

GEOSPATIAL APPLICATION DEVELOPMENT FOR PRIMARY FIELD DATA

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

**BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE &
ENGINEERING**

BY

Tanishq Dayma

EN16CS301276

Under the Guidance of
Prof. (Dr.) Om P. Damani
Prof. Sachin Solanki



Department of Computer Science & Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

MAY, 2020

GEOSPATIAL APPLICATION DEVELOPMENT FOR PRIMARY FIELD DATA

A Project-II Report

Submitted in partial fulfillment of requirement of the

Degree of

**BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE &
ENGINEERING**

BY

Tanishq Dayma
EN16CS301276

Under the Guidance of
Prof. (Dr.) Om P. Damani
Prof. Sachin Solanki



Department of Computer Science & Engineering
Faculty of Engineering
MEDI-CAPS UNIVERSITY, INDORE- 453331

MAY, 2020

Report Approval

The project work “**Geospatial Application Development for primary field data**” is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approved any statement made, opinion expressed, or conclusion drawn there in; but approve the “Project Report” only for the purpose for which it has been submitted.

Internal Examiner

Name:

Designation

Affiliation

External Examiner

Name:

Designation

Affiliation

Declaration

I hereby declare that the project entitled **“Geospatial Application Development for primary field data”** submitted in partial fulfilment for the award of the degree of Bachelor of Technology in ‘Computer Science and Engineering’ completed under the supervision of **Prof. Sachin Solanki, Department of Computer Science and Engineering**, Faculty of Engineering, Medi-Caps University, Indore and **Prof. (Dr.) Om P. Damani, Department of Computer Science and Engineering**, IIT Bombay, is an authentic work.

Further, I declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

Tanishq Dayma

EN16CS301276

Certificate

We, **Om P. Damani and Sachin Solanki** certify that the project entitled **Geospatial application development for primary field data** submitted in partial fulfillment for the award of the degree of Bachelor of Technology by **Tanishq Dayma** is the record carried out by her under my guidance and that the work has not formed the basis of award of any other degree elsewhere.

Prof. Sachin Solanki
Computer Science and Engineering
Medi-Caps University, Indore

Prof. (Dr.) Om P. Damani
Computer Science and Engineering
Indian Institute of Technology,
Bombay

Dr. Suresh Jain
Head of the Department
Computer Science & Engineering
Medi-Caps University, Indore

Acknowledgements

I would like to express my deepest gratitude to Honorable Chancellor, **Shri R C Mittal**, who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) Sunil K Somani**, Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) D K Panda**, Dean, Faculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Dr. Suresh Jain** for his continuous encouragement for betterment of the project.

I express my gratitude towards Prof. Milind Atrey, Dean (R&D), IIT Bombay for providing me with this golden opportunity of being a part of the Research Internship Awards 2019-20 at IIT Bombay. I would like to thank Prof. Harsh C Phuleria and Prof. Sunita Srivastava, IRCC Internship Coordinators at IIT Bombay for their continuous support throughout the internship.

I express my heartfelt gratitude to my External Guide and Principal Investigator, **Prof. (Dr.) Om P. Damani, Department of Computer Science and Engineering, Indian Institute of Technology Bombay** without whose continuous guidance and support the completion of this project would not have been possible. I would also like to thank to my team at IIT Bombay, Mr. Vishal Mishra, Project Manager at TDSC and Mr. Animesh Nautiyal, Project Research Assistant, TDSC who extended their kind support and help towards the completion of this project.

I would also like to extend my gratitude to our project coordinator **Prof. Sachin Solanki** who extended his support and necessary inputs important for successful submission of this project.

Tanishq Dayma (EN16CS301276)

B.Tech. IV Year

Department of Computer Science & Engineering

Faculty of Engineering

Medi-Caps University, Indore

Abstract

During this period of my internship at IIT Bombay, I have learned many technologies and worked on multiple projects. I joined on 3rd January 2020 as a research intern in Technology and Development Solutions Cell (TDSC), CTARA, IIT Bombay. It is a resource center for providing technology-based solutions for the development sector. TDSC provides resources and services to undertake projects that deliver outputs via academic research, consulting, outreach modules, entrepreneurship guidance, etc.

I worked on two primary projects for the organization:

1. First one was to work for their geospatial interface called CommunityGIS by developing it for the projects carried out in TDSC.
2. Second one was to develop a website for Unnat Maharashtra Abhiyan (UMA), a project of the Ministry of Higher and Technical Education, at IIT Bombay and Technology and Development Solutions Cell (TDSC).

CommunityGIS is a platform for spatial data infrastructure. Primary objective of it is to provide educational platform for learners to learn about spatial information and learn how these modern tools can be used to solve real life problems. As a platform it is for people who range from those who can contribute data (mobile based point data collection to digitization of published maps) to those who can analyze the data to infer useful social knowledge. It is a digital way to collect, aggregate, access, analyze data and then provide decision support to the community.

UMA has a vision to build an independent and public knowledge infrastructure for the state of Maharashtra which will bring socio-economic and cultural development for its people, especially those in the bottom 80% of the socio-economic strata.

The aim is to create map infrastructure for the various projects of TDSC like Thakkar Bappa Yojana, Rural Water Supply Scheme, Deonadi project etc. The web interface would help in viewing, editing, updating and analysing the projects by the concerned people. Modules for querying, analysing and presenting the data could also be included in the objectives. The data is to be presented on the site so that it can be easily understood and used by the various organisations and authorities or researchers planning to work on development in rural areas.

Keywords: CommunityGIS, Academic Research, Spatial Data Infrastructure, Socio-economic Strata, Web interface.

Table of Contents

		Page No.
	Report Approval	ii
	Declaration	iii
	Certificate	iv
	Acknowledgement	v
	Abstract	vi
	Table of Contents	vii
	List of figures	ix
	Abbreviations	x
Chapter 1	Introduction	
	1.1 Introduction	1
	1.2 Objectives	2
	1.3 Significance	2
	1.4 Research Design	3
	1.5 Source of Data	4
Chapter 2	System Requirement Analysis	
	2.1 Information Gathering	5
	2.2 System Feasibility	6
	2.2.1 Economical	6
	2.2.2 Technical	6
	2.2.3 Behavioral	7
	2.3 Platform Specification (Development & Deployment)	7
	2.3.1 Hardware	7
	2.3.2 Software Implementation Language/ Technology	7
Chapter 3	System Analysis	
	3.1 Information Flow Representation	8
	3.1.1 Entity-Relationship Diagram	9
	3.1.2 Use Case Diagram	10
Chapter 4	Design	
	4.1 Architectural Design	11
	4.1.1 Architectural Context Diagram	12
	4.1.2 Architectural Behavioral Diagram	12
	4.1.3 Description of Architectural Diagram	13
	4.1.4 Control Hierarchy	14
	4.2 Procedural/Modular Approach	15
	4.2.1 Software Used	15
	4.2.2 Internal Data Structures	19
	4.2.3 Coding	20

	4.3 Data Design	22
	4.3.1 Data objects and Resultant Data Structures	22
	4.4 Interface Design	23
	4.4.1 Human-Machine Interface Design Specification	23
	4.4.2 I/O Forms	25
	4.5 Reports	26
Chapter 5	Testing	
	5.1 Testing Objective	27
	5.2 Testing Scope	28
	5.3 Testing Principles	28
	5.4 Testing Methods Used	30
	5.5 Test Cases	31
	5.6 Sample Test Data & Results	31
Chapter 6	Results	35
Chapter 7	Summary and Conclusions	40
Chapter 8	Future Scope	41
Chapter 9	Bibliography and References	42

List of Figures

Figure No.	Figure Name	Page No.
1.1	Components of GIS	3
3.1	E-R Diagram of Geospatial Application	9
3.2	Use Case Diagram of Geospatial Application	10
4.1	Use Case Diagram of Geospatial Application	13
4.2	Control Hierarchy of Geospatial Application	14
4.3	GeoServer	16
4.4	Shapefiles having 6 files	16
4.5	Django Architecture	18
4.6	Domain and baseURL for the layer	18
4.7	Attributes of the layer	19
4.8	urls.py	20
4.9	views.py	20
4.10	models.py	21
4.11	Fetch Layer Function	21
4.12	Displaying Lines Function	22
4.13	CommunityGIS home page	23
4.14	Different themes of the interface	24
4.15	Different projects of TDSC	24
4.16	Login Form	25
4.17	Filters or the form to display the layer	25
4.18	Django Administration	26
5.1	Output as per input	32
5.2	Point describing the associated attributes	33
5.3	GIS mapping for the Vada Taluka and its associated attribute table	34
6.1	New interface of TDSC CommunityGIS for the project of Thakkar Bappa Yojana	35
6.2	Interface of TDSC CommunityGIS for the project of Rural Water Supply Scheme	36
6.3	UMA website home page	36
6.4	Admin interface of Wagtail	37
6.5	Participating Institutes under UMA	37
6.6	Home Page of TDSC	38
6.7	About Us Page of TDSC	38
6.8	Services Page of TDSC	39

Abbreviations

Abbreviation	Description
TDSC	Technology and Development Solution Cell
UMA	Unnat Maharashtra Abhiyan
GIS	Geospatial Information System
PI	Participating Institutes
CTARA	Centre for Technology Alternatives for Rural Areas
IFD	Information Flow Diagram
UML	Unified Modelling Language
ADL	Architectural Design Language
BRS	Business Requirement Specification
SRS	System Requirement Specification
CSV	Comma Separated Value
KML	Keyhole Markup Language
XML	Extensible Markup Language

Chapter 1

Introduction

1.1 Introduction

A geographic information system (GIS) is a system designed to capture, store, manipulate, analyze, manage, and present spatial or geographic data. GIS applications are tools that allow users to create interactive queries (user-created searches), analyze spatial information, edit data in maps, and present the results of all these operations. GIS sometimes refers to geographic information science (GIScience), the science underlying geographic concepts, applications, and systems. Since the mid-1980s, geographic information systems have become valuable tool used to support a variety of city and regional planning functions.[1]

GIS can refer to a number of different technologies, processes, techniques and methods. It is attached to many operations and has many applications related to engineering, planning, management, transport/logistics, insurance, telecommunications, and business. For that reason, GIS and location intelligence applications can be the foundation for many location-enabled services that rely on analysis and visualization.

GIS can relate unrelated information by using location as the key index variable. Locations or extents in the Earth space–time may be recorded as dates/times of occurrence, and x, y, and z coordinates representing, longitude, latitude, and elevation, respectively. All Earth-based spatial–temporal location and extent references should be relatable to one another and ultimately to a "real" physical location or extent. This key characteristic of GIS has begun to open new avenues of scientific inquiry.

Technology and Development Solutions Cell (TDSC) [2] operates as a CTARA (Centre for Technology Alternatives for Rural areas) project under IIT Bombay's Industrial Research and Consulting Centre (IRCC). TDSC does neutral third-party audits of the projects, some of which are sanctioned by the Government of Maharashtra. So, they require a system to store all their geospatial data on a centralised server that can be used to handle all the data addition and updating. This server must also have a provision to get integrated with the QGIS software, currently being utilised by the TDSC members.

Another thing was to develop a Geospatial application (CommunityGIS) that will create significant improvements in the efficiency of research work at TDSC as well as that of any other current and future research groups and members of the community. Along with the work in the geographic information system (GIS), I have been allotted work to design two websites, one was for Unnat Maharashtra Abhiyan (UMA), which is a project of the Ministry of Higher and Technical Education and other one is for TDSC. The website for UMA is hosted at (<http://104.211.185.20/>) and the TDSC website is currently under development.

1.2 Objectives

Basically, GIS system as software tools and techniques provide users with a platform where they can visualize the facilities provided by government in a more intuitive way. With this, objective of the project can be inferred as:

- To develop a geospatial application development for field data collected by members of TDSC on the CommunityGIS interface (<https://makerghat.urbansciences.in/>).

Specifically, we aim to fulfil the followings:

- To develop the interface of CommunityGIS[3] for TDSC projects.
- To set up a centralized geospatial sever for storing all the TDSC data.

1.3 Significance

The CommunityGIS is a platform for community for a spatial data infrastructure. Primary objective of this web interface is to provide data for educational platform for learners so that they can learn about spatial information and learn how these modern tools can be used to solve real life problems. As a platform it is a digital way to collect, aggregate, access, analyse data and finally provide decision support to the community manipulate reality. All the technology used is completely open-source so that anyone keen to replicate it for their use can do it

As far as the significance and need of a web GIS system, it is being highly used in many industries like telecom, agriculture, surveying etc. Such a system plays an extremely important role in the telecoms and network services. It is used in planning, collecting, analysing and storing the complex network designs that are needed to come up with a working architecture in the field of telecoms. GIS can be used to provide information that can be used to plan what

crops need to be grown in what areas of the farm depending on the soil structure and soil composition. Land use planning will depend on the information provided by GIS in deciding which part of the land should be subjected to what land use. It becomes easier to know what crops will thrive on what part of the land and this helps in improving yields. GIS stores data that can be relevant in community development. The data can provide insights that can help inform the planning of the community land based on the needs of the community.

This particular project is significant because of its very nature of providing a visualizing tool to common people. To know and understand the projects and government schemes being introduced and implemented in their location in a more intuitive and user-friendly manner.

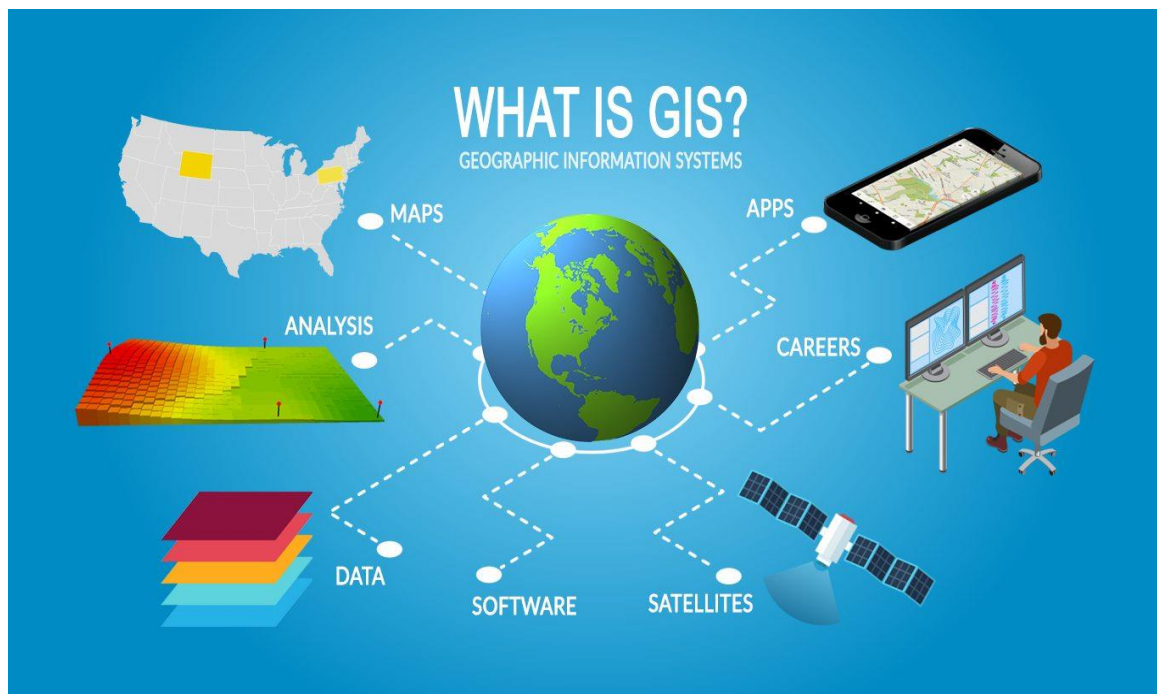


Fig 1.1. Components of GIS

1.4 Research Design

TDSC is an ongoing initiative of CTARA, which aims to supply consultancy services to the development needs of various entities at the bottom 80%, such as habitations, gram panchayats, small towns, district and taluka administrations and so on. As development progresses, the design, analysis, monitoring and evaluation needs of society at large needs the services of professionals who are trained in both engineering and governance, policy and other interdisciplinary skills. TDSC was established to meet this need.

Various data is collected by TDSC members while auditing various government schemes and implementing and finding solutions for the same. This data is research based and is mostly geospatial. Hence, the visualization of such data will require technologies that can deal with geospatial formats like shapefile. Then, once the data is collected the next big task is to convert this raw data into desired format. Data conversion is the most important and complicated step as there can be various formats of spatial data. The data we get is in the Keyhole Markup Language (KML) that contains location co-ordinates and Extensible Markup Language (XML) that contains various attributes describing the various locations. Expert discussion was required in order to find different ways to convert data into Shapefile (.shp) format, that contains the combine information of KML and XML.

The server requirement for the central repository that can store all the spatial as well as non-spatial data of TDSC was met by using Geonode after going through a number of gis servers.

1.5 Source of Data

Geospatial data is data about objects, events, or phenomena that have a location on the surface of the earth. The location may be static in the short-term (e.g., the location of a road, an earthquake event, children living in poverty), or dynamic (e.g., a moving vehicle or pedestrian, the spread of an infectious disease). Geospatial data combines location information (usually coordinates on the earth), attribute information (the characteristics of the object, event, or phenomena concerned), and often also temporal information (the time or life span at which the location and attributes exist).

TDSC is an organization that provide technology-based solutions for the development sector. The source of data are the audits done by TDSC for various projects like Thakkar Bappa Yojana and the implementation of solutions in projects like Deonadi. The data is collected using ODK Collect and other similar mobile applications. The site visit is planned by the officials in TDSC and the members of the respective projects visit the site for audits. The data collected is in the KML format for representing the latitude and longitude of locations and the XML format for storing the attribute information for the location. That's why it is called a geospatial data.

Chapter 2

System Requirement Analysis

2.1 Information Gathering

A requirement is a vital feature of a new System which may include processing or capturing the data, controlling the activities of business, producing information and supporting the management.

Information gathering is a key part of the feasibility analysis process. Information gathering is both art and a science. It is a science because it requires a proper methodology and tools in order to be effective. It is an art too, because it requires a sort of mental dexterity to achieve the best results.

Before starting building this project, I have to do a research to know about geospatial data and how to work with them. There are various formats for that and what would be useful for our requirements was a challenge. The technology and server that can host a GIS interface was another thing to think about.

While collecting information for the various GIS servers, I collected information about various services that a GIS server should provide like Web Map Service, Web Feature Service (WFS), Web Coverage Service (WCS) etc. After understanding the need of these services and comparing with our requirements, Geonode was found to meet all the requirements.

This system is all about making the interface of visualizing various schemes meant for development more intuitive.

2.2 System Feasibility

Feasibility is a process that identifies, describes and evaluates proposed system and selects the best system for the job. During the study, the problem definition is solved and all aspects of problem to be included in the system are determined. Size of project, cost and benefits are also estimated with greater accuracy. The result of feasibility study is simply a report which is a formal document detailing the nature and scope of the proposed solution.

2.2.1 Economical Feasibility (Cost)

The economic feasibility of the project includes its economic appropriateness with respect to its presented output. If a project provides results of lower significance but requiring higher budgets then that project can't be economically viable. For the case of this project, the investment is not a lot in terms of economic aspects. All the technologies being used in order to build this project is open source and thus has no major expense other than hosting.

2.2.2 Technical Feasibility (Resource Availability)

The technical feasibility of the project deals with the availability and its actual implementation ability with the existing tools and techniques available in the software market world. The following are the notable points about the technical feasibility of the project:

- The project helps in visualising all the spatial data available with the organisation, hence helping the others to access the data by providing the download option.
- Used web development technologies like HTML5, CSS, JavaScript, and Bootstrap.
- Used Django for connecting the front end to the back end which is a high-level Python web framework that enables rapid development of websites.
- Used Wagtail Content Management System (CMS) for the development of UMA website
- The advantage of using Django technology is that it is easily reusable or replicated. Since the project is made using Python, developers will find it easy as all the modules are inter-connected and defined separately. For maintenance purpose too, it is easy to understand the code due to their readability. Data is generated by TDSC for different projects, its conversion into required format.
- Using Geonode as the centralized server for internal TDSC files.

With all these perspectives taken into consideration, the project is technically feasible to implement.

2.2.3 Behavioral Feasibility (Benefits)

People are inherently resistant to change, but if the change tends to provide enough benefits such that it is worth accepting then the system possess behavioral feasibility. Our system benefits all variety of users as it is very intuitive and helps in proper visualization of geospatial data which is difficult otherwise. The CommunityGIS is not just easily understandable but also user-friendly. Hence, this system increases the value for money and this type of system is extremely helpful to not just normal people but also the authorities conducting the schemes and thus for communities in general.

2.3 Platform Specification

The software is being developed on Ubuntu 18.04 operating system with 12 GB RAM. It is a web application that will be available for everyone once deployed over a domain.

2.3.1 Hardware

Client Side:

RAM: at least 2GB

Available space: 650 MB (minimum)

Development Side:

As required by Geonode

2.3.2 Software

Client side:

Operating System: Any operating system that supports Browser (excluding MS-DOS).

Browser: Any browser compatible with IE 6.0.

Developer Side:

Operating System: Windows 2008, Ubuntu 16.04 or Mac OSX.

Browser: Any browser compatible with IE 6.0

Front-end technologies: HTML, CSS, JavaScript, Bootstrap, Leaflet

Back-end technologies: Django Framework, Wagtail CMS

Chapter 3

System Analysis

System analysis is the process of studying a procedure or business in order to identify its goals and purposes and create systems and procedures that will achieve them in an efficient way. Another view sees system analysis as a problem-solving technique that breaks down a system into its component pieces for the purpose of studying how well those component parts work and interact to accomplish their purpose.

3.1 Information Flow Representation

An information flow diagram (IFD) is a diagram that shows how information is communicated (or "flows") from a source to a receiver or target (e.g. $A \rightarrow C$), through some medium. The medium acts as a bridge, a means of transmitting the information. Examples of media include word of mouth, radio, email, etc. The concept of IFD was initially used in radio transmission. The diagrammed system may also include feedback, a reply or response to the signal that was given out. The return paths can be two-way or bi-directional: information can flow back and forth.

An IFD can be used to model the information flow throughout an organisation. An IFD shows the relationship between internal information flows within an organisation and external information flows between organisations. It also shows the relationship between the internal departments and sub-systems.

An IFD usually uses "blobs" to decompose the system and sub-systems into elemental parts. Lines then indicate how the information travels from one system to another. IFDs are used in businesses, government agencies, television and cinematic processes.[4]

3.1.1 Entity-Relationship Diagram

An entity–relationship model (or ER model) describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between entities (instances of those entity types).

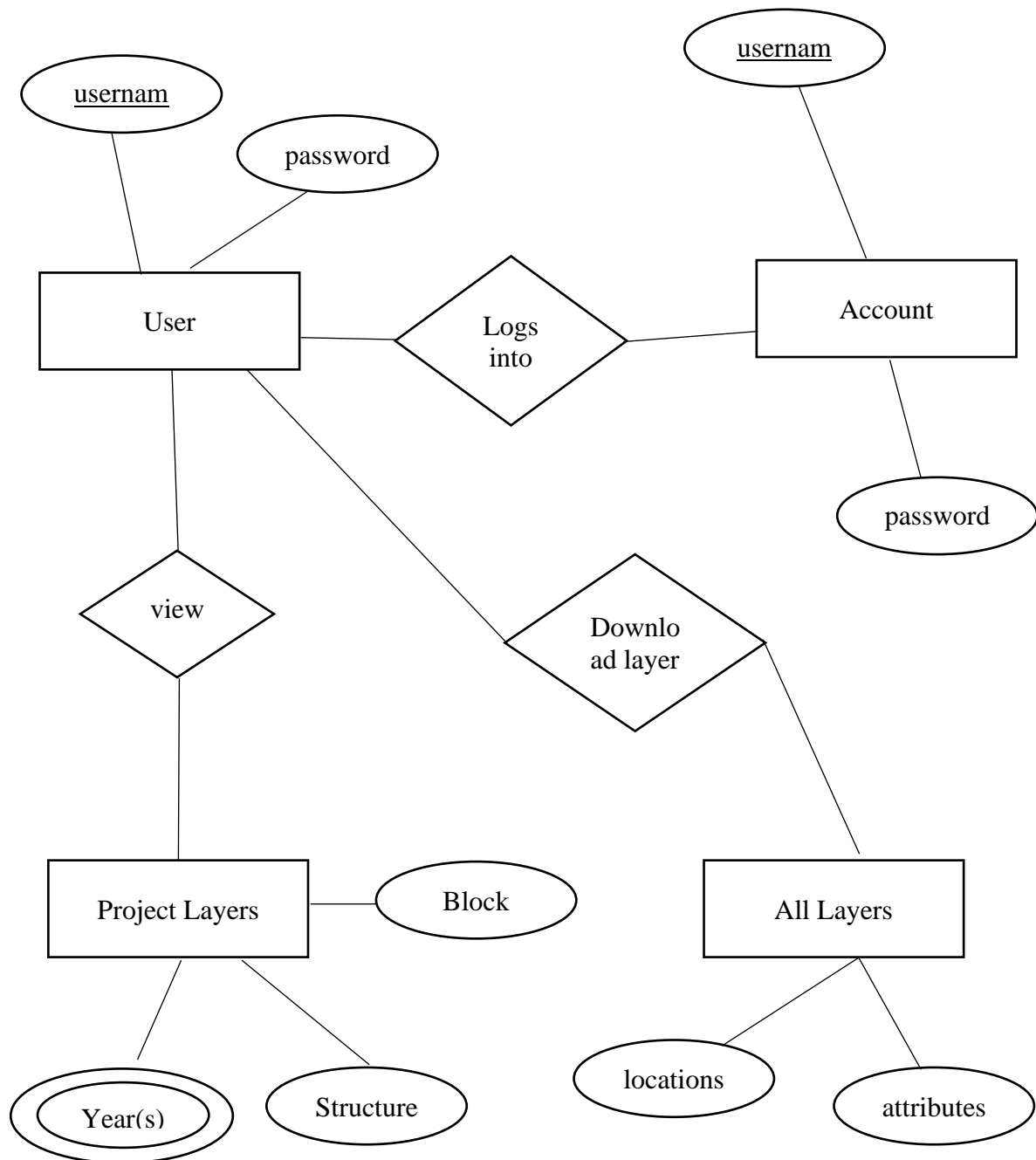


Fig 3.1. E-R Diagram of Geospatial Application

3.1.2 Use case Diagram

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform.

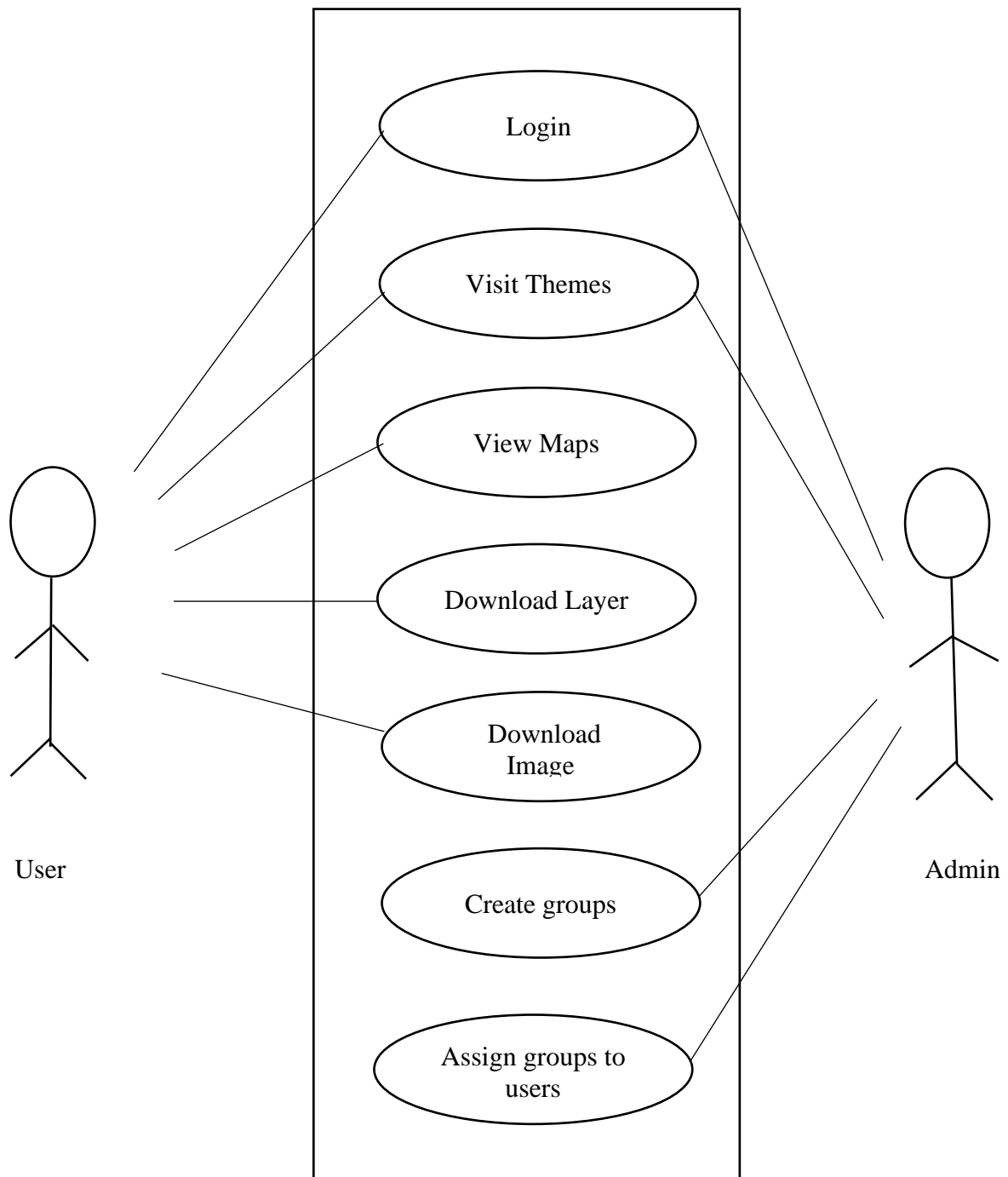


Fig 3.2. Use Case Diagram of Geospatial Application

Chapter 4

Design

Software architecture involves the high-level structure of software system abstraction, by using decomposition and composition, with architectural style and quality attributes. A software architecture design must conform to the major functionality and performance requirements of the system, as well as satisfy the non-functional requirements such as reliability, scalability, portability, and availability. A software architecture must describe its group of components, their connections, interactions among them and deployment configuration of all components.

A software architecture can be defined in many ways –

- **UML** – UML is one of object-oriented solutions used in software modeling and design.
- **Architecture View Model (4+1 view model)** – Architecture view model represents the functional and non-functional requirements of software application.
- **ADL (Architecture Description Language)** – ADL defines the software architecture formally and semantically [5]

4.1 Architectural Design

The software needs the architectural design to represent the design of software. IEEE defines architectural design as “the process of defining a collection of hardware and software components and their interfaces need to establish the framework for the development of a computer system.” The software that is built for computer-based systems can exhibit one of these many architectural styles. Each style will describe a system category that consists of:

- A set of components (e.g.: a database, computational modules) that will perform a function required by the system.
- The set of connectors will help in coordination, communication, and cooperation between the components.
- Conditions that how components can be integrated to form the system.
- Semantic models that help the designer to understand the overall properties of the system.

4.1.1 Architectural Context Diagram

The context diagram is used to establish the context and boundaries of the system to be modelled: which things are inside and outside of the system being modelled, and what is the relationship of the system with these external entities.

A context diagram, sometimes called a level 0 data-flow diagram, is drawn in order to define and clarify the boundaries of the software system. It identifies the flows of information between the system and external entities. The entire software system is shown as a single process.

In order to produce the context diagram and agree on system scope, the following must be identified:

- external entities
- data-flows

4.1.2 Architectural Behavioral Diagram

Behavioral Diagrams depict the elements of a system that are dependent on time and that convey the dynamic concepts of the system and how they relate to each other. The elements in these diagrams resemble the verbs in a natural language and the relationships that connect them typically convey the passage of time.

Use Case Diagrams:

Use Case diagrams capture Use Cases and relationships among Actors and the system; they describe the functional requirements of the system, the manner in which external operators interact at the system boundary, and the response of the system.

Actor:

An actor represents the roles that the users of the use cases play. An actor may be a person (e.g. student, customer), a device (e.g. workstation), or another system (e.g. bank, institution).

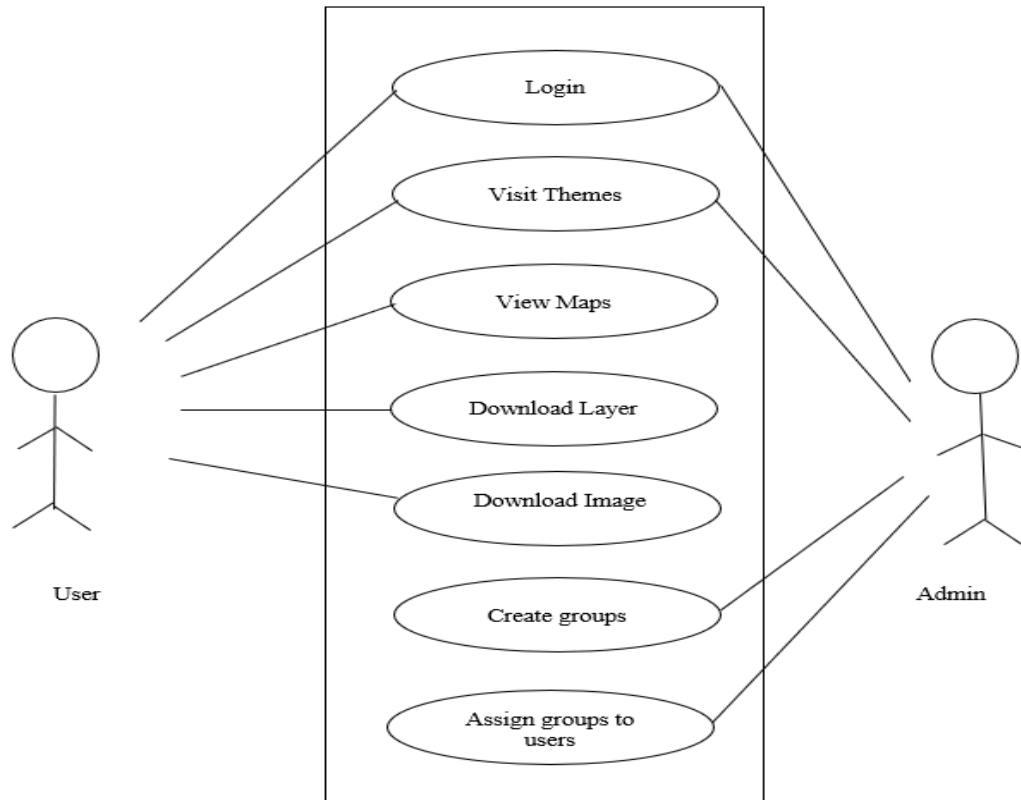


Fig 4.1. Use Case Diagram of Geospatial Application

4.1.3 Description of Architecture diagrams

Behavioral diagrams are used to depict how the software behaves and interacts with the different conditions of the environment with different actors. There are various kind of actors that can act on a software.

In our project there are two actors, one is a registered user or a guest user and other is a admin. However, the kind of actors is unlikely to change.

User can Login, Visit Theme, View Maps, Download Layer and Download Image. While the admin can Login, View Themes, create groups and assign users to groups.

As the functionalities add up the use cases will grow and also relationships between various use cases will also develop.

4.1.4 Control Hierarchy

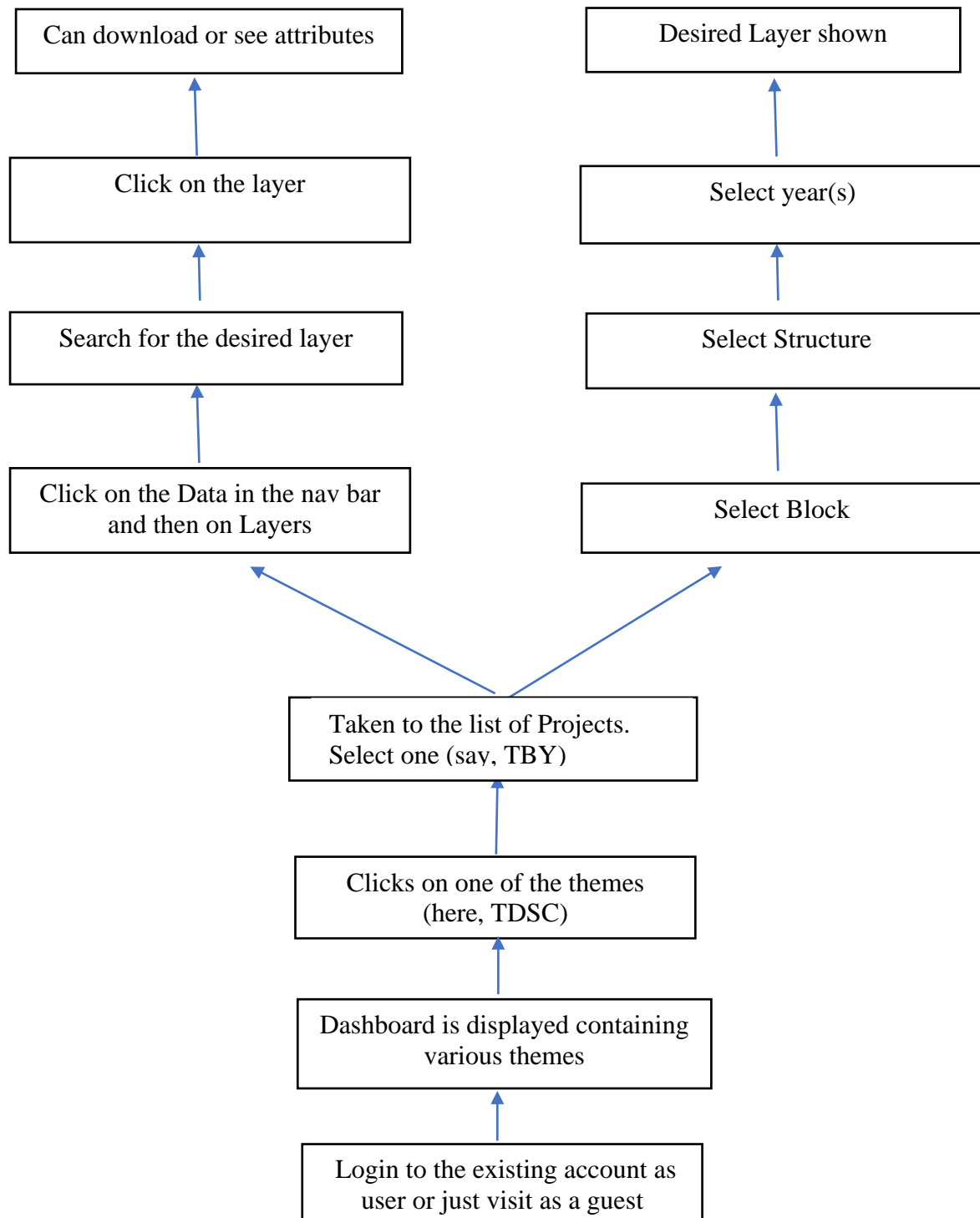


Fig 4.2. Control Hierarchy of Geospatial Application

This geospatial interface can be used to view different layers on the basis of blocks and other functionalities selected. The Interface of CommunityGIS is user-friendly and presents a good option to visualize data of the audits in GIS format to make it more intuitive. The website will open with an option of “Explore Theme”. Once selected it will take you to dashboard that contains various themes. My area of development deals with the TDSC theme. Once selected it will take you to the various projects of TDSC. Select a project and it will display a map interface. Select desired attributes, of which layer you want to display. One can reset the selected options to try some other combinations.

This interface also gives the option to download the layer or see its attributes as it is open source.

4.2 Modular and Procedural Approach

There are a number of technologies and software used, in order to build this system. Basic back-end language is Python.

4.2.1 Software Used

4.2.1.1 QGIS

QGIS is a free and open-source cross-platform desktop geographic information system (GIS) application that supports viewing, editing, and analysis of geospatial data.

QGIS integrates with other open-source GIS packages, including PostGIS, GRASS GIS, and MapServer. Plugins written in Python or C++ extend QGIS's capabilities. Plugins can geocode using the Google Geocoding API, perform geoprocessing functions similar to those of the standard tools found in ArcGIS, and interface with PostgreSQL/PostGIS, SpatiaLite and MySQL databases.[6]

4.2.1.2 GeoNode and GeoServer

Geonode version 2.8 on Ubuntu 16.04, latest is geonode 2.10 on Ubuntu 18.04. Geoserver built-in for the used geonode version is 2.15.

The following are the bare minimum system requirements: [7]

- 8GB of RAM.

- 2.2 GHz processor with 2 cores. (Additional processing power may be required for multiple concurrent styling renderings)
- 10 GB software disk usage.
- Additional disk space for any data hosted with GeoNode.
- 64-bit hardware recommended.

GeoServer is an open source server for sharing geospatial data. It is designed for interoperability and excels at publishing any major spatial data source using open standards. With suitable preparation of data, it excels at handling very large datasets, both raster and vector. It produces high quality rendering of maps and can handle hundreds to thousands of map layers easily. GeoServer is an open-source server written in Java that allows users to share, process and edit geospatial data.



Fig 4.3. Geoserver

Spatial Data uploaded on geonode contains 6 files (cpj, dbf, prj, qpj, shp, shx) for any layer.

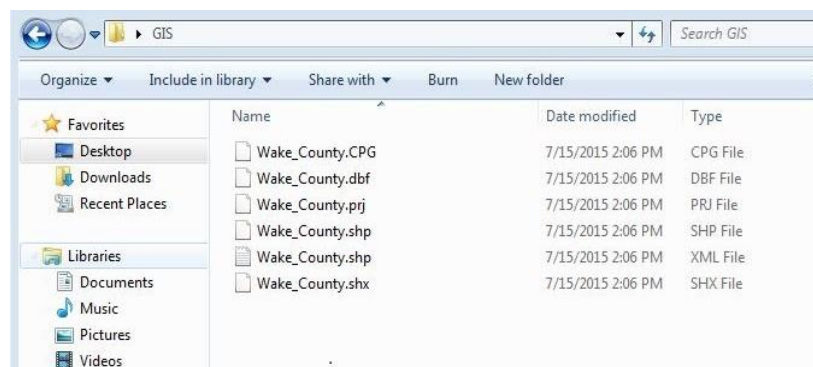


Fig 4.4. Shapefiles having 6 files

Shape file 'shp' contains the vector data (point, polyline or polygon) and 'dbf' file contains the attributes and their values. The layer uploaded on the geonode can be accessed using Leaflet (javascript library) source, via geoserver, which retrieves the shape file and display on the base map. Since using wms of geoserver, on click attribute values can also be read. [img] On uploading geoserver gives default styling to the layers, but styles can be created and customized according to the requirement. Hence, showing labels like village name was possible.

4.2.1.3 Django

Django follows MVC (model-view-controller) architecture pattern, written in Python programming language designed for creating easy and rapid development. Django contains four layers:

- URL Layer describes the different url pattern, '/admin' is by default which redirects to admin login page. Each url has a corresponding view, so on search for any url the request is sent to the corresponding view which in turn display the corresponding template.
- Models Layer describes the database schema and data structure. The database used is PostgreSQL, an extensible relational database management system, extending it with PostGIS, which supports the spatial data.
- Views Layer is UI responsible for controlling what a user sees, retrieving data from appropriate models and executing any calculation made to the data. It finally sends executed data to the template.
- Templates Layer is responsible for determining how a user sees. Also, it is responsible for describing how the data is received from the views.

Django project can contain various apps, which follows this MVC architecture. Project itself contains settings file which contains the list of settings for django project and their default values, like Allowed hosts mentions the strings representing the host or domain names that a particular Django site can serve. The host where geoserver is installed is mentioned in this, if it is on different server. Installed_Apps contains the list of strings designating all applications that are enabled in this Django installation or are to be used within the project. The Databases setting by default configure database, any number of additional databases can also be specified. When connecting to database backend, additional parameters are required.

Django's architecture

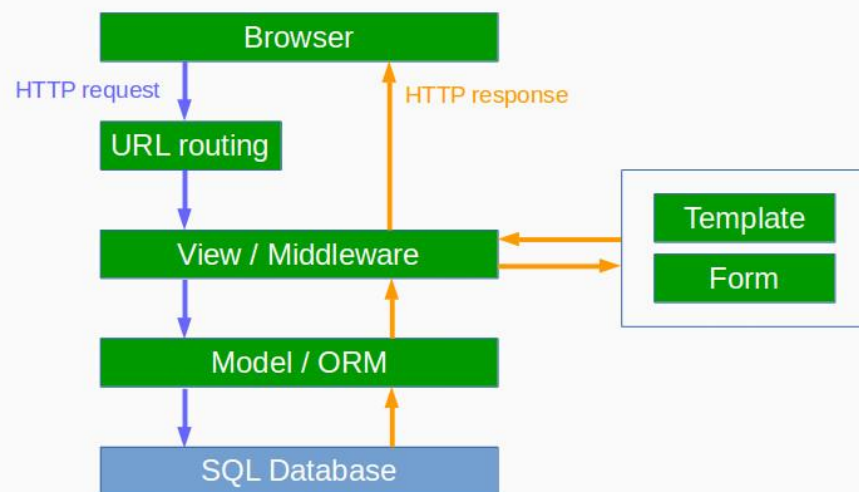


Fig 4.5. Django Architecture

4.2.1.4 Leaflet and JavaScript

Leaflet is a JavaScript library used for creating interactive maps. The project uses various JavaScript functions and leaflet features to create various templates.[8] For retrieving geoserver layer: Domain path is given, along with the services and layer name, as shown below:

```
294  
295 var geojson;  
296 const domain = ['https://makerghat.urbanosciences.in/', 'http://localhost:8080/'];  
297 var base_url = domain[0] + "geoserver/geonode/ows?service=WFS&version=1.0.0&request=";  
298 var info = L.control();  
299 var attribute_table = L.control({position: 'bottomright'});  
300 var layer_name;  
301
```

Fig 4.6. Domain and baseURL for the layer

4.2.2 Internal Data Structures

While dealing with GIS projects it becomes inevitable to deal with large chunks of data. To deal with large amounts of data it requires proper organization of data with the help of various data structures. Various data structures are used to store data in various formats. For example, an array stores homogenous data while a list can store heterogeneous data too. To deal with the large amounts of dataset that we require for our project we have used the models used in Django framework. Django by default comes with SQLite database but we have used PostgreSQL where we can store the relevant data that can be accessed. Here we have used a number of models, One for each project's each block's each structure. For example, the Thakkar Bappa Yojana was to take place in 8 districts of Palghar. Then, all the 8 blocks (Mokhada. Vasai, and so on) will have a model having fields as the attributes associated with each layer of the blocks.

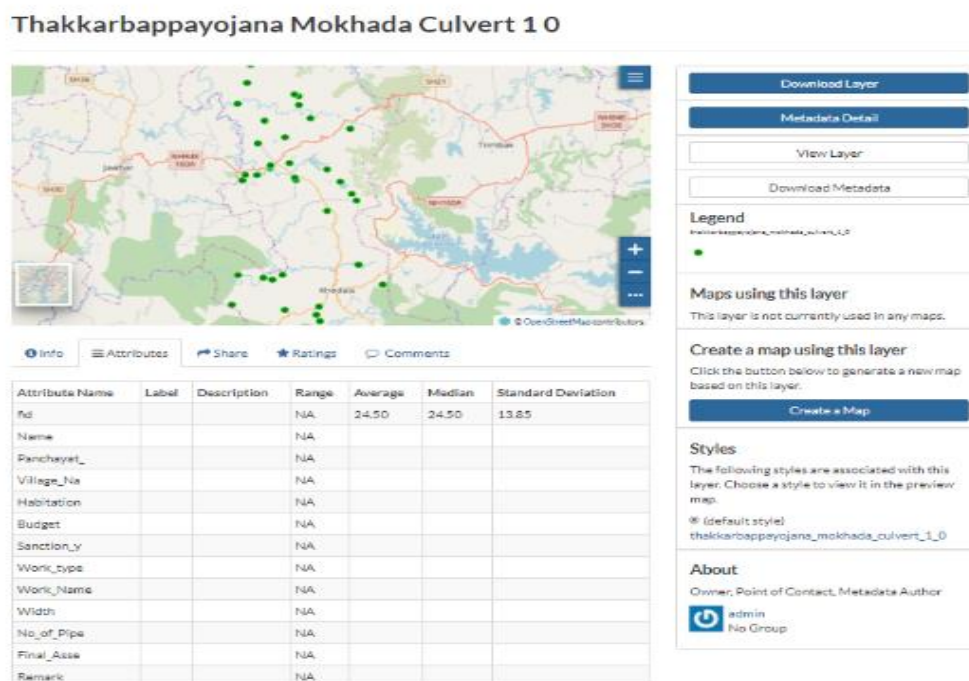


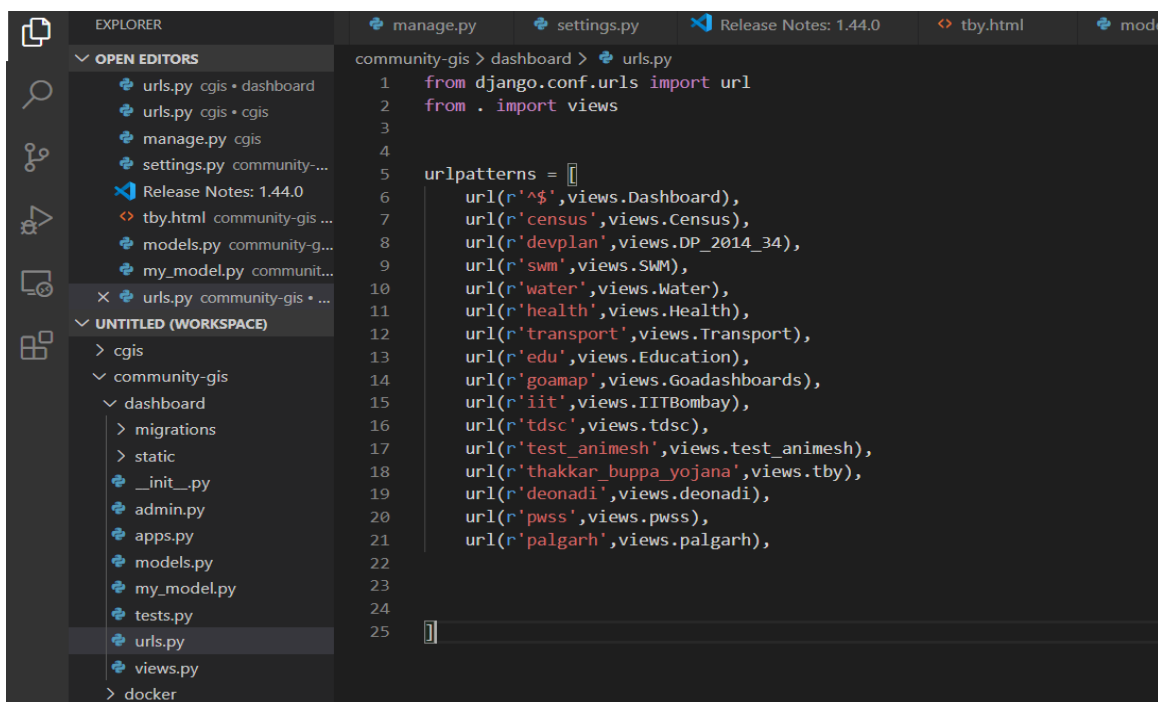
Fig 4.7. Attributes of the layer

The simple but elegant use of these data structures make the program light on the computer's memory but also makes the processing of the text faster and easier. This also makes way for dealing with much larger data in a very small secondary memory space.

4.2.3 Coding

Coding is one of the most crucial steps of the whole web development. It includes taking all the different software and technologies together. This project has included coding in the Django framework in python. For front-end, other than the basic technologies like HTML and CSS, Leaflet and JavaScript is used. For the website development of UMA, wagtail is used for the backend.

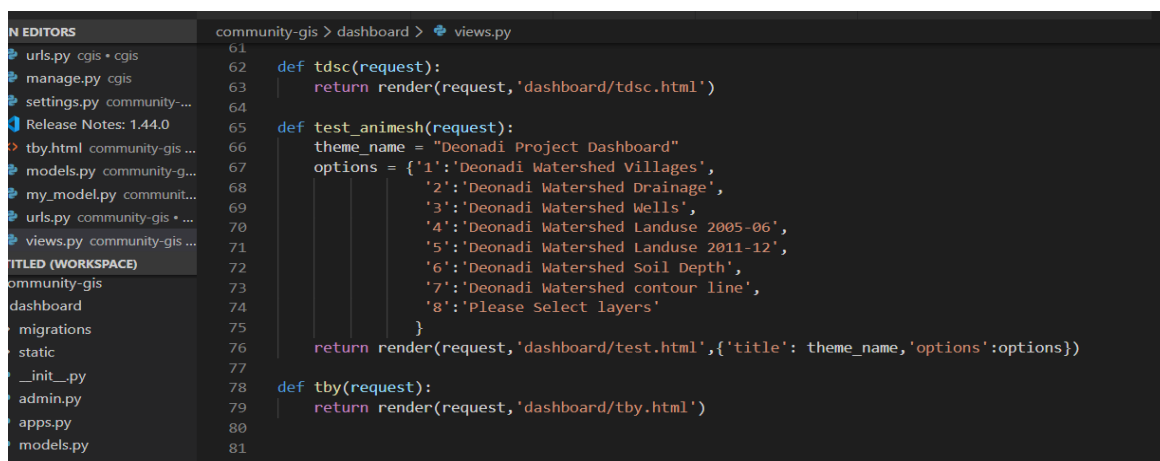
4.2.3.1 urls.py



```
community-gis > dashboard > urls.py
1  from django.conf.urls import url
2  from . import views
3
4
5  urlpatterns = [
6      url(r'^$', views.Dashboard),
7      url(r'^census', views.Census),
8      url(r'^devplan', views.DP_2014_34),
9      url(r'^swm', views.SWM),
10     url(r'^water', views.Water),
11     url(r'^health', views.Health),
12     url(r'^transport', views.Transport),
13     url(r'^edu', views.Education),
14     url(r'^goamap', views.Goadashboards),
15     url(r'^iit', views.IITBombay),
16     url(r'^tdsc', views.tdsc),
17     url(r'^test_animesh', views.test_animesh),
18     url(r'^thakkar_buppa_yojana', views.tby),
19     url(r'^deonadi', views.deonadi),
20     url(r'^pwss', views.pwss),
21     url(r'^palgarh', views.palgarh),
22
23
24
25 ]
```

Fig 4.8. urls.py

4.2.3.1 views.py



```
community-gis > dashboard > views.py
61
62 def tdsc(request):
63     return render(request, 'dashboard/tdsc.html')
64
65 def test_animesh(request):
66     theme_name = "Deonadi Project Dashboard"
67     options = {
68         '1': 'Deonadi Watershed Villages',
69         '2': 'Deonadi Watershed Drainage',
70         '3': 'Deonadi Watershed Wells',
71         '4': 'Deonadi Watershed Landuse 2005-06',
72         '5': 'Deonadi Watershed Landuse 2011-12',
73         '6': 'Deonadi Watershed Soil Depth',
74         '7': 'Deonadi Watershed contour line',
75         '8': 'Please Select layers'
76     }
77     return render(request, 'dashboard/test.html', {'title': theme_name, 'options': options})
78
79 def tby(request):
80     return render(request, 'dashboard/tby.html')
81
82 def deonadi(request):
```

Fig 4.9. views.py

4.2.3.1 models.py

```
31
32 class ThakkarbappayojanaMokhadaCulvert10(models.Model):
33     fid_1 = models.AutoField(primary_key=True)
34     the_geom = models.PointField(blank=True, null=True)
35     fid = models.IntegerField(blank=True, null=True)
36     name = models.CharField(db_column='Name', max_length=254, blank=True, null=True) # Field name made lowercase.
37     panchayat_field = models.CharField(db_column='Panchayat', max_length=254, blank=True, null=True) # Field name made lowercase. Field renamed
38     village_na = models.CharField(db_column='Village_Na', max_length=254, blank=True, null=True) # Field name made lowercase.
39     habitation = models.CharField(db_column='Habitation', max_length=254, blank=True, null=True) # Field name made lowercase.
40     budget = models.CharField(db_column='Budget', max_length=254, blank=True, null=True) # Field name made lowercase.
41     sanction_y = models.CharField(db_column='Sanction_y', max_length=254, blank=True, null=True) # Field name made lowercase.
42     work_type = models.CharField(db_column='Work_type', max_length=254, blank=True, null=True) # Field name made lowercase.
43     work_name = models.CharField(db_column='Work_Name', max_length=254, blank=True, null=True) # Field name made lowercase.
44     width = models.CharField(db_column='Width', max_length=254, blank=True, null=True) # Field name made lowercase.
45     no_of_pipe = models.CharField(db_column='No_of_Pipe', max_length=254, blank=True, null=True) # Field name made lowercase.
46     final_asse = models.CharField(db_column='Final_Asse', max_length=254, blank=True, null=True) # Field name made lowercase.
47     remark = models.CharField(db_column='Remark', max_length=254, blank=True, null=True) # Field name made lowercase.
48
49     class Meta:
50         managed = False
51         db_table = 'ThakkarBappaVojana_Mokhada_Culvert_1_0'
52
53
54 class ThakkarbappayojanaMokhadaCulvert100(models.Model):
```

Fig 4.10. models.py

4.2.3.1 Fetching Layers

```
function fetch_layer_name(){
    var layer_name,structureIcon;
    var block = document.getElementById('block').value;
    var structure = document.getElementById('structure').value;
    switch(structure){
        case "Culvert":
            layer_name = "thakkarbappayojana_mokhada_culvert_1_0";
            structureIcon = L.AwesomeMarkers.icon({icon: 'bars', markerColor: 'red', prefix: 'fa' });
            break;
        case "Dug well":
            layer_name = "thakkarbappayojana_mokhada_well_1_0";
            structureIcon = L.AwesomeMarkers.icon({icon: 'bullseye', markerColor: 'blue', prefix: 'fa' });
            break;
        case "Public Toilets":
            layer_name = "thakkarbappayojana_mokhada_publictoilet_1_0";
            structureIcon = L.AwesomeMarkers.icon({icon: 'bullseye', markerColor: 'blue', prefix: 'fa' });
            break;
        case "Concrete Gutter":
            layer_name = "thakkarbappayojana_mokhada_well_1_00";
            structureIcon = L.AwesomeMarkers.icon({icon: 'bullseye', markerColor: 'blue', prefix: 'fa' });
            break;
        case "Samaj Mandir":
            layer_name = "thakkarbappayojana_mokhada_samajmandir_1_01";
            structureIcon = L.AwesomeMarkers.icon({icon: 'bullseye', markerColor: 'blue', prefix: 'fa' });
            break;
        case "Smashan Bhoomi":
            layer_name = "thakkarbappayojana_mokhada_smashanbhoomi_1_0";
            structureIcon = L.AwesomeMarkers.icon({icon: 'bullseye', markerColor: 'blue', prefix: 'fa' });
            break;
        case "Bituminous Road":
            layer_name = "thakkarbappayojana_mokhada_bit_road_1";
            structureIcon = L.AwesomeMarkers.icon({icon: 'bullseye', Color: 'red', prefix: 'fa' });
            break;
    }
    var name_and_style = [layer_name,structureIcon];
}
```

Fig 4.11. Fetch Layer Function

4.2.3.1 Displaying Lines or Points based on the geometry

```
function displayLinesWithYear(layer_name,years){
  console.log(years);

  pointLayerList.forEach(layer => mymap.removeLayer(layer));
  mymap.spin(true,{lines: 9, length: 2, width: 20, scale: 60,radius: 70, color: "grey"});
  url = base_url + "&typeName=geonode%3A" + layer_name + "&maxFeatures=50";
  console.log(url);
  fetch(url)
  .then(
    function(response) {
      if (response.status !== 200) {
        console.log('Looks like there was a problem. Status Code: ' +
          response.status);
        return;
      }

      // Examine the text in the response
      response.json().then(function(stressData) {
        console.log(stressData);

        geojson = L.geoJson(stressData.features, {
          pointToLayer: function (feature, latlng) {
            for(var i=0;i<years.length;i++){
              if(feature.properties.cons_year === years[i]){
                console.log(years[i]);
                return L.polyline(latlng, {icon: L.AwesomeMarkers.icon({color: 'red', prefix: 'fa' })});
              }
            }
          }
        });
      });
    }
  );
}
```

Fig 4.12. Displaying Lines Function

4.3 Data Design

In this project most of the data sets used, is in the form of CSV files while training and then ultimately the models of the databases all are build by us and thus it does not involve much data design.

4.3.1 Data objects and Resultant Data Structures

A data object is a region of storage that contains a value or group of values. Each value can be accessed using its identifier or a more complex expression that refers to the object. In addition, each object has a unique data type. The data type of an object determines the storage allocation for that object and the interpretation of the values during subsequent access. The data objects used in the system are: Strings, Integers and Floating-point numbers. The resultant Data structures are in the form of tables that are the models build in Django. All the operations and functions are applied to these models to work with data and obtain results.

4.4 Interface Design

The interface of this project is quite simple and effective. The user can get away to the main point right from the beginning. As the project is open source it has a provision where the user can download the layers and see its attributes.

The interface has a good look and feel. For the map interface and layers display it uses leaflet library of JavaScript. In order to see any layer on the interface one has to select the filters. There is no need to login in order to view or download layers but the uploading of layers require login.

The interface of the UMA website is designed keeping in mind the requirements of a website for an organization of an institute.

4.4.1 Human Machine Interface Design Specification

The human machine interface design of our project gives look and feel of a website that is easy to use and handle by people of any age group with basic computer knowledge. The template of the project is simple yet effective. It goes by the understanding of the purpose of the project. The effort was to make it as intuitive as possible. The projects that TDSC deals with are for the development sector and thus this interface is designed to be used by both the government and the local people of the sector in the future, hence there was a need to balance it accordingly.

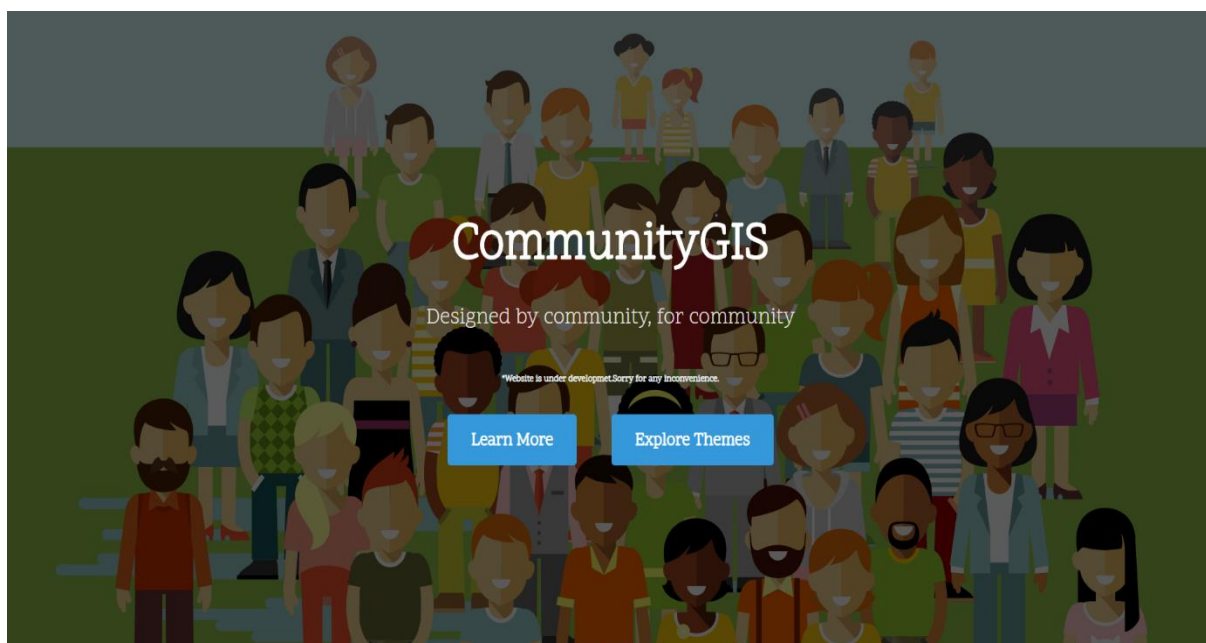


Fig: 4.13 CommunitGIS home page

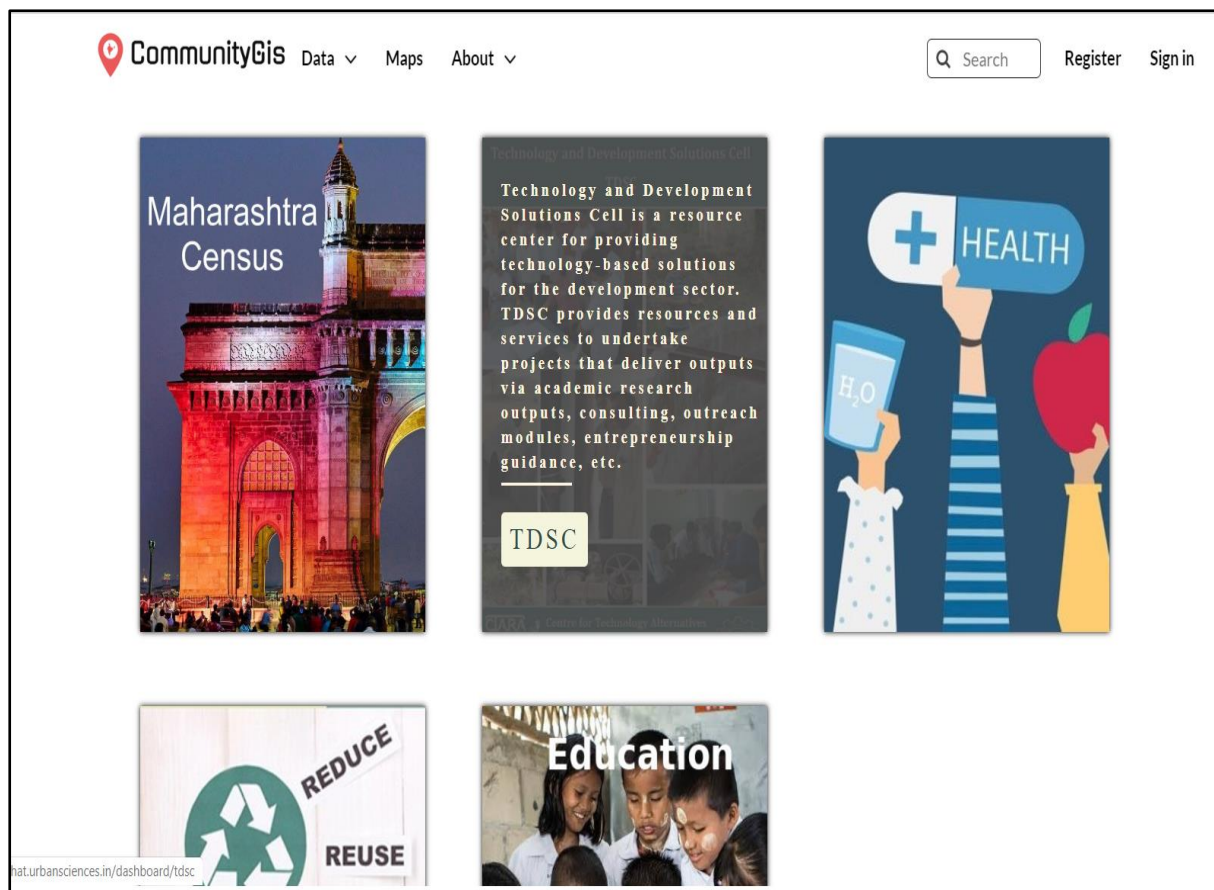


Fig: 4.14. Different themes of the interface

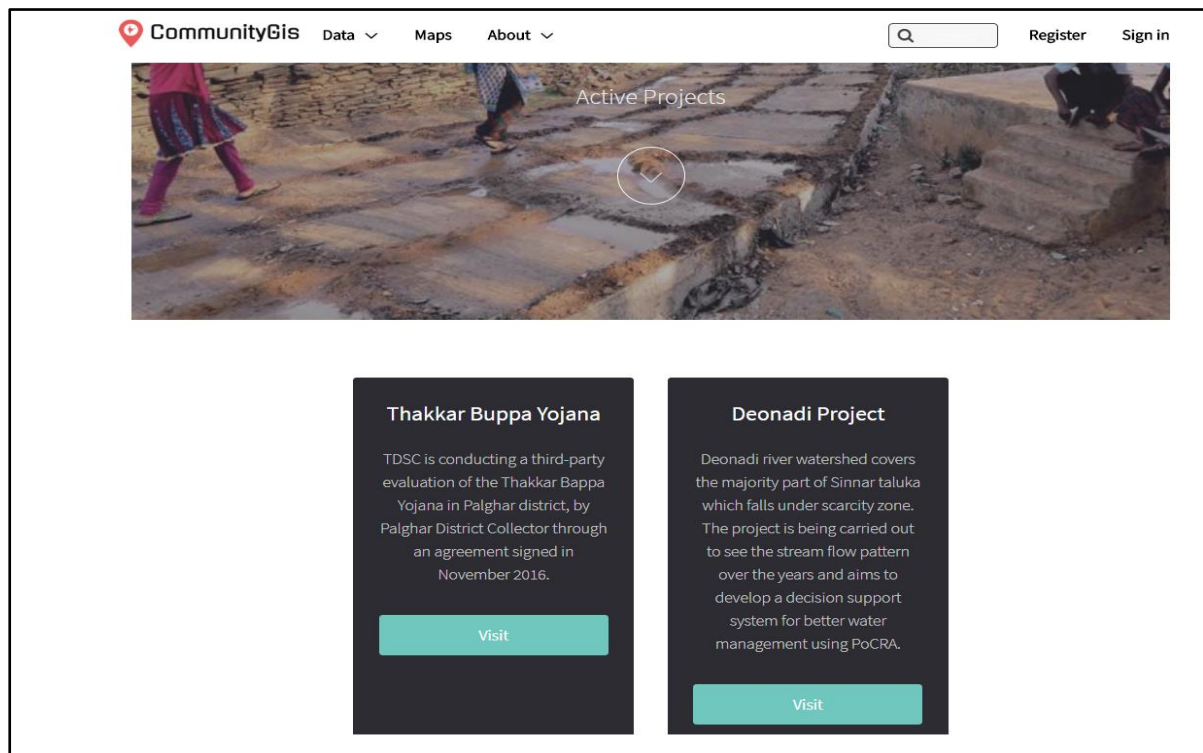
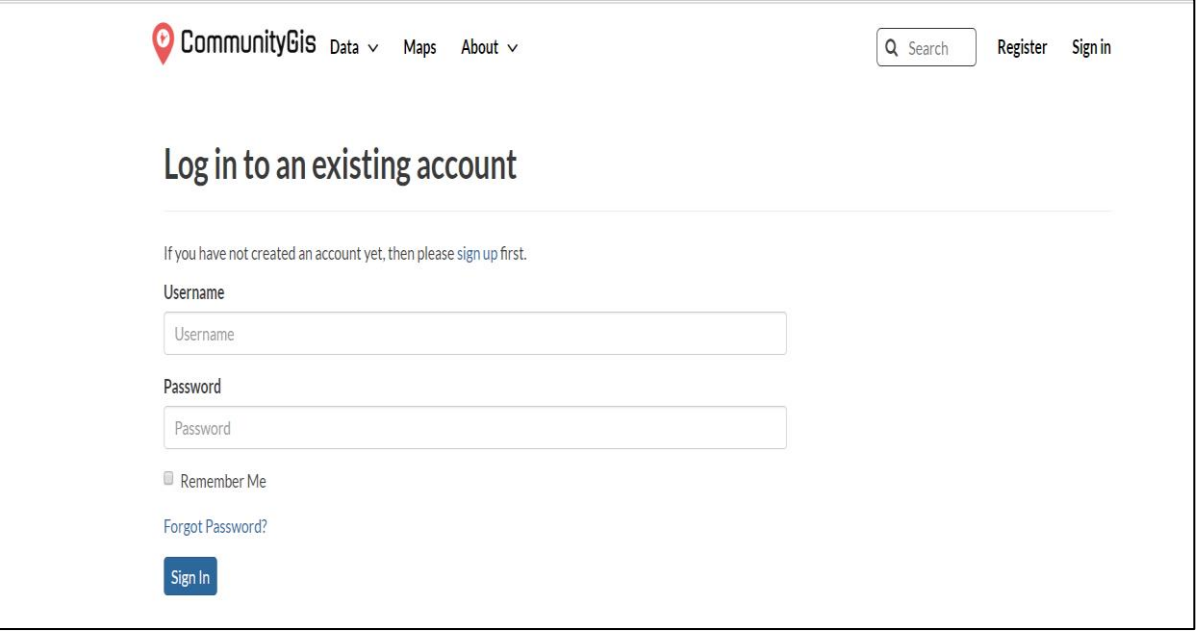


Fig: 4.15. Different projects of TDSC

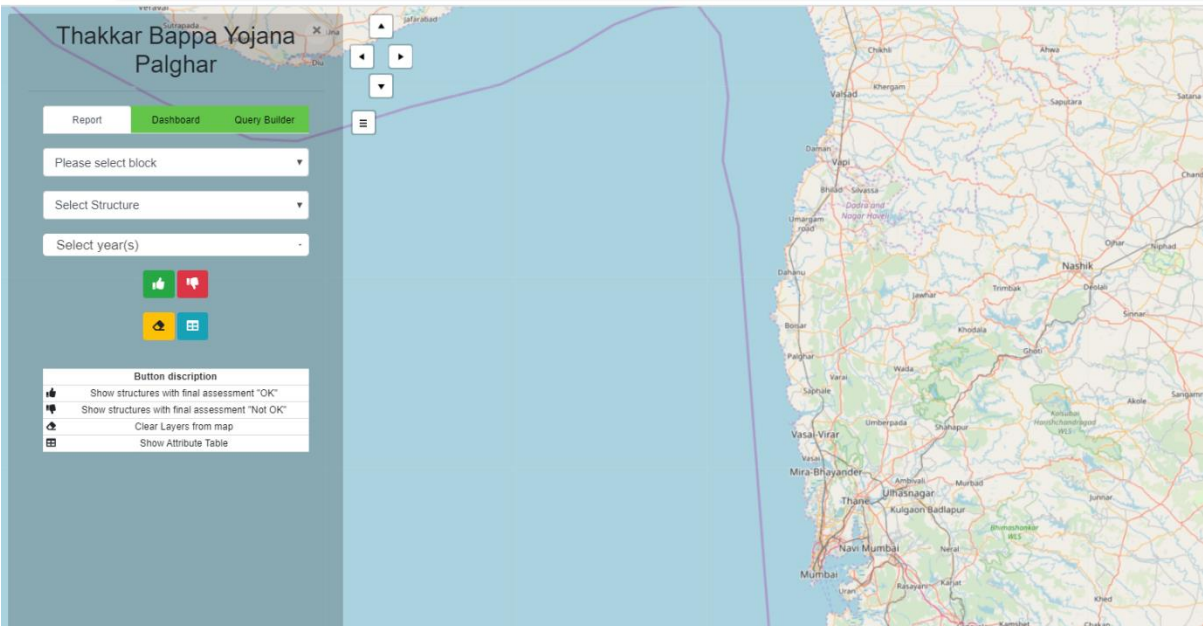
4.4.2 I/O Forms

The input of the user is taken using the website by using the frontend technologies like HTML, CSS, leaflet and JavaScript. There are two basic input forms, one is the login form taken care by the geonode. And the other one is the filters that has to be selected before displaying the layers.



The screenshot shows the 'Log in to an existing account' page on the CommunityGis website. The page has a header with the 'CommunityGis' logo, navigation links for 'Data', 'Maps', and 'About', a search bar, and links for 'Register' and 'Sign in'. The main heading is 'Log in to an existing account'. Below this, there is a message: 'If you have not created an account yet, then please [sign up](#) first.' The login form consists of two input fields: 'Username' and 'Password'. Below these fields are a 'Remember Me' checkbox and a 'Forgot Password?' link. At the bottom of the form is a blue 'Sign In' button.

Fig: 4.16. Login Form



The screenshot displays a web application interface for 'Thakkar Bappa Yojana Palghar'. On the left side, there is a sidebar with a navigation menu containing 'Report', 'Dashboard' (highlighted in green), and 'Query Builder'. Below the menu are three dropdown menus: 'Please select block', 'Select Structure', and 'Select year(s)'. There are also four colored buttons (green, red, yellow, blue) and a 'Button discription' table. The table lists four actions: 'Show structures with final assessment "OK"', 'Show structures with final assessment "Not OK"', 'Clear Layers from map', and 'Show Attribute Table'. The main area of the interface is a map of the Palghar region, showing various towns and roads. The map is overlaid with a grid and a purple line.

Fig: 4.17. Filters or the form to display the layer

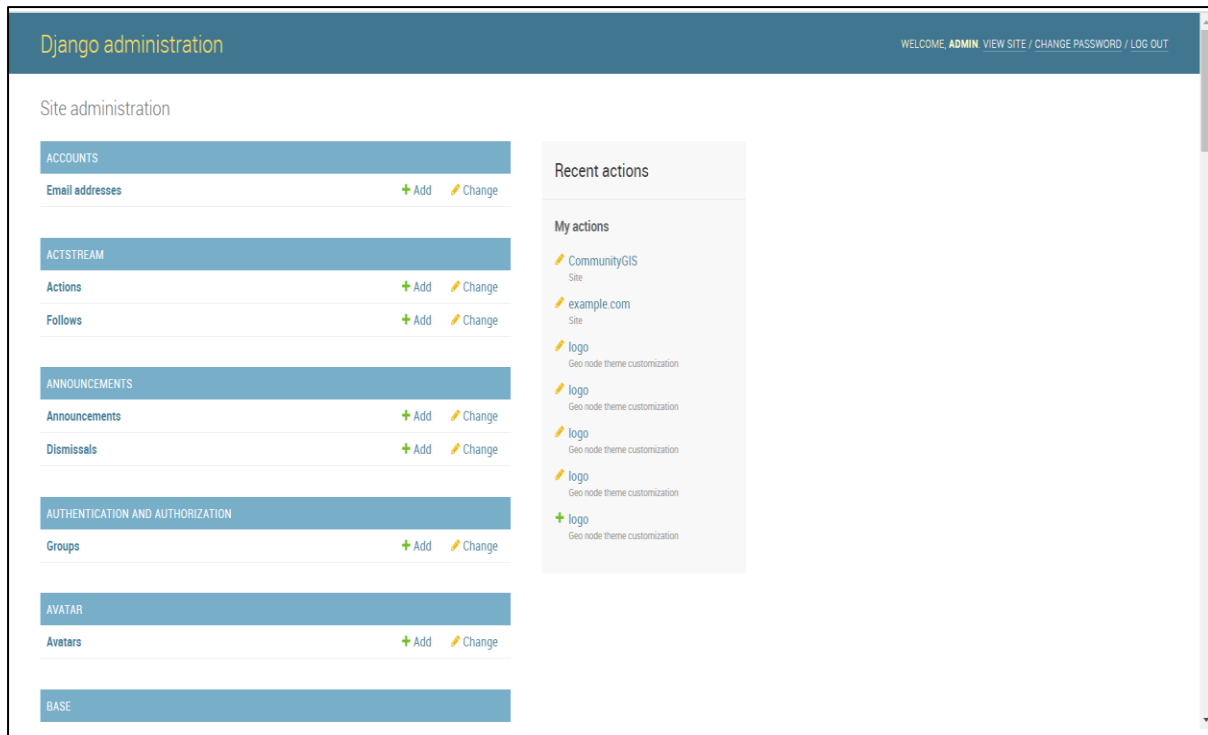


Fig: 4.18. Django Administration

4.5 Reports

The whole process went in a reverse chronological order where, first, we identified our needs from the project and then we went on creating the project.

The integral part of the project was the data conversion in the required format. The raw data is available in the KML and the XML format. In order to upload layers on the geoserver, the data needs to be in shapefile (.shp) format.

Once the data is converted the next crucial thing was to create an interface for each project based on the requirements. Also, the correct display of the layers that is intuitive to both a technical and non-technical person was important. This was achieved using the leaflet library.

Then to develop the GUI of the software, a template was required. Once the whole template is prepared, next big step was to work on the Django framework to carry out all the back-end functionalities using the models, forms and views of Django. This helped the website to be recognized as a whole working project.

Chapter 5

Testing

Software testing is a process, to evaluate the functionality of a software application with an intent to find whether the developed software met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product. Testing is conducted at the phase level in software development life cycle or at module level in program code. Software testing comprises of Validation and Verification.

5.1 Testing Objectives

Software Testing has different goals and objectives. The major objectives of Software testing are as follows:

- Finding defects which may get created by the programmer while developing the software.
- Gaining confidence and providing information about the level of quality.
- To prevent defects.
- To make sure that the end result meets the business and user requirements.
- To ensure that it satisfies the BRS that is Business Requirement Specification and SRS that is System Requirement Specifications.
- To gain the confidence of the customers by providing them a quality product.

Testing can either be done manually or using an automated testing tool:

- **Manual** – This testing is performed without taking help of automated testing tools. The software tester prepares test cases for different sections and levels of the code, executes the tests and reports the result to the manager. Manual testing is time and resource consuming. The tester needs to confirm whether or not right test cases are used. Major portion of testing involves manual testing.
- **Automated** - This testing is a testing procedure done with aid of automated testing tools. The limitations with manual testing can be overcome using automated test tools.

Tests can be conducted based on two approaches –

- Functionality testing
- Implementation testing

5.2 Testing Scope

Scope of testing includes the process to determine all those features or functionality as one may say that will be considered for Testing during a particular level of testing in a particular release. The scope is determined during the Test planning phase where in the test plan we mention the scope of testing we will consider.

The project has two parts one is the user interface, i.e. the front-end and the back-end functionality and the other one is the spatial data that contains both the location co-ordinates and the attribute information. The front-end didn't needed testing it was solely based on the look and feel that was required for a geospatial application. While the back-end that included Django and GeoNode required some amount of testing.

The main part of the project that required testing was the data collection and data conversion. It was the most crucial and complicated process. From identifying the data format required by geonode to understanding the process of conversion from the raw field data obtained from site visits was a process that needed double checking and testing.

There was a need to shorten the data conversion steps using some scripts because the current process was too hectic. Also, if the online interface is showing the correct layer and symbols and filters are intuitive enough was also something that needed testing.

5.3 Testing Principles

- Exhaustive testing is not possible. Instead, we need the optimal amount of testing based on the risk assessment of the application.
- Defect Clustering which states that a small number of modules contain most of the defects detected, i.e., approximately 80% of the problems are found in 20% of the modules.
- If the same sets of repetitive tests are conducted, the method will be useless for discovering new defects.
- Testing talks about the presence of defects and don't talk about the absence of defects. i.e. Software Testing reduces the probability of undiscovered defects remaining in the software but even if no defects are found, it is not a proof of correctness.

- It is possible that software which is 99% bug-free is still unusable.
- Testing should start as early as possible in the Software Development Life Cycle.
- Testing is context dependent which basically means that the way you test an e-commerce site will be different from the way you test a commercial off the shelf application.

Black-box testing

It is carried out to test functionality of the program. It is also called ‘Behavioral’ testing. The tester in this case, has a set of input values and respective desired results. On providing input, if the output matches with the desired results, the program is tested ‘ok’ and problematic otherwise.

In this testing method, the design and structure of the code are not known to the tester, and testing engineers and end users conduct this test on the software.

Black-box testing techniques:

- **Equivalence class** – The input is divided into similar classes. If one element of a class passes the test, it is assumed that all the class is passed.
- **Boundary values** – The input is divided into higher and lower end values. If these values pass the test, it is assumed that all values in between may pass too.
- **Cause-effect graphing** – In both previous methods, only one input value at a time is tested. Cause (input) – Effect (output) is a testing technique where combinations of input values are tested in a systematic way.
- **Pair-wise Testing** – The behaviour of software depends on multiple parameters. In pairwise testing, the multiple parameters are tested pair-wise for their different values.
- **State-based testing** – The system changes state on provision of input. These systems are tested based on their states and input.

White-box testing

It is conducted to test program and its implementation, in order to improve code efficiency or structure. It is also known as ‘Structural’ testing.

In this testing method, the design and structure of the code are known to the tester. Programmers of the code conduct this test on the code.

Below are some White-box testing techniques:

- **Control-flow testing** – The purpose of the control-flow testing to set up test cases which cover all statements and branch conditions. The branch conditions are tested for both being true and false, so that all statements can be covered.
- **Data-flow testing** – This testing technique emphasis to cover all the data variables included in the program. It tests where the variables were declared and defined and where they were used or changed.

Testing itself may be defined at various levels of SDLC. The testing process runs parallel to software development. Before jumping on the next stage, a stage is tested, validated and verified.

Testing separately is done just to make sure that there are no hidden bugs or issues left in the software. Software is tested on various levels –

Unit Testing

While coding, the programmer performs some tests on that unit of program to know if it is error free. Testing is performed under white-box testing approach. Unit testing helps developers decide that individual units of the program are working as per requirement and are error free.

Integration Testing

Even if the units of software are working fine individually, there is a need to find out if the units, if integrated together would also work without errors. For example, argument passing and data updating etc.

5.4 Testing Methods used

The testing method like white box testing and unit testing were applied at various levels that are generally carried out during the development of a software project.

The testing started in right from very initial stages, when we started dealing with the interface making and the data collection stage. The correct data format was very necessary for the correct functioning of the project. Also, the input and the output should be in sync to get the maximum efficiency from a project.

At each stage while creating a module, everyone in the team performed unit testing so as to check for the smooth working of each individual module.

5.5 Test Cases

To test the project, number of test cases were defined. We reiterated this whole process a number of times till satisfactory results were obtained.

Case 1: Checking for the correct conversion of data (as per the requirement).

Case 2: Understanding the filter inputs and looking for the correct displaying of the layer.

Case 3: To check if the point has the correct attributes associated with it.

Case 4: To check if the attribute table being fetched is correct as per the filter selected.

Using the combination of cases in Case 1, Case 2 and Case 3 we tested the software for different data set combination.

5.6 Sample Test Data and Results

Case 1: Checking the correct conversion of data.

The Data conversion was the most crucial process as it included the conversion of KML and XML format data collected from various sites through ODK Collect or other similar working applications. The location co-ordinates were stored in KML file and the attributes of the location were stored in the XML file.

Data conversion steps include:

- Conversion of XML file into CSV file using a script.
- Conversion of KML file into shapefile using QGIS software.
- Merging the shapefile obtained from KML with the CSV to obtain the final shapefile.

- Once the final shapefile is obtained test it using the QGIS software.

Case 2: Checking of the output using the input

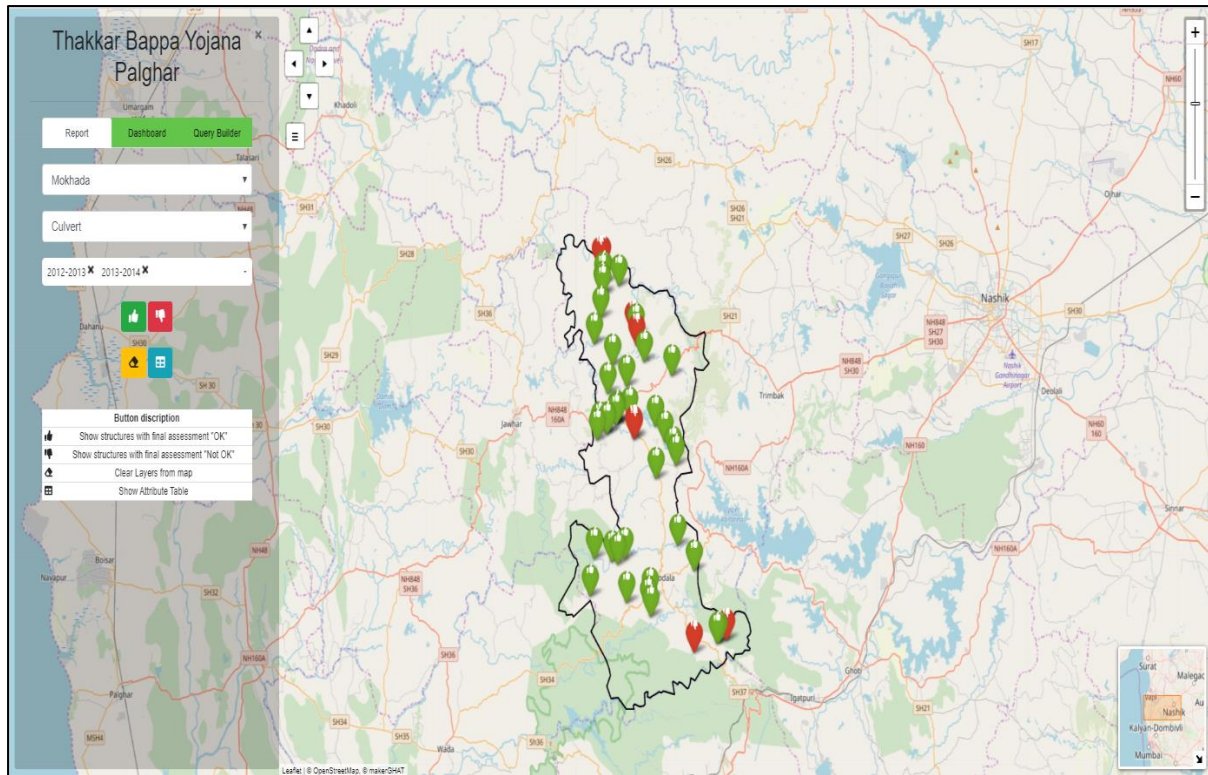


Fig: 5.1. Output as per input

The testing for the output based on the input was done by selecting the relevant options for the block, structure and year. The diagram is of the Thakkar Bappa Yojana, in which the third-party audit was done in the 8 blocks of Palghar. The example showcases the Culvert in the given years of the Mokhada Block of Palghar. The output was as per the input and thus was passed.

Case 3: Checking the attributes associated with the location geometry

Just the geometry and location co-ordinated with no description or relation with the project meant no sense. Hence, one of the crucial tasks was to be able to show attributes for every road or culvert or public toilet. This table gives the information about the geometry.

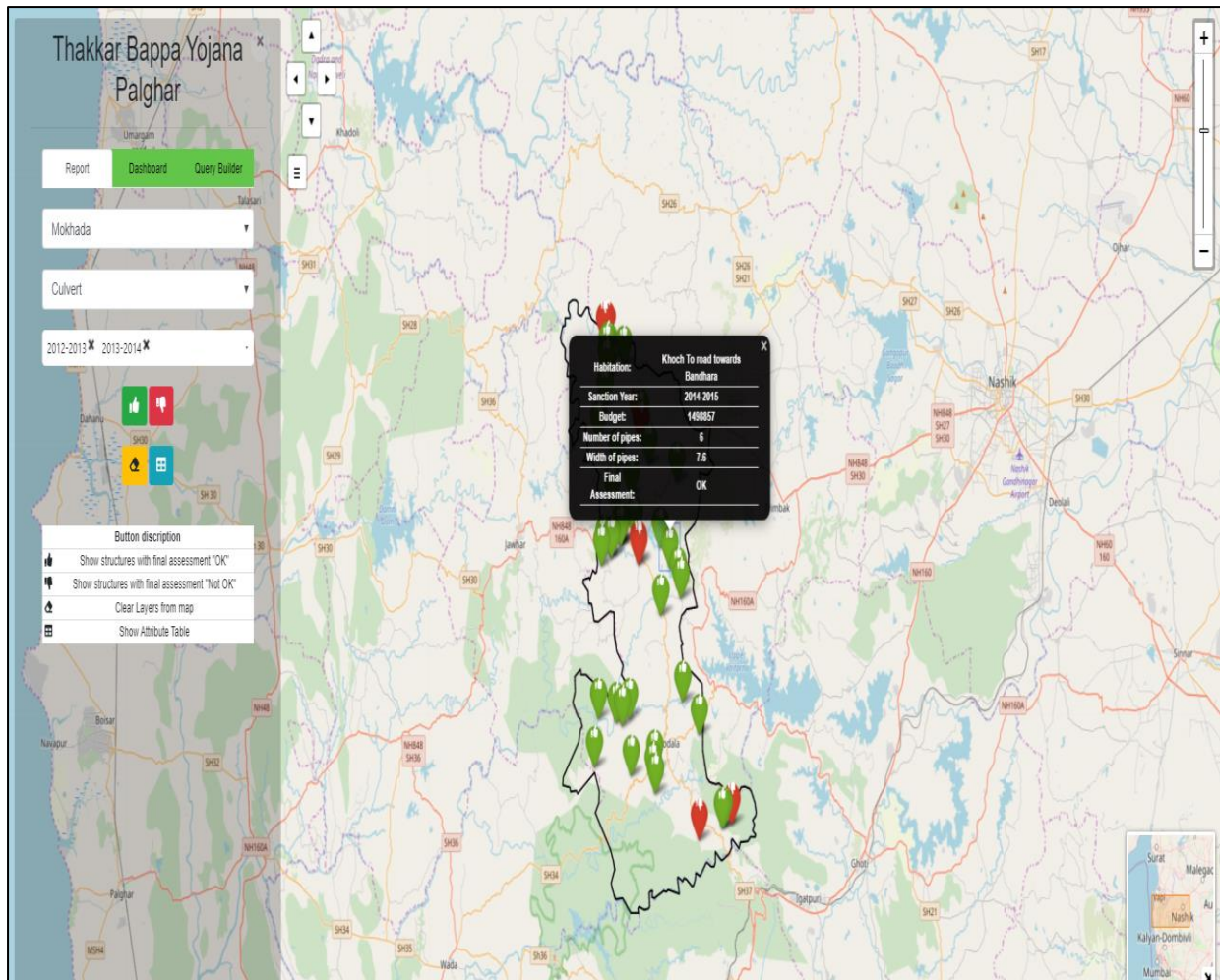


Fig: 5.2. Point describing the associated attributes

Case 3: Checking the attribute table being shown is correct as per the Taluka and villages selected

The Rural Water Supply Scheme project of TDSC, deals with the assessment of the Schemes including the construction of ESR I, ESR II, transmission main, gravity main, pump house, distribution network etc for energy need. This scheme was implemented in villages of Palghar and Thane. The GIS mapping of such a project required the proper association of attribute table with the Taluka and villages being selected. Following figure shows the attribute table corresponding to the Vada taluka and its villages.

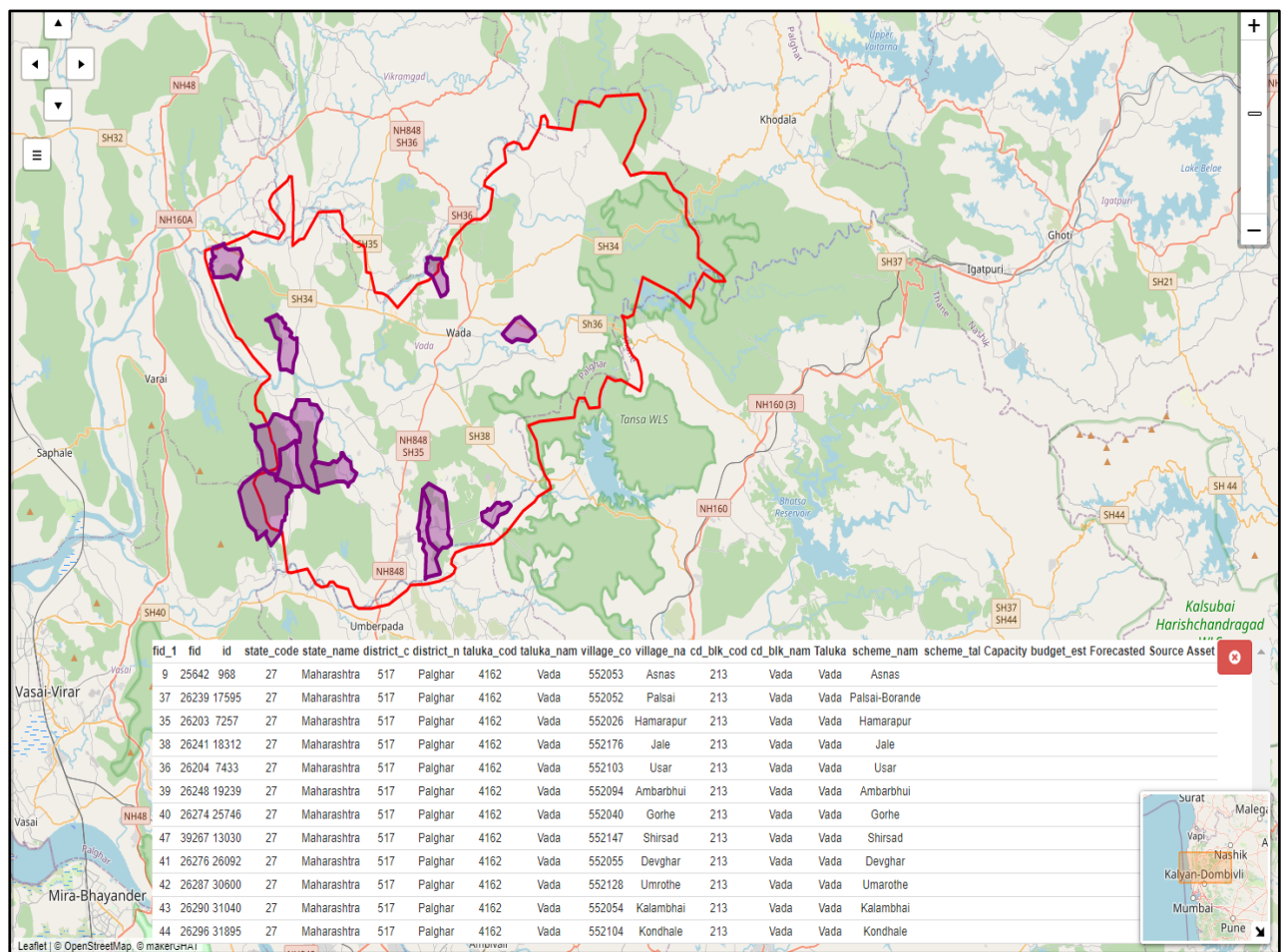


Fig: 5.3. GIS mapping for the Vada Taluka and its associated attribute table

Chapter 6

Results

During this period of my internship I went on exploring many GIS applications. Till now I have worked on two projects. They are as follows:

1. Geospatial Application Development for the primary field data:

This was the primary project and included working with GIS. TDSC wanted to establish a centralized server where they can keep all there spatial and non-spatial files. This was for their internal use.

Next task is to develop the interface of CommunityGIS for TDSC and to be able to show their projects on this interface so that both the government authorities and the local people can check about the various auditing task carried out in the districts of Maharashtra regarding the schemes for development sector.

The interface is ready and is working fine for the data of one of the blocks for Thakkar Bappa Yojana Project and for the talukas of Palghar in Rural Water Supply Scheme Project.

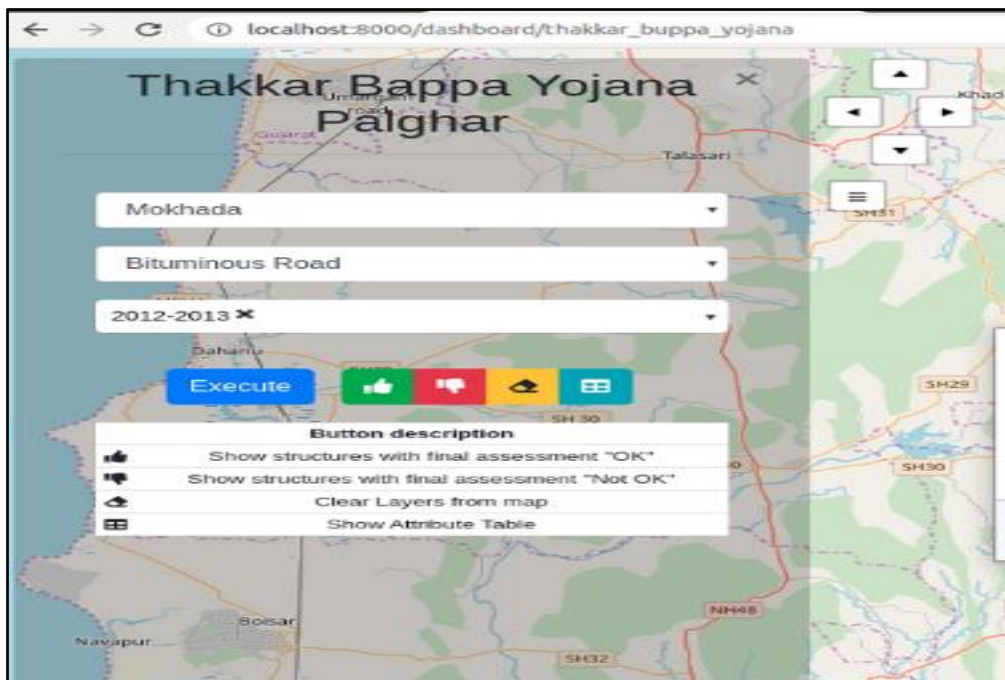


Fig: 6.1. New interface of TDSC CommunityGIS for the project of Thakkar Bappa Yojana

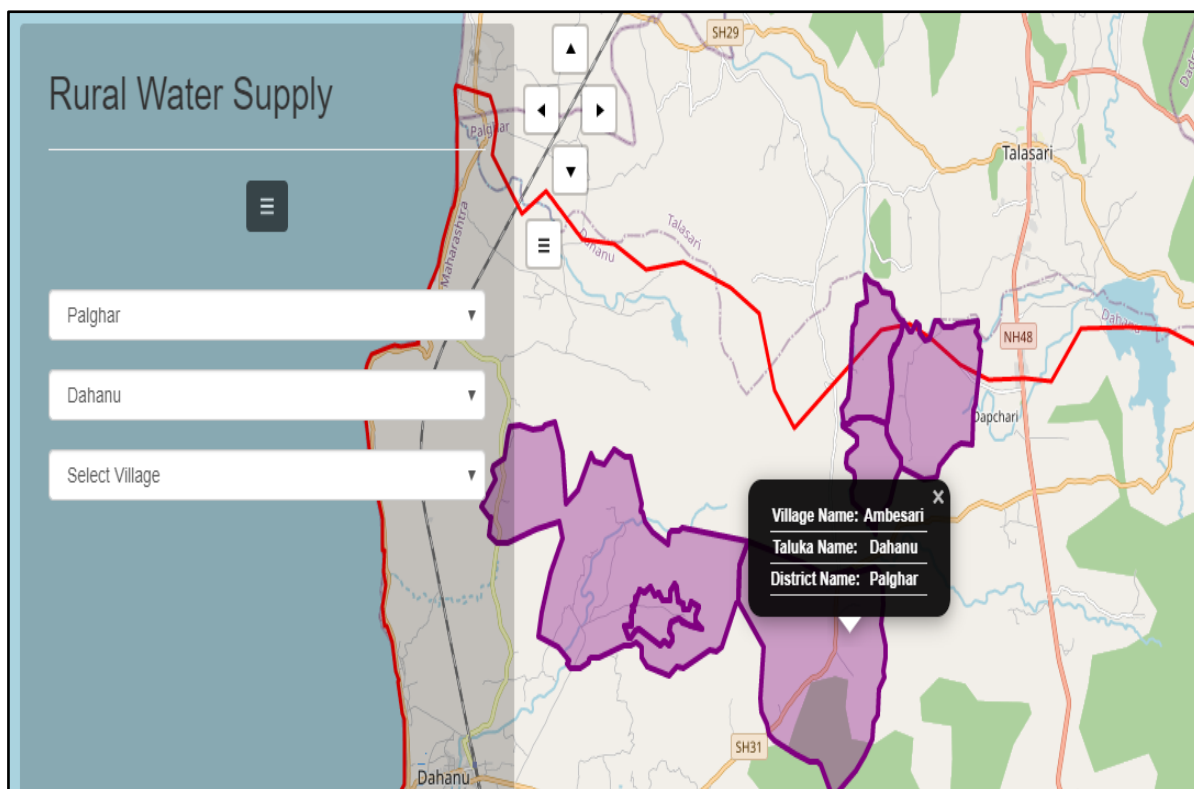


Fig: 6.2. Interface of TDSC CommunityGIS for the project of Rural Water Supply Scheme

2. Website Development of UMA:

Next project was to develop a website of Unnat Maharashtra Abhiyan. The website is ready and hosted at <http://104.211.185.20/>. The website is up and running. This website included working in CMS called Wagtail.



Fig: 6.3. UMA website home page

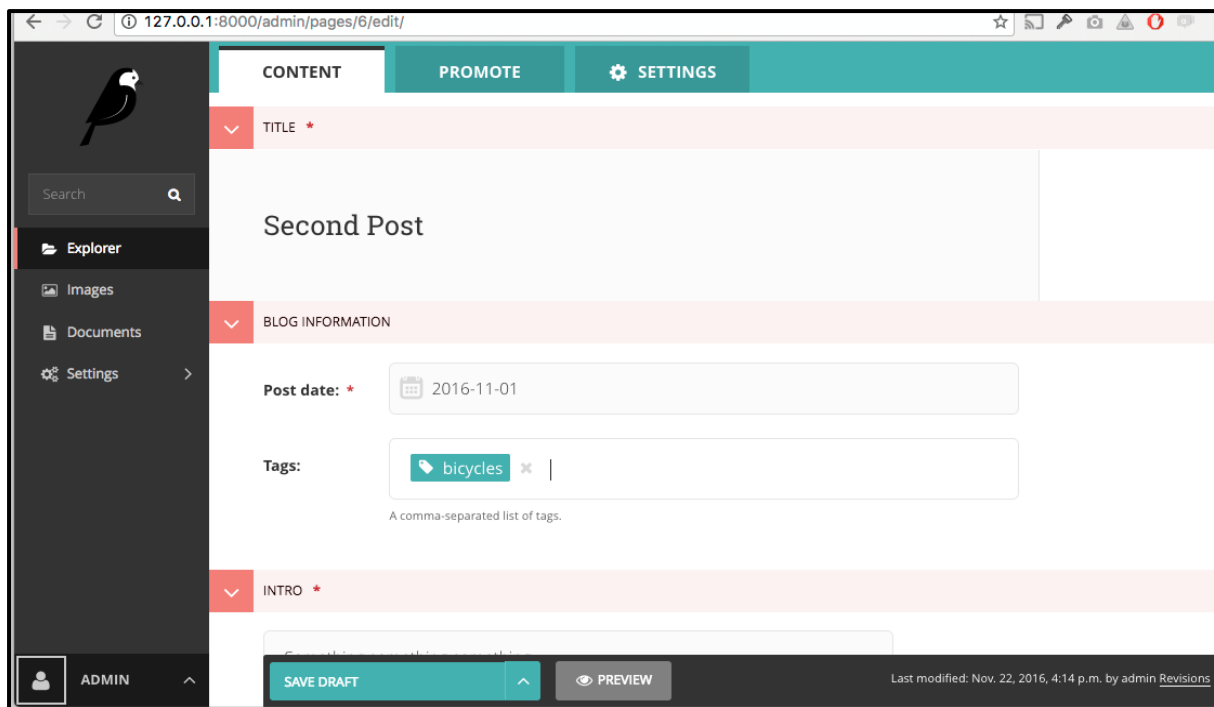


Fig: 6.4. Admin Interface of Wagtail

One of the interesting tasks of UMA website was to create a GIS system using the Google APIs to display various participating institutes of UMA and also to display if they are public or private. This was to be done by just having the names of the list of the institutes.

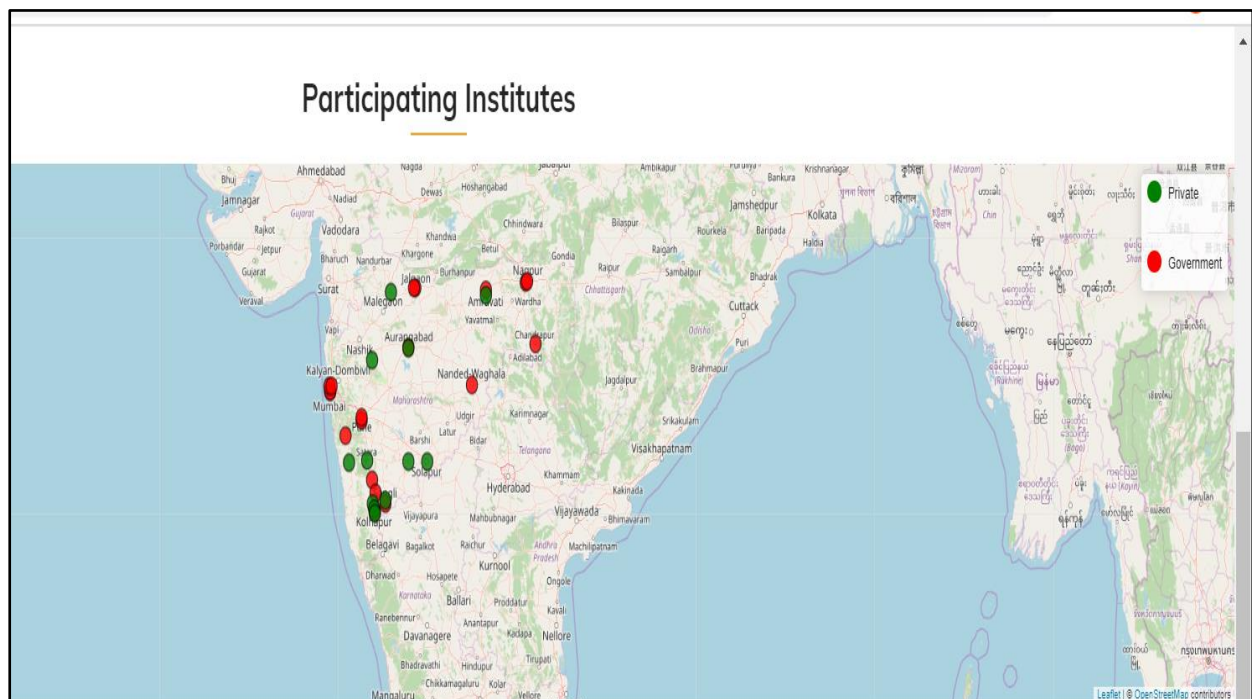


Fig: 6.5. Participating Institutes under UMA

3. Website Development of TDSC:



Fig: 6.6. Home Page of TDSC

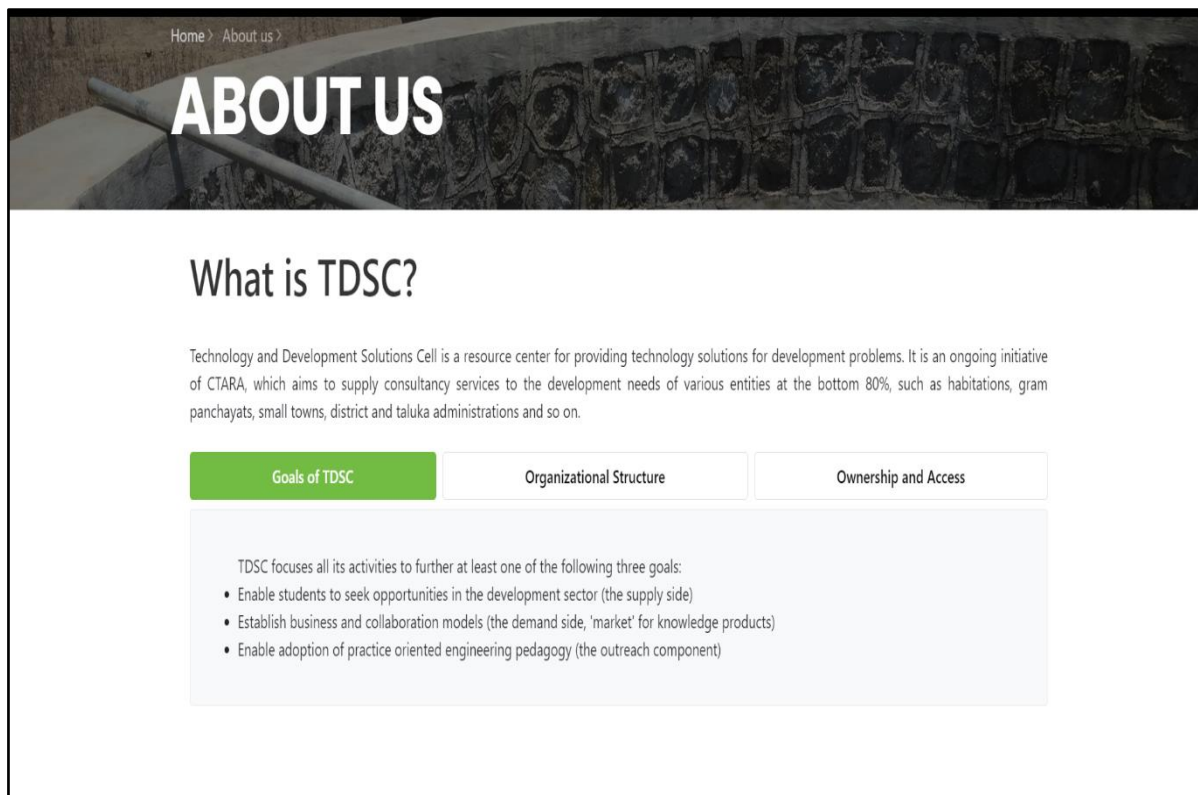


Fig: 6.7. About Us Page of TDSC

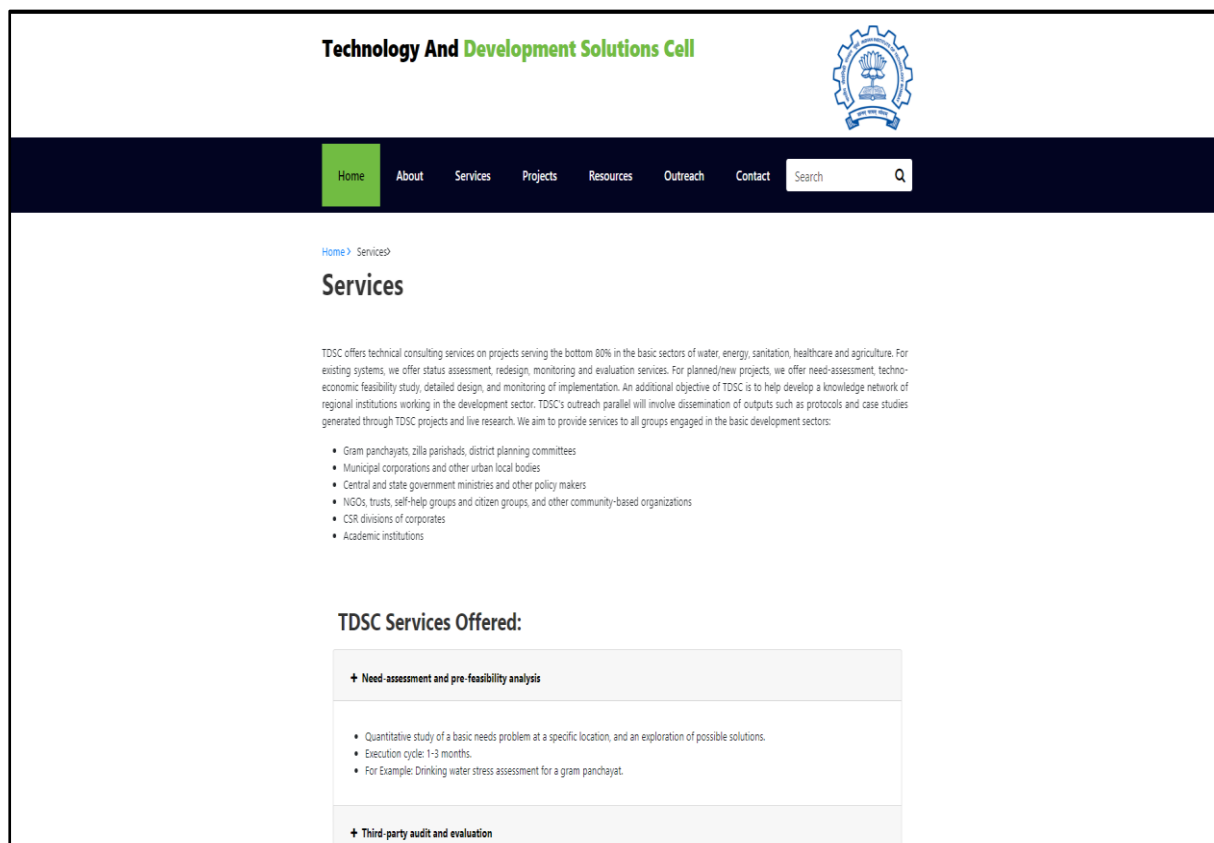


Fig: 6.8. Services Page of TDSC

Chapter 7

Summary and Conclusions

Overall, the experience of internship at IIT Bombay was very enriching. I got to learn new technologies and worked with a great team of people. Working with GIS and to be able to know how powerful it is in our day to day life was very contenting. I learned how TDSC visits site locations and collect data to make sure that correct results are obtained. While making the site for UMA, I understood the importance of such an organization and while working on the TDSC interface of CommunityGIS, I learned the importance of GIS based web system that how easily it can visualize the things that is otherwise difficult to explain.

On the technical aspect, I learned wagtail and understood the meaning of Content Management System and why is it so necessary these days. On the GIS part, I learned leaflet, GeoNode, GeoServer and data handling. One of my tasks was to maintain the current website of TDSC that helped me in learning the site maintenance.

To conclude, this internship helped me to work in a very positive environment. There was a lot of scope for me to learn new things and give suggestions. It helped me to grow multi-dimensionally.

Chapter 8

Future Scope

- In terms of application of GIS, the scope is unlimited. There are many fields that uses GIS based system these days. In many fields GIS is the future.
- This kind of a GIS system can bring revolution because it is targeting the development sector.
- In fields like Civil, the on-site work is very much and to note every single detail is also necessary. Such a system can help in proper reception of the data.
- The data collected during the field visits is in huge amount, the handling of such a data is very difficult. Such a system helps in not only the display of data but also helps in handling it by combining it as per the requirement.
- This system is being used by the Government of Maharashtra to check for the proper implementation of various schemes.
- GIS and location data are the future. Every single thing today works on the GPS, locations and maps. This project can thus help me in learning, understanding and working with GIS even more easy.

Chapter 9

Bibliography and References

- [1] en.wikipedia.com. (2020). *Geospatial Information System*. [online] Available at: https://en.wikipedia.org/wiki/Geographic_information_system [Accessed 8 Apr. 2020].
- [2] ctara.iitb.ac.in (2020). *TDSC CTARA*. [online] Available at: <http://www.ctara.iitb.ac.in/en/tdsc> [Accessed 8 Apr. 2020].
- [3] makerghat.urbansciences.in. (2020). *CommunityGIS*. [online] Available at: <https://makerghat.urbansciences.in/> [Accessed 8 Apr. 2020].
- [4] en.wikipedia.org. (2020). *Information Flow Diagram*. [online] Available at: https://en.wikipedia.org/wiki/Information_flow_diagram [Accessed 8 Apr. 2020].
- [5] tutorialpoint.com (2020). *Software Architecture Design*. [online] Available at: https://www.tutorialspoint.com/software_architecture_design/architecture_models.htm [Accessed 8 Apr. 2020].
- [6] en.wikipedia.org. (2020). *QGIS*. [online] Available at: <https://en.wikipedia.org/wiki/QGIS> [Accessed 8 Apr. 2020].
- [7] docs.geonode.org. (2020). *GeoNode Basics*. [online] Available at: <https://docs.geonode.org/en/master/start/quick/index.html#recommended-minimum-system-requirements> [Accessed 8 Apr. 2020].
- [8] leafletjs.com. (2020). *Leaflet Quick Start Guide*. [online] Available at: <https://leafletjs.com/examples/quick-start/> [Accessed 8 Apr. 2020].