

# Geoserver Lab

2021101107

## WORKSPACE CREATION

We have created a new workspace called- lab 4 in the geoserver and set it to 'default workspace'.

The screenshot shows the 'New Workspace' configuration interface in the GeoServer web application. The left sidebar contains links for 'About & Status', 'Data' (Layer Preview, Workspaces, Stores, Layers, Layer Groups, Styles), 'Services' (WMTS, WCS, WFS, WMS), and 'Settings' (Global, Image Processing, Raster Access). The main right panel has a title 'New Workspace' and a subtitle 'Configure a new workspace'. It features two tabs: 'Basic Info' (selected) and 'Security'. Under 'Basic Info', there are fields for 'Name' (set to 'lab4'), 'Namespace URI' (set to 'lab4'), and checkboxes for 'Default Workspace' (checked) and 'Isolated Workspace'. At the bottom are 'Save' and 'Cancel' buttons.

Next we are going to 'store' → create new store → add the above details and the credentials of postGIS to establish connection between geoserver and our last lab's database 'demo'.

Field 'Data Source Name' is required.

## New Vector Data Source

Add a new vector data source

PostGIS  
PostGIS Database

### Basic Store Info

Workspace \*

lab4

Data Source Name \*

lsi

Description

Enabled

Auto disable on connection failure

### Connection Parameters

host \*

localhost

port \*

5432

database

demo

schema

public

user \*

postgres

passwd

Save    Apply    Cancel

After creating a the store → go to layers → add new layer → choose the store from dropdown menu → choose "lab4:lsi" → we get all the layers.

Now we are publishing the layer called 'ind\_states' → click publish → scroll down to bounding boxes → click on 'compute from data' and 'compute from native bounds' and save → layer published. (have done for all three layers similarly)

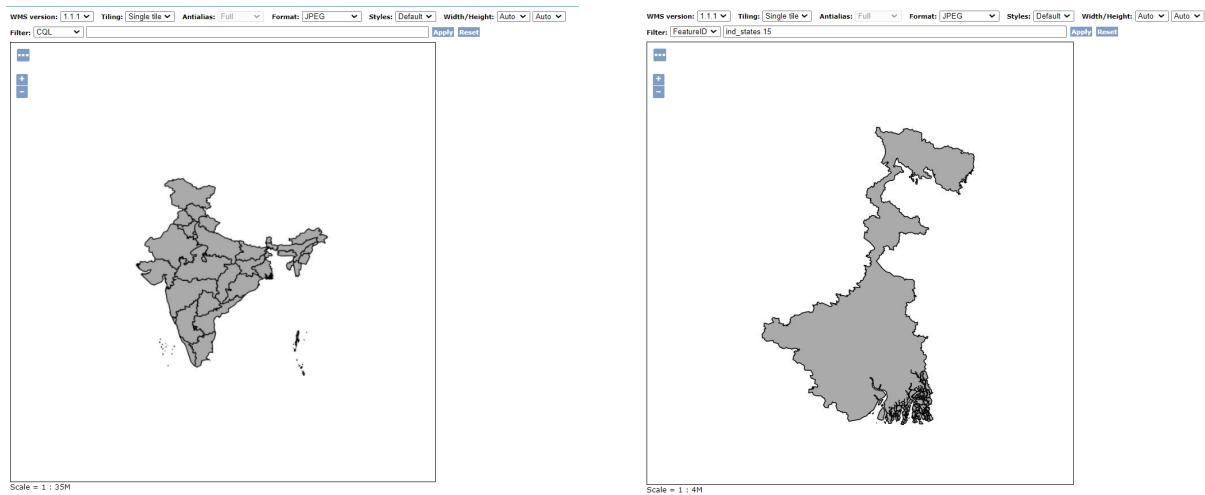
Go to layers preview and see the layers.

Type	Title	Name	Store	Enabled	Native SRS
<input type="checkbox"/>	ind_points	lab4:ind_points	lsi	✓	EPSG:4326
<input type="checkbox"/>	ind_states	lab4:ind_states	lsi	✓	EPSG:4326
<input type="checkbox"/>	water	lab4:water	lsi	✓	EPSG:4326

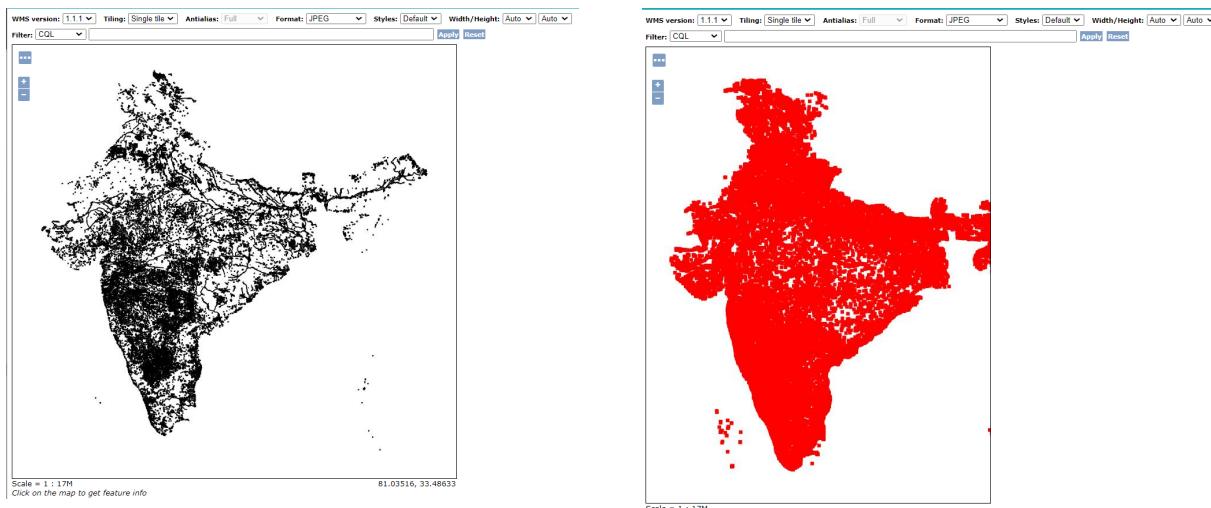
  

Type	Title	Name	Common Formats	All Formats
<input checked="" type="radio"/>	ind_points	lab4:ind_points	OpenLayers GML KML	Select one
<input checked="" type="radio"/>	ind_states	lab4:ind_states	OpenLayers GML KML	Select one
<input checked="" type="radio"/>	water	lab4:water	OpenLayers GML KML	Select one

Now go to layers preview → openLayers for ind\_states.



Layer preview for water (left) and ind\_points (right) layer:



## PARAMETER BASED LAYER

Created a new SQL view → on the edit layer page → scroll down to bounding boxes → click on 'compute from data' and 'compute from native bounds' and save.

**Create new SQL view**

Define a new SQL view and configure its identified and geometry columns

View Name: sample\_query

SQL statement:

```
select * from ind_states where name ilike '%n'
```

SQL view parameters

Guess parameters from SQL   Add new parameter   Remove selected

Name	Default value	Validation regular expression
n	telangana	^[\w\d\\$]+\$

Escape special SQL characters

Attributes

Refresh  guess geometry type and srid

Name	Type	SRID	Identifier
id	Integer		<input type="checkbox"/>
geom	Multipolygon	4326	<input type="checkbox"/>
sid	BigDecimal		<input type="checkbox"/>
name	String		<input type="checkbox"/>
area	BigDecimal		<input type="checkbox"/>

**Bounding Boxes**

Native Bounding Box

Min X	Min Y	Max X	Max Y
77.23899106648457	15.833175518135704	81.32534067420497	19.91646202070575

Compute from data  
Compute from SRS bounds

Lat/Lon Bounding Box

Min X	Min Y	Max X	Max Y
77.23899106648457	15.833175518135704	81.32534067420497	19.91646202070575

Compute from native bounds

**Curved geometries control**

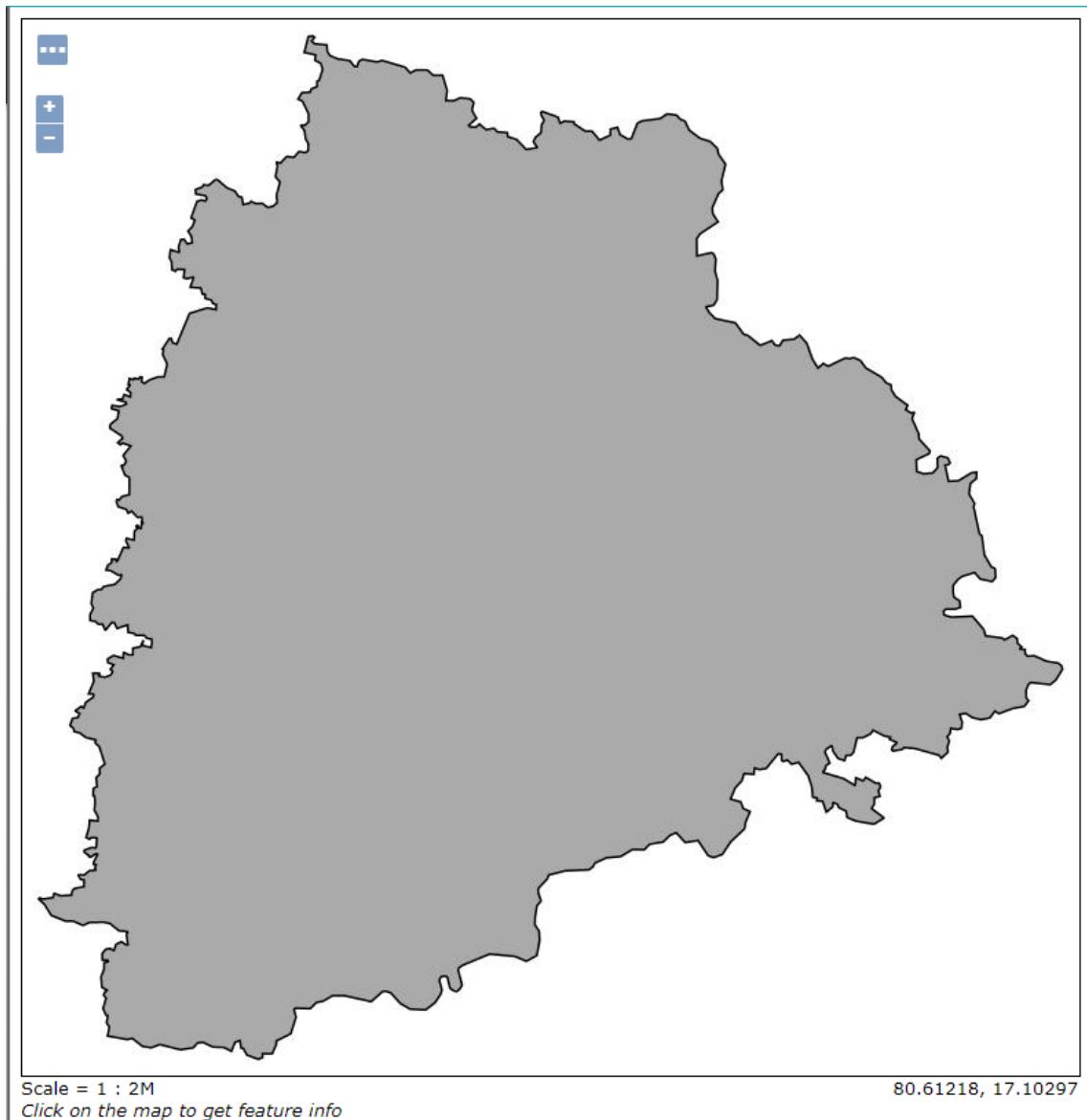
Linear geometries can contain circular arcs  
Linearization tolerance (useful only if your data contains curved geometries)

**Feature Type Details**

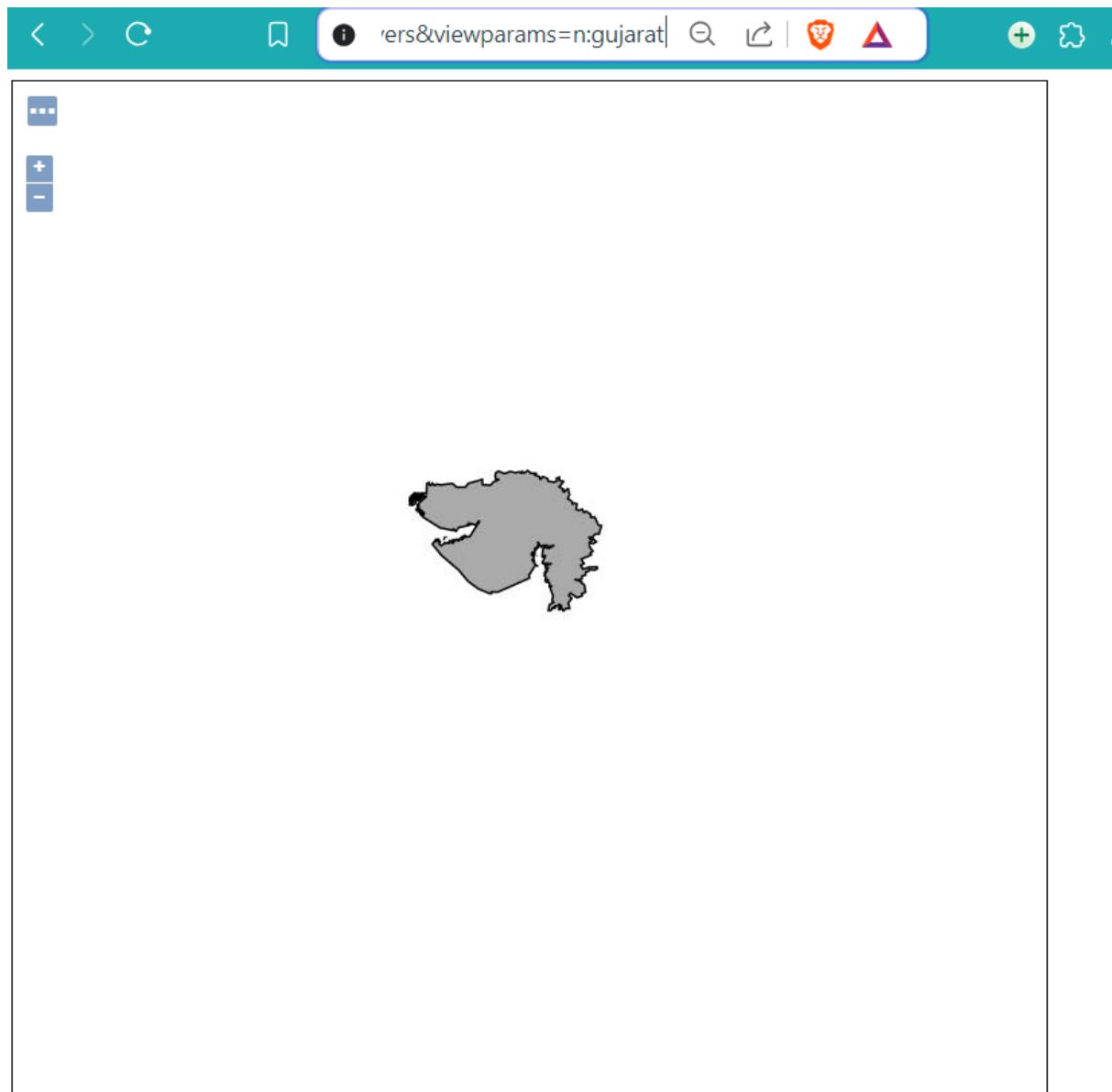
Customize attributes

Property	Type	Nullable	Min/Max Occurrences
id	Integer	false	1/1
geom	Multipolygon	true	0/1
sid	BigDecimal	true	0/1
name	String	true	0/1
area	BigDecimal	true	0/1

Go to layers preview → Isi:sample\_query → openLayers → get the output

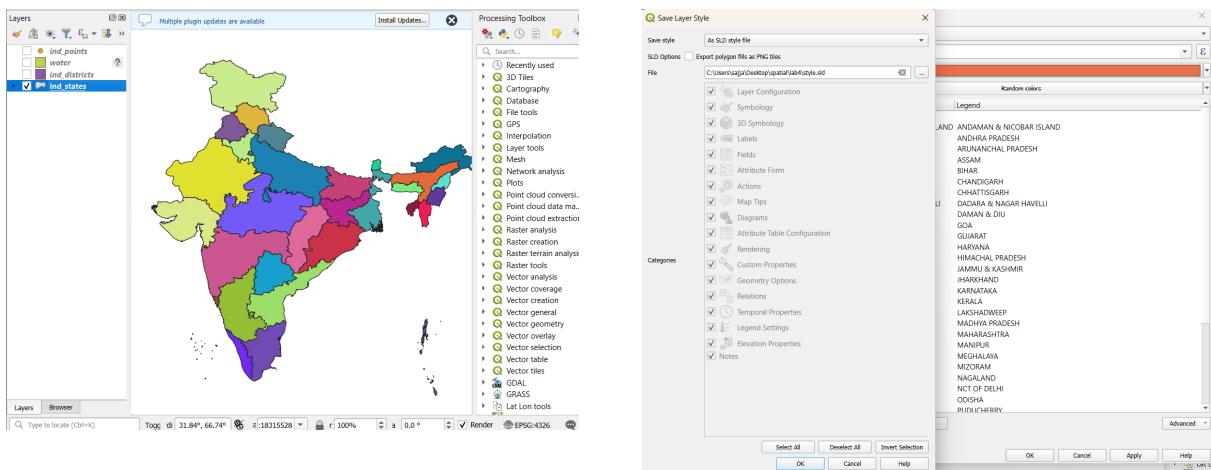


Now if we want to change the state we will add '&viewparams=n:gujarat' at the end of our link;



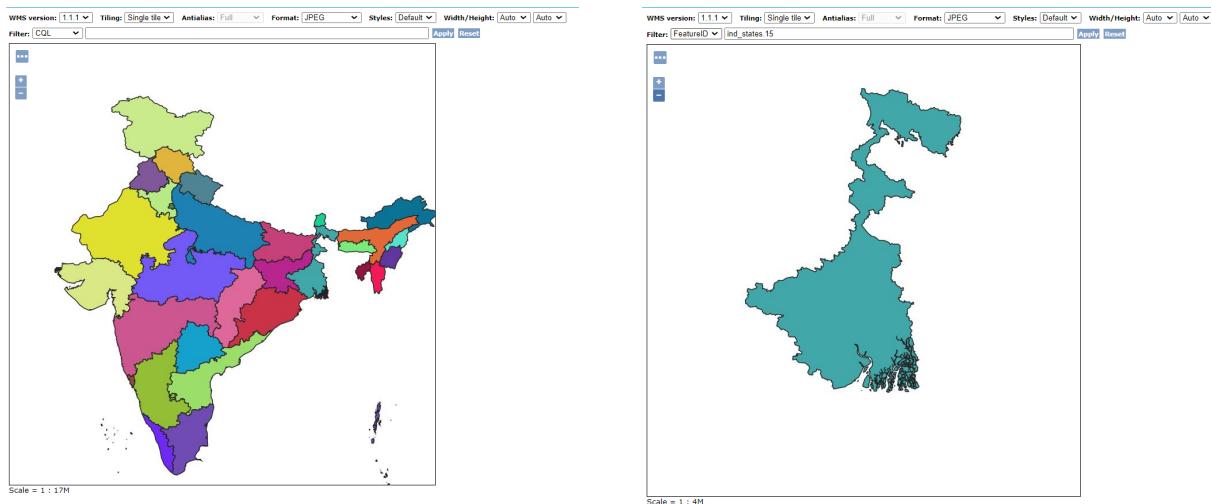
## STYLING

Go to symbology → categorised → classify on the basis of name → save style file (.sld)



Now go back to geoserver→styles→ fill in details like below and copy the xml code and apply→ now scroll up→ go to publishing and choose ind\_states→ default→ apply.

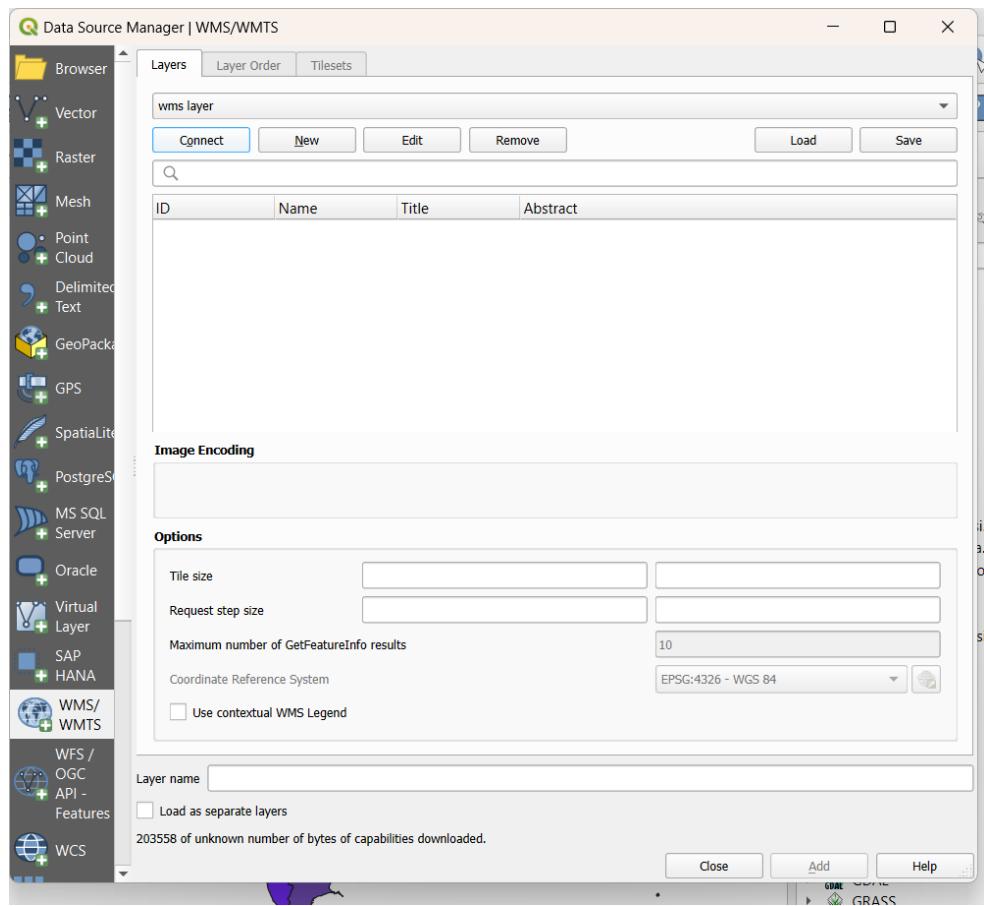
We can also see the preview of all states.



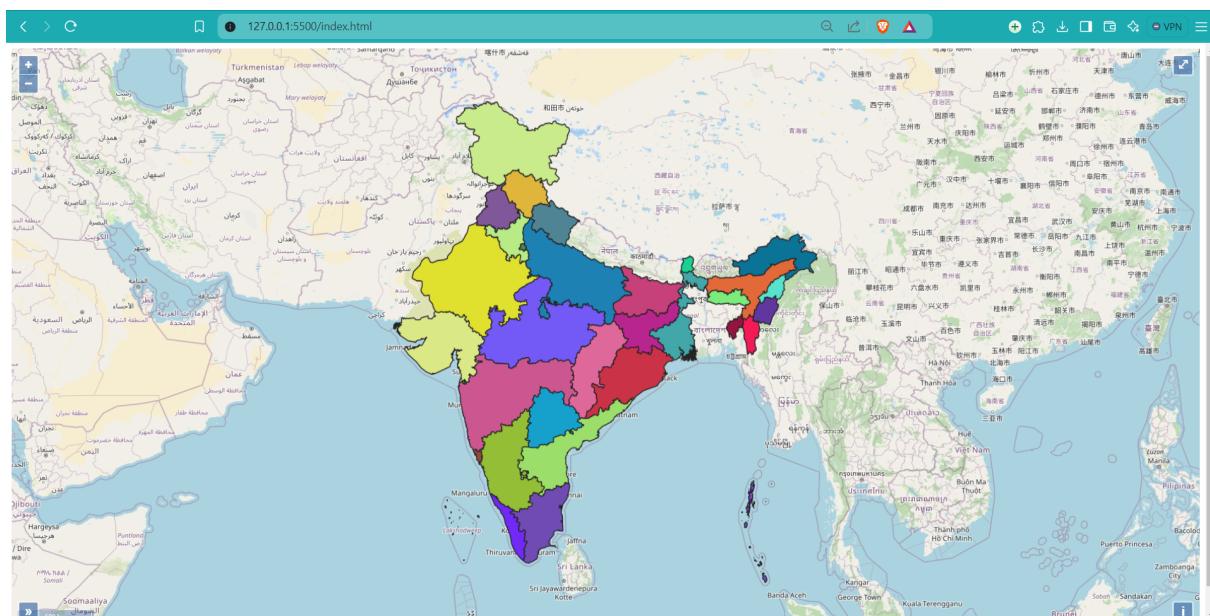
## ADDING MAP AS WMS

Opened <http://localhost:8080/geoserver/lab4/wms/>

Go to QGIS→add wms LAYER→ Add details with the url and save and click connect.



After modifying the html code, we get our ind\_states layer overlayed on osm along with the styling:



```

<!DOCTYPE html>
<html>
  <head>
    <title>Final</title>
    <link
      rel="stylesheet"
      href="https://openlayers.org/en/v4.6.5/css/ol.css"
      type="text/css"
    />
    <script src="https://openlayers.org/en/v4.6.5/build/ol.js">
    <style>
      #map {
        height: 100vh;
        width: 100%;
      }
      .query {
        position: absolute;
        top: 10px;
        left: 10px;
        background: white;
        padding: 10px;
        border-radius: 5px;
        z-index: 100;
      }
    </style>
  </head>
  <body>
    <div id="map" class="map"></div>
    <!-- Dropdown menu for selecting a state -->
    <div class="query">
      <form>
        <select id="tr_name">
          <option value="Andhra Pradesh">Andhra Pradesh</option>
          <option value="Arunachal Pradesh">Arunachal Pradesh</option>
          <option value="Assam">Assam</option>
          <option value="Bihar">Bihar</option>
          <option value="Chhattisgarh">Chhattisgarh</option>
          <option value="Goa">Goa</option>
        </select>
      </form>
    </div>
  </body>
</html>

```

```

<option value="Gujarat">Gujarat</option>
<option value="Haryana">Haryana</option>
<option value="Himachal Pradesh">Himachal Pradesh</option>
<option value="Jharkhand">Jharkhand</option>
<option value="Karnataka">Karnataka</option>
<option value="Kerala">Kerala</option>
<option value="Madhya Pradesh">Madhya Pradesh</option>
<option value="Maharashtra">Maharashtra</option>
<option value="Manipur">Manipur</option>
<option value="Meghalaya">Meghalaya</option>
<option value="Mizoram">Mizoram</option>
<option value="Nagaland">Nagaland</option>
<option value="Odisha">Odisha</option>
<option value="Punjab">Punjab</option>
<option value="Rajasthan">Rajasthan</option>
<option value="Sikkim">Sikkim</option>
<option value="Tamil Nadu">Tamil Nadu</option>
<option value="Telangana">Telangana</option>
<option value="Tripura">Tripura</option>
<option value="Uttar Pradesh">Uttar Pradesh</option>
<option value="Uttarakhand">Uttarakhand</option>
<option value="West Bengal">West Bengal</option>
<option value="Chandigarh">Chandigarh</option>
<option value="Delhi">Delhi</option>
<option value="Jammu & Kashmir">Jammu & Kashmir</option>
</select>
<button type="button" onclick="callQuery()">Show</button>
</form>
</div>
<script>
    // Initialize the map
    var map = new ol.Map({
        target: "map",
        layers: [
            new ol.layer.Tile({
                source: new ol.source.OSM(),
                title: "OSM",
            }),

```

```

        ],
        view: new ol.View({
            center: ol.proj.transform(
                [76.9639, 10.9905],
                "EPSG:4326",
                "EPSG:900913"
            ),
            zoom: 5,
        }),
        controls: ol.control
            .defaults({
                attributionOptions: {
                    collapsible: true,
                },
            })
            .extend([
                new ol.control.ScaleLine(),
                new ol.control.FullScreen(),
                new ol.control.OverviewMap(),
            ]),
    });
}

// Define the WMS layer for displaying states
var wms = new ol.layer.Tile({
    source: new ol.source.TileWMS({
        url: "http://localhost:8080/geoserver/lab4/wms",
        params: {
            LAYERS: "lab4:sample_query",
            STYLES: "lab4:style",
            FORMAT: "image/png",
            TILED: true,
        },
        serverType: "geoserver",
        projection: "EPSG:4326",
    }),
    title: "States",
    visible: true,
});

```

```

// Add the WMS layer to the map
map.addLayer(wms);

// Function to update the WMS layer based on selected state
function callQuery() {
    var stateName = document.getElementById("tr_name").value;
    // Pass the state name as a parameter to the SQL view
    wms.getSource().updateParams({
        viewparams: "n:" + stateName,
        STYLES: "lab4:style",
    });
}
</script>
</body>
</html>

```

## FINAL OUTPUT

We get the web output and i have an option choose from any of the states→ click on show and then that state will be highlighted-

