COVID -19

SOFTWARE REQUIREMENT SPECIFICATION

PRESENTED BY

KEERTHANA.V

VARSHITHA.K

Objective:

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. Latitude
    6. Longitude
    7. 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.
- Filters show Low to High or showcasing COVID19 affected area should be highlighted.
- Aadhar card 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.
- Additional requirements are introduced for apps that plan to remove contact tracing or COVID-19 status functionalities.
- Apps that provide medical, treatment, vaccine, testing, or other related information for COVID-19.
- Apps that support COVID-19-related response, containment, research, or education/training efforts.
- Apps that support services used to respond specifically to COVID-19, for example, apps that provide social support (food stamps, payment), healthcare, loans, etc., specifically in response to COVID-19.

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.
- Apps created specifically for the COVID-19 response may not access personal and sensitive data that is not required to directly support the public health emergency, and may only use the data collected to support COVID-19-related efforts or epidemiological research

- 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 heat map of the largest concentrations of confirmed covid-19 cases
- Locations of public testing centres in each state
- News updates from major health org's

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

Browser Compatible

All latest browsers

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 or http://localhost:8000/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:

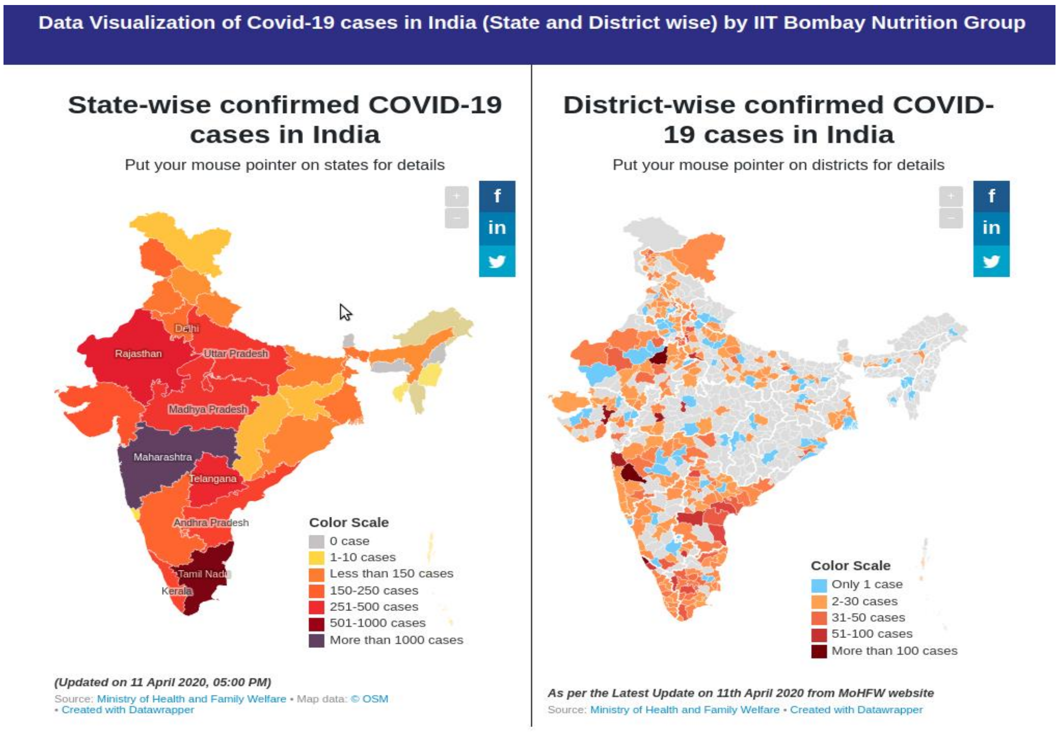
Basic mobile validation should be performed.
Aadhar Password validations should be performed

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

USER

Covid19 Heat Map


Data Visualization of Covid-19 cases in India (State and District wise) by IIT Bombay Nutrition Group
State-wise confirmed COVID-19 cases in India
Put your mouse pointer on states for details
Delhi
Rajasthan
Uttar Pradesh
Madhya Pradesh
Maharashtra
Telangana
Andhra Pradesh
Tamil Nadu
Kerala
Color Scale
0 case
1-10 cases
Less than 150 cases
150-250 cases
251-500 cases
501-1000 cases
More than 1000 cases
(Updated on 11 April 2020, 05:00 PM)
Source: Ministry of Health and Family Welfare • Map data: © OSM • Created with Datawrapper
District-wise confirmed COVID-19 cases in India
Put your mouse pointer on districts for details
Color Scale
Only 1 case
2-30 cases
31-50 cases
51-100 cases
More than 100 cases
As per the Latest Update on 11th April 2020 from MoHFW website
Source: Ministry of Health and Family Welfare • Created with Datawrapper

Frontend:

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.jsx 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 jsx file as same as component name i.e Signup.jsx file)where the new customer has options to sign up by providing their basic details.

   a. Ids:

- Aadhar number
- mobilenumber
- password
- confirmpassword
- submitButton
- signinLink
- signupBox

   b. API endpoint Url: http://localhost:8000/signup

   c. Output screenshot:

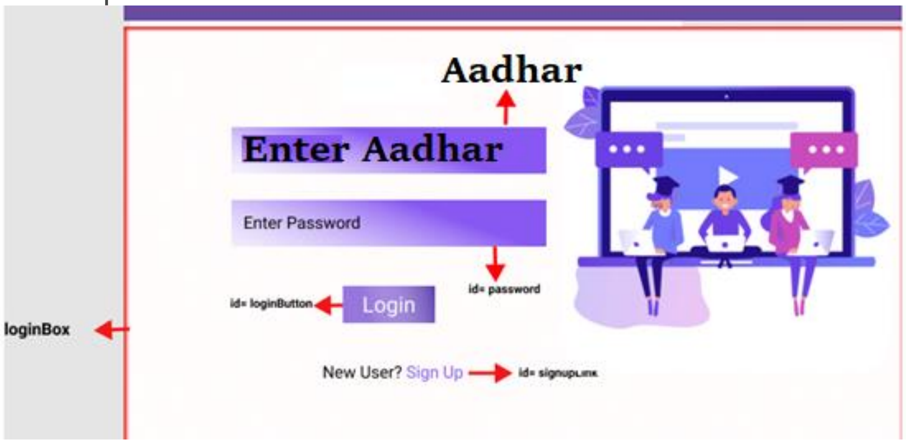
3. **Login**: Design a login page component named (Name the component as ***login*** for angular app whereas ***Login*** for react app. Once the component is created in react app, name the jsx file as same as component name i.e Login.jsx file)where the existing customer can log in using the registered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

   a. Ids:

- Aadhar number
- password
- submitButton
- signupLink
- loginBox

   b. API endpoint Url: http://localhost:8000/login

   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

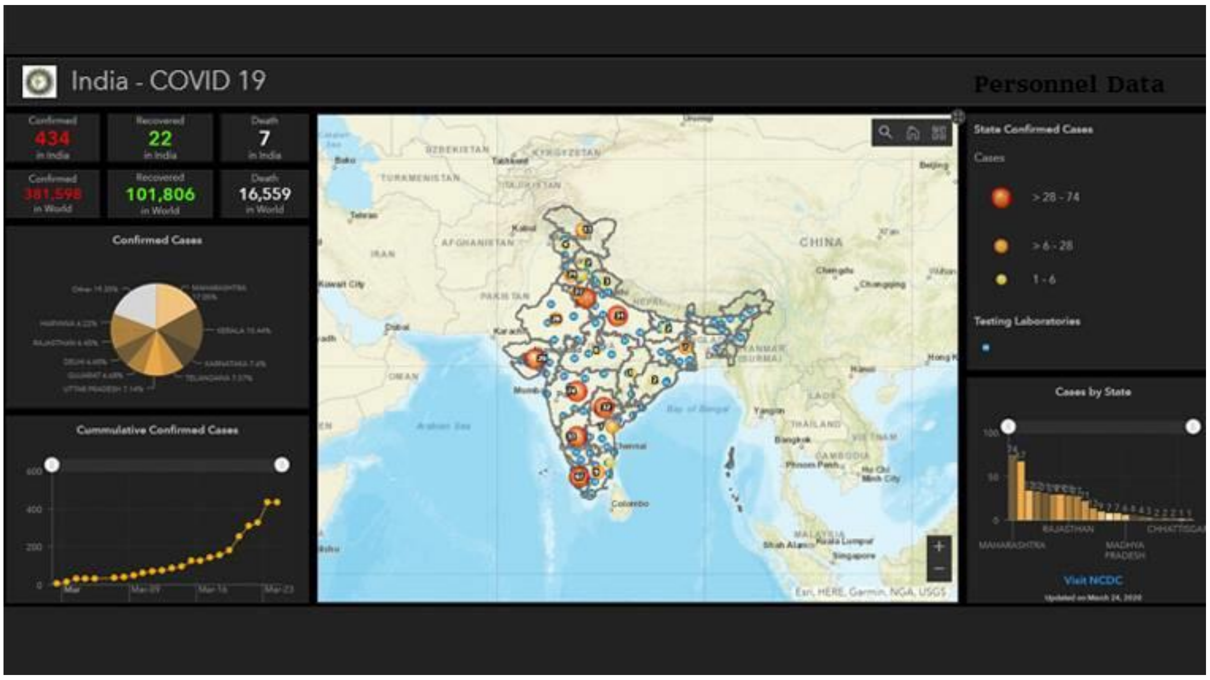
   a. Ids:

1. userNavbar

2. HomeButton
3. Personnel data
4. Over all State data
5. logoutButton

   b. API endpoint Url: http://localhost:8000/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 jsx file as same as component name i.e Dashboard.jsx 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 ***Admin Home Page*** for react app.

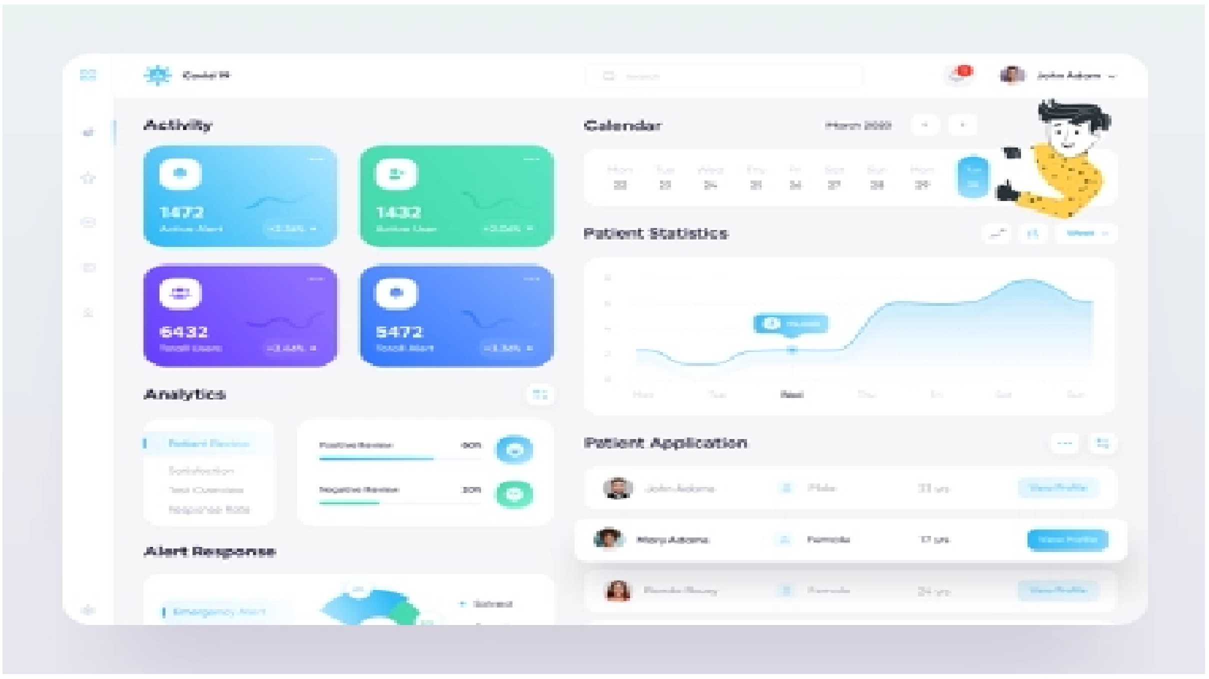
   i.Ids:

1. adminNavbar
2. adminaddtButton
3. adminconfirmButton
4. logoutButton

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

1.addnumber of affected count
2.StateName
3.District Name
4Aadhar Number
5.affected
6.Recovered

7.add data Button
ii. API endpoint Url: http://localhost:8000/addProduct
c. Screenshot



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

3. **Covid 19 Model**: This class stores the details of the patient.
   a. Attributes:
      i. StateId: String
      ii. imageUrl: String
      iii. patient name: String
      iv. Status: String

Controller Layer:

6. **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.

7. **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.

8. **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.