

**A Mini Project Report on**  
**“Stock Market Management System”**

**BACHELOR OF ENGINEERING**  
**IN**  
**ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING**

**Submitted by : MANOJ G**

# ABSTRACT

A Stock Market Management System is a software designed to automate and streamline the operations of managing stock market transactions, catering to both individual investors and brokers. This system can help to improve the user experience by allowing for efficient management of stock portfolios, executing buy and sell transactions, and tracking stocks seamlessly. The system provides distinct functionalities for regular users and brokers, enhancing the trading and management process.

This project is developed as a web application using Flask, a lightweight and flexible web framework for Python, with MySQL as the database management system. Flask is an effective framework that allows for the development of dynamic and scalable web applications, while MySQL ensures robust data handling and storage capabilities. User authentication is handled securely through Flask-Login, ensuring that only authorized users can access the system, thereby safeguarding user data and preventing unauthorized access.

Regular users can view stock prices, execute transactions, and manage their account details, while brokers have additional privileges such as adding new stocks and editing stock details. The application provides simulated stock prices with predefined fluctuation rules, offering a dynamic user experience. Detailed tracking of transaction history allows users to review past trades and analyze their investment strategies. Overall, this Stock Market Management System can help to improve the efficiency and accuracy of managing stock market transactions.

# Database Management Systems (DBMS): History and Key Concepts

A Database Management System (DBMS) is a software framework designed to handle the storage, manipulation, and retrieval of data in a structured manner. It provides an interface between users and the database, allowing for the efficient management of large volumes of data.

## History and Evolution

The concept of databases dates back to the 1960s when early systems were developed to manage large datasets. The first DBMS was introduced by IBM with the Information Management System (IMS) in 1966, primarily for hierarchical data models. The 1970s marked a significant evolution with the introduction of the relational model by Edgar F. Codd, which revolutionized data management. Codd's work led to the development of relational DBMSs (RDBMS), which organize data into tables and use SQL (Structured Query Language) for querying and manipulation. This model became the standard for database systems due to its simplicity and efficiency in handling complex queries.

In the 1980s and 1990s, the rise of object-oriented programming led to the creation of Object-Oriented DBMSs (OODBMS), which store data in the form of objects rather than tables. This approach aimed to bridge the gap between object-oriented programming languages and database management.

## Components and Functions

A DBMS consists of several key components:

- a. **Data Definition Language (DDL):** Used to define database structures.
- b. **Data Manipulation Language (DML):** For querying and manipulating data.
- c. **Data Control Language (DCL):** Manages access permissions.

A Database Management System (DBMS) provides several crucial functionalities essential for efficient and secure data handling:

- a. **Data Storage, Retrieval, and Update:** Efficiently manages data storage and retrieval.
- b. **User Management:** Controls user access and ensures data security.
- c. **Data Integrity Management:** Maintains data accuracy and consistency.
- d. **Transaction Management:** Ensures that all database transactions are processed reliably and adhere to ACID properties (Atomicity, Consistency, Isolation, Durability).
- e. **Concurrency Control:** Manages simultaneous data access for consistency.

Additional features include backup, recovery and data security, which protect data integrity, manage multiple user access, and ensure privacy. DBMSs are essential for modern computing, supporting various applications and evolving for advanced data management and analysis.

# Introduction

The stock market is a pivotal component of the global financial system, serving as a platform where shares of publicly held companies are issued, bought, and sold. It plays a crucial role in the economy by enabling businesses to raise capital and investors to earn returns on their investments. The stock market operates on the principles of supply and demand, with stock prices fluctuating based on the performance of companies, economic conditions, and various other factors. Investors buy shares at a price they believe reflects the company's value and sell them at a higher price to make a profit. The market's efficiency in aggregating information and reflecting it in stock prices makes it an essential tool for economic growth and wealth generation.

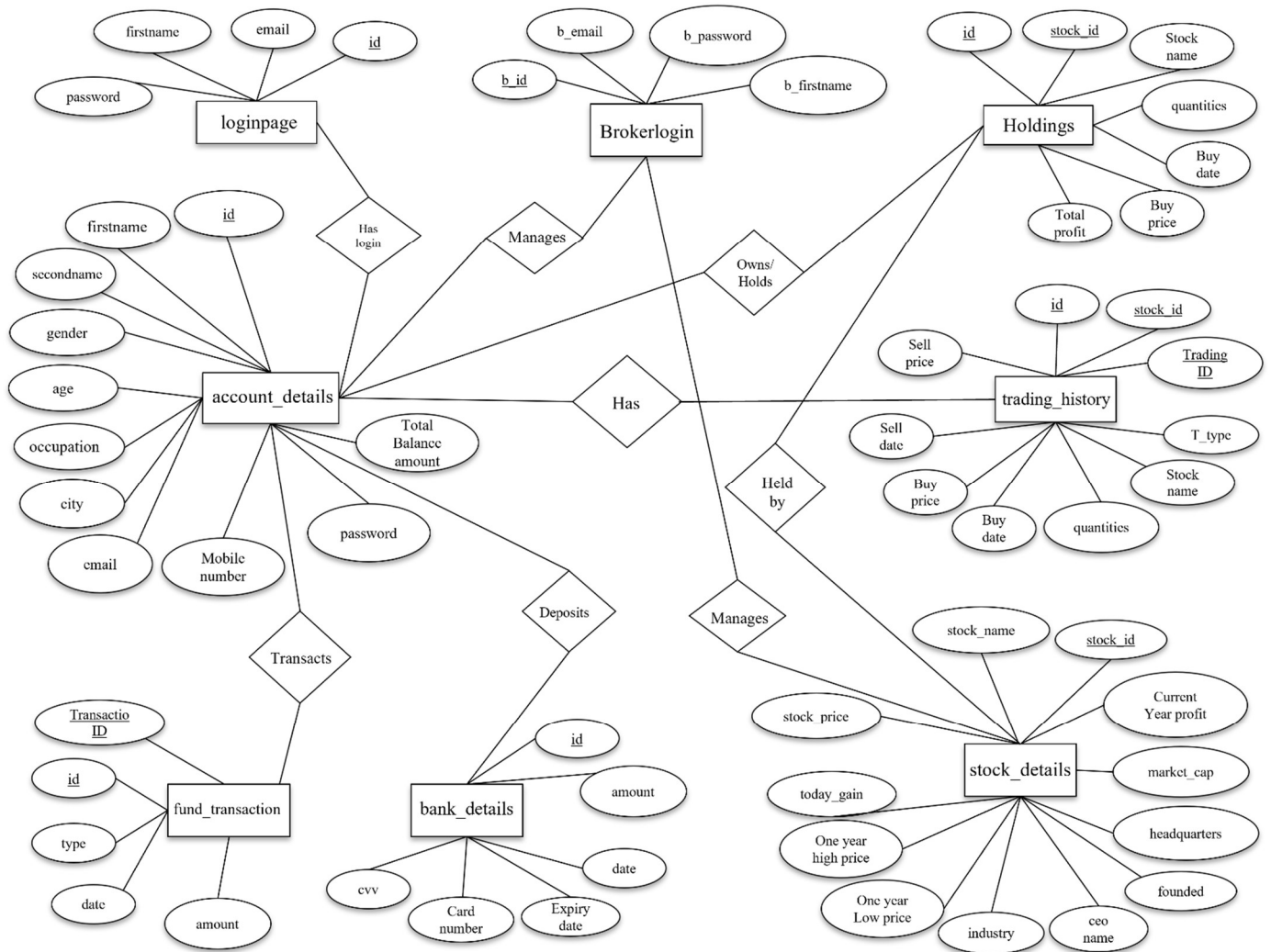
The primary function of the stock market is to facilitate the transfer of capital from investors to companies in need of funding. By purchasing shares, investors essentially buy a piece of the company and become partial owners. This ownership entitles them to a share of the company's profits, typically paid out as dividends, and the right to vote on significant company decisions. The liquidity provided by the stock market ensures that investors can quickly buy or sell shares, making it an attractive investment option compared to other less liquid assets.

In this context, our project, **Stock Base**, aims to provide a comprehensive web application for managing stock market transactions. Stock Base is designed to cater to both individual investors and brokers, offering a seamless and intuitive platform for buying, selling, and tracking stocks. The application leverages the power of Flask, a lightweight and flexible web framework for Python, and MySQL, a robust database management system, to deliver a dynamic and scalable solution.

Stock Base focuses on user authentication, ensuring secure access for authorized users through Flask-Login. Upon logging in, users are presented with interfaces tailored to their roles. Regular users can view real-time stock prices, execute transactions, and manage their portfolios, while brokers can add new stocks to the system and edit stock details. The application simulates real-time stock price updates with predefined fluctuation rules, offering a dynamic user experience that mirrors the real stock market environment.

By providing detailed transaction history tracking and real-time updates, Stock Base empowers users to make informed investment decisions. Brokers benefit from a comprehensive management interface, ensuring that stock data remains accurate and up-to-date. Overall, Stock Base aims to enhance the trading and management experience for both individual investors and brokers, making it a powerful tool in the world of stock market transactions.

# Entity Relationship(ER) diagram



## Entities and Their Attributes

1) **account\_details** : Attributes: id, firstname, secondname, gender, age, occupation, city, email, password, total\_balance\_amount, mobilenumber. Represents user account details.

2) **brokerlogin** : Attributes: b\_id, b\_firstname, b\_email, b\_password. Represents broker login details.

3) **loginpage** : Attributes: id, firstname, email, password. Represents user login information.

4) **stock\_details** : Attributes: stock\_id, stock\_name, stock\_price, today\_gain, one\_year\_lowprice, one\_year\_highprice, ceo\_name, founded, industry, headquarters, market\_cap, current\_year\_profit. Represents stock details.

5) **holdings** : Attributes: id, stock\_id, stock\_name, quantities, buy\_date, buy\_price, total\_profit. Represents stocks held by users.

**6) trading\_history** : Attributes: id, stock\_id, trading\_id, t\_type, stock\_name, quantities, buy\_date, buy\_price, sell\_date, sell\_price. Represents stock trading history.

**7) fund\_transaction** : Attributes: transaction\_id, id, type, date, amount. Represents user fund transactions.

**8) bank\_details** : Attributes: id, amount, card\_number, expiry\_date, cvv, date. Represents user bank details.

## **Relationships**

**1) Has\_login (between account\_details and loginpage)** : One-to-One: Each user has one login credential.

**2) Manages (between brokerlogin and holdings, trading\_history)** : One-to-Many: Each broker manages multiple holdings and trading histories.

**3) Owns/Holds (between account\_details and holdings)** : One-to-Many: Each user can own various stocks.

**4) Held\_by (between holdings and account\_details)** : Many-to-One: Each holding is linked to one user account (inverse of Owns/Holds).

**5) Deposits (between account\_details and bank\_details)** : One-to-Many: Each user can have multiple bank deposit records.

**6) Transacts (between account\_details and fund\_transaction)** : One-to-Many: Each user can perform multiple fund transactions.

# Process Workflow

## 1. User Registration (Signup)

- A user initiates the registration process by navigating to the registration page.
- On this page, the user fills out a form with their personal details, such as their name, email, and password.
- Upon submission, the system checks if the provided email is already associated with an existing account.
  - If the email is already in use, the system informs the user and prompts them to try again.
  - If the email is unique, the system creates a new account with the provided details and .
- The user receives a confirmation that their account has been created successfully and is encouraged to log in.

## 2. User Login

- The user navigates to the login page and enters their email and password.
- The system verifies the provided credentials by checking them against the stored account information.
- If the credentials match, the user is granted access and redirected to their account dashboard.
- If the credentials do not match, the user is informed that the login attempt failed and invited to try again.

## 3. Accessing Stock Information

- Once logged in, the user can access their stock information dashboard.
- The system retrieves the user's current stock holdings and displays relevant details, such as stock names, quantities, and current values.

## 4. Viewing Stock Details

- The user selects a specific stock to view more detailed information.
- The system provides detailed data about the selected stock, including historical performance, pricing, and other relevant metrics.

## 5. Buying Stocks

- The user decides to purchase stocks and enters the desired quantity on a purchase form.
- The system verifies whether the user has sufficient funds for the purchase.
- If the user has enough balance, the system processes the transaction, updates the user's stock holdings, and adjusts their account balance.
- If the user does not have enough funds, the system informs them of the insufficiency and prompts them to modify their purchase.

## 6. Selling Stocks

- The user chooses to sell stocks and specifies the quantity they wish to sell.
- The system checks if the user owns enough of the specified stock to proceed with the sale.

- If the user has sufficient stock, the system processes the sale, updates the user's holdings, and credits their account balance.
- If the user does not have the required amount, the system alerts them and prompts them to revise their sale request.

## **7. Managing Account Details**

- The user can view and update their account information, such as their personal details or login credentials.
- The system applies any changes made by the user to their account profile.

## **8. Viewing Holdings and Trading History**

- The user can view a summary of their current stock holdings and their trading history.
- The system displays a list of stocks the user owns and details of past transactions, including buys, sells, and other financial activities.

## **9. Deposits and Withdrawals**

- The user can add funds to their account or withdraw money.
- When depositing, the user provides payment details, and the system updates the account balance accordingly.
- When withdrawing, the user specifies the amount, and the system checks if the user has sufficient funds before processing the withdrawal and updating the balance.

## **10. Broker Login and Stock Management**

- Brokers access a separate login page to manage stock information.
- After logging in, brokers can add new stocks to the system or update existing stock details.
- The system allows brokers to enter and modify stock information, which is then updated in the stock database.

## **11. Editing Stock Details**

- Brokers can modify information about specific stocks.
- The system applies these updates to ensure the stock information remains current.

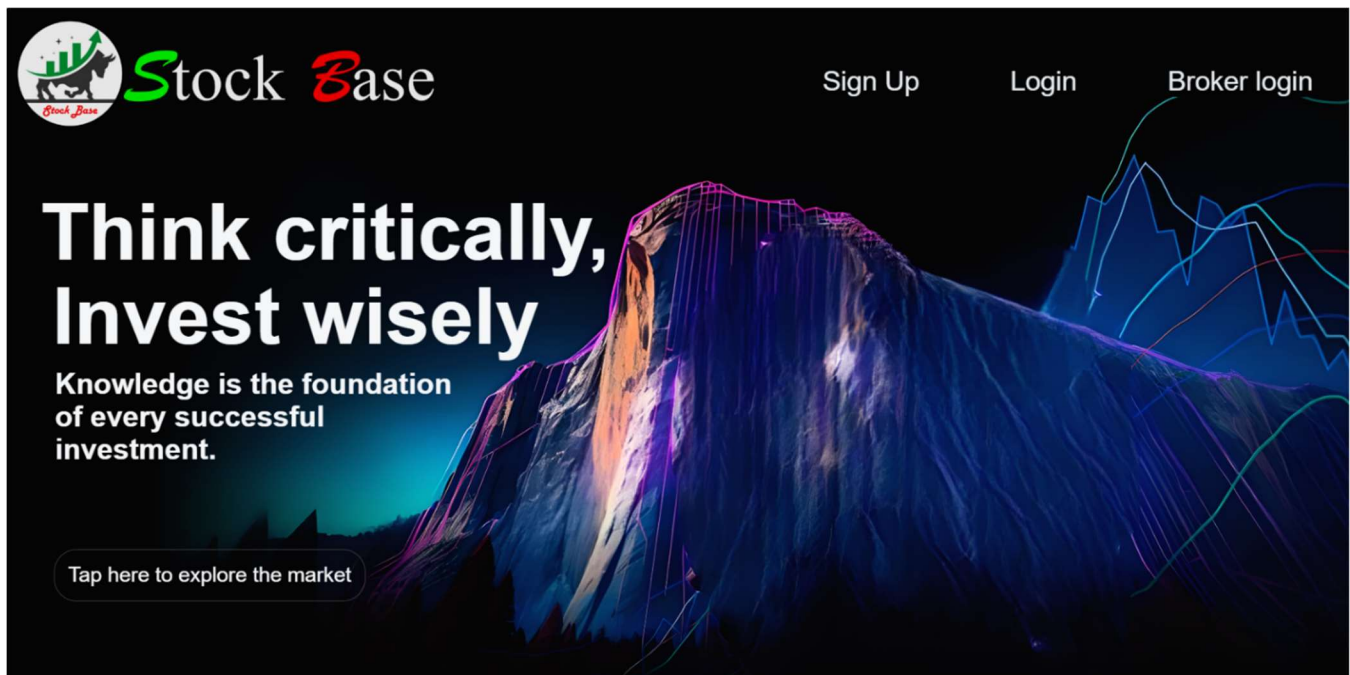
## **12. Overview and Logout**

- Users and brokers can view an overview of all accounts and stock details.
- When choosing to log out, the system terminates the user's session and redirects them to the home page or login page, ensuring the user's session data is properly cleared.

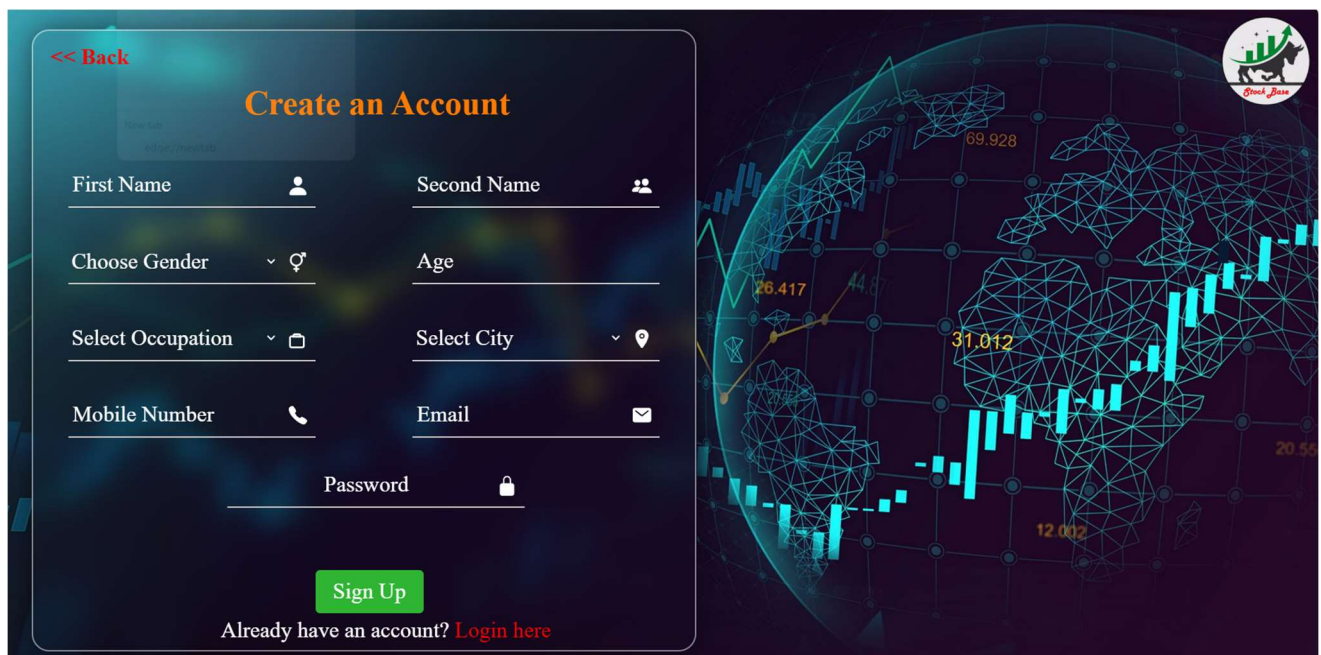


## Visual Output's of the Project

### 1) Index page



### 2) Signup / Register page



### 3) The dashboard(Home Page) that appears after login



Welcome To Stock Base, Manoj

[Home Page](#)

[Stock Holdings](#)

[Trading History](#)

[Transactions](#)

[Account Details](#)

[Log Out](#)

#### Available Stocks

Stock ID	Stock Name	Current Price	Today's Gain	
12345	Reliance Industries	₹ 2501.5	-0.75	<a href="#">View Details</a>
23456	Tata Consultancy Services	₹ 3301.5	+2.25	<a href="#">View Details</a>
34567	Infosys	₹ 1551.05	-1.5	<a href="#">View Details</a>
45678	HDFC Bank	₹ 1401.9	-0.75	<a href="#">View Details</a>
56789	ICICI Bank	₹ 701.0	-1.5	<a href="#">View Details</a>

### 4) The page that appears after selecting BUY

[<< Back To Stock Details](#)

[Log Out](#)

#### Buy Reliance Industries


Stock ID	12345
Stock Name	Reliance Industries
Stock Price	2519.5
Today's Gain	+2.25
Total available Funds	6173.5

Enter the Number of Quantities

[Click Here To Add Funds](#)

[Click Here To BUY](#)


## 5) The page displaying the user's stock holdings

Stock Base

[Home Page](#)[Stock Holdings](#)[Trading History](#)[Transactions](#)[Account Details](#)[Log Out](#)

Your Stock Holdings Details						
Stock Name	Number of Quantities	Buy Date	Buy Price	Current Price	Total Gain	
Reliance Industries	1	2024-07-24	₹ 2501.5	₹ 2519.5	+ 18.0	SELL
Infosys	1	2024-07-24	₹ 1555.05	₹ 1559.05	+ 4.0	SELL
Mahindra & Mahindra	2	2024-07-24	₹ 845.25	₹ 841.5	- 7.5	SELL
ICICI Bank	2	2024-07-24	₹ 707.0	₹ 709.0	+ 4.0	SELL
Bharti Airtel	3	2024-07-24	₹ 557.4	₹ 559.65	+ 6.75	SELL


## 6) The page where the user can view their trade history

Stock Base

[Home Page](#)[Stock Holdings](#)[Trading History](#)[Transactions](#)[Account Details](#)[Log Out](#)

Your Trading History								
Trading ID	Stock ID	Stock Name	Trading Type	Quantities	Buy Date	Buy Price	Sell Date	Sell Price
1	12345	Reliance Industries	BUY	1	2024-07-24	₹ 2501.5	-	-
2	12345	Reliance Industries	SELL	2	-	-	2024-07-24	₹ 2508.25
3	34567	Infosys	BUY	1	2024-07-24	₹ 1555.05	-	-
4	89012	Mahindra & Mahindra	BUY	2	2024-07-24	₹ 845.25	-	-
5	56789	ICICI Bank	BUY	2	2024-07-24	₹ 707.0	-	-
6	67890	Bharti Airtel	BUY	3	2024-07-24	₹ 557.4	-	-
7	67890	Bharti Airtel	SELL	1	-	-	2024-07-24	₹ 559.65

## 7) The page that shows fund transaction details and allows users to deposit/withdraw fund

 **Stock Base**


[Home Page](#) [Stock Holdings](#) [Trading History](#) [Transactions](#) [Account Details](#) [Log Out](#)

Available Fund Balance: ₹ 7413.65      Click Here To Deposit The Fund: [Deposit](#)      Click Here To Withdraw The Fund: [Withdraw](#)

**Your Transactions**

Transaction ID	Transaction Type	Amount	Date
1	DEPOSIT	₹ 10000.0	2024-07-24 06:17:59
2	DEPOSIT	₹ 5000.0	2024-07-24 06:20:50
3	WITHDRAW	₹ 1000.0	2024-07-24 06:21:46

## 8) The page shows user profile details and allows users to edit their information

 **Stock Base**

[Home Page](#) [Stock Holdings](#) [Trading History](#) [Transactions](#) [Account Details](#) [Log Out](#)

**Account Details**

Customer ID

1

First Name

Manoj

Last Name

G

Gender

Male

Email Address

man@gmail.com

Mobile Number

1234567890

Age

20

Occupation

Student

City


Bengaluru

Available Funds

₹ 7413.65

Edit

9) This page is designated for brokers, who must authenticate through the brokerlogin method, to add new stocks to the market

 **Stock Base**

[Edit Stock](#) [Add Stock](#) [Stock Trades](#) [Customer Overview](#) [Log Out](#)

---

**Add New Stock**

10) Page providing brokers with an overview of registered customer information

 **Stock Base**

[Edit Stock](#) [Add Stock](#) [Stock Trades](#) [Customer Overview](#) [Log Out](#)

---

**Customer's Overview**

Customer ID	Firstname	Gender	Age	City	Email address
1	Manoj	Male	20	Bengaluru	manoj@gmail.com
2	Kushal	Male	19	Bengaluru	kushal@gmail.com
3	Rajat	Male	34	Hyderabad	rajat123@gmail.com
4	Smriti	Female	28	Delhi	sm@gmail.com
5	Veloni	Female	65	Ahmedabad	veloni@gmail.com
6	Ratan	Male	42	Mumbai	ratant@gmail.com

## Conclusion

The Stock Base web application represents a significant advancement in the management of stock market transactions for both individual investors and brokers. By leveraging Flask and MySQL, the project effectively integrates dynamic web functionalities with robust data management capabilities, providing a comprehensive and user-friendly platform for trading and investment management.

Throughout the development of Stock Base, careful consideration was given to creating an intuitive interface and ensuring secure, efficient interactions between users and the system. The implementation of Flask-Login for user authentication highlights our commitment to maintaining the privacy and security of user data, ensuring that access to sensitive information is restricted to authorized individuals. This focus on security is complemented by the application's ability to simulate real-time stock price fluctuations, enhancing the realism and effectiveness of the trading experience.

The distinct functionalities provided for regular users and brokers reflect the application's versatility. Regular users benefit from the ability to view detailed stock information, manage their portfolios, and execute transactions, while brokers are empowered to add and modify stock data, ensuring the accuracy and relevance of the information available. The detailed tracking of transaction history further aids users in analyzing their trading strategies and making informed investment decisions.

The application's process workflow, from registration to trading and account management, is designed to be seamless and straightforward. Users can easily navigate through their account dashboard, view stock details, and perform transactions with minimal friction. The system also supports financial transactions such as deposits and withdrawals, adding to its functionality and appeal.

In conclusion, Stock Base stands out as a robust tool for managing stock market transactions, blending technical sophistication with user-centric design. The project's successful integration of Flask and MySQL, combined with its secure authentication mechanisms and comprehensive management features, positions it as a valuable resource for investors and brokers alike. As stock markets continue to evolve, Stock Base offers a scalable solution that can adapt to changing market conditions and user needs, ensuring its continued relevance and effectiveness in the dynamic world of financial trading.