

**UNIVERZITET U BEOGRADU**

**GRAĐEVINSKI FAKULTET**

**KATEDRA ZA GEODEZIJU I GEOINFORMATIKU**



## **Projekat iz Gis Programiranja**

**Profesor:**

Dr Željko Cvijetinović, dipl. inž. geod.

**Asistent:**

Stevan Milić, master inž. geod.

**Student:**

Đorđe Subotić 1548/14

## **SADRŽAJ :**

<b>1. Opis Projekta .....</b>	<b>1</b>
<b>2. Opis funkcionalnosti .....</b>	<b>2</b>
<b>3. Odabrani delovi koda .....</b>	<b>10</b>

## 1. Opis Projekta

Cilj ovog projekta je prikaz mogućnosti programskog jezika Python i drugih open source tehnologija za izradu Web aplikacija. U okviru projekta korišćene su sledeće open source tehnologije :

- HTML
- CSS
- Bootstrap
- Open Layers 3
- Postgres/Postgis 9.5
- Apache Tomcat8
- Sqlite
- Geoserver
- Python Django Web Framework 1.9.5
- Git/Github
- Eclipse Java Neon IDE

Krajnji produkt ovog projekta je PyAlergies aplikacija. Ova aplikacija daje mogućnost vizuelizacije pacijenata obolelih od alergija na području grada Valjeva i okoline. Sama suština aplikacije je da pruži mogućnost prikaza obolelih pacijenata, kao i informacije o njima. Takođe uz aplikaciju je izrađen i blog čija je svrha publikovanje i razmena radova i iskustava stručnjaka u oblasti alergologije.

U okviru sistema postoje dve vrste korisnika :

- Alergolozi(imaju mogućnost prikaza pacijenata i pristup njihovim ličnim informacijama.
- Ostali(imaju mogućnost čitanja radova I pregled karte u WMS formatu)

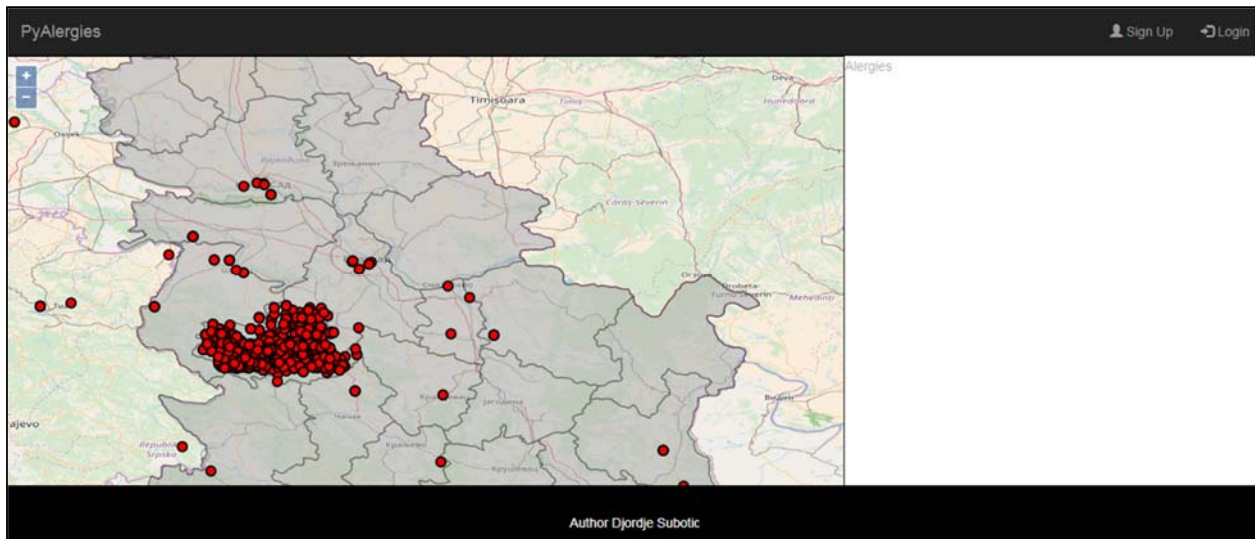
Sam kod aplikacije može se preuzeti sa github-a, na sledećem linku:

[https://github.com/GisDJordje/Gis\\_Projekat](https://github.com/GisDJordje/Gis_Projekat)

## 2. Opis Funkcionalnosti

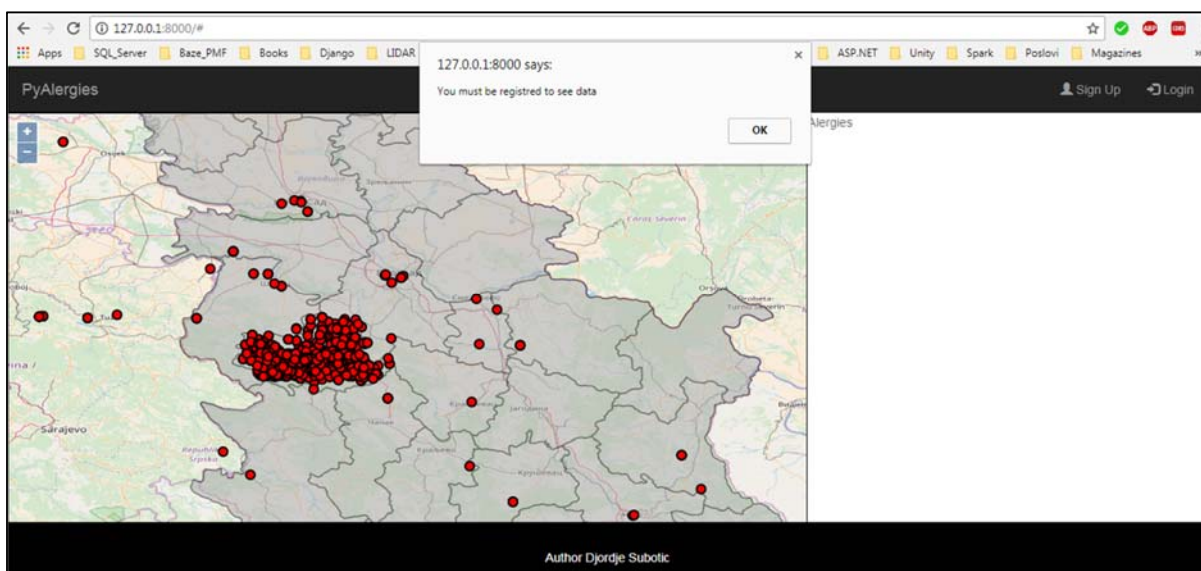
PyAlergies aplikacija je podeljena u tri modula :

- Account – služi za registraciju(Sign up i Login)
- Alergies\_Blog – služi za kreiranje, razmenu i objavu radova
- Alergies\_mapper – Vizuelizacija pacijenata



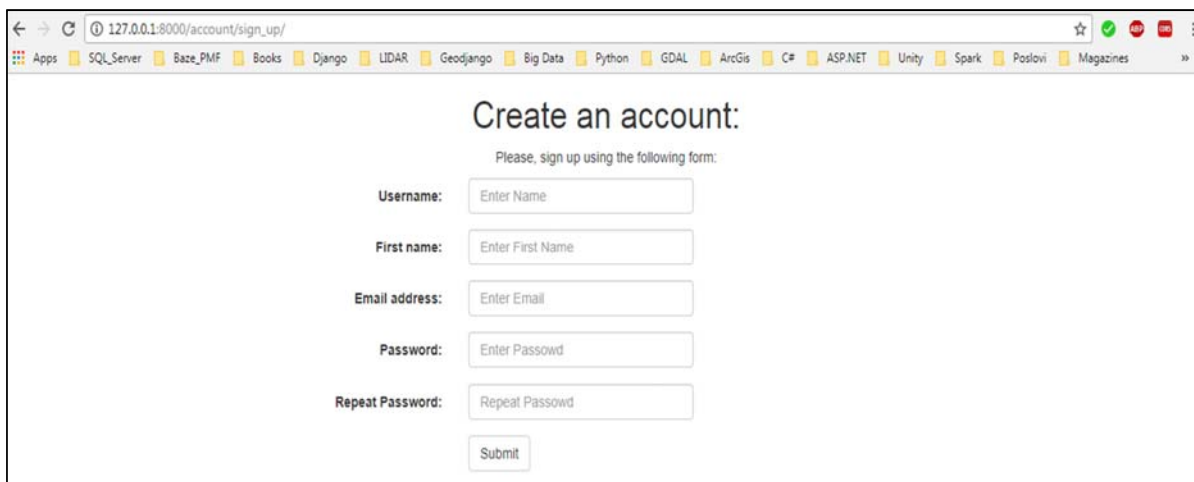
Slika 1 Home Page aplikacije

Korisniku se prikazuje karta na kojoj su tačkama predstavljeni pacijenti. Ukoliko korisnik nije ulogovan ili registrovan sistem mu ne dozvoljava pregled detaljnijih informacija klikom na mapu, što možemo videti na sledećoj slici.



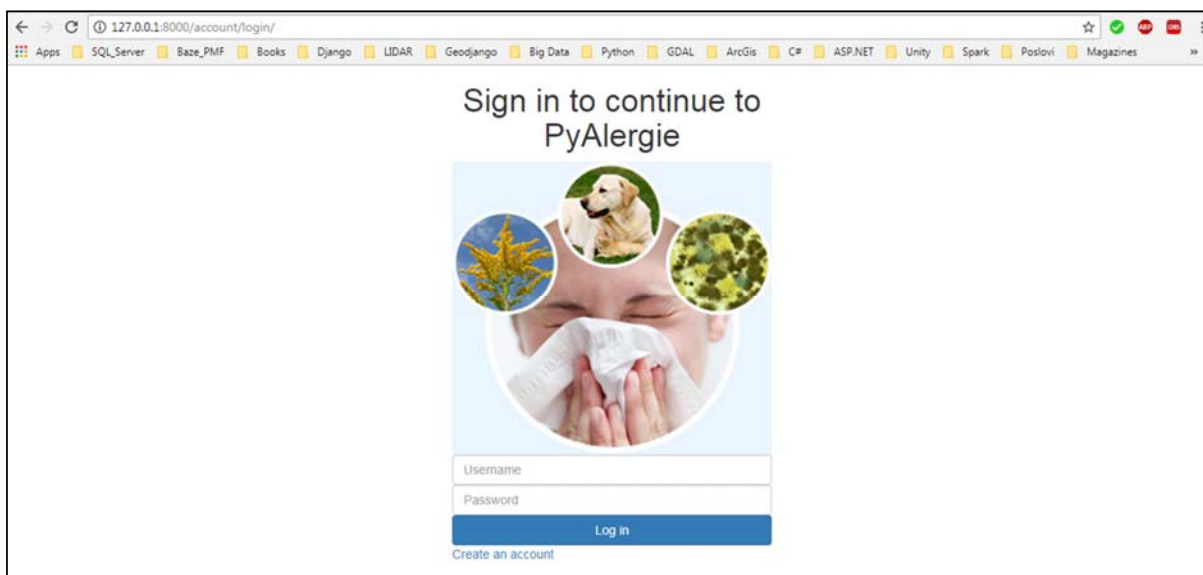
Slika 2 Prikaz pacijenata

Korisnici koji nisu registrovani klikom na Sign up polje dobijaju mogućnost registrovanja, što je prikazano na sledećoj slici.



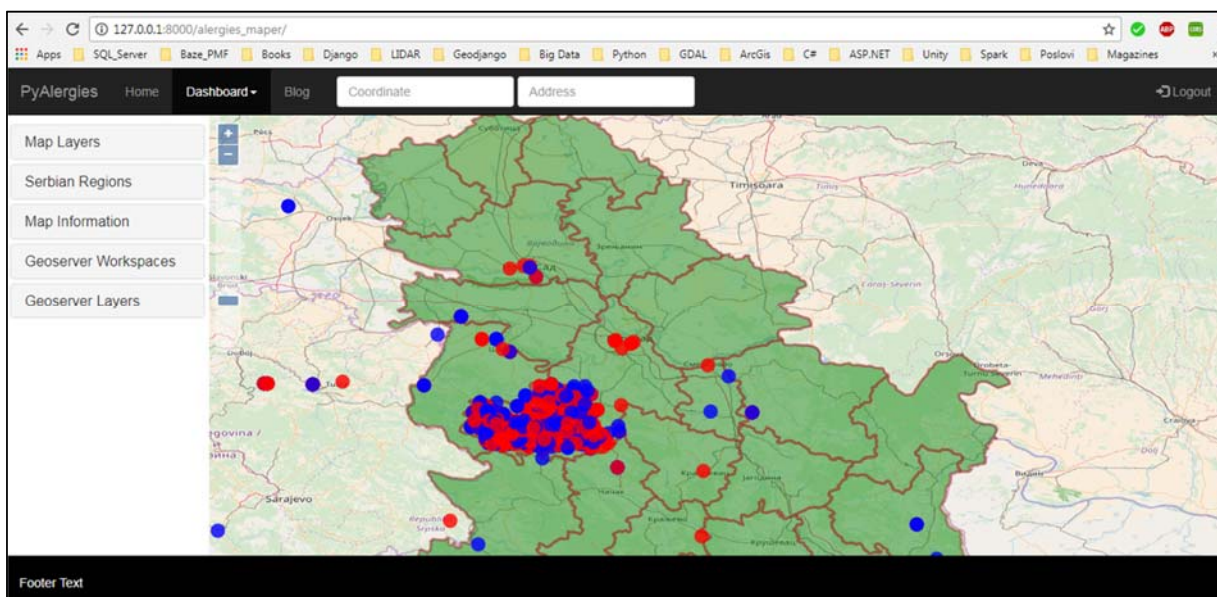
*Slika 3 Registracija korisnika*

Ukoliko korisnik ima već kreiran nalog, može se ulogovati klikom na dugme Login.



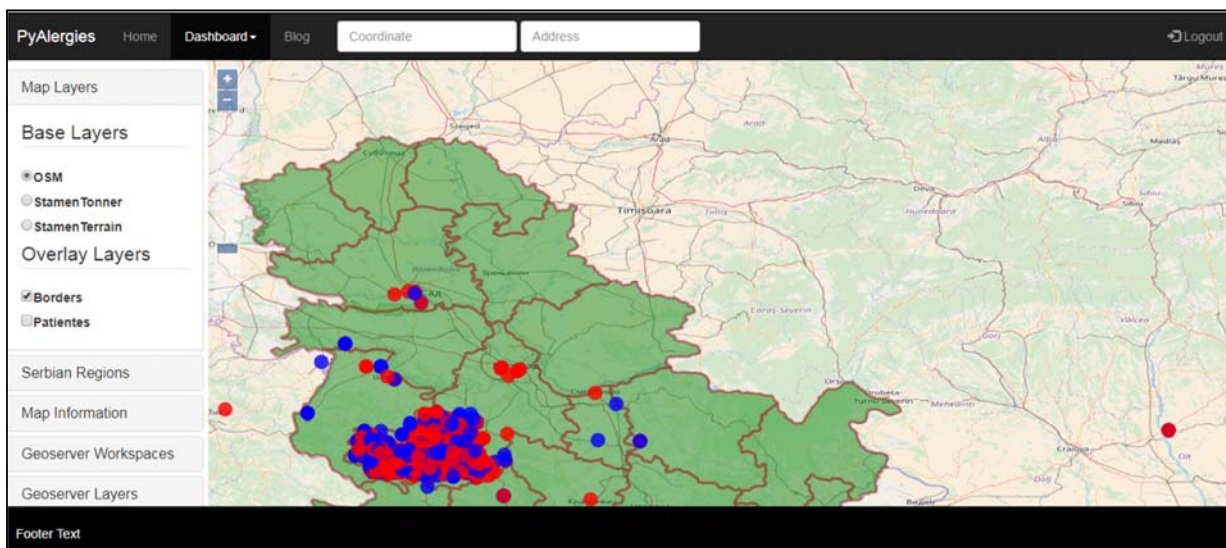
*Slika 4 Prijava korisnika na sistem*

Nakon unešenih validnih podataka korisnik se preusmerava na svoj profil(dashboard), gde mu se prikazuje mapa sa pacijentima.



Slika 5 Prikaz Dashboard-a

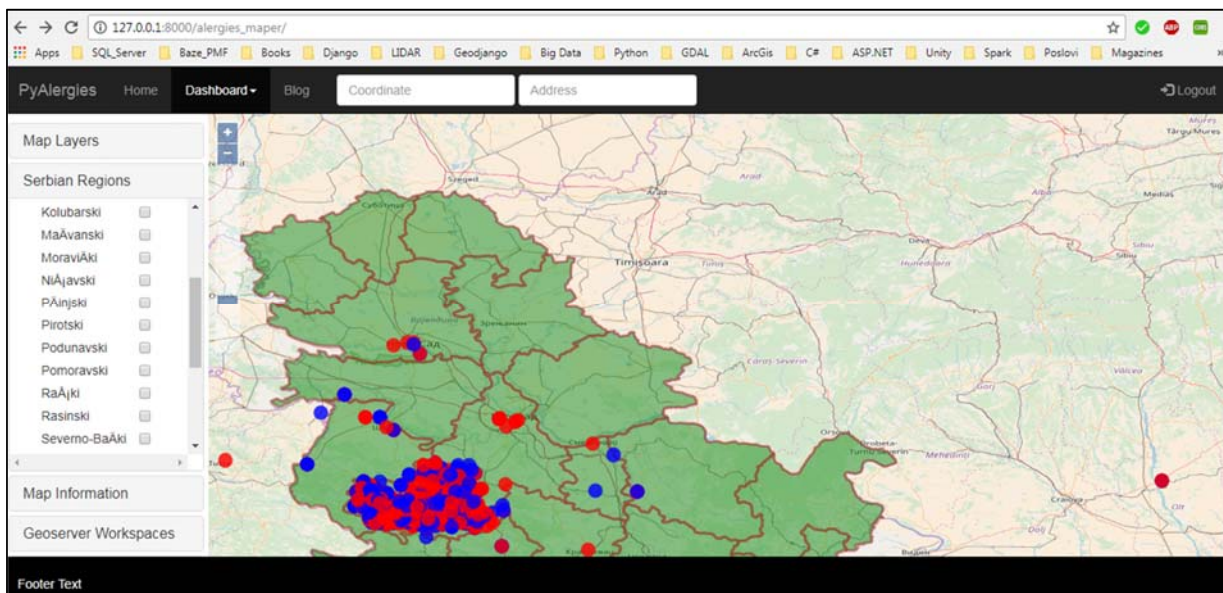
Na sledećim slikama prikazan je sadržaj svih Accordation tabova koji se nalaze levo od mape. Map Layers sadrži podatke o baznim i overlay layerima na karti.



Slika 6 Accordation Map Layer

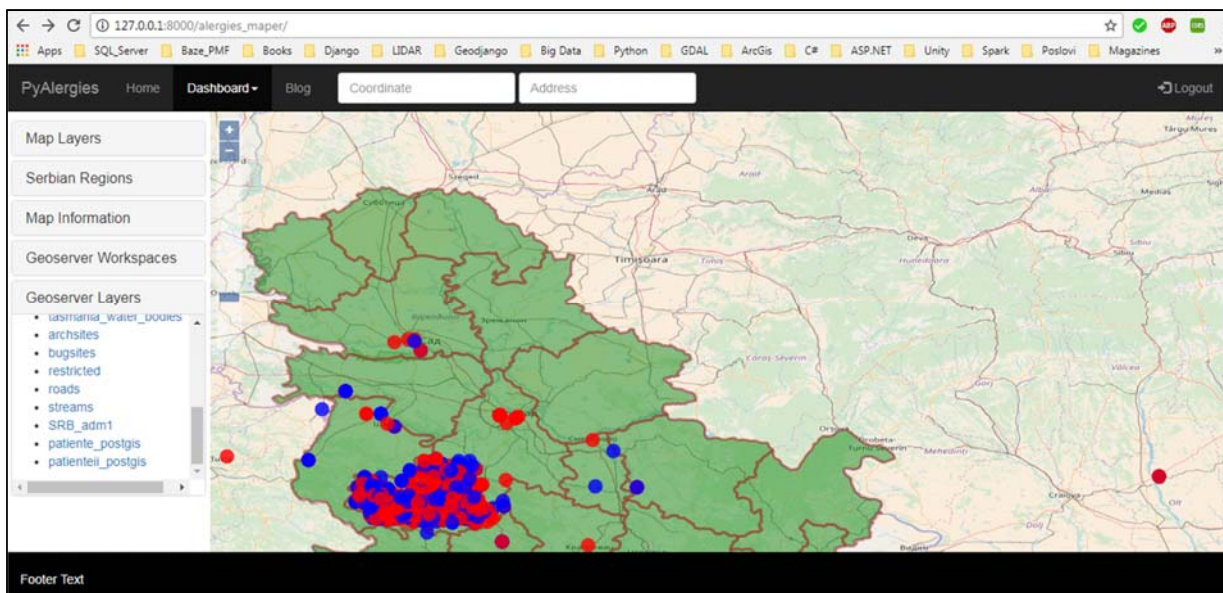


Serbian Regions predstavlja checkbox-ve pri čemu se svaki checkbox odnosi na jedan region. Checkboxovi su dinamički generisani pomoći Java Scripta, vršeći iteraciju kroz vektorski layer SRB\_adm1 (na sledećoj slici prestavljen u vidu poligona zelene boje).



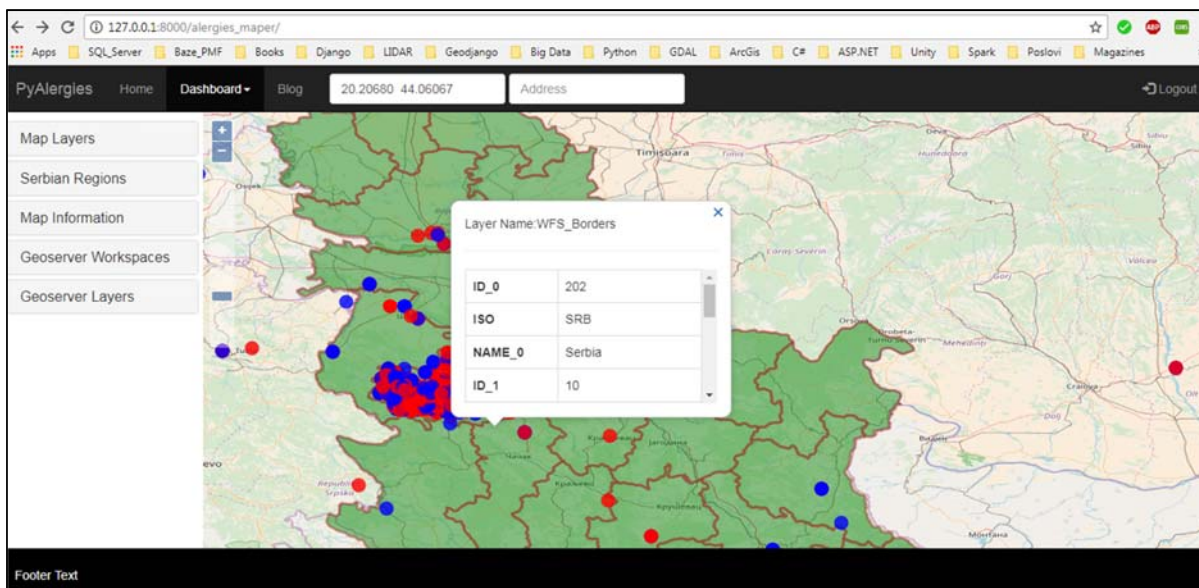
Slika 7 Accordation Serbian Regions Layer

Kartice Geoserver Workspaces i Geoserver Layers sadrže informacije o Geoserveru i dinamički su generisane koristeći Geoserver REST API. Na slici ispod možemo videti saržaj kartice Geoserver Layers. SRB\_adm1 se odnosi na okruge (zeleni poligoni na mapi) dok se laser patientii\_postgis odnosi na Postgis bazu sa pacijentima koja je postavljena kao layer na geoserver.



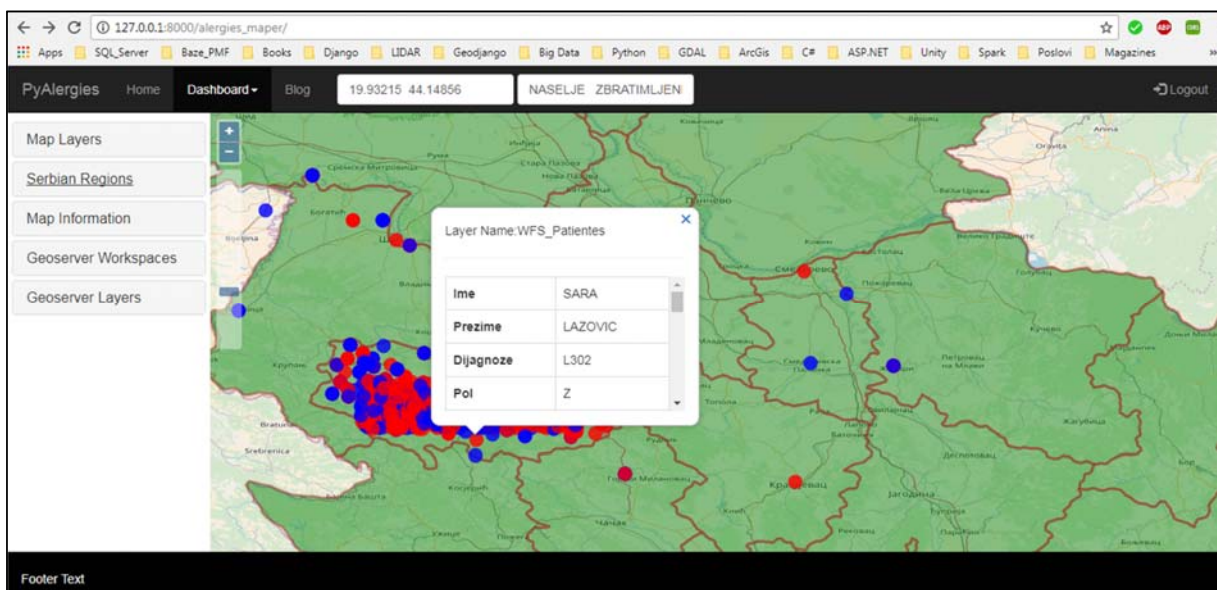
Slika 8 Accordation Geoserver Layer

Klikom na mapu prikazuju se informacije o određenim fetaure-ima koji se nalaze u layer-u na koji je kliknuto. Informacije o regionu(poligon).



Slika 9 Informacije o regionu

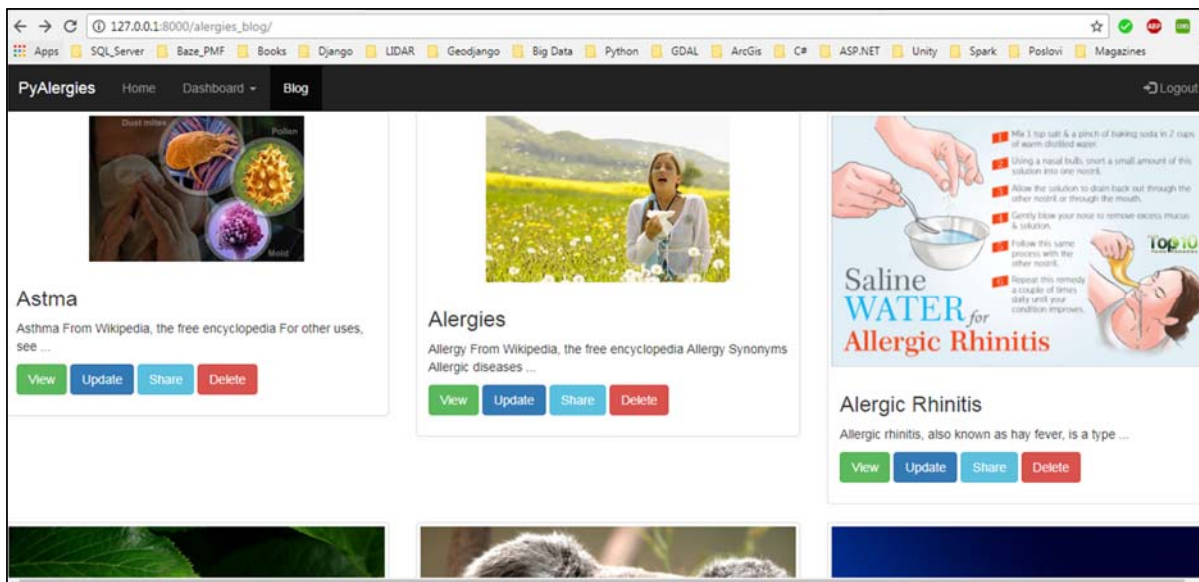
Informacije o određenom pacijentu(tačka).



Slika 10 Informacije o pacijentu

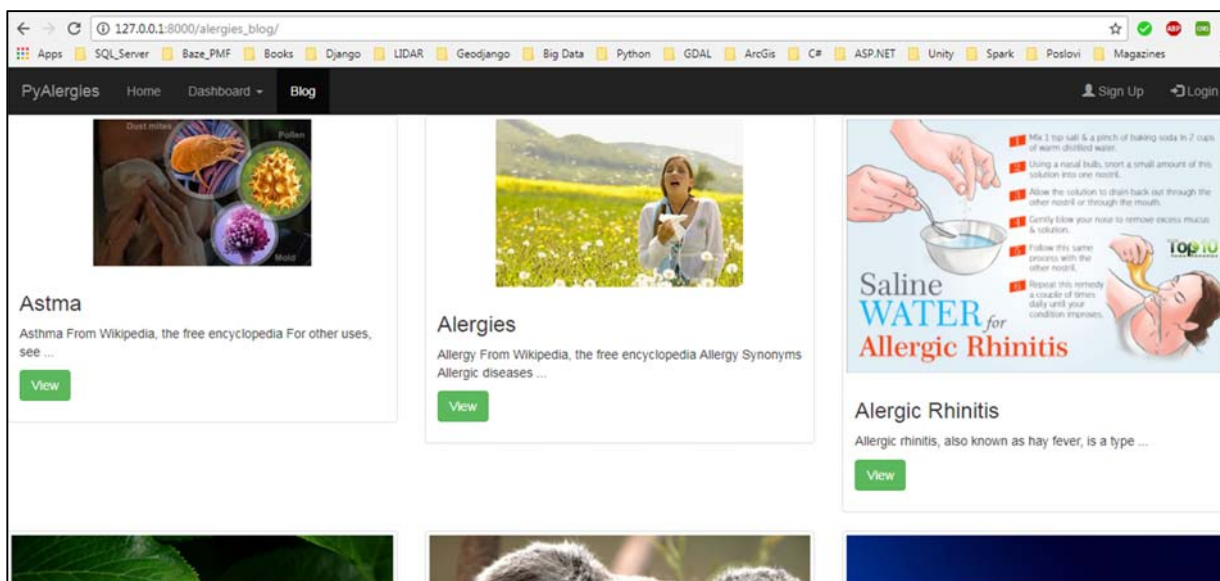


Na sledećoj slici možemo videti Home Page alergies blog aplikacije za korisnike koji su registrovani.



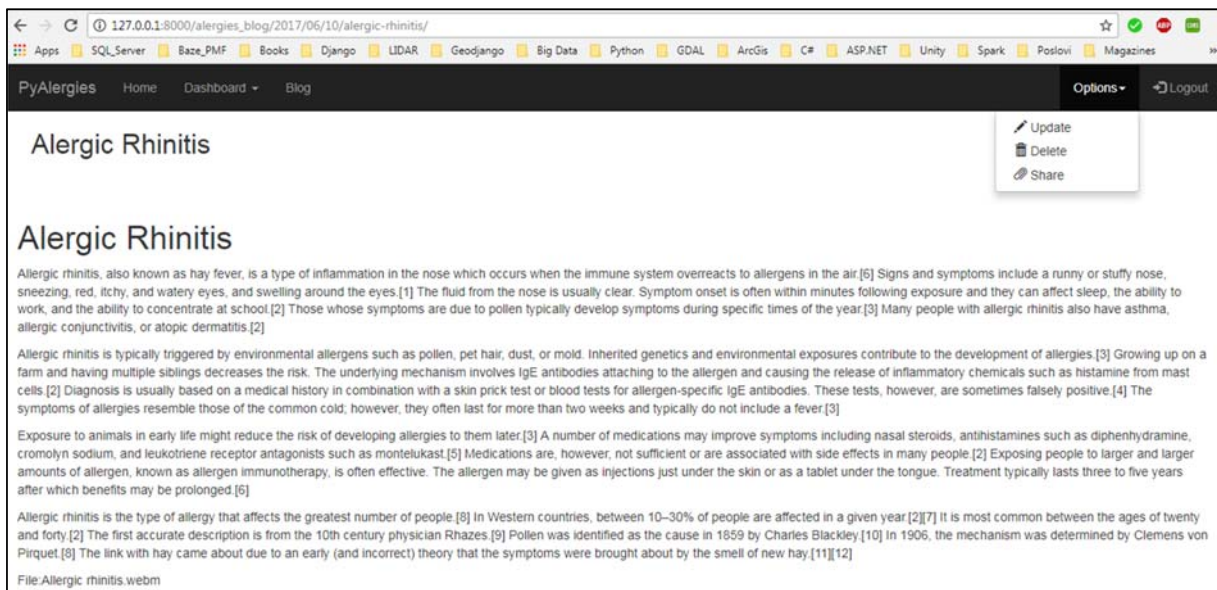
Slika 11 Home Page

Ukoliko korisnik nije registrovan, daje mu se jedino mogućnost čitanja sadržaja( dugme View).



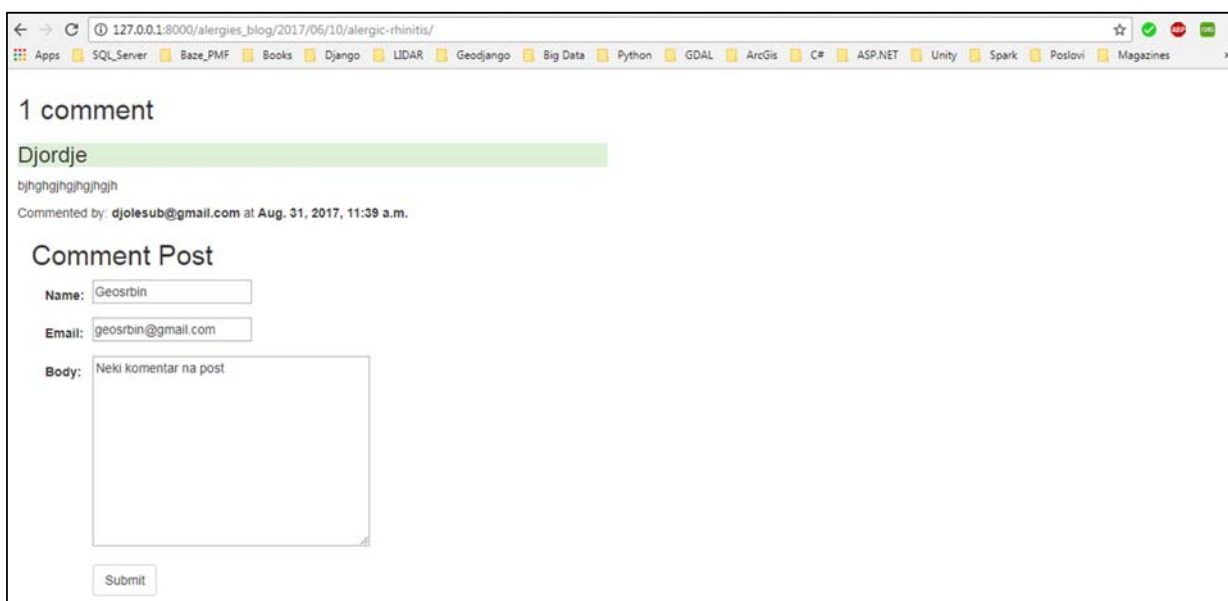
Slika 12 Prikaz sadržaja za neregistrovane korisnike

Klikom na dugme View korisnik dobija mogućnost detaljnijeg iščitavanja željenog post-a.



Slika 13 Detaljan sadržaj POST-ova

Na dnu stranice sa postoj korisniku se pruža mogućnost ostavljanja komentara.



Slika 14 Okruženje za komentarisanje

2 comments

**Geosrbini**

Neki komentar na post

Commented by: [geosrbini@gmail.com](#) at Sept. 13, 2017, 6:14 p.m.

**Djordje**

bjhghghghghghgh

Commented by: [djolesub@gmail.com](#) at Aug. 31, 2017, 11:39 a.m.

**Comment Post**

Name:

Email:

Body:

Slika 15 Okruženje za komentarisanje

Na sledećoj slici prikazana je tabela sa svim informacijama o pacijentima iz baze.

PyAlergies Home Dashboard Blog Coordinate Address Logout

Show 10 entries Search:

ID	IME	PREZIME	DIJAGNOZE	POL	VREME_RODZENJA	ADRESA	LATLON
1	SVETLANA	ACIMOVIC	Z015	Z	Jan. 29, 1981	LJIG, ZELENi VENAC 11	0101000020E61000008D93670355FA3340DD596034D023464
2	ANA	ADAMOVIC	Z015	Z	June 12, 1997	RADJEVO SELO, RADJEVO SELO	0101000020E6100000632B685A62DF3340D5642195BD24464
3	JELENA	ADAMOVIC	Z015	Z	July 7, 1996	LAJKOVAC, ZELEZNICKA 77	0101000020E6100000C3F4BD86E02C3440068195438B30464
4	LIDIJA	ADAMOVIC	Z015	Z	May 31, 1970	LJIG, SOLUNSKIH RATNIKA	0101000020E6100000394778D6C93C3440947C36BC6A1D464
5	JELENA	ADILOVIC	J450, L302, R600	Z	Dec. 23, 1983	VALJEVO, VLADE DANILOVICA 58	0101000020E610000031B9414871E333403A6865B10323464
6	MIROSLAVA	ADZIC	M350	Z	Feb. 10, 1960	SITARICE, ZASEOK ADZICI	0101000020E6100000DEE68D93C2BE33403F1D8F19A821464
7	ANDJELIJA	AKSENTIJEVIC	Z015	Z	July 8, 2009	VALJEVO, PETNICKA 24	0101000020E6100000EC67B114C9E733402074756CA920464
8	DRAGAN	AKSENTIJEVIC	J30, L302	M	May 30, 1999	LAJKOVAC, LAJKOVACKA PRUGA 61	0101000020E6100000AB2006BAF6A33340FBCA83F41427464
9	SVETLANA	ALAGIC	J30, J44	Z	Oct. 21, 1962	VALJEVO, PASTEROVA 31/A	0101000020E61000005868263E0E433409A3B9FA63B23464
10	ALEKSANDAR	ALEKSIC	J30, J45, L302	M	March 27, 1974	VALJEVO, KAMENICKA 15	0101000020E61000005A8DD81E73DE33406B7AF5961724464

Showing 1 to 10 of 2,500 entries Previous 1 2 3 4 5 ... 250 Next

Slika 16 Prikaz tabela sa informacijama o pacijentima

FILIJALA	ISPOSTAVA	KIJANJE	SEKRECIJA	ZAPUSEN_NOS	KASALJ	GUSENJE	SVIRANJE	RHINITIS_CHR	RHINITIS_NA	RHINITIS_A	ASTMA
VALJEVO	VALJEVO	Da	Da	Da	Da	Da	Da	Da	Ne	Da	Da
VALJEVO	VALJEVO	Ne	Ne	Ne	Da	Da	Da	Ne	Ne	Ne	Ne
VALJEVO	LAIKOVAC	Da	Da	Da	Ne	Ne	Ne	Da	Ne	Da	Ne
VALJEVO	LJIG	Da	Da	Da	Da	Da	Da	Da	Ne	Da	Ne
VALJEVO	VALJEVO	Da	Da	Da	Ne	Ne	Ne	Da	Ne	Da	Ne
VALJEVO	VALJEVO	Da	Da	ne	Da	Ne	Ne	Da	Ne	Da	Ne
VALJEVO	VALJEVO	Da	Da	Da	Da	Da	Ne	Da	Da	Da	Ne
VALJEVO	LAIKOVAC	Da	Da	Da	Ne	Ne	ne	Ne	Ne	Da	Ne
VALJEVO	VALJEVO	Da	Da	Da	Ne	Ne	Ne	Ne	Ne	Da	Ne
VALJEVO	VALJEVO	Da	Da	Da	Da	Da	Da	Da	Ne	Da	Da

Slika 17 Prikaz tabela sa informacijama o pacijentima

### 3. Odabrani delovi koda

Podaci koji su korišćeni u izradi aplikacije dobijeni su u exel formatu. Podaci su zatim smešteni u sqlite bazu. Potom je bilo potrebno izvršiti migraciju podataka iz sqlite baze u postgres/postgis bazu. Pošto su postojale samo adrese pacijenata bilo je potrebno izvršiti geokodiranje. Sve ovo se može videti u fajlu migration.py. Deo koda je prikazan ispod.

```
import sqlite3
import psycopg2
import geocoder
import random
```

```
from geopy.geocoders import Nominatim
from geopy import geocoders
```

```
#####
Postgres Database
#####
```

```
conn_postgres = psycopg2.connect("dbname=postgres user=postgres password=coperman")
cur_postgres = conn_postgres.cursor()
SQL = "SELECT * FROM alergies"
cur_postgres.execute(SQL)
data = cur_postgres.fetchall()
```



```
#####
                        Sqlite Database
#####
```

```
conn_sq = sqlite3.connect("AlergiesForSite.sqlite")
cur_sq = conn_sq.cursor()
cur_sq.execute("SELECT * FROM Table1")
data = cur_sq.fetchall()
```

```
coding = []
for i in range(len(data)):
```

```
    SQL1 = """INSERT INTO
alergies_site(ime,prezime,dijagnoze,pol,vreme_rodjenja,adresa,filijala,
              ispostava,kijanje,sekrecija,zapusen_nos,kasalj,gusenje,sviranje,rhinitis_chr,
              rhinitis_na,rhinitis_a,astma,konjuktivitis) VALUES """+str(data[i])
```

```
    try:
```

```
        adress = data[i][5]
        print("Adress is:",adress)
        adress.replace("Ž","Z")
        adress.replace("Š","S")
        adress.replace("Č","C")
        adress.replace("Ć","C")
        adress.replace("Đ","Dj")
```

```
        adress.replace("ž","z")
        adress.replace("š","s")
        adress.replace("č","c")
        adress.replace("ć","c")
        adress.replace("đ","dj")
```

```
        coding.append(adress)
    except UnicodeEncodeError as e:
```

```
        g = geocoder.google(adress)
        coding.append(g)
        cur_postgres.execute(SQL1)
conn_postgres.commit()
Kod za Sign Up View
```

```
def sign_up(request):
    if request.method == 'POST':
        user_form = UserRegistrationForm(request.POST)
        if user_form.is_valid():
            # Create a new user object but avoid saving it yet
            new_user = user_form.save(commit=False)
            # Set the chosen password
            new_user.set_password(user_form.cleaned_data['password'])
```

```

        # Save the User object
        new_user.save()
        return render(request, 'account/register_done.html', {'new_user': new_user})
    else:
        user_form = UserRegistrationForm()
    return render(request, 'account/register.html', {'form': user_form})

#####
                                Sign Up Forma Klasa
#####

class UserRegistrationForm(forms.ModelForm):
    password = forms.CharField(label="Password", widget=forms.PasswordInput)
    password2 = forms.CharField(label="Repeat Password", widget=forms.PasswordInput)

    class Meta:
        model = User
        fields = ('username', 'first_name', 'email')

    def clean_password2(self):
        cd = self.cleaned_data
        if cd['password'] != cd['password2']:
            raise forms.ValidationError("Password don't match")
        return cd['password2']

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
        for field in iter(self.fields):
            self.fields[field].widget.attrs.update({'class': 'form-control'})

#####
                                Prikaz svih pacijenata iz baze u tabeli View
#####

def get_all_pacientes(request):
    import psycopg2
    conn_postgres = psycopg2.connect("dbname=PyAlergies user=postgres password=coperman")
    cur_postgres = conn_postgres.cursor()
    SQL = "SELECT * FROM paciente_postgis"
    cur_postgres.execute(SQL)
    data = cur_postgres.fetchall()
    data = data
    section = 'People'
    names = ("id", "ime", "prezime", "dijagnoza", "pol", "vreme_rođenja", "adresa", "latlon", "filijala", "ispostava",
            "kijanje", "sekrecija", "zapusen_nos", "kasalj", "gusenje", "sviranje", "rhinitis_chr", "rhinitis_na", "rhinitis_a", "astma",
            "konjuktivitis"
            )
    """cur_postgres.execute("SELECT column_name FROM information_schema.columns WHERE table_name=
'paciente_postgis'")
    col_names = cur_postgres.fetchall() """
    return render(request, 'alergies_mapper/all_pacientes.html', {'data': data, 'names': names, 'section': section})

```

```
#####
View-s za preuzimanje informacija o workspace-ovima i layer-ima, korišćenjem GeoserverREST
API-ja.
#####
```

```
def get_workspace(request):
    import requests
    url = 'http://localhost:8090/geoserver/rest/workspaces'
    auth = ('admin', 'geoserver')
    headers = {'Content-Type': 'application/json', 'Accept': 'application/json'}
    r = requests.get(url, headers = headers, auth = auth)
    return HttpResponse(r, content_type="application/json")
```

```
def get_layers(request):
    import requests
    url = 'http://localhost:8090/geoserver/rest/layers'
    auth = ('admin', 'geoserver')
    headers = {'Content-Type': 'application/json', 'Accept': 'application/json'}
    r = requests.get(url, headers = headers, auth = auth)
    return HttpResponse(r, content_type="application/json")
```

```
#####
Java Script ajax kod za prikaz podataka dobijenih u funkcijama get_layers i get_workspace.
#####
```

```
<!-- Geoserver Rest Workspaces -->
<script>
$.ajax("{% url 'alergies_mapper:get_workspace' %}"),{
    type:"GET",
    headers : {"Accept":"application/json", "Content-Type":"application/json"}

}).done(function(res, status){
    alert(status)
    var data = res.workspaces.workspace
    var table = "<ul>";
    $.each(data, function(index,value){

        $.each(value,function(k,v){
            if(k == 'href'){
                var t = "<li>"+ "<a
href='"+ "'"+ "http://localhost:8090/geoserver/rest/workspaces/"+value['name']+">"+value['name']+"</a>"+ "</li>";
                table+=t;
            }
        })

    });
    table+="</ul>";
    $('#workspaces').html(table);

}).fail(function(s,v){
    alert(v)
});
```

```

<!-- Geoserver Rest Layers -->
$.ajax("{% url 'alergies_mapper:get_layers' %}",{
    type:"GET",
    headers : {"Accept":"application/json","Content-Type":"application/json"}

}).done(function(res, status){

    var data = res.layers.layer
    var table = "<ul>";
    $.each(data, function(index,value){

        $.each(value,function(k,v){
            if(k == 'href'){
                var t = "<li>"+ "<a
href="+""+"http://localhost:8090/geoserver/rest/layers/"+value['name']+""+">"+value['name']+ "</a>"+ "</li>";
                table+=t;
            }

        })
    })
}

```

```

#####
                                Open Layers stil za pacijente
#####

```

```

// Styling Patientes With Function -->
var style_Patientes = (function(){
    var woman = [new ol.style.Style({
        image: new ol.style.Circle({
            fill: new ol.style.Fill({ color: 'red'}),
            radius: 8
        }),

    }]);

    var man = [new ol.style.Style({
        image: new ol.style.Circle({
            fill: new ol.style.Fill({ color: 'blue'}),
            radius: 8
        })
    }]);

    return function(feature, resolution){
        if(feature.get('pol') == "Z"){
            return woman;
        }else {
            return man;
        }
    };
})();

```



```
#####
Dinamičko kreiranje checkboxova za svaki region(poligon) u layer-u SRB_adm1.
#####
```

```
$.ajax("http://localhost:8090/geoserver/wfs",{
    type: "GET",
    data: {
        service: "WFS",
        version: "1.0.0",
        request: 'GetFeature',
        typename: 'PythonAlergies:SRB_adm1',
        outputFormat: "JSON"
    }
}).done(function(data,b,c){

    format.readFeatures(data);
    //console.log(format);

    $.each(data,function(index, value){
        console.log(value);
        if(index == "features"){
            $.each(value, function(k,v){
                console.log(v.properties.NAME_1);
                var collapse = document.querySelector('#regionsPanel');
                var input = document.createElement('input');
                input.type = 'checkbox';
                input.name = 'regions';
                input.id = v.properties.NAME_1;
                input.setAttribute('class','col-md-2');
                var label = document.createElement('label');
                label.setAttribute('class','col-md-10');

                label.appendChild(document.createTextNode(v.properties.NAME_1));
                label.appendChild(input);

                var div = document.createElement('div');
                div.setAttribute('class','checkbox col-md-12');
                div.appendChild(label);
                div.appendChild(input);

                collapse.appendChild(div);

            })
        }
    })
});
```