-

AIES Mini Project Report

on

EXPENSE PREDICTION

Submitted by
Group id-4

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

This report details the development and implementation of an expense prediction system leveraging Long Short-Term Memory (LSTM) neural networks, a type of recurrent neural network (RNN) particularly well-suited for time-series forecasting. The system is specifically designed to assist users in managing their monthly budgets by predicting daily and weekly expenditures across several key spending categories, such as food, transportation, entertainment, and other necessities.

In today's fast-paced and consumption-driven society, it is increasingly challenging for individuals to maintain financial discipline and control their expenditures. Many people face the issue of depleting their financial resources prematurely, often due to unplanned spending or a lack of awareness about where their money is going.

To address this problem, the primary objective of this system is to analyze historical financial data and forecast future spending patterns in real time. By providing users with daily and weekly predictions, the system offers personalized insights into upcoming expenses, allowing individuals to adjust their financial behavior proactively.

Through this predictive approach, users are better equipped to avoid overspending, ensure that they stay within their monthly budget, and ultimately maintain long-term financial stability.

The use of LSTM in this context is particularly advantageous, as it is capable of capturing both short-term fluctuations and long-term trends in spending behavior, making it an ideal choice for such predictive tasks. The model's recurrent structure allows it to retain relevant information from previous days' expenditures and generate predictions that consider not only the recent patterns but also the dependencies that exist between various categories of spending.

This report explores the entire development lifecycle of the expense prediction system, beginning with an analysis of the problem space and the formulation of objectives. It also presents a comprehensive literature survey of existing techniques in financial forecasting, highlighting the limitations of traditional methods and the benefits of machine learning models, particularly LSTM networks. The report delves into the technical requirements and design choices involved in building the system, including data preprocessing, model architecture, training methodology, and evaluation metrics. Furthermore, the system's performance is rigorously evaluated through a series of experiments, which compare the model's predictions against actual user expenditure data.

Through the development of this system, we aim to bridge the gap between traditional budgeting techniques and modern technological advancements, providing users with an intelligent tool to manage their finances efficiently. The predictive nature of the system helps users take a more proactive approach to budgeting, encouraging financial discipline and enabling better decision-making when it comes to daily expenditures.

As machine learning continues to evolve, the application of advanced neural networks like LSTM in personal finance management presents a promising avenue for future research and development. This report highlights the practical implications of such systems and demonstrates how artificial intelligence can be applied to solve everyday financial challenges faced by individuals worldwide.

Introduction

In today’s fast-paced and consumption-driven world, efficient financial management has become a key challenge for individuals seeking to maintain financial stability and avoid falling into debt. As the cost of living continues to rise and personal expenditures fluctuate unpredictably, many people find it increasingly difficult to manage their finances effectively.

One of the core components of sound financial management is personal budgeting, which involves carefully planning and monitoring one’s spending habits to ensure that they do not exceed their financial means. However, personal budgeting can be an overwhelming task, especially when individuals are confronted with irregular incomes, unexpected expenses, and the multitude of financial decisions they must make on a daily basis. Without a structured approach to managing their income and expenses, many individuals often find themselves in financial distress by the end of the month.

A significant number of people struggle with managing their spending habits across various categories such as food, transportation, entertainment, and utilities, leading to budgetary imbalances and financial strain.

Impulse purchases, unforeseen costs, and a lack of visibility into future expenses further complicate the budgeting process, making it easy for individuals to overspend without realizing the long-term consequences. To address this pressing challenge, this project introduces an expense prediction system designed to provide users with actionable insights into their future spending patterns.

By leveraging Long Short-Term Memory (LSTM) neural networks, the system is able to forecast daily and weekly expenditures based on users' historical spending data, helping them better manage their finances and prevent overspending.

The underlying technology behind the expense prediction system is the LSTM neural network, a specialized type of recurrent neural network (RNN) that has proven to be highly effective in time-series predictions. LSTM networks are uniquely suited for tasks that involve sequential data, as they have the ability to retain and learn from both short-term and long-term dependencies in the data. In the context of this project, the LSTM model is trained on a user’s past spending data, learning patterns in how they allocate their funds across different categories over time.

One of the key strengths of this expense prediction system is its ability to categorize expenditures and provide detailed predictions for each category. For example, the system can predict how much a user is likely to spend on groceries, transportation, or entertainment in the coming days or weeks, enabling them to allocate their funds more efficiently.

Additionally, the system allows users to set personalized budget limits for each category, helping them stay on track and avoid overspending in any particular area. By offering users a clear view of their financial future, the system empowers them to take control of their spending habits and make more informed financial decisions.

Literature Survey

The domain of financial forecasting has been extensively researched, particularly in the context of personal finance management. Early approaches to expense prediction primarily relied on statistical models such as linear regression, moving averages, and rule-based systems. While these traditional methods provided some insights, they often struggled to capture the intricate, non-linear patterns present in real-world financial data, especially in time-series contexts. This limitation became particularly evident in personal finance, where spending habits are influenced by a wide range of factors such as income fluctuations, lifestyle changes, and unpredictable expenses. As a result, these methods often failed to provide accurate forecasts for personal expenditures.

Recent advancements in machine learning have paved the way for more robust and adaptable solutions, with neural networks, specifically Long Short-Term Memory (LSTM) networks, emerging as a popular choice for time-series forecasting. LSTM's architecture is designed to retain long-term dependencies, making it particularly effective in scenarios that require understanding past trends to predict future values. LSTM models have demonstrated success across various domains, including stock market analysis, sales forecasting, and weather prediction, all of which require the model to analyze sequential data and recognize patterns over time. However, while LSTM has been widely applied in these fields, there has been relatively little exploration into its application in predicting personal expenses.

Most existing personal finance applications focus on tracking rather than predicting expenditures. Expense tracking apps help users monitor their spending habits but do not provide proactive insights into future financial behavior. The challenge of building a predictor model lies in the inherent variability of personal financial data—spending patterns can fluctuate significantly from day to day and are influenced by a range of external factors that make prediction more difficult. Due to the complex nature of personal spending data, there has been limited research and few successful implementations of expense predictor models.

To address this gap, I conducted a thorough survey of existing expense tracker apps, research papers on financial forecasting, and stock market prediction algorithms to understand the methodologies that have been used in related fields. I analyzed how these systems handled data variability and unpredictability, and I also examined different neural network algorithms, including convolutional neural networks (CNNs) and basic recurrent neural networks (RNNs). After evaluating the strengths and weaknesses of these models, I determined that LSTM networks were the most suitable for predicting personal expenditures. LSTM's ability to process sequential data while capturing both short-term fluctuations and long-term trends made it an ideal choice for this project.

This project builds upon the body of work surrounding LSTM's success in time-series forecasting, applying the model to the domain of personal finance for the first time in a predictive context. By leveraging LSTM, this system aims to provide daily and weekly expenditure predictions across multiple categories, offering users proactive insights into their future spending patterns and helping them stay within their allocated budgets. The combination of research from expense tracker apps, stock market algorithms, and neural network models forms the foundation of this project, showcasing the potential of AI-driven predictive systems in improving personal finance management.

Project Requirements

The successful implementation of this project requires both hardware and software resources, along with a well-structured dataset containing historical financial data. The software environment for this project is built using Python, a highly versatile programming language widely adopted in the fields of machine learning and data analysis.

Essential libraries include TensorFlow for constructing and training the LSTM neural network, along with NumPy and pandas for data manipulation and preprocessing. For visualizing the results, libraries such as Matplotlib and Seaborn are employed to generate clear, insightful charts and graphs, helping users better understand their spending patterns.

One of the key components of this project is the dataset, which consists of historical spending data categorized into food, transportation, entertainment, and other recurring expenses. To meet this requirement, data has been gathered from past bank transactions shared by friends and colleagues, providing a real-world dataset for training the model.

The data is preferably stored in a CSV file format, allowing for easy integration with Python's data analysis tools. The dataset should cover at least a full month's worth of daily transactions, segmented by categories, so that the model can capture detailed spending behaviors.

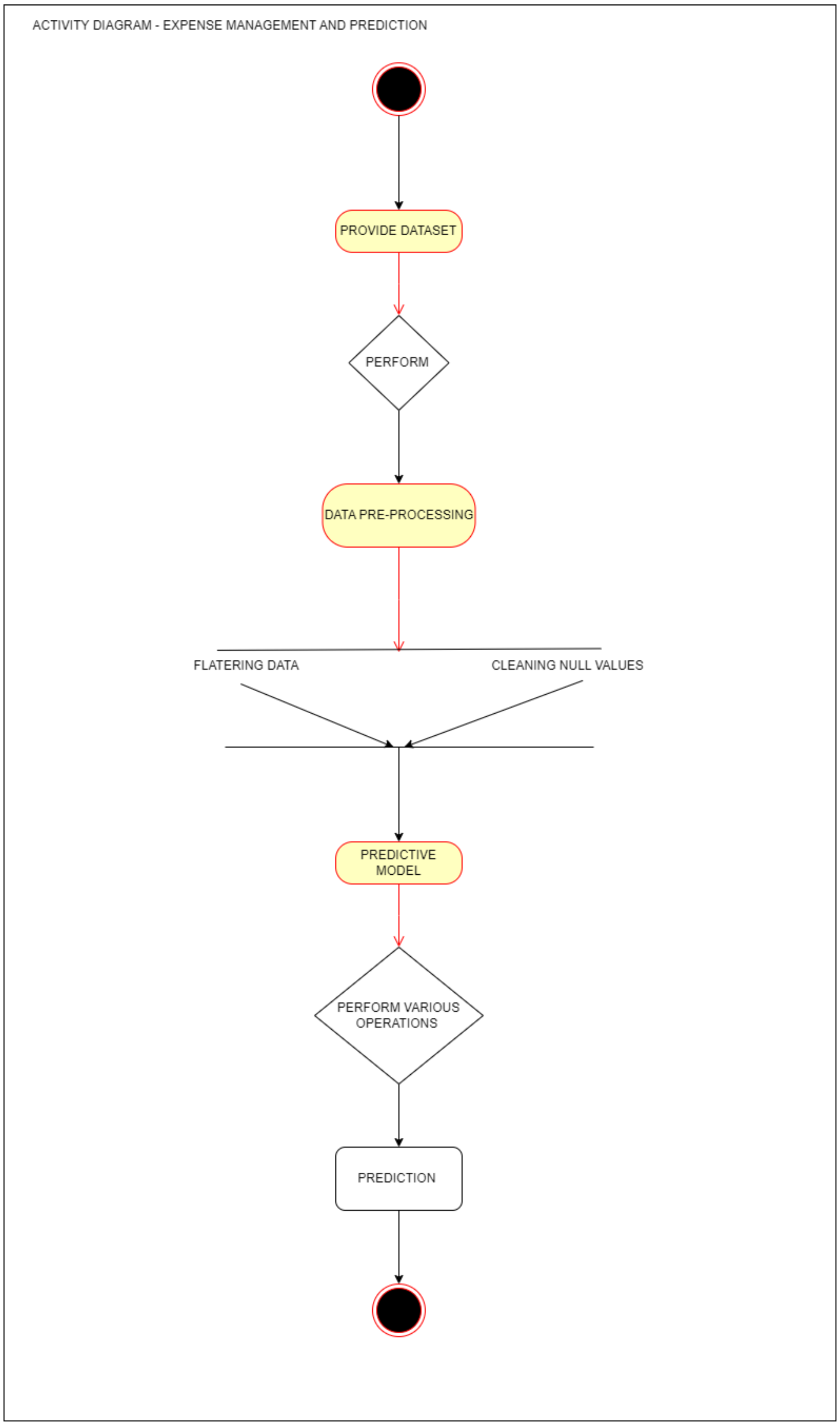
From a hardware perspective, the project demands a computer equipped with a minimum of 8 GB of RAM and a multi-core processor. These specifications are essential for handling the computational load associated with training the LSTM model, which requires significant memory and processing power due to the recurrent nature of the neural network.

Data preprocessing is a critical step, involving tasks such as normalizing the input data to ensure the model performs efficiently. The dataset is split into training and testing sets, using tools from Scikit-learn to ensure an appropriate division of the data for evaluating the model's predictive capabilities. Scikit-learn also provides essential utilities for evaluating model performance, where metrics such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) are used to gauge the accuracy of the predictions.

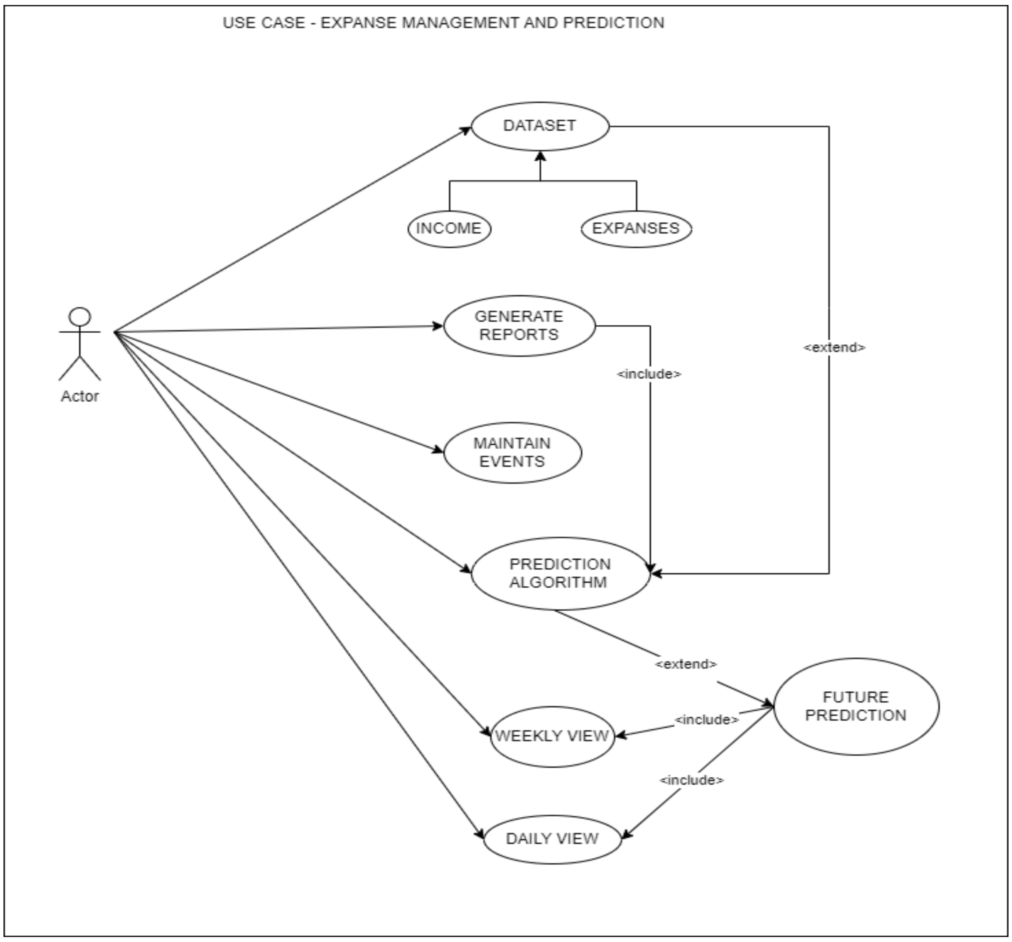
By utilizing TensorFlow for the LSTM model and Matplotlib for visualization, the system ensures a high level of functionality and usability. The combination of past bank transaction data, powerful Python libraries, and a robust computing environment forms the backbone of the project's successful implementation.

In addition to predicting daily and weekly expenditures, the project also incorporates a feature to estimate the minimum and maximum expected spending for the upcoming days. This is achieved by calculating the standard deviation (error_std) of the model's prediction errors over time. By using the error standard deviation, the system provides a confidence interval around the predicted values. The minimum prediction is calculated as the predicted value minus one standard deviation, while the maximum prediction is the predicted value plus one standard deviation. This approach allows the user to anticipate potential variations in their expenditures, offering a more comprehensive financial outlook that accounts for possible fluctuations. The min-max predictions enhance decision-making by allowing users to plan for best-case and worst-case spending scenarios within their budget limits

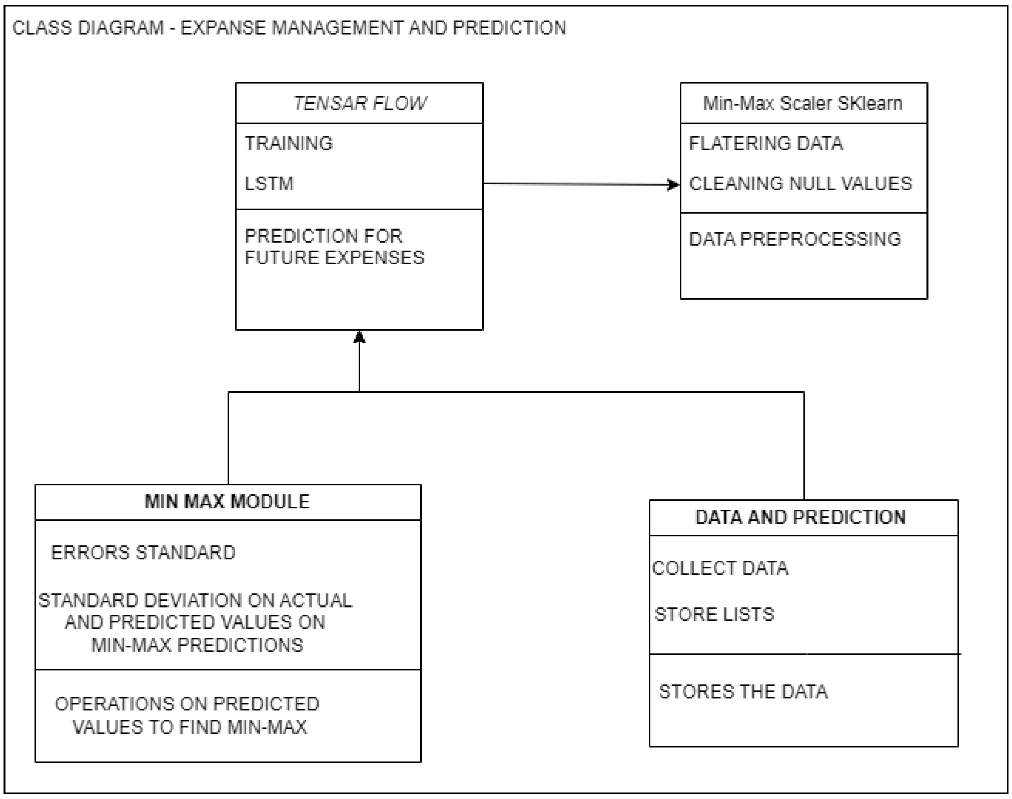
List of Figures



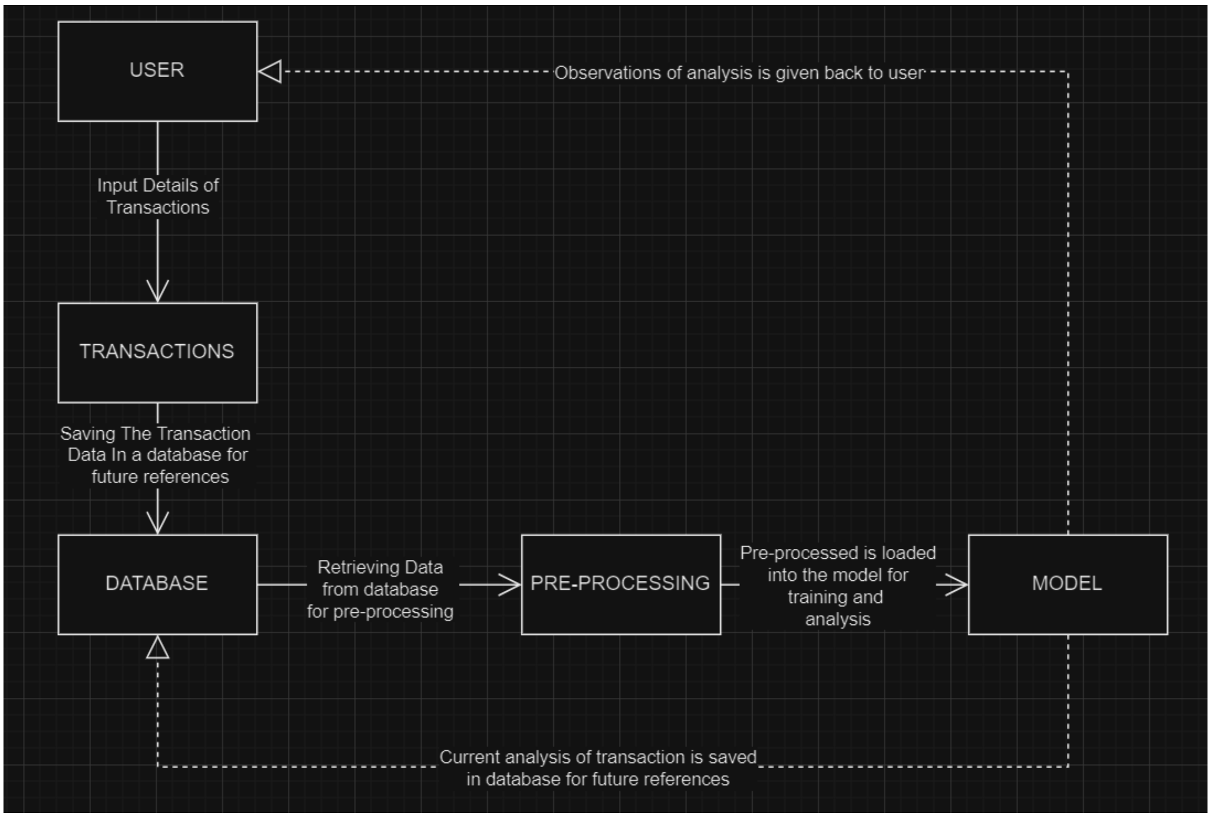
Activity Diagram



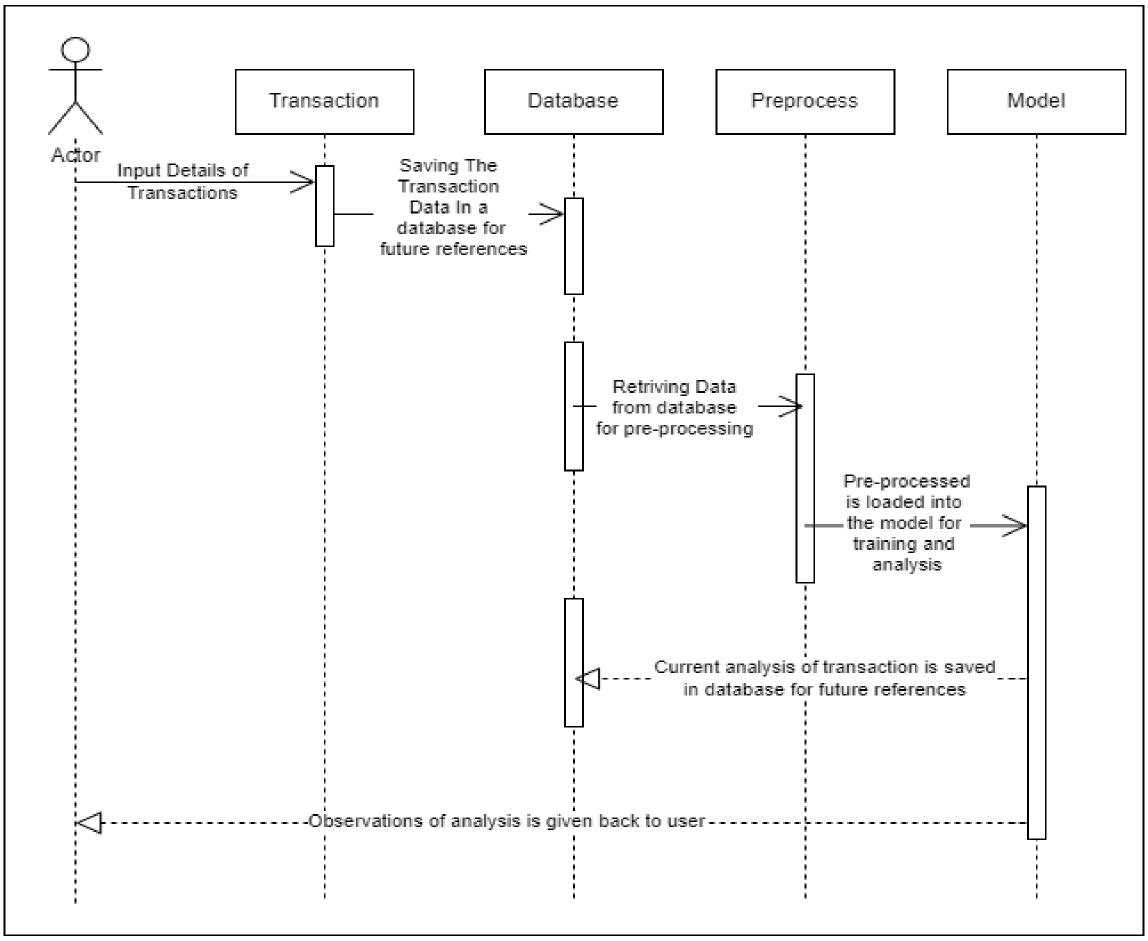
Use Case Diagram


CLASS DIAGRAM - EXPANSE MANAGEMENT AND PREDICTION
TENSAR FLOW
TRAINING
LSTM
PREDICTION FOR FUTURE EXPENSES
Min-Max Scaler SKlearn
FLATERING DATA
CLEANING NULL VALUES
DATA PREPROCESSING
MIN MAX MODULE
ERRORS STANDARD
STANDARD DEVIATION ON ACTUAL AND PREDICTED VALUES ON MIN-MAX PREDICTIONS
OPERATIONS ON PREDICTED VALUES TO FIND MIN-MAX
DATA AND PREDICTION
COLLECT DATA
STORE LISTS
STORES THE DATA

Class Diagram


USER
Observations of analysis is given back to user
Input Details of Transactions
TRANSACTIONS
Saving The Transaction Data In a database for future references
DATABASE
Retrieving Data from database for pre-processing
PRE-PROCESSING
Pre-processed is loaded into the model for training and analysis
MODEL
Current analysis of transaction is saved in database for future references

Collaboration Diagram


Actor
Transaction
Database
Preprocess
Model
Input Details of Transactions
Saving The Transaction Data In a database for future references
Retriving Data from database for pre-processing
Pre-processed is loaded into the model for training and analysis
Current analysis of transaction is saved in database for future references
Observations of analysis is given back to user

Sequence Diagram

Implementation

The implementation of this expense prediction system follows a structured approach involving data preprocessing, model development, training, and evaluation. The historical spending data is preprocessed by normalizing it to a common scale, ensuring the model does not encounter bias towards larger numerical values. The dataset is categorized based on the type of expenditure, such as food, transportation, and entertainment. This allows the model to learn distinct spending patterns for each category.

The LSTM neural network architecture is then defined, consisting of an input layer that feeds the preprocessed data, followed by multiple hidden LSTM layers that capture long-term dependencies and temporal patterns in the data. The output layer produces a prediction for the next day's expenditure.

The LSTM model is trained using backpropagation through time (BPTT) and gradient descent optimization techniques to minimize the error between the predicted and actual expenditures. After training the model on the historical dataset, it is evaluated using a separate testing set to determine its predictive accuracy.

Visualizations of predicted versus actual expenses are generated to allow users to compare their real spending behavior against the forecasted values. The system is designed to predict daily and weekly expenses, helping users adjust their spending habits accordingly to avoid overshooting their monthly budget.

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)
```

```python
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:

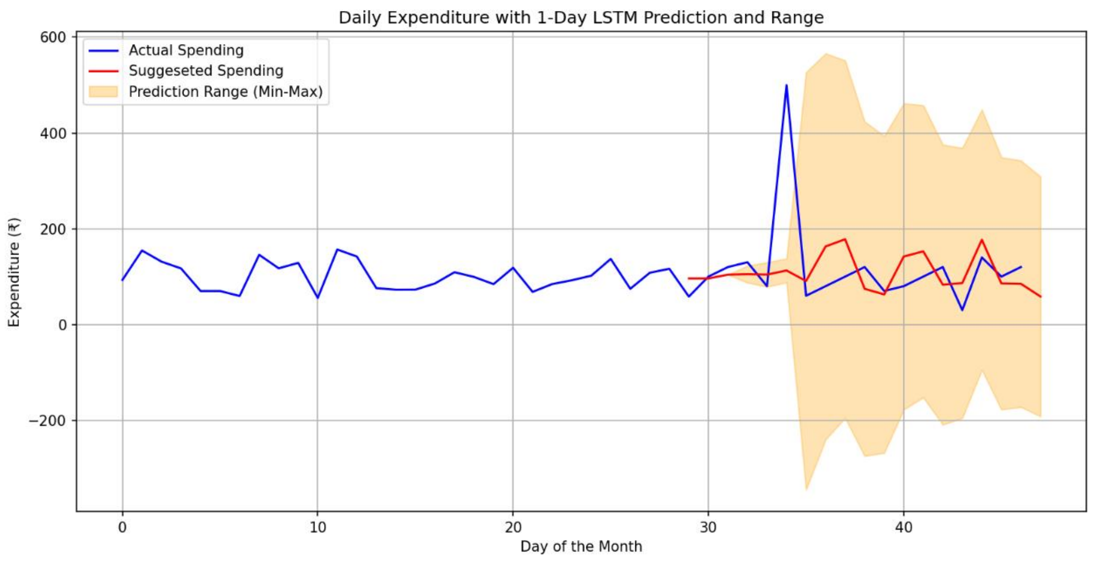
```
def update_graph(new_data):
    global all_predictions, all_min_predictions,
all_max_predictions,new_data_feeded,new_predicted_data,budget,i,avg
    # old_predictions=[]

    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_predi
cted_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)
```

and Visulalization Works As Follows:

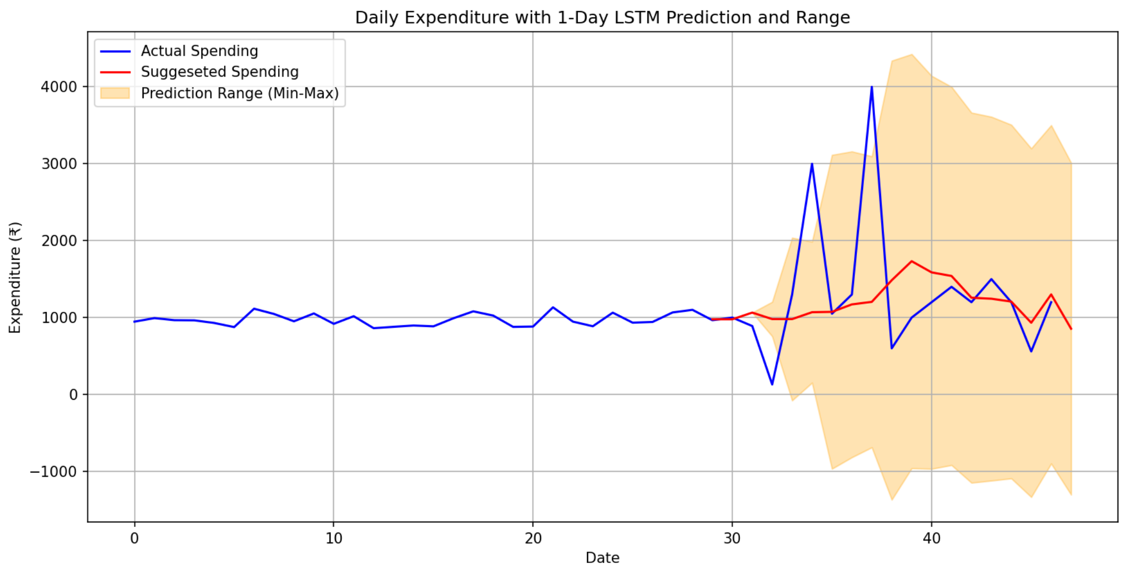
```
    ax.clear()

    # 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()
```

Result and Discussion

The results from the LSTM model indicate that it is highly effective in predicting daily and weekly expenditures across multiple categories, offering users valuable insights into their future spending. The predictive accuracy of the model was measured using standard evaluation metrics such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE), both of which indicated satisfactory performance. Visualization of the results shows that the model closely tracks actual expenditure patterns, particularly in categories where spending is more consistent, such as food and transportation. However, in categories with more irregular spending patterns, like entertainment, the model's predictions exhibit slight deviations, suggesting room for improvement in handling unpredictable expenditure spikes.



The system’s ability to forecast upcoming expenses provides users with a proactive tool for managing their finances, helping them stay within their monthly budget and avoid overspending. Additionally, the user interface allows for easy interpretation of the results, making it accessible even for non-technical users. The discussion also highlights the potential for further improvement, such as integrating external factors like holidays or special events that may influence spending behavior, thereby enhancing the model’s predictive capabilities



.

Despite the successful implementation of the expense prediction system, several flaws were identified:

1. Inconsistent Training Data: The model's predictions are less accurate due to insufficiently diverse and incomplete training data.
2. Overprediction and Underprediction: The model sometimes overpredicts or underpredicts expenses due to variations in past spending behavior.
3. Boundary Condition Limitations: Applying mathematical boundaries to prevent extreme outliers occasionally limits the model's flexibility.
4. Category Bias: The model performs better in common categories (e.g., food) but struggles with irregular or less frequent expenses (e.g., entertainment).
5. Sparse Data for Uncommon Expenses: Predictions for infrequent, high-value purchases are less accurate due to limited data points.
6. External Factors Exclusion: The model does not account for external factors such as inflation, income changes, or emergencies.
7. Limited Long-Term Accuracy: The LSTM model performs well for short-term forecasts but struggles with long-term expense predictions**.**

Conclusion

In conclusion, this expense prediction project successfully demonstrates the application of LSTM neural networks in the domain of personal financial management. By analyzing past spending behavior, the system provides users with daily and weekly expenditure forecasts, empowering them to make informed decisions about their financial habits. The model's ability to predict upcoming expenses helps users stay within their monthly budget and avoid financial strain. The results indicate that the system performs well across various expenditure categories, offering reliable predictions for routine expenses and highlighting potential areas for improvement in more irregular spending categories. Future enhancements could include the incorporation of additional data sources, such as income levels, economic trends, or seasonal factors, to further refine the model's accuracy. This project serves as a proof of concept for leveraging machine learning techniques, particularly LSTM, to solve real-world financial challenges, opening up opportunities for further research and development in this field.