Phase : 3

CODE IMPLEMENTATION

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Real-Time Stock Ticker</title>

<style>

body {

margin: 0;

font-family: 'Segoe UI', sans-serif;

background: linear-gradient(to right, #0f2027, #203a43, #2c5364);

color: #fff;

display: flex;

flex-direction: column;

align-items: center;

}

header {

padding: 20px;

text-align: center;

background-color: #111;

width: 100%;

box-shadow: 0 2px 5px rgba(0,0,0,0.5);

}

.ticker-container {

overflow: hidden;

width: 100%;

background-color: #000;

border-top: 2px solid #00ff99;

border-bottom: 2px solid #00ff99;

}

.ticker {

display: flex;

animation: scroll 25s linear infinite;

}

.stock {

padding: 10px 20px;

white-space: nowrap;

border-right: 1px solid #333;

color: #00ff99;

font-weight: bold;

transition: background 0.3s;

}

.stock:hover {

background-color: #333;

cursor: pointer;

}

.chart-container {

width: 80%;

max-width: 600px;

margin: 40px auto;

}

footer {

margin-top: auto;

padding: 10px;

font-size: 0.9em;

color: #aaa;

}

@keyframes scroll {

0% { transform: translateX(100%); }

100% { transform: translateX(-100%); }

}

</style>

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

</head>

<body>

<header>

<h1>📊 Real-Time Stock Ticker</h1>

</header>

<section class="ticker-container">

<div class="ticker" id="ticker">

<!-- Stocks will appear here -->

</div>

</section>

<section class="chart-container">

<canvas id="stockPieChart"></canvas>

</section>

<footer>

<p>Made with 💡 by SRM</p>

</footer>

<script>

const stocks = [

{ symbol: "AAPL", price: 174.55 },

{ symbol: "GOOGL", price: 138.12 },

{ symbol: "TSLA", price: 265.33 },

{ symbol: "AMZN", price: 132.45 },

{ symbol: "MSFT", price: 328.76 },

{ symbol: "INFY", price: 1450.25 },

{ symbol: "RELIANCE", price: 2480.10 },

{ symbol: "TCS", price: 3620.75 }

];

const ticker = document.getElementById("ticker");

function renderTicker() {

ticker.innerHTML = "";

stocks.forEach(stock => {

const item = document.createElement("div");

item.className = "stock";

item.textContent = ${stock.symbol}: ₹${stock.price.toFixed(2)};

ticker.appendChild(item);

});

}

function renderPieChart() {

const ctx = document.getElementById("stockPieChart").getContext("2d");

new Chart(ctx, {

type: "pie",

data: {

labels: stocks.map(s => s.symbol),

datasets: [{

label: "Stock Prices",

data: stocks.map(s => s.price),

backgroundColor: [

"#00ff99", "#00ccff", "#ffcc00", "#ff6666",

"#cc99ff", "#66ffcc", "#ff9966", "#9999ff"

],

borderColor: "#111",

borderWidth: 1

}]

},

options: {

plugins: {

legend: {

labels: {

color: "#fff"

}

}

}

}

});

}

renderTicker();

renderPieChart();

</script>

</body>

</html>

Phase : 4

🖥 Frontend (React)

App.js

import React, { useEffect, useState, lazy, Suspense } from 'react';

import io from 'socket.io-client';

import Skeleton from 'react-loading-skeleton';

import './App.css';

const StockChart = lazy(() => import('./components/StockChart'));

const socket = io('http://localhost:5000');

function App() {

const [price, setPrice] = useState(null);

const [loading, setLoading] = useState(true);

useEffect(() => {

socket.on('priceUpdate', data => {

setPrice(data);

setLoading(false);

});

}, []);

return (

<div className="dashboard">

<h1>📈 Real-Time Stock Ticker</h1>

{loading ? <Skeleton height={40} width={200} /> : <h2>${price}</h2>}

<Suspense fallback={<Skeleton height={300} />}>

<StockChart />

</Suspense>

</div> ); }

export default App;

StockChart.js

import React from 'react';

import { Line } from 'react-chartjs-2';

const StockChart = () => {

const data = {

labels: ['Mon', 'Tue', 'Wed', 'Thu', 'Fri'],

datasets: [{

label: 'Stock Price',

data: [120, 125, 130, 128, 135],

borderColor: 'blue',

fill: false }] };

return <Line data={data} />; };

export default StockChart;

🔧 Backend (Node.js + Express)

index.js

Const express = require('express');

const http = require('http');

const socketIo = require('socket.io');

const cors = require('cors');

const mongoose = require('mongoose');

const jwt = require('jsonwebtoken');

const rateLimit = require('express-rate-limit');

const swaggerUi = require('swagger-ui-express');

const swaggerDocument = require('./swagger.json');

const app = express();

const server = http.createServer(app);

const io = socketIo(server);

mongoose.connect('mongodb://localhost:27017/stocks');

app.use(cors());

app.use(express.json());

app.use('/docs', swaggerUi.serve, swaggerUi.setup(swaggerDocument));

app.use(rateLimit({ windowMs: 60000, max: 100 }));

// HTTPS enforcement

app.use((req, res, next) => {

if (req.headers['x-forwarded-proto'] !== 'https') {

return res.redirect(https://${req.headers.host}${req.url}); }

next(); });

// WebSocket for real-time price

io.on('connection', socket => {

setInterval(() => {

const price = (100 + Math.random() \* 50).toFixed(2);

socket.emit('priceUpdate', price);

}, 1000);

}); server.listen(5000, () => console.log('Server running on port 5000'));

🔐 Security & API

JWT Auth Middleware

module.exports = function(req, res, next) {

const token = req.header('Authorization');

if (!token) return res.status(401).send('Access Denied');

try {

const verified = jwt.verify(token, 'secretKey');

req.user = verified;

next();

} catch (err) {

res.status(400).send('Invalid Token');

}

};

Input Validation

const { check, validationResult } = require('express-validator');

app.post('/api/watchlist', [

check('symbol').isAlphanumeric()

], (req, res) => {

const errors = validationResult(req);

if (!errors.isEmpty()) return res.status(400).json({ errors: errors.array() });

// Save to DB

});