**DESIGN UNIVERSITY OF TEXAS AT ARLINGTON MAP ON GIS & SPATIAL DATABASE SYSTEMS**

GROUP F – Pramodh Kampalli (1001373669) and Venkatesh Suvarna(1001390986)

**High Level Project Description:**

We were given the responsibility to design the University of Texas at Arlington map by using any Geographic Information Systems(GIS) and any relevant Spatial Database and to perform certain operations and display the information on the GIS software and extract meaningful information.

Talking about the raw data, the entire map was divided into 13 sections – and each team had to develop the respective sections of the map. The resultant file format is called KML(Keyhole Markup Language) – which is very similar to XML(extensible Markup Language) except for the fact that KML is used for spatial data storage. In KML – there are three types of spatial data – polygons, lines, and points. For representing sections of the map – the main buildings were represented using polygons, the entrances of the building being represented as points, and the walking paths around the campus being represented as linestrings. We constructed our KML file using Google MyMaps – and made manual adjustments – copy pasting the coordinates by opening the KML file in any text editor to ensure that all the entrances are exactly on the polygon and the walking paths start exactly on the polygon. We then went on to merge all the 13 KML files into a single KML file for convenience. We used KML Merge v1.0 and Google Earth Pro to divide the merged KML having 3 folders – Polygons, Paths and Entrances – where all the polygons, lines and points were stored.

For the GIS software – we used QGIS which is the short form of Quantum GIS. We used that to create a .sqlite file and added the KML as a vector layer and then adding the layers to the .sqlite file. We used Spatialite – for constructing and storing the spatial data. As far as the plugin is considered – we are using Python scripts to access the database and perform queries through a user-friendly interface.

**Systems and Software used:**

Building KML file – Google MyMaps and Google Earth Pro

GIS – Quantum GIS 3.0

Spatial Database – Spatialite

Scripts – Python with mod-spatialite library

User Interface – TKInter Library for Python

**Description of the Algorithms Used:**

We have quite a few algorithms – the first algorithm is to find all the building near a particular building. This makes use of the Polygon table of the database that we have created. Example of this query could be – “find all the buildings within 500 meters of ERB”. This makes use of spatial query to create a buffer around the query building and then we check for all the buildings whether they intersect with the buffer – all those buildings are added into a new temporary layer and displayed in QGIS. All of this is done with the help of Python + TKinter scripts.

The second algorithm is basically a slightly more advanced version of the first algorithm. That is the second algorithm is to find all types of buildings within a range of the query building. Example of this query is – “Find all parking lots within 250 meters of University Center” or “Find all parks within 700 meters of Davis Hall”. For that we have made changes to the spatial database file – where we have divided all the building into 12 categories:

1. On Campus Apartments
2. Green Spaces – Gardens
3. Parking
4. Academic Building
5. Recreational Building
6. Food Place
7. Medical Care Building
8. Off Campus Residences
9. Off Campus Buildings
10. Bookstore
11. Library
12. Administrative Buildings

Once we have categorized the building, we are using a similar buffer query – and add the buildings that intersect with the buffer are added to a new layer which is displayed using QGIS. For this query we have used the first algorithm and user interface as the basis.

The third algorithm is about the shortest path algorithm – we have used the shortest path plugin(point to point) in QGIS 3.0 – to find the shortest path in a new temporary layer that is being displayed in QGIS. Our Python script then determines the length of the line and converts it into meters – and then we divide the distance by 3 mph(the average walking speed) to find the time to cover the distance and we display the time in minutes.

The fourth algorithm is straight forward where we display all the buildings of a particular type – the 13 types listed above. So using this algorithm we can find the answer to – “Find all the academic buildings in UTA” or “Find all the off campus apartments in UTA” – it will create a new temporary layer in QGIS and will be highlighted in the map accordingly.

**Project Execution Steps:**

1. Shortest path:
   1. Open QGIS 3.0 and load FullMap.sqlite DB into QGIS
   2. Import the layers from the Browser -> Spatalite -> FullMap.sqlite to the Layer column
   3. Open the shortest path (point to point) plugin – from Processing -> Toolbox
   4. Select the input vector layer as the Paths Layer
   5. Select the Start Point and End Point from the map
   6. Click on ‘Run in Background’
   7. You will see the shortest path line appended as a new layer in QGIS
   8. Run shortestpath.py in QGIS from Plugins -> Python Console
   9. The length of the line in meters and the expected time to cover that distance (in minutes) will be displayed in the console
2. Display all the buildings within a range of an input building
   1. Run neighbors.py in another Python IDE or from your current python command line
   2. Select your source building and enter the distance in meters in the input box
   3. Click on ‘Process’ button
   4. Go to QGIS python console and run the withinmiles-new.py file
   5. The resulting buildings will be added as a new result layer in QGIS
   6. Optionally – you can click on ‘Display Results’ in GUI to view names of all the result buildings
3. Display all the buildings of a category within a range of an input building
   1. Run neighbors.py in another Python IDE or from your current python command line
   2. Select your source building and enter the distance in meters in the input box
   3. Select the category of building that you want to display
   4. Click on ‘Process’ button
   5. Go to QGIS python console and run the withinmiles-new.py file
   6. The resulting buildings will be added as a new result layer in QGIS
   7. Optionally – you can click on ‘Display Results’ in GUI to view names of all the result buildings
4. Display all the buildings of a category on the map
   1. Run neighbors.py in another Python IDE or from your current python command line
   2. Select the category of building that you want to display
   3. Click on ‘Process’ button
   4. Go to QGIS python console and run the withinmiles-new.py file
   5. The resulting buildings will be added as a new result layer in QGIS
   6. Optionally – you can click on ‘Display Results’ in GUI to view names of all the result buildings

**Source Code: The Zip file name is F\_Project.zip and it contains the following files**

1. neigbours.py - The python GUI which provides a user-friendly interface for the 3 algorithms (except the shortest path)
2. shortestpath.py – This file takes information from the result layer through point to point shortest path plugin in Qgis 3.0 and displays the length of the shortest path in meters and the time(in minutes) considering a walking speed of 3 miles per hour
3. withinmiles-new.py – This file reads the input from the GUI and based on the input given, it computes a query and executes that query to show it as a result layer in QGIS