Personal Expense Tracker

PROJECT REPORT

Submitted by,

Team ID: PNT2022TMID33567

T.Periyanayaki(922519106106)

M.Pavithra(922519106105)

R.Kaviya(922519106071)

C.Midhuna(922519106091)

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

V.S.B. ENGINEERING COLLEGE, KARUR

INTRODUCTION:

Expense Tracker helps to maintain the record of daily expenses and monthly income of an users from anywhere and also generates a monthly report of the expenses in pdf format. The Expense Tracker app tracks all the expenses and helps the user to manage his/her expenses so that the user is the path of financial stability.

PROJECT OVERVIEW:

We are building an application named as “Personal Expense Tracker”. As the name suggests, this project is an app which is used to track the daily expenses of the user. It is like digital record keeping which keeps the records of expenses done by a user. The application keeps the track of the Income and Expenses both of user on a day-to-day basis. This application takes the income of a user and manage its daily expenses so that the user can save money. If you exceed daily expense allowed amount it will give you a warning, so that you don’t spend much and that specific day. If you spend less money than the daily expense allowed amount, the money left after spending is added into user’s savings. The application generates report of the expenses of each end of the month. The amount saved can be used for celebrating festivals, Birthdays or Anniversary.

PURPOSE:

The Tracking of expenses is categorised by week, month and year, it helps to see the more expenses made. To use the Expense Tracker the user has to sign up into such as name, phone no., address, email address, username, password and confirm password of the user. The user can get enlisted just a single time, per user can just one record. The remainder is set if the type future expense. The whole subtleties of the income or expense can be seen or refreshed or can be erased by long pressing the specific rundown thing. The things in the rundown can be separated by month, year and date. When the month’s end is arrived at the complete pay, all out past expense and all-out future expense are determined and shown for the user.

LITERATURE SURVEY

In a literary survey, **we analyse critically and concisely earlier research and literature related to a particular research problem, and utilize them for their own research purposes**. It helps us in understanding the significance of new research and its connections to earlier work.

LITERATURE SURVEY

In a literary survey, **we analyse critically and concisely earlier research and literature related to a particular research problem, and utilize them for their own research purposes**. It helps us in understanding the significance of new research and its connections to earlier work.

Existing Problems:

Lack of money at end of the month
Difficult to calculate net income.

References:

[1] Yusof, Suhailah Mohd and Lok- man, Sharifah Fateen Syuhada Syed, booktitle=2014 IEEE Sympo- sium on Computer Applications and Industrial Electronics (ISCAIE), title=Personal financial planner.

[2] Azhar, Nur Irdina, year = 2020, month = 01, pages = , title = Development Smart Mobile Money Management Application for Students.

[3] T. R. Masendu and A. M. Tripath, “Daily Expense Tracker”, IJRESM, vol. 5, no. 5, pp. 90–92, May 2022. Greater Noida, India.

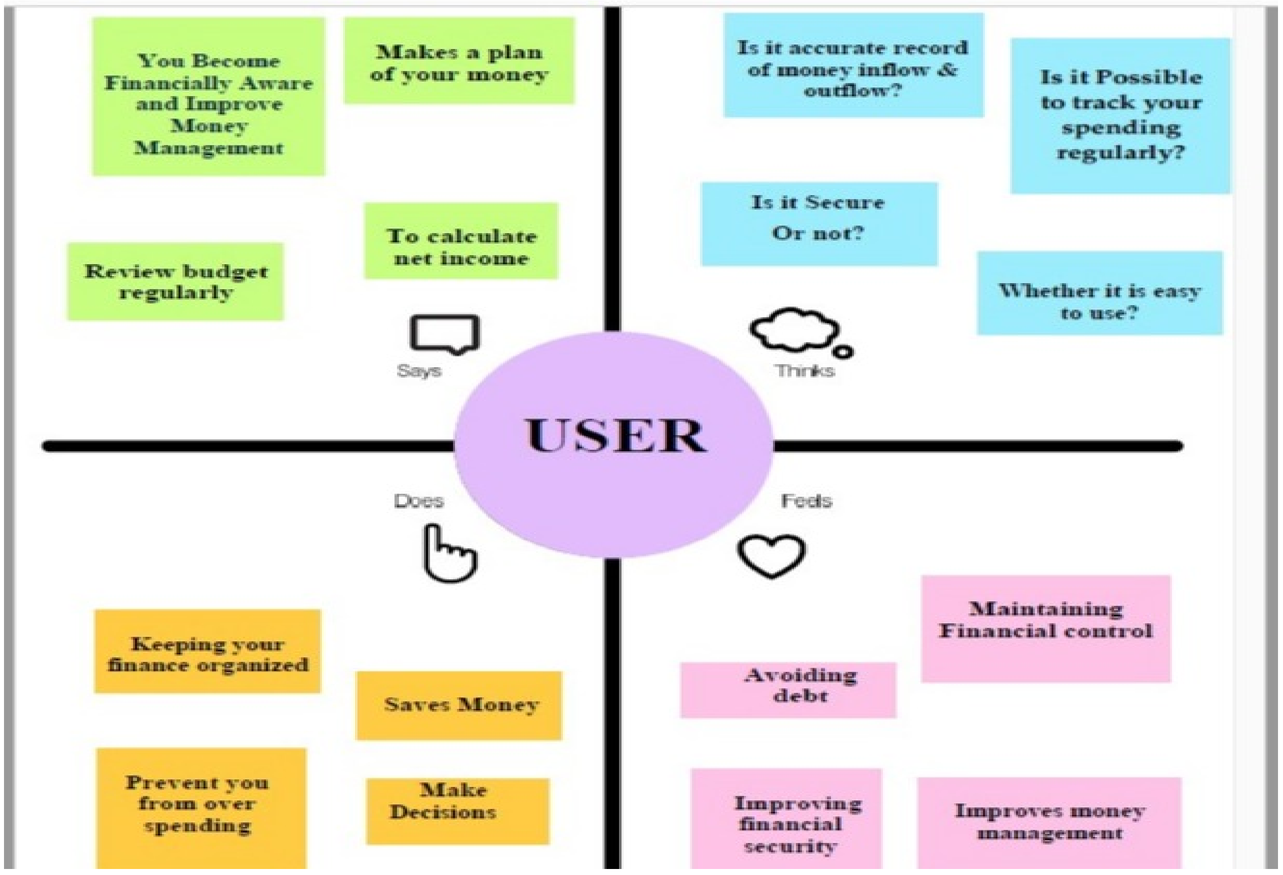
[4] Velmurugan A1, Albert Mayan J2 , Niranjana P3 and Richard Francis4 1,2Associate Professor, School of Computing, Sathyabama Institute of Science and Technology, Chennai. 3,4U.G Student, Department of CSE, Sathyabama Institute of Science and Technology, Chennai.

Problem Statement Definition :

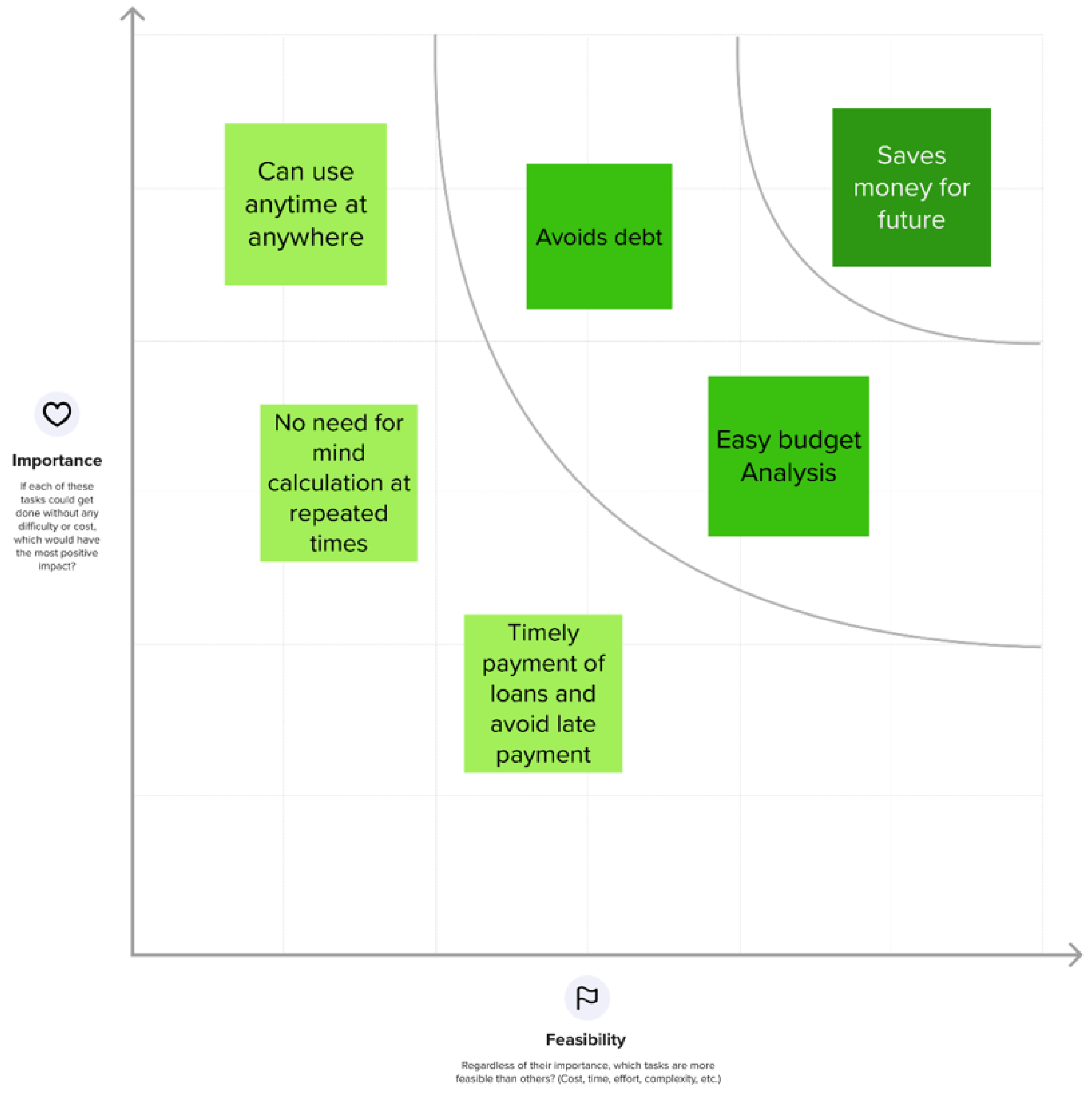
In existing, we need to maintain the Excel sheets, CSV etc. files for the user daily and monthly expenses. In existing, there is no as such complete solution to keep a track of its daily expenditure easily. To do so a personas to keep a log in a diary or in a computer, also all the calculations needs to be done by the user which may sometimes results in errors leading to losses. There can be many disadvantages of using a manual accounting system. A manual accounting system requires you to understand the accounting process in a way that may be unnecessary with a computerized accounting system. This can be an advantage or a disadvantage, depending on the person doing the bookkeeping; often, a specially trained professional is needed to ensure that accounting is done properly.To reduce manual calculations, we propose an application. This application allows users to maintain a digital automated diary. Expense Tracker application which will keep a track of Income-Expense of a user on a day to day basis.

IDEATION & PROPOSED SOLUTION:

a. Empathy Map Canvas


You Become Financially Aware and Improve Money Management
Makes a plan of your money
Review budget regularly
To calculate net income
Says
Is it accurate record of money inflow & outflow?
Is it Possible to track your spending regularly?
Is it Secure Or not?
Whether it is easy to use?
Thinks
USER
Does
Feels
Keeping your finance organized
Saves Money
Prevent you from over spending
Make Decisions
Maintaining Financial control
Avoiding debt
Improving financial security
Improves money management

Ideation & Brainstorming


Can use anytime at anywhere
Avoids debt
Saves money for future
No need for mind calculation at repeated times
Easy budget Analysis
Timely payment of loans and avoid late payment
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)

Periyanayaki T

Identify and eliminate wasteful spending habit.

spending money mindfully

ability to manage your finances wisely

Holding Yourself Accountable

Pavithra M

Avoids stress about saving for future.

Makes control over our finance

Improving financial security

It helps stick to budget

Kaviya R

Heps to set financial goals

Avoids debt

Maintains the Expense record correctly

Helps in making better budget plans

Midhuna C

Helps to meet Financial Objectives

Encourages and increase the ideas of saving

Timely payment of loans and avoid late payment

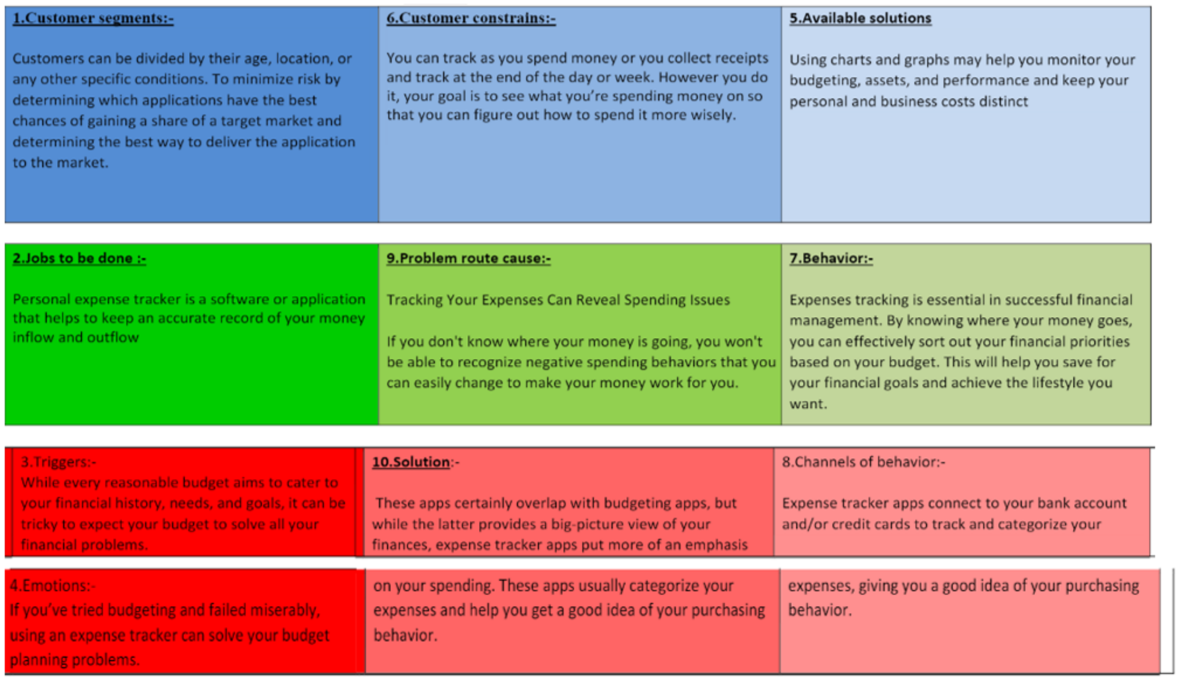
Accurate record of money inflow and outflow

Proposed Solution:

S.No.	. Parameter	Description
1	Problem Statement (Problem to be solved)	In existing, we need to maintain the Excel sheets, CSV etc. files for the user daily and monthly expenses. As now there is no as such complete solution to keep a track of its daily expenditure easily. To do so a person's to keep a log in a diary or in a computer, also all the calculations needs to be done by the user which may sometimes results in errors leading to losses. A manual accounting system requires you to understand the accounting process in a way that may be unnecessary with a computerized accounting system
2	Idea / Solution description	To reduce manual calculations, we propose an application. This application allows users to maintain a digital automated diary. Expense Tracker application which will keep a track of Income- Expense of a user on a day to day basis
3	Novelty / Uniqueness	Have a plan for your savings. This will motivate you to keep tracking, spending less and putting those saved dollars to use

4	Social Impact / Customer Satisfaction	An expense tracker helps you figure out what is happening to your money, and whether you can afford something you want
5	Business Model (Revenue Model)	Cost effective
6	Scalability of the Solution	Helps you see your money situation and figure out possible money problems before they occur

Problem Solution fit



1.Customer segments:- Customers can be divided by their age, location, or any other specific conditions. To minimize risk by determining which applications have the best chances of gaining a share of a target market and determining the best way to deliver the application to the market.	**6.Customer constrains:-** You can track as you spend money or you collect receipts and track at the end of the day or week. However you do it, your goal is to see what you're spending money on so that you can figure out how to spend it more wisely.	**5.Available solutions** Using charts and graphs may help you monitor your budgeting, assets, and performance and keep your personal and business costs distinct
2.Jobs to be done :- Personal expense tracker is a software or application that helps to keep an accurate record of your money inflow and outflow	**9.Problem route cause:-** Tracking Your Expenses Can Reveal Spending Issues If you don't know where your money is going, you won't be able to recognize negative spending behaviors that you can easily change to make your money work for you.	**7.Behavior:-** Expenses tracking is essential in successful financial management. By knowing where your money goes, you can effectively sort out your financial priorities based on your budget. This will help you save for your financial goals and achieve the lifestyle you want.
3.Triggers:- While every reasonable budget aims to cater to your financial history, needs, and goals, it can be tricky to expect your budget to solve all your financial problems.	**10.Solution**:- These apps certainly overlap with budgeting apps, but while the latter provides a big-picture view of your finances, expense tracker apps put more of an emphasis	8.Channels of behavior:- Expense tracker apps connect to your bank account and/or credit cards to track and categorize your
4.Emotions:- If you've tried budgeting and failed miserably, using an expense tracker can solve your budget planning problems.	on your spending. These apps usually categorize your expenses and help you get a good idea of your purchasing behavior.	expenses, giving you a good idea of your purchasing behavior.

REQUIREMENT ANALYSIS:

Functional requirement:

S.No	Functional Requirement	Sub Requirement (Story / Sub-Task)
1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
2	User Confirmation	Confirmation via Email Confirmation via OTP
3	Login	Enter the valid username and password
4	Calender	Personal expense tracker application shall allow user to store the daily expenses.
5	Report generation	Report generation through Message using gmail
6	Category	This application shall allow users to add categories of their expenses.

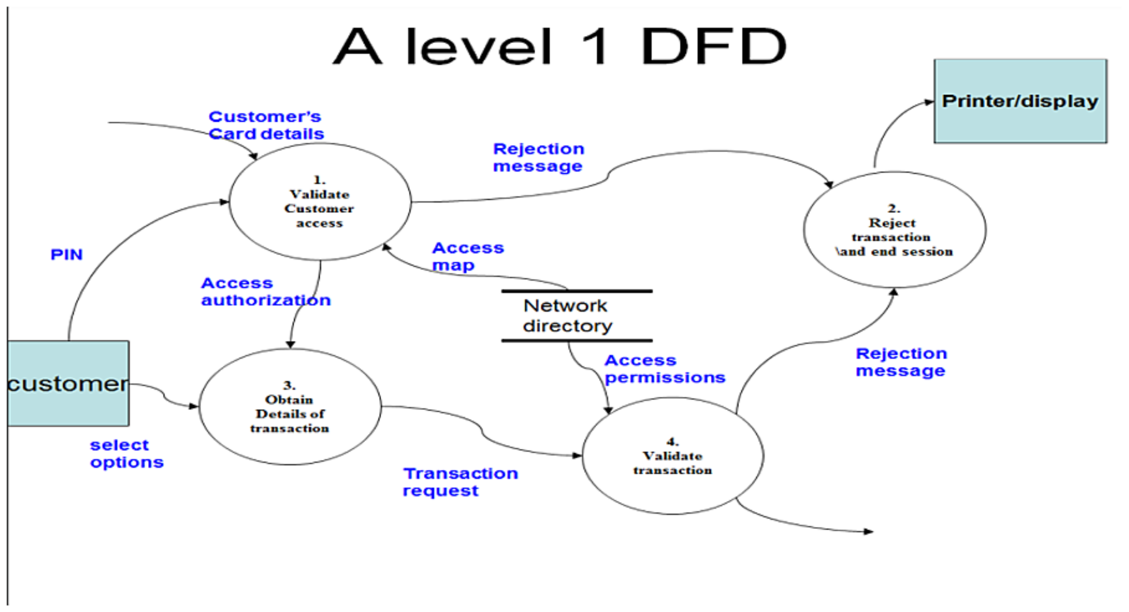
Non-Functional requirements:

S.No	Non-Functional Requirement	Description
1	Usability	Helps to keep an accurate record of your money inflow and outflow.
2	Security	Expense tracking apps are considered very safe from cyber criminal activities.
3	Reliability	Each data record is stored on a well built efficient database schema. There is no risk of data loss.
4	Performance	The types of expense are categories along with an option. Performance is very good because of the light weight database support
5	Availability	It is available all the time.No time limit for the application.
6	Scalability	The ability to appropriately handle increasing demands.

PROJECT DESIGN

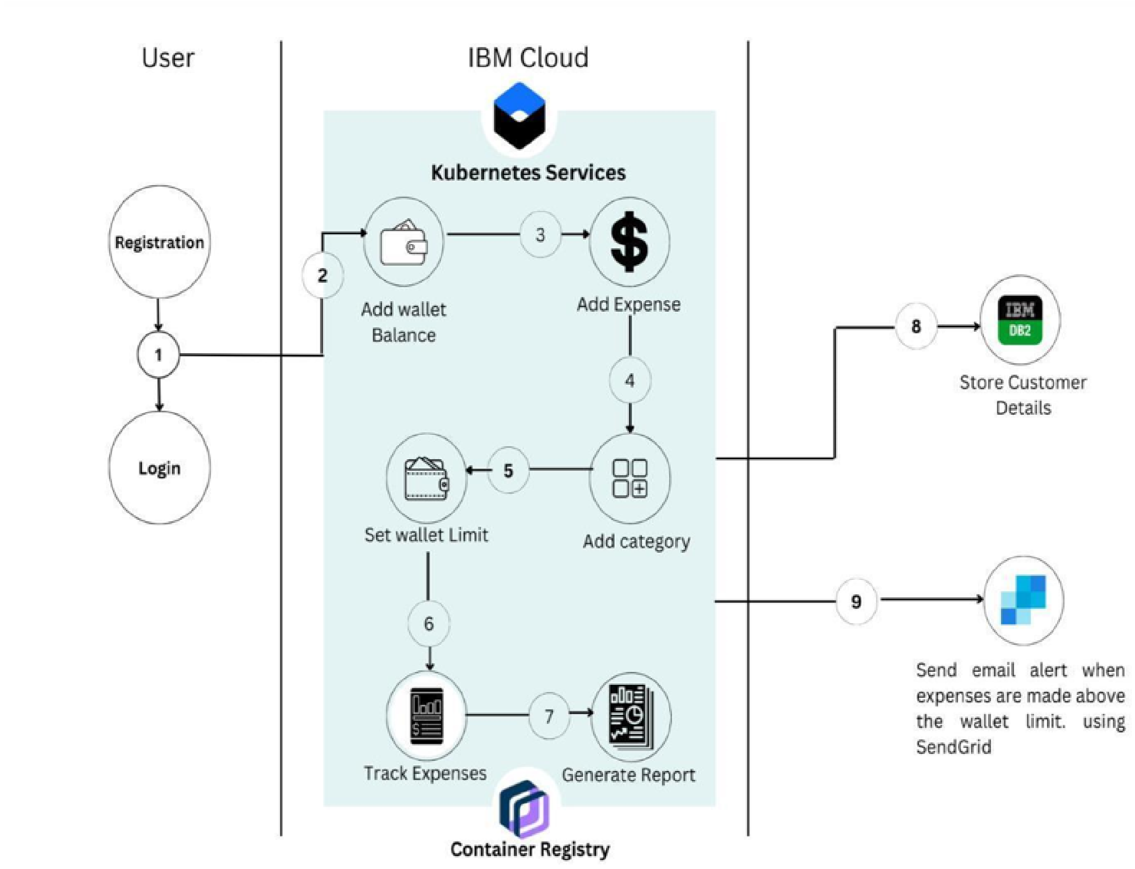
Data Flow Diagram:

A data flow diagram (DFD) **maps out the flow of information for any process or system**.
It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data
inputs, outputs, storage points and the routes between each destination.



Solution & Technical Architecture

The Deliverable shall include the architectural diagram as below and the information as per the table 1 and table 2



S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chat bot etc.	HTML, CSS, JavaScript
2.	Registration and Login	To develop the application	Python, Docker
3.	Wallet Dashboard	IBM Cloud Kubernetes Service provides a native Kubernetes experience that is secure and easy to use. This tool is used to load-balance, scale, and monitor the containers.	IBM Cloud Kubernetes Services
4.	Tracking of Expenses.	IBM Container Registry enables to store and distribute Docker images in a managed, private registry.	IBM Cloud Container Registry
5.	Database	Data Type, Configurations etc.	MySQL
6.	Cloud Database	Database Service on Cloud	IBM DB2
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	To send email alerts when the expenses are made above the wallet limit.	SendGrid

Table-2: Application Characteristics:

1.	Open-Source Frameworks	Flask is an open source framework written in Python.	Flask
2.	Security Implementations	The user accounts are configured to only allow access from users with specific privileges.	IBM DB2
3.	Scalable Architecture	Three-tier architecture- user server, application server and cloud server.	Python, IBM Cloud Services
4.	Availability	Kubernetes services, the crudest form of load balancing traffic. The most basic type of load balancing is load distribution.The Docker load balancer runs on every node and can load balance requests across any of the containers onany of the hosts in the cluster.	Kubernetes and Docker
5.	Performance	Can handle a large number of requests per second.	IBM Container Registry.

User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user & web user)	Registration	USN-1	User can register for the application by entering my email, and password, and confirming my password.	Can access my account/dashboard	High	Sprint-1

		USN-2	Then user will receive a confirmation email once they registered for the application	Can receive a confirmation email & click confirm	High	Sprint-1
		USN-3	User can register the application through Facebook3	Can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	And also,user can register for the application through a Google account.	Can register & access the dashboard with a Google Account login.	Medium	Sprint-1
	Login	USN-5	User can log into the application by entering my email & password	Can access the application.	High	Sprint-1
	Dashboard	USN-6	User can look the expenses and expenditure details regularly.	Can view the daily expenses and add the expense details.	High	Sprint-1
Customer Care Executive		USN-7	It is easy to solve the problem that is faced by the customers.	Can provide support to customers at any time 24*7.	Medium	Sprint-1
Administrator	Application	USN-8	Administrator can update the application and provide necessary upgrades.	Can fix any bugs raised by customers and upgrade the application.	Medium	Sprint-1

Project Planning Phase:

Product Backlog, Sprint Schedule and Estimation

Sprint	Functional Requirement Epic	User Story Number	User Story I Task	Story Points	Priority	Team Members
Sprint-I	Login page template	USN-I	As a user can use the login page for login purpose	1	Low	Periyanayaki.T
	Signup page template	USN-2	As a user, I can use the signup page for login purpose	1	Low	Kaviya.R
	Forget password template	USN-3	As a user, I can use the forget password page for login purpose	1	Low	Midhuna.C
	Forget password verification page team late	USN-4	As a user can view the verification of password change		Medium	Pavithra.M

	Add income page template	USN-5	As a user, I can login my income	2	High	Periyanayaki.T
	Add expense page template	USN-6	As a user, I can login my expense	2	High	Kaviya.R
	Dashboard page template	USN-7	As a user, I can view the overall states of my expense using dashboard	1	Medium	Midhuna.C
	Add budget limit	USN-8	As a user can set the budget limit in dashboard	1	Low	Pavithra.M
	Database model	USN-9	Creating a database model with SQL lite	2	High	Periyanayaki.T

Spnnt-2	Setting up IBM DB2	USN-IO	Creating and setting up IBM db2 database	2	High	Kaviya.R
	Set up IBM DB2 in flask	USN-Il	Installing and setting up the necessary tools in the flask	2	High	Midhuna.C
	Integrating IBM DB2	USN-12	Integrating IBM db2 with python flask app	2	High	Pavithra.M
	Sending data in IJI	USN-13	Sending and connecting API request response data with I-II	2	High	Periyanayaki.T
Sprint-3	IBM Watson assistant	USN-14	Creating IBM Watson assistant for chatbot service to the user	2	Medium	Kaviya.R

	Setting up send grid	USN-15	Creating send grid account and setting up the necessary libraries in the flask	1	Medium	Midhuna.C
	Integrating send grid	USN-16	By integrating send grid service you can able to receive emails with the hon flask		Medium	Pavithra.M
	Integrating chat	USN-17	By integrating chat is in the dashboard the user can overview their thin expense	2	Medium	Periyanayaki.T

Sprint-4	Containerizing app	USN-18	Containing the flask application into a docker usage	2	Medium	Kaviya.R
	Uploading to IBM cloud resist	USN-19	Uploading the docker container image to IBM Cloud	2	Medium	Midhuna.C
	Deploying in Kubernetes	USN-20	Deploying the docker container image from to the Kubernetes	2	High	Periyanayaki.T

Sprint Delivery Plan:

Project Tracker, Velocity & Burndown Chart

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date	Sprint Release Date (Actual)
Sprint-I	12	6 Days	23 Oct 2022	29 Oct 2022	20	29 Oct 2022

Sprint-2	8	6 Days	30 Oct 2022	04 Nov 2022	20	05 Nov 2022
Spnnt-3	6	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	6	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Velocity:Imagine we have a 10-day sprint duration and the velocity of the team is 20 (points per sprint). Let’s calculate the team's average velocity (AV) per iteration unit (story points per day)

AV=DURATION/VELOCITY

=20/6

=3.33

PROJECT DEVELOPMENT PHASE:

Sprint 1:

```
urlpatterns = [ path("", views.college, name="college"),
path("notice/<int:myid>/", views.notice, name="notice"),
path("application_form/", views.application_form, name="application_form"),
path("edit_application/", views.edit_application, name="edit_application"),
path("status/", views.status, name="status"), # Authentication
path("register/", views.register, name="register"), path("login/",
views.loggedin,    name="login"),                    path("logout/",    views.loggedout,
name="logout"),
# Admin     path("handle_admin/", views.handle_admin, name="handle_admin"),     path("users/",
views.users, name="users"),     path("student_application/<int:myid>/", views.student_application,
name="student_application"),     path("application_status/<int:pk>/", UpdatePostView.as_view(),
name="application_status"),     path("approved_applications/", views.approved_applications,
name="approved_applications"),     path("pending_applications/", views.pending_applications,
name="pending_applications"),     path("rejected_applications/", views.rejected_applications,
name="rejected_applications"),
]
```

Code Explanation:

It is considered to be a good practice to create a separate urls file for each app. The urls are into three parts

1) For users

2) User Authentication

3) For admin

Models.py :

```
from django.db import models from
django.contrib.auth.models import User from
django.utils.timezone import now from
django.urls import reverse


class Application(models.Model):
    COURSES = (
    ('Computer Science Engineering', 'Computer Science Engineering'),
    ('Information Technology Engineering', 'Information Technology Engineering'),
    ('Electronics and Telecommunication Engineering', 'Electronics and Telecommunication Engineering'),
('Electronics Engineering', 'Electronics Engineering'),
    )

    STATUS = (
        ('Approved', 'Approved'),
        ('Pending', 'Pending'),
        ('Rejected', 'Rejected'),
    )

    user = models.OneToOneField(User, on_delete=models.CASCADE, blank=True, null=True)
course = models.CharField(max_length=100, choices= COURSES)     name =
models.CharField(max_length=200)      email = models.CharField(max_length=200)      phone_no
= models.CharField(max_length=200)      address = models.TextField(max_length=200)
student_profile = models.ImageField(upload_to="images")      ssc_percentage =
models.DecimalField(max_digits=4, decimal_places=2, null=True)     ssc_marksheet =
models.ImageField(upload_to="images", null=True) ssc_passing_certificate =
models.ImageField(upload_to="images", null=True)     ssc_leaving_certificate =
models.ImageField(upload_to="images", null=True)     hsc_percentage =
models.DecimalField(max_digits=4, decimal_places=2, null=True)     hsc_marksheet =
models.ImageField(upload_to="images", null=True)     hsc_passing_certificate =
models.ImageField(upload_to="images", null=True)     hsc_leaving_certificate =
models.ImageField(upload_to="images", null=True)     cet_percentile =
models.DecimalField(max_digits=5, decimal_places=3, null=True)     cet_scorecard =
models.ImageField(upload_to="images", null=True)     jee_percentile =
models.DecimalField(max_digits=5, decimal_places=3, null=True)     jee_scorecard =
models.ImageField(upload_to="images", null=True)
    Application_Status = models.TextField(max_length=100, choices=STATUS, default="Pending")
message = models.TextField(max_length=100, default="")
```

```
    def str(self):
        return self.name

    def get_absolute_url(self):
return reverse('users')

class Notice(models.Model):
    title = models.CharField(max_length=200)

    def str(self):
        return self.title

class Detail(models.Model):
    title = models.ForeignKey(Notice, on_delete=models.CASCADE)     notice
= models.CharField(max_length=200)

    def str(self):
        return self.notice
```

Code Explanation:

The most important model of python college admission system is the Application model. It stores all the details of the students personal and educational details. The student while filling the application form gives all these details. The status and message are edited by the admin. Notice and Detail model stores the notice for first, second, third, and fourth year students. It is possible to add any notice for any category of students.

1. For the home page, all the notice for different year students will be shown (college.html):

```
<div class="container mt-4">
    <h1>Important Notice</h1>
<div class="row mt-4">
  {% for i in notice %}
```

```
    <div class="col-sm-6">
      <div class="card">
        <div class="card-body">
          <h5 class="card-title">{{i.title}}</h5>
          <p class="card-text"><a href="/notice/{{i.id}}/">View all recent updates.</a></p>
        </div>
      </div>
    </div>
    {% endfor %}
  </div>
  </div>
```

Views.py :

```
def college(request):
    notice = Notice.objects.all()     return render(request,
"college.html", {'notice':notice})
```

Code Explanation:

On the first page of the project all the notices will be displayed by using the for loop from the Notice model. Students can see the notice by clicking on the title regarding their year or branch.

Sprint 2:

1 . A form is created for the user registration (register.html):

```
<form action="/register/" method="POST"> {% csrf_token %}
    <div class="container mt-5">
        <div class="mb-3">
            <label for="username" class="form-label">Username</label>
            <input type="text" class="form-control" id="username" name="username">
        </div>
        <div class="mb-3">
          <label for="first_nam" class="form-label">First Name</label>
          <input type="text" class="form-control" id="first_name" name="first_name">
        </div>
        <div class="mb-3">
          <label for="last_name" class="form-label">Last Name</label>
          <input type="text" class="form-control" id="last_name" name="last_name">
        </div>
        <div class="mb-3">
          <label for="email" class="form-label">Email address</label>
          <input type="email" class="form-control" id="email" name="email">
        </div>
        <div class="mb-3">
          <label for="password1" class="form-label">Password</label>
          <input type="password" class="form-control" id="password1" name="password1">          </div>
        <div class="mb-3">
          <label for="password2" class="form-label">Confirm Password</label>
          <input type="password" class="form-control" id="password2" name="password2">
        </div>
        <button type="submit" class="btn btn-primary">Submit</button>
    </div>
</form>
```

Views.py :

```
def register(request):     if
request.method=="POST":           username =
request.POST['username']        email =
request.POST['email']
first_name=request.POST['first_name']
```

```
last_name=request.POST['last_name']

password1 = request.POST['password1']

password2 = request.POST['password2']


        if password1 != password2:
messages.error(request, "Passwords do not match.")             return
redirect('/register')


        user = User.objects.create_user(username, email, password1)
user.first_name = first_name         user.last_name = last_name         user.save()
return render(request, 'login.html')        return render(request, "register.html")
```

Code Explanation:

We create a form with a post request to take the required details from the user. These are then stored in the Django User model. These details will be further required to identify the user while logging in.

2.Then a form is created for the user login (login.html):

```
<form action="/login/" method="POST"> {% csrf_token %}
    <div class="container mt-5">
    <div class="mb-3">
      <label for="username" class="form-label">Username</label>
      <input type="text" class="form-control" id="username" name="username">
    </div>
    <div class="mb-3">
      <label for="password" class="form-label">Password</label>
      <input type="password" class="form-control" id="password" name="password">
    </div>
    <br>
    <button type="submit" class="btn btn-primary">Submit</button>
```

```
  </div>
</form>
```

Views.py :

```
  def loggedin(request):     if request.user.is_authenticated:
  return redirect("/")     else:
 if request.method=="POST":
username =
request.POST['username']
password =
request.POST['password']

user = authenticate(username=username, password=password)

if user is not None:
login(request, user)
messages.success(request,
"Successfully Logged In")
 return redirect("/")
 else:
messages.error(request, "Invalid Credentials")
return render(request, 'college.html')
return render(request, "login.html")
```

Code Explanation:

If the user wants to fill the application form, then the user has to register and then login. While login if the username and password is wrong then “Invalid Credentials” message will be shown else the user will get successfully logged in.

Sprint 3:

1.After login the student can fill the application form and if the student has already filled the application form, then the student can view and edit the form. (application_form.html):

```
{% if hide.exists %}
<div class="container mt-4">
  <div class="row">
    <h1 style="text-align: center;">Personal Details</h1>
      <hr>
    <div class="col-md-4">
      <div class="profile-img">
        {% if user.application.student_profile.url %}
        <img src="{{user.application.student_profile.url}}" alt="" width="310px" height="270px">
        {% endif %}
      </div>
    </div>
    <div class="col-md-8">
      <div class="profile-tab">
        <div class="tab-pane">
          <br><br>
          <div class="row">
            <div class="col-md-6">
              <label>Course :</label>
            </div>
            <div class="col-md-6">
              <p>{{user.application.course}}</p>
            </div>
          </div>
          <div class="row">
            <div class="col-md-6">
              <label>Full Name :</label>
            </div>
            <div class="col-md-6">
            <p>{{user.application.name}}</p>
        </div>
    </div>
    <div class="row">
        <div class="col-md-6">
            <label>Email ID :</label>
        </div>
        <div class="col-md-6">
            <p>{{user.email}}</p>
        </div>
    </div>
    {% if user.application.phone_no %}
    <div class="row">
        <div class="col-md-6">
            <label>Phone Number :</label>
        </div>
        <div class="col-md-6">
            <p>{{user.application.phone_no}}</p>
        </div>
    </div>
    {% endif %}
    <div class="row">
        <div class="col-md-6">
            <label>Address :</label>
        </div>
        <div class="col-md-6">                              <p>{{user.application.address}}</p>
        </div>
    </div>
</div>
```

```
            </div>
        </div>
</div>
<hr>
<div class="container mt-4">
    <div class="row">
        <h1 style="text-align: center;">Educational Details</h1>
            <hr>
        <div class="col-md-4">
            <div class="profile-img">
                <h3>10th Std Details</h3>
            </div>
        </div>
        <div class="col-md-8">
            <div class="profile-tab">
                <div class="tab-pane">
                    <br><br>
                    <div class="row">
                        <div class="col-md-6">
                            <label>SSC Percentage :</label>
                        </div>
                        <div class="col-md-6">
                            <p>{{user.application.ssc_percentage}}%</p>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-md-6">
                            <label>SSC Marksheet :</label>
                        </div>
                        <div class="col-md-6">
                            <p><a href="{{user.application.ssc_marksheet.url}}">View SSC Marksheet</a></p>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-md-6">
                            <label>SSC Passing Certificate :</label>
                        </div>
                        <div class="col-md-6">

<p><a href="{{user.application.ssc_passing_certificate.url}}"> View SSC Passing Certificate</a></p>
                        </div>
                    </div>
                    {% if user.application.phone_no %}
                    <div class="row">
                        <div class="col-md-6">
                            <label>SSC Leaving Certificate :</label>
                        </div>
                        <div class="col-md-6">
                            <p><a href="{{user.application.ssc_leaving_certificate.url}}">View SSC Leaving
Certificate</a></p>
                        </div>
                    </div>
                    {% endif %}
                </div>
            </div>
        </div>
</div>
<hr>
```

```
<div class="container mt-4">
    <div class="row">
        <div class="col-md-4">
            <div class="profile-img">
                <h3>12th Std Details</h3>
            </div>
        </div>
        <div class="col-md-8">
            <div class="profile-tab">
                <div class="tab-pane">
                    <br><br>
                    <div class="row">
                        <div class="col-md-6">
                            <label>HSC Percentage :</label>
                        </div>
                        <div class="col-md-6">
                            <p>{{user.application.hsc_percentage}}%</p>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-md-6">
                            <label>HSC Marksheet :</label>
                        </div>
                        <div class="col-md-6">
                            <p><a href="{{user.application.hsc_marksheet.url}}">View HSC Marksheet</a></p>
                        </div>
                    </div>
                    <div class="row">
```

```
                        <div class="col-md-6">
                            <label>HSC Passing Certificate :</label>
                        </div>
                        <div class="col-md-6">
                            <p><a href="{{user.application.hsc_passing_certificate.url}}">View HSC
Marksheet</a></p>
                        </div>
                    </div>
                    {% if user.application.phone_no %}
                    <div class="row">
                        <div class="col-md-6">
                            <label>HSC Leaving Certificate :</label>
                        </div>
                        <div class="col-md-6">
                            <p><a href="{{user.application.hsc_leaving_certificate.url}}">View HSC Leaving
Certificate</a></p>
                        </div>
                    </div>
                    {% endif %}
                </div>
            </div>
        </div>
</div>
<hr>

<div class="container mt-4">
    <div class="row">
        <div class="col-md-4">
            <div class="profile-img">
                <h3>CET and JEE Scorecard</h3>
```

```
          </div>
        </div>
        <div class="col-md-8">
          <div class="profile-tab">
            <div class="tab-pane">
              <br><br>
              <div class="row">
                <div class="col-md-6">
                  <label>CET Percentile :</label>
                </div>
                <div class="col-md-6">
                  <p>{{user.application.cet_percentile}}</p>
                </div>
              </div>
              <div class="row">
                <div class="col-md-6">
                  <label>CET Scorecard :</label>
                </div>
                <div class="col-md-6">
                  <p><a href="{{user.application.cet_scorecard.url}}">View CET Scorecard</a></p>
</div>
              </div>
              <div class="row">
                <div class="col-md-6">
                  <label>JEE Percentile :</label>
                </div>

<div class="col-md-6">
                  <p>{{user.application.jee_percentile}}</p>
```

```
                    </div>
                  </div>
                  {% if user.application.phone_no %}
                  <div class="row">
                    <div class="col-md-6">
                      <label>JEE Scorecard :</label>
                    </div>
                    <div class="col-md-6">
                      <p><a href="{{user.application.jee_scorecard.url}}">View JEE Scorecard</a></p>
                    </div>
                  </div>
                  {% endif %}
                </div>
              </div>
            </div>
        </div>
        <hr>
        <h4>Click here to edit the application</h4><a href="/edit_application/" class="btn btn-secondary"
        style="width: 20rem;">Edit Application</a>
        <br><br>
        {% else %}
        <div class="container mt-4">
          <form action="/addmission_form/" method="POST" enctype="multipart/form-data"> {% csrf_token
        %}
          {{form.as_p}}
          <button type="submit">Submit</button>
          </form>
        </div> {% endif
        %}
```

Views.py :

```
def application_form(request):
    hide = Application.objects.filter(user=request.user)     if
request.method=="POST":
        form = ApplicationForm(request.POST, request.FILES)
        if form.is_valid():
            application = form.save()            application.user
= request.user            application.save()            return
render(request, "application_form.html")
    else:
        form=ApplicationForm()     return render(request,
"application_form.html", { 'form':form,'hide':hide})
```

Code Explanation:

After the student is logged in, he/she can fill the application form on python college admission system and upload all the original documents asked in the application form.

```
hide = Application.objects.filter(user=request.user)
```

We then check that the logged in students application already exists or not and save it in the “hide” variable. Then the “hide” variable is used in the templates as follows:

```
{% if hide.exists %}
```

This means that the logged in student has already filled the application form so the whole application filled by the particular student will be displayed with an edit option. If this condition fails, that is the logged in student has not filled the application form, then the else statement will get executed, that is the entire form will be displayed on python college admission system.

2.Status of the application (status.html):

```
<div class="container w-50 mt-4 shadow">
    <h3>Your application is {{application.Application_Status}}</h3>
```

```
<br>

   <h5>{{application.message}}</h5>

</div>
```

Views.py:

```
def status(request):

   application = Application.objects.get(user=request.user)      return

render(request, "status.html", {'application':application})
```

Code Explanation:

After successfully submitting the application form, the student can check the status of the form by navigating to the application status option given on the navigation bar. At first the status will be pending, as pending status is given as default in the models.py. After viewing the application form the admin can change the status of the application only. The admin can change the status from Pending to Approved or Rejected depending upon the student's application form. Lastly, the status will get displayed with a message if any.

Sprint 4:

1.Admin's home page (handle_admin.html):

```
<div class="box">

        <h2><a href="/users/"> All Users ({{users}}) </a></h2>

     </div>

     <div class="box">

        <h2><a href="/approved_applications/"> Approved Applications ({{approve}}) </a></h2>

   </div>

   <div class="box">

      <h2><a href="/pending_applications/"> Pending Applications ({{pending}}) </a></h2>

   </div>

   <div class="box">

      <h2><a href="/rejected_applications/"> Rejected Applications ({{reject}}) </a></h2>    </div>
```

Views.py:

```
def handle_admin(request):
    if not request.user.is_superuser:        return redirect("/login")     users =
User.objects.all().count     approve =
Application.objects.filter(Application_Status='Approved').count     reject =
Application.objects.filter(Application_Status='Rejected').count     pending =
Application.objects.filter(Application_Status='Pending').count
 return render(request, "handle_admin.html", {'approve':approve, 'reject':reject, 'pending':pending,
'users':users})
```

Code Explanation:

On the admin's home page four options are created to view all the applications, the pending applications, the approved applications and the rejected applications respectively. The admin can basically check all the pending applications and then approve or reject it.

```
approve = Application.objects.filter(Application_Status='Approved').count
```

The above line basically fetches the number of applications from the Application model filtering the application status to be approved. Similarly, the number of applications is stored for all users, pending and rejected applications.

2. Admin can view and change the status of the application (student_application.html):

```
<div class="container mt-4">
    <div class="row">
        <h1 style="text-align: center;">Personal Details</h1>
            <hr>
        <div class="col-md-4">
            <div class="profile-img">
                {% if user.application.student_profile.url %}
                <img src="{{user.application.student_profile.url}}" alt="" width="310px" height="270px">
                {% endif %}
            </div>
        </div>
        <div class="col-md-8">
            <div class="profile-tab">
                <div class="tab-pane">
                    <br><br>
                    <div class="row">
                        <div class="col-md-6">
                            <label>Course :</label>
                        </div>
                        <div class="col-md-6">
                            <p>{{user.application.course}}</p>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-md-6">
                            <label>Full Name :</label>
                        </div>
                        <div class="col-md-6">
                            <p>{{user.application.name}}</p>
                        </div>
                    </div>
                    <div class="row">
                        <div class="col-md-6">
                            <label>Email ID :</label>
                        </div>
                        <div class="col-md-6">
                            <p>{{user.email}}</p>
                            </div>
                        </div>
                        {% if user.application.phone_no %}
                        <div class="row">
                            <div class="col-md-6">
                                <label>Phone Number :</label>
                            </div>
                            <div class="col-md-6">                              <p>{{user.application.phone_no}}</p>
                            </div>
                        </div>
                        {% endif %}
                        <div class="row">
                            <div class="col-md-6">
                                <label>Address :</label>
                            </div>
                            <div class="col-md-6">
                                <p>{{user.application.address}}</p>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
        <hr>

        <div class="container mt-4">
            <div class="row">
                <h1 style="text-align: center;">Educational Details</h1>
                    <hr>
```

```
<div class="col-md-4">
    <div class="profile-img">
        <h3>10th Std Details</h3>
    </div>
</div>
<div class="col-md-8">
    <div class="profile-tab">
        <div class="tab-pane">
            <br><br>
            <div class="row">
                <div class="col-md-6">
                    <label>SSC Percentage :</label>
                </div>
                <div class="col-md-6">
                    <p>{{user.application.ssc_percentage}}%</p>
                </div>
            </div>
            <div class="row">
                <div class="col-md-6">
                    <label>SSC Marksheet :</label>
                </div>
                <div class="col-md-6">
                    <p><a href="{{user.application.ssc_marksheet.url}}">View SSC Marksheet</a></p>
                </div>
            </div>
            <div class="row">
                <div class="col-md-6">
                    <label>SSC Passing Certificate :</label>
                </div>
```

```
                        <div class="col-md-6">
                            <p><a href="{{user.application.ssc_passing_certificate.url}}"> View SSC Passing
Certificate</a></p>
                        </div>
                    </div>
                    {% if user.application.phone_no %}
                    <div class="row">
                        <div class="col-md-6">
                            <label>SSC Leaving Certificate :</label>
                        </div>
                        <div class="col-md-6">
                            <p><a href="{{user.application.ssc_leaving_certificate.url}}">View SSC Leaving
Certificate</a></p>
                        </div>
                    </div>
                    {% endif %}
                </div>
            </div>
        </div>
</div>
<hr>

<div class="container mt-4">
    <div class="row">
        <div class="col-md-4">
            <div class="profile-img">
                <h3>12th Std Details</h3>
            </div>
        </div>
    <div class="col-md-8">
      <div class="profile-tab">
        <div class="tab-pane">
          <br><br>
          <div class="row">
            <div class="col-md-6">
              <label>HSC Percentage :</label>
            </div>
            <div class="col-md-6">
              <p>{{user.application.hsc_percentage}}%</p>
            </div>
          </div>
          <div class="row">
            <div class="col-md-6">
              <label>HSC Marksheet :</label>
            </div>
            <div class="col-md-6">
              <p><a href="{{user.application.hsc_marksheet.url}}">View HSC Marksheet</a></p>
            </div>
          </div>
          <div class="row">
            <div class="col-md-6">
              <label>HSC Passing Certificate :</label>
            </div>
            <div class="col-md-6">
              <p><a href="{{user.application.hsc_passing_certificate.url}}">View HSC
Marksheet</a></p>
            </div>
          </div>
```

```
                    {% if user.application.phone_no %}
                    <div class="row">
                        <div class="col-md-6">
                            <label>HSC Leaving Certificate :</label>
                        </div>
                        <div class="col-md-6">
                            <p><a href="{{user.application.hsc_leaving_certificate.url}}">View HSC Leaving
Certificate</a></p>
                        </div>

</div>
                    {% endif %}
                </div>
            </div>
        </div>
</div>
<hr>

<div class="container mt-4">
    <div class="row">
        <div class="col-md-4">
            <div class="profile-img">
                <h3>CET and JEE Scorecard</h3>
            </div>
        </div>
        <div class="col-md-8">
            <div class="profile-tab">
                <div class="tab-pane">
                    <br><br>
```

```
<div class="row">
    <div class="col-md-6">
        <label>CET Percentile :</label>
    </div>
    <div class="col-md-6">
        <p>{{user.application.cet_percentile}}</p>
    </div>
</div>
<div class="row">
    <div class="col-md-6">
        <label>CET Scorecard :</label>
    </div>
    <div class="col-md-6">
        <p><a href="{{user.application.cet_scorecard.url}}">View CET Scorecard</a></p>
    </div>
</div>
<div class="row">
    <div class="col-md-6">
        <label>JEE Percentile :</label>
    </div>
    <div class="col-md-6">
        <p>{{user.application.jee_percentile}}</p>
    </div>
</div>
{% if user.application.phone_no %}
<div class="row">
    <div class="col-md-6">
        <label>JEE Scorecard :</label>
    </div>
```

```
                    <div class="col-md-6">
                        <p><a href="{{user.application.jee_scorecard.url}}">View JEE Scorecard</a></p>
                    </div>
                </div>
                {% endif %}
            </div>
        </div>
    </div>
</div>
<hr>
<h4>Click here to edit the status</h4><a href="/application_status/{{application.id}}/" class="btn
btnsecondary" style="width: 20rem;">Edit Status</a>
```

Views.py:

```
 def student_application(request, myid):
 if not request.user.is_superuser:
 return redirect("/login")     application = Application.objects.filter(id=myid)
return render(request, "student_application.html", {'application':application[0]})
```

Code Explanation:

After clicking on one of the 4 options from all students list, approved application list, pending application list and rejected application list the list will be displayed in the form of a table. On clicking the view application button the admin can view the particular student's application on python college admission system application. Then the admin can go through the entire application, check all the submitted documents and then change the status of the application.

Python Online College Admission System Output:

College Admission Home without student logged in:

college admission home

Register:

python college registration

Application Form:

college admission application form

College Admission Admin Home page:

python college admin home

Summary

So here we come to an end of this project and we have successfully developed an online college admission management system in Django. With this project in Django, we have developed an efficient, time-saving and easy to use system for the hectic admission process

ADVANTAGES:

- **Ability to monitor costs incrementally:** Tracking expenses throughout a project provides you the ability to view various expense categories and time periods. This can help you understand how much money you can spend for the rest of the project while staying within your budget.

- **Allows for budgeting on future projects**: If you record your expenses for a project, you can use that information to budget for similar future projects. For example, if you license software to complete a project, you can include that budget if you need to use that software for other initiatives.

- **Improved decision-making:** By tracking your expenses, you can make more informed decisions than if you didn't know how much you spent. You might spend additional money on resources to keep a project on schedule, but if you knew these expenses might make the project over budget, you could explore other options.g
- **Reimbursed staff:** Sometimes, employees may pay for certain expenses while working on a project. By tracking these costs throughout a project, you can identify what costs to repay.

Organized expense tax information:

Many companies may itemize their expenses for tax purposes. By tracking these line items throughout a project, it can be easier to organize and prepare your business records for taxes and audits.

Conclusion:

In conclusion, developing a personal budget and tracking all expenses and spending is a crucial aspect of personal finances. Set aside a fixed amount in a savings account, they say you should always have three month work of your living expenses in a savings account in case of emergencies. Lastly, educating the children early in life about personal finances should be a mandatory class in every school. Parents need to proactive in teaching their children about banking, credit card, interest rates, and credit. The importance of actually seeing my spending on my budget sheet was enlightening. However, I know now where I can trim the fat, and by changing just a few items and cutting back on others, I will see a substantial increase in money that I can put into savings.

FUTURE SCOPE:

- It will have various options to keep record (for example Food, Travelling Fuel, Salary etc.).

- Automatically it will keep on sending notifications for our daily expenditure.

- In today's busy and expensive life, we are in a great rush to make moneys, but at the end of the month we broke off. As we are unknowingly spending money on title and unwanted things. So, we have come over with the plan to follow our profit.

- Here user can define their own categories for expense type like food, clothing, rent and bills where they have to enter the money that has been spend and likewise can add some data in extra data to indicate the expense.