Sri Lanka Institute of Information Technology

KidniFy - A Mobile based Chronic Kidney Disease Patient Care System Using ML and IoT

2023-032

STATUS DOCUMENT - I

Student Name: Isurika W.B.M.A.

Student ID: IT20785192

Group Details

Supervisor: Ms. Wishalya Vanshanee Tissera

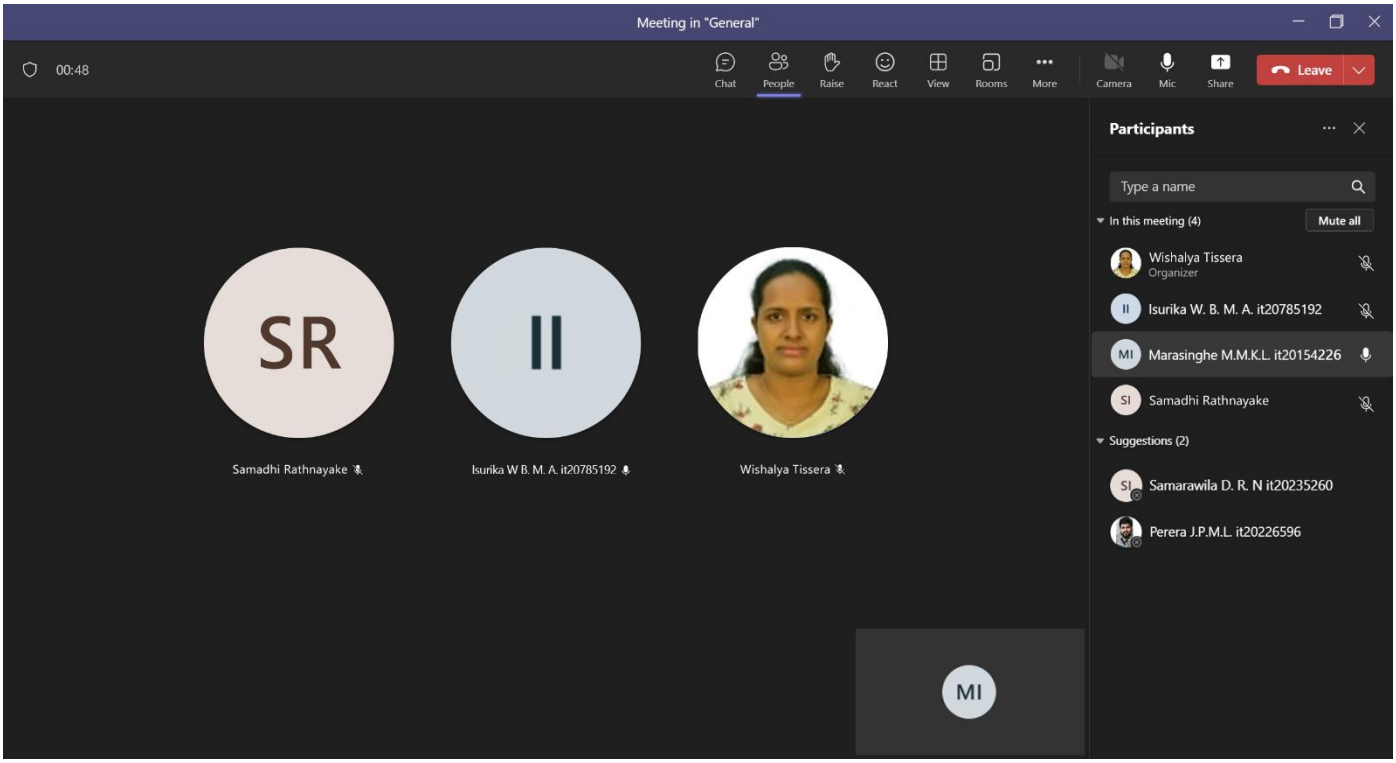
Co-Supervisor: Mr. Samadhi Chathuranga Rathnayake

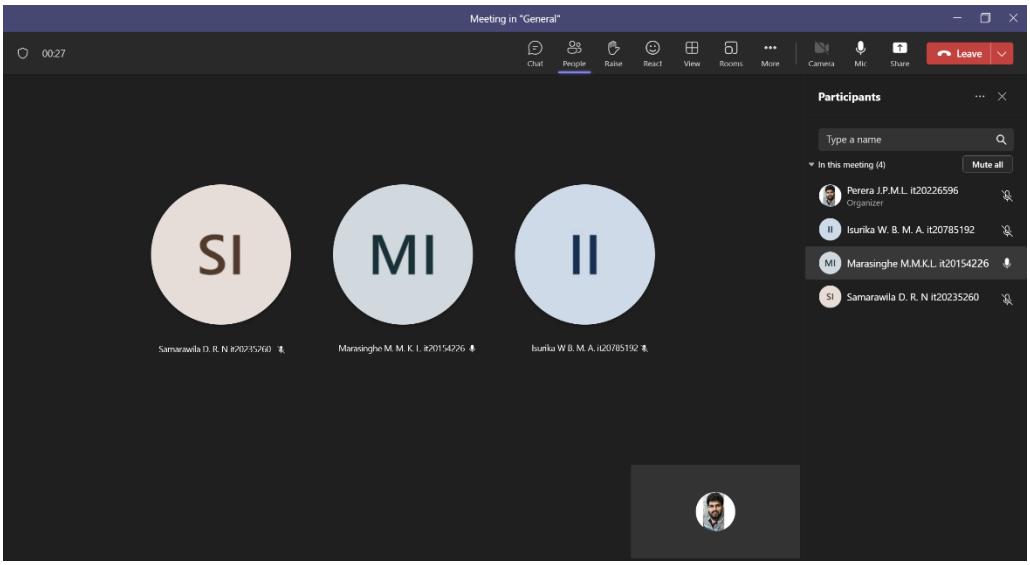
Student Name	Student ID	Contact No	Email Address
Marasinghe M.M.K.L.	IT20154226	0713037712	it20154226@my.sliit.lk
Isurika W.B.M.A.	IT20785192	0701484570	it20785192@my.sliit.lk
Perera J.P.M.L.	IT20226596	0776035479	it20226596@my.sliit.lk
Samarawila D.R.N.	IT20235260	0712421580	it20235260@my.sliit.lk

TABLE OF CONTENTS

1 Teams Meeting

1.1 Screenshots of Meetings & Calls





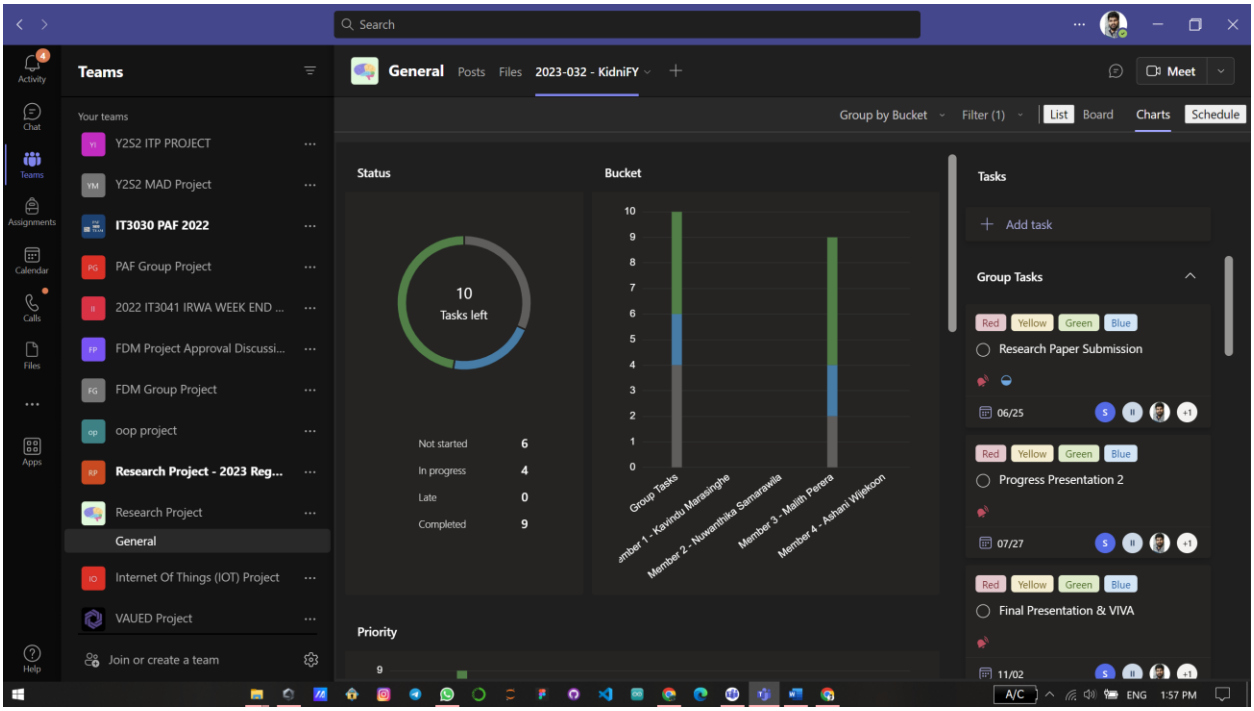
1.2 Meeting with the domain expert

Meeting with Consultant Nephrologist Dr. Pramil Rajakrishnan
At Kurunegala Teaching Hospital - Kidney dialysis Unit | 2023.04.29

Meet the Ward Master of Kidney Dialysis Unit
At Anuradhapura Teaching Hospital | 2023.05.01

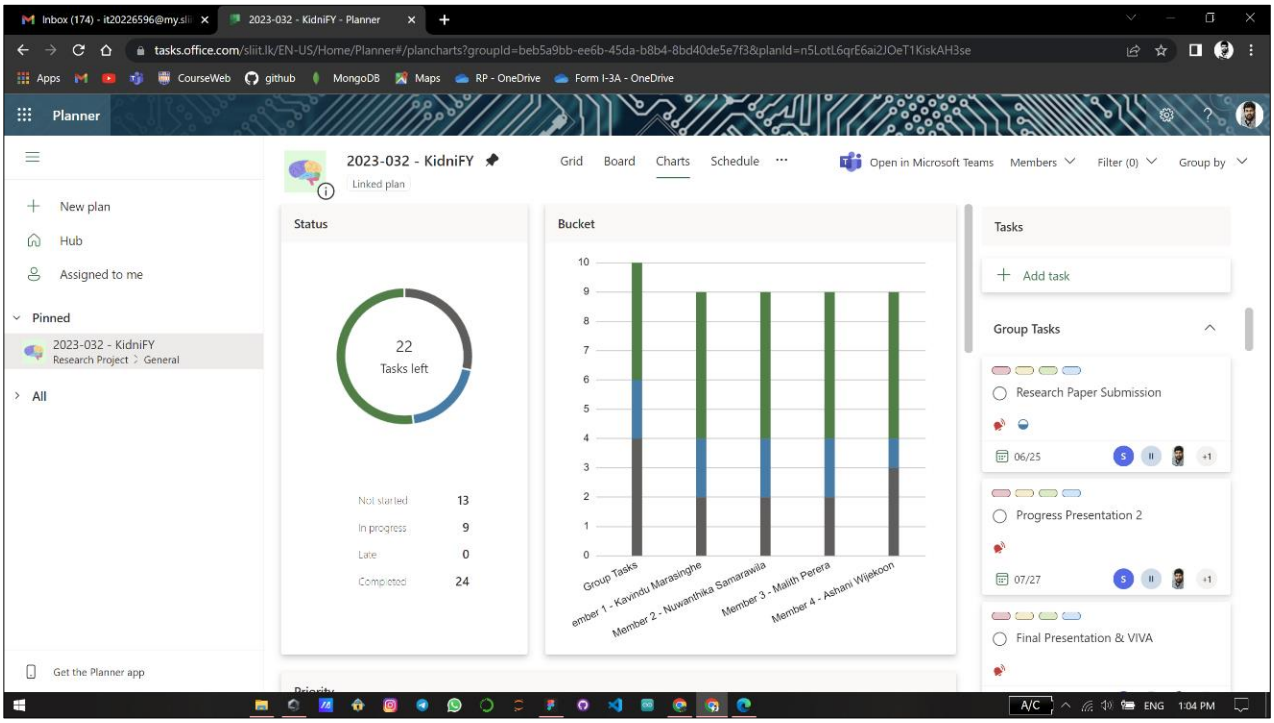
2 Screenshots of the Tasks by Planner

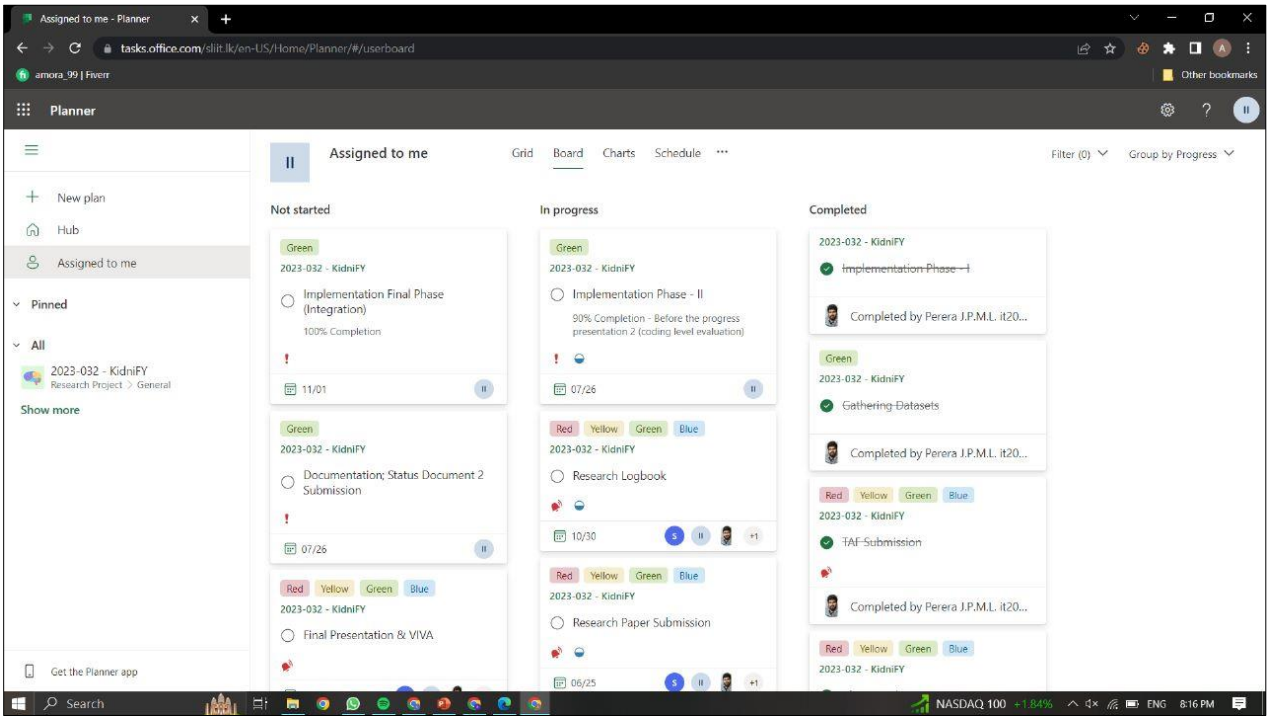
2.1 Chart Overview

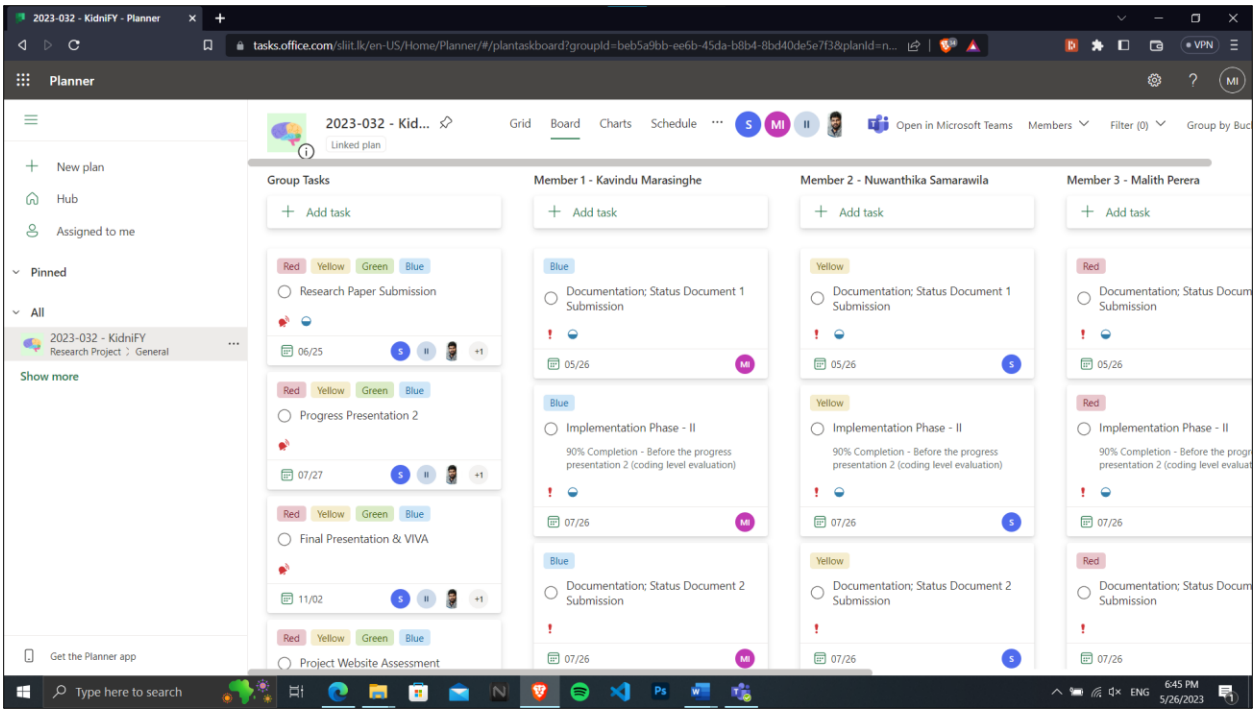


- To-do
- In Progress
- Completed

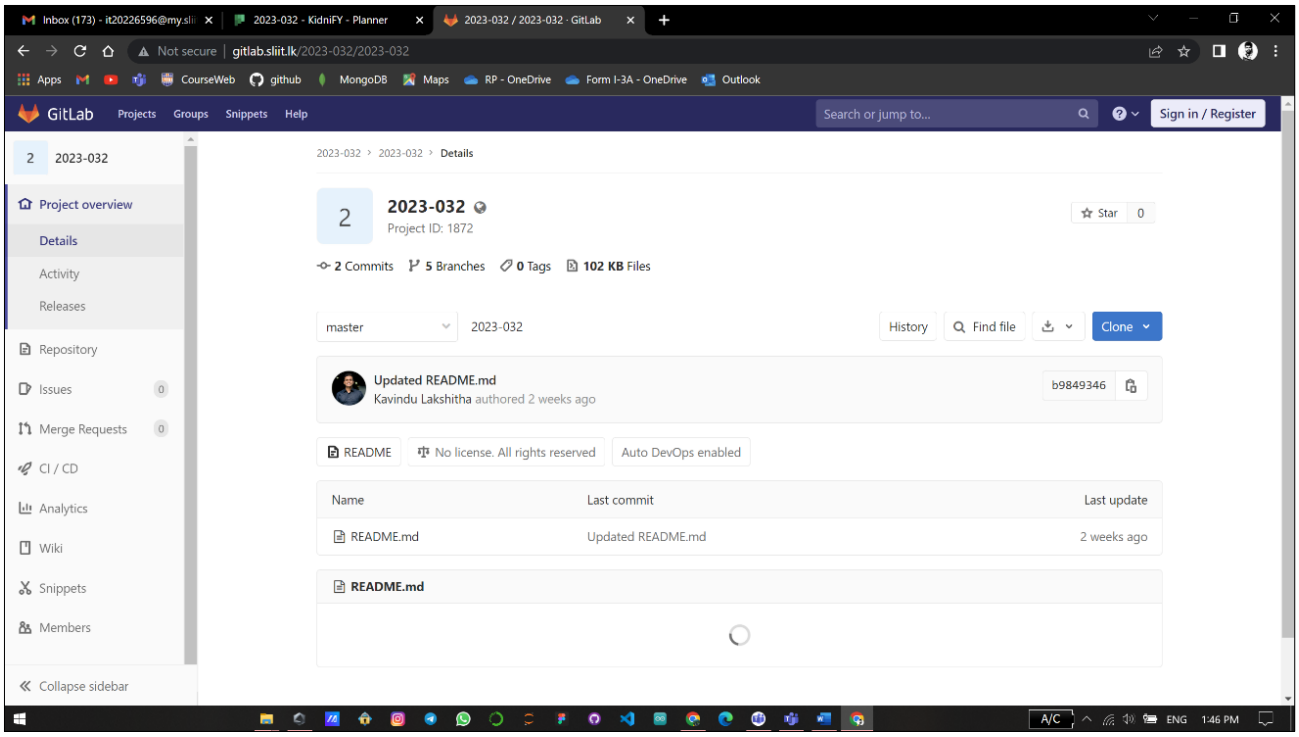
2.2 Bucket List

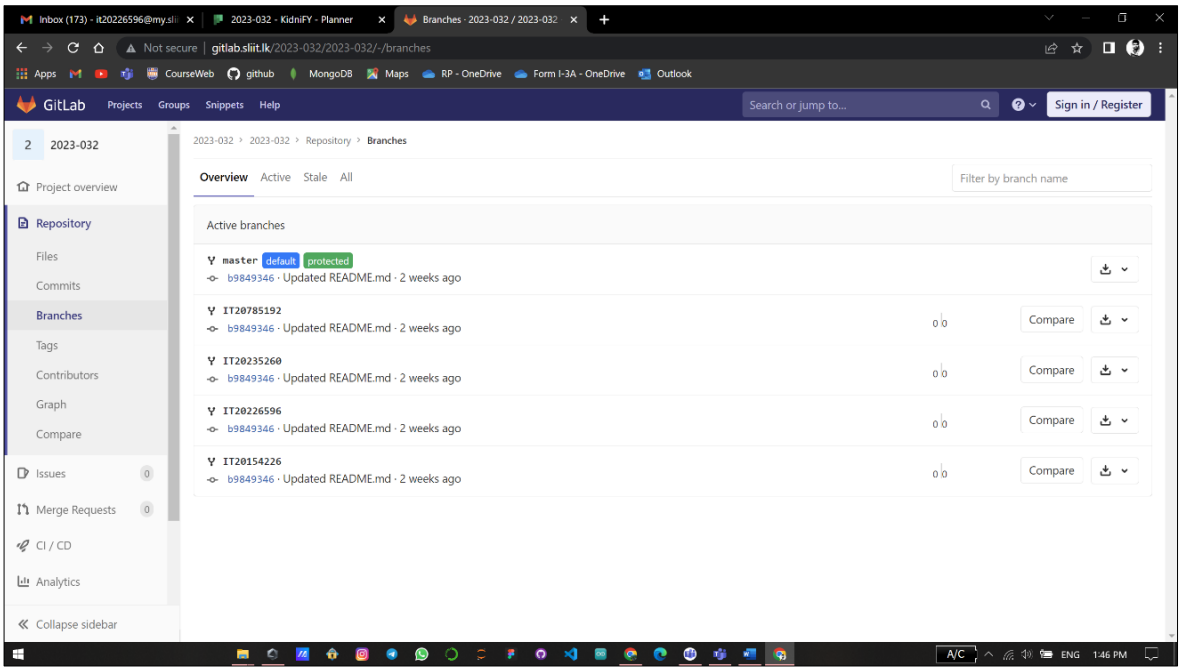






2.3 Screenshots of GitLab

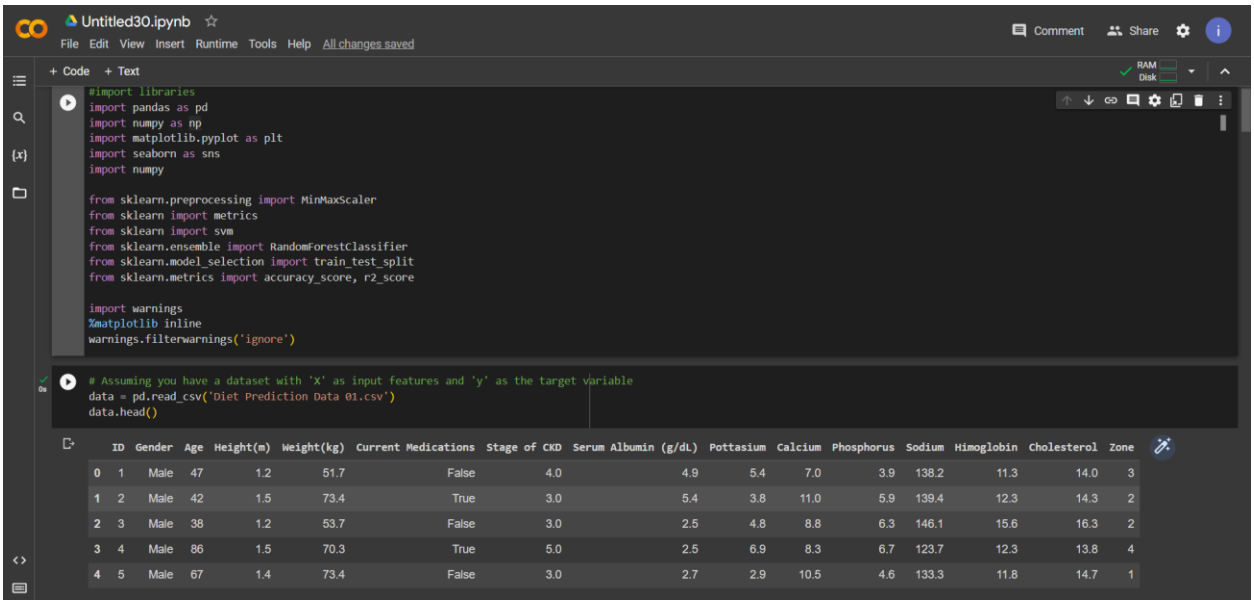




3 Project Implementation

Personalized dietary prediction for CKD patients using Machine Learning

Importing the libraries



	ID	Gender	Age	Height(m)	Weight(kg)	Current Medications	Stage of CKD	Serum Albumin (g/dL)	Pottasium	Calcium	Phosphorus	Sodium	Himoglobin	Cholesterol	Zone
0	1	Male	47	1.2	51.7	False	4.0	4.9	5.4	7.0	3.9	138.2	11.3	14.0	3
1	2	Male	42	1.5	73.4	True	3.0	5.4	3.8	11.0	5.9	139.4	12.3	14.3	2
2	3	Male	38	1.2	53.7	False	3.0	2.5	4.8	8.8	6.3	146.1	15.6	16.3	2
3	4	Male	86	1.5	70.3	True	5.0	2.5	6.9	8.3	6.7	123.7	12.3	13.8	4
4	5	Male	67	1.4	73.4	False	3.0	2.7	2.9	10.5	4.6	133.3	11.8	14.7	1

Data preprocessing

```
[145] #To view the last few rows,
      data.tail()
```

	ID	Gender	Age	Height(m)	Weight(kg)	Current Medications	Stage of CKD	Serum Albumin (g/dL)	Pottasium	Calcium	Phosphorus	Sodium	Himoglobin	Cholesterol	Zone
721	722	Female	35	1.4	82.76	False	3.0	6.3	3.5	9.1	3.4	147.0	13.2	15.2	2
722	723	Female	29	1.0	67.30	False	1.0	4.6	5.0	9.2	6.9	125.1	13.5	10.9	2
723	724	Female	61	1.1	100.45	False	3.0	4.4	6.5	9.9	2.7	139.8	14.2	10.5	4
724	725	Female	31	1.2	110.13	False	3.0	3.4	2.5	10.0	5.7	147.5	13.5	16.8	1
725	726	Female	44	1.6	95.65	False	3.0	4.0	6.1	6.7	2.3	140.3	13.1	14.8	4

```
[146] #get the column headings of the data set
      data.columns.values

array(['ID', 'Gender', 'Age', 'Height(m)', 'Weight(kg)',
       'Current Medications', 'Stage of CKD', 'Serum Albumin (g/dL)',
       'Pottasium', 'Calcium', 'Phosphorus', 'Sodium', 'Himoglobin',
       'Cholesterol', 'Zone'], dtype=object)
```

```
[146] #get the column headings of the data set
      data.columns.values

array(['ID', 'Gender', 'Age', 'Height(m)', 'Weight(kg)',
       'Current Medications', 'Stage of CKD', 'Serum Albumin (g/dL)',
       'Pottasium', 'Calcium', 'Phosphorus', 'Sodium', 'Himoglobin',
       'Cholesterol', 'Zone'], dtype=object)

[147] #Explore the data types of the columns
      data.dtypes

ID                        int64
Gender                   object
Age                       int64
Height(m)               float64
Weight(kg)              float64
Current Medications      object
Stage of CKD            float64
Serum Albumin (g/dL)    float64
Pottasium               float64
Calcium                 float64
Phosphorus              float64
Sodium                  float64
Himoglobin              float64
Cholesterol             float64
Zone                      int64
dtype: object
```

```
[148] #describe the data set
      data.describe()
```

	ID	Age	Height(m)	Weight(kg)	Stage of CKD	Serum Albumin (g/dL)	Pottasium	Calcium	Phosphorus	Sodium	Himoglobin	Cholesterol	Zone
count	726.000000	726.000000	726.000000	726.000000	716.000000	726.000000	726.000000	720.000000	726.000000	726.000000	726.000000	720.000000	726.000000
mean	363.500000	57.071625	1.344766	75.338802	3.001397	4.442287	4.770386	9.230139	4.429063	140.465840	13.429477	14.487361	2.442149
std	209.722436	17.026562	0.157983	19.197597	1.097358	1.177497	1.306648	1.587852	1.487213	11.533203	2.135800	2.254686	1.035288
min	1.000000	28.000000	1.000000	48.000000	1.000000	2.400000	2.400000	6.500000	1.900000	120.100000	10.500000	10.500000	1.000000
25%	182.250000	42.000000	1.200000	60.100000	2.750000	3.500000	3.625000	7.900000	3.100000	130.425000	11.700000	12.600000	2.000000
50%	363.500000	57.000000	1.300000	72.200000	3.000000	4.400000	4.800000	9.300000	4.400000	141.400000	13.100000	14.500000	2.000000
75%	544.750000	72.000000	1.500000	88.152500	3.000000	5.400000	5.900000	10.600000	5.700000	150.100000	14.500000	16.325000	3.000000
max	726.000000	87.000000	1.600000	119.690000	5.000000	6.500000	7.000000	12.000000	7.000000	160.000000	18.500000	18.500000	4.000000

```
[149] #To get an idea about the null values
      data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 726 entries, 0 to 725
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   ID                      726 non-null    int64
 1   Gender                  726 non-null    object
 2   Age                     726 non-null    int64
 3   Height(m)               726 non-null    float64
 4   Weight(kg)              726 non-null    float64
 5   Current Medications     704 non-null    object
 6   Stage of CKD            716 non-null    float64
 7   Serum Albumin (g/dL)    726 non-null    float64
 8   Pottasium               726 non-null    float64
 9   Calcium                 720 non-null    float64
 10  Phosphorus              726 non-null    float64
 11  Sodium                  726 non-null    float64
 12  Himoglobin              726 non-null    float64
 13  Cholesterol             720 non-null    float64
 14  Zone                    726 non-null    int64
dtypes: float64(10), int64(3), object(2)
```

```
[150] #Getting the null value count
      data.isnull().sum()

      ID                      0
      Gender                  0
      Age                     0
      Height(m)               0
      Weight(kg)              0
      Current Medications    22
      Stage of CKD           10
      Serum Albumin (g/dL)    0
      Pottasium               0
      Calcium                 6
      Phosphorus              0
      Sodium                  0
      Himoglobin              0
      Cholesterol             6
      Zone                    0
      dtype: int64
```

```
      #replace null values using mean and value counts
      Df_dataset = data.apply(lambda x: x.fillna(x.mean())
                        if x.dtype =='float'
                        else x.fillna(x.value_counts().index[0]))
```

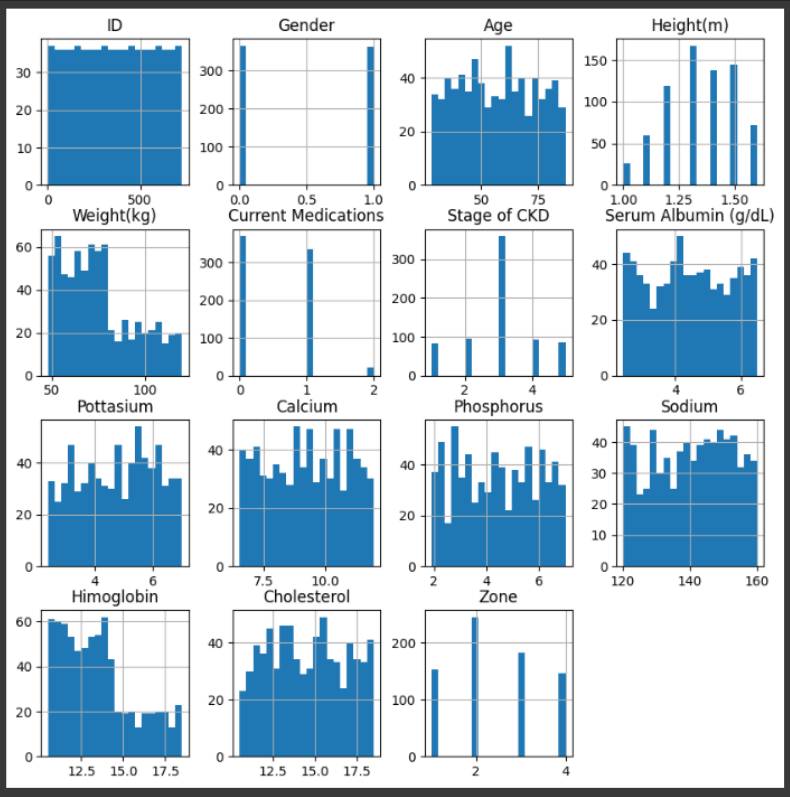
```
[153] from sklearn.preprocessing import LabelEncoder

      # Convert categorical variable using LabelEncoder
      label_encoder = LabelEncoder()
      data['Gender'] = label_encoder.fit_transform(data['Gender'])
      data['Current Medications'] = label_encoder.fit_transform(data['Current Medications'])
```

```
[154] data.info()

      <class 'pandas.core.frame.DataFrame'>
      RangeIndex: 726 entries, 0 to 725
      Data columns (total 15 columns):
       #   Column                Non-Null Count  Dtype
      ---  ------                --------------  -----
       0   ID                    726 non-null    int64
       1   Gender                726 non-null    int64
       2   Age                   726 non-null    int64
       3   Height(m)             726 non-null    float64
       4   Weight(kg)            726 non-null    float64
       5   Current Medications   726 non-null    int64
       6   Stage of CKD          716 non-null    float64
       7   Serum Albumin (g/dL)  726 non-null    float64
       8   Pottasium             726 non-null    float64
       9   Calcium               720 non-null    float64
       10  Phosphorus            726 non-null    float64
       11  Sodium                726 non-null    float64
       12  Himoglobin            726 non-null    float64
       13  Cholesterol           720 non-null    float64
       14  Zone                  726 non-null    int64
      dtypes: float64(10), int64(5)
      memory usage: 85.2 KB
```

```
[155] data
```

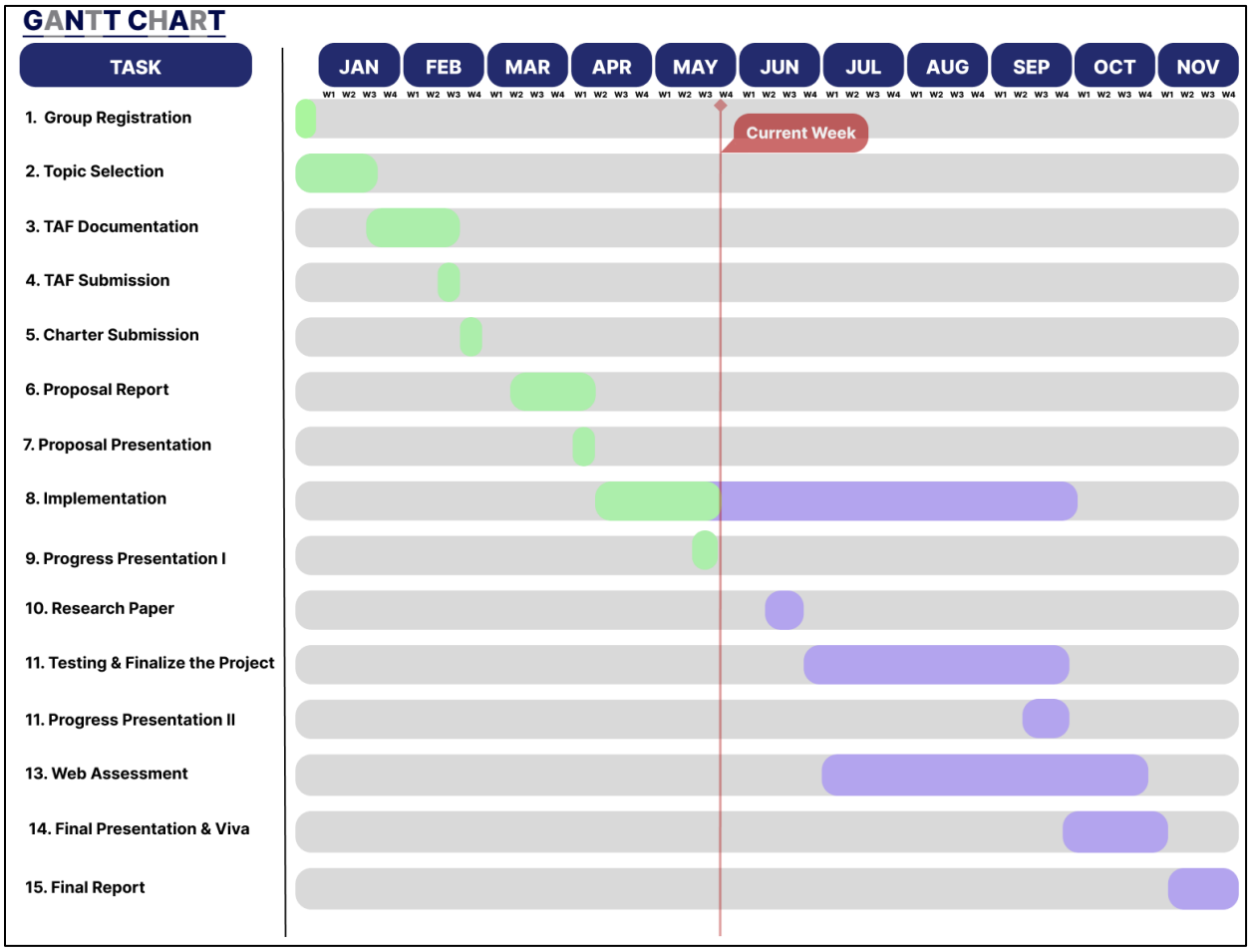
	ID	Gender	Age	Height(m)	Weight(kg)	Current Medications	Stage of CKD	Serum Albumin (g/dL)	Pottasium	Calcium	Phosphorus	Sodium	Himoglobin	Cholesterol	Zone
0	1	1	47	1.2	51.70	0	4.0	4.9	5.4	7.0	3.9	138.2	11.3	14.0	3
1	2	1	42	1.5	73.40	1	3.0	5.4	3.8	11.0	5.9	139.4	12.3	14.3	2
2	3	1	38	1.2	53.70	0	3.0	2.5	4.8	8.8	6.3	146.1	15.6	16.3	2
3	4	1	86	1.5	70.30	1	5.0	2.5	6.9	8.3	6.7	123.7	12.3	13.8	4
4	5	1	67	1.4	73.40	0	3.0	2.7	2.9	10.5	4.6	133.3	11.8	14.7	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
721	722	0	35	1.4	82.76	0	3.0	6.3	3.5	9.1	3.4	147.0	13.2	15.2	2
722	723	0	29	1.0	67.30	0	1.0	4.6	5.0	9.2	6.9	125.1	13.5	10.9	2
723	724	0	61	1.1	100.45	0	3.0	4.4	6.5	9.9	2.7	139.8	14.2	10.5	4

```
[156] data.hist(bins=20, figsize=(10, 10))
      plt.show()
```




```
[158] #split the dataset before balancing the class.
      X = Df_dataset.drop(columns=[ 'Zone'], axis 1)
      y = Df_dataset['Zone']

[159] X.columns.values

      array(['ID', 'Gender', 'Age', 'Height(m)', 'Weight(kg)',
             'Current Medications', 'Stage of CKD', 'Serum Albumin (g/dL)',
             'Pottasium', 'Calcium', 'Phosphorus', 'Sodium', 'Himoglobin',
             'Cholesterol'], dtype=object)

[160] y.value_counts()

      2    245
      3    182
      1    153
      4    146
      Name: Zone, dtype: int64

      #delete missing values rows
      data = data.dropna(axis=0)

[163] # Split the data into train and test sets
      X_train, X_test, y_train, y_test = train_test_split(data.drop("Zone", axis=1), data["Zone"], test_size=0.25)

      # Create the random forest model
      rf_model = RandomForestClassifier(n_estimators=100, max_depth=10)

      # Fit the model to the training data
      rf_model.fit(X_train, y_train)

      # Predict the labels for the test data
      y_pred = rf_model.predict(X_test)

      # Calculate the accuracy of the model
      accuracy = accuracy_score(y_test, y_pred)
      print("Accuracy:", accuracy)

      # Calculate the R-squared value of the model
      r2 = r2_score(y_test, y_pred)
      print("R-squared:", r2)

      Accuracy: 0.95
      R-squared: 0.94269853783
```

4 Gantt Chart



- Completed

- In Progress

5 Work Breakdown Structure