Covid-19 Tracing System

We all know that the world has been shaken by the Pandemic COVID-19 which has infected millions of people and affected all the humanity stature. And the governments are trying their level best in controlling the situations. But in India with a population of over 1.3 billon it's difficult to keep track of details.

So Design a Project to track the details of State situation with regarding to Covid-19. Create a State Class and store details of Group of States where each State contains the following details ->

1. Name of the State.

2. Zone (RED, GREEN, YELLOW) depicting the severity.

3. Number of Positive Cases Registered.

4. Number of Deaths Registered**.**

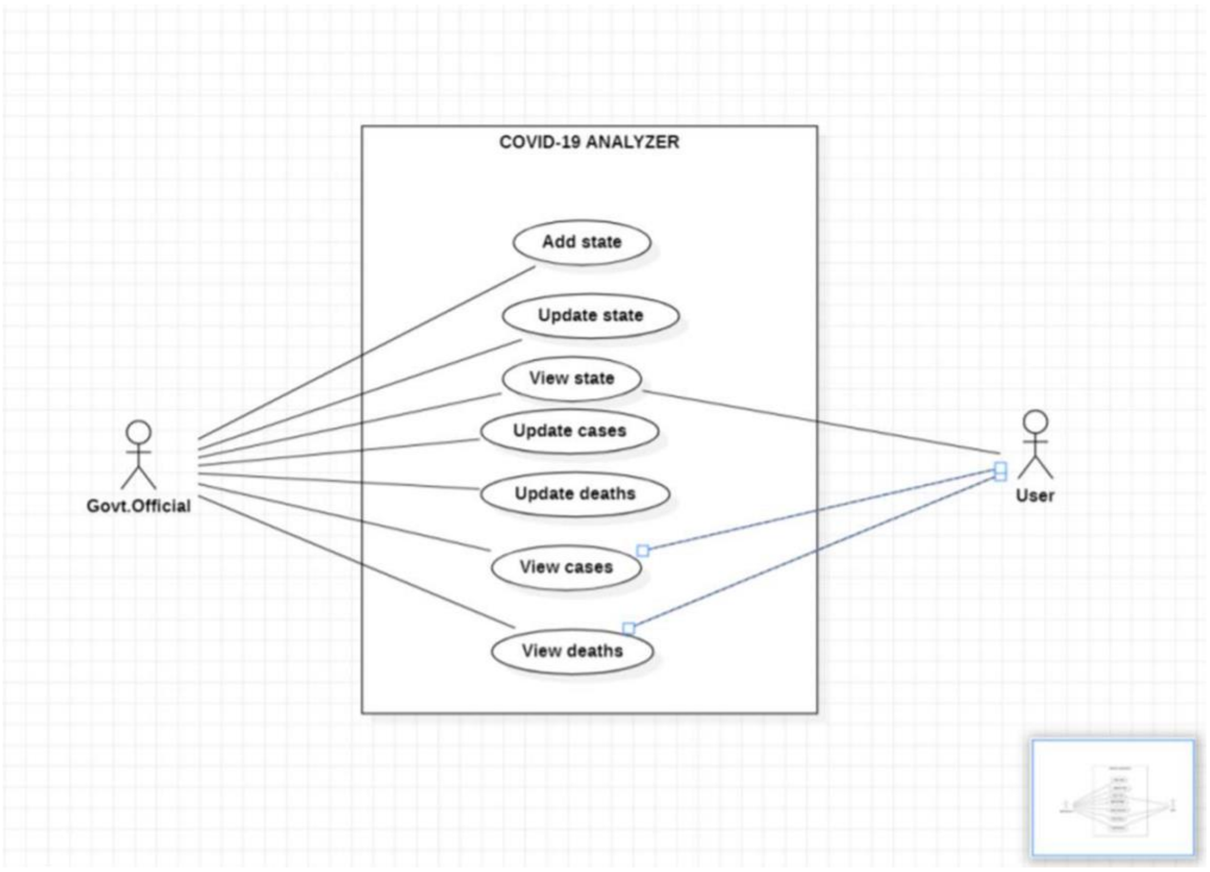
Design two different console based programs for Citizen of State and Government Officials.

Design the Citizen's Console as Citizen can check what the status of the states is and act accordingly. Collect details of Citizen and should have the features to check his State's Status and details of it.

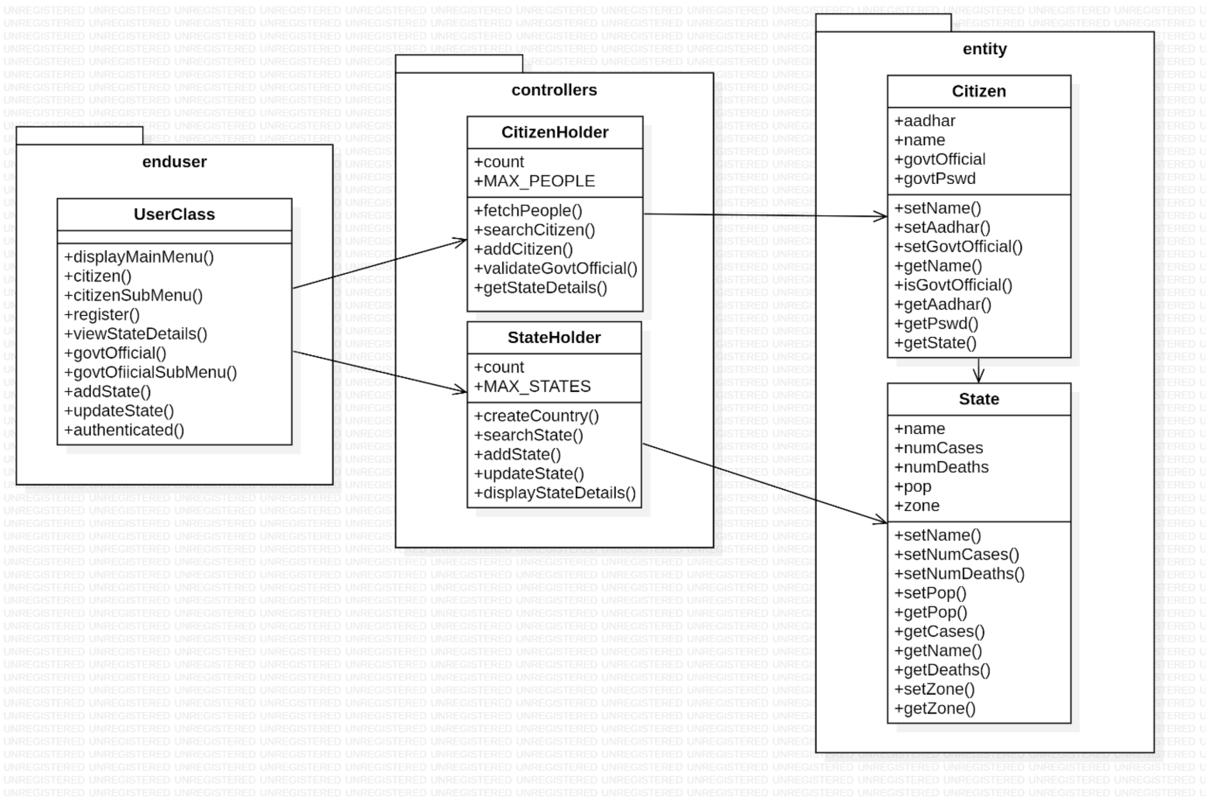
And for the Government Officials to access the State information can enter new State details, update the details. In order to access the Government Official Console provides a prefixed password so that only they can access these features.

+ Use modularization techniques and also unit-test case development while building the project.

1. Draw the Use Case Diagrams



2. Draw the Class Diagrams.



3. Code of the project.

CITIZENHOLDER CLASS

```
package controllers;

import entity.Citizen;

public class CitizenHolder {
	private Citizen[] people;
	private int count;
	private static final int MAX_PEOPLE=10000;
	private static final CitizenHolder ch=new CitizenHolder();
	public CitizenHolder()
	{
		this.people=new Citizen[MAX_PEOPLE];
		this.count=0;
	}
	public static CitizenHolder fetchPeople()
	{
		return ch;
	}
	private Citizen searchCitizen(long a)
	{
		for(int i=0;i<count;i++)
		{
			if(people[i].getAadhar()==a)
				return people[i];
		}
		return null;
	}
	public boolean addCitizen(String n,long a)
	{
		Citizen c=this.searchCitizen(a);
		if(c==null && count<MAX_PEOPLE)
		{
			this.people[count++]=new Citizen(n,a);
			return true;
		}
		return false;
	}
	public boolean validateGovtOfficial(long a,String Pswd)
	{
		return (this.searchCitizen(a)!=null &&
		Pswd.equals(Citizen.getPswd()));
	}
	public boolean getStateDetails(long a)
	{
		Citizen c=this.searchCitizen(a);
		if(c!=null)
		{
			System.out.println(c.getState());
		}
		return false;
	}
}
```

CITIZEN CLASS

```
package entity;

public class Citizen {
	//members
	long aadhar;
	String name;
	private State s;
	boolean govtOfficial;
	private static final String govtPswd="Govt@123";

	//methods
	//constructor

	private Citizen() {
		this.aadhar=-1;
		this.name="no name";
		this.govtOfficial=false;
	}

	public Citizen(String name, long aadhar) {
		this();
		if(name!=null && !(name.isEmpty())) this.name=name;
		if(aadhar>0) this.aadhar =aadhar;
	}

	//mutator
	public boolean setName(String n) {
		if(n!=null && !(n.isEmpty())) {
			this.name=n;
			return true;
		}
		return false;
	}
	public void setAadhar(long n) {
//		if(n>100000000000 && n<1000000000000) {
			this.aadhar=n;
//			return true;
//		}
//		return false;
	}
	public void setGovtOfficial(boolean b) {
		this.govtOfficial=b;
	}

	//accessor
	public String getName() {
		return this.name.toUpperCase();
	}
	public boolean isGovtOfficial() {
		return this.govtOfficial;
	}

	public long getAadhar() {
		return this.aadhar;
	}
	public static String getPswd() {
		return govtPswd;
```

```java
	}
	public State getState() {
		return this.s;
	}

}
```

STATEHOLDER CLASS

```java
package controllers;
import entity.State;

public class StateHolder {
	//members
	private State[] country;
	private int count;
	//if a member is final - indicates that it can only be initialized once
	//static - it belongs to class and there is only one copy of the variable
	//instance - it belongs to the object of the class and there is a copy of
the variable for every object
	private static final int MAX_STATES =29;
	private static StateHolder sh = null;

	//private no-parameter constructor is to initialize the members especially
when the members are reference type
	public StateHolder() {
		country= new State[MAX_STATES];
		this.count=0;
	}

	//do other classes create objects of this class?
	//assumptions in the project - one country
	//my code should let me create objects but should not allow me to create
more than one object
	//alternate ways to create objects other than using public constructors

	//creational design pattern - singleton pattern (lazy instantiation and
early instantiation)
	public static StateHolder createCountry() {
		if(sh==null) sh=new StateHolder();
		return sh;
	}

	//functional methods
	//search data
	//add data
	//update data

	private State searchState(String n) {
		State s = null;
		//logic
		for(int i=0;i<count;i++) {
			if(country[i].getName().equalsIgnoreCase(n)) s=country[i];
		}
		return s;
	}

	public boolean addState(String n,long p) {
		//logic
		//conditions - n should not already exists, data structure is not
full, n is not null, n is not empty,
		if(n!=null && !(n.isEmpty()) && searchState(n)==null &&
count<MAX_STATES) {
			//logic to add state
			country[count++]= new State(n,p);
			return true;
		}
		return false;
	}

	public boolean updateState(String n,int nc,int nd) {
		//logic
		State s=this.searchState(n);
		if(s!=null) {
			//logic to update
			s.setNumCases(nc);
			s.setNumDeaths(nd);
			s.setZone();
			return true;
		}
		return false;
	}

	public String displayStateDetails(String n) {
		State s=this.searchState(n);
		if(s!=null) return s.toString();
		else return "State with name "+n+" doesn't exist";
	}
}
```

STATE CLASS

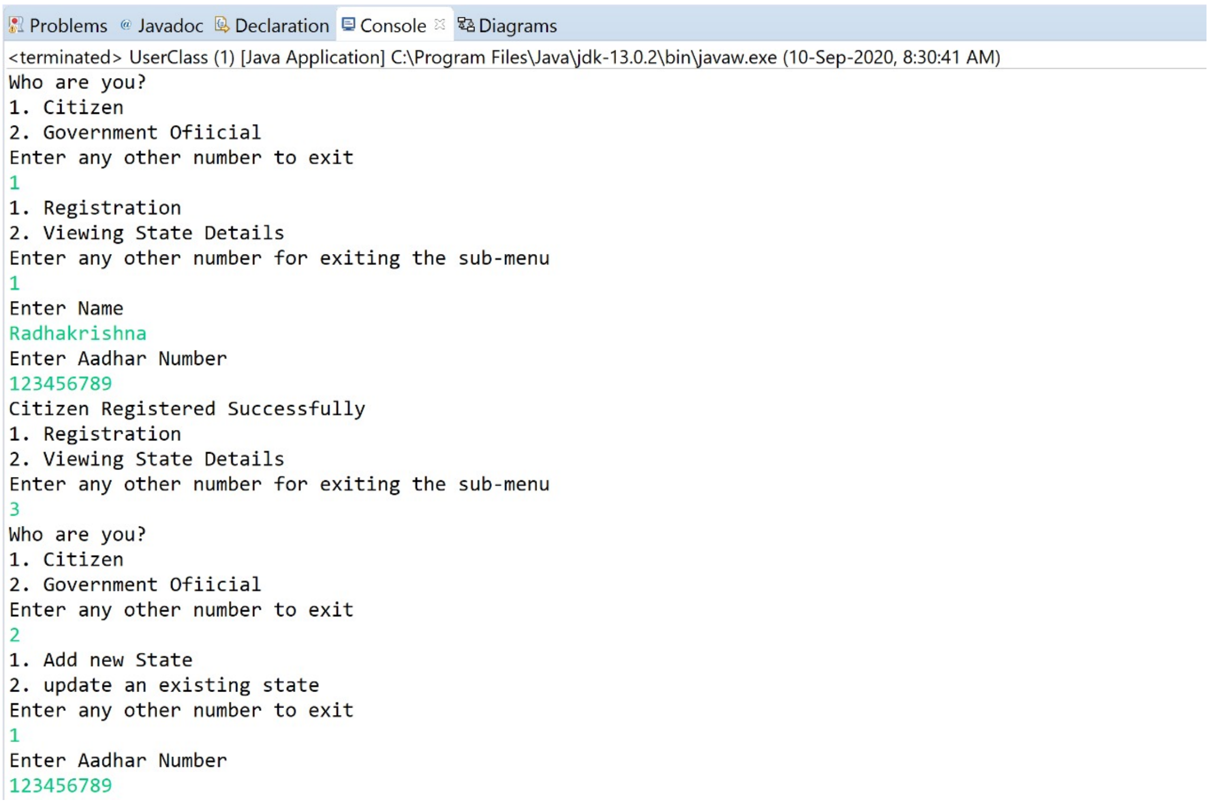
```
package entity;

public class State {
	//members;
	String name;
	long numCases;
	long numDeaths;
	long pop;
	String zone;

	//methods
	//constructors
	//accessors and mutators
	//functional methods
	//display methods

	private State() {
		this.name="no name";
		this.numCases=0;
		this.numDeaths=0;
		this.pop=0;
		this.zone="no zone set";
	}
	public State(String name, long pop) {
		this(); //chaining this constructor with no parameter constructor
		this.setName(name);
		this.setPop(pop);
	}

	//mutator method
	public boolean setName(String name) {
		if(name!=null) {
			this.name=name;
			return true;
		}
		return false;
	}
	public boolean setNumCases(long n) {
		if(n>=0) {
			this.numCases=n;
			return true;
		}
		return false;
	}
	public boolean setNumDeaths(long n) {
		if(n>=0) {
			this.numDeaths=n;
			return true;
		}
		return false;
	}
	public boolean setPop(long p) {
		if(p>10000) {
			this.pop=p;
			return true;
		}
		return false;
	}
	//no mutator for zone - zone is calculated or determined, it is not set
	//accessor methods
	public String getName() {
		return this.name.toUpperCase();
	}
	public long getCases() {
		return this.numCases;
	}
	public long getDeaths() {
		return this.numDeaths;
	}
	public long getPop() {
		return this.pop;
	}
	public String getZone() {
		this.setZone();
		return this.zone.toUpperCase();
	}

	//functional methods
	public void setZone() {
		if (this.getCases()*100/this.getPop()>30) this.zone="RED";
		else if (this.getCases()*100/this.getPop()>20) this.zone="YELLOW";
		else this.zone="GREEN";
	}

	//display method
	public String toString() {
		String output = "";
		output= String.format("State Name: %S%nNumber of +ve Cases: 
%d%nNumber of  Deaths=%d%nYour State is in %s zone%n", this.getName() , 
this.getCases(), this.getDeaths() , this.getZone());
		return output;
	}
}
```

USERCLASS

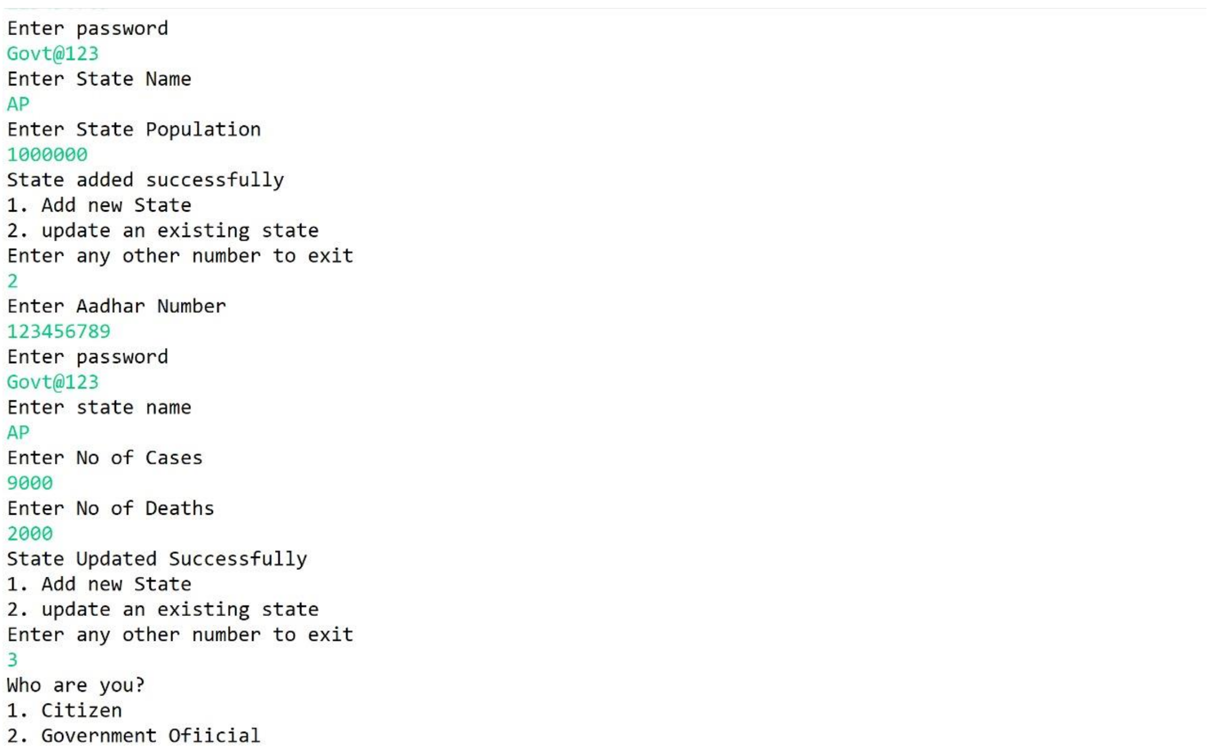
```
package enduser;
import java.util.*;

import controllers.CitizenHolder;
import controllers.StateHolder;

public class UserClass {
	//Take input
	//use classes and their functionality
	//give output
	static CitizenHolder ch=new CitizenHolder();
	static StateHolder sh=new StateHolder();

	private static final Scanner s=new Scanner(System.in);
	public static void main(String[] args) {
		boolean repeat =true;
		while(repeat) {
			switch(displayMainMenu()) {
			case 1:
				citizen();
				break;
			case 2:
				govtOfficial();
				break;
			default:
				repeat = false;
			}
		}
	}

	private static int displayMainMenu() {
		System.out.println("Who are you?");
		System.out.println("1. Citizen");
		System.out.println("2. Government Ofiicial");
		System.out.println("Enter any other number to exit");
		return s.nextInt();
	}

	private static void citizen() {
		//write sub-menu for citizen
		boolean repeat=true;
		while(repeat) {
			switch(citizenSubMenu()) {
			case 1: register();
					break;
				case 2: viewStateDetails();
						break;
				default: repeat=false;
				}
			}
		}

		private static int citizenSubMenu() {
			System.out.println("1. Registration");
			System.out.println("2. Viewing State Details");
			System.out.println("Enter any other number for exiting the sub-
menu");
			return s.nextInt();
		}

		private static void register() {
			System.out.println("Enter Name");
			String name=s.next();
			System.out.println("Enter Aadhar Number");
			long a = s.nextLong();
			if(ch.addCitizen(name,a)) System.out.println("Citizen Registered
Successfully");
			else System.out.println("Citizen cannot be registered");
		}

		private static void viewStateDetails() {
			//logic citizen enters state name, search for state and view details
of state
			System.out.println("Enter state name");
			String name=s.next();
			System.out.println(sh.displayStateDetails(name));
		}

		private static void govtOfficial() {
			//all operations as govt Official coded here
			//logic - display sub-menu for govt official
			//functionalities - add state, update state
			//functionality - authenticate a govt official
			//functionality - display all citizens details
			boolean repeat=true;
			while(repeat) {
				switch(govtOfficialSubMenu()) {
				case 1: addState();
						break;
				case 2: updateState();
						break;
				default: repeat=false;
				}
			}
		}

		private static int govtOfficialSubMenu() {
			System.out.println("1. Add new State");
			System.out.println("2. update an existing state");
			System.out.println("Enter any other number to exit");
			return s.nextInt();
		}

		private static void addState() {
```

```
		//logic - if authenticated then add state by taking name and pop 
from the console
		if(authenticated()) {
			System.out.println("Enter State Name");
			String sname=s.next();
			System.out.println("Enter State Population");
			long p=s.nextLong();
			if(sh.addState(sname,p)) System.out.println("State added 
successfully");
			else System.out.println("State cannot be added");
		}
		else System.out.println("User not authenticated");
	}

	private static void updateState() {
		//logic - authenticate, ask for state name and if state exists then 
update state
		if(authenticated()) {
			System.out.println("Enter state name");
			String sname=s.next();
			System.out.println("Enter No of Cases");
			int nc=s.nextInt();
			System.out.println("Enter No of Deaths");
			int nd=s.nextInt();
			if(sh.updateState(sname,nc,nd)) System.out.println("State 
Updated Successfully");
			else System.out.println("State "+sname+" doesn't exist");
		}
		else System.out.println("User not authenticated");
	}

	private static boolean authenticated() {
		//logic need aadhar and password
		//if citizen exists with aadhar and if password is matching then 
return true
		System.out.println("Enter Aadhar Number");
		long a=s.nextLong();
		System.out.println("Enter password");
		String pswd=s.next();
		return ch.validateGovtOfficial(a,pswd);
	}

}
```

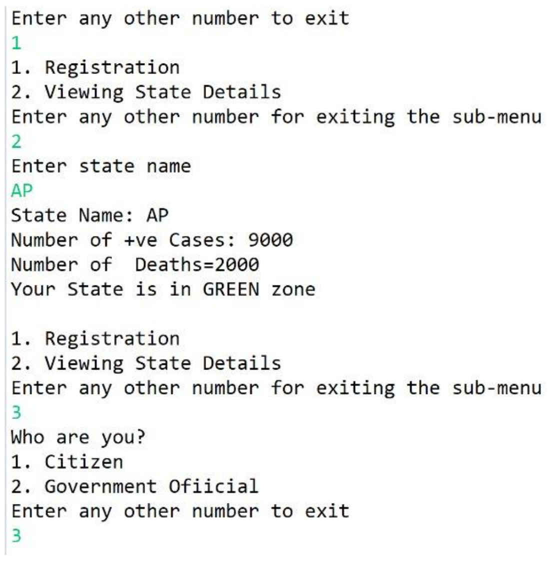
OUTPUT



```
<terminated> UserClass (1) [Java Application] C:\Program Files\Java\jdk-13.0.2\bin\javaw.exe (10-Sep-2020, 8:30:41 AM)
Who are you?
1. Citizen
2. Government Ofiicial
Enter any other number to exit
1
1. Registration
2. Viewing State Details
Enter any other number for exiting the sub-menu
1
Enter Name
Radhakrishna
Enter Aadhar Number
123456789
Citizen Registered Successfully
1. Registration
2. Viewing State Details
Enter any other number for exiting the sub-menu
3
Who are you?
1. Citizen
2. Government Ofiicial
Enter any other number to exit
2
1. Add new State
2. update an existing state
Enter any other number to exit
1
Enter Aadhar Number
123456789
```



```
Enter password
Govt@123
Enter State Name
AP
Enter State Population
1000000
State added successfully
1. Add new State
2. update an existing state
Enter any other number to exit
2
Enter Aadhar Number
123456789
Enter password
Govt@123
Enter state name
AP
Enter No of Cases
9000
Enter No of Deaths
2000
State Updated Successfully
1. Add new State
2. update an existing state
Enter any other number to exit
3
Who are you?
1. Citizen
2. Government Ofiicial
```



```
Enter any other number to exit
1
1. Registration
2. Viewing State Details
Enter any other number for exiting the sub-menu
2
Enter state name
AP
State Name: AP
Number of +ve Cases: 9000
Number of  Deaths=2000
Your State is in GREEN zone

1. Registration
2. Viewing State Details
Enter any other number for exiting the sub-menu
3
Who are you?
1. Citizen
2. Government Ofiicial
Enter any other number to exit
3
```