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

Date	15 October 2022
Team ID	PNT2022TMID48705
Project Name	Estimation of crop yield and data analytics
Maximum Marks	4 Marks

Technical Architecture:

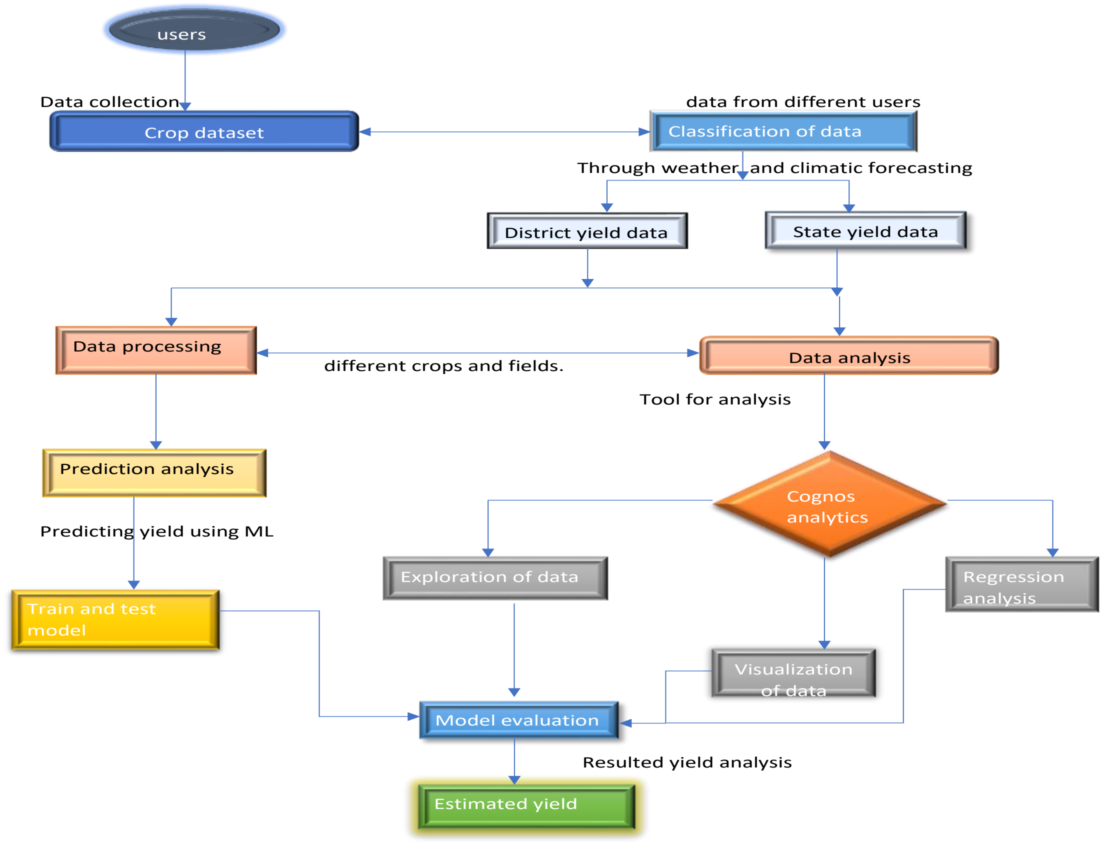

Table-1: Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The user interacts with the application through the web UI.	HTML, CSS, JavaScript, React Js
2.	Application Logic-1	User's account registration, user's login & user's input entry tab is made using this web framework	Python (Flask /Django)
3.	Application Logic-2	Analysis and prediction input data is done using python SciKit Learn	Python
4.	Application Logic-3	The base data set is pre-processed & trained using python Scikit Learn	Python
5.	Application logic-4	The visualization in dashboard is analysed & shown by using python analytics tool (Matplotlib/Seaborn)	Python
6.	Application logic-5	The Estimation of cropyield is done using python	Python
7.	Database	Contains information of registered users, user's health related data	PostgreSQL/MongoDB
8.	Cloud Database	Database Service on Cloud	IBM DB2
9.	File Storage	File storage requirements	Local Filesystem
10.	External API-1	one clicks signup feature can be implemented using API's	Google API and Facebook API.

11.	Machine Learning Model	The main objective of The Estimation of cropyield system is to discover and extract hidden knowledge associated with parameters of historical crop yield data set and to identify theinefficiency, trend spots, weather ,soli condition ,production respective to profit. The Estimation of cropyield system aims to exploit data mining techniques.	Predictive and Classification model.
12.	Infrastructure (Server / Cloud)	Application Deployment on Cloud Cloud Server Configuration: Kubernetes	Kubernetes

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask/Django	Flask-Python microservice web frame work for web development / Django-Python-based web framework that follows the model–template–views architectural pattern
2.	Security Implementations	Security / access controls implemented, use of firewalls for web app security	Encryptions, WAF, OWASP.
3.	Scalable Architecture	Scalability consists of 3-tiers	Client -HTML, CSS, JavaScript, Server-Python Database- IBM Cloud
4.	Availability	The application is available for all the users and largely scalable as it is deployed in cloud. Kubernetes is a Load balancer	Kubernetes
5.	Performance	As the application is developed with Flask/Django, it can handle large number of requests so our application will be available to large number of users and process many numbers of requests	Flask/Django