Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	14 October 2022
Team ID	PNT2022TMID15282
Project Name	Personal Expense Tracker Application
Maximum Marks	4 Marks

Technical Architecture:

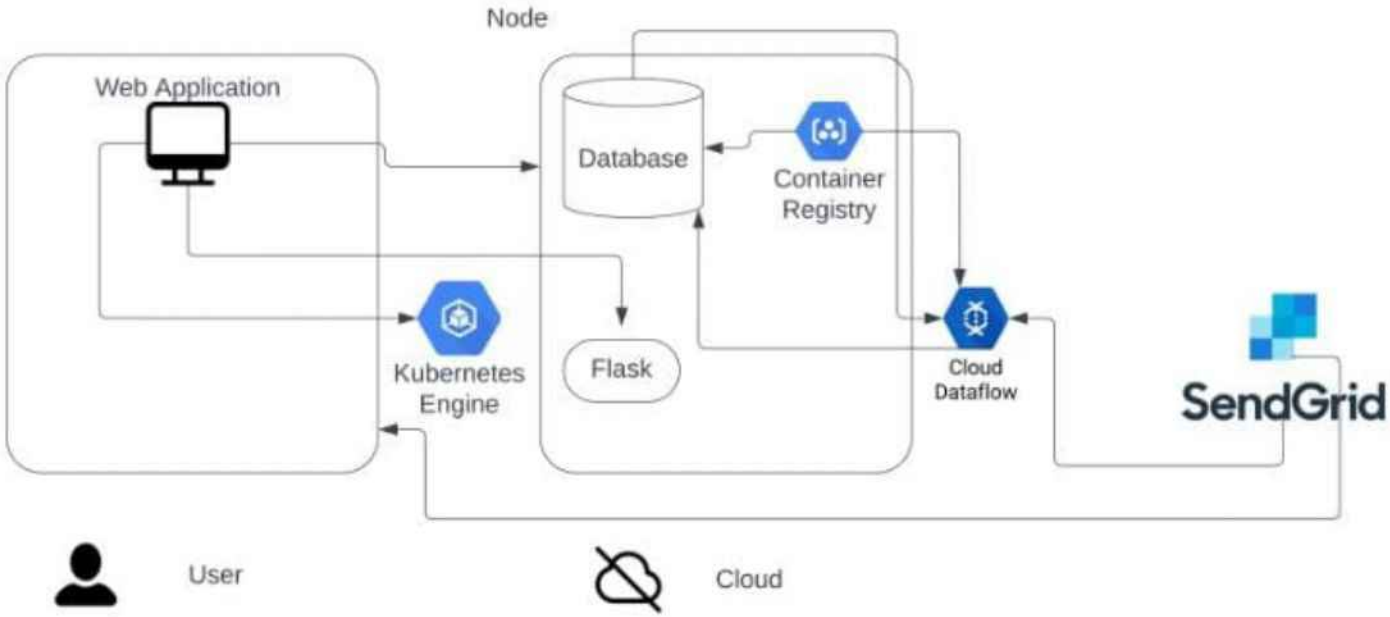

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript,Flask,Python.
2.	Application Logic-Creating an account	Users using App;s UI can create an account	Python
3.	Application Logic-Logging in	Users can log in with user name and password after creating an account.	Flask App running using Kubernetes Cluster.
4.	Application Logic-Creating An Expense	Users can add expenses in the portal using App's UI	Flask App running using Kubernetes Cluster.
5.	Application Logic-Updating An Expense	Users can update the expenses.	Flask App running using Kubernetes Cluster.
6.	Application Logic-Adding the bills	Users can add their bills using App's UI components.	Flask App running using Kubernetes Cluster.
7.	Database	Data types will be user dependent as it is customisable,user can define it according to their use.	IBM DB2
8.	Cloud Database	Kubernetes Cluster services,Cloud's other services will be used.	IBM DB2
9.	File Storage	This requires hard disk of size 8GB RAM	HardDisk
10.	External API-sendGrid	The app sendGrid will be used for giving alerts.	sendGrid services.
11.	Cloud	Application Deployment on Cloud Cloud Server Configuration : Virtual server for VPC	Local, Cloud Foundry, Kubernetes, etc.
12.	Deployment	Application will be Deployment on Local System / Cloud Local Server Configuration: The application will be running on the local server/client side to allow user to interact with Web UI components.	Cloud Foundry, Kubernetes,Cloud Registry.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask is a open source web framework, that is a Python module that lets you to develop web applications easily. It has a small and easy-to-extend core: it is a microframework that does not include an ORM (Object Relational Manager) or such features.	Flask services
2.	Security Implementations	Access to the DB2 database system is managed by facilities that reside outside the DB2 database system (authentication). Kubernetes expects that all API communication in the cluster is encrypted by default with TLS, and the majority of installation methods will allow the necessary certificates to be created and distributed to the cluster components.	IAM Controls will be used for implementations.
3.	Scalable Architecture	3-tier architecture will be employed: Presentation tier – The UI is fixed for the application and thus scalability doesn't really apply to this tier. Application tier – This tier comprises of the Python logic that will be used to provide the main functionality to the application	IBM DB2, IBM Cloud Object Storage, Kubernetes to run new container images
4.	Availability	On a large scale, as majority of the application depends on the cloud, CDN's can be used and the storage can be distributed, i.e. data of the user would be stored in areas close to the user based on the availability of cloud servers which will enhance performance.	IBM Cloud Object Storage, Kubernetes, Docker Images, IBM DB2, SendGrid
5.	Performance	The performance of the application would currently be limited as the services employed belong to the trial version and would thus enable only a certain number of users etc, i.e., the performance of the application depends on the storage and compute sources available.	IBM Cloud Object Storage, Kubernetes, Docker Images, IBM DB2, SendGrid