

**KATHMANDU UNIVERSITY**  
**SCHOOL OF ENGINEERING**  
**DEPARTMENT OF GEOMATICS ENGINEERING**



**REPORT ON**  
**FINDING SHORTEST ROUTE FROM**  
**DHANGADHI TO KATHMANDU UNIVERSITY**  
**USING PgRouting**

Submitted by: Aarya Pant (029022-21)

Level: UNG/ III Year/ I Semester

14<sup>th</sup> June 2024

# Introduction

The problem of determining the shortest path is currently very interesting to discuss in everyday life. This problem is related to finding the shortest route and the minimum budgeted costs. Unintentionally, we experience this problem almost every day. For example, traveling from one place to another, telephone networks, electricity networks and so on. Another example is the online bikes and taxies services which is currently very popular in Nepal. Online bikes and taxies trips always look for the shortest path from the starting city point to the final destination city point. This is so that every online bike or taxi driver can quickly get customers according to the target set by the company every day. There it appears that information technology is increasingly advanced and sophisticated today. Therefore, many people who want to travel by looking for the shortest route or distance to reach the trip using technology (Alam & Faruq, 2019; CHOOSUMRONG et al., 2012; Nurhasanah et al., 2021; pgRouting Community, 2016; Pritee et al., 2019; Seeta, 2021; Singh et al., 2015). So, here we use Dijkstra's algorithm to find the shortest distance. Dijkstra's algorithm calculates the path based on the shortest distance travelled. For our project we also used PgRouting (an extension for PostgreSQL).

Geographic information systems (GISs) are becoming the most popular field in recent years. A GIS is an application or system which is designed to capture, storage, manipulation, analysis, and presentation of spatial or geographic information. Geographic location is the key term or information for the geographic information without which the data can't be said to be spatial or geographic. This project explores the working of QGIS and Databases technologies to find out the shortest path between two places (source and destination) in road network (geospatial data) using various algorithm approaches of PgRouting [extension of PostgreSQL database]. In this project, Dijkstra's algorithm is implemented to calculate the best and an optimal shortest path between two nodes(CHOOSUMRONG et al., 2012).

The purpose of this project is to find the shortest route from Dhangadhi to Kathmandu University using PgRouting. This study aims to provide an efficient route planning solution that can be utilized for various applications, including logistics and transportation.

## Dijkstra's Algorithm

Dijkstra's algorithm is a popular algorithm in computer science used for finding the shortest path between nodes in a graph. It's named after Dutch computer scientist Edsger W. Dijkstra, who developed it in 1956 (Alam & Faruq, 2019).

(Seeta, 2021) The result of Dijkstra's algorithm is the shortest path tree from the source node to all other reachable nodes in the graph, with the minimum distance from the source node to each node. It's often used in routing and network algorithms, such as finding the shortest route in transportation networks or the optimal path in communication networks. The working of this algorithm has been explained using six steps:

- I. Choose search region providing set S of vertices and select source and destination vertices.
- II. Initialize the source node to zero which is solved and identify rest unsolved node connected to source node which is assumed as origin.
- III. Calculate candidate distance by adding distance to the solved node and length of arc connecting between solved and unsolved nodes.
- IV. After that algorithm chooses smallest candidate distance.
- V. Then unsolved is changed into solved node.
- VI. If there will be tie in candidate distance, then choose arbitrary.
- VII. Repeat these steps with all unsolved node until the destination is reached.

## PgRouting

PgRouting is an extension for PostgreSQL, the open-source object-relational database system. It adds routing capabilities to PostgreSQL (one of the Structured Query Language specially used for Relational Databases), allowing users to perform routing queries and computations within the database. This extension is particularly useful for applications that require routing functionality, such as navigation systems, logistics, and transportation planning (Pritee et al., 2019). PgRouting provides a set of functions and algorithms for various routing tasks, including shortest path calculation, traveling salesman problem solving, and network analysis. It utilizes graph theory and algorithms like Dijkstra's algorithm and A\* search to find the most efficient paths between points on a network. With PgRouting, users can store and manage spatial data in PostgreSQL databases and perform routing computations directly within the database, leveraging the power and flexibility of SQL queries. This integration simplifies the development and maintenance of routing applications by centralizing data storage and processing (Singh et al., 2015).

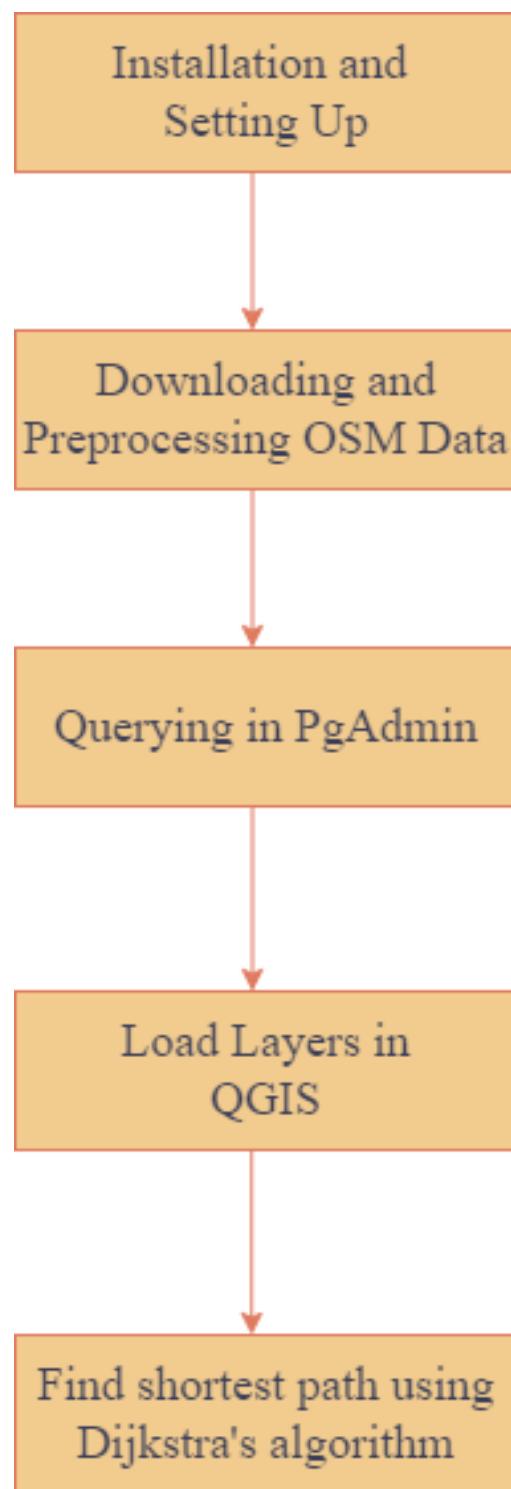
Advantages of the database routing approach are:

Data and attributes can be modified by many clients, like Quantum GIS and uDig through JDBC, ODBC, or directly using PL/pgSQL. The clients can either be PCs or mobile devices. Data changes can be reflected instantaneously through the routing engine. There is no need for pre calculation. The “cost” parameter can be dynamically calculated through SQL and its value can come from multiple fields or tables. PgRouting provides functions for: Shortest Path Dijkstra: routing algorithm without heuristics; Shortest Path A-Star: routing for large datasets (with heuristics); Shortest Path Shooting-Star: routing with turn restrictions (with heuristics), Traveling Salesperson Problem, Driving Distance Calculation (Isolines) (Nurhasanah et al., 2021).

This project encompasses the development and implementation of an efficient route planning solution using Geographic Information Systems (GIS) and PgRouting, a powerful extension of the PostgreSQL database. The scope includes the acquisition and management of road network data within QGIS, the application of Dijkstra's algorithm via PgRouting to determine the shortest path between Dhangadhi and Kathmandu University, and the evaluation of this method's performance in dynamic, real-world scenarios. By leveraging dynamic cost calculations, this project aims to adapt to real-time variations in the road network, such as traffic conditions and road closures. The outcome will be a robust and scalable solution for optimal route planning, applicable to various fields including transportation, demonstrating the practical benefits of integrating advanced geospatial data analysis with database management systems.

The study area for this project is the road network between Dhangadhi and Kathmandu University in Nepal. This region encompasses a diverse range of geographic and infrastructural features, making it an ideal candidate for evaluating the effectiveness of GIS-based routing solutions. Dhangadhi is located in the farwestern part of Nepal, Dhangadhi is one of the major cities in the Sudurpashchim Province, characterized by a mix of urban and semi-urban road networks whereas Kathmandu University is situated in Dhulikhel, approximately 30 kilometers east of Kathmandu, the university represents a significant endpoint in the hilly terrain of central Nepal.

## Workflow



## Description

### 1. Database download:

The installation of PostgreSQL had to be done. PostGIS is a spatial extension for PostgreSQL. It can be installed using the link provided above. Also, After installing PostgreSQL on your system, PostGIS can be installed using Stack Builder Application. PgRouting is included with the PostGIS bundle.

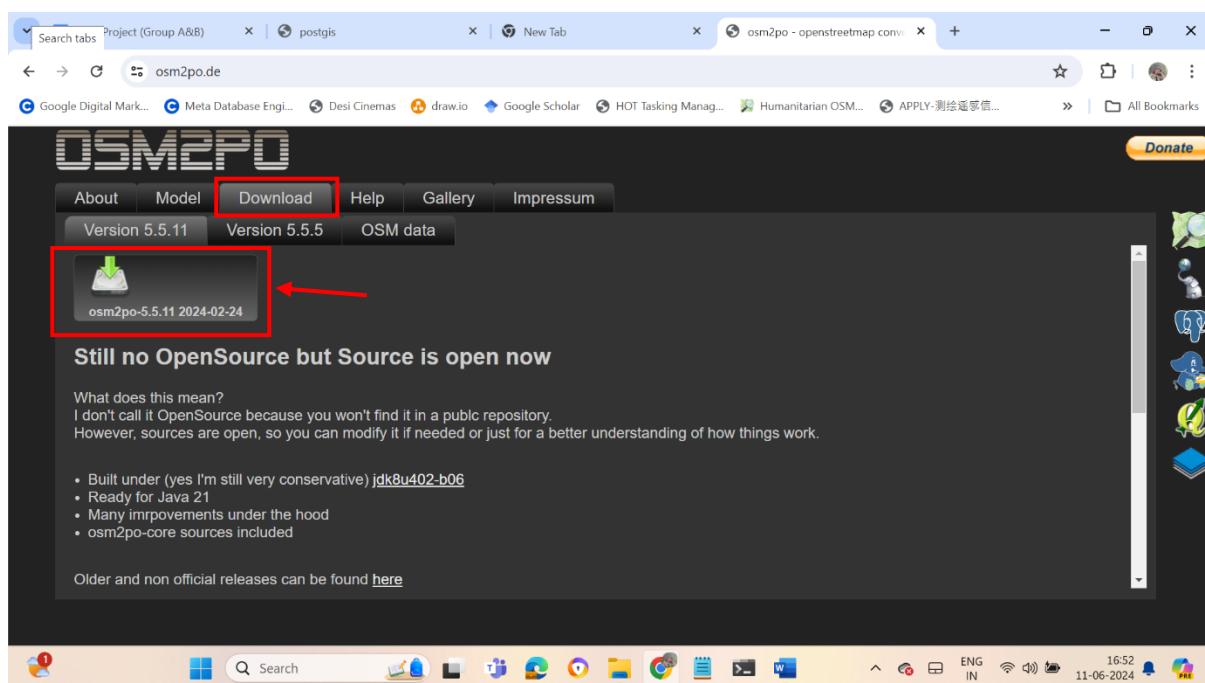
### 2. GIS:

This Open-Source GIS applicaton was to be downloaded.

### 3. osm2po:

The osm2po package was downloaded from the link provided by the lecturer. This package takes as input the raw OSM data (eg. .PBF format) and converts the data into a SQL file which can be imported into our Spatial Database and used for the pgRouting implementation. The following steps can be followed for the installation of osm2po.

- Visit: <https://osm2po.de/>
- Download: osm2po-5.5.11

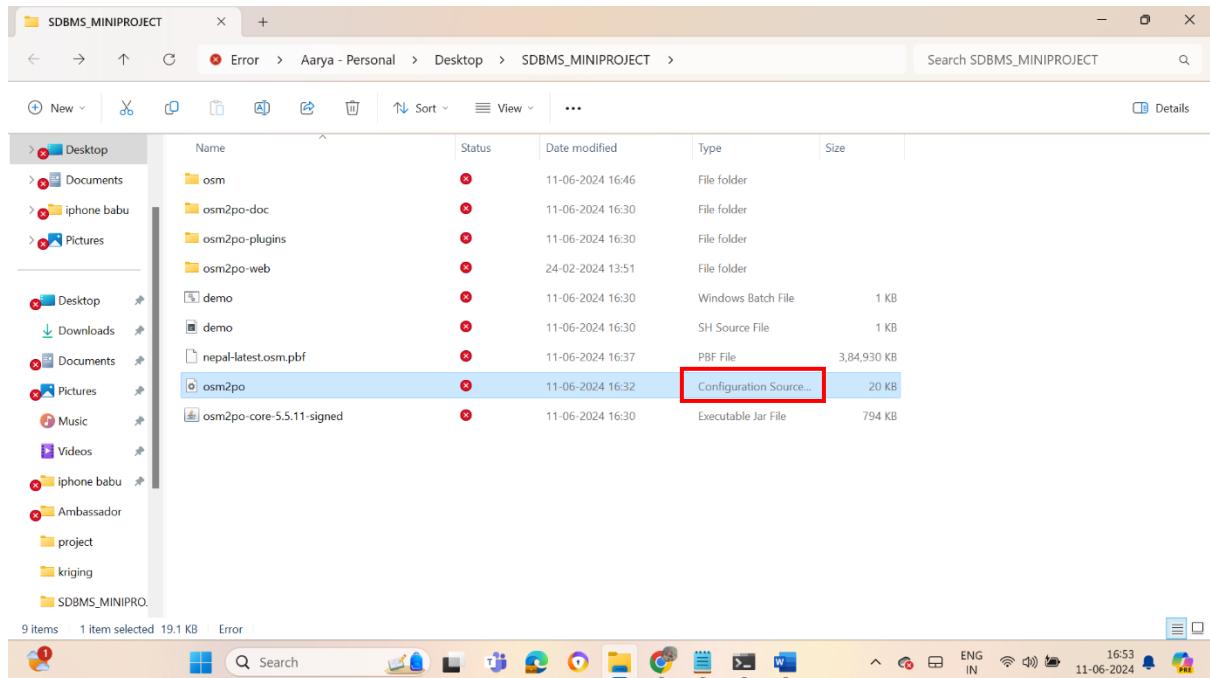


### 4. Extracting the zipped file:

- Extract files from: osm2po-5.5.11

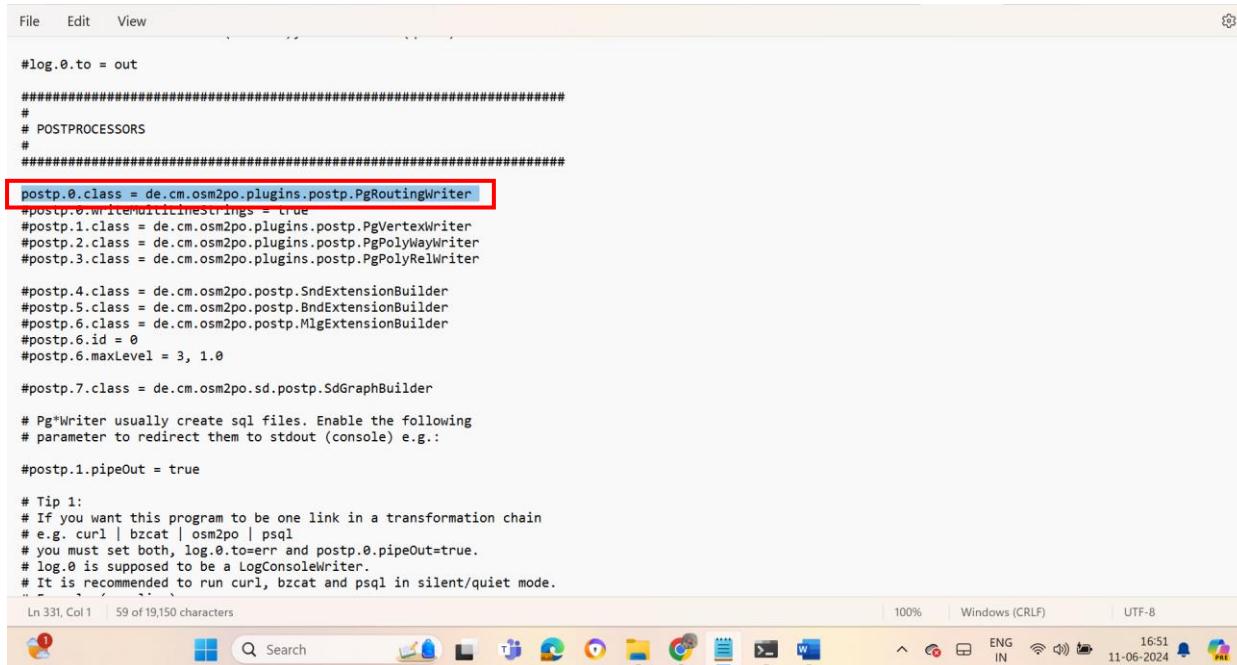
## 5. Editing the data

- osm2po.config was to be opened using Notepad



Editing was to be done on:

- Line 330: Removed # in the beginning
- Line 179: wtr.finalMask = car,foot,bike



```
File Edit View

#log.0.to = out
#####
#
# POSTPROCESSORS
#
#####
postp.0.class = de.cm.osm2po.plugins.postp.PgRoutingWriter
#postp.0.writeMultiLineString = true
#postp.1.class = de.cm.osm2po.plugins.postp.PgVertexWriter
#postp.2.class = de.cm.osm2po.plugins.postp.PgPolyWayWriter
#postp.3.class = de.cm.osm2po.plugins.postp.PgPolyRelWriter

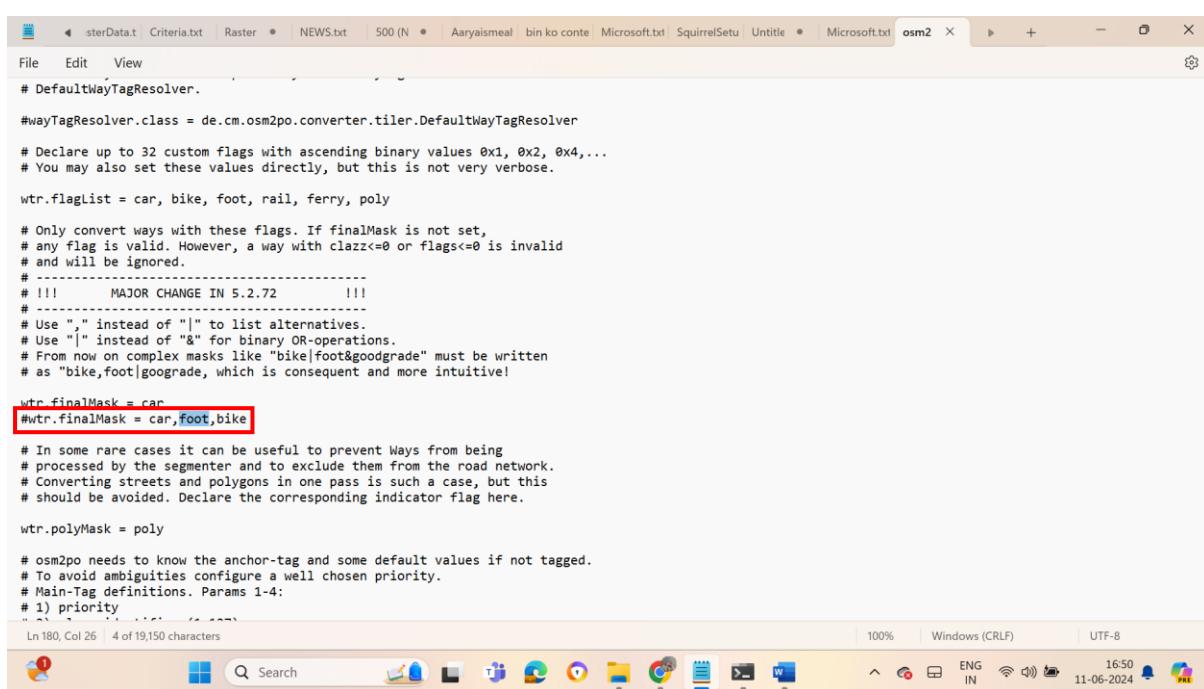
#postp.4.class = de.cm.osm2po.postp.SndExtensionBuilder
#postp.5.class = de.cm.osm2po.postp.BndExtensionBuilder
#postp.6.class = de.cm.osm2po.postp.MlgExtensionBuilder
#postp.6.id = 0
#postp.6.maxLevel = 3, 1.0

#postp.7.class = de.cm.osm2po.sd.postp.SdGraphBuilder

# Pg*Writer usually create sql files. Enable the following
# parameter to redirect them to stdout (console) e.g.:
#postp.1.pipeOut = true

# Tip 1:
# If you want this program to be one link in a transformation chain
# e.g. curl | bzip2 | osm2po | psql
# you must set both, log.0.to=err and postp.0.pipeOut=true.
# log.0 is supposed to be a LogConsoleWriter.
# It is recommended to run curl, bzip2 and psql in silent/quiet mode.

Ln 331, Col 1 | 59 of 19,150 characters | 100% | Windows (CRLF) | UTF-8
```



```
File Edit View
# DefaultWayTagResolver.

#wayTagResolver.class = de.cm.osm2po.converter.tiler.DefaultWayTagResolver

# Declare up to 32 custom flags with ascending binary values 0x1, 0x2, 0x4,...
# You may also set these values directly, but this is not very verbose.

wtr.flagList = car, bike, foot, rail, ferry, poly

# Only convert ways with these flags. If finalMask is not set,
# any flag is valid. However, a way with clazz<=0 or flags<=0 is invalid
# and will be ignored.
#
# -----
# !!!      MAJOR CHANGE IN 5.2.72      !!!
# -----
# Use "," instead of ";" to list alternatives.
# Use "|" instead of "&" for binary OR-operations.
# From now on complex masks like "bike|foot&goodgrade" must be written
# as "bike,foot|goodgrade", which is consequent and more intuitive!

wtr.finalMask = car
#wtr.finalMask = car,foot,bike

# In some rare cases it can be useful to prevent Ways from being
# processed by the segmenter and to exclude them from the road network.
# Converting streets and polygons in one pass is such a case, but this
# should be avoided. Declare the corresponding indicator flag here.

wtr.polyMask = poly

# osm2po needs to know the anchor-tag and some default values if not tagged.
# To avoid ambiguities configure a well chosen priority.
# Main-Tag definitions. Params 1-4:
# 1) priority ...
```

## 6. Pre-processing OSM Data in osm2po:

- Open Command Prompt and type the following command:

```
C:\Users\asus>java -jar C:\Users\asus\OneDrive\Desktop\SDBMS_MINIPROJECT\osm2po-core-5.5.11-signed.jar cmd=c prefix=nep C:\Users\asus\OneDrive\Desktop\SDBMS_MINIPROJECT\nepal-latest.osm.pdf
```

```
Microsoft Windows [Version 10.0.22631.3593]
(c) Microsoft Corporation. All rights reserved.

C:\Users\asus>java -jar C:\Users\asus\OneDrive\Desktop\SDBMS_MINIPROJECT\osm2po-core-5.5.11-signed.jar cmd=c prefix=nep C:\Users\asus\OneDrive\Desktop\SDBMS_MINIPROJECT\nepal-latest.osm.pdf

v.5.5.11

OpenStreetMap-Data to Topology Converter with integrated RoutingEngine.
(c) 2024 - Carsten Moeller, info@osm2po.de, Pinneberg, Germany

Licence: This software is FreeWare without nuisance. You are free to
make copies, give exact copies of the original to anyone, distribute
it in its unmodified form via electronic means. You may not reverse
engineer, de-compile or disassemble it, rent, lease, lend or sell it.
This software is provided 'AS IS', without warranty of any kind,
so use it at your own risk.

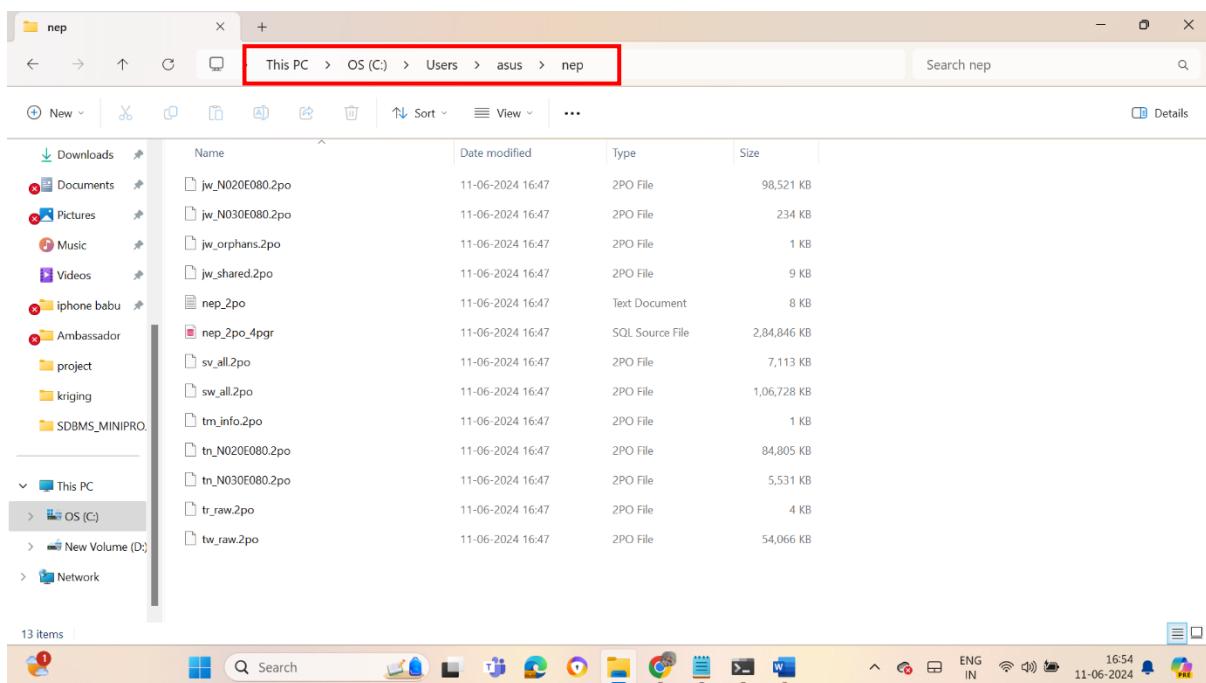
INFO  Reading Configuration from file:/C:/Users/asus/OneDrive/Desktop/SDBMS_MINIPROJECT/osm2po.config
INFO  Plugin /C:/Users/asus/OneDrive/Desktop/SDBMS_MINIPROJECT/osm2po-plugins/osm2po-plugins-5.5.11.jar loaded
INFO  Running osm2po 5.5.11 with cmd=tjsp - 1,892M
INFO  Java 19.0.2 (Oracle Corporation)
INFO  Starting Tiler at Tue Jun 11 16:47:03 NPT 2024
INFO  Reading from file:/C:/Users/asus/OneDrive/Desktop/SDBMS_MINIPROJECT/nepal-latest.osm.pbf
```

- Navigate to the converted file.

```
... 100,000 of 209,307 ways segmented (N020E080) - 1,4... 200,000 of 209,307 ways segmented (N020E080) - 1,4INFO
209,307 ways analyzed, 397,110 segments created (N020E080) - 1,452M
INFO 330,756 vertices of 6,187,065 nodes written - 1,432M
INFO 15,517 of 15,717 WayNodes cached (N030E080) - 1,867M
INFO 190 ways analyzed, 241 segments created (N030E080) - 1,865M
INFO 231 vertices of 15,517 nodes written - 1,865M
INFO 4 ways analyzed, 4 segments created (SHARED) - 1,865M
INFO 8 vertices of 571 nodes written - 1,865M
INFO Segmenter finished at Tue Jun 11 16:47:52 NPT 2024
INFO Starting PostProcessor[0] at Tue Jun 11 16:47:52 NPT 2024
de.cm.osm2po.plugins.postp.PgRoutingWriter
INFO Creating sql file nep\nepl2po_4pgr.sql
INFO 397,355 Segments written.
INFO commandline template:
psql -U [username] -d [dbname] -q -f "C:\Users\asus\nepl2po_4pgr.sql"
INFO PostProcessor finished at Tue Jun 11 16:47:58 NPT 2024
INFO Config closed at 240611-16:47.58944

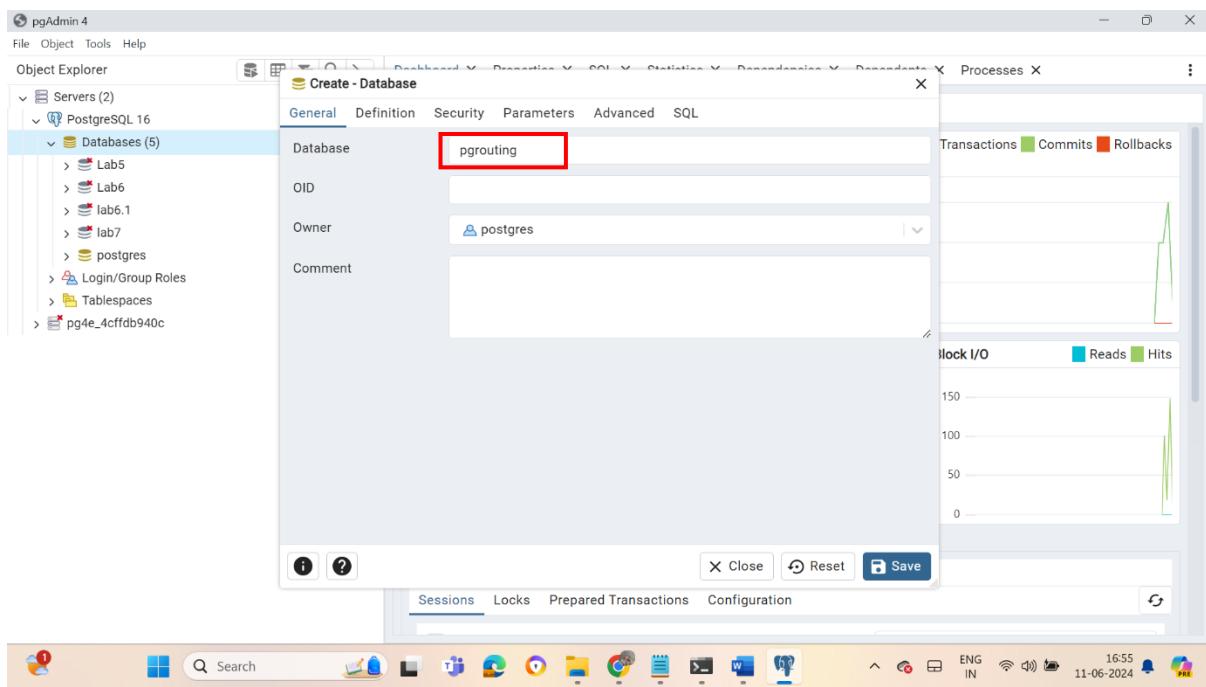
C:\Users\asus>
```

- Navigate to the converted file in Windows Explorer.

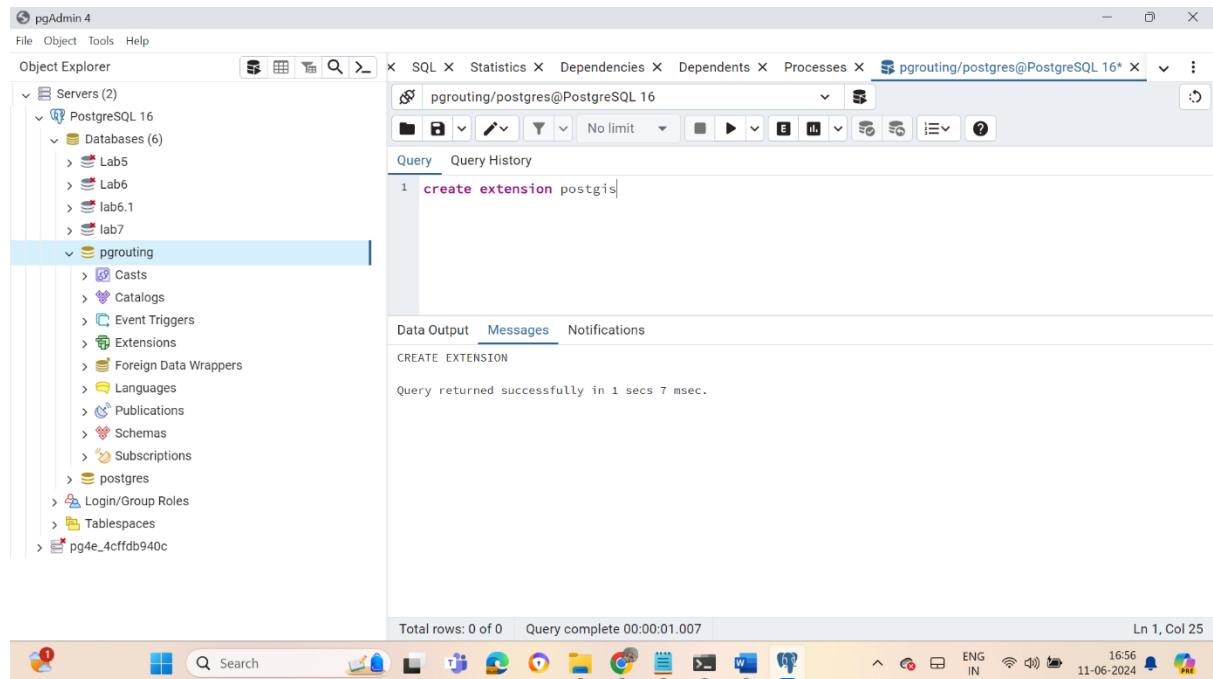


## 7. pgAdmin:

- Create Database pgRouting.

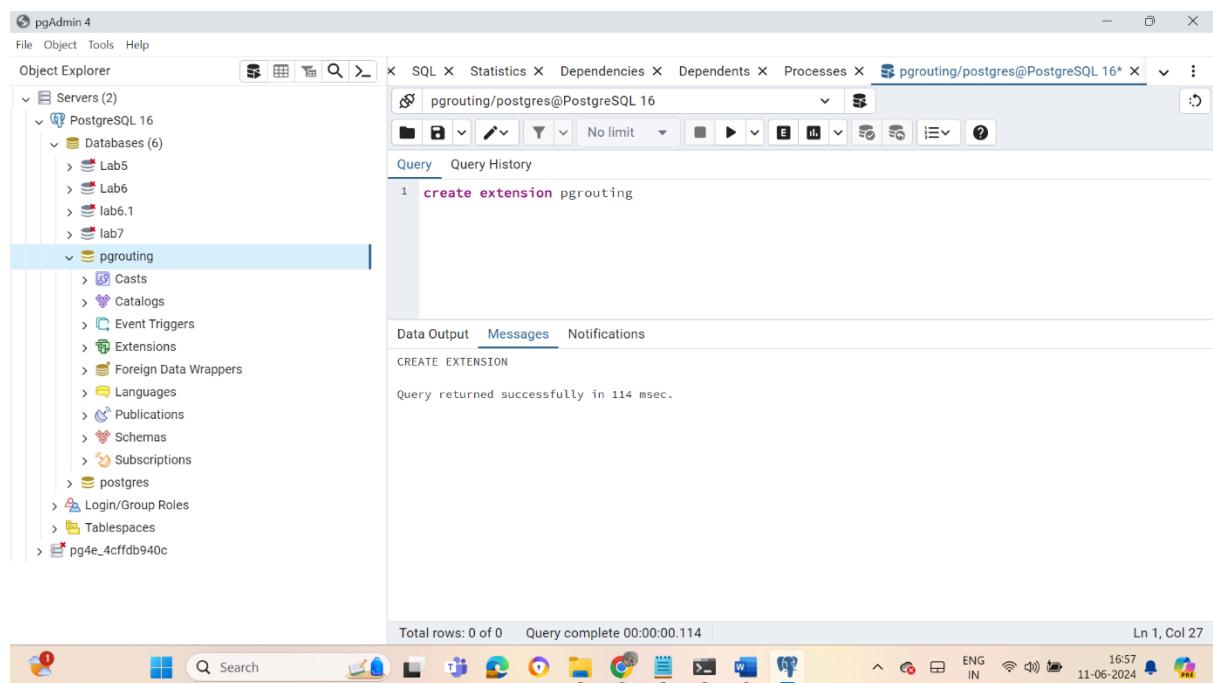


- Create Extension postgis.



The screenshot shows the pgAdmin 4 interface. In the Object Explorer, under the 'Servers' section, 'PostgreSQL 16' is selected. Within 'PostgreSQL 16', the 'Databases' folder contains six databases: Lab5, Lab6, lab6.1, lab7, postgres, and pg4e\_4cffdb940c. Under the 'pgRouting' folder, there are various objects like Casts, Catalogs, Event Triggers, Extensions, Foreign Data Wrappers, Languages, Publications, Schemas, Subscriptions, and Postgres. The 'SQL' tab is active, showing the command: `1 create extension postgis;`. The 'Messages' tab shows the output: `CREATE EXTENSION` and `Query returned successfully in 1 secs 7 msec.`. The status bar at the bottom indicates 'Total rows: 0 of 0 Query complete 00:00:01.007 Ln 1, Col 25'. The taskbar at the bottom of the screen shows various application icons.

- Create Extension pgRouting.



The screenshot shows the pgAdmin 4 interface. The setup is identical to the previous one, with 'PostgreSQL 16' selected in the Object Explorer. The 'pgRouting' folder contains the same set of objects. The 'SQL' tab is active, showing the command: `1 create extension pgRouting;`. The 'Messages' tab shows the output: `CREATE EXTENSION` and `Query returned successfully in 114 msec.`. The status bar at the bottom indicates 'Total rows: 0 of 0 Query complete 00:00:00.114 Ln 1, Col 27'. The taskbar at the bottom of the screen shows various application icons.

- Open .sql file converted earlier and execute.

The screenshot shows two instances of pgAdmin 4 running on a Windows desktop. Both instances are connected to a PostgreSQL 16 server named 'PostgreSQL 16'. A red arrow points from the left instance to the right instance, highlighting the 'Processes' tab in the top bar of the right window.

**Left Instance (Object Explorer):**

- Servers: PostgreSQL 16
- Databases: Lab5, Lab6, lab6.1, lab7, pgrouting, nep\_2po\_4pgr
- File Explorer: Shows files like 'kriging', 'SDBMS\_MINIPR', 'OS (C)', 'New Volume (D)', and 'Network'.
- SQL File: nep\_2po\_4pgr.sql (selected)

**Right Instance (Object Explorer):**

- Servers: PostgreSQL 16
- Databases: Lab5, Lab6, lab6.1, lab7, pgrouting, nep\_2po\_4pgr, pg4e\_4cffdb940c
- Processes: pgRouting/postgres@PostgreSQL 16 (selected)
- Query History: Shows the execution of 'CREATE EXTENSION pgRouting'.
- Messages: Shows the message 'Query returned successfully in 114 msec.'

**Bottom Status Bar:**

- Total rows: 0 of 0
- Query complete 00:00:00.114
- Ln 1, Col 1
- 16:58 11-06-2024 ENG IN

pgAdmin 4

File Object Tools Help

Object Explorer

Servers (2)

- PostgreSQL 16
  - Databases (6)
    - Lab5
    - Lab6
    - lab6.1
    - lab7
    - pgrouting**
    - Cast
    - Catalogs
    - Event Triggers
    - Extensions
    - Foreign Data Wrappers
    - Languages
    - Publications
    - Schemas
    - Subscriptions
    - postres
    - Login/Group Roles
    - Tablespaces

ard X Properties X SQL X Statistics X Dependencies X Dependents X Processes X nep\_2po\_4pgr.sql X

pgrouting/postgres@PostgreSQL 16

Query Query History

```

1 -- Created by : osm2po-core
2 -- Version   : 5.5.11
3 -- Author (c) : Carsten Moeller - info@osm2po.de
4 -- Date      : Tue Jun 11 16:47:52 NPT 2024
5
6 SET client_encoding = 'UTF8';
7
8 DROP TABLE IF EXISTS nep_2po_4pgr;
9 -- SELECT DropGeometryTable('nep_2po_4pgr');
10
11 CREATE TABLE nep_2po_4pgr(id integer, osm_id bigint, osm_name character varying, osm_meta character varying);
12 SELECT AddGeometryColumn('nep_2po_4pgr', 'geom_way', 4326, 'LINESTRING', 2);
13

```

Data Output Messages Notifications

addgeometrycolumn	text

Total rows: 0 of 0    Query complete 00:01:35.534    Ln 1, Col 1

pgAdmin 4

File Object Tools Help

Object Explorer

Servers (2)

- PostgreSQL 16
  - Databases (6)
    - Lab5
    - Lab6
    - lab6.1
    - lab7
    - pgrouting**
    - Cast
    - Catalogs
    - Event Triggers
    - Extensions
    - Foreign Data Wrappers
    - Languages
    - Publications
    - Schemas
    - Subscriptions
    - postres
    - Login/Group Roles
    - Tablespaces

SQL X Statistics X Dependencies X Dependents X Processes X pgRouting/postgres@PostgreSQL 16\* X

Query Query History

```

1 select * from public.nep_2po_4pgr
2 order by id ASC

```

Data Output Messages Notifications

	id [PK] integer	osm_id bigint	osm_name character varying	osm_meta character varying	osm_source_id bigint	osm_target_id bigint	clazz integer
1	1	4825621	F26	[null]	38921333	3514581859	
2	2	4925621	F26	[null]	3514581859	2169105896	
3	3	4825621	F26	[null]	2169105896	1280107732	
4	4	4825621	F26	[null]	1280107732	38921343	
5	5	4825630	कान्ति पथ	[null]	31019141	2126598729	
6	6	4825630	कान्ति पथ	[null]	2126598729	1273136891	
7	7	4825671	धावाई सादक	[null]	268301866	4723674796	
8	8	4839933	[null]	[null]	7070924827	3339466444	
9	9	4839936	[null]	[null]	5477725560	31147572	
10	10	4839958	[null]	[null]	31147586	31147591	
11	11	4840030	बरीसपुत्री मार्ग	[null]	4879891022	2201363919	
12	12	4840030	बरीसपुत्री मार्ग	[null]	2201363919	1835392411	

Total rows: 1000 of 397355    Query complete 00:00:03.774    Ln 2, Col 16

pgAdmin 4

File Object Tools Help

Object Explorer

- Servers (2)
  - PostgreSQL 16
    - Databases (6)
      - Lab5
      - Lab6
      - lab6.1
      - lab7
- pgrouting
  - Casts
  - Catalogs
  - Event Triggers
  - Extensions
  - Foreign Data Wrappers
  - Languages
  - Publications
  - Schemas
  - Subscriptions
  - postgres
  - Login/Group Roles
  - Tablespaces
- pg4e\_4cffdb940c

SQL X Statistics X Dependencies X Dependents X Processes X pgRouting/postgres@PostgreSQL 16\*

Query Query History

```
1 select * from public.nep_2po_4pgr
2 order by id ASC
```

Data Output Messages Geometry Viewer X Notifications

	x	y	precision	y2	double precision	geom
1	85.3854271	27.7221233	0102000020E6100000030000017E41D96BC5855409CBE43F6E6B83B40170FEF39B0585540042C0318E0E			
2	85.3846481	27.7220403	0102000020E6100000020000007F5F6DD6AA585540FA2D9512DBB83B40FD7D10139E585540DFE412A2D7			
3	85.382829	27.7218308	0102000020E610000003000000FD7D10139E585540DFE412A2D7B83B404873BF9EB5A585540195EA4ABC			
4	85.3827653	27.7218239	0102000020E61000000200000077B0344580585540DA3E3FCE7C9883B40CB62073A7F5855409FF87B73C9			
5	85.3143478	27.7084885	0102000020E6100000040000069684D3D1C545540FF55EC041B53B404A7A185A1D5455408B1A06D3E			
6	85.3143863	27.7085323	0102000020E61000000300000073243C461E545540941799805FB53B402CBAF59A1E5455403F41182E61E			
7	85.3113434	27.7143814	0102000020E610000007000000EF007AA3FE535540AF6076AAD7B63B40CFEA319EF753554013BE52F1DA			
8	85.3562535	27.7007657	0102000020E610000007000000596C9FC6565540764F1E166AB33B40E73AE86DC7565540D69F0E7569F			
9	85.3537169	27.7006099	0102000020E610000007000000D08DFA35A35655407A1DCC8179B33B400CD759E3A25655407C71F3D77E			
10	85.3536764	27.7011217	0102000020E610000006000000ECE44389C5655404190CD9F80B33B40F4DFEE9D565540BF9044847F			
11	85.3426811	27.7064885	0102000020E6100000050000002D10F99FB5555402935C52D30B53B4041C6CFE2F555554063C5BA021B			
12	85.3422499	27.7055176	0102000020E610000003000000445DB57CEE55554006802A6EDCB43B40584AE020EB555540F23515F3BD			

Total rows: 1000 of 397355 Query complete 00:00:02.285 Ln 1, Col 1

pgAdmin 4

File Object Tools Help

Object Explorer

- Servers (2)
  - PostgreSQL 16
    - Databases (6)
      - Lab5
      - Lab6
      - lab6.1
      - lab7
- pgrouting
  - Casts
  - Catalogs
  - Event Triggers
  - Extensions
  - Foreign Data Wrappers
  - Languages
  - Publications
  - Schemas
  - Subscriptions
  - postgres
  - Login/Group Roles
  - Tablespaces
- pg4e\_4cffdb940c

SQL X Statistics X Dependencies X Dependents X Processes X pgRouting/postgres@PostgreSQL 16\*

Query Query History

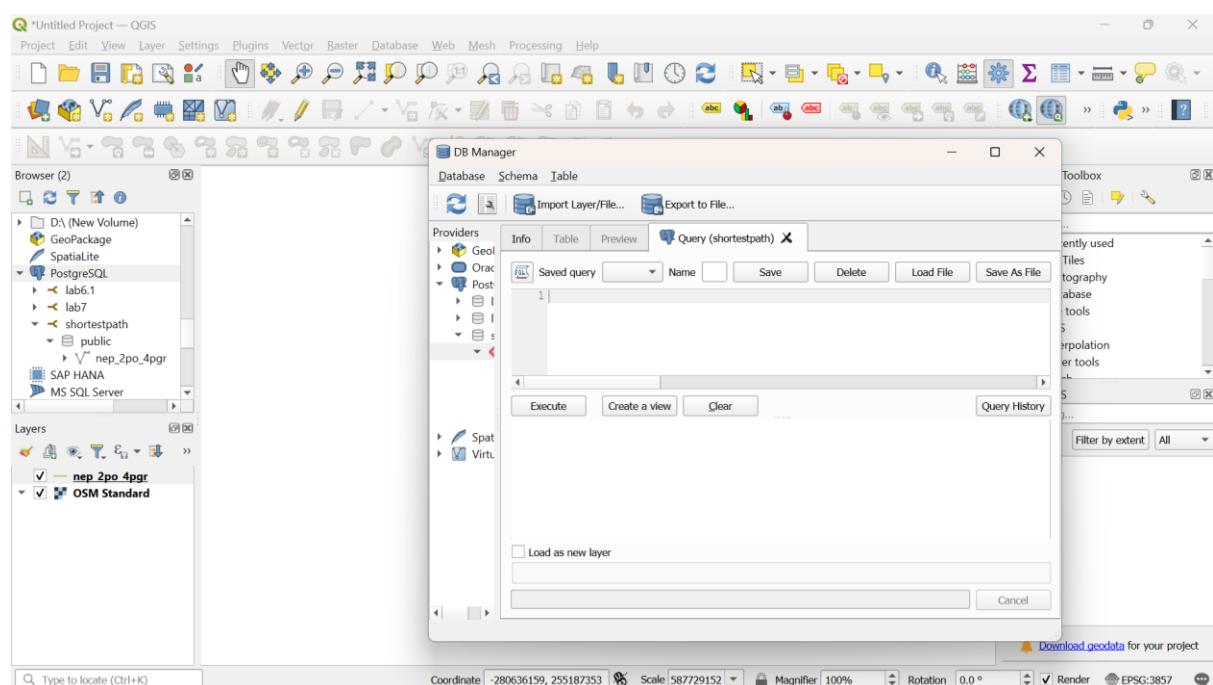
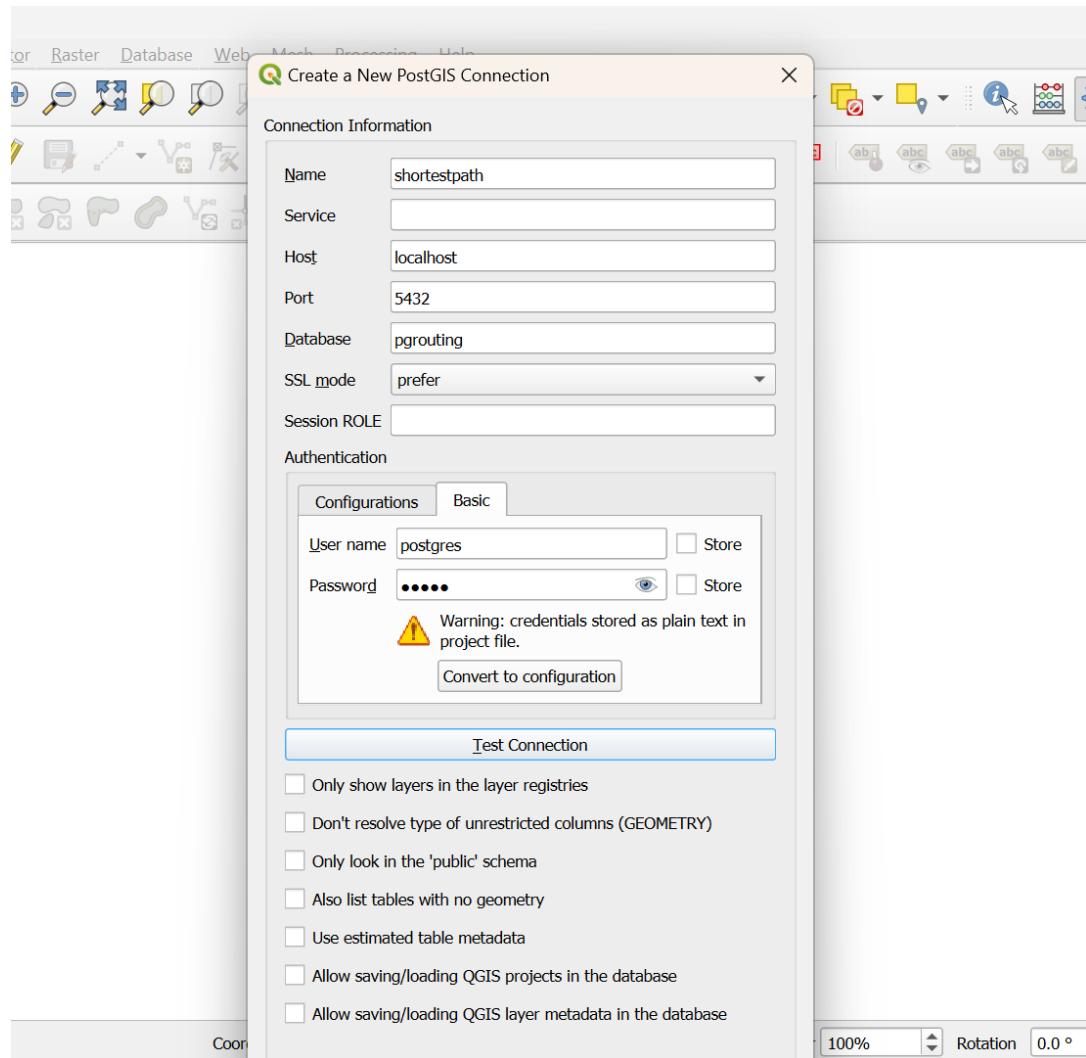
```
1 select * from public.nep_2po_4pgr
2 order by id ASC
```

Data Output Messages Geometry Viewer X Notifications

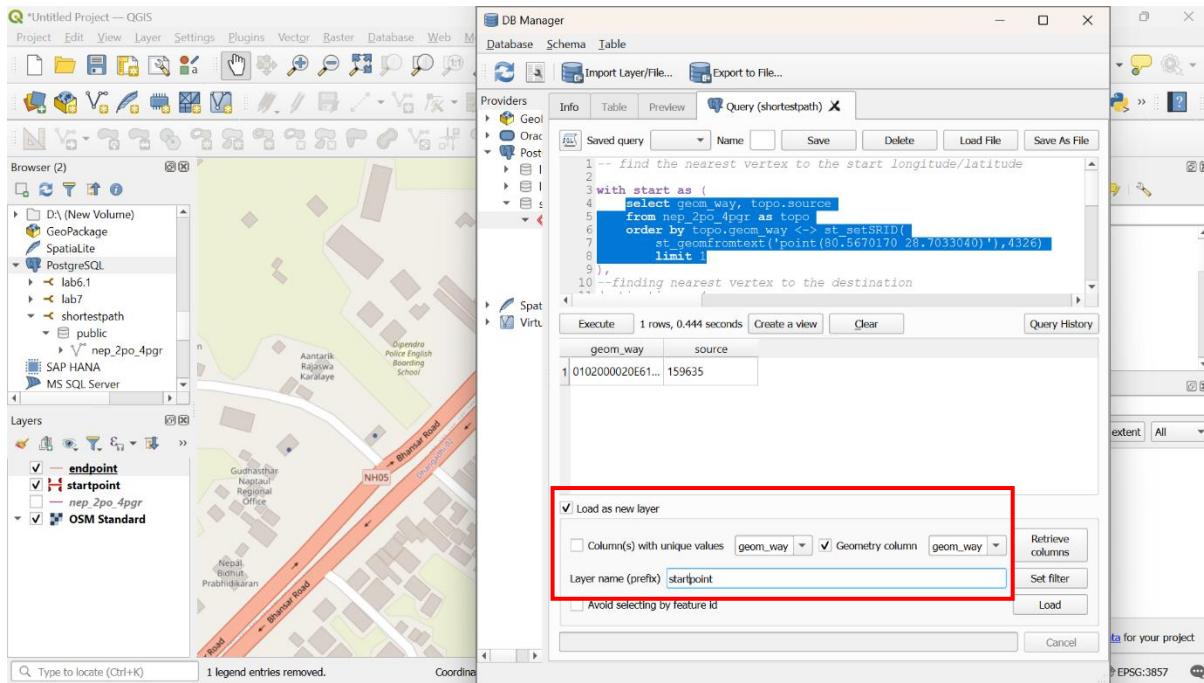
Total rows: 1000 of 397355 Query complete 00:00:02.285 Ln 1, Col 1

## 8. QGIS

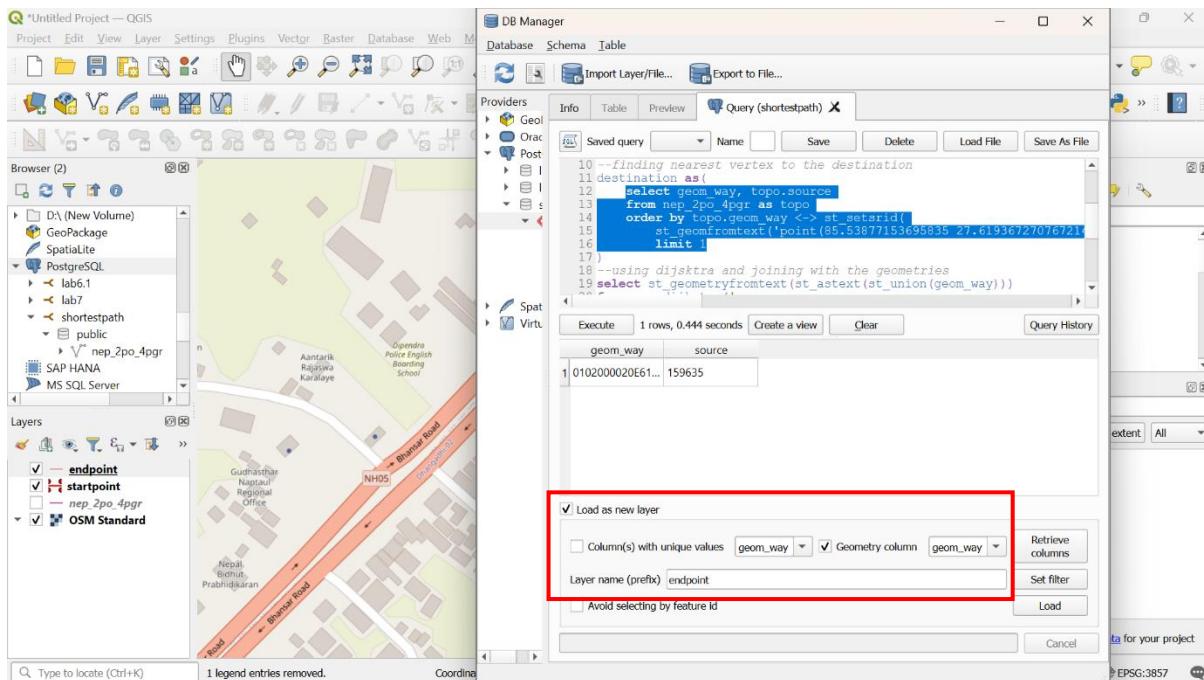
- Open QGIS and Connect to database pgrouting created earlier.

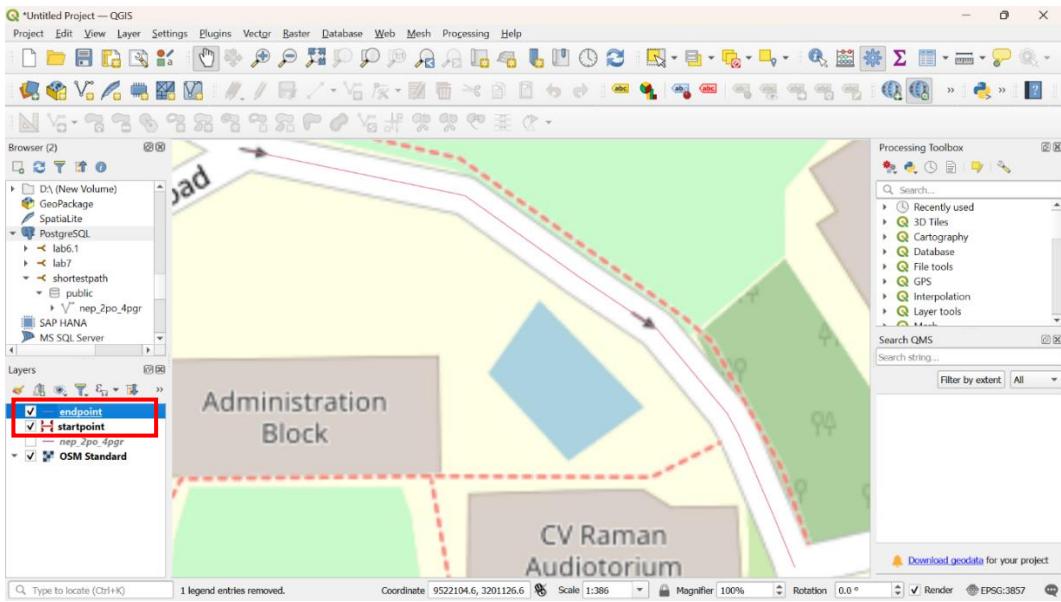


- We find the nearest vertex to start
- Origin: Dhangadhi
- While executing query, we also checked the ‘load as new layer’ option and named the layer ‘startpoint’.

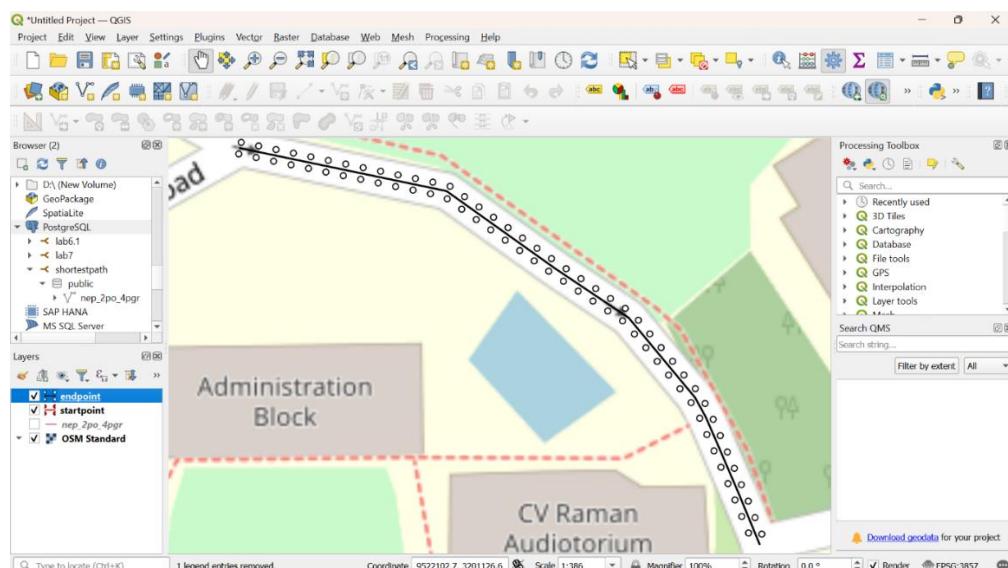
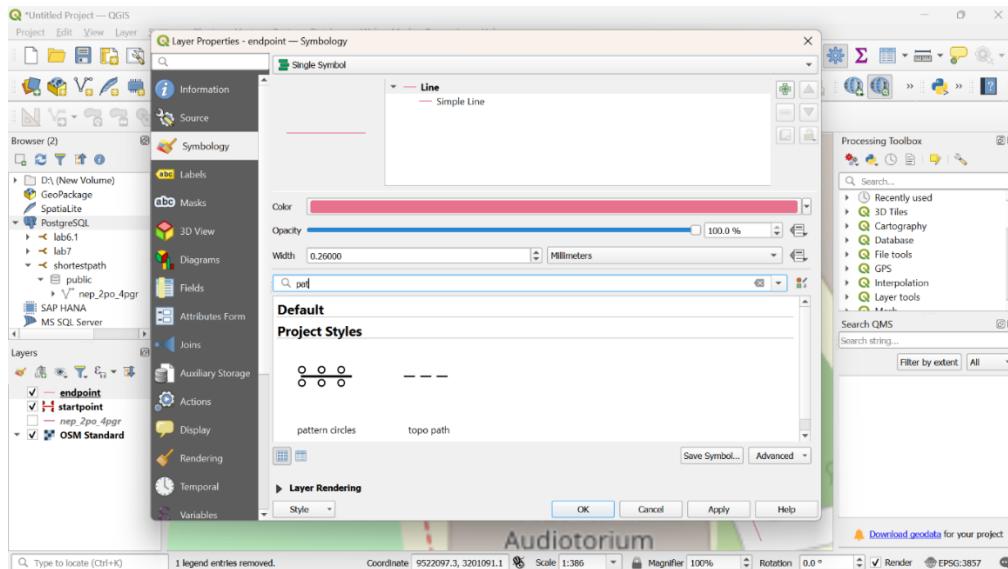


- Destination: Kathmandu University
- While executing query, we also checked the ‘load as new layer’ option and named the layer ‘endpoint’.

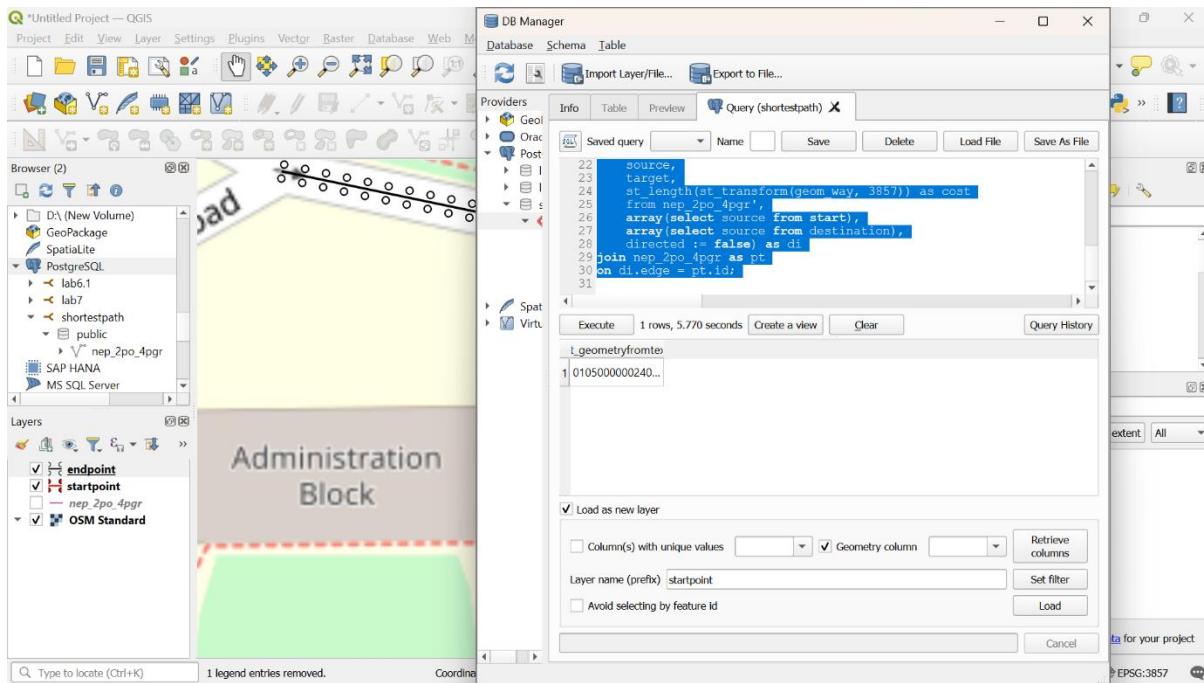




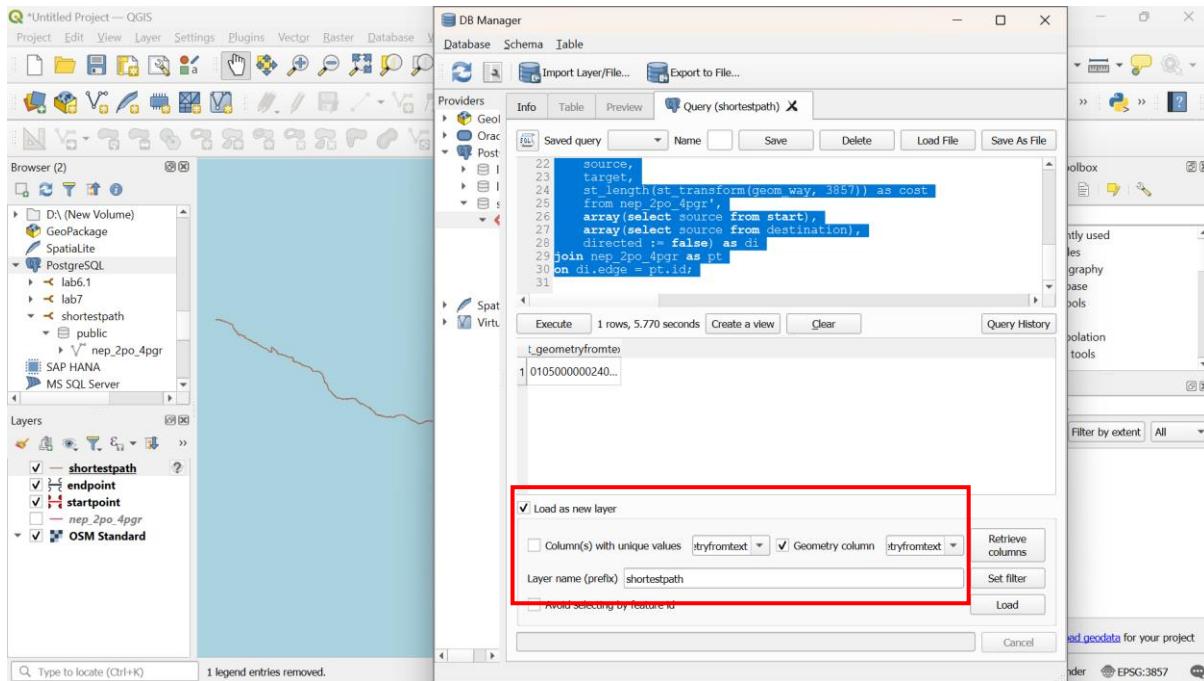
- Changing the symbols to more appropriate ones:



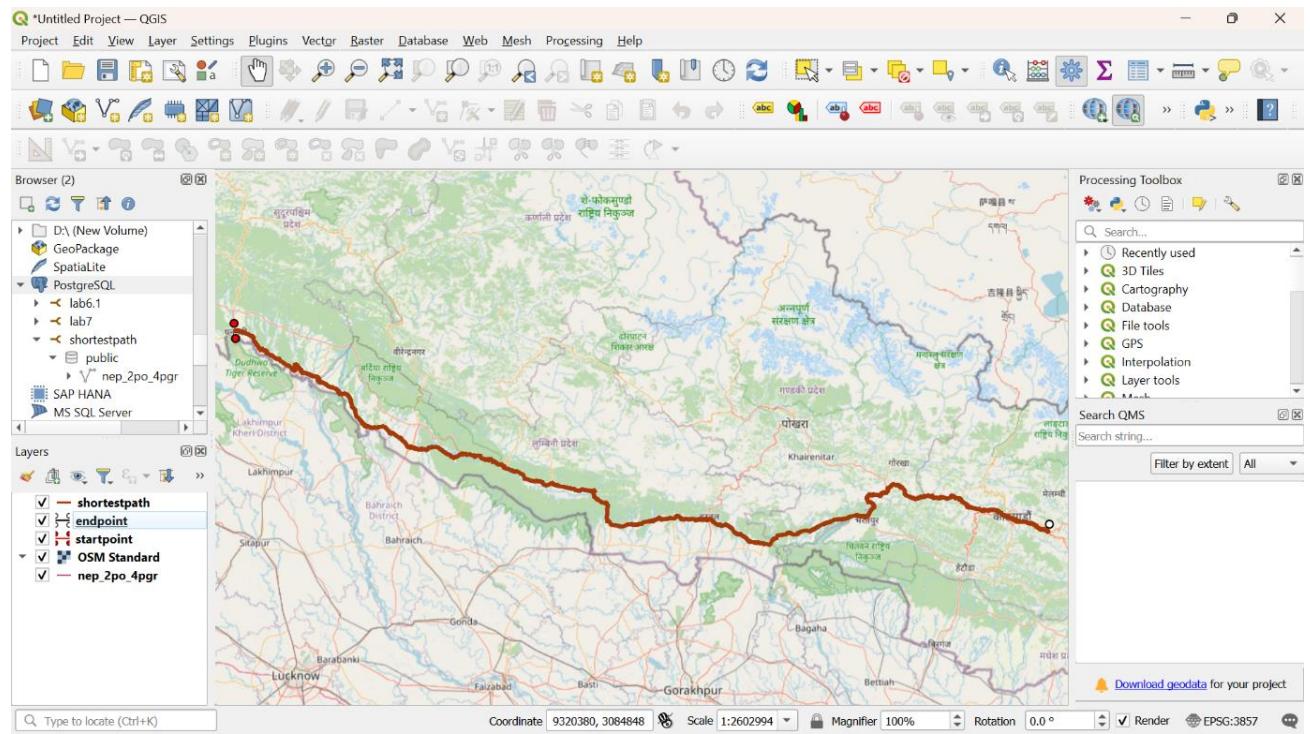
- Applying the Dijkstra's Algorithm to find the shortest path:



- Loading a new layer named as 'shortestpath'.



## Final Result:



## References

- Alam, Md. A., & Faruq, Md. O. (2019). Finding Shortest Path for Road Network Using Dijkstra's Algorithm. *Bangladesh Journal of Multidisciplinary Scientific Research*, 1(2).  
<https://doi.org/10.46281/bjmsr.v1i2.366>
- CHOOSUMRONG, S., RAGHAVAN, V., & BOZON, N. (2012). Multi-Criteria Emergency Route Planning Based on Analytical Hierarchy Process and pgRouting. *GEOINFORMATICS*, 23(4).  
<https://doi.org/10.6010/geoinformatics.23.159>
- Nurhasanah, F. Y., Gata, W., Riana, D., Jamil, Muh., & Saputra, S. F. (2021). Shortest Path Finding Using Dijkstra's Algorithm. *PIKSEL : Penelitian Ilmu Komputer Sistem Embedded and Logic*, 9(1). <https://doi.org/10.33558/piksel.v9i1.2365>
- pgRouting Community. (2016). pgRouting Project — Open Source Routing Library. In *pgRouting*.
- Pritee, K., Garg, R. D., & Ohri, A. (2019). Windows Implementation of PgRouting to find Shortest Path using Dijkstra's Algorithm for Varanasi Road Network. *Journal of Basic and Applied Engineering Research*, 2(2).
- Seeta, V. (2021). Finding the Shortest Path using Dijkstra's Algorithm. *International Journal of Innovative Research in Engineering & Multidisciplinary Physical Sciences*, 9(5).  
<https://doi.org/10.37082/ijirmps.2021.v09si05.017>
- Singh, P. S., Lyngdoh, R. B., Chutia, D., Saikhom, V., Kashyap, B., & Sudhakar, S. (2015). Dynamic shortest route finder using pgRouting for emergency management. *Applied Geomatics*, 7(4).  
<https://doi.org/10.1007/s12518-015-0161-4>