

# Democratic Map: Customized for GNDEC

Submitted for the partial fulfilment of the Degree  
of  
Bachelor of Technology  
(Computer Science Engineering)



**Submitted By:**  
Amisha Budhiraja  
1410808  
145010

**Submitted To:**  
Sukhjit Singh Sehra  
Training Co-ordinator  
CSE Department

---

Department of Computer Science & Engineering  
Guru Nanak Dev Engineering College  
Ludhiana 141006

# Abstract

This project mainly discuss about OpenStreetMap. Geographical data (geo data) is not free in many parts of the world. Generally these places have given the task of mapping to various government agencies who in return get to make money by selling the data back to you and me. The main disadvantage of Google Maps is that data is copyrighted and owned by multiple organisations like the Ordnance Survey. Google/whoever just licenses it. If we were to use it, we'd have to pay for it. This leads to the increasing demand of OSM.

You can use OSM by picking an area that you know well and use the OpenStreetMap viewer to see how well the map data corresponds to your own knowledge. As on Wikipedia, it's easy to edit, so you can help!. Also, this project is completely open source and the entire code is available to the user as and when required.

There is complete developer's Blog reference alongwith it that helps using it a lot easier. The data and software is owned by you, the contributors. By making your system an OSM tile server not only you can edit the map but can use it offline also. You can change the styling of the map like color of the roads fonts style and amny more as per your requirments.

The core part of OSM is implemented using Mapnik library and database for rendering, mod tile and slippy for web interface. Bash Shell Scripting has been used to automate the installation. There is an organisation called the OpenStreetMap Foundation which exists to protect, promote, and support the project, but does not own the data. There are lots of ways to contribute to the OpenStreetMap project.

# Acknowledgements

I, student of Guru Nanak Dev Engineering College, Ludhiana, have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.

The author is highly grateful to Dr. Sehijpal Singh, Principal of Guru Nanak Dev Engineering College, Ludhiana for providing her with the opportunity to carry out her Six month Training at Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana.

The author would like to whole heartedly thank Dr. H.S. Rai Dean, Testing and Consultancy Cell, Guru Nanak Dev Engineering College, Ludhiana who is a vast sea of knowledge and without whose constant and never ending support and motivation, it would never have been possible to complete the project and other assignments so efficiently and effectively.

Finally, I would thanks to all whoever have contributed in this report work with Amritpal Singh (D4 IT) and all other trainees. Without their encouragement, it would not have been possible to complete this project in such an efficient manner.

Amisha Budhiraja

<b>1</b>	<b>Introduction To Organisation</b>	<b>1</b>
1.1	Testing and Consutancy Cell . . . . .	1
<b>2</b>	<b>Introduction to Project</b>	<b>3</b>
2.1	Overview . . . . .	3
2.2	Existing System . . . . .	3
2.3	User Requirement Analysis . . . . .	4
2.3.1	Users of the System . . . . .	4
2.3.2	Functional Requirements . . . . .	5
2.3.3	Non functional requirements . . . . .	6
2.4	Feasibility Study . . . . .	6
2.4.1	Types of Feasibility Study . . . . .	6
2.4.1.1	Technical Feasibility . . . . .	6
2.4.1.2	Economic Feasibility . . . . .	7
2.4.1.3	Behavioral Feasibility . . . . .	7
2.5	Objectives of Project . . . . .	8
<b>3</b>	<b>PROJECT DESIGN</b>	<b>10</b>
3.1	Product Perspective . . . . .	10
3.2	Product Functions . . . . .	10
3.3	User Characteristics . . . . .	11
3.3.1	The General User . . . . .	11
3.3.2	Designers . . . . .	12
3.3.3	Developers . . . . .	12
3.4	Constraints . . . . .	12
3.5	Flowchart . . . . .	12
3.5.1	Detailed Description . . . . .	13
3.5.2	DFD's . . . . .	13
3.6	Database design . . . . .	13
3.7	Table Structure . . . . .	17
3.8	Assumptions and Dependencies . . . . .	17
3.8.1	Dependency Graph . . . . .	18
3.8.2	Class Diagrams . . . . .	18
3.9	Entity Relationship(ER) Diagram . . . . .	20
3.10	Specific Requirements . . . . .	20
<b>4</b>	<b>Development and Implementation</b>	<b>24</b>
4.1	Introduction to Languages . . . . .	24
4.1.1	HTML . . . . .	24
4.1.2	CSS . . . . .	25
4.1.3	CMake . . . . .	26
4.1.4	Shell Scripting . . . . .	26
4.1.5	Python . . . . .	28
4.1.6	Features of Python . . . . .	28
4.1.7	JSON . . . . .	28

4.2	Ubuntu: An open source OS . . . . .	28
4.3	Introduction To Doxygen . . . . .	29
4.3.1	Installation of Doxygen . . . . .	30
4.4	Introduction to L <sup>A</sup> T <sub>E</sub> X . . . . .	30
4.4.1	Typesetting . . . . .	32
4.5	Introduction to Github . . . . .	32
4.5.1	What is Git? . . . . .	34
4.5.2	Installation of Git . . . . .	35
4.5.3	Various Git Commands . . . . .	35
4.5.3.1	Create Repositories . . . . .	35
4.5.3.2	Make Changes . . . . .	35
4.5.3.3	Group Changes . . . . .	36
4.5.3.4	Synchronize Changes . . . . .	36
4.6	Introduction to Reveal-js & Reveal-md . . . . .	36
4.6.1	Installation of reveal-md . . . . .	37
4.7	Working with Experimental Server . . . . .	37
4.8	Python Django Framework . . . . .	38
4.9	OSM and its Components . . . . .	39
4.9.1	Benifits Of Own Tile Server . . . . .	39
4.9.2	Postgresql / postgis . . . . .	39
4.9.3	Osm2pgsql . . . . .	40
4.9.4	Openstreetmap-carto . . . . .	40
4.9.5	OpenLayer.js . . . . .	41
4.9.6	Geographic Information Systems (GIS) . . . . .	41
4.10	Implementation . . . . .	41
4.11	Testing . . . . .	47
<b>5</b>	<b>Conclusion and Future Scope</b>	<b>50</b>
5.1	Conclusion . . . . .	50
5.2	Future Scope . . . . .	50
	<b>BIBLIOGRAPHY</b>	<b>50</b>

1.1	Guru Nanak Dev Engineering College . . . . .	1
2.1	OpenStreetMap . . . . .	3
3.1	Flowchart of Democratic Maps . . . . .	13
3.2	Flow Chart to Serve a Tile . . . . .	14
3.3	Data Flow Diagram Level 0 . . . . .	14
3.4	Data Flow Diagram Level 1 . . . . .	15
3.5	Data Flow Diagram Level 2 . . . . .	15
3.6	Database design . . . . .	16
3.7	Dependency graph of osmium index object . . . . .	18
3.8	Caller graph of expiration of the tiles in osm2pgsql . . . . .	19
3.9	Caller graph to generate_road_colours . . . . .	19
3.10	Caller graph of tag tranform in postgresql . . . . .	19
3.11	Class Diagram for generating road colours . . . . .	20
3.12	Class Diagram for tagtransform in osm2pgsql . . . . .	21
3.13	Class Diagram of osmdata . . . . .	22
3.14	Class Diagram for reprojection . . . . .	23
3.15	Component model of Democratic maps . . . . .	23
4.1	HTML5 Logo . . . . .	24
4.2	CSS3 . . . . .	25
4.3	Javascript . . . . .	26
4.4	Bootstrap . . . . .	26
4.5	test.cpp file . . . . .	27
4.6	cmake file . . . . .	27
4.7	Python . . . . .	28
4.8	Ubuntu . . . . .	29
4.9	Doxygen . . . . .	29
4.10	Main Page of pbOSM . . . . .	30
4.11	Namespace documentation of pbOSM . . . . .	31
4.12	Donald Knuth, Inventor Of T <sub>E</sub> X typesetting system . . . . .	31
4.13	Output of the avove program . . . . .	33
4.14	Github Logo . . . . .	33
4.15	Git Logo . . . . .	34
4.16	MD & JS . . . . .	36
4.17	Server Communication . . . . .	37
4.18	Python Django Framework . . . . .	38
4.19	OpenStreetMap . . . . .	39
4.20	Postgresql . . . . .	40
4.21	Openstreetmap-carto Style . . . . .	40
4.22	OSM Map on Web browser . . . . .	42
4.23	Building with customize color . . . . .	42
4.24	International Boundary of India . . . . .	43
4.25	Customized icon representing Pub shop . . . . .	43

4.26	Map of Punjab in Punjabi . . . . .	44
4.27	India divided into states with different color . . . . .	44
4.28	India divided into districts with different color . . . . .	44
4.29	Increased zoom level for indoor mapping . . . . .	45
4.30	User Input Page . . . . .	45
4.31	Php Page . . . . .	45
4.32	Verifing system to be OSM server using script . . . . .	46
4.33	Representing 3-D view of GNE . . . . .	46
4.34	Searched place and pop up menu . . . . .	46
4.35	Representing Animations . . . . .	47
4.36	Map on remote server . . . . .	47

3.1	Map uses different database schemas . . . . .	16
3.2	Database tables . . . . .	17
4.1	Unit testing . . . . .	48
4.2	Integration testing . . . . .	49





Figure 1.1: Guru Nanak Dev Engineering College

I had my Six Month Industrial Training at TCC-Testing And Consultancy Cell under the guidance of Dr. H.S.Rai Dean TCC, GNDEC Ludhiana. Guru Nanak Dev Engineering College was established by the Nankana Sahib Education Trust Ludhiana. The Nankana Sahib Education Trust i.e NSET was founded in memory of the most sacred temple of Sri Nankana Sahib, birth place of Sri Guru Nanak Dev Ji. With the mission of Removal of Economic Backwardness through Technology Shiromani Gurudwara Parbandhak Committee i.e SGPC started a Poly technical was started in 1953 and Guru Nanak Dev Engineering College was established in 1956.

The main goal of this institute is:

- To build and promote teams of experts in the upcoming specialisations.
- To promote quality research and undertake research projects keeping in view their relevance to needs and requirements of technology in local industry.
- To achieve total financial independence.
- To start online transfer of knowledge in appropriate technology by means of establishing multipurpose resource centres.

## 1.1 Testing and Consutancy Cell

Testing and Consultancy Cell was established in the year 1979 with a basic aim to produce quality service for technical problems at reasonable and affordable rates as a service to society in general and Engineering fraternity in particular.

Consultancy Services are being rendered by various Departments of the College to the industry, State Government Departments and Entrepreneurs and are extended in the form of expert advice in design, testing of materials & equipment, technical surveys, technical audit, calibration of instruments, preparation of technical feasibility reports etc. This consultancy cell of the college has given a new dimension to the development programmers of the College. Consultancy projects of over Rs. one crore are completed by the Consultancy cell during financial year 2009-10.

Ours is a pioneer institute providing Consultancy Services in the States of Punjab, Haryana, Himachal, J&K and Rajasthan. Various Major Clients of the Consultancy Cell are as under:

- Northern Railway, Govt. of India
- Indian Oil Corporation Ltd.
- Larson & Turbo.
- Multi National Companies like AFCON & PAULINGS.
- Punjab Water Supply & Sewage Board

## 2.1 Overview

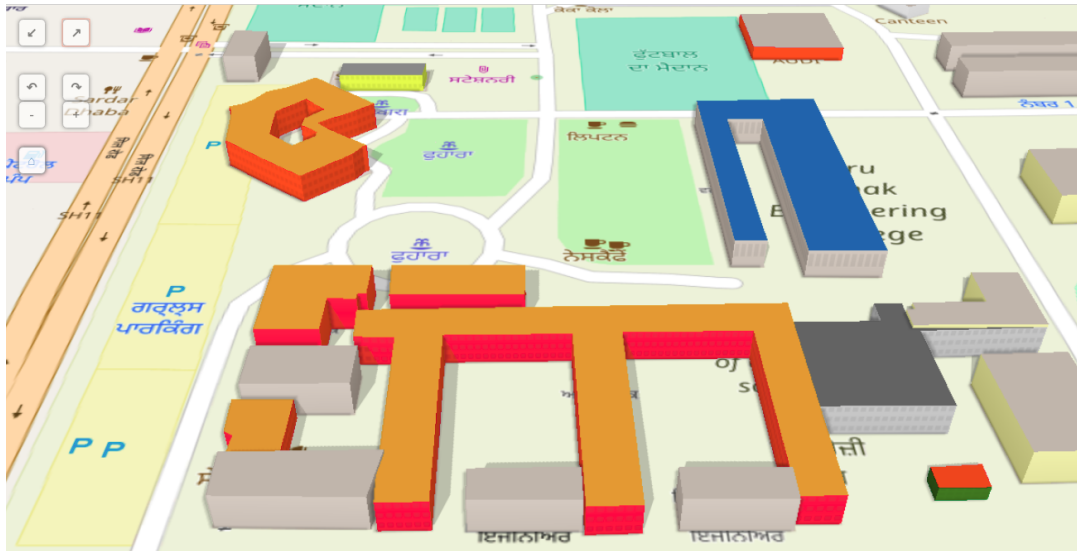


Figure 2.1: OpenStreetMap

OpenStreetMap (OSM) is an open-source, free web-based software, owned by you, the contributors. OpenStreetMap is an online open data platform to collect the world's geographic data based on the Wikipedia model of crowdsourcing. The project started in 2004 by Steve Coast and is now governed by the non profit OpenStreetMap Foundation based in the UK.

OpenStreetMap is a free editable map of the whole world. It is made by people like you. Which means the database will always be subject to the whims, experimentation, and mistakes of the community. This is precisely OSMs strength since, among other things, it allows our data to quickly accommodate changes in the physical world.

By making your system an OSM tile server not only you can edit the map but can use it offline also. You can change the styling of the map like color of the roads fonts style and amny more as per your requirments.

The core part of OSM is implemented using Mapnik library and database for rendering, mod.tile and slippy for web interface. Bash Shell Scripting has been used to automate the installation.

My training being not based on particular language or technology, different type of open-source softwares and technologies are used in this project and many during my training which are not used in this project like CGI (for web interface through c++).

## 2.2 Existing System

Geographical data (geo data) is not free in many parts of the world. If you collect data from Google Maps in this way, you are creating a "derived work". Any such data retains the copy-

right conditions of the original. In practice, this means your data is subject to the licensing fees, and contractual restrictions, of these map providers. That's exactly what OpenStreetMap is trying to avoid. The data is copyrighted and owned by multiple organisations like the Ordnance Survey. Google/whoever just licenses it. If we were to use it, we'd have to pay for it.

In areas where there are no such data sources (most areas) we have to start from a blank slate, and head out there to survey the streets ourselves. Despite starting from scratch, we have achieved a good level of completion in many places.

"Also, you may not use Google Maps in a manner which gives you or any other person access to mass downloads or bulk feeds of numerical latitude and longitude coordinates."

### **Limitations of the existing system**

- We can't edit the maps.
- Data may be inaccurate.
- They are costly.
- Can't create own map server.
- Mass downloads or bulk feeds of numerical latitude and longitude coordinates is sometime impossible.

## **2.3 User Requirement Analysis**

User Requirements Analysis for a software system is a complete description of the requirements of the User. It includes functional Requirements and Non-functional Requirements. Non-functional requirements are requirements which impose constraints on the design or implementation.

- **Purpose:** OpenStreetMap (OSM) is an open collaborative project to create a free editable map of the world and the main purpose of this project is to:
  1. To create a free editable map of the world.
  2. To gather location data using GPS, local knowledge, and other free sources of information and upload it.
  3. To encourage the growth, development and distribution of free geospatial data.
  4. To provide geospatial data for anyone to use and share.
  5. Reduce the time for analysis.
  6. The OpenStreetMap Foundation is an international not-for-profit organization supporting, but not controlling, the OpenStreetMap Project.

### **2.3.1 Users of the System**

1. Provides beautiful GUI (Graphical User Interface) for GNE Tour and animations.
2. A full editing history is stored for each user.

3. Provide on-line way to analysis so that individual does not have to install anything.
4. Users can attach Wikipedia-like edit summaries to their edits, and there is a History tab on the main page that shows recent edits to the selected area.
5. The user can download the data in \*.pbm or \*.osm file format.
6. Both technical and non-technical users can use OSM.
7. User can make own OSM tile server.
8. User can run script for automatic installation.
9. They can search places with ease.

### 2.3.2 Functional Requirements

- **Specific Requirements:** This phase covers the whole requirements for the system. After understanding the system we need the input data to the system then we watch the output and determine whether the output from the system is according to our requirements or not. So what we have to input and then what we'll get as output is given in this phase. This phase also describe the software and non-function requirements of the system.
- **Input Requirements of the System**
  1. Guess points and name of the places.
  2. Precision
  3. Required point at which value is to be found
  4. Knowledge of latitude and longitude.
- **Output Requirements of the System**
  1. Final output of the location of the particular area.
  2. Shops, restaurants and many more are represented through icon and images.
- **Special User Requirements**
  1. Taking bulk input values through html forms.
- **Software Requirements**
  1. Programming language: C++, Python
  2. software:  $\text{\LaTeX}$
  3. Web Languages: php, javascript, html
  4. Database: Postgresql
  5. Documentation: Doxygen 1.8.3

6. Text Editor: Vim
7. Operating System: Ubuntu 14.04 or 15.10
8. Revision System: Git

### 2.3.3 Non functional requirements

1. Scalability: System should be able to handle a number of users. For e.g., handling around thousand users at the same time.
2. Usability: Simple user interfaces that a layman can understand.
3. Speed: Processing input should be done in reasonable time i.e. we can say maximum 24 hrs.

## 2.4 Feasibility Study

Feasibility study aims to uncover the strengths and weaknesses of a project. In its simplest term, the two criteria to judge feasibility are cost required and value to be attained. As such, a well-designed feasibility analysis should provide a historical background of the project, description of the project or service, details of the operations and management and legal requirements.

The objective of the feasibility study is to establish the reasons for developing the software that is acceptable to users, adaptable to change and conformable to established standards. Objectives of feasibility study are listed below:

- To analyze whether the software will meet organizational requirements.
- To determine whether the software can be implemented using the current technology and within the specified budget and schedule.
- To determine whether the software can be integrated with other existing software.

Generally, feasibility analysis precedes technical development and project implementation. These are some feasibility factors by which we can determine that the project is feasible or not:

### 2.4.1 Types of Feasibility Study

Various types of feasibility that are commonly considered include technical feasibility, economic feasibility, and behavioural feasibility.

#### 2.4.1.1 Technical Feasibility

The Technical feasibility assessment is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of the proposed system. It is an evaluation of the hardware and software and how it meets the need of the proposed system. This assessment is based on an outline design of system requirements, to determine whether the company has the technical expertise to handle completion of the project.

This whole project is based on Open Source Environment and is part of an open source software which would be deployed on any OS.

The project is developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology. Through the technology may become obsolete after some period of time, due to the fact that never version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using Java the project is technically feasible for development.

Democratic Maps is technically feasible as it is built up using various open source technologies and it can run on any platform.

### **2.4.1.2 Economic Feasibility**

The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification of all the benefits expected. This assessment typically involves a cost/ benefits analysis.

Economic feasibility is the cost and logistical outlook for a business project or endeavor. Prior to embarking on a new venture, most businesses conduct an economic feasibility study, which is a study that analyzes data to determine whether the cost of the prospective new venture will ultimately be profitable to the company. Economic feasibility is sometimes determined within an organization, while other times companies hire an external company that specializes in conducting economic feasibility studies for them.

In addition, it is necessary to consider the benefits that can be achieved by developing the software. Software is said to be economically feasible if it focuses on the issues listed below.

- Cost incurred on software development to produce long-term gains for an organization.
- Cost required to conduct full software investigation (such as requirements elicitation and requirements analysis).
- Cost of hardware, software, development team, and training.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.4.1.3 Behavioral Feasibility**

Behavioral feasibility assesses the extent to which the required software performs a series of steps to solve business problems and user requirements. It is a measure of how well the solution of problems or a specific alternative solution will work in the organization. It is also measure of how people feel about the system. If the system is not easy to operate, than operational process would be difficult. The operator of the system should be given proper training. The system should be made such that the user can interface the system without any problem.

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the

requirements identified in the requirements analysis phase of system development. The operational feasibility assessment focuses on the degree to which the proposed development projects fits in with the existing business environment and objectives with regard to development schedule, delivery date, corporate culture, and existing business processes.

To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters such as reliability, maintainability, supportability, usability, producibility, disposability, sustainability, affordability and others. These parameters are required to be considered at the early stages of design if desired operational behaviors are to be realized. A system design and development requires appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineered into the design. Therefore, operational feasibility is a critical aspect of systems engineering that needs to be an integral part of the early design phases. This feasibility is dependent on human resources (software development team) and involves visualizing whether the software will operate after it is developed and be operative once it is installed. Operational feasibility also performs the following tasks.

- Determines whether the problems anticipated in user requirements are of high priority.
- Determines whether the solution suggested by the software development team is acceptable.
- Analyzes whether users will adapt to a new software.
- Determines whether the organization is satisfied by the alternative solutions proposed by the software development team.

This includes the following questions:

- The project provides sufficient support for the users as the tiles are already stored in system.
- The proposed system would not cause any harm as it is running on a server rather than a client side.
- The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

## 2.5 Objectives of Project

The main objective of this project is to help GNE freshers to locate the places like labs, Admin Block, TCC etc from phone or laptop easily. The map is provided in Punjabi language. They can easily search the place by typing in the search button. In order to entertain them, the projects includes animations, GNE Tour and lot more.

1. The map includes 3-D View with the shadows of buildings.



2. Styling of the map by adding international boundary of India.
3. Automation for making the system an OSM tile server.

### 3.1 Product Perspective

This product is supposed to be part of an open source, under the GNU General Public license. It is a software to create the maps by using various softwares like mapnik, database, mod\_tile, openstreetmap-carto, osm2pgsql. Democratic maps is the idea given by the user's only to fulfill their needs. It had been in demand even before its consentment. It will extend this already powerful system by allowing the user to create an OSM tile server with automation script. It will also provide a way to modify the set of parameters using config file.

The following are the main features that are included in Democratic maps.

1. **Cross platform support:** Offers operating support for most of the known and commercial operating systems in form of binaries and also it can be compiled on other platforms.
2. **Styling of Map:** Colors of the buildings, roads, primary lines, secondary lines etc have been customized and then re-render the map to view the changes in the map.
3. **Language Customizer:** Customize the language of the map by applying Algorithm- if Punjabi name is provided then first priority goes to it followed by hindi and then English.
4. **Configurable OSM Automation tile server:** It is the shell non-interactive, one time configurable script (user has to change hardly two three parameters inside it) at the initial stage and then can run the script.
5. **Event Handling:** It is used to control the movement of the map through the arrow keys of keyboard.
6. **3-D View of the Map:** It represents OSM Buildings with multiple stories, shadows of the building, sun, sky and many more. Most of these features are implemented with JOSM tool like creating 3-D tank.

### 3.2 Product Functions

Functions performed by Customizer are:

- **Provision to give your own stylesheets:** This means there is a syntax which could be used to generate maps with different stylesheets. The stylesheets that we intend to support are:
  1. Shapefiles
  2. Directly from postgresql database.
  3. Protocolbuffer Binary Format.
  4. xml file
  5. osm file

- **Provide Syntax for adding ten layers in mod\_tile:** Customer also provide more than one mod\_tile layers and store tiles separately for each layers. It is used to run more than one style sheets at a time.
- **Provides Syntax to increase the zoom level to 28:** This means user can zoom into the 28 levels of the map by providing an algorithm.
- **Provides Syntax to see the difference between two data files:** This feature is used to see the difference between two data files using osmosis.
- **Provides Syntax to convert format of data file from one to other:** This feature is used to convert format of data file from one to other using shp2pgsql.
- **Provides Syntax to make certain parameters Global** User is free to change domain name, postgresql database name, unix user name for tile server.
- **Provides Syntax to change the colour of polygon at different zoom levels:** This means user can change the colour and transparency of the highlighted area in the maps.
- **Provides option to see GNE tour:** This feature describes the important places of the camp by viewing a GNE tour.
- **Provides Save the set of parameters in JSON file:** This feature saves the OSM-Buildings data in JSON file with each building as an object.
- **Provides support to storing the tiles in png format** This means the after generating the tiles they are stored in png format.
- **Provides Cmd-line support to render the maps:** This means that maps can also be render from the command line.

### 3.3 User Characteristics

We have identified three potential classifications of users of our system:

1. Designers: Designers are the people who create model for their own use or for commercial use.
2. The Client: These are the people which will customzie view to their need made by openstreetmap-carto before ordering or printing model drawing themselves.
3. Developers: These are people who might want to integrate this new feature of Democratic maps into their systems.

#### 3.3.1 The General User

All users can be assumed to have the following characteristics:

1. Ability to read and understand English.
2. Familiarity with the operation of the basic Graphical User Interface (GUI) components of Democratic maps.
3. Beyond the above, no further facility with computer technology can be assumed.

### 3.3.2 Designers

The Designer can be assumed to have the following characteristics:

1. Basic Knowledge of Democratic maps.
2. Basic Knowledge of Creating maps.
3. Basic coding skills.
4. Optional experience of cmd-line.

### 3.3.3 Developers

The Developers can be assumed to have the following characteristics:

1. Basic Knowledge of programming.
2. Basic Knowledge of Democratic maps.
3. Ability to program in cmd-line.

## 3.4 Constraints

The project constraint, is any restriction that defines a project's limitations; the scope, for example, is the limit of what the project is expected to accomplish.

The following are the main constraints while making your system an OSM server.

1. **Minimum 2GB RAM:** The system requires minimum 2 GB RAM to compile `mod_tile` and `mapnik`.
2. **Core processors:** The number of core processors is directly dependent on the dumping the data into database. More the database more CPU processor. For downloading the world's data it requires all core processors.
3. **Hard disk Memory:** OSM data take 30 GB to download the world's data. If the data is require for commercial use than fine else if require for developing than extract small amount of data.

## 3.5 Flowchart

A flowchart is a type of diagram that represents an algorithm, work flow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows and the flowchart 3.1 of Democratic maps showing the flow of control and Data in the software.

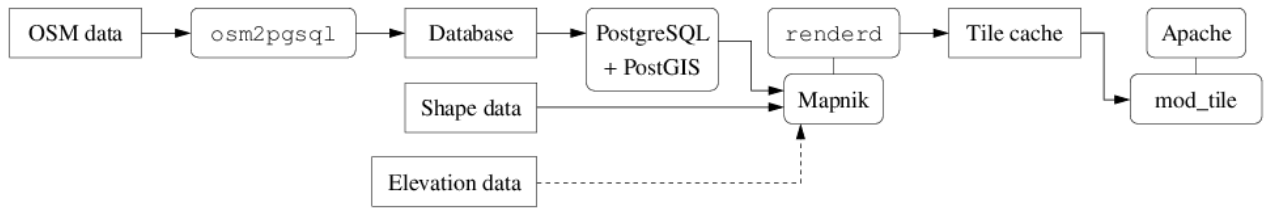


Figure 3.1: Flowchart of Democratic Maps

### 3.5.1 Detailed Description

The basic implementation of this project is almost done in form of prototype. There is need to modify the structure of the project. We have to divide the task into there parts:

1. **Front end** It will deal with how the Democratic maps will look to the user like in form of toolbars, menus etc. This part will include two parts:
  - (a) **Rebar Addon toolbar/menus** It contains a list of different animations.
  - (b) **Dialog box** User can input latitude and longitude of the point to locate the point.
2. **Back End** The slippymap send to render the tiles on the fly. The tiles are stored for caching the tiles mapnik creates the tiles. The utility osm2pgsql is used to convert raw data to postgresql database. At backend openstreetma-carto fetch the database and create beautiful maps and displayed on the browser.

### 3.5.2 DFD's

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs to serve a tile is as following:

## 3.6 Database design

The database contains tables for each Element type (nodes, ways, relations). In fact for each of these there are several database tables: current, history, current\_tags, history\_tags. In addition there are database tables for storing changeset, gpx\_files, users, diary entries, sessions, oauth etc. Democratic maps uses different database schemas for different applications.

1. **Updatable:** This can be extremely important for keeping world-wide databases up-to-date, as it allows the database to be kept up-to-date without requiring a complete (and space- and time-consuming) full, worldwide re-import. However, if you only need a small extract, then re-importing that extract may be a quicker and easier method to keep up-to-date than using the OsmChange diffs.

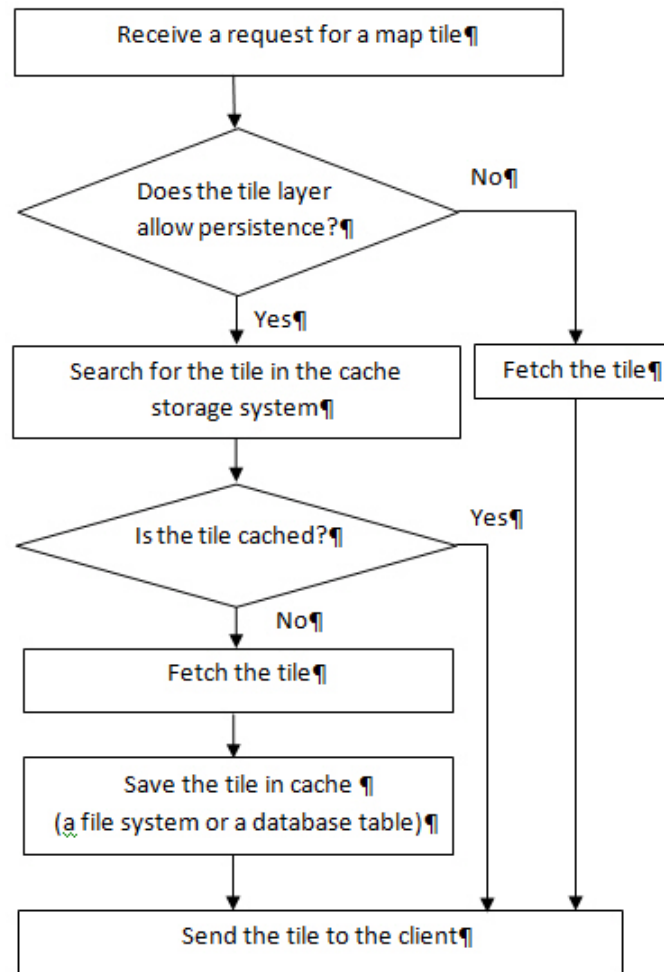


Figure 3.2: Flow Chart to Serve a Tile

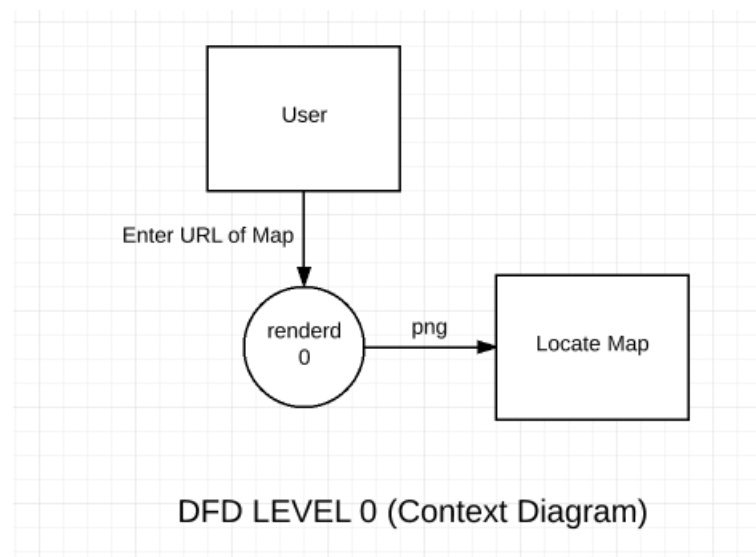


Figure 3.3: Data Flow Diagram Level 0

2. **Geometries:** Some database schemas provide native (e.g: PostGIS) geometries, which allows their use in other pieces of software which can read those geometry formats. Other

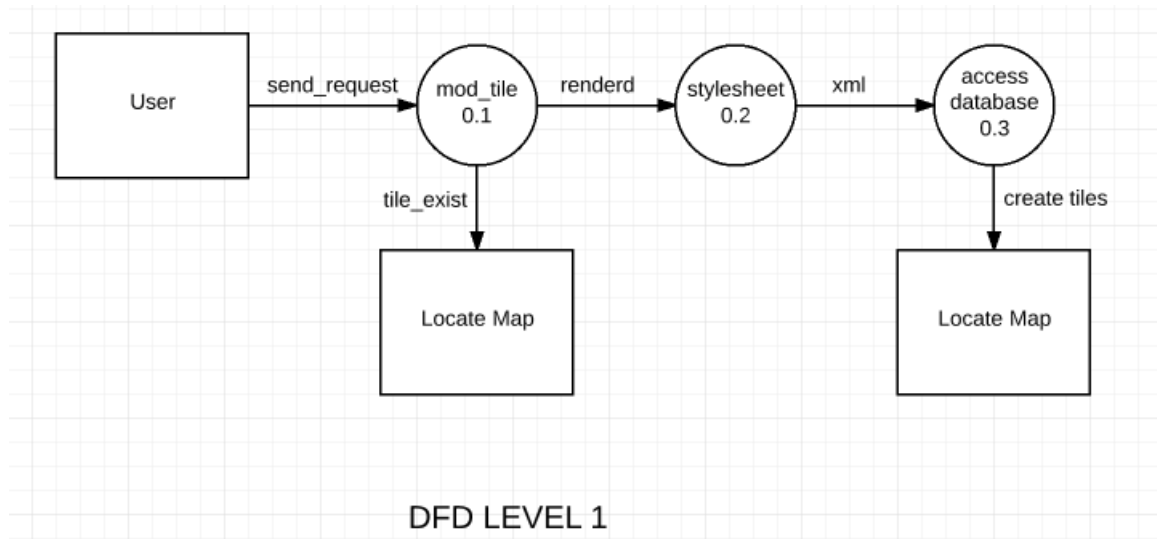


Figure 3.4: Data Flow Diagram Level 1

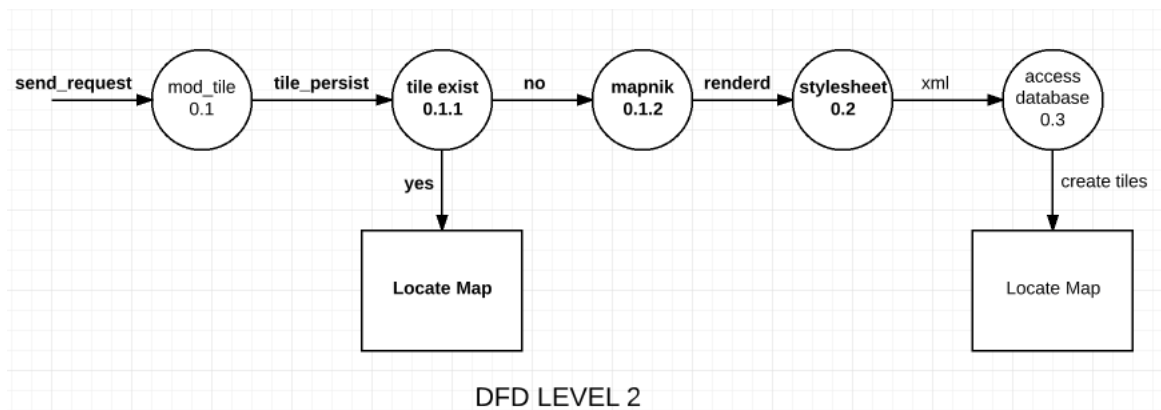


Figure 3.5: Data Flow Diagram Level 2

database schemas may provide enough data to produce the geometries (e.g: nodes, ways, relations and their linkage) but not in a native format. Some can provide both. If you want to use the database with other bits of software such as a GIS editor then you probably want a schema with these geometries pre-built.

3. **Lossless:** Some schemas will retain the full set of OSM data, including versioning, user IDs, changeset information and all tags. This information is important for editors, and may be of importance to someone doing analysis.
4. **hstore columns:** hstore is perhaps the most straightforward approach to represent OSM's freeform tagging in PostgreSQL. However, not all tools use it and other databases might not have (or need) an equivalent.

Table 3.1: Map uses different database schemas

Schema name	Created with	Primary use case	Updatable	Geometries	Lossless	hstore columns
osm2pgsql	osm2pgsql	Rendering	yes	yes	no	optional
nomina-tim	osm2pgsql	search, geocoding	yes	yes	no	optional
apidb	osmosis	Mirroring	yes	no	yes	no
way-change	SQL	Data cache and analysis	only a schema	optional	optional	no
osm2pgsql	osm2pgsql	Rendering	yes	yes	no	optional

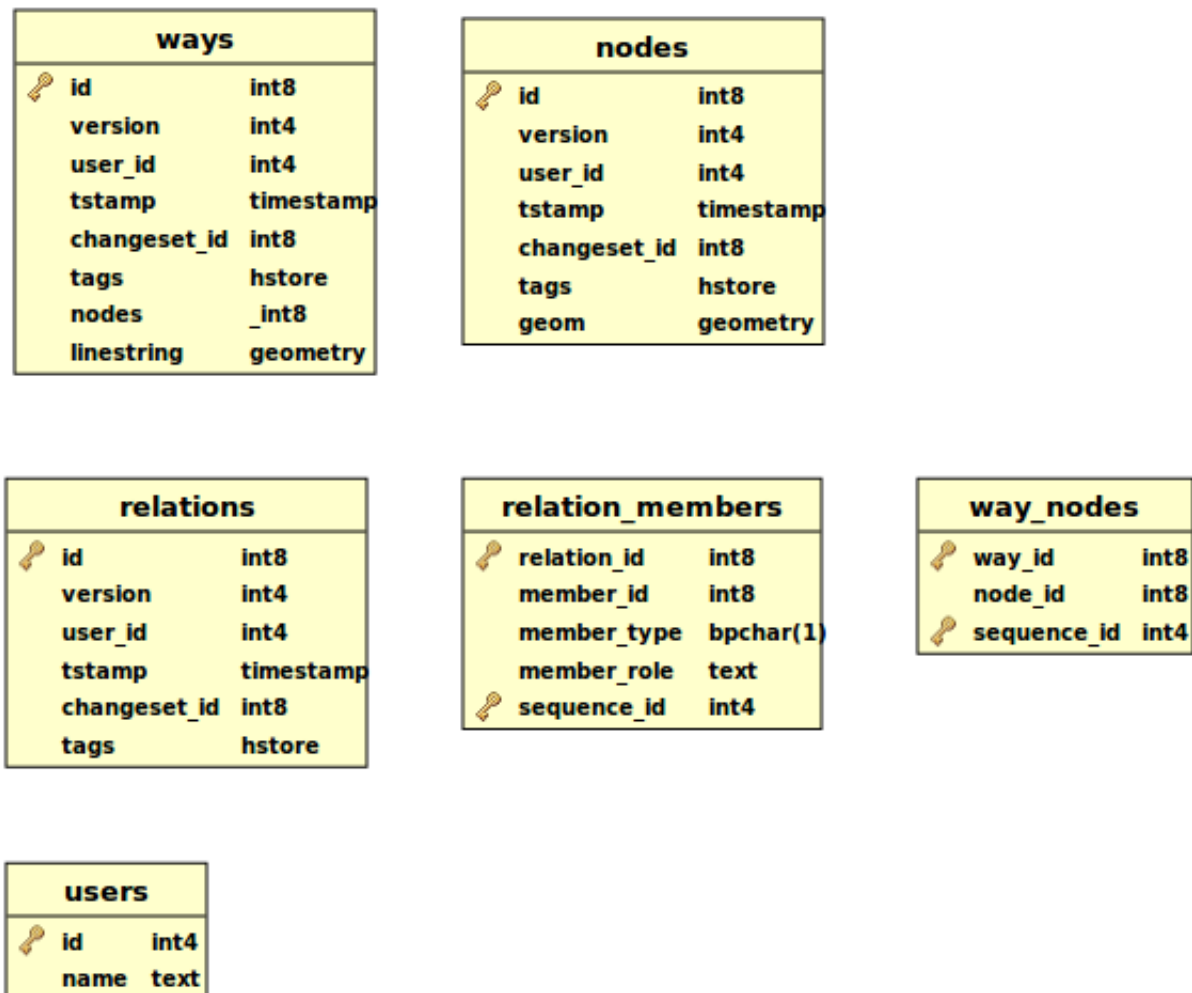


Figure 3.6: Database design



Table 3.2: Database tables

Schema	Name	Type	Owner	Size
public	geography_columns	view	postgres	0 bytes
public	geometry_columns	view	amisha	0 bytes
public	planet_osm_line	table	amisha	796MB
public	planet_osm_nodes	table	amisha	2838MB
public	planet_osm_point	table	amisha	172MB
public	planet_osm_polygon	table	amisha	1232MB
public	planet_osm_rels	table	amisha	14MB
public	planet_osm_roads	table	amisha	131MB
public	planet_osm_ways	table	amisha	1397MB
public	raster_columns	view	postgres	0 bytes
public	raster_overviews	view	postgres	0 bytes
public	spatial_ref_sys	table	amisha	4008 KB

## 3.7 Table Structure

## 3.8 Assumptions and Dependencies

- Operating System: Linux/Windows
- Processor Speed: 512KHz or more
- RAM: Minimum 2GB
- Library: Mapnik
- Modules: Mod\_tile
- Compiler: CartoCSS
- Stylesheet: OSMBright
- Programming Language: C++, Python

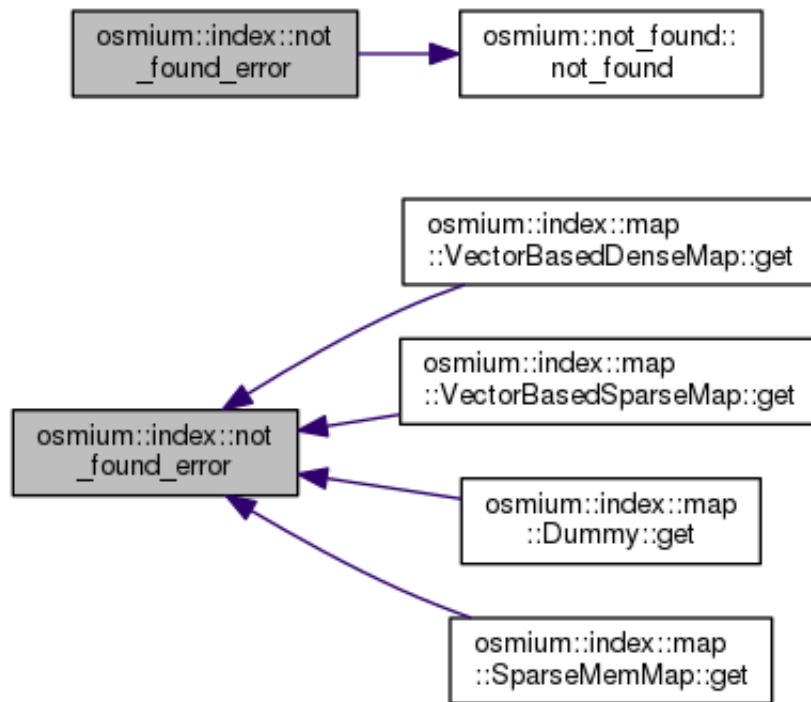


Figure 3.7: Dependency graph of osmium index object

### 3.8.1 Dependency Graph

A Dependency Graph is a graphical representation of the which module is dependent on which other modules. A Dependency Graph is often used as a preliminary step to creating an overview of the system. Dependency Graph also gives overview of how good is the design of the system. FreeCAD being were huge software it would be difficult to make the dependency graph of whole software. So, here is Dependency Graph of openstreetmap-carto is as following:

1. **Caller graph of osmium index object:** Figure 3.7 shows the modules that use Key-value containers with unique integer values for a key.
2. **Caller graph of expiration of the tiles in osm2pgsql:** Figure 3.8 shows the modules that expire the tile in osm2pgsql. If bounding box too big - just expire tiles on the line
3. **Caller graph to generate\_road\_colours:** Figure 3.9 show the modules that generate the road colours based on primary, secondary roads.
4. **Caller graph of tag tranform in postgresql:** Figure 3.10 show the modules that uses tags tranform with transform lua filters.

### 3.8.2 Class Diagrams

Class Diagrams describe the static structure of the system. Following classes diagram represent the relationship between different classes in :

1. Figure 3.11 shows the class diagram of the road colours class which is the base class of which is the base class of lch, rgb, CIE color space.

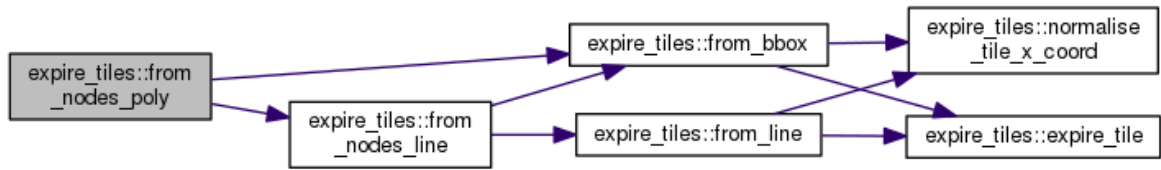


Figure 3.8: Caller graph of expiration of the tiles in osm2pgsql

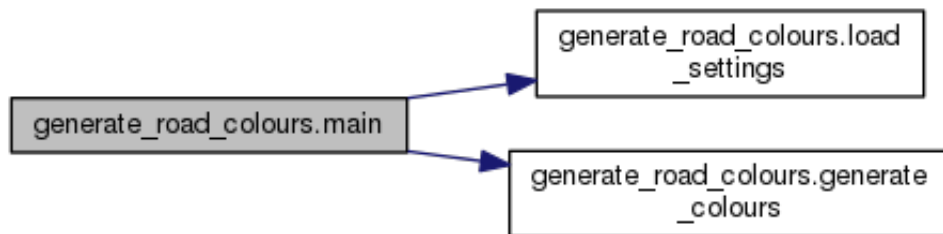


Figure 3.9: Caller graph to generate\_road\_colours

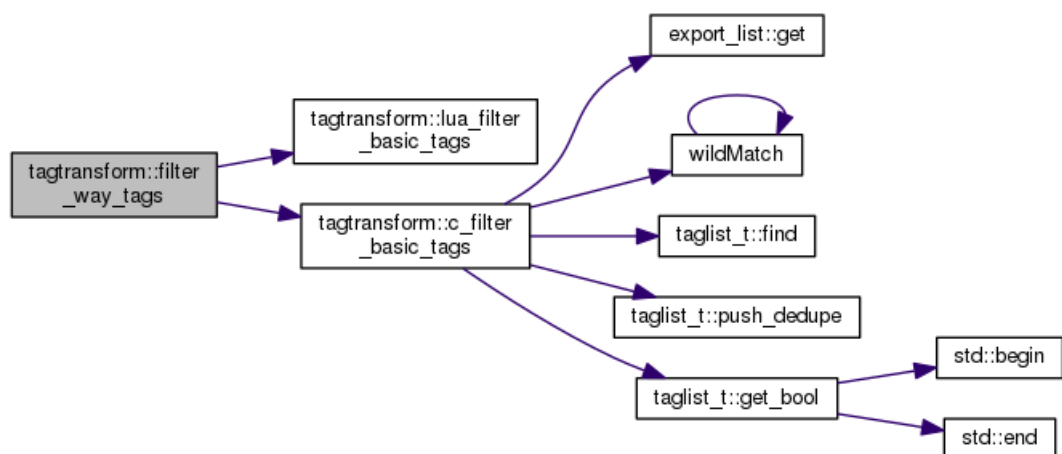


Figure 3.10: Caller graph of tag tranform in postgresql

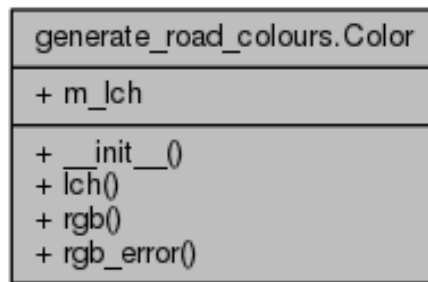


Figure 3.11: Class Diagram for generating road colours

2. Figure 3.12 shows the class diagram of the `tagstranform` which is the main class of `database_options`, `lua filter tags`, `bbox`, `hstore_columns`, `slim`, `cache`.
3. Figure 3.13 shows the class diagram of the `osmdata` which is the main class of `node_add`, `way_add`, `relation_add`, `relation_modify`, `node_delete`, `way_delete`, `relation_delete`.
4. Figure 3.14 shows the inheritance diagram of `reprojecion` which has `target_latlon`, `target_to_tile`, `reproject`, `create_projection`.

## 3.9 Entity Relationship(ER) Diagram

An entity relationship model, also called an entity-relationship (ER) diagram, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data-an object or concept about which data is stored.

## 3.10 Specific Requirements

Before starting the project,

### Learn Linux:

Before starting with project, we have to install various things to make our system an OSM server. So, for that you should know terminal commands because I gonna explain it for Ubuntu only. It is possible on other OS also but you have to work it own. I have provided some basic command also for Linux.

### Learn Postgresql:

We have to go through the basics of postgresql(database) also, such that there should not be any problem proceeding with project.

### Make or Cmake

The softwares like `mapnik`, `mod_tile`, `osm2pgsql`, are compiled through the Cmake which is basically language. So, we should the basics of it.

### Languages:

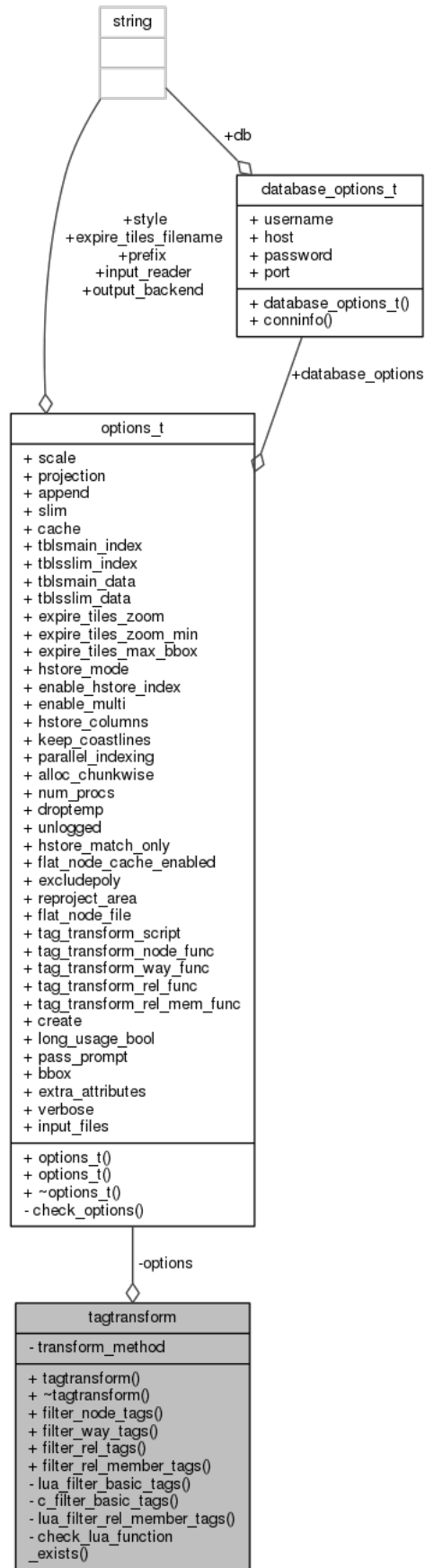


Figure 3.12: Class Diagram for tagtransform in osm2pgsql

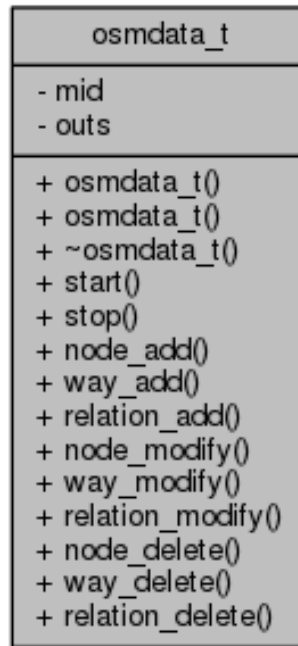


Figure 3.13: Class Diagram of osmdata

We should the basics of the languages like C++, javascript, python etc for manipulating the styling and rendering of the map.

### **Input:**

Input values are taken from user or default values defined in the file are used.

### **Output:**

According to input values we will get the particular location of the map.

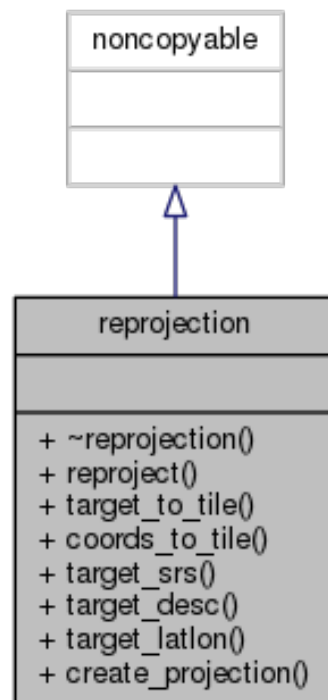


Figure 3.14: Class Diagram for reprojection

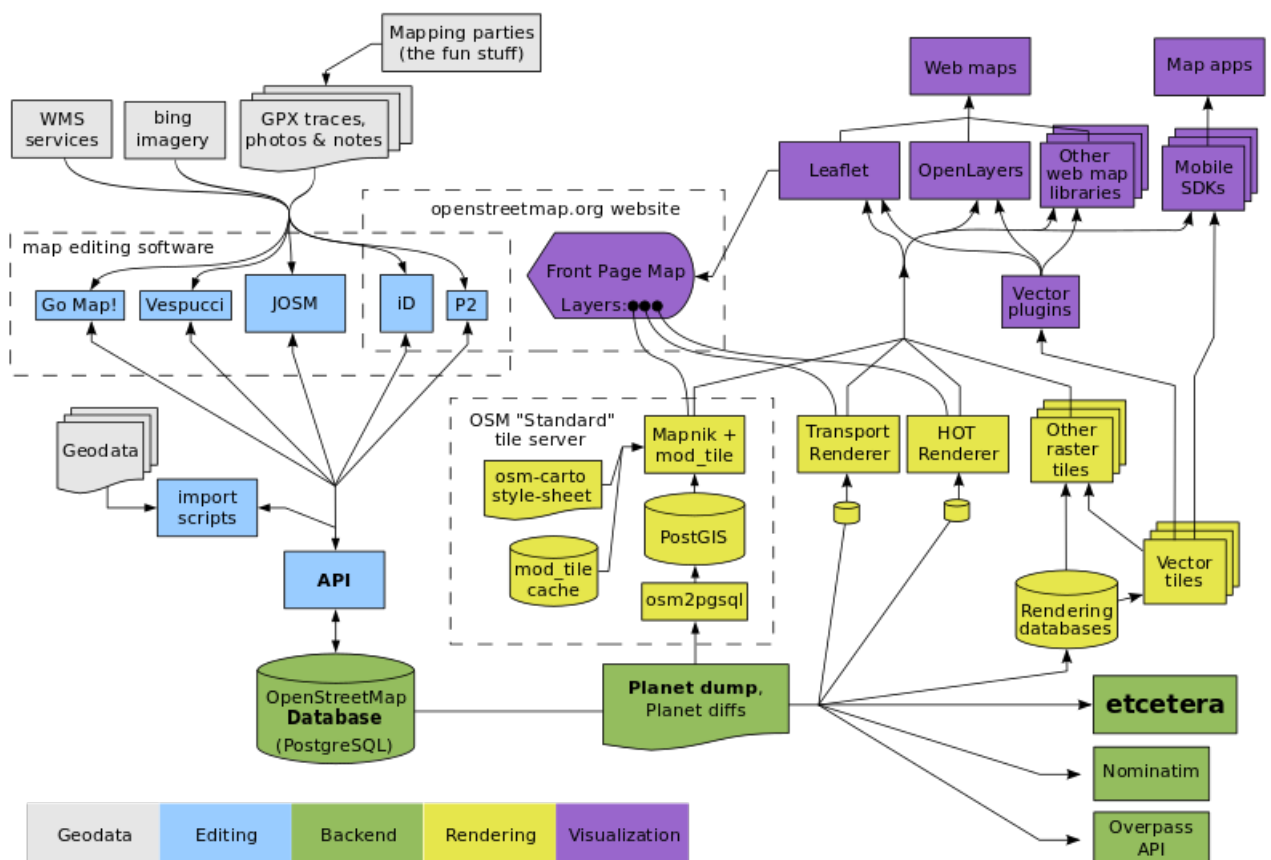


Figure 3.15: Component model of Democratic maps

## 4.1 Introduction to Languages

Front End languages are language that are used to give better user experience and user interface. These mainly include HTML, CSS, Javascript. Some Frameworks like Bootstrap are also used with these basic languages.

### 4.1.1 HTML



Figure 4.1: HTML5 Logo

HyperText Markup Language, commonly referred to as HTML, is the standard markup language used to create web pages. Along with CSS, and JavaScript, HTML is a cornerstone technology, used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications. Web browsers can read HTML files and render them into visible or audible web pages. HTML describes the structure of a website semantically along with cues for presentation, making it a markup language, rather than a programming language.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```



### 4.1.2 CSS



Figure 4.2: CSS3

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers. CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions

```
p {  
    color: red;  
    text-align: center;  
}
```

JavaScript (/dvskrpt/) is a high-level, dynamic, untyped, and interpreted programming language. It has been standardized in the ECMAScript language specification. Alongside HTML and CSS, it is one of the three essential technologies of World Wide Web content production; the majority of websites employ it and it is supported by all modern web browsers without plug-ins. JavaScript is prototype-based with first-class functions, making it a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles. It has an API for working with text, arrays, dates and regular expressions, but does not include any I/O, such as networking, storage or graphics facilities, relying for these upon the host environment in which it is embedded.

Bootstrap is a free and open-source collection of tools for creating websites and web applications. It contains HTML and CSS-based design templates for typography, forms, buttons,



Figure 4.3: Javascript



Figure 4.4: Bootstrap

navigation and other interface components, as well as optional JavaScript extensions.

It aims to ease the development of dynamic websites and web applications.

Bootstrap is a front end framework, that is, an interface for the user, unlike the server-side code which resides on the "back end" or server.

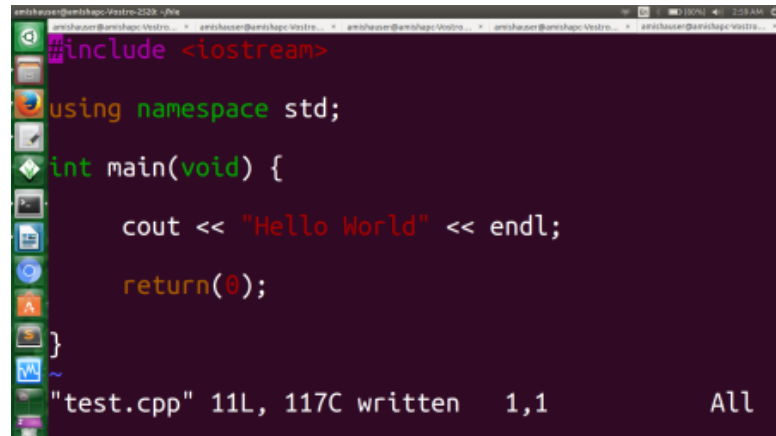
### 4.1.3 CMake

CMake is a language(for generator of build systems) with abstract build rules and gnu make is a dependency resolves that executes programs. Cmake takes information on how to build programs generates makefiles that build the program.

Simple Program

### 4.1.4 Shell Scripting

A shell script is a text file that contains a sequence of commands for a UNIX-based operating system. It's called a shell script because it combines into a "script" in a single file a sequence of commands that would otherwise have to be presented to the system from a keyboard one at a time. The shell is the operating system's command interpreter and the set of commands you use to communicate with the system. A shell script is usually created for command sequences for which a user has a repeated need. You initiate the sequence of commands in the shell script

A screenshot of a code editor window showing the contents of a file named test.cpp. The code is written in C++ and includes the <iostream> header, uses the std namespace, and contains a main function that prints "Hello World" to the console and returns 0. The status bar at the bottom indicates the file is 11 lines long, 117 characters wide, and was written at 1,1. The editor has a dark theme and a sidebar on the left with various icons.

```
#include <iostream>

using namespace std;

int main(void) {

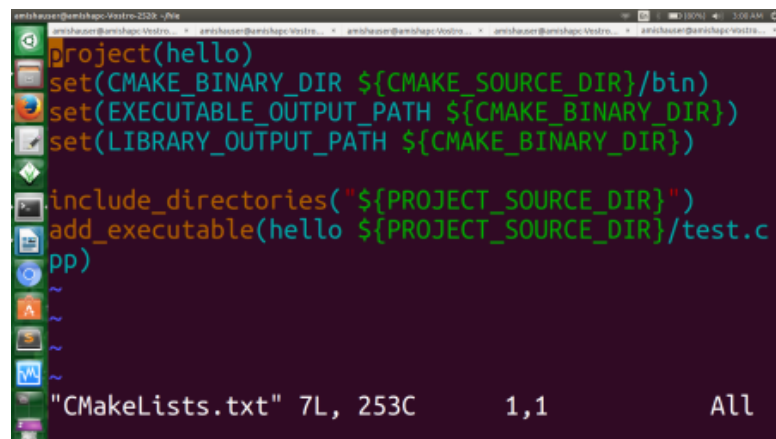
    cout << "Hello World" << endl;

    return(0);

}
```

"test.cpp" 11L, 117C written 1,1 All

Figure 4.5: test.cpp file

A screenshot of a code editor window showing the contents of a file named CMakeLists.txt. The code is written in CMake and defines a project named 'hello', sets the binary directory, and adds an executable named 'hello' from the source file 'test.cpp'. The status bar at the bottom indicates the file is 7 lines long, 253 characters wide, and was written at 1,1. The editor has a dark theme and a sidebar on the left with various icons.

```
project(hello)

set(CMAKE_BINARY_DIR ${CMAKE_SOURCE_DIR}/bin)
set(EXECUTABLE_OUTPUT_PATH ${CMAKE_BINARY_DIR})
set(LIBRARY_OUTPUT_PATH ${CMAKE_BINARY_DIR})

include_directories("${PROJECT_SOURCE_DIR}")
add_executable(hello ${PROJECT_SOURCE_DIR}/test.cpp)
```

"CMakeLists.txt" 7L, 253C 1,1 All

Figure 4.6: cmake file

by simply entering the name of the shell script on a command line.

1. Shell script can take input from user or file and output them on screen.
2. Useful to create our own commands.
3. Save lots of time.
4. To automate the task of day to day life.
5. System Administration part can be also automated.

**Execute your script as syntax:**

```
chmod 755 your-script-name
sh your-script-name
./your-script-name
```



Figure 4.7: Python

### 4.1.5 Python

Python is a dynamic language, as in python coding is very easy and also it require less coding and about its interpreted nature it is just excellent. Python is a high level programming language and Django which is a web development framework is written in python language.

Python is an easy to learn, powerful programming language. Python runs on Windows, Linux/Unix, Mac OS X. Python is free to use, even for commercial products. Python can also be used as an extension language for existing modules and applications that need a programmable interface. Python is free to use, even for commercial products, because of its OSI-approved open source license.

### 4.1.6 Features of Python

- Very clear, readable syntax.
- Strong introspection capabilities.
- Intuitive object orientation.

### 4.1.7 JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

## 4.2 Ubuntu: An open source OS

During my training, I also got familiar with a great and open source Operating System, Ubuntu. Firstly, it was quite difficult for a regular MS Windows user to port to Ubuntu. I did all of



Figure 4.8: Ubuntu

my project work using this vast operating system. Ubuntu is a Debian-based Linux operating system, with Unity as its default desktop environment. It is based on free software and named after the Southern African philosophy of ubuntu (literally, "human-ness"), which often is translated as "humanity towards others" or "the belief in a universal bond of sharing that connects all humanity".

Ubuntu's goal is to be secure "out-of-the box". By default user's programs run with low privileges and cannot corrupt the operating system or other user's files. For increased security, the sudo tool is used to assign temporary privileges for performing administrative tasks, which allows the root account to remain locked and helps prevent inexperienced users from inadvertently making catastrophic system changes or opening security holes.

### 4.3 Introduction To Doxygen



Figure 4.9: Doxygen

Doxygen is a documentation generator, a tool for writing software reference documentation. The documentation is written within code, and is thus relatively easy to keep up to date. Doxygen can cross reference documentation and code, so that the reader of a document can easily refer to the actual code.

Doxygen supports multiple programming languages, especially C++, C, C#, Objective-C, Java, Python, IDL, VHDL, Fortran and PHP.[2] Doxygen is free software, released under the terms of the GNU General Public License.

- Requires very little overhead from the writer of the documentation. Plain text will do, Markdown is support, and for more fancy or structured output HTML tags and/or some of doxygen's special commands can be used.
- Cross platform: Works on Windows and many Unix flavors (including Linux and Mac OS X).
- Comes with a GUI frontend (Doxywizard) to ease editing the options and run doxygen. The GUI is available on Windows, Linux, and Mac OS X.
- Automatically generates class and collaboration diagrams in HTML (as clickable image maps) and L<sup>A</sup>T<sub>E</sub>X (as Encapsulated PostScript images).
- Allows grouping of entities in modules and creating a hierarchy of modules.
- Doxygen can generate a layout which you can use and edit to change the layout of each page.
- Can cope with large projects easily.

### 4.3.1 Installation of Doxygen

Doxygen can be installed using following commands:

```
$ git clone https://github.com/doxygen/doxygen.git
$ cd doxygen
$ ./configure
$ make
```

Documentation of this project.

#### pbOSM

Main Page	Related Pages	Namespaces	Classes	Files
<b>pbOSM Documentation</b>				
<p>You are here because you want beautiful styles of your city "Punjab" in OpenStreetMap.</p> <p>Installation of pbOSM</p> <ol style="list-style-type: none"> <li>1. Clone the repository from <a href="https://github.com/GreatDevelopers/pbOSM.git">https://github.com/GreatDevelopers/pbOSM.git</a></li> <li>2. Give path of style.xml file to render.conf file.</li> <li>3. Browse to the beautiful map.</li> </ol> <p>Style.xml file is using gis database name. If you have database with other name just run the script <a href="#">getxml.py</a>.</p> <p>The script will download style.xml and rename database from gis to your own database. The thing you have to do is rename the database osmpb to your own database.</p>				

Figure 4.10: Main Page of pbOSM

## 4.4 Introduction to L<sup>A</sup>T<sub>E</sub>X

L<sup>A</sup>T<sub>E</sub>X, I had never heard about this term before doing this project, but when I came to know about it's features, found it excellent. L<sup>A</sup>T<sub>E</sub>X is a document markup language and document

# pbOSM

Main Page	Related Pages	Namespaces	Classes	Files
Namespace List	Namespace Members			
<b>generate_road_colours Namespace Reference</b>				
<b>Classes</b>				
class <b>Color</b>				
<b>Functions</b>				
def <b>load_settings</b> ()				
def <b>generate_colours</b> (settings, section)				
def <b>main</b> ()				
<b>Function Documentation</b>				
<pre>def generate_road_colours.generate_colours ( settings,  section  )</pre>				

Figure 4.11: Namespace documentation of pbOSM



Figure 4.12: Donald Knuth, Inventor Of  $\text{\TeX}$  typesetting system

preparation system for the  $\text{\TeX}$  typesetting program. Within the typesetting system, its name is styled as  $\text{\LaTeX}$ .

Within the typesetting system, its name is styled as  $\text{\LaTeX}$ . The term  $\text{\LaTeX}$  refers only to

the language in which documents are written, not to the editor used to write those documents. In order to create a document in  $\text{\LaTeX}$ , a `.tex` file must be created using some form of text editor. While most text editors can be used to create a  $\text{\LaTeX}$  document, a number of editors have been created specifically for working with  $\text{\LaTeX}$ .

$\text{\LaTeX}$  is most widely used by mathematicians, scientists, engineers, philosophers, linguists, economists and other scholars in academia. As a primary or intermediate format, e.g., translating DocBook and other XML-based formats to PDF,  $\text{\LaTeX}$  is used because of the high quality of typesetting achievable by  $\text{\TeX}$ . The typesetting system offers programmable desktop publishing features and extensive facilities for automating most aspects of typesetting and desktop publishing, including numbering and cross-referencing, tables and figures, page layout and bibliographies.

$\text{\LaTeX}$  is intended to provide a high-level language that accesses the power of  $\text{\TeX}$ .  $\text{\LaTeX}$  essentially comprises a collection of  $\text{\TeX}$  macros and a program to process  $\text{\LaTeX}$  documents. Because the  $\text{\TeX}$  formatting commands are very low-level, it is usually much simpler for end-users to use  $\text{\LaTeX}$ .

#### 4.4.1 Typesetting

In preparing a  $\text{\LaTeX}$  document, the author specifies the logical structure using familiar concepts such as chapter, section, table, figure, etc., and lets the  $\text{\LaTeX}$  system worry about the presentation of these structures. It therefore encourages the separation of layout from content while still allowing manual typesetting adjustments where needed.

```
\documentclass[12pt]{article}
\usepackage{amsmath}
\title{\LaTeX}
\date{}
\begin{document}
  \maketitle
  \LaTeX{} is a document preparation system
  for the \TeX{} typesetting program.
\end{document}
```

### 4.5 Introduction to Github

GitHub is a Git repository web-based hosting service which offers all of the functionality of Git as well as adding many of its own features. Unlike Git which is strictly a command-line tool, Github provides a web-based graphical interface and desktop as well as mobile integration. It also provides access control and several collaboration features such as wikis, task management, and bug tracking and feature requests for every project.

GitHub offers both paid plans for private repo handle everything from small to very large projects with speed and efficiency. ositories, and free accounts, which are usually used to host open source software projects. As of 2014, Github reports having over 3.4 million users, making it the largest code host in the world.





Figure 4.13: Output of the above program



Figure 4.14: Github Logo

GitHub has become such a staple amongst the open-source development community that many developers have begun considering it a replacement for a conventional resume and some employers require applications to provide a link to and have an active contributing GitHub account in order to qualify for a job.

The Git feature that really makes it stand apart from nearly every other Source Code Management (SCM) out there is its branching model.

Git allows and encourages you to have multiple local branches that can be entirely independent of each other. The creation, merging, and deletion of those lines of development takes seconds.

This means that you can do things like:

- **Frictionless Context Switching.**  
Create a branch to try out an idea, commit a few times, switch back to where you branched from, apply a patch, switch back to where you are experimenting, and merge it in.
- **Role-Based Code lines.**  
Have a branch that always contains only what goes to production, another that you merge work into for testing, and several smaller ones for day to day work.

- Feature Based Work flow.  
Create new branches for each new feature you're working on so you can seamlessly switch back and forth between them, then delete each branch when that feature gets merged into your main line.
- Disposable Experimentation.  
Create a branch to experiment in, realize it's not going to work, and just delete it - abandoning the work with nobody else ever seeing it (even if you've pushed other branches in the meantime).

Notably, when you push to a remote repository, you do not have to push all of your branches. You can choose to share just one of your branches, a few of them, or all of them. This tends to free people to try new ideas without worrying about having to plan how and when they are going to merge it in or share it with others.

There are ways to accomplish some of this with other systems, but the work involved is much more difficult and error-prone. Git makes this process incredibly easy and it changes the way most developers work when they learn it.

### 4.5.1 What is Git?



Figure 4.15: Git Logo

Git is a distributed revision control and source code management (SCM) system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows. Git was initially designed and developed by Linus Torvalds for Linux kernel development in 2005, and has since become the most widely adopted version control system for software development.

As with most other distributed revision control systems, and unlike most clientserver systems, every Git working directory is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server. Like the Linux kernel, Git is free and open source software distributed under the terms of the GNU General Public License version 2 to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

### 4.5.2 Installation of Git

Installation of git is a very easy process. The current git version is: 2.0.4. Type the commands in the terminal:

```
$ sudo apt-get update
```

```
$ sudo apt-get install git
```

This will install the git on your pc or laptop.

### 4.5.3 Various Git Commands

Git is the open source distributed version control system that facilitates GitHub activities on your laptop or desktop. The commonly used Git command line instructions are:-

#### 4.5.3.1 Create Repositories

Start a new repository or obtain from an exiting URL

```
$ git init [ project-name ]
```

Creates a new local repository with the specified name

```
$ git clone [url ]
```

Downloads a project and its entire version history

#### 4.5.3.2 Make Changes

Review edits and craft a commit transaction

```
$ git status
```

Lists all new or modified files to be committed

```
$ git diff
```

Shows file differences not yet staged

```
$ git add [file ]
```

Snapshots the file in preparation for versioning

```
$ git commit -m "[descriptive message ]"
```

Records file snapshots permanently in version history

### 4.5.3.3 Group Changes

Name a series of commits and combine completed efforts

#### **\$ git branch**

Lists all local branches in the current repository

#### **\$ git branch [branch-name ]**

Creates a new branch

#### **\$ git checkout [branch-name ]**

Switches to the specified branch and updates the working directory

#### **\$ git branch -d [branch-name ]**

Deletes the specified branch

### 4.5.3.4 Synchronize Changes

Register a repository bookmark and exchange version history

#### **\$ git fetch [bookmark ]**

Downloads all history from the repository bookmark

#### **\$ git merge [bookmark /[branch]]**

Combines bookmarks branch into current local branch

#### **\$ git push [alias [branch]]**

Uploads all local branch commits to GitHub

#### **\$ git pull**

Downloads bookmark history and incorporates changes

## 4.6 Introduction to Reveal-js & Reveal-md

# REVEAL MD

Figure 4.16: MD & JS

Reveal-js is one of the framework of Javascript. This can be used for presentations purpose. Now before going to reveal-md lets talk about some fundamental things.

What is a Markup language?

Markup languages are designed for the processing, definition and presentation of text. The language specifies code for formatting, both the layout and style, within a text file. HTML and Markdown is an example of a widely known and used markup language.

Markdown is a lightweight Markup Language with simple plain text formatting syntax designed so that it can be converted to HTML and many other formats. It is created by John Gruber. It had '.md' or '.markdown' extension.

"Markdown is a text-to-HTML conversion software tool written in Perl for web writers."

Moreover, to enable markdown feature of reveal.js, we need reveal-md. The Markdown feature of reveal.js is awesome, and has an easy (and configurable) syntax to separate slides. Use three dashes surrounded by two blank lines.

### 4.6.1 Installation of reveal-md

Installation of reveal-md is a very easy process. Type the commands in the terminal:

```
$ sudo apt-get install npm
```

```
$ sudo apt-get install nodejs-legacy
```

```
$ sudo npm install -g reveal-md
```

This will install reveal-md on your pc or laptop.

## 4.7 Working with Experimental Server



Figure 4.17: Server Communication

I had also done the whole project on ubuntu experimental server and had also learnt about making your system a server.

What is a Remote Server?

In simple words its nothing much but a Computer that is not attached to a users keyboard but over which he or she has some degree of control (like can see data of that computer, can retrieve or send data etc.)

For going deep you need to know about ssh (Secure Shell). I had written about it in my old blogs. You can Google it too.

I had done it using SSH. There are few terms related to this :

- SSH: It is a Secure Socket Shell, is a network protocol that provides administrators with a secure way to access a remote computer.
- MOSH: It is a software tool used to connect from a client computer to a server over the Internet, to run a remote terminal.
- Tmux: tmux is basically a terminal multiplexer. It is used so that within one terminal window we can open multiple windows and split-views.
- OpenSSH: It is a freely available version of the Secure Shell (SSH) protocol family of tools for remotely controlling or transferring files between computers. Traditional tools used to accomplish this is telnet which is not much secure.

In Unix, you can use SCP (the scp command) to securely copy files and directories between remote hosts without starting an FTP session or logging into the remote systems explicitly. The scp command uses SSH to transfer data, so it requires a password.

Some of the useful commands in this for checking errors or for other purposes are:

- ll: This command is used to list the detail information of files and folder of a current directory.
- tail -f error.log: This is used for checking errors.
- sudo apt-get install openssh-server
- sudo vim /etc/ssh/sshdconfig  
(To edit this as per your preferences. But first take a backup of this file for later default configurations if needed.)
- sudo restart ssh  
(To check your ssh daemon is running or not.)
- ssh user@hostip  
(To enter into a remote server from some other system. )

## 4.8 Python Django Framework



Figure 4.18: Python Django Framework

Django is a web development framework that assists in building and maintaining quality web applications. Django helps eliminate repetitive tasks making the development process an

easy and time saving experience. It is a Python web framework.

During the training period it's being used in making LuvLdh-Webapp. An app used for publish post on a facebook page using a Graph Facebook API with Python SDK-Development Kit.

Librehatti, an Enterprise Resource Planning(ERP) software is wholly build in Django with the vast extensive knowledge, datastructures. It is completely open source available in Github. During training, it is being used to make a module Dispatch\_register and made GST enabled module.

## 4.9 OSM and its Components

It is a collaborative project to create a free editable map of the world. Before beginning to make your own tile server, review some terminologies.



Figure 4.19: OpenStreetMap

### 4.9.1 Benifits Of Own Tile Server

OSM map is accessible even when internet provider is down or when the power is off or both. It won't take much for to see the benefit of having your own piece of OpenStreetMap infrastructure.

Now it's turn to install, setup and configure all the necessary software to operate own tile server. All the instructions are illustrated in blog "<https://amisha2016.wordpress.com>"

These instructions build what OpenStreetMap calls a "tile server". That is, a computer that uses the OSM data set to create map images that are suitable for a website. Not every OpenStreetMap function is supported, but you will be able to create a local map, keep it up to date and customize it for your own purposes.

### 4.9.2 Postgresql / postgis

PostGIS is a spatial database extender for PostgreSQL object-relational database. It adds support for geographic objects allowing location queries to be run in SQL.

Most spatial databases allow representing simple geometric objects such as points, lines and polygons. Some spatial databases handle more complex structures such as 3D objects, topological coverages, linear networks, and TINs.



Figure 4.20: Postgresql

On Ubuntu there are pre-packaged versions of both `postgis` and `postgresql`, so these can simply be installed via the Ubuntu package manager.

### 4.9.3 Osm2pgsql

`osm2pgsql` is under active development and is best compiled from source.

`osm2pgsql` is a command-line based program that converts OpenStreetMap data to postGIS-enabled PostgreSQL databases.

Mapnik is an open source mapping toolkit for desktop- and server-based map rendering, written in C++.

One of its many users is the OpenStreetMap project (OSM), which uses it in combination with an Apache Web Server module (`mod_tile`) to render tiles that make up the OSM Slippy Map Layer.



Figure 4.21: Openstreetmap-carto Style

### 4.9.4 Openstreetmap-carto

Openstreetmap-carto is a sensible starting point for quickly making beautiful maps based on an OpenStreetMap database. It is written in the Carto styling language and can be opened as



a project in TileMill.

The style is still a work in progress and you are encouraged to use the issue tracker to note missing features or problems with the current implementation.

### 4.9.5 OpenLayer.js

OpenLayers makes it easy to put a dynamic map in any web page. It can display map tiles, vector data and markers loaded from any source. OpenLayers has been developed to further the use of geographic information of all kinds. It is completely free.

### 4.9.6 Geographic Information Systems (GIS)

GIS is a computer system for capturing, storing, checking, and displaying data related to positions on Earth's surface. GIS can show many different kinds of data on one map. This enables people to more easily see, analyze, and understand patterns and relationships.

The Global Positioning System (GPS) is a space-based navigation system that provides location and time information in all weather conditions, anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites.

## 4.10 Implementation

Development of OSM started with development in phases which focus on particular need of project. Various phases and their detail are given below :-

- Phase I (Setup OSM Server) :-  
During Phase I, install all the dependencies(components) as mentioned above to make your own osm tile sever. After installing the softwares download the map in pbf(may be osm) format and render your own tile server. You can see your map on the browser after moving to the location which is being downloaded.
- Phase II (Styling of Map) :-  
During phase II, there is a lot of customization as listed below:-
  - Colors of the buildings, roads, primary lines, secondary lines etc have been customized and then re-render the map to view the changes in the map.
  - Added international boundary of India with customizable color and pixels.
  - Modified the icons of the nodes.
  - Customize the language of the map by applying Algorithm- if Punjabi name is provided then first priority goes to it followed by hindi and then English.
  - Admin levels at different zoom levels with different colors and with a names displayed over each boundary area.
- Phase III (Increased zoom levels to 28 for indoor mapping) :-  
The purpose of phase III was to increase the zoom levels to more than 19 so to create the

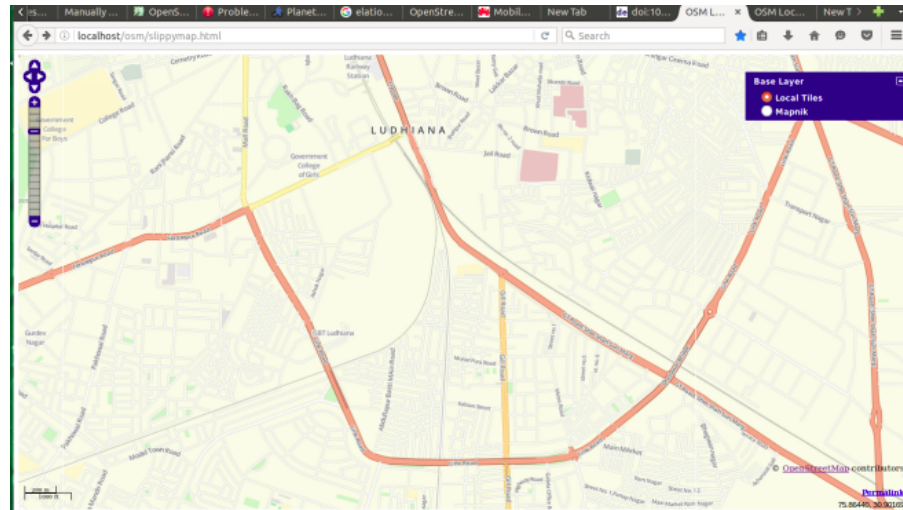


Figure 4.22: OSM Map on Web browser

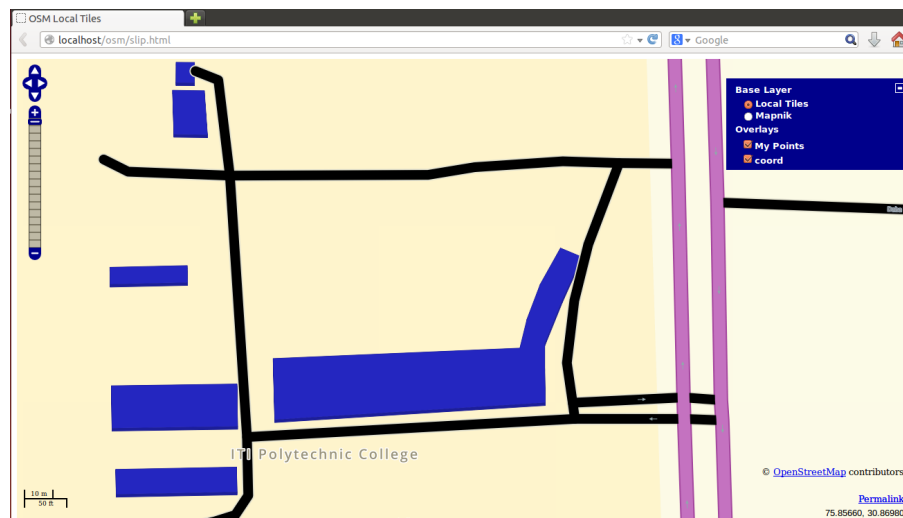


Figure 4.23: Building with customize color

space for indoor mapping. It is done in mod\_tile module by applying another different algorithm.

- Phase IV (User Input Map) :-  
During phase III, we made the html and php pages in which user can input latitude, longitude and zoom level of his own choice and if the tile image of that location is downloaded then on one click the map of that particular location will be visible. The functioning is done with the help of Javascript.
- Phase V (Configurable OSM Installation Script) :-  
Building your system as an OSM server, is standalone task that can acquire atleast 15 days for beginner. So the purpose for phase IV is to make a script as easy Wordpress Installation. It is the shell non- interactive, one time configurable script (user have to change hardly two three parameters inside it) at the initial stage and then can run the script and can go for a cup of coffee. The source code of the script is placed under the repository [https://github.com/amisha2016/osm\\_installation\\_script](https://github.com/amisha2016/osm_installation_script). After running script

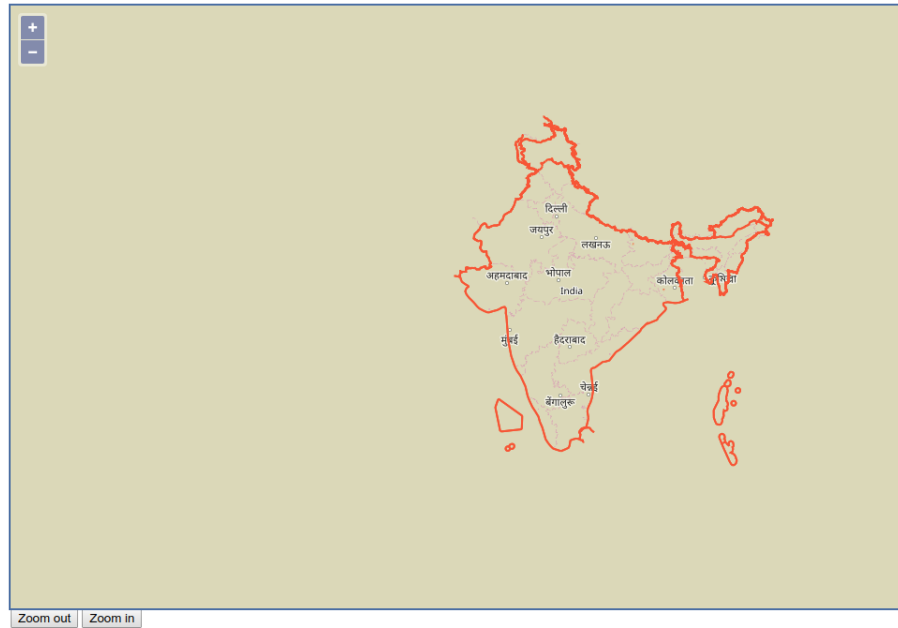


Figure 4.24: International Boundary of India

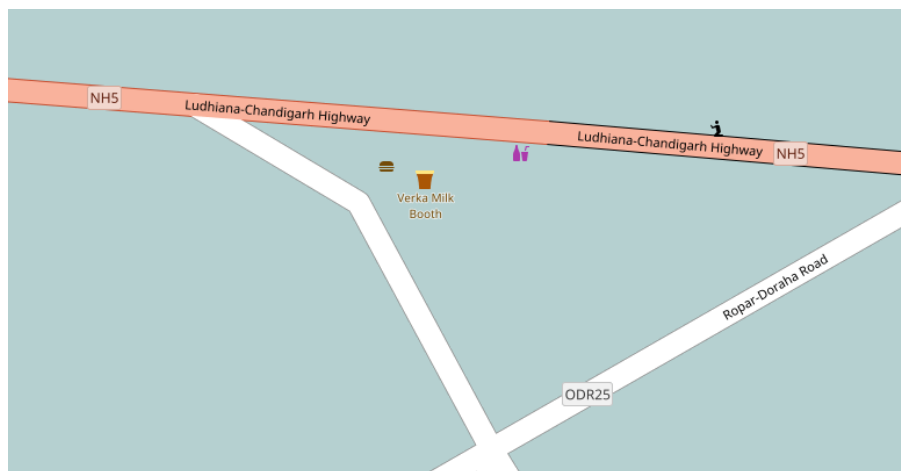


Figure 4.25: Customized icon representing Pub shop

you will be able to browse to [http://your-server-address/osm\\_tiles/0/0/0.png](http://your-server-address/osm_tiles/0/0/0.png) as shown below.

- Phase VI (Event Handling) :-  
The next task was to control the movement of the osm map through the arrow keys of keyboard. Again it is done with the help of Javascript with the concept of event handling. Various formulas are being applied and testing have been done while doing it. The code for the same is on the experimental server. Now, the map can be controlled through arrow keys also. Isn't it amazing?
- Phase VII (3-D View of the Map) :-  
The main motive of Phase VII was to make map more realistic. So, by creating the 3-D View of the map it has achieved the greater milestone. It represents OSM Buildings with multiple storys, shadows of the building, sun, sky and many more. Most of these features



Figure 4.26: Map of Punjab in Punjabi

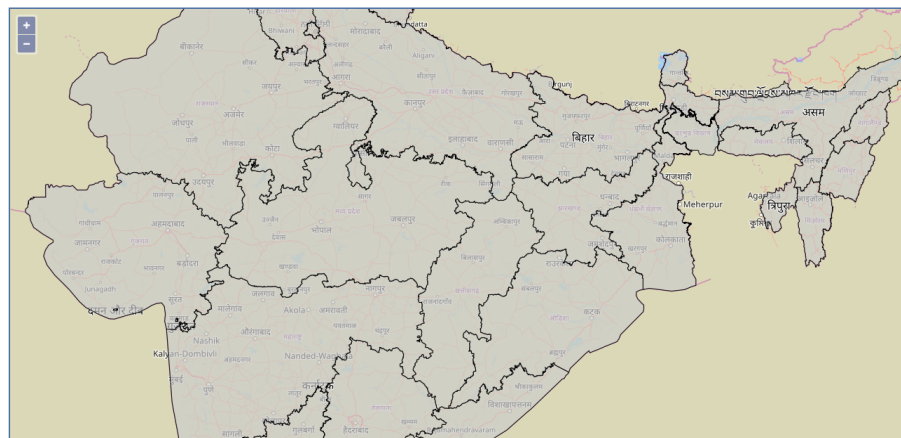


Figure 4.27: India divided into states with different color

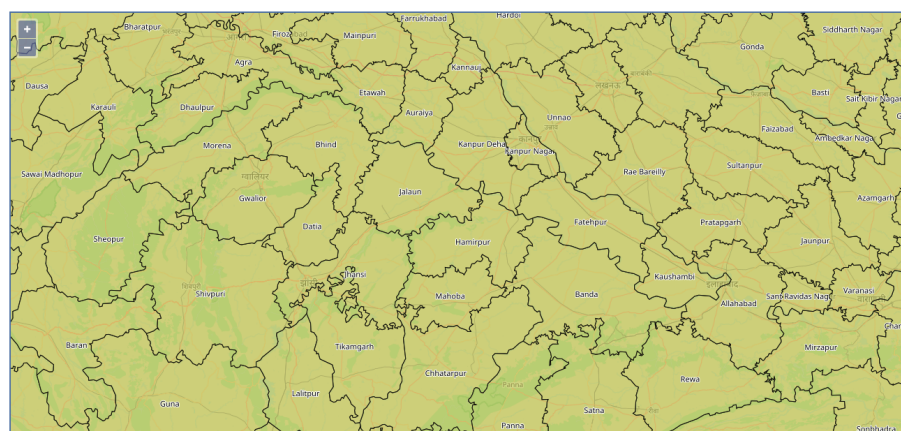


Figure 4.28: India divided into districts with different color

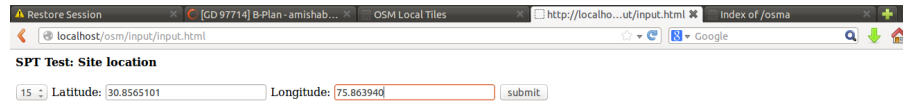


Figure 4.29: Increased zoom level for indoor mapping

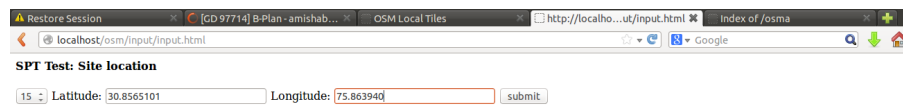


Figure 4.30: User Input Page

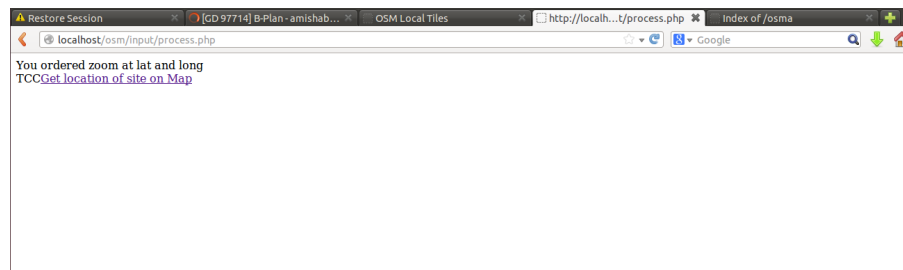


Figure 4.31: Php Page

are implemented with JOSM tool like creating 3-D tank. The best part of the map is we can tilt and rotate to see a beautiful look.

- Phase VIII (Searching the map, Animations, GNE Tour, Popup Menu and Icons) -: During phase VIII, frontend of the map is improved by adding various features like search button, animations like rotate clockwise, spinning etc. A very beautiful tour of GNE college is represented and the highlighted placed represented by icons. On clicking the map, latitude and longitude is popup.
- Phase X (Map On Remote Server) -: All the tasks were achieved on local server first. To make it for others to use for people like you everything is implemented on remote server i.e <https://lab.gdy.club>. Moreover, it has been kept for back up purpose also so that the code and project retains on other system also.

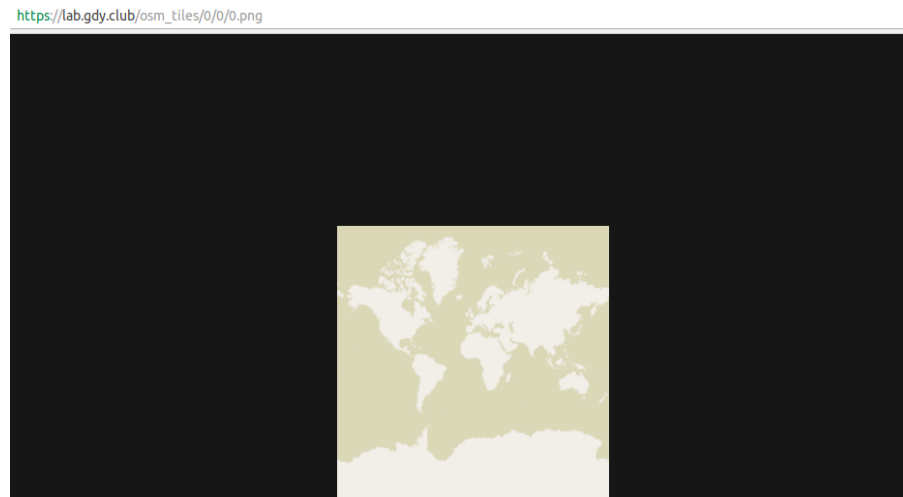


Figure 4.32: Verifying system to be OSM server using script

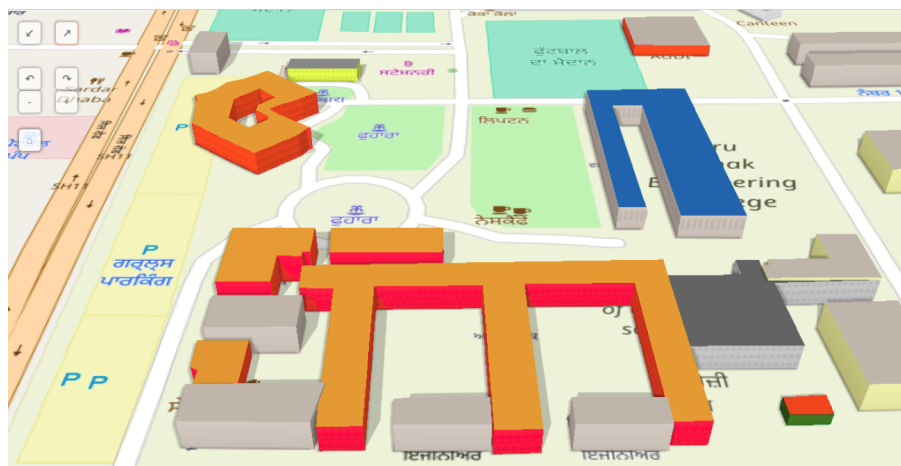


Figure 4.33: Representing 3-D view of GNE

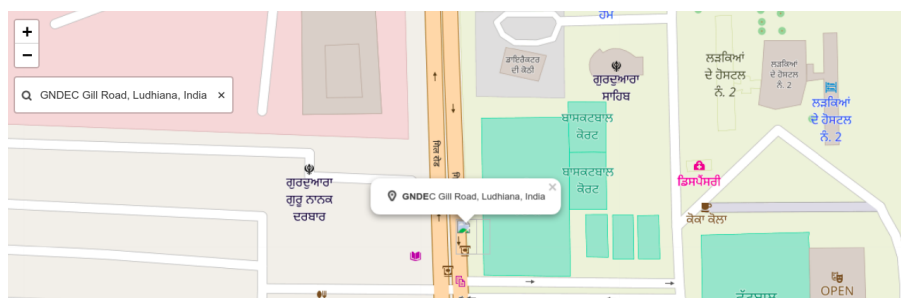


Figure 4.34: Searched place and pop up menu

- Phase XI (Documentation) :-

During final phase, the documentation of the project( developers documentation and README.md) has been made using doxygen and wrote the report in latex for this software.

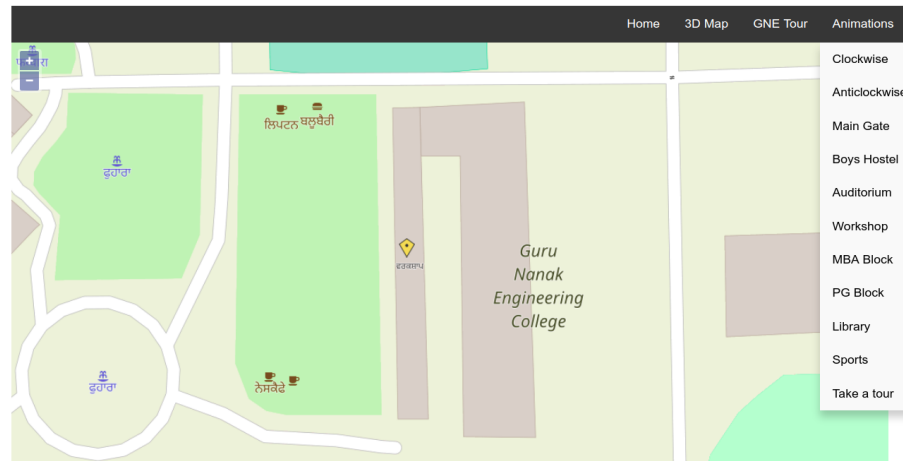


Figure 4.35: Representing Animations

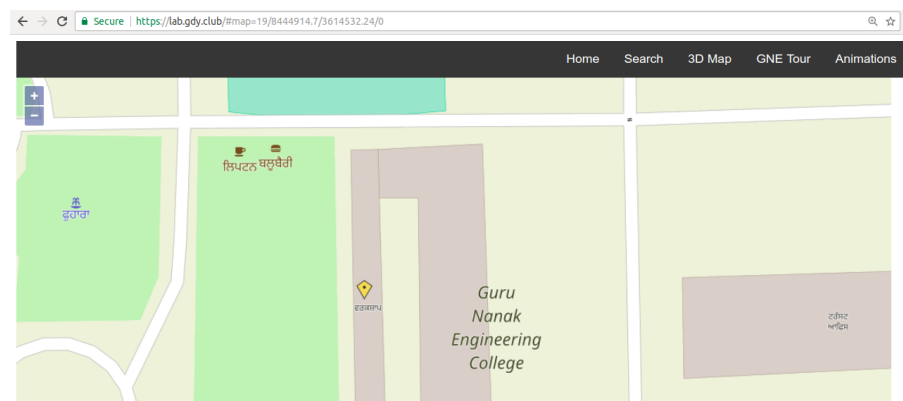


Figure 4.36: Map on remote server

## 4.11 Testing

Testing a program consists of providing the program with a set of test inputs (or test cases) and observing if the program behaves as expected. If the program fails to behave as expected, then the conditions under which failure occurs are noted for later debugging and correction.

This project had been taken through rigorous test to fully found potential causes of error and system failure and full focus have been given to cover all possible exceptions that can occur and cause failure of the software.

As this project is based on intensive background process it have been taken care that if correct input are given then processing of user job can even continue or a least automatically restart even after server shuts down or even crash.

If user don't enter any input then it by default takes latitude and longitude 0 respectively and zom is also set with default value 15. We can default value also.

When we click on a link "Permalink" the current latitude and longitude become a default. On refreshing we get the same location.

Table 4.1: Unit testing

Test Case ID	Test Case Name	Description	Pre-Condition	Execution Step	Expected Result	Status
1	OSM tile server Test	OSM installation script run by the user will make the system OSM server.	1. Proper network connection . 2. Minimum RAM space required is 2GB.	Execute the script file install_osm.sh.	If localhost output map of the world then OSM server is installed.	passed
2	Database Test	1.Dumping the raw data to database.	1. Proper database administration rights must be awarded. 2. The storage space must be sufficient.	1. Give the correct path of the raw data file. 2. All core processers should be free for easy dump.	It would result in building 9-tables and time elapsed for completion of command.	passed
3	Mod_tile Test	Mod_tile Installation server for slippymap to render.	Proper internet connection.	Compile the mod_tile.	In case localhost/mod_tiles outputs then mod_tile server is installed.	passed



Table 4.2: Integration testing

Test Case ID	Test Case Name	Description	Pre-Condition	Execution Step	Expected Result	Status
1	Searching Places Test	Searching the name of place from database.	1. Database dump with tag as name . 2. Proper Internet connection.	Write name of place in search button.	Map move to the required place.	passed
2	Language ordering Test	Language of the map should be in order Punjabi followed by Hindi then English.	Carto style sheet should be correctly compiled.	Open slip-pymap in browser.	Map with customize language in a defined order.	passed

## 5.1 Conclusion

The core of OpenStreetMap is a collection of map data which can be used for many different purposes. Embedding and using different technology in one software. Everybody can use OpenStreetMap's data with very few restrictions, many different third party services are built on OpenStreetMap's data. How to work like in group for development of software and how to apply juggaar(innovated) in softwares to get problem solved.

A great deal of things are learned while working on this project. The learning was not limited to project only but the whole experience of working as a trainee under Dr. H.S. Rai at TCC, developer at OpenSCAD and also as a mentor for Google Code In was immensely educational. Working with the Open Source Community and a variety of people of different age group, one is always challenged by the fundamental difference between classroom coaching and real World experience. But such a challenge is exactly the purpose of six months training.

The whole experience of working on this project and contributing to a few others has been very rewarding as it has given great opportunities to learn new things and get a firmer grasp on already known technologies.

## 5.2 Future Scope

OpenStreetMap being an open source project and supported by a large open Source community have a lot of scope for future improvements and additions as other individuals can also contribute in it and add additional functionality. One of the examples is my project only customizer for OpenStreetMap.

Being an Open Source project there is a constant flow of suggestion and demands by people for additional functionality and improvement in existing features. Here is a small list of the Features that would be added in near future.

1. **Adding support for Indoor Mapping:** It was decided much at beginning of the project that this feature will be ported using a special syntax then relying on the comment based syntax. The discussion has been going on related to deciding the syntax for the customizer.
2. **Option to Change Languages:** It has been proposed that there should be a feature to switch to different languages through Javascript.
3. **Option to import BIM model in JOSM:** This feature will help people especially Civil Engineers to import BIM models in JOSM and customize directly their.
4. **Styling the map:** There is more need to style a map like adding icy effect near glaciers, show particular tags at particular levels.
5. **Improving GUI part:** There have been some suggestions related to improving the GUI of the Customizer.

- [1] J. Levine, Flex & bison, 1st ed. Sebastopol, Calif.: O'Reilly Media, 2009.
- [2] C. Donnelly and R. Stallman, CartoCSS, 1st ed. Corto, MA: Free Software Foundation, 1999.
- [3] "User:amisha2016 - OpenStreetMap", openstreetmap.org, 2016. [Online]. Available: <https://www.openstreetmap.org/user/amisha2016>. [Accessed: 27- Nov- 2017].
- [4] "Customizing OSM Map", Amisha Budhiraja, 2016. [Online]. Available: <https://amisha2016.wordpress.com/2016/11/13/customizing-osm-map/>. [Accessed: 27- Nov- 2017].
- [5] "OpenStreetMap User Manual - Wikibooks, open books for an open world", En.wikibooks.org, 2016. [Online]. Available: [http://wiki.openstreetmap.org/wiki/Beginners\\_Guide\\_1.6](http://wiki.openstreetmap.org/wiki/Beginners_Guide_1.6). [Accessed: 27- Nov- 2017].
- [6] "User: Amisha Budhiraja pbOSM", GitHub, 2017. [Online]. Available: <https://github.com/amisha2016/pbOSM>. [Accessed: 27- Nov- 2017].
- [7] "3D OSM Buildings with leaflet", Amisha Budhiraja, 2017. [Online]. Available: <https://amisha2016.wordpress.com/2017/07/11/integration-of-3d-osmbuilding-with-leaflet-and-openlayers/>. [Accessed: 27- Nov- 2017].
- [8] "openstreetmap", GitHub, 2017. [Online]. Available: <https://github.com/openstreetmap>. [Accessed: 27- Nov- 2017].
- [9] "Doxygen: Main Page", Doxygen.org, 2016. [Online]. Available: <http://www.doxygen.org>. [Accessed: 27- Nov- 2017].
- [10] <http://www.openstreetmap.org/>
- [11] "Increased zoom level to 28 in OSM", Amisha Budhiraja, 2017. [Online]. Available: <https://amisha2016.wordpress.com/2017/07/24/inreased-zoom-level-to-28-in-osm/>. [Accessed: 27- Nov- 2017].
- [12] "OpenStreetmap - Blogs", openstreetmap.org, 2016. [Online]. Available: <https://blogs.openstreetmap.org/>. [Accessed: 27- Nov- 2017].
- [13] "JOSM (software)", En.wikipedia.org. [Online]. Available: <http://wiki.openstreetmap.org/wiki/JOSM>. [Accessed: 27- Nov- 2017].
- [14] "Playing with Languages in OSM", Amisha Budhiraja, 2017. [Online]. Available: <https://amisha2016.wordpress.com/2017/07/03/playing-with-languages-in-osm/>. [Accessed: 27- Nov- 2017].
- [15] "Travis CI", En.wikipedia.org, 2017. [Online]. Available: [https://en.wikipedia.org/wiki/Travis\\_CI](https://en.wikipedia.org/wiki/Travis_CI). [Accessed: 27- Nov- 2017].
- [16] "CMake/Testing With CTest - KitwarePublic", Cmake.org, 2017. [Online]. Available: [https://cmake.org/Wiki/CMake/Testing\\_With\\_CTest](https://cmake.org/Wiki/CMake/Testing_With_CTest). [Accessed: 27- Nov- 2017].