Cloud Application Development:-

Use-case 1:- Personal Expense Tracker Application

All the Financial Decisions and activities that you make are unable to keep a track of it

This app makes your life easier by helping you to manage your Finances Efficiently.

A Personal Finance app will not only help you with budgeting and accounting but also give you helpful insights about Financial management.

Social Impact:-

It will help the People to track their Expenses and also starts when you exceed the limit of your budget.

Business model / Impact:-

we can provide the application in a Subscription based

Existing Solution:-

https://www.spendee.com/

Recommended Technology stack:-

Python, flask, IBM DB2, Docker, IBM cloud, etc.

Reference:-

https://www.researchgate.net/Publication/347972

Expenses Manager Application

Use case 2:- Nutrition Assistant Application:-

→ Due to the improvement in People's standards of living obesity Rates are increasing at an alarming speed. And this is Reflective to the Risks in People's Health. People need to control their daily calorie intake by eating healthier food, which is the most basic Method of avoid obesity.

→ However, although food Packaging comes with nutrition (and calorie) labels, it's still not very convenient for People to Refer.

→ APP-based Nutrient dashboard systems which can analyze Real time, images of meal and analyze it for analyze it for nutritional content can be very handy and improve the dietary habit

Social Impact:-

It will help People with Providing Proper nutrition and helps in maintaining a Healthy lifestyle.

Business model/Impact:-

→ Social media is the best way to spread the world about our application

Need for Flask:-

-> Easy to learn
-> written in Python
-> Routes by decorates
-> Import only what you need
-> New plugins every day.

Flask Installation:

open Command Prompt

http://flask.

$ pip install Flask

Static Files:

Dynamic web applications also need static files.

Pgm:-

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello_world():
    return 'Hello World'
if __name__ == '__main__':
    app.run()
```

(local host: 5000/)

Flask:-

A micro framework written in

Flask is based on Werkzeug and Sinja 2

Werkzeug is comprehensive web Server Gateway Interface (WSGI) utility library.

WSGI:-

WSGI is the web Server Gateway Interface.

other Python Framework based on WSGI

-> Django
-> Bottle
-> Pyramid
-> Webapy

framework for building web application in Python.

```
<!doctype html>
<html>
  <body>
    <h3> Hello world </h3>
  </body>
</html>
```

Dependencies:-

These distributions will be installed automatically when installing Flask.

Werkzeug implements WSGI, the standard Python interface between applications and servers.
Jinja is a template language that renders the pages your application serves.
Markupsafe comes with Jinja. It escapes untrusted input when rendering templates to avoid injection attacks.
Itsdangerous securely signs data to ensure it's integrity. This is used to protect Flask's session cookie.
click: click is a framework for writing command line applications. It provides the Flask command and allows adding custom management commands

Optional dependencies:-

These distribution will not be installed automatically.

→ Blinker provides support for signals.
→ watchdog provides a faster, more efficient reloader for the development server.
→ Python-dotenv enables support for Environment variable from .flaskenv when running flask commands