Multiple Company Stock App

1. **index.html**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Multi-Company Stock Data</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <div class="container">
        <h1>Stock Data</h1>
        <select id="company-select">
            <option value="AAPL">Apple Inc. (AAPL)</option>
            <option value="GOOGL">Alphabet Inc. (GOOGL)</option>
            <option value="MSFT">Microsoft Corporation (MSFT)</option>
            <option value="NVDA">N-VIDIA (NVDA)</option>
            <option value="TSLA">Tesla, Inc. (TSLA)</option>
            <option value="INFY">Infosys Ltd.</option>
            <!-- Add more options as needed -->
        </select>
        <table>
            <thead>
                <tr>
                    <th>Date</th>
```

```
                <th>Volume</th>
                <th>VWAP</th>
                <th>Open</th>
                <th>Close</th>
                <th>High</th>
                <th>Low</th>
                <th>Trades</th>
            </tr>
        </thead>
        <tbody id="stock-data">
            <!-- Data will be populated here -->
        </tbody>
    </table>
</div>
<script src="script.js"></script>
</body>
</html>
```

2. script.js

```
document.addEventListener("DOMContentLoaded",  async function() {
    const select = document.getElementById("company-select");
    const tbody = document.getElementById("stock-data");

    async function fetchData(ticker) {
        try {
            const response = await fetch(`/api/stock/${ticker}`);
```

```
        const data = await response.json();
        tbody.innerHTML = '';

        data.results.forEach(result => {
            const row = document.createElement("tr");

            const date = new Date(result.t);
            const dateString = date.toLocaleDateString();

            row.innerHTML = `
                <td>${dateString}</td>
                <td>${result.v}</td>
                <td>${result.vw.toFixed(2)}</td>
                <td>${result.o.toFixed(2)}</td>
                <td>${result.c.toFixed(2)}</td>
                <td>${result.h.toFixed(2)}</td>
                <td>${result.l.toFixed(2)}</td>
                <td>${result.n}</td>
            `;

            tbody.appendChild(row);
        });
    } catch (e) {
        console.error('An error occurred while fetching data:', e);
    }
}
```

```
    select.addEventListener("change", function() {
        fetchData(select.value);
    });

    fetchData(select.value); // Fetch data for the default selected company on page load
});
```

3. style.css

```
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f9;
    margin: 0;
    padding: 0;
}

.container {
    width: 80%;
    margin: 50px auto;
    background: #fff;
    padding: 20px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

h1 {
```

```
    text-align: center;
    color: #333;
}

select {
    display: block;
    margin: 20px auto;
    padding: 10px;
    font-size: 16px;
}

table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 20px;
}

thead {
    background-color: #333;
    color: #fff;
}

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

```
}

tbody tr:nth-child(odd) {
    background-color: #f9f9f9;
}

tbody tr:nth-child(even) {
    background-color: #ececec;
}

tbody tr:hover {
    background-color: #d3d3d3;
}
```

These three files are inside public folder at project's root

4. **server.js**

```
const express = require('express');
const path = require('path');
const dotenv = require('dotenv');
const { restClient } = require("@polygon.io/client-js");

dotenv.config();
const app = express();
const port = process.env.PORT || 3000;
```

```
const rest = restClient(process.env.POLYGON_API_KEY);


app.use(express.static(path.join(__dirname, 'public')));


app.get('/api/stock/:ticker', async (req, res) => {
    try {
        const ticker = req.params.ticker.toUpperCase();
        const data = await rest.stocks.aggregates(ticker, 1, "day", "2025-01-01", "2025-05-11");
        res.json(data);
    } catch (e) {
        res.status(500).json({ error: "An error happened: " + e.message });
    }
});


app.listen(port, () => {
    console.log(`Server is running on http://localhost:${port}`);
});


.env with POLYGON_API_KEY=My_Secret_Key
.gitignore with  code as
echo ".env" >> .gitignore
git add .gitignore
git commit -m "Updated .gitignore to exclude .env"
git push origin main
```

package.json

```
{
  "name": "multi-company-stock-app",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "@polygon.io/client-js": "^7.3.2",
    "dotenv": "^16.4.7",
    "express": "^4.21.2"
  }
}
```

And package-lock.json file alongwith node_modules folder in the VS Code Project root

Help these code files as a base to create a brand new VS Code Project with vercel's serverless functions that can be deployable to vercel and can load the companies data from the past mentioned dates.