SOFTWARE ENGINEERING LAB

PROJECT REPORT

On

Motor Part Shop Software

Submitted By:

Suwesh Prasad Sah	(120CS1069)
Awantika Mallick	(120CS1108)
Akash Poudel	(120CS1109)
Prabesh Kumar Sah	(120CS1110)

Submitted To:

Dr. Puneet Kumar Jain

Department of Computer Science and Engineering

Department of Computer Science and Engineering

NATIONAL INSTITUTE OF TECHNOLOGY

ROURKELA

APRIL, 2023

ABSTRACT

Our motor parts shop software is a comprehensive solution designed to streamline the inventory management process, minimize inventory overheads, and sustain selling for about one week. The software is equipped with advanced features that enable the shop owner to efficiently manage the inventory of motor parts, generate orders, and track sales performance.

The software allows the shop owner to maintain different parts in wall-mounted and numbered racks, add, modify, and delete parts as needed, and calculate the threshold value for each item based on the average number of parts sales for one week. The threshold value ensures that items are ordered as soon as the number of items in the inventory reduces below a certain value.

The software system generates the items to be ordered, the amount required, and the address of the vendor supplying the part at the end of each day, based on the threshold value and the current inventory level. The system also calculates the revenue for each day based on the number of parts sold and the selling price of each part.

At the end of the month, the software generates a graph showing the sales for each day of the month. The graph enables the shop owner to analyze the sales trend and make informed decisions about inventory management.

The software has user management features, reporting capabilities, and security features to ensure the confidentiality, integrity, and availability of the data. The software can also be integrated with other systems, such as accounting software and CRM software, to enable seamless data exchange and avoid data duplication.

Overall, our motor parts shop software is a user-friendly, scalable, and customizable solution that empowers the shop owner to efficiently manage the inventory and optimize the business performance.

TABLE OF CONTENTS

List of Figures

1. INTRODUCTION

1.1. Background

As we're closer to a completely digital era, we depend more and more on software-based solutions for our tasks. However, we could not find any relevant software in the market currently for the niche i.e, Motor Parts Shop/Automobile spare parts automation. So, to address the particular problem, we decided to build a new software highly customized and targeted towards the Automobile parts shop.

1.2. Motivation

We wanted to build software targeting the huge market of smaller businesses like Motor Part Shops. The motivation was to make businesses grow by making everything a seamless experience. Every tiresome work related to the business can be made with this within just a few clicks or touches. This would highly impact the positive growth of their business.

1.3. Objectives

1. To automate the tasks for a Motor Parts Shop.
2. To help the shops grow in revenue by providing a seamless experience.

1.4. Problem statement

The motor parts shop deals with large number of motor parts of various manufacturers and various vehicle types. Some of the motor parts are very small and some are of reasonably large size. The shop owner maintains different parts in wall mounted and numbered racks. The shop owner maintains as few inventory for each item as reasonable, to reduce inventory overheads after being inspired by the just-in-time (JIT) philosophy. Thus, one important problem the shop owner faces is to be able to order items as soon as the number of items in the inventory reduces below a threshold value. The shop owner wants to maintain parts to be able to sustain selling for about one week. To calculate the threshold value for each item, the software must be able to calculate the average number of parts sales for one week for each part. At the end of each day, the shop owner would request the computer to generate the items to be ordered. The computer should print out the part number, the amount required and the address of the vendor supplying the part. The computer should also generate the revenue for each day and at the end of the month, the computer should generate a graph showing the sales for each day of the month.

1.5. Scope of Project

The scope of this project is to develop a software system for a motor parts shop that will enable the shop owner to manage the inventory efficiently, minimize inventory overheads, and sustain selling for about one week. The software should be able to calculate the average number of parts sales for one week for each item based on historical sales data. The system should also be able to generate the items to be ordered, the amount required, and the address of the vendor supplying the part at the end of each day.

2. REQUIREMENT ANALYSIS

2.1. Feasibility Study:

The feasibility study for the motor parts shop software project involves analyzing various aspects of the project to determine its practicality, viability, and profitability. The feasibility study includes the following:

2.1.1. Technical Feasibility:

The proposed software project is technically feasible as it requires standard hardware, software, and networking components. The development team has the required expertise and experience to develop the software project, and the necessary technology is readily available.

2.1.2. Economic Feasibility:

The proposed software project is economically feasible as it is expected to generate significant revenue and reduce inventory overheads. The development, maintenance, and operational costs of the project are within the budget, and the payback period is estimated to be less than one year.

2.1.3. Operational Feasibility:

The proposed software project is operationally feasible as it meets the needs and expectations of the stakeholders. The software project is designed to streamline the inventory management process, minimize inventory overheads, and sustain selling for about one week. The software is user-friendly, scalable, and customizable, and it integrates seamlessly with other systems, such as accounting software and CRM software.

In conclusion, the feasibility study indicates that the motor parts shop software project is practical, viable, and profitable. The project team can proceed with the project with confidence, knowing that it meets the technical, economic, and operational requirements of the stakeholders.

2.2. Selection of Process Models:

The software life cycle process model is a framework that outlines the various stages involved in the development of a software application. So, choosing a life cycle process model is the stepping stone to the development of a software product.
For this project, we have chosen the RAD model among all the models.

2.2.1. RAD Model:

The rapid application development also known as rapid prototyping model aims at decreasing the time taken and cost incurred to develop software systems. This software development model facilitates accommodating change requests as early as possible before large investments have been made in development. An important underlying principle of RAD is "Make only short-term plans and make heavy reuse of existing code."

2.2.2. Why we chose RAD?

RAD is the most suitable SDLC model because of the following reasons:

- The software to be developed is a customized product for the owner and only a few customers to use at a time.
- This product does not critically require high performance and reliability.
- The system can be split into several independent modules.
- The project schedule is highly constrained.
- RAD facilitates faster development by testing the core modules thoroughly using specialized tools.
- This reduces the chances of error in the final software.
- RAD uses reusable components, i.e., reusable code reducing the effort required for testing.
- This leads to lower development and maintenance costs.

2.2.3. RAD methodology in the development of this product

- The plans are made for one increment at a time.
- Each iteration enhances the implemented functionality of the software by a little.
- During each iteration, a quick and dirty prototype style software for some selected functionality is developed.
- The customer evaluates the prototype and gives his/her feedback.
- The prototype is then refined based on customer feedback.

2.3. Introduction for Software Requirement Specification

2.3.1. Purpose

The purpose of the document is to serve as a guide to designers, developers, and testers who are responsible for the engineering of the Motor Part Shop project. It should give the engineers all of the information necessary to design, develop and test the software.

2.3.2. Intended Audience and Reading Suggestions

This SRS is for developers, project managers, users, and testers. Further, the discussion will provide all the internal, external, functional, and also non-functional information about motor parts shop software (MPSS).

2.3.3. Project Scope

A small automobile spare parts shop sells spare parts for vehicles of several models. Also, each part is typically manufactured by several small industries. To streamline the sales and supply ordering, the shop owner has asked us to develop the following motor part shop software. The motor part shop deals with large no. of motor parts from various manufacturers and various vehicle types. Some of the motor parts are very small, and some are very large. The owner maintains different parts in wall-mounted and numbered racks. The shop owner maintains as little inventory for each item as reasonable, to reduce inventory overheads after being inspired by the"just in time (JIT) philosophy". Now to facilitate the sales and stocking up the motor parts, we are required to develop the required software.

2.3.4. Environment Characterstics

The software is compatible with Windows version 7 and above for greater flexibility across all systems. The hardware requirements involve RAM greater than 512 MB, with Pentium core and above. The hard disk capacity needs to exceed 20 GB for this software to function smoothly.

2.4. Overall Description

2.4.1. Product Perspective

The Motor Part Shop Software will be a newly developed and self-contained product. It will help in restocking motor parts for the intended owner's shop as well as it will show the required sales statistics to the owner. The Software can be modified in further versions to add complex functionalities.

2.4.2. Product Features

The Product Feature includes:

- Login Authentication.
- Search for a motor part
- Checking current stock of motor parts in the store
- Setting the threshold for a motor part
- Billing a customer
- Checking the sales statistics for the shop
- Getting the list of parts that needs to be restocked by the end of the day
- Updating the database when new stocks are added or when new parts are bought for the shop.
- Adding a new motor part to the inventory

2.4.3. User Classes and Characteristics

The various user classes that are expected to use this software as follows:

- Owner: Can use the software the look into the sales statistics and the current stock of the motor parts.
- Shopkeeper: Can use the software for billing purposes.
- Salesman: Can update the current stock of the motor parts after getting the list from the software.

2.4.4. Operating Environment

The software is compatible with the operating system windows version 7 and above and MAC OS. The database used is MySQL. Python is used for the back end, and Python's GUI Library Tkinter is used for the front end of the software. The hardware requirements involve RAM greater than 512 MB, with Pentium core and above. The hard disc capacity needs to exceed 20 GB for this software to function smoothly.

2.4.5. Design and Implementation Constraints

The information on all the motor parts must be stored in a database that is accessible by the motor part shop software. No internet is required for the functioning of the motor part shop software, so the motor part shop software should be able to run all 24 hours a day. The billing system is connected to the motor part shop software, and the database used by the billing system must be compatible with the interface of the motor part shop software. The users must have their correct usernames and passwords to enter into the motor part shop software.

2.4.6. User Documentation

A user Manual of the motor part shop software would be provided at the time of delivery of the software to the required authority, which can be used to solve any troubleshooting issues regarding this software. If the user manual does not help, the developers are always ready to help with any troubleshoot who can be contacted by the number given in the manual.

2.5. Functional Requirements

2.5.1. Login Authentication

This is the first functionality that the user encounters when it uses the software. There are three classes of users, which include owner, shop keepers and the sales executive. Users are required to enter their username and password to login into the software.

- Input: Username and Password.
- Authentication: Username and Password are matched in the database.
- Output: A successful message or failure message is displayed, once the credentials are checked.

2.5.2. Search for a motor part

This feature can be used to search for a motor part that is present in the inventory. This feature returns the location of the motor part and the number of parts left in the inventory. This feature can be accessed by the owner, the shop keeper and the sales executive

- Input: Motor Part ID
- Output: The rack location where the motor part is stored, and the number of parts stored there. If no parts are present, a warning message is to be displayed.

2.5.3. Checking current stock of motor parts in the store

This feature is kept to check the current stock of any motor part within the store. A list of all the existing motor parts is displayed when the user wishes to check the stock showing the part's number, part's name and the number of items in the shop currently

- Input: User asks for the current stock by clicking on the ”Check current stock” button.
- Process: The current stock is fetched from the database in ascending order of the part ID.
- Output: he current stock is displayed in a list showing the part ID, part name and the number of parts in the inventory currently.

2.5.4. Setting the Threshold for a motor part

This functionality is required to calculate the threshold of each motor part based on the number of parts sold previously so that the shop does not run out of the specified part. The average of number of parts previously sold is calculated to get the threshold.

- Input: The stock that is sold in recent weeks.
- Process: Average of the stocks is calculated for each part.
- Output: The threshold for each motor part, if the current stock falls below the threshold for a motor part, the motor part is restocked.

2.5.5. Billing a Customer

This feature is used to bill a customer when it wishes to buy some required parts from the motor part shop. After the billing is done, the appropriate change is done in the database, which is reflected over the sales statistics.

- Input: Parts bought by user.
- Output: The stock is reduced in the database

2.5.6. Checking the sales statistics for the shop

The functionality has got two parts, as follows;

- The owner can check the sale of motor parts from the shop in the current day. This includes the parts that were sold beginning from the day itself. The user can also check the revenue of the day with the help of this feature.
  - Input: Owner asks for sale of motor parts of the current day
  - Output: Revenue for the day is calculated and displayed along with the sold parts details.
- The owner can check the sales graph which shows the daily sales of the shop over a period of a month. This includes all the parts in the shop. The graph would also show the revenue generated every day in the current month.

  - Input: Owner asks for sale of motor parts of the current month.
  - Output: A graph is displayed which shows the sale of motor parts for each day of the month.

2.5.7. Getting the list of parts that needs to be restocked

The user can get a list of all the motor parts that need to be restocked using the software. The software compares the current stock of an item with the average number of sales of the item over a week. If the current stock is low, the user needs to restock the motor part to carry on the functioning of the shop in a smooth manner.

- Input: Sales executive wants a list of parts that need to be restocked
- Output: A list is displayed showing the ID, name, vendor address and the number of the motor parts that need to be restocked.

2.5.8. Updating the Current Stock

The user can increase the existing number of a motor part after it has been restocked using the software. The change would be reflected in the database, from which the upcoming purchases would be carried on.

- Input:The sales executive adds the number of parts bought for the shop against the ID of the motor part.
- Output: The number of motor parts is increased in the database and is reflected while checking the current stock.

2.5.9. Adding a new motor part to the inventory

This functionality enables the sales executive to add a new motor part into the inventory. To add a new part, the sales executive needs to enter the part ID, part name, the address of the vendor who currently sells the motor part. The initial count for a new part is set to 0, which can be incremented accordingly by the sales executive.

- Input: Sales Executive adds the id of the new part, name of the part, address of the vendor from which the part is bought along with the number of the parts.

- Output: New part is added in the database which can be checked by the current stock.

2.6. External Interface Requirements

2.6.1. User Interfaces

When the user will open the software, a welcome page asking the user id and password will be displayed. The software would entertain three categories of users, as it has been mentioned in the user classes. Upon logging in, the software will display four buttons along with a billing button and an exit button. Upon the clicking the exit button, the current user will be logged out and the login page will be displayed again. The billing button will direct the user to the billing page. The four functionalities include, Stocks, Sales, Requires Parts, Add parts. The stocks button will show the list of current stock of motor parts. The sales button will display the sales statistics of the shop. The requires parts button will display the motor parts that need to be restocked by the end of the day. The add parts page will help in adding the recent number of restocked motor parts. The UI is kept simple and understandable for the user, so that it can work with it without hassle.

2.6.2. Hardware Interfaces

Since the software is supposed to be run on a single system, we do not require cloud based hosting solution here. If more than one system needs to be connected, cloud based solution is required to store the database.

2.6.3. Software Interfaces

A firewall will be used with the server to prevent unauthorized access to the system

2.6.4. Communication Interfaces

The software is to be kept connected to the internet for continuous use in multiple systems.

2.7. Other Non-Functional Requirements

2.7.1. Maintainability Requirements

The software must be robust enough to require as lesser maintenance as possible. Given a glitch in the software, the administrators must be capable enough to sort out the bug quickly to prevent delay in the shop's functionality.

2.7.2. Portability Requirements

The software should be able to be deployed in any machine.

2.7.3. Performance Requirements

The software must be developed using an object oriented model. The performance of every existing module in this software must be robust. This software should be able to run on various operating systems steadily as it as been specified before. Overall the performance of the software must be reliable and the data kept must be safe in case of a power failure.

2.7.4. Safety Requirements

Passwords must be kept different from the student id's for safety purpose. A mail must be provided for emergency queries regarding the software, so that the software can be used without concerns. The mail must be replied by the admins of the software for quick responses.

2.7.5. Security Requirements

Passwords must be stored in hashed format with some salt to prevent any data leak. The authentication used in the login page should work without fault. And finally the whole software is completely secured from outside accessing.

3. SYSTEM ARCHITECTURE AND METHODOLOGY

The whole system architecture is divided into various parts. We have represented the design of the system in following UML diagrams:

3.1. Use Case Diagram

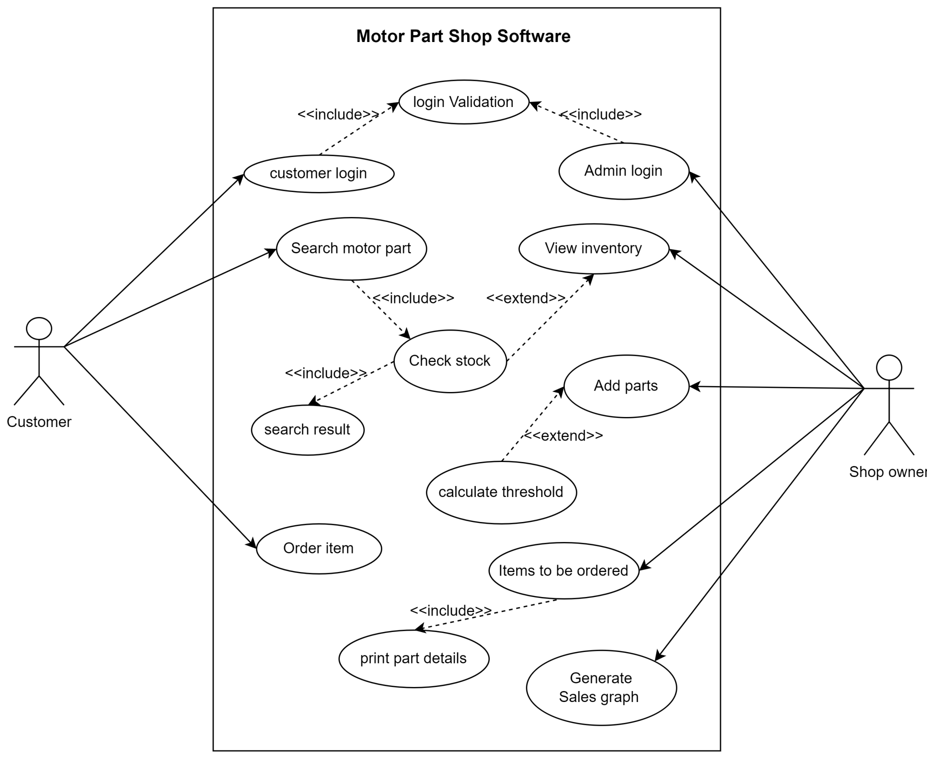


Figure: Use Case Diagram

3.2. Activity Diagram

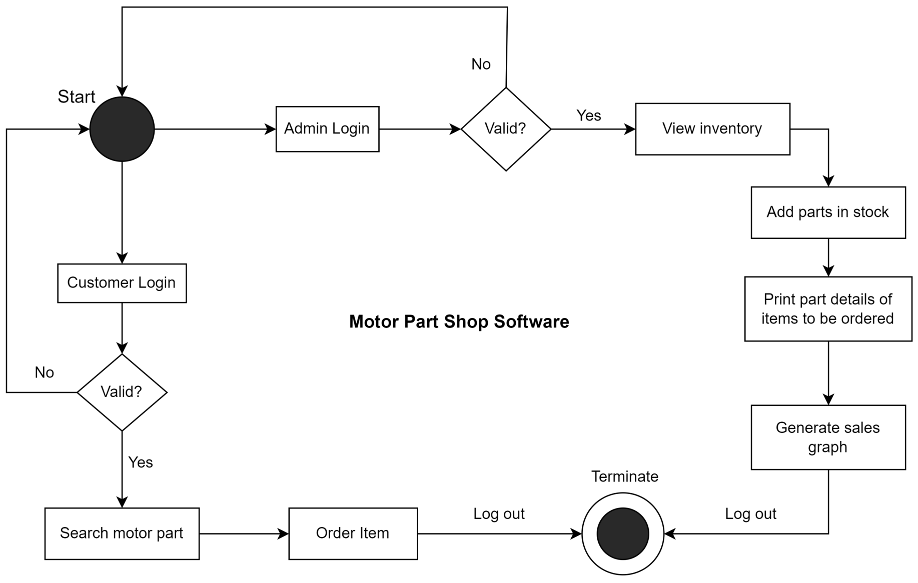


Figure: Activity Diagram

3.3. Class Diagram

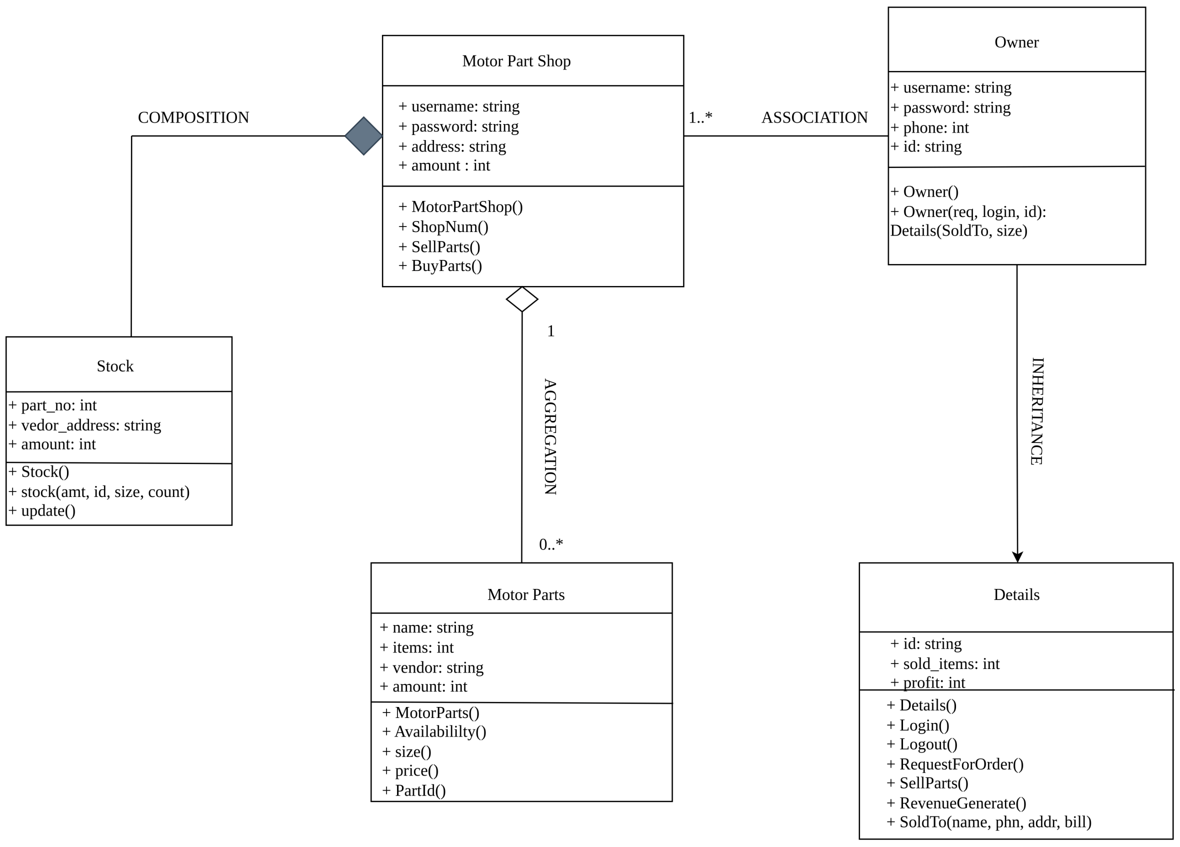


Figure: Class Diagram

3.4. Object Diagram

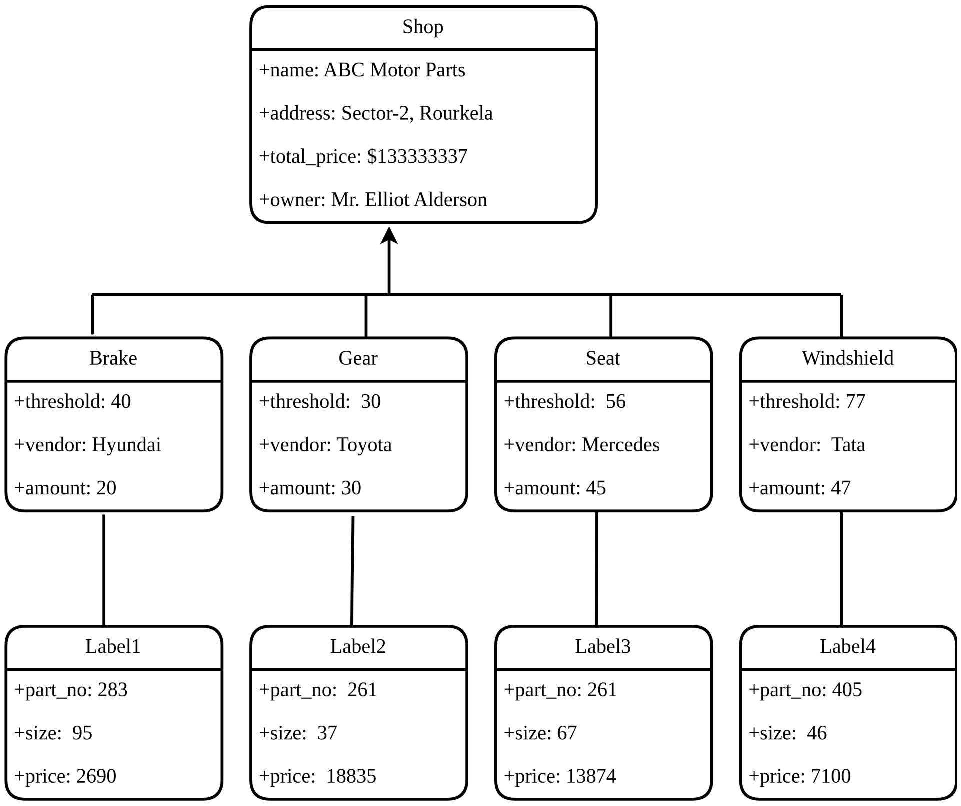


Figure: Object Diagram for Motor Part Shop Software

3.5. Sequence Diagram

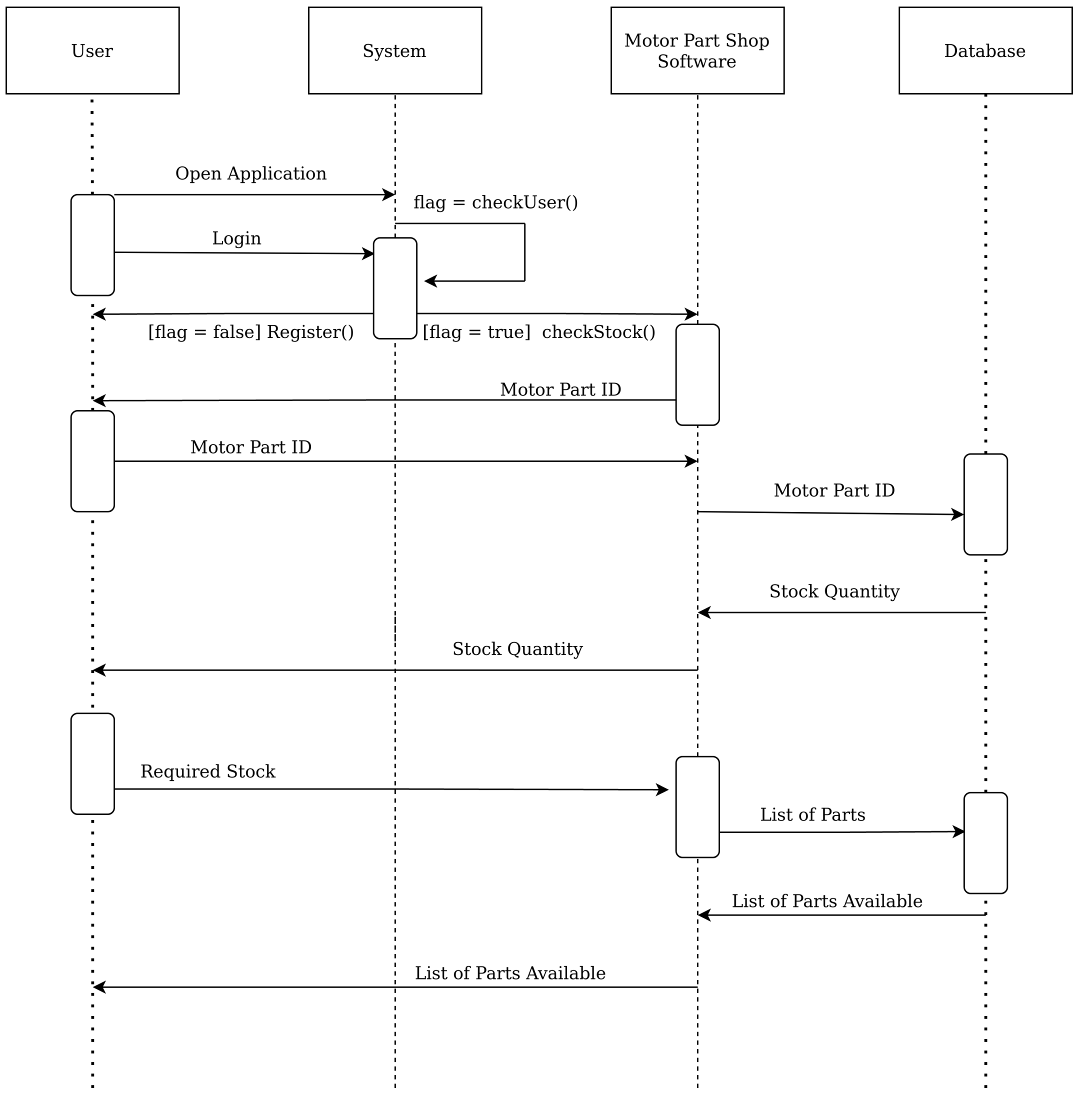


Fig: Sequence Diagram

3.6. Component Diagram

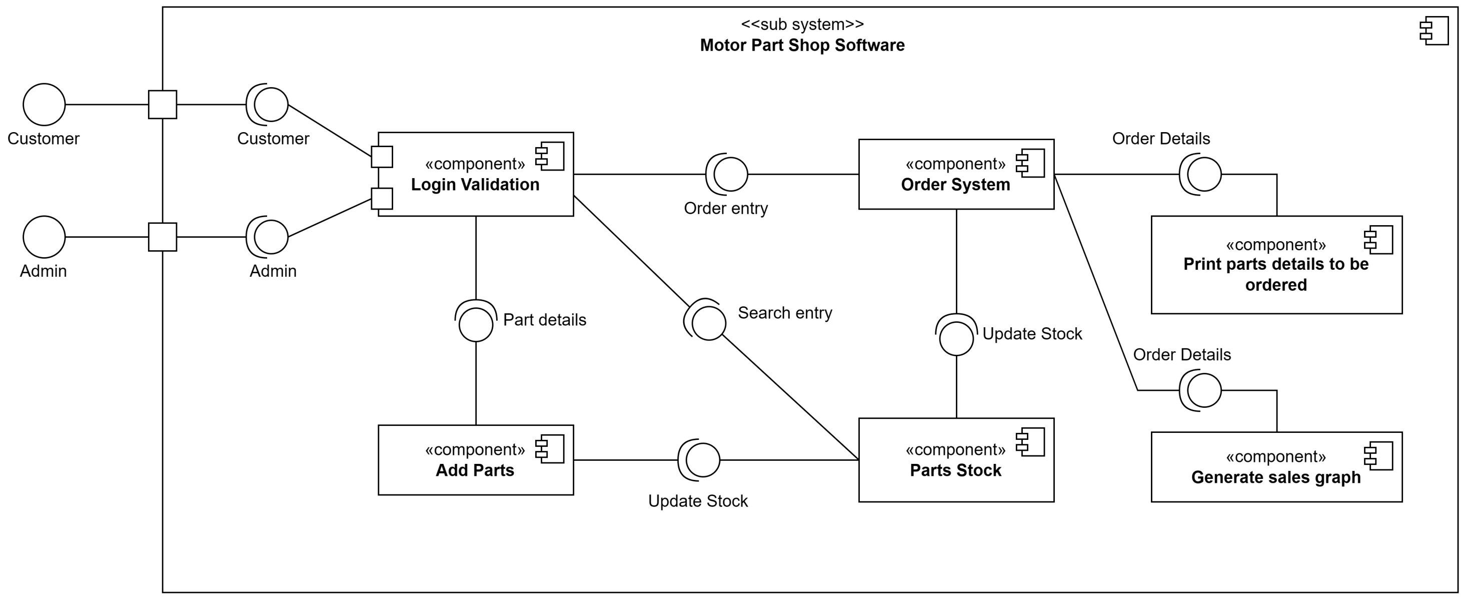
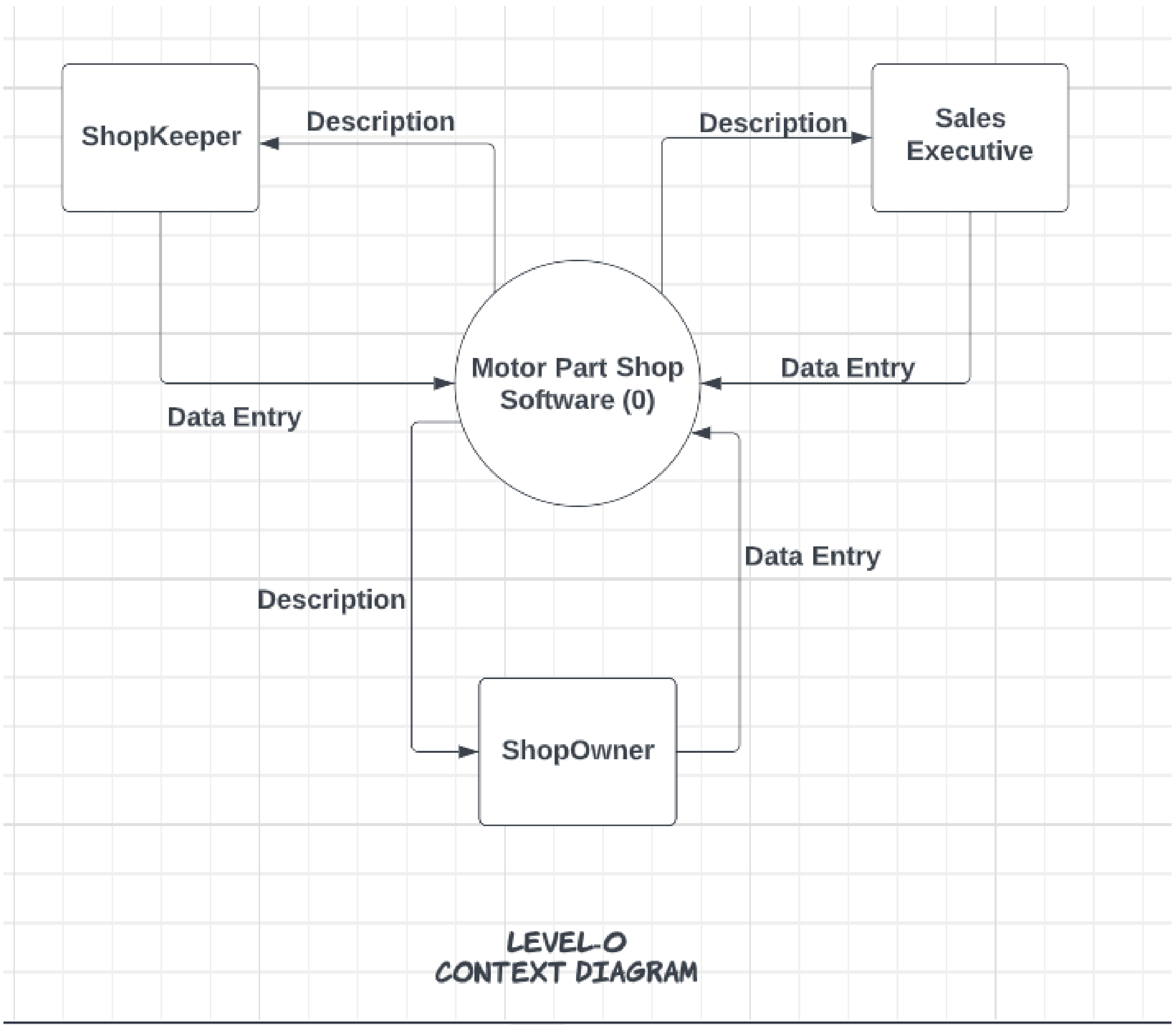
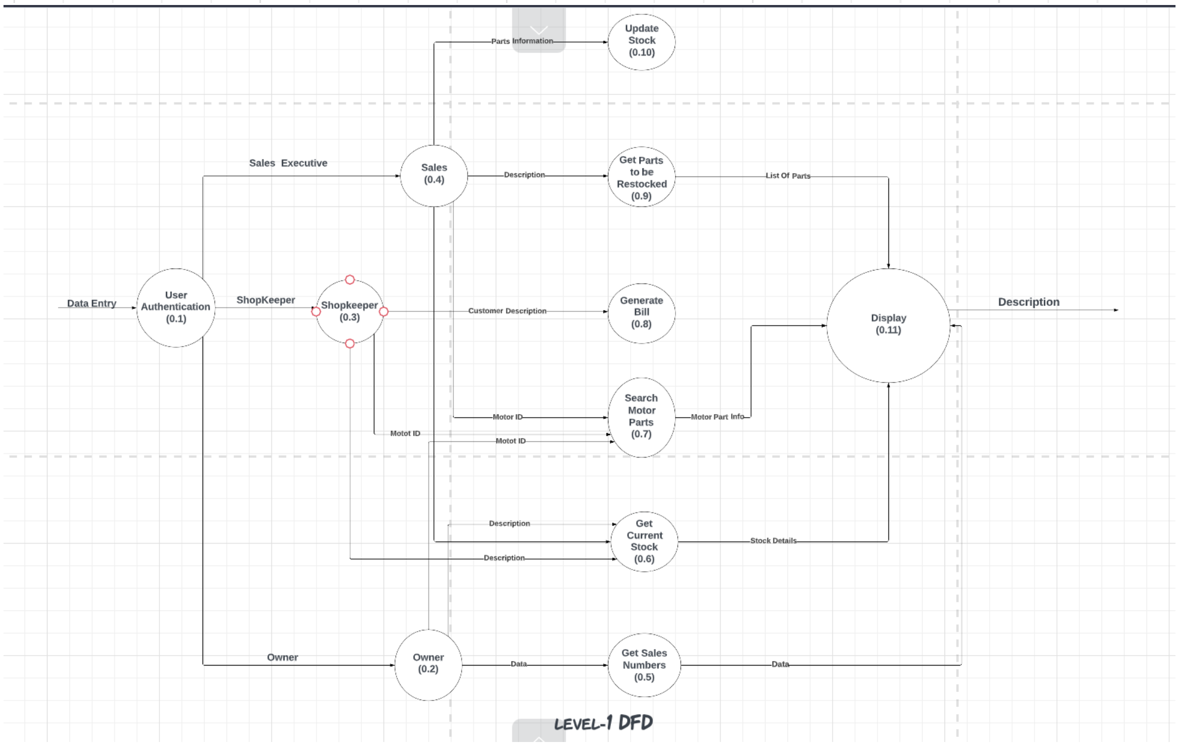


Figure: Component Diagram

3.7. Data Flow Diagram





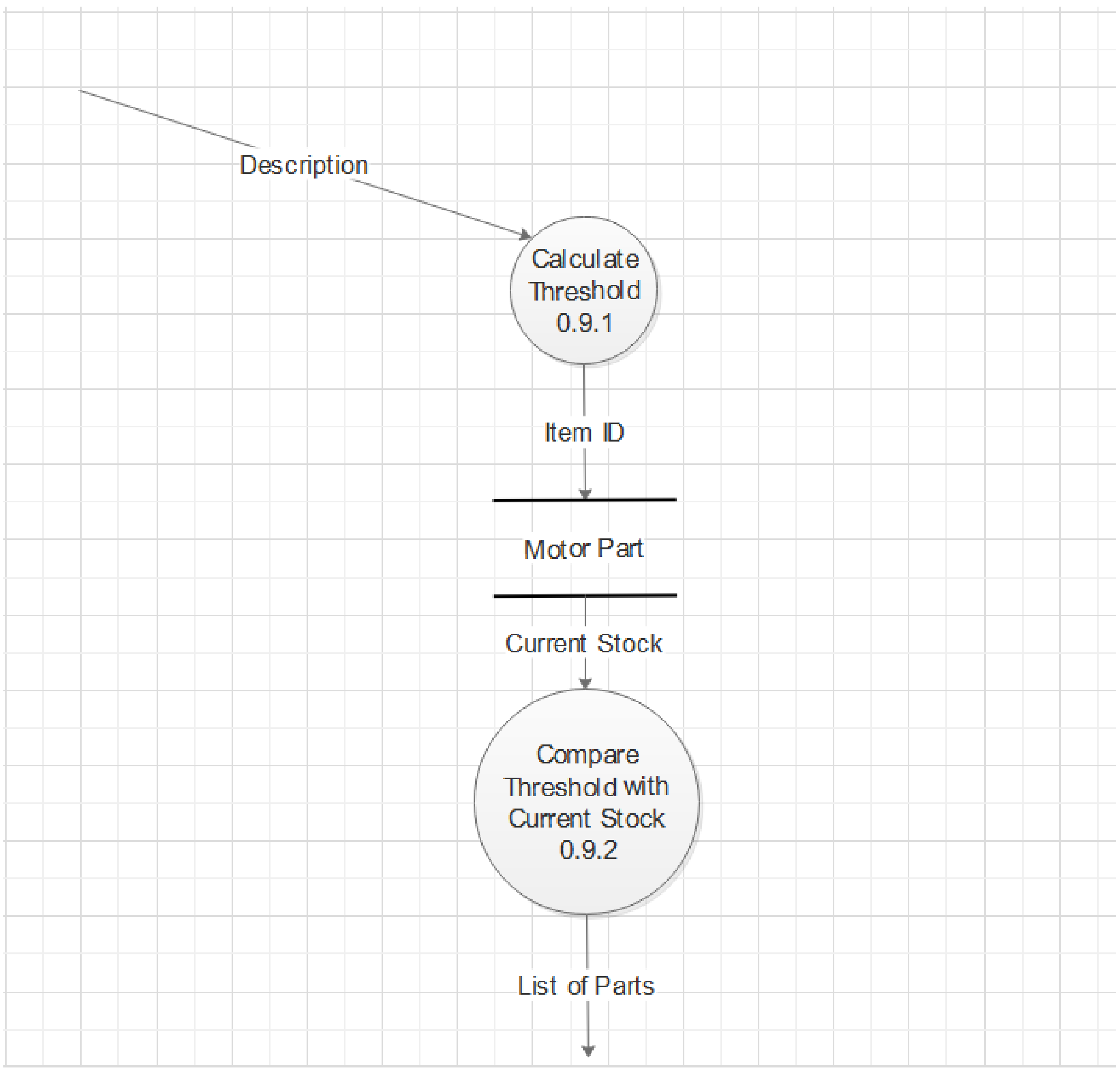
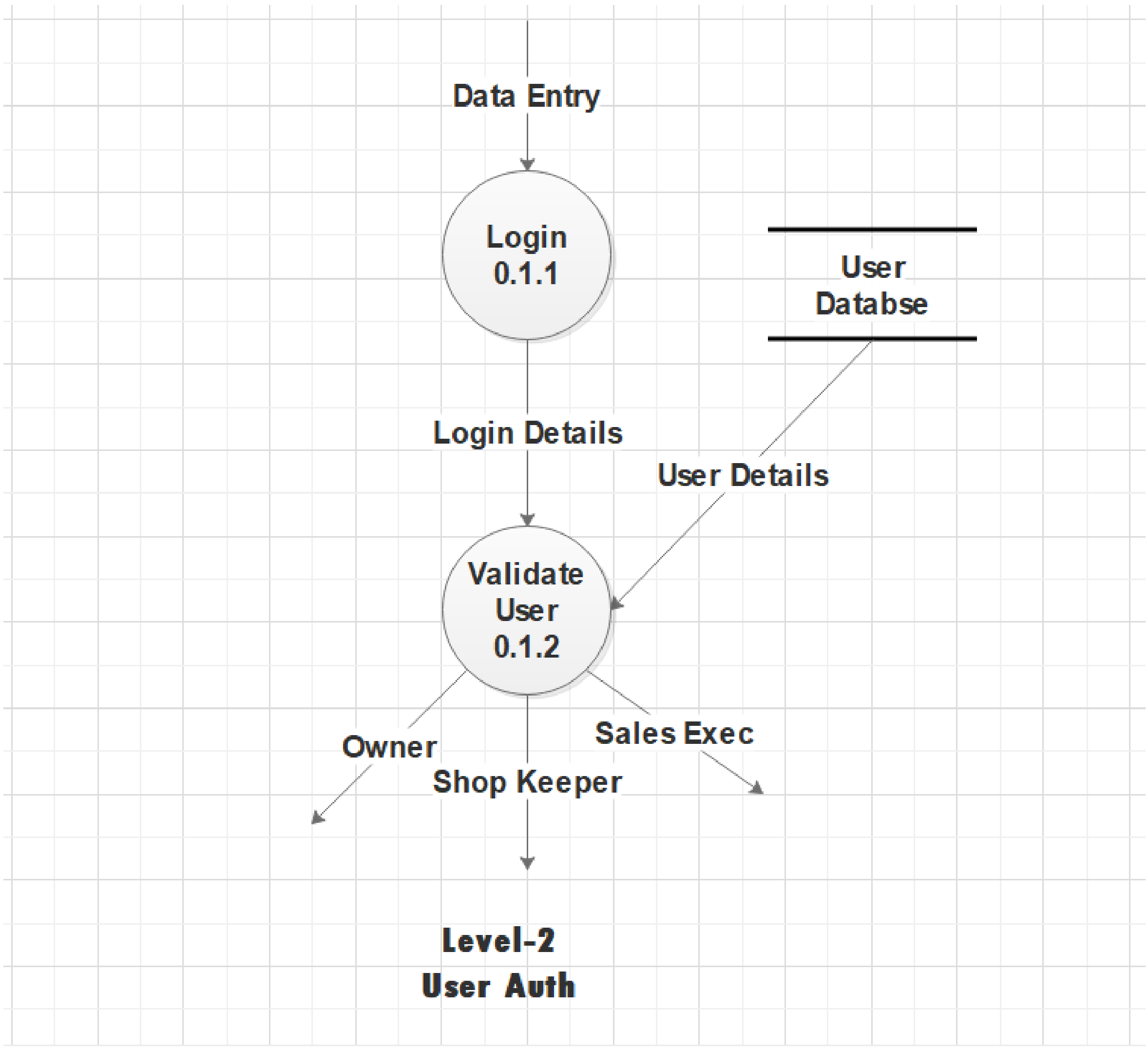

Description
Calculate Threshold 0.9.1
Item ID
Motor Part
Current Stock
Compare Threshold with Current Stock 0.9.2
List of Parts

Figure: Level 2 Data Flow Diagram for Restocking Parts



Level-2
User Auth

3.8. Statechart Diagram

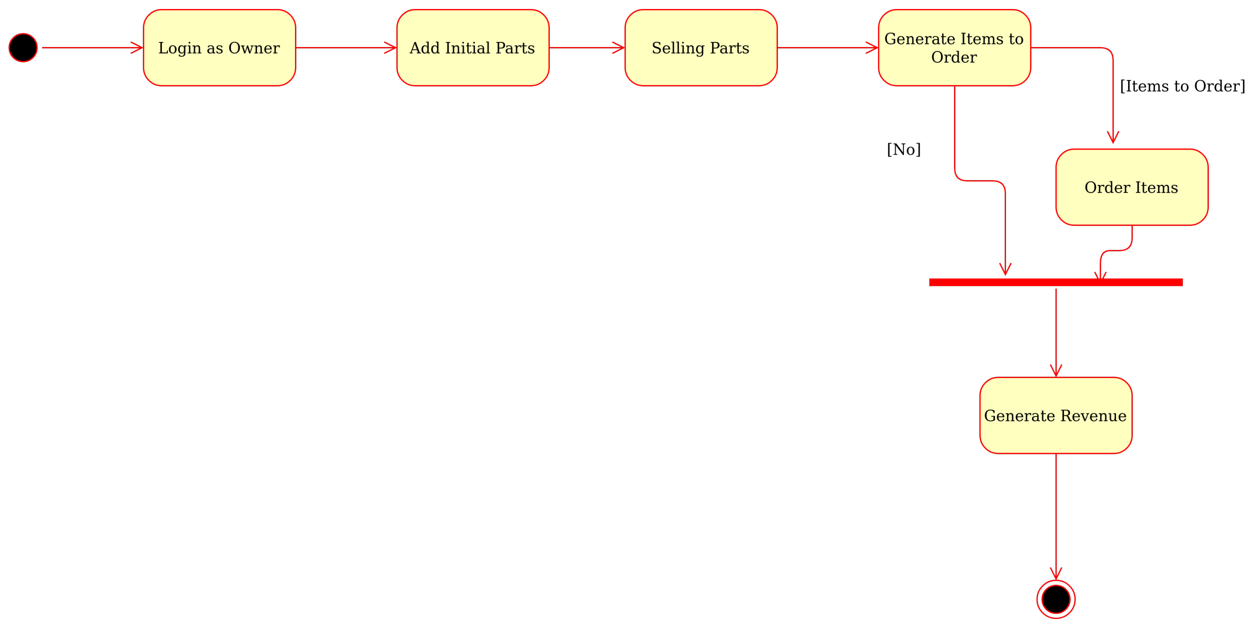


Figure: State Chart Diagram

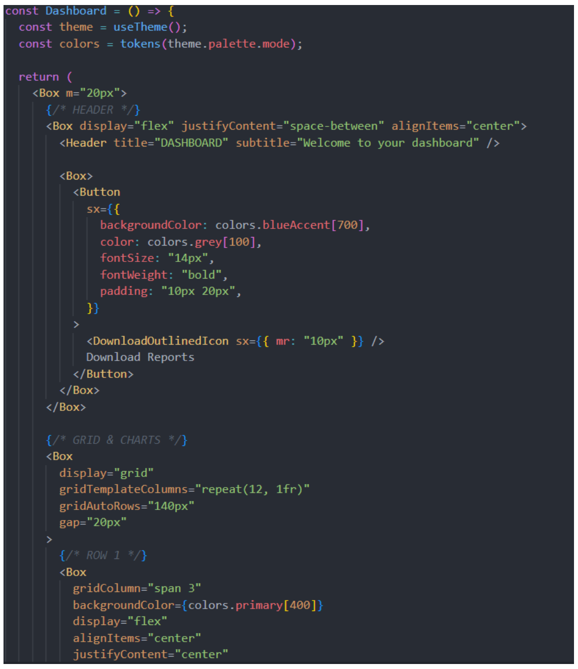
4. IMPLEMENTATION DETAILS

4.1. Introduction

This report describes the hardware and implementation details of a project that uses the Javascript language and Reactjs framework to create a front-end web application. The project uses the Material UI library for the user interface and MongoDB as the database to store collected sample data from the internet.

4.2. Implementation Details

The frontend web application was implemented using Javascript and the Reactjs framework. The application uses the Material UI library for the user interface, which provides pre-designed components to make the development process faster and easier. The fundamental principles of Reactjs were used to create a user-friendly and responsive interface. The project mainly focuses on the dashboard, which displays a summary of the project. The dashboard contains information about the admin and customer details, revenue generated in terms of graphs using bar and line charts for a day and month, and customer locations in terms of geographical graphs. The recent transactions and the number of new users and emails sent are also displayed on the dashboard.

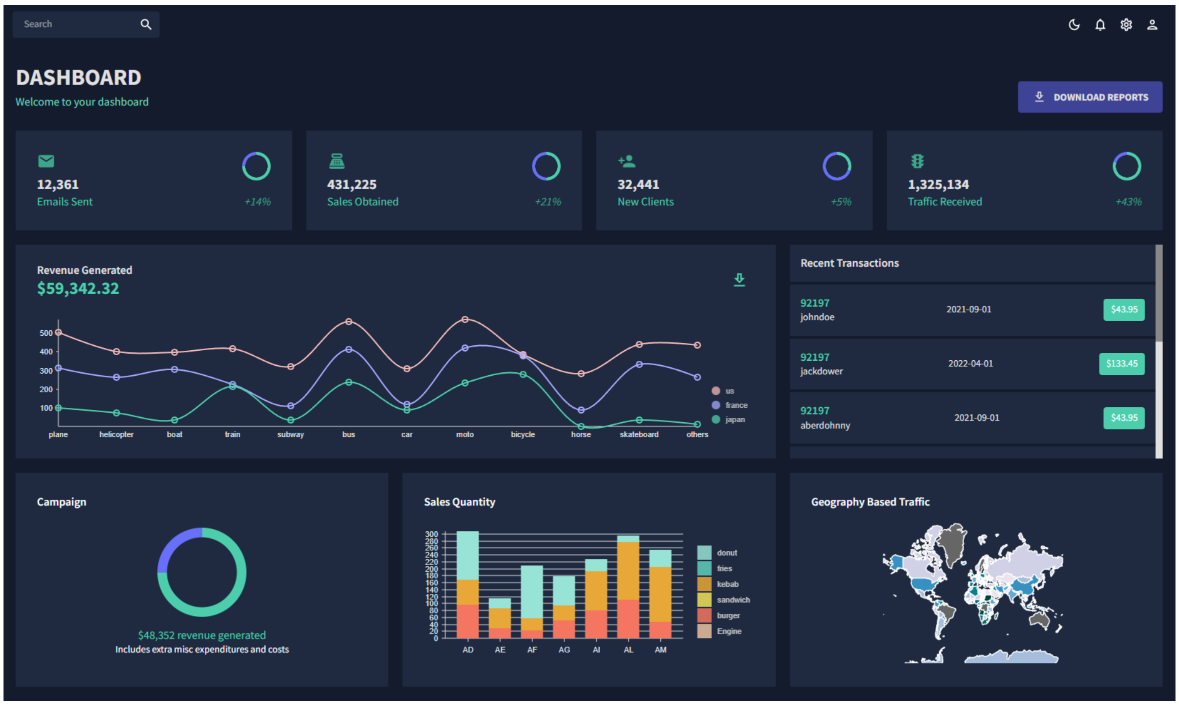


```
const Dashboard = () => {
  const theme = useTheme();
  const colors = tokens(theme.palette.mode);

  return (
    <Box m="20px">
      {/* HEADER */}
      <Box display="flex" justifyContent="space-between" alignItems="center">
        <Header title="DASHBOARD" subtitle="Welcome to your dashboard" />

        <Box>
          <Button
            sx={{
              backgroundColor: colors.blueAccent[700],
              color: colors.grey[100],
              fontSize: "14px",
              fontWeight: "bold",
              padding: "10px 20px",
            }}
          >
            <DownloadOutlinedIcon sx={{ mr: "10px" }} />
            Download Reports
          </Button>
        </Box>
      </Box>

      {/* GRID & CHARTS */}
      <Box
        display="grid"
        gridTemplateColumns="repeat(12, 1fr)"
        gridAutoRows="140px"
        gap="20px"
      >
        {/* ROW 1 */}
        <Box
          gridColumn="span 3"
          backgroundColor={colors.primary[400]}
          display="flex"
          alignItems="center"
          justifyContent="center"
```



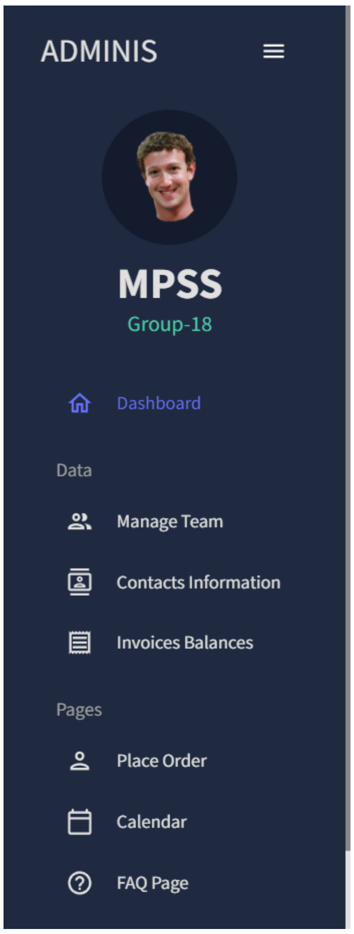
12,361 Emails Sent		+14%
431,225 Sales Obtained		+21%
32,441 New Clients		+5%
1,325,134 Traffic Received		+43%

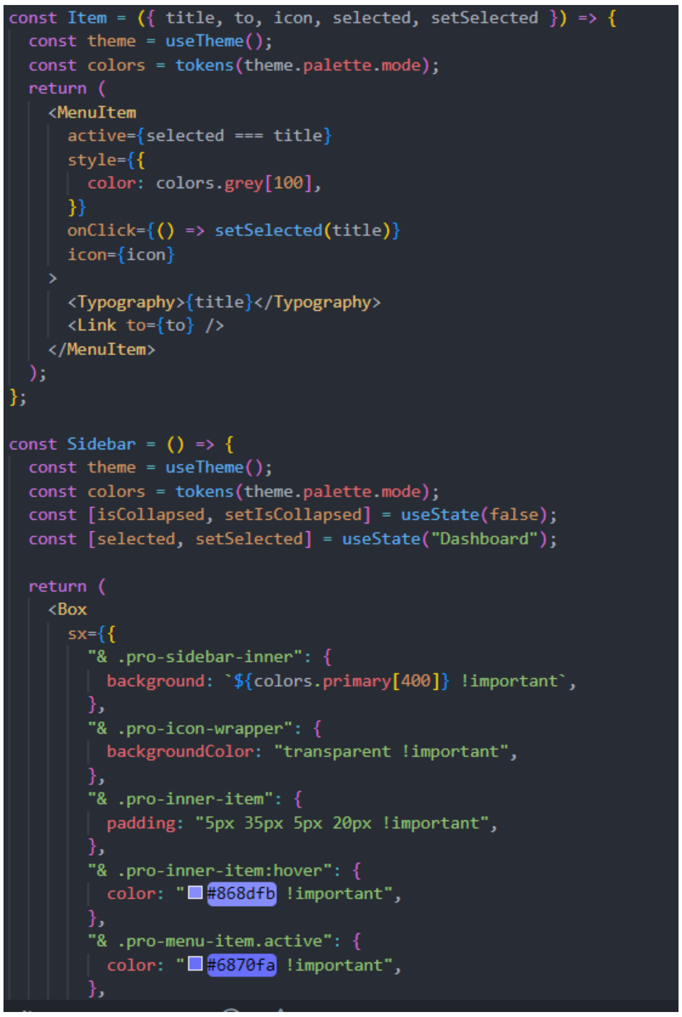
Recent Transactions

92197 johndoe	2021-09-01	$43.95
92197 jackdower	2022-04-01	$133.45
92197 aberdohnny	2021-09-01	$43.95

$48,352 revenue generated
Includes extra misc expenditures and costs

The sidebar of the application has various functionalities, including the ability to place orders. Customers can enter their details and choose the parts they want to buy. The details are stored in the database, and the admin can access them from the backend. Another feature on the sidebar is the number of stocks available in the inventory. The application also provides information on invoices made.





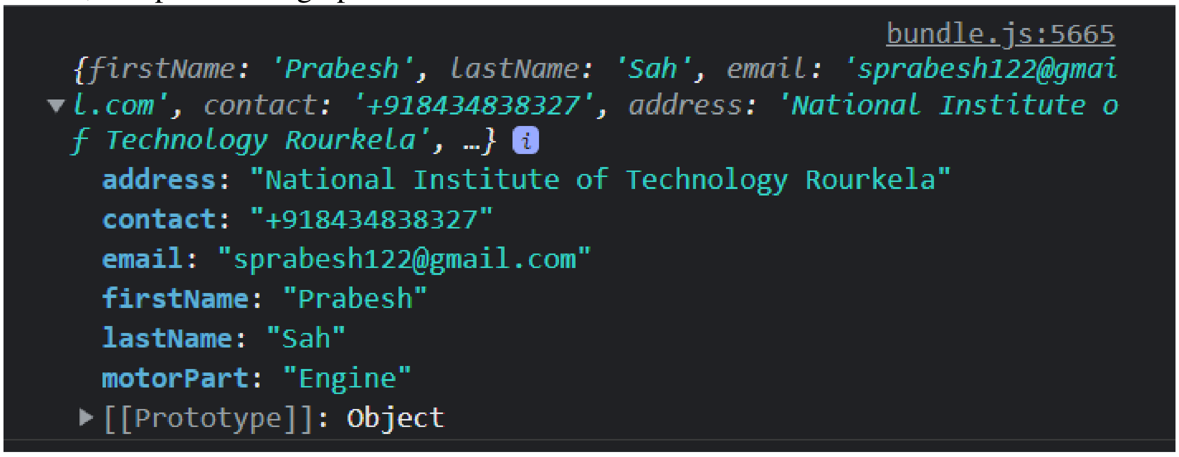
```
const Item = ({ title, to, icon, selected, setSelected }) => {
  const theme = useTheme();
  const colors = tokens(theme.palette.mode);
  return (
    <MenuItem
      active={selected === title}
      style={{
        color: colors.grey[100],
      }}
      onClick={() => setSelected(title)}
      icon={icon}
    >
      <Typography>{title}</Typography>
      <Link to={to} />
    </MenuItem>
  );
};

const Sidebar = () => {
  const theme = useTheme();
  const colors = tokens(theme.palette.mode);
  const [isCollapsed, setIsCollapsed] = useState(false);
  const [selected, setSelected] = useState("Dashboard");

  return (
    <Box
      sx={{
        "& .pro-sidebar-inner": {
          background: `${colors.primary[400]} !important`,
        },
        "& .pro-icon-wrapper": {
          backgroundColor: "transparent !important",
        },
        "& .pro-inner-item": {
          padding: "5px 35px 5px 20px !important",
        },
        "& .pro-inner-item:hover": {
          color: "#868dfb !important",
        },
        "& .pro-menu-item.active": {
          color: "#6870fa !important",
        },
```

4.3. Database

MongoDB was used to store the collected sample data from the internet. MongoDB is a document-oriented database that allows for the storage of unstructured data in JSON format. This makes it easy to store and retrieve data from the database. MongoDB is also scalable, secure, and provides high performance.



```
                                                        bundle.js:5665
 {firstName: 'Prabesh', lastName: 'Sah', email: 'sprabesh122@gmai
▼l.com', contact: '+918434838327', address: 'National Institute o
 f Technology Rourkela', …} i
   address: "National Institute of Technology Rourkela"
   contact: "+918434838327"
   email: "sprabesh122@gmail.com"
   firstName: "Prabesh"
   lastName: "Sah"
   motorPart: "Engine"
 ▶ [[Prototype]]: Object
```

4.4. Conclusion

In conclusion, this project was implemented using the Javascript language and Reactjs framework, with Material UI library for the user interface. MongoDB was used to store the collected data from the internet. The application provides a user-friendly interface that displays a summary of the project on the dashboard and allows customers to place orders. The sidebar of the application provides various functionalities, including the number of stocks available in the inventory and information on invoices made. Overall, the project provides a simple, effective, and efficient solution for managing customer orders and data.

5. RESULTS AND DISCUSSIONS

The motor parts shop software project was successfully completed and implemented within the planned budget and timeline. The software was designed to manage the inventory of motor parts, generate orders for items as soon as the number of items in the inventory reduces below a threshold value, and calculate the average number of parts sales for one week for each part. The software was also designed to generate revenue for each day and at the end of the month and to generate a graph showing the sales for each day of the month.

The software project has improved the inventory management process of the motor parts shop by providing accurate and real-time inventory data, automating the ordering process, and reducing inventory overheads. The shop owner can now order items as soon as the number of items in the inventory reduces below the threshold value and maintain parts to be able to sustain selling for about one week. The software has also helped the shop owner to optimize the inventory level, reduce the risk of stockouts, and improve customer satisfaction.

The software project has also helped the motor parts shop to increase revenue and profitability by providing real-time sales data, enabling the shop owner to make informed decisions about pricing, marketing, and promotions. The software has also helped to reduce the cost of inventory management and improve the efficiency of the shop's operations.

Overall, the motor parts shop software project has been a success, meeting the technical, economic, and operational requirements of the stakeholders. The software has provided significant benefits to the motor parts shop, improving inventory management, increasing revenue, and improving customer satisfaction. The project team has delivered a high-quality software product, meeting the expectations of the stakeholders and demonstrating the value of software development in improving business processes and operations.

6. CONCLUSION

In conclusion, the motor parts shop software project has been a success, meeting the technical, economic, and operational requirements of the stakeholders. The software has significantly improved the inventory management process, reduced inventory overheads, optimized the inventory level, and improved customer satisfaction. The software has also helped to increase revenue and profitability by providing real-time sales data and enabling the shop owner to make informed decisions. The project team has delivered a high-quality software product within the planned budget and timeline, demonstrating the value of software development in improving business processes and operations. Overall, the motor parts shop software project has provided a significant return on investment, and the shop owner can now focus on growing the business and providing better service to customers

A. APPENDIX

It may contains the additional topics or data sheets or reference sheets or even user manual. Project Budget (Detailed Breakdown of Costs), Project Timeline (Gantt chart), Circuit Diagrams (Should be Clear and Legible), PCB Designs (Should be Clear and Legible), Module Specifications (Should be brief - Keep only necessary tables and figures), Details of Dataset can be included here.

References