TEAM NO 017

AI-Enhanced Intelligent Money Monitoring System

PROJECT WORK I (REVIEW 3)

Submitted by

JANARTHANAN V K 2122222230051

in partial fulfillment for the

award of the degree of

BACHELOR OF TECHNOLOGY

In

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

SAVEETHA ENGINEERING COLLEGE, THANDALAM

An Autonomous Institution Affiliated to

ANNA UNIVERSITY - CHENNAI 600 025

NOVEMBER 2025

SAVEETHA ENGINEERING COLLEGE

AUTONOMOUS

NAAC A GRADE

Affiliated to Anna University | Approved by AICTE

ANNA UNIVERSITY, CHENNAI

BONAFIDE CERTIFICATE

Certified that this Project report **" AI-Enhanced Intelligent Money Monitoring System "** is the bonafide work of **Janarthanan V K (212222230051),** who carried out this project work under my supervision.

SIGNATURE

Mr.Tenali Ravi Kumar

Professor

SUPERVISOR

Dept of Computer Science and Engineering

Saveetha Engineering College, Thandalam, Chennai 602105

SIGNATURE

Dr. Karthi Govindharaju, M.E., Ph.D.,

Professor

HEAD OF THE DEPARTMENT

Dept of Artificial Intelligence and Data Science

Saveetha Engineering College, Chennai 602105.

DATE OF THE VIVA VOCE EXAMINATION: ……

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to our esteemed Founder President **Dr. N. M. Veeraiyan**, our President **Dr. Saveetha Rajesh**, our Director **Dr. S. Rajesh**, and the entire management team for providing the essential infrastructure.

I extend my sincere appreciation to our principal, **Dr. V. Vijaya Chamundeeswari, M.Tech., Ph.D.,** for creating a supportive learning environment for this project.

I am very thankful to our Dean of ICT, **Mr. Obed Otto,** M.E., for facilitating a conducive atmosphere that allowed me to complete my project successfully.

My thanks go to **Dr. Karthi Govindharaju, M.E., Ph.D.,**, Professor and Head of the Department of Artificial Intelligence and Data Science at Saveetha Engineering College, for his generous support and for providing the necessary resources for my project work.

I would also like to express my profound gratitude to my Supervisor, **Mr.Tenali Ravi Kumar, Professor**, and my Project Coordinator **Dr. N.S. Gowri Ganesh**, Associate Professor at Saveetha Engineering College, for their invaluable guidance, suggestions, and constant encouragement, which were instrumental in the successful completion of this project. Their timely support and insights during the review process were greatly appreciated.

I am grateful to all my college faculty, staff, and technicians for their cooperation throughout the project. Finally, I wish to acknowledge my loving parents, friends, and well-wishers for their encouragement in helping me achieve this milestone.

ABSTRACT

In today's world, the rapid and widespread digital shift in banking and commerce has changed how people manage their finances. This ongoing tech revolution includes online banking, digital wallets, investment apps, and e-commerce platforms, which create a constant and overwhelming flow of transactional data. While this influx of information offers new chances for personalized financial analysis and planning, it also creates a heavy mental load for many users. A lot of people struggle to consolidate, manage, and understand their expenses across different digital platforms. As a result, many lack a clear understanding of their spending habits, which can lead to poor financial choices and increased vulnerability. This issue worsens due to the limited capabilities of many financial management tools. These tools often require tedious data entry and leave the interpretation to the user, making it hard to maintain financial responsibility.

In response to these challenges, our project, AIMMS (AI-Enhanced Intelligent Money Monitoring System), presents a next-generation platform designed to simplify the complicated and often stressful processes of managing personal finances. The system's approach is divided into two main parts: fully automated expense analysis and personalized financial guidance. The AIMMS platform uses advanced Artificial Intelligence AI and machine learning ML algorithms to autonomously and accurately categorize a wide range of financial transactions. At the same time, it detects the unique spending patterns that come from each user’s financial history and behavior.

These detailed analyses are processed and turned into clear, easy-to-understand insights. Users receive these insights through engaging, interactive digital dashboards that transform raw data into meaningful financial stories. By providing a continuous stream of personalized, data-driven recommendations and a goal-tracking system, the AIMMS platform helps users become active participants in their financial management. It offers practical suggestions, like identifying redundant subscriptions or pointing out unusual spending, which allows users to improve their saving habits and positively change their long-term financial behaviors.

The goal-setting feature empowers users to achieve short-term objectives through real-time feedback, effectively reducing financial anxiety. By fostering sustainable habits, AIMMS serves as an essential, intelligent ally in securing long-term financial wellness.

SUSTAINABLE DEVELOPMENT GOALS (SDGS) OF THE PROJECT

Decent Work and Economic Growth

- **SDG 8:** Promotes individual economic resilience and financial stability by empowering users to effectively track expenses, manage debt, and achieve short-term savings targets.

Industry, Innovation, and Infrastructure

- **SDG 9:** Leverages AI technologies like TF-IDF and Naive Bayes within a scalable microservices architecture to build innovative, privacy-first financial digital infrastructure.

Responsible Consumption and Production

- **SDG 12:** Encourages sustainable spending habits and reduces financial waste by providing real-time alerts and goal-centric visualizations to curb impulsive consumption.

Quality Education & Financial Literacy

- **SDG 4:** Enhances financial literacy through interactive dashboards and AI-generated insights, educating users on essential budgeting strategies and resource management skills.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AIMMS - AI-Enhanced Intelligent Money Monitoring System

AI - Artificial Intelligence

ML - Machine Learning

UI - User Interface

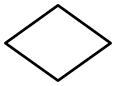
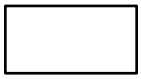
SQL - Structured Query Language

OCR - Optical Character Recognition

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

LIST OF SYMBOLS

S.NO.	SYMBOL NAME	SYMBOL
1.	Usecase	
2.	Actor	
3.	Process	
4.	Start	
5.	Decision	
6.	Unidirectional	
7.	Entity set	
8.	Stop	

Chapter 1

INTRODUCTION

2.1 OVERVIEW OF THE PROJECT

In our connected and digital world, the rapid transformation of global banking and commerce has changed how we manage our money. This digital shift, driven by fintech innovations and the rise of online platforms, generates a constant flow of complex financial data from many sources. While this data-rich environment offers opportunities for personalized financial analysis and wealth management, many users face challenges. The sheer amount and fragmentation of this data often feel overwhelming, making it hard for individuals to keep a clear and accurate view of their financial health. As a result, the need for smart tools that can navigate this complex digital landscape is more important than ever.

Artificial Intelligence (AI) is the key technology that can turn this stream of complex financial data into clear and actionable insights. Using AI in personal finance is not just about convenience; it is an important step in improving financial literacy, encouraging regular saving habits, and helping people take control of their financial futures. In this context, we introduce our project, AIMMS (AI-Enhanced Intelligent Money Monitoring System). This system is carefully designed to be an intelligent and proactive companion for achieving everyday financial wellness. By harnessing the analytical power of AI, the AIMMS platform goes beyond simple expense tracking. It offers users valuable insights into their spending habits, creates personalized saving suggestions, and lights the way toward achieving their key financial goals.

The main goal of this paper is to explore the specific methods and technologies that support our AIMMS platform. In the following sections, we will outline the system's structure, its analytical engine, and the design principles that shape its interface. By the end, readers will understand the

importance of automated financial analysis today, the techniques we used to implement it, and how a system like AIMMS can help users, reduce financial stress, and promote good money management practices.

2.2 PROBLEM DEFINITION

In the contemporary digital economy, managing personal finances is increasingly complex due to data fragmentation across diverse channels like digital wallets and credit cards. This creates a "Financial Fog," preventing individuals from maintaining a consolidated view of their fiscal health. Current solutions remain inefficient, relying on labor-intensive manual entry that leads to user fatigue and data errors.

Furthermore, existing tools lack the analytical capabilities to transform raw transactions into actionable insights. Without automated categorization or visualization, users struggle to identify spending patterns or align daily behaviors with long-term goals. This disconnect fosters financial uncertainty, where immediate spending undermines future stability. Consequently, there is a critical need for an intelligent, unified platform utilizing Natural Language Processing to automate tracking and empower users with the data-driven clarity essential for sustained financial wellness.

Chapter 2

LITERATURE SURVEY

3.1 INTRODUCTION

A literature survey or a literature review in a project report is that section which shows various analysis and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of the project. Once the programmers start building the tool programmers need a lot of external support. This support can be obtained from senior programmers, books or from the websites. It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem statement. Literature survey is the most important sector in the software development process. Before developing the tools and the associated designing the software it is necessary to determine the survey, the time factor, resource requirement etc., The consumer needs regarding online customer service differs from person to person. The needs are also based on each person's personal needs. We need to identify and anticipate these needs in order to completely and accurately meet them.

3.2 LITERATURE SURVEY

3.2.1 TrackEZ Expense Tracker

Author Name : Bhatele P, Mahajan D, Mahajan D, Mahajan N, Mahajan B, & Mahajan P.
Year of Publish : 2023

The paper "TrackEZ Expense Tracker" presents a web-based expense management system designed to eliminate the inefficiencies of manual record-keeping methods like diaries and spreadsheets. The authors developed the application using the Django framework with Python for the backend and SQLite for the database, creating a two-tier architecture that efficiently handles financial data. The system features a user-friendly interface built with HTML, CSS, and JavaScript that allows users to input income and expenses, which are then visualized through dynamic charts and graphs using the ChartJS library. The study emphasizes the application's capability to store transaction history individually for each user, providing a secure and

organized way to monitor daily financial activities and analyze spending patterns over time, ultimately helping users maintain better control over their economic management.

3.2.2 FINANCEGPT: PRECISION FINANCIAL FORECASTING AND BUDGETING FOR SMARTER INVESTMENT STRATEGIES

Author Name : Divya S, Suvarna Smita R S, Sheethal R, Teja V D, Sri Balaji S.
Year of Publish : 2025

The paper "FinanceGPT: Precision Financial Forecasting and Budgeting for Smarter Investment Strategies" introduces an AI-powered financial management tool designed to streamline expense tracking and assist with loan and insurance decisions. The system comprises two primary modules: an Expense Tracker that utilizes Optical Character Recognition (OCR) to extract text from bills and a BERT-based model to classify expenses into predefined categories, and a Loan & Insurance Predictor that evaluates financial profiles. The authors implemented a feature that calculates the Fixed Obligations to Income Ratio (FOIR) to determine a user's borrowing capacity and affordable EMI ranges, while also providing tailored term insurance recommendations. The study demonstrates how integrating AI-driven analytics and threshold-based alerts enhances budget management by automating data entry and providing personalized financial insights, effectively acting as an intelligent personal financial advisor.

3.2.3 Expense Tracking with Tesseract Optical Character Recognition v5: A Mobile Application Development

Author Name : Koo, X. T., & Khor, K. C.
Year of Publish : 2023

The paper "Expense Tracking with Tesseract Optical Character Recognition v5: A Mobile Application Development" proposes a mobile application specifically designed to assist users in tracking daily expenditures using automated receipt scanning. The authors utilized the React Native framework for cross-platform development and integrated Tesseract OCR v5, which implements

Long Short-Term Memory (LSTM) networks, to extract text and prices from physical receipts. The study includes a performance evaluation using real-world grocery receipts, reporting a low Character Error Rate (CER) of 2.26% for unit prices, alongside a System Usability Scale (SUS) score of 71.5%, indicating satisfactory usability. Overall, the application addresses the inconvenience of manual data entry and aims to improve financial literacy by providing a convenient, automated method for visualizing and monitoring personal spending habits.

3.2.4 Design and Development of Expense App

Author Name : Girdhar, G., Bhardwaj, A., Kumar, S., & Sharma, M.
Year of Publish : 2024

The paper "Design and Development of Expense App" presents a smartphone application developed using the Flutter framework and Firebase/SQLite to assist individuals in managing their daily financial activities. The authors aim to replace traditional, error-prone manual methods like notebooks and Excel sheets with a digital solution that tracks earnings and outlays over specific time frames. The proposed system features a user-friendly interface that allows users to categorize expenses, set monthly budget limits, and receive automated alerts if those limits are exceeded. Additionally, the application utilizes graphical representations, such as charts and graphs, to provide users with a clear visual summary of their weekly, monthly, and annual spending habits. Overall, the study concludes that the tool significantly aids in improving financial literacy and helps users develop healthier spending habits by minimizing human calculation errors and organizing financial data efficiently.

3.3 LITERATURE SURVEY SUMMARY

S.No	Research	Technique	Features Used	Domain	Disadvantage / Advantage	Future Direction
1.	TrackEz Expense Tracker	Django Web Framew-o rk Python, SQLite	Manual Expens e Entry	Personal Finance	**Advantage:** Saves user time compared to non-digital methods. **Disadvantage:** No automation or OCR capabilities.	Integration of a budgeting and alert system.
2.	The Daily Expense Tracker System	Web technol ogies (HTML, CSS, PHP, SQL)	Manual Expens e Entry, Data Encrypt ion	Personal Finance	**Advantage:**Uses strong AES encryption for security. **Disadvantage:** No intelligent features or AI-based analysis.	Integrating more advanced technologies
3.	FINANCE GPT	AI, Optical Character Recognitio n (OCR), BERT-base d NLP	Automate d expense tracking via OCR, expense	Personal Finance Managem ent	**Advantage:** Automates expense tracking from receipts, eliminating manual entry, and provides	Improving OCR accuracy for varied receipt qualities and exploring multi-modal

		model	categorization, loan eligibility prediction, and personalized insurance recommendations		a comprehensive financial overview with loan and insurance advice. **Disadvantage:** OCR accuracy is limited when processing low-quality images or receipts with unusual formatting	chatbots for a better user experience.
4.	Design and Development of Expense App	Flutter, Node.js, Firebase, SQLite	Daily/monthly expense tracking, budget limit alerts and categorization of expenses	Personal Finance / Mobile App Development	**Advantage:** Reduces manual calculation errors, provides visual spending insights, and sends automatic alerts when budget limits are exceeded. **Disadvantage:** Currently lacks direct integration with banking apps, requiring users to manually input transaction data.	Integration with online payment applications to directly record and analyze debits, and advanced analysis of spending patterns based on denomination size.

Chapter 3

SYSTEM ANALYSIS

3.3 EXISTING SYSTEM

Advanced NLP models, particularly if utilizing deep learning or Transformer architectures, require computational resources that can impact the latency of the real-time categorization API on cost effective cloud instances. The classification model is prone to overfitting, where it becomes too specialized to specific vendor names in the training data (e.g., recognizing "Starbucks" but missing a local coffee shop) and performs poorly on unique or unseen transactions. While the AI can achieve high accuracy in sorting expenses, it often lacks interpretability, making it difficult to understand the specific logic behind why a transaction was assigned a certain category. The model may struggle to generalize to new vendors, slang, or evolving transaction formats outside of its training distribution, limiting its effectiveness for diverse user bases. Furthermore, the system is highly sensitive to the quality of the input text, such as typos, abbreviations, or vague descriptions (e.g., "Transfer" or "Misc"). This can lead to reduced categorization accuracy in real-world usage. Overall, while AI is effective for the AIMMS "Type and Go" system, it is important to consider these potential disadvantages when designing the fallback mechanisms and user correction flows.

3.4 DISADVANTAGES OF EXISTING SYSTEM

- **Reliance on manual data entry** which is tedious, time-consuming, and prone to user abandonment.
- **Invasive requirements for direct bank integration**, forcing users to share sensitive credentials and compromising privacy.
- **Lack of adaptive learning**, as systems often rely on rigid, rule-based categorization that fails to understand context or unique spending habits.
- **Static reporting** that provides only historical charts without offering actionable, real-time insights or short-term goal tracking.

3.5 PROPOSED SYSTEM

In the proposed method, we design and develop an AI-Enhanced Intelligent Money Monitoring System (AIMMS) to automate the tracking and analysis of personal finances. Unlike traditional methods that rely on manual input, this system utilizes Natural Language Processing (NLP) and machine learning algorithms to automatically categorize transaction descriptions entered by the user. The system is built on a robust web framework that serves as a centralized hub for consolidating data from various sources.

The core of the system is the AI classification engine, which processes raw text (e.g., "Starbucks Coffee") and instantly maps it to the correct expense category (e.g., "Food & Drink") with high accuracy. This allows for real-time updates to the user's financial dashboard. Furthermore, the system includes dynamic visualization modules that transform this categorized data into actionable insights, such as spending trends and budget adherence reports. By integrating a Goal Management module, the system not only tracks past behavior but actively assists users in planning for future financial milestones.

3.6 ADVANTAGES OF PROPOSED SYSTEM

- **Advanced AI Integration:** The proposed system uses state-of-the-art NLP algorithms to understand transaction context, significantly outperforming rule-based keywords.
- **Enhanced Financial Discipline:** By visualizing spending habits clearly, the approach encourages users to adhere to their budgets and follow better financial practices.
- **Automated Organization:** The system eliminates the need for manual categorization, saving time and reducing human error in data entry.
- **Actionable Insights:** Instead of just storing data, the system processes it to provide meaningful alerts and progress tracking towards specific saving goals.

3.7 FEASIBILITY STUDY

The feasibility study and prototype development confirm that AIMMS is a viable, scalable, and economically sustainable solution. It offers a robust framework for financial clarity that is accessible to students and young professionals alike. Moving forward, the system establishes a

strong foundation for future enhancements, such as optical character recognition (OCR) for receipts and predictive forecasting, positioning AIMMS as a comprehensive, intelligent ally in the journey toward financial wellness.

3.8 HARDWARE ENVIRONMENT

4 Processor : Pentium Dual Core 2.00 GHz

5 Hard disk : 120 GB

6 RAM : 2GB (minimum)

7 Keyboard : 110 keys enhanced

3.9 SOFTWARE ENVIRONMENT

8 Operating system : Windows7 (with service pack 1), 8, 8.1 ,10 and 11

9 Language : Java

3.10 TECHNOLOGIES USED

10 IDE - Visual Studio

11 Framework - Springboot

12 Deep Learning

13 Machine Learning

3.10.1 Python

Python is a high-level, interpreted programming language that is widely used in various domains such as web development, data science, artificial intelligence, scientific computing, and more. It was first released in 1991 and has since become one of the most popular programming languages in the world. Some key features of Python include:

14 **Easy to Learn:** Python has a simple and easy-to-learn syntax, which makes it an ideal language for beginners.

15 **Interpreted Language:** Python is an interpreted language, which means that the code is

executed line by line, making it easier to test and debug.

16 **Cross-Platform:** Python can be run on various platforms, including Windows, macOS, and Linux.

17 **Large Standard Library:** Python has a large standard library that provides a wide range of built-in modules for various tasks, such as file I/O, regular expressions, networking, and more.

18 **Open Source:** Python is open-source software, which means that the source code is freely available to anyone and can be modified and redistributed.

19 **Object-Oriented:** Python is an object-oriented language, which means that it supports object-oriented programming concepts such as encapsulation, inheritance, and polymorphism.

3.10.2 Deep Learning

In the realm of intelligent financial management, the utilization of deep learning techniques has proven pivotal for advancing automated expense tracking capabilities. Specifically, our project employs the state-of-the-art Natural Language Processing (NLP) model **DistilBERT** (Distilled Bidirectional Encoder Representations from Transformers) to achieve real-time and highly accurate transaction categorization. DistilBERT's distinctive ability to understand the semantic context of short text descriptions makes it an ideal choice for our system, offering the performance of larger transformer models while being lightweight enough for efficient deployment.

Trained on a diverse dataset of financial entries, the deep neural network is finely tuned to discern critical spending categories such as "Food & Drink," "Transport," and "Utilities" from raw user inputs. Additionally, the system integrates **CRNN** (Convolutional Recurrent Neural Network) for its Optical Character Recognition (OCR) module. Unlike traditional computer vision models, CRNN integrates feature extraction and sequence modeling into a single trainable framework, allowing it to accurately recognize text in receipt images of varying dimensions. Through this comprehensive exploration of deep learning methodologies, our system achieves a nuanced understanding of user spending patterns, providing a robust foundation for personalized financial insights.

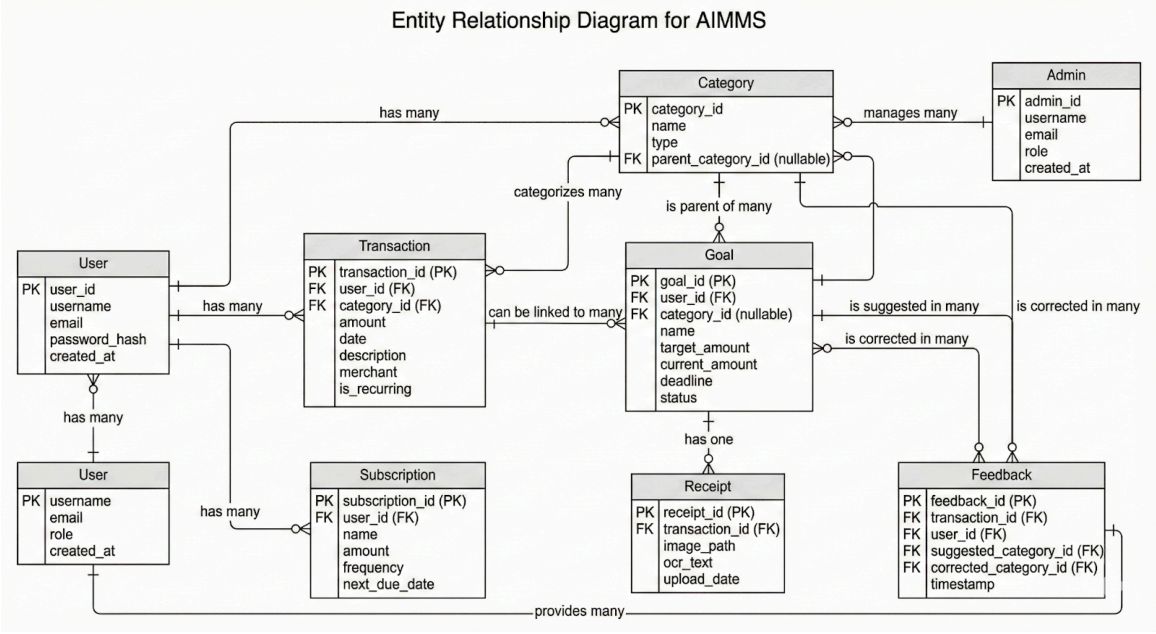
Chapter 4

SYSTEM DESIGN

19.1 ENTITY-RELATIONSHIP DIAGRAM

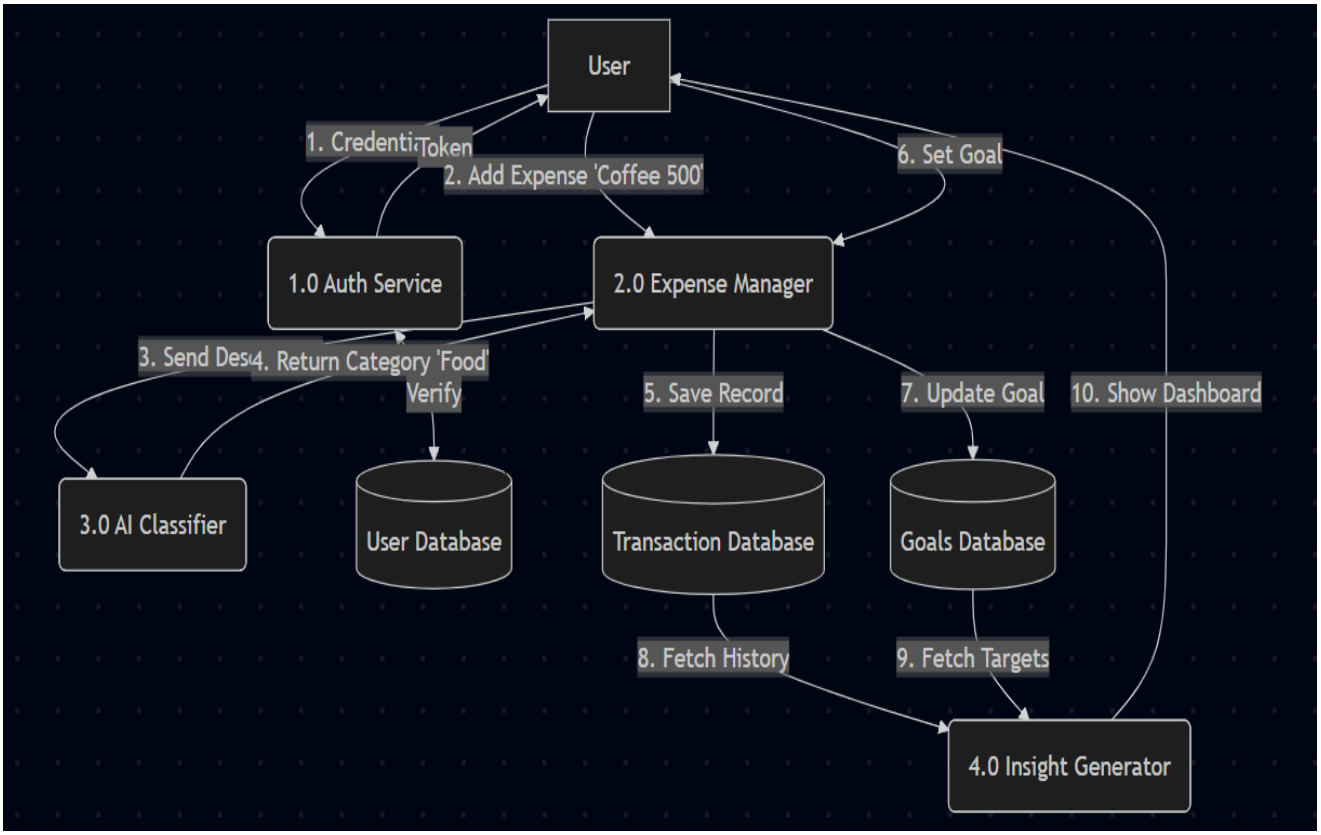
The relationships between database entities can be seen using an entity- relationship diagram (ERD). The entities and relationships depicted in an ERD can have further detail added to them via data object descriptions. In software engineering, conceptual and abstract data descriptions are represented via entity- relationship models (ERMs). Entity-relationship diagrams (ERDs), entity- relationship diagrams (ER), or simply entity diagrams are the terms used to describe the resulting visual representations of data structures that contain relationships between entities. As such, a data flow diagram can serve dual purposes. To demonstrate how data is transformed across the system. To provide an example of the procedures that affect the data flow.

Fig 4.1 Entity Relationship Diagram



19.2 DATA FLOW DIAGRAM (DFD)

The whole system is shown as a single process in a level DFD. Each step in the system's assembly process, including all intermediate steps, are recorded here. The "basic system model" consists of this and 2-level data flow diagrams.They are often elements of a formal methodology such as Structured Systems Analysis and Design Method (SSADM). Superficially, DFDs can resemble flow charts or Unified Modeling Language (UML), but they are not meant to represent details of software logic. DFDs make it easy to depict the business requirements of applications by representing the sequence of process steps and flow of information using a graphical representation or visual representation rather than a textual description.



19.3UML DIAGRAMS

19.3.1 Use Case Diagram

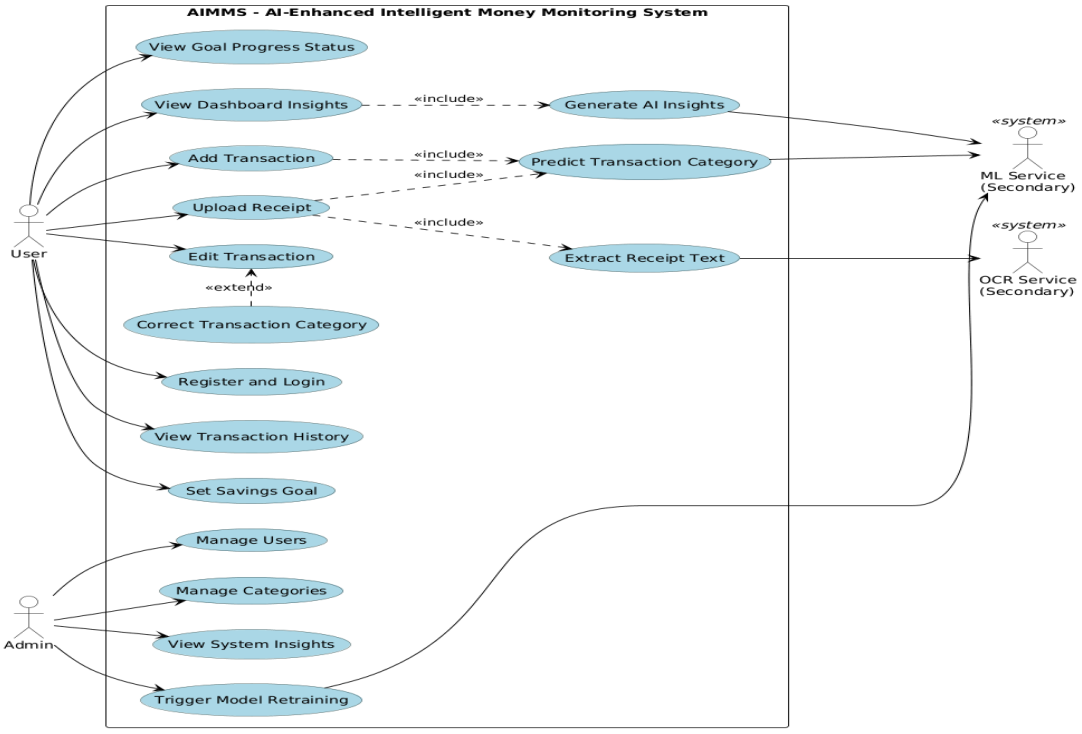
A use case diagram is a type of Unified Modeling Language (UML) diagram that represents the interactions between a system and its actors, and the various use cases that the system supports. It is a visual representation of the functional requirements of the system and the actors that interact with it. Use case diagrams typically include the following elements:

20 Actors: Actors are external entities that interact with the system. They can be human users, other systems, or devices.

21 Use Cases: Use cases are the specific functions or tasks that the system can perform. Each use case represents a specific interaction between an actor and the system.

22 Relationships: Relationships are used to indicate how the actors and use cases are related to each other. The two main relationships in a use case diagram are "uses" and "extends". "Uses" relationship indicates that an actor uses a specific use case, while "extends" relationship indicates that a use case extends or adds functionality to another use case.

23 System Boundary: The system boundary is a box that contains all the actors and use cases in the system. It represents the physical or logical boundary of the system being model



23.1.1 Class Diagram

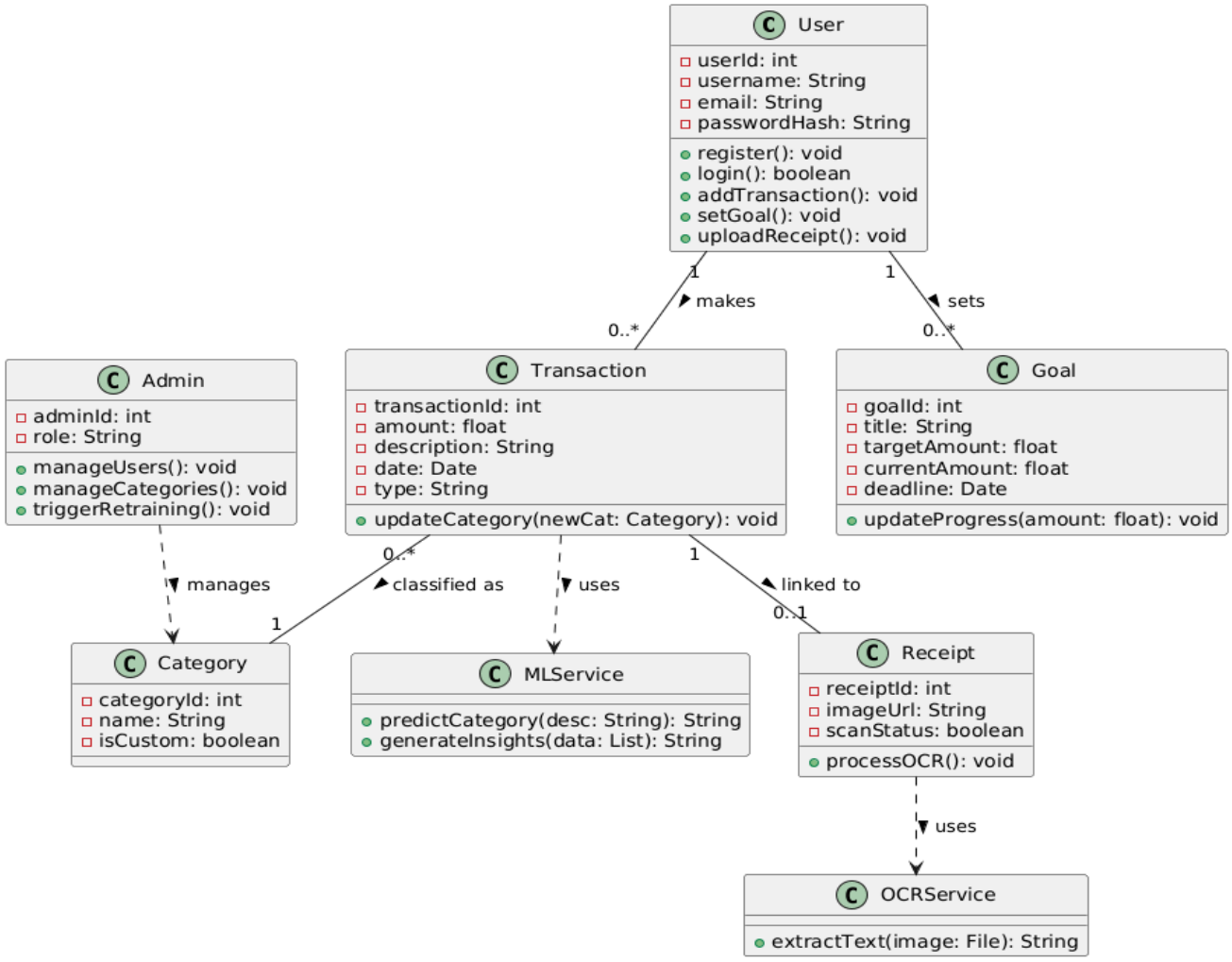
In essence, this is a "context diagram," another name for a contextual diagram. It simply stands for the very highest point, the 0 Level, of the procedure. As a whole, the system is shown as a single process, and the connection to externalities is shown in an abstract manner.

24 A + indicates a publicly accessible characteristic or action.

25 A - a privately accessible one.

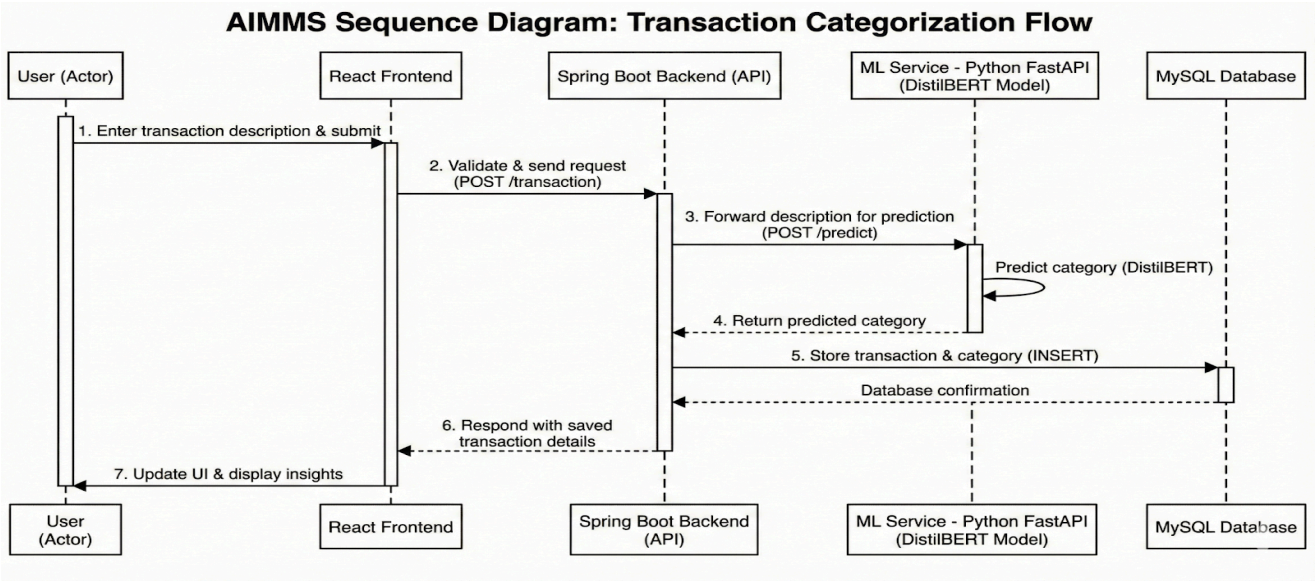
26 A # a protected one.

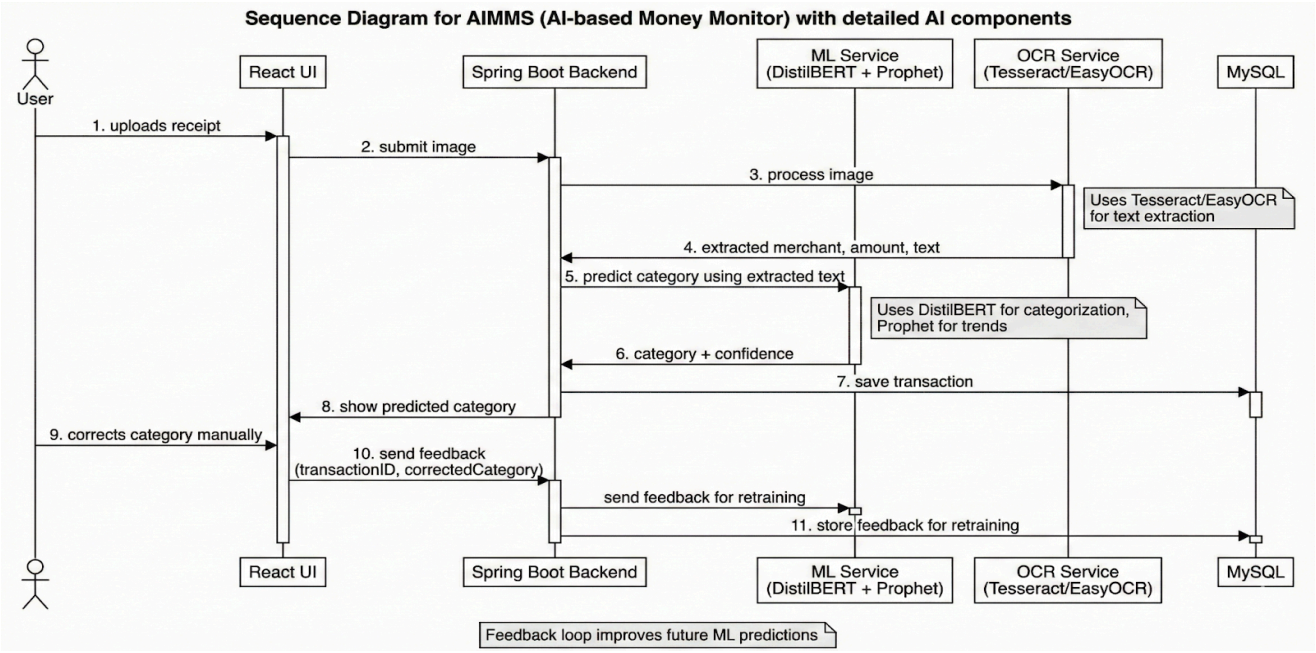
27 A - denotes private attributes or operations.



27.1.1 Sequence Diagram

These are another type of interaction-based diagram used to display the workings of the system. They record the conditions under which objects and processes cooperate. It is a construct of Message Sequence diagrams sometimes called event diagrams, event sceneries and timing diagram.





Chapter 5

SYSTEM ARCHITECTURE

5.1 ARCHITECTURE DIAGRAM

This graphic provides a concise and understandable description of all the entities currently integrated into the system. The diagram shows how the many actions and choices are linked together. You might say that the whole process and how it was carried out is a picture. The figure below shows the functional connections between various entities.

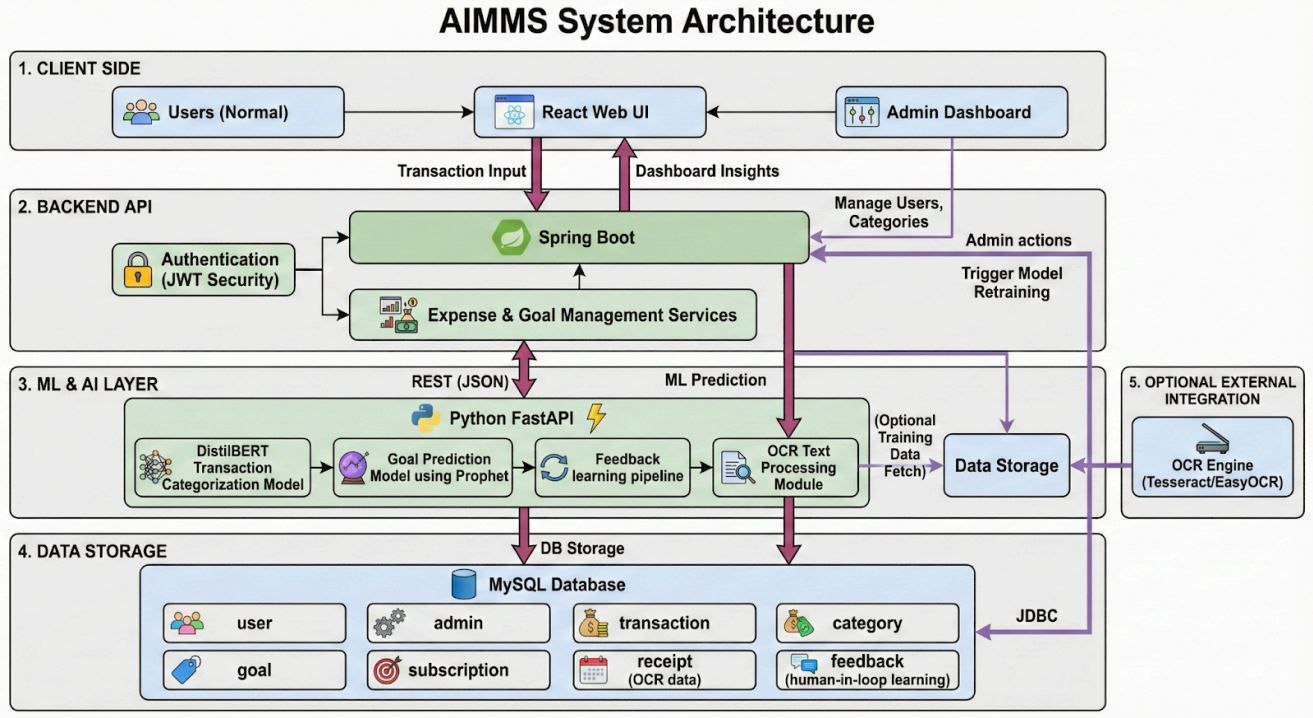


Fig 5.1 Architecture Diagram

The system architecture of the fig 5.1 clearly shows that the input is given as raw text then using the **Multinomial Naive Bayes** model the transactions are analyzed and classified based on the probability. After the classification of the expense, the real-time insight is generated and the budget status is displayed to the user through the dynamic dashboard.

5.2 ALGORITHMS

5.2.1 Optical Character Recognition (OCR)

CRNN (Convolutional Recurrent Neural Network) is a hybrid neural architecture specifically designed for image-based sequence recognition tasks, such as Optical Character Recognition (OCR). Unlike traditional computer vision models that require fixed input dimensions, CRNN can operate on images of varying widths, making it highly effective for recognizing text in natural scenes. The architecture integrates feature extraction and sequence modeling into a single trainable framework.

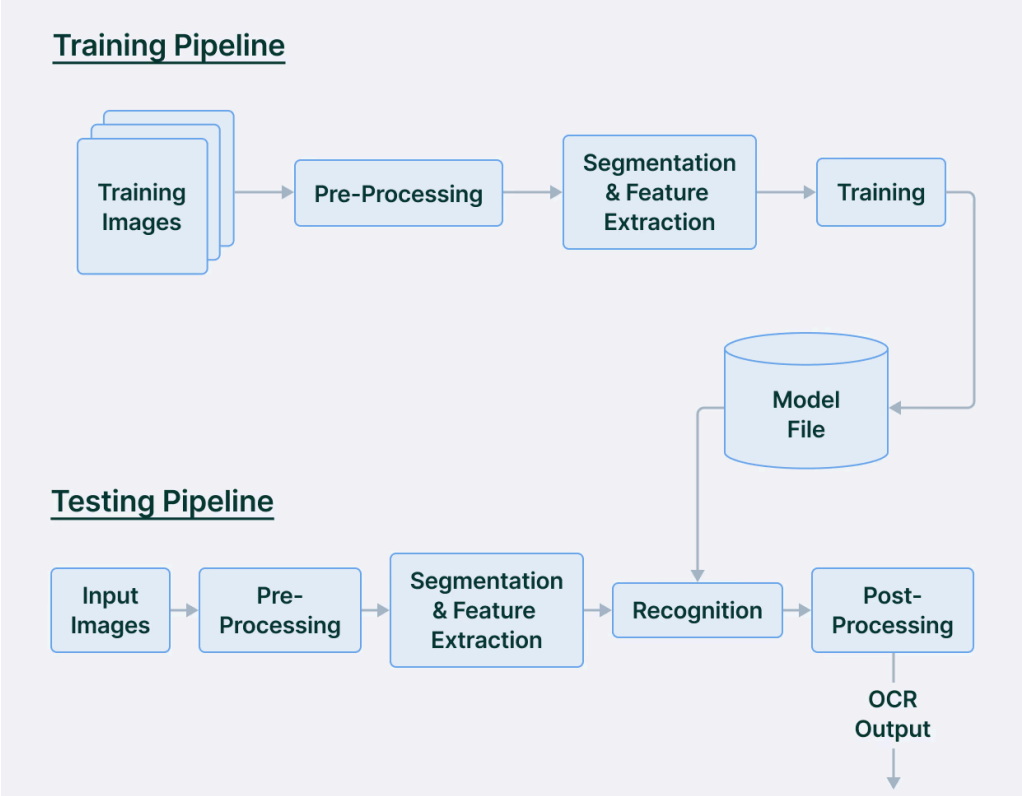


Fig 5.2.1 TEXT RECOGNIZATION MODEL

The CRNN architecture consists of three distinct components:

- **Convolutional Layers:** This is the bottom part of the network, which automatically extracts a feature sequence from each input image. It typically uses a standard CNN architecture, such as VGG or ResNet, to create feature maps that represent the visual appearance of characters.

- **Recurrent Layers:** This component is built on top of the convolutional layers and predicts a label distribution for each frame of the feature sequence. It typically utilizes a Bidirectional LSTM (Long Short-Term Memory) network to capture contextual information from both directions of the text sequence.
- **Transcription Layer:** This is the top component of the architecture, responsible for translating the per-frame predictions into the final label sequence. It merges the redundant characters and removes blanks to form the final recognized word or sentence.

CRNN relies on a specific loss function called Connectionist Temporal Classification (CTC). CTC allows the network to be trained on unsegmented data, meaning the system does not require the exact position of each character in the training images. By using CTC, CRNN eliminates the need for labor-intensive character-level annotations. Another significant advantage of CRNN is its end-to-end trainability, allowing the entire system to be optimized simultaneously for the global objective of sequence recognition, resulting in higher accuracy and robustness compared to systems that train components separately.

Chapter 6

SYSTEM IMPLEMENTATION

6.1 MODULE 1: DATA COLLECTION AND PREPROCESSING

The data collection and preprocessing module for the AI-Enhanced Intelligent Money Monitoring System (AIMMS) plays a foundational role in the overall development of an accurate and robust financial categorization engine. This module involves two primary phases: the aggregation of financial transaction data and the rigorous preparation of this data for Natural Language Processing (NLP) tasks.

Data collection is the first step in building the predictive capabilities of the system. Unlike image-based systems, AIMMS relies on textual data representing financial transactions. The module requires collecting a diverse set of transaction descriptions, merchant names, and amounts. To ensure the model is robust enough to handle real-world ambiguity, the dataset is constructed using a combination of synthetic data generation and real-world transaction examples. This dataset must be extensive and balanced across various spending categories such as "Food & Drink," "Transport," "Utilities," and "Shopping" to prevent the model from becoming biased toward frequently occurring expenses.

The next step is data labeling and tokenization, which is critical for the DistilBERT-based architecture. The raw text descriptions often contain noise, such as random transaction IDs or payment codes (e.g., "UPI-REF-123456"). The preprocessing phase involves cleaning this text by removing irrelevant punctuation, normalizing case, and filtering out non-informative alphanumeric strings. Following this, the text is passed through a specific tokenizer compatible with the DistilBERT model. This process converts human-readable text into numerical input IDs and attention masks that the neural network can interpret. Additionally, the categorical labels (e.g., "Groceries") are encoded into numerical formats using label encoding techniques.

In conclusion, the data collection and preprocessing module for AIMMS is a crucial step in developing an intelligent financial assistant. The module requires careful curation of varied transaction examples and precise linguistic preprocessing to convert raw financial logs into a structured, numerical format suitable for training advanced transformer models.

6.2 MODULE 2: MODEL TRAINING

The model training module of AIMMS forms the core intelligence layer of the entire system, integrating multiple machine learning and deep learning models to deliver a comprehensive financial monitoring experience. This module involves training three major models DistilBERT, Prophet, and LSTM each responsible for a different segment of the system's intelligence. The first model, DistilBERT, is a transformer-based NLP architecture that is fine-tuned to categorize user transactions into specific financial categories such as groceries, utilities, entertainment, rent, health, and subscriptions.

The training data consists of thousands of manually labeled transaction descriptions, including variations in merchant names, UPI messages, bill descriptions, and receipt text extracted through OCR. During training, the DistilBERT model undergoes tokenization, embedding, contextual learning, and backpropagation, enabling it to understand financial language, semantic patterns, and merchant-category relationships. The second major model used in AIMMS is Prophet, a robust time-series forecasting model designed to analyze daily savings and expenditure patterns. Prophet is trained using historical savings data, computed from daily expenses and budgets, enabling it to learn long-term trends, recurring seasonal patterns such as weekend expenditure spikes, and user-specific financial behaviors. Prophet decomposes the time-series data into trend, seasonal, and residual components, providing interpretable and reliable financial forecasts suitable for goal prediction and monthly planning. The third model, LSTM (Long Short-Term Memory Network), is a deep learning architecture specifically trained to learn sequential dependencies and long-range financial patterns. AIMMS trains the LSTM model on extended sequences of user savings and spending data, allowing the model to capture complex patterns such as recurring monthly bills, subscription cycles, festival or holiday spikes, and unexpected fluctuations. The training process involves multiple epochs of forward and backward passes, sequence windowing, memory state updates, and gradient optimization, making the LSTM model highly effective for predicting long-term financial behavior and potential overspending risks. Together, the training of DistilBERT, Prophet, and LSTM represents a comprehensive multi-model training pipeline that equips AIMMS with the ability to classify transactions accurately, forecast user savings trends, and detect or predict spending anomalies. The success of this module depends heavily on the quality and quantity of the training data, the diversity of the financial scenarios included, and proper hyperparameter tuning.

6.3 MODULE 3: PREDICTION OF OUTPUT

The prediction of the output module for the Optical Character Recognition (OCR)–based receipt and bill scanning system represents the final and most crucial stage in the development of an accurate and intelligent document analysis solution. This module focuses on using the trained OCR and NLP models to predict and extract meaningful information from receipt images and scanned documents, such as merchant details, invoice numbers, dates, item descriptions, tax values, and total amounts. The prediction process begins by providing the preprocessed input data, which includes cleaned, deskewed, and enhanced images or PDFs, to the trained OCR engine such as PaddleOCR or EasyOCR. The OCR model analyzes the visual input and detects text regions within the document, generating bounding boxes along with the recognized textual content. These bounding boxes represent the spatial location of the detected text, while the recognized text provides the raw information present in the receipt. Following text detection and recognition, the extracted raw text and positional data are passed to the NLP and information extraction module. Key entities including merchant name, bill number, date and time, item lists, tax details, and total amount are identified and structured into a standardized JSON format. The predicted output can be displayed to the user through a web or mobile interface in real time or stored in a database for further analysis. The effectiveness of this module depends on the quality of image preprocessing, the robustness of the trained OCR and NLP models, and the efficiency of the post-processing strategies employed. In conclusion, the prediction of the output module for the OCR-based receipt and bill scanning system is fundamental to delivering a reliable and intelligent document understanding solution. The output prediction module is the final, crucial step in the OCR-based receipt and bill scanning system. It uses trained OCR (like PaddleOCR, EasyOCR) and NLP models (like DistilBERT, LayoutLM, Donut) to extract and structure key information (merchant, date, total, items, etc.) from preprocessed document images into a standardized JSON format. The process involves OCR detecting text regions and generating bounding boxes, followed by NLP interpreting the raw text and positional data. Post-processing, including confidence thresholding and consistency checks, is applied to ensure accuracy and reliability. The predicted output is displayed, stored, or used to trigger automated actions (expense categorization). The module's success is vital for real-time financial tracking and depends on the quality of preprocessing, model robustness, and post-processing efficiency. Its effective implementation requires expertise in machine learning, computer vision, and NLP.

Chapter 7

SYSTEM TESTING

7.1 BLACK BOX TESTING

During this kind of testing, the user does not have access to or knowledge of the internal structure or specifics of the data item being tested. In this method, test cases are generated or designed only based on the input and output values, and prior knowledge of either the design or the code is not necessary. The testers are just conscious of knowing about what is thought to be able to do, but they do not know how it is able to do it.

Fig 7.1 Black Box Testing

For example, without having any knowledge of the inner workings of the website, we test the web pages by using a browser, then we authorize the input, and last, we test and validate the outputs against the intended result.

7.2 WHITE BOX TESTING

During this kind of testing, the user is aware of the internal structure and details of the data item, or they have access to such information. In this process, test cases are constructed by referring to the code. Programming is extremely knowledgeable of the manner in which the application of knowledge is significant. White Box Testing is so called because, as we all know, in the tester's eyes it appears to be a white box, and on the inside, everyone can see clearly.This is how the testing got its name.

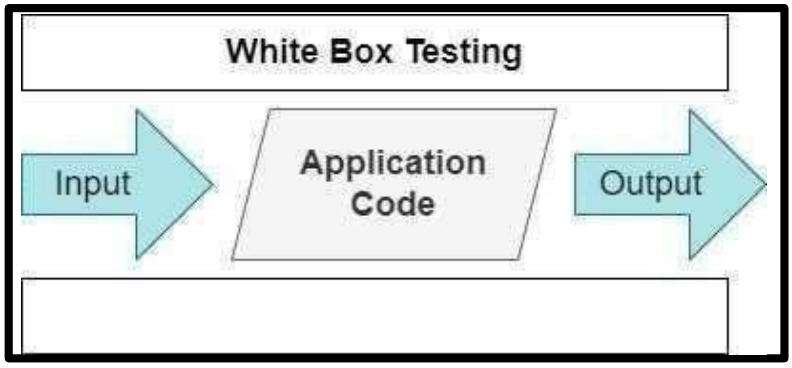


Fig 7.2 White Box Testing

As an instance, a tester and a developer examine the code that is implemented in each field of a website, determine which inputs are acceptable and which are not, and then check the output to ensure it produces the desired result. In addition, the decision is reached by analyzing the code that is really used.

8.1 TEST CASE

TEST REPORT: 01

PRODUCT: AI-ENHANCED INTELLIGENT MONEY MONITORING SYSTEM

USE CASE: ENTER DATA MANUALLY

TEST CASE ID	TEST CASE/ ACTION TO BE PERFORMED	EXPECTED RESULT	ACTUAL RESULT	PASS/FAIL
01	Enter data manually	Uploaded	As Expected	PASS

Table-7.3.1 Test Case For Manual Entry

TEST REPORT: 02

PRODUCT: AI-ENHANCED INTELLIGENT MONEY MONITORING SYSTEM

USE CASE: CATEGORIZATION

TEST CASE ID	TEST CASE/ ACTION TO BE PERFORMED	EXPECTED RESULT	ACTUAL RESULT	PASS/FAIL
01	Extract the category	Extracted Successfully	As Expected	PASS

Table-7.3.2 Test Case For Categorization

Chapter 8

CONCLUSION AND FUTURE ENHANCEMENT

10.1 CONCLUSION

In conclusion, the development of AIMMS (AI-Enhanced Intelligent Money Monitoring System) represents a transformative step toward intelligent, automated, and user-centric financial management. By integrating advanced machine learning and deep learning technologies such as DistilBERT for transaction categorization, Prophet for goal and savings forecasting, and LSTM for deep financial pattern prediction, the system delivers a high degree of accuracy, adaptability, and real-time responsiveness. The system is designed to interpret transaction descriptions, predict future patterns in user spending behavior, and provide intelligent insights that guide users toward better financial decisions. Through extensive model training and evaluation, AIMMS demonstrates its ability to categorize transactions with high precision, forecast long-term savings behavior reliably, and detect potential overspending or financial instability through sequence-based learning. These results highlight AIMMS as not only a reliable analytical tool but also a proactive financial assistant capable of supporting users at every stage of their financial journey.

The system's practical impact lies in its seamless ability to automate routine tasks such as transaction categorization, expense tracking, and budget monitoring, drastically reducing the manual effort typically required in personal finance management. AIMMS further strengthens the user experience by providing goal completion predictions, personalized recommendations, and smart alerts designed to keep users financially aware and in control. Its modular architecture ensures compatibility with real-world systems, making it suitable for integration into personal finance applications, digital banking platforms, and financial advisory tools. The OCR module enhances the system's usability by enabling the extraction of financial data from receipts with minimal user intervention, thereby improving efficiency and data completeness.

Beyond its current capabilities, AIMMS lays the foundation for future enhancements that can further elevate its intelligence and real-world applicability. The inclusion of user feedback loops, retraining mechanisms, and adaptive learning ensures that the system continuously evolves along with

user behavior. As personal finance becomes increasingly digital and data-driven, AIMMS stands as a comprehensive and forward-thinking solution capable of contributing significantly to financial literacy, informed decision-making, and long-term financial well-being. Ultimately, AIMMS demonstrates the potential of AI-powered systems to revolutionize personal finance management, offering accurate predictions, seamless automation, and proactive guidance making it a meaningful contribution to the broader domain of financial technologies.

8.2 FUTURE ENCHANCEMENT

Looking ahead, AIMMS presents a wide range of opportunities for future enhancements that can significantly strengthen its intelligence, adaptability, and real-world impact. One promising direction involves integrating Reinforcement Learning (RL), enabling the system to proactively recommend personalized financial strategies based on long-term rewards rather than static predictions. By learning from user actions and the outcomes of financial decisions, the system can dynamically suggest optimal spending adjustments, budget reallocations, or saving strategies that improve the likelihood of achieving financial goals. This turns AIMMS into not just a monitoring tool, but a smart financial coach capable of guiding users toward healthier financial habits.

Another major enhancement involves integrating banking APIs and UPI transaction services to automate data collection. This eliminates the need for manual entry and OCR uploads, allowing AIMMS to process real-time financial data directly from bank statements and digital payment platforms. Such integration creates a fully automated ecosystem, enhancing accuracy and ensuring that users always have up-to-date financial insights. Additionally, AIMMS can be extended with fraud detection and anomaly identification models, enabling early detection of unusual transactions or potential security risks using advanced AI techniques.

Chapter 9

APPENDIX 1 – SAMPLE CODING

analyze_data.py

```
import csv
from collections import Counter
def analyze_dataset(file_path):
    categories = []
    try:
        with open(file_path, 'r', ='utf-8') as f:
            reader = csv.DictReader(f)
            for row in reader:
                if 'category' in row:
    categories.append(row['category'])
        counts = Counter(categories)
        print(f"Total records:
    {len(categories)}")
        print("Category distribution:")
        for cat, count in
    counts.most_common():
            print(f"{cat}: {count}")
    except FileNotFoundError:
        print(f"File not found: {file_path}")
```

```
    if __name__ == "__main__":
        analyze_dataset('sample_dataset.csv')
```

predict.py:

```
import joblib
import torch
from transformers import DistilBertTokenizer, DistilBertForSequenceClassification
import numpy as np

MODEL_DIR = './'

def load_models():
    print("Loading models...")
    # Load Logistic Regression
    lr_model = joblib.load(f'{MODEL_DIR}logistic_regression_model.pkl')

    # Load DistilBERT
    bert_path = f'{MODEL_DIR}distilbert_transaction_model'
    tokenizer = DistilBertTokenizer.from_pretrained(bert_path)
    bert_model = DistilBertForSequenceClassification.from_pretrained(bert_path)
    le = joblib.load(f'{MODEL_DIR}distilbert_label_encoder.pkl')

    return lr_model, tokenizer, bert_model, le

def predict_lr(model, text):
    return model.predict([text])[0]

def predict_bert(tokenizer, model, le, text):
    inputs = tokenizer(text, return_tensors="pt", truncation=True, padding=True, max_length=64)
    with torch.no_grad():
```

```
        logits = model(**inputs).logits
    predicted_class_id = logits.argmax().item()
    return le.inverse_transform([predicted_class_id])[0]

if __name__ == "__main__":
    lr_model, tokenizer, bert_model, le = load_models()

    test_cases = [
        "Starbucks Coffee",
        "Uber Ride to Office",
        "Netflix Subscription",
        "Rent Payment for Oct",
        "Electricity Bill TNEB",
        "Salary Credited", # Unknown/Misc
        "Movie at PVR",
        "Medicine from Apollo",
        "Grocery from BigBasket"
    ]

    print("\n--- Predictions ---")
    print(f"{'Transaction':<25} | {'Logistic Regression':<20} | {'DistilBERT':<20}")
    print("-" * 70)

    for text in test_cases:
        pred_lr = predict_lr(lr_model, text)
        pred_bert = predict_bert(tokenizer, bert_model, le, text)
        print(f"{text:<25} | {pred_lr:<20} | {pred_bert:<20}")
```

Dashboard.jsx:

```
import React, { useEffect, useState } from 'react'
import { getDashboardStats } from '../services/api'

export default function Dashboard() {
  const [stats, setStats] = useState({ totalUsers: 0, recentActivity: [] });
  const [loading, setLoading] = useState(true);

  useEffect(() => {
    const fetchStats = async () => {
      try {
        const data = await getDashboardStats();
        setStats(data);
      } catch (error) {
        console.error("Error loading dashboard stats:", error);
      } finally {
        setLoading(false);
      }
    };
    fetchStats();
  }, []);

  if (loading) return <div className="p-6">Loading dashboard...</div>;

  return (
    <div className="grid grid-cols-3 gap-6">
      <div className="col-span-2 bg-white p-6 rounded shadow">
        <h2 className="text-xl font-semibold">Overview</h2>
        <div className="mt-4 grid grid-cols-2 gap-4">
          <div className="bg-blue-50 p-4 rounded">
            <h3 className="text-lg font-medium text-blue-700">Total Users</h3>
```

```
          <p className="text-3xl font-bold text-blue-900">{stats.totalUsers}</p>
        </div>
        <div className="bg-green-50 p-4 rounded">
          <h3 className="text-lg font-medium text-green-700">System Status</h3>
          <p className="text-lg font-bold text-green-900">Active</p>
        </div>
      </div>
    </div>
    <div className="bg-white p-6 rounded shadow">
      <h3 className="font-semibold">Notifications</h3>
      <ul className="mt-2 text-sm text-slate-600">
        <li>No new notifications.</li>
      </ul>
    </div>
    <div className="col-span-3 bg-white p-6 rounded shadow">
      <h3 className="font-semibold">Recent activity</h3>
      <p className="mt-2 text-sm text-slate-600">
        {stats.recentActivity.length > 0 ? (
          <ul>
            {stats.recentActivity.map((act, i) => <li key={i}>{act}</li>)}
          </ul>
        ) : (
          "No recent activity to show."
        )}
      </p>
    </div>
  </div>
 )
}
```

Transactions.jsx:

```
import React, { useEffect, useState } from 'react'
import { getTransactions, createTransaction, deleteTransaction, getUsers } from
'../services/api'


export default function Transactions() {
  const [transactions, setTransactions] = useState([]);
  const [loading, setLoading] = useState(false);
  const [userId, setUserId] = useState(1); // Default to user 1 for demo
  const [formData, setFormData] = useState({
    amount: '',
    description: '',
    merchant: '',
    txnDate: new Date().toISOString().split('T')[0]
  });

  // Check if current user is admin (read-only access for admins)
  const isAdmin = localStorage.getItem('userType') === 'admin';
  const userRole = localStorage.getItem('role');

  useEffect(() => {
    console.log("Transactions component mounted");
    // Fetch users to get a valid ID
    const fetchUsers = async () => {
      try {
        const users = await getUsers();
        console.log("Fetched users:", users);
        if (users.length > 0) {
          console.log("Setting userId to:", users[0].userId);
          setUserId(users[0].userId);
```

```
      }
    } catch (err) {
      console.error("Failed to fetch users", err);
    }
  };

  fetchUsers();
}, []);

useEffect(() => {
  console.log("userId changed to:", userId);
  if (userId) {
    loadTransactions();
  }
}, [userId]);

const loadTransactions = async () => {
  console.log("Loading transactions for userId:", userId);
  setLoading(true);
  try {
    const data = await getTransactions(userId);
    console.log("Loaded transactions:", data);
    setTransactions(data);
  } catch (err) {
    console.error("Failed to load transactions", err);
  } finally {
    setLoading(false);
  }
};

const handleInputChange = (e) => {
```

```
  const { name, value } = e.target;
  setFormData(prev => ({ ...prev, [name]: value }));
};

const handleSubmit = async (e) => {
  e.preventDefault();
  console.log("Submitting transaction:", formData, "for userId:", userId);
  try {
    const res = await createTransaction(userId, formData);
    console.log("Transaction created:", res);
    setFormData({
      amount: '',
      description: '',
      merchant: '',
      txnDate: new Date().toISOString().split('T')[0]
    });
    loadTransactions();
  } catch (err) {
    alert("Failed to create transaction");
    console.error("Create transaction error:", err);
  }
};

const handleDelete = async (id) => {
  if (!window.confirm("Delete transaction?")) return;
  try {
    await deleteTransaction(id);
    setTransactions(transactions.filter(t => t.transactionId !== id));
  } catch (err) {
    console.error(err);
  }
```

```
  };

  return (
    <div className="p-6">
      <div className="flex items-center justify-between mb-6">
        <h2 className="text-2xl font-bold">Transactions</h2>
        {isAdmin && (
          <span className="bg-yellow-100 text-yellow-800 text-sm font-medium px-3 py-1
rounded-full">
            Admin View (Read-Only)
          </span>
        )}
      </div>

      {/* Simple User Selector for Demo */}
      {isAdmin && (
        <div className="mb-6">
          <label className="mr-2 font-medium">View User Transactions (User ID):</label>
          <input
            type="number"
            value={userId}
            onChange={(e) => setUserId(e.target.value)}
            className="border rounded p-1 w-20"
          />
        </div>
      )}

      <div className={`grid grid-cols-1 ${!isAdmin ? 'md:grid-cols-3' : ''} gap-6`}>
        {/* Form - Only show for non-admin users */}
        {!isAdmin && (
          <div className="bg-white p-6 rounded shadow h-fit">
            <h3 className="text-lg font-semibold mb-4">Add Transaction</h3>
```

```
          <form onSubmit={handleSubmit} className="space-y-4">
            <div>
              <label className="block text-sm font-medium text-gray-700">Amount</label>
              <input type="number" name="amount" value={formData.amount}
onChange={handleInputChange} required className="mt-1 block w-full border rounded p-2"
/>
            </div>
            <div>
              <label className="block text-sm font-medium text-gray-700">Description</label>
              <input type="text" name="description" value={formData.description}
onChange={handleInputChange} required className="mt-1 block w-full border rounded p-2"
/>
            </div>
            <div>
              <label className="block text-sm font-medium text-gray-700">Merchant</label>
              <input type="text" name="merchant" value={formData.merchant}
onChange={handleInputChange} required className="mt-1 block w-full border rounded p-2"
/>
            </div>
            <div>
              <label className="block text-sm font-medium text-gray-700">Date</label>
              <input type="date" name="txnDate" value={formData.txnDate}
onChange={handleInputChange} required className="mt-1 block w-full border rounded p-2"
/>
            </div>

            <button type="submit" className="w-full bg-blue-600 text-white py-2 rounded
hover:bg-blue-700">Add Transaction</button>
          </form>
        </div>
      )}

      {/* List */}
      <div className={`${!isAdmin ? 'md:col-span-2' : ''} bg-white rounded shadow
overflow-hidden`}>
```

```
<div className="overflow-x-auto">
  <table className="min-w-full divide-y divide-gray-200">
    <thead className="bg-gray-50">
      <tr>
        <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase">Date</th>
        <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase">Merchant</th>
        <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase">Amount</th>
        <th className="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase">Category</th>
        {!isAdmin && <th className="px-6 py-3 text-right text-xs font-medium text-gray-500 uppercase">Actions</th>}
      </tr>
    </thead>
    <tbody className="divide-y divide-gray-200">
      {loading ? (
        <tr><td colSpan={isAdmin ? "4" : "5"} className="p-4 text-center">Loading...</td></tr>
      ) : transactions.length === 0 ? (
        <tr><td colSpan={isAdmin ? "4" : "5"} className="p-4 text-center text-gray-500">No transactions found.</td></tr>
      ) : (
        transactions.map(t => (
          <tr key={t.transactionId}>
            <td className="px-6 py-4 whitespace-nowrap text-sm text-gray-500">{t.txnDate}</td>
            <td className="px-6 py-4 whitespace-nowrap text-sm text-gray-900">{t.merchant}</td>
            <td className="px-6 py-4 whitespace-nowrap text-sm font-bold text-gray-900">${t.amount}</td>
            <td className="px-6 py-4 whitespace-nowrap text-sm text-gray-500">
              <span className="px-2 inline-flex text-xs leading-5 font-semibold rounded-full bg-green-100 text-green-800">
```

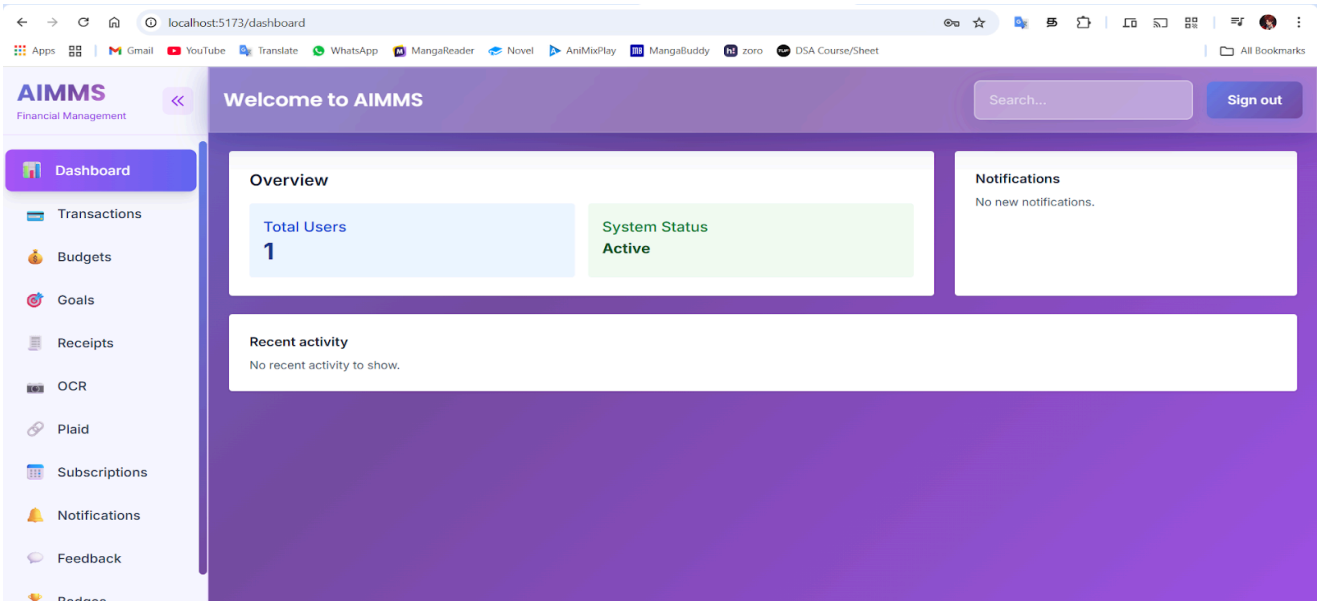
```
                    {t.predictedCategory || t.category?.name || 'Uncategorized'}
                  </span>
                </td>
                {!isAdmin && (
                  <td className="px-6 py-4 whitespace-nowrap text-right text-sm
font-medium">
                    <button onClick={() => handleDelete(t.transactionId)}
className="text-red-600 hover:text-red-900">Delete</button>
                  </td>
                )}
              </tr>
            ))
          )}
        </tbody>
      </table>
    </div>
  </div>
 </div>
 )
}
```

Chapter 10

APPENDIX 2 – SAMPLE OUTPUT

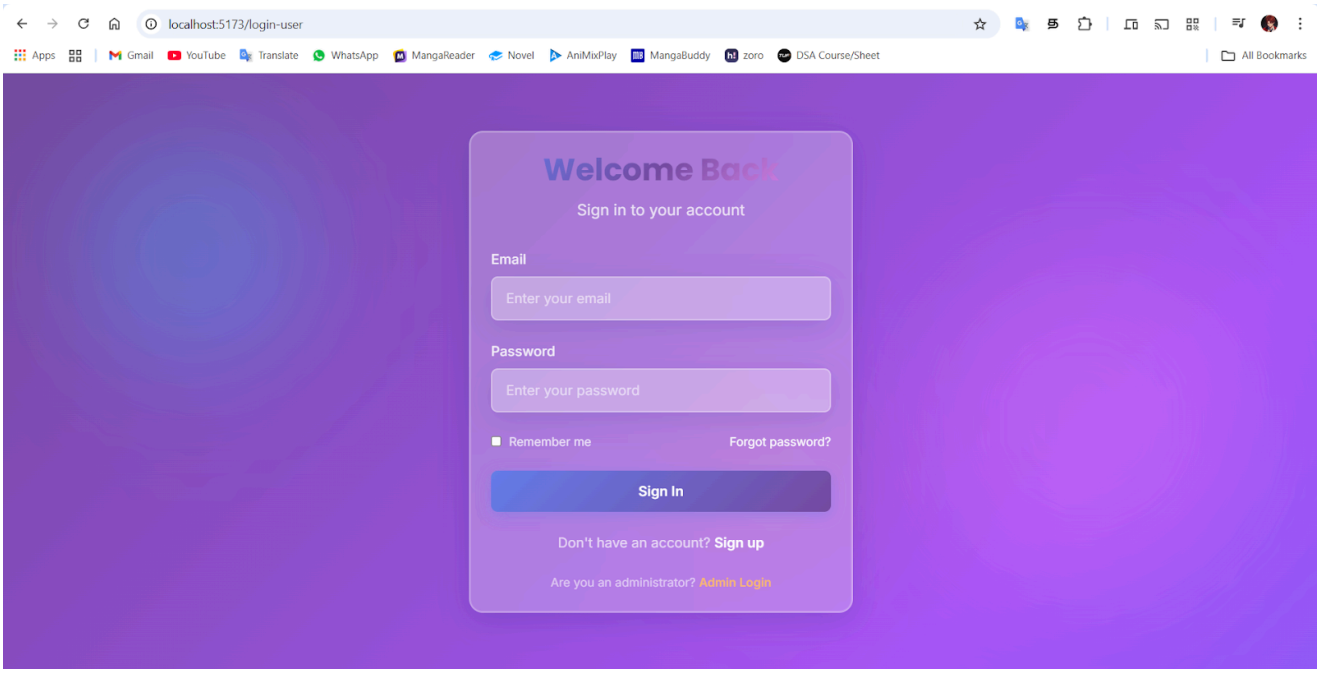
10.1 Home Page

The project output screenshots are shown as follows:



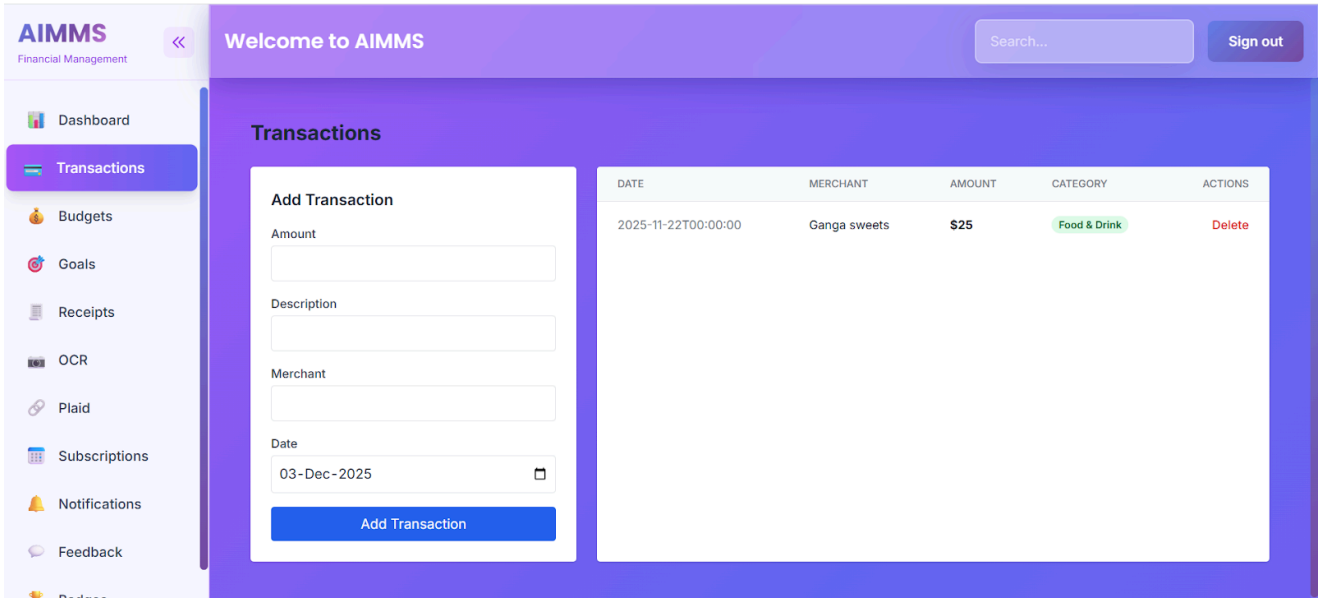
This is the dashboard or home page of a financial management application called **AiMMS**. The design features a purple background with a sidebar on the left for navigation, which includes links to the **Dashboard**, **Transactions**, **Budgets**, **Goals**, and other features. The main content area displays an **Overview** section showing **Total Users** and a **System Status**. There are also panels for **Notifications** and **Recent activity**, both of which currently show no new items. A search bar and a "Sign out" button are located in the top right corner.

10.2 Login Page



This is the login page for the application, featuring a glassmorphism design on a solid purple and blue gradient background. A translucent, rounded-rectangle panel is centered on the screen and contains the login elements. The text "Welcome Back" is displayed at the top, followed by "Sign in to your account." The page includes two input fields: one for **Email** and one for **Password**. Below the input fields, there are options for "Remember me" and a "Forgot password?" link. The main call-to-action is a large, blue-to-purple gradient **"Sign In"** button. Finally, the page provides links for new users to "Sign up" and a separate link for an "Admin Login."

10.3 Transaction page



In the figure 8.3 , the **Transactions** section of the AiMMS financial management application, accessible via the highlighted link in the left sidebar. The page is divided into two main areas: a form on the left titled "**Add Transaction**" with input fields for Amount, Description, Merchant, and Date; and a transaction list on the right. The transaction list shows columns for Date, Merchant, Amount, Category, and Actions, currently displaying one example transaction for **Ganga sweets**..

Chapter 11

REFERENCES

[1] D. S, S. R. S. Suvarna Smita, R. Sheethal, T. V. Dixit, S. Sri Balaji, and T. D. Sivadharishana, "FINANCEGPT: PRECISION FINANCIAL FORECASTING AND BUDGETING FOR SMARTER INVESTMENT STRATEGIES," in *2025 6th International Conference on Control, Communication and Computing (ICCC)*, 2025.

[2] S. Shelke, S. Shaikh, M. Shingre, and S. Lebisha, "Smart BAT - Smart Budget Analyzer and Tracker," in *2023 International Conference on Advanced Computing Technologies and Applications (ICACTA)*, 2023.

[3] X. Koo and K. Khor, "Expense Tracking with Tesseract Optical Character Recognition v5: A Mobile Application Development," in *2023 IEEE Symposium on Industrial Electronics & Applications (ISIEA)*, 2023.

[4] S. Singla et al., "Unveiling Financial Insights: The Daily Expense Tracker System Approach," in *2024 International Conference on Emerging Innovations and Advanced Computing (INNOCOMP)*, 2024.

[5] P. Bhatele et al., "TrackEZ Expense Tracker," in *2023 4th International Conference for Emerging Technology (INCET)*, pp. 1-5, 2023.

- **Github :** https://github.com/Janarthanan2/AIMMS.git
- **Youtube link of the project:** https://youtu.be/ZVaoVND3Q3I