**Designing University of Texas at Arlington's map on a GIS and Spatial Database Systems**

**REPORT BY:**

**GROUP M**

AJINKYA GHADGE

AND

SHUBHAM PATEL

**Abstract:** This project aims at modelling The University of Texas at Arlington campus map, the entire design and modelling has been done by using keyhole markup language files (KML) and converted into shape files and is studied in detail. The campus map covers entire UTA campus including structures like academic buildings, playgrounds, parking lots, administrative buildings, residence halls, on-campus/off-campus apartments. The entire system is built on top of QGIS interface, Google Earth, and Python programming language. The map is represented using three geometric representations in QGIS points, lines and polygons. Alongside map visualization, features like shortest walking path (between two points) calculation using Dijkstra's algorithm, nearest neighbour network analysis has been implemented using Python. Table of Contents

[Abstract](#_3znysh7) 2

[List of Illustrations](#_1fob9te) 4

[Introduction](#_2et92p0) 5

Overview and FlowChart5

Systems and Tools 8

Implementation and Description9

Results 13

Comparison and Analysis17

[Conclusion 1](#_3rdcrjn)7

[References 1](#_3rdcrjn)8

# List of Illustrations

[Figure 1](#_26in1rg) 10

[Figure 2](#_lnxbz9) 11

[Figure 3](#_35nkun2) 12

Figure 4 13

Figure 5 14

Figure 6 15

Figure 7 16

Figure 8 17

**INTRODUCTION**

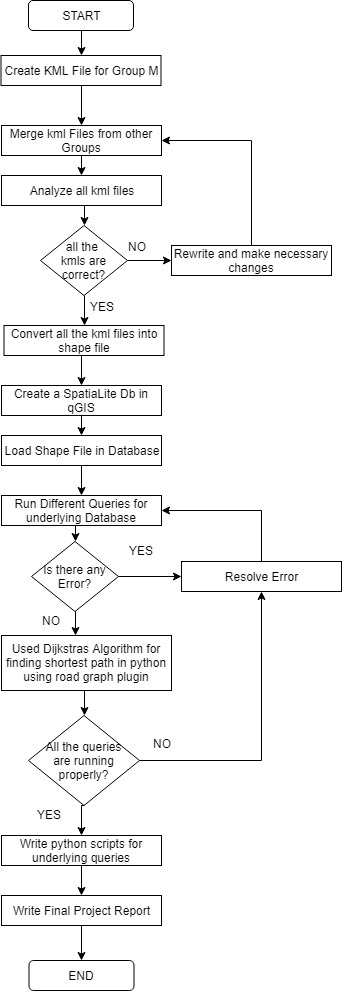
Modelling real world maps and performing geographic study for a specific problem has been one of the most prevailing tasks for any admirer of Geographic information system. The developed system tries to solve one such problem of modelling the University of Texas at Arlington campus map and perform computations like shortest walking path calculations between two points, nearest neighbour analysis etc. The shortest path is computed across the walking paths of the campus by the walking speed of 3 miles/hour. The nearest neighbour analysis is implemented in a way that allows to list any specified structure/structures from any point on the map.

**OVERVIEW AND FLOWCHART**

The high-level description of the project will be as follows.

1. Classify the University of Texas at Arlington (UTA) campus map based on the structures like academic buildings, administrative buildings, on-campus/off-campus apartments, playgrounds, parking lots and residence halls.
2. Build Keyhole Markup Language (KML) files for the entire map using Google Earth and convert into Esri format shape files which can be integrated into any GIS system.
3. Integrate the shape files together, the map is created using Google Earth represented as points, polygons and lines.
4. Once the road network is constructed, the shortest path can be calculated using the road graph plugin of QGIS.
5. Nearest neighbours are constructed using the Python script which displays the results on the screen by highlighting the corresponding points, lines and polygons.
6. The Python script is written in such a way that it can give the results which can display the lines (roads), points(entrances) and polygons based on any input query.

**FlowChart:**

****

**SYSTEMS AND TOOLS**

The tools used for the system are:

1. Google Earth [1]
2. QGIS [2]
3. Python programming Language
4. SpatiaLite database [3]

**Google Earth [1]**

Google earth uses satellite images, GIS data, and aerial photos to model a 3D representation of the earth's surface. The user interface allows using a keyboard/mouse to navigate along with latitude and longitude coordinates. It is also crowd sourced, meaning it allows users to upload their KML files through differents sources like blogs and forums.

**QGIS [2]**

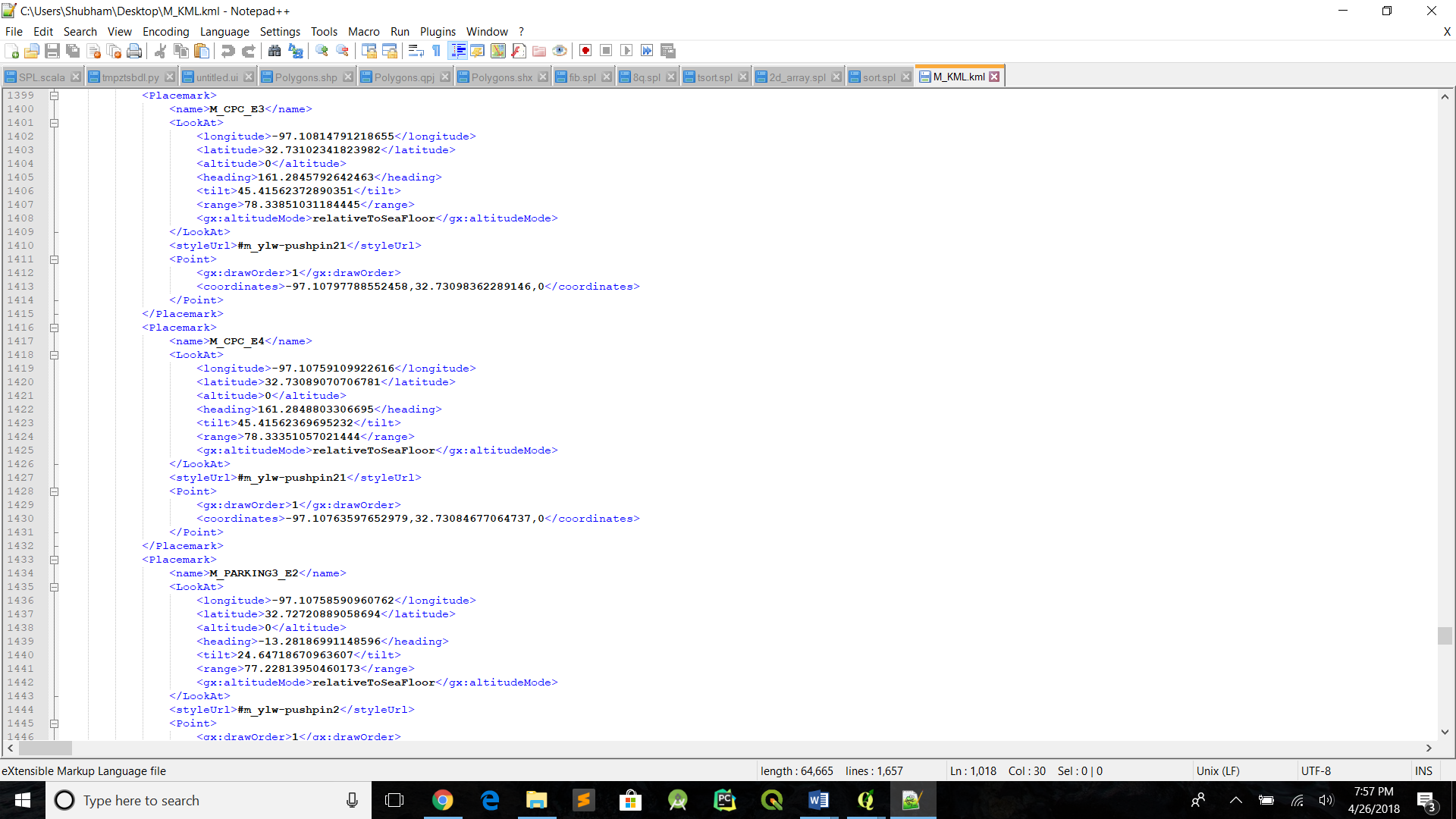
QGIS is an open source GIS application that allows users to view, update and analyze, and perform computations on geospatial data. QGIS can support different formats like shapefiles, PostGIS, coverages etc. Different packages can be integrated with other applications and plugins programmed in Python and C++.

Plugins can be geocoded using Google geocoding API, perform geoprocessing operations and interface with MySQL, PostgreSQL and SQLite databases.

**SpatiaLite Database [3]**

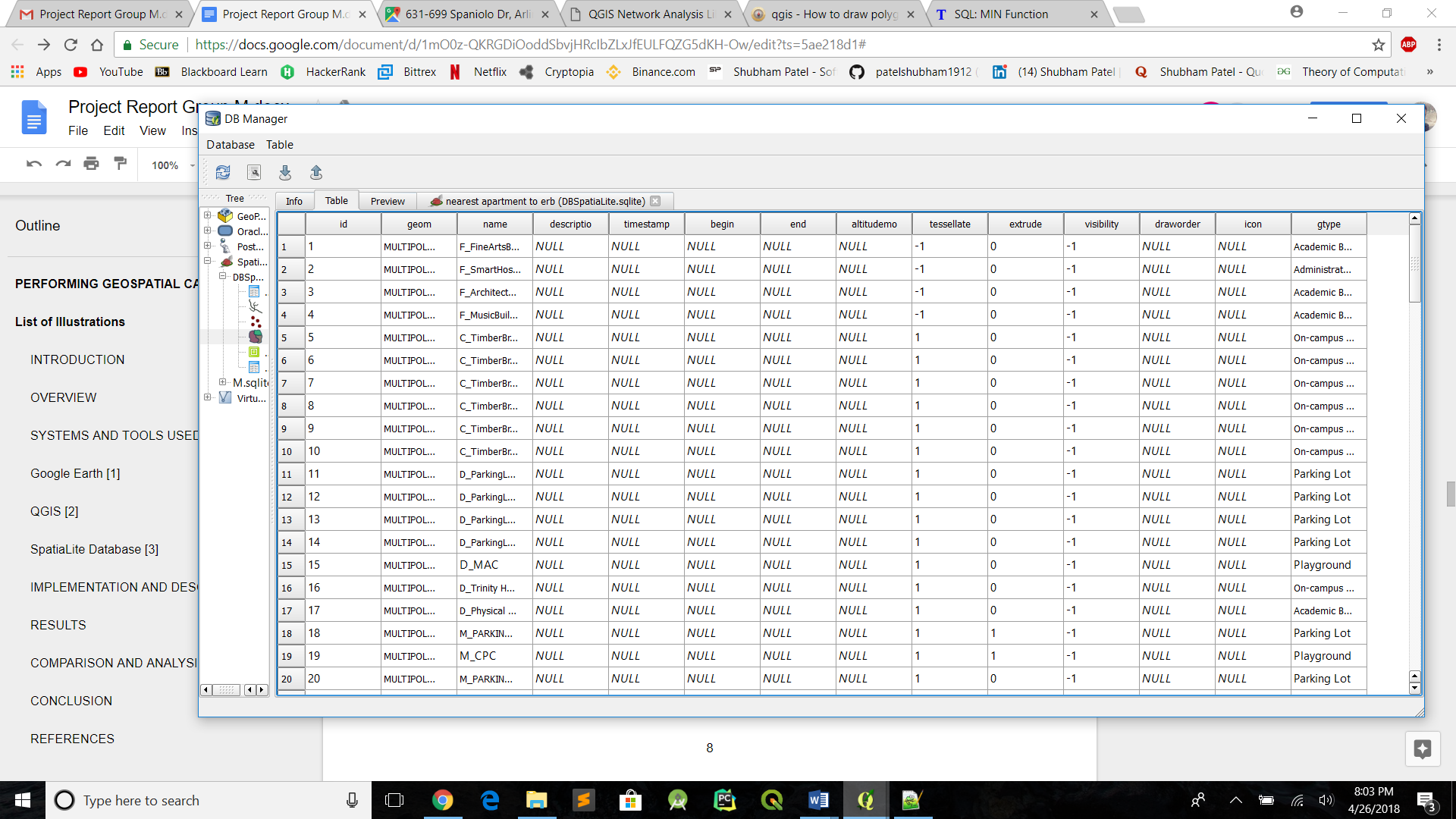
SpatiaLite database is an extension to the SQLite database and provides geodatabase functionality. In SpatiaLite database the whole sql engine is embedded within the application which means that the database file can be transferred freely and would still run on any computer or operating system. It is similar to Oracle Spatial and PostGIS databases.

**IMPLEMENTATION AND DESCRIPTION**

1. *Constructing KML file of UTA Campus*: Using Google Earth, KML file of different sections of the campus were generated individually specifying different structures as points, lines and polygons and merged together into a 3 layered ESRI format shape files. Following is an example of kml code snippet: 

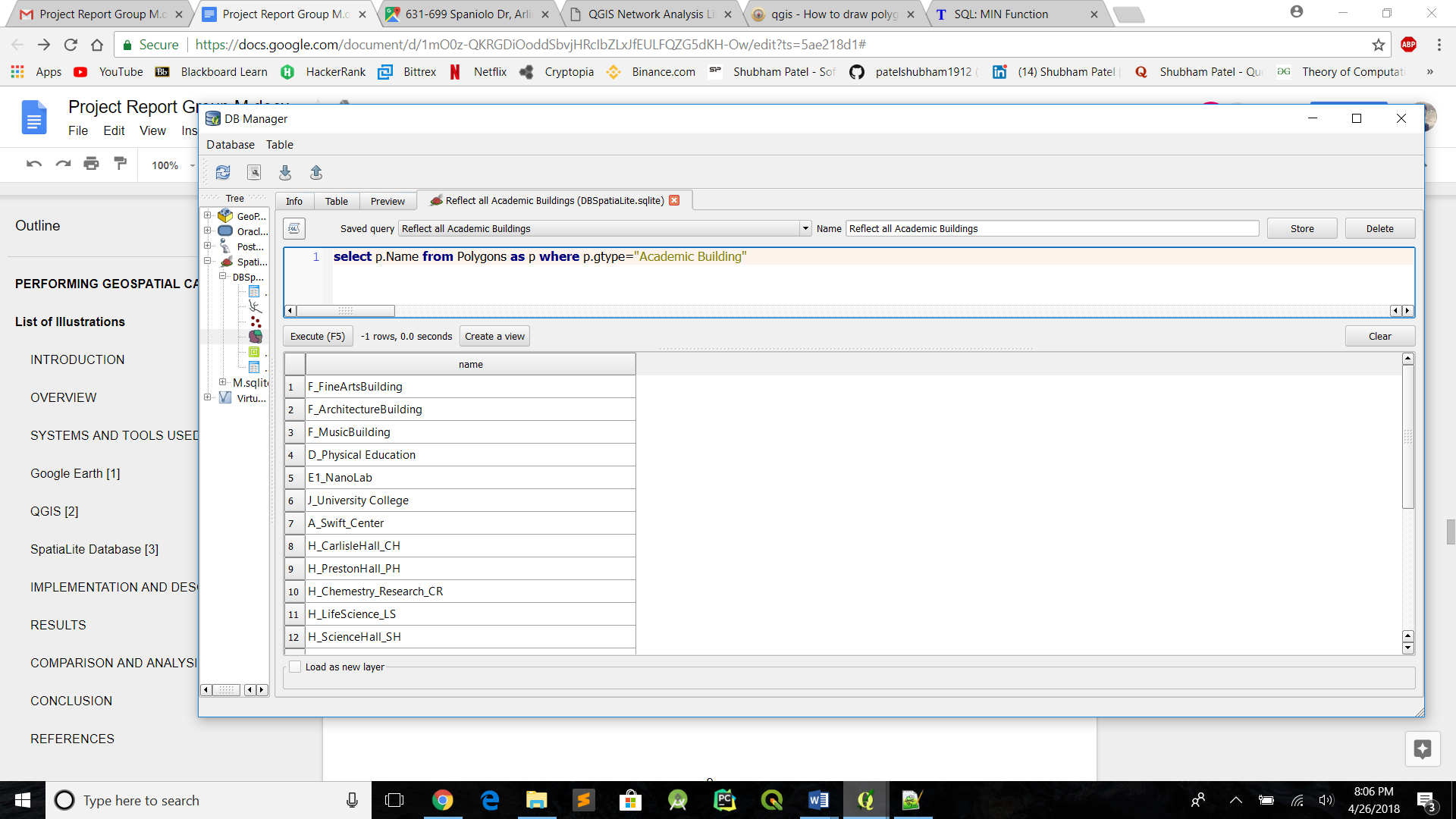
*Figure 1*

1. *Loading the shape files into SpatiaLite database:* Create a SpatiaLite database with the help of a plugin in QGIS and all three shape files were imported as 3 tables in the database. Following is the snippet of spatialite polygon table;



*Figure 2*

1. *Identifying the points of interest:* In the polygon table we add another column named “GTYPE” which represents the category in which the polygons falls into example: academic buildings, administrative buildings, on/off campus apartments, residence halls, parking lots etc.
2. *Running queries on SpatiaLite:* Test the SpatiaLite database to check if all the queries run smoothly. Following is an example of query that we ran to check if data is retrieved accurately.

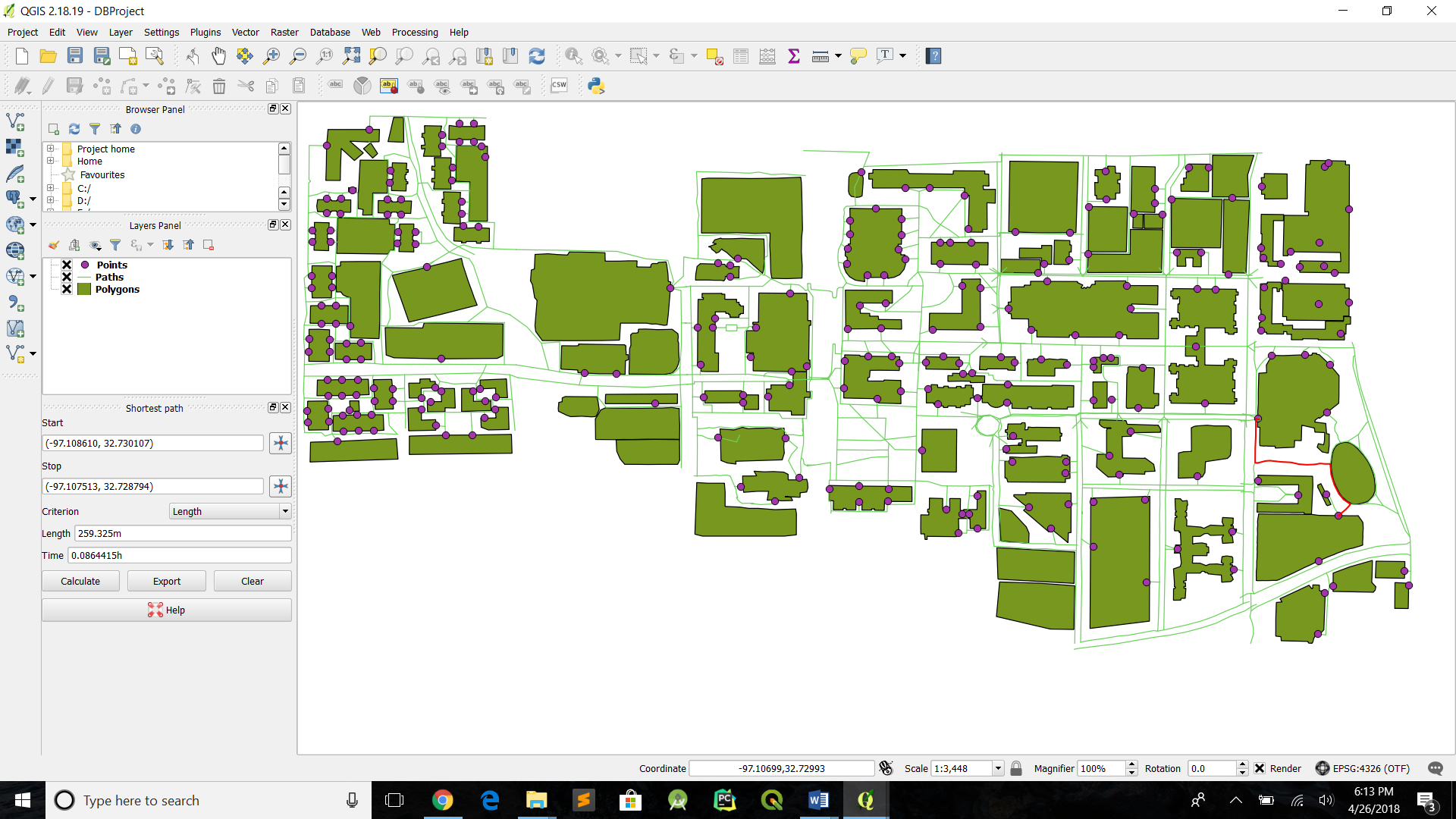


*Figure 3*

1. *Computing shortest path:* QGIS inbuilt plugin named 'Road graph plugin' was used to calculate the shortest path between any two entrances of any two polygons was used to show the shortest walking route between them in metres.
2. *Displaying the output of queries:* Python script [4,5] was written to display the output of each spatiaLite query in different layers in QGIS.

**RESULTS**

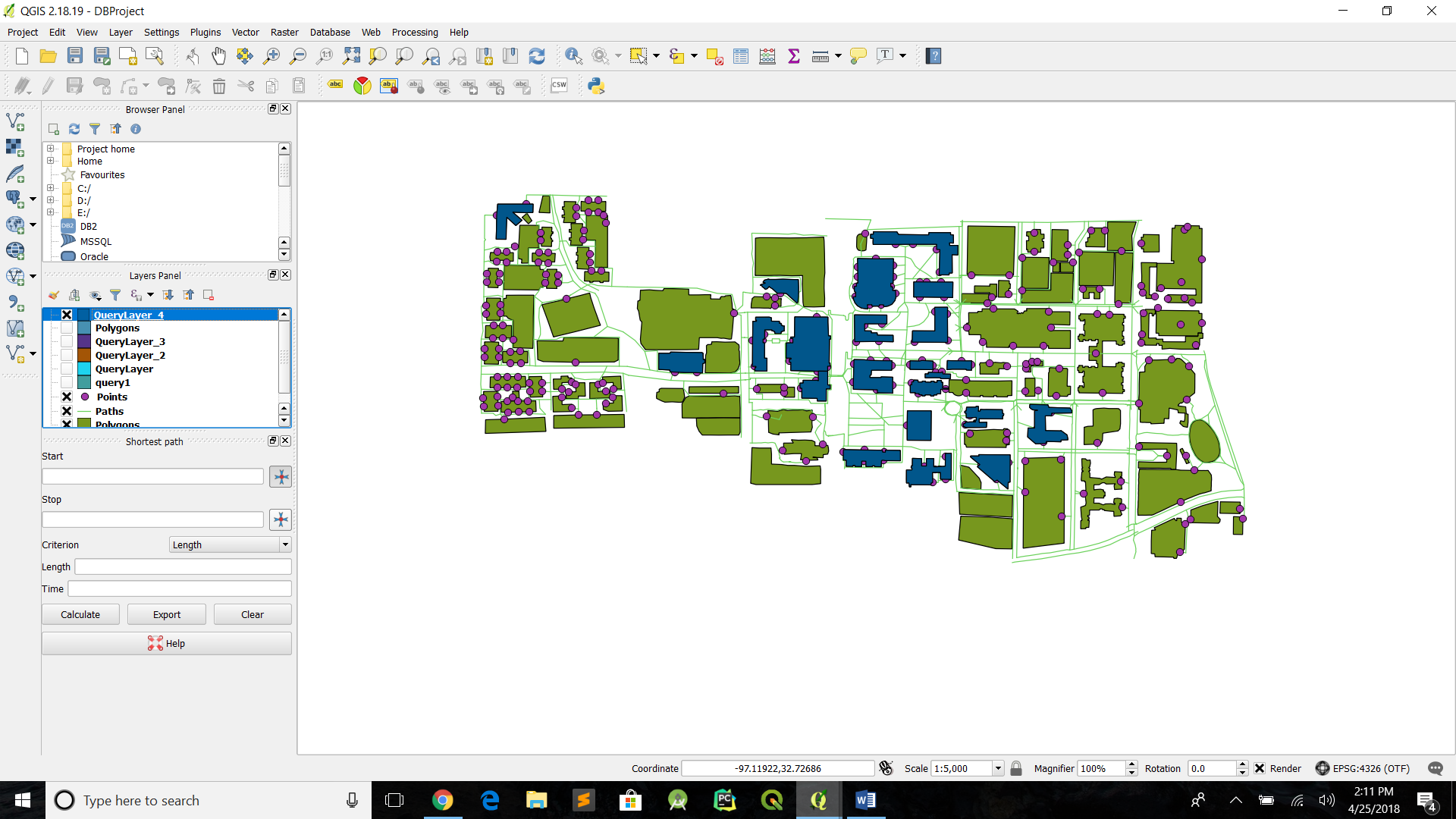
1. **Computing shortest walking path between two entrances of any two polygon**

****

*Figure 4*

As shown in figure 1, we have computed shortest path between CPC Entrance 1 and M\_PARKING Entrance 1, which is evaluated as 259.32 m. Road Graph plugin of QGIS has been used to display the same. On the other hand we have also written a python script named “shortestpaths.py” which works on the functionality of the same plugin to calculate the shortest path between two coordinates.

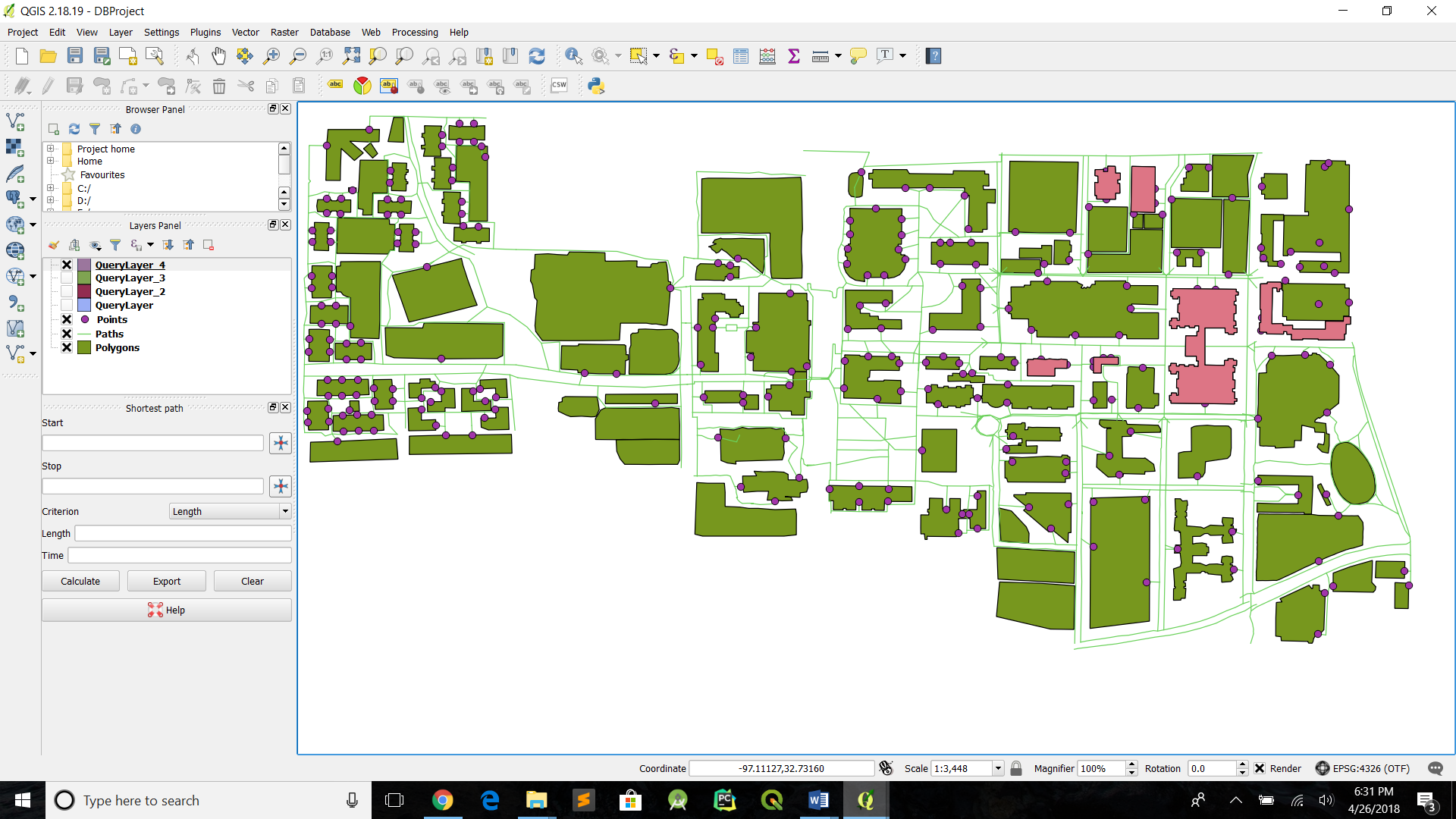
2. **Query to display all the academic buildings within our map**



*Figure 5*

The following query has been used to retrieve the polygons whose GTYPE in the database is “Academic Buildings” : “*select p.geom from Polygons as p where p.gtype=’Academic Building’*”. This has been executed by writing a python script named “AcademicBuildings.py” to load the data of query result into different qgis layer for display (as shown in figure 2, blue color region).

3. **Query to select on campus apartments within 500 m of ERB**



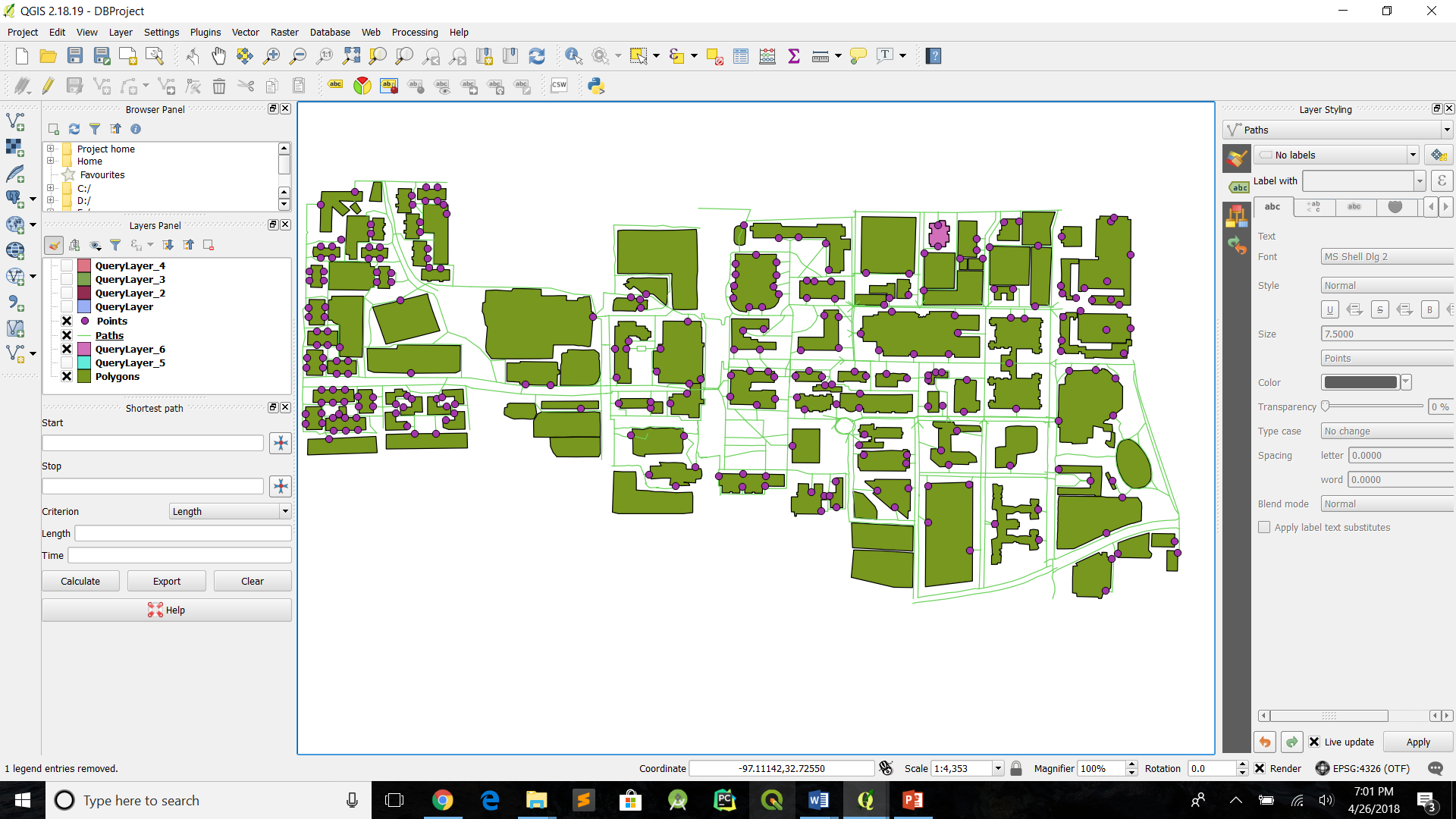
*Figure 6*

In figure 3, All the on campus apartments are displayed in pink color. Following query is used for the same:

“*select e.geom from Polygons as p, Polygons as e where st\_distance (st\_transform(st\_centroid(p.geom),2163), st\_transform( st\_centroid (e.geom) ,2163)) <500 and p.Name='G\_ERB' and e.gtype='On-campus Housing'* “

To calculate the distance between two polygons, distance between centroid of each polygon is considered. This has been implemented by writing a python script named “500mERB.py” .

4. **Which is the nearest on-campus apartment from ERB**



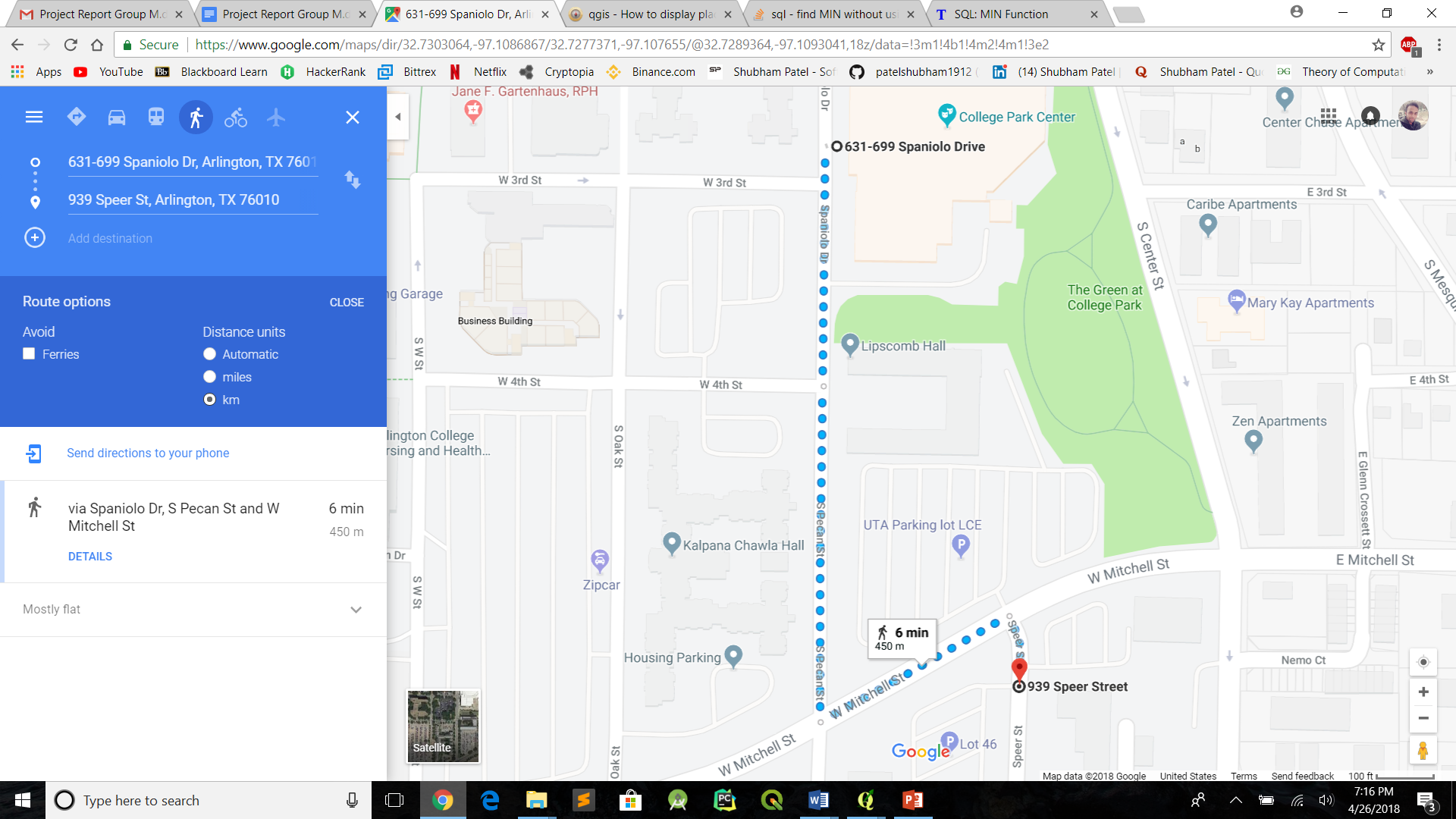
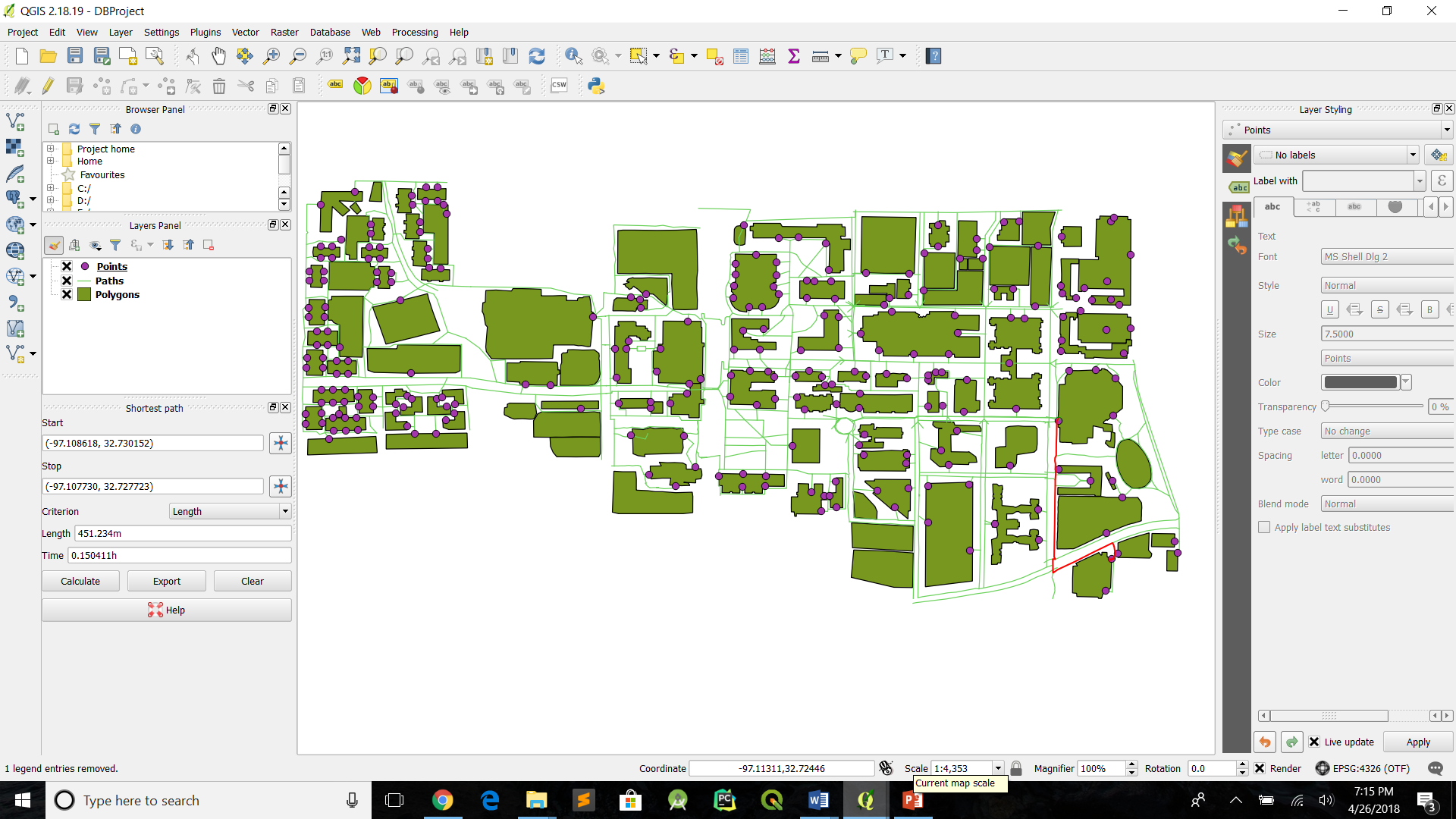
*Figure 7*

Figure 4 shows that "Garden Club" apartment displayed in pink color is the nearest on campus apartment to “ERB”. Following query is used to display the same.

“*select e.geom from Polygons as p, Polygons as e where p.Name='G\_ERB' and e.gtype='On-campus Housing' and st\_distance (st\_transform (st\_centroid (p.geom),2163),st\_transform(st\_centroid(e.geom),2163)) = (select min( st\_distance( st\_transform (st\_centroid (p.geom), 2163), st\_transform (st\_centroid (e.geom),2163))) from Polygons as p, polygons e where p.Name='G\_ERB' and e.gtype='On-campus Housing')*”. This has been implemented by writing a python script named “nearestERB.py”.

Above all queries are modified in python script so that we can retrieve the coordinates of boundaries of polygon instead of polygon itself. The same coordinates are then redrawn in different layer when displayed in graphical user interface.

**COMPARISON AND ANALYSIS**

**Comparison of Road Graph shortest walking path with respect to google maps:**

*Figure 8 (Road Graph walking Path - 451.23m , Google Maps - 450 m)*

**CONCLUSION**

From the results we can see that the nearest neighbour queries run smoothly displaying the correct polygons and the accuracy of shortest path computation is equivalent with Google Maps.

**REFERENCES**

1. Google Earth, Wikipedia, "<https://en.wikipedia.org/wiki/Google_Earth>"
2. QGIS, Wikipedia, "<https://en.wikipedia.org/wiki/QGIS>"
3. SpatiaLite database, Wikipedia, "<https://en.wikipedia.org/wiki/SpatiaLite>"
4. QGIS Package reference, “<https://gis.stackexchange.com/questions/86812/how-to-draw-polygons-from-the-python-console>”
5. QGIS Network Analysis, “<http://gis-lab.info/qa/qgis-network-analysis-lib.html>”