# EXPENSE MANAGEMENT
# A MINI-PROJECT REPORT

*Submitted by:*

**PRATIBA.S    241901080**

*in partial fulfillment of the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING(Cyber Security)**



**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**An Autonomous Institute**

**CHENNAI**

**NOVEMBER 2025**

# BONAFIDE CERTIFICATE

Certified that this project **"EXPENSE MANAGEMENT"** is the bona fide work of **"PRATIBA.S",** who carried out the project work under my supervision.

**SIGNATURE**

**MS.M. FOWZIA SIHANA**

**ASSISTANT PROFESSOR (SS)**

**Department of Computer Science and Engineering (Cyber Security)**

**Rajalakshmi Engineering College**

**Chennai**

This mini project report is submitted for the viva voice examination to be held on _____ .

**INTERNAL EXAMINER**                                    **EXTERNAL EXAMINER**

# ABSTRACT

        In today's competitive business world, efficient project expense management is vital for organisational success. Companies often struggle to track expenses, manage vendors, monitor budgets, and generate reports. This project aims to develop a comprehensive database system that centralises expense tracking across multiple projects, vendors, and cost categories. The system enables organisations to control spending, prevent budget overruns, and produce insightful financial reports, ultimately optimising financial operations and improving decision-making.

# ACKNOWLEDGEMENT

# TABLE OF CONTENTS

# CHAPTER 1:

# INTRODUCTION

## 1.1   INTRODUCTION:

The Project Expense Management System automates expense tracking,   budgeting, and reporting for projects. It ensures accuracy, security, and real-time financial insights through role-based access and secure data management.

## 1.2   SCOPE OF THE WORK:

The Project Expense Management System encompasses the following functionalities: User Management: Register and manage multiple users with different Business Management: Store and organize multiple business entities Account Management: Maintain financial accounts with balance tracking Vendor Management: Store vendor information and contact details Customer Management: Track customer information and contact details Category Management: Define and manage expense categories with type classification Transaction Management: Record all financial transactions with date, amount, and descriptions Invoice Management: Generate and track customer invoices with payment status Budget Management: Set monthly or yearly budgets for expense categories with limit tracking Reporting: Generate expense reports, budget vs. actual analysis, and vendor payment summaries.

## 1.3   PROBLEM STATEMENT:

Organisations struggle with manual expense management due to scattered tracking, budget overruns, poor vendor management, weak audit trails, time-consuming reporting, and a lack of authorisation control.

## 1.4    AIM AND OBJECTIVES OF THE PROJECT:

The project aims to build a user-friendly system for efficient expense tracking and analysis. Its objectives include centralised data storage, secure role-based access, real-time budget monitoring, vendor management, automated reporting, audit trails, and multi-project tracking.

### KEY OBJECTIVES:

1. Provide a centralised database for storing all expense-related information.
2. Implement role-based access control for enhanced security.
3. Enable real-time budget monitoring and alert generation for budget overruns.
4. Facilitate efficient vendor and customer payment management.
5. Generate comprehensive expense reports and financial analytics.
6. Maintain audit trails for compliance and verification.
7. Automate expense categorisation and allocation.
8. Provide multi-project expense tracking capabilities.

# CHAPTER 2

# SYSTEM SPECIFICATIONS

## 2.1 SOFTWARE SPECIFICATIONS:

| | | |
|---|---|---|
| Operating System | **:** | WINDOWS 11 |
| Front – End | **:** | Python |
| Back - End | **:** | SqlLite |
| Language | **:** | python,SQL |

## 2.2    HARDWARE SPECIFICATIONS

| | | |
|---|---|---|
| Processor | **:** | Intel i5 |
| Memory Size | **:** | 8GB (Minimum) |
| HDD | **:** | 1 TB (Minimum) |

# CHAPTER 3

# MODULE DESCRIPTION

## Module 1: User Authentication & Management

This module handles user registration, login, and role-based access control. Users can have roles such as Admin (full access), Manager (create and view expenses), Accountant (view and process payments), and Viewer (read-only access).

*Key Features*:

- User registration with email verification
- Secure password authentication
- Role-based access control (RBAC)
- Session management
- Password recovery and reset

## Module 2: Business & Account Management:

This module allows organizations to manage multiple businesses and their associated financial accounts.

*Key Features:*

- Create and manage business profiles
- Track multiple bank/financial accounts per business
- Real-time account balance updates
- Account statement generation

## Module 3: Expense & Transaction Management:

This module is the core of the system, enabling users to record and categorize expenses.

*Key Features:*

- Record transactions with category assignment
- Support for multiple vendors and customers
- Transaction history and audit trails
- Transaction search and filtering
- Receipt attachment capability

## Module 4: Budget Management:

Allows setting and monitoring budgets for different expense categories.

### *Key Features:*

- Set budget limits per category and time period
- Real-time budget vs. actual tracking
- Budget alert notifications for overruns
- Budget variance analysis

## Module 5: Reporting & Analytics:

Generates comprehensive financial reports and analytics.

### Key Features:

- Expense reports by category, vendor, or time period
- Budget variance reports
- Cash flow analysis
- Vendor payment summaries
- Export to CSV/PDF formats

## Module 6: Vendor & Customer Management:

Manages vendor and customer information with payment tracking.
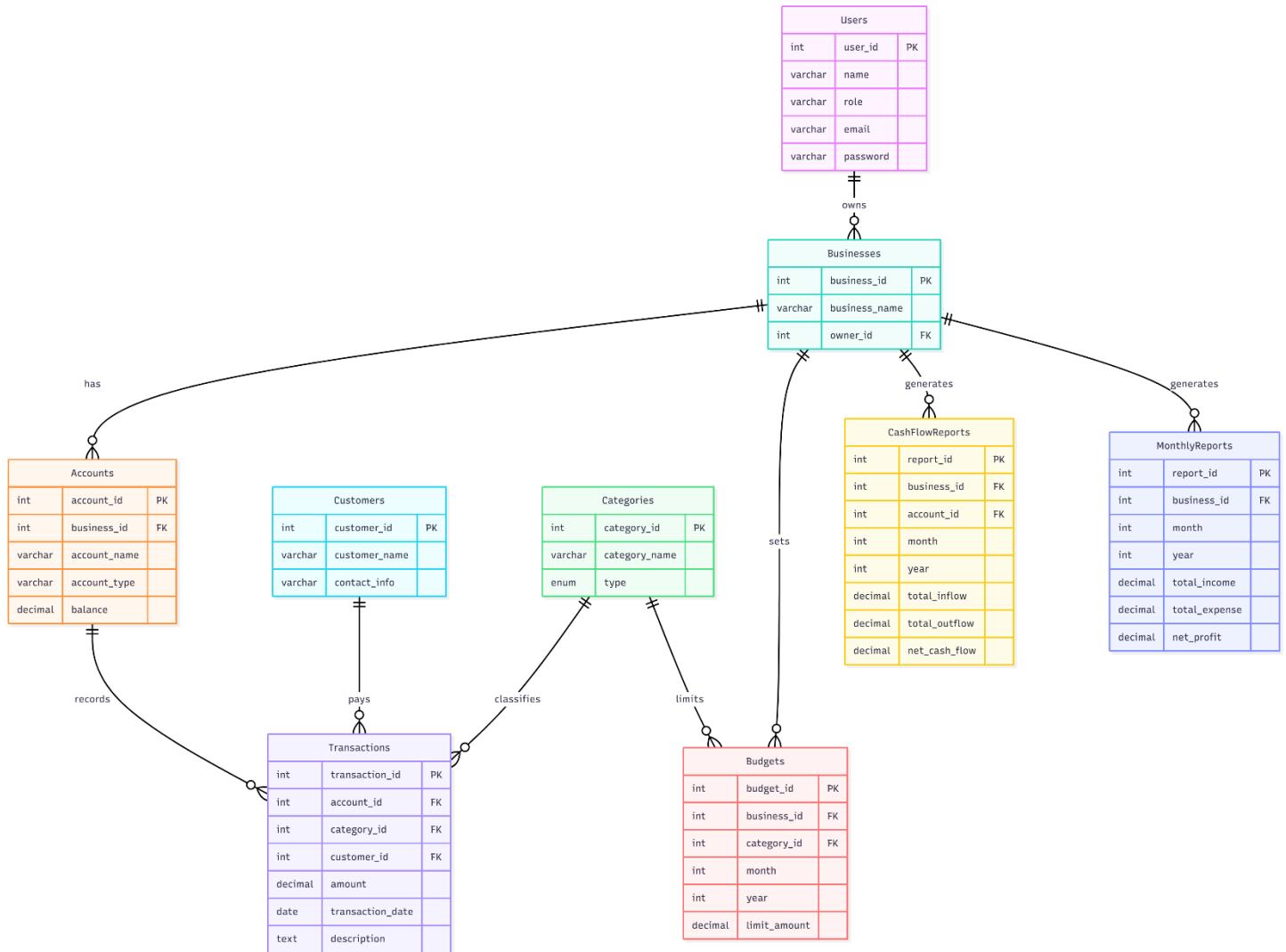
### *Key Features:*

- Vendor profile management
- Customer invoice tracking
- Payment status monitoring
- Customer communication history

# CHAPTER 4

# DATABASE DESIGN

## 4.1 ENTITY RELATIONSHIP DIAGRAM

# 4.2 TABLE STRUCTURE

## Users Table:

| Users | | |
|---|---|---|
| int | user_id | PK |
| varchar | name | |
| varchar | role | |
| varchar | email | |
| varchar | password | |

## Businesses Table:

| Businesses | | |
|---|---|---|
| int | business_id | PK |
| varchar | business_name | |
| int | owner_id | FK |

## Accounts Table:

| Accounts | | |
|---|---|---|
| int | account_id | PK |
| int | business_id | FK |
| varchar | account_name | |
| varchar | account_type | |
| decimal | balance | |

## Customers Table:

| Customers | | |
|---|---|---|
| int | customer_id | PK |
| varchar | customer_name | |
| varchar | contact_info | |

## Categories Table:

| Categories | | |
|---|---|---|
| int | category_id | PK |
| varchar | category_name | |
| enum | type | |

## Transactions Table:

| Transactions | | |
|---|---|---|
| int | transaction_id | PK |
| int | account_id | FK |
| int | category_id | FK |
| int | customer_id | FK |
| decimal | amount | |
| date | transaction_date | |
| text | description | |

## Budgets Table:

| Budgets | | |
|---|---|---|
| int | budget_id | PK |
| int | business_id | FK |
| int | category_id | FK |
| int | month | |
| int | year | |
| decimal | limit_amount | |

## CashFlowReports Table:

| CashFlowReports | | |
|---|---|---|
| int | report_id | PK |
| int | business_id | FK |
| int | account_id | FK |
| int | month | |
| int | year | |
| decimal | total_inflow | |
| decimal | total_outflow | |
| decimal | net_cash_flow | |

## MonthlyReports Table:

| MonthlyReports | | |
|---|---|---|
| int | report_id | PK |
| int | business_id | FK |
| int | month | |
| int | year | |
| decimal | total_income | |
| decimal | total_expense | |
| decimal | net_profit | |

# CHAPTER 5

# SQL IMPLEMENTATION

## Creation of Table:

```
expense_management > apps > users > 🐍 models.py > 🔱 Users
        You, 4 weeks ago | 1 author (You)
   1    from django.db import models
   2
   3
   4
        You, 4 weeks ago | 1 author (You)
   5    class Users(models.Model):          You, 4 weeks ago • Initial commi
   6        user_id = models.IntegerField(primary_key = True)
   7        name = models.CharField(max_length = 100)
   8        role = models.CharField(max_length = 50)
   9        email = models.CharField(max_length = 100)
  10        password = models.CharField(max_length = 100)
  11
  12        def __str__(self):
  13            return f"{self.name} ({self.role})"
  14
```

## Registrations Of The Table:

```
expense_management > apps > users > 🐍 admin.py
        You, 1 second ago | 1 author (You)
   1    from django.contrib import admin
   2    from .models import Users
   3
   4    admin.site.register(Users)
   5
   6
   7
   8    |
```

## Insertion Of Values Into Table:



| | custo... | customer... | email | user_id |
|---|---|---|---|---|
| | Filte | Filte | Filter.. | Filte |
| 1 | 1 | kumar | kumar@gmail.com | NULL |
| 2 | 4 | rahul | rahul@gmail.com | NULL |
| 3 | 5 | sundar | sundar@gmail.com | NULL |
| 4 | 6 | janr | dd@gmail.com | NULL |
| + | 5 | | | |

## Logic Behind The Code:

```python
from django.shortcuts import render, redirect, get_object_or_404
from apps.users.decorators import customer_required
from .models import Customer
from .forms import CustomerForm

@customer_required
def customer_list(request):
    customers = Customer.objects.all()
    return render(request, 'customers/customer_list.html', {'customers': customers})

def customer_create(request):
    if request.method == 'POST':
        form = CustomerForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('customer_list')
    else:
        form = CustomerForm()
    return render(request, 'customers/customer_form.html', {'form': form})

def customer_update(request, pk):
    customer = get_object_or_404(Customer, pk=pk)
    if request.method == 'POST':
        form = CustomerForm(request.POST, instance=customer)
        if form.is_valid():
            form.save()
            return redirect('customer_list')
    else:
        form = CustomerForm(instance=customer)
    return render(request, 'customers/customer_form.html', {'form': form})

def customer_delete(request, pk):
    customer = get_object_or_404(Customer, pk=pk)
    if request.method == 'POST':
        customer.delete()
        return redirect('customer_list')        You, 4 weeks ago • Initial commit …
    return render(request, 'customers/customer_confirm_delete.html', {'customer': customer})
```

# Frontend For The Code:

## Customers_list HTML:

```html
{% extends 'customers/base.html' %}          You, 3 weeks ago • Updated quiz feature logic
{% load static %}

{% block title %}Customers{% endblock %}

{% block content %}
  <h2>Customers</h2>
  <p><a class="btn" href="{% url 'customer_create' %}">Add customer</a></p>
  <table class="table">
    <thead>
      <tr>
        <th>ID</th>
        <th>Name</th>
        <th>Email</th>
        <th>Actions</th>
      </tr>
    </thead>
    <tbody>
      {% for c in customers %}
      <tr>
        <td>{{ c.customer_id }}</td>
        <td>{{ c.customer_name }}</td>
        <td>{{ c.email }}</td>
        <td>
          <a href="{% url 'customer_update' c.customer_id %}">Edit</a> |
          <a href="{% url 'customer_delete' c.customer_id %}">Delete</a>
        </td>
      </tr>
      {% empty %}
      <tr><td colspan="4">No customers yet.</td></tr>
      {% endfor %}
    </tbody>
  </table>
{% endblock %}
```

## Customers_Delete HTML:

```html
{% extends 'customers/base.html' %}          You, 3 weeks ago • Updated quiz feature logic and UI i
{% load static %}

{% block title %}Delete Customer{% endblock %}

{% block content %}
  <h2>Delete Customer</h2>
  <p>Are you sure you want to delete customer {{ customer.customer_name }} ({{ customer.custome
  <form method="post">
    {% csrf_token %}
    <button type="submit" class="btn">Yes, delete</button>
    <a href="{% url 'customer_list' %}">Cancel</a>
  </form>
{% endblock %}
```

## Customers_Form:

```
expense_management > apps > customers > templates > customers > dj customer_form.html
        You, 3 weeks ago | 1 author (You)
    1   {% extends 'customers/base.html' %}        You, 3 weeks ago • Updated quiz feature logic and UI
    2   {% load static %}
    3
    4   {% block title %}Customer Form{% endblock %}
    5
    6   {% block content %}
    7     <h2>Customer</h2>
    8     <form method="post">
    9       {% csrf_token %}
   10       <table>
   11         {{ form.as_table }}
   12       </table>
   13       <button type="submit" class="btn">Save</button>
   14       <a href="{% url 'customer_list' %}">Cancel</a>
   15     </form>
   16   {% endblock %}
   17
```

## Urls For The Site:

```
expense_management > apps > customers > urls.py > ...
        You, 4 weeks ago | 1 author (You)
    1   from django.urls import path
    2   from . import views
    3
    4   urlpatterns = [
    5       path('', views.customer_list, name='customer_list'),
    6       path('add/', views.customer_create, name='customer_create'),
    7       path('edit/<int:pk>/', views.customer_update, name='customer_update'),
    8       path('delete/<int:pk>/', views.customer_delete, name='customer_delete'),
    9   ]
   10
```

# CHAPTER 6:

# SCREENSHOTS

## Home Page:



**Expense Management**   Logged in as **Jeffery**   ☰   Home   Businesses   Accounts   Customers   Categories   Transactions   Budgets   Reports   Dashboard   Logout

## Welcome to Users

This dashboard shows a quick summary of recent activity and links to reports generated from your data.

### Current month (11/2025)

- Transactions this month: **0**
- Total income: **0**
- Total expenses: **0**

### Quick links

- Reports home
- Monthly report
- Cashflow report
- View all users

# User Page:

Expense
Management

Logged in as
**Jeffery**

☰

Home   Businesses   Accounts   Customers   Categories   Transactions   Budgets

Reports   Dashboard   Logout

## Users

[Add new user](#)

| ID | Name | Email | Actions |
|----|------|-------|---------|
| 1 | Tom (Sales) | Tom@gmail.com | Edit | Delete |
| 2 | Sam (Finances) | Sam@gmail.com | Edit | Delete |
| 3 | Max (Toys Business) | Max@gmail.com | Edit | Delete |

© Expense Management

# Businesses:

Expense
Management

Logged in as
**Jeffery**

☰

Home   Businesses   Accounts   Customers   Categories   Transactions   Budgets

Reports   Dashboard   Logout

## Businesses

**Add business**

**Soap company**

Owner: **Tom (Sales)**

Edit   **Delete**

**Little Toys**

Owner: **Max (Toys Business)**

Edit   **Delete**

**Helping Finances**

Owner: **Sam (Finances)**

Edit   **Delete**

**Iron export**

Owner: **Nandy (Business)**

Edit   **Delete**

**Software As A Service**

Owner: **George (Business)**

Edit   **Delete**

# Accounts For the Businesses Created:

Expense Management | Logged in as **Jeffery** | ☰ | Home | Businesses | Accounts | Customers | Categories | Transactions | Budgets | Reports | Dashboard | Logout

## Accounts

**Add account**

| ID | Business | Name | Type | Balance | Actions |
|----|----------|------|------|---------|---------|
| 1 | Soap company (Owner: Tom) | Business account | debit card | 1000000.0 | Edit \| Delete |
| 2 | Little Toys (Owner: Max) | Business account | credit | 500000.0 | Edit \| Delete |
| 3 | Helping Finances (Owner: Sam) | Business account | debit card | 300000.0 | Edit \| Delete |
| 4 | Iron export (Owner: Nandy) | Business account | credit | 100000.0 | Edit \| Delete |
| 5 | Software As A Service (Owner: George) | Business account | debit card | 250000.0 | Edit \| Delete |

# Customers Accounts For The Businesses:

Expense Management | Logged in as **Jeffery** | ☰ | Home | Businesses | Accounts | Customers | Categories | Transactions | Budgets | Reports | Dashboard | Logout

## Customers

**Add customer**

| ID | Name | Email | Actions |
|----|------|-------|---------|
| 1 | kumar | kumar@gmail.com | Edit \| Delete |
| 4 | rahul | rahul@gmail.com | Edit \| Delete |
| 5 | sundar | sundar@gmail.com | Edit \| Delete |
| 6 | janu | dd@gmail.com | Edit \| Delete |

# Different Type Of Categories:

Expense
Management

Logged in as
**Jeffery**

☰

Home    Businesses    Accounts    Customers    Categories    Transactions    Budgets

Reports    Dashboard    Logout

## Categories

**Add category**

| ID | Name | Type | Actions |
|----|------|------|---------|
| 1 | Helping Finances | finance | Edit \| Delete |
| 2 | Iron export | Export | Edit \| Delete |
| 3 | Software As A Service | Online Service | Edit \| Delete |
| 4 | Little Toys | Toys | Edit \| Delete |

# Transaction History From Company to Customers:

Expense
Management

Logged in as
**Jeffery**

☰

Home    Businesses    Accounts    Customers    Categories    Transactions    Budgets

Reports    Dashboard    Logout

## Transactions

**Add transaction**

| ID | Account | Category | Customer | Amount | Date | Actions |
|----|---------|----------|----------|--------|------|---------|
| 1 | Business account (debit card) - Balance: 1000000.0 | soap Company (Soap) | kumar (kumar@gmail.com) | 50000.00 | Nov. 5, 2025 | Edit \| Delete |
| 2 | Business account (credit) - Balance: 500000.0 | Little Toys (Toys) | rahul (rahul@gmail.com) | 60000.00 | Nov. 5, 2025 | Edit \| Delete |
| 3 | Business account (debit card) - Balance: 300000.0 | Software As A Service (Online Service) | sundar (sundar@gmail.com) | 80000.00 | Nov. 5, 2025 | Edit \| Delete |
| 4 | Business account (credit) - Balance: 100000.0 | Iron export (Export) | janu (dd@gmail.com) | 45000.00 | Nov. 5, 2025 | Edit \| Delete |

## Setting Budgets For Next Transaction:

**Expense Management**   Logged in as **Jeffery**   ☰   Home   Businesses   Accounts   Customers   Categories   Transactions   Budgets   Reports   Dashboard   Logout

### Budgets

Add budget

| ID | Business | Customer | Period | Limit | Actions |
|----|----------|----------|--------|-------|---------|
| 1 | Soap company (Owner: Tom) | kumar (kumar@gmail.com) | 11/2025 | 25000.0 | Edit \| Delete |
| 2 | Little Toys (Owner: Max) | rahul (rahul@gmail.com) | 11/2025 | 45000.0 | Edit \| Delete |
| 3 | Helping Finances (Owner: Sam) | sundar (sundar@gmail.com) | 11/2025 | 56000.0 | Edit \| Delete |
| 4 | Iron export (Owner: Nandy) | janu (dd@gmail.com) | 11/2025 | 23000.0 | Edit \| Delete |

## Monthly Reports Based On The Transaction:

**Expense Management**   Logged in as **Jeffery**   ☰   Home   Businesses   Accounts   Customers   Categories   Transactions   Budgets   Reports   Dashboard   Logout

### Monthly Report — November 2025

Transactions this month: **4**

- Total income: **0.00**
- Total expenses: **235000.00**
- Net profit: **-235000.00**
- Budget total: **149000.00**

Income change vs previous month: **N/A**

Expenses change vs previous month: **N/A**

3-month rolling average expenses: **78333.33**

**Top expense categories**

No expense transactions this month.

**Category breakdown (by expense share)**

# Separate Credentials Login For Different Users:

Expense Management | Not logged in | ☰ | Home | Businesses | Accounts | Customers | Categories | Transactions | Budgets | Reports | Dashboard | Login

## Login

Username:

Password:

Log in

### Are you a customer or a vendor?

If you're a customer or vendor, use the links below to enter your information and then log in.

- Customer registration / enter info (creates a customer record)
- Vendor / Business registration (create your business)

Back to home

# CHAPTER 7

# CONCLUSION AND FUTURE ENHANCEMENT

## Conclusion

The Project Expense Management System provides organisations with a comprehensive solution for tracking, managing, and analysing project expenses. By implementing this system, organisations can:

- Achieve better financial visibility and control
- Prevent budget overruns through real-time monitoring
- Reduce time spent on expense management and reporting
- Ensure compliance and maintain audit trails
- Make data-driven financial decisions

This system demonstrates the practical application of database design principles and provides a scalable foundation for enterprise expense management.

## Future Enhancements:

1. **Mobile Application**: Develop mobile apps for on-the-go expense tracking.

2. **AI-Powered Categorisation**: Implement machine learning for automatic expense categorisation.

3. **Integration with Banking Systems**: Direct bank feed integration for automated transaction import.

4. **Advanced Analytics**: Implement predictive analytics for budget forecasting.

5. **Multi-Currency Support**: Enable expense tracking in multiple currencies.

6. **Integration with Accounting Software**: Connect with QuickBooks, SAP, or other ERP systems.

7. **Real-time Notifications**: Email/SMS alerts for budget overruns and invoice due dates.

8. **Customizable Dashboards**: Allow users to create personalised dashboard view.

# REFERENCES

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). Database System Concepts.
McGraw-Hill.
https://dev.mysql.com/doc/
https://www.postgresql.org/docs/
https://www.oracle.com/database/documentation/
https://docs.djangoproject.com/
https://spring.io/projects/spring-boot
https://www.w3schools.com/sql/
https://en.wikipedia.org/wiki/Database_management_system