

MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
SUMY STATE UNIVERSITY
FACULTY OF ELECTRONICS AND INFORMATION TECHNOLOGY
COMPUTER SCIENCE DEPARTMENT

"Approved for defense."

Acting Head of the Department

_____ Igor SHELEHOV
(signature)

09.06.2023

GRADUATION THESIS

for obtaining the educational degree of Bachelor

in the specialty 122 - Computer Science,
educational-professional program "Informatics"
on the topic: "AUTOMATION OF AN INVENTORY SYSTEM"
by the student of group IN-95AH, OKOTIE PHILIP MAMUS.

The Bachelor Graduation Thesis contains the results of original research. The use of ideas, results, and texts of other authors is properly referenced to the respective sources.

OKOTIE PHILIP MAMUS



(signature)

Supervisor
Candidate of Technical Sciences,
Associate Professor.

Natalia Barchenko

(signature)

Sumy – 2023

SUMY STATE UNIVERSITY
FACULTY OF ELECTRONICS AND INFORMATION TECHNOLOGY
COMPUTER SCIENCE DEPARTMENT

«Approved»

Acting Head of the Department

Igor SHELEHOV

(signature)

TASK FOR THE GRADUATION THESIS

to obtain the educational degree of Bachelor

in the specialty 122 - Computer Science,

educational-professional program "Informatics"

by the student of group IN-95AH, Okotie Philip Mamus.

1. Topic of the Bachelor Graduation Thesis: "AUTOMATION OF AN INVENTORY SYSTEM"
approved by the order of SumDU on June 1, 2023. № 0475-VI

2. The deadline for the submission of the Bachelor Graduation Thesis until June 9, 2023.

3. Input data for the qualification work

4. Table of Contents for the Explanatory Memorandum (List of questions to be addressed)

1) Analysis of the subject area, defining the purpose, and forming the tasks of the Bachelor Graduation Thesis. 2) Review of the theoretical material. 3) Development of the automation of an inventory system. 4) Analysis of the obtained results.

5. List of graphic materials (with specific mention of mandatory drawings)

6. Project consultants (with the corresponding sections of the project they are associated with).

Section	Consultant	Signature, date	
		The assignment has been issued	The assignment has been accepted

7. Date of assignment issuance «____» _____ 20 ____

The assignment has been
accepted for execution

Supervisor

(signature)

(signature)

CALENDAR PLAN

№	Titles of the stages of the Bachelor Graduation Thesis	Deadline	Note
1	<i>Analysis of the subject area, defining the purpose, and forming the tasks of the Bachelor Graduation Thesis</i>		
2	<i>Review of the theoretical material</i>		
3	<i>Development of the automation of an inventory system</i>		

4	<i>Analysis of the obtained results</i>		
5	<i>Bachelor Graduation Thesis Formatting</i>		

Higher education student

Supervisor



(signature)

(signature)

ABSTRACT

Note: 35 pages, 5 figures, 0 tables., 1 appendix, 11 reference.

Justification of the relevance of the research topic – the topic of the qualification work is relevant, as it is devoted to the solution of an important practical task of inventory automation.

Research object – inventory process.

Research objective – automation of the inventory system.

Research methods – Front-End And Back-End Technologies To Manage Inventory Data, Automate Processes, And Provide A User Interface For Interacting With The System.

Results - the creation of a software package for an Inventory system, which is able to reduce time and human resources.

DBMS, DESIGN OF THE DATABASE, ER-MODEL, INVENTORY SYSTEM

TABLE OF CONTENT

INTRODUCTION	5
1 ANALITICAL REVIEW	6
1.1 Automated Inventory System	6
1.2 Objectives of an Automated Inventory System	11
1.3 Task Formulation	13
2 SELECTION OF PROBLEM SOLVING METHODS	14
2.1 Statement of the problem	14
2.2 Database design	15
3 SOFTWARE IMPLEMENTATION	20
3.1 Database Design	20
3.2 Database Structure	21
3.3 Data Structure Pacakge	26
CONCLUSIONS	28
REFERENCES	30
APPENDIX	31

INTRODUCTION

Relevance – the topic of the qualification work is relevant, as it is devoted to the solution of an important practical task of inventory automation. Inventory is one of the methods of financial accounting and an important means of control over the preservation and consolidation of material values for certain objects and materially responsible persons. This allows for consistent and reliable accounting. Automated inventory systems can help businesses maintain regulatory compliance by providing accurate, up-to-date inventory data and facilitating traceability throughout the supply chain. This can be particularly important for businesses in regulated industries, such as pharmaceuticals, food and beverage, or aerospace.

Object of research – inventory process.

Subject of research – automation of the inventory system.

Hypothesis – the development of an automatic inventory system allows to simplify the labor-intensive accounting process at the enterprise.

Novelty – developed information technology that allows you to conveniently perform the inventory process.

Structure. This work consists of an introduction, an analysis of publications, a statement of the research problem, a choice of methods and tools for solving the problem, a description of the software implementation of information technology, conclusions, a list of used sources and applications.

1 ANALITICAL REVIEW

1.1 Automated Inventory System

Inventory management is a process that helps businesses keep track of their products and raw materials [1]. By tracking inventory, manufacturers can ensure that they have the right amount of stock on hand to meet customer demand. An effective inventory management system can help businesses improve their bottom line by reducing the cost of goods sold and increasing sales. Inventory management software is made up of several key components working together to create a cohesive inventory of many organization's systems. These features include Reorder Point, Asset tracking, Service management, Production Identification, Inventory optimization [1-3].

Reorder point: A Reorder Point system is an inventory control that helps businesses maintain an appropriate inventory level and avoid shortages. Should inventory reach a specific threshold, a company's inventory management system can be programmed to tell managers to reorder that product. This helps companies avoid running out of products or tying up too much capital in inventory [1] .

Asset tracking: Asset Tracking refers to the method of tracking physical assets either by scanning barcode labels attached to the assets or using tags using GPS, BLE, LoRa or RFID that broadcast their location .Asset tracking is just as important as managing your inventory, because you need to know the location, status, maintenance schedule, and other important information about your organization's physical assets. Indeed, asset tracking is important to your organization's bottom line and compliance, as you are responsible for locating and replacing lost or missing physical assets, as well as those that have come to the end of their lifecycle [1].

Service Management: A set of specific organizational capabilities for providing consumers with value in the form of services is referred to as service management. Businesses must constantly strive to raise the standard of their services while guaranteeing client satisfaction, cost restraint, and operational effectiveness. Various

frameworks, approaches, and standards can be used to accomplish effective service management.

Following are some crucial elements of service management [2]:

1. Information Technology Infrastructure Library (ITIL): The ITIL framework is a well-known one for IT service management (ITSM). It offers a list of best practices for supplying high-quality IT services, putting the focus on matching business requirements with IT procedures. Service strategy, design, transfer, operation, and continuous improvement are among the topics covered by ITIL.

2. Service Design: Planning and organizing the infrastructure and resources needed to perform a service are the goals of service design. It entails identifying the service requirements, specifying the customer demands, and developing the infrastructure, tools, and processes required to satisfy those needs.

3. Service Transition: The purpose of this phase is to put the designed services into use. To guarantee a seamless transfer of new or modified services, it involves tasks like change management, release and deployment management, and knowledge management.

4. Service Operation: This phase handles the regular administration of services. For the purpose of ensuring that services are provided effectively and efficiently, it comprises procedures like incident management, problem management, and event management.

5. Continual Service Improvement (CSI): CSI strives to continuously assess and enhance services. For better service quality and efficiency, it entails identifying areas for improvement, reviewing performance, and making changes.

6. SLA (Service Level Agreement): An SLA is a contract that outlines the standard of service that a customer expects from a supplier. It contains key performance indicators (KPIs) to make that the service lives up to the customer's expectations, like response times, availability, and reliability.

7. IT Service Desk: Customers can report incidents, request services, and get help from the support desk, which serves as a single point of contact. It is in charge of

directing customer communications and coordinating the handling of problems and service requests.

8. IT Service Portfolio: A company's complete inventory of all IT services is contained in its service portfolio. It includes details about the objective, elements, price, and value that the service offers to clients.

9. Service Catalog: The active services that are currently offered to clients can be found in a service catalog, which is a subset of the service portfolio. Customers can use it as a communication tool for understanding the services offered to them and the expenses involved.

10. Service Management Tools: These tools are designed to assist firms in more efficient service management. Processes can be automated, incidents and service requests can be tracked, performance can be measured, and reports may be generated for analysis and improvement. Organizations may improve the quality of their services, increase customer happiness, and maximize operational efficiency by putting service management strategies into effect.

Organizations may improve the quality of their services, increase customer happiness, and maximize operational efficiency by putting service management strategies into effect.

Production identification: This is the practice of identifying and tracking items in a unique way throughout the course of their lifecycle, from production to distribution and finally to the final consumer. This method is necessary for effective supply chain operations, inventory management, and quality control.

Here are a few typical techniques and equipment for production identification:

1. barcode: A barcode scanner can read a barcode as a series of lines or patterns that convey data. They are employed to encrypt data about the product, including its maker, identification number, and other specifics. For the purpose of maintaining and tracking products, barcodes are frequently used in logistics, warehousing, and retail.

2. QR Codes: Also known as "Quick Response" codes, these two-dimensional barcodes have a larger data storage potential than standard barcodes. Smartphones or

specific QR code readers may scan them. Many applications for QR codes exist, including inventory control, marketing, and product identification.

3. RFID (Radio Frequency Identification): RFID is a wireless technology that transmits data between a tag and a reader via electromagnetic communication. Products with RFID tags connected to them can be traced and identified throughout the supply chain. Compared to barcodes, RFID has a number of benefits, including a greater read range and the capacity to read several tags at once. Monitoring and handling products via warehousing and logistics.

4. Serial numbers are individual alphanumeric codes that are assigned to every product during production. They support the tracking of specific items by suppliers and manufacturers for warranty, quality assurance, and recall purposes.

5. Batch or Lot Numbers: A specific collection of products that were created together under the same circumstances are identified by their batch or lot number. They are particularly significant in sectors like food, medicine, and chemicals, where monitoring and traceability are essential for quality assurance and security.

6. Global Trade Item Number (GTIN): GTIN is an international standard for product identification. Retailers, producers, and suppliers utilize it to manage products across various marketplaces and distribution channels. For convenient identification and tracking, GTINs can be encoded in barcodes or RFID tags.

7. Product Labels: Product labels provide crucial details about a product, including its name, brand, components, and usage guidelines. Additionally, barcodes, QR codes, and serial numbers are frequently used on labels to help customers identify products.

8. PIM systems are software tools that centralize and manage product information, including product identifiers, descriptions, photos, and other data. They assist businesses in making sure that accurate and consistent product information is accessible via all platforms and points of contact.

Effective production identification methods must be put into place in order to manage inventory, ensure quality control, and sustain a productive supply chain. Businesses may monitor and handle their products throughout their lifecycle by

utilizing technology like barcodes, RFID, and PIM systems, ultimately enhancing operational efficiency and customer happiness.

Inventory optimization: The process of ensuring that the appropriate amount of stock is available at the appropriate time and location to satisfy customer demand while reducing costs and optimizing profitability is known as inventory optimization. For businesses to avoid stockouts, overstocking, and obsolescence, which can have a negative influence on customer satisfaction, cash flow, and overall business performance, efficient inventory management is essential.

The following are some ways and tactics for optimizing inventory:

1. Forecasting demand: For the purpose of optimizing inventories, accurate demand forecasting is crucial. Businesses can forecast future demand and change their inventory levels accordingly by examining historical sales data, seasonality patterns, and market trends.
2. Safety stock: Safety stock is extra inventory kept to reduce the danger of stockouts brought on by demand fluctuations or long lead times. The ideal safety stock level is determined by variables including the desired service level, demand volatility, and lead time volatility.
3. . Reorder point: To replenish stock before it runs out, a fresh order should be placed at that inventory level. It is determined by taking lead time demand and safety stock levels into account. Stockouts and overstocking can be avoided by routinely keeping an eye on inventory levels and establishing suitable reorder points.
4. Inventory turnover ratio: This performance indicator gauges how frequently a company's inventory is sold and replaced over a predetermined time frame. A greater turnover ratio denotes better inventory control. Monitoring inventory turnover on a regular basis can give information about how well inventory optimization measures are working.
5. Inventory optimization software: To automate the operations of demand forecasting, inventory tracking, and replenishment, many organizations employ

inventory optimization software. These technologies can assist businesses in increasing the effectiveness of their inventory management while minimizing manual labour.

Businesses may guarantee that they have the appropriate amount of inventory on hand to meet consumer demand while minimizing expenses and boosting revenue by employing inventory optimization tactics. As a result, there may be an increase in customer satisfaction, increased cash flow, as well as supply chain efficiency.

1.2 Objectives of an Automated Inventory System

To provide a more comprehensive understanding of automated inventory systems, let's discuss some additional aspects and benefits of implementing such a system in your business [1-3]:

1. **Scalability:** Automated inventory systems can grow with your business, accommodating increased product ranges, storage locations, and transaction volumes. This scalability ensures that your inventory management processes remain effective and efficient as your business expands.

2. **Customization:** Many automated inventory systems offer customization options, allowing you to tailor the system to your specific business requirements. This might include custom reporting, unique workflows, or integration with industry-specific tools or platforms. Customization can help ensure that the system meets your unique needs and improves overall inventory management.

3. **Streamlined communication:** An automated inventory system can facilitate better communication and collaboration within your organization. By providing real-time access to inventory data, team members across various departments can make informed decisions and work together more effectively. This can help ensure that sales, purchasing, and warehouse teams are aligned and working towards common goals.

4. **Regulatory compliance:** Automated inventory systems can help businesses maintain regulatory compliance by providing accurate, up-to-date inventory data and facilitating traceability throughout the supply chain. This can be particularly important

for businesses in regulated industries, such as pharmaceuticals, food and beverage, or aerospace.

5. Improved customer service: By maintaining optimal stock levels and enhancing order fulfilment processes, automated inventory systems can help improve customer service. This can lead to increased customer satisfaction, repeat business, and positive brand reputation.

6. Sustainability: Implementing an automated inventory system can contribute to your business's sustainability efforts by reducing waste and promoting more efficient use of resources. By optimizing stock levels and reducing excess inventory, you can minimize waste and lower your environmental impact.

7. Competitive advantage: Businesses that adopt automated inventory systems can gain a competitive advantage over their rivals by streamlining operations, reducing costs, and improving decision-making. This technology-driven approach to inventory management can help your business stay ahead of industry trends and adapt to changing market conditions.

In summary, automated inventory systems offer a wide range of benefits and can play a crucial role in optimizing inventory management processes. By considering the various aspects, objectives, and advantages discussed above, you can better understand the value of implementing an automated inventory system and make an informed decision for your business.

1.3 Task Formulation

This software package is considered as one of the stages in the development of an automated system called "Computer and Office Equipment Inventory Management." Therefore, it will address the following tasks:

1. Maintenance of directories of equipment used in the enterprise.
2. Inventory management based on storage and installation locations.
3. Compilation of aggregated data on components used in computer equipment.
4. Preparation of data for conducting inventories.

The creation of this product aims to achieve the following objectives:

1. Automation and improvement of inventory management and control.
2. Reduction of labour intensity and organization of management activities.

In this regard, the following requirements should be taken into account:

1. The database should be capable of storing information on over 100 units of equipment.
2. The user interface style should include resizable graphical dialog windows with convenient mouse navigation.
3. User-friendliness.
4. Low software and hardware requirements.

2 SELECTION OF PROBLEM SOLVING METHODS

2.1 Statement of the problem

Production management and organisation play a significant role in the creation of software packages. The goal of this last qualifying project is to create a software package for creating a goods inventory. Developing and implementing an automated inventory system can be a complex process, with numerous challenges that may arise [3-5]. Some common problems include:

1. **Data accuracy and consistency:** an effective inventory system requires accurate and consistent data. This includes keeping product information, quantities, and locations up to date. Stock outs, overstocking, and other inventory management issues can be caused by inaccurate data.
2. **Integration with existing systems:** it can be difficult to integrate the automated inventory system with other systems like point-of-sale, enterprise resource planning, or warehouse management systems. Maintaining accurate inventory levels and preventing errors require seamless data flow between these systems.
3. **Scalability:** the inventory system must be able to handle increasing amounts of data and transactions as the business grows. This necessitates nimble and scalable system architecture capable of accommodating expansion without sacrificing performance.
4. **User adoption and training:** when implementing a new system, employees must often be trained to become familiar with the system's functionalities. it can be difficult to encourage user adoption and make sure the system is user-friendly.
5. **Costs and resource allocation:** developing and implementing an automated inventory system can be costly, particularly for small and medium-sized businesses. Allocating appropriate resources to the project while staying within a reasonable budget can be difficult.
6. **Security and data privacy:** protecting sensitive business information is critical, and any inventory system must have strong security measures in Place. It Is Also Critical To Ensure Data Privacy And Compliance With Applicable Regulations.

7. Customization: Each Business Has Different Inventory Management Needs. A One-Size-Fits-All Solution May Not Be Appropriate, And Customization May Be Required To Meet Specific Business Requirements.
8. Technology Selection: Choosing The Right Technology Stack For The Automated Inventory System Is Critical To Its Success. This Includes Choosing The Right Hardware, Software, And Programming Languages, As Well As Taking Into Account Factors Like Usability, Flexibility, And Integration Capabilities.
9. Real-Time Data Processing: To Provide Accurate Information On Stock Levels, Sales, And Other Relevant Metrics, An Automated Inventory System Must Process Data In Real-Time. To Handle Large Volumes Of Data Without Delays, Efficient Algorithms And Data Processing Techniques Are Required.
10. System Maintenance And Updates: An Automated Inventory System, Like Any Other Software System, Requires Ongoing Maintenance And Updates To Ensure That It Remains Effective And Compatible With Other Systems. This Includes Bug Fixes, Feature Updates, And Ensuring The System Remains Current With Industry Trends And Best Practices.

2.2 Database design

There are two approaches to database design: bottom-up design and top-down design [4].

Top-down design begins by defining datasets, then defining the data elements for each of those datasets. This process involves identifying the different types of entities and defining the attributes of each entity. Top-down design includes decomposition operations, which involves replacing the original set of relations included in the database schema with another set of relations that are projections of the original relations.

This approach is recommended in cases where the number, variety, and complexity of entities, relationships, and transactions are significant in size. The most common models for this design are entity-relationship models [5].

Bottom-up design begins by identifying data elements, which are then grouped into data sets. First, attributes are defined, which are then combined into entities. Bottom-up design includes synthesis operations, which involves the execution of a layout from a given set of functional dependencies between objects of the subject area of the source relations of the database scheme [6].

This approach is recommended if you are developing a small database with a small number of objects, attributes, and transactions. The main stages of database design are on the fig.2.1.[7]

Conceptual (infological) design — construction of a semantic model of the subject area, that is, an information model of the highest level of abstraction. Such a model is created without focusing on any particular DBMS and data model. The terms "semantic model", "conceptual model" and "infological model" are synonymous. At this stage, objects, relationships between objects, attributes, and key attributes are defined [8].

The specific form and content of the conceptual model of the database is determined by the formal apparatus chosen for this purpose. Graphical notations similar to ER diagrams are commonly used.

Most often, the conceptual model of the database includes:

description of information objects or concepts of the subject area and connections between them.

description of integrity constraints, i.e. requirements for permissible data values and relationships between them.

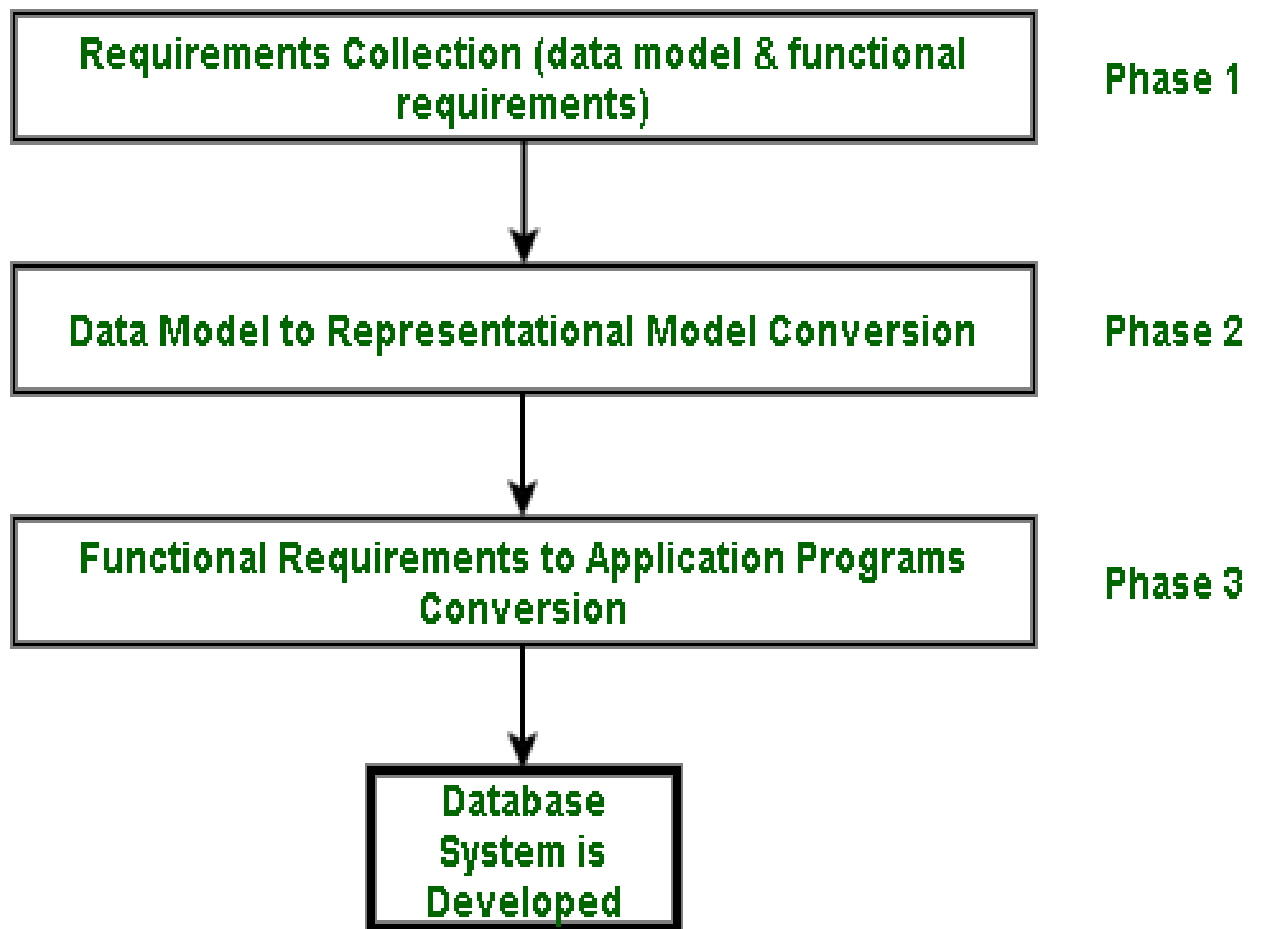


Figure 2.1 - The main stages of database design

Logical design — creating a database scheme based on a specific data model, for example, a relational database model. For a relational data model, this is a set of relationship schemas, usually specifying primary keys, as well as "links" between relationships, which are foreign keys.

The transformation of a conceptual model into a logical model is usually carried out according to formal rules. This step can be largely automated.

At the stage of logical design, the specifics of a specific data model are taken into account, but the specifics of a specific DBMS may not be taken into account. Physical design — creation of a database scheme for a specific DBMS. The specifics of a specific DBMS may include restrictions on the naming of database objects, restrictions on supported data types, and others. In addition, the specificity of a specific

DBMS during physical design includes the choice of solutions related to the physical data storage environment (choice of disk memory management methods, database division by files and devices, data access methods), creation of indexes, and others [9,10].

The entity-relationship model (ER-model) (Entity-relationship model or entity-relationship diagram) is a data model proposed by P. Chen [11], which allows describing conceptual schemes using generalized block structures. An ER model is a data meta-model, that is, a means of describing data models. There are a number of models for knowledge representation, but one of the most convenient tools for a unified representation of data independent of the software that implements it is the entity-relationship model. It is important that all existing data models (hierarchical, network, relational, object) can be generated from the "entity-relationship" model, therefore it is the most general [8].

The main advantages of ER models [9]:

clearness;

models allow designing databases with a large number of objects and attributes;

ER models are implemented in many automated database design systems;

The main elements of ER models (fig.2.2):

objects (entities);

attributes of objects;

connections between objects.

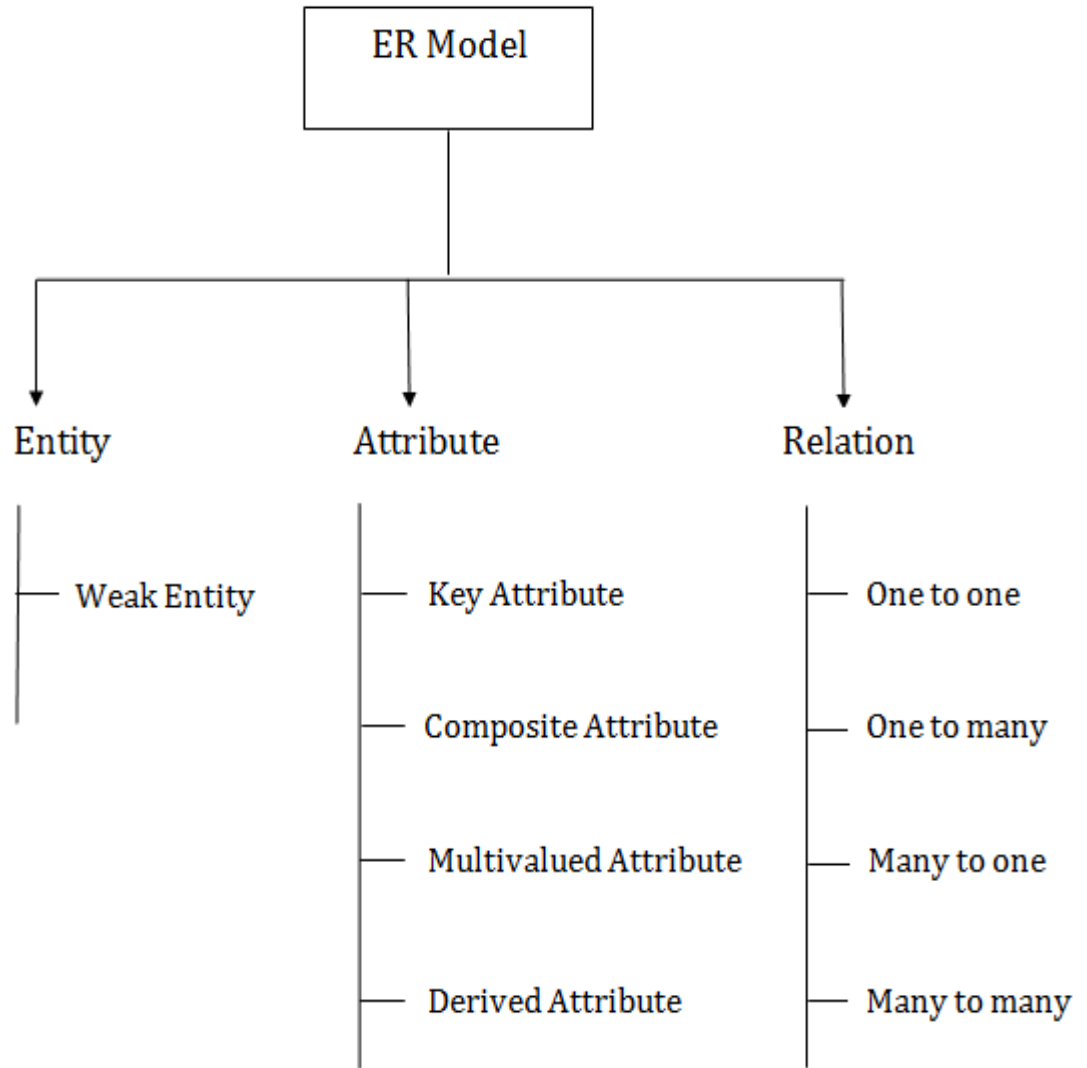


Figure 2.2 – The main elements of ER models

3 SOFTWARE IMPLEMENTATION

3.1 Database Design

A Javascript-Based Inventory System Typically Combines Front-End And Back-End Technologies To Manage Inventory Data, Automate Processes, And Provide A User Interface For Interacting With The System. All Information Must Be Entered Into The Javascript Database For The Inventory System Being Developed, Which Can Be Controlled By Javascript Frameworks.

Working Directly With Databases Is Not A Common Use For Javascript. Javascript, However, Can Be Used To Develop Web Applications That Communicate With Databases Using Apis Or Server-Side Scripting Languages. These Are Some Of The Factors That Influence My Decision To Build Web Applications Using Javascript.

1. **Data Storage:** The Inventory System Keeps Data In A File System Or Database. This Data Consists Of Details Regarding The Products, Stock Levels, Orders, And Other Pertinent Information.
2. **Front-End Interface:** To Create A User Interface That Enables Users To Interact With The Inventory System, Javascript Is Used. This Interface Can Be Accessed Through A Web Browser Or Other Client Application
3. **Back-End Processing:** The Inventory System Processes Data, Automates Tasks, And Interacts With The Database Using Back-End Technologies Like Php, Node.Js, Or Python. Adding And Updating Inventory Data, Processing Orders, And Producing Reports Are All Tasks That This Back-End Code Takes Care Of.
4. **Automated Stock Level Updates, Order Processing, And Alerts** When Inventory Levels Drop Below A Predetermined Level Are Just A Few Examples Of The Automation Features That The Inventory System May Offer.
5. **Reporting:** The Inventory System Is Capable Of Producing Reports On Stock Levels, Sales, Orders, And Other Pertinent Statistics. These Reports Can Be Viewed In The User Interface Or Exported To Other Applications.

6. Integration: The Inventory System Can Be Integrated With Other Programs Like Accounting Software, E-Commerce Platforms, And Shipping Systems. The Need For Manual Data Entry Is Diminished Thanks To This Integration, Which Enables Seamless Communication Between Systems.

In Order To Manage Inventory Data, Automate Procedures, And Provide A User Interface For Interacting With The System, A Javascript-Based Inventory System Uses Both Front-End And Back-End Technologies. The Requirements Of The Business And The Expertise Of The Development Team Will Determine The Precise Features And Functionality Of The System.

3.2 Database Structure

The Database Architecture Of A Javascript-Based Inventory System. As Opposed To Javascript Objects Or Arrays, The Data Will Be Saved In A Database. Here Is An Illustration Of How The Database Structure For A Javascript-Based Inventory System Might Appear:

1. Products Table: This Table Houses Data On The Items In The Inventory, Including Product Name, Sku, Description, Price, And Quantity On Hand.

Productid: A Distinctive Identifier For Every Product.

Name: The Item Name

Sku: A Specific Product Identifier Used For Inventory Management.

Description: A Succinct Description Of The Thing

Price: The Price At Which The Good Is Sold.

Quantityinstock: The Quantity Of The Product In Stock At The Time Of Your Query.

Kruidavt Inventory System

Item Name:

Quantity:

Price:

Location:

Submit

Item Name	Quantity	Price	Location	Actions
aloe vera drin	536	1.00	eden	Delete
loft soap	900	2.22	Ash	Delete
dior perfume	6	30.00	wuse abuja	Delete
ultra light bulb	50	10.00	benin	Delete

Figure 3.1 Inventory System

2. Orders Table: Data About Customer Orders, Including Order Date, Customer Name, Shipping Address, Total Cost, And Status, Are Kept In This Table.

Orderid: A Unique Identifier For Each Order

Orderdate: The Day The Purchase Was Made.

Customername: The Name Of The Client Who Made The Purchase

Shippingaddress: The Place Where The Order Will Be Delivered.

Including Taxes And Shipping, The Order's Total Price Is Known As The Ordertotal.

Orderstatus: The Order's Status (Such As Pending, Shipped, Or Cancelled).

Kruidavt Inventory System

Item Name:

Quantity:

Price:

Location:

Item Name	Quantity	Price	Location	Actions
aloe vera drin	536	1.00	eden	<input type="button" value="Delete"/>
loft soap	900	2.22	Ash	<input type="button" value="Delete"/>
dior perfume	6	30.00	wuse abuja	<input type="button" value="Delete"/>
ultra light bulb	50	10.00	benin	<input type="button" value="Delete"/>

Figure 3.2 Order Table

3. Order Items Table: This Table Keeps Track Of Details Like Product Ids, Quantity Ordered, And Prices For The Items In Each Order.

Orderitemid: A Distinguishing Number For Each Order Item

Orderid: A Foreign Key Connecting The Order Item To The Order It Belongs To

Productid: A Foreign Key Connecting The Order Item To The Product It Refers To

Quantityordered: The Quantity Of The Product That Was Ordered.

Cost: The Item's Unit Cost At The Time The Order Was Placed.

4. Users Table: This Table Contains Data On System Users, Including Usernames, Passwords, Email Addresses, And Roles.

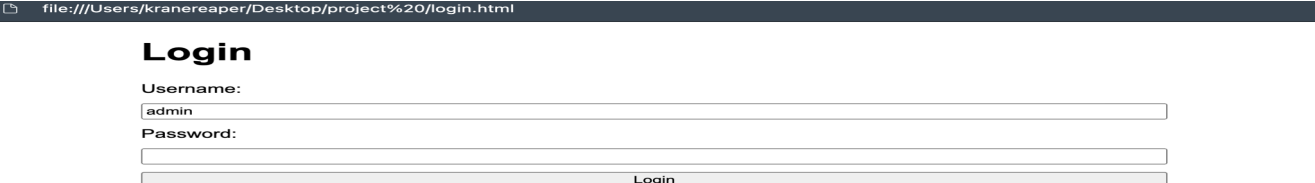
Userid: Each User's Individual Identifier

Username: The Username That A User Enters When Accessing A System

Password: The User's System Login Password, Which Is Encrypted And Kept On File.

Email: The User Account's Associated Email Address

Role: The User's Role (For Example, Administrator Or Regular User).



file:///Users/kranereaper/Desktop/project%20/login.html

Login

Username:
admin

Password:

Login

Figure 3.3 Login Page To Inventory System

5. Reports Table: This Table Contains Details About The Reports The System Generates, Including The Type Of Report, The Date It Was Generated, And The Parameters Used.

Reportid: Each Report's Individual Identification Number

Reporttype: The Classification Of The Report (E.G., Sales Report, Inventory Report).

Dategenerated: The Date The Report Was Generated

Parameters: Any Parameters Used To Generate The Report (E.G., Date Range, Product Category)

6. Suppliers Table: This Table Contains Information About The Product's Suppliers, Such As The Supplier's Name, Contact Information, And Product Id.

Supplierid: A Unique Identifier For Each Supplier.

Suppliername: The Supplier's Name

Contactinformation: The Supplier's Contact Information (For Example, Phone Number, Email Address).

Productid: A Foreign Key That Connects The Supplier To The Products They Provide.

7. Table Of Categories: This Table Contains Information About The Product Categories, Such As The Category Name, Description, And Product Id.

Categoryid: A Unique Identifier For Each Category.

Categoryname: The Category's Name

Description: A Category Description

Productid: A Foreign Key That Connects A Category To The Products Within That Category.

Foreign Keys, Which Are Fields That Reference The Primary Keys Of Other Tables, Can Connect The Tables. For Example, The Orders Table May Contain A Foreign Key That Refers To The Users Table To Indicate Which User Placed The Order. To Indicate Which Product Was Ordered, The Order Items Table May Have A Foreign Key That References The Products Table.

Overall, The Database Structure For An Inventory System Built With Javascript Should Be Designed To Store And Retrieve Data About Products, Orders, Users, And Reports As Efficiently As Possible. The Specific Structure Will Be Determined By The

System's Requirements And The Data That Must Be Stored. Javascript And Server-Side Technologies Such As Node.js, Express, And MongoDB Can Be Used To Access And Manipulate The Database.

3.3 Data Structure Package

The Structure Of A Javascript Software Package Can Vary Depending On The Application And Framework Used. However, Here Is An Example Of A Possible Structure For An Inventory Management Software Package Written In Javascript:

1. Server-Side Files, Such As Node.js Files, Contain Code That Runs On The Server And Handles Client Requests.
2. Client-Side Files: These Files Contain Client-Side Code, Such As Html, Css, And Javascript Files For Creating The User Interface.
3. Model Files: These Files Contain Code That Defines The Inventory Management System's Data Models, Such As Product, Order, And User Models.
4. Controller Files: These Files Contain Code That Handles The Inventory Management System's Business Logic, Such As Creating And Updating Products, Processing Orders, And Generating Reports.
5. View Files: These Files Contain Code For Creating The Inventory Management System's User Interface, Such As Html, Css, And Javascript Files For Creating Web Pages And Forms.
6. Database Management Files: These Files Contain Code That Interacts With The Database, Such As Creating Tables, Inserting And Updating Data, And Executing Queries.
7. Configuration Files: These Files Contain Configuration Settings For The Inventory Management System, Such As Database Connection Details, System Settings, And User Roles And Permissions.
8. Utility Files: These Files Contain Code That Offers The Inventory Management System's Necessary Features, Including Date And Time Functions, Encryption And Decryption Capabilities, And Input Validation Capabilities.

9. Testing Files: These Files Include Code For Unit, Integration, And Acceptance Tests Of The Inventory Management System.

A Software Package For An Inventory Management System Using Javascript Should Have A Modular, Organized, And Simple To Maintain And Update Structure Overall. Additionally, It Ought To Be Built To Be Scalable So That It Can Manage Growing Volumes Of Data And Traffic As The Business Expands. The Framework Or Libraries Being Used, As Well As The Requirements Of The Application, Will Determine The Precise Structure.

CONCLUSIONS

The purpose of this final qualifying work is the development of a software package for an inventory system . Based on the goal and objectives, the structure of this final qualification work was determined, during which the analysis of the activities of the institution, the analysis of algorithms for creating the complex, the analysis of tools, the software implementation for writing, and the result of the work of the software complex were described.

The result of solving the tasks set is the creation of a software package for an Inventory system , which is able to reduce time and human resources.

At the beginning of the final qualification work, the theoretical foundations for the construction of the inventory system were outlined.

Well-known similar developments are considered - programs for an automated inventory system , their functionality is briefly described and illustrations of their interface are given. The prototype system is described with all types of support: organizational, informational, technical and software algorithmic.

At the end of the analytical, a problem was formulated on this object. The existing Inventory system for computer and office Items of the enterprise is formed by information on paper and partially in MS Excel files. The current equipment Inventory system is chaotic, the form and structure of files are not unified. Thus, it is not able to fully cope with the volume of work that exists at the present time.

The problem considered in the final qualification work, within the framework of certain criteria and restrictions, was divided into two particular tasks: to create a database necessary for the operation of the system, and to create a software package for Inventory of goods and services .

In the analytical part, the statement of the problem of building a database and the statement of the problem of building a software package were performed. During the design of the database, all design stages were passed: description of information objects

of the subject area, design of the infological model of the subject area, logical and physical design of the database. Javascript was chosen as the DBMS.

For development, the concept of a Windows application based on a client-server architecture was chosen. The HTML was chosen as the development environment.

As a result, a software package was created that represents a client application for Inventory system, planning preventive work, based on data from previous preventive work. The program meets the main stated requirements and implements an intuitive interface.

REFERENCES

1. M. Cohen, "Inventory management in the age of big data," *Harvard Business Review*, 2015.
2. W. Muchaendepi, C. Mbohwa, T. Hamandishe and J. Kanyepe, "Inventory Management and Performance of SMEs in the Manufacturing Sector of Harare," *Procedia Manufacturing*, vol. 33, pp. 454-461, 2019
3. H. Inegbedion, S. Eze, A. Asaleye and A. Lawal, "Inventory Management and Organisational Efficiency," *The Journal of Social Sciences Research*, vol. 5, no. 3, pp. 756-763, 2019.
4. Harrington, Jan L. *Relational database design and implementation*. Morgan Kaufmann, 2016.
5. Chin, Francis Y., and Gultekin Ozsoyoglu. "Statistical database design." *ACM Transactions on Database Systems (TODS)* 6.1 (1981): 113-139.
6. Bagui, Sikha, and Richard Earp. *Database design using entity-relationship diagrams*. Crc Press, 2011.
7. Lightstone, Sam S., Toby J. Teorey, and Tom Nadeau. *Physical Database Design: the database professional's guide to exploiting indexes, views, storage, and more*. Morgan Kaufmann, 2010.
8. Rishe, Naphtali. *Database design: the semantic modeling approach*. McGraw-Hill, Inc., 1992.
9. Harrington, Jan L. *Relational database design and implementation*. Morgan Kaufmann, 2016.
10. Storey, Veda C. "Relational database design based on the Entity-Relationship model." *Data & knowledge engineering* 7.1 (1991): 47-83.
11. Nijssen, Gerardus Maria, and Terence A. Halpin, eds. *Conceptual Schema and Relational Database Design: a fact oriented approach*. Prentice-Hall, Inc., 1989.

APPENDIX

Using Html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Kruidavt Inventory System</title>
  <link rel="stylesheet" href="styles.css">
  <script src="scripts.js" defer></script>
</head>
<body style="background-color: lightblue;">
  <h1>Kruidavt Inventory System</h1>
  <form id="addItemForm">
    <label for="itemName">Item Name:</label>
    <input type="text" id="itemName" name="itemName" required>
    <label for="itemQuantity">Quantity:</label>
    <input type="number" id="itemQuantity" name="itemQuantity" required>
    <label for="itemPrice">Price:</label>
    <input type="number" name="itemPrice" id="itemPrice" step="0.01" min="0"
required>
    <label for="itemLocation">Location:</label>
    <input type="text" name="itemLocation" id="itemLocation" required>
    <button type="submit">Submit</button>
  </form>
  <table>
    <thead>
      <tr>
        <th>Item Name</th>
        <th>Quantity</th>
        <th>Price</th>
        <th>Location</th>
        <th>Actions</th>

```

```
    </tr>
  </thead>
  <tbody id="inventoryBody"></tbody>
</table>
</body>
</html>
```

Using CSS

```
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  max-width: 800px;
  margin: auto;
}
```

```
form {
  display: flex;
  flex-direction: column;
  gap: 10px;
}
```

```
table {
  width: 100%;
  border-collapse: collapse;
}
```

```
th, td {
  border: 1px solid #ccc;
  padding: 10px;
  text-align: left;
```

```

}

th {
  background-color: lightblue;
}

```

Using JavaScript

```

document.addEventListener("DOMContentLoaded", () => {
  const addItemForm = document.getElementById("addItemForm");
  const inventoryBody = document.getElementById("inventoryBody");

  class KruidavtInventorySystem {
    constructor() {
      this.loadInventoryItems();
    }

    addItem(itemName, itemQuantity, itemPrice, itemLocation) {
      console.log(`Adding item: ${itemName}`);
      const newRow = inventoryBody.insertRow();
      newRow.innerHTML = `
        <td>${itemName}</td>
        <td>${itemQuantity}</td>
        <td>${itemPrice.toFixed(2)}</td>
        <td>${itemLocation}</td>
        <td><button class="deleteBtn">Delete</button></td>
      `;

      newRow.querySelector(".deleteBtn").addEventListener("click", () => {
        newRow.remove();
        this.removeItemFromLocalStorage(itemName);
      });
    }
  }
}

```

```

    this.addItemToLocalStorage(itemName, itemQuantity, itemPrice, itemLocation);
  }

  addItemToLocalStorage(itemName, itemQuantity, itemPrice, itemLocation) {
    const items = this.getItemsFromLocalStorage();
    items[itemName] = { quantity: itemQuantity, price: itemPrice, location:
itemLocation };
    localStorage.setItem("inventoryItems", JSON.stringify(items));

    console.log(`Added item to localStorage: ${JSON.stringify(items)}`);
  }

  removeItemFromLocalStorage(itemName) {
    const items = this.getItemsFromLocalStorage();
    delete items[itemName];
    localStorage.setItem("inventoryItems", JSON.stringify(items));
  }

  loadInventoryItems() {
    const items = this.getItemsFromLocalStorage();
    console.log(`Loading items from localStorage: ${JSON.stringify(items)}`);
    for (const itemName in items) {
      const item = items[itemName];
      this.addItem(itemName, item.quantity, item.price, item.location);
    }
  }

  getItemsFromLocalStorage() {
    const itemsJSON = localStorage.getItem("inventoryItems");
    return itemsJSON ? JSON.parse(itemsJSON) : {};
  }
}

```

```
const inventorySystem = new KruidavtInventorySystem();

addItemForm.addEventListener("submit", (event) => {
  event.preventDefault();
  const itemName = event.target.itemName.value;
  const itemQuantity = event.target.itemQuantity.value;
  const itemPrice = parseFloat(event.target.itemPrice.value);
  const itemLocation = event.target.itemLocation.value;

  // Add item to the table and localStorage
  inventorySystem.addItem(itemName, itemQuantity, itemPrice, itemLocation);

  // Reset the form
  addItemForm.reset();
});
});
```