FINANCIAL ADVISORY CHATBOT

Project by: Group 4

Created by JUNIHERS GROUP 4:
Aastha Khandelwal
Dakshita Tripathi
Pranjal Adwani
Soumi Datta
Vedika Chandra

Under the mentorship of: Sandeep Manthi

Introduction

“Money is always eager and ready to work for anyone who is ready to employ it.”

— Idowu Koyenikan, Wealth for All: Living a Life of Success at the Edge of Your Ability

Finance management has always been one of the greatest concerns for most of us in our daily lives. From our grandparents and parents calculating their expenses at the end of every month in a notebook and making impulsive and intuitive decisions on investments, to us using chatbots to manage finance, we have come a long way. Money management is not an easy task, and involves a lot of analysis and discipline. We have tried to simplify the task by making a personal finance manager chatbot in integration with Telegram.

We implemented features such as portfolio management, real-time market information, plots of stocks as well as a unique goal setting program that motivates the user to save. For someone who is setting out on the path of finance management, the distributed and vast amount of information and advice available on the Internet can be overwhelming and confusing. Afterall when it comes to money, we all look for clarity, accuracy, transparency and security. Our chatbot strives to be that one-stop destination for all your finance management related issues, so that one does not have to spend hours struggling to find the correct information on the Internet.

Goals and Objectives

1) **Financial Goal Planning**

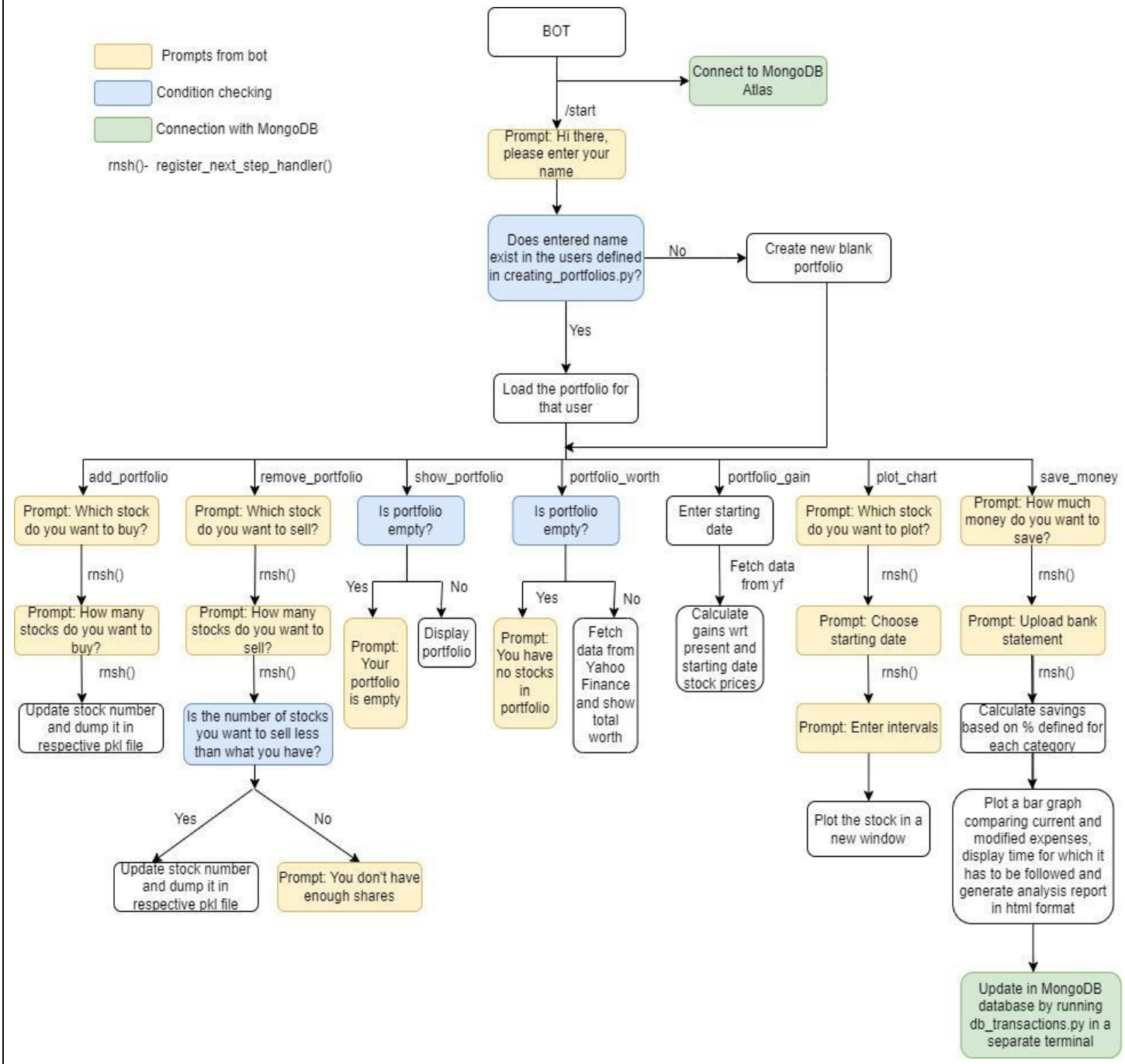
   Our chatbot lets users upload their current month's bank statement in the form of a csv file along with their goal and comes up with an updated monthly budget according to the goal of the user.

2) **Portfolio Management**

   The chatbot allows the user to add and remove stocks from his portfolio, as well as view and analyse it.

3) **Real Time Market Information**

   The chatbot gives real time information and plots when prompted, and also calculates relative and absolute gains from a given time point, and current portfolio valuation too.


Prompts from bot
Condition checking
Connection with MongoDB
rnsh()- register_next_step_handler()
BOT
Connect to MongoDB Atlas
/start
Prompt: Hi there, please enter your name
Does entered name exist in the users defined in creating_portfolios.py?
No
Create new blank portfolio
Yes
Load the portfolio for that user
add_portfolio
remove_portfolio
show_portfolio
portfolio_worth
portfolio_gain
plot_chart
save_money
Prompt: Which stock do you want to buy?
rnsh()
Prompt: How many stocks do you want to buy?
rnsh()
Update stock number and dump it in respective pkl file
Prompt: Which stock do you want to sell?
rnsh()
Prompt: How many stocks do you want to sell?
rnsh()
Is the number of stocks you want to sell less than what you have?
Yes
No
Update stock number and dump it in respective pkl file
Prompt: You don't have enough shares
Is portfolio empty?
Yes
No
Prompt: Your portfolio is empty
Display portfolio
Is portfolio empty?
Yes
No
Prompt: You have no stocks in portfolio
Fetch data from Yahoo Finance and show total worth
Enter starting date
Fetch data from yf
Calculate gains wrt present and starting date stock prices
Prompt: Which stock do you want to plot?
rnsh()
Prompt: Choose starting date
rnsh()
Prompt: Enter intervals
Plot the stock in a new window
Prompt: How much money do you want to save?
rnsh()
Prompt: Upload bank statement
rnsh()
Calculate savings based on % defined for each category
Plot a bar graph comparing current and modified expenses, display time for which it has to be followed and generate analysis report in html format
Update in MongoDB database by running db_transactions.py in a separate terminal

Components of the Project

Creation of Telegram Bot

1) Getting the API Key

The Bot API is an HTTP based interface created for developers having a keen interest in building bots on Telegram. The bot is accessed over the servers using this API Key. For getting the API Key, we need to interact with a bot on telegram named the BotFather. Now, follow the commands, choose a unique username for the bot and the authorization token would be generated.

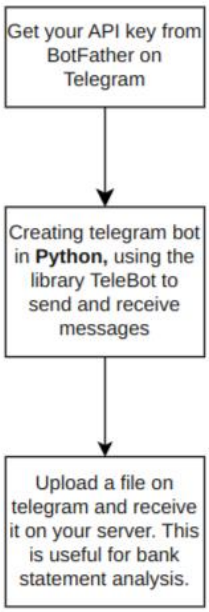
2) Creating telegram bot in Python, using the library Telebot

Firstly install the required python packages. Then import Telebot for setting up the bot and the `os` library in order to read the environment variables stored in our system. Next to set up the initial bot we can define a simple message handler that takes in the command and has a function defined based on the result returned.

3) Upload a file on telegram and receive it on your server

For making a complete financial analysis, we would require the bank statement. The user would upload the csv file and the bot needs to receive it on the server. When the bot gets a content from the user which has the type 'document' then it gets the url of the file, uses the pandas module to read the csv file through the url, performs the analysis and returns an html file if the csv file has been received successfully else throws an error.

Flowchart



Creation of Dummy Data

For simplification, four sets of dummy data were created, which are very similar to the original bank statements in csv format. We used Mockaroo for the dummy data creation. To make it as realistic as possible

yet simple, the various columns added in the dataset were **row** (row number), **Txn Date** (transaction date), **Description** (category of expense like grocery, books, etc), **Transfer To/ From** (true value in the column indicates debit and a false value indicates credit), **Ref No./ Cheque No**., **Debit**, **Credit**, **Balance** and **Class** (essential, non-essential, important, non-important tags).

We randomly assigned the Class tags in the dummy data for the sake of simplicity. We intend to incorporate an AI model in the future to classify the expenses into the 4 categories mentioned. The Balance column, which required computations based on the previous row Balance amount and the current row Debit and Credit, was generated in Excel. The final dataset was then converted to the csv format.

Here is a preview of what our dummy dataset looks like.

```
[2]: data.head(5)
```

[2]:

	row	Txn Date	Description	Transfer To/From	Ref No./Cheque No.	Debit	Credit	Balance	Class
0	1	25/12/2022	Home	True	1216711285639	4305.53	0.00	366736.40	Important
1	2	25/12/2022	Movies	False	1628188928766	0.00	4931.10	371667.50	Non-important
2	3	25/12/2022	Tools	False	1845200544154	0.00	1068.89	372736.39	Essential
3	4	25/12/2022	Computers	True	1158881602293	3224.64	0.00	369511.75	Non-important
4	5	25/12/2022	Baby	True	1155313314792	3929.41	0.00	365582.34	Non-Essential

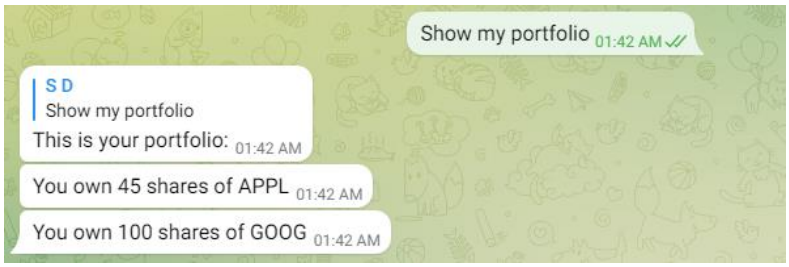
Portfolio Management

The bot allows the user to manage their portfolio, by giving them access to the following features:

- Adding stocks to portfolio

- Removing stocks from portfolio

- Showing their portfolio



These features are very useful to the newbie trader, who can view and modify their portfolio with a simple chat prompt.

On initially running the file 'creating_portfolios.py', the user creates 5 default portfolios on their system, each belonging to the members of this group. Now, when the bot is started using the /start command, you can login by entering the name of the portfolio holder. Now you will successfully be able to access and modify your portfolio, which is always kept up to date with all the transactions you have done in previous logins.

Real Time Market Information

The bot continuously accesses real time market information through a library called Yahoo Finance, which fetches market information in the form of a Pandas DataFrame containing columns that are highly suitable for out analysis.

This library enables us to run the following features:

- **Displaying portfolio value:** Fetches the present day price for all the stocks in your portfolio and shows a total valuation.

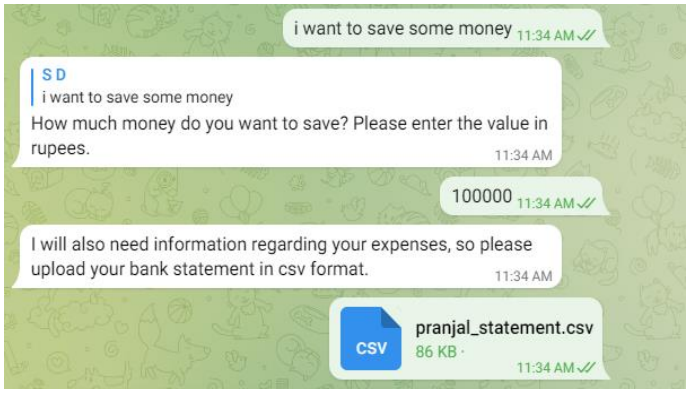
- **Displaying portfolio gains over time:** Enables you to choose a starting date, and then fetches the price of the stocks in your portfolio on that date, and on the present day. Then it computes the absolute gains(difference in value of portfolio) and relative gains(as a percentage) from your portfolio, and hence you can analyze whether you should or should not sell or buy certain stocks.

- **Plotting the candlestick chart for a stock:** The candlestock chart is something that is of very high importance to a trader. Upon being prompted, the bot plots a highly interactive chart for a ticker from a specified starting date till the present date, enabling the user to analyse price change patterns over a certain period of time and with an interval of their choice too. The plot is made interactive, if the user hovers their mouse on any of the candlesticks, they would be able to see more information of the movement of the price in that interval too.

Bank Statement Analysis

In this project, we have enabled a feature where user could upload their financial goal and monthly expense along with the tags whether they were important/ non-important and our bot outputs an estimated monthly budget and the time duration for which they should follow the new budget so as to additionally save the money required for their goal.

The program calculates the user's original savings using all the credits and debits of his statement and then prepares a new budget and using that it calculates for how long the user will have to save to be able to reach their goal. Additionally, it also displays a bar graph showing the difference in current and expected expenditures.

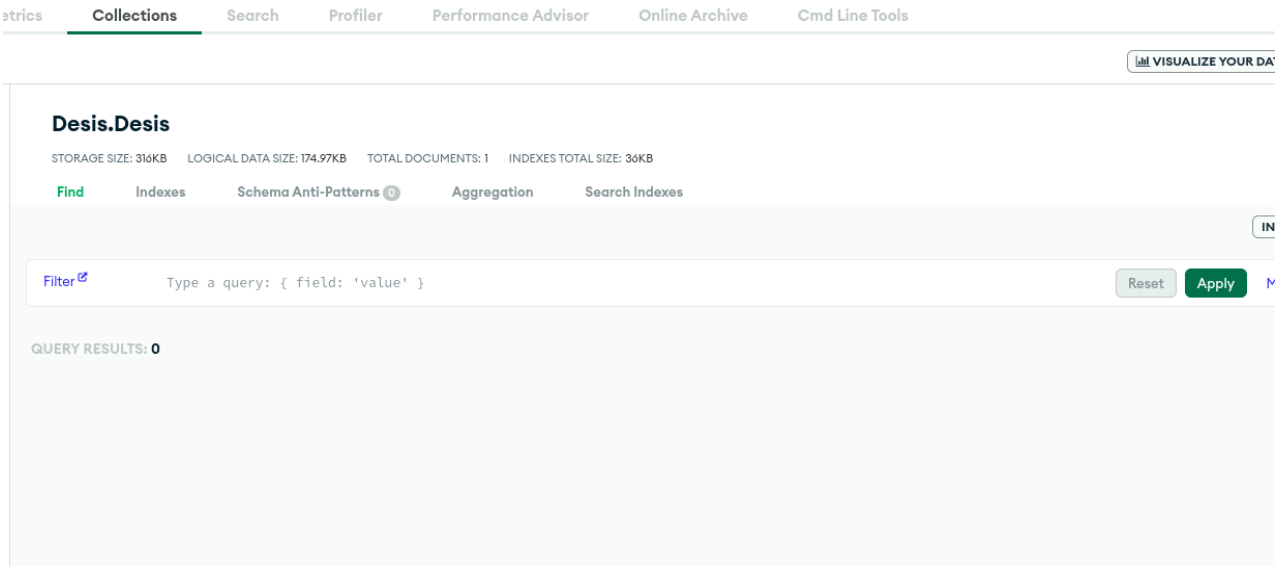


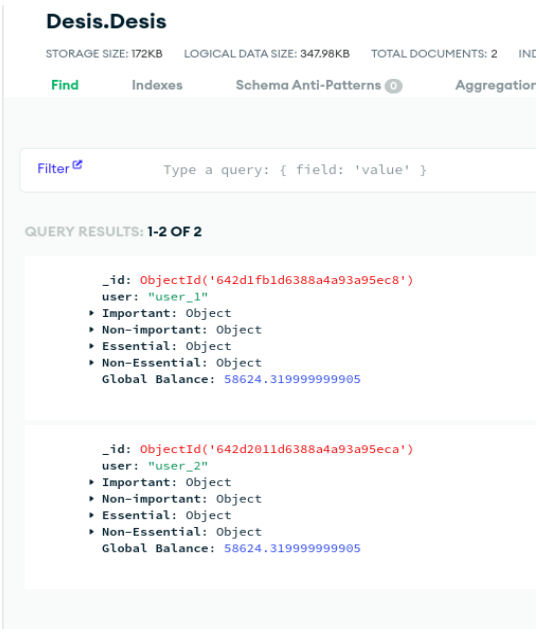
	Unnamed: 0	row	Txn Date	Description	Transfer To/From	Ref No./Cheque No.	Debit	Credit	Balance	Class	User	Estimated Expense	Saving
0	0	1	23/12/2022	Toys	False	2235056301457	0.00	1453.89	540283.71	Important	pranjal	0.00	0.00
1	1	2	23/12/2022	Clothing	False	2153874362292	0.00	1995.38	542279.09	Essential	pranjal	0.00	0.00
2	2	3	23/12/2022	Industrial	True	2156143492489	2899.33	0.00	539379.76	Important	pranjal	2899.33	0.00
3	3	4	23/12/2022	Games	False	2003697616024	0.00	1532.03	540911.79	Non-Essential	pranjal	0.00	0.00
4	4	5	23/12/2022	Clothing	True	2086344215661	2355.73	0.00	538556.06	Non-Essential	pranjal	1884.58	471.15
5	5	6	23/12/2022	Music	True	2490790716171	1973.11	0.00	536582.95	Essential	pranjal	1775.80	197.31
6	6	7	23/12/2022	Outdoors	True	2149767778002	2604.32	0.00	533978.63	Important	pranjal	2604.32	0.00
7	7	8	23/12/2022	Grocery	True	2016170275768	2910.13	0.00	531068.50	Essential	pranjal	2619.12	291.01
8	8	9	23/12/2022	Grocery	False	2096494119394	0.00	1794.64	532863.14	Non-Essential	pranjal	0.00	0.00
9	9	10	23/12/2022	Health	False	2717243753532	0.00	2805.57	535668.71	Important	pranjal	0.00	0.00
10	10	11	23/12/2022	Home	True	2694610074296	1295.42	0.00	534373.29	Essential	pranjal	1165.88	129.54
11	11	12	23/12/2022	Toys	True	2917605618306	1727.37	0.00	532645.92	Non-Essential	pranjal	1381.90	345.47
12	12	13	23/12/2022	Computers	False	2279594515245	0.00	2885.79	535531.71	Essential	pranjal	0.00	0.00
13	13	14	23/12/2022	Beauty	False	2742918496447	0.00	2915.08	538446.79	Non-Essential	pranjal	0.00	0.00
14	14	15	24/12/2022	Clothing	False	2153385917337	0.00	1948.11	540394.90	Non-Essential	pranjal	0.00	0.00
15	15	16	24/12/2022	Grocery	True	2648207424484	1613.02	0.00	538781.88	Essential	pranjal	1451.72	161.30
16	16	17	24/12/2022	Books	True	2558846261879	2146.17	0.00	536635.71	Essential	pranjal	1931.55	214.62
17	17	18	24/12/2022	Baby	False	2222389139458	0.00	1622.53	538258.24	Essential	pranjal	0.00	0.00
18	18	19	24/12/2022	Toys	False	2386456912918	0.00	748.71	539006.95	Non-Essential	pranjal	0.00	0.00
19	19	20	24/12/2022	Baby	True	2584315428394	1180.67	0.00	537826.28	Important	pranjal	1180.67	0.00
20	20	21	24/12/2022	Music	False	2637505662152	0.00	1477.54	539303.82	Non-important	pranjal	0.00	0.00

Database Management using MongoDB

- **Connecting to a MongoDB Atlas** database using the MongoClient class from the pymongo module. After connecting to the MongoDB Atlas database, the code creates a dictionary called user_data. This dictionary will store the processed data before it is inserted into the database. It contains keys for the user name, financial categories, and a global balance. By creating this dictionary, the code can organize the financial data in a structured way, making it easier to insert into the database.
- **Reading data from a CSV file** using the csv.DictReader class from the csv module. For each row in the CSV file, creating a dictionary transaction containing information about a financial transaction, such as the transaction date, description, and amount.
- **Determining the category for each transaction**, and storing the transaction in the appropriate sub-dictionary within the user_data dictionary.
  Once the transactions have been categorized, the code updates the balance for each category and for the overall account. By doing this, the code can keep track of the overall financial situation and make sure that everything is balanced correctly.

- **Updating the balance for each category** and for the overall account. Inserting the user_data dictionary into the user_data_collection in the MongoDB database using the insert_one method. By inserting the data into the database, the code can access and analyze the data more easily and efficiently.





Libraries Used

The main libraries we used are:

1. **<u>NeuralIntents (modified!)</u>:** The main library of this program is NeuralIntents. It is a PyPI library, developed by the open source community of Python programmers. This library created a neural network based NLP model that is capable of taking sentences as inputs and mapping them to the intents specified in the intents.json file with certain probabilities. However, this library is long due updating, and the functions, frameworks and optimizers used in the original library are now deprecated. In order to make the model actually work and further make mappings stronger, we had to make multiple changes in the library, such as changing the older versions, implement a function that displays probability of mapping

an input to an intent as well as improving and adding to the intents.json file so it has more options to map the input to.

2. **Matplotlib:** For plotting expected savings plot, i.e., bar graph showing the difference in current and expected expenditures.
3. **Ploty:** Used for plotting interactive candlestick charts when the bot is prompted. The chart displays extra information about each candlestick on hovering the mouse pointer on it, which is given by Yahoo Finance.
4. **Yahoo Finance:** A well maintained library that keeps track of real time stock prices in the form of a Pandas DataFrame with columns that are just suitable for our analysis. Data for a particular stock can be downloaded from this library by specifying a start date and an interval period. It maintains a huge database of information, and can fetch very old as well as very new information, with interval period as long or as short as the program requires.
5. **Telebot:** Used for integrating functions that were functional from the terminal to a highly interactive bot on telegram, an interface that is used all across the globe and is very familiar to the newbie user. Initially, the bot was fully operational from the terminal, but our team quickly realized that this isn't user friendly, and that defeats the purpose of this project. Hence, we integrated our code with the functioning and replying systems of a Telegram Bot by using this library. Each reply from the bot and each input from the user corresponds to one function, and we had to implement that very specifically, forming an intricate flow of information between a variety of functions while also performing mathematical analysis, handling databases and plotting data.
6. **Pandas:** Used for handling the csv bank statement, as well as data received from Yahoo Finance
7. **Numpy:** Used for mathematical calculations
8. **PyMongo:** Used for integrating Python bot with MongoDB Atlas.

and many more…

Challenges We Faced

- Initially we found it difficult to upload the bank statement in our telegram bot and use the file to get an analysis report.
- Another problem we faced was the transfer of graphs plotted on the bank statement analysis report by matplotlib to telegram chat.
- We had tried using the Neural Intents library to work with intents which were to be used in our chatbot. However, the original library used older versions of keras. To use it in our code, we had to update the functions ourselves.
- Integrating terminal operational functions with the messaging system in Telegram.
- Integrating MongoDB Atlas with the bot required the bank statement data to be updated every time a new statement was uploaded, this was a highly tricky task to figure out the proper table structure to input all the use cases, and integrate the backend mongo dp with bot framework.

What we learnt from the project

Through this project, we learnt the following skills:

- Integration of telegram bot with Python script.
- Using NeuralIntents library to be able to create interaction between user and chatbot.

- Getting files from telegram to Python script and then returning them back to the user.
- Using MongoDB to store the user's bank statement details and integrating it with python.

Future Scope

In the future, we would like to improve certain features in our project, which we couldn't implement due to lack of time or knowledge.

- Using an AI model to work on bank statements which classifies user's expenses into criterias such as important/ essential and so on.
- Deploy the chatbot on Azure so that others can use our bot.
- Use a stronger NLP model to map intents to input texts, to increase user freedom.

References

- For bot creation: YouTube
- For backend bot framework: YouTube
- For python and mongodb integration: Documentation
- NeuralIntents Library: GitHub
- Stackoverflow for error resolving

Acknowledgements

We would like to express our heartfelt gratitude to the DESIS Ascend Educare 2022 team for giving us the wonderful opportunity to put our knowledge and skills that we have gained from the program, to test through this project. We are highly thankful to our mentor Mr. Sandeep Manthi for taking out time from his busy schedule and guiding us at every step.