

Projektdokumentation

Wertpapiersimulation

Ausbildungsberuf

Fachinformatiker/-in Fachrichtung: Anwendungsentwicklung

Berufsschule

Brühlwiesenschule

Auszubildende

Danny Nicolay Walter Binder, Cristian Felipe Castillo-Barrero, Moritz Heinke

E-Mail-Adresse: moritz.heinke@deutsche-boerse.com

Betrieblicher Ansprechpartner: Jan Patrick Drehwald

E-Mail-Adresse: jan.patrick.drehwald@clearstream.com | Telefon: +49 69 211-1 41 41

Thema der Projektarbeit:

Entwicklung und Implementierung einer plattformunabhängigen Simulation eines Wertpapierdepots

Inhaltsverzeichnis

1	Einleitung.....	3
1.1	Vorwort	3
1.2	Projektauftrag.....	3
1.3	Projektumfeld.....	3
2	Projektplanung	3
2.1	Funktionale Anforderungen	3
2.2	Ressourcen und Kostenplanung	3
2.2.1	Zeitliche Planung	3
2.2.2	Kostenplanung	3
3	Projektdurchführung	4
3.1	Vorbereitung.....	4
3.2	Defintion	4
3.3	Entwurf	4
3.4	Implementation	5
3.4.1	Entwicklungsumgebung.....	5
3.4.2	Einrichtung der Datenbank	5
3.4.3	Verwendung von yfinance.....	5
3.4.4	Aufbau der Webanwendung	5
3.5	Testen	6
3.5.1	Komponententest	6
3.5.2	Integration Test.....	6
3.6	Installation.....	6
4	Fazit.....	7
4.1	Ist/Soll Vergleich.....	7
4.2	Ausblick	7
4.2.1	Weitere Funktionalitäten	7
4.3	Gewonnene Erkenntnisse	7
5	Glossar	8
6	Anhang.....	8
7	Quellen.....	13
8	Eigenständigkeitserklärung.....	13

1 Einleitung

1.1 Vorwort

Verwendete Fachbegriffe und Abkürzungen sind in Kapitel 5 (Glossar) erwähnt und erklärt.

1.2 Projektauftrag

Der Auftrag besteht in der Entwicklung einer Simulationsplattform zum spielerischen Erlernen von praxisnahem und lehrreichem Verständnis für den Handel mit Wertpapieren. Die Plattform ermöglicht es den Benutzern mithilfe von virtuellem Kapital, Käufe und Verkäufe von Wertpapieren durchzuführen. Das Gesamtkapital des Benutzers wird in Echtzeit basierend auf den aktuellen Börsenkursen berechnet und angepasst. Darüber hinaus soll die Simulation die Dynamiken des echten Börsenhandels visuell abbilden.

1.3 Projektumfeld

Das Projekt wird innerhalb der Brühlwiesenschule, für die Lernfelder zehn bis zwölf, innerhalb des Schulblocks vom 26.08.2024 – 20.09.2024 durchgeführt. Innerhalb des Gruppenprojektes werden die bislang erlernten Fertigkeiten der letzten beiden Lehrjahre genutzt. Neue Konzepte, wie Model-View-Trennung, Barrierefreiheit und plattformübergreifende Kompatibilität werden durch Eigeninitiative erlernt und angewandt. Das Projekt wird im folgenden Schulblock durch eine Präsentation und eine Live-Demo vorgestellt.

2 Projektplanung

2.1 Funktionale Anforderungen

Im Vorfeld der Anforderungsbesprechung war es notwendig, das Projekt gründlich zu durchdenken und zu planen. Dabei ging es darum, eine klare Vision für die Simulationsplattform zu entwickeln. Es mussten Überlegungen angestellt werden, wie die Benutzererfahrung gestaltet werden kann, welche Funktionen integriert werden sollen und wie die Echtzeit-Berechnung des Kapitals sowie die visuelle Darstellung der Börsendynamiken umgesetzt werden können. Diese Vorarbeit war entscheidend, um sicherzustellen, dass alle Anforderungen und Ziele des Projekts klar definiert und realisierbar sind.

2.2 Ressourcen und Kostenplanung

2.2.1 Zeitliche Planung

Der Zeitplan des Projekts wurde aus dem Projektantrag übernommen und definiert den Ablauf des geplanten Projekts (Tabelle 2: Zeitplanung). Wegen der konkreten Anforderungen und des zeitlichen Horizonts bietet sich das erweiterte Wasserfallmodell als geeignetes Vorgehensmodell an. Die geplanten Abläufe werden nach dem erweitertem Wasserfallmodell nacheinander abgearbeitet, können aber mit darüberliegenden Phasen interagieren.

2.2.2 Kostenplanung

Die Gesamtkosten für das Projekt setzen sich aus Personal- sowie Dienstleistungskosten zusammen. Die Personalkosten sind von HR festgelegt und belaufen sich für den Arbeitgeber auf 11€ pro Stunde. Die angegebenen Dienstleistungskosten decken die Nutzung der GCP Infrastruktur ab, die für die Implementierung und das Testen des Programms notwendig waren. Der Stundensatz für die eingesetzte E2-Standardmaschinentypen (4 virtuelle CPUs und 16GB Arbeitsspeicher) ist der Preistabelle für die erwähnten Spezifikationen mit $\approx 0,18\$ / 0,16€$ zu entnehmen.

Personal	Stundenlohn	Stunden	Kosten
Auszubildender	11,00€	232	2552€
Google-Cloud	0,16€	142	22,72€
Summe			2574,72€

Tabelle 1 : Kostenplanung

3 Projektdurchführung

3.1 Vorbereitung

Um eine effiziente und transparente Verwaltung unserer Projektdokumentation zu gewährleisten, setzen wir auf ein Versionskontrollsystem. Hierfür erstellen wir ein Git-Repository. Git ist ein Tool zur Versionskontrolle, das es ermöglicht, Änderungen an unseren Dokumenten im Laufe der Zeit nachzuverfolgen und zu verwalten.

3.2 Definition

In der Projektdurchführung haben wir uns entschieden, die Yahoo Finance API als Datenquelle zu nutzen und ein Python Backend mit Flask zu entwickeln. Als Datenbank kommt SQLite zum Einsatz.

Yahoo Finance API: Die Yahoo Finance API bietet einen umfangreichen und leicht zugänglichen Datensatz zu Finanzinformationen. Sie stellt historische Kursdaten und diverse Finanzkennzahlen für eine Vielzahl von Unternehmen bereit. Die API ist gut dokumentiert und wird von einer großen Entwicklergemeinschaft genutzt, was die Integration in unser Projekt erleichtert.

Python und Flask: Python ist eine flexible Programmiersprache, die sich durch ihre Lesbarkeit und die große Auswahl an Bibliotheken auszeichnet. Flask, ein leichtgewichtiges Web-Framework für Python, ermöglicht es, einfache Webanwendungen zu entwickeln. Im Gegensatz zu Django ist es schlanker und wegen des geringen Umfangs schneller zu lernen.

Als Datenbank haben wir uns für SQLite entschieden, da diese für kleinere Projekte wie unseres ideal ist und sich leicht in Python-Anwendungen integrieren lässt. Des Weiteren ist SQLite kostenlos und Kenntnisse sind durch ein anderes Projekt bereits vorhanden.

Die technische Produktumgebung und die abgesprochenen Muss- sowie Wunschkriterien wurden im Pflichtenheft festgehalten.

3.3 Entwurf

Für die Konzeptionierung des Projekts haben wir eine detaillierte Modellierung durchgeführt. Use-Case-Diagramme dienten dazu, die funktionalen Anforderungen aus Benutzersicht zu erfassen und zu dokumentieren. Dabei wurden alle Use-Cases aller verfügbaren Seiten berücksichtigt. In dieser Dokumentation sind aus Platz- und Übersichtsgründen die zwei einflussreichsten Use-Cases mit den meisten Funktionalitäten enthalten (Abbildung 3: UseCase Startseite; Abbildung 4: UseCase User Depot). Diese Diagramme geben einen klaren Überblick über die von den Benutzern erwarteten Funktionen. Das ERM visualisiert die Entitäten und ihre Beziehungen zueinander. Auf Basis des ERM (Abbildung 1: ERM) konnten wir die Datenbanktabellen entwerfen. Ergänzend dazu haben wir ein Aktivitätsdiagramm (Abbildung 2: Aktivitätsdiagramm Login) erstellt, um eine Teilalgorithmik der Anwendung abzubilden. Diese Modellierungstechniken unterstreichen das Vorgehen nach dem Top-down-Prinzip.

3.4 Implementation

3.4.1 Entwicklungsumgebung

Die Einrichtung einer virtuellen Python-Umgebung und die Installation der erforderlichen Pakete waren entscheidende Schritte für die Implementation. Durch diese Maßnahmen besteht eine kontrollierte Umgebung, in der die Abhängigkeiten der Anwendung präzise verwaltet werden können. Dies minimiert das Risiko von Konflikten, die durch unterschiedliche Paketversionen oder globale Installationen entstehen können. Notwendig ist die Installation von `python3-venv`. Weitere Installationsschritte sind im Kapitel „Installation“ erwähnt (3.6) erwähnt.

3.4.2 Einrichtung der Datenbank

Das Fundament unserer Datenbankstruktur bildet die Datei `schema.sql`. Diese Datei fungiert als Gerüst und definiert die Tabellen, Spalten und Beziehungen, aus denen die Datenbank besteht. Die Methode `init_db_command()` erstellt eine Datenbank, sofern keine Datenbank vorhanden ist, indem es die `sqlite DB-API` verwendet und mit dem Keyword `detect_types=sqlite3.PARSE_DECLTYPES` die `schema.sql` einliest. Zu erwähnen ist, dass der Befehl `ON DELETE CASCADE` Löschanomalien verhindert, indem beim Löschen eines Datensatzes in einer Tabelle eine Kaskade von Löschvorgängen in anderen, verknüpften Tabellen ausgelöst wird. Des Weiteren werden durch den `init`-Befehl die Stammdaten der angebotenen Produkte in der Tabelle „Produkte“ eingetragen.

3.4.3 Verwendung von `yfinance`

Die Aktualisierung der Chartbilder erfolgt primär über die Funktion `yfinance.download()`, die eine Schnittstelle zur Yahoo Finance API bietet. Durch Übergabeparameter können spezifische Wertpapiere und Zeiträume ausgewählt werden. In Kombination mit der Visualisierungsbibliothek `plotly` können so historische Kursdaten visuell dargestellt werden. Für die Abfrage des aktuellen Kurses einzelner Wertpapiere wird hingegen die Funktion `yfinance.Ticker` eingesetzt. Diese Funktion ist effizienter für die Beschaffung einzelner Kursdatenpunkte und liefert die Informationen in einem strukturierten Format, das leicht weiterverarbeitet werden kann.

3.4.4 Aufbau der Webanwendung

In dem Projekt ist `Flask` die Kernkomponente, um die Webanwendung zu erstellen. Entscheidung war die Implementierung von Routen, die es uns ermöglicht haben, verschiedene URL-Pfade unserer Anwendung gezielt zu steuern und zu verarbeiten. Durch die Definition von Routen konnten wir spezifische Funktionen den jeweiligen URLs zuordnen. Die unterschiedlichen HTTP-Methoden wie `GET` und `POST` strukturieren die Webanwendung. Bei `GET`-Anfragen hat die Formularseite geladen, während die `POST`-Anfragen die übermittelten Daten verarbeiten und die Antwort zurückgegeben haben.

Über die Template-Engine `Jinja2` werden die HTML-Seiten dynamisch mit den Daten aus der jeweiligen Route befüllt. Der Datenaustausch zwischen Frontend und Backend erfolgt im `JSON`-Format, welches einem `Python-Dictionary` ähnelt. So konnte eine saubere Trennung zwischen Logik und Präsentation beibehalten werden.

An einem `Flask`-Objekt (`app`) kann die Methode `app.run()` aufgerufen werden, welche die Webanwendung zum Laufen bringt (in `Development` `localhost` IP-Adresse: `127.0.0.1`). Zusätzlich kann mit dem Keyword `ssl_context=('cert.pem', 'key.pem')` ein `SSL-Zertifikat` und der dazugehörige private Schlüssel übergeben werden. Sobald eine Anfrage eingeht, überprüft `app`, ob eine sichere Verbindung gewünscht wird (`HTTPS`). Ist dies der Fall, wird mit dem bereitgestellten `SSL-Zertifikat` und dem Schlüssel die Verbindung

verschlüsselt. Im Development wird der cert.pem und key.pem selbstsigniert, sodass andere Browser diese Verbindung nicht als sicher ansehen. In einer Produktionsumgebung müssten diese Dateien von einer Authentifikation signiert werden. Weitere Informationen stehen im Kapitel „Installation“ (3.6).

3.5 Testen

Um die Qualität der Anwendung sicherzustellen, wurde als erste Testmaßnahme ein Abnahmetest durchgeführt. Dabei wurde die Anwendung auf einem unabhängigen System nach Installationsanweisung (3.6) vorbereitet und anschließend ausgeführt. Die Anforderungen aus dem Projektantrag sowie Lastenheft wurden überprüft.

Durch Vorerfahrung haben wir uns dazu entschieden, das Python-Modul unittest zu verwenden, um Testfälle zu definieren und automatisiert laufen zu lassen. Diese Testfälle erzeugen beim Start eine eigene Testdatenbank, sodass die Tests unabhängig von der Produktion ausgeführt werden können. Dafür erbt die Testklasse von unittest.TestCase, sodass die Testklasse die Funktionalität von unittest enthält. Eine setup()-Methode kreiert eine Testumgebung. Mithilfe von Assertions (z.B. assertEquals, assertTrue) vergleichen wir das erwartete Ergebnis mit dem tatsächlichen Ergebnis der Funktion.

3.5.1 Komponententest

Der Komponententest dient dazu, einzelne isolierte Funktionen oder Methoden zu testen. In unserem Testbereich liegt die user_db_requests.py-Datei, die Schnittstelle der Flask.Routen und der Datenbank ist. Aufruf: `python3 -m unittest discover -s tests/unit_tests`

3.5.2 Integration Test

Integrationstests überprüfen, wie verschiedene Komponenten einer Anwendung zusammenarbeiten. Dafür werden GET und POST Request erstellt und an die Webanwendung gesendet. Ein POST Request übermittelt Informationen an die Routen im JSON-Format. Die response enthält ein Feedback, inwiefern der Request erfolgreich war. So können Login sowie Registrierung getestet werden. Der GET Request testet die Erreichbarkeit aller Routen. Für die Erreichbarkeitsprüfung des Depotbereichs ist ein eingeloggter User Voraussetzung. Aufruf: `python3 -m unittest discover -s tests/integration_tests`

3.6 Installation

Damit andere Menschen die Webanwendung nutzen können, muss die Anwendung auf einem Server aufgesetzt werden, der ständig online und von überall erreichbar ist.

Da der Quellcode in Python geschrieben ist, wird die Installation eines Python-Interpreters vorausgesetzt. Darüber hinaus müssen die benötigten Pakete und Module installiert werden. Die zu installierenden Pakete sind im requirements.txt File aufgelistet. Dabei besteht die Möglichkeit, die Installation in einer virtuellen Umgebung vorzunehmen, um ungewünschte Seiteneffekte zu vermeiden. Dies geht mit den Befehlen:

- `pip install virtualenv`
- `python3 -m venv`
- `source .venv/bin/activate`

Um eine verschlüsselte Verbindung zu gewährleisten, ist es zwingend notwendig, im Vorfeld authentifizierte Zertifikate (cert.pem) und einen private Key (key.pem) in das Verzeichnis (instance/certs) abzulegen. Der Anbieter Let's Encrypt bietet mit dem Programm Certbot eine kostenlose Möglichkeit an.

Als Nächstes muss die Datenbank mit ihren Stammdaten aufgesetzt werden:

- `flask --app app init-db`

Die Anwendung kann im Development mit Debug-Level und ohne vorhandene Zertifikate gestartet werden:

- `flask --app main.py --debug run`

Die Anwendung sollte mit diesem Befehl in Produktion aufgesetzt werden:

- `python3 -m app.main`

4 Fazit

4.1 Ist/Soll Vergleich

Während der Durchführung des Projekts gab es an wenigen Stellen Abweichungen zum Projektantrag. Der geplante zeitliche Ablauf hat sich leicht verschoben, jedoch ohne den gesamten Zeitrahmen zu verändern (Tabelle 3: Soll/Ist Zeitplanung). Die Implementation des Projekts hat durch eingetretene Probleme mit GitHub mehr Zeit in Anspruch genommen als geplant. Hauptursache waren Merge-Konflikte, `__pycache__` Dateien und unterschiedliche Versionen der Datenbank. Beim Testen und Probelauf lief die Durchführung schneller als geplant ab, sodass Zeit zurückgewonnen werden konnte.

4.2 Ausblick

Aufgrund der Abschlussprüfungen muss das Projekt TradeHub pausiert werden. Wir laden jeden dazu ein, mitzumachen und gemeinsam an diesem Projekt zu wachsen. Das GitHub-Repository ist öffentlich und für jeden frei zugänglich.

4.2.1 Weitere Funktionalitäten

Für eingeloggte User soll es möglich sein in dem Fenster Detailpage (Darstellung des Aktienkurses) direkt Aktien kaufen oder verkaufen zu können. Damit kann der Benutzer schneller handeln und spart sich so das Navigieren zum Ordermanager und das Eingeben des Aktienamens.

Eine vielversprechende Erweiterung für unser Projekt wäre die Integration eines SMTP-Servers. Dies würde es uns ermöglichen, E-Mails zu versenden und somit eine Vielzahl von Anwendungsfällen zu realisieren:

- **Passwort-Wiederherstellung:** Bei einem vergessenen Passwort könnte dem Nutzer automatisch eine E-Mail mit einem Link zum Zurücksetzen des Passworts zugesendet werden.
- **Bestätigung der Registrierung:** Neue Nutzer könnten eine E-Mail erhalten, um ihre E-Mail-Adresse zu bestätigen und den Registrierungsprozess abzuschließen.

4.3 Gewonnene Erkenntnisse

Die Arbeit an diesem Projekt war für uns eine wertvolle Erfahrung. Wir haben gelernt, wie wichtig es ist, auf die Stärken und Schwächen der anderen Teammitglieder einzugehen. Kommunikation war in unserem Fall ein wesentlicher Faktor, da jedes Teammitglied unterschiedliche Ansichten und Ideen hat. Wir hatten die Erkenntnis, dass nicht alles planbar und Flexibilität unverzichtbar ist.

5 Glossar

GCP: Die Google Cloud Platform, kurz GCP, ist eine umfassende Sammlung von Cloud-Computing-Diensten, die von Google bereitgestellt wird.

API: Application Programming Interface ist im Grunde eine Schnittstelle, die es verschiedenen Softwareanwendungen ermöglicht, miteinander zu kommunizieren.

ERM: Entity-Relationship-Modell. Es ist eine grafische Darstellung, die dazu dient, Datenstrukturen zu beschreiben.

Top-down: Durch die Abstraktion am Anfang erhält man einen klaren Überblick über das gesamte System und kann komplexe Probleme besser verstehen.

Stammdaten: Stammdaten liefern grundlegende Information und werden in der Zukunft wenig bis gar nicht geändert.

SSL: Secure Sockets Layer ist ein Sicherheitsprotokoll, das die Kommunikation über das Internet verschlüsselt.

Localhost: Diese Adresse wird als Loopback-Adresse bezeichnet und leitet Anfragen zurück an das eigene System. Keine anderen Nutzer können die Webanwendung erreichen.

IP-Adresse: Internet Protocol-Adresse ist eine numerische Kennung, die jedem Gerät in einem Netzwerk zugewiesen wird, um dessen Standort und Kommunikation zu ermöglichen.

SMTP-Servers: Simple Mail Transfer Protocol ist das zugrunde liegende Protokoll, das den Versand von E-Mails im Internet regelt. Es definiert die standardisierten Regeln und Verfahren, nach denen E-Mail-Nachrichten zwischen verschiedenen Mailservern übertragen werden.

6 Anhang

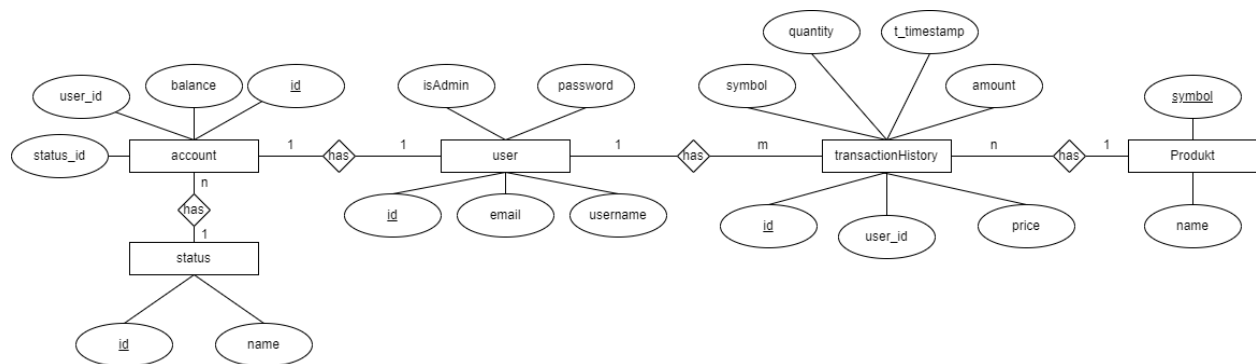


Abbildung 1: ERM

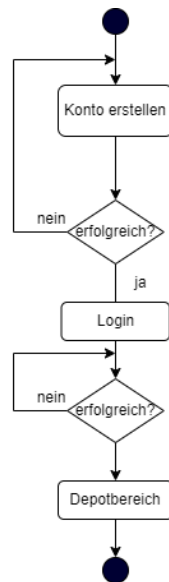


Abbildung 2: Aktivitätsdiagramm Login



Abbildung 3: UseCase Startseite



Abbildung 4: UseCase User Depot

Phase	Aufgabe	Cristian (Std.)	Danny (Std.)	Moritz (Std.)	Gesamtstunden pro Phase (Std.)
Planung	Projektantrag	5	7	6	18
Definition	Pflichtenheft	1	1	1	22
	Auswahl Schnittstellen	2	2	2	
	Auswahl Software	1	1	1	
	Auswahl Frameworks	1	1	1	
	Auswahl Bibliotheken	1	1	1	
	Erstellung Softwareplan	1	2	1	
Entwurf	Use-Case Diagramm	1	1	1	20
	Klassendiagramme	3	2	3	
	Aktivitätsdiagramm	2	3	2	
	ER-Modell	-	1	1	
Implementierung	Klassen	3	3	4	92
	Methoden	7	6	7	
	Datenbank aufsetzen, Kommunika- tion zum Backend, Definition der Queries	-	-	12	
	User Interface Design	30	-	-	
	API-Anbindung	-	20	-	
Tests	Testplanung, -vorbereitung und - spezifikation	13	13	14	50
	Komponententests	1	1	2	
	Integrationstests	1	1	1	
	Systemtests	1	1	1	
Dokumentation	Dokumentation	10	10	10	30
					232

Tabelle 2: Zeitplanung

Phase	Aufgabe	Cristian (Std.)	Danny (Std.)	Moritz (Std.)	Gesamtstunden pro Phase (Std.)
Planung	Projektantrag	5	7	6	18
Definition	Pflichtenheft	1	1	1	22
	Auswahl Schnittstellen	2	2	2	
	Auswahl Software	1	1	1	
	Auswahl Frameworks	1	1	1	
	Auswahl Bibliotheken	1	1	1	
	Erstellung Softwareplan	1	2	1	
Entwurf	Use-Case Diagramm	1	1	1	20/6
	Klassendiagramme	3/0	2/0	3/0	
	Aktivitätsdiagramm	2/0	3/0	2/1	
	ER-Modell	-	1	1	
Implementierung	Klassen	3	3	4	92/117
	Methoden	7	6/10	7	
	Datenbank aufsetzen, Kommunika- tion zum Backend, Definition der Queries	0/5	0/12	12/15	
	User Interface Design	30	-	-	
	API-Anbindung	-	20	0/1	
Tests	Testplanung, -vorbereitung und - spezifikation	13/5	13	14	50/39
	Komponententests	1/0	1	2	
	Integrationstests	1/0	1	1	
	Systemtests	1/0	1	1	
Dokumentation	Dokumentation	10/5	10/5	10/40	30
					232

Tabelle 3: Soll/Ist Zeitplanung

7 Quellen

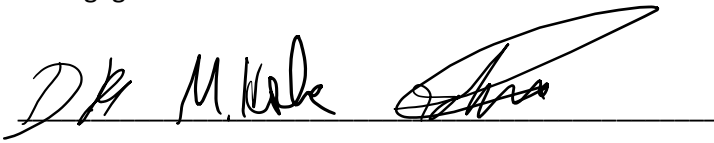
GCP-Preistabelle: <https://cloud.google.com/compute/all-pricing?hl=de>

Certbotbot: <https://certbot.eff.org/>

GitHub: <https://github.com/McLovin-81/trade-hub/tree/main>

8 Eigenständigkeitserklärung

Hiermit bestätigen wir, dass der vorliegende Projektantrag selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt wurden.



(20.09.2024, Unterschrift Auftragnehmer: Danny Binder, Moritz Heinke, Cristian Felipe Castillo-Barrero)

(20.09.2024, Unterschrift Auftraggeber)

Projektantrag

Wertpapiersimulation

Ausbildungsberuf

Fachinformatiker/-in Fachrichtung: Anwendungsentwicklung

Berufsschule

Brühlwiesenschule

Auszubildende

Danny Nicolay Walter Binder, Cristian Felipe Castillo-Barrero, Moritz Heinke

E-Mail-Adresse: moritz.heinke@deutsche-boerse.com

Betrieblicher Ansprechpartner: Jan Patrick Drehwald

E-Mail-Adresse: jan.patrick.drehwald@clearstream.com | Telefon: +49 69 211-1 41 41

Thema der Projektarbeit:

Entwicklung und Implementierung einer plattformunabhängigen Simulation eines Wertpapierdepots zum spielerischen Erlernen der Börsenumgebung.

1 Projektbeschreibung

Das Projektziel besteht in der Entwicklung einer Simulationsplattform zum spielerischen Erlernen von praxisnahem und lehrreichen Verständnis für den Handel mit Wertpapieren. Die Plattform ermöglicht es den Benutzern, mithilfe von virtuellem Kapital, Käufe und Verkäufe von Wertpapieren durchzuführen. Das Gesamtkapital des Benutzers wird in Echtzeit basierend auf den aktuellen Börsenkursen berechnet und angepasst. Darüber hinaus soll die Simulation die Dynamiken des echten Börsenhandels visuell abbilden.

Das Projekt ist erfolgreich abgeschlossen nach bestandenen Tests, Inbetriebnahme und Übergabe an den Kunden.

2 Projekttitel

Entwicklung und Implementierung einer plattformunabhängigen Simulation eines Wertpapierdepots zum spielerischen Erlernen der Börsenumgebung.

3 Durchführungszeitraum

Projektstart: 26.08.2024

Projektende: 20.09.2024

4 Ausgangssituation

4.1. Projektumfeld

Das Projekt wird innerhalb der Brühlwiesenschule, für die Lernfelder zehn bis zwölf, innerhalb des Schulblocks vom 26.08.2024 – 20.09.2024 durchgeführt. Innerhalb des Gruppenprojektes werden die bislang erlernten Fertigkeiten der letzten beiden Lehrjahre genutzt. Neue Konzepte, wie Model-View-Trennung, Barrierefreiheit und Plattformübergreifende Kompatibilität werden durch Eigeninitiative erlernt und angewandt. Per Gruppenentscheidung ist das Vorgehensmodell das erweiterte Wasserfallmodell. Das Projekt wird im folgenden Schulblock durch eine Präsentation und eine Live-Demo vorgestellt.

4.2. Anforderungen

Bei der Kontoerstellung müssen Benutzer einen Usernamen und ein Passwort angeben. Nach der erfolgreichen Registrierung wird jedem neuen Benutzer ein festgelegtes Startkapital gutgeschrieben. Im Rahmen der Kontoverwaltung haben Benutzer die Möglichkeit, ihr Passwort bei Bedarf zurückzusetzen und eine Anfrage auf Löschung stellen. Administratoren sind befugt, auf Anfrage, Konten zu resettet oder vollständig zu löschen.

In der Depotverwaltung können Benutzer Orders zum Kauf und Verkauf von Aktien erstellen und den aktuellen Wert ihres Depots jederzeit einsehen. Das zugewiesene Startkapital bildet dabei die Grundlage für alle weiteren Transaktionen. Benutzer haben außerdem die Möglichkeit, den aktuellen Marktwert bestimmter Aktien abzurufen, um Entscheidungen treffen zu können. Es gibt eine Bestenliste für die aktuelle Woche und eine separate für die Gesamte Zeit. Die Bestenliste zeigt die Nutzer mit den meisten gewinnen in absteigender Reihenfolge an. Bei einem Reset des Kontos wird der Nutzer auch aus den Bestenlisten herausgenommen, bis er das Handeln wieder beginnt.

5 Projektziel

Das Projektziel besteht in der Entwicklung der Simulationsplattform zum spielerischen Erlernen praxisnahem und lehrreichen Verständnis für den Handel mit Wertpapieren. Die Plattform ermöglicht es den Benutzern, mithilfe von virtuellem Kapital Käufe und Verkäufe von Wertpapieren durchzuführen. Das Gesamtkapital des Benutzers wird in Echtzeit basierend auf den aktuellen Börsenkursen berechnet und angepasst. Darüber hinaus soll die Simulation die Dynamiken des echten Börsenhandels abbilden und den Benutzern ein praxisnahes und lehrreiches Verständnis für den Handel mit Wertpapieren vermitteln.

Das Projekt ist erfolgreich abgeschlossen nach bestandenen Tests, Inbetriebnahme und Übergabe an den Kunden.

6 Zeitplanung

<i>Phase</i>	<i>Aufgabe</i>	<i>Cristian (Std.)</i>	<i>Danny (Std.)</i>	<i>Moritz (Std.)</i>	<i>Gesamtstunden pro Phase (Std.)</i>
<i>Planung</i>	Projektantrag	5	7	6	18
<i>Definition</i>	Pflichtenheft	1	1	1	22
	Auswahl Schnittstellen	2	2	2	
	Auswahl Software	1	1	1	
	Auswahl Frameworks	1	1	1	
	Auswahl Bibliotheken	1	1	1	
	Erstellung Softwareplan	1	2	1	
<i>Entwurf</i>	Use-Case Diagram	1	1	1	20
	Klassendiagramme	3	2	3	
	Aktivitätsdiagramm	2	3	2	
	ER-Modell	-	1	1	
<i>Implementierung</i>	Klassen	3	3	4	92
	Methoden	7	6	7	
	Datenbank aufsetzen, Kommunikation zum Backend, definition der Queries	-	-	12	
	User Interface Design	30	-	-	
	API-Anbindung	-	20	-	
<i>Tests</i>	Testplanung, -vorbereitung und - spezifikation	13	13	14	50
	Komponententests	1	1	2	
	Integrationstests	1	1	1	
	Systemtests	1	1	1	
<i>Dokumentation</i>	Dokumentation	10	10	10	30
					232

7 Anlagen

Wireframes des Projekts

8 Präsentationsmittel

Vorhanden: Präsentationsfläche, Beamer

Mitgebracht: Präsentter, Notebook

9 Eigenständigkeitserklärung

Hiermit bestätigen wir, dass der vorliegende Projektantrag selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt wurden.



(29.08.2024, Unterschrift Auftragnehmer: Danny Binder, Moritz Heinke, Cristian Felipe Castillo-Barrero)

(29.08.2024, Unterschrift Auftraggeber)