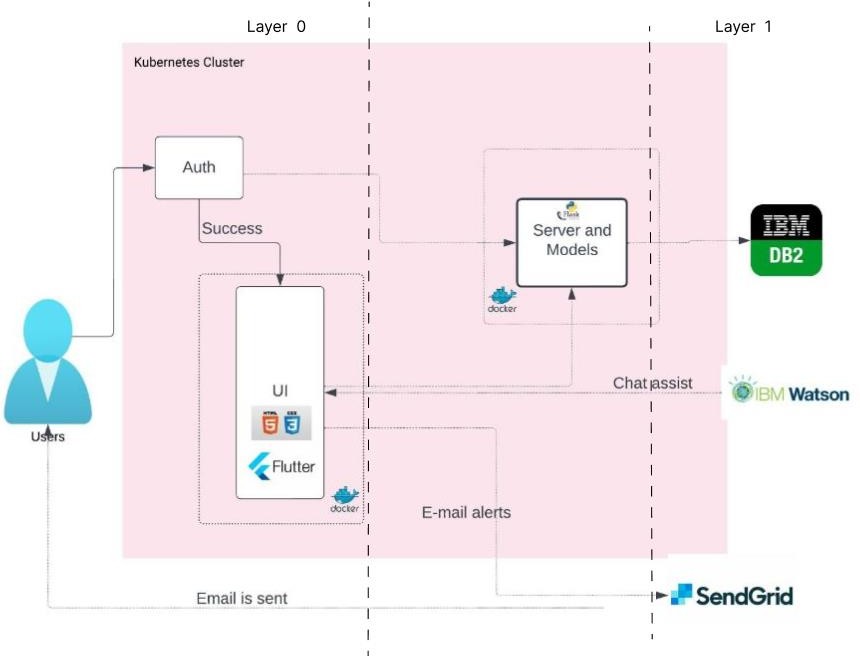
PROJECT DESIGN PHASE-II TECHNOLOGY STACK (ARCHITECTURE & STACK)

|  |  |
| --- | --- |
| Date | 03 November 2022 |
| Team ID | PNT2022TMID03619 |
| Project Name | Project – Personal Expense Tracker Application |
| Maximum Marks | 4 Marks |

# *Technical Architecture:*

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2



# *Table-1 : Components & Technologies:*

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | The user can interface with the application. E.g., web UI, mobile App, chart bot. | Flutter, HTML, CSS, JavaScript. |
| 2. | User Login | Application contains the Sign In/Sign Up page.  Where the user can Login and register | Flutter and Flask (Python) |
| 3. | Application Logic-1 | Users can create a budget | Flutter |
| 4. | Application Logic-2 | User can check the report of the budget | Flask |
| 5. | Application Logic-3 | Calculate balance and alert | Flask, SendGrid |
| 6. | Database | Type of database used | MySQL |
| 7. | Cloud Database | Database Service on Cloud | IBM DB2 |
| 8. | File Storage | File storage on Cloud | IBM Block Storage |
| 9. | Infrastructure (Server / Cloud) | Application Deployment on Local System / Cloud Local Server Configuration: localhost:8080  Cloud Server Configuration: | Local, Cloud, docker. |

***Table-2: Application Characteristics:***

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | List of open-source frameworks used | Flask, Docker, Kubernetes, Flutter |
| 2. | Security Implementations | Unique Identity key in generated and validated | Python – Flask and Flutter |
| 3. | Scalable Architecture | We use automatic deployment and load balancer | Docker and Flask Queue |
| 4. | Availability | This can be accessed by Cross platform application from all part of the world | Flutter, IBM Cloud |
| 5. | Performance | Designed in particular way to handle 1000s  request per second | Docker and IBM cloud Kubernetes. |