Rīgas Tālmācības vidusskola

**Piekļuves darba programmēšanā**

**dokumentācija**

Darba autors: Kristofers Jurevics

**SATURA RADĪTĀJS**

lpp.

1. Problēmas izpēte un analīze ............................................................................................... 3
2. Programmatūras prasību specifikācija ................................................................................ 4
3. Produkta vizuālā un datu struktūras skice........................................................................... 5
4. Programmēšanas valoda un izstrādes vide ......................................................................... 7
5. Programmatūras izstrādes plāns ......................................................................................... 8
6. Atkļūdošanas pārskats pēc izstrādes .................................................................................. 10
7. Lietotāja ceļvedis ............................................................................................................... 11
8. **Problēmas izpēte un analīze**

Sākotnēji zināju, ka vēlos savu programmatūras projektu saistīt ar kādu no nozarēm, kuras labi pārzinu un kuras man personīgi interesē. Uzskatu, ka, strādājot pie projekta, kas saistīts ar sev tuvām tēmām, ir iespējams nodrošināt augstāku kvalitāti un lielāku iesaisti, kā arī izstrādāt risinājumu detalizētāk un praktiskāk. Man tuvas nozares ir futbols, videospēles un finanšu tirgus.

Lai noteiktu, kāda veida projekts varētu būt vislietderīgākais, veicu izpēti, izmantojot pašnovērtējumu un salīdzinošo analīzi, izvērtējot, kādi rīki ir pieejami katrā no man interesējošajām nozarēm un kādas ir to stiprās un vājās puses.

Izskatīju vairākas idejas:

* futbola spēļu statistikas izvērtēšanas un analīzes sistēmu,
* zināšanu pārbaudes spēli videospēļu tematikā,
* ekonomisko ziņu apkopošanas un attēlošanas mājaslapu.

Neviena no šīm idejām tomēr nelikās pietiekami specifiska vai noderīga man kā lietotājam. Tāpēc turpināju izpēti, balstoties uz jautājumu, kas man personīgi pietrūkst ikdienā, darbojoties šajās nozarēs un kā es pats varētu izmantot šo programmu reālajā dzīvē.

Šī analīze ļāva nonākt līdz konkrētai problēmai: tirdzniecības jeb “trading” žurnāla trūkumam, kas būtu īpaši pielāgots finanšu tirgus dalībniekiem, treideriem. Līdz šim savus darījumus biju pierakstījis vienkāršā piezīmju žurnālā, kas radīja vairākas problēmas:

* informācija nebija attiecīgi strukturēta,
* trūka iespējas pievienot konkrētas tirdzniecības detaļas
* datu analīze bija praktiski neiespējama, jo nebija automātisku statistikas apkopošanas rīku,
* vizuāli žurnāls nebija pārskatāms un nespēja attēlot darījumu rezultātus dinamiski.

Kvalitatīva darījumu dokumentēšana finanšu tirgū ir ārkārtīgi svarīga, lai varētu gūt noderīgu informāciju par savām tirdzniecības darbībām. Ir pierādīts, ka regulāra žurnāla aizpildīšana palīdz treideriem noteikt, kuras stratēģijas strādā, kādas kļūdas tiek atkārtotas, un kā uzlabot risku pārvaldību. Tāpat arī atmiņa nav uzticams rīks – pēc laika nav iespējams precīzi atcerēties katru darījumu, tāpēc ir svarīgi, lai būtu iespēja nepieciešamības gadījumā atgriezties pie konkrēta ieraksta.

Ņemot vērā, ka finanšu tirgū šobrīd iesaistās arvien vairāk cilvēku pateicoties gan pieejamībai no jebkuras viedierīces, gan arī kriptovalūtu popularitātei, pieprasījums pēc šāda specializēta rīka tikai pieaug.

Izvērtējot visas iespējamās idejas, pieņēmu lēmumu izstrādāt trading žurnāla aplikāciju, kas būtu paredzēta tieši treideriem un nodrošinātu gan ērtu darījumu pierakstu, gan analītisku pārskatu par to rezultātiem.

1. **Programmatūras prasību specifikācija**

*Risinājuma mērķauditorijas izvēle un tās raksturojums*

Izstrādājamās programmatūras mērķauditorija ir finanšu tirgus aktīvie dalībnieki – treideri, kuri regulāri veic darījumus finanšu tirgos (Forex, akcijas, kriptovalūtas u.c.) un meklē efektīvu veidu, kā dokumentēt un analizēt savu darbību.

Tipiski šie lietotāji ir:

* pašmācībā apguvuši tirgus pamatus vai darbojas individuāli bez finanšu institūciju starpniecības;
* ieinteresēti savas stratēģijas pilnveidošanā, izmantojot statistiku;
* informēti par tirgus psiholoģijas un disciplīnas nozīmi;
* tehniski zinoši un novērtē modernas un specifiskas lietotnes.

Šie lietotāji nereti izmanto vispārīgus rīkus (Excel, OneNote, papīra žurnālus), kas nespēj nodrošināt vajadzīgo vizualizāciju, statistiku vai datu analīzi. Tāpēc šī programmatūra kalpo kā specializēts risinājums, kas paredzēts tieši treideriem un pielāgots viņu specifiskajām vajadzībām.

*Programmatūras tās funkciju apraksts*

**Lietotāja pārvaldība:**

* reģistrācija un pieteikšanās -Epasts un parole (jābūt vismaz 8 rakstzīmes garai);
* paroļu droša uzglabāšana ar kriptogrāfias šifrēšanu.

**Darījumu reģistrēšana un pārvaldība:**

* darījuma ievades forma: datums, nosaukums, pozīcijas tips, peļņa/zaudējums, risks, pārliecība, komentāri;
* dzēšana, ja nepieciešams;
* datu validācija.

**Statistika un analītika:**

* kopējā peļņa/zaudējumi;
* uzvaras procenti;
* vidējā peļņa/zaudējums;
* lielākā peļņa/zaudējums;
* kopējais tirgojumu daudzums;
* peļņas pret zaudējuma attiecība;
* nosaukums (aktīvam, crypto vai valūtai) ar kuru visvairāk darījumi ir veikti.

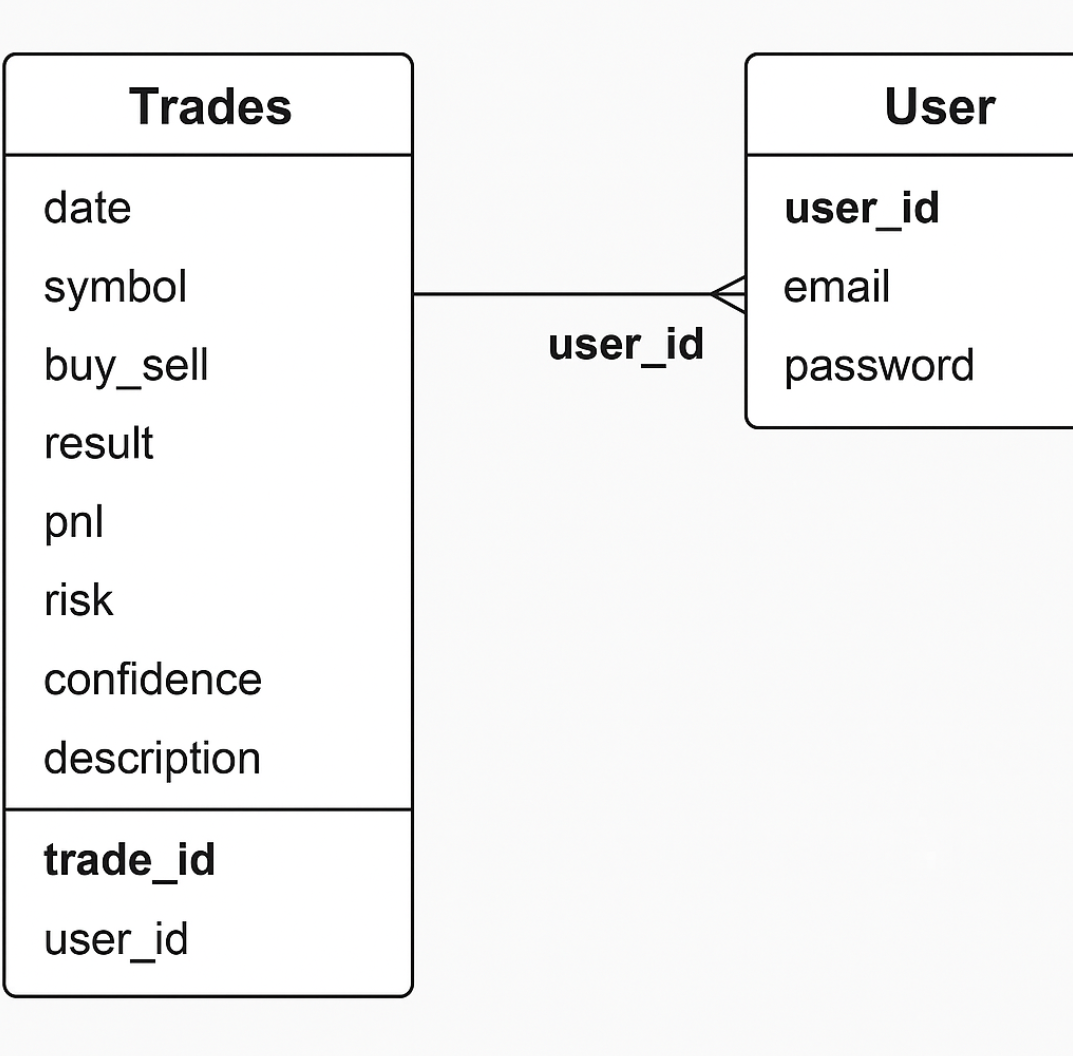
**Ekonomisko ziņu integrācija (API):**

* automātiska jaunāko ekonomisko ziņu ielāde no ārēja API;
* tiek attēlots ziņu nosaukums, ietekmējošā valūta un datums.

**Datubāze (sqlite3):**

Programmatūrai datubāzē būs divas sava starpā saistītas tabulas – viena “User”, kas saturēs informāciju par lietotāju un otra “Trades”, kas saturēs informācija par veiktajiem tirgojumiem. Tabulas tiks saistītas caur “user\_id”, kas automātiski tiek padots katram profilam, kad tas tiek izveidots skaitļa formā, un, kad lietotājs pierakstās sistēmā, tas tiek saglabāts sesijas laikā, lai varētu ievietot tabulā “Trades” kopā ar tirgojuma datiem.

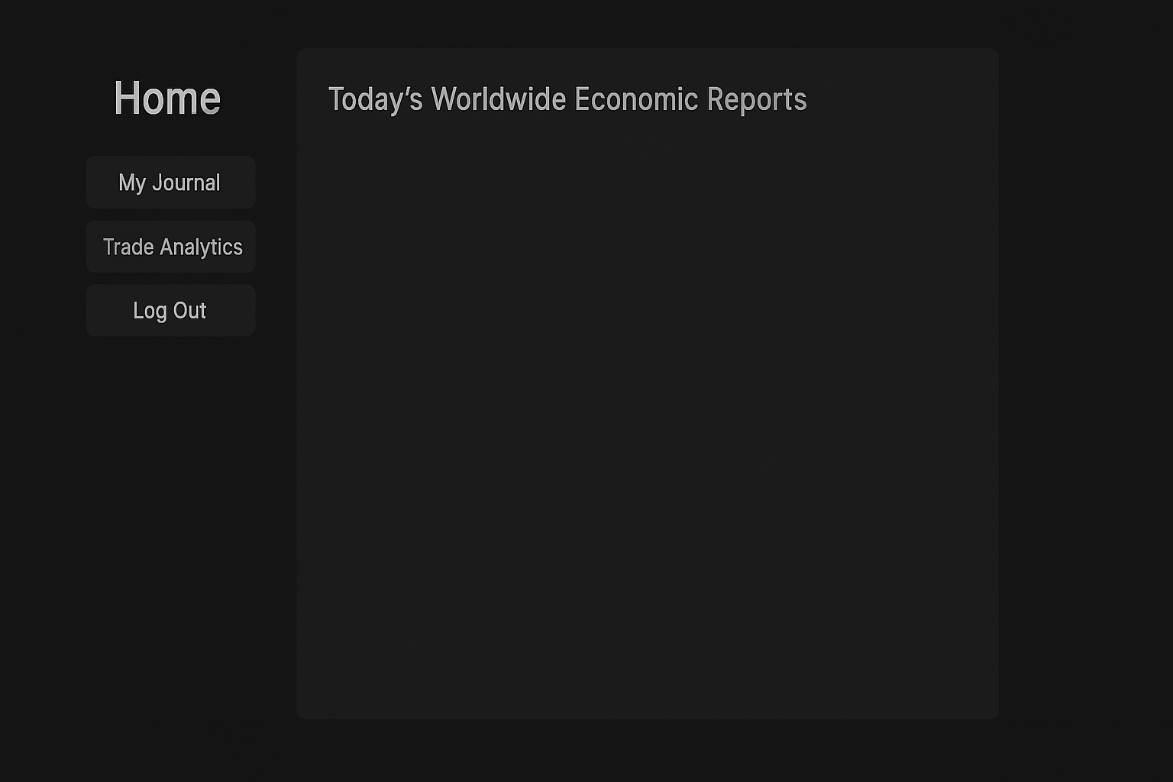
Datu bāzes modelis



1. **Produkta vizuālā un datu struktūras skice**

Programmatūra vizuāli tiks veidota tumšā un profesionālā stilā,– tas ir vizuāli mierīgs, un rada koncentrētības sajūtu. Tumšais dizains nodrošina labāku kontrastu datu attēlošanai, kas ir būtiski darbā ar skaitļiem un analīzi. Turklāt šāds vizuālais noformējums atbilst arī profesionālai videi, kurā vizuālā vienkāršība un struktūra ir svarīgāka par liekām krāsām un elementiem. Lietotāji sagaida mūsdienīgu un nopietnu izskatu, kas atspoguļo programmatūras funkcionalitāti un tās pielietojumu finanšu tirgos. Programmatūras lietotāja saskarnes teksts tiks veidots angļu valodā, jo lielākā daļa ar finanšu tirgiem un tirdzniecību saistīto terminu tiek visbiežāk lietoti tieši angļu valodā – gan profesionālajā vidē, gan izglītības materiālos, gan arī pašās tirdzniecības platformās. Tas nodrošina labāku saprotamību starptautiskai auditorijai un samazina tulkošanas neprecizitātes, kas šajā nozarē var būt ļoti būtiskas.

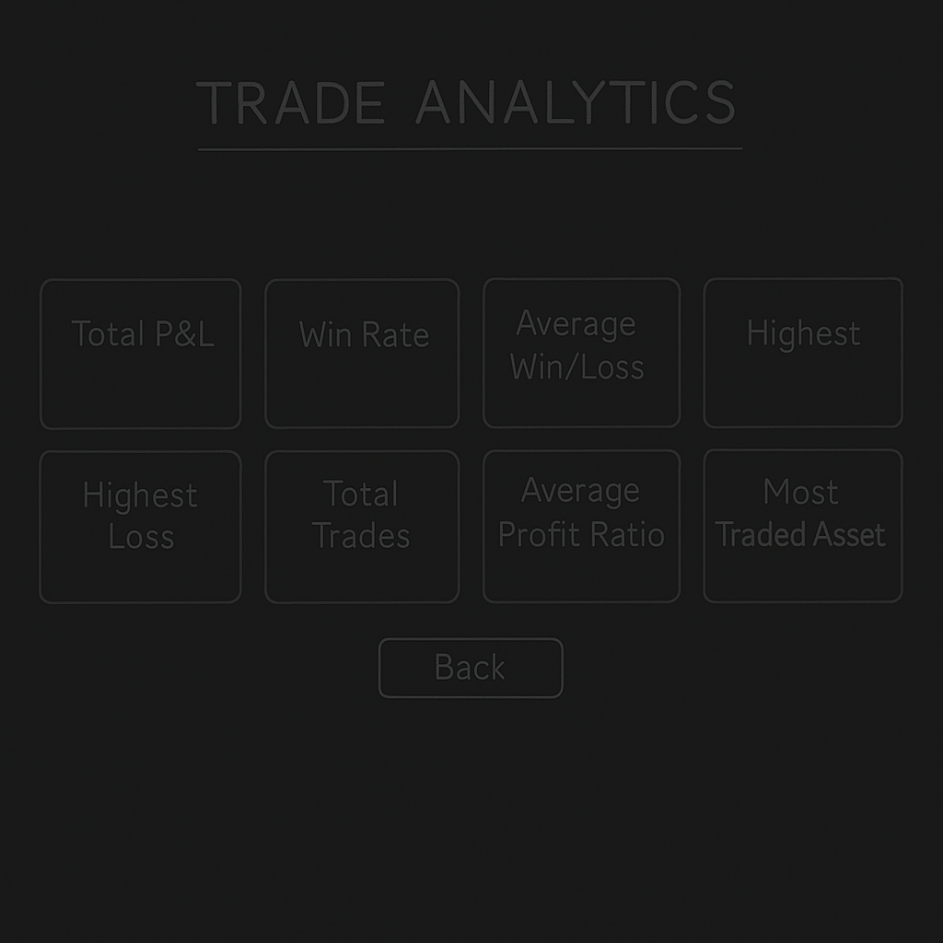
Programatūras sākuma lapas skice



Žurnāla lapas skice



Analītikas lapas skice



1. **Programmēšanas valoda un programmatūras izstrādes vide**

Programmatūras izstrādei izvēlēta Python programmēšanas valoda kā galvenā loģikas un servera puses valoda, savukārt HTML un CSS tika izmantoti lietotāja saskarnes (frontend) izstrādei.

Python tika izvēlēta, jo piedāvā daudz noderīgu, kas ļauj ērti strādāt ar datubāzēm, vizualizēt datus un izveidot API savienojumus. Python arī ļauj viegli apstrādāt statistiku, kas ir būtiska šāda veida žurnāla sistēmā, kur treideriem svarīga ir piekļuve pārskatāmam un analizējamam tirdzniecības datu apkopojumam.Man arī ir salīdzinoši lielākā pieredze ar python.

HTML un CSS tika izvēlēti lietotāja interfeisa veidošanai, jo tie ir industrijas standarti, kuri ļauj veidot pārskatāmu, strukturētu un profesionālu saskarni. HTML veido lapas struktūru, savukārt CSS nodrošina stilu un dizaina pielāgošanu.

Šī projekta izstrādei tiks izmantots Flask ietvars kopā ar SQLite3 datubāzi. Šī izvēle ir piemērota, ņemot vērā projekta mērogu, specifiku un izstrādes mērķi – izveidot funkcionālu, vieglu un lokāli darbināmu treidinga žurnāla lietotni ar iespējām pierakstīt un analizēt finanšu tirgus darījumus.

Flask ir viegls, elastīgs un populārs Python ietvars, kas ļauj ātri izveidot tīmekļa lietotnes, nodrošinot labu kontroli pār saskarnes izkārtojumu, funkcionalitāti un datu apstrādi. Tas ir lieliski piemērots nelieliem līdz vidējiem projektiem, kur nepieciešama vienkārša un stabila aizmugures loģika.

SQLite3 tika izvēlēta kā datubāze, jo tā ir viegla, uz failu balstīta sistēma, kas neprasa sarežģītu servera konfigurāciju. Tā ir piemērota gadījumiem, kad dati tiek glabāti lokāli vai projekta sākumstadijā – piemēram, lietotāju reģistrācijas dati, darījumu ieraksti un statistikas rezultāti.

Tādējādi programmatūras izstrāde balstīsies uz šādiem tehniskajiem pamatprincipiem:

* Backend loģika: Python + Flask
* Datu glabāšana: SQLite3
* Frontend saskarne: HTML, CSS (tumšs, profesionāls dizains)

Šāds risinājums nodrošinās gan efektīvu prototipa izstrādi, gan iespēju viegli paplašināt funkcionalitāti nākotnē, piemēram, pārejot uz sarežģītāku datubāzi (piemēram, PostgreSQL) vai integrējot frontend rīkus, kā React vai JavaScript interaktivitātei**.**

1. Lietotāja autentifikācija -Reģistrācijas un pieslēgšanās funkcijas ar paroles šifrēšanu un sesiju pārvaldību.
2. Sākuma skats – Pogas uz žurnālu un analītikas skatu uni ziņu API integrācija, aktuālās ekonomiskās ziņas tiek ielādētas pēc lietotāja pieslēgšanās
3. Tirdzniecības žurnāls - Kalendārs ar katras dienas rezultātu atspoguļošanu. Dienām piesaistīti darījumi.
4. Jauns darījums - Formas ievade, datu validācija un saglabāšana datubāzē
5. Analītika – Lietotāja statistikas par pievienotajiem darijumiem

Kā izstrādes vide izvēlēta Visual Studio Code, jo tā ir viegli pārskatāma un ir pieredze darbā ar to. Github arī tiks izmantots versiju kontrolei.

1. **Programmatūras izstrādes plāns**

Izstrādājot tirdzniecības žurnāla lietotni, nepieciešams izvēlēties piemērotāko programmatūras izstrādes modeli. Pastāv vairāki modeļi, taču visbiežāk izmantotie ir ūdenskrituma modelis un ātrās prototipēšanas modelis jeb Agile.

Ūdenskrituma modelis paredz, ka visa izstrādes gaita tiek sadalīta vairākos secīgos posmos, kur katrs nākamais posms tiek uzsākts tikai pēc iepriekšējā pabeigšanas. Šajā modelī visas prasības tiek noteiktas jau projekta sākumā un netiek mainītas procesa laikā.

Savukārt, Agile modelis balstās uz ciklisku un elastīgu izstrādes pieeju, kur lietotājam tiek rādīti prototipi un pēc katra cikla iespējams veikt izmaiņas. Tas ļauj pakāpeniski precizēt sistēmas funkcionalitāti, bet apgrūtina precīzu resursu plānošanu, jo nav iepriekš zināms, cik daudz izmaiņu būs nepieciešams veikt.

Tā kā tirdzniecības žurnāla sistēmai ir konkrēti un jau zināmi mērķi un šīs prasības nav paredzēts mainīt izstrādes laikā, piemērotākais ir tieši ūdenskrituma modelis. Šī modeļa izmantošana ļauj strukturēti plānot katru posmu un nodrošināt kvalitatīvu, pakāpenisku izstrādi bez pārsteigumiem.

Vienkāršots izstrādes plāns (pēc ūdenskrituma modeļa):

1. Prasību analīze.

Šajā posmā tiek noteiktas visas sistēmas funkcionalitātes prasības – ko lietotājs varēs darīt, kādi dati tiks apstrādāti, kā notiks autentifikācija un kā integrēsies ārējais ekonomisko ziņu avots. Rezultātā tiek izveidota prasību specifikācija.

1. Sistēmas projektēšana.

Tiek izstrādāts sistēmas arhitektūras un datu modelis – tiek noteikts, kādi datubāzes lauki būs nepieciešami, kā tiks organizētas programmas daļas, kā izskatīsies lietotāja saskarne. Šeit arī veidojas skices un struktūras apraksti.

1. Izstrāde jeb programmēšana.

Notiek kodēšana – tiek realizēta lietotāju sistēma, tirdzniecības darījumu reģistrācija un pārvaldība, API integrācija ar ekonomiskajām ziņām, statistikas attēlošana un lietotāja saskarne.

1. Testēšana.  
   Šajā posmā tiek veikta gan vienībtestēšana, gan integrācijas pārbaudes, lai pārliecinātos, ka visas sistēmas daļas strādā korekti, dati tiek pareizi saglabāti, un lietotāja darbības rada gaidīto rezultātu. Atklātās kļūdas tiek labotas.
2. Ieviešana.  
   Projekts tiek prezentēts pasūtītājam.
3. Uzturēšana.  
   Ja sistēmas lietošanas laikā parādās kādas kļūdas vai nepieciešami sīki uzlabojumi, tie tiek veikti uzturēšanas fāzē. Tā kā prasības sākumā jau bija fiksētas, izmaiņas šajā posmā tiek veiktas tikai kļūdu labojumu vai nelielu pielāgojumu formā.
4. **Atkļūdošanas pārskats pēc programmatūras izstrādes**

Pēc tirdzniecības žurnāla programmatūras izstrādes tika identificētas divas būtiskas problēmas, kuras tika analizētas un īstermiņā atrisinātas.

*1.problēma* ar ekonomisko ziņu API

Konstatēts, ka izmantotajam ekonomisko ziņu API ir ierobežojums – pieprasījumu var veikt tikai vienu reizi piecās minūtēs. Šis ierobežojums radīja problēmas, jo API dati tika attēloti mājaslapas sākumlapā katru reizi, kad lietotājs tai piekļuva. Sākotnēji tas neradīja traucējumus, bet, ja lietotājs no kādas citas lapas (piemēram, žurnāla) atgriezās uz sākumlapu mazāk nekā piecu minūšu laikā, API pieprasījums netika apstrādāts, un dati netika atgriezti.

Risinājums:

API pieprasījuma maršruts tika mainīts. Tā vietā, lai pieprasījums tiktu izsaukts katru reizi pie sākumlapas ielādes (/home maršruts), tas tagad tiek izpildīts tikai pieteikšanās brīdī – kad lietotājs ievada pareizu e-pastu un paroli. Saņemtie API dati tiek saglabāti Flask session mainīgajos, un tiek izmantoti visas sesijas laikā, tādejādi novēršot atkārtotus pieprasījumus īsā laika periodā.

Nākotnes plāns:

Šis risinājums samazina, bet pilnībā neizslēdz API ierobežojuma ietekmi. Ilgtermiņā problēma tiks atrisināta, ja projektam tiks piešķirts lielāks budžets, kas ļaus iegādāties API maksas versiju ar neierobežotu pieprasījumu skaitu.

*2.problēma* ar tirdzniecības analītiku

Konstatēta kļūda analītikas funkcionalitātē – ja lietotājs vēl nebija reģistrējis vismaz vienu darījumu ar peļņu un vienu ar zaudējumu, tad analītikas aprēķinu laikā notika Python matemātiska kļūda (piemēram, dalīšana ar nulli, aprēķinot vidējo zaudējumu utt.).

Risinājuma izvērtēšana:

* 1.variants:

Pārbaudīt, vai lietotājam ir vismaz viens darījums ar peļņu un viens ar zaudējumu. Ja šie nosacījumi nav izpildīti, lietotājs saņem paziņojumu ar skaidrojumu par to, kādi dati ir nepieciešami, lai analītika tiktu parādīta.

* 2.variants:

Veikt pārbaudes pie katra konkrētā analītikas rādītāja aprēķina un, ja dati nav pieejami, atstāt attiecīgo lauku tukšu.

Izvēlētais risinājums:

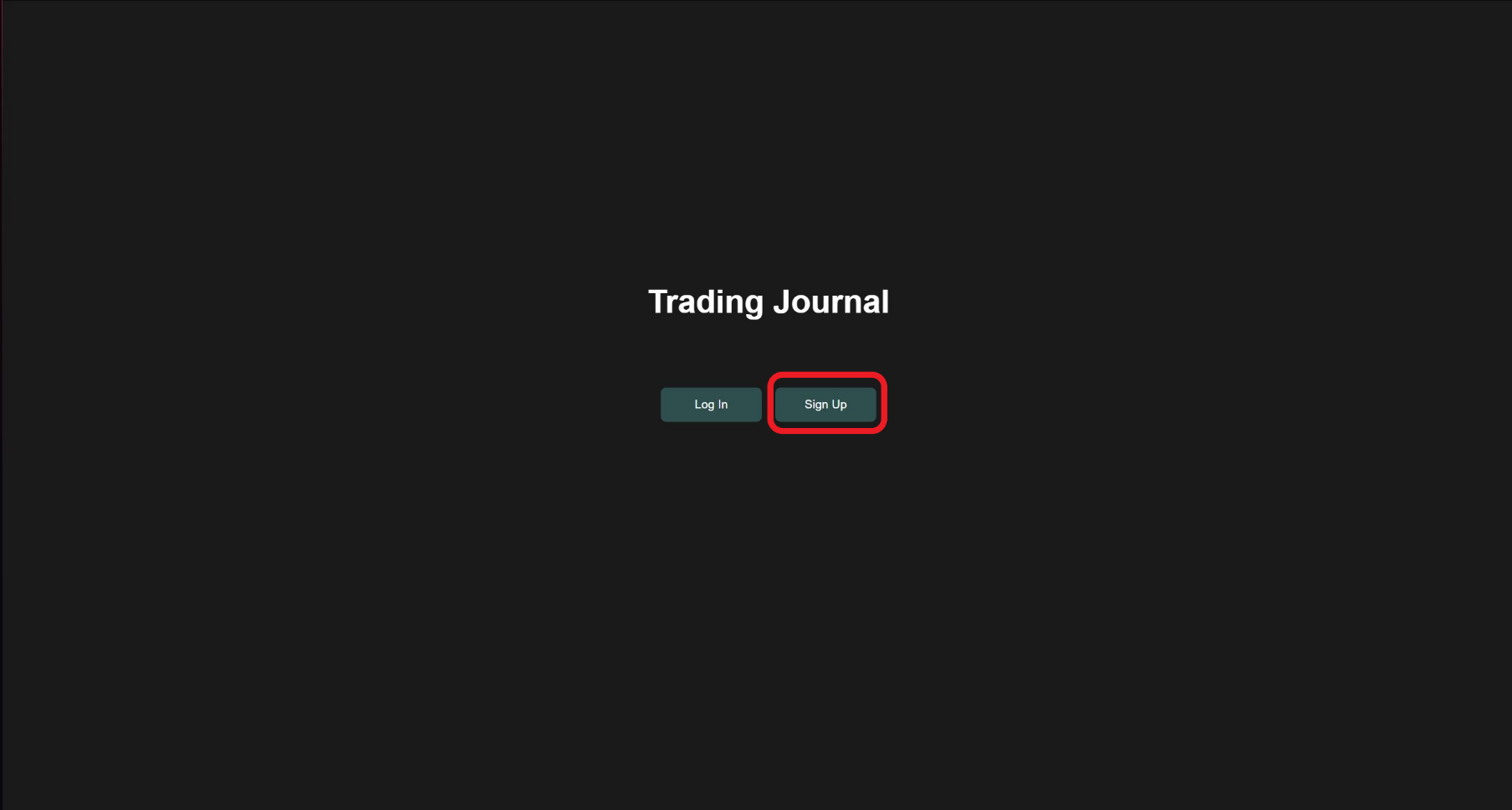
Izvēlēta pirmā pieeja, jo rādīt tikai daļēju analītiku vai atstāt laukus tukšus būtu maldinoši un nepilnīgi. Uzticamai analīzei ir būtiski, lai būtu pieejama informācija gan par peļņu, gan zaudējumiem. Prasība pēc vismaz viena peļņas un viena zaudējuma darījuma ir minimāla un viegli izpildāma.

Veiktie atkļūdošanas darbi atrisināja divas nozīmīgas problēmas sistēmas sākotnējā versijā. Abiem gadījumiem tika izvēlēti efektīvi īstermiņa risinājumi, vienlaikus apzinoties, ka ilgtermiņā nepieciešami uzlabojumi – īpaši API darbības optimizēšanā, ko varēs īstenot ar papildus finansējumu. Citas kļūdas sistēmā netika konstatētas.

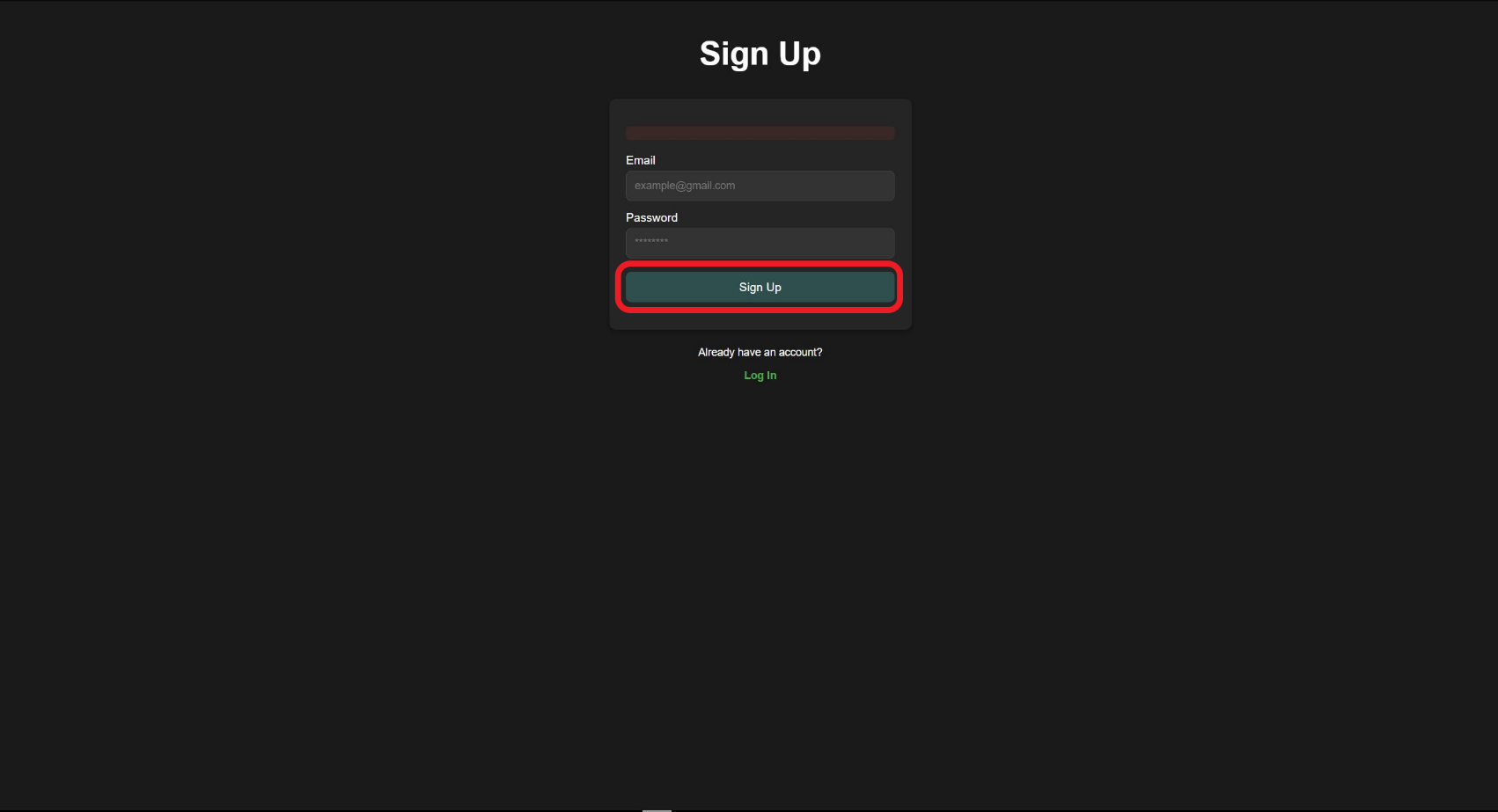
**7. Lietotāja ceļvedis**

*Lietotāja konta izveide/pieslēgšanās sistēmai*

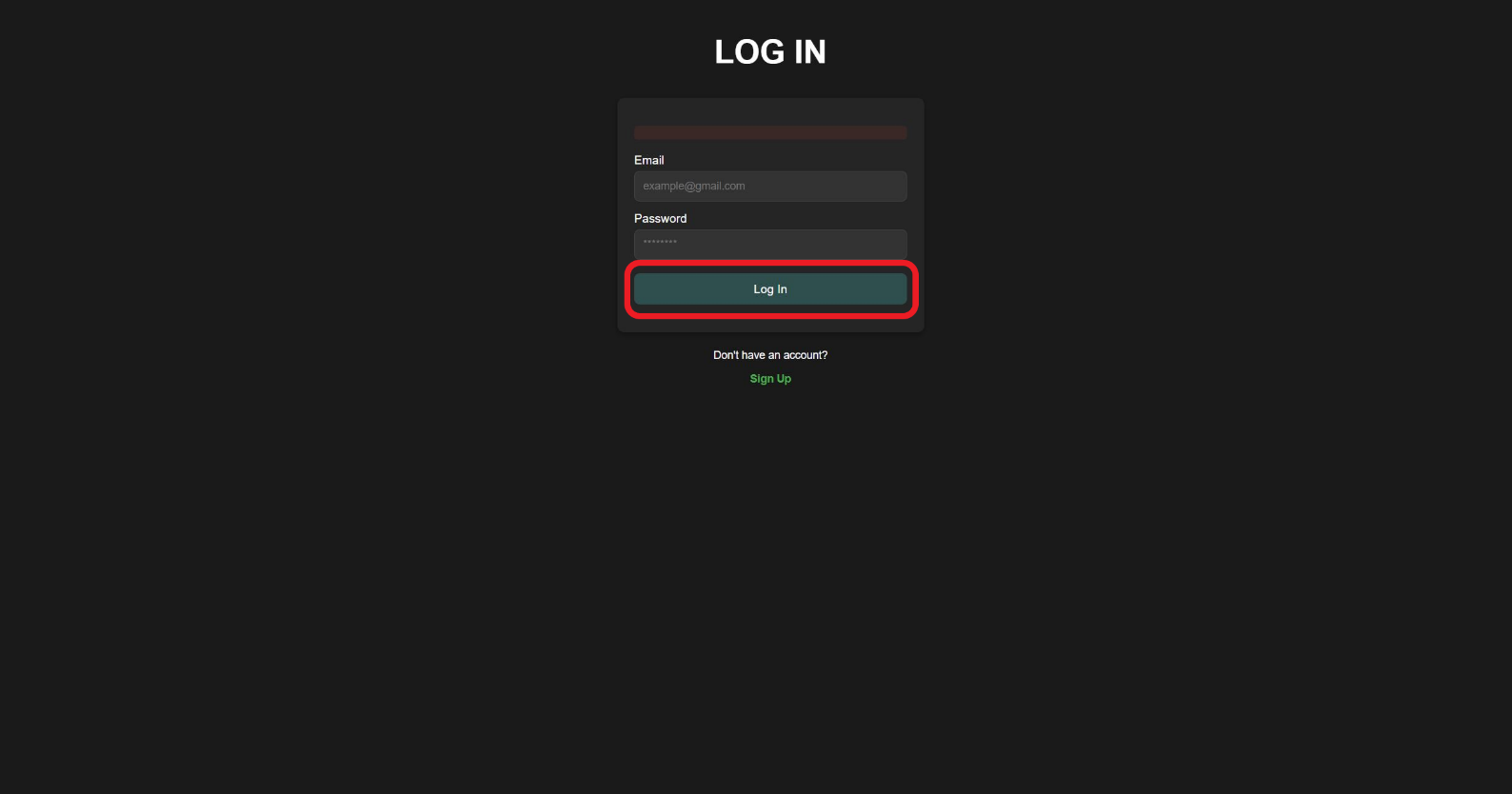
Nonākot sākuma skatā, lietotājs spiež uz ”Sign Up”, lai izveidotu profilu –

****

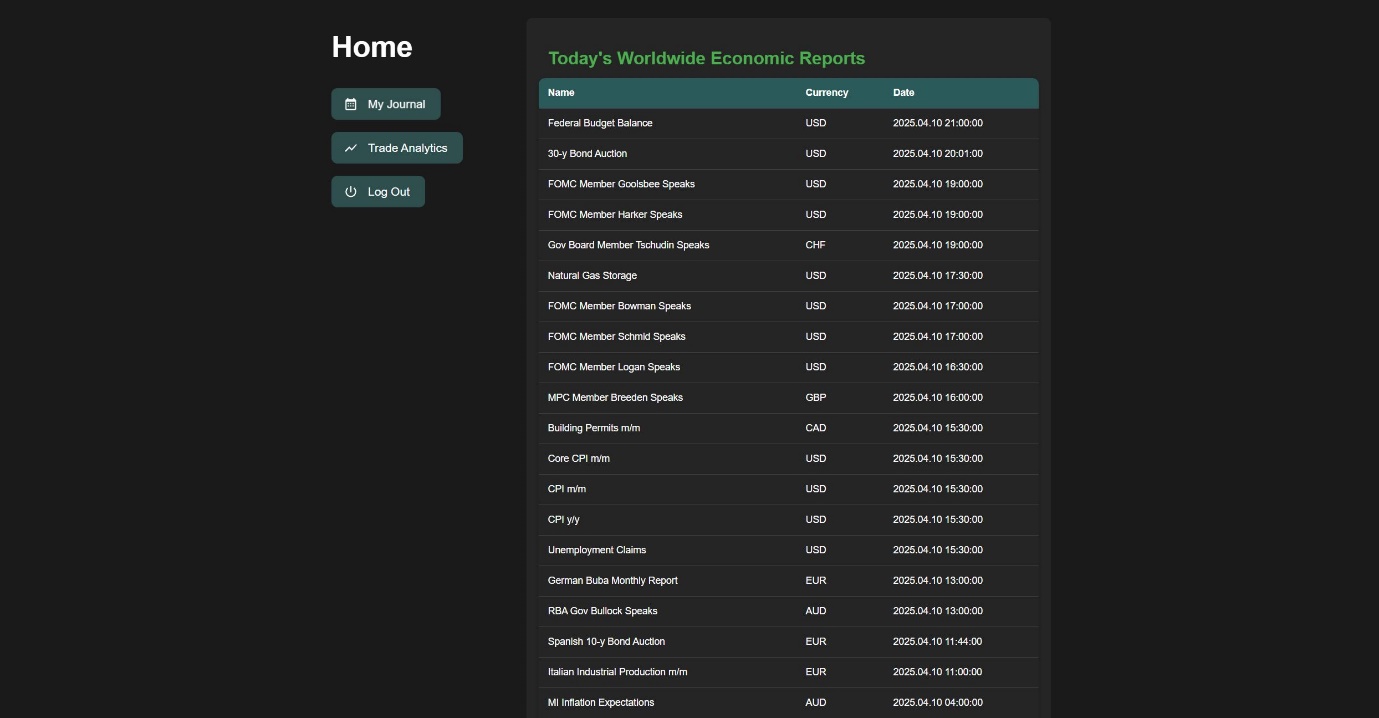
Nonākot reģistrēšanās skatā, lietotājs ievada sava e-pasta adresi un paroli, kas būs turpmāk jāizmanto, lai pieslēgtos sistēmai. E-pasta un paroles ievadlauki ir reģistrjūtīgi - tos aizpildot, ir jāņem vērā lielie un mazie burti. E-pasta un paroles ievadlauki ir obligāti aizpildāmi. Pēc piekļuves datu ievades lietotājam jānospiež poga ”Sign Up” –

****

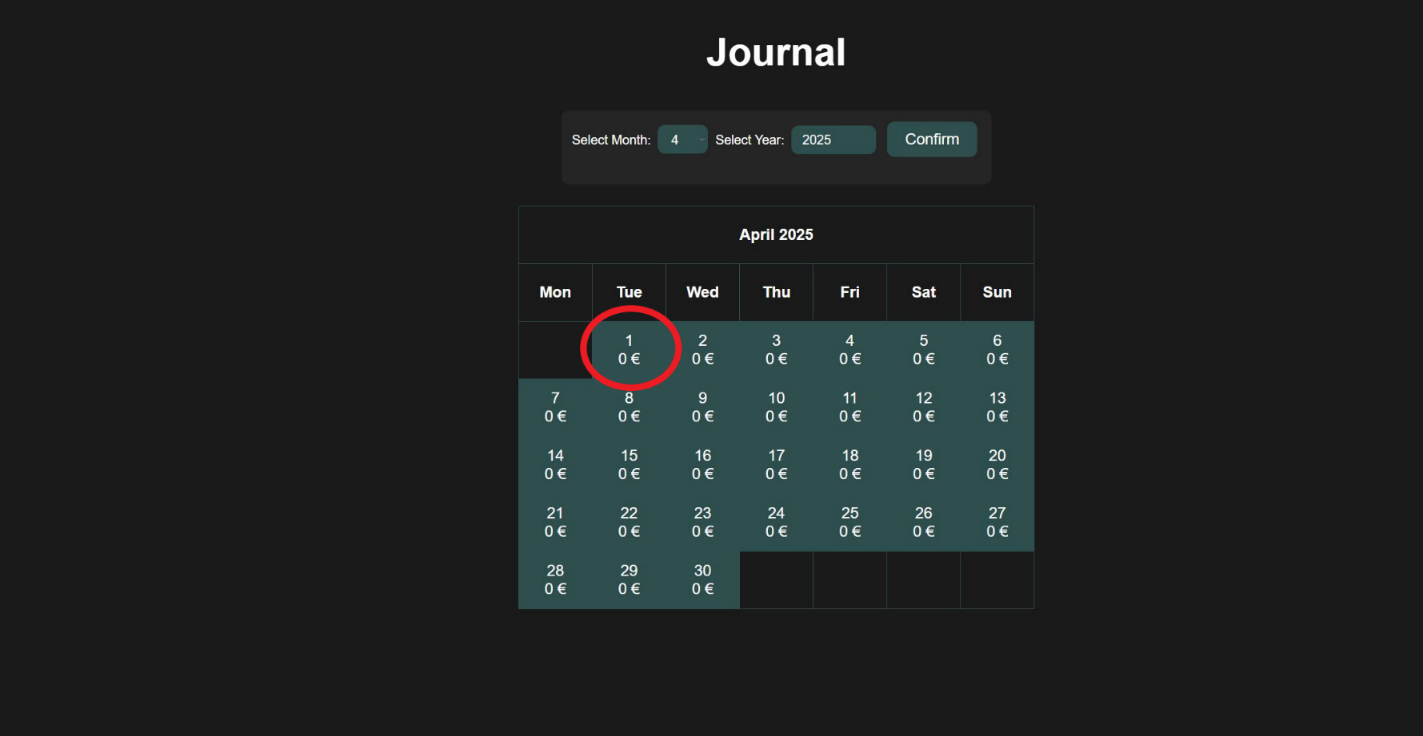
Lai izveidotu kontu, var lietot jebkuru savu e-pastu, un parolei jābūt vismaz 8 rakstzīmes garai, un parole tiks aizsargāta ar kriptogrāfijas šifrēšanu. Ja aizpildītie lauki būs atbilstoši prasībām, lietotājs tiks maršutēts uz pieteikšanās skatu, kur būs tikko izveidotā profila detaļas. Kad tās ir ievadītas, jāspiež uz “Log In” lai ieietu sistēma –



Pēc veiksmīgas pieteikšanās lietotājs nokļūst pasaules ekonomisko ziņu un informācijas lapā un arī pogām, kas aizved vai nu uz žurnālu, tirgojumu analītiku vai iziešanu no sistēmas –

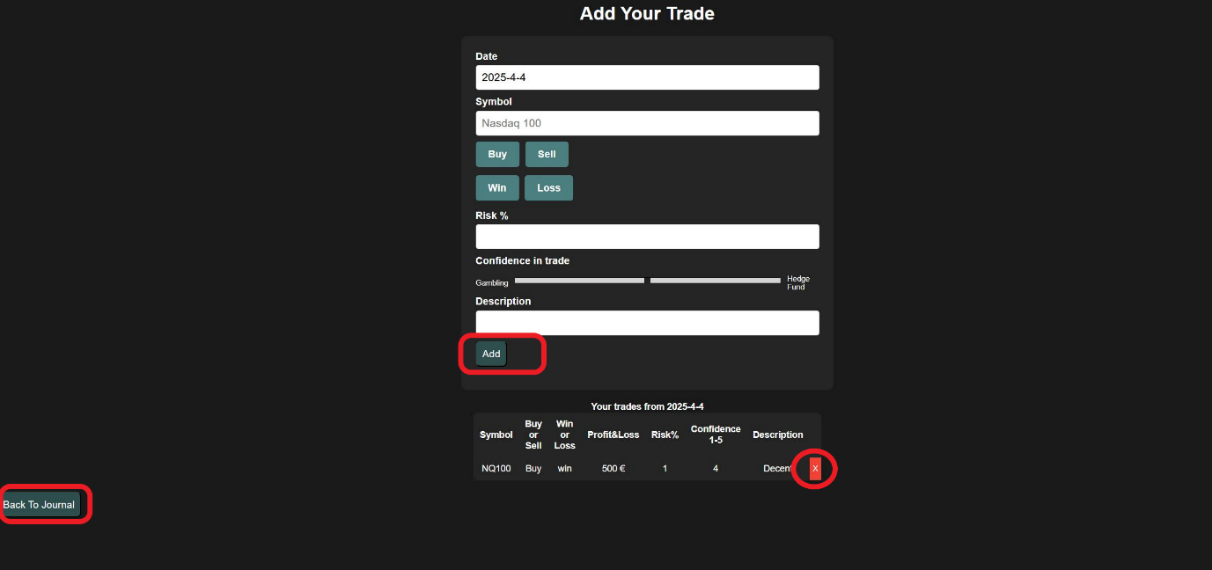


Uzspiežot uz pogas “My Journal”, lietotājs nokļūst savā žurnālā, kas ir kalendāra formātā, kuram pēc vajadzības var nomainīt gadu un mēnesi, un, uzspiežot uz kādu no izvēlētajām dienām, lietotājs var pierakstīt informāciju par tirgojumu no tās dienas –

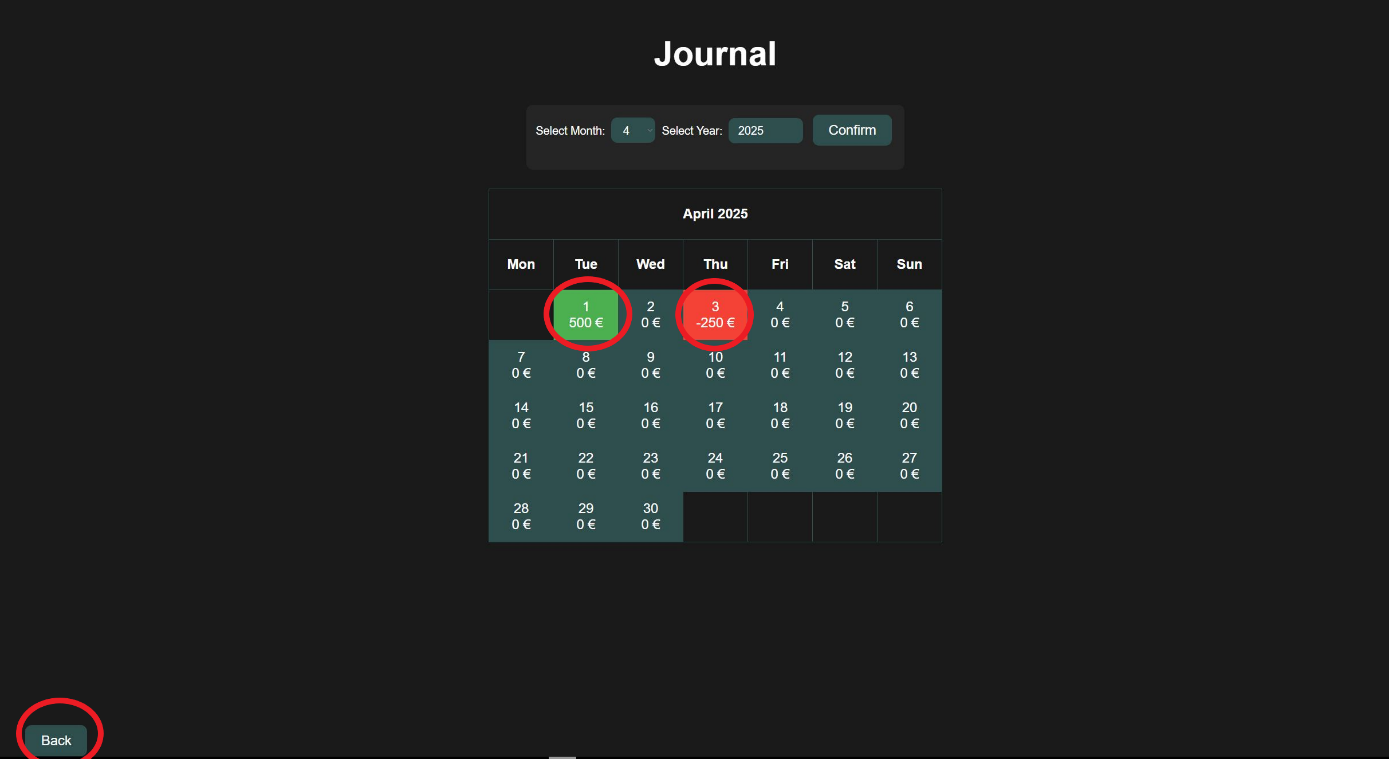


Kad kāda no dienām tiks izvēlēta, atvērsies tirgojuma pievienošanas skats, kur katrs lauks ir jāaizpilda. Datums tiks automātiski pievienots. Otrais lauks ir nosaukums tirgotajai akcijai, valūtai, kriptovalūtai utt. Trešajā laukā jāizvēlas, vai tirgojums bija pirkšana (“buy”) vai pārdošana (“sell”). Ceturtajā laukā – vai tirgojumam bija peļņa (“win”) vai zaudējums (“loss”). Izvēloties kādu no opcijām ceturtajā laukā, parādīsies piektais lauks, kurā jāievada vai nu cik nopelnīts vai attiecīgi cik zaudēts attiecībā no iepriekšēja lauka izvēles. Sestajā laukā jāievada cik procenti tika riskēti no kopējā konta bilances attiecīgajā tirgojumā. Septītajā laukā jāizvēlas cik liela pārliecība bija par tirgojumu, kur minimālā vērtība ir loterija, kazino (“gambling”) un maksimālā vērtība ir riska ieguldījumu fonds (“hedge fund”). Astotajā laukā jāievada apraksts par tirgojumu, informācija varbūt, kas netika piedāvāta parējos ievades laukos.

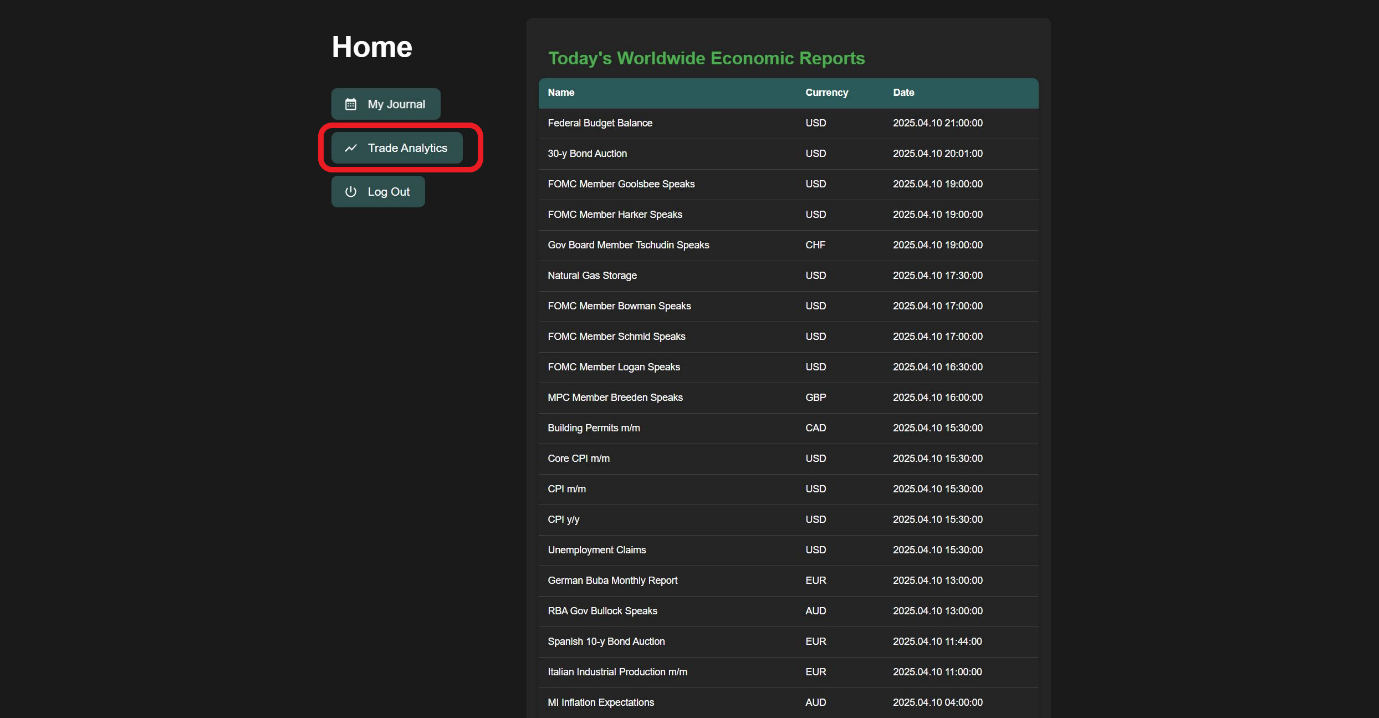
Pēc informācijas uzrakstīšanas, spiežot uz pogas “Add”, tirgojumu var pievienot. Zem pievienošanas formas ir redzama tirgojumu vēsture izvēlētajā diena un tabulas labajā pusē ir sarkana poga “X”, ar kuras uzspiešanu var izdzēst, ja nepieciešams, kādu tirgojumu. Kreisajā skata pusē ir poga ”Back To Journal”, ja lietotājs tomēr negrib pievienot tirgojumu –



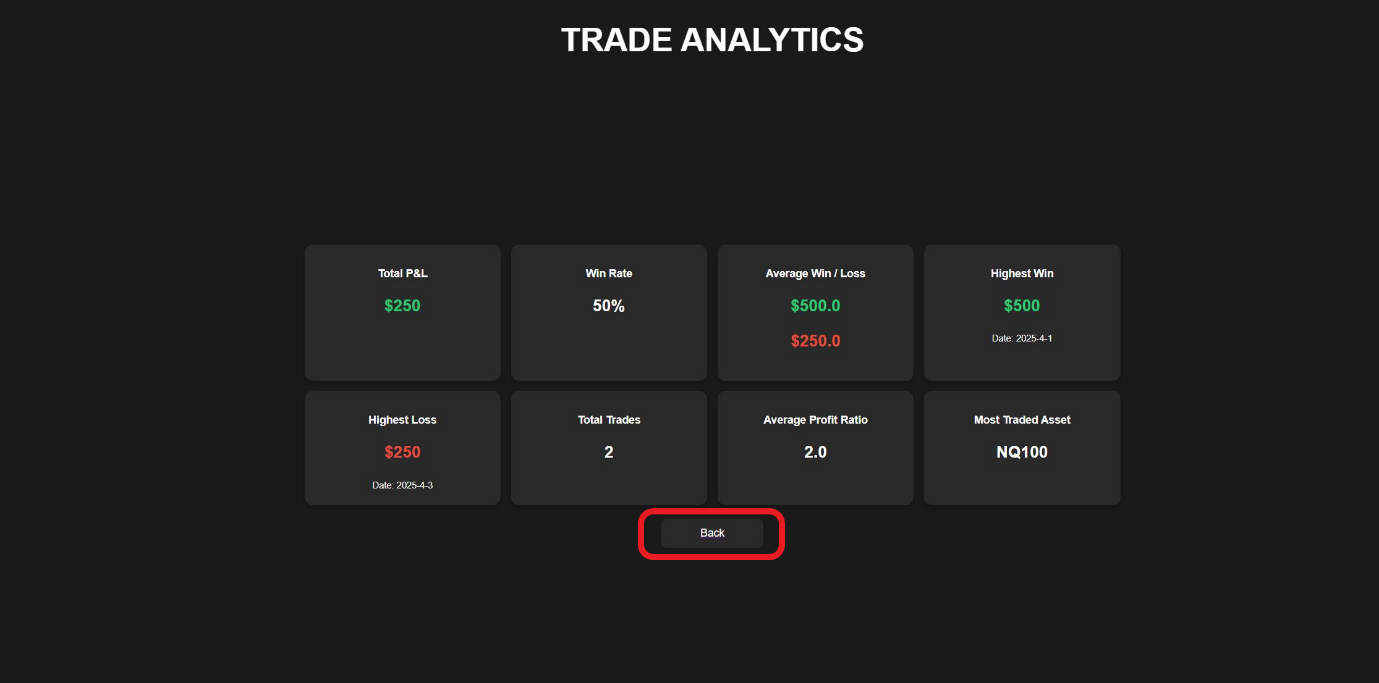
Kad lietotājs izvēlas doties atpakaļ uz žurnālu, būs redzams, ka kalendārā pie izvēlētās dienas tagad ir parādīts nopelnītais vai zaudētais šajā dienā, kas arī iekrāsots attiecīgā krāsā. Ja vēlas atgriezties uz sākumu, tad kreisajā apakšējajā stūrī ir poga ”Back” –



Nonākot sākuma skatā, lietotājs var uzspiest uz ”Trade Anlytics”, lai apskatītu savu statistiku par veiktajiem darijumiem –



Ja lietotājām bija veikts darījums ar peļņu un darījums ar zaudējumiem, tad atvērsies analītikas skats par veiktajiem darījumiem. Zem statistikām ir poga “Back”, ar kuru var doties atpakaļ uz iepriekšējo skatu –



**Pieliekums**

**Failu mapju struktūras attēlojumu**

TRADING JOURNAL/

│── main.py

│── .gitattributes

├── static/

│ ├── style\_analytics.css

│ ├── style\_auth.css

│ ├── style\_home.css

│ ├── style\_journal.css

│ └── style.css

│

├── templates/

│ ├── analytics.html

│ ├── error.html

│ ├── home.html

│ ├── journal.html

│ ├── login.html

│ ├── newtrade.html

│ ├── signup.html

│ ├── style.css

│ └── welcome.html

│

main.py--------------------------------------------------------------------------------------

from flask import Flask, render\_template , request , session ,redirect , url\_for

import sqlite3

import os

import calendar

import requests

from datetime import datetime

from werkzeug.security import generate\_password\_hash, check\_password\_hash

app = Flask(\_\_name\_\_)

app.secret\_key = os.urandom(24)

# SQL funkcija

def execute\_sql(cmd , vals=None):

    conn = sqlite3.connect("main.db")

    cursor = conn.cursor()

    if vals:

        cursor.execute(cmd , vals)

    else:

        cursor.execute(cmd)

    res = cursor.fetchall()

    conn.commit()

    conn.close()

    return res

@app.route("/")

def welcome():

    return render\_template("welcome.html")

# Vairāku skatu izveidošana un maršutēšana

@app.route("/login")

def login ():

    return render\_template("login.html")

@app.route("/signup")

def signup():

    return render\_template("signup.html")

@app.route("/home")

def home():

    return render\_template ("home.html", news = session["news"])

# Izveidoju tabulu ,kur tiks saglabāti ieraksti par akcijas, kripto utt tirgu pirkumiem

@app.route("/journal" , methods=[ "POST" , "GET"])

def journal():

    execute\_sql("CREATE TABLE IF NOT EXISTS Trades(\

                date DATE ,\

                symbol TEXT,\

                buy\_sell TEXT,\

                win\_loss TEXT,\

                pnl INTEGER,\

                risk INTEGER,\

                confidence TEXT,\

                description TEXT,\

                trade\_id INTEGER PRIMARY KEY,\

                user\_id INTEGER )")

    # Kalendāra izveide ar importētām funkcijām

    current\_date = datetime.now()

    year = request.form.get("year", current\_date.year, type=int)

    month = request.form.get("month", current\_date.month, type=int)

    full\_calendar=""

    cal = calendar.HTMLCalendar().formatmonth(year, month)

    user\_id = session["user\_id"]

    for day in range(1, 32):

        try:

            date = (f"{year}-{month}-{day}")

            # Katrai dienai aprēķinu kopēju peļņu/zaudi

            pnl = execute\_sql("SELECT COALESCE((SELECT SUM(pnl) FROM Trades WHERE win\_loss = 'win' AND date = ? AND user\_id = ?), 0)-COALESCE((SELECT SUM(pnl) FROM Trades WHERE win\_loss = 'loss' AND date = ? AND user\_id = ?), 0) AS result ", (date, user\_id , date , user\_id))            # Pārbaudu katrai dienai vai ir peļņa , zaude vai pa nullēm ,lai varētu katrai but savādāks dizains

            if pnl[0][0] < 0:

                id = "-day"

            elif pnl[0][0] > 0 :

                id = "+day"

            else:

                id = "day"

            day\_str = f'>{day}<'

            if day\_str in cal:

                # Kalendāram katrai dienai izveidoju pogu ,kuru uzspiežot var pievienot "trade" priekš tās dienas ,neievadot datumu

                cal = cal.replace(day\_str, f'><a href="/new\_trade-{year}-{month:02d}-{day:02d}" id="{id}"><button type="button">{day} <br> {pnl[0][0]} € </button></a><')

        except ValueError:

            continue

    full\_calendar += f"{cal}"

    return render\_template("journal.html", full\_calendar=cal, year=year, month=month)

@app.route("/new\_trade-<int:year>-<int:month>-<int:day>")

# funkcija ,kas tiek palaista uzspiežot uz jebkuru dienu kalendārā ,ar padoto datumu linkā

def new\_trade(year, month, day):

    user\_id = session["user\_id"]

    date = f"{year}-{month}-{day}"

    # tiek saglabāti "tradi" no konkrēta datumu ,lai tos varētu parādīt ,kā vēsturi

    trades = execute\_sql("SELECT  date , symbol , buy\_sell , win\_loss, pnl , risk , confidence , description ,trade\_id FROM Trades WHERE date  = ? AND user\_id = ?" , (date , user\_id) )

    return render\_template("newtrade.html", year=year, month=month, day=day , trades = trades )

# Skats kur varēs apskatīt analītikas un statistikas par veiktajiem "tradiem".

@app.route("/analytics")

def stats ():

    user\_id = session["user\_id"]

    wins = execute\_sql("SELECT COUNT(\*) FROM Trades Where user\_id = ? AND win\_loss = 'win' " , (user\_id,))

    losses = execute\_sql("SELECT COUNT(\*) FROM Trades Where user\_id = ? AND win\_loss = 'loss' " , (user\_id,))

    if wins[0][0] > 0 and losses[0][0] > 0:

        trade\_amount = execute\_sql("SELECT COUNT(\*) FROM Trades Where user\_id = ?  " , (user\_id,))

        win\_rate = round(wins[0][0]/trade\_amount[0][0] \* 100)

        highest\_win = execute\_sql("SELECT pnl , date FROM Trades WHERE win\_loss = 'win' AND user\_id = ? ORDER BY pnl DESC LIMIT 1  " , (user\_id,))

        highest\_loss = execute\_sql("SELECT pnl , date FROM Trades WHERE win\_loss = 'loss' AND user\_id = ? ORDER BY pnl DESC LIMIT 1  " , (user\_id,))

        total\_pnl = execute\_sql("SELECT COALESCE((SELECT SUM(pnl) FROM Trades WHERE win\_loss = 'win' AND user\_id = ?), 0) - ""COALESCE((SELECT SUM(pnl) FROM Trades WHERE win\_loss = 'loss' AND user\_id = ?), 0)",(user\_id, user\_id))

        average\_win = execute\_sql("SELECT AVG(pnl) FROM Trades WHERE win\_loss = 'win' AND user\_id = ? " , (user\_id,))

        average\_loss = execute\_sql("SELECT AVG(pnl) FROM Trades WHERE win\_loss = 'loss' AND user\_id = ? " , (user\_id,))

        win\_loss\_ratio = round( average\_win[0][0] / average\_loss[0][0] ,1)

        most\_traded = execute\_sql("SELECT symbol, COUNT(symbol) AS  symbol\_count FROM Trades WHERE user\_id = ? GROUP BY symbol ORDER BY symbol\_count DESC LIMIT 1" , (user\_id,))

        return render\_template("analytics.html" ,win\_rate = win\_rate ,trade\_amount = trade\_amount[0][0] , highest\_win = highest\_win[0][0] , highest\_win\_date = highest\_win[0][1] , highest\_loss = highest\_loss[0][0] ,highest\_loss\_date = highest\_loss[0][1] , total\_pnl = total\_pnl[0][0] , average\_win = round(average\_win[0][0],2), average\_loss =round( average\_loss[0][0],2) ,win\_loss\_ratio = win\_loss\_ratio ,most\_traded = most\_traded[0][0])

    else:

        # Lai analītikas par "traidiem" varētu but aprēķinātas ir vajadzīga vismaz viena uzvara un viena zaude

        return render\_template("error.html", message = "Not Enough Trades For Analytics .Minimum : 1 Win & 1 Loss .")

@app.route("/submit" , methods = ["POST" , "GET"])

def submit ():

    if request.method == "POST" :

        email = request.form["email"]

        password = request.form["password"]

        # Noņem liekas atstarpes

        email = email.strip()

        password = password.strip()

        # Paroli teik "hashota" ar kriptogrāfiju , lai lietotāja dati būtu pasargāti

        hashed\_password = generate\_password\_hash(password)

        # Pārbauda vai ievadītais emails vai parole nav tuksš pēc atstarpju noņemšanas

        if email =="" or password =="":

            return render\_template ("error.html", message="Email/Password cannot be empty ")

        # Pārbauda vai ievadītais epasts jau netiek lietots

        check = execute\_sql("SELECT EXISTS (SELECT 1 FROM User WHERE email = ?  )" , (  email ,    )   )

        if check[0][0] == 0:

            # Pievieno no datubāzei

            execute\_sql("INSERT INTO User (email , password) VALUES (?,?) ",  (email , hashed\_password))

            return render\_template("login.html")

        else :

            return render\_template ( "signup.html", message ="An account with this email already exists")

@app.route("/submitlog", methods=["POST", "GET"])

def submitlog():

    if request.method == "POST":

        email = request.form["email"]

        password = request.form["password"]

        user = execute\_sql("SELECT user\_id, password FROM User WHERE email = ?", (email,))

        # Pārbauda vai ievadīta parole sakrīt ar "hashoto" paroli

        if user and check\_password\_hash(user[0][1], password):

            # saglabā lietotāja id , priekš sesijas ,lai vēlāk varētu to lietotot priekš datubāzes lietošanas

            session["user\_id"] = user[0][0]

            # API finkcija ,kas savāc datus par pasaules ekomiskajām ziņām lietošanas bŗīža dienā ar limitu requestiem vienreiz piecās minūtēs

            url: str = "https://www.jblanked.com/news/api/forex-factory/calendar/today/"

            headers = {

                "Content-Type": "application/json",

                "Authorization": f"Api-Key DYibXX4P.rn7fKN0jPM0oozVFrkWO3VA32JU1Ch5J ",

            }

            response = requests.get(url, headers=headers)

            news = []

            if response.status\_code == 200:

                data = response.json()

                # Ja datu koppa nav tukša tiek atgriezti datu nosaukums , kuru valūtu ziņas ietekmē un datums

                for result in data:

                    news.append([result["Name"], result["Currency"], result["Date"]])

            else:

                print(f"Error: {response.status\_code}")

                print(response.json())

            session["news"] = news

            return render\_template("home.html", news=news)

        else:

            return render\_template("login.html", message="Incorrect email or password")

@app.route("/submit\_trade" , methods = ["POST"])

# funkcija kas apstrādā ievadītos datus par akciju tirgu tirgojumiem un ievieto tos datubāze

def submit\_trade():

    if request.method == "POST":

        date = request.form["date"]

        symbol = request.form["symbol"]

        buy\_sell = request.form["buy\_sell"]

        win\_loss = request.form["win\_loss"]

        risk = request.form["risk"]

        confidence =request.form["confidence"]

        description = request.form["description"]

        # saglabāto sessijas lietotāja ID ievieto kopā ar tirgojumu ,lai varētu kategorizēt datus priekš katra lietotāja

        user\_id = session["user\_id"]

        print(win\_loss)

        if win\_loss == "win":

            pnl = request.form["profit"]

        else:

            pnl = request.form["deficit"]

        execute\_sql("INSERT INTO Trades (date , symbol , buy\_sell , win\_loss, pnl , risk , confidence , description , user\_id) VALUES(?,?,?,?,?,?,?,?,?) " , (date , symbol , buy\_sell , win\_loss , pnl , risk , confidence , description , user\_id))

    year, month, day = date.split("-")

    return redirect(url\_for("new\_trade", year=year, month=month, day=day))

# Opcija izdzēst kādu ievadījumu no datubāzes

@app.route ("/delete" , methods = ["POST"])

def delete():

    trade\_id = request.form["trade\_id"]

    date = request.form["date"]

    execute\_sql("DELETE FROM Trades WHERE trade\_id = ?" , (trade\_id,))

    year, month, day = date.split("-")

    return redirect(url\_for("new\_trade", year=year, month=month, day=day))

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(host='0.0.0.0', port=8080 ,debug=True )

.gitattributes------------------------------------------------------------------------------------------------

# Auto detect text files and perform LF normalization

\* text=auto

style\_analytics.css-------------------------------------------------------------------------------------------------

body {

    font-family: Arial, sans-serif;

    background-color: #1a1a1a;

    color: white;

    margin: 0;

    padding: 0;

    display: flex;

    align-items: center;

    justify-content: center;

    height: 100vh;

}

.dashboard {

    width: 90%;

    max-width: 1200px;

}

.grid {

    display: grid;

    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

    gap: 15px;

}

.stat-box {

    background: #2a2a2a;

    padding: 20px;

    border-radius: 12px;

    text-align: center;

    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

}

.stat-box h2 {

    font-size: 16px;

}

p {

    font-size: 24px;

    font-weight: bold;

}

.win {

    color: #2ecc71;

}

.loss {

    color: #e74c3c;

}

.negative {

    color: #e74c3c;

}

.positive {

    color: #2ecc71;

}

.back-button {

    display: block;

    width: 150px;

    margin: 20px auto;

    padding: 12px;

    background-color: #2a2a2a;

    border: none;

    border-radius: 8px;

    color: white;

    font-size: 16px;

    cursor: pointer;

}

.back-button:hover {

    background-color: #0056b3;

}

.page-title {

    text-align: center;

    font-size: 48px;

    font-weight: bold;

    margin-top: 20px;

    margin-bottom: 30px;

    text-transform: uppercase;

    position: absolute;

    top: 20px;

    left: 50%;

    transform: translateX(-50%);

}

style.auth.css---------------------------------------------------------------------------------------

h1 {

  display: flex;

  justify-content: center;

}

body {

  background-color: #1a1a1a;

  color: white;

  font-family: arial;

}

.radio-group {

  display: flex;

  gap: 10px;

  margin-bottom: 10px;

}

.radio-group input[type="radio"] {

  display: none;

}

.radio-group label {

  display: inline-block;

  padding: 10px 20px;

  background: rgb(76, 128, 128);

  color: white !important;

  font-size: 1.2rem;

  border-radius: 5px;

  cursor: pointer;

  border: 2px solid transparent;

  text-align: center;

  transition: all 0.3s ease;

}

#win:checked + label {

  background: green;

  color: white !important;

  border-color: green;

}

#loss:checked + label {

  background: red;

  color: white !important;

  border-color: red;

}

.radio-group input[type="radio"]:checked + label {

  background: darkslategrey;

  border-color:darkslategrey;

}

.radio-group label:hover {

  background: darkgray;

}

#win:checked ~ .profit-container {

  display: block;

}

#loss:checked ~ .deficit-container {

  display: block;

}

.profit-container{

  display: none;

}

.deficit-container {

  display: none;

}

#utility {

  background-color: darkslategray;

  border-color:  darkslategray;

  color: white;

  padding: 10px;

  border-radius: 5px;

  font-size: 1.1rem;

  cursor: pointer;

  font-family: arial;

  border-radius: 8px;

}

.form-container, #history {

  width: 60%;

  max-width: 600px;

  margin: 0 auto;

  padding: 25px;

  border-color: #252525;

  border-radius: 10px;

  background: #252525;

  color: white;

  margin-bottom: 20px;

}

table {

  width: 100%;

  border-collapse: collapse;

  border-color: dark;

  border-color: #252525;

}

td, th {

  padding: 10px;

  border: 0.5px solid rgb(152, 151, 151);

  text-align: center;

  position: relative;

  border-color: #252525;

}

.form-container input {

  width: 100%;

  padding: 10px;

  font-size: 1.2rem;

  border: 1px solid #ccc;

  border-radius: 5px;

  margin-bottom: 10px;

}

.confidence-container {

  display: flex;

  align-items: center;

  gap: 10px;

  width: 100%;

}

.form-container input[type="radio"] {

  transform: scale(1.5);

  margin-right: 10px;

}

.form-container label {

  font-weight: bold;

  font-size: 1.1rem;

  margin-bottom: 5px;

  display: block;

}

#remove\_button {

  width: 100%;

  height: 100%;

  background-color: #F44336;

  color: white;

  border: none;

  position: absolute;

  top: 0;

  left: 0;

  cursor: pointer;

}

style\_home.css----------------------------------------------------------------------------------------------

body {

    background-color: #1a1a1a;

    color: white;

    font-family: Arial, sans-serif;

    margin: 0;

    padding: 30px;

    min-height: 100vh;

}

h1 {

    text-align: left;

    font-size: 3em;

    margin: 20px 0 40px 0;

    width: 100%;

    max-width: 1200px;

    padding-left: 20px;

}

.content-container {

    display: flex;

    flex-direction: column;

    gap: 40px;

    width: 100%;

    max-width: 1200px;

    margin: 0 auto;

    padding: 0 20px;

}

@media (min-width: 992px) {

    .content-container {

        flex-direction: row;

        align-items: flex-start;

    }

}

.left-column {

    width: 100%;

}

@media (min-width: 992px) {

    .left-column {

        width: 300px;

        position: sticky;

        top: 20px;

    }

}

.right-column {

    flex: 1;

    width: 100%;

}

.button-container {

    display: flex;

    flex-direction: column;

    gap: 20px;

    width: 100%;

    max-width: 400px;

    margin-top: 30px;

    padding-left: 20px;

}

button {

    background-color: darkslategray;

    color: white;

    border: none;

    padding: 15px 25px 15px 60px;

    font-size: 1.2em;

    border-radius: 10px;

    cursor: pointer;

    width: 100%;

    transition: all 0.2s ease;

    position: relative;

    text-align: left;

}

button:hover {

    transform: scale(1.05);

    background-color: #2a5a5a;

}

button::before {

    content: "";

    position: absolute;

    left: 20px;

    top: 50%;

    transform: translateY(-50%);

    width: 24px;

    height: 24px;

    background-size: contain;

    background-repeat: no-repeat;

    background-position: center;

}

a[href="/journal"] button::before {

    background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' fill='white'%3E%3Cpath d='M19 4h-1V2h-2v2H8V2H6v2H5c-1.1 0-2 .9-2 2v14c0 1.1.9 2 2 2h14c1.1 0 2-.9 2-2V6c0-1.1-.9-2-2-2zm0 16H5V8h14v12zM9 10H7v2h2v-2zm4 0h-2v2h2v-2zm4 0h-2v2h2v-2zm-8 4H7v2h2v-2zm4 0h-2v2h2v-2zm4 0h-2v2h2v-2z'/%3E%3C/svg%3E");

}

a[href="/analytics"] button::before {

    background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' fill='white'%3E%3Cpath d='M3.5 18.49l6-6.01 4 4L22 6.92l-1.41-1.41-7.09 7.97-4-4L2 16.99z'/%3E%3C/svg%3E");

}

a[href="/login"] button::before {

    background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' fill='white'%3E%3Cpath d='M13 3h-2v10h2V3zm4.83 2.17l-1.42 1.42C17.99 7.86 19 9.81 19 12c0 3.87-3.13 7-7 7s-7-3.13-7-7c0-2.19 1.01-4.14 2.58-5.42L6.17 5.17C4.23 6.82 3 9.26 3 12c0 4.97 4.03 9 9 9s9-4.03 9-9c0-2.74-1.23-5.18-3.17-6.83z'/%3E%3C/svg%3E");

}

a {

    text-decoration: none;

    display: block;

    width: fit-content;

}

table {

    width: 100%;

    border-collapse: collapse;

    margin-top: 0;

    background-color: #252525;

    border-radius: 10px;

    overflow: hidden;

    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

}

th, td {

    padding: 15px;

    text-align: left;

    border-bottom: 1px solid #444;

}

th {

    background-color: #2a5a5a;

    font-weight: bold;

    color: white;

}

.table-header {

    margin: 30px 0 15px 0;

    font-size: 1.8em;

    color: #4CAF50;

    position: relative;

    padding-left: 15px;

}

.table-section {

    background-color: #252525;

    padding: 20px;

    border-radius: 10px;

}

style\_journal.css--------------------------------------------------------------------------------------------

body {

  background-color: #1a1a1a;

  color: white;

  font-family: Arial;

  padding: 30px;

  font-size: 18px;

}

h1 {

  display: flex;

  justify-content: center;

  font-size: 3em;

  margin: 20px 0;

}

.calendar-container {

  display: flex;

  flex-direction: column;

  align-items: center;

  height: auto;

  min-height: 80vh;

  padding: 30px;

  margin: 0px auto;

  max-width: 1200px;

}

#utility {

  background-color: darkslategrey;

  border: white;

  color: white;

  padding: 10px 15px;

  font-size: 1.1em;

  margin: 5px;

}

select#utility,

input#utility {

  height: 40px;

  font-size: 1em;

  border-radius: 10px;

}

table {

  border-collapse: collapse;

  width: 80%;

  max-width: 900px;

  margin: 10px auto;

  font-size: 1.2em;

  table-layout: fixed;

}

td, th {

  border: 0.5px solid darkslategray;

  text-align: center;

  padding: 0;

  min-width: 60px;

  height: 80px;

  position: relative;

}

#day button,

#\-day button,

#\+day button {

  position: absolute;

  top: 0;

  left: 0;

  right: 0;

  bottom: 0;

  width: 100%;

  height: 100%;

  border: none;

  font-family: Arial;

  color: white;

  font-size: 1.0em;

  display: flex;

  flex-direction: column;

  justify-content: center;

  align-items: center;

  transition: all 0.2s ease;

}

#day button {

  background-color: darkslategray;

}

#\-day button {

  background-color: #F44336;

}

#\+day button {

  background-color: #4CAF50;

}

button#utility:hover,

a[href="/home"] button#utility:hover {

  transform: scale(1.05);

  transition: transform 0.2s ease;

}

#day button:hover,

#\-day button:hover,

#\+day button:hover {

  transform: scale(1.05);

  box-shadow: 0 0 10px rgba(255,255,255,0.2);

  z-index: 1;

}

.selecter  {

  margin-bottom: 20px;

  padding: 15px;

  background-color: #252525;

  border-radius: 10px;

}

.selecter button{

  border-radius: 10px;

}

button#utility {

  font-size: 1.2em;

  padding: 12px 25px;

  margin-top: -10px;

  border-radius: 10px;

}

style.css------------------------------------------------------------------------------------------------------

h1 {

  display: flex;

  justify-content: center;

}

body {

  background-color: #1a1a1a;

  color: white;

  font-family: arial;

}

.radio-group {

  display: flex;

  gap: 10px;

  margin-bottom: 10px;

}

.radio-group input[type="radio"] {

  display: none;

}

.radio-group label {

  display: inline-block;

  padding: 10px 20px;

  background: rgb(76, 128, 128);

  color: white !important;

  font-size: 1.2rem;

  border-radius: 5px;

  cursor: pointer;

  border: 2px solid transparent;

  text-align: center;

  transition: all 0.3s ease;

}

#win:checked + label {

  background: green;

  color: white !important;

  border-color: green;

}

#loss:checked + label {

  background: red;

  color: white !important;

  border-color: red;

}

.radio-group input[type="radio"]:checked + label {

  background: darkslategrey;

  border-color:darkslategrey;

}

.radio-group label:hover {

  background: darkgray;

}

#win:checked ~ .profit-container {

  display: block;

}

#loss:checked ~ .deficit-container {

  display: block;

}

.profit-container{

  display: none;

}

.deficit-container {

  display: none;

}

#utility {

  background-color: darkslategray;

  border-color:  darkslategray;

  color: white;

  padding: 10px;

  border-radius: 5px;

  font-size: 1.1rem;

  cursor: pointer;

  font-family: arial;

  border-radius: 8px;

}

.form-container, #history {

  width: 60%;

  max-width: 600px;

  margin: 0 auto;

  padding: 25px;

  border-color: #252525;

  border-radius: 10px;

  background: #252525;

  color: white;

  margin-bottom: 20px;

}

table {

  width: 100%;

  border-collapse: collapse;

  border-color: dark;

  border-color: #252525;

}

td, th {

  padding: 10px;

  border: 0.5px solid rgb(152, 151, 151);

  text-align: center;

  position: relative;

  border-color: #252525;

}

.form-container input {

  width: 100%;

  padding: 10px;

  font-size: 1.2rem;

  border: 1px solid #ccc;

  border-radius: 5px;

  margin-bottom: 10px;

}

.confidence-container {

  display: flex;

  align-items: center;

  gap: 10px;

  width: 100%;

}

.form-container input[type="radio"] {

  transform: scale(1.5);

  margin-right: 10px;

}

.form-container label {

  font-weight: bold;

  font-size: 1.1rem;

  margin-bottom: 5px;

  display: block;

}

#remove\_button {

  width: 100%;

  height: 100%;

  background-color: #F44336;

  color: white;

  border: none;

  position: absolute;

  top: 0;

  left: 0;

  cursor: pointer;

}

analytics.html------------------------------------------------------------------------------------------------

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Trade Analytics</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style\_analytics.css') }}">

</head>

<body>

    <h1 class="page-title">Trade Analytics</h1>  <!-- Moved to the absolute top -->

    <div class="dashboard">

        <div class="grid">

            <div class="stat-box">

                <h2>Total P&L</h2>

                <p class="{% if total\_pnl < 0 %}negative{% else %}positive{% endif %}">${{ total\_pnl }}</p>

            </div>

            <div class="stat-box">

                <h2>Win Rate</h2>

                <p>{{ win\_rate }}%</p>

            </div>

            <div class="stat-box">

                <h2>Average Win / Loss</h2>

                <p class="win">${{ average\_win }}</p>

                <p class="loss">${{ average\_loss }}</p>

            </div>

            <div class="stat-box">

                <h2>Highest Win</h2>

                <p class="win">${{ highest\_win }}</p>

                <small>Date: {{ highest\_win\_date }}</small>

            </div>

            <div class="stat-box">

                <h2>Highest Loss</h2>

                <p class="loss">${{ highest\_loss }}</p>

                <small>Date: {{ highest\_loss\_date }}</small>

            </div>

            <div class="stat-box">

                <h2>Total Trades</h2>

                <p>{{ trade\_amount }}</p>

            </div>

            <div class="stat-box">

                <h2>Average Profit Ratio</h2>

                <p>{{ win\_loss\_ratio }}</p>

            </div>

            <div class="stat-box">

                <h2>Most Traded Asset</h2>

                <p>{{ most\_traded }}</p>

            </div>

        </div>

        <a href="/home"><button class="back-button">Back</button></a>

    </div>

</body>

</html>

error.html----------------------------------------------------------------------------------------------------

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

    <h1>error</h1>

    <h2>{{message}}</h2>

    <a href="/home"><button type="button">Back</button></a>

</head>

<body>

</body>

</html>

home.html----------------------------------------------------------------------------------------------------

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Trade Journal Home</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style\_home.css') }}">

</head>

<body>

    <div class="content-container">

        <!-- Left Column -->

        <div class="left-column">

            <h1>Home</h1>

            <div class="button-container">

                <a href="/journal"><button type="button">My Journal</button></a>

                <a href="/analytics"><button type="button">Trade Analytics</button></a>

                <a href="/login"><button type="button">Log Out</button></a>

            </div>

        </div>

        <!-- Right Column -->

        <div class="right-column">

            <div class="table-section">

                <h2 class="table-header">Today's Worldwide Economic Reports</h2>

                <table>

                    <thead>

                        <tr>

                            <th>Name</th>

                            <th>Currency</th>

                            <th>Date</th>

                        </tr>

                    </thead>

                    <tbody>

                        {% for row in news %}

                            <tr>

                                {% for item in row %}

                                    <td>{{ item }}</td>

                                {% endfor %}

                            </tr>

                        {% endfor %}

                    </tbody>

                </table>

            </div>

        </div>

    </div>

</body>

</html>

journal.html--------------------------------------------------------------------------------------------------

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Journal</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style\_journal.css') }}">

</head>

<body>

<h1>Journal</h1>

<div class="calendar-container">

    <div class="selecter">

        <form method="POST" action="/journal">

            <label for="month">Select Month:</label>

            <select name="month" id="utility">

                {% for m in range(1, 13) %}

                    <option  value="{{ m }}" {% if m == month %}selected{% endif %}>{{ m }}</option>

                {% endfor %}

            </select>

            <label for="year">Select Year:</label>

            <input type="number" name="year" id="utility" value="{{ year }}" min="2000" max="2100">

            <button type="submit" id="utility">Confirm</button>

        </form>

    </div>

    <div class="calendar">

        {{ full\_calendar | safe }}

    </div>

</div>

<a href="/home" ><button type="button" id="utility">Back</button></a>

</body>

</html>

login.html----------------------------------------------------------------------------------------------------

<head>

    <meta charset="UTF-8">

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

    <title>Log In</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style\_auth.css') }}">

</head>

<body>

    <h1>LOG IN</h1>

    <div class="auth-form">

        <p class="error-message">{{message}}</p>

        <form action="/submitlog" method="POST">

            <label for="email">Email</label>

            <input type="email" name="email" id="email" placeholder="example@gmail.com">

            <label for="password">Password</label>

            <input type="password" name="password" id="password" placeholder="\*\*\*\*\*\*\*\*">

            <button type="submit">Log In</button>

        </form>

    </div>

    <div class="auth-footer">

        <p>Don't have an account?</p>

        <a href="/signup" class="auth-link">Sign Up</a>

    </div>

</body>

newtrade.html-----------------------------------------------------------------------------------------------

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style.css') }}">

    <h1>Add Your Trade</h1>

</head>

<body>

<div class="form-container">

    <form action="/submit\_trade" method="post" >

        <label for="date">Date</label>

        <input type="text" name="date" id="date" value="{{ year }}-{{ month }}-{{ day }}" readonly><br>

        <label for="symbol">Symbol</label>

        <input type="text" name="symbol" id="symbol" placeholder="Nasdaq 100" required><br>

        <!-- Ar css palīdzību uztaisiju funkciju ,kas ,kad uzpiež uz "win" vai "loss" tad parādas ievades lauks ar attiecīgu "profit" vai "loss" ,kas tiks ievietots datubāzē -->

        <div class="radio-group">

            <input type="radio" id="buy" name="buy\_sell" value="Buy" required>

            <label for="buy">Buy</label>

            <input type="radio" id="sell" name="buy\_sell" value="Sell" required>

            <label for="sell">Sell</label>

        </div>

        <div class="radio-group">

            <input type="radio" id="win" name="win\_loss" value="win" >

            <label for="win">Win</label>

            <input type="radio" id="loss" name="win\_loss" value="loss" >

            <label for="loss">Loss</label>

            <div class="profit-container">

                <label for="profit">Profit €</label>

                <input type="number" name="profit" id="profit" >

            </div>

            <div class="deficit-container">

                <label for="deficit">Loss €</label>

                <input type="number" name="deficit" id="deficit" >

            </div>

        </div>

        <label for="risk">Risk % </label>

        <input type="number" name="risk" id="risk" min="0.01" max="100" step="0.01" required><br>

        <label for="confidence">Confidence in trade</label>

        <div class="confidence-container">

            <small>Gambling</small>

            <input type="range" name="confidence" id="confidence" min="1" max="5" required>

            <small>Hedge Fund</small>

        </div>

        <label for="description">Description</label>

        <input type="text" id="description" name="description"><br>

        <button type="submit" id="utility">Add</button>

    </form>

</div>

{%if trades%}

    <table id="history">

        <caption><strong>Your trades from {{trades[0][0]}}</strong></caption>

            <tr>

                <th>Symbol</th>

                <th>Buy or Sell</th>

                <th>Win or Loss</th>

                <th>Profit&Loss</th>

                <th>Risk%</th>

                <th>Confidence 1-5</th>

                <th>Description</th>

            </tr>

            {%for trade in trades%}

            <tr>

                <td>{{trade[1]}}</td>

                <td>{{trade[2]}}</td>

                <td>{{trade[3]}}</td>

                <td>{{trade[4]}} €</td>

                <td>{{trade[5]}}</td>

                <td>{{trade[6]}}</td>

                <td>{{trade[7]}}</td>

                <form action="/delete" method="POST">

                    <td>

                        <input type="hidden" name="trade\_id" value="{{trade[8]}}">

                        <input type="hidden" name = "date" value="{{trade[0]}}">

                        <button type="submit" id="remove\_button">X</button>

                    </td>

                </form>

            </tr>

            {%endfor%}

    </table>

{%else%}

    {##}

{%endif%}

    <a href="/journal" ><button type="button" id="utility">Back To Journal</button></a>

</body>

</html>

signup.html--------------------------------------------------------------------------------------------------

<head>

    <meta charset="UTF-8">

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

    <title>Sign Up</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style\_auth.css') }}">

</head>

<body>

    <h1>Sign Up</h1>

    <div class="auth-form">

        <p class="error-message">{{message}}</p>

        <form action="/submit" method="POST">

            <label for="email">Email</label>

            <input type="email" name="email" id="email" placeholder="example@gmail.com" required>

            <label for="password">Password</label>

            <input type="password" name="password" id="password" placeholder="\*\*\*\*\*\*\*\*" minlength="8" required>

        <button type="submit">Sign Up</button>

        </form>

    </div>

    <div class="auth-footer">

        <p>Already have an account?</p>

        <a href="/login" class="auth-link">Log In</a>

    </div>

</body>

welcome.html-----------------------------------------------------------------------------------------------

<head>

    <meta charset="UTF-8">

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

    <title>Welcome</title>

    <link rel="stylesheet" href="{{ url\_for('static', filename='style\_auth.css') }}">

</head>

<body>

    <div class="welcome-container">

        <h1>Trading Journal</h1>

        <div class="button-container">

            <a href="/login"><button class="welcome-button">Log In</button></a>

            <a href="/signup"><button class="welcome-button">Sign Up</button></a>

        </div>

    </div>

</body>