# Spatio-Temporal Data Mining based Task Reminder System

## Final Year Project – Mid Report

### Session 2016-2020

A project submitted in partial fulfilment of the

COMSATS University Degree

of

BS in Computer Science / Software Engineering (CUI)


Department of Computer Science

COMSATS University Islamabad, Lahore Campus

14 May 2020

## Evaluation

| **Project Group Members** (To be filled by students) | | | | (To be filled by supervisor) |
|---|---|---|---|---|
| Sr.# | Reg. # | Student Name | *Signature | Obtained Marks (Total Marks: 10) |
| (i) | FA16-BCS-037 | Abdul-Dayyan | | |
| (ii) | FA16-BCS-184 | Soha Bilal | | |
| (iii) | | | | |

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others.

Supervisor Name: Dr. Hamid Turab Mirza Supervisor Signature: _____

Remarks (if any): _____

_____

_____

# Abstract

People of today's age and specifically businessmen have been that much busy that they are really unable to remember what tasks they had to accomplish at a specific location. And there comes the need of the reminder system which reminds the user of the tasks to be accomplished based on the past activities and route patterns. But the problem arises when to remind the user of a specific reminder? The user might be busy or may not want to accomplish the task yet or may be too distant from the location. The solution to this problem is the intelligent reminder system which doesn't remind the user based on time like the typical mobile phone reminders does but on the past routine of the user. The problem with the temporal reminder is that it reminds the user on a specific time. The location based reminder is not also very efficient. The user needs to set all of the reminders before getting any reminder. And location based reminders work on the basis of Euclidean distance which is not efficient. Whereas, the route patterns based reminder works efficiently. The route pattern based reminders consider the actual routes of the user and then perform data mining algorithms on them to determine where the user travels frequently. So, they are efficient compared to the temporal reminders and LRBs. This project aims to build the reminder system with route patterns mining in mind. For the data mining algorithms, any of the sequential pattern mining algorithms may be used. But this project aims to use PrefixSpan which is derived from the FreeSpan algorithm which is a sequential pattern algorithm. Furthermore, CRPM (Continuous Route Pattern Mining) algorithm is used to record the trajectory data of the user. The route patterns are extracted from that trajectory data in the form of Geo Hash codes which is the input of the PrefixSpan algorithm to extract the frequently travelled patterns of the user. The users doesn't need to set reminders. And the system actually calculates the distance except calculating by Euclidean distance. It determines the probable next routes of the user by the trajectory data stored of the user. Then it can calculate the exact road distances between the user and the destination.

# Table of Contents

# List of Tables

# List of Figures

# 1 Introduction

Today's era is the 21st century, an era where human beings have started being dependent on computers. It would not be wrong to say that humans have been surrounded by the machines, instead of machines are surrounded (used) by them. People are becoming fully dependent on the Artificially Intelligent computers which can think like humans and take decisions like a human brain. All of this has become possible just due to the emerging field of Data Science. While there are self-driving cars implemented by Google Inc.,[1] Hollo Lens by Microsoft[2], voice assistants like Cortana and Siri [1], self-learning algorithms in Neural Networks [2] which executes and train themselves like human brain does, there is also a need to develop a system which can remind the user of the tasks based on the daily routine.

With the emergence of the fields of Data Science, Data Mining and Big Data which are all tightly coupled, software engineers started to develop applications based on users' activities and behaviours. And with the increasing demand of reminder systems which most of the people use to get a notification from their electronic gadgets mostly mobile phone or smart watch, the scientists planned to develop an intelligent reminder system based on the user's routine routes and activities. The simple time-based reminders systems are not much helpful because even the user doesn't know when he/she will be at a certain location. An intelligent reminder system notifies the user of the set reminders based on the collected data from the user's daily routine and not on a specific time. Instead of setting the reminder based on a specific time, or setting a reminder at all, the system will learn from the user's daily routine, the paths of travelling and the activities and then it will notify the user of a specific tasks after learning from the routine.

There are usually three types of reminder systems: temporal reminder system which is simply a time based reminder system [3], location based reminder system which reminds the user of the set reminders based on the location and future based reminders which learn from the route patterns of the user and then remind the user.

In this project, we will be developing a very similar intelligent reminder system which will access the user's location using GPS and will teach itself about the user's routine and route patterns. The system will first store the patterns of the user's routes, which turn the user takes from a specific road to determine the routes the user uses the most. The user will first have to store the locations to get reminders to those locations before going there after learning from the user's routine. Then it will store the user data of all the frequently used routes. Then the system will analyse and determine the next probable task of the user and will trigger the message based on what the user does on that specific time of the week in a month or some other time span.

There are two ways to trigger the reminder based on the location: either by calculating a Euclidean distance and setting a threshold distance or by storing the user routes and

[1] https://waymo.com/
[2] https://www.microsoft.com/en-us/hololens

drawing the next probable routes of the user by using the data mining algorithms. The Euclidean distance is the straight line distance between two points on the plane. The Euclidean way of triggering the notification is not very efficient as it only calculates the distance between the user and the set location without considering the routes. It is possible that the user is near to the location, the reminder is triggered and there is no road to the destination. So, the system will be storing the data of the user's routes and will apply the data mining algorithms on the dataset and will consider the next estimated tasks of the user to remind the user of a task. Before this, the user will store the locations frequently visited so that the system can specially keep record of route patterns of those locations.

# 2 Objectives

The main end of this project is to provide users with a system which can remind them of their set reminders based on their location. However, the goals and objectives to be achieved by this project are as follows:

- Remind the users of the set reminders based on their past routine
- Using spatio-temporal datasets instead of Euclidean datasets to improve accuracy
- Using the different settings for different users to activate notifications depending upon their daily routines and activities.
- Easy to use design and interface

# 3 Problem statement

The main problem with most of the reminder systems is that they are temporal reminders. They are out of the race initially. Then there comes location based reminders which remind the users based on their location. But the problem arises that how much times they provide the user with the accurate reminder on accurate time. How many times they trigger the reminder when the user really need it. And how many times the decision of the reminder system gets wrong. The accuracy of the location based reminders is the case under discussion. There are some reminders built like PlaceIt [4] and PlaceMail [5] which work totally on the basis of Euclidean distance. They simply set the reminders of the specific locations. And they access the user's location using GPS. A threshold distance is set to trigger the reminder. Whenever the user is near to the location and the distance between the user and the specified location is below the threshold distance, the reminder gets triggered and notifies the user. But the Euclidean way of triggering the reminder is not correct always. The reminder simply assumes the Earth's plan as a co-ordinate plane system and calculates the distance using the distance formula used in geometry:

*Distance between user and location:* $|AB| = sqrt( (x_B-x_A)^2 + (y_B-y_A)^2 )$

We need a system that could also store and mine the user's routine patterns and then combine it with location based reminder so that it can trigger the reminders in a more accurate way. For this we need the spatio-temporal dataset of the user of at least two weeks in order to determine his routine patterns. Then we can combine the user's frequently visited patterns with the location based reminders [6] to improve the time and location of triggering the reminder. An application named as "iReminder" [7] is built using this way and is probably the only application built using the data mining technique.

How the routine patterns of the user can be mined once the data is collected? For this we need a data mining algorithm. The algorithms based on the Sequential pattern mining techniques are Apriori [8] and the others derived from it such as GSP [9], FreeSpan [10], SPADE [11], PrefixSpan [12] and CRPM [13] algorithm which is derived from the PrefixSpan [12]. The Apriori [8] and GSP [9] are not efficient as they do a lot of calculations. They first calculate all of the candidate subsequence and then mine the frequent ones from them. This takes a lot of time and computer memory. The FreeSpan [10] and its improved version which is PrefixSpan [12] work more efficiently. They calculate the frequent items in first iteration. Then the algorithm uses these frequent items as prefix and calculate the suffixes to find the 2-item frequent subsequence. The CRPM (Continuous Route Pattern Mining) [13] algorithm further enhances the capabilities of the PrefixSpan [12] and outperforms all of the other algorithms. So the proposed system will either be working on PrefixSpan [12] or CRPM [13].

## 3.1 Assumptions & constraints

Although the route pattern mining based reminders outperform the classic location based reminders but at a cost; the user either first needs to get his data stored in the application by recording the data into the application or provide already recorded data to the reminder so that it can process the data and calculate the user's frequent patterns. So there is a constraint of first recording the user's routine patterns before getting benefit from the reminder. So one assumption is that the data will be already provided to the system or the user will need to wait for first two weeks before actually getting benefit of the more accurate routine pattern based reminders.

## 3.2 Project scope

The system can be used by anyone who wants an efficient reminder system and wants to get the messages delivered in a more accurate way than the other systems built to date. The user needs to provide his geolocation data so that application can work on it and find the

user's frequently visited patterns. The system's performance is limited by the fact that it needs to be trained first before using it unlike other location based reminders which don't work on the basis of frequent pattern generation. And the system will try its best to implement the user's routine patterns and integrate with the location based reminder technology to provide the maximum possible results. And of course, its efficiency will be far better than traditional location based reminders but it won't provide 100% accurate results. In fact, no system can implement 100% performance. Although the accuracy of the system can be improved more and more.

# 4 Requirements Analysis

## 4.1 Literature review / Existing system study

People of these days are busy that they forget to do daily routine tasks. There must be a hint to let them remind about these tasks. According to Donald A. Norman, the two aspects of reminding something are:

1. The Signal (something to remember)
2. The Message (what to remember by that signal)

The reminders can be divided into three categories: time based reminders which just set reminder for a specific time without keeping record of the user's position and activities, Location Based Reminders (LRBs) which remind the user of a task based on his location & Routine patterns based reminders which are more intelligent than any other kind of reminders because they keep track of the user's routine and deliver reminder at very accurate position and time keeping a view on his routine. The time based reminder systems are not much accurate and are not used anymore because every routine is not dependent on time. Some tasks are done by the current location. Location based reminders are helpful and are more demanding these days. But do they deliver message on accurate time? Not every time. So we need to improve the accuracy of location based reminders. And there comes the idea of spatio-temporal data based reminder systems which are actually integrated with the location based reminders to improve their accuracy.

Location Based Reminder Systems (LBRs) are not much accurate due to the reason that they calculate the distance between the user and the task location using the Euclidean distance formula. The Euclidean distance formula considers a straight line distance between two points without noticing any roads or tracks. When the distance between the user and the task location is below the threshold distance, the message is delivered to the user without caring about whether there is a road to the location or not.


*Figure 1: Euclidean Based Reminder showing Threshold Perimeter*

A study by the Sohn et al showed that people prefer location based reminders in the nuanced ways. Besides the availability of GPS technology in the mobile phones by the start of 21st century also helped in realising the importance of LBRs. The first location based reminder for the iPhone users was the ***Vittici*** which was launched in 2011 at the time of iOS 5. Then first location based reminder for Android was built in 2012 named as ***"Yepa"***. Some other applications have also been built using the location based technology.

**comMotion:**

In 2000, Schmandt and Marmasse built comMotion, an application which is installed in the user's mobile or any wearable computer. The application learns from the user's travelling and places to visits. The more it save the places as time passes, the more accurate results it provide. The user can set reminders on it and even send emails. He/she can record a voice message through it and then set a place to let it remember. The interaction with the application is possible using two different ways: either using the voice command and text-to-speech or simply using the interface of the application. And it provides the reminders then and there. It means that it will only remind the user of shopping some grocery if he

will be passing outside of some grocery store. So its results are very precise and accurate. And it learns all of this from the user's travelling locations.

**iReminder:**

iReminder is probably the first reminder application which mines the user's routine patterns and then suggests the reminders based on the predicted future routes user mat take while travelling. The main difference between the traditional reminders and iReminder is that traditional reminders triggers the notification of the reminder when the user enters a virtual field around the location set as the perimeter for triggering the message while the iReminder first predict the user's next probable routes by learning from the data of past routine patterns and triggers the message only if user is going near to the location and location will be in his or her predicted future route. It outperforms almost all of the currently available reminder systems. It mines the routine patterns using the Continuous Route Pattern Mining algorithm (CRPM).


*Figure 2: iReminder System, Finding Patterns by Mining User Patterns*

## 4.2 Stakeholders list (Actors)

The stakeholders of the system includes every person who is interacting with the application directly or indirectly. From design to test to use the application actually, all of the persons who have interacted with the application can be included in the stakeholders list. Some of the stakeholders are as follows:

- The customers of the application who want to get the application build for use. They may use or not use the application at all. They may only want to get it developed for other people.
- The investors who might be interested in investing in the project
- The software engineers who are interacting with the customer and specifying the problem and requirements
- The designers of the application who are creating the user interfaces for the application
- The development team of the system
- The testing and maintenance team
- The test users who are willing to let their data record and wants to contribute in testing the performance of the application in real world before getting deployed
- The actual users of the application

## 4.3 Functional requirements

| Req. ID | Requirement | Description | Priority |
|---|---|---|---|
| ST_R_01 | Sign in | The entry point of the application. It is a window to enter the user name and credentials to get the user access the application. | 1 |
| ST_R_02 | Forget Password | It is the feature to let the user change the password in case he/she forget it. It may require some security questions or alternative credentials. | 2 |
| ST_R_03 | Sign up | This functionality is used for the new user of the application to get them registered. The users need to complete a form by entering their personal information and setting up a password for their account. | 1 |
| ST_R_04 | View Reminder | This window is used to view a specific reminder along with its details such as title, description, reminder message, date and time created. When the user clicks on a reminder from the reminders list, this window pops up. | 2 |
| ST_R_05 | Add Reminder | This feature is used to add new reminders. The user can simply add details and click on the create button to create a new reminder. | 1 |
| ST_R_06 | Edit Reminder | This feature is used to edit the existing reminders created. | 1 |
| ST_R_07 | Delete Reminder | This feature simply deletes an existing reminder in case the user don’t need it. | 1 |
| ST_R_08 | Sign out | This option is used to logout the user’s account from the application. | 2 |
| ST_R_09 | Upcoming | This window will contain all of the upcoming reminders’ list in ascending order. The user can click on any of them to see its details. | 3 |
| ST_R_10 | Calendar | This option will also contain all of the upcoming reminders but on a calendar. So the user can see all of his reminders of a month at a single window. | 3 |

| | | | |
|---|---|---|---|
| ST_R_11 | Settings | This option is used to change the settings of the application. The user can change ringtone, application look and feel and other settings related to reminders. | 2 |
| ST_R_12 | User profile | This option can be used to maintain the profile of the user. The user can update his profile like name, password, profile picture and other info. | 2 |
| ST_R_13 | Multiple logins | This feature of the system will allow user to login to the multiple devices simultaneously. So if the user uses more than one phones or wearables, he can login to all of them to ensure the availability of the reminder service. | 3 |

*Table 1: Functional Requirements*

| Priority | Value |
|---|---|
| High | 1 |
| Medium | 2 |
| Low | 3 |

*Legend: Priorities*

## 4.4 Non-functional requirements

According to IEEE, the non-functional requirements are those requirements which don't explain what the system will do but these are used to explain how the system will perform them.

- **Easy to use interface**

The interface of the application should be easy to use and understand. It should be that much easy to adopt by the novice users in very less time. As the application would also be used by users who are not familiar with the complex application designs, so the application should be designed by keeping in view those users first.

- **Usability**

The application should identify all of the user's needs and should be able to achieve all of those need effectively and efficiently. Usability is a combination of the many factors but all can be combined to achieve targets like ease of learning, memorability and efficiency etc.

- **Accessible anywhere**

According to this non-functional requirement, the application should be easy to access and should be accessible anywhere. The user can login to multiple devices simultaneously to get reminders on multiple devices by just signing in to his account. The user can access his reminders anywhere on the go by just one of his account.

- **Effectiveness and efficiency**

The application should be effective and efficient to use. The effectiveness means that it performs all of the required tasks. Efficiency means that it performs all of those actions in minimum effort and time.

- **Deliver reminders on accurate date and time**

The application should be fast enough to provide all of the reminders on accurate place and date & time before the arrival of the user. Otherwise, that reminder would not be useful for the user.

## 4.5 Software development life cycle model

Software Development Life Cycle is a series of steps that are performed in a sequence to develop a working system. This process is followed by software organizations in developing successful software. Each process contains some specific tasks and time required. Different type of SDLC models may perform these activities in different sorting depending on the type of the software. The typical phases of SDLC include:

1. Problem definition
2. Specify requirements and analysis
3. Design the software
4. Building the software according to design
5. Testing
6. Deployment and maintenance

The model we will be using for developing our application is the ***Evolutionary Prototyping Model.*** The reason for using this prototype is that it requires much client involvement as our system will also need the involvement of the client and interaction with the users to keep knowing if the application is working fine or not. Another reason for choosing this prototype is that we can iterate in the design process until we reach at the design we want. So rapid & iterative development and evaluation are also the reasons for choosing this prototype.

**Advantages:**

- This model is very suitable for developing Research based projects. It starts from a minimalist design and idea. And then increases the scope of the software. For example, in Artificial Intelligence like speech recognition, it is possible to take a small vocabulary and develop a system. After successful implementation, you can gradually increase your vocabulary.
- If the developers are working at a big project, they can stop the project in between. And they can resume anytime from the previous working prototype.
- It is cost efficient compared to the other software development models.

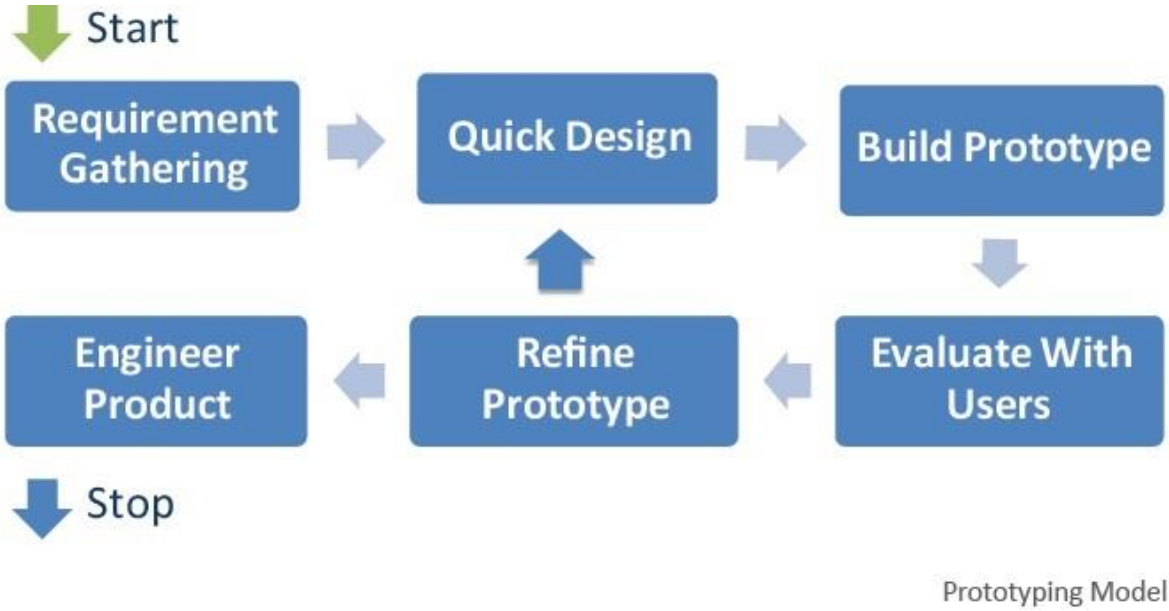


*Figure 3: Evolutionary Prototyping model*

# 5 System Design

## 5.1 Activity diagram

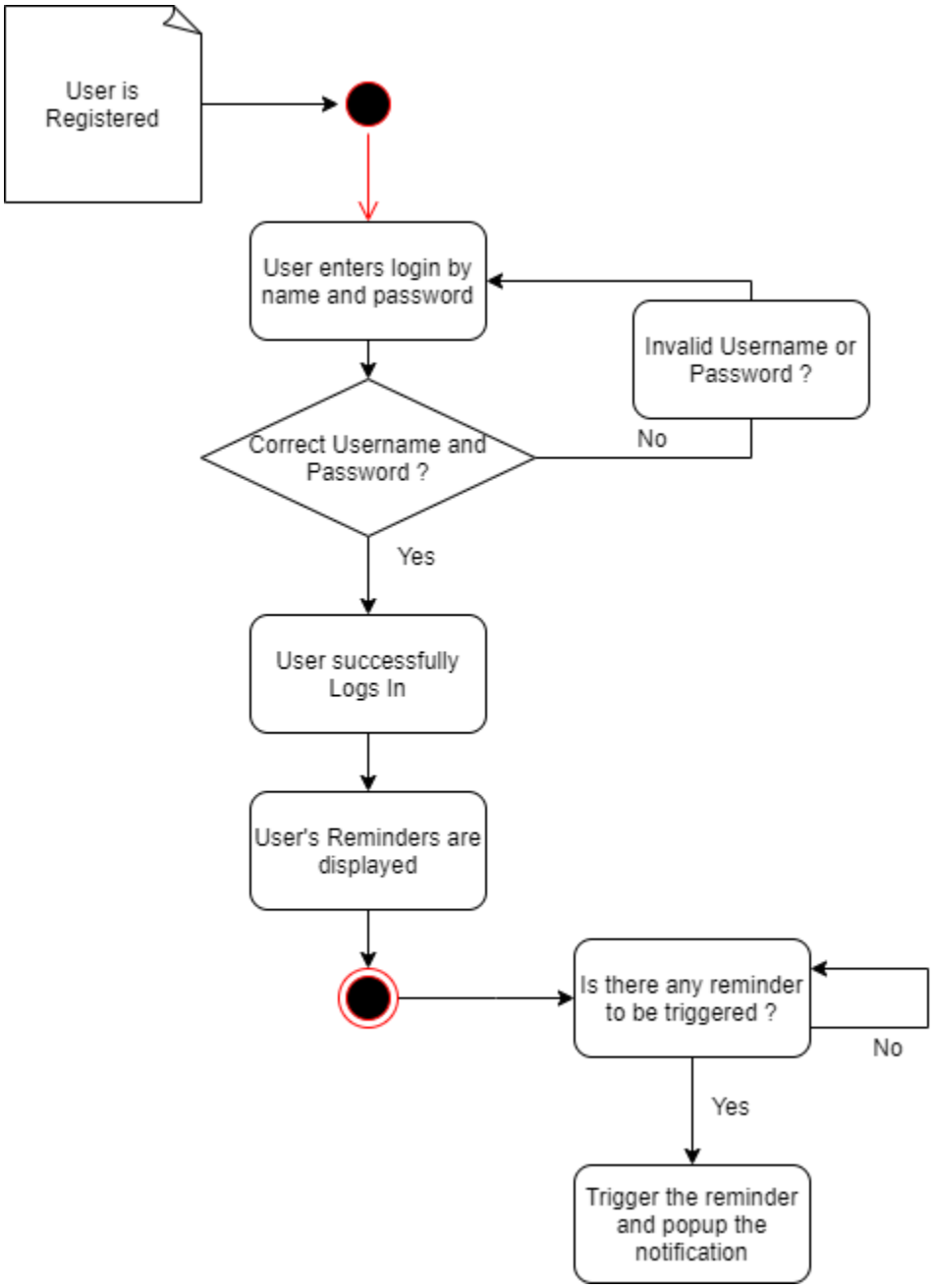


*Figure 4: Activity Diagram of the System*

The activity diagram is used to show the possible activities performed by the user in a specific sequence in the form of a flow chart like diagram. Each activity is represented in an activity window. The activity windows are connected by arrows which represents the

flow and sequence of activities performed after one another. The diagram shows if the user has signed up by creating an account and registering with the application, then he needs to verify his/her account by entering his name and password. If the name and password entered is correct, then the user successfully logs in, otherwise the login fails and the user can retry logging in to the account. After logging in, the reminders of the user are displayed on the screen. The user needs to quit from the application without signing out, so his account kept logged in, and if there is any reminder to trigger, it gets triggered as shown in the figure 4.

## 5.2 Software architecture

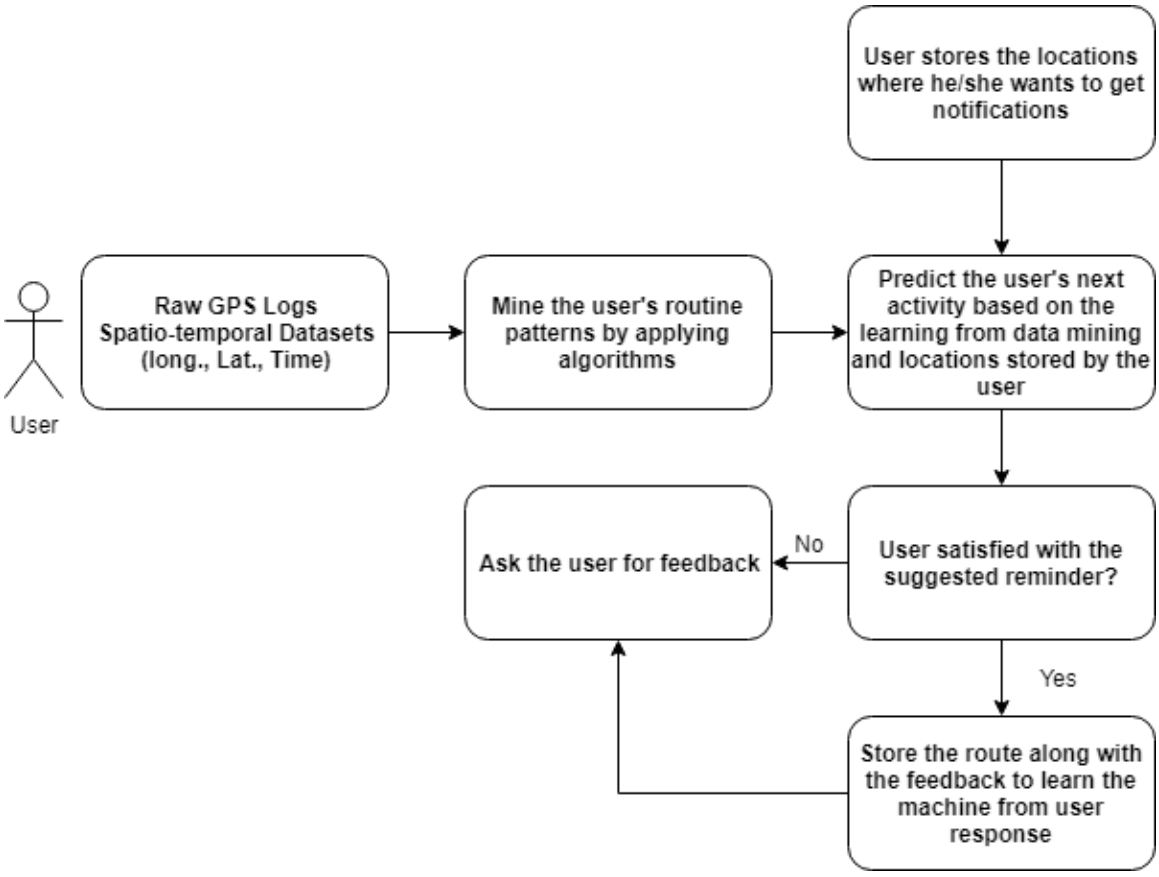


*Figure 5: Software Architecture*

The software architecture is used to show the main functions implemented during the implementation of the system. The set of functionalities and activities performed during the implantation of the system are shown in figure 5 whose detail is as follows:

- **Raw GPS Logs Spatio-temporal Datasets**

In this phase, we gathered the users' log GPS (Global Positioning System) data. Actually spatio-temporal data was required to keep record of the user activities and the route patterns the user travels the most. The data was gathered using the Google Timeline which keeps the places where the users visits each day. Then we can download the data in KML format. Then the data can the converted from KML to CSV using an online conversion tool. The CSV file contained other fields along with the required fields such as description, Name, Altitude, etc. The data contained a total of 18 fields.

- **Mine the user's routine patterns by applying algorithms**

After the data had been recorded using the user's GPS location, it had to be mined by applying the data mining algorithms to draw out the patterns of interest from the dataset of the activities of the user. The patterns are very beneficial for generating and setting the reminders for the user so that the user doesn't need to set the reminders for himself. For the mining of algorithms, the sequential pattern mining technique can be used which can mine the patterns which occur in a sequence extensively. The Apriori algorithm is used for the mining of sequential datasets. There are algorithms which have been derived from the Apriori algorithm such as GSP (Generalized Sequential Patterns), FreeSpan, etc. Then, PrefixSpan is further an extension of FreeSpan algorithm which outperforms all of these algorithms and find patterns at a very low cost and also save a lot of computer memory in comparison to other algorithms such as Decision Tree Algorithms which generate so much binary trees in the memory to perform the data mining. So the project used the PrefixSpan Algorithm to mine the patterns. Before mining the patterns, the geo hashes of the trajectory data was required which was generated by the CRPM (Continuous Route Pattern Mining) algorithm which has been derived from the PrefixSpan algorithm.

- **Predict user's next activity based on data and location**

For generating the reminders automatically for the user, the prediction of the next probable route and current location is required so that we can determine which turn the user is going to take next. The location can be accessed using the GPS technology in the mobile of the user. The data is already stored from the user's past activities and so by applying the data mining on the previous activities and using the current location, the prediction of the next probable route is possible.

- **User stores the desired locations**

The user will need to store the desired location where he/she is trying to get the reminders and travels frequently. So for those locations, there is no need to set reminders for such locations. And the system will set the reminders of these location and nearby places itself in case the user forgets and too busy to set the reminder.

- **User satisfied with the suggested reminder?**

Is the user satisfied with the reminder he/she gets finally after the mining of locations of interest and predicting the next probable route? If the answer is yes, then system will store this reminder for future use. If the user is not satisfied and reminder is not accurate, then the system doesn't store it and reinforces itself to learn from this mistake so that next time it doesn't generate such kind of reminders for the user. Hence, the system learns from the user's behaviour. The system also asks the user why the suggested route or reminder wasn't of the interest so it can also learn from that response of the user and it avoids generating such reminders in future.

## 5.3 Class diagram

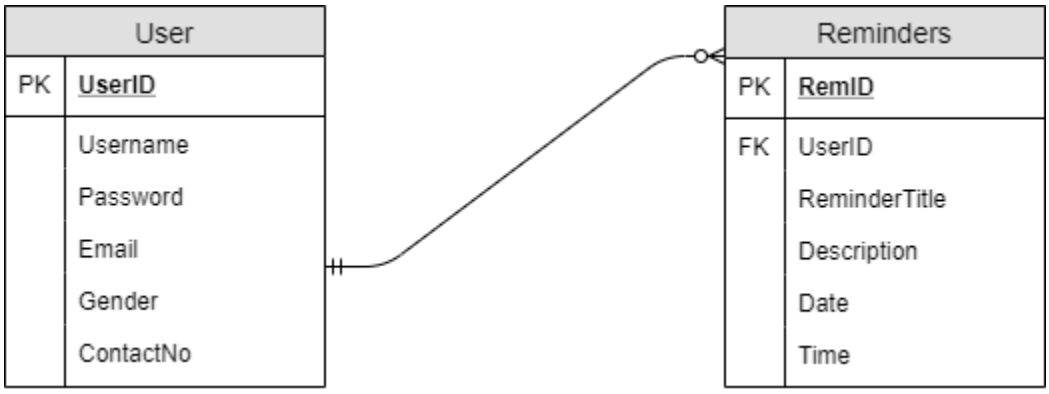


*Figure 6: Class Diagram showing Relationship b/w Classes*

The class diagram shows the possible relationships in a database just like the database diagram does. It shows all of the main classes and the relationship between those classes. It can be shown from the figure 6 that the system will mainly have two relations, one for the users and other for their reminders generated. Each user is assigned a unique ID while creating an account. This ID is created automatically. One user can have many reminders or no reminder at all yet in case if the user has just created an account. Each reminder created for the user also gets a unique ID. And the user ID is also associated with each reminder so as to keep record of which reminder belongs to which user in the database.

## 5.4 Network diagram

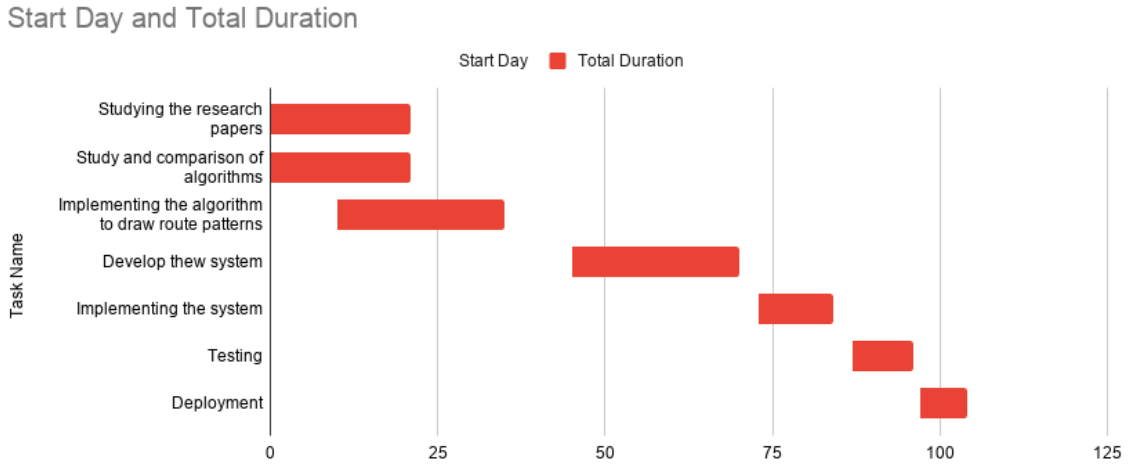


*Figure 7: Network Diagram (Gantt Chart)*

The network diagram is just like a Gantt chart except that it keeps record of what has been done to date and what needs to be accomplished yet. The network diagram is showing that the group members will be studying the research papers for further discussion of algorithms and the application design. Then further algorithms will be implemented to compare their performance and then the best algorithm will be chosen. Then the system will be developed followed by the implementation, testing and deployment of the application which is shown in the figure 7. Until now, we have planned the system, analysed it, studied the research papers in this field and implemented some algorithms. We have also recorded the users' data and pre-processed it.

## 5.5 Implementation

### 5.5.1 User Interfaces

The prototypes for the design of the User Interfaces have been generated using a Prototype application called Justinmind Prototyper which was even then evaluated by many evaluation techniques to find the abnormalities in the design such as Heuristics evaluation for the UIs. The screenshots of the interfaces generated as a prototype are as follows:

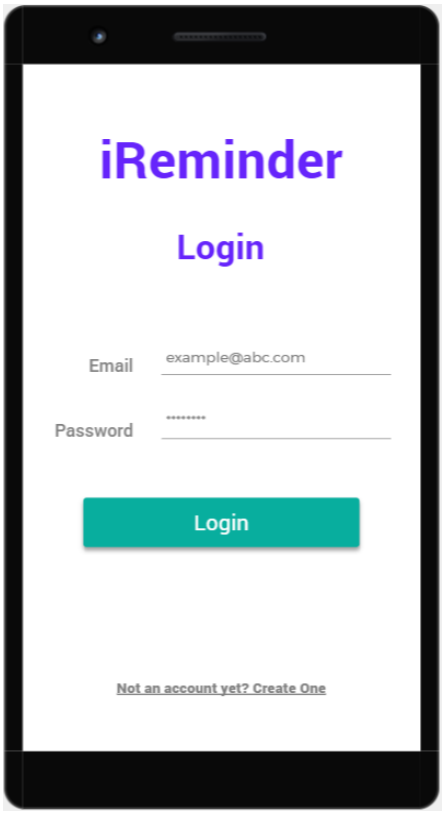


*Figure 8: Sign In Screen*

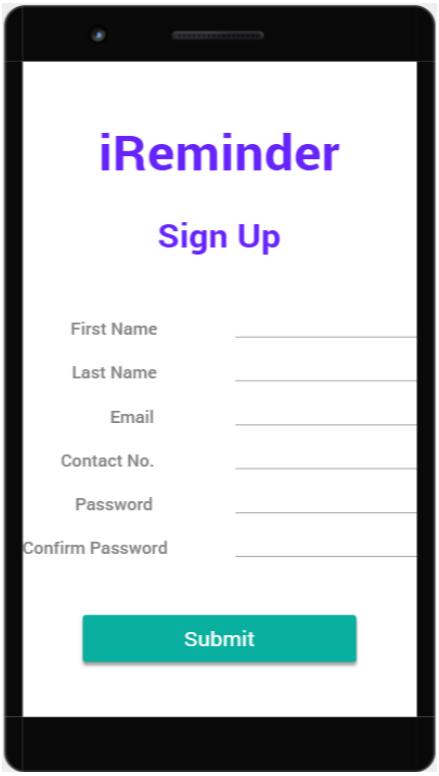


*Figure 9: Sign up screen*

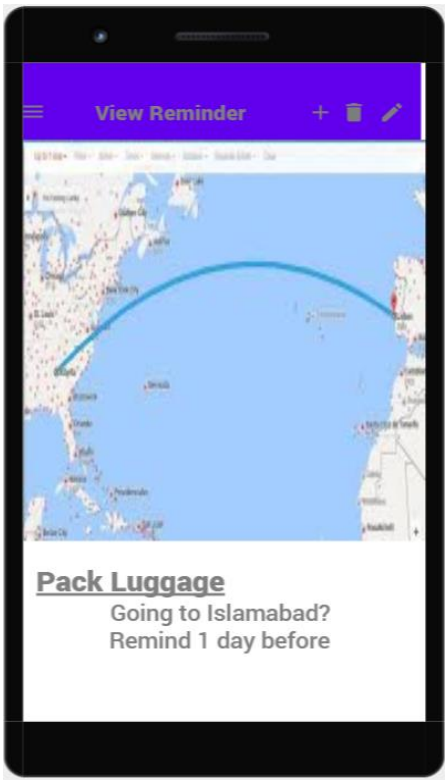


*Figure 10: View Reminder screen*

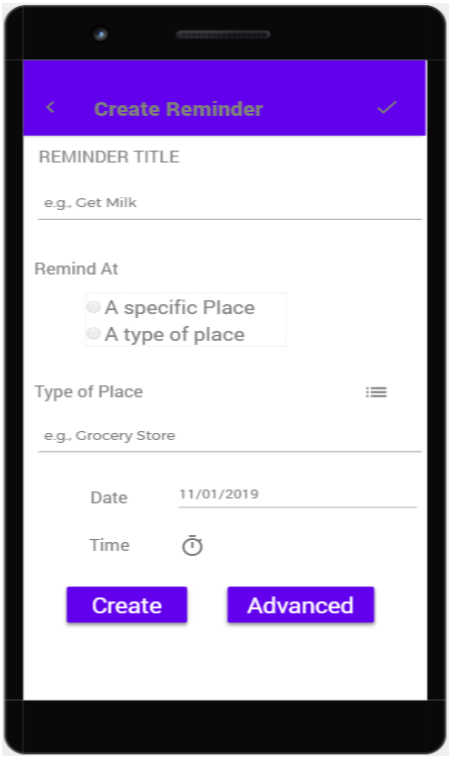


*Figure 11: Create Reminder screen*

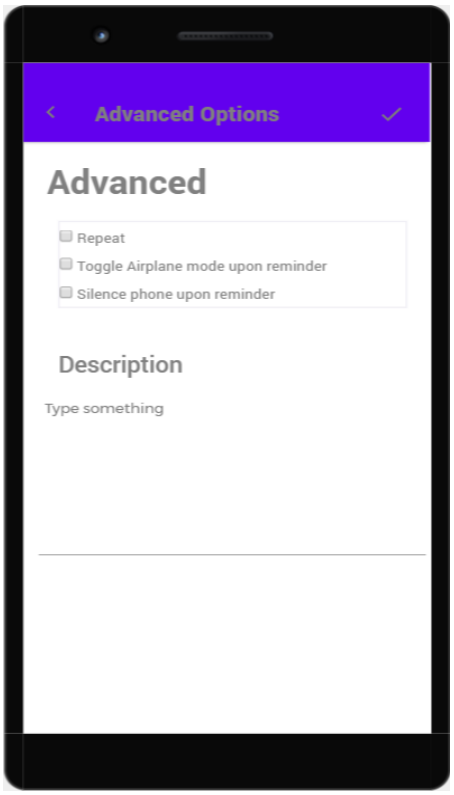


*Figure 12: Advanced options for creating reminder*

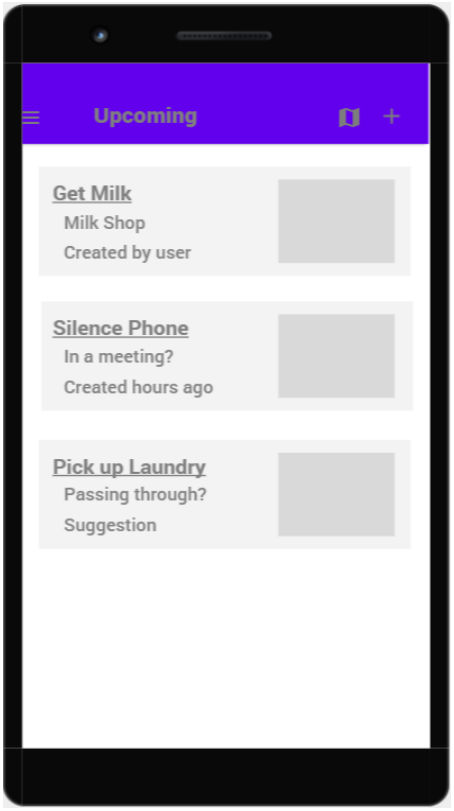


*Figure 13: List of all upcoming reminders*

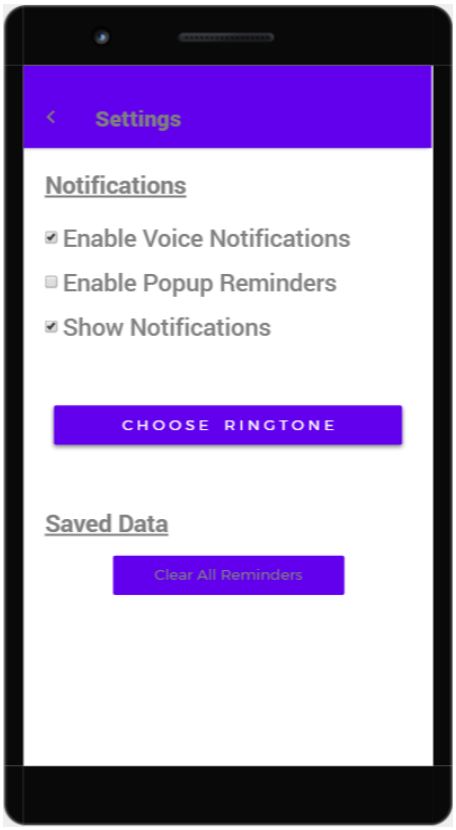


*Figure 14: Settings window*

### 5.5.2 Algorithms

Until now, the only algorithm used is the PrefixSpan algorithm which takes the geo hash codes as input and generates the frequent patterns. But we recorded the user's GPS data from Google timeline, processed it in a form to get it accepted by the PrefixSpan algorithm. So, a lot of data pre-processing was required to make the data in workable form for the algorithm. The algorithm is as follows:

**Input:** A sequence database *S*, and the minimum support threshold *min_sup*

**Output:** The complete set of sequential patterns

**Method:** Call PrefixSpan(<>, 0, S).

**Subroutine:** PrefixSpan(*a*, *l*, $S|_a$)

**Parameters: *a*:** a sequential pattern; ***l***: the length of a; **$S|_a$:** the a-projected database, if $a \neq <>$; otherwise, the sequence database S.

**Method:**

1. Scan $S|_a$, once, find the set of frequent items b such that

(a) b can be assembled to the last element of a to form a sequential pattern; or

(b) <b> can be appended to a to farm a sequential pattern.

2. For each frequent item b, append it to *a* to form a sequential pattern *a*', and output *a*';

3. For each *a*', construct a'-projected database $S|_a$ and call PrefixSpan (*a*', *l* + 1, $S|_{a'}$).

# 6 Conclusion

## 6.1 Problems faced and lessons learned

This projects aims at building a system which can generate reminders for the users based on daily routine and where they travel. The most important thing for this kind of project is the data of the user, i.e., the system requires the users' GPS log data to mine their routine patterns. So the main problem to tackle for this project was to get the accurate log data of test users. The issue is that no one is willing to get their routine recorded by the application for the sake of privacy. The test users were afraid of their confidential information to be leaked. So the main issue arises is of recording the GPS log data. The second main problem

faced during the project was that the data was calculated using the Google's timeline. The data was in so much raw form. A lot of data effort is required before working on the data. It means we need to pre-process the data a lot before working on it. If the data had been collected using some other tool, that hard work of data pre-processing wouldn't have been required. So the lesson learned from this problem was to always choose such a tool for recording the user data which provide cleansed data. So we can directly start the working on it without the worries of data pre-processing. The final problem faced during the project up till now was the studying of some of the algorithms whose documentation or support was not much available on the internet. Those algorithms should be avoided without the clear and proper documentation.

## 6.2 Project summary

The project is to develop a reminder system which could learn from the daily life routine of the users and after learning from those routines, it can set the reminders for them itself without the effort of the user. Most of the times, users wish to set a reminder for a particular task but failed to set it just for the reason that they forget. It is all because of the busy life of the today's age where everyone is just rushing to its daily life tasks and in this hurry, one forgets to do some important tasks. So this application can remind those important tasks to the user very efficiently and without the collaboration of the user.

## 6.3 Future work

As seen in the network diagram that the work done until now is to plan for the project, analysis of the main problem, getting the log data of the users, studying the research papers related to the area of interest and studying different kind of algorithms as some of these activities are not included in the Gantt chart anymore. Although study of different research papers will still be continued along with the study and comparison of different algorithms which can find patterns from the data. Along this, the studied algorithms will be implemented on the data and tested in the future work. After all of this, the system will be developed and implemented. Then the final testing of the application will be conducted after which the application will be deployed on some local server.

# 7. References

[1]. Matthew B. Hoy (2018) Alexa, Siri, Cortana, and More: An Introduction to Voice Assistants, Medical Reference Services Quarterly, 37:1, 81-88, https://doi.org/10.1080/02763869.2018.1404391

[2]. Andre Esteva, Brett Kuprel, Roberto A. Novoa, Justin Ko, Sussan M. Swetter, Helen M. Blau and Sebastian Thrun (2017) Dermatologist-level classification of skin cancer with deep neural networks, DOI: https://doi.org/10.1038/nature21056

[3]. David Graus, Paul N. Bennett, Ryen W. White, and Eric Horvitz. 2016. Analyzing and Predicting Task Reminders. In Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization (UMAP '16). ACM, New York, NY, USA, 7-15. DOI: https://doi.org/10.1145/2930238.2930239

[4]. Sohn T., Li K.A., Lee G., Smith I., Scott J., Griswold W.G. (2005) Place-Its: A Study of Location-Based Reminders on Mobile Phones. In: Beigl M., Intille S., Rekimoto J., Tokuda H. (eds) UbiComp 2005: Ubiquitous Computing. UbiComp 2005. Lecture Notes in Computer Science, vol 3660. Springer, Berlin, Heidelberg

[5]. Because I Carry My Cell Phone Anyway: Functional Location-Based Reminder Applications - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/The-PlaceMail-web-user-interface-design-Specifying-Message-Delivery-Users-can-specify_fig1_221517760 [accessed 17 Sep, 2019]

[6]. Yao Wang and Manuel A. Perez-Quinones. 2015. Beyond "Geofencing": Specifying Location in Location-Based Reminder Applications. In Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15). ACM, New York, NY, USA, 1767-1772. DOI: https://doi.org/10.1145/2702613.2732780

[7]. You Tu, Ling Chen, Mingqi Lv, Youbiao Ye, Weikai Huang & Gencai Chen (2013) iReminder: An Intuitive Location-Based Reminder That Knows Where You Are Going, International Journal of Human–Computer Interaction, 29:12, 838-850, DOI: 10.1080/10447318.2013.796440

[8] Rakesh Agrawal, and Ramakrishman Srikant. Fast algorithms for mining association rules in large databases. In orge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, Proceedings of 20th International Conference on Very Large Data Bases, pages 487–499, Santiago, Chile, September 12–15 1994.

[9] Hirate, Yu & Yamana, Hayato. (2006). Generalized Sequential Pattern Mining with Item Intervals. Journal of Computers. 1. 10.4304/jcp.1.3.51-60.

[10] FreeSpan: Frequent pattern-projected sequential pattern mining Han J., Pei J., Mortazavi-Asl B., Chen Q., Dayal U., Hsu M.-C. (2000) Proceeding of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 355-359.

[11] Zaki, M.J. SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning **42,** 31–60 (2001) doi:10.1023/A:1007652502315

[12] Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., & Hsu, M. (2001). PrefixSpan,: mining sequential patterns efficiently by prefix-projected pattern growth. Proceedings 17th International Conference on Data Engineering, 215-224.

[13] Ye, Q., Chen, L. & Chen, G. Personal continuous route pattern mining. J. Zhejiang Univ. Sci. A **10,** 221–231 (2009) doi:10.1631/jzus.A0820193

## Spatio-Temporal Data Mining based Task Reminder System

ORIGINALITY REPORT

| 5% | 3% | 4% | 4% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

MATCH ALL SOURCES (ONLY SELECTED SOURCE PRINTED)

2%

★ Niranjan, Utpala, V.V. Krishna, Kanduri Srividya, and V. Khanaa. "An efficient web recommendation system based on modified IncSpan algorithm", International Journal of Knowledge and Web Intelligence, 2011.

Publication