

# Stock Tracker

Thomas Clermont

# Was sollte erreicht werden?

- Sign in/ Sign up **Anmeldefunktionen**, um Benutzerdaten und Portfolios zu speichern.
- **Portfolioverwaltung** Benutzer können mehrere Portfolios erstellen und verwalten.  
Portfolios koennen beliebige Aktien beinhalten.
- **Echtzeit-Aktiendaten** Echtzeit-Aktiendaten von API (wsl. Yahoo Finance) um relevante Daten anzuzeigen.
- **Portfolioübersicht** Überblick über die Leistung des Portfolios (wert, Gewinne/Verluste).
- **Aktiensuche** Aktien nach Symbol oder Namen suchen und Hinzufügen zu Portfolio.
- **Aktiendetails** Informationen zu jeder Aktie (evtl. diagramme, Unternehmensinfos, aktuelle Nachrichten).
- **Responsive Website** Website zugaenglich auf Geräten und Bildschirmgrößen.
- **User-friendly** Benutzerfreundliche Oberfläche mit klarer Navigation.

# Was wurde umgesetzt?

✓ Sign in/ Sign up **Anmeldefunktionen**, um Benutzerdaten und **Portfolios zu speichern**.

Portfolioverwaltung Benutzer können mehrere Portfolios erstellen und verwalten. Portfolios können beliebige Aktien beinhalten.

✓ **Echtzeit-Aktiendaten** von API (wsl. Yahoo Finance) um relevante Daten anzuzeigen.

Portfolioübersicht Überblick über die Leistung des Portfolios (Wert, Gewinne/Verluste).

✓ Aktiensuche **Aktien nach Symbol oder Namen suchen** und Hinzufügen zu Portfolio.

Aktiendetails Informationen zu jeder Aktie (evtl. Diagramme, Unternehmensinfos, aktuelle Nachrichten).

✓ Responsive Website Website **zugänglich auf allen Geräten und Bildschirmgrößen**.

✓ **User-friendly** Benutzerfreundliche Oberfläche mit klarer Navigation.

## Weiteres:

- ✓ **News seite** mit top 50 stocks discussed stocks
- ✓ **Chart.js** mit zeitspanne und stockname
- ✓ **Seperate funktionen** für logged in und nicht logged in

# Was wurde weggelassen?

Sign in/ Sign up Anmeldefunktionen, um Benutzerdaten und Portfolios zu speichern.

✗ Portfolioverwaltung Benutzer können **mehrere Portfolios erstellen und verwalten**. Portfolios können beliebige Aktien beinhalten.

Echtzeit-Aktiendaten Echtzeit-Aktiendaten von API (wsl. Yahoo Finance) um relevante Daten anzuzeigen.

✗ Portfolioübersicht **Überblick über die Leistung des Portfolios** (wert, Gewinne/Verluste).

Aktiensuche Aktien nach Symbol oder Namen suchen und Hinzufügen zu Portfolio.

✗ Aktiendetails Informationen zu jeder Aktie (evtl. diagramme, **Unternehmensinfos, aktuelle Nachrichten**).

Responsive Website Website zugänglich auf Geräten und Bildschirmgrößen.

User-friendly Benutzerfreundliche Oberfläche mit klarer Navigation.

# Besondere Bestandteile:

- chart.js
- database implementation

# Chart.js implementation (Stockdata API request)

```
function fetchHistoricalStockData(stockName, startDate, endDate) {  
  const apiUrl = `https://api.polygon.io/v2/aggs/ticker/${stockName}/range/1/day/${startDate}/${endDate}?unadjusted=false&apiKey=${apiKey}`;  
  
  return fetch(apiUrl)  
    .then((response) => {  
      if (!response.ok) {  
        throw new Error("Network response was not ok");  
      }  
      return response.json();  
    })  
    .then((data) => {  
      if (!data || !data.results || data.results.length === 0) {  
        throw new Error("API response did not contain valid data");  
      }  
      const historicalData = data.results.map((result) => ({  
        timestamp: new Date(result.t),  
        open: result.o,  
        high: result.h,  
        low: result.l,  
        close: result.c,  
      }));  
      return historicalData;  
    })  
    .catch((error) => {  
      console.error("Error fetching historical stock data:", error);  
      return null;  
    });  
}
```

# Chart.js implementation (chartjs-chart-financial library)

```
/*!
 * @license
 * chartjs-chart-financial
 * http://chartjs.org/
 * Version: 0.1.0
 *
 * Copyright 2021 Chart.js Contributors
 * Released under the MIT license
 * https://github.com/chartjs/chartjs-chart-financial/blob/master/LICENSE.md
 */
(function (global, factory) {
  typeof exports === 'object' && typeof module !== 'undefined' ? factory(require('chart.js'), require('chart.js/helpers')) :
  typeof define === 'function' && define.amd ? define(['chart.js', 'chart.js/helpers'], factory) :
  (global = typeof globalThis !== 'undefined' ? globalThis : global || self, factory(global.Chart, global.Chart.helpers));
})(this, (function (chart_js, helpers) { 'use strict';

  /**
   * Computes the "optimal" sample size to maintain bars equally sized while preventing overlap.
   * @private
   */
  function computeMinSampleSize(scale, pixels) {
    let min = scale._length;
    let prev, curr, i, ilen;

    for (i = 1, ilen = pixels.length; i < ilen; ++i) {
      min = Math.min(min, Math.abs(pixels[i] - pixels[i - 1]));
    }

    for (i = 0, ilen = scale.ticks.length; i < ilen; ++i) {
      curr = scale.getPixelForTick(i);
      min = i > 0 ? Math.min(min, Math.abs(curr - prev)) : min;
      prev = curr;
    }

    return min;
  }

  /**
   * This class is based off controller.bar.js from the upstream Chart.js library
   */
  class FinancialController extends chart_js.BarController {

    getLabelAndValue(index) {
      const me = this;
      const parsed = me.getParsed(index);
      const axis = me._cachedMeta.isScale.axis;
    }
  }
})
```

<https://www.chartjs.org/chartjs-chart-financial/chartjs-chart-financial.js>

```
<script src="https://www.chartjs.org/chartjs-chart-financial/chartjs-chart-financial.js"></script>
```







# Chart.js implementation (create chart)

```
function drawChart() {  
  fetchHistoricalStockData(stockName, startDate, endDate).then((data) => {  
    if (data) {  
      // Check if the chart already exists and destroy it  
      if (stockChart) {  
        stockChart.destroy();  
      }  
  
      var ctx = document.getElementById("stockChart").getContext("2d");  
  
      // Die Charts sollten die Screenbreite nicht überschreiten  
      const canvasWidth = window.innerWidth - 20;  
      ctx.canvas.width = canvasWidth;  
  
      ctx.canvas.height = 800;  
  
      // Data for chart  
      var barData = data.map((result) => ({  
        x: result.timestamp.valueOf(), //use luxon to convert date input to right format  
        o: result.open,  
        h: result.high,  
        l: result.low,  
        c: result.close,  
      }));  
    }  
  });  
}
```



```
    // Create a new Chart chart  
    stockChart = new Chart(ctx, {  
      type: "candlestick",  
      data: {  
        datasets: [  
          {  
            label: stockName + " candlestick chart",  
            data: barData,  
          },  
        ],  
      },  
    });  
  }  
});  
drawChart();  
}
```

# Database (table structure)

## logindata

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	<b>Username</b>	varchar(20)	utf8mb4_general_ci		No	None			 Change  Drop More
<input type="checkbox"/>	2	<b>Password</b>	varchar(20)	utf8mb4_general_ci		No	None			 Change  Drop More

## userstocks

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	<b>userID</b>	varchar(50)	utf8mb4_general_ci		No	None			 Change  Drop More
<input type="checkbox"/>	2	<b>stockID</b>	varchar(50)	utf8mb4_general_ci		No	None			 Change  Drop More

# Database (establish connection)

```
if ($_SERVER["REQUEST_METHOD"] == "POST") { // if login form has been submitted with username and password
    $username = $_POST['username'];
    $password = $_POST['password'];

    $conn = new mysqli("localhost", "root", "", "stocktrackerlogin"); // connect to database "stocktrackerlogin"

    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error); // throw error if connection fails
    }
}
```

## Database (match user input with database)

```
// check if the provided username and password match database
$stmt = $conn->prepare("SELECT * FROM logindata WHERE Username = ? AND Password = ?");
$stmt->bind_param("ss", $username, $password); //"ss" --> (string, string)
```

Database (match user input with database)

```
$conn->close();
```