

**Project Report On**

Inventory Management System

Submitted By

SOURAV

1911985018

# ACKNOWLEDGEMENT

We take this occasion to thank God, almighty for blessing us with his grace and taking our endeavor to a successful culmination. We extend our sincere thanks to our esteemed guides, Mr. Mukesh Sir and Mr. Mananjay Sir, for providing us with the right guidance and advice at the crucial junctures and for showing me the right way. I am overwhelmed in all humbleness and gratefulness to acknowledge my depth to all those who have helped me to put these ideas, well above the level of simplicity and into something concrete.

**PURPOSE**

The Inventory management system is a web-based project that seeks to provide effective management of stock data and data regarding the purchase and sale of the said stock. It consists of a few modules: a stock module, a purchased stock module, a stock sale module, etc. While the stock module is concerned with the management including the addition and deletion of products. The purchased stock module is concerned with keeping a record of all the purchased products. The stock sale module is for clients concerned with the management of products they sold.

**Module Description:**

It’s an online Inventory Management service which is named IMGT. It’s convenient, it gives you the widest choice possible and it can be done sitting in the privacy of one’s cubicle. Today when professionals across the world are spending 10-12 hours at work every day, Online Inventory Management reduces the time and complexity.

Number of Modules

The system after a careful analysis has been identified to be presented with the following modules:

## The modules involved are:

* STOCK MODULE
* PURCHASED STOCK MODULE
* SOLD STOCK MODULE
* BILL MODULE

## Module Description:

It has mainly divided into four modules

1. **STOCK MODULE**

In this module, the admin maintains the stock data, consisting of the number of stocks and their respective amount, needed to buy them. Through this, they can add stock to the existing product list while filling out the necessary details required, such as product name, transaction mode, amount, sold by, etc. The products listed then are stored in the system with other existing ones to be sold or purchased.

1. **PURCHASED STOCK MODULE**

This will hold the details of the stocks purchased by the person at any given time, while also keeping a record of all the transactions made, with their bills and receipts. This module helps you manage all of your products conveniently and safely.

This module helps you manage all the products that you purchased while also keeping a record of the necessary details of the products purchased.

1. **SOLD STOCK MODULE**

This module helps you manage all the products that, you created in the system, that have been sold to other clients with all the required details such as payment mode, payment by, etc.

This module in particular holds the data of all the products sold by one another through this system while also keeping a record of the many important details of the transaction.

1. **BILL MODULE**

This module helps you manage the bills and receipts of all the payments made, transactions made in the system generate bills for every product purchased, and a receipt for every product sold. The bills would show the payment mode and other necessary details required by the user or for the user. Similarly, receipts would show the products sold and their required details.

# Technologies Used:

Various web development technologies are used to create IMGT. These are:

**Front-end technologies:** HTML, Material UI, JS, React JS, Axios, Yup, React Router.

* HTML: (HyperText Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. HTML is the standard markup language for Web pages. With HTML we can create our Website.
* Material UI: Material-UI is simply a library that allows us to import and use different components to create a user interface in our React applications. This saves a significant amount of time since the developers do not need to write everything from scratch. Material UI is highly customizable with which designers can create tons of designs. But it may produce inconsistency among components.
* JS: It is a lightweight, interpreted programming language. It is designed for creating network-centric applications. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform. JavaScript can be used for client-side developments as well as server-side developments. It is used to make web pages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.
* React JS: It is a JavaScript library that combines the speed of JavaScript and uses a new way of rendering web pages, making them highly dynamic and responsive to user input. aces. React is not a framework – it's not even exclusive to the web. It's used with other libraries to render to certain environments. React's primary goal is to minimize the bugs that occur when developers are building UIs. It does this through the use of components — self-contained, logical pieces of code that describe a portion of the user interface. These components can be composed together to create a full UI, and React abstracts away much of the rendering work, leaving you to concentrate on the UI design.
* Axios: It is a javascript used to make HTTP requests to the server to perform certain actions. These requests are promise-based and use the concept of asynchronous javascript.
* Yup: It is a js library used to perform JS object validation.
* React Router: It is a standard library for routing in React. It enables the navigation among views of various components in a React Application, allows changing the browser URL, and keeps the UI in sync with the URL.

**Backend technologies:**  ExpressJS, JWT, NodeJS, MongoDB, SendGrid Mail, BCryptJS, CORS.

* ExpressJS: Express.js, or simply Express, is a back-end web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js. Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.
* JWT: To authenticate a user, a client application must send a JSON Web Token (JWT) in the authorization header of the HTTP request to your backend API. API Gateway validates the token on behalf of your API, so you don't have to add any code in your API to process the authentication. JSON Web Token (JWT) is a standard RFC 7519 for exchanging cryptographically signed JSON data. It is probably the most popular current standard of authorization on the web, especially when it comes to microservices and distributed architecture.
* NodeJS: Node.js is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project. Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant. A Node.js app runs in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.
* MongoDB: It is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server-Side Public License (SSPL). MongoDB, Inc. is an American software company that develops and provides commercial support for the source-available database MongoDB, a NoSQL database that stores data in JSON-like documents with flexible schemas. MongoDB Atlas makes it easy to control access to your database. Your database instances are deployed in a unique Virtual Private Cloud (VPC) to ensure network isolation. Other security features include IP whitelisting or VPC Peering, always-on authentication, encryption at rest and encryption in transit, sophisticated role-based access management, and more.
* SendGrid Mail: SendGrid provides a cloud-based service that assists businesses with email delivery. The service manages various types of email including shipping notifications, friend requests, sign-up confirmations, and email newsletters. It also handles Internet service provider (ISP) monitoring, domain keys, sender policy framework (SPF), and feedback loops. Additionally, the company provides link tracking and open rate reporting. It also allows companies to track email opens, unsubscribes, bounces, and spam reports. Beginning in 2012, the company integrated SMS, voice, and push notification abilities into its service through a partnership with Twilio.
* BCryptJS: bcrypt is a password-hashing function designed by Niels Provos and David Mazières, based on the Blowfish cipher and presented at USENIX in 1999. Besides incorporating a salt to protect against rainbow table attacks, bcrypt is an adaptive function: over time, the iteration count can be increased to make it slower, so it remains resistant to brute-force search attacks even with increasing computation power. Blowfish is notable among block ciphers for its expensive key setup phase. It starts with subkeys in a standard state, then uses this state to perform block encryption using part of the key, and uses the result of that encryption (which is more accurate at hashing) to replace some of the subkeys.
* CORS: Cross-origin resource sharing (CORS) is a mechanism that allows restricted resources on a web page to be requested from another domain outside the domain from which the first resource was served. A web page may freely embed cross-origin images, stylesheets, scripts, iframes, and videos. Certain "cross-domain" requests, notably Ajax requests, are forbidden by default by the same-origin security policy. CORS defines a way in which a browser and server can interact to determine whether it is safe to allow the cross-origin request. It allows for more freedom and functionality than purely same-origin requests but is more secure than simply allowing all cross-origin requests.

# UML Diagrams:

|  |
| --- |
| Stock |
| Stock\_name |
| Stock\_amount |
| Stock\_quantity |
| Stock\_category |
| Stock\_sellerName |
| Stock\_sellerAddress |
| Stock\_sellerPhoneNo |
| Stock\_transactionMode |
| Stock\_transactionID |

## ER Diagram:

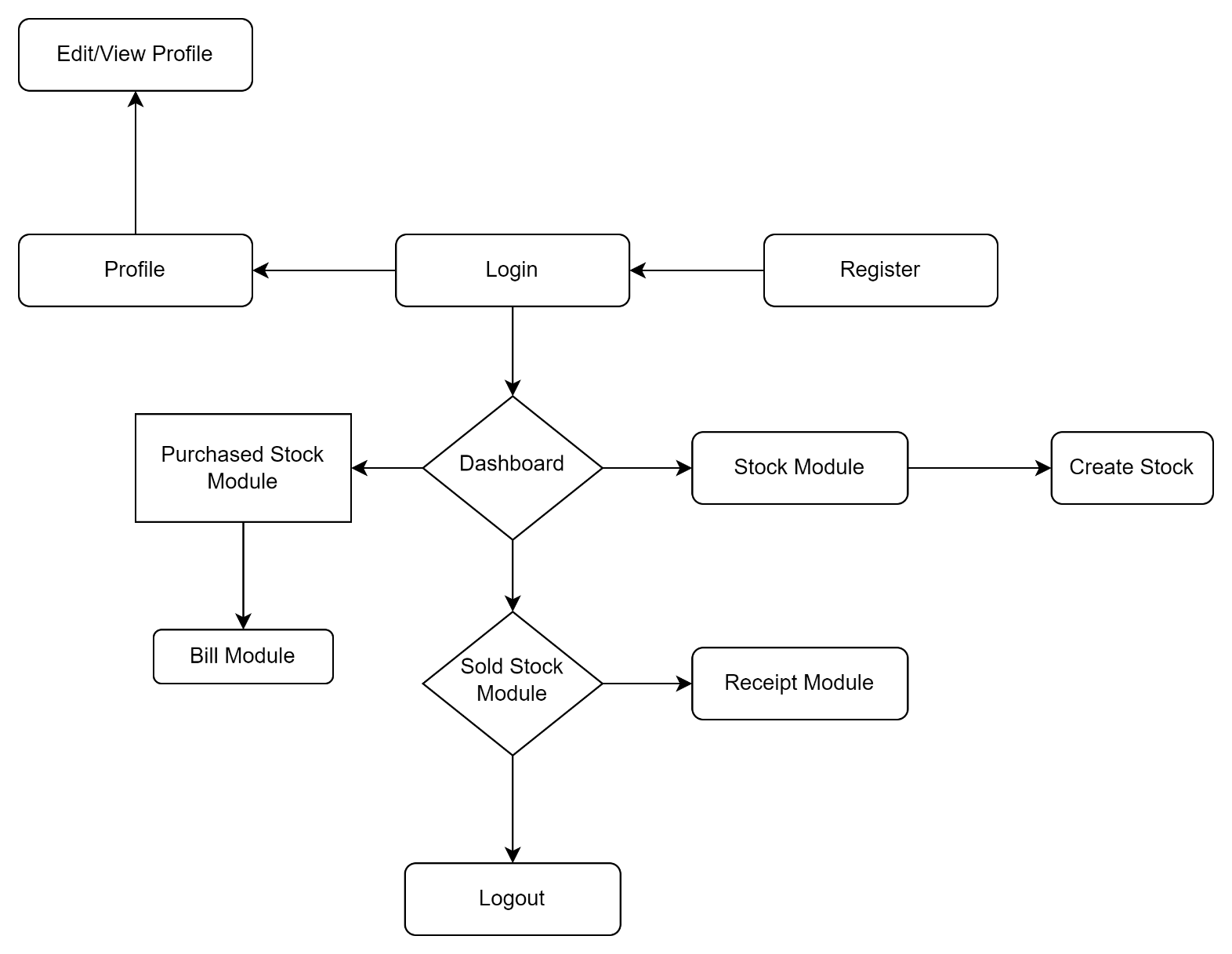
|  |
| --- |
| Purchased Stock |
| Stock\_name |
| Stock\_seller |
| Stock\_category |
| Stock\_purchasedDate |
| Stock\_amount |
| Stock\_quantity |

|  |
| --- |
| Login |
| Admin\_name |
| Admin\_email |
| Admin\_number |

|  |
| --- |
| Sold Stock |
| Stock\_name |
| Stock\_seller |
| Stock\_category |
| Stock\_purchasedDate |
| Stock\_amount |
| Stock\_quantity |

|  |
| --- |
| Bill |
| Bill\_name |
| Bill\_id |
| Bill\_createdAt |
| Bill\_paymentMode |
| Bill\_transactionId |
| Bill\_category |
| Bill\_amount |
| Bill\_quantity |

**Data Flow Diagram:**

****

# SCREENSHOTS

# 

# FUTURE ASPECTS

The proposed system ‘Inventory Management’ can be further developed into a separate, automated system with the following enhancements:

* Addition of chat application between seller and buyers.
* Additional security additions.
* Forgot password feature.
* Data visualization of patient’s transactional history.
* Bill Management System.

Addition of Recent Orders