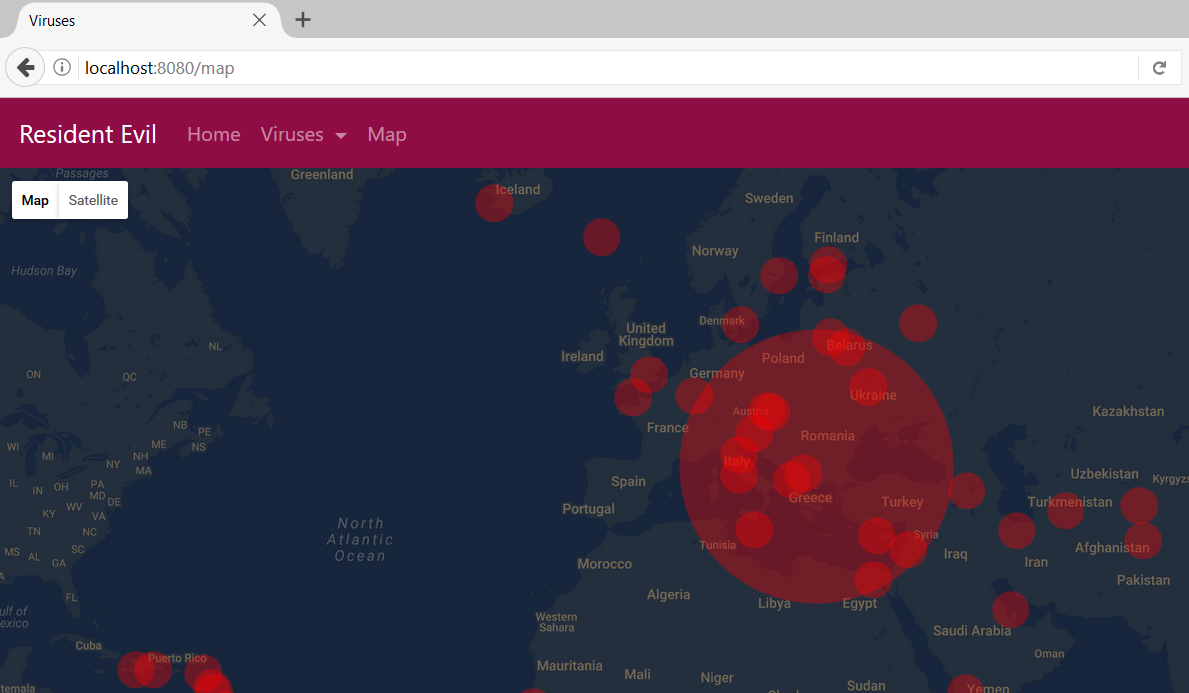
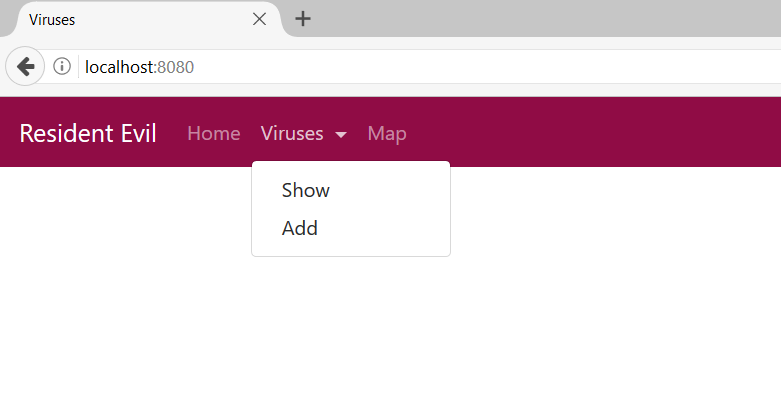
# Resident Evil Part 1

Resident Evil is a system that spreads viruses across the world. Your task is to build one. You have to use **Spring Boot** and **Thymeleaf** for that purpose. Your application needs to have **three** pages.



## Views

You would need couple of views. Separate the **menu and the forms** in html files and include them by using **th:replace**. Use a design that you find appropriate. The examples use **Bootstrap 4**. Here is link to find good-looking examples: [click](https://v4-alpha.getbootstrap.com/components/forms/)



* **Home**
  + Entry point of the application
* **Viruses**
  + All the viruses are shown here
  + You can add, delete and edit them
* **Map**
  + Shows a Google Maps with all the viruses

## Data Entities

Create the required **entities**. Use the appropriate **data types**.

* **Virus**
  + Name
    - Not Blank
    - Between 3 and 10 symbols
  + Description
    - Represented as Text in the database
    - Not Blank
    - Between 5 and 100 symbols
  + Side Effects
    - Up to 50 symbols
  + Creator
    - Should contains Corp or corp
  + Is Deadly
  + Is Curable
  + Mutation
    - ZOMBIE or
    - T\_078\_TYRANT or
    - GIANT\_SPIDER
    - Not Null
  + Turnover Rate
    - Between 0 and 100
  + Hours Until Turn (to a mutation)
    - Between 1 and 12
  + Magnitude
    - Low, Medium or High
  + Rreleased On
    - Should not be in the past
  + Capitals
    - Many **Viruses** are spread in many **Capitals**
* **Capitals**
  + Name
  + Latitude
  + Longitude

## Establish the Back-End

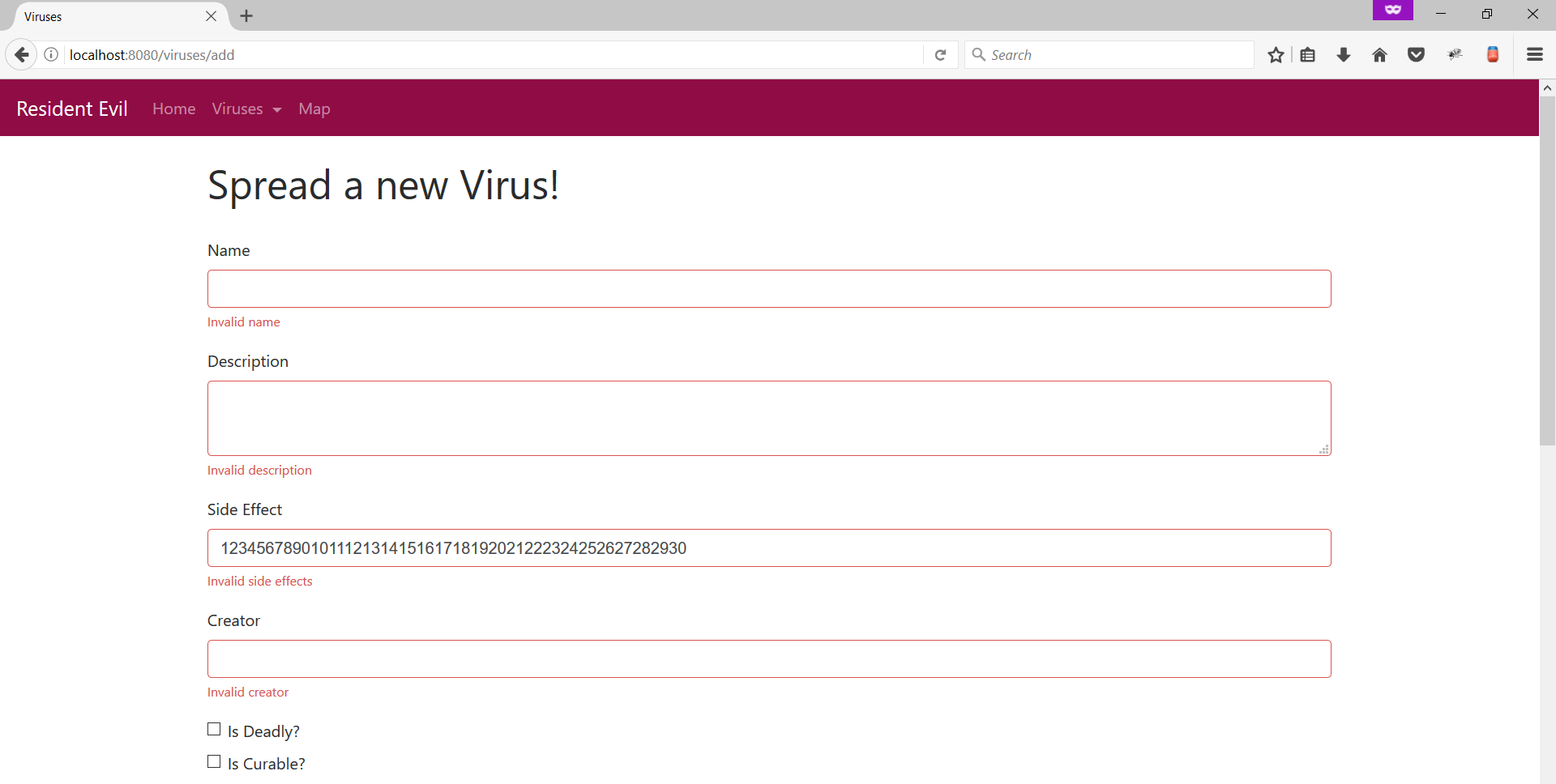
Create the required:

* **Entities**
* **Models**
* **Repositories**
* **Services**
* **Controllers**

Load the capitals by the provided [**SQL**](http://svn.softuni.org/admin/svn/web-development/Jan-2017/Java-Frameworks-Spring/03.%20Java-Frameworks-Thymeleaf-Engine)

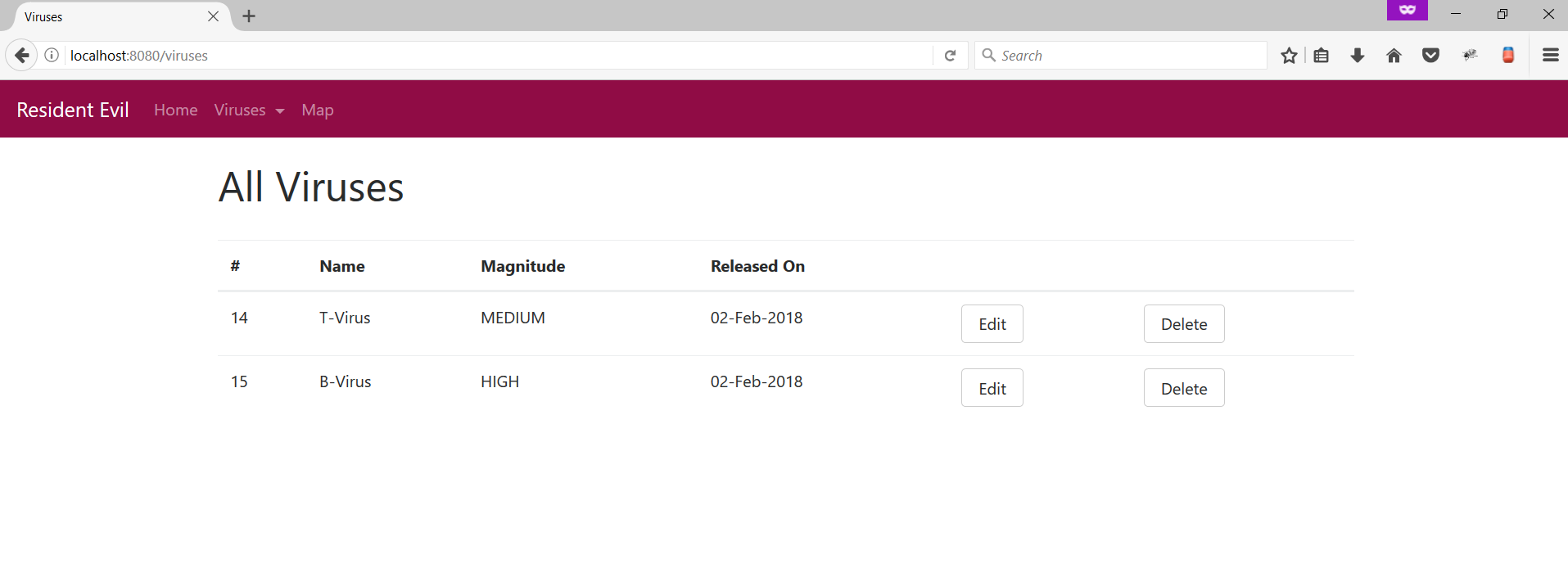
## Add Viruses

Create a functionality to **add** Viruses. Make the necessary **validations**. Create a custom annotation to validate the Release Date.



## Show Viruses

Create a functionality to **show** all the available viruses.



## Edit Viruses

Create a functionality to **edit** Viruses. You should be able to edit everything **except the release date**. Make the necessary **validations.**

## Delete Viruses

Create a functionality to **delete** Viruses

## \*\*\*Google Maps

Do you want to see the damage you did to the world? Add Google Maps API to enjoy your work.

To make it a bit easier here is the code that adds the map.

|  |
| --- |
| **map.html** |
| <!DOCTYPE **html**> <**html lang="en" xmlns:th="http://www.thymeleaf.org"**> <**head**>  <**meta charset="UTF-8"**/>  <**title**>Viruses</**title**>  <**link rel="stylesheet" href="../static/bootstrap/css/bootstrap.min.css"  th:href="@{bootstrap/css/bootstrap.min.css}"**/>  <**style**>  */\* Always set the map height explicitly to define the size of the div  \* element that contains the map. \*/* **#map** {  **height**: 100%;  }   */\* Optional: Makes the sample page fill the window. \*/* **html**, **body**, **main** {  **height**: 100%;  **margin**: 0;  **padding**: 0;  }  </**style**>  </**head**> <**body**> <**header th:replace="fragments/parts::menu"**> </**header**>  <**main**>  <**div id="map"**></**div**> </**main**> <**script th:src="@{jquery/jquery.min.js}"**></**script**> <**script th:src="@{bootstrap/js/bootstrap.min.js}"**></**script**> <**script th:src="@{scripts/maps.js}"**></**script**> <**script th:inline="javascript"**> **var *geoJson*** = [[${geoJson}]];  **var *data*** = ***JSON***.parse(***geoJson***);  **console**.log(***data***);   **var *map***;   **function** *initMap*() {  map = **new** google.maps.Map(**document**.getElementById(**'map'**), {  **center**: {**lat**: 23, **lng**: 42},  **zoom**: 3,  **styles**: [  {**elementType**: **'geometry'**, **stylers**: [{**color**: **'#242f3e'**}]},  {**elementType**: **'labels.text.stroke'**, **stylers**: [{**color**: **'#242f3e'**}]},  {**elementType**: **'labels.text.fill'**, **stylers**: [{**color**: **'#746855'**}]},  {  **featureType**: **'administrative.locality'**,  **elementType**: **'labels.text.fill'**,  **stylers**: [{**color**: **'#d59563'**}]  },  {  **featureType**: **'poi'**,  **elementType**: **'labels.text.fill'**,  **stylers**: [{**color**: **'#d59563'**}]  },  {  **featureType**: **'poi.park'**,  **elementType**: **'geometry'**,  **stylers**: [{**color**: **'#263c3f'**}]  },  {  **featureType**: **'poi.park'**,  **elementType**: **'labels.text.fill'**,  **stylers**: [{**color**: **'#6b9a76'**}]  },  {  **featureType**: **'road'**,  **elementType**: **'geometry'**,  **stylers**: [{**color**: **'#38414e'**}]  },  {  **featureType**: **'road'**,  **elementType**: **'geometry.stroke'**,  **stylers**: [{**color**: **'#212a37'**}]  },  {  **featureType**: **'road'**,  **elementType**: **'labels.text.fill'**,  **stylers**: [{**color**: **'#9ca5b3'**}]  },  {  **featureType**: **'road.highway'**,  **elementType**: **'geometry'**,  **stylers**: [{**color**: **'#746855'**}]  },  {  **featureType**: **'road.highway'**,  **elementType**: **'geometry.stroke'**,  **stylers**: [{**color**: **'#1f2835'**}]  },  {  **featureType**: **'road.highway'**,  **elementType**: **'labels.text.fill'**,  **stylers**: [{**color**: **'#f3d19c'**}]  },  {  **featureType**: **'transit'**,  **elementType**: **'geometry'**,  **stylers**: [{**color**: **'#2f3948'**}]  },  {  **featureType**: **'transit.station'**,  **elementType**: **'labels.text.fill'**,  **stylers**: [{**color**: **'#d59563'**}]  },  {  **featureType**: **'water'**,  **elementType**: **'geometry'**,  **stylers**: [{**color**: **'#17263c'**}]  },  {  **featureType**: **'water'**,  **elementType**: **'labels.text.fill'**,  **stylers**: [{**color**: **'#515c6d'**}]  },  {  **featureType**: **'water'**,  **elementType**: **'labels.text.stroke'**,  **stylers**: [{**color**: **'#17263c'**}]  }  ]   });   map.**data**.addGeoJson(***data***);   map.**data**.setStyle(**function** (feature) {  **var** mag = ***Math***.exp(parseFloat(feature.getProperty(**'mag'**))) \* 0.1;  **var** color = feature.getProperty(**'color'**);  **return** */\*\** ***@type*** *{google.maps.Data.StyleOptions} \*/*({  **icon**: {  **path**: google.maps.SymbolPath.CIRCLE,  **scale**: mag,  **fillColor**: color,  **fillOpacity**: 0.35,  **strokeWeight**: 0  }  });  });  }  </**script**>  <**script async="true" defer="true"  src="https://maps.googleapis.com/maps/api/js?key=YOUR\_API\_KEY&amp;callback=initMap"**> </**script**> </**body**> </**html**> |

#### 8.1 Add the view to your project

#### 8.2 Get Google API Key

Go to [Google Maps API](https://developers.google.com/maps/documentation/javascript/get-api-key) and get a key for free. Follow the instructions there.

#### 8.3 Add the Key to key=YOUR\_API\_KEY

#### 8.4 Send the data to JS

Get all viruses and their capitals. Generate a string with the required data and send it the JavaScript file. The format should like this.

|  |
| --- |
| **Geo Json** |
| {  **"type"**: **"FeatureCollection"**,  **"features"**: [{  **"type"**: **"Feature"**,  **"properties"**: {  **"mag"**: 5,  **"color"**: **"#f00"** },  **"geometry"**: {  **"type"**: **"Point"**,  **"coordinates"**: [  43.939999,  12.450000  ]  }  },  …  ]  }; |

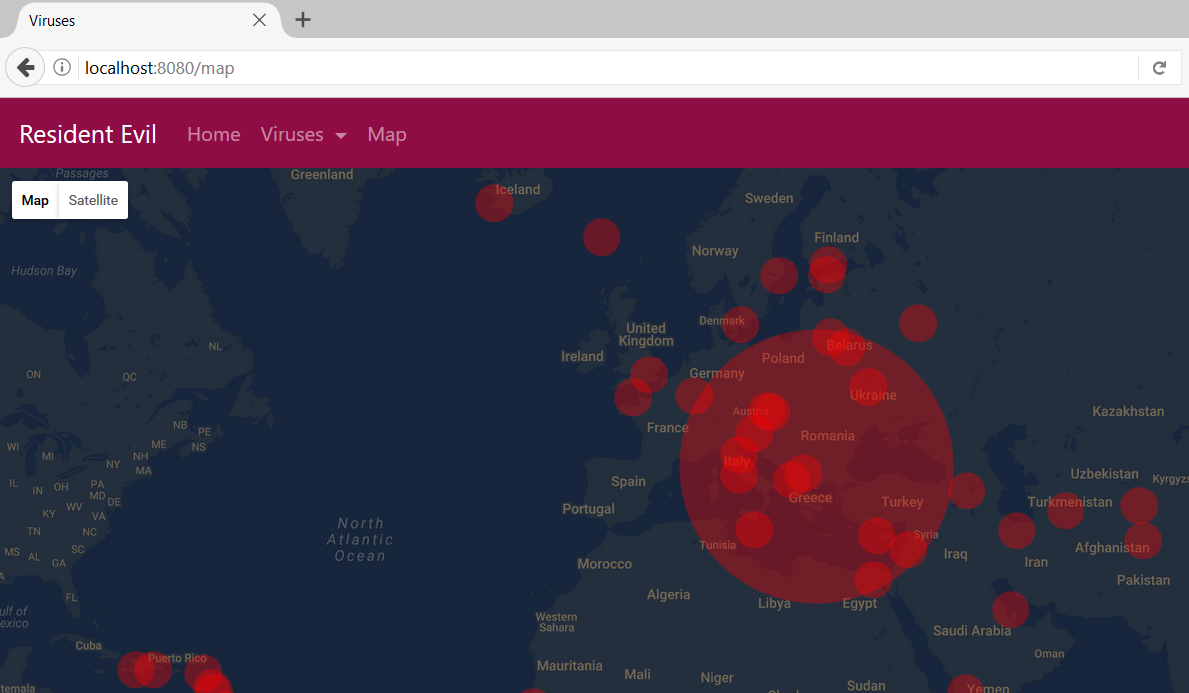
Weird, right? It is called **GeoJson** and it is massively used to **show geo data**. You have to create a **string** that looks like this with all the data. Here is an example of the required **controller**.

|  |
| --- |
| **MapController.java** |
| @Controller **public class** MapController {   @Autowired  **private** VirusService **virusService**;   @GetMapping(**"/map"**)  **public** String getMapPage(Model model){  String geoJson = **this**.**virusService**.findAllMapViruses();  model.addAttribute(**"geoJson"**, geoJson);  **return "map"**;  } } |

The data is **passed** to JS like this:

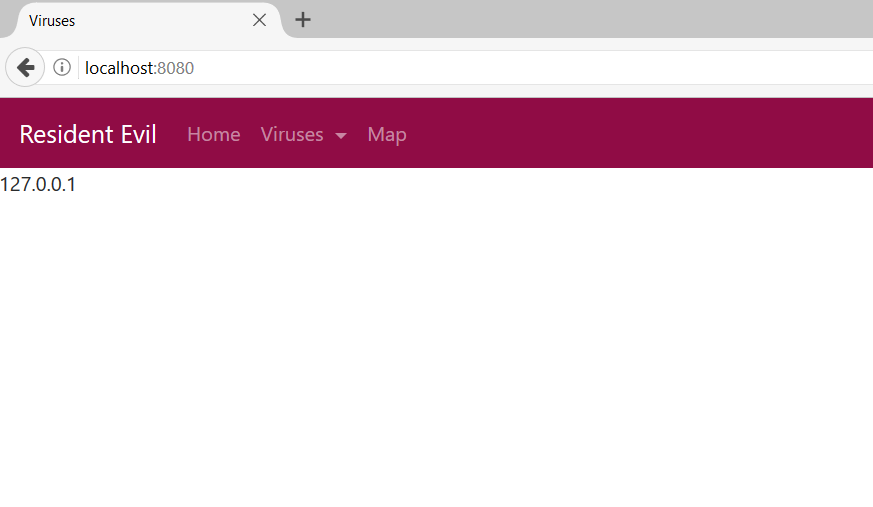
|  |
| --- |
| **JavaScript** |
| **var *geoJson*** = [[${geoJson}]]; |

You have the last line in your **map.html**. Here is the final result:

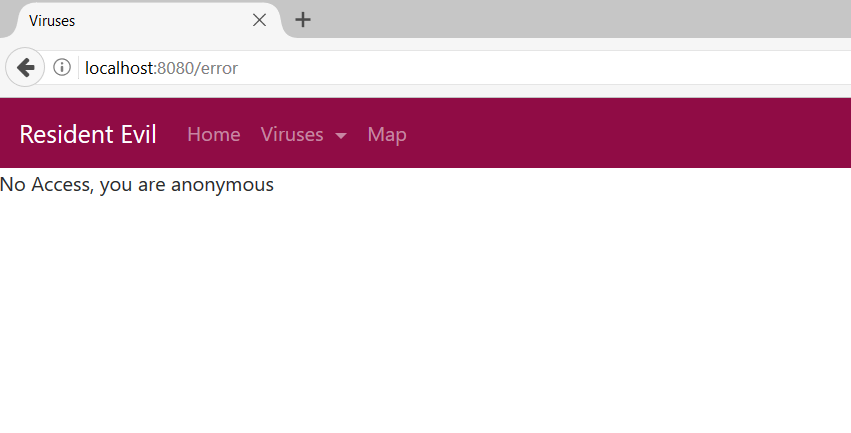


## Get IP Address

Whenever someone access your **application** print his **IP**. Use **Interceptor**! If the user uses anonymous browser redirect to **/error**.



.



## Counter

Create an **interceptor** that **counts** the number of time your **home** page is **visited**.

