**COVID 19-SRC**

**Aim:**

Building an UI/UX for Covid19 Open API tracker site.

**Users of the System:**

1.   Government

2.   Public

**Functional Requirements:**

* Build an application that Public can access and view the COVID19 status.
* The application should have signup, login, profile, dashboard page and individual records.

1. State Name
2. Confirmed
3. Deaths
4. Fatality Rate
5. Last Update

* This application should have a provision to maintain a database for individual information, public information and COVID19 portfolio.
* Also, an integrated platform required for government and public.
* Aadhar integration for intimating individual reports to the public.
* Clarified that all contact tracing and COVID-19 status apps, including ones that merely store an individual’s vaccination or test records, need to complete the "COVID-19 contact tracing and status apps" section in the App content page.

**Output/ Post Condition:**

* Records Persisted in Success & Failure Collections
* Standalone application / Deployed in an app Container

**Non-Functional Requirements:**

**Security**

* Apps must have a publicly accessible privacy policy that comprehensively discloses the access, collection, use and sharing of personal and sensitive user data
* You must disclose the use of COVID-19 related data in user-facing privacy disclosures (e.g., privacy policy and/or in-app disclosures) for your app(s).
* Apps must handle all personal or sensitive user data securely, including transmitting it using modern cryptography (for example, over HTTPS).

**Standard Features**

* Number of tested
* Confirmed cases
* Deaths in the country
* A heatmap of the largest concentrations of confirmed covid-19 cases
* Locations of public testing centres in each state

**Logging**

* The system should support logging(app/web/DB) at all levels

**Cloud**

* The Solution should be made Cloud-ready and should have a minimum impact when moving away to Cloud infrastructure

**Technology Stack**

* HTML&CSS
* JavaScript

**Key points to remember:**

1.   The id (for frontend) and attributes(backend) mentioned in the SRS should not be modified at any cost. Failing to do may fail test cases.

2.   Remember to check the screenshots provided with the SRS. Strictly adhere to id mapping and attribute mapping. Failing to do may fail test cases.

3.   Strictly adhere to the proper project scaffolding (Folder structure), coding conventions, method definitions and return types.

Adhere strictly to the endpoints given below.

**Application assumptions:**

1.   The login page should be the first page rendered when the application loads.

2.   Manual routing should be restricted by using AuthGuard by implementing the can Activate interface. For example, if the user enters as [http://localhost:8000/signup](http://localhost:4200/signup) or [http://localhost:8000/home](http://localhost:4200/home) the page should not navigate to the corresponding page instead it should redirect to the login page.

3.   Unless logged into the system, the user cannot navigate to any other pages.

4.   Logging out must again redirect to the login page.

5.   To navigate to the admin side, you can store a user type as admin in the database with a username and password as admin.

6.   Use admin/admin as the username and password to navigate to the admin dashboard.

**Validations:**

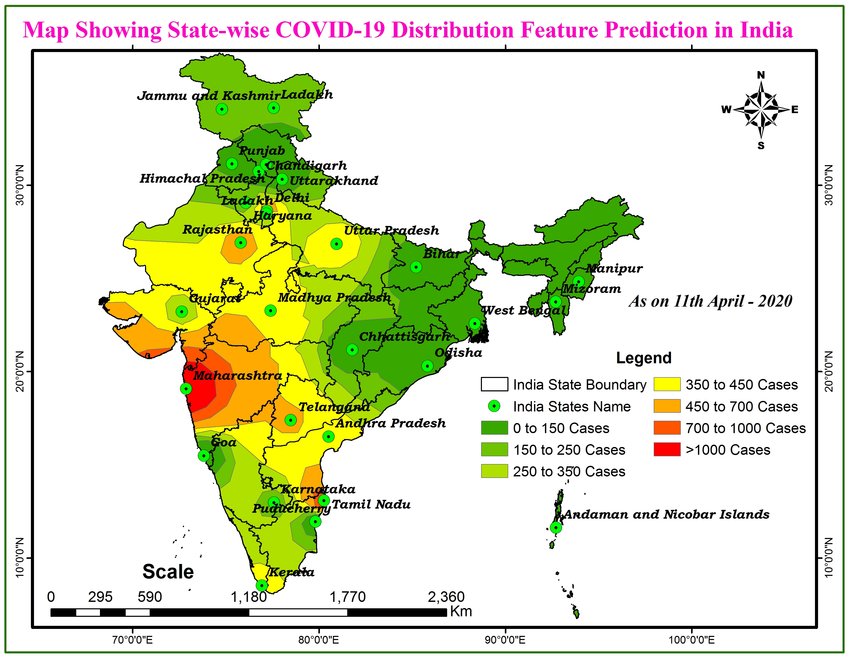
   Basic mobile validation should be performed.

Aadhar Password validations should be performed

**Project Tasks:**

* Complete the "COVID-19 contact tracing and status apps" section in the App content page
* Submit proof of eligibility via the Advance Notice form
* Privacy requirements
* App visibility and user awareness
  + For apps that collect information in the foreground or use foreground service
  + For apps that collect information when running as a background service
* API requirements
* Editorial and quality requirements
* App review and visibility

COVID-19=THERMAL MAP



**Customer:**

1. **Auth**: Design an auth component (Name the component as ***auth***for angular app whereas ***Auth***for react app. Once the component is created in react app, name the jsx file as same as component name i.e Auth.jdm file) where the customer can authenticate login and signup credentials

2.   **Signup**: Design a signup page component (Name the component as ***signup***for angular app whereas ***Signup***for react app. Once the component is created in react app, name the jdm file as same as component name i.e Signup.jdm file)where the new customer has options to sign up by providing their basic details.

a.   Ids:

* Aadhar number
* Mobile number
* password
* confirm password
* submit Button
* Sign in Link
* signup Box j

b.   API endpoint Url: [http://localhost:8000/signup](http://localhost:4200/signup)

c.   Output screenshot:

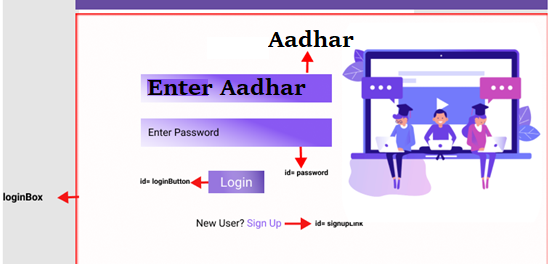
3.   **Login**: Design a login page component named (Name the component as ***login*** for angular app whereas ***Login*** for react app. Once the component is created in react app, name the jdm file as same as component name i.e Login.jdm file) where the existing customer can log in using the registered email id and password.

a.   Ids:

* Aadhar number
* password
* submitButton
* signupLink
* loginBox

b.   API endpoint Url: [http://localhost:8000/login](http://localhost:4200/login)

c.   Output screenshot:



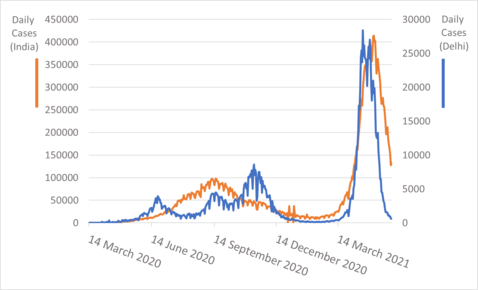
4.   **Dashboard / Home**: Design a home page component named (Name the component as ***homepage***for angular app whereas ***HomePage***for react app. Once the component is created in react app, name the jsx file as same as component name i.e HomePage.jsx file) that has the navigation bar

a.   Ids:

1. user Number
2. Home Button
3. Personnel data
4. Over all State data
5. Logout Button

b.   API endpoint Url: [http://localhost:8000/](http://localhost:4200/login)home

c. Screenshot



**Admin:**

6.   **Admin Dashboard:** Design a dashboard page named (Number of affected as ***dashboard*** for angular app whereas ***Dashboard*** for react app. Once the numbers created in react app, name the jdm file as same as component name i.e Dashboard.jdm file) where the number of affected persons is displayed on the admin side.

a. **Admin Navigation**: Design a navigation component (Name the component as ***adminhomepage*** for angular app whereas ***AdminHomePage***for react app.

      i.Ids:

1. admin Number
2. admin add Button
3. admin confirm Button
4. logout Button

b. **Add number of patients affected**: Design an add product component (Name the component as ***addpatient*** for angular app whereas ***AddPatient*** for react app.

1.addnumber of affected count

2.StateName

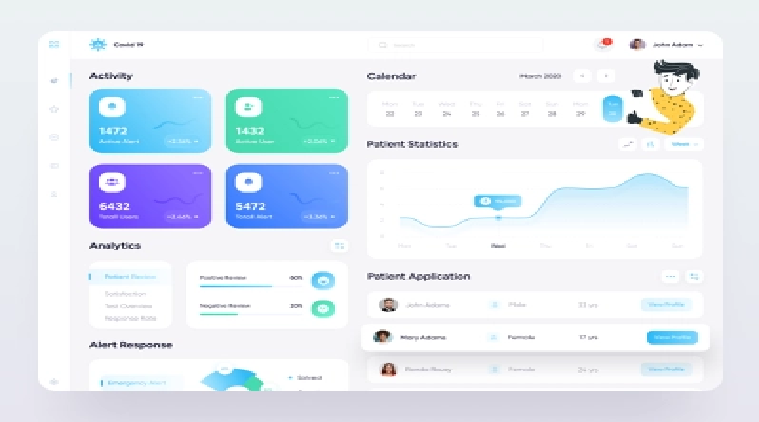
3.District Name

4Aadhar Number

5.affected

6.Recovered

API endpoint Url: [http://localhost:8000/ad](http://localhost:4200/ad)dProduct



**Backend:**

**Class and Method description:**

**Model Layer:**

1.   **UserModel**: This class stores the user type (admin or the customer) and all user information.

a.   Attributes:

                                                         i.   Aadhar: String

                                                       ii.   password: String

                                                      iii.   mobileNumber: String

                                                       iv.   active: Boolean

                                                      v.   role: String

2.   **LoginModel**: This class contains the email and password of the user.

a.   Attributes:

                                                         i.   Aadhar: String

                                                       ii.   password: String

**Controller Layer:**

3.   **SignupController**: This class control the user signup

a.   Methods:

                 i.   saveUser(UserModel user): This method helps to store users in the database and return true or false based on the database transaction.

4.   **LoginController**: This class controls the user login.

a.   Methods:

              i.   checkUser(LoginModel data): This method helps the user to sign up for the application and must return true or false.

5.   **Patient Controller**: This class controls the add/edit/update/view number of person affected by Covid 19.

a.   Methods:

                 i.   List<state> getstate(): This method helps the admin to fetch all datas from the database.

                ii.   List<District> getDistric (): This method helps to retrieve all the datas from the database.

               iii. Patient Details EditData(String id): This method helps to retrieve a affected details from the database based on the Aadhar id.

            iv.   Patient Details Edit Save(Patient Details data): This method helps to edit a Patient Details and save it to the database.

               v.   Patient Details Save(Patient Details data): This method helps to add a new Patient Details to the database.

               vi.   Patient Details Delete (String id): This method helps to delete a Patient Details from the database.

BY TEAM-6

1.SACHIN

2.VIJAY SAI RAJ

3.SATHVIK

4.THARUN KUMAR

5. SAKTHI ROHITH