the delivery date of this project, I did not have time to make this feature work, there is a problem on the prepared statement syntax.

Sending an email every morning to an admin email informing all products with low stock was one of the ideas presented on the project proposal, this idea evolved to a table localized in the stock area, that was an important feature for future clients but ended up dropped due because I could not develop it.

The stock calculations overall need improvement, the system now is limited to addition or subtraction of stock. I would like to add some stock management algorithms to calculate and inform the user the exact reposition time based not only on reaching the reorder point quantity but also include lead time and sales history.

Another dropped idea that unfortunately could be one of the selling points of this application was the cocktail creation and cost calculation, it would allow users to create cocktails using the products already registered as its components and calculate the cost for them, there was a lack of time and expertise to develop it.

# Further development or research

Stock management is a vast area, so the application has a lot of room to grow also, due to the short time to deliver this project, many features idealized at first had to be dropped to deliver a functional and running application.

Some of the features to be added in the future:

* Area displaying all items that reached the reorder point.
* Improve the calculations on the system including stock management algorithms instead of addition and subtraction.
* Area to allow the user to create and calculate the cost for cocktails.
* Create a table plan where the drinks can be associated to a table.
* Improve the point of sales screen, making easier to operate.
* Area to generate purchase orders to suppliers.

# References

https://www.business.org/finance/inventory-management/why-is-inventory-management-important/

https://www.unleashedsoftware.com/blog/small-businesses-implement-inventory-management

<https://stackoverflow.com/questions/46326822/java-regex-first-name-validation>

<https://mailtrap.io/blog/java-email-validation/>

[https://lucid.app/lucidchart/7a9b35a8-d03a-4e25-8651-859319110d9f/edit?beaconFlowId=5742BE28E167B437&page=0\_0#](https://lucid.app/lucidchart/7a9b35a8-d03a-4e25-8651-859319110d9f/edit?beaconFlowId=5742BE28E167B437&page=0_0)

<https://docs.oracle.com/javase/8/docs/api/index.html?javax/swing/text/PlainDocument.html>

<https://stackoverflow.com/questions/2739617/what-is-the-regular-expression-for-valdating-a-price-in-a-jtextfield>

<https://www.mariadbtutorial.com/mariadb-basics/mariadb-select/>

<https://www.netsuite.com/portal/resource/articles/inventory-management/inventory-management.shtml>

<https://www.guru99.com/usability-testing-tutorial.html>

<https://usabilitygeek.com/how-to-use-the-system-usability-scale-sus-to-evaluate-the-usability-of-your-website/>

<https://uiuxtrend.com/measuring-system-usability-scale-sus/#calculation>

https://blog.prototypr.io/10-usability-heuristics-with-examples-4a81ada920c

# Appendix

## Project Proposal

### Objectives

The objective of this project is to development of program to manage stocks in pubs. The program developed in Java aims the automatization of this time-consuming activity in small business, reducing shortages or surpluses.

The application will have the following functionalities:

* **User authentication**: Only the accounts set as admin will be allowed to create new users.
* **Product module:** this module allows adding, deleting, editing, or viewing products and product costs calculation.
* **Stock module:** allows the user to enter, view or change stock.
* **Supplier module:** allows the user to create, delete, edit, or view suppliers.
* **Sales module:** the only module non admins will be able to use. Allows to record the sales and waste. It will be done by clicking the buttons corresponding to the drinks, it will generate a receipt with the total sale.
* **Reports:** every module will generate reports according to the client’s necessity such as: total sales report, stock details, supplier details, etc.

### Background

With nearly nine years’ experience in Supply Chain and Purchasing, and three years’ experience working in pubs in Ireland, I couldn’t help noticing how time consuming and problematic the stock management of small pubs is. The task is done by employees with no experience in stock management and sometimes by untrained personnel (due to holidays for examples). It leads to constant shortages or surpluses and discard of expired products, that is not good for businesses, especially small ones where the profits are usually smaller.

This app will be a point-of-sale system, designed especially for pubs with stock management and purchasing features, usually the applications for bar management needs to be integrated with a POS system, leading to more expenses to owners.

### Technical Approach

The objective of this project is to create an application of simple understanding, once the final users will be Bar Managers and Head Bartenders, not exactly professionals with specialized stock management background. The program will be developed in a way that all the features can be easily found and reports easily generated, it will do all the stock calculations and send the admin notifications when products reach a determined stock level.

The application will be developed in Java via InteliJ, the database adopted will be MySQL. Requirements for the project:

* Create methods for Stock management calculations.
* How to generate reports in excel.
* How to generate e-mail with stock levels.
* How to generate notifications of products with low level stock.

### Special resources required

GitHub and StackOverflow, Stock Management books.

## Project Plan

Timeline

Description automatically generated

### Technical Details

The programming language chosen to this project is Java using InteliJ IDE. For the Database, MySQL will be adopted and Microsoft Project for the project plan.

### Evaluation

The application will be tested by Head bartenders of small pubs.

## Requirement Specification

### Purpose

The purpose of this document is to set out the requirements for the development of a Pub management system. This Java application allows sales control and stock management for small pubs. It is intended to be simple and easy to understand, the application focuses on helping users to control stock levels in an automated way.

The intended customers are all pub staff specially ones responsible for stock reposition.

The main idea is to gather a POS and stock management into one simple application, discarding the use of another system to control sales and automating the stock control reducing shortages and surplus of products.

### Project Scope

The scope of the project is to develop a user-friendly Java application. Helping small business to control stocks and avoid unnecessary expenses, also helping owners to have a better overview of their stocks and sales. The system shall have an easy and clear navigation template.

#### Project objectives

Users not classified as admin will be able to register sales and take payments from customers and their change password. Users classified as admin will be allowed to use all features from the system. After logging in, the dashboard will be displayed, and the user can choose which page to go.

Other’s page: the admin will be allowed to create new users from name and password, the system will create and ID number automatically. In this section a range of sales and stock reports will be available and will be generated in excel.

Products and suppliers: the admin will be allowed to create, edit, and delete information regarding both, input authentication will be adopted accordingly and presented to the user.

Sales: all products will be available in this dashboard, the user will be allowed to record all products sold, quantities, check total sales amount. Admin will be allowed to record waste products when necessary.

Stock management: in this section the user will input or edit all necessary information to the stock calculation for example, minimum stock necessary, lead time. Also modify stock quantities when goods are received.

Every data will be stored in a Database and will be available whenever needed. The system will notify the user all items that reached reorder point via email.

#### Motivation for this project

The motivation to develop this project was based on my experience working in the environment in Ireland and talking to bar managers, usually smaller business has only a POS available, and the stock management is controlled manually, it is a time-consuming task to workers that are already overloaded.

### Definitions, Acronyms, and Abbreviations

POS Point of Sale

GUI Graphical User Interface

BOM Bill of Material

### User Requirements Definition

The users require an application that can be easily understood and easy to manage and work with. The pub environment is usually fast paced, and the staff turnover is constant, so the features must be easy to find and easy to learn how to deal with. Also, the stock control must be as simple as possible once the responsible for the task is not a Supply chain specialist.

### Requirements Specification

The system requires thirty minutes training for bartender users and three hours training for admin users.

#### Functional requirements

Make sale: system shall record all sales made by bartenders

Maintain stock: update stocks according to sales and stock input.

Maintain product: allow creation, deletion end edition of products.

Maintain supplier: keep record of suppliers’ information.

Create report: allow user to generate diverse reports to check sales performance and stock levels

Register new user: create a new user ID and accept a password to allow system navigation.

##### Use Case Diagram

### Diagram Description automatically generated

##### Requirement 1 <MAKE SALE>

#### Description & Priority

Using the sales section, the user inputs all the products and quantities sold to the customer and payment type. The screen will display the total amount of the purchase so the customer can be charged correctly. In case of any error, the transaction can be voided. This use case has high priority to the systems proper functioning.

#### Use Case

**Scope**

The scope of this use case is to allow the user to input every sale on the system.

**Description**

This use case describes the procedure to input sales and payment type and void incorrect transactions.

**Use Case Diagram**

Diagram, letter

Description automatically generated

**Flow Description**

**Precondition**

The user already logged in and entered the sales module.

**Activation**

This use case starts when a user makes a sale.

**Main flow**

1. The user enters the sales module on the system
2. The user inputs the customer order
3. The user inputs the payment type
4. The system displays total amount of the order
5. The user finishes the sale <See E1>

**Exceptional flow**

E1 : <Incorrect sale input>

1. The user chooses the void feature
2. The user inputs the wrong product from the sale
3. The system voids the sale
4. The system returns to the sales module

**Termination**

The use case terminates when the user finishes the sale.

**Post condition**

Stocks and sales reports are updated.

### Requirement 2 <Register user>

#### Description & Priority

The admin user accesses the register module and creates a new user, inputting name and password. The system will automatically generate a numerical ID to the new user. The new user can choose to change the password or keep the one created by the admin. The priority for the system is low once the admin has access to every feature on the system.

#### Use Case

**Scope**

The scope of this use case is to allow the user to create new users and change password.

**Description**

This use case describes the procedure to register new users on the system and how to change password when necessary.

**Use Case Diagram**

Diagram

Description automatically generated

**Flow Description**

**Precondition**

The system is in register user module.

**Activation**

This use case starts when an admin enters the register user module.

**Main flow**

1. The system identifies the admin account type
2. The user inputs name and password to the new user
3. The system creates a numerical login and displays it.
4. The admin informs the new user login details
5. The user enters the system using login and password<See A1>

**Alternate flow**

A1 : <Change password>

1. The user inputs login and password
2. The user clicks “Change password” button
3. System checks details
4. The system opens a screen to change password
5. User inputs old password and new password
6. User clicks “Change password” button
7. System check details and display change password message
8. System goes back to login area.

**Termination**

The system terminates when it displays the login info to the new user.

**Post condition**

The system goes into a wait state

### Requirement 3 <Maintain product>

#### Description & Priority

The admin will sign in and create, edit, or delete products. High priority, requirement is vital to systems correct operation.

#### Use Case

**Scope**

The scope of this use case is to allow the user to create, edit or delete products from the system and create products with a BOM.

**Description**

This use case describes the correct procedure to maintain products data up to date.

**Use Case Diagram**

Diagram

Description automatically generated**Flow Description**

**Precondition**

The user logged in using an admin account.

**Activation**

This use case starts when an admin goes to the products module.

**Main flow**

1. The user chooses the edition or creation of product
2. The system displays the correct form
3. The user created or edit product <See A2>
4. The system records the edition or creation
5. The system updated the database

**Alternate flow**

A2 : <Deletion of product>

1. The user will choose edition mode
2. The user finds product to be edited
3. The system displays product information
4. The user clicks the “Delete” button
5. The system updates the database

**Termination**

The system terminates when the user clicks Create, delete, or edit buttons.

**Post condition**

The system goes into a wait state

### Requirement 4 <Maintain supplier>

#### Description & Priority

The admin will sign in and create, edit, or delete suppliers. Low priority, at this stage of the project, suppliers are only for information.

#### Use Case

**Scope**

The scope of this use case is to allow the user to create, edit or delete suppliers from the system.

**Description**

This use case describes the correct procedure to maintain suppliers’ data up to date.

**Use Case Diagram**

Diagram

Description automatically generated

**Flow Description**

**Precondition**

The user logged in using an admin account.

**Activation**

This use case starts when an admin goes to the supplier’s module.

**Main flow**

1. The user chooses the edition or creation of supplier’s data
2. The system displays the correct form
3. The user created or edit supplier <See A3>
4. The system records the edition or creation
5. The system updates the database

**Alternate flow**

A3 : <Deletion of product>

1. The user will choose edition mode
2. The user finds supplier to be edited
3. The system displays supplier information
4. The user clicks the “Delete” button
5. The system updates the database

**Termination**

The system terminates when the user clicks Create, delete, or edit buttons.

**Post condition**

The system goes into a wait state

### Requirement 5 <Maintain stock>

#### Description & Priority

The admin will sign in and create, edit stock data. High priority, requirement is vital to systems correct operation.

#### Use Case

**Scope**

The scope of this use case is to allow the user to create, edit stocks from the system. Also allows waste input.

**Description**

This use case describes the correct procedure to maintain stocks up to date.

**Use Case Diagram**

Diagram

Description automatically generated**Flow Description**

**Precondition**

The user logged in using an admin account.

**Activation**

This use case starts when an admin goes to the stock management module.

**Main flow**

1. The user chooses to edit stock data or add waste
2. The system displays the stock form
3. The user updates stock
4. The system records the change
5. The system updates the database

**Termination**

The system terminates when the user clicks save button.

**Post condition**

Stocks are updated.

### Non-Functional Requirements

The non-functional attributes required by the system are:

#### Performance/Response time requirement

The performance will depend on the user’s device, its operating system and memory.

#### Availability requirement

The application will require internet connection and a personal computer, laptop. It will be available on GitHub for download.

#### Recover requirement

The application backup information everyone hour.

#### Security requirement

System will not allow non authorized people to use the application. Only admin accounts will be able to make all input on the system and create new users, other accounts will only be allowed to register sales and waste.

#### Maintainability requirement

Code will be properly divided in sections and comment to facilitate any updates.

#### Interface requirements

### GUI

The first page displayed the user will be the login page, after the successful login, the user will have access to the dashboard, it will display the other system sections: Sales, products and suppliers, stock management and others.

LOG IN PAGE

A picture containing diagram

Description automatically generated

DASHBOARD

Graphical user interface, diagram

Description automatically generated

Graphical user interface

Description automatically generatedSALES SECTION

OTHER FEATURES SECTION

A picture containing chart

Description automatically generated

PRODUCTS AND SUPPLIERS SECTION

Graphical user interface

Description automatically generated with low confidence

STOCK SECTION

Diagram

Description automatically generated with medium confidence

### System Architecture

Diagram

Description automatically generated

Each requirement will be your own class, the user class will be divided into user and admin user.

### System Evolution

The next steps for the project will be electronic invoice reading to update stocks automatically and send PO`s automatically to suppliers. Also make the application available for smartphones and developing an inventory module which calculates the quantity on open bottles via camera.