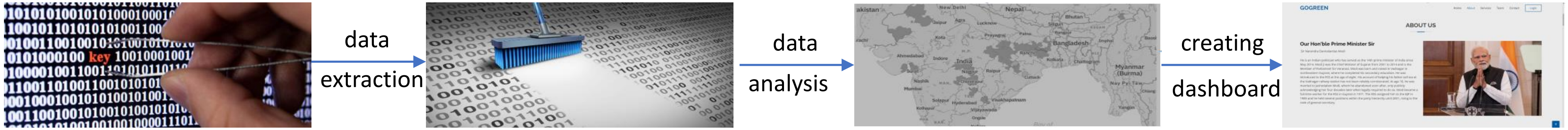
GoGreen

Chandradeo Prasad

(Team Leader) , Front-end Programmer

Cse2024chandradeo43@iesbpl.ac.in

ROADMAP:



data extraction

data analysis

creating dashboard

Prototyping Strategies:

- Data Extraction and Cleaning.
- Development of Analytical Models.
- Socio-economic Analysis Models.
- Data Handling and Quality Assurance.
- Accuracy models .

Our Dashboard include several features:-

i) **Barren Land Information:** Providing data on lands post-coal mining completion, helping stakeholders understand the transformation and potential for future use.

ii) **Coal Mine Details:** Offering a detailed breakdown of areas where coal mining has been completed, aiding in land-use planning and resource allocation.

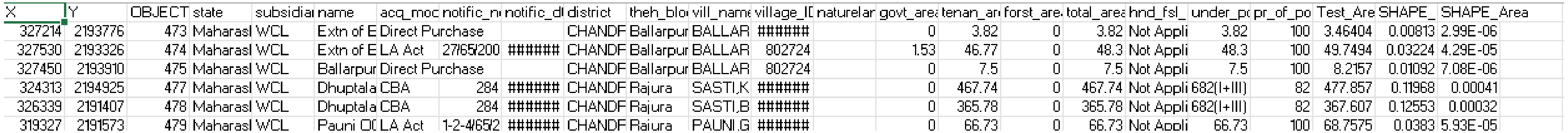
iii) **Geospatial Imaging:** Regular geospatial imagery captured every six months provides a dynamic view of changing land conditions, facilitating informed decision-making.

iv) **Soil Cover Percentage Analysis:** Utilizing advanced techniques, we assess soil cover percentages, aiding in identifying lands suitable for agriculture beds based on predefined criteria.

v) Illicit Land Occupation Analysis: Investigation into the nature and extent of unauthorized land use on Coal India's properties.

vi) **Dynamic Hazard Assessment Dashboard**:-Development of a cutting-edge dashboard to assess the safety risks associated with barren lands, ensuring timely intervention.

Prototype Journey: Bringing Ideas to Life

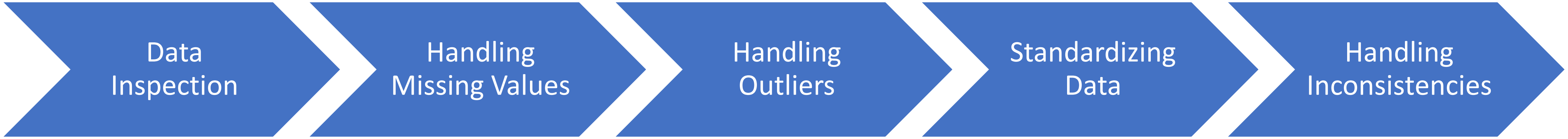


X	Y	OBJECT	state	subsidiai	name	acq_moc	notific_n	notific_dl	district	theh_blo	vill_name	village_ID	naturelar	govt_are	tenan_ar	forst_are	total_are	hnd_fsl_	under_pc	pr_of_po	Test_Are	SHAPE_	SHAPE_Area
327214	2193776	473	Maharasl	WCL	Extn of E	Direct Purchase			CHANDF	Ballarpur	BALLAR	######		0	3.82	0	3.82	Not Appli	3.82	100	3.46404	0.00813	2.99E-06
327530	2193326	474	Maharasl	WCL	Extn of E	LA Act	27/65/200	######	CHANDF	Ballarpur	BALLAR	802724		1.53	46.77	0	48.3	Not Appli	48.3	100	49.7494	0.03224	4.29E-05
327450	2193910	475	Maharasl	WCL	Ballarpur	Direct Purchase			CHANDF	Ballarpur	BALLAR	802724		0	7.5	0	7.5	Not Appli	7.5	100	8.2157	0.01092	7.08E-06
324313	2194925	477	Maharasl	WCL	Dhuptala	CBA	284	######	CHANDF	Rajura	SASTI,K	######		0	467.74	0	467.74	Not Appli	682(I+III)	82	477.857	0.11968	0.00041
326339	2191407	478	Maharasl	WCL	Dhuptala	CBA	284	######	CHANDF	Rajura	SASTI,B	######		0	365.78	0	365.78	Not Appli	682(I+III)	82	367.607	0.12553	0.00032
319327	2191573	479	Maharasl	WCL	Pauni OC	LA Act	1-2-4/65/2	######	CHANDF	Rajura	PAUNI,G	######		0	66.73	0	66.73	Not Appli	66.73	100	68.7575	0.0383	5.93E-05

- The above data given by the respected evaluators, is a Govt. Land Information System(GLIS) Data.
- The data works on the following parameters:

Name of state, subsidiaries in state, accusation mode, district name, block name, village name and id, government land, tenant area along with the shape-values of the length and area.

Data Cleaning:



Data Inspection

Handling Missing Values

Handling Outliers

Standardizing Data

Handling Inconsistencies

1.Understand Data Structure:
Familiarize with GLIS geospatial data types and structure.

2.Data Inspection:
Identify issues like missing values, outliers, and inconsistencies.

3.Coordinate System Consistency:
Ensure consistent use of coordinate systems across the dataset.

4.Standardize Data Formats:
Standardize date formats, land use codes, and categorical variables.

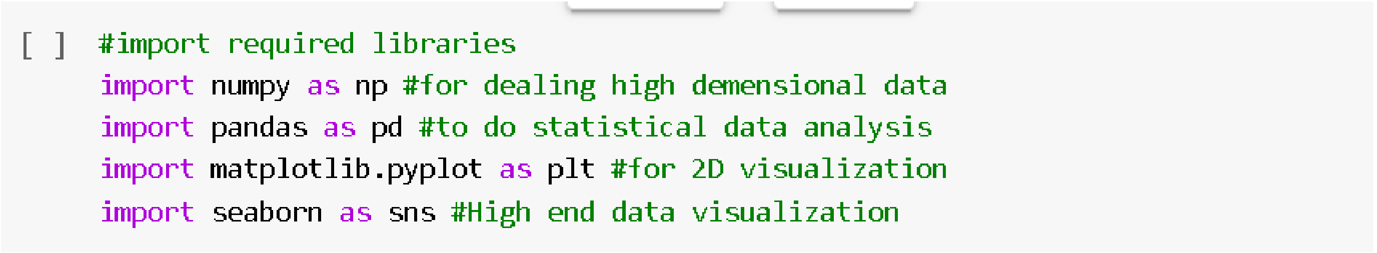
5.Topological Checks:
Perform checks to ensure spatial relationships and integrity.

6.Documentation and Iterative Cleaning:
Document cleaning steps and decisions, and iterate as needed.

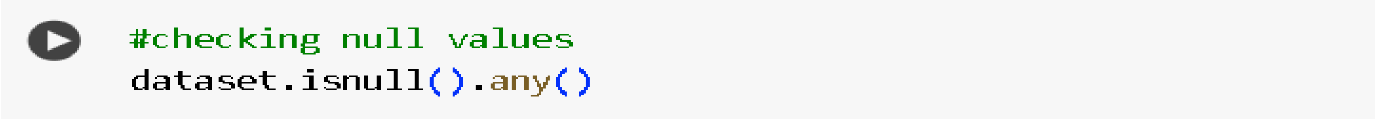
7.Quality Assurance:
Conduct quality assurance to ensure data accuracy and reliability.

Machine Learning Model:

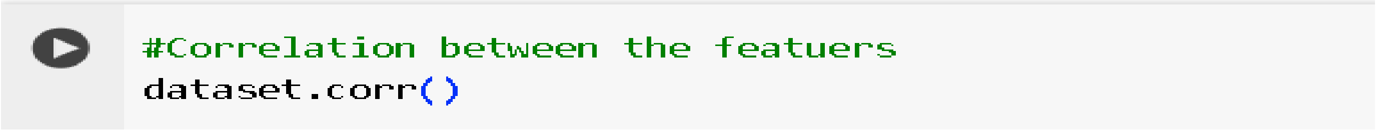
1. **Data Collection and Exploration:**
   - Gather historical data and explore its characteristics, addressing missing values and outliers.
2. **Feature Selection and Preprocessing:**
   - Select relevant features and preprocess the data, handling scaling, encoding, and other quality issues.
3. **Data Splitting:**
   - Split the data into a training set (e.g., 80%) and a testing set (e.g., 20%).
4. **Model Selection and Training:**
   - Choose a suitable machine learning algorithm and train the model on the training set.
5. **Model Evaluation on Training Set:**
   - Assess the model's performance on the training set using appropriate metrics.
6. **Model Testing and Evaluation:**
   - Test the model on the unseen testing set to evaluate its generalization performance.
7. **Hyperparameter Tuning and Deployment:**
   - Fine-tune model hyperparameters, deploy the final model, and consider ongoing monitoring and maintenance.



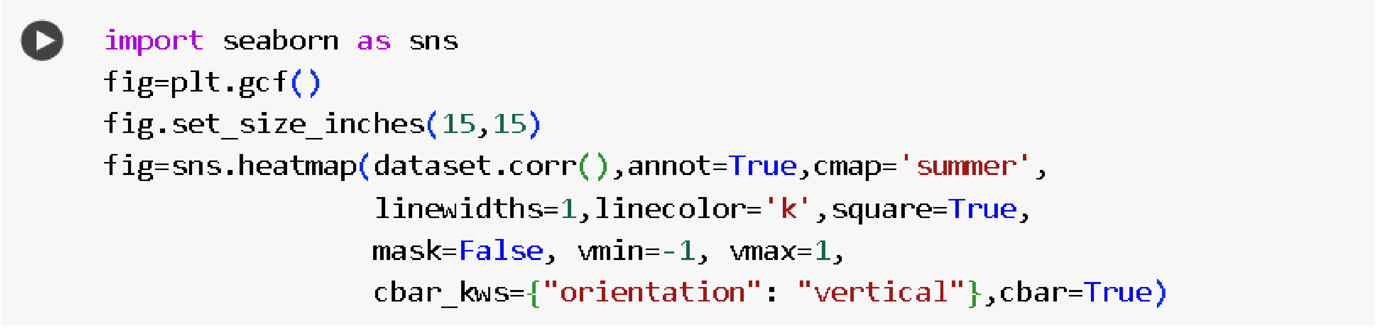
```
[ ] #import required libraries
    import numpy as np #for dealing high demensional data
    import pandas as pd #to do statistical data analysis
    import matplotlib.pyplot as plt #for 2D visualization
    import seaborn as sns #High end data visualization
```



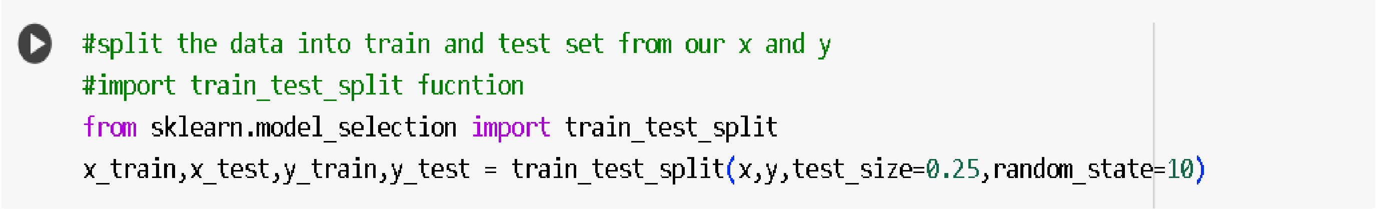
```
#checking null values
dataset.isnull().any()
```



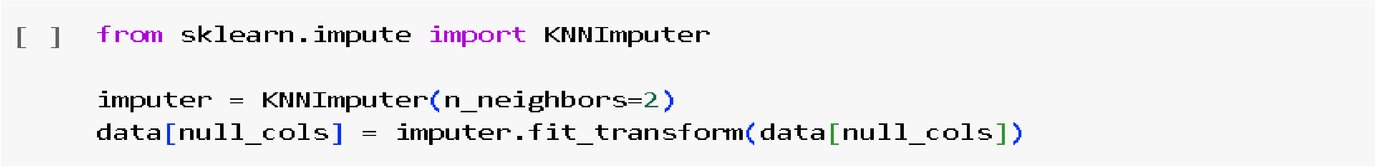
```
#Correlation between the featuers
dataset.corr()
```



```
import seaborn as sns
fig=plt.gcf()
fig.set_size_inches(15,15)
fig=sns.heatmap(dataset.corr(),annot=True,cmap='summer',
                linewidths=1,linecolor='k',square=True,
                mask=False, vmin=-1, vmax=1,
                cbar_kws={"orientation": "vertical"},cbar=True)
```

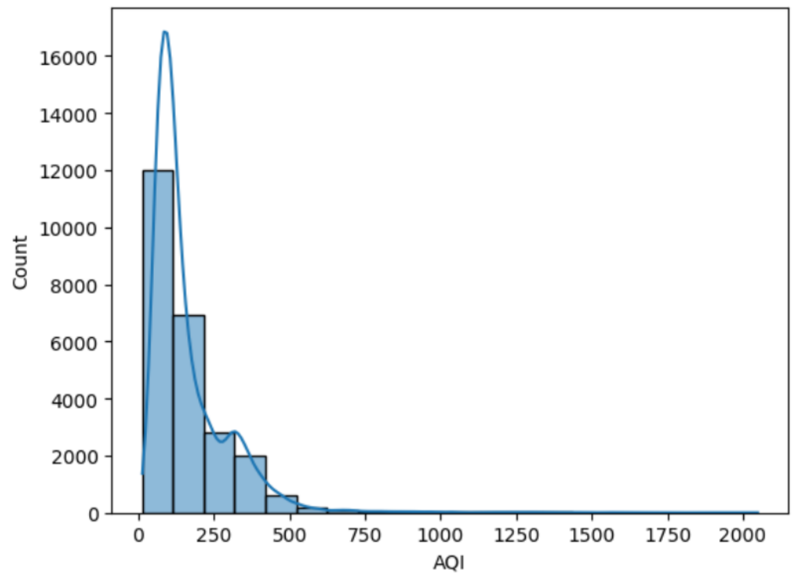


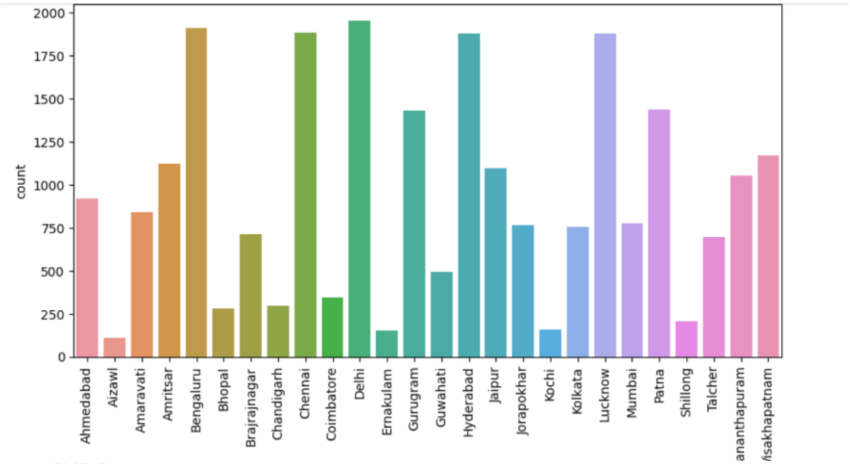
```
#split the data into train and test set from our x and y
#import train_test_split fucntion
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=10)
```

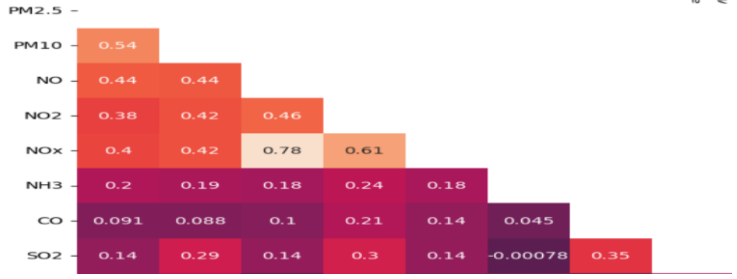


```
[ ] from sklearn.impute import KNNImputer

    imputer = KNNImputer(n_neighbors=2)
    data[null_cols] = imputer.fit_transform(data[null_cols])
```

```
<Axes: xlabel='AQI', ylabel='Count'>
```

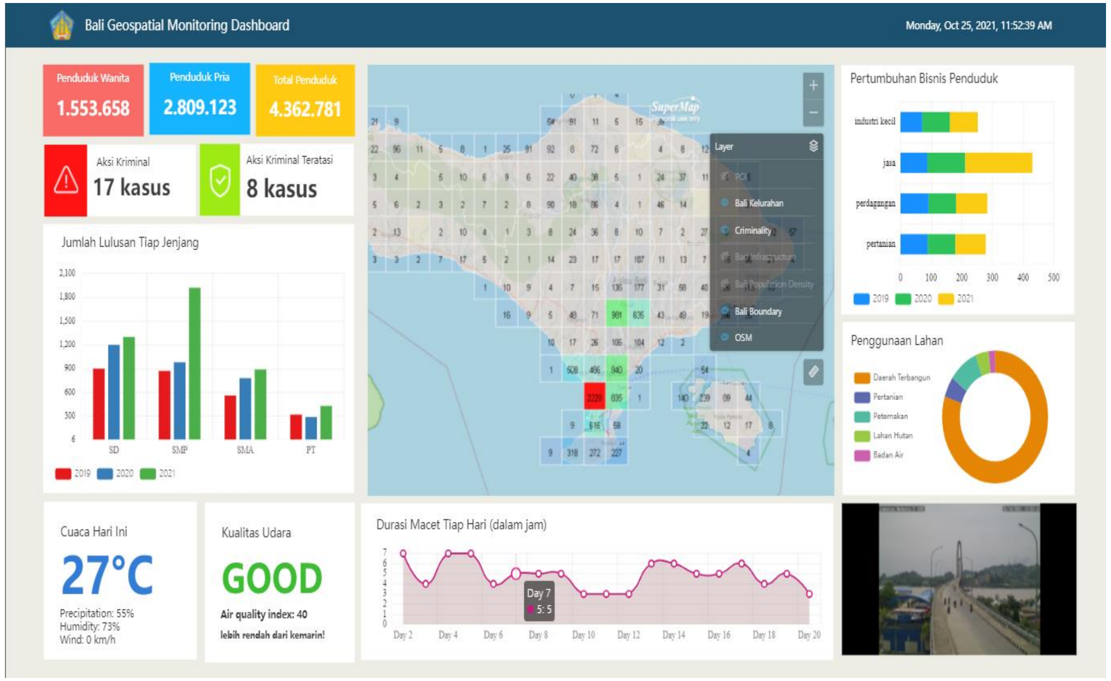




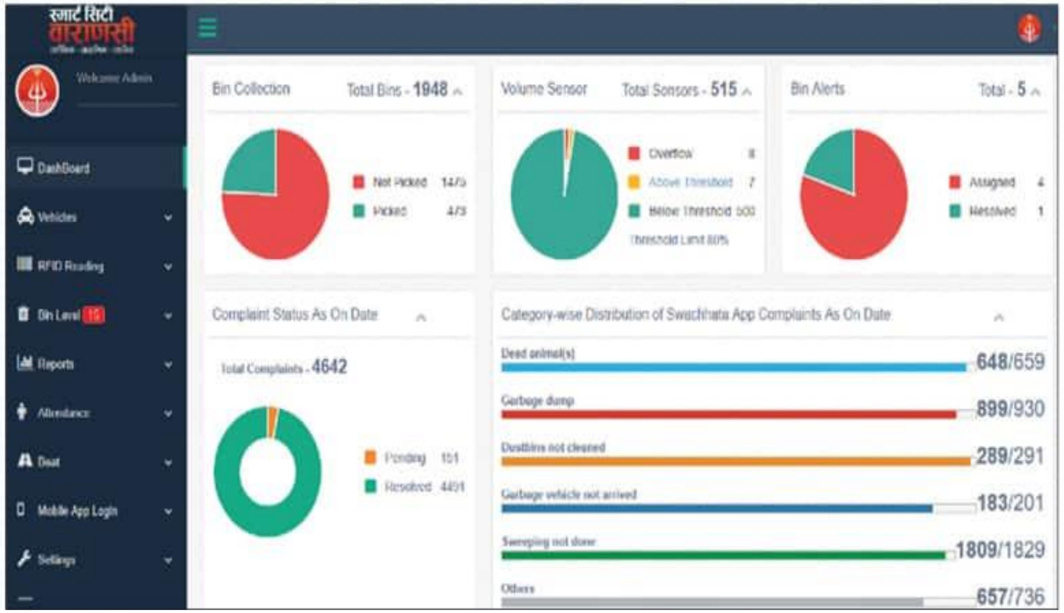


SMART URBAN PLANNIG DASHBOARD

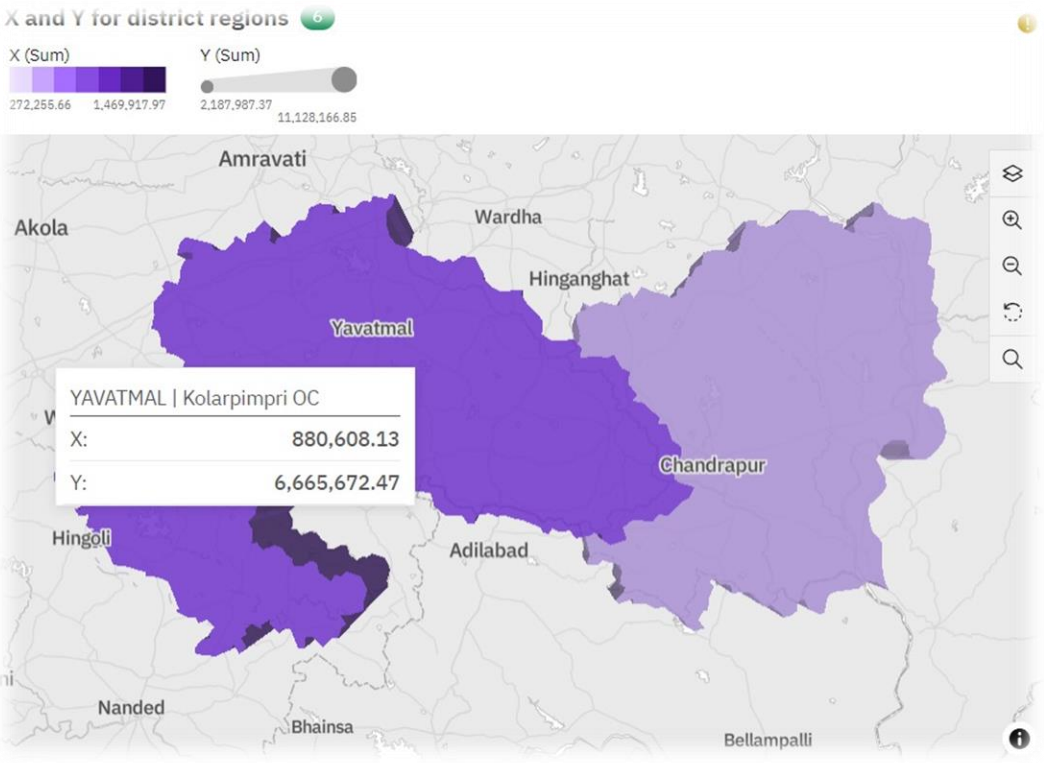
We are Creating a Smart Urban Planning Dashboard using GLIS Data which become a super tool for evidence based decision making.



DASHBOARD – Monitoring inside a city (Varanasi)



Dashboard Designing



X and Y for district regions 6

X (Sum)

272,255.66 1,469,917.97

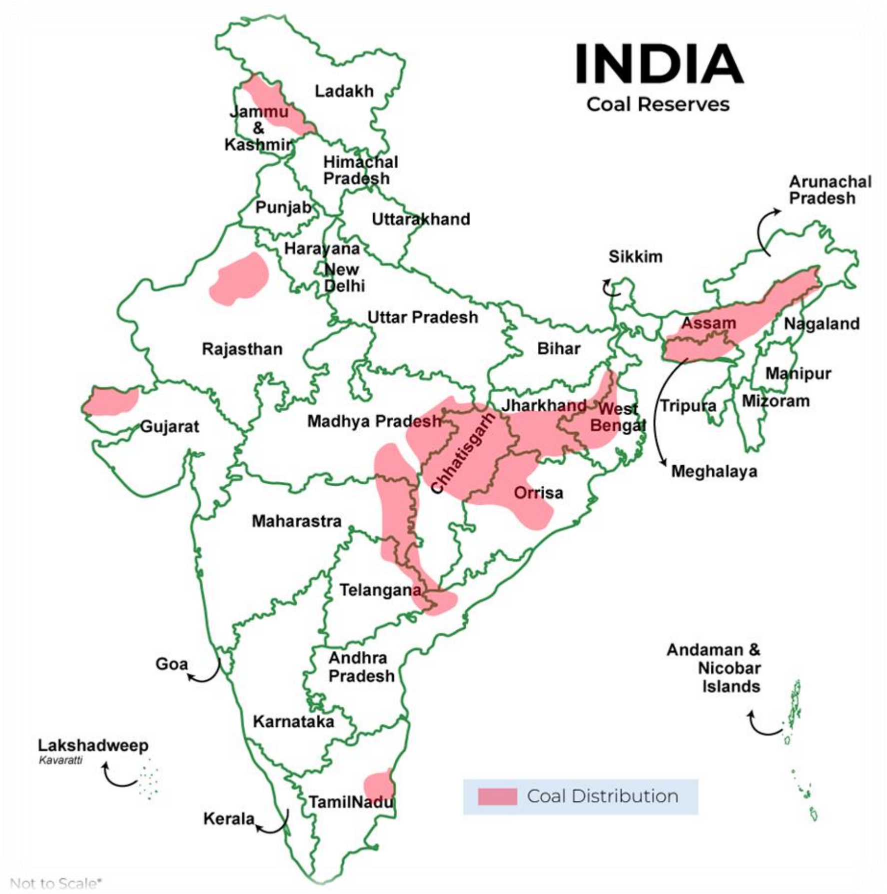
Y (Sum)

2,187,987.37

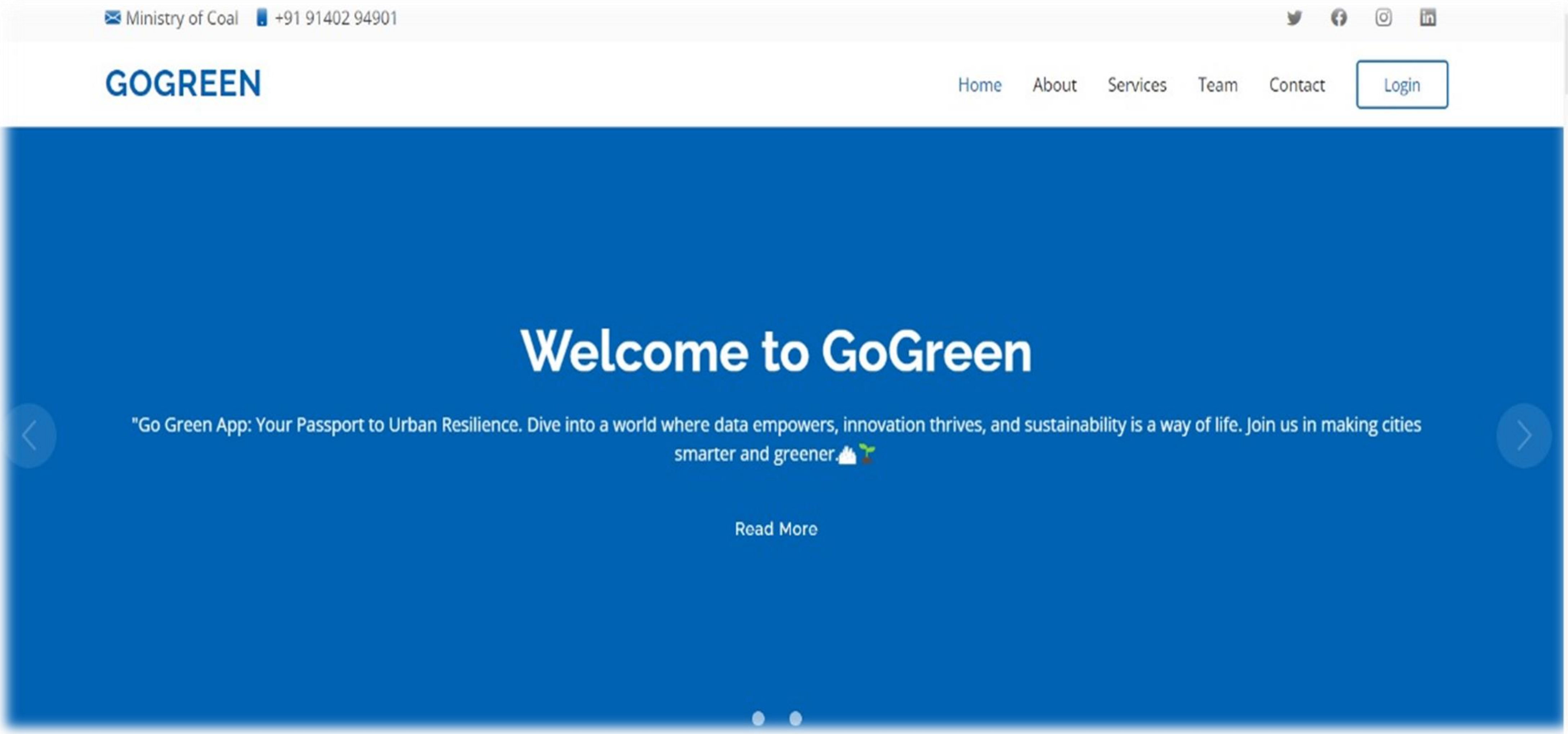
11,128,166.85

YAVATMAL | Kolarpimpri OC

X:	880,608.13
Y:	6,665,672.47



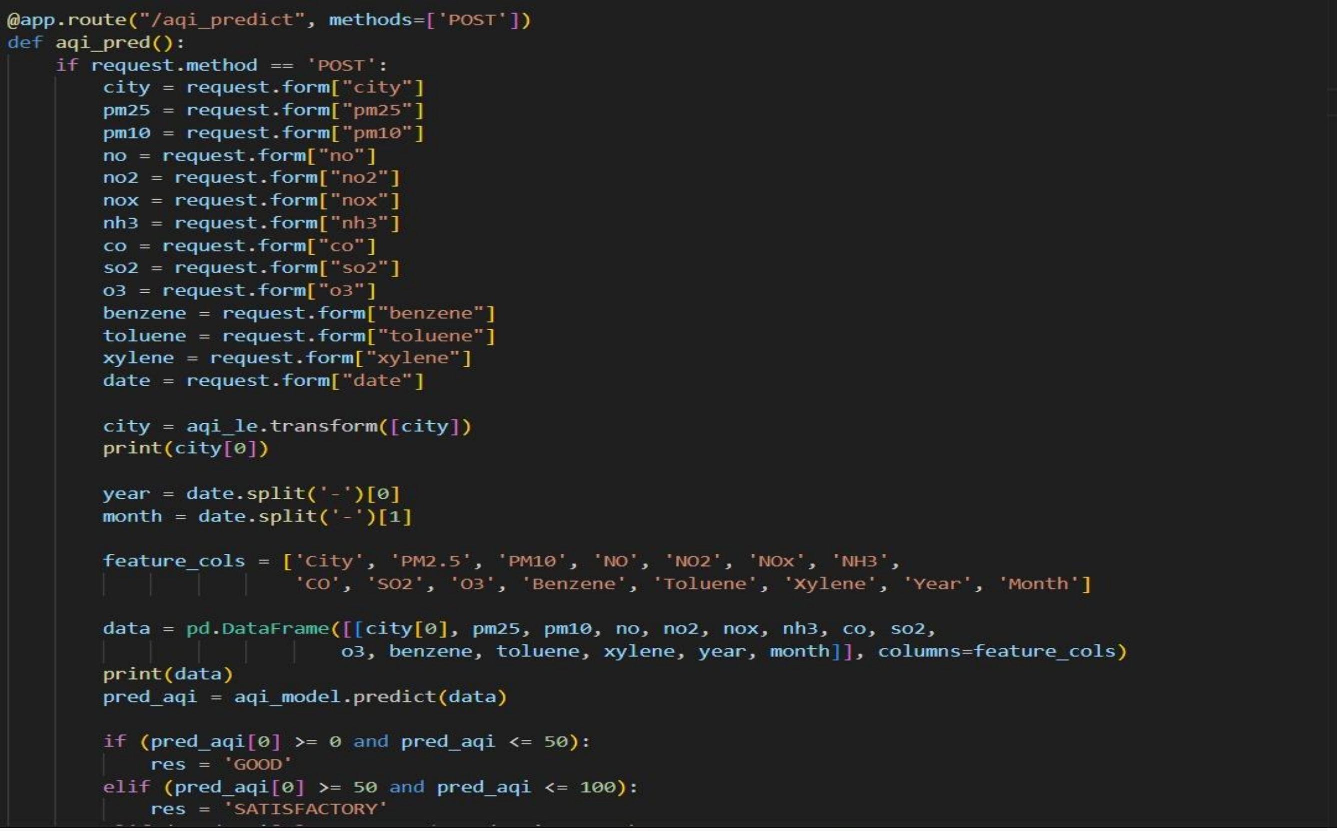
Front-end Side



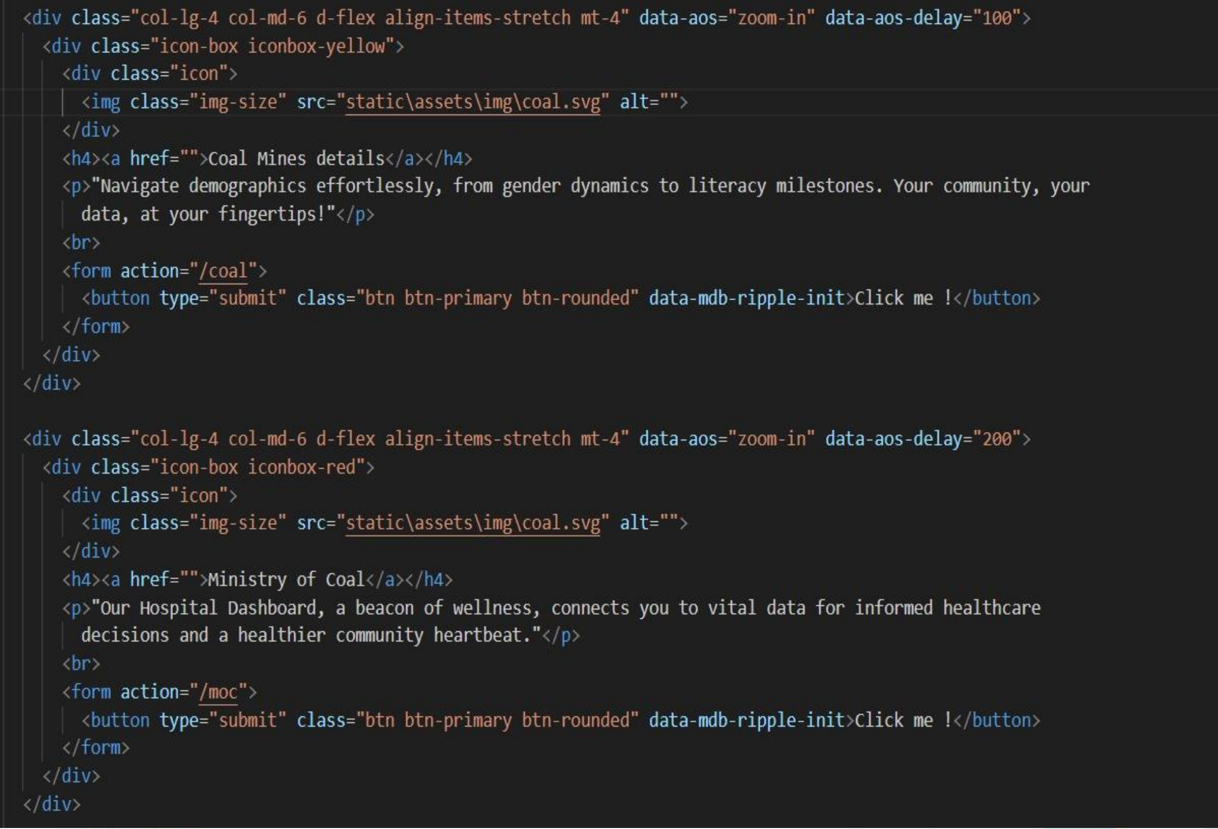
Welcome to GoGreen

"Go Green App: Your Passport to Urban Resilience. Dive into a world where data empowers, innovation thrives, and sustainability is a way of life. Join us in making cities smarter and greener.🏙️🌱

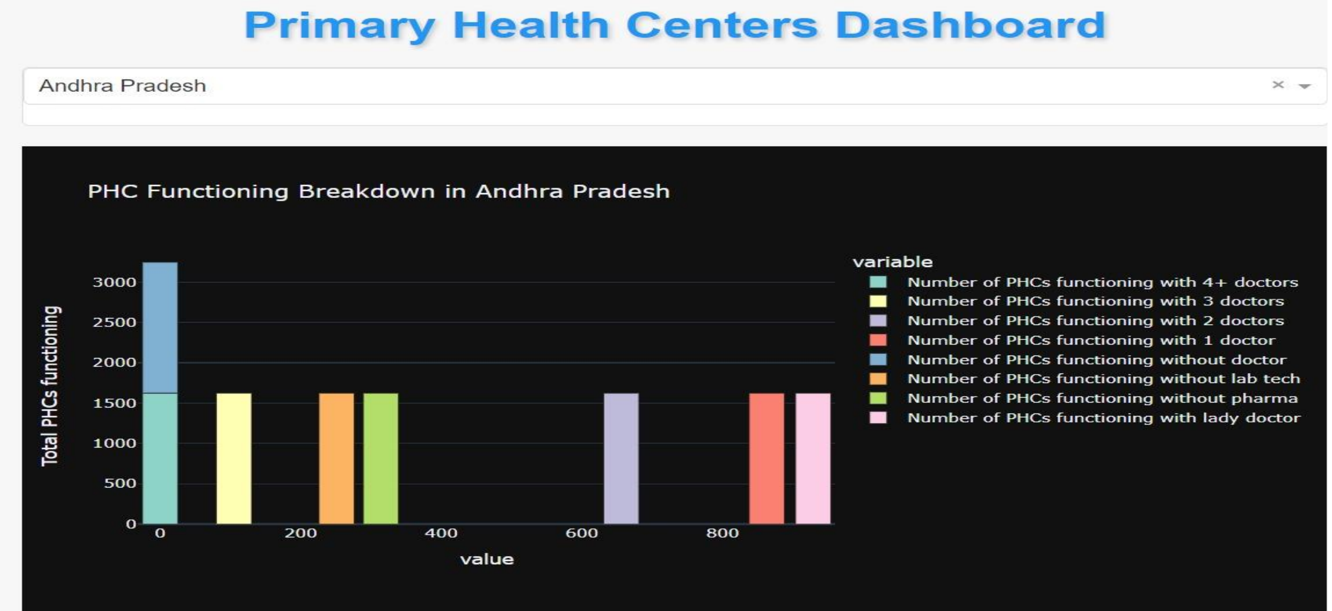
Integration



```python
@app.route("/aqi_predict", methods=['POST'])
def aqi_pred():
    if request.method == 'POST':
        city = request.form["city"]
        pm25 = request.form["pm25"]
        pm10 = request.form["pm10"]
        no = request.form["no"]
        no2 = request.form["no2"]
        nox = request.form["nox"]
        nh3 = request.form["nh3"]
        co = request.form["co"]
        so2 = request.form["so2"]
        o3 = request.form["o3"]
        benzene = request.form["benzene"]
        toluene = request.form["toluene"]
        xylene = request.form["xylene"]
        date = request.form["date"]

        city = aqi_le.transform([city])
        print(city[0])

        year = date.split('-')[0]
        month = date.split('-')[1]

        feature_cols = ['City', 'PM2.5', 'PM10', 'NO', 'NO2', 'NOx', 'NH3',
                        'CO', 'SO2', 'O3', 'Benzene', 'Toluene', 'Xylene', 'Year', 'Month']

        data = pd.DataFrame([[city[0], pm25, pm10, no, no2, nox, nh3, co, so2,
                              o3, benzene, toluene, xylene, year, month]], columns=feature_cols)
        print(data)
        pred_aqi = aqi_model.predict(data)

        if (pred_aqi[0] >= 0 and pred_aqi <= 50):
            res = 'GOOD'
        elif (pred_aqi[0] >= 50 and pred_aqi <= 100):
            res = 'SATISFACTORY'
```



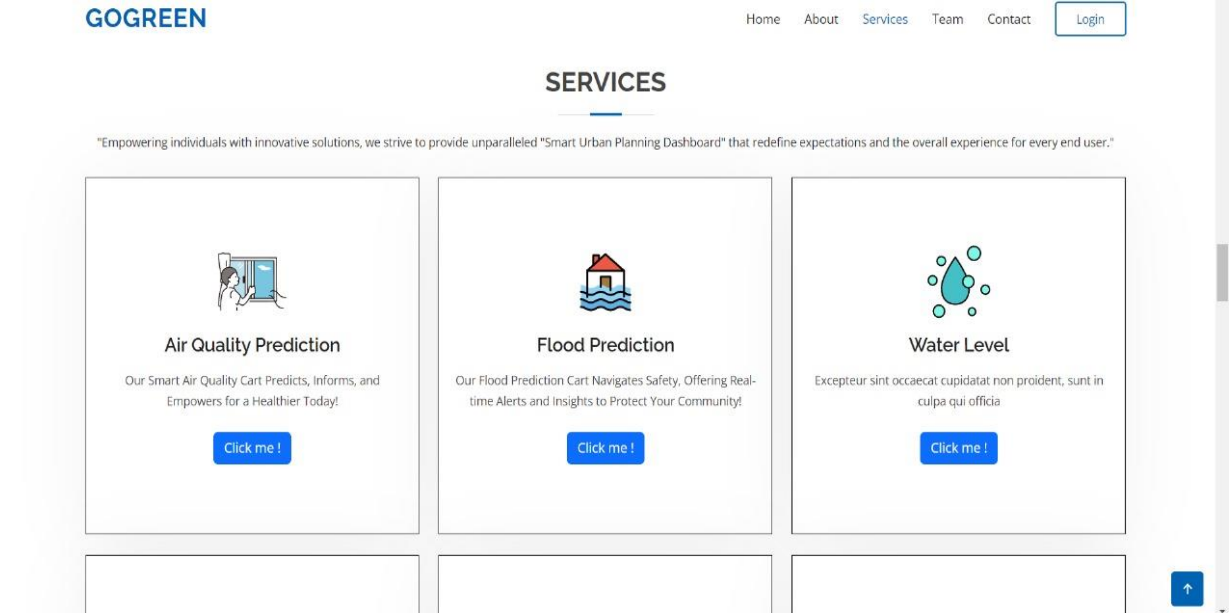
```html
<div class="col-lg-4 col-md-6 d-flex align-items-stretch mt-4" data-aos="zoom-in" data-aos-delay="100">
  <div class="icon-box iconbox-yellow">
    <div class="icon">
      <img class="img-size" src="static\assets\img\coal.svg" alt="">
    </div>
    <h4><a href="">Coal Mines details</a></h4>
    <p>"Navigate demographics effortlessly, from gender dynamics to literacy milestones. Your community, your
      data, at your fingertips!"</p>
    <br>
    <form action="/coal">
      <button type="submit" class="btn btn-primary btn-rounded" data-mdb-ripple-init>Click me !</button>
    </form>
  </div>
</div>

<div class="col-lg-4 col-md-6 d-flex align-items-stretch mt-4" data-aos="zoom-in" data-aos-delay="200">
  <div class="icon-box iconbox-red">
    <div class="icon">
      <img class="img-size" src="static\assets\img\coal.svg" alt="">
    </div>
    <h4><a href="">Ministry of Coal</a></h4>
    <p>"Our Hospital Dashboard, a beacon of wellness, connects you to vital data for informed healthcare
      decisions and a healthier community heartbeat."</p>
    <br>
    <form action="/moc">
      <button type="submit" class="btn btn-primary btn-rounded" data-mdb-ripple-init>Click me !</button>
    </form>
  </div>
</div>
```



Primary Health Centers Dashboard

Andhra Pradesh

PHC Functioning Breakdown in Andhra Pradesh



SERVICES

"Empowering individuals with innovative solutions, we strive to provide unparalleled "Smart Urban Planning Dashboard" that redefine expectations and the overall experience for every end user."

Air Quality Prediction

Our Smart Air Quality Cart Predicts, Informs, and Empowers for a Healthier Today!

Flood Prediction

Our Flood Prediction Cart Navigates Safety, Offering Real-time Alerts and Insights to Protect Your Community!

Water Level

Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia

COMPONENT COST

GOOGLE CLOUD COST :Rs. 16,250/year

IBM COGNOS:- Rs. 15,000/year

TOTAL:-Rs. 31,250/year

Implementation cost:-

- Domain Cost:-Rs. 1,502/year
- Server Infrastructure Development cost:-Rs. 15,000/year
- Development Cost : - Rs. 1 Lakh
- Total Implementation Cost:- Rs. 1,16,502

Total Application Cost:-

- IMPLEMENTATION COST + COMPONENT COST + = Rs. 1,16,502 + Rs. 31,250=Rs.1,47,752-Rs.1,50,000
- AMC(Annual Maintenance Charge) :- 20% OF TOTAL COST AFTER 1 YEAR FOR MAINTAINENCE.

Contract Based Application Cost –(yearly):-

- 40 % Of Implementation cost + server charge = Rs. 60,000 + Rs. 16,250=76,250

- NOTE:-IN YEARLY PURCHASE, WE PROVIDE UPGRADE VERSION OF OUR APPLICATION.

Revenue Model:

- Advertisements:

I. Free Basics Access Models: Offers basic features and limit access to GUIs analysis for free, to attract wide user base.

II. Premium Model: Introduce premium subscription plan with advance analysis of real-time update and additional functionalities as a subscription fee.

- Marketing and Outreach:

I. Develop a comprehensive marketing strategy to promote your platform. Utilize digital marketing, content creation, social media, and other channels to reach your target audience.

- Platform Charges

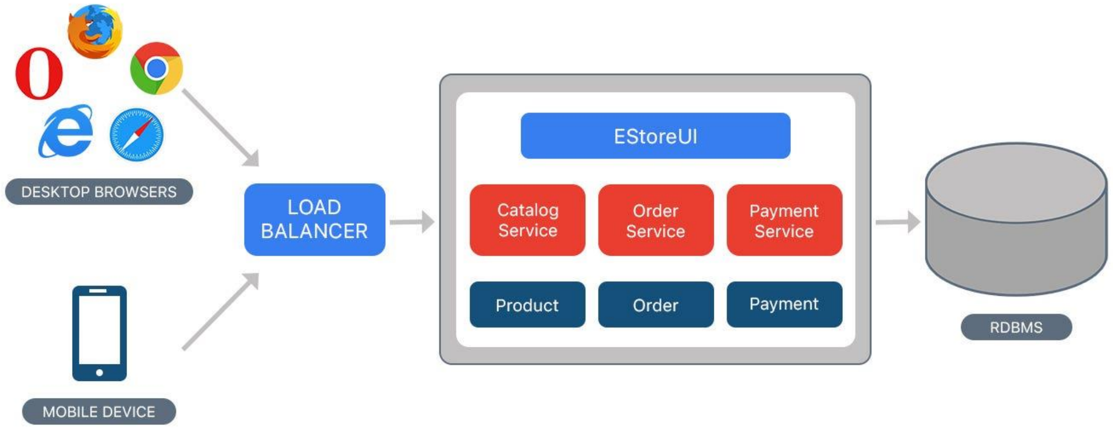
Monolith Vs Microservices

Monolithic Architecture:

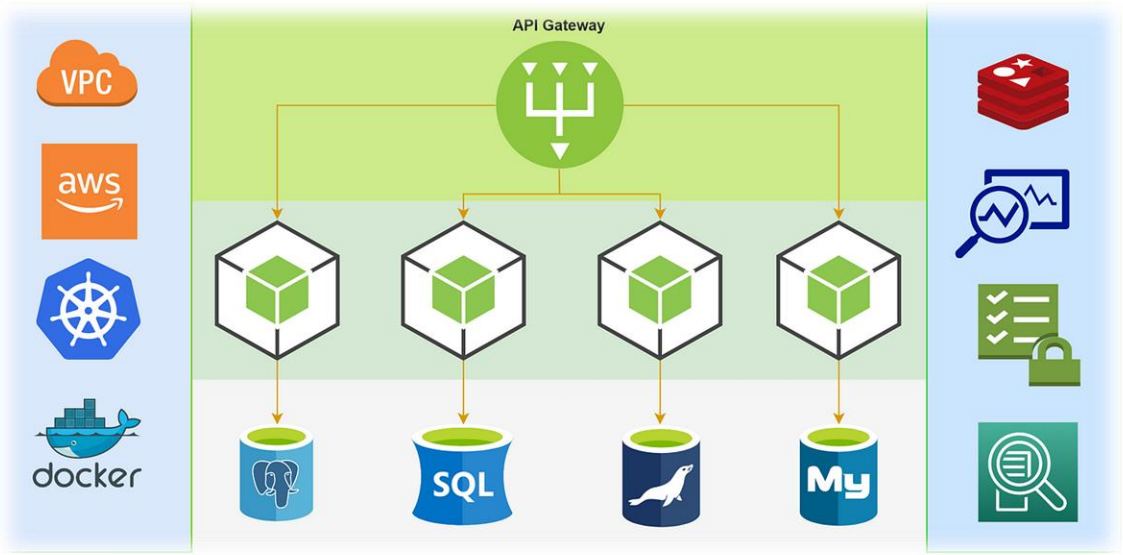
- A monolithic architecture is a traditional approach where an entire application is built as a single, tightly integrated unit.

- In a monolithic application, all components (such as user interface, business logic, and data access) are tightly coupled and run as a single process.

- Scaling a monolithic application typically involves replicating the entire application, which can be inefficient and resource-intensive.



Monolith Vs Microservices



Microservices Architecture:

- Microservices is an architectural style where an application is built as a collection of small, independent services that communicate with each other through well-defined APIs.
- Each microservice is a self-contained unit with its own specific business functionality and can be developed, deployed, and scaled independently of other services.
- Microservices promote flexibility, scalability, and maintainability by allowing different services to be developed and deployed independently.

Comparison

Scalability:

Monolithic applications scale by replicating the entire application, while microservices allow for independent scaling of individual services based on their specific needs.

Flexibility and Maintenance:

Microservices provide flexibility as each service can be developed, deployed, and maintained independently. Changes to one service do not affect others.

Monolithic applications may require more coordination when making changes, as all components are tightly coupled.

Fault Isolation:

In a monolithic architecture, a failure in one module can potentially bring down the entire application.

Microservices are designed for fault isolation. If one service fails, it doesn't necessarily impact the entire application.

Development Speed:

Microservices can allow for faster development cycles as smaller, independent teams can work on different services concurrently.

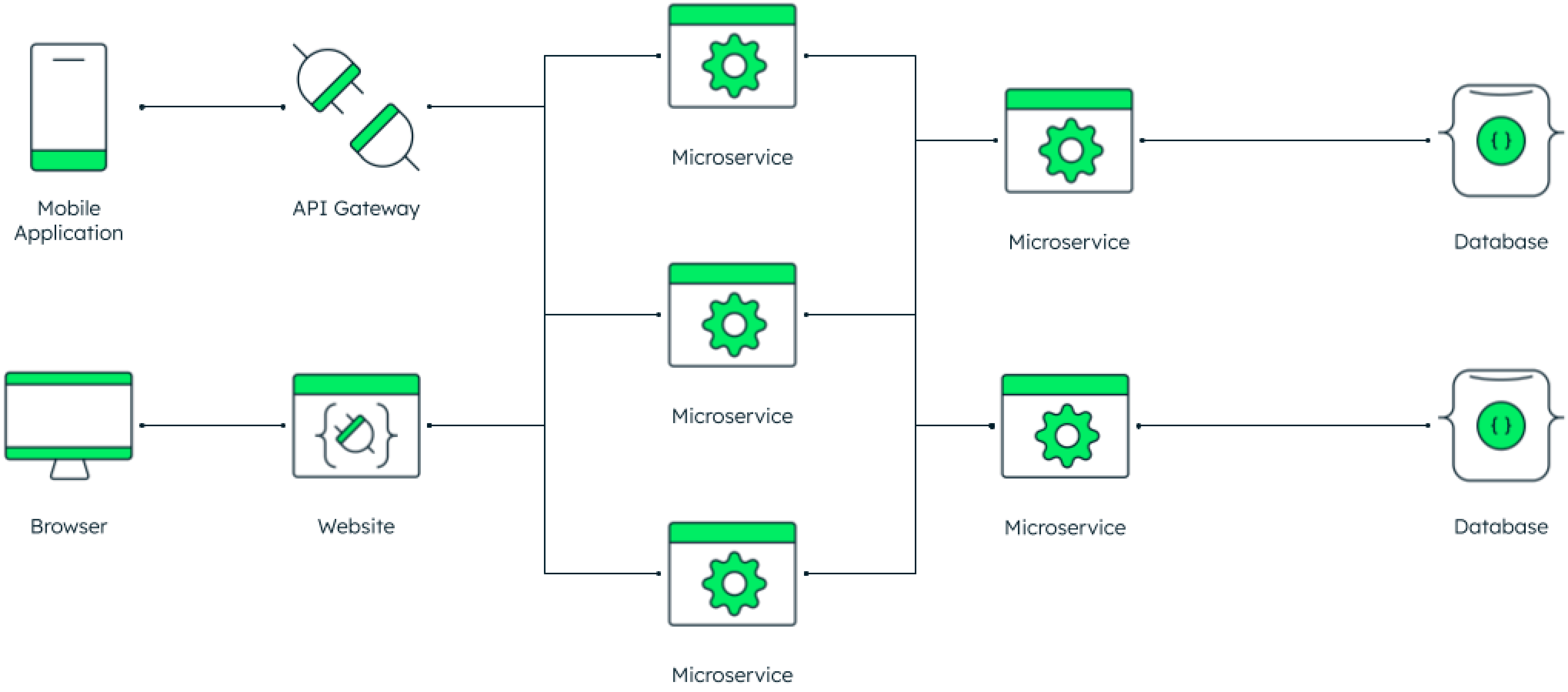
Monolithic applications might face challenges when multiple teams are working on the same codebase, leading to coordination and integration issues.

Complexity:

Microservices introduce distributed system complexities, such as service discovery, inter-service communication, and data consistency.

Monolithic applications are generally simpler to develop and deploy but may become complex and hard to maintain as they grow.

Micro-Services



TEAM DETAILS:

Team Leader Name: CHANDRADEO PRASAD		
Branch : B.tech	**Stream : CSE**	**Year : IV**
Team Member 1 Name: DIVYA KUMARI		
•Branch : B.tech	**Stream : CSE**	**Year : IV**
•Team Member 2 Name: ADITYA RAJ		
•Branch : B.tech	**Stream : CSE**	**Year : IV**
Team Member 3 Name: SANIA CHARPE		
Branch : B.tech	**Stream : CSE**	**Year : III**
Team Member 4 Name: SAANVI SHUKLA		
•Branch : B.tech	**Stream : CSE**	**Year : III**
•Team Member 5 Name: SHRUTI SRIVASTAVA		
•Branch : B.tech	**Stream : CS**	**Year : II**
•Team Mentor 1 Name: Prof. Vijay Dhote Sir		
Category (Academic)	**Expertise (ML)**	**Domain Experience (5 years)**
Team Mentor 2 Name: Mr. Kuldeep Kr Mishra		
Category (Industry):	**Expertise (ML)**	**Domain Experience (10 years)**