Global Geo-Informatics Options by Contexts (GeOC)

Installation Guide (for programmers)

The Global Geo-informatics Context and Options (GeCO) is a new web-based GIS tool that enables its users to define, monitor, assess and co-create knowledge and learning on relevant Sustainable Land Management (SLM) options that match the social-ecological context at global, regional and national scales. The GeOC tool aims to support the implementation of SLM practices by the local international communities by providing them with context-specific information that is required to make sound investment decisions for agricultural and rural development. The GeOC is designed to provide land users, development projects or programs, and policy decision-makers with plausible, robust extrapolation domains for guiding decisions on the selection and use of SLM options, and an open platform for docking different disciplinary projects into integrative/holistic and converging actions for promoting SLM at scale.

GeOC is the result of the synergic efforts by CGIAR Research Program on Dryland Systems (CRP-DS), the German Federal Ministry for Economic Cooperation and Development (BMZ), the International Center for Agricultural Research in the Dry Areas (ICARDA), ICARDA Geoinformatics Unit (GU), the CGIAR Research Program on Water, Land and Ecosystems (CRP-WLE) and is powered by iMMAP, Codeobia, D-Space and Amazon Web Services.

For more information, please visit:
Main website: https://geoc.mel.cgiar.org
Guide: https://cgiarmel.atlassian.net/wiki/spaces/MEL/pages/15794182/GeOC+Programmer+Guide+-+Global+Geo-informatics+Options+by+Contexts

AUTHORS
Jim Jaspe[2], Bashar Ayyash[2], Badabate Diwediga[2], Mahdi Hamdan[3], Valerio Graziano[1], Fajr Fradi[1];

CONTRIBUTORS
Khaled El-Shamaa[1], Jalal Omari[4], Quang Bao Le[1], Enrico Bonaiuti[1], Chandrashekhar Biradar[1],;

SUGGESTED CITATION
ICARDA, iMMAP and Codeobia (2017). Global Geo-Informatics Options by Contexts (GeOC): Installation Guide (for programmers). The International Center for Agricultural Research in Dry Areas, Amman, Jordan.

[1] International Center for Agricultural Research in Dry Areas (ICARDA)
[2] Information Management and Mine Action Programs (iMMAP)
[3] Codeobia
[4] Sooq.com (Amazon)

Tables of contents

1. Introduction

The Web-based Geographic Information System (WebGIS) application is designed as an integral part of the web-based Monitoring, Evaluation and Learning (MEL) platform. MEL enables better result-based management including planning, reporting, coordination, risk management, performance evaluation, management of legal mechanisms in place among partners, as well as knowledge sharing and learning amongst different groups of stakeholders (donors, partners, project/program implementers, managers and collaborators) within and across Projects and Programs. WebGIS extends the capabilities of MEL by integrating GIS, mapping, visualization, charting, analysis, reporting and dissemination of spatial data. In addition, it provides the mapping capabilities for Sustainable Land Management database as part of the MEL platform.

The main objective of the Option-by-Context Web-GIS platform is to support and guide decisions on Sustainable Land Management. The level of the analysis is scalable and enables analysis from local to sub-national, to national, to regional, to global level. The analysis will be location based exploring global contextual options and best practices. The availability of 30 arc-second resolution spatially explicit databases enables the end user to extract information on options-by-context considering drivers, trends and patterns.

The Option-by-Context Web-GIS platform is accessed in MEL platform through the WebGIS menu where user can submit new SLM metadata form, and visualize WebGIS tool. While the admin can approve, reject or delete the submitted SLMs.

This document is a living document and it will be updated over the lifetime of iMMAP's engagement with ICARDA as the system evolves. The document is used by all Partners[1] customizing and developing the system in order to support broad understanding and report funds allocated for more efficient and effective Monitoring and Evaluation. It will serve as a guide to document the system and its evolution.

The present document is designed to be open access and accessible to all stakeholders in order to increase customization and adaptation of different institutions partners of this initiative. As per CGIAR

[1]2016: MEL System development partners: CGIAR Research Program on Roots, Tubers and Bananas (RTB) - http://www.rtb.cgiar.org; International Potato Center (CIP) - http://cipotato.org; International Center for Agricultural Research in Dry Areas (ICARDA) - http://icarda.org; IMMAP - http://www.immap.org; CGIAR Dryland Cereals (DC) http://drylandcereals.cgiar.org; CGIAR Grain Legumes (GL); http://grainlegumes.cgiar.org.

Policy[2] the information products including codes and software should be accessible and use granted free of cost using appropriate licence (https://creativecommons.org).

1.1 Purpose

The purpose of this document is to outline the technical design of WebGIS, and provide a step-by-step guidance to build and deploy WebGIS tool.

1.2 Intended Audience

The intended audiences are:

- Supervisors, to analyse the design and implementation of WebGIS tool
- WebGIS team members in charge of developing the system further
- Future developers to extend or use some ideas of WebGIS tool and SLM form

1.3 Design Goals

The following design goals were considered during developing the WebGIS tool:

- To be developed based on industry best practices, including the use of recommended naming conventions including the use of recommended naming conventions, cascading style sheets (css) for separation of layout and content, and W3C standard-compliant HTML.
- To be developed based on J2EE design patterns. Some of the design patterns are used in the development of WebGIS Tool:
    - ✓ Front Controller
    - ✓ Session Façade
    - ✓ Business Delegate
    - ✓ Data Access Object
    - ✓ Value Object
- To be scalable: the ability of the WebGIS Tool to scale both up and down to support varying numbers of users or transaction volumes. The WebGIS tool should be able to scale horizontally (by adding more servers) or vertically (by increasing hardware capacity or software efficiency).
- To be flexible: the ability of the WebGIS Tool to adapt and evolve to accommodate new requirements without affecting the existing operations. This relies on a modular architecture,

which isolates the complexity of integration, presentation, and business logic from each other to allow for easy integration of new technologies and processes within the application.

Design goals taken during integrating WebGIS tool with MEL platform (SLM approval and submission):

- Back-end developed based on PHP (Zend Framework 1.12)
- Front-end developed on Metronic html theme v.4.7
- Database tables to store SLM metadata

1.4 Assumptions and Constraints

The following assumptions and constraints have been made in regards to the design, support and enhancements of the WebGIS application:

- A key architectural goal is to leverage industry best practices for designing and developing a scalable, enterprise- wide J2EE application. To meet this goal, the design of the WebGIS Tool will be based on core J2EE patterns as well as the industry standard development guidelines for building the WebGIS Tool.
- WebGIS is to be compatible with latest Internet Explorer, latest Chrome, latest Firefox, and latest Safari.
- Developers have familiarity with JavaScript, JQuery, Java, PostgreSQL, PostGIS, GeoServer, ArcGIS or QGIS

2. General Overview and Conceptual Framework

2.1 General Overview

Over the recent years, ICARDA has developed the conceptual framework of the Options-by-Context that provides spatially explicit contextual information on SLM practices. The conceptual high-level diagram below (Figure 1) indicates the relationships between "Options-by-Context" approach through the relevance, effectiveness and stress/potential links.

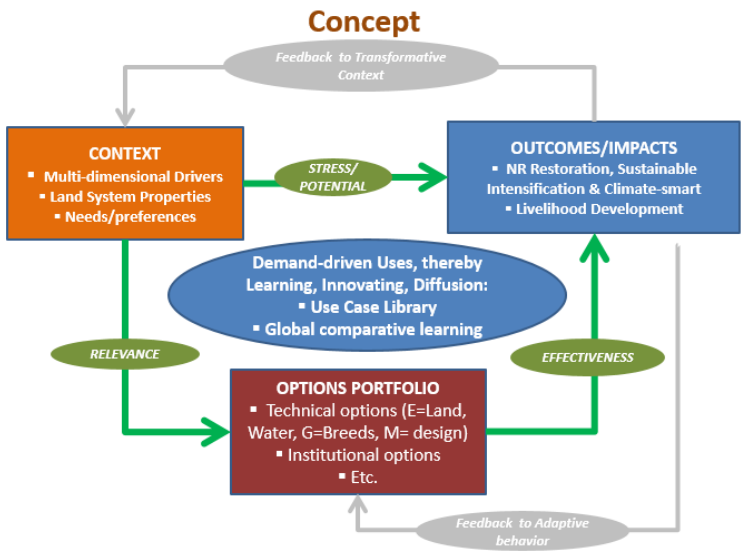

Concept
Feedback to Transformative Context
CONTEXT
▪ Multi-dimensional Drivers
▪ Land System Properties
▪ Needs/preferences
STRESS/ POTENTIAL
OUTCOMES/IMPACTS
▪ NR Restoration, Sustainable Intensification & Climate-smart
▪ Livelihood Development
Demand-driven Uses, thereby Learning, Innovating, Diffusion:
▪ Use Case Library
▪ Global comparative learning
RELEVANCE
EFFECTIVENESS
OPTIONS PORTFOLIO
▪ Technical options (E=Land, Water, G=Breeds, M= design)
▪ Institutional options
▪ Etc.
Feedback to Adaptive behavior

Figure 1: Options-by-context conceptual framework

The main "Context" and "Options" under "Systems Transition" is indicated on the flowchart below (Figure 2).

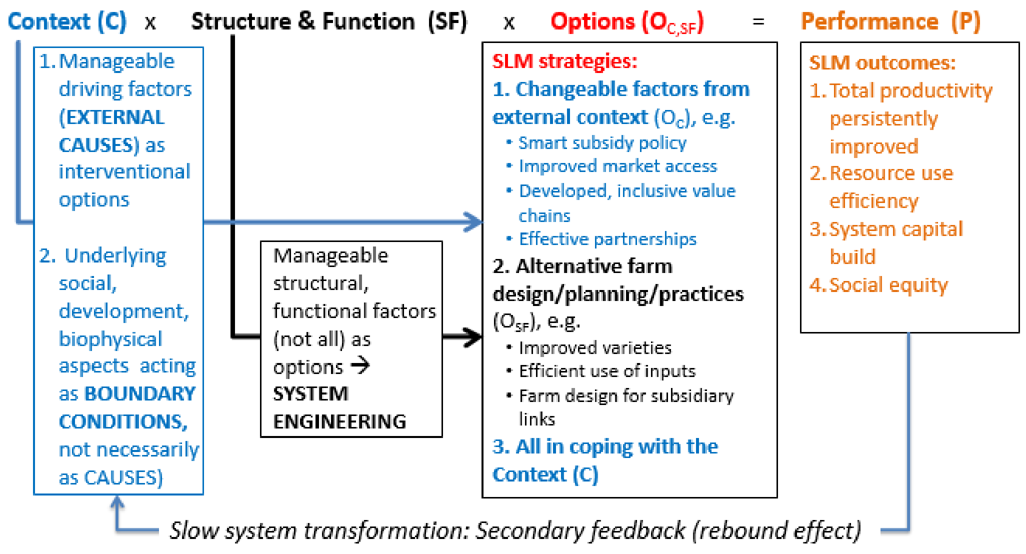

Context (C) x Structure & Function (SF) x Options ($O_{C,SF}$) = Performance (P)
1. Manageable driving factors (EXTERNAL CAUSES) as interventional options
2. Underlying social, development, biophysical aspects acting as BOUNDARY CONDITIONS, not necessarily as CAUSES)
Manageable structural, functional factors (not all) as options → SYSTEM ENGINEERING
SLM strategies:
1. Changeable factors from external context (O_C), e.g.
• Smart subsidy policy
• Improved market access
• Developed, inclusive value chains
• Effective partnerships
2. Alternative farm design/planning/practices (O_{SF}), e.g.
• Improved varieties
• Efficient use of inputs
• Farm design for subsidiary links
3. All in coping with the Context (C)
SLM outcomes:
1. Total productivity persistently improved
2. Resource use efficiency
3. System capital build
4. Social equity
Slow system transformation: Secondary feedback (rebound effect)

Figure 2: Context and Options under System Transition

Based on these principles the following high-level representation (Figure 3) of the functionality is designed to be adopted by the WebGIS.

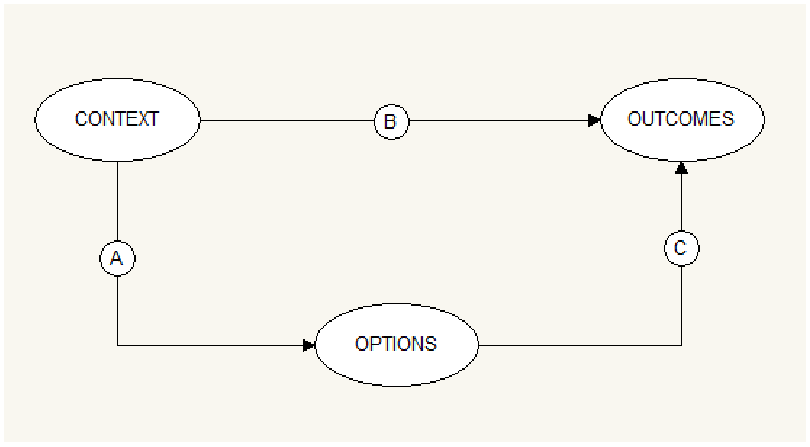


Figure 3: High-Level Options-by-Context representation

2.2 Main Functionalities

In addition to the normal display, search, pan, zoom, navigate, download, login, register, feedback, help buttons, the following functionality is designed to:

- A-1 Map areas vs contextual criteria (defined by SES-TYPE and user defined query)
- A-2 List of Options vs. contextual criteria (defined by SES-TYPE and user defined query)
- A-3 Stats (Mean, Max, Min, STD, Range, Sum, Weighted Average)
- B-1 Map of degraded / improved areas vs contextual criteria
- B-2 Stats Area on Degraded/Improved areas vs contextual criteria
- B-3 Stats Number of people (male/female, urban, rural) affected
- C-1 List of successful working stories vs contextual criteria (Through MEL)
- C-2 List of research / management gaps (Through MEL)
- C-3 List of references to documents (Through MEL)

3. Technology and Infrastructure

3.1 Application Technologies

The technologies used to develop and deploy this system are:

- Java Runtime Environment 8
- Apache Tomcat 8.5.6
- PostgreSQL 9.5
- PostGIS 2.3.0
- Apache 2.4

The languages used are:

- Java 8
- HTML/CSS

- Javascript / JQuery

Used Software in MEL and WebGIS Integration

- METRONIC HTML theme v.4.7
- Javascript & jQuery
- Leaflet JS
- Mapbox

3.2 Hardware & Operating System Requirements

- Amazon Instance Server: Linux Operating System x64 (Ubuntu 16.04.1 LTS Xenial)
- Processor(s): minimum 2.5GHz per server core (3.0Ghz or faster recommended): Quad Core (Optimal 8 cores)
- Minimum 16GB RAM: 32GB recommended for large queries
- Minimum of 50GB available HDD space: SSD volume type recommended

3.3 Web Browsers

- 1024x768 minimum screen resolution (optimal viewing requires higher resolution)
- Compatible browsers: IE 10 or Newer, Latest Chrome, Latest Firefox, Explorer and Apple Safari.
- To get the best-possible experience, we recommend using a browser that has full HTML-5 compatibility (Chrome, Firefox, IE10, IE11)

4. Application Architecture

Application architecture defines the various components and their interactions in context of a whole system. Application architecture is the critical software that bridges the architectural gap between the application server and the application's business logic, thereby eliminating the complexities and excessive costs of constructing, deploying and managing distributed enterprise applications.

4.1 Physical Architecture

The diagram below (Figure 4) provides illustration of the System Architecture along with various system components used in architecting the WebGIS tool.

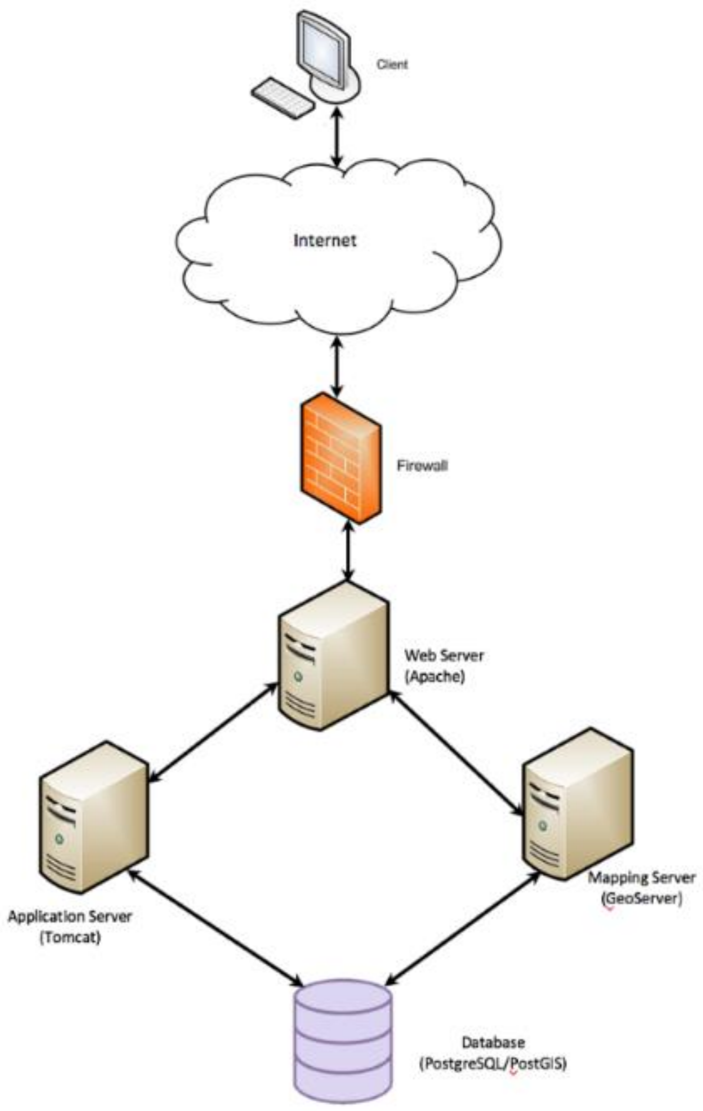


Figure 4: Overview of the WebGIS Physical Architecture

Interaction of software components along with its responsibilities are explained below:

- **Web server :** Web Server is responsible for serving web pages, mostly HTML pages, via the HTTP protocol to clients. The Web server sends out web pages in response to requests from browsers. A page request is generated when a client clicks a link on a web page in the browser.
- **Apache Tomcat Application Server** : Application server hosts the business logic and the business model classes of WebGIS and GeoServer applications. It serves requests for dynamic HTTP web pages from Web servers.
- **PostgreSQL/PostGIS:** PostgreSQL is responsible for storing object-relational data while PostGIS is responsible for collecting, processing, and storing spatial data.

- **HTTP:** Hyper Text Transport Protocol is the communication protocol used to connect to servers on the World Wide Web. The primary function of HTTP is to establish a connection with a Web server and transmit HTML pages to the user's browser.
- **JDBC:** Java Database Connectivity is an application program interface (API) specification for connecting programs written in Java to the data in popular databases. The application program interface lets you encode access request statements in structured query language (SQL) that are then passed to the program that manages the database. It returns the results through a similar interface.
- **JSON:** It is used primarily to transmit data between a server and web application, as an alternative to XML.

4.2 System Architecture (Atlassian version)

The system architecture is based on AWS services, Free and Open Source Software innovative technologies, supporting large web computing.

- **Hardware Architecture**

GeOC is hosted on Amazon Web Server with the following specification:

Hosting provider

- Amazon
- IP address: 54.171.74.139
- OS: Amazon Linux Instance
- Web server: Tomcat
- Nameserver:

The technical implementation of the GeOC interface consists of four primary steps:.

- Web Interface
- Web Server
- Database Server
- Map Server (Geoserver/ArcGIS for Server WMS and ArcGIS for Server Imagery Service)

The GeOC application uses HTML forms and JavaScript to communicate with Java REST API on the web server. The REST API are used to process users request and communicate with the database by extracting the requested data and prepare the results in the requested format. If the user chooses an interactive visual, the data is sent to GeoServer/ArcGIS Server, which prepares the data to be displayed on the website using Leaflet/ Openlayers.

A schematic representation of the process is shown below (Figure 5):

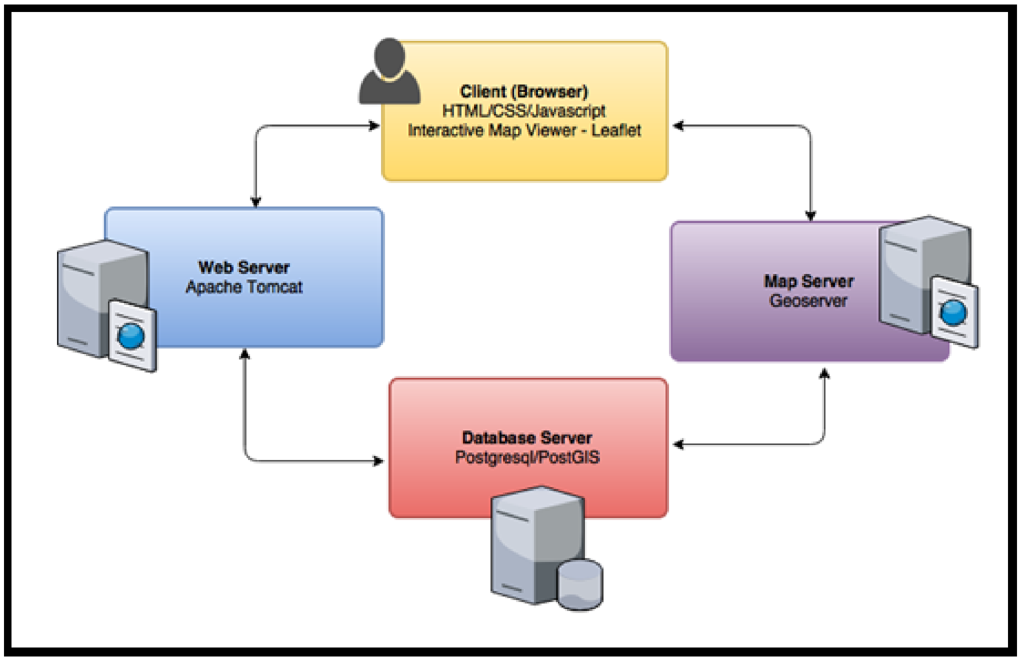


Figure 5. Schematic representation of the Web-GIS implementation

4.3 Performance Hardware Architecture

GeOC has no dedicated hardware to enhance the system performance, nothing of the following were implemented: load balancer, CDN, clusters; however it is implemented on AWS.

4.4 Software Architecture

These are the GeOC components and libraries:

- Leaflet API
- Open Layers
- Smart Admin
- JavaScript
- Spring 4.3.x and Java REST API
- PostgreSQL 9.5
- PostGIS 2.2
- GeoServer 2.9.1
- ArcGIS Server
- Encoding: UTF-8

UTF-8 (8-bit UCS/Unicode Transformation Format) is a variable-length character encoding for Unicode. It is the preferred encoding for web pages.

4.5 Internal Communications Architecture

MEL System components interact with each other in systematic, specific ways. Their definition depends on either Zend Framework logic itself, or a customized way built for a custom feature in the system (i.e. Open Access repository). These internal interactions are exemplified below with code snippets. Figure 6 shows a static representation of the subsystem composing the GeOC application:

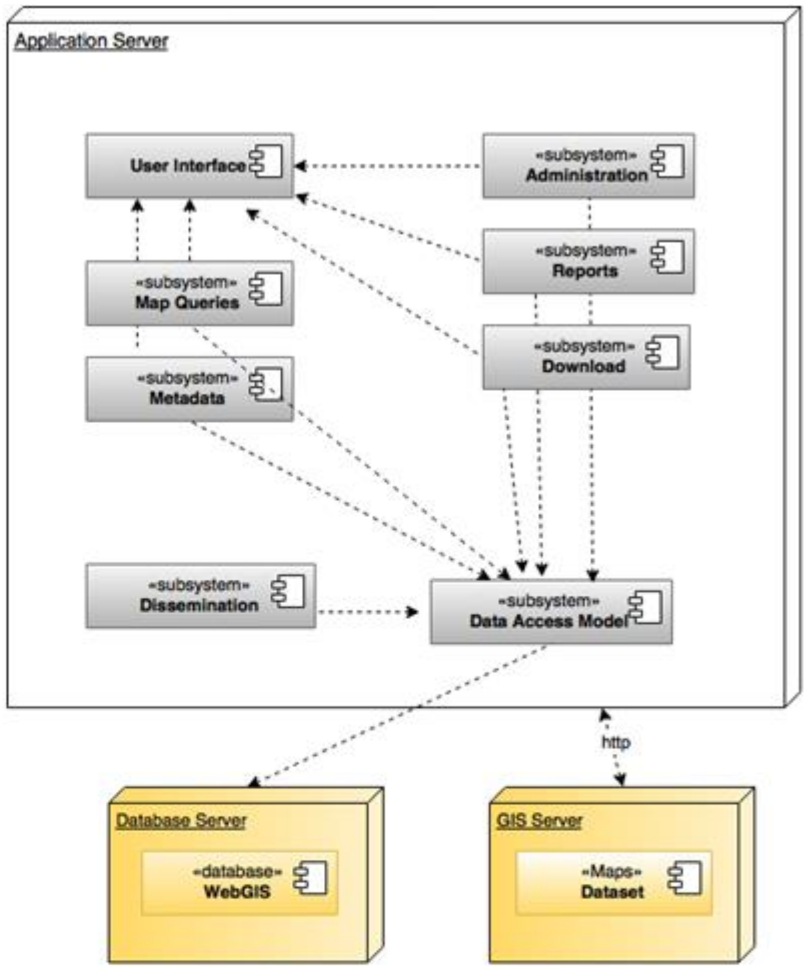
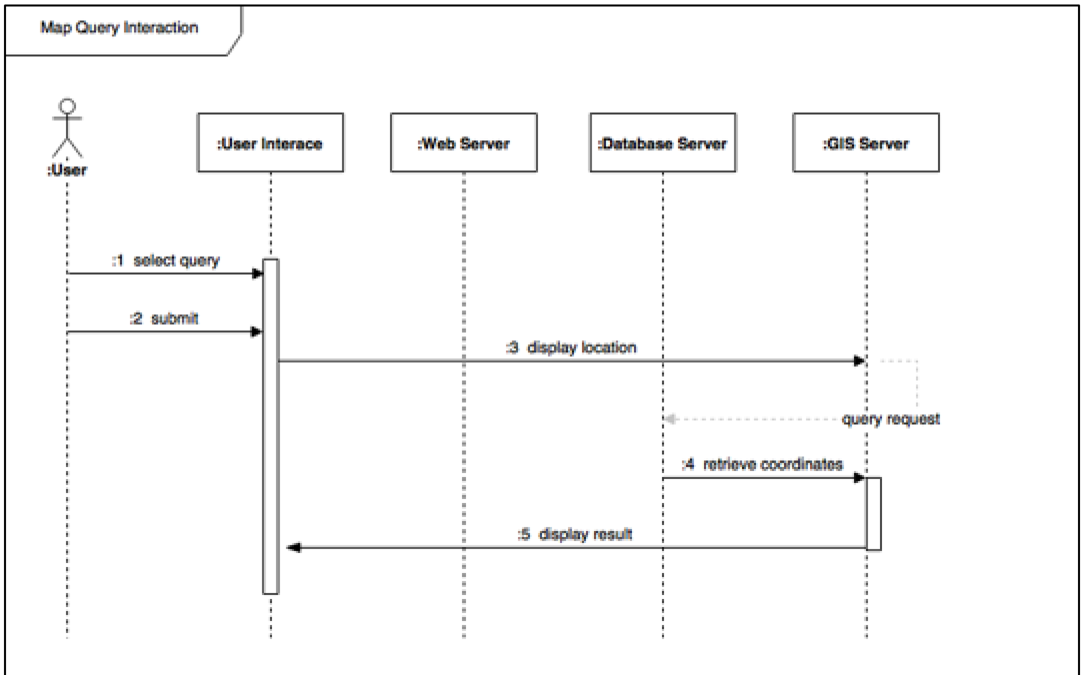


Figure 6: GeOC component diagram

The communication between the Application Server and the Database Server is the standard JDBC interface. Similar to the former, the communication between the Application Server and the GIS Server is presented in the diagram. The communication between the User Interface, Application Server, Database Server and GIS Server are more articulated and are explained more in detail in the following sequence diagram (Figure 7):



Interaction	Description
1	User query by Administrative Unit, Theme, Sub-Theme, Dataset and Options Portfolio
2	User submit the selected query for processing
3	Display location map based on area of interest.
4	Retrieve the coordinates and execute spatial query to PostgreSQL / PostGIS
5	Display results back to user for analysis and option to download the map and statistics.

Figure 7: GeOC Map Query Interaction UML Diagram

4.6 System Architecture Diagram

The following diagram (Figure 8) illustrates the high-level architecture of the GeOC application main components, including user interaction with the system and the dissemination systems.

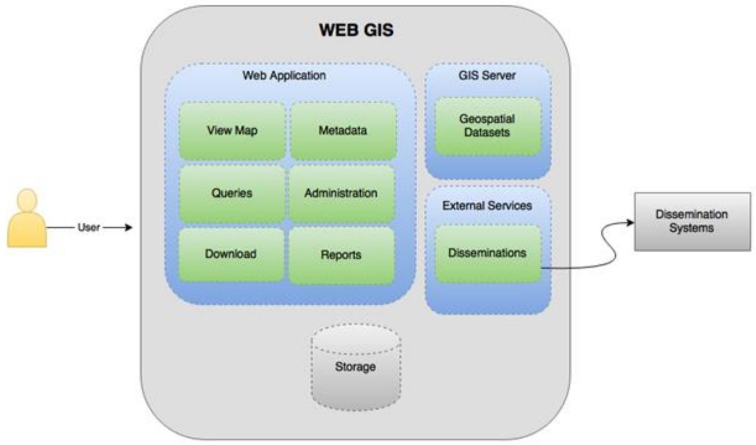


Figure 8: GeOC high-level architecture design

4.7 Block Architecture

The diagram below (Figure 9) provides illustration of the principal parts of WebGIS tool.

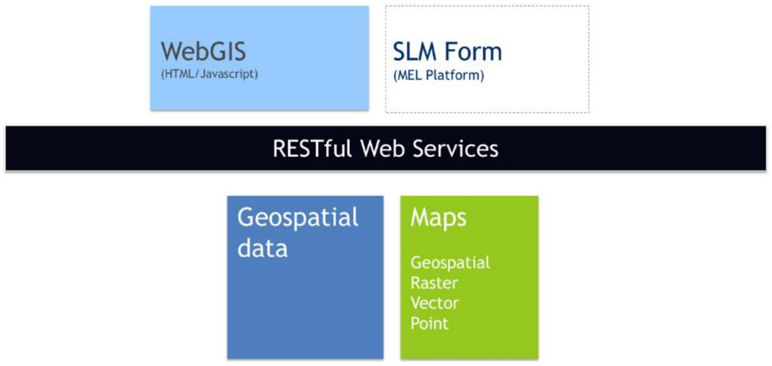


Figure 9: Overview of Principal Parts of WebGIS Tool

4.8 High Level Architecture Layers

The WebGIS Tool architecture can be represented in the following layers illustrated by the diagram which provides some of the key features below.

- **Structure** - Organizing applications along business-level boundaries and not technical boundaries
- **Speed & Flexibility** - Making application changes through configuration and not programming
- **Control** - Modifying, extending or overwriting any architectural element.

- **Reuse** - Achieving greater reusability and integration by loosely coupling application logic to infrastructure.

The WebGIS Tool design is based on a n-tier layer approach (Figure 10). “A tier is a logical partition of the separation of concerns of the system. Each tier is assigned its unique responsibility in the system. We view each tier as logically separated from one another. Each tier is loosely coupled with the adjacent tier.”

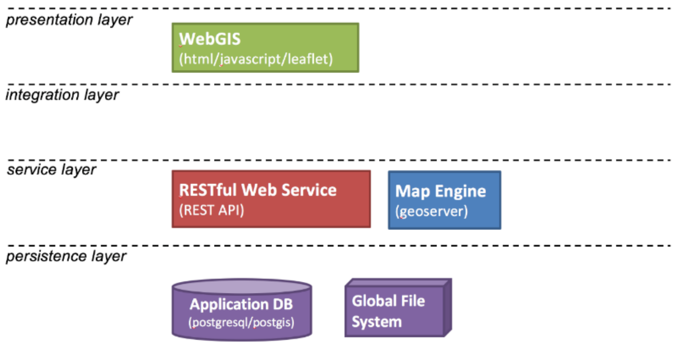


Figure 10: High Level Architecture Layers of WebGIS

- **Presentation Layer**

The presentation layer is where the graphical user interface is dynamically generated. HTML pages are delivered to the client browser by the WebGIS tool upon a user request.

- **Integration Layer**

An integration layer can automate complex business processes or provide unified access to information that is scattered across many systems.

- **Service Layer / Business Layer**

The service layer implements the business rules of the application. It abstracts business logic and data access. The business logic layer will run under the “Application Server” environment. Application Servers provide support for transaction control, thread management and other run-time services that make application development much simpler and more reliable. Business components are generally computation-intensive. They will use Data Access objects (DAO) to communicate with the database.

- **Persistence Layer / Data Access Layer**

The persistence layer is responsible for manipulating the database, and it is used by the service layer. Persistence can be complex in large applications using protocols like JDBC/JNDI. Neither the client nor the business component needs to be aware of this complexity. Decoupling the persistence logic from the business components and client allows for a flexible, easy to maintain application. The Data Access Object (DAO) pattern allows for the abstraction of the persistence from the business component. The

Data Access Object manages the connection to the data source to obtain and store data. It encapsulates all access to the data store.

4.9 Interfaces

The following diagram (Figure 11) describes the interfaces between the WebGIS, GeoServer and SLM Form.

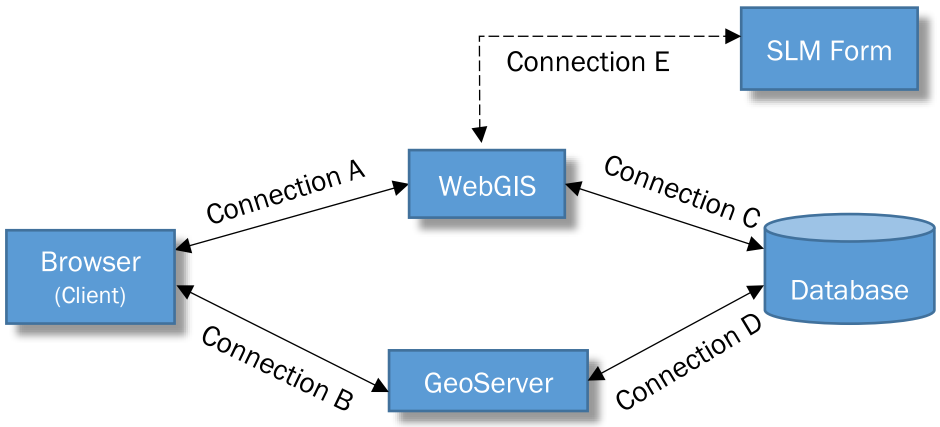


Figure 1: Applications that Interfaces with the WebGIS Tool

- **Connection A – Browser to WebGIS**

This is the connection between the end user's web browser and the web server (Amazon Cloud) hosting the WebGIS application. Transfer consists of standard HTTP/HTTPS requests to the server to render HTML to the client.

- **Connection B – Browser to GeoServer**

This is the connection between the end user's web browser and the mapping server (GeoServer). Requests are issued using JavaScript in conjunction with the Leaflet API to issue HTTP/HTTPS requests.

Connection C – WebGIS to Database

This is the connection between WebGIS application to PostreSQL/PostGIS database. The connection requests are issued using JNDI connection pooling.

- **Connection D – GeoServer to Database**

This is the connection between GeoServer to PostreSQL/PostGIS database. The connection requests are issued using JNDI connection pooling.

- **Connection E – SLM Form to WebGIS**

This is the connection between SLM Form to WebGIS application. Requests are issued using JavaScript in conjunction with REST API to issue HTTP/HTTPS requests.

The SLM form stores the Meta data for designated area where the SLM has been implemented, the required data are split into multiple form elements split into, wizard like, pages, and use MySQL database to store the data into JSON format inside table records.

5. Database Architecture

The main repository for WebGIS data is a relational database utilizing PostgreSQL and PostGIS as database management system hosted on Amazon cloud services. This repository is where all application data and spatial data are stored and maintained.

5.1 Data Models

This diagram in Figure 12 represents the WebGIS database schema and its direct table relationship. Figure 13 shows the SLM database schema

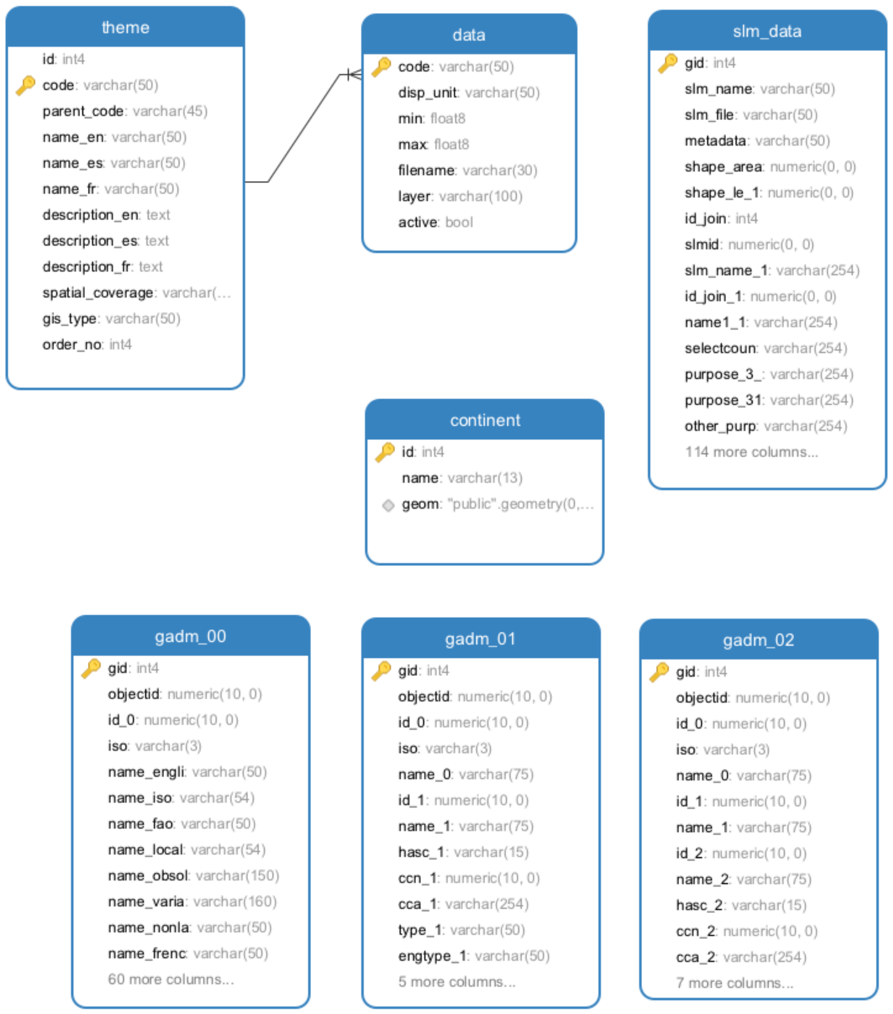


Figure12: WebGIS Data Model

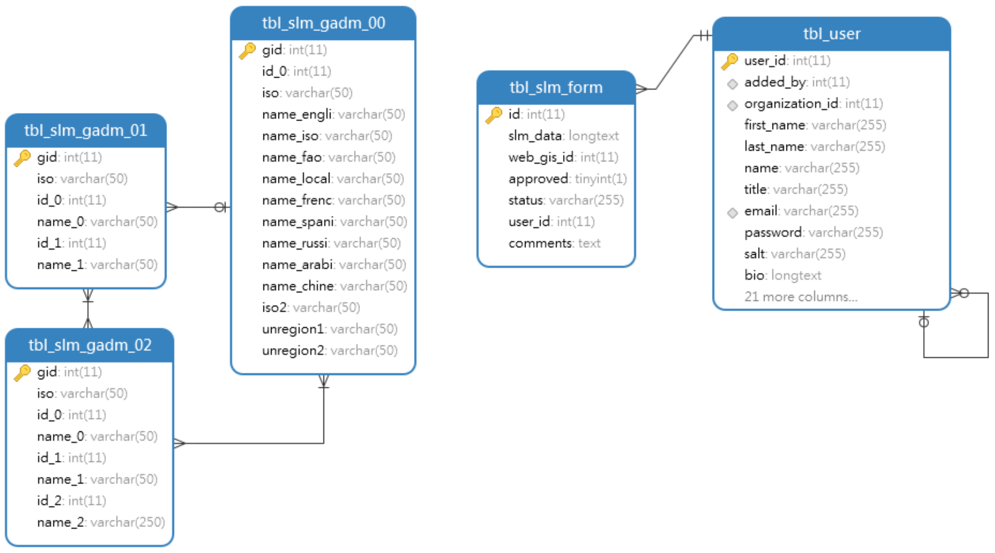


Figure 5: SLM Data Model

5.2 Tables

- **Theme table**

This table stores the information about the themes and sub-themes.

✓ Table name: Theme

✓ Schema: webgis

Column Name	Type	Description
id	serial	This field is unique and auto generated
code	character varying(50)	This field is contains the identifier of the theme
parent_code	character varying(45)	Contains the code of the parent theme.
name_en	character varying(50)	Contains a short description of the theme in English
name_es	character varying(50)	Contains a short description of the theme in Spanish
name_fr	character varying(50)	Contains a short description of the theme in French
description_en	text	Contains a long description of the theme English
description_es	text	Contains a long description of the theme in Spanish
description_fr	text text	Contains a long description of the theme in French
spatial_coverage	text	Contains the spatial coverage
order_no	integer	This field contains the order number of this theme

- **Data table**

This table stores the raster datasets information.

- ✓ Table name: data
- ✓ Schema: webgis

Column Name	Type	Description
code	character varying(50)	This field is contains the theme identifier.
min	double precision	Contains the minimum value of the raster.
max	double precision	Contains the maximum value of the raster.
filename	character varying(30)	The filename of the raster
layer	character varying(100)	The name of the layer
Active	Boolean	This field contains the status of the layer

- **Global Administrative Unit Layers tables**

- ✓ Table names: continent, gaul_00, gaul_01, gaul_02
- ✓ Schema: webgis

These tables store the Global Administrative Unit Layers (GAUL) is a spatial database of the location of the world's administrative areas. The data are downloaded from GADM website and extracted by Continent, Region, Province, and District.

- **SLM Data table**

This table stores the SLM data information.

- ✓ Table name: slm_data
- ✓ Schema: webgis

Column Name	Type	Description
gid	serial	This field is contains the slm identifier.
slm_name	character varying(50)	Contains the name of the SLM.
purpose_31	character varying(254)	Contains the description of the SLM.
geom	geometry(MultiPolygon,4326)	Contains the SLM geometry
status	smallint	Contains the status of the SLM; 0 - Submitted; 1 - Pending; 2 - Approved

- **SLM Form table**

This table stores the SLM metadata information.

- ✓ Table name: tbl_flm_form
- ✓ Schema: crpcoreix

Column Name	Type	Description
id	Unique id	This field is contain slm form identifier.
slm_data	Long text	Contains the information for slm form stored as serialized json string
web_gis_id	Unique integer	Contains the ids for the slm in webgis tool
approved	Integer	Contains the value of approved status for slm form
status	String	Contains the status of the SLM; Approved, Draft, Deleted, Rejected, and Review
user_id	Integer	Contains the id for the user who submitted the slm form
comments	Text	Contains the comments for the rejection of slm form

6. System Design: Business Requirements

The following table represents MEL System business processes with a detailed description of actors, process flow, and requirements.

Actors	• **GeOC database custodian** • **GeOC system administrator** • **Admin user** • **User**
Process Flow	• **GeOC database custodian:** GeOC database custodian can initiate any database management tasks and update the system. • **GeOC System administrator :** GeOC System Administrator can approve users, user updates and new records, initiate a discussion during the review/approval process and manage any changes on the GeOC system and database. • **Admin user:** Admin users can approve other users, user updates, new records and initiate a discussion during the review/approval process. • **User:** Guests and Users can initiate any SLM CO session and update/insert new database entries in the GeOC.
Requirements	GeOC access approval

7. Database Design

A physical GeOC data model (GeOC-PDM) is provided in a HTML format file named Full GeOC Physical Report.html. The file supports the analysis and documenting of the underlining tables, views, and other objects in a database, including the multidimensional objects necessary for data warehousing. A PDM is more solid than a conceptual (CDM) or logical (LDM) data model. You can operate by modelling, reverse-engineering, and generate on all the most popular DBMSs.

A physical GeOC data diagram provides a graphical view of your database structure, and helps to analyze its tables (including related columns, indexes, and triggers), views, and procedures, and the references between them.

Figure 14 shows some parts of the GeOC data model:

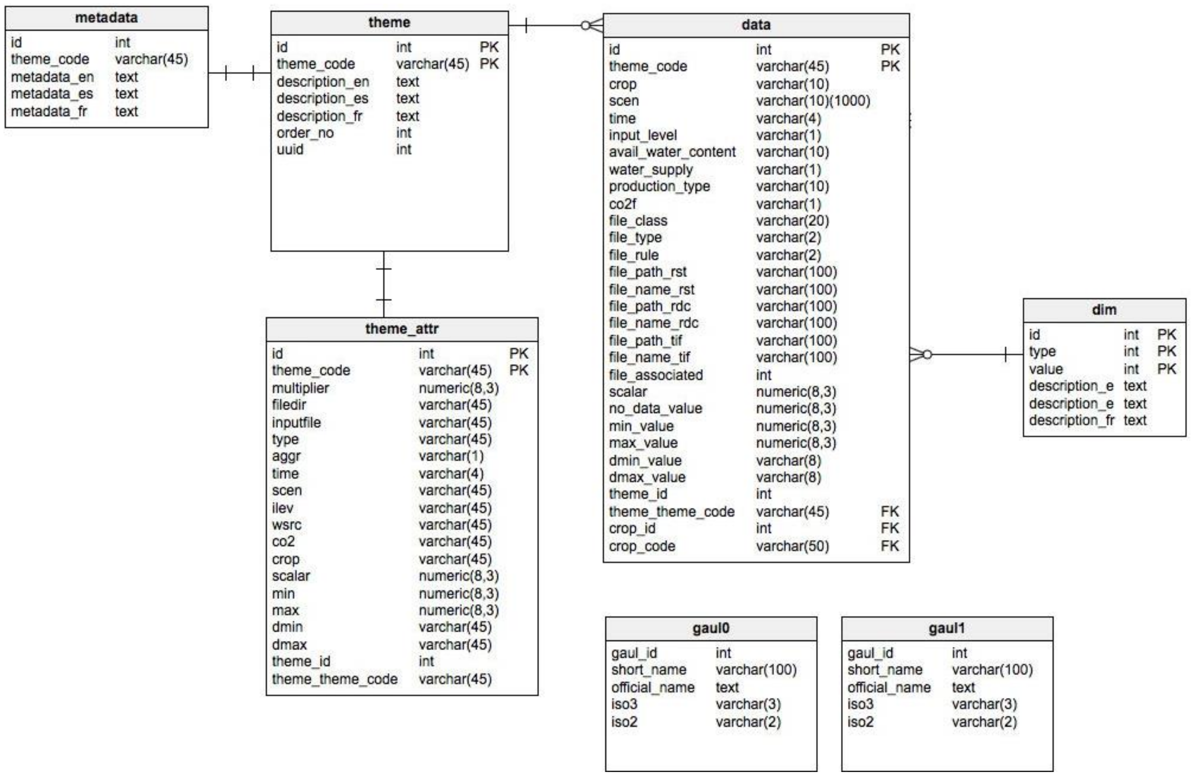
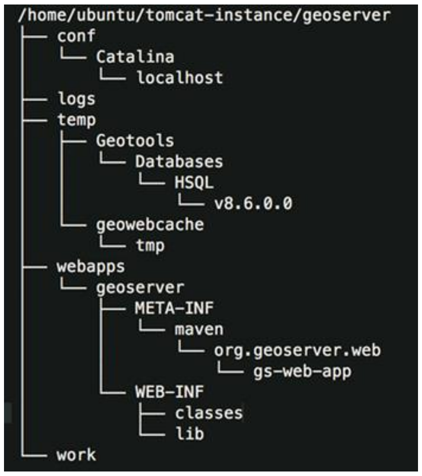
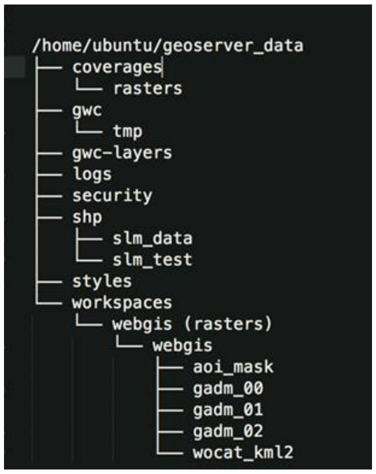
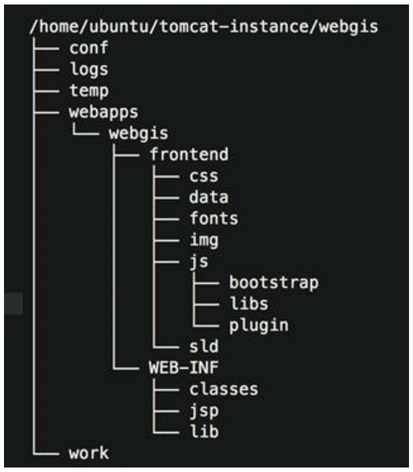


Figure 6: GeOC Data Mode

7.1 File and Database Structures

The following figures describe the usage cases for each directory:

GeoServer Directory Structure	GeoServer Data Directory Structure	GeOC Directory Structure
		

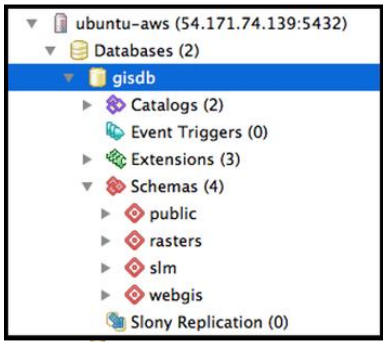
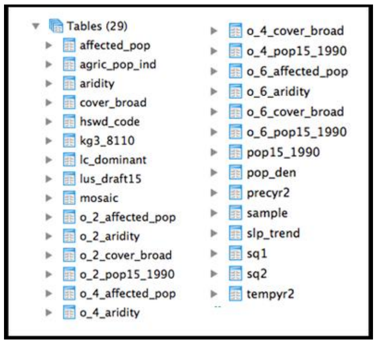
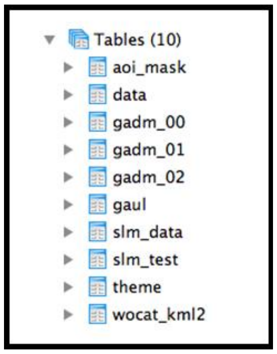
```
/home/ubuntu/tomcat-instance/geoserver
├── conf
│   └── Catalina
│       └── localhost
├── logs
├── temp
│   ├── Geotools
│   │   └── Databases
│   │       └── HSQL
│   │           └── v8.6.0.0
│   └── geowebcache
│       └── tmp
├── webapps
│   └── geoserver
│       ├── META-INF
│       │   └── maven
│       │       └── org.geoserver.web
│       │           └── gs-web-app
│       └── WEB-INF
│           ├── classes
│           └── lib
└── work
```

```
/home/ubuntu/geoserver_data
├── coverages
│   └── rasters
├── gwc
│   └── tmp
├── gwc-layers
├── logs
├── security
├── shp
│   ├── slm_data
│   └── slm_test
├── styles
└── workspaces
    └── webgis (rasters)
        └── webgis
            ├── aoi_mask
            ├── gadm_00
            ├── gadm_01
            ├── gadm_02
            └── wocat_kml2
```

```
/home/ubuntu/tomcat-instance/webgis
├── conf
├── logs
├── temp
├── webapps
│   └── webgis
│       ├── frontend
│       │   ├── css
│       │   ├── data
│       │   ├── fonts
│       │   ├── img
│       │   ├── js
│       │   │   ├── bootstrap
│       │   │   ├── libs
│       │   │   └── plugin
│       │   └── sld
│       └── WEB-INF
│           ├── classes
│           ├── jsp
│           └── lib
└── work
```

7.2 Database Management System Files

GeOC system database consists of independent and dependent tables, in order to store and split system data.

Database: gisdb Schema: rasters, slm, GeOC	Schema: rasters	Schema: GeOC
		

```
▼ ubuntu-aws (54.171.74.139:5432)
  ▼ Databases (2)
    ▼ gisdb
      ▶ Catalogs (2)
        Event Triggers (0)
      ▶ Extensions (3)
      ▼ Schemas (4)
        ▶ public
        ▶ rasters
        ▶ slm
        ▶ webgis
        Slony Replication (0)
```

```
▼ Tables (29)
  ▶ affected_pop          ▶ o_4_cover_broad
  ▶ agric_pop_ind         ▶ o_4_pop15_1990
  ▶ aridity               ▶ o_6_affected_pop
  ▶ cover_broad           ▶ o_6_aridity
  ▶ hswd_code             ▶ o_6_cover_broad
  ▶ kg3_8110              ▶ o_6_pop15_1990
  ▶ lc_dominant           ▶ pop15_1990
  ▶ lus_draft15           ▶ pop_den
  ▶ mosaic                ▶ precyr2
  ▶ o_2_affected_pop      ▶ sample
  ▶ o_2_aridity           ▶ slp_trend
  ▶ o_2_cover_broad       ▶ sq1
  ▶ o_2_pop15_1990        ▶ sq2
  ▶ o_4_affected_pop      ▶ tempyr2
  ▶ o_4_aridity
```

```
▼ Tables (10)
  ▶ aoi_mask
  ▶ data
  ▶ gadm_00
  ▶ gadm_01
  ▶ gadm_02
  ▶ gaul
  ▶ slm_data
  ▶ slm_test
  ▶ theme
  ▶ wocat_kml2
```

7.3 Non-Database Management System Files

MEL has two backups tasks performed, and are done as follows:

- A server level backup policy (daily)
- A full local database backup (every 30 Minutes)

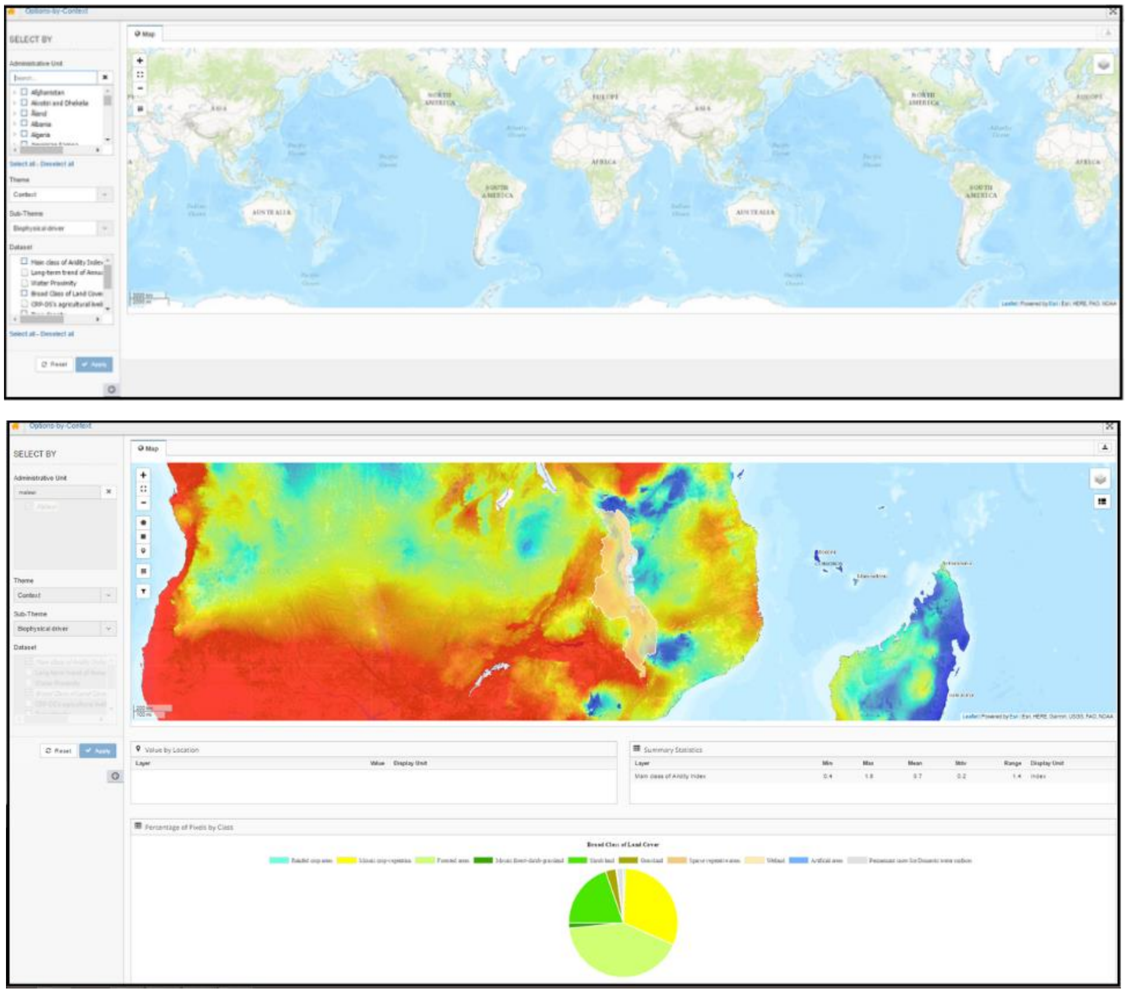
7.4 Database Information

GeOC as a single database hosted on Amazon Web Server with the following specifications:

- Database name: gisdb
- Database instance: Amazon Ubuntu linux
- Database host: 54.171.74.139
- Database user name: postgres
- Database password: adm-aws001
- Current database size: 50GB

7.5 User Interface Design

GeOC User Interface consists of: header, top menu, content, side bar, user login area, download button and other buttons and tables as shown below (Figure 15):



Layer	Value	Display Unit

Layer	Min	Max	Mean	Stdv	Range	Display Unit
Main class of Aridity Index	0.4	1.8	0.7	0.2	1.4	index

Figure 15: GeOC User Interface

8. Operational Scenarios: Data Flow Diagrams

The data flow diagram illustrates how data moves through the GeOC application, from where the data comes and where it goes and how it is stored in the three use cases: account management, query and submission of SLM.

8.1 Use Case 1: Manage the user account

To receive a user account, a request must be sent to an administrator. Once the request is approved, the user will be able to manage the settings and the assigned profile, which can always be edited by an administrator. Figure 16 shows the flow diagram user account management.

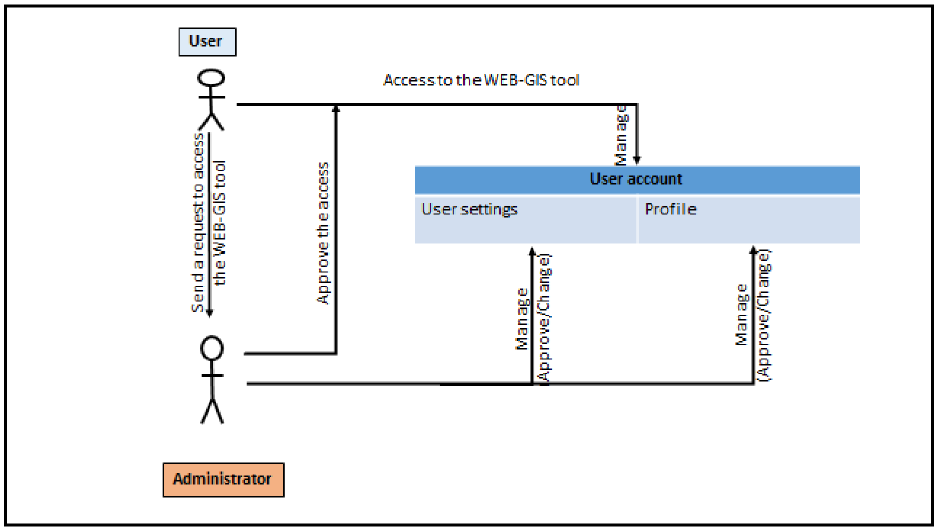


Figure 7: User account management

8.2 Use Case 2: Query of SLM

In the process of the query (Figure 17), the user needs to specify the administrative unit, theme and related sub-theme. Therefore, several datasets will be available for selection and visualization on the map. The query is not processed live and must be triggered with the button "Apply".

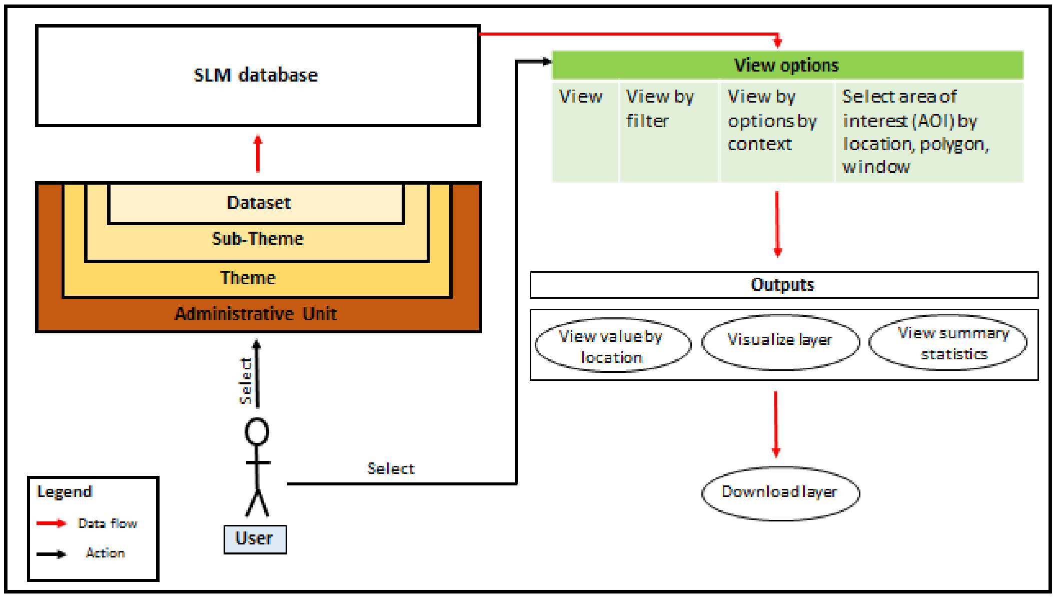


Figure 17: Query process

8.3 Use Case 3: Submission of SLM

In the process of submission of SLM (Figure 18), the user needs the approval of an Admin in ordor to integrate his submitted SLM into the SLM database. The Admin user has the right to approve, edit or initiate a discussion about the submitted SLM with the user who submit it or the other user. The process of SLM submission implies that an Admin will review the proposed dataset. Admins can approve, edit, reject or initiate a discussion about the submitted SLM with the submitter or other users.

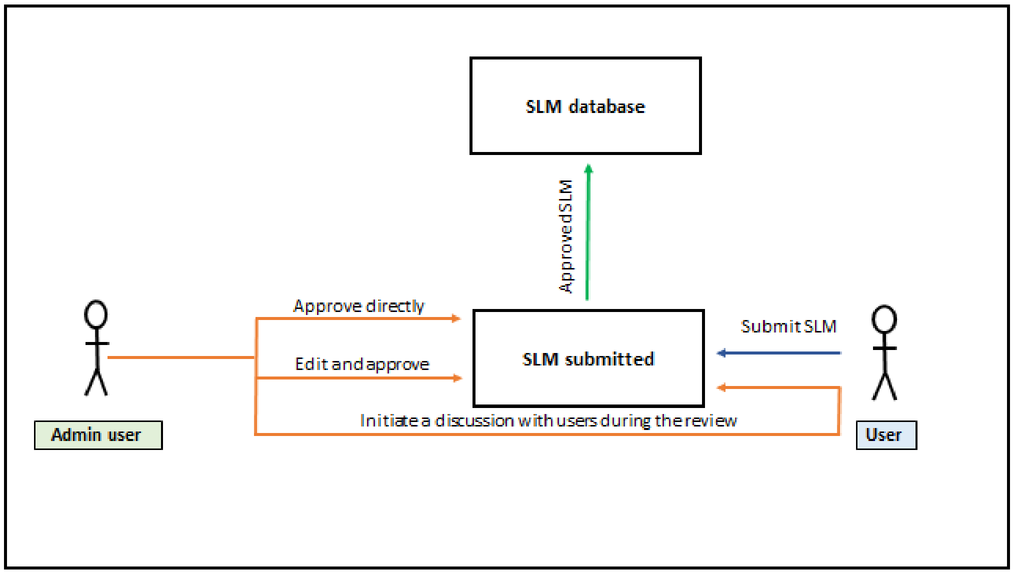


Figure 18: Submission of an SLM dataset

9. Application Implementation

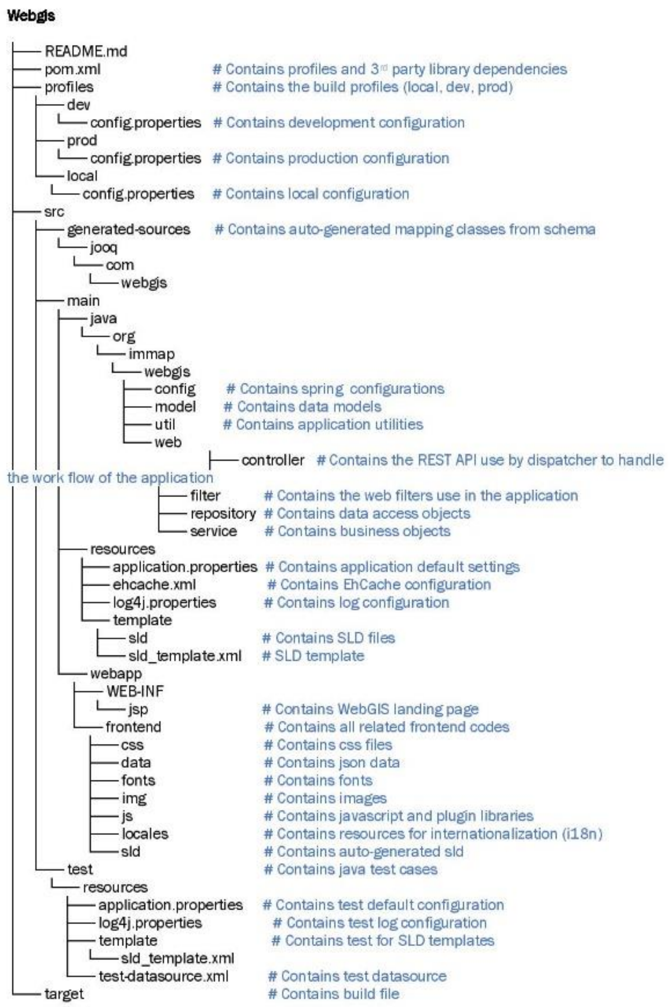
9.1 Source Codes

The WebGIS source codes is available in Bitbucket repository on the following link: https://libra007@bitbucket.org/libra007/webgis.git

The SLM Form and MEL integration code available in MEL repository: https://github.com/CodeObia/MEL

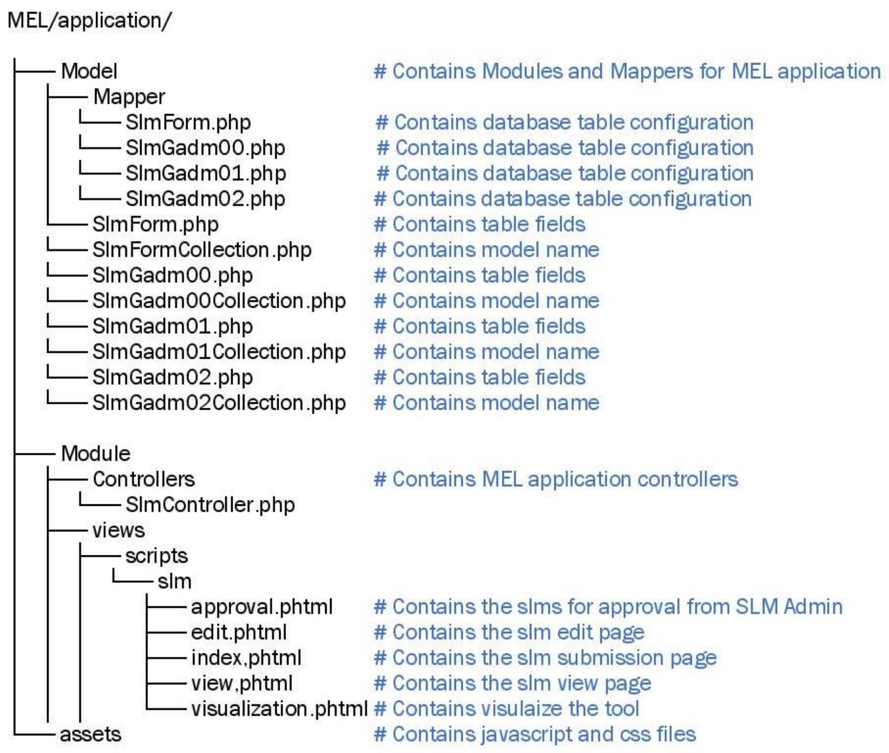
9.2 Project Structure

The project structures for the WebGIS application is arranged as the following image shows:



```
Webgis

├── README.md
├── pom.xml                      # Contains profiles and 3rd party library dependencies
├── profiles                     # Contains the build profiles (local, dev, prod)
│   ├── dev
│   │   └── config.properties    # Contains development configuration
│   ├── prod
│   │   └── config.properties    # Contains production configuration
│   └── local
│       └── config.properties    # Contains local configuration
├── src
│   ├── generated-sources        # Contains auto-generated mapping classes from schema
│   │   └── jooq
│   │       └── com
│   │           └── webgis
│   ├── main
│   │   ├── java
│   │   │   └── org
│   │   │       └── immap
│   │   │           └── webgis
│   │   │               ├── config      # Contains spring configurations
│   │   │               ├── model       # Contains data models
│   │   │               ├── util        # Contains application utilities
│   │   │               └── web
│   │   │                       ├── controller   # Contains the REST API use by dispatcher to handle
the work flow of the application
│   │   │                   ├── filter       # Contains the web filters use in the application
│   │   │                   ├── repository   # Contains data access objects
│   │   │                   └── service      # Contains business objects
│   │   ├── resources
│   │   │   ├── application.properties # Contains application default settings
│   │   │   ├── ehcache.xml            # Contains EhCache configuration
│   │   │   ├── log4j.properties       # Contains log configuration
│   │   │   └── template
│   │   │       ├── sld                # Contains SLD files
│   │   │       └── sld_template.xml   # SLD template
│   │   └── webapp
│   │       ├── WEB-INF
│   │       │   └── jsp                # Contains WebGIS landing page
│   │       └── frontend               # Contains all related frontend codes
│   │           ├── css                # Contains css files
│   │           ├── data               # Contains json data
│   │           ├── fonts              # Contains fonts
│   │           ├── img                # Contains images
│   │           ├── js                 # Contains javascript and plugin libraries
│   │           ├── locales            # Contains resources for internationalization (i18n)
│   │           └── sld                # Contains auto-generated sld
│   └── test                           # Contains java test cases
│       └── resources
│           ├── application.properties # Contains test default configuration
│           ├── log4j.properties       # Contains test log configuration
│           ├── template               # Contains test for SLD templates
│           │   └── sld_template.xml
│           └── test-datasource.xml    # Contains test datasource
└── target                             # Contains build file
```

The SLM Form folder structure is arranged as the image shows:



```
MEL/application/

├── Model                             # Contains Modules and Mappers for MEL application
│   ├── Mapper
│   │   └── SlmForm.php               # Contains database table configuration
│   │   └── SlmGadm00.php             # Contains database table configuration
│   │   └── SlmGadm01.php             # Contains database table configuration
│   │   └── SlmGadm02.php             # Contains database table configuration
│   ├── SlmForm.php                   # Contains table fields
│   └── SlmFormCollection.php         # Contains model name
│   └── SlmGadm00.php                 # Contains table fields
│   └── SlmGadm00Collection.php       # Contains model name
│   └── SlmGadm01.php                 # Contains table fields
│   └── SlmGadm01Collection.php       # Contains model name
│   └── SlmGadm02.php                 # Contains table fields
│   └── SlmGadm02Collection.php       # Contains model name
│
├── Module
│   ├── Controllers                   # Contains MEL application controllers
│   │   └── SlmController.php
│   ├── views
│   │   ├── scripts
│   │   │   └── slm
│   │   │       ├── approval.phtml    # Contains the slms for approval from SLM Admin
│   │   │       ├── edit.phtml        # Contains the slm edit page
│   │   │       ├── index.phtml       # Contains the slm submission page
│   │   │       ├── view.phtml        # Contains the slm view page
│   │   │       ├── visualization.phtml # Contains visulaize the tool
└── assets                            # Contains javascript and css files
```

9.3 Front-End

This section describes the location of the main files used in the front-end layer.

Path	Files	Description
webgis/src/main/webapp/WEB-INF/frontend/jsp	Index.jsp	The landing page of WebGIS Tool.
webgis/src/main/webapp/frontend/js	app.chart.js app.filters.js app.map.js app.map.util.js app.validator.js	These are the files use in the charts, filters and maps. The files are organized into modules for easily understand.
webgis/src/main/webapp/frontend/css	webgis.css	Main style of the application.

SLM Form Front-End Files: This section describes the location of the main files of the SLM Form used in the front-end layer.

Path	Files	Description
Mel/application/modules/default/views/scripts/slm	index.phtml edit.phtml view.phtml visualization.phtml approval.phtml	The main pages of the SLM Form
Mel/assets/admin/pages/scripts/slm	app.js table-managed.js form-wizard.js .	These are the files use in the form and to communicate with webgis tool api.

Mel/assets/admin/pages/css	slm-module.css	Main style of the form.

9.4 Back-End

Package	Files	Description
org.immap.webgis.config	AppConfig.java AppContext.java CachingConfig.java CorsFilter.java EhCacheConfiguration.java PersistenceContext.java WebMvcConfig.java	These are the application configuration.
org.immap.webgis.model	AdmUnit.java ChartData.java ChartDataList.java ClassData.java ColorMap.java Country.java Data.java Label.java Layer.java LayerClass.java LayerDownload.java LayerStats.java LayerSummary.java Polygon.java Region.java SLM.java SLMData.java SLMForm.java SLMPolygon.java SLMSimilarity.java SLMStatus.java SearchLayer.java SearchLayerClass.java Similarity.java Stats.java Theme.java	These are the Value Objects use to transfer data between frontend and backend.
org.immap.webgis.controller	ApiController.java FileDownloadController.java FrontendController.java GAULController.java ThemeController.java	These are the controller that acts on both model and view. It controls the data flow into model object and updates the view whenever data changes. It keeps view and model separate.
org.immap.webgis.service	CountryService.java RasterService.java RepositoryCountryService.java RepositoryRasterService.java RepositoryThemeService.java ThemeService.java	These are the Service Layer that helps to reduce coupling between presentation layers and persistent layers.
org.immap.webgis.repository	CountryRepository.java CountryRepositoryImpl.java RasterRepository.java RasterRepositoryImpl.java ThemeRepository.java	These are the Data Access Objects that helps to decouple the business layer from the database

	ThemeRepositoryImpl.java	thus increasing the portability of the application.

SLM Form Back-End Files

Path	Files	Description
Mel/application/modules/default/controllers	SlmController.php	The form controller
Mel/application/Model	SlmForm.php SlmFormCollection.php SlmGadm00.php SlmGadm00Collection.php SlmGadm01.php SlmGadm01Collection.php SlmGadm02.php SlmGadm02Collection.php	The form Models
Mel/application/Model/Mapper	SlmForm.php SlmGadm00.php SlmGadm01.php SlmGadm02.php	The form mappers

9.5 Configuration

This section describes the location of the configuration files used in the WebGIS tool.

- **Build configuration (Development, Staging, Production)**

The WebGIS tool is built using Maven build lifecycle framework. It handles compilation, documentation, and distribution of the project. Maven increases reusability and takes care of most of build related tasks.

The maven profiles are specified in pom.xml file using its active profile elements and are triggered in variety of ways. Profiles modify the POM at build time, and are used to give parameters different target environments like local, development, and production.

Maven build file is located in webgis/pom.xml

```
        <profiles>
                <profile>
                        <id>dev</id>
                        <activation>
                                <activeByDefault>true</activeByDefault>
                        </activation>
                        <properties>
                                <build.profile.id>dev</build.profile.id>
                                <skipMinify>true</skipMinify>
                                <suffix />
                                <warSourceExcludes />
                                <packageSourceExcludes />
                        </properties>
                </profile>

                <profile>
                        <id>staging</id>
                        <properties>
                                <build.profile.id>staging</build.profile.id>
                                <skipMinify>false</skipMinify>
                                <warSourceExcludes>
                                        frontend/css/font-google.css,
                                        frontend/css/webgis.css,
                                        frontend/js/app.chart.js,
                                        frontend/js/app.config.seed.js,
                                        frontend/js/app.filters.js,
                                        frontend/js/app.map.js,
                                        frontend/js/app.map.util.js,
                                        frontend/js/app.seed.js,
                                        frontend/js/app.validator.js
                                </warSourceExcludes>
                        </properties>
                </profile>

                <profile>
                        <id>prod</id>
                        <properties>
                                <build.profile.id>prod</build.profile.id>
                                <skipMinify>false</skipMinify>
                                <warSourceExcludes>
                                        frontend/css/font-google.css,
                                        frontend/css/webgis.css,
                                        frontend/js/app.chart.js,
                                        frontend/js/app.config.seed.js,
                                        frontend/js/app.filters.js,
                                        frontend/js/app.map.js,
                                        frontend/js/app.map.util.js,
                                        frontend/js/app.seed.js,
                                        frontend/js/app.validator.js
                                </warSourceExcludes>

                        </properties>
                </profile>

        </profiles>
```

Each profile id (highlighted in yellow) is associated with a configuration file. The configuration files are located in webgis/profiles folder.

- **Profile Id:** dev
- **Config file:** webgis/profiles/dev/config.properties

```
#jOOQ Configuration
jooq.sql.dialect=POSTGRES

#sld path
path.sld = /Users/jimjaspe/tomcat-instance/webgis/webapps/webgis/frontend/sld

#sld template
path.sld.template = /Users/jimjaspe/tomcat-instance/webgis/webapps/webgis/WEB-
INF/classes/template

# temp path
```

```
path.temp = /tmp

# server geoserver
server.host.geoserver = localhost:9000
server.host.application = localhost:8000

# MEL Login
app.mel.login = https://mel.cgiar.org/user/login

# MEL SLM viewer
app.mel.metadata = https://mel.cgiar.org/slm/view/webgisid
```

- **Profile Id:** staging
- **Config file:** webgis/profiles/staging/config.properties

```
#jOOQ Configuration
jooq.sql.dialect=POSTGRES

#sld path
path.sld = /home/webgis/tomcat-instance/webgis/webapps/webgis/frontend/sld

#sld template
path.sld.template = /home/webgis/tomcat-instance/webgis/webapps/webgis/WEB-
INF/classes/template

# temp path
path.temp = /tmp

# server geoserver
server.host.geoserver = 192.168.1.41
server.host.application = 192.168.1.41

# MEL Login
app.mel.login = https://mel.cgiar.org/user/login

# MEL SLM Metadata
app.mel.metadata = https://mel.cgiar.org/slm/view/webgisid
```

- **Profile Id:** prod
- **Config file:** webgis/profiles/prod/config.properties

```
#jOOQ Configuration
jooq.sql.dialect=POSTGRES

#sld path
path.sld = /home/ubuntu/tomcat-instance/webgis/webapps/webgis/frontend/sld

#sld template
path.sld.template = /home/ubuntu/tomcat-instance/webgis/webapps/webgis/WEB-
INF/classes/template

# temp path
path.temp = /tmp

# server geoserver
server.host.geoserver = webgis.world
server.host.application = webgis.world
```

```
# MEL Login
app.mel.login = https://mel.cgiar.org/user/login

# MEL SLM metadata
app.mel.metadata = https://mel.cgiar.org/slm/view/webgisid
```

- **Database configuration**

The WebGIS tool use the Java^{TM} Naming and Directory Interface (JNDI) data source for database connection pooling. The context configuration file below must be copied in the **$CATALINA_BASE/conf** directory.

Source file: ~/tomcat-instance/webgis/conf/context.xml

```
<Context>
<Resource name="jdbc/gisdb"
      auth="Container"
      type="javax.sql.DataSource"
      driverClassName="org.postgresql.Driver"
      url="jdbc:postgresql://localhost:5432/gisdb"
      username="postgres"
      password="xxxxxxx"
      maxTotal="100"
      initialSize="0"
      minIdle="0"
      maxIdle="30"
      maxWaitMillis="10000"
      timeBetweenEvictionRunsMillis="30000"
      minEvictableIdleTimeMillis="60000"
      testWhileIdle="true"
      validationQuery="SELECT 1"
    />
<Context>
```

NOTE: To be able to access the database server it will be necessary to change the credentials highlighted above

- **Cache configuration**

The WebGIS tool is configured to use EhCache for better scalability and improve the application performance.

- **Configuration file:** webgis/src/main/resources/echcache.xml

```
<ehcache xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="ehcache.xsd"
    updateCheck="true"
    monitoring="autodetect"
    dynamicConfig="true">

    <diskStore path="user.home/tomcat-instance/webgis/cache"/>

    <defaultCache
        maxElementsInMemory="1"
        eternal="true"
        overflowToDisk="true"
        diskPersistent="true"
        memoryStoreEvictionPolicy="LRU" />
```

```
       <Cache name="findAllGADM_00"
         maxElementsInMemory="1"
         eternal="true"
         overflowToDisk="true"
         diskPersistent="true"
         memoryStoreEvictionPolicy="LRU" />

       <Cache name="findAllRegions"
         maxElementsInMemory="1"
         eternal="true"
         overflowToDisk="true"
         diskPersistent="true"
         memoryStoreEvictionPolicy="LRU" />

        <Cache name="findSubRegionById"
         maxElementsInMemory="1"
         eternal="true"
         overflowToDisk="true"
         diskPersistent="true"
         memoryStoreEvictionPolicy="LRU" />

       <Cache name="findCountriesByRegions"
         maxElementsInMemory="1"
         eternal="true"
         overflowToDisk="true"
         diskPersistent="true"
         memoryStoreEvictionPolicy="LRU" />

      <Cache name="findSelectionByAOI"
         maxElementsInMemory="1"
         eternal="true"
         overflowToDisk="true"
         diskPersistent="true"
         memoryStoreEvictionPolicy="LRU" />

     <Cache name="findStatsByAOI"
         maxElementsInMemory="1"
         eternal="true"
         overflowToDisk="true"
         diskPersistent="true"
         memoryStoreEvictionPolicy="LRU" />

         more ....

</ehcache>
```

The DiskStore provides a disk spooling facility that can be used for additional storage during cache operation and for persisting caches through system restarts.

The <diskStore> element has one attribute called “path” that tells the directory location to create the cache.

- **Cache management**

Flushing the WebGIS application cache is necessary when the following tables are modified (theme, data, gadm_01, gadm_02, gadm_03, continent, slm_data)

To flush the cache, just delete the content of the "cache" folder. It's not advisable to delete a subset of it as the data are linked to each other and the caches are smaller in size.

- **Log configuration**

WebGIS tool is configured to use SLF4j logging framework to handle all application logs into different levels.

- **Configuration file:** webgis/src/main/resources/log4j.properties

```
log4j.appender.Stdout=org.apache.log4j.ConsoleAppender
log4j.appender.Stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.Stdout.layout.conversionPattern=%-5p - %-26.26c{1} - %m\n

log4j.rootLogger=INFO,Stdout
log4j.logger.org.springframework=INFO
log4j.logger.org.dbunit=OFF
```

- **Backup and Restore**

Backing up the WebGIS database can be done via command line below:

Syntax: pg_dump -U postgres -h <hostname> -n <target_schema> <database> > <output_file>

where: -U = user

-h = hostname

-n = schema

For reference, please follow the link: https://www.postgresql.org/docs/9.4/static/app-pgdump.html

To backup webgis database:

pg_dump -U postgres -h localhost -n webgis gisdb > webgis.sql

To restore webgis backup:

psql -U postgres -h localhost -d gisdb < webgis.sql

Note: If space is not an issue, you can also backup the "rasters" schema which contains the raster datasets.

10. Detailed Installation and Design

The WebGIS can be installed on local servers or on Amazon Web Services (AWS) linux instances. The installation procedures are the same on the local server and on AWS. Only the type of connection to the server is different and depends on the platform chosen by the user.

This part is the WebGIS Installation Guide for deployment on AWS Ubuntu instance. It describes the complete installation procedures for the WebGIS application. The document scope is to support the end user (typically the system administrator, the database administrator and geodatabase administrator roles) to install, maintain and monitor the WebGIS application. The document covers all aspect of the installation including all various software components needed and it is needed, and how to deploy the web services in your machine.

10.1 Prerequisites for a New Installation of the WebGIS application

The following procedures are intended for use with Amazon Web Services Ubuntu Instance. It is assumed that the user has already launched an Amazon Web instance, has created a public DNS that is reachable from the internet. The security group(s) are configured to allow access through the Secured Shell (SSH) on the default port 22, HTTP is enabled on the default port 80, and HTTPS (port 443) connections are enabled.

10.2 System Requirements

Please ensure that your system meets the minimum requirements.

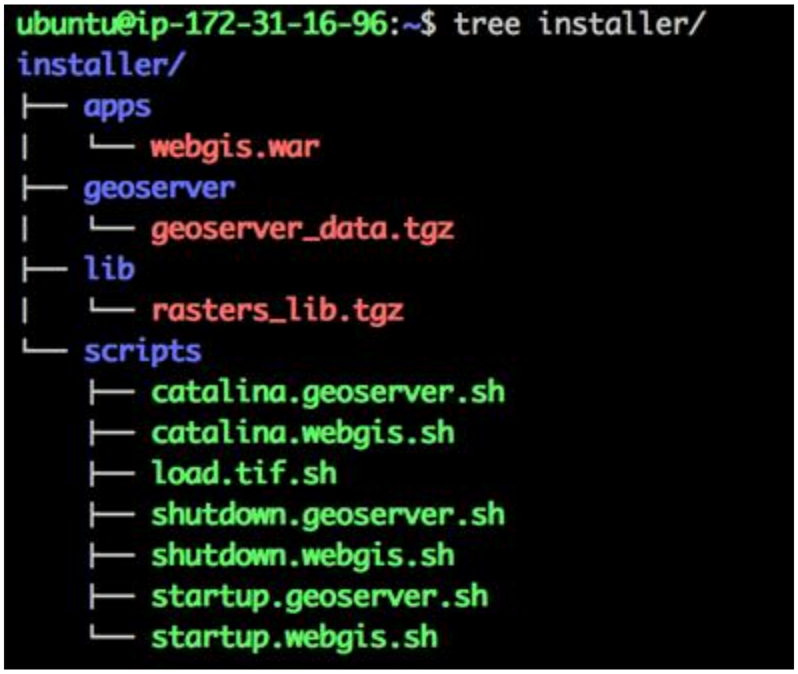
Amazon Instance Server	Third-party Applications	Web Browsers
• Compatible Operating System: Linux Operating System x64 (Ubuntu 16.04.1 LTS Xenial) • Processor(s): minimum 2.5GHz per server core (3.0Ghz or faster recommended): Quad Core (Optimal 8 cores) • Minimum 16GB RAM: 32GB recommended for large queries • Minimum of 50GB available HDD space: SSD volume type recommended	• Java Runtime Environment 8 • Apache Tomcat 8.5.6 • PostgreSQL 9.5 • PostGIS 2.3.0 (Geos 3.5/ Proj 4.9.2/ Gdal 1.11.3/ LibXML 2.9.3) • Apache 2.4	• 1024x768 minimum screen resolution (optimal viewing requires higher resolution) • Compatible browsers: IE 10 or Newer, Latest Chrome, Latest Firefox, Explorer and Apple Safari. • To get the best-possible experience, we recommend using a browser that has full HTML-5 compatibility (Chrome, Firefox, IE10, IE11)

You need to verify the distribution files. I don't see the link between these sentences. This goes to the preparation to verify that you have the intaller so before the connection.

The installer directory contains (normally/ must?) the following files:

- WebGIS application archive – webgis.war
- GeoServer extension libraries for Rasters and ImageMosaic – rasters_lib.tgz
- Tomcat scripts for GeoServer and WebGIS applications – scripts

- Geoserver Data with Sample Datasets – geoserver_data.tgz



```
ubuntu@ip-172-31-16-96:~$ tree installer/
installer/
├── apps
│   └── webgis.war
├── geoserver
│   └── geoserver_data.tgz
├── lib
│   └── rasters_lib.tgz
└── scripts
    ├── catalina.geoserver.sh
    ├── catalina.webgis.sh
    ├── load.tif.sh
    ├── shutdown.geoserver.sh
    ├── shutdown.webgis.sh
    ├── startup.geoserver.sh
    └── startup.webgis.sh
```

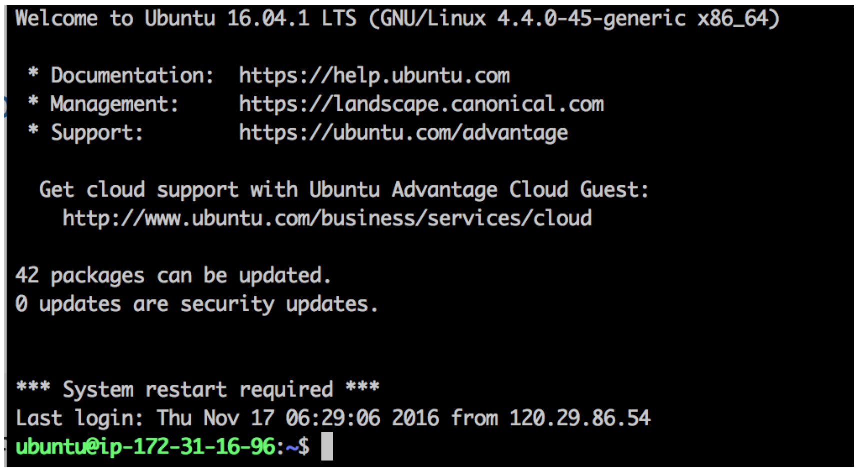
10.3 Preparing to Install

Before you begin the installation, verify that you are able to connect to your Linux instance.

```
$ ssh -i WebGISkey.pem ubuntu@54.171.74.139
```

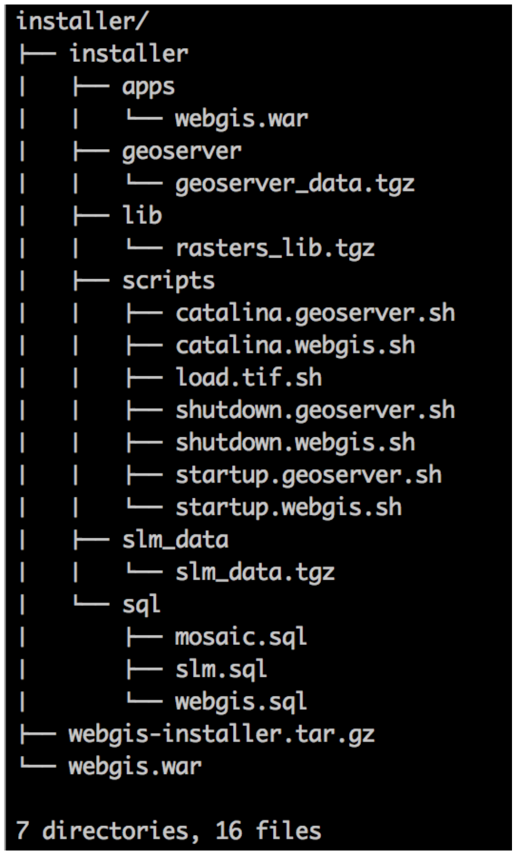
10.3.1. Step 1: Connecting to your Amazon Linux Instance using SSH

If connected using the previous code, you will see a response like the following then continue to the next step.



```
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-45-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud

42 packages can be updated.
0 updates are security updates.


*** System restart required ***
Last login: Thu Nov 17 06:29:06 2016 from 120.29.86.54
ubuntu@ip-172-31-16-96:~$
```

Verify the distribution files: The installer directory contains the following files.



```
installer/
├── installer
|   ├── apps
|   |   └── webgis.war
|   ├── geoserver
|   |   └── geoserver_data.tgz
|   ├── lib
|   |   └── rasters_lib.tgz
|   ├── scripts
|   |   ├── catalina.geoserver.sh
|   |   ├── catalina.webgis.sh
|   |   ├── load.tif.sh
|   |   ├── shutdown.geoserver.sh
|   |   ├── shutdown.webgis.sh
|   |   ├── startup.geoserver.sh
|   |   └── startup.webgis.sh
|   ├── slm_data
|   |   └── slm_data.tgz
|   └── sql
|       ├── mosaic.sql
|       ├── slm.sql
|       └── webgis.sql
├── webgis-installer.tar.gz
└── webgis.war

7 directories, 16 files
```

- WebGIS application archive – **webgis.war**
- GeoServer extension libraries for Rasters and ImageMosaic – **rasters_lib.tgz**
- Tomcat scripts for GeoServer and WebGIS applications – **scripts**
- Geoserver Data with Sample Datasets – **geoserver_data.tgz**
- SLM Data – **slm_data.tgz**

10.3.2. Step 2: Configuring the Java Platform

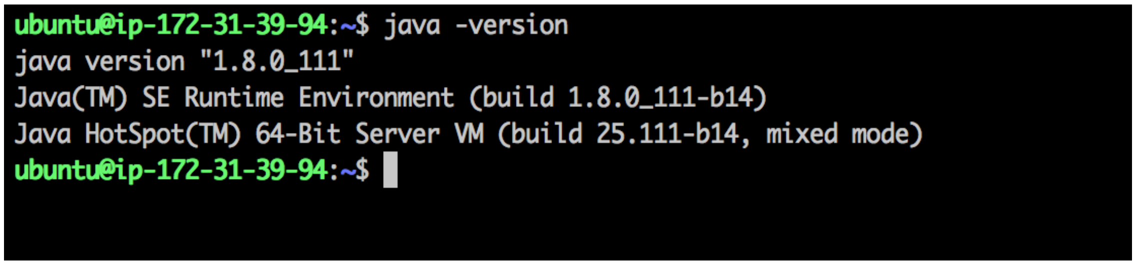
This step requires that you install the Oracle JDK 8 first. For that, you have to add Oracle's PPA. First, then update your package repository.

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
```

Next, you have to install the latest stable version of Java at time of writing and the recommended version to install. You can do it using the following command:

```
$ sudo apt-get install oracle-java8-installer
```

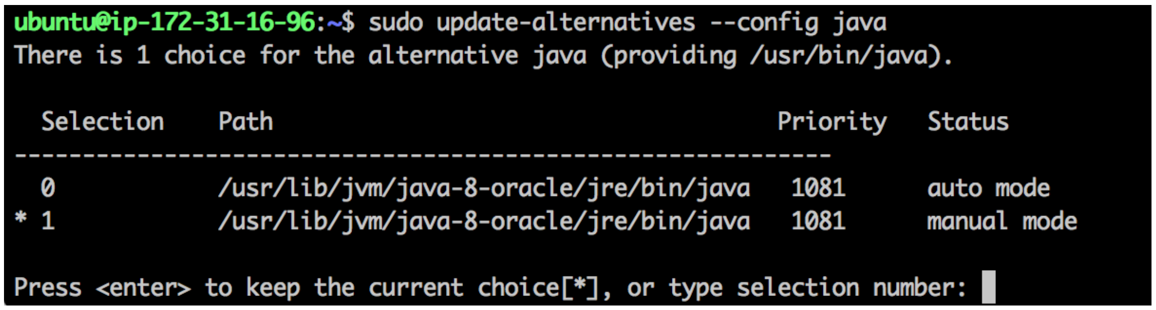
When the installation is done, check the version of Java again by typing "java –version" at the prompt and press Enter. You will have a similar message like the following figure shows. It also indicates the version of Java you just installed.



```
ubuntu@ip-172-31-39-94:~$ java -version
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)
ubuntu@ip-172-31-39-94:~$
```

10.3.3. Step 3: Setting the JAVA_HOME environment variable

To set this environment variable, you need first to find out where Java is installed. You can do this by executing this command:



```
ubuntu@ip-172-31-16-96:~$ sudo update-alternatives --config java
There is 1 choice for the alternative java (providing /usr/bin/java).

  Selection    Path                                     Priority   Status
------------------------------------------------------------
  0            /usr/lib/jvm/java-8-oracle/jre/bin/java   1081      auto mode
* 1            /usr/lib/jvm/java-8-oracle/jre/bin/java   1081      manual mode

Press <enter> to keep the current choice[*], or type selection number:
```

You copy the path from your preferred installation and then you open /etc/environment using vi or your favourite text editor.

```
$ sudo vi /etc/environment
```

At the end of this file, you add the following line. You make sure to replace the highlighted path with your own copied path.

```
/etc/environment
JAVA_HOME="/usr/lib/jvm/java-8-oracle/jre/bin/java"
```

You save, exit the file, and reload it.

```
$ source /etc/environment
```

10.3.4. Step 4: Install Maven environment

Open a terminal window and excute the following command.

```
$ sudo apt-get update
$ sudo apt-get install maven
```

We can verify whether Maven is installed successfully or not by typing the command:

```
$ mvn –v

Apache Maven 3.3.9
Maven home: /usr/share/maven
Java version: 1.8.0_111, vendor: Oracle Corporation
Java home: /usr/lib/jvm/java-8-oracle/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "4.4.0-45-generic", arch: "amd64", family: "unix"
```

10.3.5. Step 5: Build WebGIS

To build the WebGIS project you need to change directory to webgis then execute the command below.

Syntax: mvn clean package -DskipTests –P<dev|staging|prod>

Where: clean – cleans up artifacts created by prior builds.

package – take the compiled code and package it to war file.

-DskipTests – skip all test cases

-P – Profile to build (dev, staging, prod)

```
$ mvn clean package -DskipTests -Pdev
```

```
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building webgis 1.0-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ webgis ---
[INFO] Deleting /Users/jimjaspe/Documents/workspace-prj/webgis/target
[INFO]
[INFO] --- properties-maven-plugin:1.0.0:read-project-properties (default) @ webgis ---
[INFO]
[INFO] --- jooq-codegen-maven:3.8.3:generate (default) @ webgis ---
[INFO] License parameters
[INFO] ----------------------------------------------------------
[INFO]   Thank you for using jOOQ and jOOQ's code generator
[INFO]
[INFO] Database parameters
[INFO] ----------------------------------------------------------
[INFO]   dialect                : POSTGRES
[INFO]   URL                    : jdbc:postgresql://localhost:5432/gisdb
[INFO]   target dir             : /Users/jimjaspe/Documents/workspace-prj/webgis/src/generated-
sources/jooq
[INFO]   target package         : com.webgis.jooq.model
[INFO]   includes               : [.*]
[INFO]   excludes               : []
[INFO]   includeExcludeColumns  : false
[INFO] ----------------------------------------------------------
[INFO]
[INFO] DefaultGenerator parameters
[INFO] ----------------------------------------------------------
[INFO]   strategy               : class org.jooq.util.DefaultGeneratorStrategy
[INFO]   deprecated             : true
[INFO]   generated annotation   : true
[INFO]   JPA annotations        : false
[INFO]   validation annotations : false
[INFO]   instance fields        : true
[INFO]   records                : true
[INFO]   pojos                  : false
[INFO]   immutable pojos        : false
[INFO]   interfaces             : false
[INFO]   daos                   : false
[INFO]   relations              : true
[INFO]   table-valued functions : false
[INFO]   global references      : true
[INFO] ----------------------------------------------------------
[INFO]
[INFO] Generation remarks
[INFO] ----------------------------------------------------------
[INFO]   none
[INFO]
[INFO] ----------------------------------------------------------
[INFO] Generating catalogs      : Total: 1
[INFO] No schema version is applied for catalog . Regenerating.
[INFO]


[INFO] Generating catalog       : DefaultCatalog.java
[INFO] ==========================================================
[INFO]

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@  @@        @@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@        @@@@@@@@@@
@@@@@@@@@@@@@@@@  @@  @@    @@@@@@@@@@
@@@@@@@@@@  @@@@  @@  @@    @@@@@@@@@@
@@@@@@@@@@        @@        @@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@        @@        @@@@@@@@@@
@@@@@@@@@@    @@  @@  @@@@  @@@@@@@@@@
@@@@@@@@@@    @@  @@  @@@@  @@@@@@@@@@
@@@@@@@@@@        @@  @  @  @@@@@@@@@@
@@@@@@@@@@        @@        @@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@  @@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@  Thank you for using jOOQ 3.8.3

 [INFO]
[INFO] --- maven-war-plugin:2.5:exploded (default) @ webgis ---
[INFO] Exploding webapp
```

```
[INFO] Assembling webapp [webgis] in [/Users/jimjaspe/Documents/workspace-prj/webgis/target/webgis]
[INFO] Processing war project
[INFO] Copying webapp resources [/Users/jimjaspe/Documents/workspace-prj/webgis/src/main/webapp]
[INFO] Webapp assembled in [1331 msecs]
[INFO]
[INFO] --- replacer:1.5.3:replace (default) @ webgis ---
[INFO] Replacement run on 3 files.
[INFO]
[INFO] --- maven-war-plugin:2.5:war (default-war) @ webgis ---
[INFO] Packaging webapp
[INFO] Assembling webapp [webgis] in [/Users/jimjaspe/Documents/workspace-prj/webgis/target/webgis]
[INFO] Processing war project
[INFO] Copying webapp resources [/Users/jimjaspe/Documents/workspace-prj/webgis/src/main/webapp]
[INFO] Webapp assembled in [52 msecs]
[INFO] Building war: /Users/jimjaspe/Documents/workspace-prj/webgis/target/webgis.war
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 18.378 s
[INFO] Finished at: 2017-04-05T21:45:14+08:00
[INFO] Final Memory: 36M/481M
[INFO] ------------------------------------------------------------------------
```

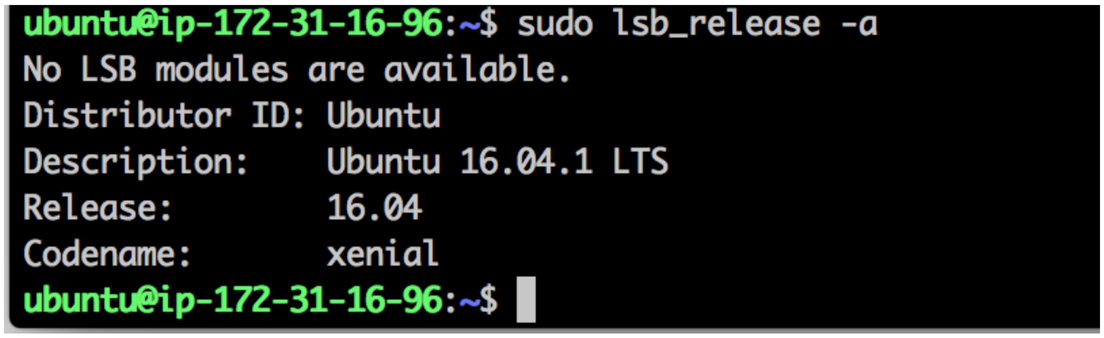
When build is successful, the web archive will be created in “target/webgis.war” folder.

10.3.6. Step 6: Configuring the Database Server

The following describes how to install PostgreSQL 9.5, PostGIS 2.2, pgRouting on Ubuntu version 16.04. Please verify what version of Ubuntu you are running.

```
$ sudo lsb release -a
```

The output should be as the following figure shows:



```
ubuntu@ip-172-31-16-96:~$ sudo lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 16.04.1 LTS
Release:        16.04
Codename:       xenial
ubuntu@ip-172-31-16-96:~$
```

For Ubuntu release 16.04, the codename is “xenial”. You add Respository to sources.list. You will need to replace the codename below with what you are running.

```
$ sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt xenial-pgdg main" >> /etc/apt/sources.list'
```

You add Keys.

```
$ wget --quiet -O - http://apt.postgresql.org/pub/repos/apt/ACCC4CF8.asc | sudo apt-key add -
$ sudo apt-get update
```

10.3.7. Step 7: Installing PostgresSQL and PostGIS

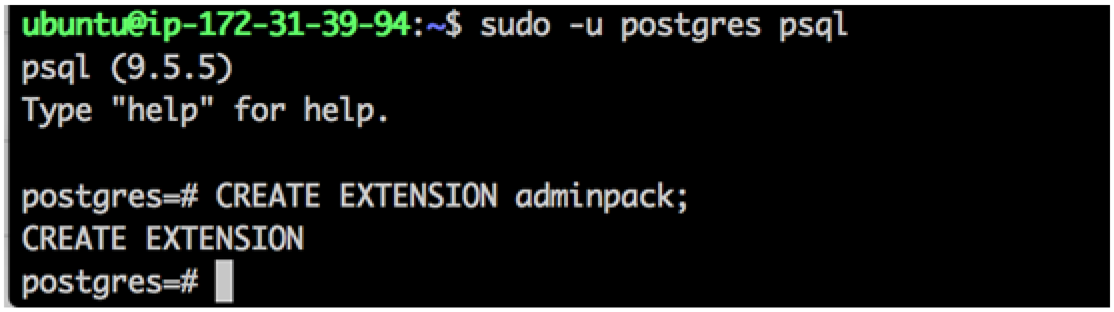
The following command will allow you to install postgresql 9.5, PostGIS 2.2, pg Routing 2.1 and additional supplied modules including the **adminpack** extension:

```
$ sudo apt-get install postgresql-9.5-postgis-2.2 postgresql-contrib-9.5
```

You install pg Routing 2.1 package.

```
$ sudo apt-get install postgresql-9.5-postgis-2.2 postgresql-contrib-9.5
```

You enable **adminpack**: While in terminal, you log in to the psql console as postgres user:



```
ubuntu@ip-172-31-39-94:~$ sudo -u postgres psql
psql (9.5.5)
Type "help" for help.

postgres=# CREATE EXTENSION adminpack;
CREATE EXTENSION
postgres=#
```

In order to successfully show the commands above, you need to run locale to list what locales are currently defined for the current user account:

```
$ locale

LANG=C
LANGUAGE=
LC_CTYPE=f_FI.UTF-8
LC_NUMERIC="C"
LC_TIME="C"
LC_COLLATE=f_FI.UTF-8
LC_MONETARY="C"
LC_MESSAGES=f_FI.UTF-8
LC_PAPER="C"
LC_NAME="C"
LC_ADDRESS="C"
LC_TELEPHONE="C"
LC_MEASUREMENT="C"
LC_IDENTIFICATION="C"
```

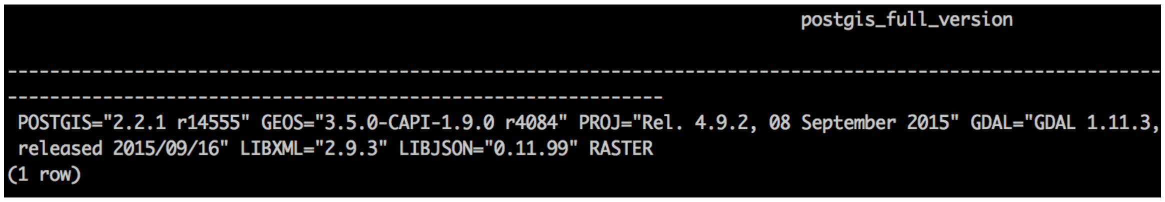
```
$ sudo locale-gen "en_US.UTF-8"
Generating locales...
en_US.UTF-8... done
Generation complete.
$ sudo dpkg-reconfgure locales
Generating locales...
en_US.UTF-8... up-to-date
Generation complete.
```

10.3.8. Step 8: Creating a database and enabling PostGIS extension and GDAL drivers

The following command will allow you to enable PostGIS and GDAL.

```
postgres=# CREATE DATABASE gisdb;
postgres=# \connect gisdb;

postgres=# CREATE EXTENSION postgis;
postgres=# SELECT postgis_full_version();
```

The output should be as the following figure shows:

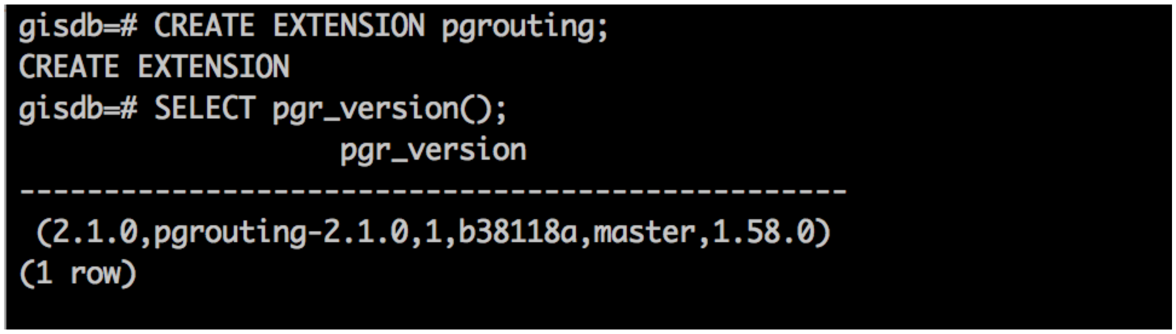


```
                                                                                   postgis_full_version
-------------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------
 POSTGIS="2.2.1 r14555" GEOS="3.5.0-CAPI-1.9.0 r4084" PROJ="Rel. 4.9.2, 08 September 2015" GDAL="GDAL 1.11.3,
 released 2015/09/16" LIBXML="2.9.3" LIBJSON="0.11.99" RASTER
(1 row)
```

You enable pgRouting extension.

```
postgres=# CREATE EXTENSION pgrouting;
postgres=# SELECT pgr_version();
```

The output should be like the following figure shows:

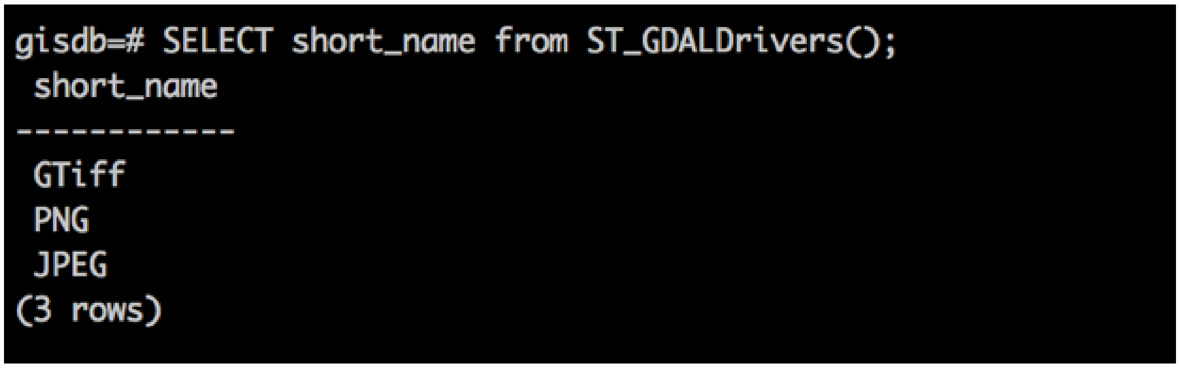


```
gisdb=# CREATE EXTENSION pgrouting;
CREATE EXTENSION
gisdb=# SELECT pgr_version();
                        pgr_version
--------------------------------------------------
 (2.1.0,pgrouting-2.1.0,1,b38118a,master,1.58.0)
(1 row)
```

You enable GDAL Drivers to GTiff, PNG, JPEG.

```
gisdb=# ALTER DATABASE gisdb SET postgis.gdal_enabled_drivers TO 'GTiff PNG JPEG';

gisdb=# SELECT short_name FROM ST_GDALDrivers();
```

The output should be like the following figure shows:



```
gisdb=# SELECT short_name from ST_GDALDrivers();
 short_name
------------
 GTiff
 PNG
 JPEG
(3 rows)
```

In order to successfully show the command above, the user must follow the commands below :

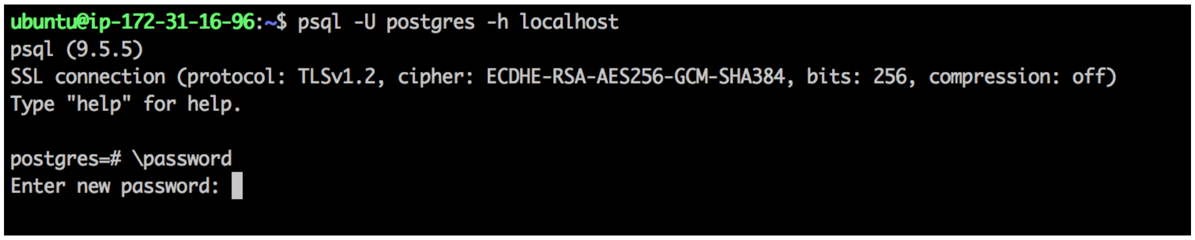
```
sudo nano /etc/postgresql/9.5/main/environment
add:
POSTGIS_GDAL_ENABLED_DRIVERS=ENABLE_ALL
POSTGIS_ENABLE_OUTDB_RASTERS=1
Ctrl+X-Y-Enter
sudo service postgresql restart
```

Then, you exit psql console:

```
postgres=# \q
```

10.3.9. Step 9: Changing PostgreSQL password

While in terminal, you log in to the psql console as postgres user:



```
ubuntu@ip-172-31-16-96:~$ psql -U postgres -h localhost
psql (9.5.5)
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256, compression: off)
Type "help" for help.

postgres=# \password
Enter new password:
```

For this setup, postgres user password is set to “adm-aws001”. In case the user is reinstalling the webgis, he should first reset the password using the following command:

```
sudo -U postgres psql
```

You should see: postgres=# \password

Enter new password: Then enter a new password using the password mentioned above.

Then, you exit psql console:

```
postgres=# \q
```

10.3.10. Step 10: Creating and using a .pgpass file

A .pgpass file will allow you to use postgres CLI tools such as psql and pg_dump without having to manually enter a password. To create the .pgpass file, you execute the command below.

```
$ cd ~\
> echo localhost:*:*:postgres:adm-aws001 > .pgpass
$ chmod 600 .pgpass
```

To use postgres command line, you can use either "-w" or "--no-password". It will never issue a password prompt.

10.3.11. Step 11: Setting up PostgresSQL for remote connection (optional)

Following the steps below, you will be able to enable remote access to PostgreSQL database server. First, relevant IP addresses of remote machines that are supposed to have access to the database need to be added to the **pg_hba.conf** configuration. You may need to edit pg_hba.conf and postgresql.conf to allow external access.

```
$ sudo vi /etc/postgresql/9.5/main/pg_hba.conf
```

At the end of the file, you add the following line and save your changes.

P.S: you replace “0.0.0.0/0” with the relevant IP addresses of the remote machines.

```
host all all  0.0.0.0/0 md5
```

You change the address PostgreSQL listens to:

```
$ sudo vi /etc/postgresql/9.5/main/postgresql.conf
```

We're specifically looking for a line that says “listen_addresses”. We're going to modify where PostgreSQL is listening from “localhost” to all (“*”).

```
listen_addresses = '*'
```

You restart PostgresSQL for these changes to take place.

```
$ sudo vi /etc/postgresql/9.5/main/postgresql.conf
```

NOTE: *PostgreSQL server is listening on (default: 5432), must be added to the exception lists of all firewalls that are between the server and the web. Otherwise interactions with remote machines are consequently blocked and the database cannot be reached from outside.*

10.3.12. Step 12: Configuring the Web Application Server

This step describes how to install Apache Tomcat, and to create instances for WebGIS and GeoServer application.

Installing Apache Tomcat:

First, you change to your home directory:

```
$ cd ~
```

Then use **wget** and paste in the link to download the Tomcat 8 archive. User can find the latest version of Tomcat 8 at the Tomcat 8 Downloads page. At the time of writing, the latest version is **8.5.8**. Under the **Binary Distributions** section, you find the **Core** list. Under this, you copy the link to the "tar.gz". Then you download the latest binary distribution to your home directory.

```
$ wget http://www-eu.apache.org/dist/tomcat/tomcat-8/v8.5.8/bin/apache-tomcat-8.5.8.tar.gz
```

Tomcat will be installed in the ~/tomcat directory. You create the directory, then extract the archive with these commands:

```
$ mkdir tomcat
$ tar xzvf apache-tomcat-8.5.8.tar.gz -C tomcat --strip-components=1
```

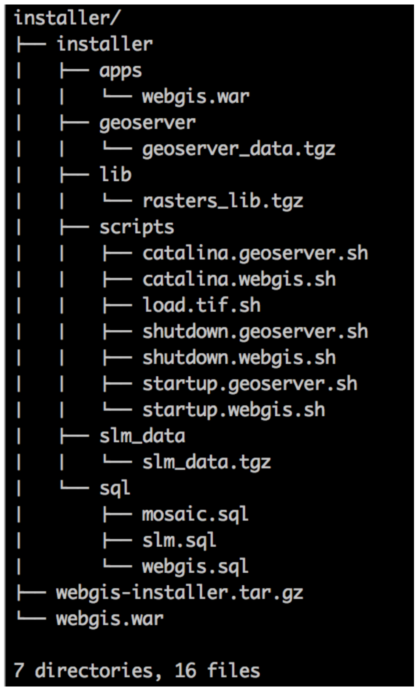
Next, you need to copy the scripts to tomcat/bin folder.

```
$ cp webgis/scripts/* ~/tomcat/bin
```

The version of Tomcat 8.5.8 may be different from your latest version. Note that you can replace this version by your latest.

Before typing the command below, please make sure to have the installer directory. To install directory, please use the following command:

```
$ mkdir installer
$ cd installer
$ tar xzvf ~/webgis.installer.tar.gz
```



```
installer/
├── installer
|   ├── apps
|   |   └── webgis.war
|   ├── geoserver
|   |   └── geoserver_data.tgz
|   ├── lib
|   |   └── rasters_lib.tgz
|   ├── scripts
|   |   ├── catalina.geoserver.sh
|   |   ├── catalina.webgis.sh
|   |   ├── load.tif.sh
|   |   ├── shutdown.geoserver.sh
|   |   ├── shutdown.webgis.sh
|   |   ├── startup.geoserver.sh
|   |   └── startup.webgis.sh
|   ├── slm_data
|   |   └── slm_data.tgz
|   └── sql
|       ├── mosaic.sql
|       ├── slm.sql
|       └── webgis.sql
├── webgis-installer.tar.gz
└── webgis.war

7 directories, 16 files
```

Next, because of the command used above making a directory installed inside another directory, we use the following command:

```
$ cd ~/installer/installer/geoserver
```

If you are using Amazon Linux (Ubuntu 16.04) in the cloud, the command below works without problem. But if you do not have the Amazon Linux (Ubuntu 16.04) in the cloud, the 'Cxzvf' will be replaced by 'xzvf'.

```
$ tar Cxzvf ~/installer/installer/geoserver~/geoserver_data_tgz
```

Creating Tomcat Instances: You have to create the directory for geoserver and webgis instance first.

```
$ cp ~/installer/scripts/* ~/tomcat/bin
```

Next, you change to tomcat directory and copy the copy conf, temp, webapps and logs to tomcat-instance/geoserver and tomcat-instance/webgis folder.

```
$ mkdir tomcat-instance
$ mkdir tomcat-instance/{geoserver,webgis}
```

```
$ cd tomcat
$ cp conf /  temp/  webapps/   logs/   -R  ~/tomcatinstance/geoserver
$ cp  conf /  temp/  webapps/  logs/ -R  ~/tomcatinstance/webgis
```

Configuring the WebGIS Application Port

The WebGIS application port is configured in ~/tomcat-instance/webgis/conf/server.xml.

You set the connector port to 8000.

```
<Connector port="8000" protocol="HTTP/1.1"
connectionTimeout="20000"
redirectPort="8443" />
```

You set the Server port to 8005.

```
<Server port="8005" shutdown="SHUTDOWN">
```

You save the server.xml and exit.

Create JNDI DataSource for Database Connection:

To declare a JNDI Datasource for PostgreSQL, you need to modify context.xml in ~/tomcat-instance/geoserver/conf and ~/tomcat-instance/webgis/conf folder. You add the following lines between the <context> </context> tag.

```
$ nano ~/tomcat-instance/geoserver/conf/context.xml
```

```
<Resource name="jdbc/gisdb"
     auth="Container"
     type="javax.sql.DataSource"
     driverClassName="org.postgresql.Driver"
     url="jdbc:postgresql://localhost:5432/gisdb"
     username="postgres" password="adm-aws001"
     maxTotal="100"
     initialSize="0"
     minIdle="0"
     maxIdle="30"
     maxWaitMillis="10000"
     timeBetweenEvictionRunsMillis="30000"
     minEvictableIdleTimeMillis="60000"
     testWhileIdle="true"
     validationQuery="SELECT 1" />

</Context>
```

You save **context.xml** file and exit.

10.3.13. Step 13: Deploying Web-GIS Application

This section describes how to deploy Web-GIS application after you configure your Apache Tomcat application server. To deploy the WebGIS application:

- Copy the file "webgis.war" from the installer/apps folder to the **<tomcat_root_path>/tomcat_instance/webgis/webapps/** directory.

Note: Assume that **tomcat_root_path** is in /home/Ubuntu

```
$ cd installer/installer
$ cp apps/webgis.war ~/tomcat-instance/webgis/webapps
```

The commands below should be executed before starting Tomcat

```
$ cd ~/tomcat/bin
$ ./startup.sh
$ nano catalina.webgis.sh
```

You should edit catalina.webgis.sh inside ~/tomcat/bin folder and change the following home folder settings:

```
CATALINA_HOME=~/tomcat
CATALINA_BASE=~/tomcat-instance/webgis
```

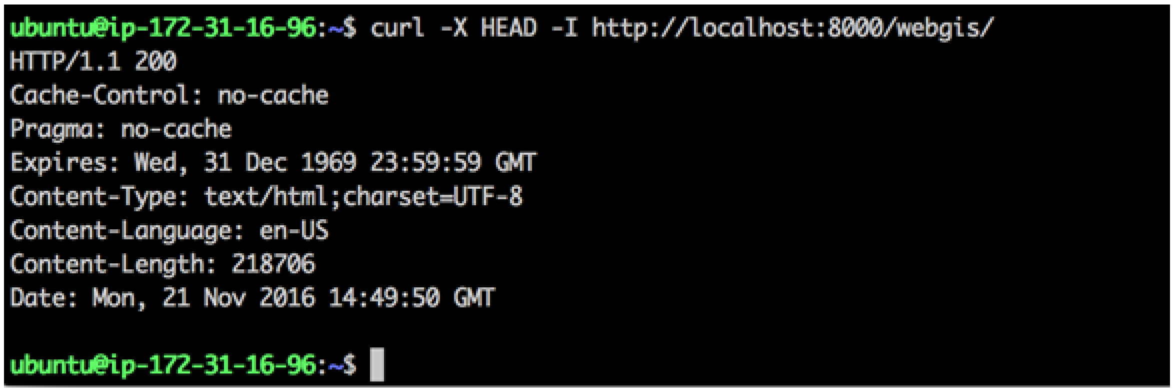
- Start Tomcat: This will extract the WAR file.

```
$ catalina.webgis.sh start
```

- After the application deploys, shutdown the server and delete the webgis.war file from <tomcat_path>\tomcat_instance\webgis\webapps\ directory.

```
$ ./catalina.webgis.sh stop
```

The output should be like the following figure shows:



```
ubuntu@ip-172-31-16-96:~$ curl -X HEAD -I http://localhost:8000/webgis/
HTTP/1.1 200
Cache-Control: no-cache
Pragma: no-cache
Expires: Wed, 31 Dec 1969 23:59:59 GMT
Content-Type: text/html;charset=UTF-8
Content-Language: en-US
Content-Length: 218706
Date: Mon, 21 Nov 2016 14:49:50 GMT

ubuntu@ip-172-31-16-96:~$
```

If you do not remove the WAR file after deployment, Tomcat redeploys the application each time the server starts.Use the following command lines to remove the webgis.war and shut down the server :

```
$ cd ~/tomcat-instance/webgis/webapps
$ rm -rf webgis.war
$ sudo shutdown -r now
```

To reconnect with the server, type the following command:

```
$ ssh -i WebGISkey.pem ubuntu@54.171.74.139
```

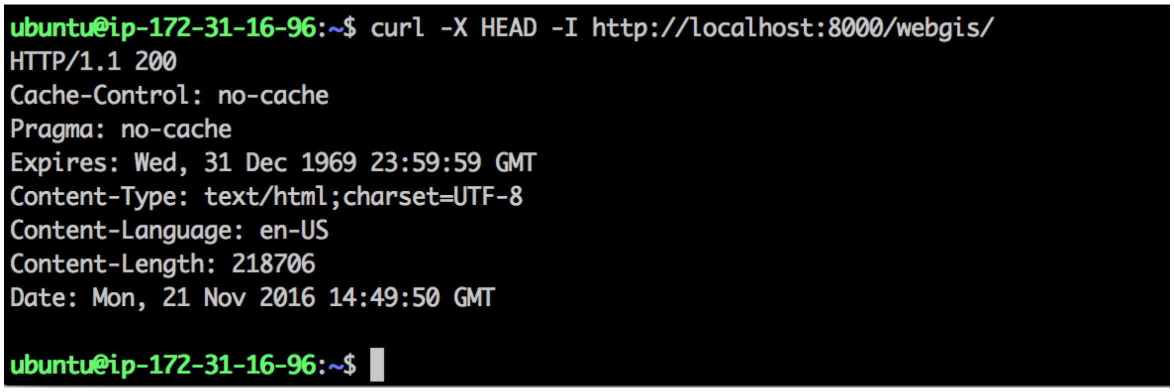
After reconnecting to the server, the Catalina should be restarted using the command:

```
$ ./catalina.webgis.sh start
```

Once Tomcat is up and running, you may run a health check to the WebGIS Page.

```
$  curl -X HEAD -I http://localhost:8000/webgis/
```

You should get a response like this.


```
ubuntu@ip-172-31-16-96:~$ curl -X HEAD -I http://localhost:8000/webgis/
HTTP/1.1 200
Cache-Control: no-cache
Pragma: no-cache
Expires: Wed, 31 Dec 1969 23:59:59 GMT
Content-Type: text/html;charset=UTF-8
Content-Language: en-US
Content-Length: 218706
Date: Mon, 21 Nov 2016 14:49:50 GMT

ubuntu@ip-172-31-16-96:~$
```

10.3.14. Step 14: Setting the Web-GIS Application

This section describes how to load the webgis database and configure the API URL's of the front-end application. To load the webgis database into PostgreSQL execute this command.

```
$ psql -U postgres -d gisdb -h localhost -f ~/installer/installer/sql/webgis.sql
```

To change the API URL's of the front-end application, you need to edit the file "app.map.js".

```
$ catalina.webgis.sh stop
$ vi /home/ubuntu/tomcat_instance/webgis/webapps/webgis/frontend/js/app.map.js
```

You need to change the value of the variable serverHost to the Public IP or DNS name of your server. If you have app.map.js, you could change the following:

```
var serverHost = '54.171.74.139';
var apiUrl = 'http://'+ serverHost +'/webgis/api';
var sldUrl = 'http://'+ serverHost +'/webgis/frontend/sld';
var geoUrl = 'http://'+ serverHost +'/geoserver';
var owsUrl = geoUrl + '/ows?';
var wmsUrl = geoUrl + '/wms?';
```

You save your changes and restart WebGIS application.

```
$ catalina.webgis.sh start
```

Install GeoServer :

To insall the Geoserver, you need to download GeoServer Binary. You can find the latest version of GeoServer at the GeoServer Download page. At the time of writing, the latest version is 2.9.2. Under the Maintenance Distributions section, you click the version 2.9.2, then you copy the link of the "Web Archived".

First, you need to change to your home directory:

```
$ cd ~
```

Then you use **wget** and you paste in link to download GeoServer 2.9.2.

```
$ wget http://sourceforge.net/projects/geoserver/files/GeoServer/2.9.2/geoserver-2.9.2-war.zip
```

We're going to install GeoServer to the ~/tomcat-instance/geoserver/webapps directory. If **unzip** is not installed, you may need to install using this command.

```
$ sudo apt-get install unzip
```

10.3.15. Step 15: Deploying GeoServer Application

To deploy the GeoServer application:

- Unzip the file geoserver-2.9.2-war.zip
- Then, copy geoserver.war to ~/tomcat-instance/geoserver/webapps directory.

```
$ unzip geoserver-2.9.2-war.zip
$ cp geoserver.war ~/tomcat-instance/geoserver/webapps
```

10.3.16. Step 16: Setting the GeoServer Application

This step is about how to setup the GeoServer environment variables and Geoserver Data. You have to extract the geoserver_data.tgz into /home/ubuntu folder. This folder contains the geoserver settings and workspace for the layers.

```
$ cd ~/installer/installer/geoserver
$ tar Cxzvf geoserver_data.tgz /home/ubuntu/
```

You have to modify the following files and point the IP address of your server:

Example:

- ~/geoserver_data/workspaces/webgis/namespace.xml

<uri>http://54.171.74.139/geoserver</uri>

- ~/geoserver_data/workspaces/webgis/webgis/datastore.xml

<entrykey="namespace">http:// 54.171.74.139/geoserver</entry>

The GeoServer **Data Directory** location and Tomcat environment options are set in ~/tomcat/bin/catalina.geoserver.sh.

The GeoServer **Port** is configured in ~/tomcat-instance/geoserver/conf/server.xml. You need to edit this file with the following setting. You set the connector to port 9000.

```
<Connector port="9000" protocol="HTTP/1.1"
               connectionTimeout="20000"
               redirectPort="8443" />
```

You set the Server port to 8006.

```
<Server port="8006" shutdown="SHUTDOWN">
```

You set the AJP 1.3 Connector to port 8010.

```
<!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector port="8010" protocol="AJP/1.3" redirectPort="8443" />
```

You Save your changes and exit.

You may need to edit .bash_profile to set POSTGIS settings.

```
$ vi ~/.bash profile
```

Then, you add the following lines to the end of the file.

```
CATALINA_HOME=$HOME/tomcat
PATH=$PATH:$HOME/.local/bin:$HOME/bin:$CATALINA_HOME/bin

                          POSTGIS_GDAL_ENABLED_DRIVERS=ENABLE_ALL
POSTGIS_ENABLE_OUTDB_RASTERS=1

export PATH POSTGIS_GDAL_ENABLED_DRIVERS POSTGIS_ENABLE_OUTDB_RASTERS
```

You save the changes, re-load the profile, and start GeoServer application.

```
$  source .bash_profile
$  catalina.geoserver.sh start
```

Once GeoServer is up and running, you may run a health check to the Web Admin Page.

You should edit catalina.geoserver.sh inside ~/tomcat/bin folder and change the following home folder settings:

```
$ cd ~/tomcat/bin
$ ./startup.sh
$ nano catalina.geoserver.sh
```

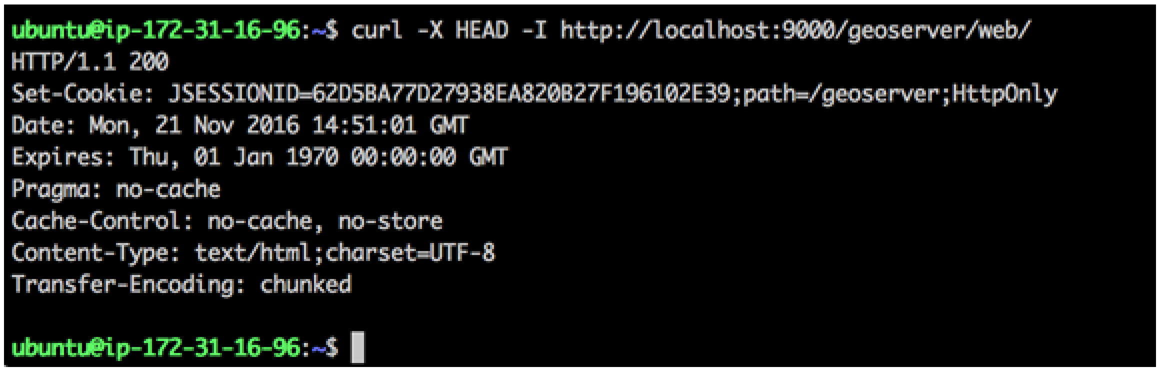
Start Tomcat. This will extract the WAR file.

```
$  source .bash_profile
$  catalina.geoserver.sh start
```

```
$  curl http://localhost:9000/geoserver/web/
```

```
CATALINA_HOME=~/tomcat
CATALINA_BASE=~/tomcat-instance/geoserver
```

The right response should be as the following figure shows:



```
ubuntu@ip-172-31-16-96:~$ curl -X HEAD -I http://localhost:9000/geoserver/web/
HTTP/1.1 200
Set-Cookie: JSESSIONID=62D5BA77D27938EA820B27F196102E39;path=/geoserver;HttpOnly
Date: Mon, 21 Nov 2016 14:51:01 GMT
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Pragma: no-cache
Cache-Control: no-cache, no-store
Content-Type: text/html;charset=UTF-8
Transfer-Encoding: chunked

ubuntu@ip-172-31-16-96:~$
```

10.3.17. Step 17: configuring Apache HTTP Server

This step is about how to install Apache HTTP server and configure reverse proxy.

```
$  sudo apt-get update
$  sudo apt-get install apache2
```

Activating the Modules: Before configuring Apache,you need to enable the necessary modules that you will be using in the application. You run the following command to get a list of available Apache modules:

```
$ a2enmod

# You will be presented with an output similar to:

# Which module(s) do you want to enable (wildcards ok)?
```

Once you are prompted with the choice of modules you desire, you can pass the below line:

```
proxy proxy_ajp proxy_http rewrite deflate headers proxy_balancer proxy_connect proxy_html
```

P.S: Some modules are likely to be enabled by default. Trying to enable them twice will just ensure that they are active.

10.3.18. Step 18: Modifying the default configuration

In this step, you will see how to modify the default configuration file 000-default.conf inside /etc/apache2/sites-enabled to set up "proxying" functionality. You run the following command to edit the default Apache virtual host using the vi text editor:

```
$ vi /etc/apache2/sites-enabled/000-default.conf
```

Here, you will define a proxy virtual host using mod_virtualhost and mod_proxy together. You copy-and-paste the below block of configuration, amending it to suit your needs:

```
<VirtualHost *:80>
        ...
ProxyPass /webgis/ http://127.0.0.1:8000/webgis/
ProxyPassReverse /webgis/ http://127.0.0.1:8000/webgis/

ProxyPass /geoserver/ http://127.0.0.1:9000/geoserver/
ProxyPassReverse /geoserver/ http://127.0.0.1:9000/geoserver/

</VirtualHost>
```

You save the changes.

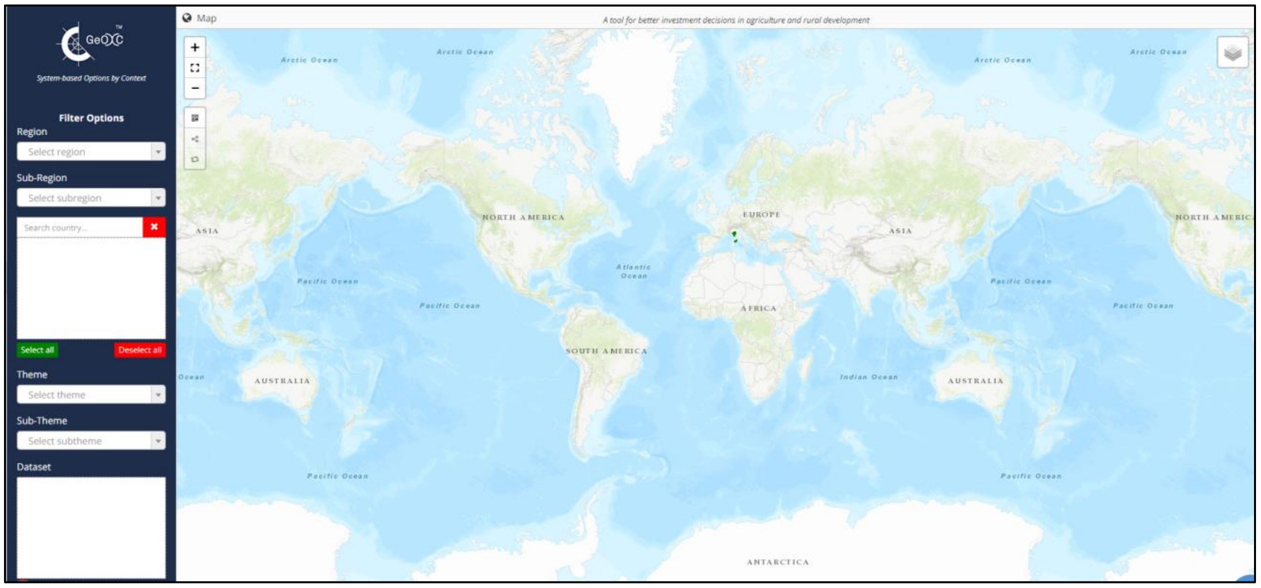
Restarting Apache: you execute the following command to restart apache.

```
$ service apache2 restart
```

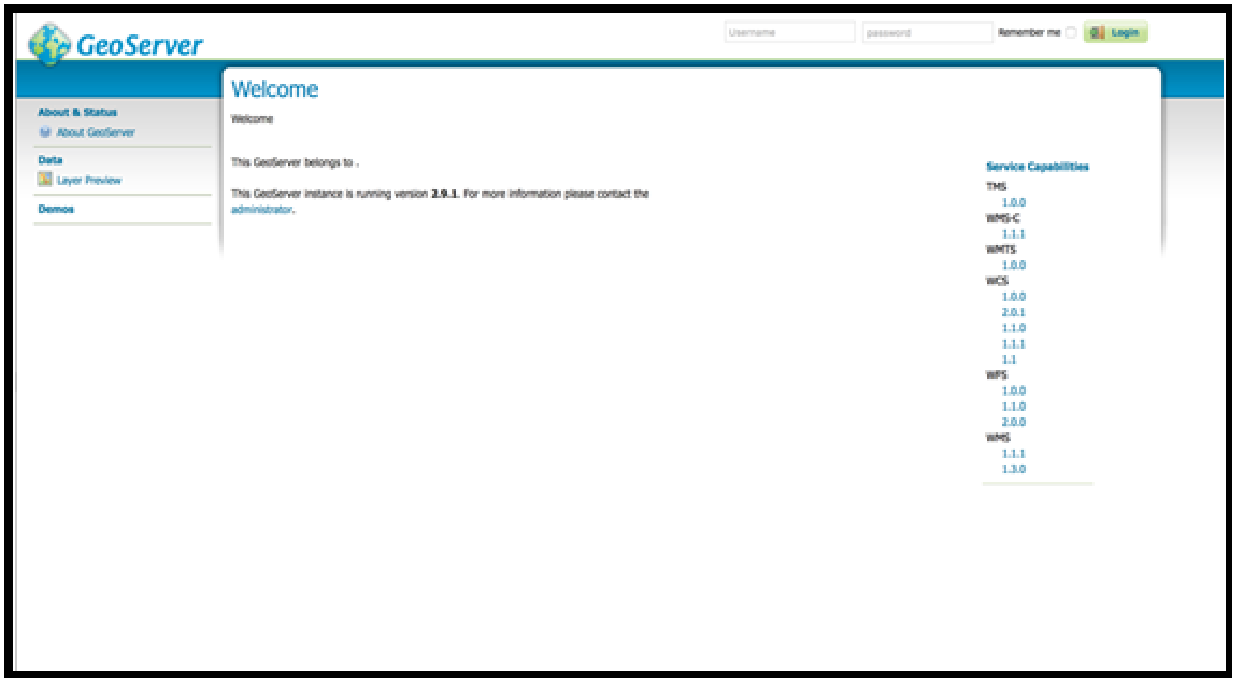
Congartulations! You can now access the WebGIS and GeoServer applications outside the network via browser with the following URL.

WebGIS Application: http://192.168.1.41:8000/webgis

https://mel.cgiar.org/slm/visualization



GeoServer Application: http://54.171.74.139/geoserver



Welcome

Welcome

This GeoServer belongs to .

This GeoServer instance is running version **2.9.1**. For more information please contact the administrator.

Service Capabilities

TMS
1.0.0
WMS-C
1.1.1
WMTS
1.0.0
WCS
1.0.0
2.0.1
1.1.0
1.1.1
1.1
WFS
1.0.0
1.1.0
2.0.0
WMS
1.1.1
1.3.0

10.3.19. Step 19: Install native JAI and ImageIO extensions

In this step, we are going to setup Java Advanced Imaging API (JAI) for improve the performance of all raster processing.

Preparation: To avoid conflict in geoserver lib

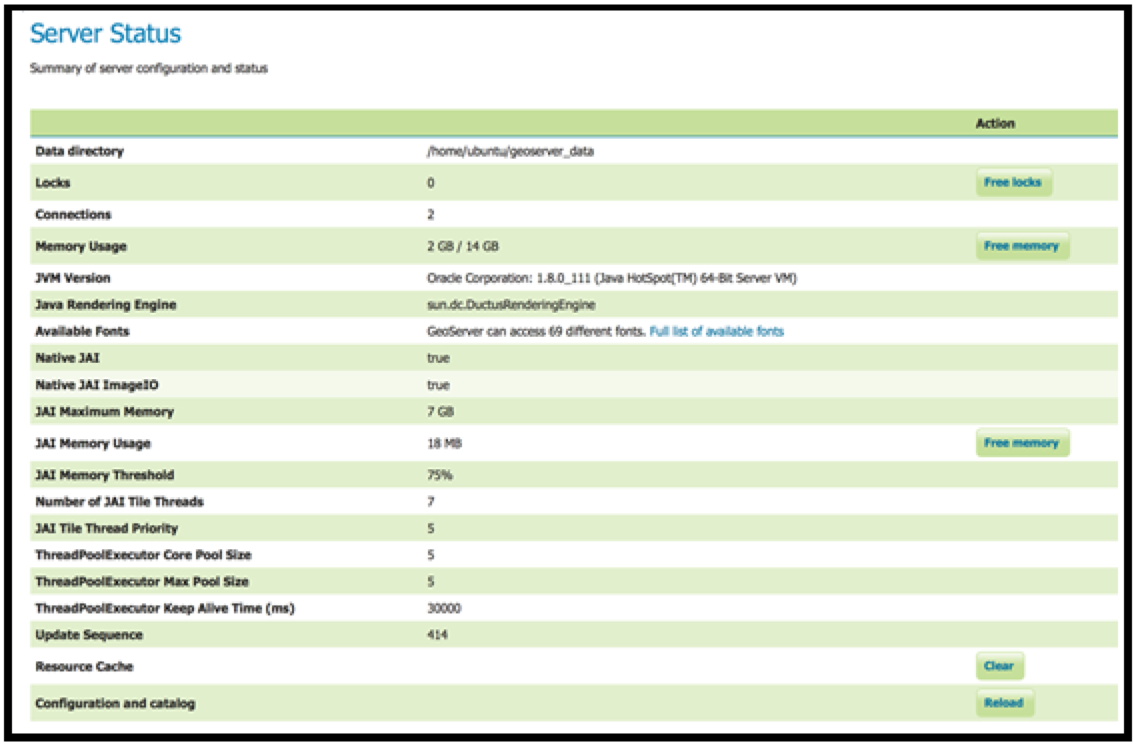
1. Install JAI 1.1.3, copy and paste the command below into the terminal window.

```
$ cd /tmp
$ sudo wget http://download.java.net/media/jai/builds/release/1_1_3/jai-1_1_3-lib-
linux-amd64.tar.gz

$ sudo gunzip -c jai-1_1_3-lib-linux-amd64.tar.gz | tar xf -
$ sudo cp /tmp/jai-1_1_3/lib/*.jar $JAVA_HOME/jre/lib/ext/
$ sudo cp /tmp/jai-1_1_3/lib/*.so $JAVA_HOME/jre/lib/amd64/

$ rm /tmp/jai-1_1_3-lib-linux-amd64.tar.gz
$ rm -r /tmp/jai-1_1_3
```

2. Install JAI Image I/O

```
$ cd /tmp
$ sudo wget http://download.java.net/media/jai-imageio/builds/release/1.1/jai_imageio-
1_1-lib-linux-amd64.tar.gz

$ sudo gunzip -c jai_imageio-1_1-lib-linux-amd64.tar.gz | tar xf -
$ sudo cp /tmp/jai_imageio-1_1/lib/*.jar $JAVA_HOME/jre/lib/ext/
$ sudo cp /tmp/jai_imageio-1_1/lib/*.so $JAVA_HOME/jre/lib/amd64/

$ rm /tmp/jai_imageio-1_1-lib-linux-amd64.tar.gz
$ rm -r /tmp/jai_imageio-1_1
```

3. To verify if JAI is successfully installed, login to your GeoServer application and click "Server Status".Native JAI and Native JAI ImageIO should be set to "true".



Server Status

Summary of server configuration and status

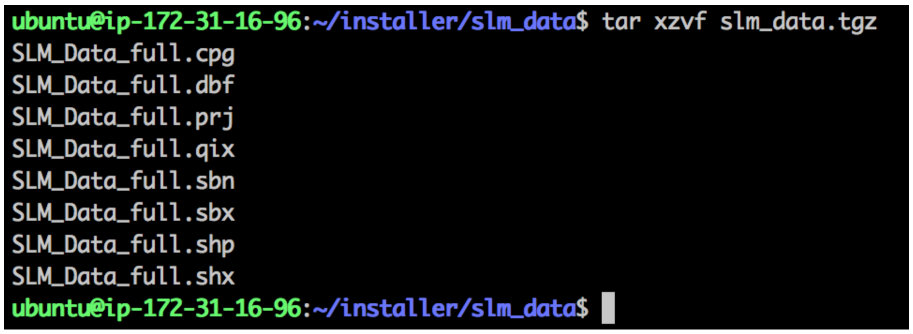
		Action
Data directory	/home/ubuntu/geoserver_data	
Locks	0	Free locks
Connections	2	
Memory Usage	2 GB / 14 GB	Free memory
JVM Version	Oracle Corporation: 1.8.0_111 (Java HotSpot(TM) 64-Bit Server VM)	
Java Rendering Engine	sun.dc.DuctusRenderingEngine	
Available Fonts	GeoServer can access 69 different fonts. Full list of available fonts	
Native JAI	true	
Native JAI ImageIO	true	
JAI Maximum Memory	7 GB	
JAI Memory Usage	18 MB	Free memory
JAI Memory Threshold	75%	
Number of JAI Tile Threads	7	
JAI Tile Thread Priority	5	
ThreadPoolExecutor Core Pool Size	5	
ThreadPoolExecutor Max Pool Size	5	
ThreadPoolExecutor Keep Alive Time (ms)	30000	
Update Sequence	414	
Resource Cache		Clear
Configuration and catalog		Reload

11. Loading SLM data into PostGIS

PostGIS provides a "shp2pgsql" tool to convert ESRI shape files into SQL suitable for insertion into PostGIS/PostgreSQL database.

Preparation

- Connect to your AWS Linux instance or localhost.
- Extract slm_data.tgz located in the installer/slm_data directory.

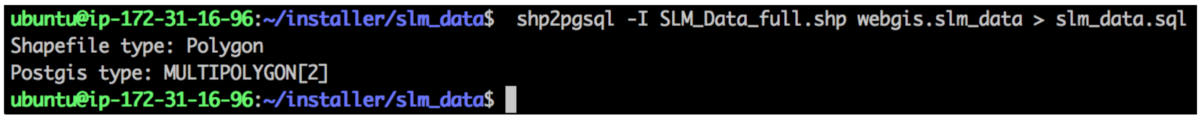


```
ubuntu@ip-172-31-16-96:~/installer/slm_data$ tar xzvf slm_data.tgz
SLM_Data_full.cpg
SLM_Data_full.dbf
SLM_Data_full.prj
SLM_Data_full.qix
SLM_Data_full.sbn
SLM_Data_full.sbx
SLM_Data_full.shp
SLM_Data_full.shx
ubuntu@ip-172-31-16-96:~/installer/slm_data$
```

11.1 Loading Data

1. Convert the shape file to SQL script.

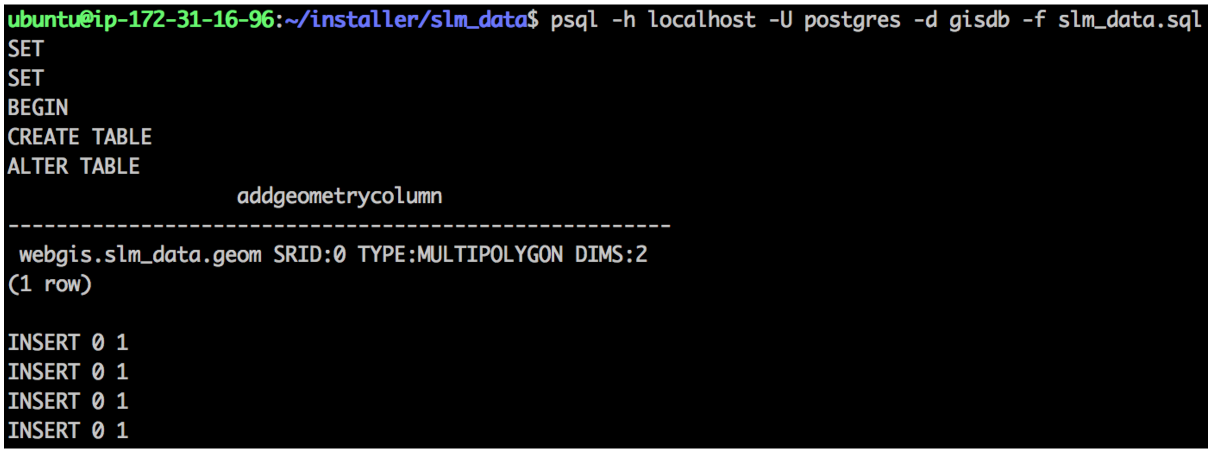
```
$ shp2pgsql -I -s4326 -d slm_filled.shp webgis.slm_data > slm_data.sql
```



```
ubuntu@ip-172-31-16-96:~/installer/slm_data$  shp2pgsql -I SLM_Data_full.shp webgis.slm_data > slm_data.sql
Shapefile type: Polygon
Postgis type: MULTIPOLYGON[2]
ubuntu@ip-172-31-16-96:~/installer/slm_data$
```

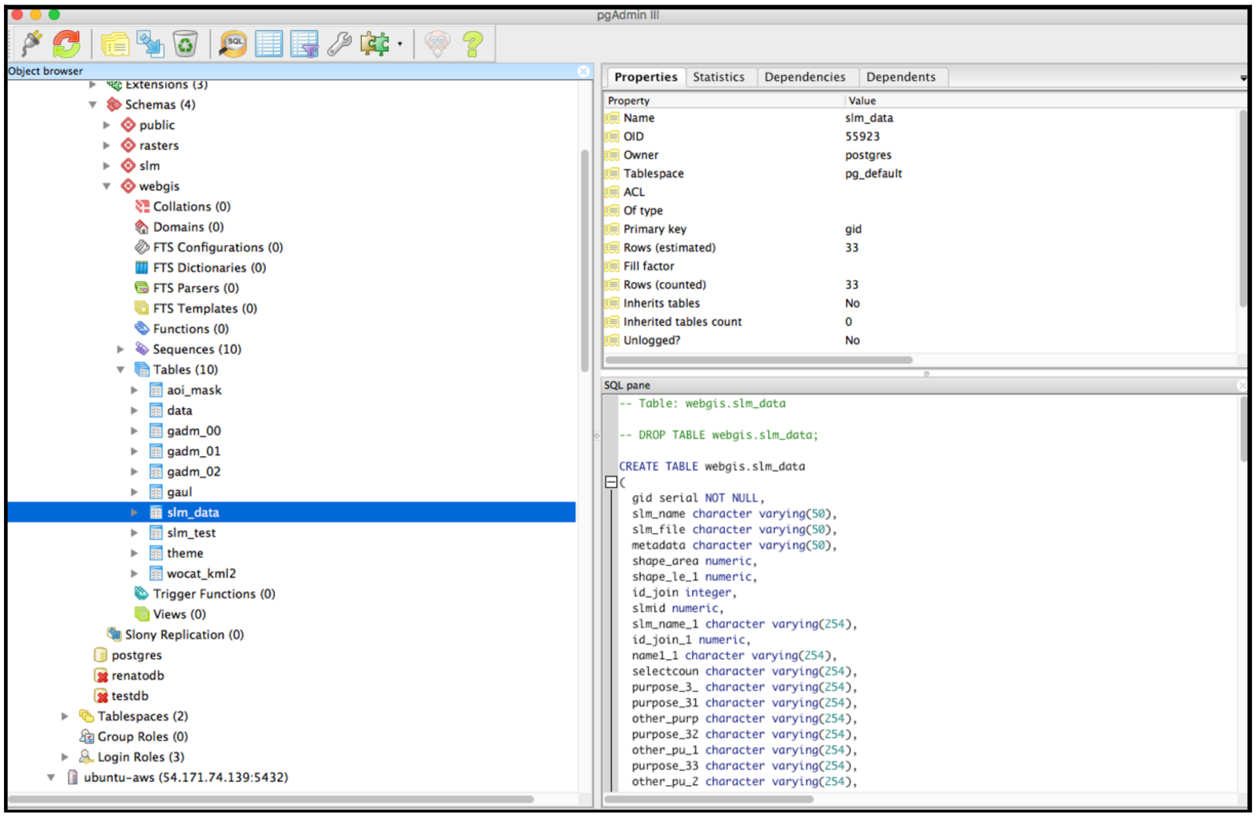
2. Load the SQL script to PostGIS database.

```
psql -h localhost -U postgres -d gisdb -f slm_data.sql
```



```
ubuntu@ip-172-31-16-96:~/installer/slm_data$ psql -h localhost -U postgres -d gisdb -f slm_data.sql
SET
SET
BEGIN
CREATE TABLE
ALTER TABLE
                   addgeometrycolumn
-------------------------------------------------------
 webgis.slm_data.geom SRID:0 TYPE:MULTIPOLYGON DIMS:2
(1 row)

INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
```

3. Using PgAdmin tool you can verify the table “slm_data” is created in webgis schema.



Property	Value
Name	slm_data
OID	55923
Owner	postgres
Tablespace	pg_default
ACL	
Of type	
Primary key	gid
Rows (estimated)	33
Fill factor	
Rows (counted)	33
Inherits tables	No
Inherited tables count	0
Unlogged?	No

```
-- Table: webgis.slm_data

-- DROP TABLE webgis.slm_data;

CREATE TABLE webgis.slm_data
(
  gid serial NOT NULL,
  slm_name character varying(50),
  slm_file character varying(50),
  metadata character varying(50),
  shape_area numeric,
  shape_le_1 numeric,
  id_join integer,
  slmid numeric,
  slm_name_1 character varying(254),
  id_join_1 numeric,
  name1_1 character varying(254),
  selectcoun character varying(254),
  purpose_3_ character varying(254),
  purpose_31 character varying(254),
  other_purp character varying(254),
  purpose_32 character varying(254),
  other_pu_1 character varying(254),
  purpose_33 character varying(254),
  other_pu_2 character varying(254),
```

4. Finish. Next publish SLM to geoserver.

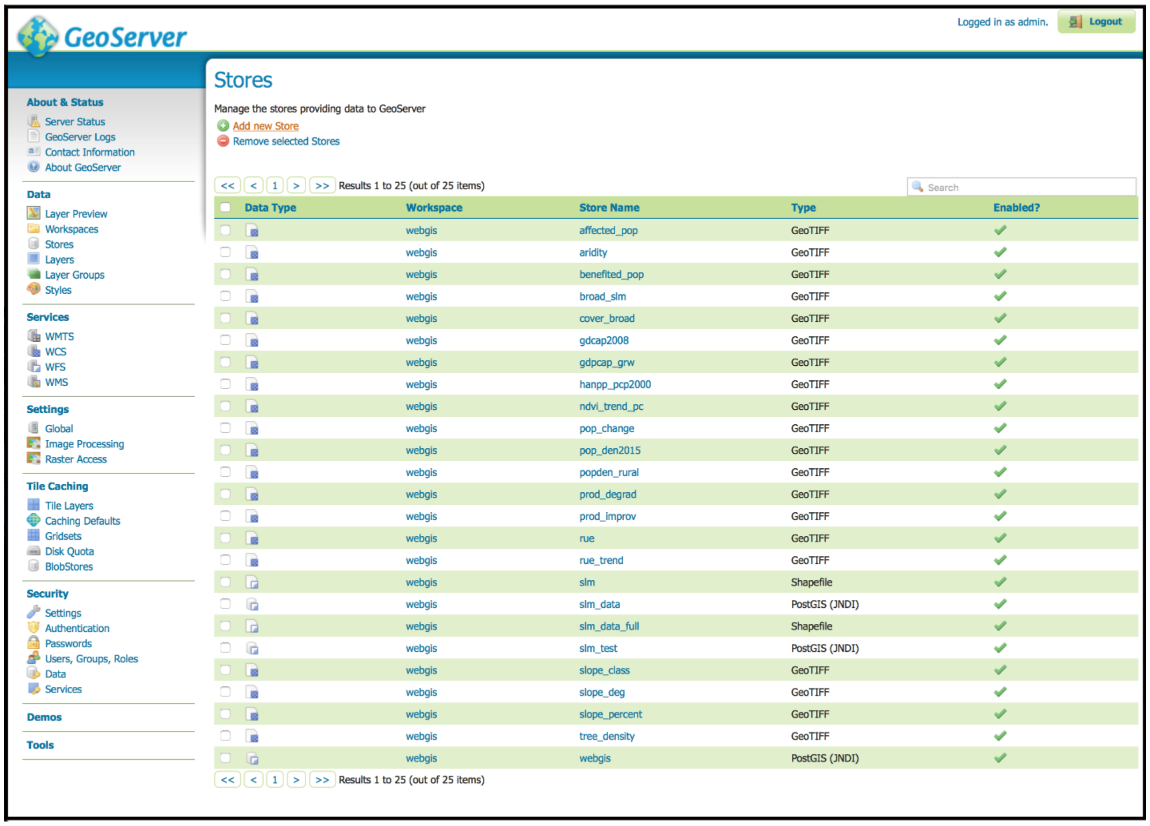
11.2 Publish SLM data to GeoServer

In this section, we are going to publish slm data into GeoServer.

1. In your web browser, navigate to http://54.171.74.139/geoserver/
2. Login to GeoServer. For this installation, the credentials are:

Username: admin
Password: adm-aws001

3. Navigate to Stores ->Add new Store



Stores

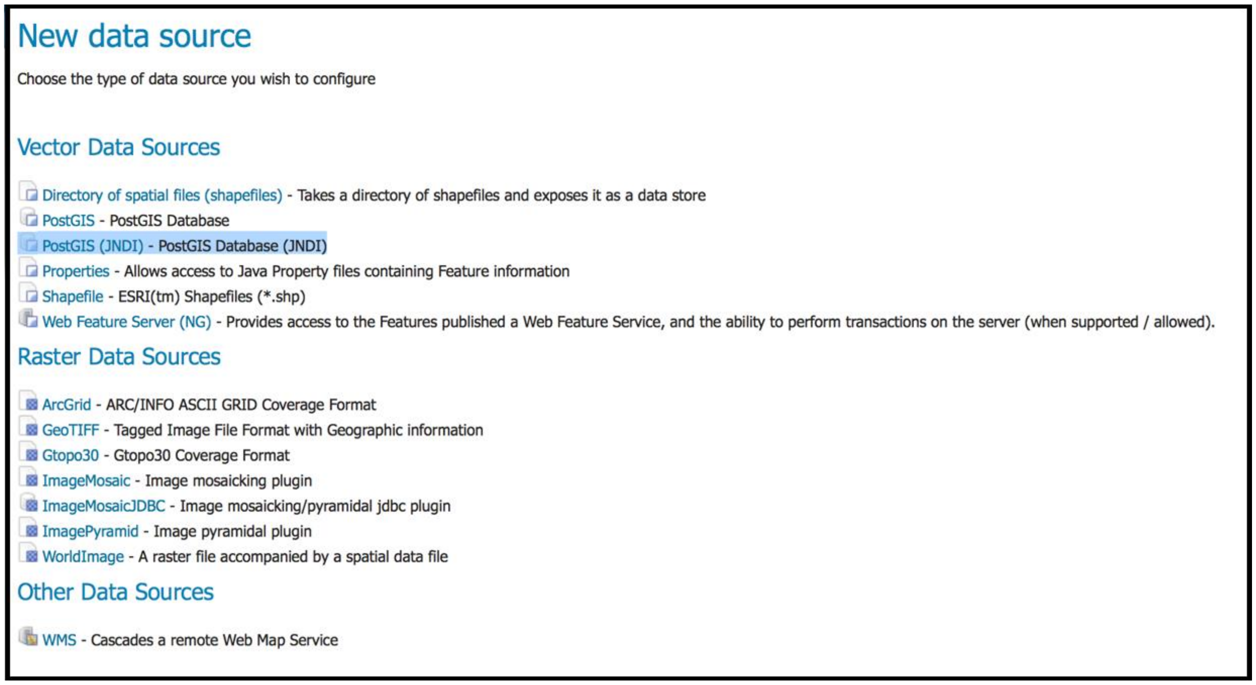
Manage the stores providing data to GeoServer

Results 1 to 25 (out of 25 items)

Data Type	Workspace	Store Name	Type	Enabled?
	webgis	affected_pop	GeoTIFF	✓
	webgis	aridity	GeoTIFF	✓
	webgis	benefited_pop	GeoTIFF	✓
	webgis	broad_slm	GeoTIFF	✓
	webgis	cover_broad	GeoTIFF	✓
	webgis	gdcap2008	GeoTIFF	✓
	webgis	gdpcap_grw	GeoTIFF	✓
	webgis	hanpp_pcp2000	GeoTIFF	✓
	webgis	ndvi_trend_pc	GeoTIFF	✓
	webgis	pop_change	GeoTIFF	✓
	webgis	pop_den2015	GeoTIFF	✓
	webgis	popden_rural	GeoTIFF	✓
	webgis	prod_degrad	GeoTIFF	✓
	webgis	prod_improv	GeoTIFF	✓
	webgis	rue	GeoTIFF	✓
	webgis	rue_trend	GeoTIFF	✓
	webgis	slm	Shapefile	✓
	webgis	slm_data	PostGIS (JNDI)	✓
	webgis	slm_data_full	Shapefile	✓
	webgis	slm_test	PostGIS (JNDI)	✓
	webgis	slope_class	GeoTIFF	✓
	webgis	slope_deg	GeoTIFF	✓
	webgis	slope_percent	GeoTIFF	✓
	webgis	tree_density	GeoTIFF	✓
	webgis	webgis	PostGIS (JNDI)	✓

Results 1 to 25 (out of 25 items)

4. Select "PostGIS (JNDI)"



New data source

Choose the type of data source you wish to configure

Vector Data Sources

- Directory of spatial files (shapefiles) - Takes a directory of shapefiles and exposes it as a data store
- PostGIS - PostGIS Database
- PostGIS (JNDI) - PostGIS Database (JNDI)
- Properties - Allows access to Java Property files containing Feature information
- Shapefile - ESRI(tm) Shapefiles (*.shp)
- Web Feature Server (NG) - Provides access to the Features published a Web Feature Service, and the ability to perform transactions on the server (when supported / allowed).

Raster Data Sources

- ArcGrid - ARC/INFO ASCII GRID Coverage Format
- GeoTIFF - Tagged Image File Format with Geographic information
- Gtopo30 - Gtopo30 Coverage Format
- ImageMosaic - Image mosaicking plugin
- ImageMosaicJDBC - Image mosaicking/pyramidal jdbc plugin
- ImagePyramid - Image pyramidal plugin
- WorldImage - A raster file accompanied by a spatial data file

Other Data Sources

- WMS - Cascades a remote Web Map Service

5. Fill the following fields as shown in the following screenshot.

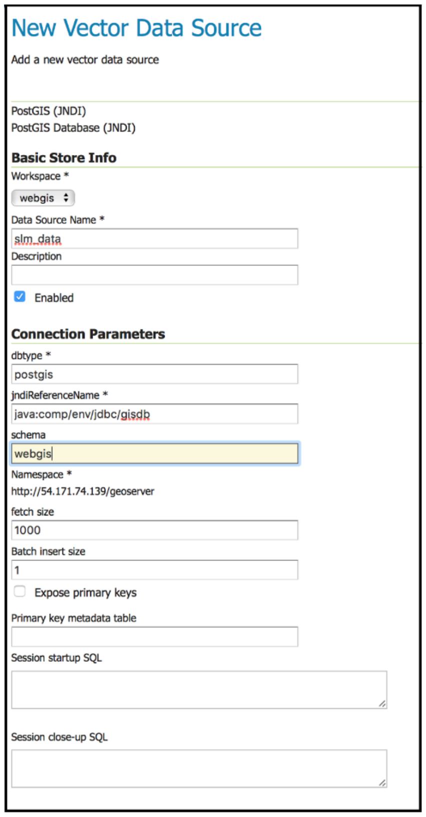
- **Basic Store Info**

Data Source Name : slm_data

- **Connection Parameters**

dbtype: postgis

jndiReferenceName: java:comp/env/jdbc/gisdb



New Vector Data Source

Add a new vector data source

PostGIS (JNDI)
PostGIS Database (JNDI)

Basic Store Info

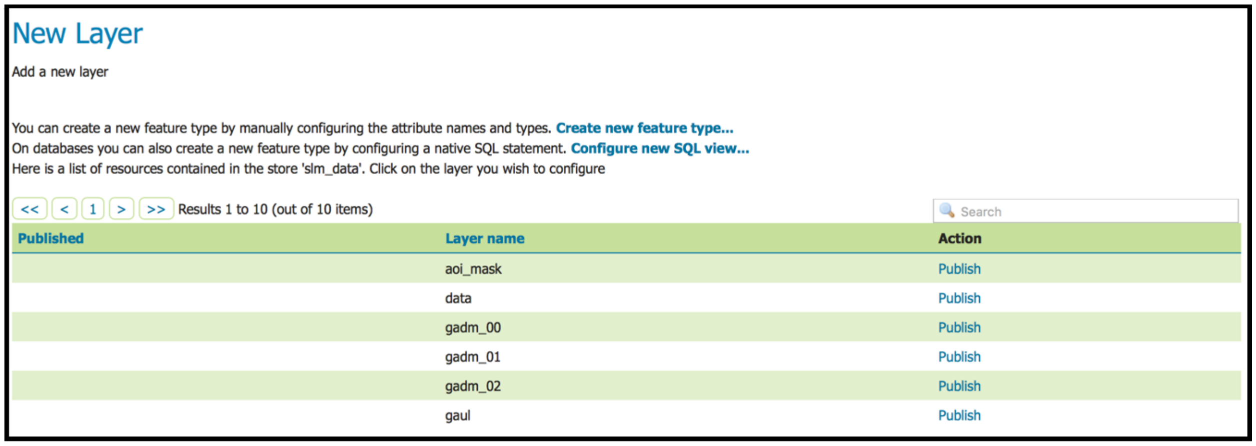
Workspace *	webgis
Data Source Name *	slm_data
Description	
Enabled	✓

Connection Parameters

dbtype *	postgis
jndiReferenceName *	java:comp/env/jdbc/gisdb
schema	webgis
Namespace *	http://54.171.74.139/geoserver
fetch size	1000
Batch insert size	1
Expose primary keys	
Primary key metadata table	
Session startup SQL	
Session close-up SQL	

6. Click Save

7. Navigate on slm_data, in the action column click Publish.



New Layer

Add a new layer

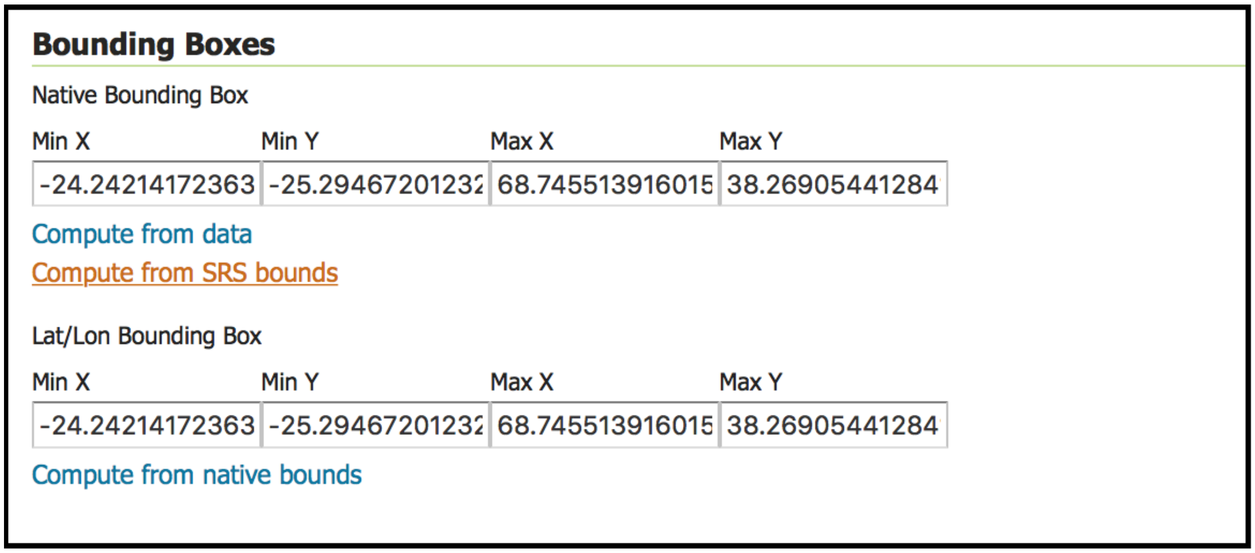
You can create a new feature type by manually configuring the attribute names and types. **Create new feature type...**
On databases you can also create a new feature type by configuring a native SQL statement. **Configure new SQL view...**
Here is a list of resources contained in the store 'slm_data'. Click on the layer you wish to configure

<< < 1 > >> Results 1 to 10 (out of 10 items)

Published	Layer name	Action
	aoi_mask	Publish
	data	Publish
	gadm_00	Publish
	gadm_01	Publish
	gadm_02	Publish
	gaul	Publish

8. In the Data tab, navigate to Bounding Box section. Click on the following link to calculate the bounding box.

- Compute from data
- Compute from native bounds



Bounding Boxes

Native Bounding Box

Min X	Min Y	Max X	Max Y
-24.24214172363	-25.29467201232	68.745513916015	38.26905441284

Compute from data

Compute from SRS bounds

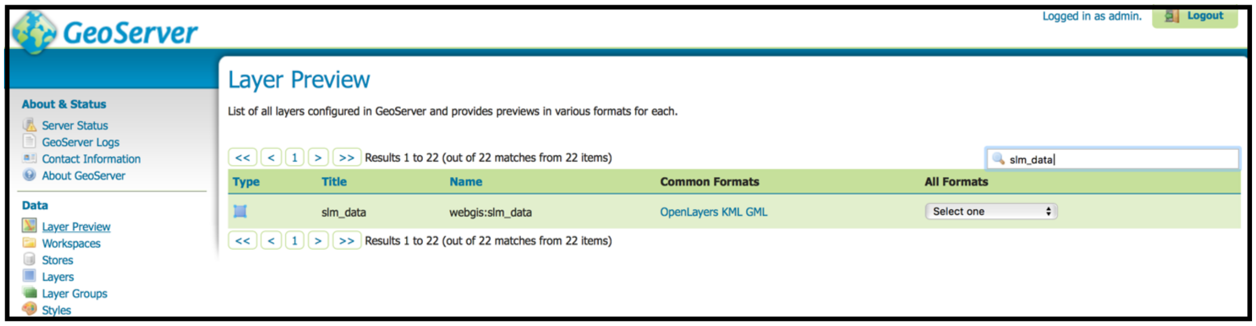
Lat/Lon Bounding Box

Min X	Min Y	Max X	Max Y
-24.24214172363	-25.29467201232	68.745513916015	38.26905441284

Compute from native bounds

9. Click Save

10. To preview the slm_data layer, navigation to Layer Preview in the menu (left side).



Layer Preview

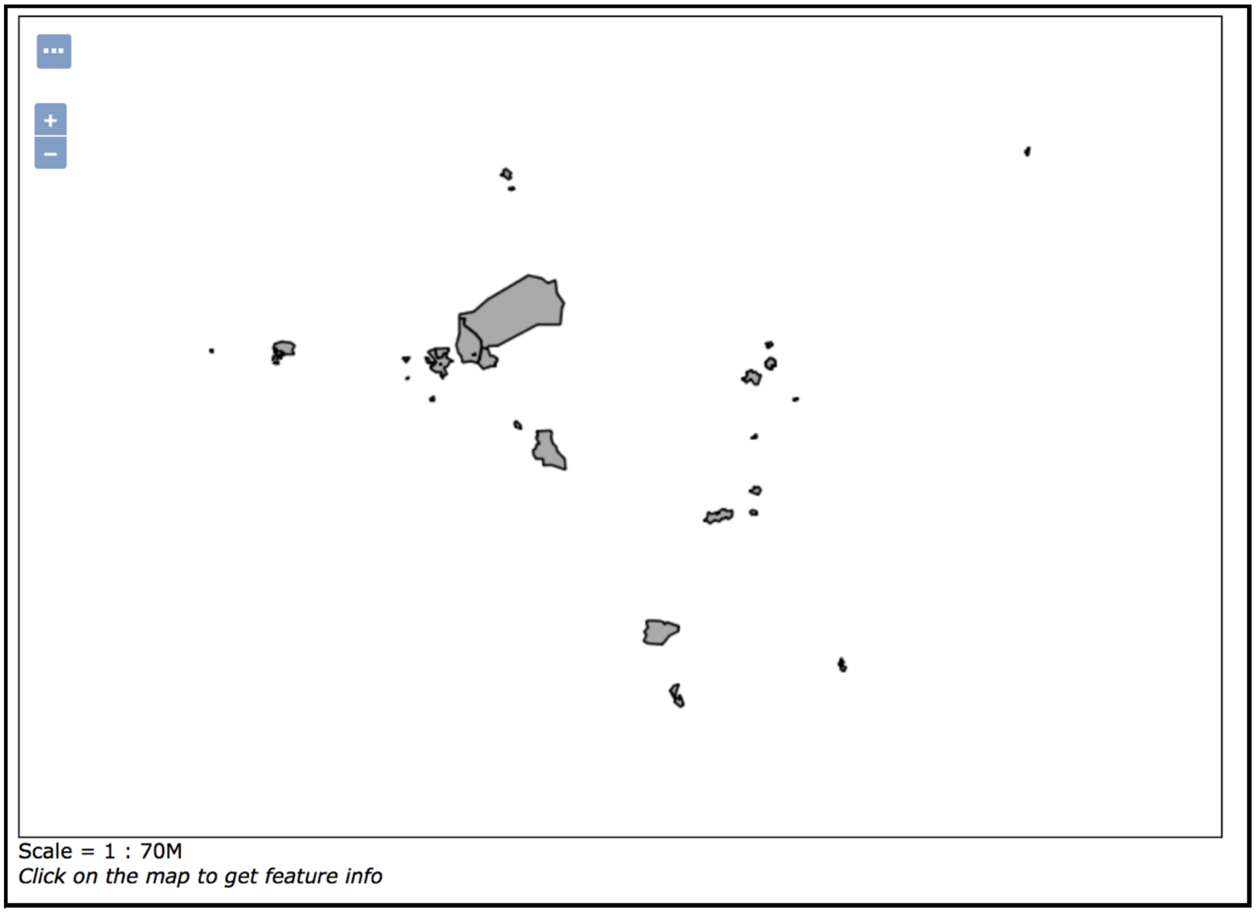
List of all layers configured in GeoServer and provides previews in various formats for each.

<< < 1 > >> Results 1 to 22 (out of 22 matches from 22 items)

Type	Title	Name	Common Formats	All Formats
▦	slm_data	webgis:slm_data	OpenLayers KML GML	Select one

<< < 1 > >> Results 1 to 22 (out of 22 matches from 22 items)

11. Navigate to slm_data and in the Column Format column click OpenLayers to preview.



12. Finish.

To update your SLM database, user need to (i) convert the shape file to SQL script, (ii) Load the SQL script to PostGIS database, and (iii) optionally, use PgAdmin tool to refresh the table "slm_data" in webgis schema.

12. Loading Raster Data into PostGIS

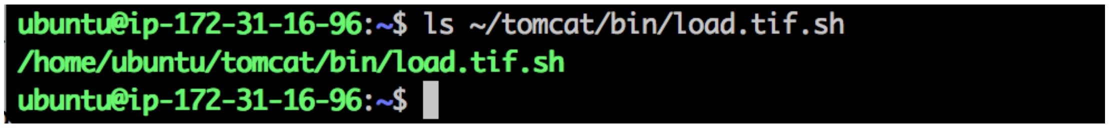
PostGIS provides a "raster2pgsql" tool for converting raster data sources into database tables. This tool is wrapped into "load.tiff.sh" which will automate the process of conversion, renaming, and loading to database. This section describes how to use this tool to load a single or multiple raster files.

12.1. Preparation

- Connect to your AWS Linux instance.
- Verify the script “load.tif.sh” is exists by executing this command.

```
$ ls ~/tomcat/bin/load.tif.sh
```

You should get a response like this.



```
ubuntu@ip-172-31-16-96:~$ ls ~/tomcat/bin/load.tif.sh
/home/ubuntu/tomcat/bin/load.tif.sh
ubuntu@ip-172-31-16-96:~$
```

- Run the following command to view the content of the loader script.

```
$ cat ~/tomcat/bin/load.tif.sh
```

/home/ubuntu/tomcat/bin/load.tif.sh

```
#!/bin/bash


start=`date +%s`
source_file=$@

if [ ! -d "processed" ]; then
    mkdir -p processed/{tif,sql}
fi

# extract sql
for f in $source_file; do target_file=`echo $f|tr '-' '_'`;name=`echo "$target_file" | cut -
d'.' -f1`;raster2pgsql -s 4326 -I -M -C $f -t 100x100 rasters.$name >
processed/sql/$name.sql;mv $f processed/tif;done

cd ~/raster-data/processed/sql

# load to postgis
for f in *.sql; do psql -U postgres -d gisdb -h localhost -f $f; tar cvzf $f.tgz $f;rm -f
$f;done

# compressed tif
cd ~/raster-data/processed/tif
for f in *.tif; do tar cvzf $f.tgz $f;rm -f $f;done

cd ~/raster-data
end=`date +%s`

runtime=$((end-start))

echo Begin: $start
echo End: $end
echo Elapsed: $runtime
```

- Select the raster files(s) you wish to load. You could simply copy them into a folder named “raster-data”.

12.2. Loading Data

1. Change directory to the raster folder.

```
$ cd raster-data
```

2. Execute the loader script by following this syntax.
   To load a single raster file:

   load.tif.sh <RASTER_FILE>

   for e.g. *load.tif.sh cover-broad.tif*

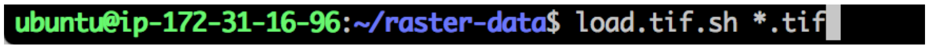
   To load multiple raster files:

   load.tif.sh <RASTER_FILE1> <RASTER_FILE2…>

   or using wildcard characters.

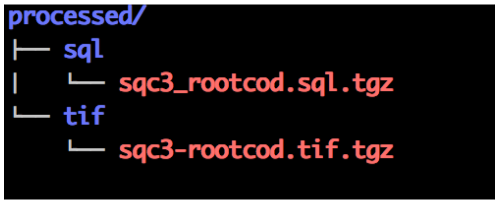
   load.tif.sh <.tif>*

   In this example, we load multiple rasters:



```
ubuntu@ip-172-31-16-96:~/raster-data$ load.tif.sh *.tif
```

This will process all the rasters by extracting the data, load to database and compress them when completed. You will be prompted to enter postgres password. Enter postgres password you assigned during installation. You will see same screenshot below when the process is completed. The raster files that has been processed are copied and compressed into processed folder. Below shows the directory structure.



```
processed/
├── sql
│   └── sqc3_rootcod.sql.tgz
└── tif
    └── sqc3-rootcod.tif.tgz
```

If loaded successfully, new table are created in the rasters schema.

12.3. Publish Raster to GeoServer

In this section, we are going to publish raster data into GeoServer.
Preparation:

- Create a tiff folder in the ~/geoserver_data workspace.

```
$ cd ~/geoserver_Data
$ mkdir tiff
```

- Next, copy all the tiff files you want to publish into this folder.

  In this example, "*tree-density.tif*" is copied in ~/geoserver_data/tiff folder.

- Rasters styles are stored in the **installer**->**styles** folder.

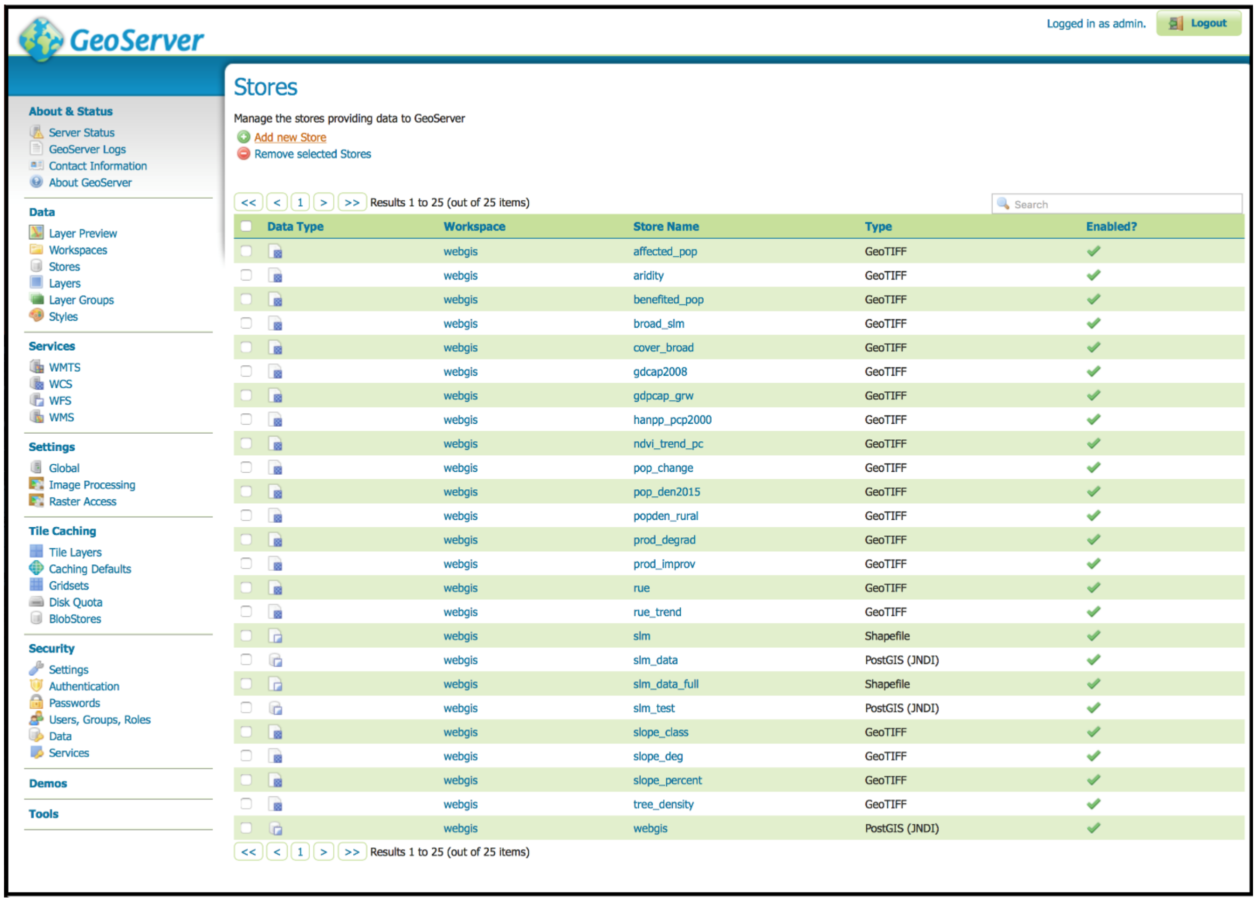
- Proceed on the next section to publish the raster.

  In your web browser, navigate to http://54.171.74.139/geoserver/

- Login to GeoServer. For this installation, the credentials are:

  Username: admin
  Password: adm-aws001

- Navigate to **Stores** ->**Add new Store**



Stores

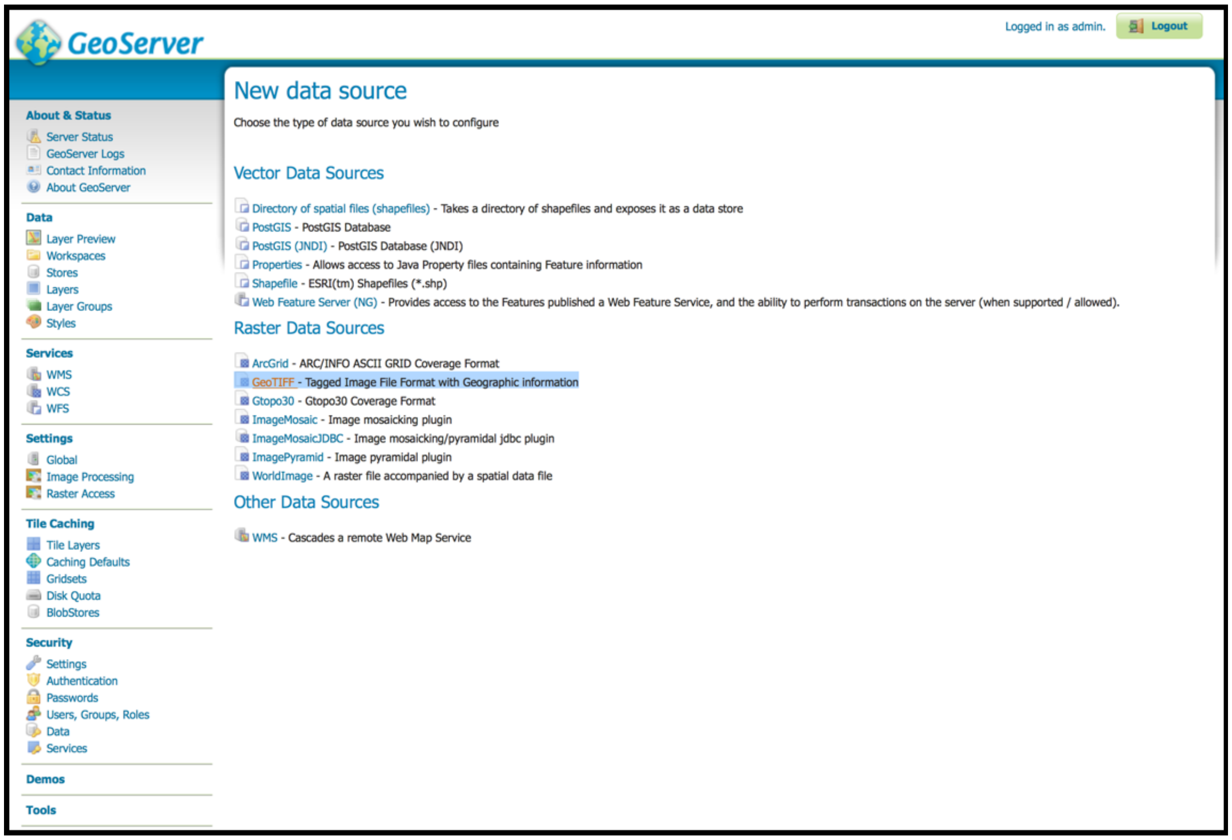
Manage the stores providing data to GeoServer

<< < 1 > >> Results 1 to 25 (out of 25 items)

Data Type	Workspace	Store Name	Type	Enabled?
	webgis	affected_pop	GeoTIFF	✓
	webgis	aridity	GeoTIFF	✓
	webgis	benefited_pop	GeoTIFF	✓
	webgis	broad_slm	GeoTIFF	✓
	webgis	cover_broad	GeoTIFF	✓
	webgis	gdcap2008	GeoTIFF	✓
	webgis	gdpcap_grw	GeoTIFF	✓
	webgis	hanpp_pcp2000	GeoTIFF	✓
	webgis	ndvi_trend_pc	GeoTIFF	✓
	webgis	pop_change	GeoTIFF	✓
	webgis	pop_den2015	GeoTIFF	✓
	webgis	popden_rural	GeoTIFF	✓
	webgis	prod_degrad	GeoTIFF	✓
	webgis	prod_improv	GeoTIFF	✓
	webgis	rue	GeoTIFF	✓
	webgis	rue_trend	GeoTIFF	✓
	webgis	slm	Shapefile	✓
	webgis	slm_data	PostGIS (JNDI)	✓
	webgis	slm_data_full	Shapefile	✓
	webgis	slm_test	PostGIS (JNDI)	✓
	webgis	slope_class	GeoTIFF	✓
	webgis	slope_deg	GeoTIFF	✓
	webgis	slope_percent	GeoTIFF	✓
	webgis	tree_density	GeoTIFF	✓
	webgis	webgis	PostGIS (JNDI)	✓

<< < 1 > >> Results 1 to 25 (out of 25 items)

- Select “GeoTIFF”



Logged in as admin. Logout

New data source

Choose the type of data source you wish to configure

Vector Data Sources

- Directory of spatial files (shapefiles) - Takes a directory of shapefiles and exposes it as a data store
- PostGIS - PostGIS Database
- PostGIS (JNDI) - PostGIS Database (JNDI)
- Properties - Allows access to Java Property files containing Feature information
- Shapefile - ESRI(tm) Shapefiles (*.shp)
- Web Feature Server (NG) - Provides access to the Features published a Web Feature Service, and the ability to perform transactions on the server (when supported / allowed).

Raster Data Sources

- ArcGrid - ARC/INFO ASCII GRID Coverage Format
- GeoTIFF - Tagged Image File Format with Geographic information
- Gtopo30 - Gtopo30 Coverage Format
- ImageMosaic - Image mosaicking plugin
- ImageMosaicJDBC - Image mosaicking/pyramidal jdbc plugin
- ImagePyramid - Image pyramidal plugin
- WorldImage - A raster file accompanied by a spatial data file

Other Data Sources

- WMS - Cascades a remote Web Map Service

- In the Add Raster DataSource, fill in basic store info and connection parameters.

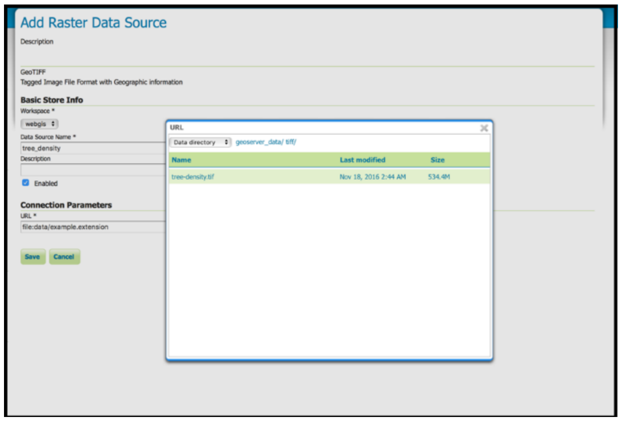
  Workspace: **webgis**

  Data Source Name: **tree_density**

  Description: **tree_desnity**

  NOTE: In the data source name, please use “_” (underscore) instead of “-” (hyphen) to match the table name in postgis.

- Connection Parameters, click Browse and navigate to “tiff” folder.



Add Raster Data Source

Description

GeoTIFF
Tagged Image File Format with Geographic information

Basic Store Info

Workspace *: webgis

Data Source Name *: tree_density

Description:

☑ Enabled

Connection Parameters

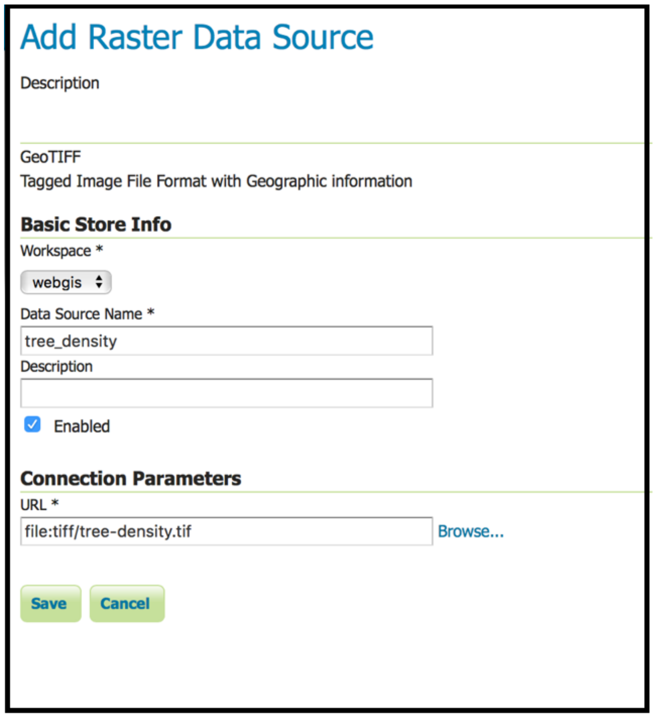
URL *: file:data/example.extension

URL

Data directory: geoserver_data/ tiff/

Name	Last modified	Size
tree-density.tif	Nov 18, 2016 2:44 AM	534.4M

- Select the “tree-density.tiff” as shown below screenshot.



Add Raster Data Source

Description

GeoTIFF
Tagged Image File Format with Geographic information

Basic Store Info

Workspace *: webgis

Data Source Name *: tree_density

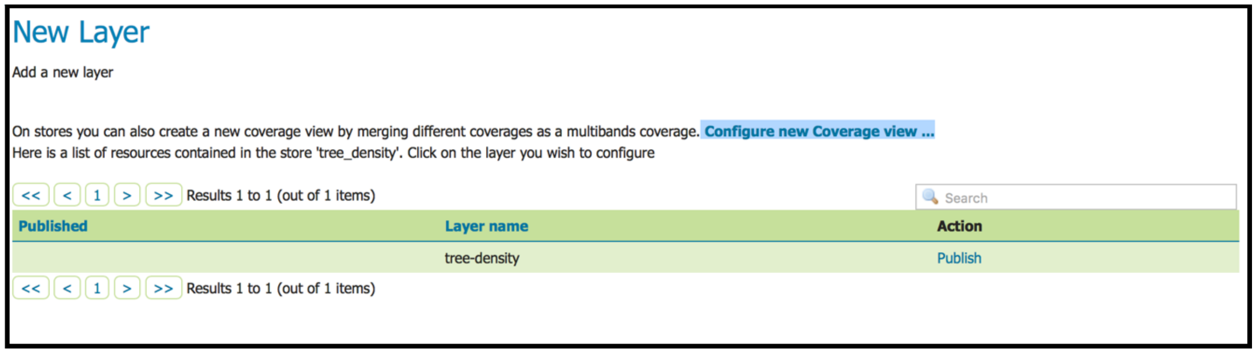
Description:

☑ Enabled

Connection Parameters

URL *: file:tiff/tree-density.tif

- Click Save. If no errors are returned, you will be redirected to New Layer page in order to configure the tree_density layer.
- Now, that the store is loaded.


New Layer

Add a new layer

On stores you can also create a new coverage view by merging different coverages as a multibands coverage. Configure new Coverage view ...

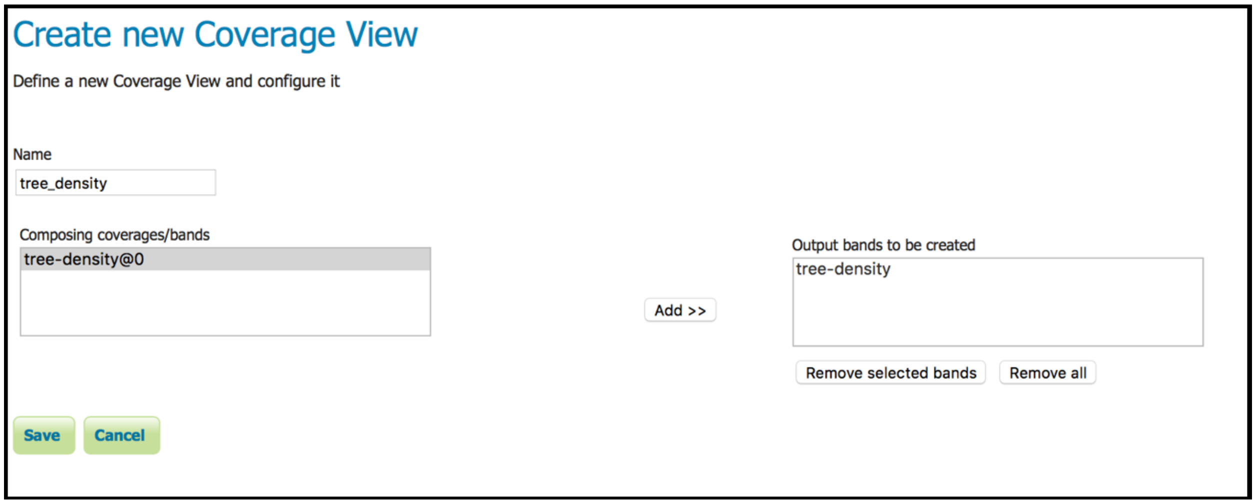
Here is a list of resources contained in the store 'tree_density'. Click on the layer you wish to configure

<< < 1 > >> Results 1 to 1 (out of 1 items)

Published	Layer name	Action
	tree-density	Publish

<< < 1 > >> Results 1 to 1 (out of 1 items)

- Click on Configure new Coverage View….
- Change the name to “tree_density”. Click Add button to add Composing coverages/bands to Output bands to be created.


Create new Coverage View

Define a new Coverage View and configure it

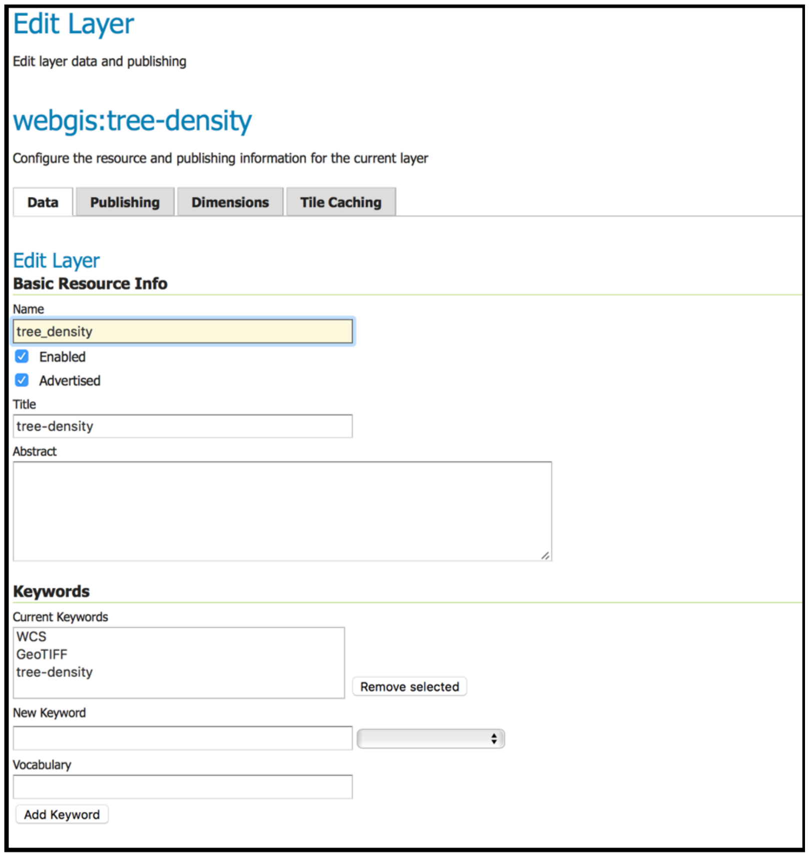
Name: tree_density

Composing coverages/bands: tree-density@0

Output bands to be created: tree-density

- Click Save
- Edit the Layer page defines the data and publishing parameters for a layer: Enter a Short Title and an Abstract for the tree_density layer

NOTE: In the Name field, please use “_” (underscore) instead of “-” (hyphen) to match the table name in postgis.


Edit Layer

Edit layer data and publishing

webgis:tree-density

Configure the resource and publishing information for the current layer

Edit Layer

Basic Resource Info

Name: tree_density

☑ Enabled

☑ Advertised

Title: tree-density

Abstract:

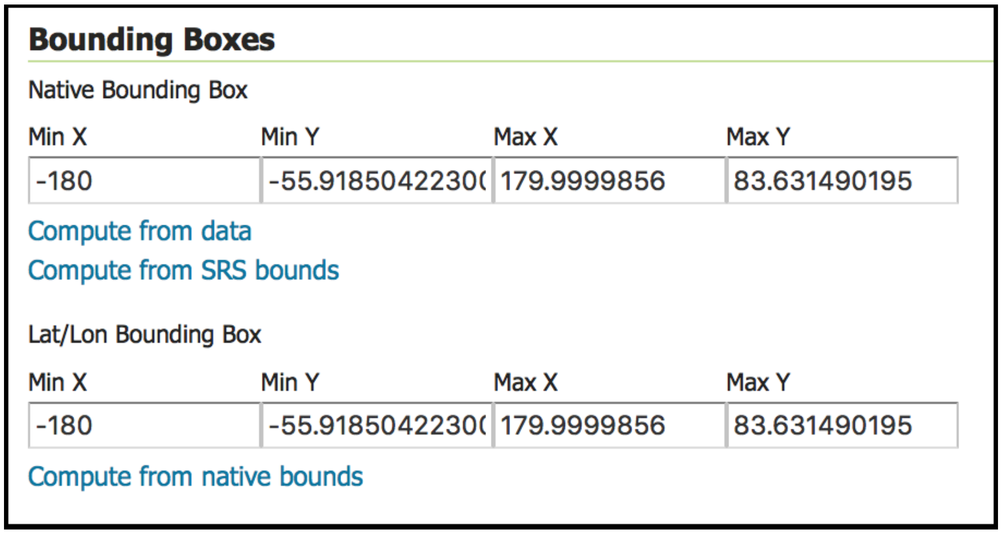
Keywords

Current Keywords:
WCS
GeoTIFF
tree-density

New Keyword:

Vocabulary:

- Generate the layer's bounding boxes by clicking the Compute form data and then compute from native bounds links.


Bounding Boxes

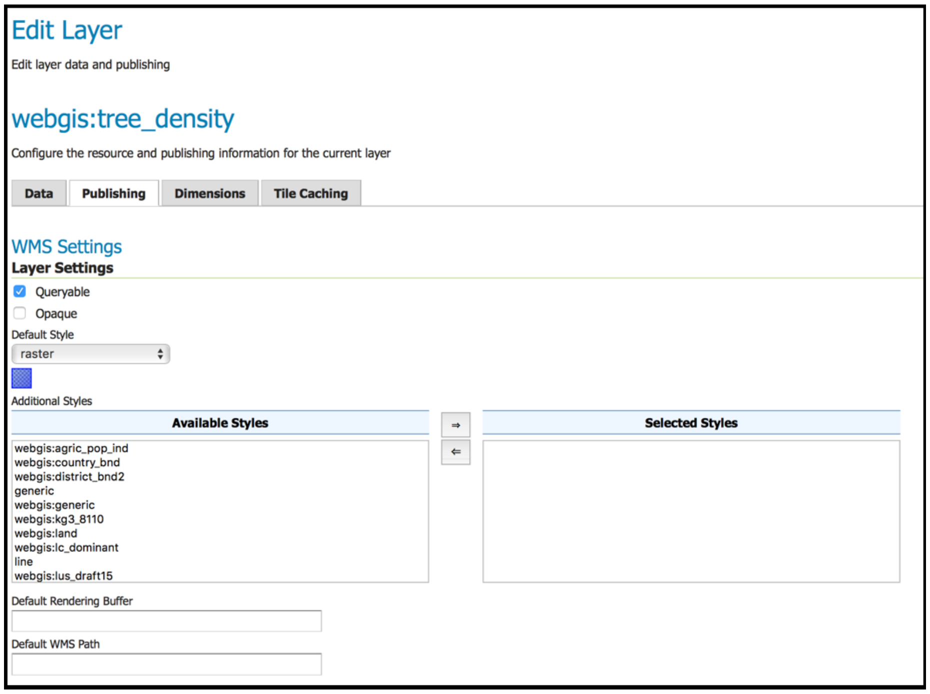
Native Bounding Box

Min X	Min Y	Max X	Max Y
-180	-55.91850422230(	179.9999856	83.631490195

Lat/Lon Bounding Box

Min X	Min Y	Max X	Max Y
-180	-55.91850422230(	179.9999856	83.631490195

- Click the Publishing tab at the top of the page.



Edit Layer

Edit layer data and publishing

webgis:tree_density

Configure the resource and publishing information for the current layer

Data | Publishing | Dimensions | Tile Caching

WMS Settings

Layer Settings

☑ Queryable

☐ Opaque

Default Style

raster

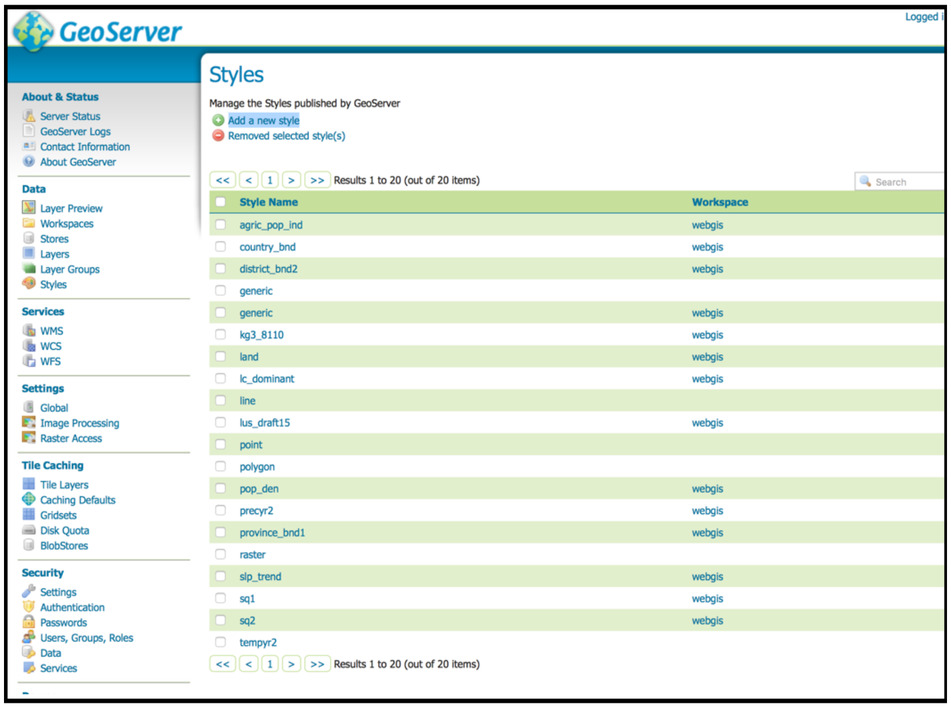
Additional Styles

Available Styles	Selected Styles
webgis:agric_pop_ind webgis:country_bnd webgis:district_bnd2 generic webgis:generic webgis:kg3_8110 webgis:land webgis:lc_dominant line webgis:lus_draft15	

Default Rendering Buffer

Default WMS Path

- In the WMS Settings, tick Queryable and leave the default style to raster. Next section, we are going to create a style for this layer.
- Finalize the layer configuration by scrolling to the bottom of the page and clicking Save.
- Create SLD Style
- Navigate to Styles->Add a new style



GeoServer

Logged i

About & Status
Server Status
GeoServer Logs
Contact Information
About GeoServer

Data
Layer Preview
Workspaces
Stores
Layers
Layer Groups
Styles

Services
WMS
WCS
WFS

Settings
Global
Image Processing
Raster Access

Tile Caching
Tile Layers
Caching Defaults
Gridsets
Disk Quota
BlobStores

Security
Settings
Authentication
Passwords
Users, Groups, Roles
Data
Services

Styles

Manage the Styles published by GeoServer

Add a new style

Removed selected style(s)

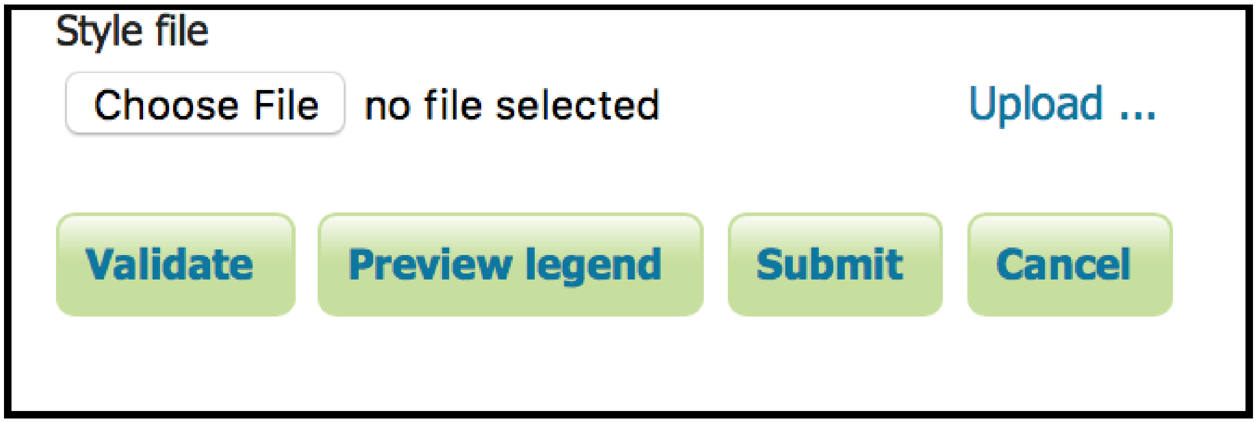
<< < 1 > >> Results 1 to 20 (out of 20 items)

Search

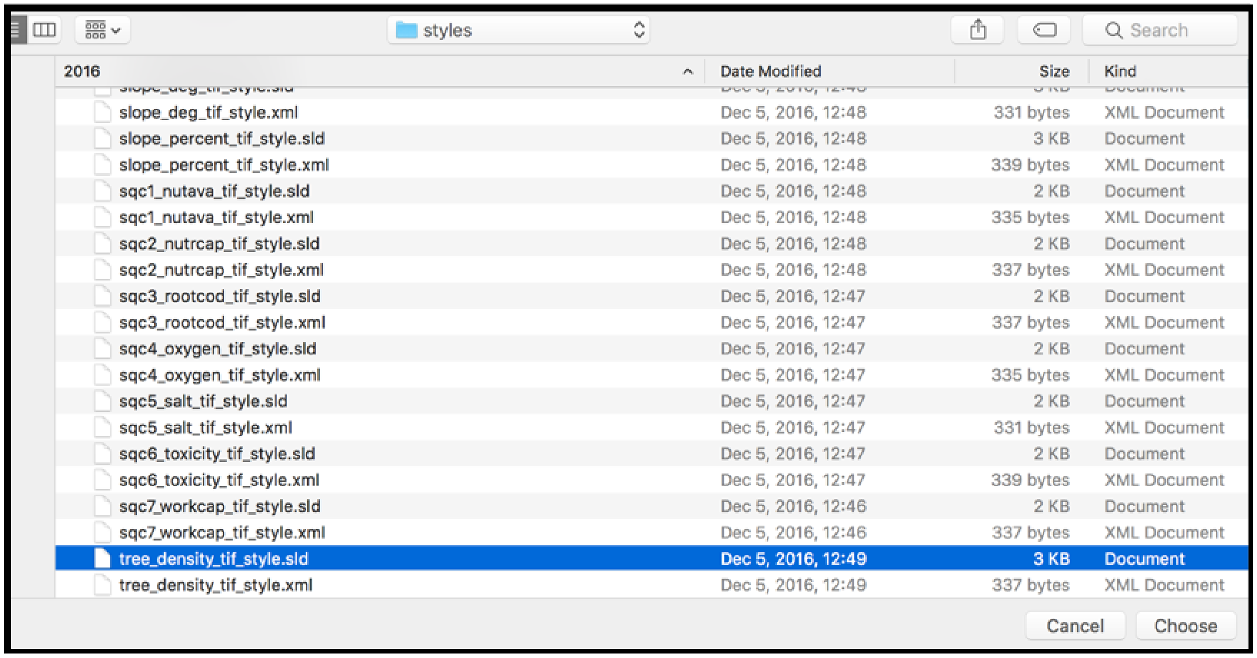
Style Name	Workspace
agric_pop_ind	webgis
country_bnd	webgis
district_bnd2	webgis
generic	
generic	webgis
kg3_8110	webgis
land	webgis
lc_dominant	webgis
line	
lus_draft15	webgis
point	
polygon	
pop_den	webgis
precyr2	webgis
province_bnd1	webgis
raster	
slp_trend	webgis
sq1	webgis
sq2	webgis
tempyr2	

<< < 1 > >> Results 1 to 20 (out of 20 items)

- Scroll to the bottom and click “Choose File”.

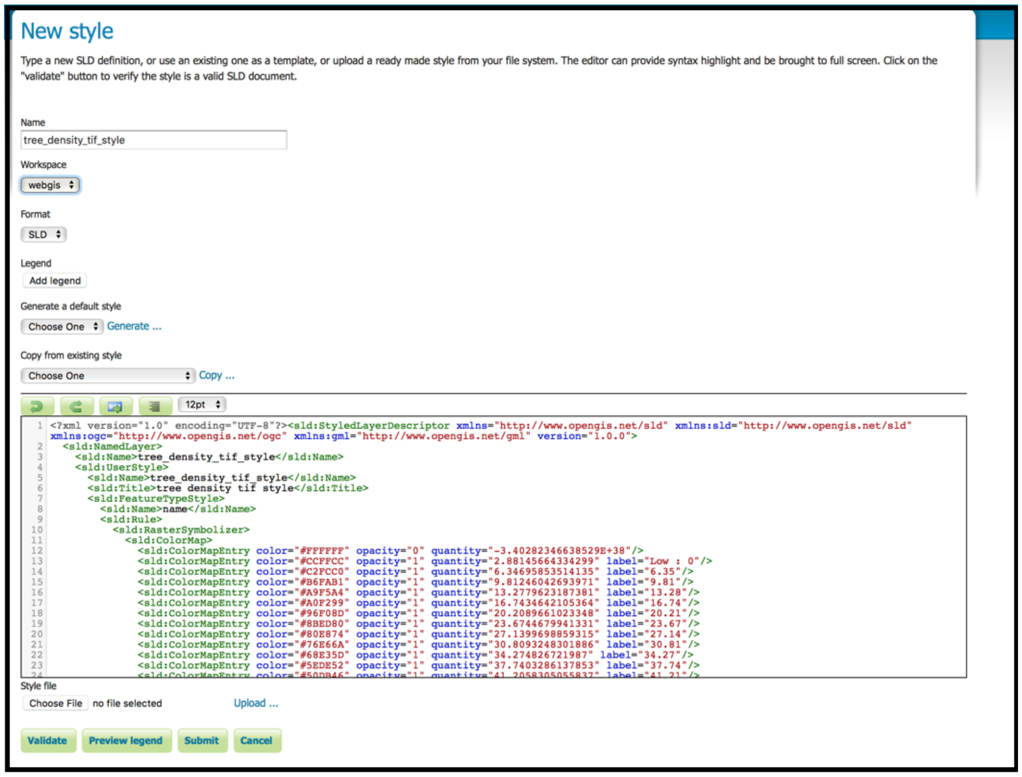


- Navigate to “Installer->styles” folder and locate “tree_density_tif_style” and click choose.



2016	Date Modified	Size	Kind
slope_deg_tif_style.xml	Dec 5, 2016, 12:48	331 bytes	XML Document
slope_percent_tif_style.sld	Dec 5, 2016, 12:48	3 KB	Document
slope_percent_tif_style.xml	Dec 5, 2016, 12:48	339 bytes	XML Document
sqc1_nutava_tif_style.sld	Dec 5, 2016, 12:48	2 KB	Document
sqc1_nutava_tif_style.xml	Dec 5, 2016, 12:48	335 bytes	XML Document
sqc2_nutrcap_tif_style.sld	Dec 5, 2016, 12:48	2 KB	Document
sqc2_nutrcap_tif_style.xml	Dec 5, 2016, 12:48	337 bytes	XML Document
sqc3_rootcod_tif_style.sld	Dec 5, 2016, 12:47	2 KB	Document
sqc3_rootcod_tif_style.xml	Dec 5, 2016, 12:47	337 bytes	XML Document
sqc4_oxygen_tif_style.sld	Dec 5, 2016, 12:47	2 KB	Document
sqc4_oxygen_tif_style.xml	Dec 5, 2016, 12:47	335 bytes	XML Document
sqc5_salt_tif_style.sld	Dec 5, 2016, 12:47	2 KB	Document
sqc5_salt_tif_style.xml	Dec 5, 2016, 12:47	331 bytes	XML Document
sqc6_toxicity_tif_style.sld	Dec 5, 2016, 12:47	2 KB	Document
sqc6_toxicity_tif_style.xml	Dec 5, 2016, 12:47	339 bytes	XML Document
sqc7_workcap_tif_style.sld	Dec 5, 2016, 12:46	2 KB	Document
sqc7_workcap_tif_style.xml	Dec 5, 2016, 12:46	337 bytes	XML Document
tree_density_tif_style.sld	Dec 5, 2016, 12:49	3 KB	Document
tree_density_tif_style.xml	Dec 5, 2016, 12:49	337 bytes	XML Document

- Click “Upload…” to load the styles into the editor and set the workspace to “webgis”.



New style

Type a new SLD definition, or use an existing one as a template, or upload a ready made style from your file system. The editor can provide syntax highlight and be brought to full screen. Click on the "validate" button to verify the style is a valid SLD document.

Name: tree_density_tif_style

Workspace: webgis

Format: SLD

Legend: Add legend

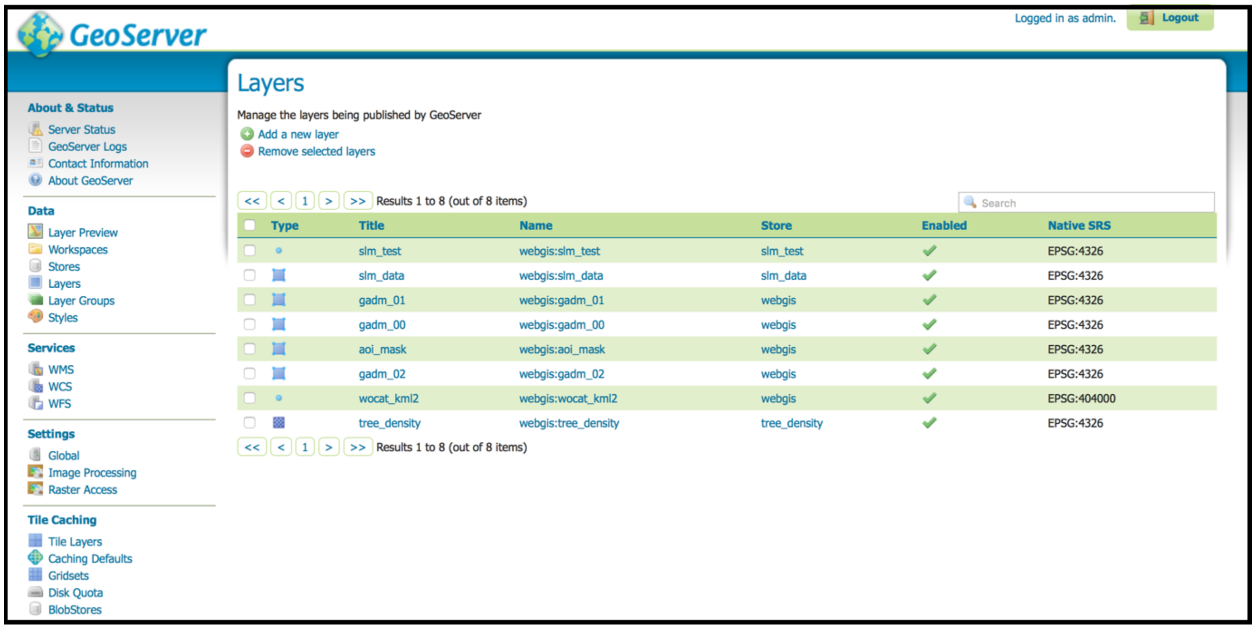
Generate a default style: Choose One — Generate ...

Copy from existing style: Choose One — Copy ...

```
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml="http://www.opengis.net/gml" version="1.0.0">
  <sld:NamedLayer>
    <sld:Name>tree_density_tif_style</sld:Name>
    <sld:UserStyle>
      <sld:Name>tree_density_tif_style</sld:Name>
      <sld:Title>tree density tif style</sld:Title>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:RasterSymbolizer>
            <sld:ColorMap>
              <sld:ColorMapEntry color="#FFFFFF" opacity="0" quantity="-3.40282346638529E+38"/>
              <sld:ColorMapEntry color="#CCFFCC" opacity="1" quantity="2.88145664334299" label="Low : 0"/>
              <sld:ColorMapEntry color="#C2FCC0" opacity="1" quantity="6.34695853514135" label="6.35"/>
              <sld:ColorMapEntry color="#B6FAB1" opacity="1" quantity="9.81246042693971" label="9.81"/>
              <sld:ColorMapEntry color="#A9F5A4" opacity="1" quantity="13.2779623187381" label="13.28"/>
              <sld:ColorMapEntry color="#A0F299" opacity="1" quantity="16.7434642105364" label="16.74"/>
              <sld:ColorMapEntry color="#96F08D" opacity="1" quantity="20.2089661023348" label="20.21"/>
              <sld:ColorMapEntry color="#8BED80" opacity="1" quantity="23.6744679941331" label="23.67"/>
              <sld:ColorMapEntry color="#80E874" opacity="1" quantity="27.1399698859315" label="27.14"/>
              <sld:ColorMapEntry color="#76E66A" opacity="1" quantity="30.8093248301886" label="30.81"/>
              <sld:ColorMapEntry color="#68E35D" opacity="1" quantity="34.274826721987" label="34.27"/>
              <sld:ColorMapEntry color="#5EDE52" opacity="1" quantity="37.7403286137853" label="37.74"/>
              <sld:ColorMapEntry color="#50DB46" opacity="1" quantity="41.2058305055837" label="41.21"/>
```

Style file: Choose File — no file selected — Upload ...

- Click Validate, to check the style for errors.
- If no error, click Submit.
- Next, navigate to Layers.
- In the title column, click tree_density layer.



Layers

Manage the layers being published by GeoServer

Add a new layer

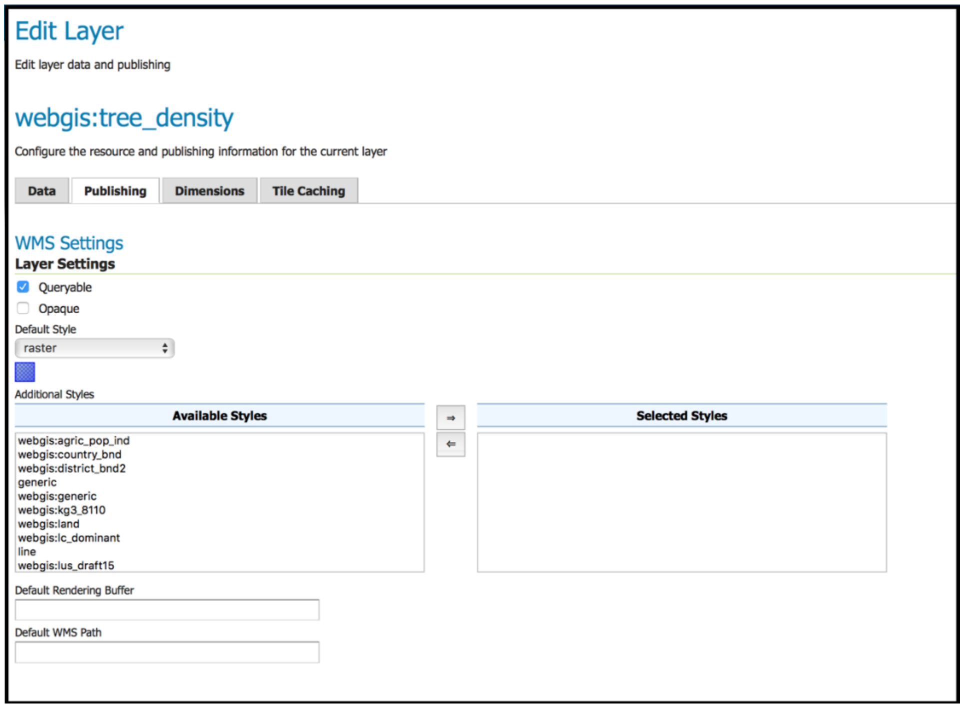
Remove selected layers

<< < 1 > >> Results 1 to 8 (out of 8 items)

	Type	Title	Name	Store	Enabled	Native SRS
☐	●	slm_test	webgis:slm_test	slm_test	✓	EPSG:4326
☐	▦	slm_data	webgis:slm_data	slm_data	✓	EPSG:4326
☐	▦	gadm_01	webgis:gadm_01	webgis	✓	EPSG:4326
☐	▦	gadm_00	webgis:gadm_00	webgis	✓	EPSG:4326
☐	▦	aoi_mask	webgis:aoi_mask	webgis	✓	EPSG:4326
☐	▦	gadm_02	webgis:gadm_02	webgis	✓	EPSG:4326
☐	●	wocat_kml2	webgis:wocat_kml2	webgis	✓	EPSG:404000
☐	▦	tree_density	webgis:tree_density	tree_density	✓	EPSG:4326

<< < 1 > >> Results 1 to 8 (out of 8 items)

- Click Publishing tab.



Edit Layer

Edit layer data and publishing

webgis:tree_density

Configure the resource and publishing information for the current layer

WMS Settings

Layer Settings

☑ Queryable

☐ Opaque

Default Style: raster

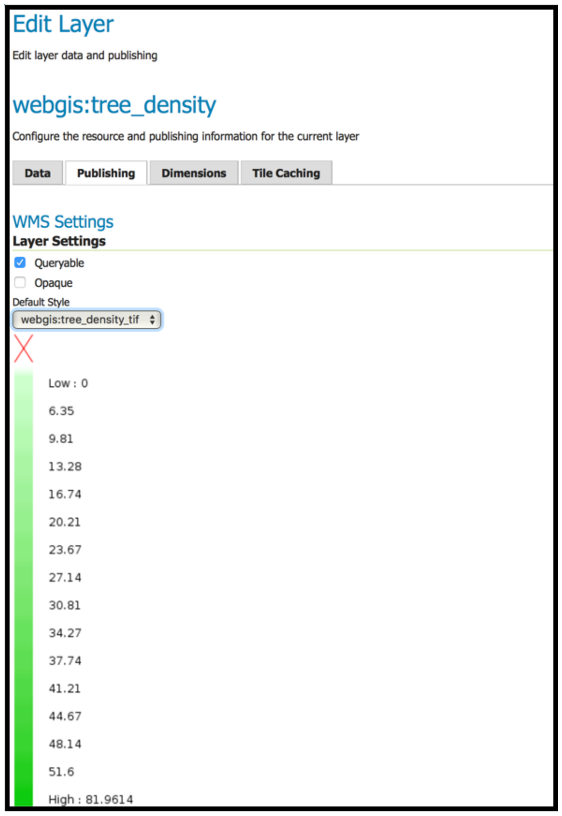
Additional Styles

Available Styles	Selected Styles
webgis:agric_pop_ind	
webgis:country_bnd	
webgis:district_bnd2	
generic	
webgis:generic	
webgis:kg3_8110	
webgis:land	
webgis:lc_dominant	
line	
webgis:lus_draft15	

Default Rendering Buffer

Default WMS Path

- In the Default Style, select the style name "webgis:tree_density_tif_style" and click Save.



Edit Layer

Edit layer data and publishing

webgis:tree_density

Configure the resource and publishing information for the current layer

WMS Settings

Layer Settings

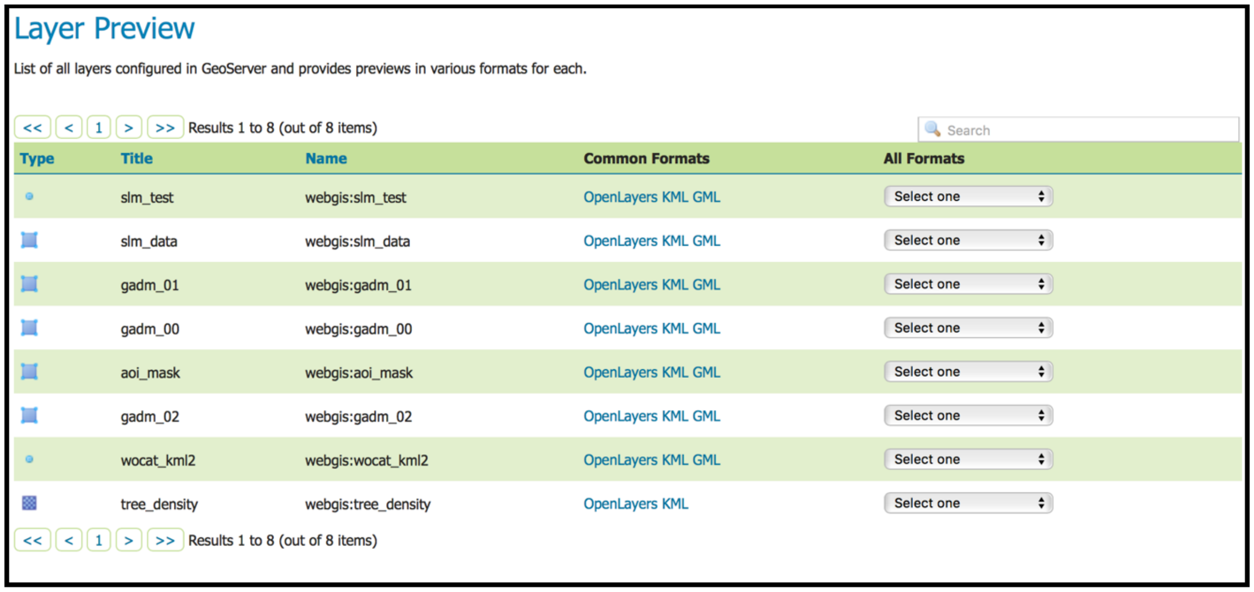
☑ Queryable

☐ Opaque

Default Style: webgis:tree_density_tif

Low : 0
6.35
9.81
13.28
16.74
20.21
23.67
27.14
30.81
34.27
37.74
41.21
44.67
48.14
51.6
High : 81.9614

- Preview the Layer: In order to verify that the **tree_density** layer is published correctly, we can preview the layer.
- Click on the Layer Preview in the left side menu.
- In the Layer Preview screen find **tree_density** layer.



Layer Preview

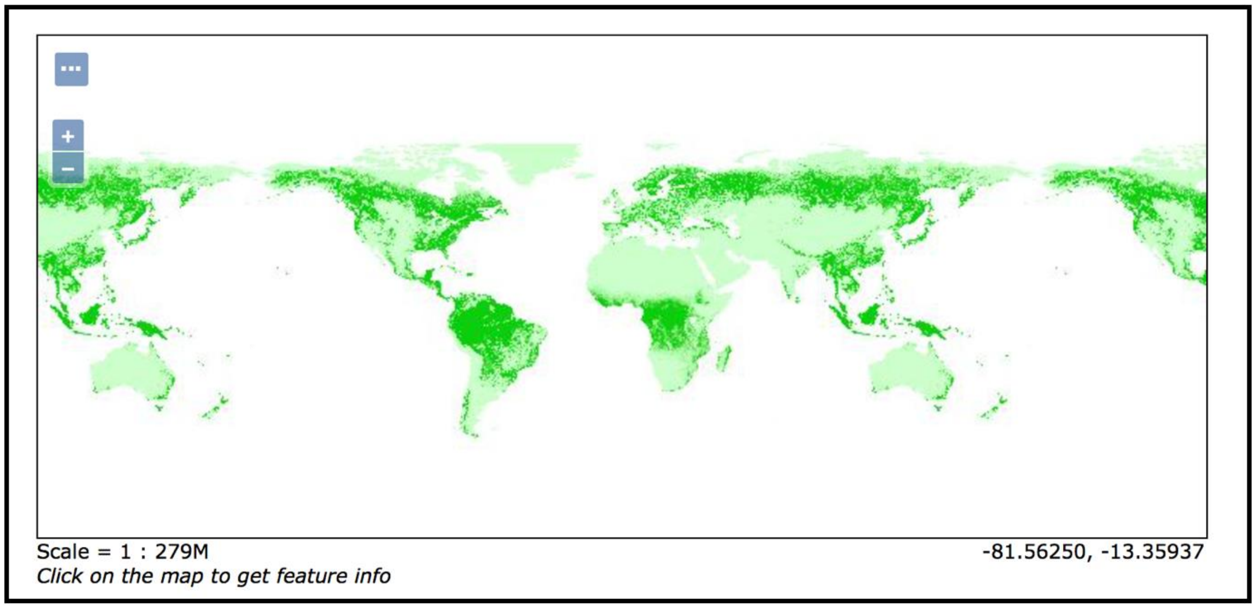
List of all layers configured in GeoServer and provides previews in various formats for each.

<< < 1 > >> Results 1 to 8 (out of 8 items)

Type	Title	Name	Common Formats	All Formats
●	slm_test	webgis:slm_test	OpenLayers KML GML	Select one
▨	slm_data	webgis:slm_data	OpenLayers KML GML	Select one
▨	gadm_01	webgis:gadm_01	OpenLayers KML GML	Select one
▨	gadm_00	webgis:gadm_00	OpenLayers KML GML	Select one
▨	aoi_mask	webgis:aoi_mask	OpenLayers KML GML	Select one
▨	gadm_02	webgis:gadm_02	OpenLayers KML GML	Select one
●	wocat_kml2	webgis:wocat_kml2	OpenLayers KML GML	Select one
▦	tree_density	webgis:tree_density	OpenLayers KML	Select one

<< < 1 > >> Results 1 to 8 (out of 8 items)

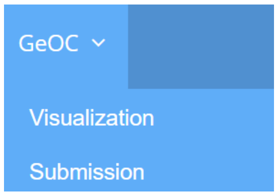
- Click the OpenLayers link in the Common Formats column.
- An OpenLayers map will load in a new tab and display the raster.



- Repeat Step 3 if you wish to publish another raster.

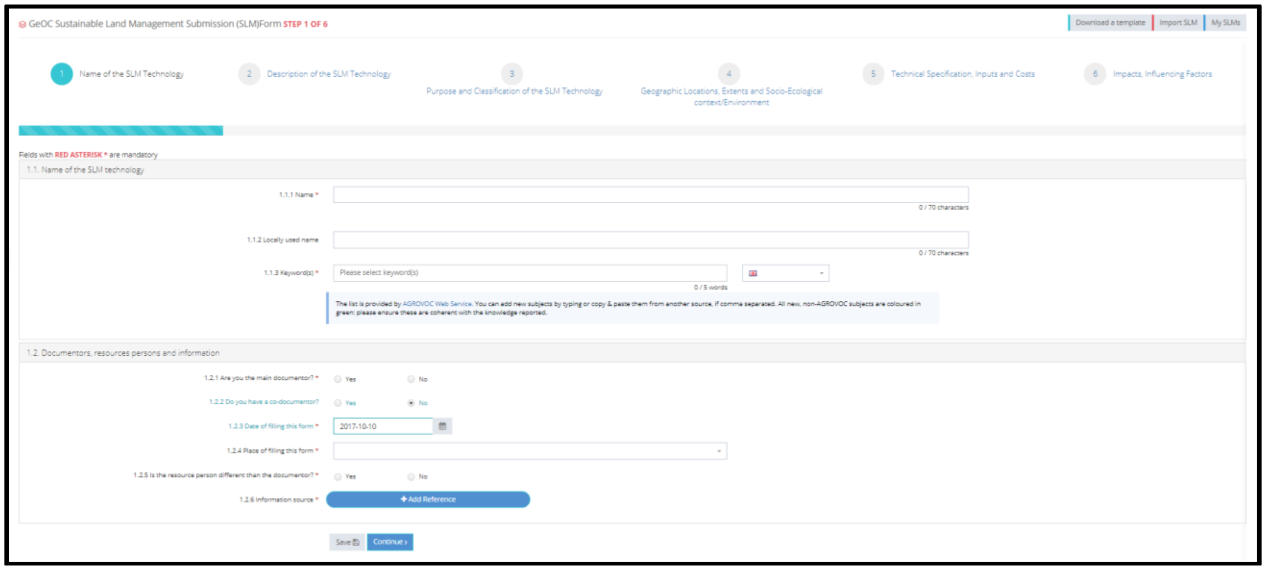
13. SLM Form Implementation

This section shows the code for SLM module, this module can be accessed from Web GIS menu.



13.1 SLM Submission Form

The submission form is the main part of SLM Module where user can enter the metadata for SLM and when the submitted SLM approved users can view the SLMs from Visualization menu item, while the SLM Approval is only visible for the SLM Admin / MEL Admin.



GeOC Sustainable Land Management Submission (SLM)Form STEP 1 OF 6

Download a template | Import SLM | My SLMs

1 Name of the SLM Technology
2 Description of the SLM Technology
3 Purpose and Classification of the SLM Technology
4 Geographic Locations, Extents and Socio-Ecological context/Environment
5 Technical Specification, Inputs and Costs
6 Impacts, Influencing Factors

Fields with RED ASTERISK * are mandatory

1.1. Name of the SLM technology

1.1.1 Name * — 0 / 70 characters

1.1.2 Locally used name — 0 / 70 characters

1.1.3 Keyword(s) * — Please select keyword(s) — 0 / 5 words

The list is provided by AGROVOC Web Service. You can add new subjects by typing or copy & paste them from another source, if comma separated. All new, non-AGROVOC subjects are coloured in green: please ensure these are coherent with the knowledge reported.

1.2. Documentors, resources persons and information

1.2.1 Are you the main documentor? * — Yes No

1.2.2 Do you have a co-documentor? — Yes [✓ No]

1.2.3 Date of filling this form * — 2017-10-10

1.2.4 Place of filling this form *

1.2.5 Is the resource person different than the documentor? * — Yes No

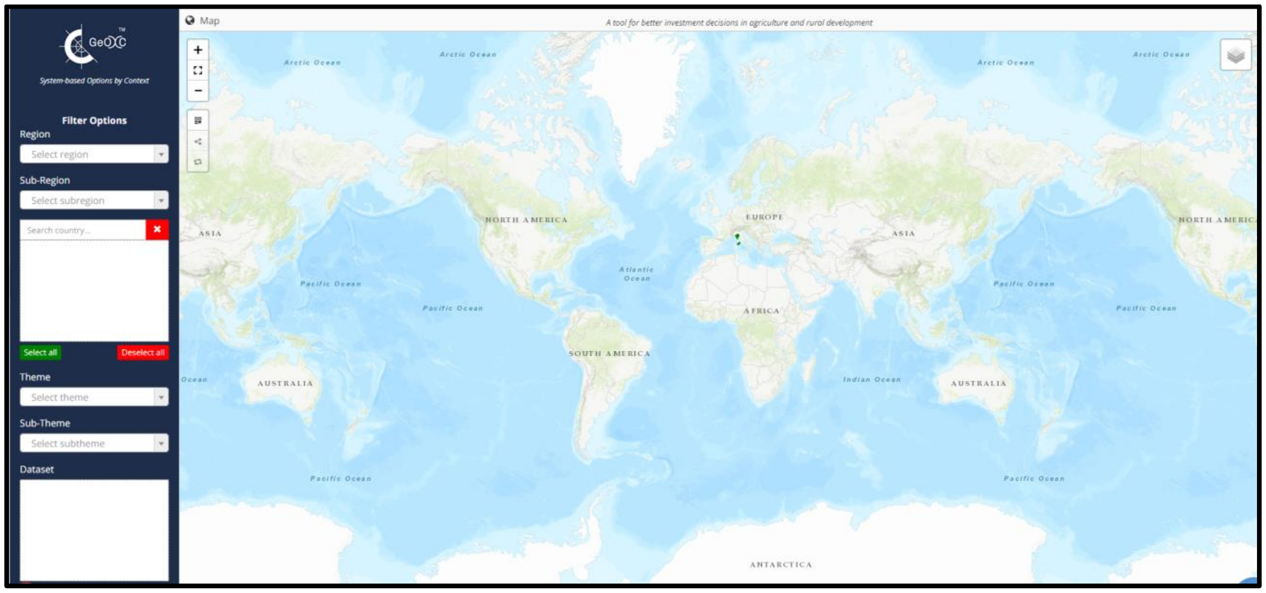
1.2.6 Information source * — + Add Reference

Save | Continue ›

- **SLM Form Files**

```
slm
 │  ├──── index.phtml                                    # Contains Html input elements and webgis polygon integration
 │  ├──── assets/admin/pages/scripts/slm/form-wizard.js  # Contains initialization javascript code for the form
 │  ├──── assets/admin/pages/scripts/slm/app.js          # Contains code for integration with webgis
 │  ├──── assets/admin/pages/scripts/slm/utils.js        # Contains code for configuration webgis connectio
 │  ├──── assets/admin/pages/css/slm-module.css          # Contains form styles
 └────
```

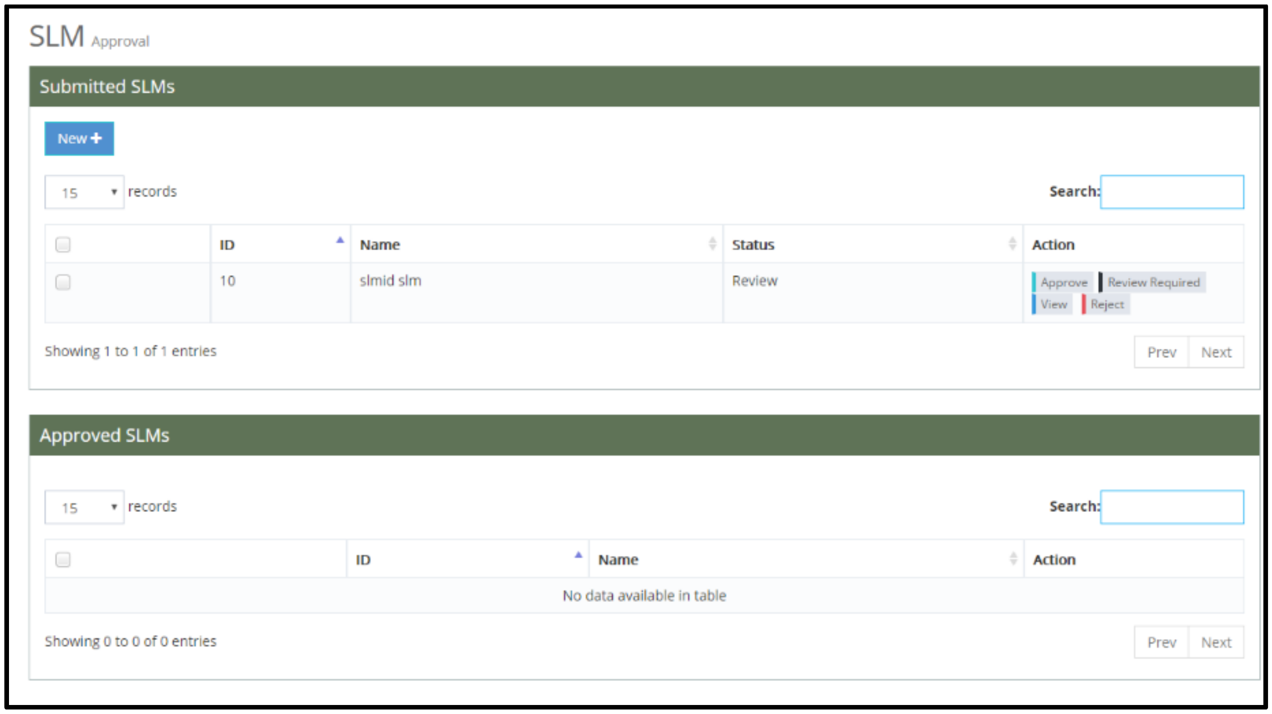
13.2 SLM Visualization



- **SLM Visualization Files**

```
slm
│ ├── visualization.phtml                          # Contain Html code for viewing the webgis tool
│ ├── assets/admin/pages/css/slm-module.css        # Contain styles
└──
```

13.3 SLM Approval



SLM Approval

Submitted SLMs

New +

15 records Search:

	ID	Name	Status	Action
☐	10	slmid slm	Review	Approve Review Required View Reject

Showing 1 to 1 of 1 entries Prev Next

Approved SLMs

15 records Search:

☐	ID	Name	Action
No data available in table			

Showing 0 to 0 of 0 entries Prev Next

- **SLM Approval Files**

```
slm
 │   ├──── approval.phtml                                              # Contain jquery data tables for approve submitted slms
 │   ├──── assets/admin/pages/scripts/slm/approval/table-managed.js    # Contain initialization javascript code for the page
 │   ├──── assets/admin/pages/scripts/slm/approval/form-validation.js  # Contains code for validation and jQuery.form
 │   ├──── assets/admin/pages/css/slm-module.css                       # Contains form styles
 └────
```

Appendix A: Acronyms

Table 1. Acronyms

Acronym	Literal Translation
API	Application Program Interface.
CDM	Conceptual Data Model.
CDN	Content Data Network.
COTS	Commercial Off-the-Shelf.
CSS	Cascading Style Sheet.
CRPs	CGIAR Research Programs.
CRUD	Create, Read, Update and Delete.
DAO	Data Access Object.
DC	Developing Countries
DDD	Database Design Document.
DS	Dryland Systems.
GL	Grain Legumes.
GUI	Graphical User Interface.
HTML	Hyper Text Markup Language.
ICD	Interface Control Document.
IE	Internet Explorer.
IDO	Intermediate Development Outcomes.
LAN	Local Area Network.
LDM	Logical Data Model.
MEL	Monitoring, Evaluation & Learning framework.
MVC	Model-View-Controller.
PDM	Physical Data Model.
PHP	PHP: Hypertext Preprocessor.
OCS	One Corporate System.
RDBMS	Relational Database Management System.
RTP	Research program on Roots, Tubers, and Bananas.
SCP	Strategy and Corporate Plan.
SDD	System Design Document.
SLOC	Source Lines of Code.
SLO	System-Level Outcome.
SRF	Strategy and Results Framework.
UTF	Unicode Transformation Format.
WAN	Wide Area Network.
W1-W2	Window 1 – Window 2.
W3/bilateral	Window 3 / bilateral.
XML	Extensible Markup Language.
ZF1	Zend Framework 1.

Appendix B: Glossary

Instructions: Provide clear and concise definitions for terms used in this document that may be unfamiliar to readers of the document. Terms are to be listed in alphabetical order.

Table 2. Glossary

Term	Definition
API	Set of routines, protocols, and tools for building software and applications.
CDM	Map of concepts and their relationships used for databases.
CDN	Globally distributed network of proxy servers deployed in multiple data centers to deliver webpages and other Web content to a user based on the geographic locations of the user.
COTS	Products that are commercially available and can be bought.
CSS	Style sheet language used for describing the presentation of a document written in a markup language.
CapDev	The process through which individuals, organizations and societies obtain, strengthen and maintain the capabilities to set and achieve their own development objectives over time.
CRUD	The four basic functions of persistent storage.
DAO	Object that provides an abstract interface to some type of database or other persistence mechanism.
Deliverable	This is intended to be a sub-component of an output that a scientist can use to show the contribution to a specific output. An integrated set of deliverable constitute the output. The deliverable can be a document (publication) a dataset, a training/workshop report/material.
DS	Partnership of several dozen actors, including national research systems from 28 countries, universities, extension agents, civil society organizations, advanced research centers, CGIAR partners, and other development partners.
GIT	Is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency
GUI	Type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation.
HTML	Standardized system for tagging text files to achieve font, colour, graphic, and hyperlink effects on World Wide Web pages.
ICD	The interface or interfaces between subsystems or to a system or subsystem.
Impact Pathway	Impact pathways describe these result chains, showing the linkages between the sequence of steps in getting to impact.
Indicator	Indicators are the measures that help define the change at the appropriate level – whether output, outcome or impact. They are precise statements that should be SMART (Specific, Measurable, Accurate, Reliable and Time-bound).

Term	Definition
LAN	Group of computers and associated devices that share a common communications line or wireless link to a server.
MVC	Software architectural pattern for implementing user interfaces on computers.
OCS	Software to help staff manage projects, human resources, and finances, and perform other administrative functions.
Output	Output is the products and services resulting directly and attributably from the activities undertaken. Outputs can be ‘bought’ in the sense that all costs associated with them should be clear and associated with the program. An output is an integrated set of deliverables for which it is possible to define a budget. The output type falls into categories defined by the CO (Technologies, Policies, Tools, Framework/Concept, Value Chain and Agro-Ecosystems assessments, and Innovation Platform).
Outcome	Outcomes are the changes (intended or realized) in the target individuals, institutions and/or systems. These changes result from the range of outputs, working together – typically mixing product-related outputs such as tools and databases, and service-related outputs around training, workshops and other fora to raise awareness, build interest and demand.
PDM	Representation of a data design which takes into account the facilities and constraints of a given database management system.
PHP	Server scripting language, and a powerful tool for making dynamic and interactive Web pages.
RDBMS	Database management system (DBMS) that is based on the relational model.
SCP	Organization's process of defining its strategy, or direction, and making decisions on allocating its resources to pursue this strategy.
SDD	Document includes the design of system components, modules, interfaces, and data for a system to satisfy specified requirements.
SLOC	Software metric used to measure the size of a computer program by counting the number of lines in the text of the program's source code.
UTF	Character encoding capable of encoding all possible characters, or code points, defined by Unicode.
WAN	Computer network that spans a relatively large geographical area.
XML	Markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

Appendix C: Approvals

Instructions: List the individuals whose signatures are desired. Examples of such individuals are Business Owner, Project Manager (if identified), and any appropriate stakeholders. Add additional lines for signature as necessary.

The undersigned acknowledge that they have reviewed the System Design Document and agree with the information presented within this document. Changes to this Document will be coordinated with, and approved by, the undersigned, or their designated representatives.

Signature:		Date:	
Print Name:	Enrico Bonaiuti		
Title:	Research Program Coordinator (DS)		
Role:	Business Owner		
Signature:		Date:	
Print Name:			
Title:			
Role:			
Signature:		Date:	
Print Name:			
Title:			
Role:			
Signature:		Date:	
Print Name:			
Title:			
Role:			