-

Mini Project Report

on

Expense Prediction

Submitted by

Group id - 04

Project Members

KRISHNA HITNALIKAR (1032221979)
RONAK PATIDAR(1032221964)
HARSH KATIYAR(1032221852)
AMAN KANTHALIA(1032221904)

Under the Guidance of

Prof. Pramod Mali

School of Computer Engineering and Technology
MIT World Peace University, Kothrud,
Pune 411 038, Maharashtra - India
2024-2025

Abstract

The design and deployment of the long short-term memory neural networks are explored, which fall in the category of recurrent neural networks (RNNs) for general time-series forecasting. The system will be used to help people manage their budgets by predicting daily and weekly expenses from various categories such as food, transportation, entertainment, and other items that users might need.

In this day and age, with speed dominating society and consumption characterizing an era, it is tough for human beings to exercise financial discipline over their proper spending. Many people spend their savings too early, mostly because of unintended spending or lack of knowledge about where they spend their money.

The system essentially looks at historical financial data and makes future predictions about spending by individuals in real time. It provides users with day-to-day and weekly predictions of money to be spent and personalizes insights toward future expenses so that an individual can adjust his or her financial behavior proactively.

This approach of predicting will be placing individuals better in avoiding over-spending hence remain within their monthly budgeting and, therefore eventually maintain long-term financial stability.

This is even more applicable with the use of an LSTM because it will reflect not only the short-term fluctuations but also the long-term trends in spending behavior. It is, therefore, well suited to such a prediction task. Its recurrent structure enables it to keep track of relevant information from previous days' expenditure and generate predictions that take into account not only the patterns of recent days but also the dependency that exists between different types of spending.

It details all the processes involved in the lifecycle of an expense prediction system, starting with problem-domain analysis, where objectives were formulated. It also provides a comprehensive literature survey of the existing techniques in financial forecasting and enumerates the weaknesses of traditional methods, thereby pointing out the advantage of using machine learning models, especially LSTM networks. Such a system discusses technical requirements and design choices related to a model of data preprocessing, model architecture, training methodology, and evaluation metrics. Besides that, the system's performance will be exhaustively experimented with -- a sequence of experiments made on the model which compares the prediction made by it to a series of actual user expenditure data.

This system shall bridge this gap between these conventional budgeting techniques and modern technological advances, helping the user provide an intelligent tool through which he can manage his finances more efficiently. The prediction nature of the system helps the users approach their budgeting in a more proactive manner, thus providing a sense of financial discipline that offers great advantages in decisions made about daily expenditures.

With the advancement of machine learning, this path can be considered one of the possible further developments in using advanced neural networks like LSTM for personal finance management. This report illustrates some practical applications of such systems and demonstrates how artificial intelligence can apply to everyday financial challenges faced by people around the world.

List of figures

List of tables

Contents

Chapter 1:
Introduction

1.1 Challenges in Personal Financial Management

With the hurried lifestyle and increased consumption, proper financial management is a major challenge to one who wishes to hold everything together financially without drowning in debt. People thus struggle, while living expenses continually rise and personal spending grows at unpredictable intervals.

Personal budgeting is, after all a core element of sound financial management. Here, most people have to plan and monitor their spending habits very carefully so as not to go above their limits. However, personal budgeting can sometimes be such a challenge to many people once they are faced with irregular sources of income, unexpected expenses, and the numerous financial decisions individuals have to make daily. Most people end up in financial difficulties by the end of each month due to not planning on how they should manage their incomes and expenses.

1.2 Budgetary Limits and Financial Control

In addition to predicting expenses, the expense prediction system allows users to set individual budgetary limits for various spending categories, such as groceries, transportation, entertainment, and utilities. By setting these limits, users can gain control over their spending in specific areas, ensuring that they do not exceed their planned budget for any category. This feature encourages responsible financial behavior by providing clear boundaries for spending, making it easier for individuals to avoid overspending.

The system will alert users when they approach or exceed their set limits, allowing them to adjust their behavior in real-time. This not only prevents financial imbalances but also helps users identify areas where they may need to cut back. With a clear overview of how their money is being spent across different categories, users are empowered to make informed decisions and prioritize their expenses based on their financial goals.

By incorporating budgetary limits, the system reinforces financial discipline and promotes a proactive approach to managing money, helping users maintain long-term financial stability.

1.2 Expense Prediction System Using LSTM Networks

A good number of people lack control of their spending habits in food, transport, entertainment, and other utilities, which often results in imbalances in the budget and creates pressure on finances.

Impulse buying, unaccounted for expenditures, as well as a lack of transparency as to the expenditures that will be incurred in the future all make it more complicated to budget, and with ease one will find them over-spending without realizing the implications at the long term. In countering this major challenge, this project will introduce an expense prediction system aimed to enable users to gain insight into their spending patterns in the near future.

By exploiting the LSTM neural networks, the system can predict the daily and weekly expenses based on the historical data of the spending of users, so that the users can have a good sense of spending and thus avoid excessive expenditure.

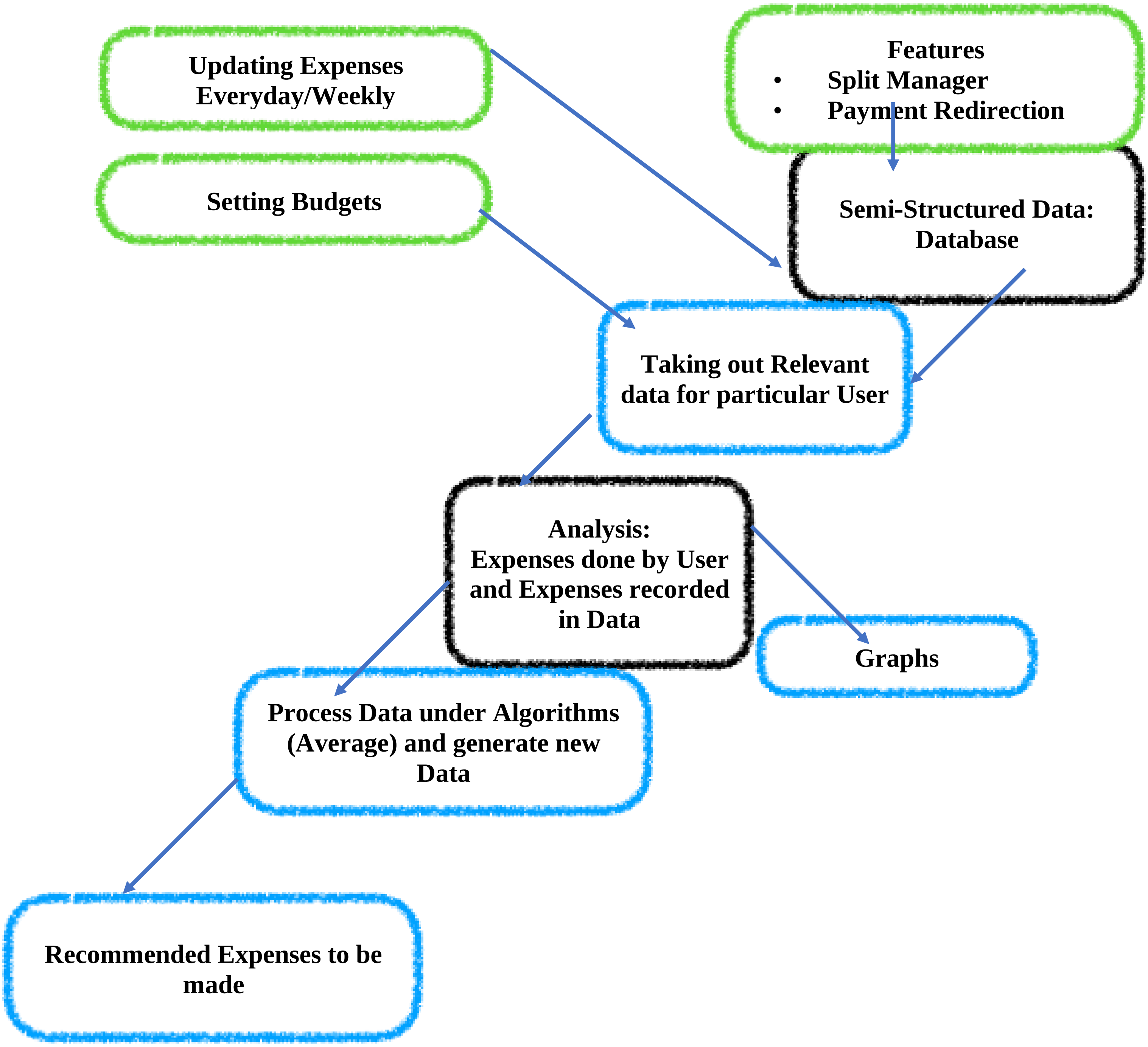
The actual core technology associated with the prediction system of expenses is the LSTM neural network, which is a special kind of recurrent neural network that has been successfully applied for time-series predictions. The use of LSTM networks for tasks that imply sequential dependency in data is apt since the networks retain both short-term and longterm dependencies in the data. This project trains an LSTM model on a user's past spending data, learning how to recognize patterns in the way a person allocates his or her funds across different categories over time. This cost prediction system is very effective for the fact that it can classify expenses and provide a detailed forecast regarding the amount of money an individual would spend on specific groups, such as grocery expenditures, transportation, entertainment, and many others, so that they can identify where their money should be used correctly.

Apart from this, the system has a provision where users can set individual budgetary limits for every category so that there will be no overspending in any particular area. Offering users a clear view of their financial future, it empowers them to control their spending habits and make more informed financial decisions.

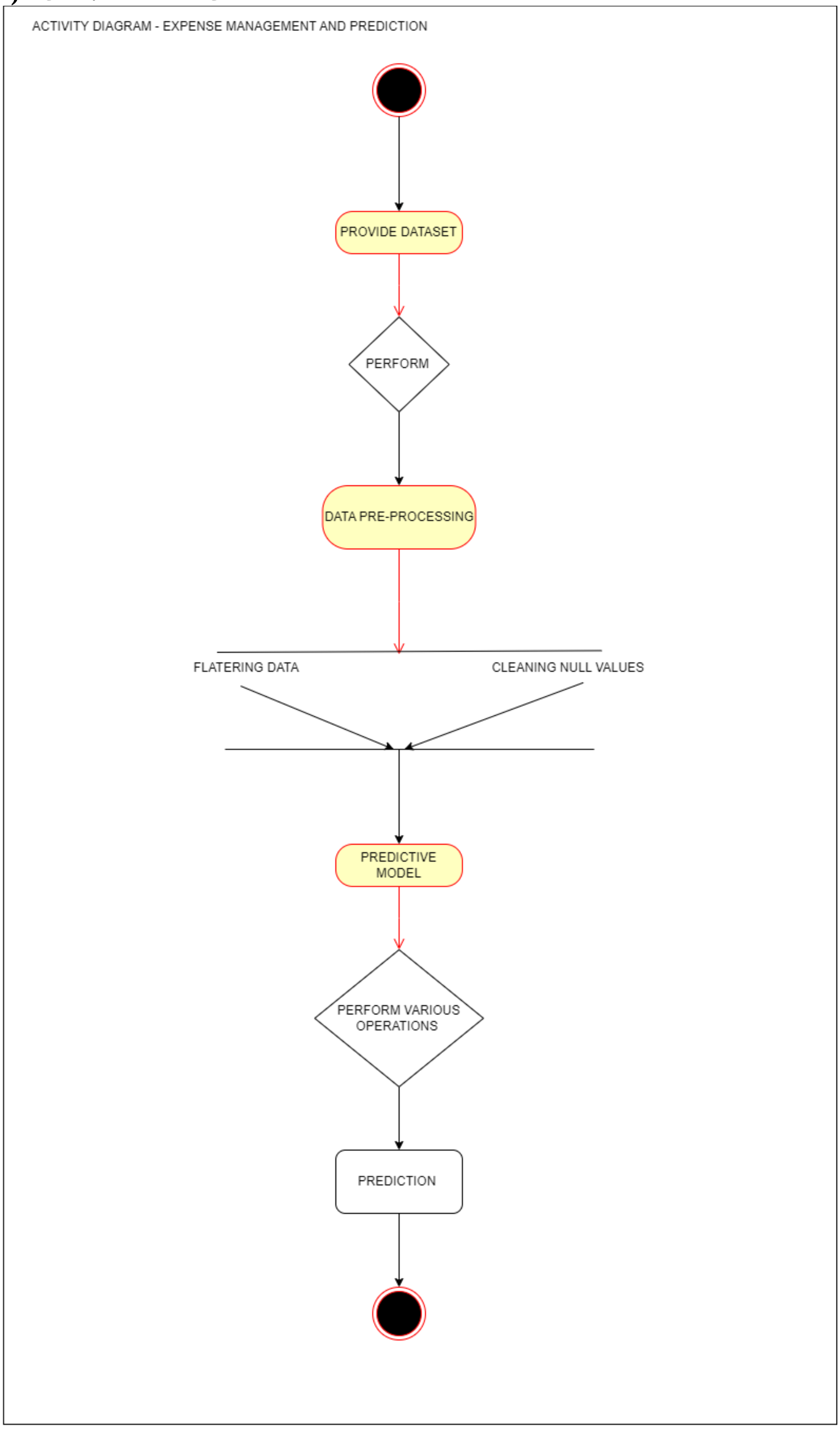
Chapter 2:

BLOCK DIAGRAMS

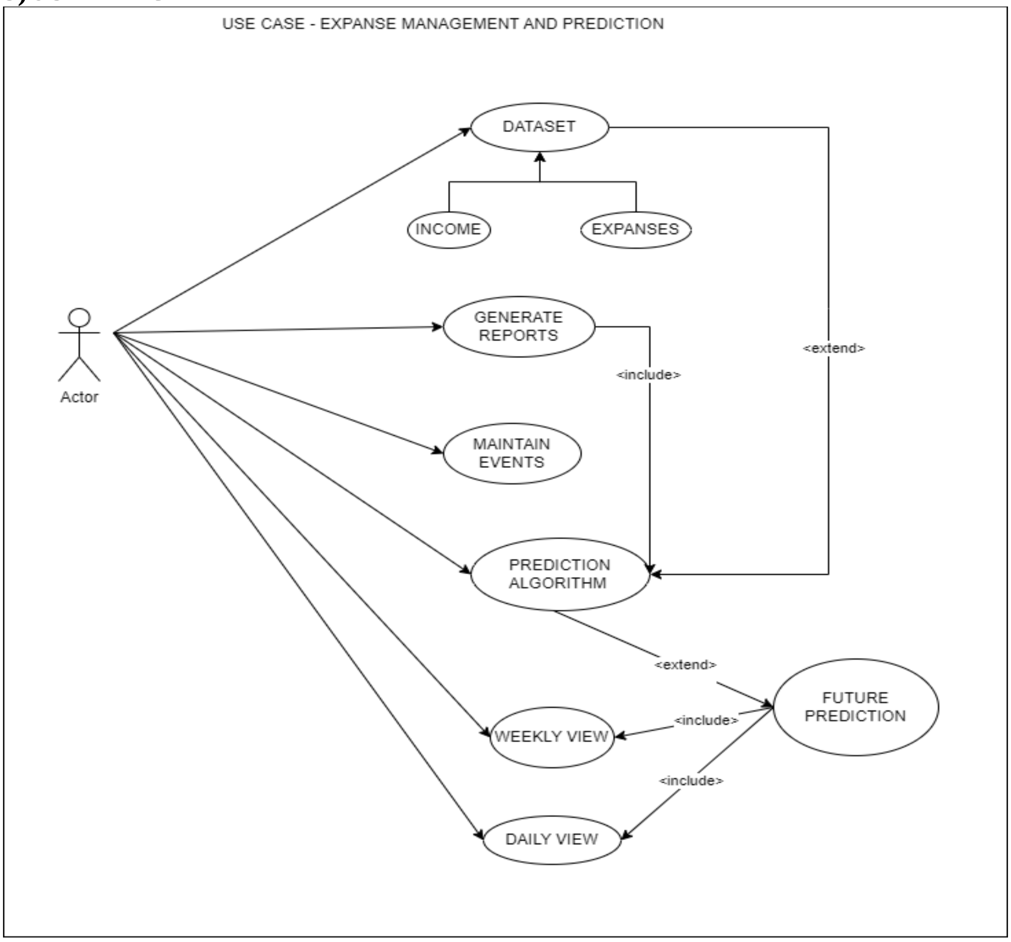
1)Architecture:



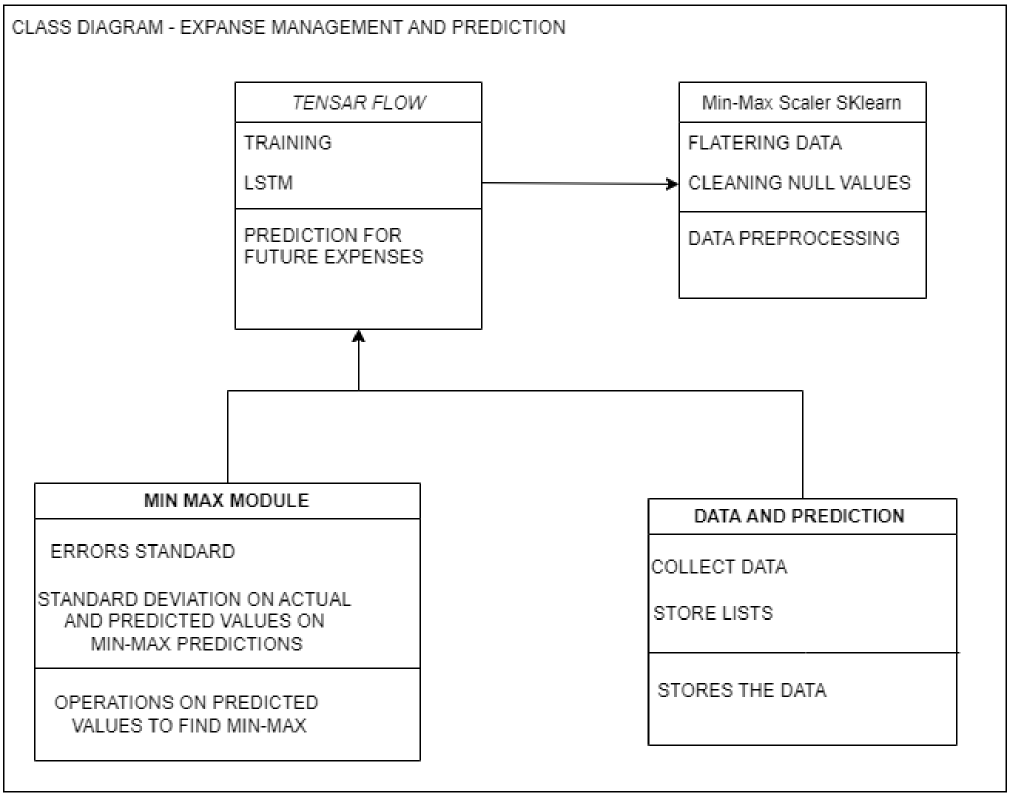
2)ACTIVITY DIAGRAM -



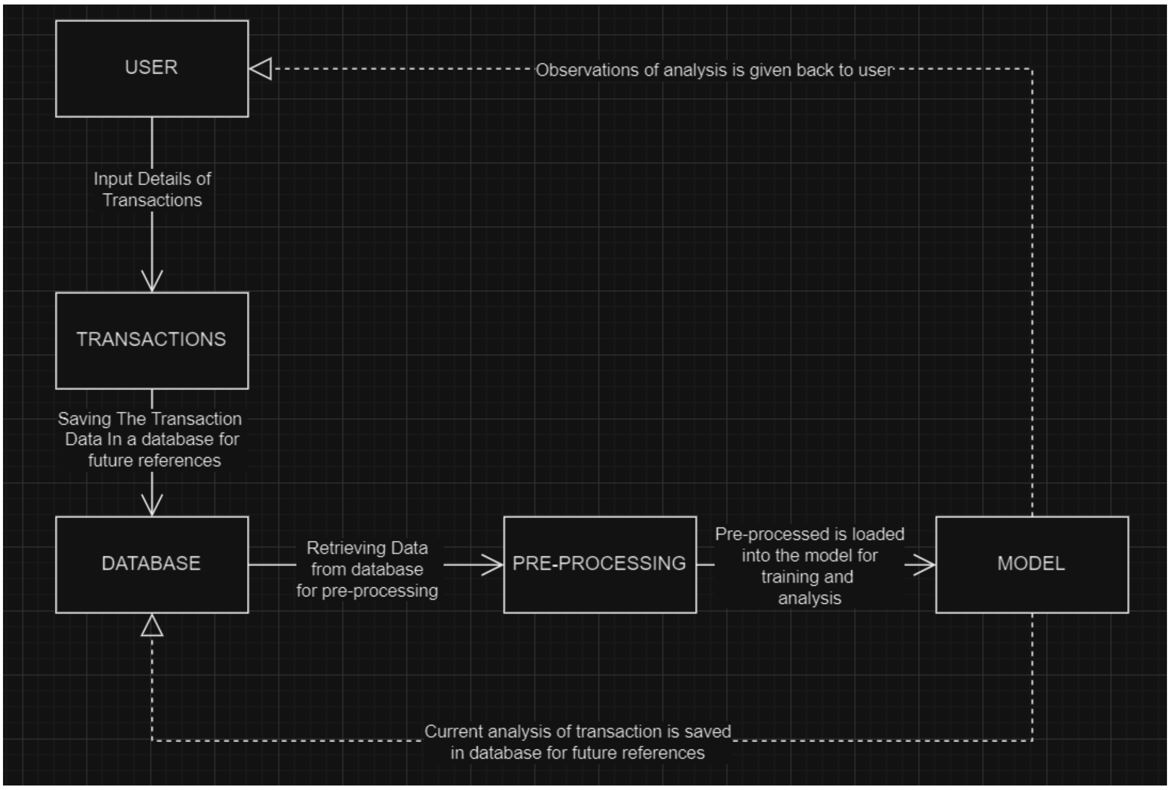
3)USE DIAGRAM -



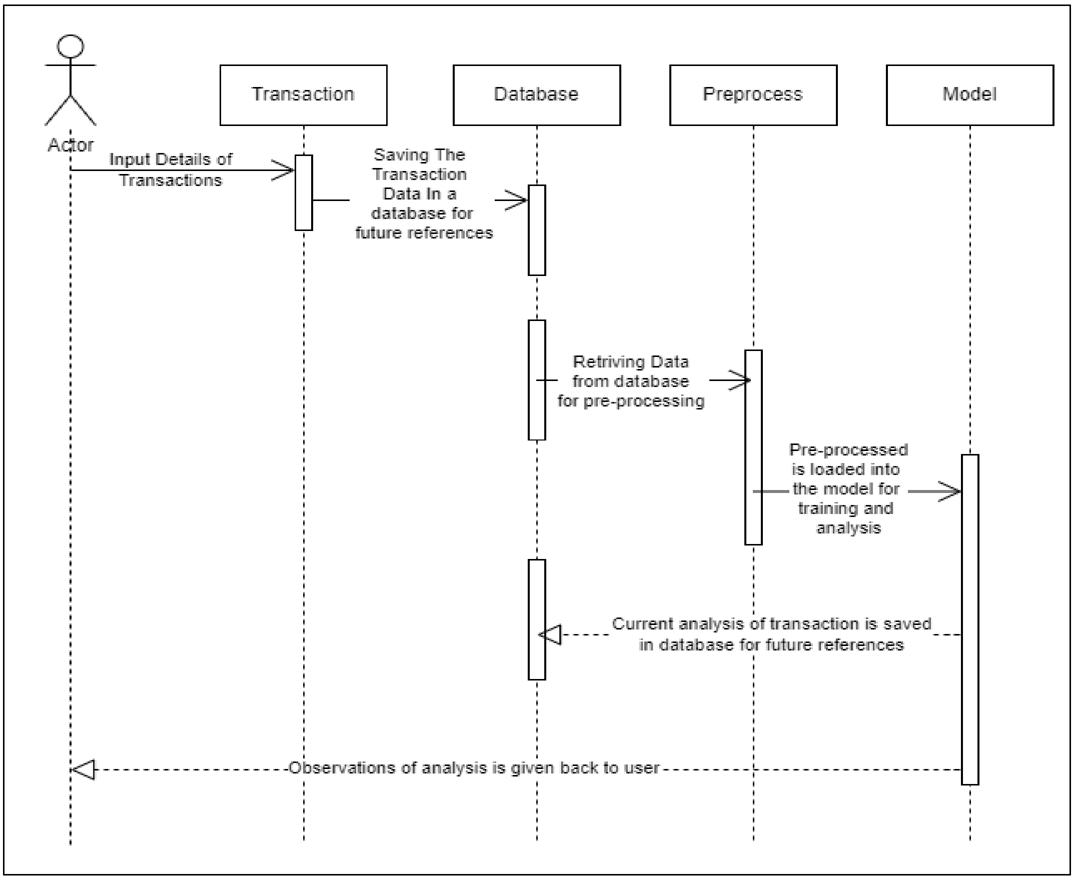
4)CLASS DIAGRAM -



5)COLLABORATION DIAGRAM -



6)SEQUENCE DIAGRAM -



Chapter 3:
Literature Survey

In particular, the area of financial forecasting has been intensively studied, especially with direct relevance to personal finance management. Early techniques for expense forecasting very often employed statistical models such as linear regression, moving averages, and rule-based systems. These types of earlier methods did capture some measure of insight but were often not very effective at properly grasping the complex, non-linear patterns found in many real-world financial data sets, particularly in time-series contexts. But it is that this assumption was not sufficient to cover personal finance and spending habits, which depend on a host of factors; income fluctuations, lifestyle changes, and lots more unexpected expenditures. Thus, these methods would often not accurately forecast personal expenditures.

It has opened the doors for more powerful and flexible solutions, or nowadays, with neural networks specifically with the type called LSTM, becoming a favorite for time-series forecasting. The architecture of LSTM is mainly designed to capture long-term dependencies and is highly useful in the situation where knowledge of the past trends will be helpful in predicting future values. These have successfully been applied in many domains such as the analysis of data in the stock market, sales, and weather forecasting, all of which demand sequential analysis of data and require patterns to be identified over time. So, though LSTM has extensively been used in all of these fields, very little effort has been made to apply it in the prediction of personal expenses.

Most of the personal finance applications available are specifically designed to track rather than predict expenditure. The applications that help a user track expenses are essentially the kind that monitor the spending habits of users rather than providing proactive insight into the future nature of the behavior. Building a predictor model is slightly challenging owing to the apparent nature of variability in personal financial data-spending behaviors can change from day to day, influenced by a myriad of outside elements that make prediction a little tougher. Due to the complexity of personal spending data, very few studies and fewer practical implementations of expense predictor models have been done.

To close the gap, I undertook a detailed survey of all available expense tracker applications, along with research in finance on forecasting and stock market prediction algorithms, to comprehend methodologies applied in areas similar to this. I analyzed how systems manage data variability and unpredictability as well as dived deeper into numerous neural network algorithms, from CNNs to basic RNNs. Considering the advantages and disadvantages of those models, I chose LSTM network since it satisfies all requirements for personal expenditure prediction-the need to use the processing of sequential data while illustrating short-term spikes or dips and long-term trends was a crucial aspect of the given project.

This project sits atop the body of work regarding the success of LSTM in the domain of time-series forecasting. For the very first time, this model is applied within the context of personal finance in terms of a prediction-based scenario. Leverage the power of LSTM to provide daily and weekly expenditure projections across categories with proactive insights into future spending trends toward helping them stay within allocated budgets. Built using expense tracker apps, as well as stock market algorithms and neural network models, this project's basis believes that it is possible for AI-driven predictive systems to create personal finance management.

Chapter 4:
Problem Requirements

4.1 Project Scope-

The scope of the project shall be designing a financial management tool strong enough to predict daily and weekly expenditures as a function of historical financial data. Further, this shall inform users of spending patterns using LSTM neural networks, in addition to aiding the users in their projections of future costs. This actually covers expenditure on food, transport, entertainment, and more while making it possible for a user to set some limits around a budgetary spending. Lastly, it has min-max predictions that allow the system to account for variable changes to spending, thereby providing a clear view of the user's financial outlook.

The main programming environment used in this project is Python, with accompanying libraries TensorFlow when applying the LSTM model on the project, with pandas pertaining to data manipulation, and Matplotlib and Seaborn for visualization. The data source applied in this project falls under the categorization of expenses with their respective data types and is holding real-time bank transaction data in the form of CSV for direct insertion into the data tools of Python. The minimum hardware requirements needed are at least 8 GB RAM with a multi-core processor so that the computational load of the training model can be run smoothly.

4.2 Project Aims-

1. Develop an Expense Prediction System

- The fundamental focus is to design the expense predictive system by using LSTM neural networks to predict one's daily and weekly expenditures with the help of historical financial transactions.

2. Budget Constraints
It allows limits to be put across the categories, including food, transport, and entertainment, in a way that prevents unauthorized overexpansion in any one area.
3. Analysis and Visualization of Data
Makes use of the Python libraries pandas for the manipulation of data as well as Matplotlib and Seaborn for visualization purposes to produce transparent yet insightful analyses.
4. Incorporation of Actual Financial Data
-Use a formated set of past historical data on spending from bank transactions. Such imported dataset is in CSV format for seamless integration and processing.
5. Min-Max Predictions
Design a feature that will compute the standard deviation of the model's prediction errors so that users know what is the confidence interval for variations in expected spending that they will need to prepare for the best cases and worst cases of scenarios.
6. Hardware Requirements and System Performance
-Hardware requirement of at least 8 GB of RAM and multi-core processor, which will be used in carrying out the computation for training the LSTM neural network model; this would enable the system to carry out its processes without hitch.

4.3 Project Resources:

1. Human Resources:

- Data Scientists/Analysts: To design and develop the LSTM model for expense prediction and preprocessing and analysis of the financial data.
- Software Developers: This prediction system will be developed by integrating it with a user interface, so that the users can input budget limits and also view any displayed visualization.
- Project Manager: Over the entire project; the completion will be within time, and management will be ensured over the development team.
- UI/UX Designers: The interface to be developed should be user-friendly to display the predicted expenditure and budget limits.
- Testing Team: Test the software on bugs and checking if it works as desired

2. Software reusable Components

- Data Preprocessing: refers to a set of functions to clean up and preprocess financial data; for example, handling missing values and normalization of data. It could be reused in the context of other financial prediction tasks or reused in other projects.
- Visualization Module: The code for expense graphs and prediction visualizations will be reusable and applicable on any kind of financial or nonfinancial data.

LSTM Model: Applying the same concept of expense prediction model to other kinds of time-series prediction is achievable, such as sales forecasts or even stock prices.

Budget Limiting Module: The codes on the budget limiting module enables the users to limit all budgets in a given category, such as food and transport. This can be reused for financial applications in other industries

3. Software & Hardware Requirements:

Software Requirements

Programming Language Python

Libraries

TensorFlow(for use of LSTM neural network)

- pandas - for data manipulation
- NumPy - for numerical computations
- Matplotlib/Seaborn - to analyze data.
- Scikit-learn - for preprocessing.

Database: A structured CSV file containing real-time bank transactions or financial data categorized as expenses.

-Hardware Requirements:

-Minimum RAM: 8 GB

-Processor: Multi-core processor (Intel i5 or above, equivalent, etc.)

-Storage: at least 500 GB (for data storage and running machine learning models)

Optional : Dedicated GPU: There exists a dedicated GPU like NVIDIA CUDA-enabled for accelerated training on better performance level for LSTM models.

These resources ensure efficient development, testing, and deployment of the expense prediction system and provide users with accurate forecasts and budget management.

Chapter 5:
Implementation

This kind of expense forecasting system was set using a structured approach in terms of preprocessing the data, developing the model, training it, and evaluating it. The historical spending data were normalized to the same scale so as not to bias the model towards more significant numerical values. Also, the data were categorized into different types of expenditure: food, transportation, and entertainment. In this way, the model will learn specific patterns in spending for every category.

The LSTM network architecture defines an input layer, which feeds the preprocessed data to it. It then consists of multiple hidden LSTM layers to capture long-term dependencies and consequently generate temporal patterns from the data. Then finally, the output layer produces a prediction for next-day expenditure.

LSTM Model Back propagation through time and gradient descent are used for optimization of the model in training. It minimizes the error between the predicted expenditures and actual ones. Now, after training this model on the historical dataset, it is evaluated with a separate testing set in terms of accuracy.

The forecasted versus actual expenses visualized enable the user to compare his or her real spending behavior against the forecast values. The system should predict daily and weekly expenses so that users can plan for those periods and avoid overshooting their monthly budgets.

5.1) Code-

- **Model Creation and Training Works as Follows**:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

# Load data
file_path = 'weekly_limit.csv'
df = pd.read_csv(file_path)

df['month'] = 9  # September
df = df.sort_values(by='day_of_month')

data = df[['daily_limit']].values

# Normalize the data
scaler = MinMaxScaler(feature_range=(0, 1))
scaled_data = scaler.fit_transform(data)

def create_sequences(data, seq_length):
    X = []
    for i in range(len(data) - seq_length):
```

```
        X.append(data[i:i + seq_length])
    return np.array(X)

seq_length = 7  # History sequence
X = create_sequences(scaled_data, seq_length)

# Split data into training and testing
train_size = int(len(X) * 0.8)
X_train, X_test = X[:train_size], X[train_size:]

model = Sequential([
    LSTM(50, activation='relu', input_shape=(X_train.shape[1], X_train.shape[2])),
    Dense(1)
])
model.compile(optimizer='adam', loss='mse')

# Train the model
model.fit(X_train, X_train, epochs=50, verbose=1)

all_data = list(data.flatten())
all_predictions = [None] * len(all_data)
all_min_predictions = [None] * len(all_data)
all_max_predictions = [None] * len(all_data)
new_data_feeded=[]
new_predicted_data=[]

plt.ion()  # Turn on interactive mode
fig, ax = plt.subplots(figsize=(12, 6))

def calculate_min_max_with_error(prediction, error_std,actual ,predicted):
    """Returns a tuple with min and max predictions based on error standard deviation."""
    adjusted_error = error_std * np.where(actual > predicted, 1.5, 0.5)
    min_pred = prediction - 1.96 * adjusted_error
    max_pred = prediction + 1.96 * adjusted_error
    return min_pred, max_pred

def calculate_accuracy(actual_values, predicted_values):
    """Calculate accuracy of predictions based on actual values."""
    actual = np.array(actual_values)
    predicted = np.array(predicted_values)

    # Calculate Mean Absolute Percentage Error (MAPE)
    mape = np.mean(np.abs((actual - predicted) / actual)) * 100

    # Calculate Mean Absolute Error (MAE)
    mae = np.mean(np.abs(actual - predicted))

    return mape, mae
```

```
scaled_data_with_input = scaler.transform(np.array(all_data).reshape(-1, 1))
new_seq = scaled_data_with_input[-seq_length:]
new_seq = new_seq.reshape((1, seq_length, 1))

# Make prediction for the next day
new_prediction = model.predict(new_seq).flatten()
new_prediction_inv = scaler.inverse_transform(new_prediction.reshape(-1, 1)).flatten()

# Calculate min and max predictions

# Update the last element in `all_predictions`, `all_min_predictions`,
`all_max_predictions`
all_predictions[len(all_data) - 1] = new_prediction_inv[0]
all_min_predictions[len(all_data) - 1] = None
all_max_predictions[len(all_data) - 1] = None

all_predictions.append(None)
all_min_predictions.append(None)
all_max_predictions.append(None)

scaled_data_with_input = scaler.transform(np.array(all_data).reshape(-1, 1))

# Update the sequence with new data and predict the next 1 day
new_seq = scaled_data_with_input[-seq_length:]
new_seq = new_seq.reshape((1, seq_length, 1))

# Make prediction for the next day
new_prediction = model.predict(new_seq).flatten()
new_prediction_inv = scaler.inverse_transform(new_prediction.reshape(-1, 1)).flatten()

new_predicted_data.append(new_prediction_inv[0])

# Update predictions
all_predictions[len(all_predictions) - 1] = new_prediction_inv[0]
all_min_predictions[len(all_predictions) - 1] = None
all_max_predictions[len(all_predictions) - 1] = None
all_min_predictions = np.array(all_min_predictions, dtype=np.float64)
all_max_predictions = np.array(all_max_predictions, dtype=np.float64)
```

- **Model Prediction Works as Follows:**

```
def update_graph(new_data):
          global all_predictions, all_min_predictions,
all_max_predictions,new_data_feeded,new_predicted_data,budget,i,avg
   # old_predictions=[]

   budget=budget-new_data
   present_day_avg=budget/(30-i)
```

```
    new_data_feeded.append(float(new_data))

    # Calculate errors between actual and predicted values (after the look-back period)
    errors = np.abs(np.array(new_data_feeded[:]) - np.array(new_predicted_data[:]))

    # Compute the standard deviation of the errors (this helps assess the spread of the predictions)
    error_std = np.std(errors)

    # Update the actual data with new input
    all_data.append(new_data)

    # Expand the all_predictions, all_min_predictions, and all_max_predictions lists
    all_predictions.append(None)
    all_min_predictions.append(None)
    all_max_predictions.append(None)
    # old_predictions.append(None)

    scaled_data_with_input = scaler.transform(np.array(all_data).reshape(-1, 1))

    # Update the sequence with new data and predict the next 1 day
    new_seq = scaled_data_with_input[-seq_length:]
    new_seq = new_seq.reshape((1, seq_length, 1))

    # Make prediction for the next day
    new_prediction = model.predict(new_seq).flatten()
    new_prediction_inv = scaler.inverse_transform(new_prediction.reshape(-1, 1)).flatten()

    print("presnt day average:",present_day_avg)
    # old_predicted_data=new_prediction_inv[0]
    print("Original Prediction:",new_prediction_inv[0])

    percentage=None


    ideal_avg=1/30*i
    if new_limit>=present_day_avg and (present_day_avg/new_prediction_inv[0]<0.8 or new_prediction_inv[0]/present_day_avg<0.8):
        actual_avg=budget/3000
        if (ideal_avg/actual_avg)>=0.5:
            decrement=1-(ideal_avg/actual_avg)
        else:
            decrement=0.5-(ideal_avg/actual_avg)
        new_prediction_inv[0]=new_prediction_inv[0]*decrement
        if new_prediction_inv[0]<=5:
            new_prediction_inv[0]=present_day_avg*0.8
    elif new_limit>=present_day_avg and (present_day_avg/new_prediction_inv[0]>0.8 or new_prediction_inv[0]/present_day_avg>0.8):
        if present_day_avg>new_prediction_inv[0]:
```

```
            new_prediction_inv[0]=new_prediction_inv[0]*1.08
        else:
            new_prediction_inv[0]=new_prediction_inv[0]*0.92
    else:
        actual_avg=budget/3000
        if ideal_avg<actual_avg:
            if new_prediction_inv[0]>new_limit*0.3 and new_prediction_inv[0]<new_limit*1.4:
                incremenet=1+(ideal_avg/actual_avg)
            else:
                if new_limit<present_day_avg and new_limit<new_prediction_inv[0]*0.7 :
                    print("hii")
                    if ideal_avg/actual_avg<0.5:
                        incremenet=1+(ideal_avg/actual_avg)
                    else :
                        incremenet=0.5+(ideal_avg/actual_avg)
                else:
                    print("Hello")
                    incremenet=1-(ideal_avg/actual_avg)
        else:
            print("Hello2")
            incremenet=0.6+(actual_avg/ideal_avg)

        new_prediction_inv[0]=new_prediction_inv[0]*incremenet

    print("Percentage:",percentage)

    # new_prediction_inv[0]=new_prediction_inv[0]*((100+(percentage))/100)
    print( "calculated Prediction :",new_prediction_inv[0])


    prediction=new_prediction_inv[0]
    if new_data>=2.5*(all_predictions[len(all_predictions) - 2]):
        min_pred, max_pred = calculate_min_max_with_error(present_day_avg,error_std,new_data_feeded[0],new_predicted_data[0])
    else:
        min_pred, max_pred = calculate_min_max_with_error(prediction,error_std,new_data_feeded[0],new_predicted_data[0])

    mape, mae = calculate_accuracy(new_data_feeded, new_predicted_data)
    print(f'MAPE: {100-mape:.2f}%, MAE: {mae:.2f}₹')
    new_predicted_data.append(new_prediction_inv[0])
    # Update predictions
    all_predictions[len(all_predictions) - 1] = new_prediction_inv[0]
    all_min_predictions[len(all_predictions) - 1] = min_pred
    all_max_predictions[len(all_predictions) - 1] = max_pred
    all_min_predictions = np.array(all_min_predictions, dtype=np.float64)
    all_max_predictions = np.array(all_max_predictions, dtype=np.float64)
```

- **Visulalization Works As Follows:**

```
ax.clear()

# Plot the actual expenditures
ax.plot(np.arange(len(all_data)), all_data, label='Actual Spending', color='blue')

# Plot the predicted spending
 ax.plot(np.arange(len(all_predictions)), all_predictions, label='Suggeseted Spending',
color='red')
    # ax.plot(np.arange(len(all_predictions)), old_predicted_data, label='Predicted
Spending', color='gree/n')

# Plot the min and max predictions as bands
ax.fill_between(np.arange(len(all_predictions)), all_min_predictions,
all_max_predictions ,
            color='orange', alpha=0.3, label='Prediction Range (Min-Max)')

ax.text(1, 13, f'Accuracy: {100-mape:.2f}%', fontsize=12, color='black',
   bbox=dict(facecolor='none', edgecolor='black', boxstyle='round,pad=0.5'))


ax.set_xlabel('Day of the Month')
ax.set_ylabel('Expenditure (₹)')
ax.set_title(f'Daily Expenditure with 1-Day LSTM Prediction and Range    Accuracy:
{100-mape:.2f}%')
ax.legend()
ax.grid(True)

# Redraw the updated plot
fig.canvas.draw()
fig.canvas.flush_events()
```

EXPLANATION OF CODE –

Develop a machine learning model that predicts how much each day will expend with the help of historical sequences. This loads and reads the data set, that is, daily expenditure limits of each day in a month. The Data file holds in CSV named `weekly_limit.csv` and sets the month to September. After loading the data, they are then sorted based on the day of the month, and daily limits are extracted for model training. To ensure uniformity, spending data is normalized or scaled between 0 and 1 using `MinMaxScaler`. This keeps all values that enter into the model in a fixed range, thus making the model handle the data better.

The code employs a function, `create_sequences()`, that creates sequences of historical data that the model will be fed. Each of these sequences comprises spending data for the preceding 7 days as `seq_length = 7`. The data was split between the training and testing sets with 80% given to the training set. After this setup of data, the code goes on to build a **LSTM** neural network model using Keras. The model consists of an LSTM layer of 50 units with a dense output layer used to predict what to spend on the next day. It is compiled using an

Adam optimizer and MSE as the loss function. Then, train it in prepared data for 50 epochs. At those 50 epochs, the model actually adjusts its weights to minimize errors in predictions.

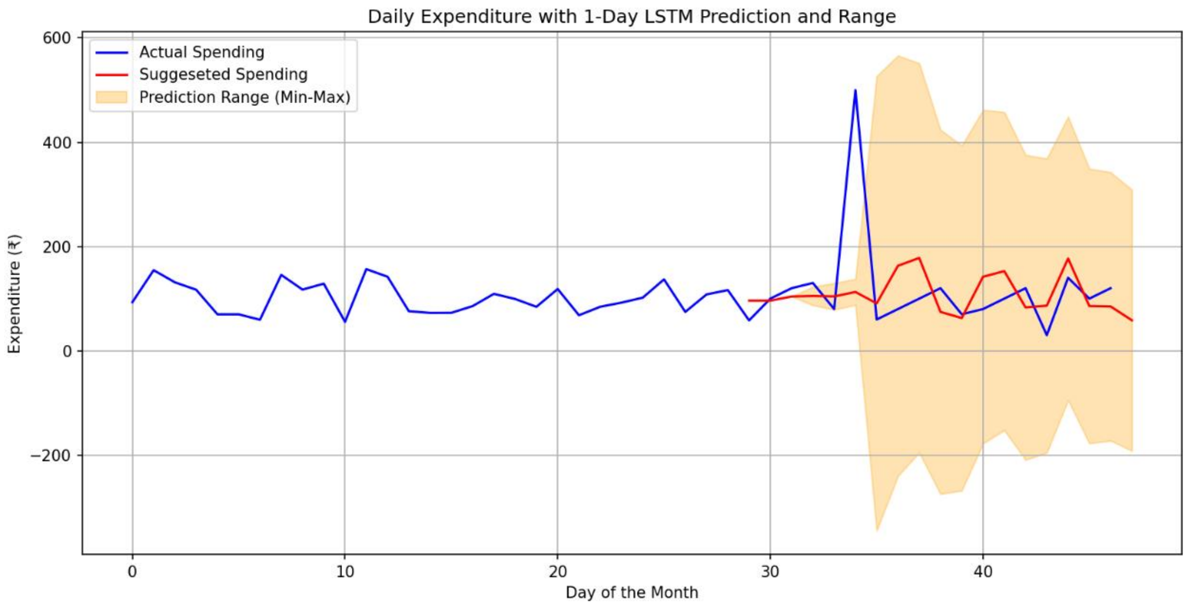
Once the model is trained, it applies the last 7 days of spending to predict the next day's spending. It first is in scaled format, and it calls the function of the inverse of scaling to transform the result into its natural range. This prediction is stored inside a list called `all_predictions`. Then min and max predictions are made for places in `all_min_predictions` and `all_max_predictions`. This enables the tracking of the model's predictions and their corresponding confidence intervals for future predictions.

Now, the graph is updated upon new spending data. As each day's spending data passes through the model, the system appropriately corrects predictions. The remaining budget for the month is updated with the average budget for the remaining days. It also computes the error of actual versus predicted spending, which is an evaluating feature of the model's performance. Whenever a huge difference exists between real and predicted spending, the model adjusts future predictions in order to make them correlate with real data. Further, on the basis of the margin of error, it computes a confidence interval (min and max predictions) that makes an even more precise range of prediction. Besides that, it provides an estimate of the closeness of predictions to real values through the computation of MAPE (Mean Absolute Percentage Error) and MAE (Mean Absolute Error).

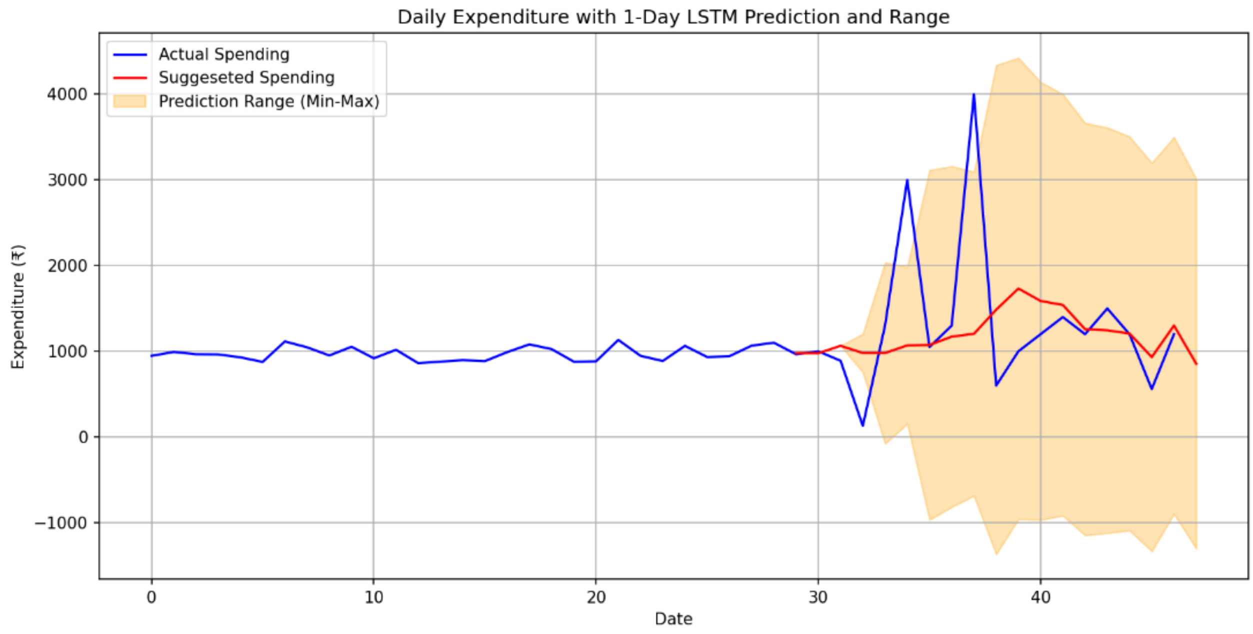
Visualization deals with plotting the actual against the predicted amount of spending. It plots the actual spending data as a blue line, the predicted spending as a red line, and shades the area between the min and max predictions (orange) to indicate the uncertainty in the predictions. The graph is also updated with accuracy metrics, based on MAPE, which gives a clear illustration of how well the model is performing. The graph contains a title, titles for the x and y axes, and it reflects the daily expenditure and prediction accuracy, giving an overall view of actual and forecasted spending.

Chapter 6:
Result and Discussion

The performance of the LSTM model shows promise in predicting daily and weekly expenditure on multi-category levels, which can perhaps reflect the spending in the future by its users. To evaluate the performance of the model, the author adopted some conventional evaluation metrics such as MSE and RMSE, both of which showed satisfactory performance. Results visualization shows the model closely follows actual patterns of spending, especially where budgeting is more regulated, such as food and transport. However, in categories with less regular expenditure patterns, such as entertainment, the model's predictions show slight deviations, indicating an area for improvement related to managing unpredictable spikes in expenditure.



The system can forecast what the near future is going to cost by showing a proactive approach to expense management, thus staying within one's budget of the month without overspending. Once again, the process is user-friendly for even nontechnical people because of how easily one can interpret the results. Further scope for discussion would include the improvement of the model by introducing external influences such as holidays or other special events that may influence the behavior of the people involved in order to enhance the model's predictability.



Despite the effective implementation of the expense prediction system, some drawbacks were detected:
1. Inadequate Training Data: the model fails to make better predictions and calculation as training data are not sufficiently broad and comprehensive.

2. Overprediction and Underprediction: There are instances when the model tends either over to predict or under predict expenses due to unique patterns in previous spending behavior.

3. Boundary Condition-Limitation: Creating mathematical boundaries to avoid extreme outliers at times reduces the flexibility of the model.

4. Category Bias: The model appears to perform better with mundane categories; food items are quite convenient and comfortable, but less frequent or rare expenses such as entertainment do not work well.

5. Uncommon Expenses Sparse Data: Forecasts about infrequent events with high expenditure are not very accurate because the number of instances is far too small.

6. Exclusion of External Factors: The model cares not for how inflationary pressures built on top of previous years, a shift in income, or an emergency may run.

7. Poor Long-Term Accuracy: LSTMs will do very well with short-term predictions but have poor accuracy with respect to long-term expenses.

Chapter 7: Conclusion

In essence, this expenditure prediction project well demonstrates the applicability of LSTM artificial neural networks within the sphere of personal finance management. Through analyzing past spending behavior, the system provides users with daily and weekly expenditure forecasts, enabling them to make more informed choices in their spending habits. The ability of this model to forecast future expenditures puts users within their monthly budgets, saving them the grievous inconvenience of financially crippling stress. The results then indicate that the system is fairly good in most of the categories of expenditure and presents rather reliable predictions in routine expenditure while pointing out areas for potential improvements in more irregular categories of expenditure. All potential improvements would include adding more sources of data-for example, additional information like income levels, economic trends, and seasonal influences-that would tend to improve the model. This project could be viewed as a proof of concept using machine learning techniques with LSTM at the core to tackle real financial problems and thus open doors for further research and development in this field.