ESCORT

TRAJECTORY ANALYSIS FOR TRAFFIC PATTERNS

By: Ahmed Wael Samy
Amr Ali Farrag
Loai Hamdy Amin
Mina Kamal youssef

Supervised by: Eng. Youssef Mubarak, Dr.Mustafa Abdo and Dr. Mai El-Shehaly

WE AIM TO ACHIEVE SOLUTIONS FOR PLACING BILLBOARD ADVERTISEMENT IN RECOMMENDED LOCATIONS FROM OUR SYSTEM.

DATASET

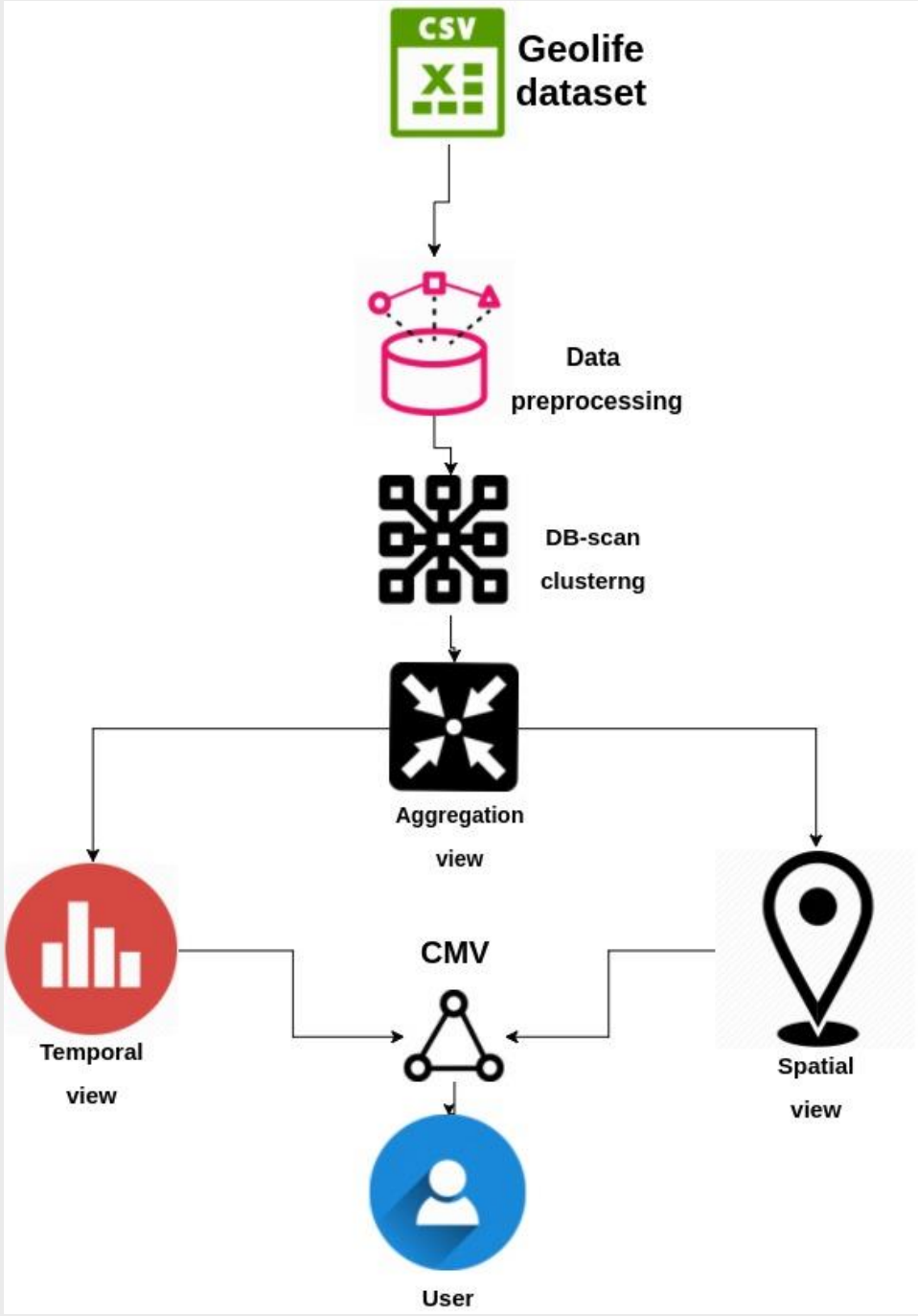
Microsoft

lat	lng	date	time
39.9748	116.313	2007-04-12	14:30:45
39.9747	116.312	2007-04-12	14:31:09

Fig.(1): GPS Finger Print

(1)This GPS trajectory dataset was collected in (Microsoft Research Asia) Geolife by 178 users in a period of over four years (from April 2007 to August 2012)

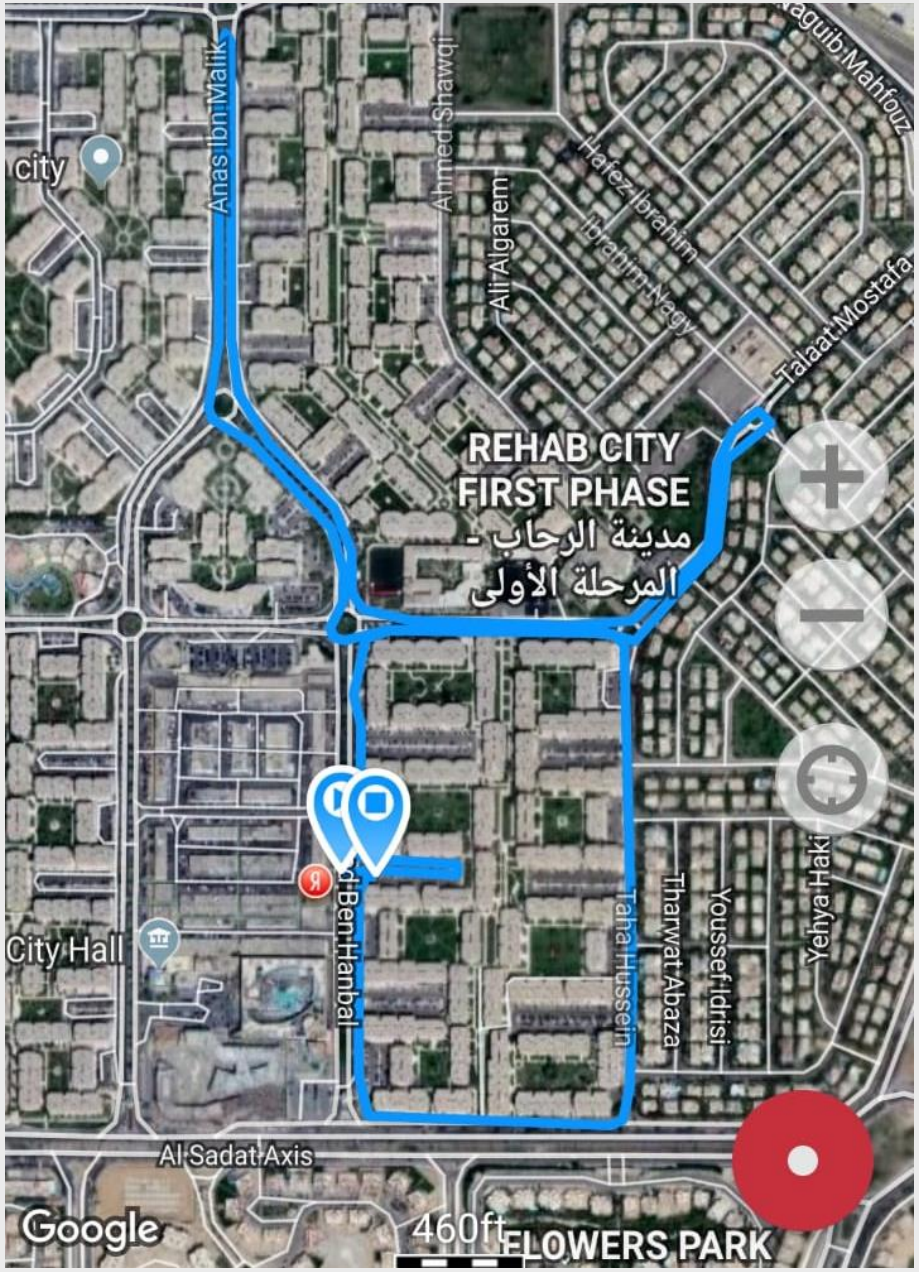
SYSTEM OVERVIEW



TOOLS AND ALGORITHM USED

ADDITIONAL TESTING DATA

- We tried developing GPS trajectories of our own by recording our GPS track while cruising in new Cairo, but it wasn't enough to develop clusters.



EVALUATION: EXPERIMENT 2/2
DOMAIN EXPERTS TRYING THE SYSTEM

Goals

In this experiment we want to know whether the system is suitable for the market, also whether it will be a useful asset for the market and whether it is a user friendly or not, so we picked two domain experts to test the system and give us their feedback.

Domain experts are:

- Mr. Tamer Adel Osman (Advertisement agency CEO)

- Mr. George M. Hanna (Managing Director of Squares Smart Solutions)

COMPETITIONS

Advances in Data Analysis and Classification Journal

Evaluation: Experiment 2/2
Domain experts trying the system

Misr International University
جامعة مصر الدولية

Result

- Develop Mobile Application.
- Don’t forget Government Restriction.

SYSTEM PRICE

IBM **Cloud**

CloudDigitalSales@us.ibm.com
Toll Free: +1 (866) 403-7638
Local: +1 (214) 442-0603
Fax: + 1(214) 442-0601

Estimate generated on IBM Cloud Calculator tool:
http://ibm.biz/pricing-calculator

Estimate Summary

Description	Quantity	Price (USD)
Virtual Server	1	$380.25
	Estimated Total:	$380.25

DEMO

ANY QUESTIONS ??

APPENDIX

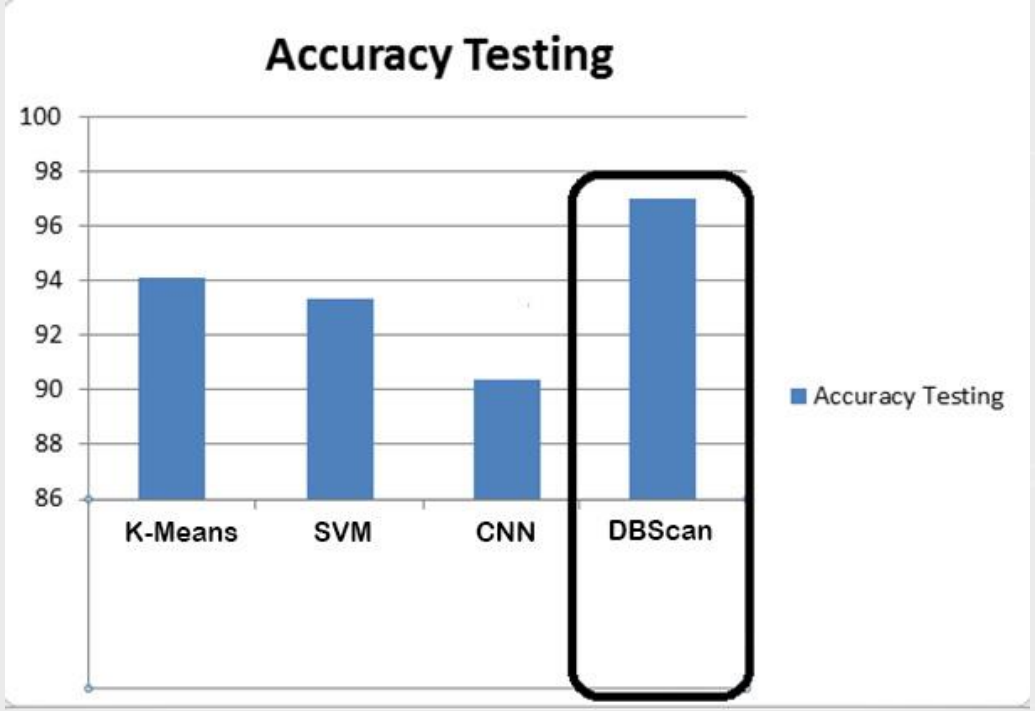
EVALUATION: EXPERIMENT 1/3

- Setup
  - The dataset used in this experiment is an open source data taken from Microsoft Asia recording GPS data in a time span (2007-2011) from 128 users, dataset contains 17,621 trajectories with a total distance of 1,251,654 kilometers and a total duration of 48,203 hours.
- We used four main algorithms in this experiment, all in machine learning
  - DB-scan.
  - KNN.
  - SVM.
  - CNN.

Evaluation: Experiment 1/3

RESULTS:

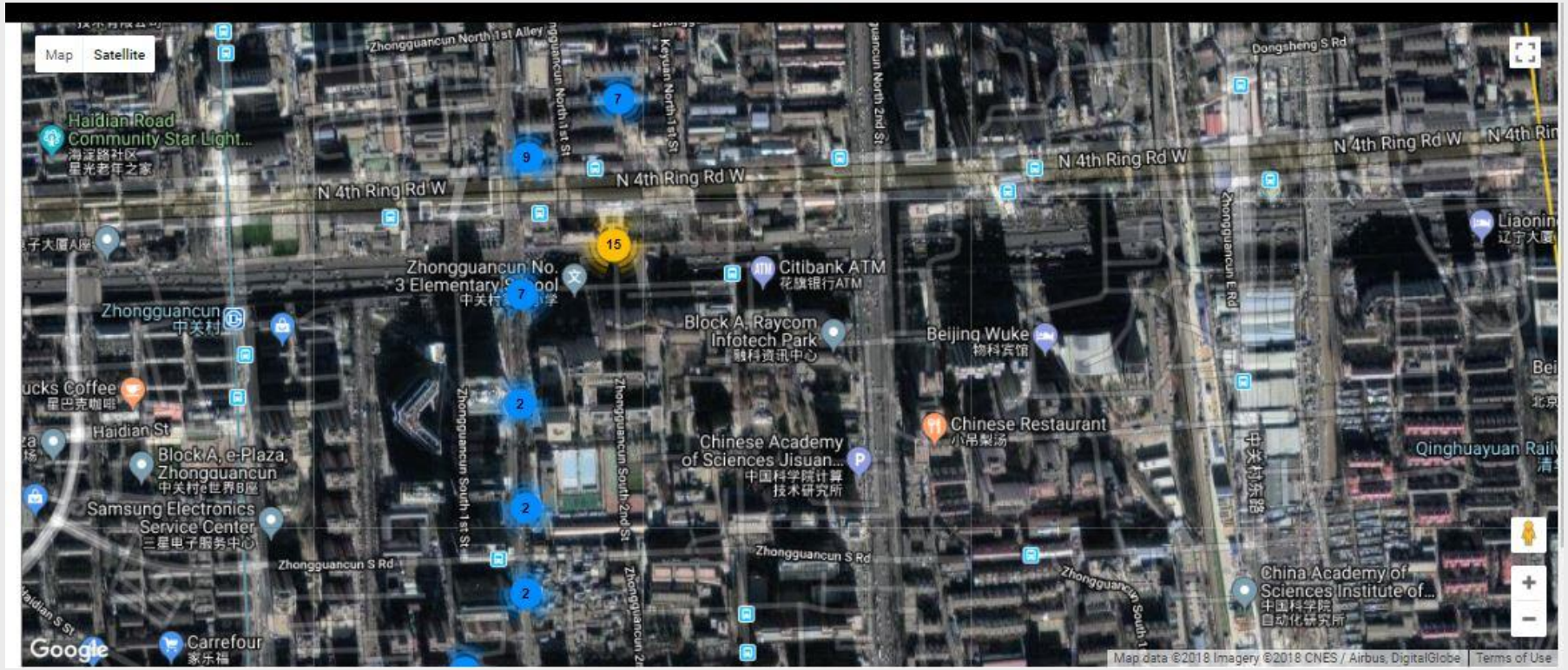
DB-scan algorithm have the best results in case of (Accuracy-Time complexity), because It is a density-based clustering algorithm: it groups together points that are closely packed together (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions (whose nearest neighbors are too far away). so it best suits geographic dataset.



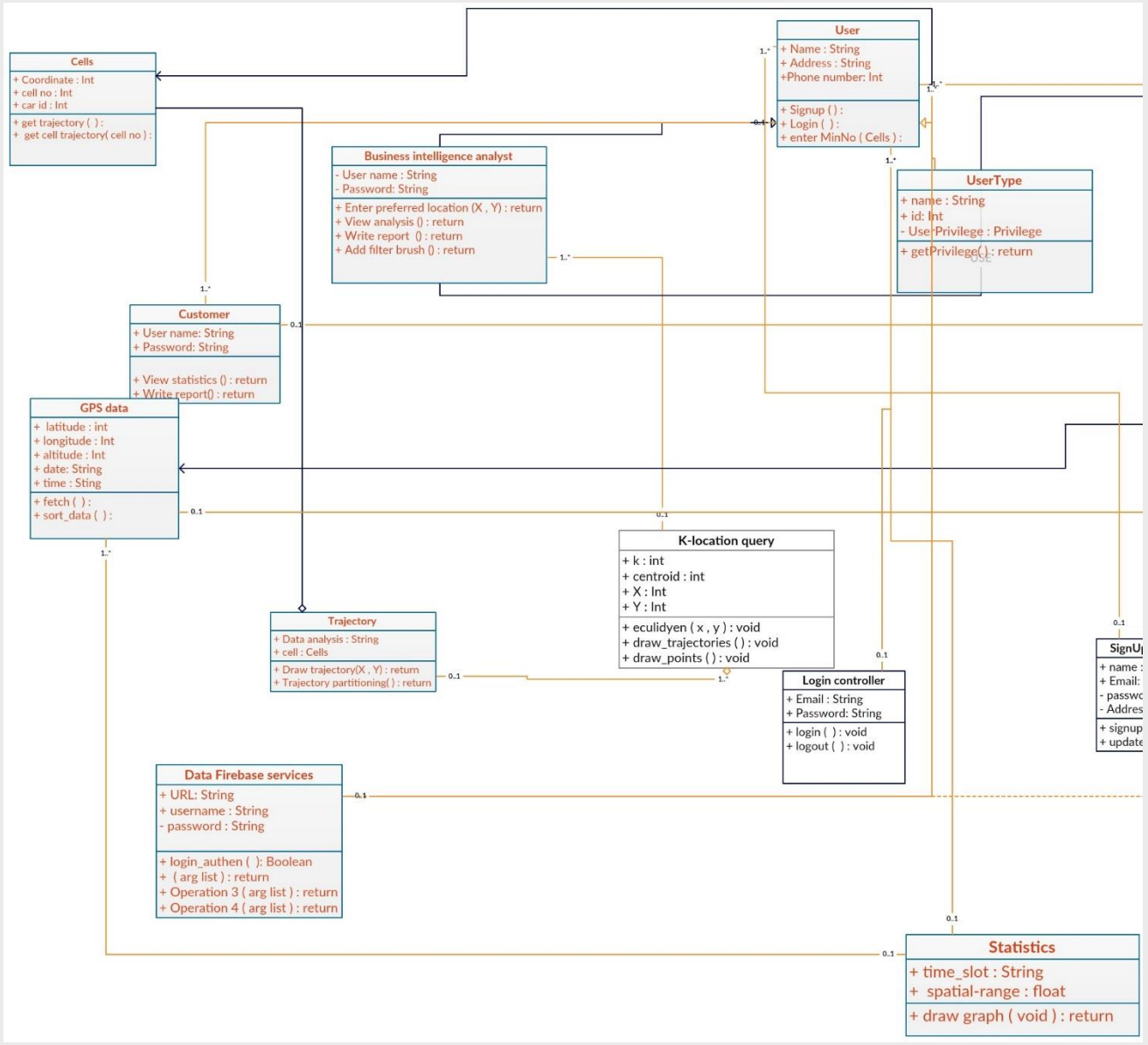
DBSCAN ALGORITHM

```
1:  DBSCAN(D, epsilon, min_points):
2:      C = o
3:      for each unvisited point P in dataset
4:          mark P as visited
5:          sphere_points = regionQuery(P, epsilon)
6:          if sizeof(sphere_points) < min_points
7:              ignore P
8:          else
9:              C = next cluster
10:             expandCluster(P, sphere_points, C
                  , epsilon, min_points)

11: expandCluster(P, sphere_points, C
              , epsilon, min_points):
12:     add P to cluster C
13:     for each point P' in sphere_points
14:         if P' is not visited
15:             mark P' as visited
16:             sphere_points' = regionQuery(P', epsilon)
17:             if sizeof(sphere_points') >= min_points
                    sphere_points = sphere_points joined
                      with sphere_points'
18:             if P' is not yet member of any cluster
19:                 add P' to cluster C

20: regionQuery(P, epsilon):
21:     return all points within the n-dimensional
                sphere centered at P with radius
                epsilon (including P)
```

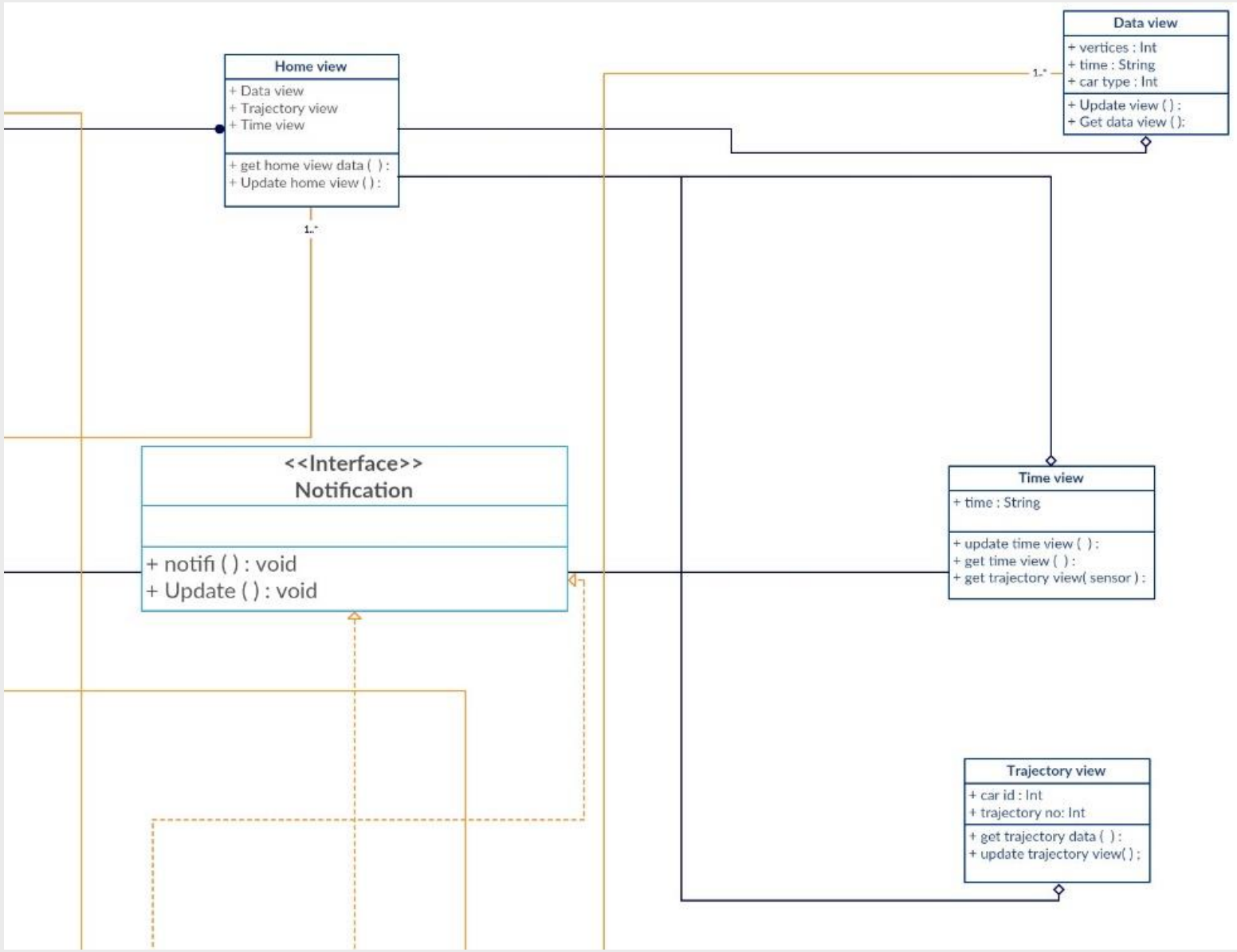
FINAL OUTPUT



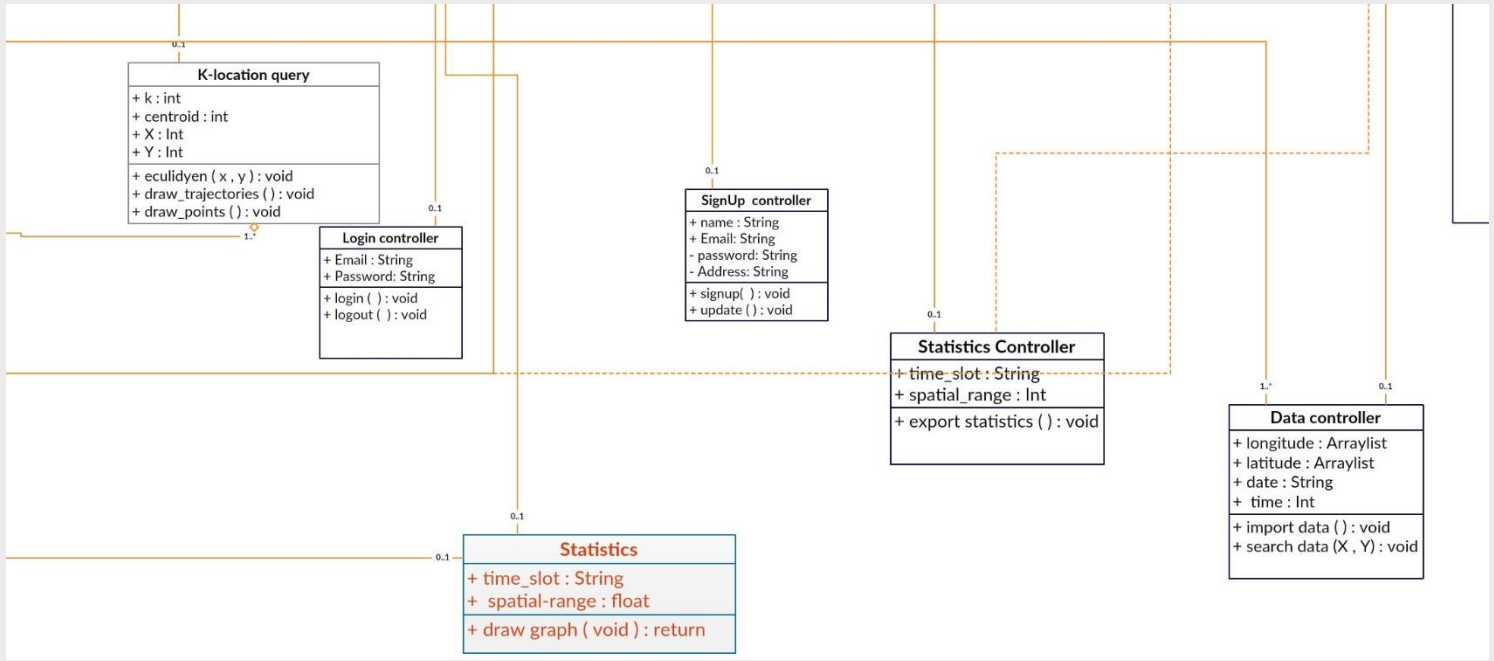
MODEL


Cells
+ Coordinate : Int
+ cell no : Int
+ car id : Int
+ get trajectory ( ) :
+ get cell trajectory(cell no) :
User
+ Name : String
+ Address : String
+Phone number: Int
+ Signup ( ) :
+ Login ( ) :
+ enter MinNo (Cells) :
Business intelligence analyst
- User name : String
- Password: String
+ Enter preferred location (X , Y) : return
+ View analysis () : return
+ Write report () : return
+ Add filter brush () : return
UserType
+ name : String
+ id: Int
- User Privilege : Privilege
+ getPrivilege( ) : return
Customer
+ User name: String
+ Password: String
+ View statistics () : return
+ Write report() : return
GPS data
+ latitude : int
+ longitude : Int
+ altitude : Int
+ date: String
+ time : Sting
+ fetch ( ) :
+ sort_data ( ) :
K-location query
+ k : int
+ centroid : int
+ X : Int
+ Y : Int
+ eculidyen (x , y) : void
+ draw_trajectories ( ) : void
+ draw_points ( ) : void
Trajectory
+ Data analysis : String
+ cell : Cells
+ Draw trajectory(X , Y) : return
+ Trajectory partitioning( ) : return
Login controller
+ Email : String
+ Password: String
+ login ( ) : void
+ logout ( ) : void
Data Firebase services
+ URL: String
+ username : String
- password : String
+ login_authen ( ): Boolean
+ (arg list) : return
+ Operation 3 (arg list) : return
+ Operation 4 (arg list) : return
Statistics
+ time_slot : String
+ spatial-range : float
+ draw graph (void) : return

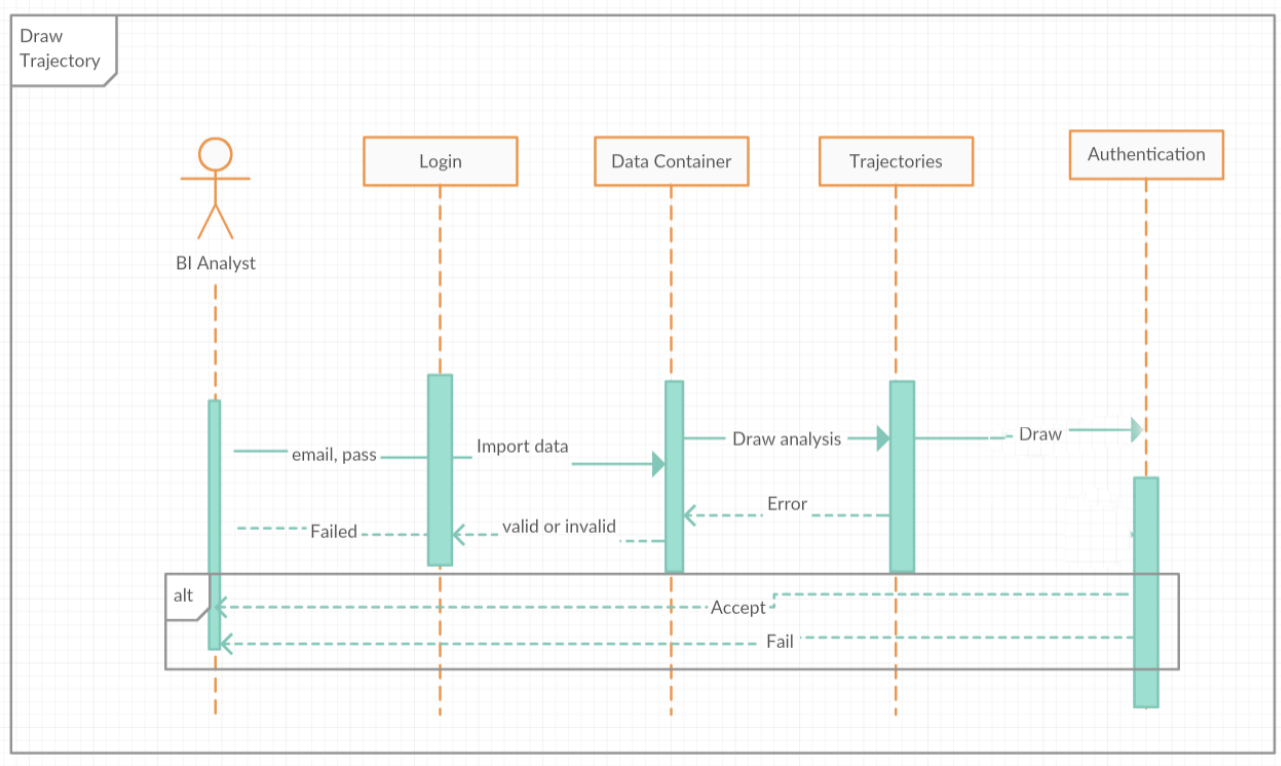
VIEW


Home view
+ Data view
+ Trajectory view
+ Time view
+ get home view data ( ) :
+ Update home view ( ) :
1..*
1..*
Data view
+ vertices : Int
+ time : String
+ car type : Int
+ Update view ( ) :
+ Get data view ( ):
<<Interface>>
Notification
+ notifi ( ) : void
+ Update ( ) : void
Time view
+ time : String
+ update time view ( ) :
+ get time view ( ) :
+ get trajectory view(sensor) :
Trajectory view
+ car id : Int
+ trajectory no: Int
+ get trajectory data ( ) :
+ update trajectory view( ) ;

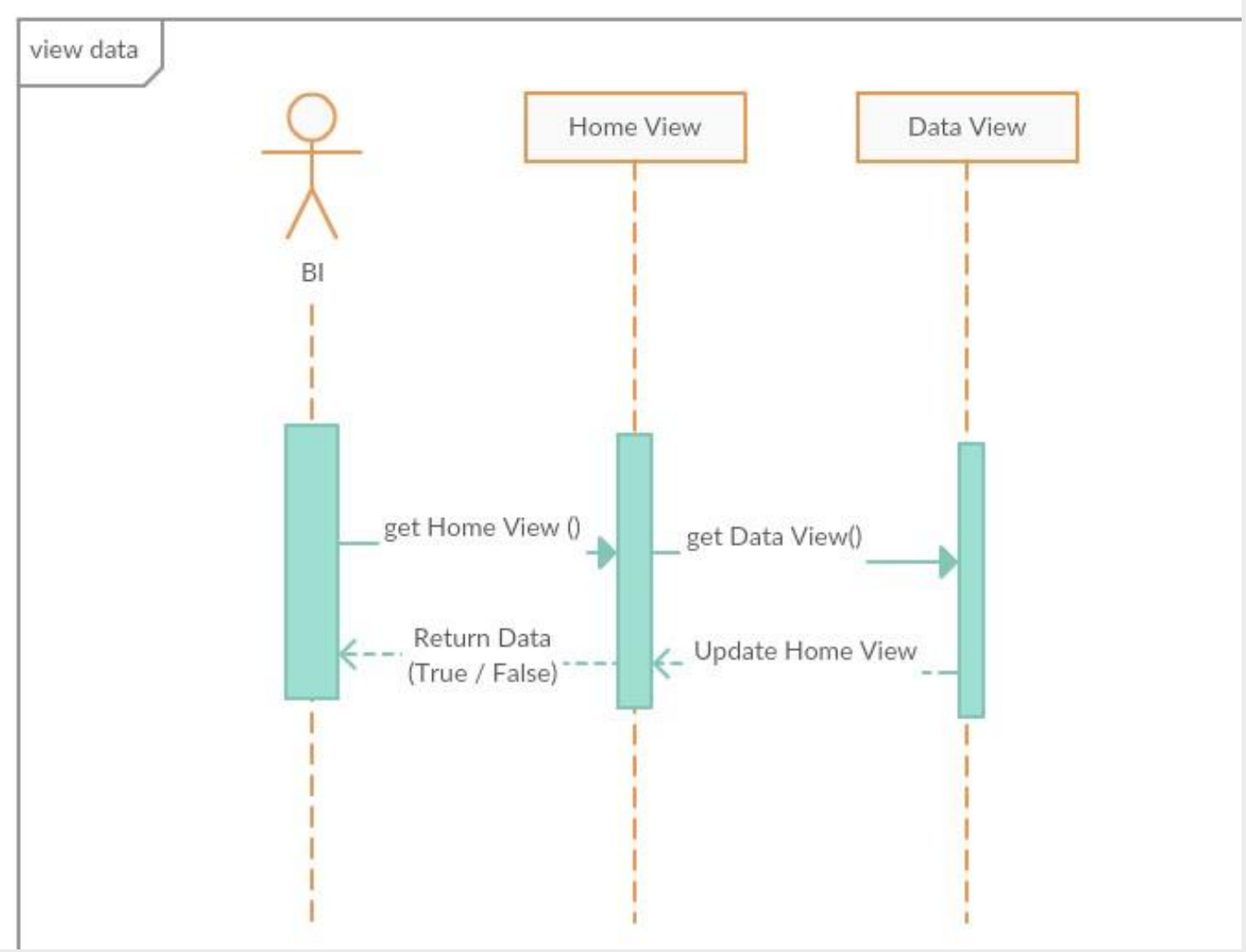
CONTROLLER


K-location query
+ k : int
+ centroid : int
+ X : Int
+ Y : Int
+ eculidyen (x , y) : void
+ draw_trajectories ( ) : void
+ draw_points ( ) : void
Login controller
+ Email : String
+ Password: String
+ login ( ) : void
+ logout ( ) : void
SignUp controller
+ name : String
+ Email: String
- password: String
- Address: String
+ signup( ) : void
+ update ( ) : void
Statistics Controller
+ time_slot : String
+ spatial_range : Int
+ export statistics ( ) : void
Data controller
+ longitude : Arraylist
+ latitude : Arraylist
+ date : String
+ time : Int
+ import data ( ) : void
+ search data (X , Y) : void
Statistics
+ time_slot : String
+ spatial-range : float
+ draw graph (void) : return

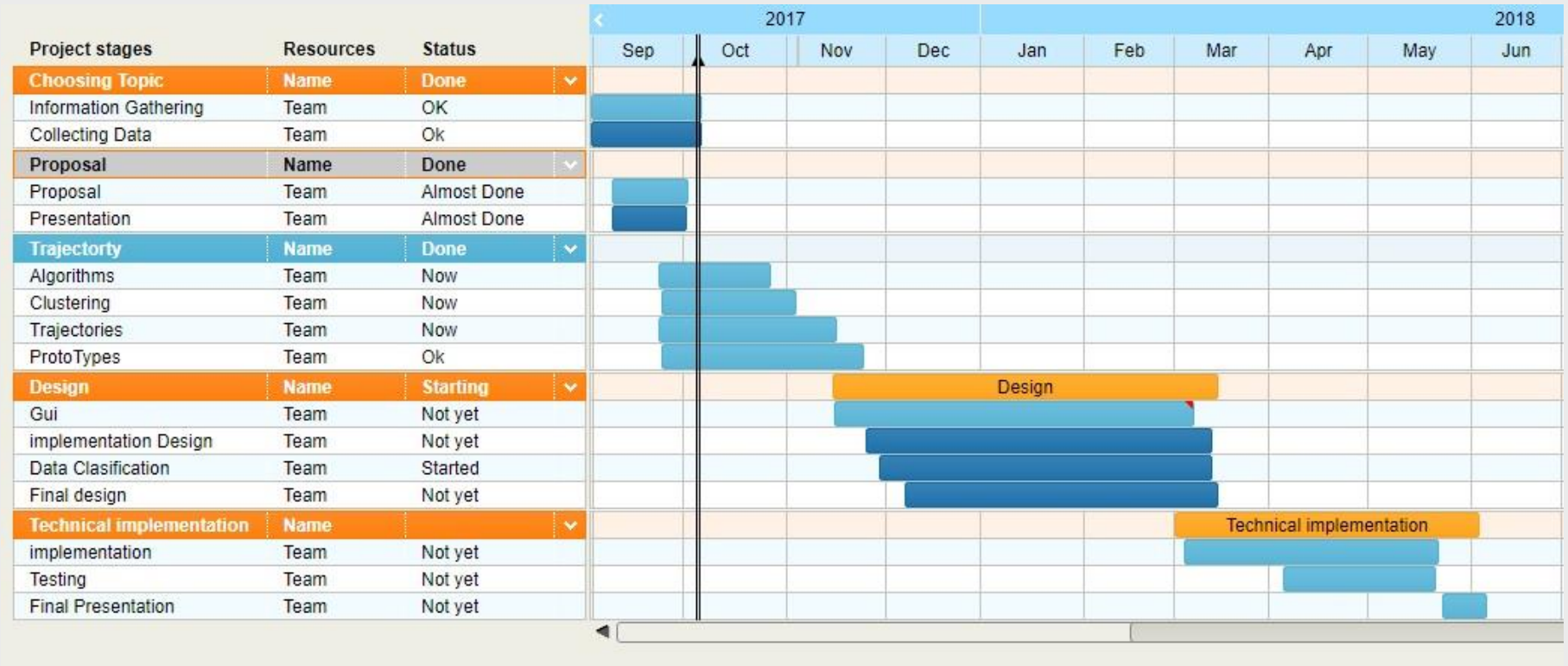
DRAW TRAJECTORY SEQUANCE



VIEW ALL DATA

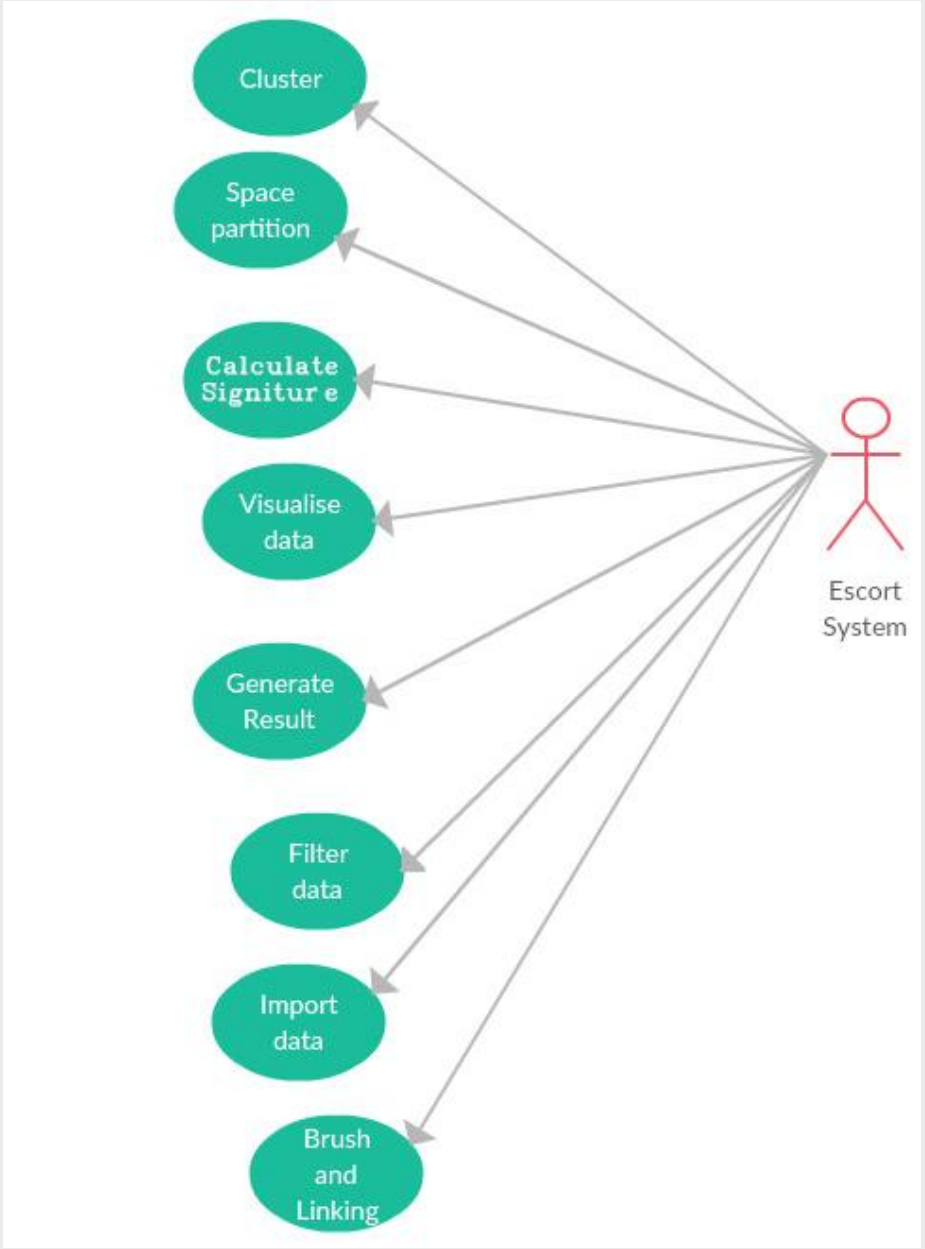


TIME PLAN

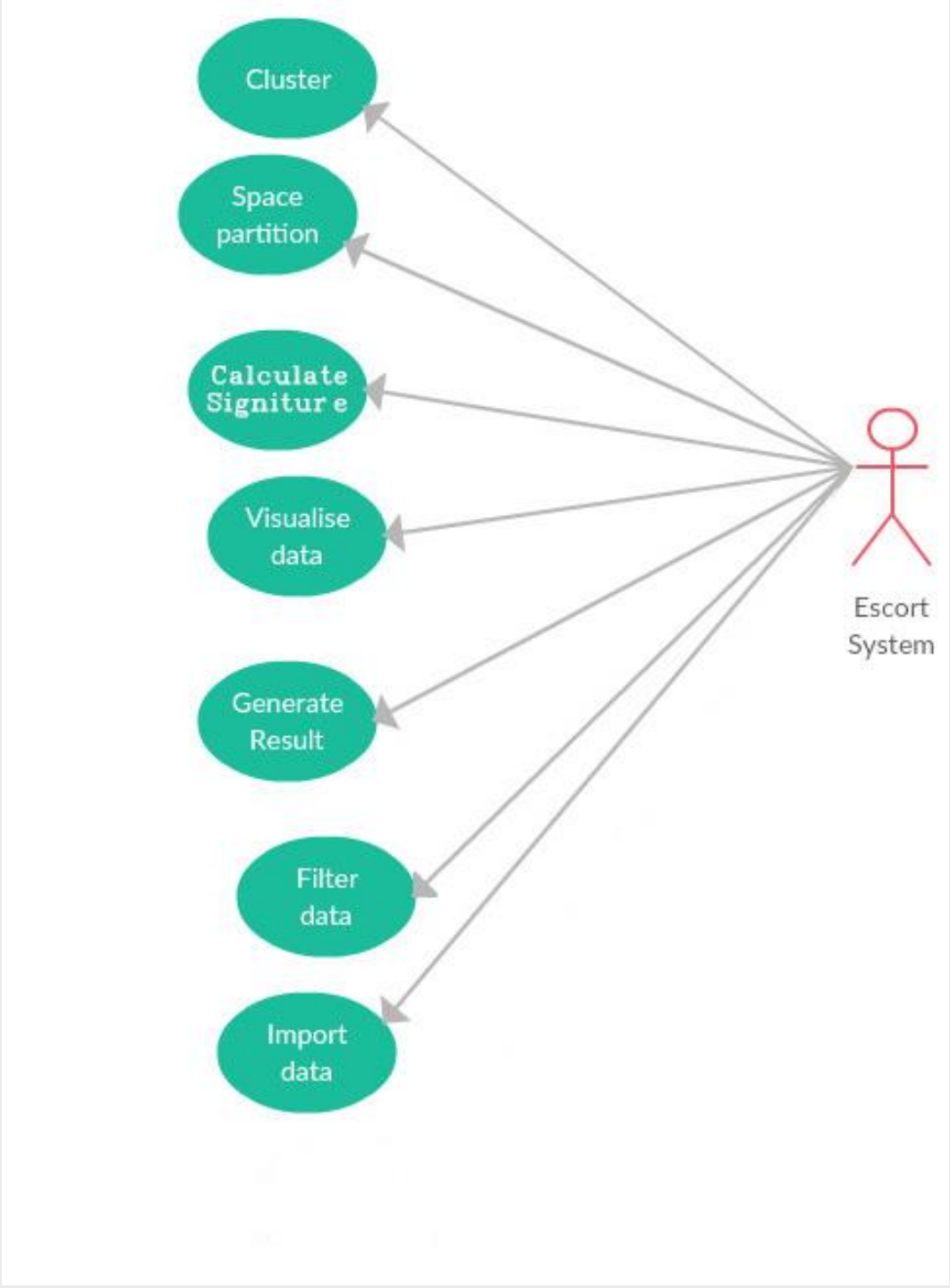


Project stages	Resources	Status
Choosing Topic	**Name**	**Done**
Information Gathering	Team	OK
Collecting Data	Team	Ok
Proposal	**Name**	**Done**
Proposal	Team	Almost Done
Presentation	Team	Almost Done
Trajectorty	**Name**	**Done**
Algorithms	Team	Now
Clustering	Team	Now
Trajectories	Team	Now
ProtoTypes	Team	Ok
Design	**Name**	**Starting**
Gui	Team	Not yet
implementation Design	Team	Not yet
Data Clasification	Team	Started
Final design	Team	Not yet
Technical implementation	**Name**	
implementation	Team	Not yet
Testing	Team	Not yet
Final Presentation	Team	Not yet

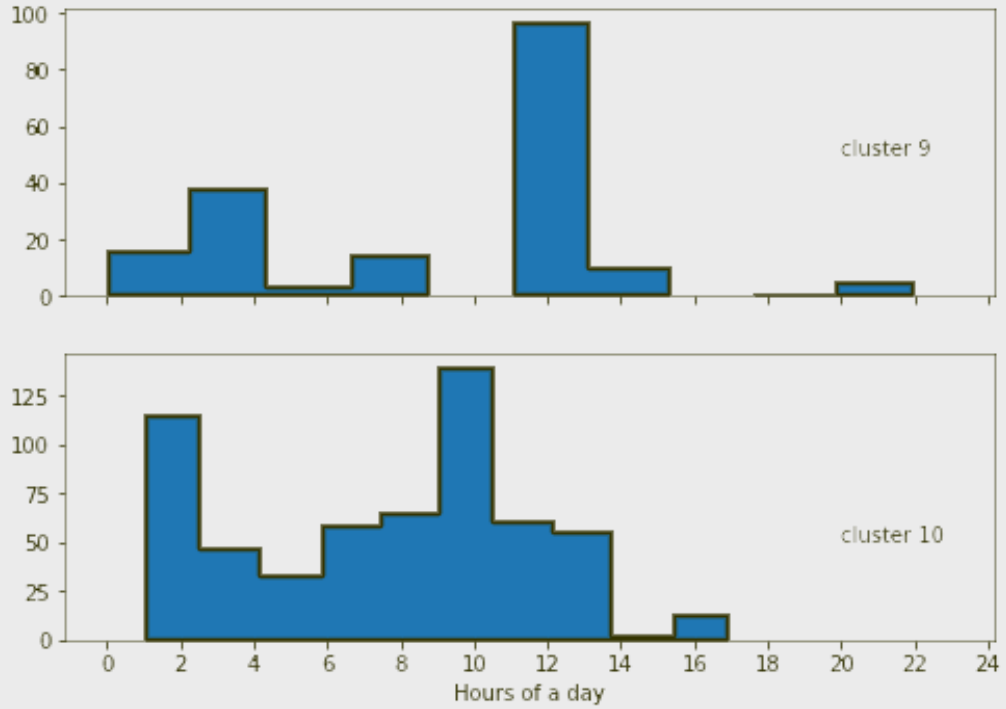
E CASE 1/2


Cluster
Space partition
Calculate Signitur e
Visualise data
Generate Result
Filter data
Import data
Brush and Linking
Escort System

E CASE 2/2


Cluster
Space partition
Calculate Signitur e
Visualise data
Generate Result
Filter data
Import data
Escort System

RESULTS OF TEMPORAL VIEW



THANK YOU