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

Date	15 October 2022
Team ID	PNT2022TMID20341
Project Name	Personal Expense Tracker Application
Maximum Marks	4 Marks

Technical Architecture:

The Technical diagram is processed as below. The application is built using front end tech stack – HTML /CSS /JS and using Flask we form our basic functionality. Docker provides the ability to package and run an application in a loosely isolated environment called a container. Kubernetes is open-source orchestration software that provides an API to control how and where those containers will run. The data is stored in DB2 and cloud object storage. We use SendGrid to notify user when the set budget limit exceeds and other alerts.

Technology Architecture Diagram

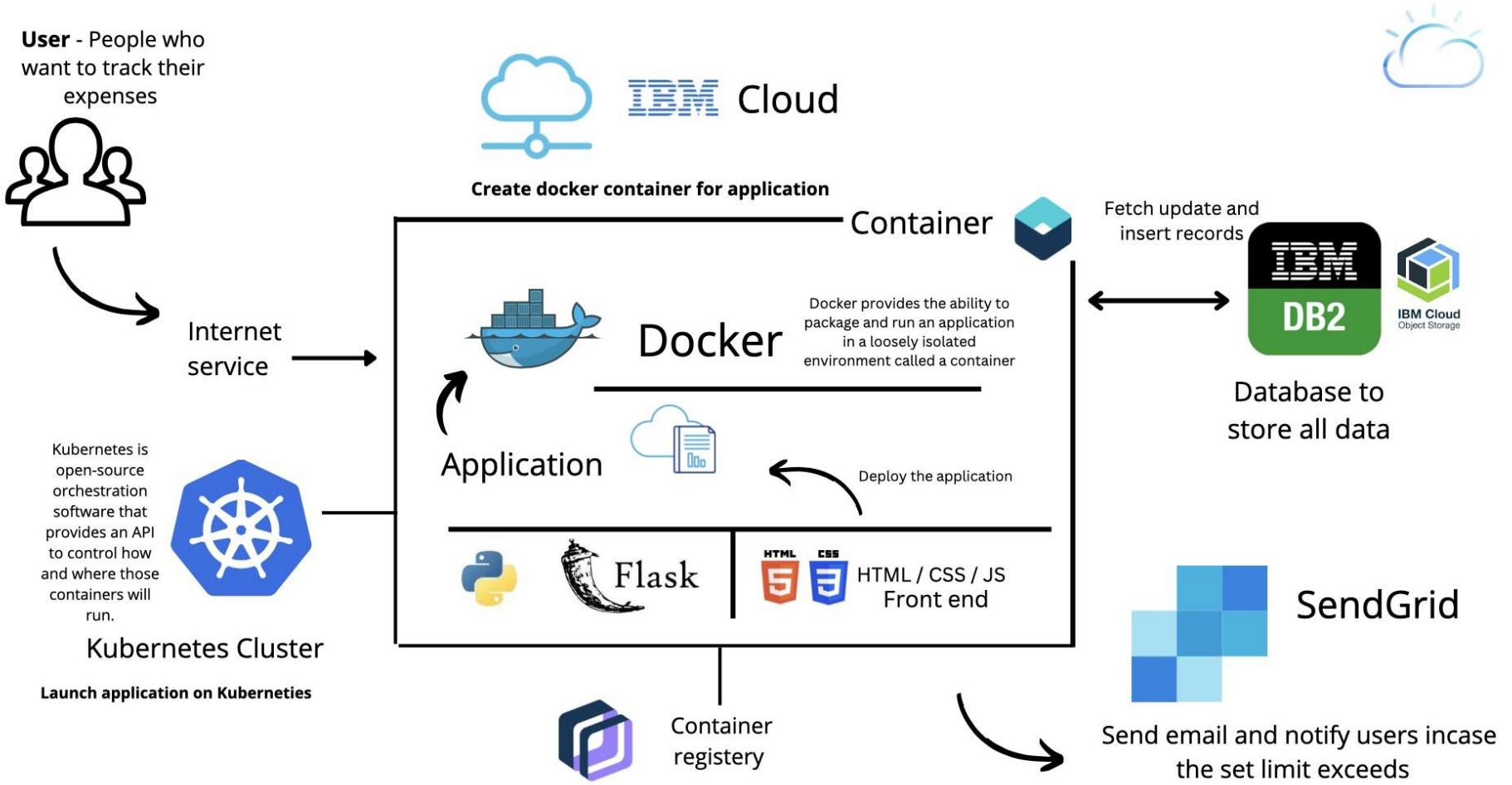

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application Web UI for navigation – signup, login, navigating through different modules.	HTML, CSS, JavaScript
2.	Direct users and add expense	Perform specific features to calculate finances and function and add, Fetch and update the database.	Python
3.	Track progress	Fetch relevant up to date details for processing.	Python
4.	IBM Watson	Chat bot easy assistance	IBM Watson Assistant
5.	Database	User profile data, Expense and files	MySQL, NoSQL,
6.	Cloud Database	Database Service on Cloud to store expense and user info.	IBM DB2, IBM Cloudant.
7.	File Storage	File storage requirements	IBM Cloud object storage
8.	External API: To send email SendGrid	Send email and notify users in-case the set limit exceeds	SendGrid
9.	Infrastructure (Server / Cloud)	Application Deployment Cloud	IBM - Docker – container, Cloud Foundry, Kubernetes container

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask – Python based framework provides structural and functional units of the web application. Navigate and integrate with other services and the functioning of the application logic.	Flask
2.	Security Implementations	Security / access controls implemented, use of IBM services for stable and secure access of data and its transactions.	SHA-256, Encryptions, IAM Controls
3.	Scalable Architecture	3 – tier, Micro-services and layered structure so app is scalable and functional according to future trends and needs.	IBM services, Flask, Front end stack(HTML / CSS / JS)
4.	Availability	Use of load balancers, distributed servers, through Kubernetes and containerized model of application.	Docker, Kubernetes, Container registry, IBM DB2, Cloud Object storage.
5.	Performance	Performance of the application optimized through IBM services where the app is deployed so that there is no lag or hinderance to user using the application.	IBM Container registry