**Title: Investment RoboAdvisor**

**Project Repository: https://github.com/Edward0728/RoboAdvisor\_4FD3**

**Team members:**

**Qin Yang 400420584** [**yangq90@mcmaster.ca**](mailto:yangq90@mcmaster.ca)

**Hua Yao 400344368** [**yaoh17@mcmaster.ca**](mailto:yaoh17@mcmaster.ca)

**Abubakir Myrzaly 400268356** [**myrzalya@mcmaster.ca**](mailto:myrzalya@mcmaster.ca)

**Hongliang Wang 400220416** [**wangh258@mcmaster.ca**](mailto:wangh258@mcmaster.ca)

**Introduction: This chatbot asks users questions regarding their risk appetite, fund size preference, return expectation, and volatility tolerance as inputs. The backend algorithms will use these inputs to search in the updated product database and give funds matching the user’s requirements. In the second half semester, more advanced features using more algorithms would be integrated to provide more valuable solutions for the users.**

**Objectives: By chatting with the bot, the user could quickly get the product recommendation fitting their situation. This help user avoids subscription to financial website, look into hundreds of products, and manually filter the products without much investment knowledge.**

**Solution methodology:**

**Software requirements:**

**ChatterBot 1.0.4**

**Django 4.1.1**

**chatterbot-corpus 1.2.0**

**Flask 2.2.2**

**Mathparse 0.1.2**

**Matplotlib 3.6.1**

**nltk 3.7**

**Hardware requirements:**

**App host on cloud (TBD)**

**Steps of implementation:**

|  |  |
| --- | --- |
| **Steps** | **First half Implementation** |
| **1** | **Project Specification Initial** |
| **2** | **Setup GitHub repo and split tasks based on backend and frontend scopes** |
| **3** | **Frontend: draft chatbot completed. Backend: product data source research done** |
| **4** | **Frontend: finalize chatbot. Backend: automated data retrieval, storage and updated processes** |
| **5** | **Frontend and backend interface API development** |
| **6** | **Frontend: hook chatbot UI with API. Backend: solution design** |
| **7** | **Solution development (user input retrieval, solution functions, solution rendering)** |
|  | **Second half Implementation** |
| **8** | **Frontend: UI update. Backend: stock solution development** |
| **9** | **Frontend: UI update. Backend: advanced feature development** |
| **10** | **Project server deployment** |
| **11** | **Complete project testing and troubleshooting** |
| **12** | **Prepare for the final submission.** |

**Validation strategy:**

|  |  |
| --- | --- |
| **Validation Stage** | **How to measure performance** |
| **Unit validation** | **Run each unit code on source data and get desired outputs for the following process.**  **Code is bug-free, modularized and easy to troubleshoot and expand.** |
| **Integration validation** | **Run pipeline scripts to get data updates from the data source**  **Run API scripts to get expected data output from solution scripts**  **Run app scripts to render the correct product list from API function calls**  **Code is bug-free, modularized and easy to troubleshoot and expand** |
| **Complete validation** | **Visit the project on the webpage server.**  **Test data update, user interactions, solution rendering, project start and quit functions from the user experience.**  **Check the chatbot behaviors as designed such as correct greetings, right questions and answers, meaningful solutions, proper endings, etc.** |

**Ethics and sustainability considerations:**

* **This software is provided to all races and all genders. While using the software, no gender, the racial or religious question is asked. None of these criteria is in use to assist investment decisions.**
* **All data collection is for financial product analyzing purposes only and can be deleted by the user at any time. AI chatbot does not save user information without consent from the user.**
* **With this chatbot, customers may save phone or in-person appointments, which consumes more energy and produces more carbon emissions. Furthermore, having the communication stored electronically eliminates the paper print-out—all these benefits are helpful to more sustainable development.**

**Commercialization (Bonus):**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 1. **Importance** | 1. **Innovation** | 1. **Competition** |
| **Rank (1-10)** | **Rank: 5**  **Professional investors have all kinds of tools. But we provide an easy-use solution for the average population to get investment suggestions.** | **Rank: 7**  **This automated and customized solution could be the mainstream service type in the future financial industry.** | **Rank: 4**  **Cheaper, Easier to use, Performs faster and consistently. Special features (coming)**  **At the first release stage, the value we can provide to the users might be limited. But this business model could win.** |

**We are considering two ways of commercialization for our project:**

1. **Subscription**

**Users can use the trial version free of charge. To access advanced features such as user accounts and stock investment recommendations, Users need to subscribe and pay a monthly fee.**

1. **Source Code Package**

**We can also sell this chatbot software with customized AI algorithms to other vendors and commercial websites.**

**Proposed timeline: Include the table (see below) indicating the individual activities and the weeks during which they will be undertaken.**

Table 1: Project activity schedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone** | **Individual Work Done** | **Group Work Combined Deadline** | **Course Submission Deadline** |
| **Project Specification Initial** | **Friday, Sept 16th** | **Saturday, Sept 17th** | **Sunday, Sept 18th** |
|  |  |  |  |
| **Clarify Objectives and distribute Workload** | **Week 3** |  |  |
| **Code individually and Intergrade as Group** | **Weeks 4, 5, 6, 7** |  |  |
|  |  |  |  |
| **Documentation Submission Preparation** | **Week 8** |  |  |
| **Midterm Document Section Submission** | **N/A** | **Sunday, Oct 30th** | **Sunday, Oct 30th** |
| **Midterm Prototype Submission (Code)** | **Friday, Oct 28th** | **Saturday, Oct 29th** | **Sunday, Oct 30th** |
| **Midterm Video Section Submission** | **N/A** | **Sunday, Oct 10th** | **Sunday, Oct 30th** |
|  |  |  |  |
| **Clarify second half Objectives and distribute Workload** | **Week 9** |  |  |
| **Code individually and Intergrade as Group** | **Weeks 10,11,12** |  |  |
| **Doc and PowerPoint Submission Preparation** | **Week 12** |  |  |
|  |  |  |  |
| **Final implementation (Code)** | **Monday, Nov 7th** | **Monday, Nov 14th** | **Monday, Nov 21st** |
| **Final Project report** | **Friday, Nov 18th** | **Monday, Nov 21st** | **Monday, Nov 28th** |
| **Final Presentation Preparation** | **Friday, Nov 18th** | **Monday, Nov 21st** | **Monday, Nov 28th** |

**Please note:**

**For all milestone submissions, we will manage with 3 stages:**

1. **The first stage is Individual work done. All members will submit their portion at the end of this stage.**
2. **The second stage is Group Work Integration. We will review the group work and combine individual work. Meanwhile, we will preview the next milestone and divide it into separate segments.**
3. **Third Stage is final submission. As planned, Group work submission should always be one week earlier than Course Submission Deadline.**

Requirements

**Product use cases**

User can interact with robot advisor and obtain investment information.

Investment information

(Chatbot chatting Picture needed. One for stock, One for mutual fund)

Robot-advisor provides financial planning services that are automated with algorithms without any human control.

A Robot-advisor works in the following ways such as first gathering information that a client will provide using an online survey and then automatically considering and providing advice to the client based on those answers.

Robo-advisors use passive index investing strategies to provide the necessary investment advice that many new people are straggling with nowadays.

**Functional requirements**

Functional requirements are product features or functions that developers must implement to enable users to accomplish their tasks

RoboAdvisor asks user information  
The syst calculates user risk tolerance level

The system gives recommended stock investment information

RoboAdvisor provides recommended a list of recommended mutual funds including names and selected information such as year rate of return.

**Data requirements**

Data requirements definition establishes the process used to identify, prioritize, precisely formulate, and validate the data needed to achieve predicting and advising objectives.

Data Used in API:

Aphap Vantage API returns intraday time series of the equity specified, covering extended trading hours where applicable (e.g., 4:00am to 8:00pm Eastern Time for the US market). The intraday data is derived from the Securities Information Processor (SIP) market-aggregated data. You can query both raw (as-traded) and split/dividend-adjusted intraday data from this endpoint

Data used in LSTM model:

ENB.TO starts from 2016-01-01 to 2022-11-18

ENB.TO is the stock symbol for Enbridge Incorporated on the Toronto Stock Exchange (TSX).

Enbridge Incorporated is a Canadian multinational energy transportation company based in Calgary, Alberta. It focuses on the transportation, distribution and generation of energy, primarily in North America.

SU.TO starts from 2016-01-01 to 2022-11-18

Suncor Energy is a Canadian integrated energy company based in Calgary, Alberta. It specializes in production of synthetic crude from oil sands

Data used in Mutual fund:

The Globe and Mail offers the most authoritative news and data in Canada, featuring corporate data, and national and international news. Company stock data. The most comprehensive database of Canadian corporate financial information in spreadsheet format. The information allows you to better understand the top companies in Canada, to make it easier to sell to, invest in and communicate with these companies.

**Performance requirements**  
Performance requirements define how well the software system accomplishes certain functions under specific conditions. Examples include the software's speed of response, throughput, execution time and storage capacity. The service levels comprising performance requirements are often based on supporting end-user tasks.

(Need timer some numbers)

**Security requirements**  
A security requirement is **a goal set out for an application at its inception**. Every application fits a need or a requirement. For example, an application might need to allow customers to perform actions without calling customer service

**Maintenance & support requirements**

**Operation and environment requirements**

The servers have been tested to meet all functional requirements when operating in the operating environmental limits presented in [Environmental Specifications](https://docs.oracle.com/cd/E19910-01/E21500-01/z40000c41010849.html). Operating computer equipment in extremes of temperature or humidity increases the failure rate of hardware components. To minimize the chance of component failure, use the server within the optimal temperature and humidity ranges

Software Enviroment requirements:

ChatterBot 1.0.4

Django 4.1.1

chatterbot-corpus 1.2.0

Flask 2.2.2

Mathparse 0.1.2

Matplotlib 3.6.1

nltk 3.7

yifinance 0.1.87

tensorflow 2.8.0

absl-py==1.0.0  
appdirs==1.4.4  
asgiref==3.5.2  
astunparse==1.6.3  
async-generator==1.10  
attrs==22.1.0  
blessed==1.19.1  
cachetools==5.0.0  
certifi==2021.10.8  
charset-normalizer==2.0.12  
ChatterBot==1.0.4  
chatterbot-corpus==1.2.0  
click==8.1.3  
codefind==0.1.3  
contourpy==1.0.5  
cycler==0.11.0  
Django==4.1.1  
et-xmlfile==1.1.0  
exceptiongroup==1.0.0rc9  
Flask==2.2.2  
flatbuffers==2.0  
fonttools==4.38.0  
gast==0.5.3  
google-auth==2.6.6  
google-auth-oauthlib==0.4.6  
google-pasta==0.2.0  
grpcio==1.44.0  
gunicorn==20.1.0  
h11==0.14.0  
h5py==3.6.0  
idna==3.3  
importlib-metadata==4.11.3  
itsdangerous==2.1.2  
Jinja2==3.1.2  
joblib==1.2.0  
jurigged==0.5.3  
keras==2.8.0  
Keras-Preprocessing==1.1.2  
kiwisolver==1.4.4  
libclang==14.0.1  
lxml==4.9.1  
Markdown==3.3.6  
MarkupSafe==2.1.1  
mathparse==0.1.2  
matplotlib==3.6.1  
multitasking==0.0.11  
nltk==3.7  
numpy==1.22.2  
oauthlib==3.2.0  
openpyxl==3.0.9  
opt-einsum==3.3.0  
outcome==1.2.0  
ovld==0.3.2  
packaging==21.3  
pandas==1.3.4  
Pillow==9.2.0  
Pint==0.19.2  
protobuf==3.20.1  
pyasn1==0.4.8  
pyasn1-modules==0.2.8  
pymongo==3.12.3  
pyparsing==3.0.9  
PySocks==1.7.1  
python-dateutil==2.7.3  
python-dotenv==0.21.0  
pytimeparse==1.1.8  
pytz==2021.3  
regex==2022.9.13  
requests==2.27.1  
requests-oauthlib==1.3.1  
rsa==4.8  
scikit-learn==1.0.2  
scipy==1.9.3  
selenium==4.5.0  
six==1.16.0  
sniffio==1.3.0  
sortedcontainers==2.4.0  
SQLAlchemy==1.2.19  
sqlparse==0.4.3  
tensorboard==2.8.0  
tensorboard-data-server==0.6.1  
tensorboard-plugin-wit==1.8.1  
tensorflow==2.8.0  
tensorflow-io-gcs-filesystem==0.25.0  
termcolor==1.1.0  
tf-estimator-nightly==2.8.0.dev2021122109  
threadpoolctl==3.1.0  
tqdm==4.64.1  
trio==0.22.0  
trio-websocket==0.9.2  
twelvedata==1.2.9  
typing\_extensions==4.2.0  
urllib3==1.26.9  
watchdog==2.1.9  
wcwidth==0.2.5  
webdriver-manager==3.8.3  
Werkzeug==2.2.2  
wrapt==1.14.0  
wsproto==1.2.0  
XlsxWriter==3.0.2  
yfinance==0.1.87  
zipp==3.8.0

Hardware requirements:

App host on cloud (TBD)