2014-07-11

Integrating Data Mining Techniques and Social Networking into Effective Recommendation Framework for Improved Shopping Experience

Jarada, Tamer Nsr

UNIVERSITY OF CALGARY

Integrating Data Mining Techniques and Social Networking into Effective Recommendation Framework for Improved Shopping Experience

by

Tamer Nsr Jarada

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER IN SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

May, 2014

Abstract

The application of data mining in the shopping domain has received a considerable attention for its key role in improving the marketing quality in the last two decades. The main data mining technique that can be used is association rules mining (ARM) though other techniques like clustering and classification are useful but they are beyond the scope of the work described in this thesis. Market basket analysis (MBA) is the most famous example as an application for ARM. MBA's applications have emerged from retail stores' perspective to gain the benefit. In this thesis, we have designed and implemented a framework that considers the shopping process from consumers' perspective to turn it into an interactive process, speed it up, save money, and keep the environment clean. Our proposed solution, backed by experimental results, discovers the frequent items that are usually purchased by the consumer; this helps us to introduce them as recommended items. Also, it helps in finding the nearest stores and introduces a navigational map to be used inside the store. Moreover, our proposed solution has been integrated with the social network analysis concept to improve the shopping process quality by providing the opportunity to involve family and friends.

Acknowledgements

I would like to express my sincere gratitude, appreciation, and thanks to my great supervisor Dr. Reda Alhajj for his continuous advice, support, guidance, and enthusiasm. I have learnt a lot from him in the past years and this thesis would not be possible without him. It is really so hard to list everything he has done for me. His valuable efforts will always be remembered. Here I cannot pass without mentioning the great support and help received from Dr. Tansel Özyer from whom I learned the dedication, the motivation, a lot of computing techniques and the research style.

I also want to thank Dr. Jon Rokne and Dr. Rafaat El-Hacha, my committee members, for their time and effort in reviewing this work and their feedback.

I would like to highly appreciate the help, support, and collegiality of my friends and all the members in the research group of Dr. Alhajj. Thanks for helping me to get through difficult times smoothly and their moral support. Special thanks to Omar Addam, Abdullah Sarhan, and Anas Atieh.

I also would like to thank the Department of Computer Science at the University of Calgary for providing all the necessary resources for successfully completing my thesis.

Acknowledgment is extended to Alberta Innovates Technology Futures for providing the financial support for my research as well as for support the Microsoft Imagine Cup 2013 Team of which I was member.

Finally and most importantly, I am heartily grateful to my family for their love, encouragement, support, patience, prayers, and sacrifices throughout my life.

To my greatest father Nsr, and most kindness mother Nima.

To my dear brother Mahmood.

To my gorgeous sisters Ashjan, Haneen, Nisreen, and Marwa.

To my first and only love, my wife Wafaa.

To my country Palestine.

Table of Contents

List of Tables

List of Figures and Illustrations

List of Symbols, Abbreviations and Nomenclature

Symbol	Definition
SaS	Shop and Socialize
DM	Data Mining
KDD	Knowledge Discovery in Databases
SP	Shopping Process
SN	Social Network
SNM	Social Network Model
SNS	Social Network Service
ARM	Association Rules Mining
MBA	Market Basket Analysis
SPM	Shopping Process Model
DBMS	Database Management Systems
ECLAT	Equivalence Class Clustering And Bottom-up Lattice Traversal
FP-Growth	Frequent Pattern Growth
FP-Tree	Frequent-Pattern Tree
GCS	Geographic Coordinate System
GIS	Geographic Information Systems
WWW	World Wide Web
HTML	Hyper Text Markup Language
CSS	Cascade Style Sheet
ER	Entity Relationship

Chapter 1

Introduction

The rapid increase in the amount of data generated by the different resources and the huge amount of public data makes the processes for the extraction of hidden information manually very difficult. Over the past years, a large number of solutions has come up to handle this issue.

Data mining (DM) or knowledge discovery in databases (KDD) is one of the most effective solutions that allow automatizing the processes for the extraction of hidden information by identifying patterns from large quantities of data and repositories that help to get interesting knowledge, regularities, rules, patterns, and constraints.

Data mining combines knowledge and theories from different fields like databases, artificial intelligence, machine learning, statistics, optimization, and data visualization. It is a powerful technology that has been widely applied in various real-life applications like Health Care, Biology, Finance, Educational System, Supermarket Transactions Analysis, Store Layout and Promotions on the Items, Shopping System, and Web Search, among others. [34].

Supermarkets and shopping centers have mountains of administrative data about items, prices, customers, etc. With the huge amount of data, tendency for using the data mining highly increases, the reason behind that is to get benefit from the huge amount of data in this domain and improve the quality of the shopping systems' performance. For instance, a data mining framework for shopping centers was described in [10] and [22].

In addition to data mining, Social Network (SN) has become one of the highly demand fields in the last years. Social Network is a social structure which contains objects that form communities which may have some sort of relationships.

A social network service (SNS) is a platform, or website that focuses on setting up the social relations among people who share interests, activities, backgrounds, or real-life connections etc. Most social network services are web-based services and provide means for users to interact over the internet, such as e-mails and instant messages. Social network services allow users to share their activities, events, and interests within their individual networks.

On other hand, mobile devices and applications received a great attention in the last couple years. According to [8], smartphones outsold PCs for the first time ever in the fourth quarter of 2010, and 998 million smartphone shipments shipped in 2013, a 44% increase compared to 2012. Also, 86.4% of Canadian smartphone owners use their smartphones in-store [36]; the number will be much more in the upcoming years [14].

I argue that there is a great opportunity for customers shopping in a retail store to benefit from the rapid development in technology and computing techniques. Accordingly, this thesis develops a framework that provides customers with a mobile app which combines data mining techniques and social network analysis concepts to turn shopping into an attractive and enjoyable process. A customer will decide on the shopping list before entering the store; the proposed solution will get the shopping list to process properly and will help the customer in the shopping experience. The system will apply data mining techniques to derive possible combinations of items which could be purchased by the customer together based on his/her history captured by the system. Further, it is also possible to utilize purchase history of other members while deciding on the items to be brought to the attention of the customer. The mobile application also provides the opportunity to notify family members and/or friends of a customer about his/her presence in the store in case the customer preauthorized the server to do so. The notified bodies may be authorized to add more items to the initial list of the customer.

1.1 Problem Definition and Motivation

The term "Data mining" was introduced as a scientific field around 1990s in the database community, but data mining is the evolution of a field with a long history. Since 1990s, data mining become a widely used technology of extracting knowledge hidden from large volumes of raw data; this knowledge can enhance the quality of our life, health, and economy etc.

The data mining functions are of different types, depending on the intended data mining result; the data mining functions are classified as [12], [25]:

- **Exploratory Data Analysis**

  This type of data mining employs a variety of techniques, mostly graphical, in extracting the knowledge for what the user is searching for and analyzing it.

- **Descriptive Modeling**

  It is a mathematical process that describes the historical data, includes all the probability distributions of it, partitions it into groups and models that are describing the relationships between its variables in order to look at past performance and understand it.

- **Predictive Modeling**

  It is the process of using the descriptive model, mathematical science and machine learning to predict a value of one variable from known values of other variables and suggest decision options to take advantage of the predictions.

- **Discovering Patterns and Rules**

  It is primarily used to find patterns of common behavior from the data in order to focus on them or use them later in the prediction process.

- **Retrieving Similar Objects**

  The main objective of this task is to find patterns similar to the pattern of interest in the dataset.

In 1993, Rakesh Agrawal introduced the association rules mining (ARM); the first algorithm to extract the frequent itemsets and generate all the association rules between items in a large database of transactions. This algorithm was powerful for identifying the frequent patterns by using the historical data that will be discussed in the next chapter [1].

The Market could get benefit from data mining techniques to improve its quality; the main data mining technique that can be used is association rules mining (ARM). Market basket analysis (MBA) is the most famous example as an application for the association rules mining algorithm. MBA's applications have received a great deal of interest in the last two decades. It takes its name from the fact that a customer in a supermarket places all of his/her purchased items into the market basket or shopping cart. Customers buying habits are discovered by finding associations between different items that customers place in their shopping baskets.

Nowadays, the market basket analysis concept is not limited to the supermarket. It could be applied to any transactional database such as bank, webpages, cataloger and so on and so forth [7].

One of the most important human life activities is shopping. A store or retailer is a business that presents a collection of goods or services in order to trade or sell them to others for money or other goods. Shopping is an activity where a customer browses goods or services which are presented by a store with the intent to purchase some of them [48]. We have studied the shopping process model (SPM) and we explain the flow of this process in Figure 1.1 [49].

1. **Problem Recognition**

   When the consumer has a desire or need, he/she initiates an action and the shopping process begins, (e.g., when the consumer is hungry, the process of seeking out and purchasing food begins).

2. **Information Search**

   Different sources of information are utilized to gain understanding about various products/items and options available, as well as the nature of the consumer problem.

3. **Alternative Evaluation**

   The various alternatives presented to the consumer are evaluated based on comparison of the previous experience and the gathered information.

4. **Purchase Decision**

   An item is decided upon and a purchase occurs from a store.

5. **Post-purchase Behavior**

   The performance of the chosen process is considered against other possible excluded options and expectations. The item experience may play a part in shaping future decisions involving it, as well as helping in to increase or decrease cognitive dissonance about the purchase.

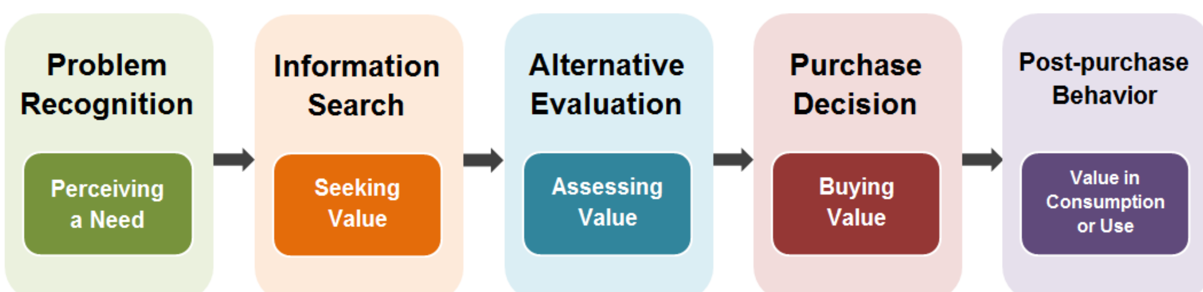

Figure 1.1: Shopping Process Model

The large and more complex the integrated shopping process is the more complicated and, hence, important the problem of finding the appropriate item and store becomes.

According to [30]; Americans spent on average 43.2 minutes/day shopping in 2011. A significant percentage of this time is spent finding grocery-items in a store. All of us remember times when we felt lost in a large grocery-store or had to do many trip to the store because we forgot to purchase an item or two.

We studied the shopping process model and consumer shopping behavior which lead to identify two distinct types of consumer shopping behavior, Shopping for a specific need or shopping for specific items:

Shopping based on a need is a common behavior when we want to eat, watch a movie, or go for shopping for joy. If the consumer does not know the appropriate item to be purchased, he/she may:

- Spend time thinking about his/her choices.
- Draw on his/her past experiences.
- Ask others' opinion.
- Go directly to any store browse for items to be purchased.

Shopping for specific items is a common behavior when we perform our regular grocery-shopping or similar activities. If the consumer does not know an appropriate store to be visited, he/she may:

- Spend some time looking for an appropriate store.
- Go to a store that may or may not satisfy all his/her needs.

This procedure, indeed, causes the following problems:

- Wasting valuable time in completing the shopping process.
- Consuming more fuel by driving around to find the appropriate store.
- Causing traffic congestion and unnecessary in-store crowd.
- Polluting the environment.

The reason behind studying and focusing on the shopping process model, as opposed to more traditional types of consumer research, is that it provides a more sophisticated understanding of consumer behavior. The model gives a new way of thinking about the strengths and weaknesses of the shopping process, what is working well and changes what is not. In this respect, it helps to save time and money and to keep the environment clean during the shopping process.

In this thesis, our motivation is to solve the problems mentioned above related to the shopping system and try to speed up the shopping process especially in selecting the items that a consumer wants to purchase and the store that he/she should visit to purchase the selected items. Since the consumer faces a lot of problems in finding the appropriate item and store, we have introduced a module that gets benefit from the social network analysis field by letting the consumer to notify his/her family and friends that he/she is going to do shopping, so they can ask him/her to purchase the needed items. We then, introduce a module that discovers the items that can be considered as frequent items from the consumer history and recommend him/her to purchase them. In other words, using the previous consumer transactions, we discover which items should be targeted by the consumer for future shopping. In addition, we introduce another module which recommends the best store to be visited to purchase the selected items based on the consumer's location and shopping list; this model is highly supported in the shopping system in terms of shopping traffic. In other words, the consumer will visit the closest store that almost has his/her needed items. Finally, we introduce a module that helps the consumer inside the store by providing him/her a navigational map showing to the locations of the items to minimize time and effort.

1.2 Proposed Solution

Figure 1.2 shows the framework of our proposed solution, it is called Shop and Socialize (SaS), it has four components:

1. Database Component.
2. Windows Communication Foundation Component.
3. Administrator Side Component.
4. Client Side Component.

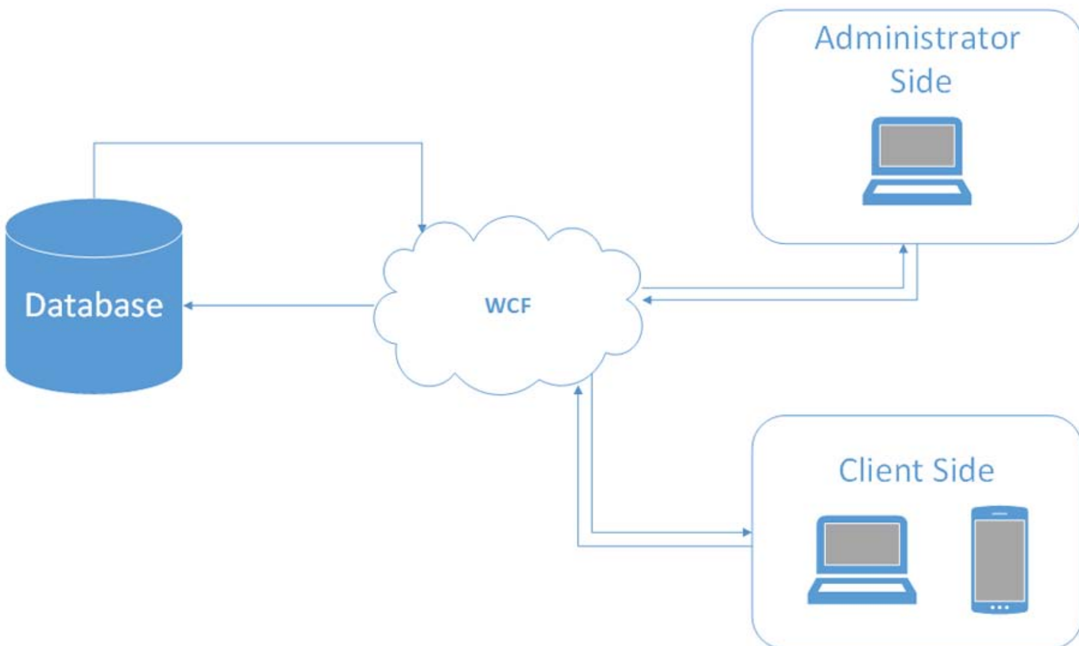

Figure 1.2: The Proposed Solution Framework (SaS)

Each one of these components has several functions and below is a description of each component and they will be described in more details in Chapter Three.

1. **Database Component**

   In this component, framework's data will be stored in appropriate formats inside tables to be used easily later on. Also, many stored procedures and functions were implemented to manage the data inside the database and the connections between this database and the other framework's components. In addition, some stored procedures and functions were implemented to help in building some modules in the proposed solution.

2. **Windows Communication Foundation Component**

   This is the most important component in this framework. In this component, we will be responsible for organizing the communication between the framework's other components. Also in this component, we will use a data mining technique (described in Chapter Two) to handle the major function for one of the modules in our proposed solution.

3. **Administrator Side Component**

   This component will be responsible for administrating our framework. The administrator will be able to manage stores, items, items inside each store, and system's administrators and clients.

4. **Client Side Component**

   In this component, system's client will be able to manage his/her account and benefit from using the proposed solution in order to make the shopping process easier.

1.3 Contributions

The proposed solution is a very useful approach to turn shopping into an attractive process that combines socializing, targeted shopping and saving. The main goal is to develop a robust framework that helps to speed up the shopping process and saves consumer's money. The contributions of our research as described in this thesis are summarized below:

1. We incorporate social network analysis concepts with the shopping process model to help the consumer to increase the benefit from the process.

2. We use data mining techniques to discover frequent items that were purchased by the consumer to be used in future shopping. This enhancement will increase the quality of the shopping system.

3. We use some SQL Server methods to help the consumer in finding the nearest stores or that are within couple of kilometers away to be visited by him/her.

4. We implemented an algorithm to help the consumer inside the target store to be visited by generating a map to be used for better planning to pick his/her items in short time.

1.4 Thesis Organization

To summarize, in this chapter we have described the problem tackled in this thesis and the motivation based on very important real world domain, i.e., the shopping domain. The proposed solution framework is presented and the main contributions of this work are listed.

The rest of this thesis is organized in four chapters:

In Chapter Two, we introduce the basic information about the commonly used shopping process model. Then, we describe basic information about social network analysis. After that, we introduce a detailed background about data mining and specifically the technique used in this research. Then, we talk about mobile applications, platforms, and market. Then, we introduce basic information about geometry concepts used in our work. Finally, we discuss the related work.

In Chapter Three, we present our proposed solution and methodology to solve the problems that we mentioned in the shopping system.

In Chapter Four, we discuss the experimental results of our framework's proposed solution described in Chapter Three, we use synthetic data, based on the nature of the real data available in stores, to build several scenarios in order to demonstrate the correctness and the performance of our proposed solution.

Chapter Five summarizes and concludes the thesis and discusses the limitations of our work. It also describes the future work and possible research directions.

Chapter 2

Background and Related Work

In this chapter, we present the necessary background information for the reader to understand the proposed solution. We first discuss the shopping process model. After that, we talk about the social network analysis concepts. Then, we focus, in details, on data mining techniques as relevant to our research. Then, we talk about mobile applications. Then, we talk about geographic coordinate system. At the end, we will cover the related work to our research.

2.1 Shopping Process Model

Canada has one of the biggest shopping systems worldwide. According to [31]; the Canadian sales volumes for all the retail categories in 2013 was 481.55 Billion Canadian Dollar. Also, based on the data from the Centre for the Study of Commercial Activity (CSCA), Edmonton (27.53), Halifax (26.33), and Calgary (24.84) have the highest retail real estate inventory square feet per capita to service their residences, followed by Toronto (21.93) and Saskatoon (20.10).

The shopping process model describes the process a consumer goes through in when purchasing a product. The shopping process model has passed through a lot of interpretation by scientists [21], [29]. In spite of having the models vary, there is a shared theme for the shopping process steps. These steps were first introduced by John Dewey [11]. The steps are:

1. Problem Recognition.
2. Information Search.
3. Alternative Evaluation.

4. Purchase Decision.

5. Post-purchase Behavior.

These steps form a good model to evaluate the consumer's shopping process. However, it is not necessary that a consumer gets through each step, nor it is necessary that he/she proceeds in any particular order. For instance, if a consumer has a need to buy bread, he/she might go straight to the purchase decision step, skipping the first three steps [24].

2.1.1 Problem Recognition

It is the first and most important step in the shopping process model. Without the recognition of the need, a product purchase cannot take place. The need can be triggered by internal stimuli such as hunger and thirst, or by external stimuli like marketing efforts [24].

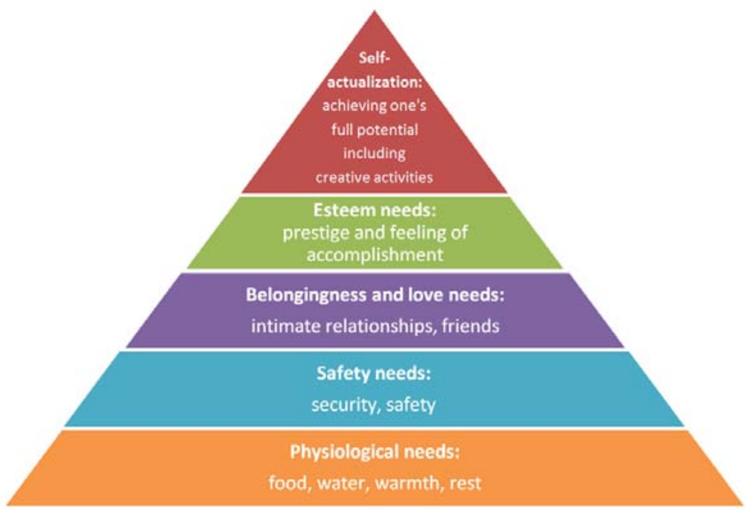

Figure 2.1: Maslow's Hierarchy

According to Maslow's hierarchy [26], shown in Figure 2.1, the human has different needs that could occur at any time, only when a human has fulfilled the needs at a certain level, he/she can move to the next level. The problem must be addressed through the available products.

2.1.2 Information Search

The consumer comes to this step after he/she has recognized the need in order to find out what is the optimal solution. In this step, the consumer looks for information about products such as price, options, features, and availability, he/she may use internal sources like scanning his/her memory to recall previous experiences. When previous experience is insufficient, the risk of making a wrong decision is high. Also, the consumer may use external sources like asking salespeople, friends, and family or accessing advertisements or websites.

2.1.3 Alternative Evaluation

In this step, the consumer evaluates each alternative based on varying product's attributes, and weather this product satisfies his/her needs. The consumer may care about objective attribute(s) such as the volume of the milk container, or subjective attribute(s) such as his/her prestige. Also, the decision in this step of the process may be affected by budget, personal likes and dislikes, and recommendations from family or friends.

2.1.4 Purchase Decision

In this step, the purchase will take place. The consumer will be mainly focusing on looking for a store to be visited in order to buy the needed product which depends on some sort of considerations like previous experience buying from this store, terms of sale, and return policy. Also he/she will be interested in scheduling a time to visit the targeted store which can be affected by the atmosphere, time pressure, and sale.

2.1.5 Post-purchase Behavior

After purchasing a product, the consumer takes some action related to the product, he/she compares the product with expectations and whether he/she is satisfied or dissatisfied. Satisfaction or dissatisfaction plays part in affecting the shopping process for a similar purchase from the same store in the future and spreading either positive or negative feedback about the product.

Not every product's purchase should go through all the five steps. For example, impulse buys might move directly from problem recognition to purchase decision such as I need that soft drink. Also, a consumer might decide to drop out of the shopping process at one of the steps.

2.2 Social Network Analysis

The notation of social network, coined by Barnes [5], is defined as a set of actors that are connected by one or more types of socially meaningful relationships, these nodes may represent people, organizations, genes, diseases, or other social entities [46], [47]. By realizing this concept, sociologists developed the social network model (SNM); this model is consisting of nodes (representing individual actors within the network) and edges (which represent relationships between the individuals, such as friendship, organizational position, etc.)

Based on SNM and graph theory, SNA was founded to understand, explain, and predict the communities' structure by description, visualization and statistical modeling. SNA has emerged as a key technique in modern sociology, anthropology, medicine, biology, communication studies, economics, geography, history, information science, organizational studies, political science, social psychology, development studies, and sociolinguistics etc. [48]. To build the SNM, it is necessary to define the actors (nodes) first and then the links (edges) are added based on the social relationship to be studied. For instance, in biology, actors could be genes and a directed link between gene A to gene B reflects the effectives of gene A to gene B; in another model the link may be undirected and reflects the number of diseases in which the two genes are affected.

One of the main objectives from using SNA is to understand the interaction between the actors in the SNM. This understanding is helping to enhance the organizational structures and the process flows.

Social network service is the first application in which people explicitly articulate their own social networks. Commercialized SNSs sites, like facebook and friendster etc. have incorporated a lot of social network theory concepts in their applications. Each user in a social network service is represented by a public profile which contains links to his/her selected members. These links build the network through which information is distributed, while each user can define his/her communities [3]. Researchers have concluded that the Internet complements traditional communications and enhances traditional social relationships [6], [16].

2.3 Data Mining

In traditional database management systems (DBMS), deductive information are retrieved in response to a query processing, queries are coded to retrieved existing information. Data mining or knowledge discovery in databases differs from traditional information retrieval from database. Data mining is defined as the extraction of interesting non-trivial, implicit, previously unknown, and potentially useful information from large volumes of raw data. Due to the perception of *"we are data rich but information poor"*, data mining has received a considerable attention for its key role in turning the huge amount of data into useful information and knowledge [18], [38].

2.3.1 Data Mining Life Cycle

Figure 2.2 shows the life cycle of a data mining process, it has six phases. The sequence of the phases is not strict. Moving between different phases is always required. It depends on the outcome of each phase. The main phases are described next [25], [20]:

1. **Business Understanding**

   This phase focuses on understanding the requirements, then converting this knowledge into a data mining problem definition and trying to achieve the objectives.

2. **Data Understanding**

   It starts with a sample data collection in order to get familiar with the data by identifying data quality problems, building first impression about the data, or detecting interesting subsets to form rules for hidden information.

3. **Data Preparation**

   This phase outputs the final dataset to be used by constructing the varieties of the activities from the initial dataset.

4. **Modeling**

   Different techniques are selected and applied in this phase, the parameters for the selected techniques are standardized to get optimal results.

5. **Evaluation**

   In this stage, a decision on using the data mining results might be reached, the model is evaluated and reviewed in order to construct the solution that achieves the requirements.

6. **Deployment**

   This phase is to increase the knowledge quality by organizing and presenting in a way that can be used easily.

Data mining has several tasks including association rules mining, clustering, and classification etc., and there are many different algorithms and techniques for each task. In this section, we explorer in details the data mining task relevant to our research which is association rules mining (ARM) and its corresponding techniques.

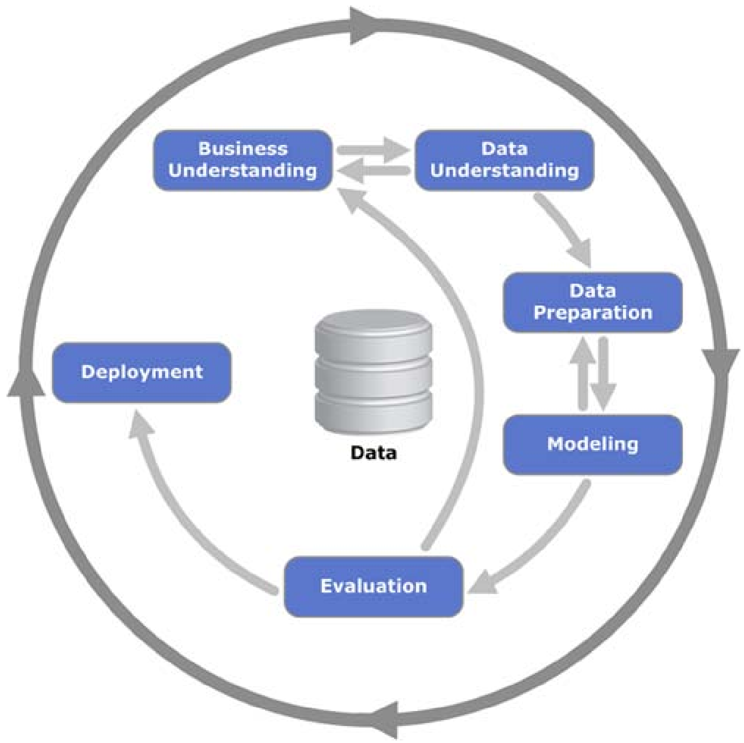

Figure 2.2: Data Mining Life Cycle
Adopted from [http://www.wikipedia.org]

2.3.2 Association Rules Mining (ARM)

Association rules mining is a technique that could be successfully applied to any dataset characterized by a many-to-many relationship between two sets of entities, e.g., transactions and items. The process detects relationships or associations between patterns that appear in a dataset frequently. For instance, a set of items such as bread and butter that appear together frequently in transactions dataset is called a frequent pattern/itemset. Finding frequent patterns in a dataset plays an integral part in mining associations, correlations, and other interesting relationships. Thus, it has a wide range of applications in many different fields like customer preferences and human resources management etc.

With the enormous amounts of data that are continuously being collected and stored, a lot of people are becoming interested in gaining some benefits from their repositories. Discovering the hidden relationships among any kind of transactional database might help in the decision making process such as shopping [18].

Market basket analysis explores customer's baying habits and discovers a set of associations in the different items that the customer places in his/her shopping basket and it is shared among a dataset. For instance, if a customer is buying milk, on the same trip to the store, is he/she going to buy bread? and what kind of bread?. Discovering which items are purchasing together can help to develop marketing strategies. Also, it is useful to [13]:

- Segment the customers
- Decide on layout of items in the market
- Decide on promotions plan
- Stock control
- Seasonal sales

ARM uses some form of statistical analysis. Rule's support and confidence are two statistical measures of rule interestingness. Since the dataset is large and domain's users are only concerned about the frequent patterns, thresholds of support and confidence are defined by domain's users in order to eliminate those rules which are out of interest. The two thresholds are called minimal support and minimal confidence, respectively.

Let $I = \{i_1,\ i_2,\ i_3,\ \ldots,\ i_n\}$ be a set of all available items. Let D be a set of all transactions where each transaction T is a set of items such that $T \subseteq I$. Let X, Y be sets of items such that $X, Y \subseteq I$. An association rule is an implication in the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$ and $X \cap Y = \phi$ [1].

Support

The support of an association rule $X \Rightarrow Y$ is defined as the number of transactions that contain $X \cup Y$ divided by the total number of transactions among the whole dataset. The count for $X \cup Y$ is increased by one every time $X \cup Y$ is encountered in different transaction T in D. Rules that have a support greater than or equal to s, which was specified by the domain's expert, is said to have minimum support and it will be an interesting association rule.

Support is calculated by the following formula:

$$Support\ (XY) = \frac{Support\ count\ of\ XY}{Total\ number\ of\ transactions\ in\ D}$$

Given the transactional dataset shown in Table 2.1:

- $Support\ (AB) = \frac{2}{5} = 40\%$
- $Support\ (BC) = \frac{3}{5} = 60\%$
- $Support\ (ABC) = \frac{1}{5} = 20\%$

We can say that, $Support\ (ABC) = 20\%$ means that 20% of the transactions in D contain purchasing of items $A, B, and\ C$ together. The retailer might not be interested in such kind of items that are not purchased very frequently, obviously association rules with high support is desired for more interestingness.

TID	**Items**
1	A, B, C
2	A, B, D
3	B, C
4	A, C
5	B, C, D

Table 2.1: Transactional Dataset

Confidence

The confidence for a rule $X \Rightarrow Y$ is defined as the number of transactions that contain $X \cup Y$ divided by the number of transactions that contain X among the whole dataset. An association rule that has a confidence greater than or equal to c which was specified by the domain's expert is said to have minimum confidence and it will be an interested association rule.

For the dataset shown in Table 2.1:

- $Confedence\ \{A \Rightarrow B\} = \frac{2}{3} = 66.67\%$
- $Confedence\ \{B \Rightarrow C\} = \frac{3}{4} = 75\%$
- $Confedence\ \{AB \Rightarrow C\} = \frac{1}{2} = 50\%$

We can say that, $Confedence\ \{B \Rightarrow C\} = 75\%$ means that 75% of the transactions that contain B also contain C together.

2.3.3 Association Rules Mining Algorithms

Many algorithms have been developed to find out association rules that satisfy the pre-defined minimum support and confidence from a given dataset. They mainly vary in how to generate the candidate itemsets and how to calculate the support for each candidate itemset. Some of these algorithms are very well known; because they were the first to define the concepts of the association rules and frequent itemsets. Other algorithms are variations which bring improvements mainly in terms of running time.

In this subsection, we will discuss in brief some of the most important association rules mining algorithms.

2.3.3.1 Agrawal, Imielinski, Swami Algorithm (AIS)

AIS was the first algorithm that introduced the association rules mining problem. It was introduced by Agrawal et al in [1]. AIS uses candidates itemsets generation in order to detect the frequent itemsets. It generates a set of all possible combination of items and then counts the support for each combination. The main drawback of this algorithm is generating too many candidate itemsets which finally turn out to be small; this process requires more space and work that ends to be useless. Also, AIS algorithm needs in general one extra pass in addition to a total number of passes equals to the size of the largest frequent itemset(s) discovered.

2.3.3.2 Apriori Algorithm

Apriori is the most important algorithm for association rules mining. It was first proposed by Agrawal and Srikant in [2]. The AIS algorithm is a naive approach, the number of possible combinations increase exponentially as the number of items increases making this approach impractical. Apriori is more efficient in the candidate itemsets generation, it uses the prior knowledge of the dataset to generate the candidates itemsets. The idea behind this algorithm is that for an itemset to be frequent, each of its subsets must be also frequent, so a k-itemset can be generated by extending a frequent (k-1)-itemset with a frequent 1-itemset. Figure 2.3 shows an example for Apriori algorithm.

Apriori Algorithm

Input:

D, a database of transactions

Min_sup, the minimum support count threshold.

Output:

L, frequent itemsets in D.

Method:

$L_1 = \{frequent\ items\}$

$for\ (k = 1; L_{k-1} \neq \phi; k++)\{$

$C_{k+1}\ =\ candidates\ generated\ from\ L_k;$

$for\ each\ transaction\ t\ \in D\ \{\ //scan\ D\ for\ count$

$increment\ the\ count\ of\ all\ candidates\ in\ C_{k+1}\ that\ are\ contained\ in\ t$

$L_{k+1} =\ candidates\ in\ C_{k+1}\ with\ min\_support$

$\}$

$\}$

$return\ \bigcup_k L_k$

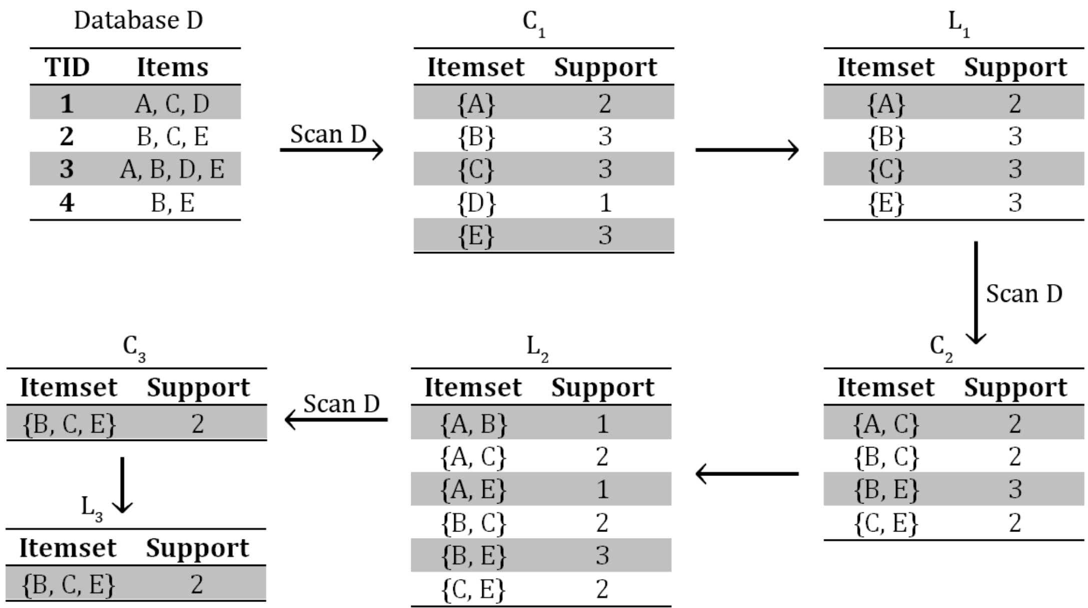

Figure 2.3: Apriori Example

Apriori has a drawback which is scanning over the dataset for each k-candidate set produced. Based on Apriori algorithm, many algorithms were introduced with some enhancements and modifications. Apriori-TID and Apriori-Hybrid [2], SON [40], and DHP [33].

2.3.3.3 Equivalence Class Clustering And Bottom-up Lattice Traversal (ECLAT)

ECLAT is the first association rules mining algorithm that uses a vertical dataset layout. ECLAT transforms a given transactional dataset in TID-itemset horizontal format, shown in Table 2.2, into item-TID_set vertical format, shown in Table 2.3. Then, it uses TID_set intersections to calculate the support of a candidate itemset, avoiding the generation of subsets that do not exist. Figure 2.4 shows an example to determine the support of k-itemset by intersecting TID_sets of two of its (k-1)-itemsets.

The drawback in ECALT is to have a large initial TID-sets table in case of having large dataset. Also, it finds the count for each individual item like in Apriori.

TID	**Items**
1	A, B, E
2	B, C, D
3	C, E
4	A, C, D
5	A, B, C, D
6	A, E
7	A, B
8	A, B, C
9	A, C, D
10	B

Table 2.2: Horizontal Transactional Dataset Layout

A	**B**	**C**	**D**	**E**
1	1	2	2	1
4	2	3	4	3
5	5	4	5	6
6	7	8	9	
7	8	9		
8	10			
9				

Table 2.3: Vertical Transactional Dataset Layout

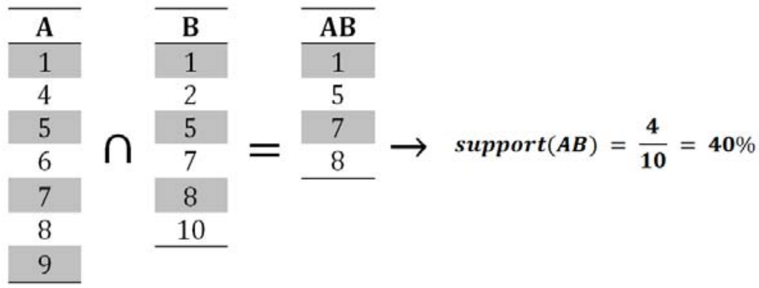

Figure 2.4: ECLAT Example

2.3.3.4 Frequent Pattern Growth (FP-Growth) Algorithm

FP-Growth algorithm was introduced by Han et al in [19]. It uses an extended prefix-tree structure and a (divide and conquer) strategy in order to extract the frequent patterns from a database. FP-Growth generates the frequent patterns with only two passes over the database and without any candidate generation.

This approach has two steps:

1. Build a compact tree data structure called the frequent-pattern tree (FP-Tree).
2. Extract frequent patterns directly from the FP-Tree.

Build FP-Tree

The first step is to build the FP-Tree which is used to store frequent patterns of a database. Having frequent patterns stored in a tree structure helps to avoid scanning the whole database in case of needing any information. FP-Tree is constructed by using 2 passes over the dataset:

In the first pass:

- The dataset is scanned to find the support for each individual item.
- Items that have support less than the minimum support are ignored.
- Frequent items are sorted in descending order based on their support.

The sorted list, shown in Figure 2.5, is used to build the FP-Tree, each node in the FP-Tree represents an item and has a counter.

Transactional Database D

TID	Items
1	A, B
2	B, C, D
3	A, C, D, E
4	A, D, E
5	A, B, C
6	A, B, C, D
7	A
8	A, B, C
9	A, B, D
10	B, C, E

The sorted list for D items

Item	Pointer
A	8
B	7
C	6
D	5
E	3

Figure 2.5: Transactional Database and Its Items Sorted List

In the second pass:

- Each transaction in the database is read and sorted based on the sorted list that we had in the first pass, and then it is mapped to its path in the FP-Tree, shown in Figure 2.6.
- Paths can overlap when more than one transaction share items, in other words, when some transactions have the same prefix. In this case, counters for the shared items are increased without duplicating the path. Thus an FP-tree is a compact structure.
- Pointers are maintained between the FP-Tree's nodes containing the same item, as shown in Figure 2.7.

The more paths that overlap we have, the higher the compression will be.

Frequent Patterns Extraction

Once an FP-tree has been constructed, it uses a recursive (divide and conquer) strategy to extract the frequent patterns.

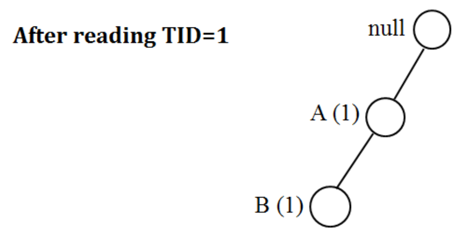

Figure 2.6: FP-Tree after Reading the First Transaction

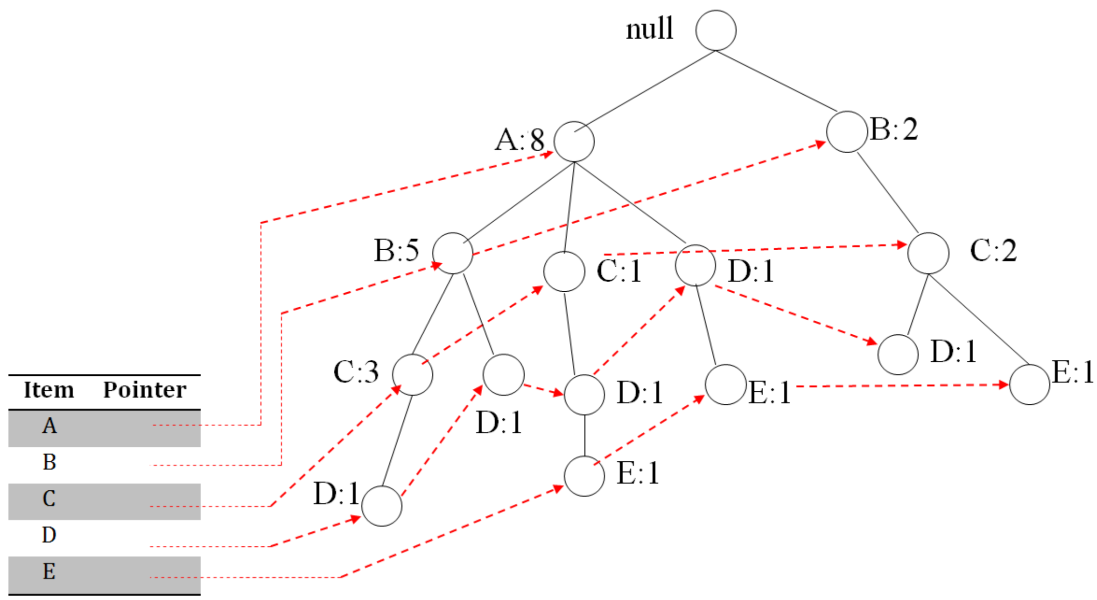

Figure 2.7: FP-Tree after Reading all the Transaction

Given Figure 2.7, FP-Growth starts from the leaves towards the root, it looks for frequent patterns ending with E first, followed by D, C, B, and finally A. Frequent patterns ending with E are driven by examining only the paths containing E, these paths can be accessed by using the pointers associated with E. Frequent patterns found in the given tree are shown in Table 2.4, they are ordered by their suffix and the order in which they are found.

FP-Growth has some limitations, e.g., the FP-Tree may not fit in the memory. Also, it is difficult to use in interactive systems. Users may change the threshold of support that may lead to repetition of the whole mining process. Another limitation is that the FP-Tree is not suitable for incremental mining; because the databases keep changing over time, this may lead to a repetition of the whole mining process as well.

Suffix	Frequent Patterns
E	(E), (D,E), (A,D,E), (C,E), (A,E)
D	(D), (C,D), (B,C,D), (A,C,D), (B,D), (A,B,D), (A,D)
C	(C), (B,C), (A,B,C), (A,C)
B	(B), (A,B)
A	(A)

Table 2.4: Frequent Patterns Based on Figure 2.7 Tree

2.4 Mobile Applications

In addition to a wallet and keys, mobile devices are becoming an essential commodity to be carried in our daily life. Mobile devices play an integral part in organizing time, communicating with others, running business, and surfing the Internet. Millions of mobile applications have been developed to handle personal needs. A mobile application, or mobile app, is software application designed to run on smartphones, tablets, and other mobile devices.

2.4.1 Mobile Application Development

Mobile apps can be classified into three categories: Mobile web applications, native mobile applications, and hybrid mobile applications.

2.4.1.1 Mobile Web Applications

A mobile web application is a website that has been developed by using HTML5, CSS and JavaScript to be run on mobiles. It accesses the World Wide Web (WWW) by using a mobile web browser. A mobile web application can work with all the mobile devices that have browser, regardless of the mobile's operating system or browser type and can be developed quickly and cheaply. Also, it is updated once the application's code on the server is updated.

The main drawback in this category is not to get benefit from the mobile's features like camera and accelerometer.

2.4.1.2 Native Mobile Applications

A native mobile application is a software that is developed for a specific mobile operating system such as Windows Phone, Android, and iOS. Developing native mobile applications needs knowledge of the native platform's programming language like C# for Windows Phone and Java for Android. One of the main advantages in this kind of mobile applications is to get benefit from the mobile features such as camera and accelerometer. Also, it works faster than a mobile web application and provides better control through the user interface.

Accessing this kind of applications goes through app stores, such as Windows store, Google play, and Apple's app store. Native mobile applications are not updated automatically. The user needs to access his apps store in order to update to new version.

2.4.1.3 Hybrid Mobile Applications

A hybrid mobile application is a software that can be developed by using HTML5, CSS and JavaScript; and then it is wrapped in a native mobile application container to get access to platform features. In other words, it gets benefit from the mobile features such as camera and accelerometer. The main advantage in this category is to have applications that maintain the design and usability across the different platforms without developing different applications. Also it does not require knowledge in the mobile's platform programming language.

Accessing hybrid mobile applications goes through apps stores and the user needs to access the apps store for any updates.

2.4.2 Mobile Applications Platforms and Market

Mobile application development is experiencing significant growth; it is quickly becoming an area that can no longer be ignored. Mobile application development domain has been completely reshaped since the beginning of the recent boom of smartphone and tablet computer market growth. More than 102 billion mobile apps were downloaded in 2013. It is estimated that the total number of mobile applications downloads will be more than double over the next few years, reaching 268 billion by 2017 [14]. Domain experts predict that this growth will continue over the next years, ensuring that the demand for mobile applications will escalate [32].

In response to the growing popularity, many mobile platforms are being developed to help in accessing the needs. According to [36] and many other resources, the leading mobile's platforms are: iOS, Android, Widows Phone and Blackberry. They are battling out for domination in every country around the world. Table **??** shows a comparison between iOS, Android, Widows Phone and Blackberry platforms [48].

Feature	**iOS**	**Android**	**Widows Phone**	**Blackberry**
Company	Apple	Google/ Open Handset Alliance	Microsoft	BlackBerry
Market share	0.134	0.813	0.041	0.01
Official application store	App Store	Google Play	Windows Phone Store	BlackBerry World
Current version	7.0.5	4.4.2	8	10.2.1
OS	Darwin	Linux	Windows CE7, NT, 8	QNX (Unix-like)
Programming Languages	C, C++, Objective-C	C, C++, Java	C#, VB.NET, Silverlight, C/C++, XMLA, DirectX	C, C++, ActionScript, Java

Table 2.5: Comparison between iOS, Android, Widows Phone and Blackberry Platforms

2.5 Geographic Coordinate System (GCS)

Geographic coordinate system is one of the most commonly used coordinate systems. It specifies precisely every location on the earth's surface based on specified horizontal (latitude) and vertical (longitude) lines, shown in Figure 2.8, which are measured in decimal degrees or degrees, minutes and seconds, each degree is divided into 60 minutes, each of which is divided into 60 seconds [44].

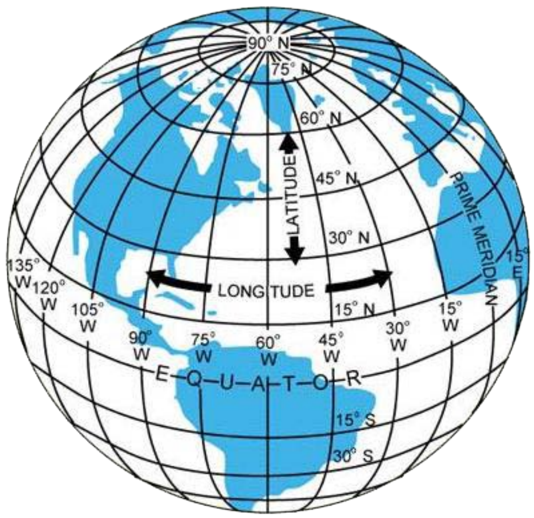

Figure 2.8: Latitude and Longitude Lines
Adopted from [http://www.dauntless-soft.com]

Latitude's lines run from east to west as circles parallel to the equator line and divide the earth's surface from north to south into 180 equal portions. The equator line is the reference latitude's line and each hemisphere is divided into 90 degrees. In the northern hemisphere, degrees of latitude are from 0 at the equator line to 90 at the North Pole. In the southern hemisphere, degrees of latitude are from 0 at the equator line to -90 at the South Pole. The distance between any two latitude's lines is the same which is 60 nautical miles.

Longitude's lines run from north to south parallel as circles perpendicular to the equator line and divide the earth's surface from east to west into 360 portions. The reference longitude's line runs from the North Pole to the South Pole through Greenwich in England and each hemisphere is divided into degrees. In the eastern hemisphere, degrees of longitude are from 0 at Greenwich line to 180 at the point parallels with Greenwich. In the western hemisphere, degrees of latitude are from 0 at Greenwich line to -180 at the point parallels with Greenwich. As we move towards the poles, the distance between any two longitude's lines becomes progressively less until, at the exact location of the pole, all 360 longitude's lines are represented by a single point.

Coordinates in decimal degrees are often used in the geographic information systems (GIS). In decimal degree system, they use the degree unit and decimal points as a percentage of a degree. The decimal points can be up to 4 points. When we use the 4 decimal points, the decimal degree system is accurate to within ± 11.12 m (36.5 feet). Often, the 4 decimal points are rounded up to 3 decimal points which results in an accuracy of ± 111.2 m (364.8 feet).

An example of the two measurement's systems, the location of ICT building at the University of Calgary in degree, minute and second system is $51^{\circ}\ 4'\ 47.99''$ N, $-114^{\circ}\ 7'\ 42.96''$ W. And in the decimal degree system is 51.0800 N, -114.1285 W.

Many formulas are proposed to calculate the distance between two points over the earths surface. Haversine formula is one of the most common formulas. The details of Haversine formula as follow [45]:

$$a = \ sin^2\left(\frac{\triangle\varphi}{2}\right) + \cos\left(\varphi_1\right)\ .cos\left(\varphi_2\right).sin^2\left(\frac{\triangle\lambda}{2}\right)$$

$$c = 2.\text{atan2}\left(\sqrt{a}, \sqrt{(1-a)}\right)$$

$$d = R.c$$

Where φ_1 and λ_1 are latitude and longitude for the first point, φ_2 and λ_2 are latitude and longitude for the second point, R is earth's radius, and angles in radians.

2.6 Related Work

Recent work in association rules mining focuses on proposing efficient algorithms that overcome the popular and computationally expensive task of generating the association rules such as reducing the number of passes over the database, reducing the size of the database to be scanned in each pass, and reducing the number of candidate patterns by using different techniques. The AIS algorithm was the first induction tool for discovering the association rules in large databases [1]. Some modifications have been proposed to improve its efficiency in [2] and then in [43]. Other algorithms have been proposed to extract the frequent patterns by using tree structure and (divide and conquer) strategy as FP-Growth algorithm. Also, a lot of algorithms have been proposed in order to induce more expressive rules such as by extracting generalized rules [42] or by mining over multiple abstraction levels [17]. In addition, algorithms for mining maximal frequent patterns, which are the itemsets that have no supersets that are frequent, have been proposed such as Max-Miner that was proposed in [39] to extract only the maximal frequent patterns by combining a bottom-up traversal with a top-down traversal in order to quickly find the maximal frequent patterns.

On the other hand, market basket analysis as an example for association rules mining has received a great deal of interest from the business perspective in recent years [7]. A lot of commercial market basket analysis applications and packages have recently been introduced to the research field and market [28], [9], [4].

Recommendation systems as business tools have received a great attention since the nineties from the last century. Shafer et al. say [41], as Pine states [35], that recommendation systems are one solution for businesses customization to serve multiple needs.

A retail prediction and recommendation system for a chain of retail stores was proposed in [15]. To improve sales forecasting, the author built a model that depends on four different types of datasets and used many data mining techniques such as K-means which is one of the commonly used clustering methods. BY providing the demographic information for the population around a new store, the model will be able to predict the expect sales for each item in the retailer. A worthy impact for the economic perspective was seen as an outcome for the proposed system.

A study has been done for recommendation systems in the electronic retail websites to understand the peculiar characteristics and requirements in that environment. A hybrid model has been proposed in [37] to generate dynamic recommendations and eliminate the drawbacks that exist. The proposed model takes advantages of the item category hierarchies and uses data mining techniques and collaborative filtering to satisfy its objective.

A personalized recommendation system based on web mining, association rules mining, and decision tree induction has been introduced in [23]. The authors applied the proposed model to an internet shopping mall for evaluation and got a high quality recommendation results.

In [27], the authors discussed the impact of using mobiles in shopping from the consumer and the store perspectives. For the consumer, integrating the mobile into the shopping process can make life easier by helping in getting what he/she wants more quickly. In addition to that, it helps to make life more meaningful by providing the information in the right time to make perfect choices. For store, using the mobile technology to provide information and services can increase sales, consumer satisfaction and loyalty, and the value of the physical products by promoting them through different mobile applications and services. Also, the authors discussed the future directions to the interaction between the consumer and the store through mobiles such as in-store navigation, self-scanning & self-checkout, payment, etc.

Chapter 3

The Proposed Solution and Methodology

Most stores around the world collect data about their customers. Using automated data mining approaches helps in gaining a lot of benefits for the store and the customer perspectives. Also, combining the social network concept with the shopping process model will be very useful in order to save the customer's time and money and the environment as well.

In this chapter, we introduce our proposed solution to solve the problems defined earlier in Chapter One. The framework developed is a web-based & mobile-based application. The system is mainly used by administrators or customers to improve the shopping process quality. To save time, money and keep the environment as clean as possible, the system recommends, based on data mining techniques, to the person who wants to go shopping a list of potential items that he/she may add to his/her cart based on the historical information and the items that he/she already added to his/her cart. Also, the system will notify his/her family and friends to be able to add items that they want to purchase to his/her cart. In addition, the system will give the completed and confirmed stores' list in order to purchase the need items. Finally, based on the items' locations inside the selected store, the system will generate a map for the customer to be followed inside the store in order to save some time.

3.1 User Requirements

There are two kinds of users for our proposed solution, administrators and customers.

Administrators will basically be responsible to manage stores, items, and items inside the stores. Other responsibilities of administrators are to manage system's users and their privileges. A user requires to have a computer connected to internet in order to access SaS.

For the customer, he/she mainly requires to have mobile device that has SaS mobile's app installed and an internet connection in order to access SaS.

3.2 System Architecture and Function

Our proposed solution framework, shown in Figure 1.2, contains four components:

1. Database Component.
2. Windows Communication Foundation (WCF) Component.
3. Administrator Side Component.
4. Client Side Component.

Each component has some functions to be done. The details of each component will be described in this Chapter.

3.2.1 Database Component

The system is connected to SQL SERVER relational database to store customer information including customer's profile and social network, favorite items, and all his/her historical purchases. The database also contains tables that have information about stores (including name, address, longitude, latitude, telephone, etc.) and items (including name, category, price, location, etc.) that exist inside these stores.

Generating the recommended items for the customer is one of our main contributions in the proposed solution. Customers purchasing transactions are stored in a table in the database. The data in this transactional table is a key to our system as we use this data to calculate the frequent items.

In addition, many stored procedures and functions are implemented for different purposes such as managing the data inside the database, retrieving the data from the database based on different input parameters, connecting the database with other components in the system, and finally helping in building some modules in the proposed solution.

The entity relationship (ER) data model or the conceptual data model is the detailed description of the real world application requirements as a set of entities (objects) that include attributes, relationships between entities, and constraints over the attributes. The ER diagram for our proposed solution is shown in Figure 3.1.

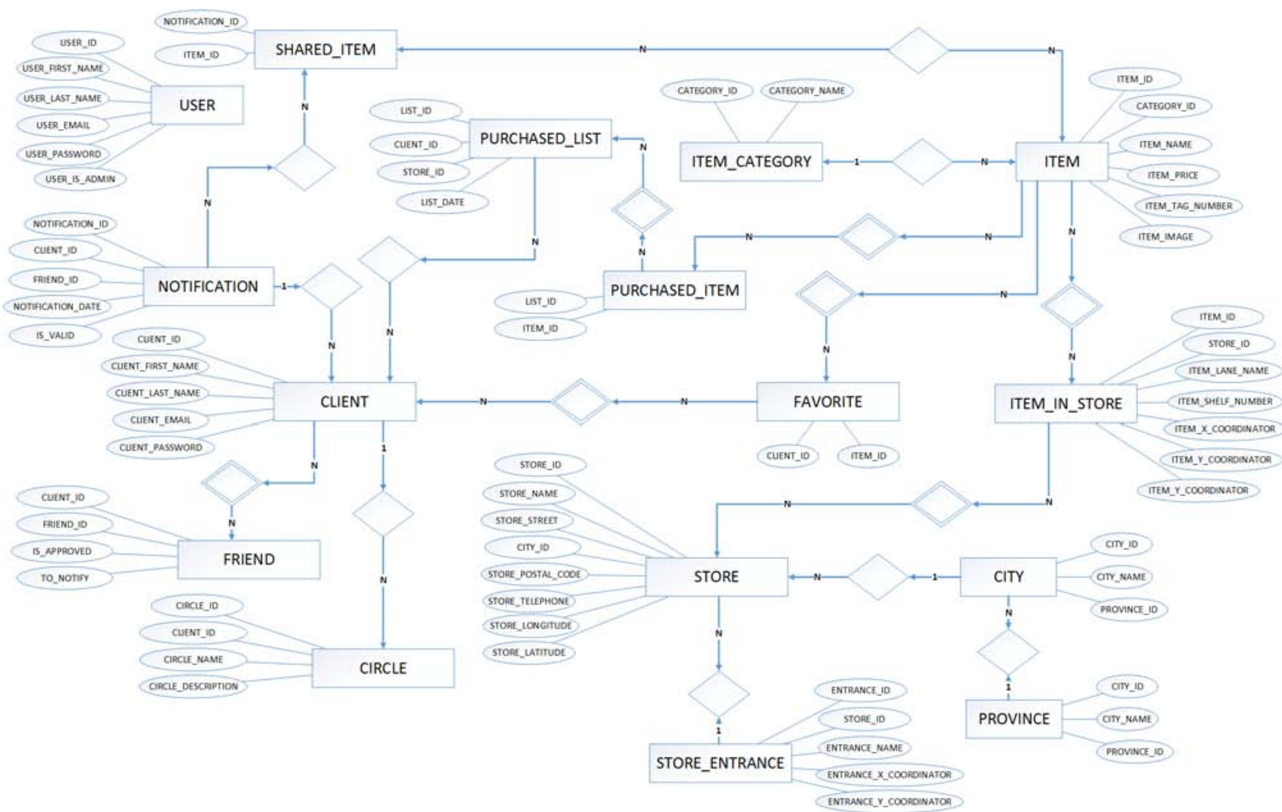

Figure 3.1: Entity Relationship Data Model Diagram

Figure 3.2 shows the relational database model diagram for our system. The relational database model represents the system in terms of tables (relations) and attributes. Each table contains a list of attributes. Each attribute has a name and domain.

A description of the data required by the system is presented in Table 3.1 3.1 as a data dictionary for the system's database.

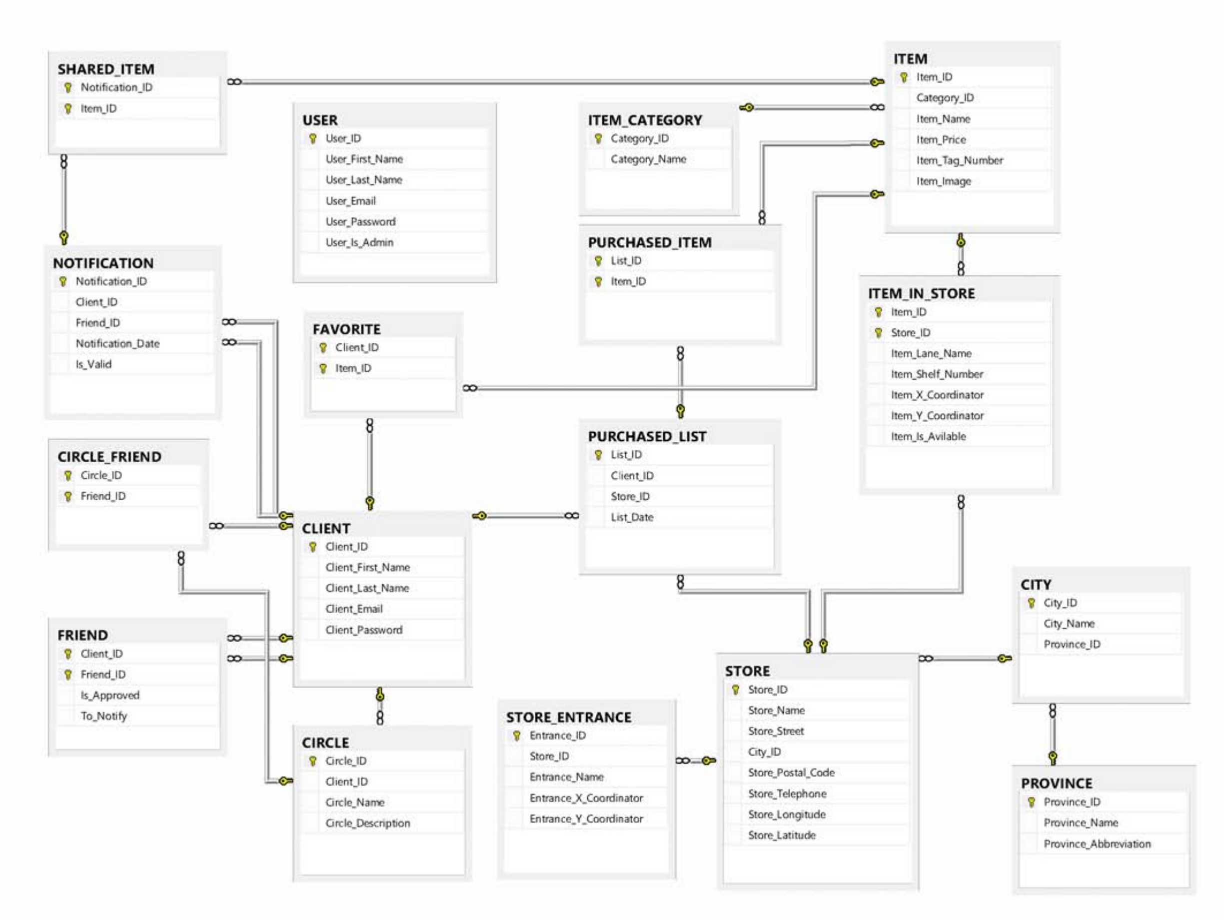

Figure 3.2: Relational Database Model Diagram

Entity	**Attribute**	**Data Type**	**Description**
STORE	STORE_ID	INTEGER	Key attribute to identify each store
	STORE_NAME	STRING	The name of the store
	STORE_STREET	STRING	The street of the store
	CITY_ID	INTEGER	The ID of the store's city
	STORE_POSTAL_CODE	STRING	The postal code of the store

Continued on next page

Table 3.1 – continued from previous page

Entity	Attribute	Data Type	Description
	STORE_TELEPHONE	STRING	The telephone of the store
	STORE_LONGITUDE	DECIMAL	The longitude of the store
	STORE_LATITUDE	DECIMAL	The latitude of the store
STORE_ENTRANCE	ENTRANCE_ID	INTEGER	Key attribute to identify each entrance
	STORE_ID	INTEGER	The ID of the entrance's store
	ENTRANCE_NAME	STRING	The name of the entrance
	ENTRANCE_X_ COORDINATOR	DECIMAL	The x coordinator of the entrance
	ENTRANCE_Y_ COORDINATOR	DECIMAL	The y coordinator of the entrance
ITEM_CATEGORY	CATEGORY_ID	INTEGER	Key attribute to identify each category for group of items
	CATEGORY_NAME	STRING	The name of the category
ITEM	ITEM_ID	INTEGER	Key attribute to identify each item
	CATEGORY_ID	INTEGER	The ID of the item's category
	ITEM_NAME	STRING	The name of the item
	ITEM_PRICE	DECIMAL	The price of the item
			Continued on next page

Table 3.1 – continued from previous page

Entity	Attribute	Data Type	Description
	ITEM_TAG_NUMBER	INTEGER	The tag number of the item
	ITEM_IMAGE	STRING	The link of the item's image
ITEM_IN_STORE	ITEM_ID	INTEGER	The ID of the item inside the store
	STORE_ID	INTEGER	The ID of the store
	ITEM_LANE_NAME	STRING	The lane name of the item
	ITEM_SHELF_NUMBER	INTEGER	The shelf number of the item
	ITEM_X_ COORDINATOR	DECIMAL	The x coordinator of the item
	ITEM_Y_ COORDINATOR	DECIMAL	The y coordinator of the item
	ITEM_IS_AVILABLE	BOOLEAN	Is this item amiable in the store?
PURCHASED_LIST	LIST_ID	INTEGER	Key attribute to identify each list
	CLIENT_ID	INTEGER	The ID of the client who purchased the list
	STORE_ID	INTEGER	The ID of the store where the list was purchased

Continued on next page

Table 3.1 – continued from previous page

Entity	Attribute	Data Type	Description
	LIST_DATE	DATATIME	The date and the time when the list was purchased
SHARED_ITEM	NOTIFICATION_ID	INTEGER	The ID of the notification that was sent to client's friend
	ITEM_ID	INTEGER	The ID of the item that was shared
PURCHASED_ITEM	LIST_ID	INTEGER	The ID of the list that was purchased
	ITEM_ID	INTEGER	The ID of the item that was purchased
USER	USER_ID	INTEGER	Key attribute to identify each user
	USER_FIRST_NAME	STRING	The first name of the user
	USER_LAST_NAME	STRING	The last name of the user
	USER_EMAIL	STRING	The email of the user
	USER_PASSWORD	STRING	The password of the user
	USER_IS_ADMIN	BOOLEAN	The privileges of the user
CLIENT	CLIENT_ID	INTEGER	Key attribute to identify each client

Continued on next page

Table 3.1 – continued from previous page

Entity	Attribute	Data Type	Description
	CLIENT_FIRST_NAME	STRING	The first name of the client
	CLIENT_LAST_NAME	STRING	The last name of the client
	CLIENT_EMAIL	STRING	The email of the client
	CLIENT_PASSWORD	STRING	The password of the client
FRIEND	CLIENT_ID	INTEGER	The ID of the client
	FRIEND_ID	INTEGER	The ID of the client's friend
	IS_APPROVED	BOOLEAN	Is this friendship approved?
	TO_NOTIFY	BOOLEAN	Should the system notify this friend?
CIRCLE	CIRCLE_ID	INTEGER	Key attribute to identify each circle
	CLIENT_ID	INTEGER	The ID of the client who has this circle
	CIRCLE_NAME	STRING	The circle name
	CIRCLE_DESCRIPTION	STRING	The circle description
NOTIFICATION	NOTIFICATION_ID	INTEGER	Key attribute to identify each notification
	CLIENT_ID	INTEGER	The ID of the client's
	FRIEND_ID	INTEGER	The ID of the client's friend
	NOTIFICATION_DATE	DATATIME	The date of the notification

Continued on next page

Table 3.1 – continued from previous page

Entity	Attribute	Data Type	Description
	IS_VALID	BOOLEAN	Is this notification valid?
FAVORITE	CLIENT_ID	INTEGER	The ID of the client
	ITEM_ID	INTEGER	The ID of the item
PROVINCE	PROVINCE_ID	INTEGER	Key attribute to identify each province
	PROVINCE_NAME	STRING	The name of the province
	PROVINCE_ ABBREVIATION	STRING	The abbreviation of the province
CITY	CITY_ID	INTEGER	Key attribute to identify each city
	CITY_NAME	STRING	The name of the city
	PROVINCE_ID	INTEGER	The ID of the city's province

Table 3.1: Database's Data Dictionary

3.2.2 Windows Communication Foundation (WCF) Component

WCF is a framework designed using service-oriented architecture (SOA) principle to support remote computing. It is used for implementing and deploying service oriented applications where services are loosely coupled and have remote clients. Clients can access multiple services; services can be accessed by multiple clients. Services have a web services description language (WSDL) interface that any WCF client can use to access any service regardless of the service platform. Any WCF application has endpoints. An endpoint has an address and it is a client or a service. Each service presents its contract via one or more endpoints. WCF client can send a request to a service endpoint in order to get information from it.

There are many reasons behind using WCF as a part of our framework. Actually, the proposed solution is targeting mobile users and there are many different mobile platforms. Further, the platform independent feature of WCF means that WCF services can be accessed from different platforms.

In addition, due to the limited speed of mobiles processors, we used WCF to handle all the computations and the calculations within the framework. Therefore, any information that is required by any application within our framework will be obtained through WCF. Applications will call WCF services and WCF services will be responsible for doing the computations and the calculations as well as accessing the database in order to get the required information.

Other reason to use WCF is related to the security perspective. As it is known, each platform has its level of authentication and authorization. However, one of the main objectives in our framework is to keep the integrity between its components. Hence, we used WCF to administrate the connections between our framework's components and to be responsible for the authentication and the authorization processes.

The frequent items that will be presented to the customer in his/her mobile application as recommended items will be delivered by WCF. WCF service will receive a list of the targeted items form the client, hence it will retrieve the historical data from the framework' database and apply the association rules mining algorithm to generate the recommended items.

The WCF component mainly contains six classes which are Database_Manager, User_Manager Client_Manager, Item_Manager, Store_Manager, and Encryption. The class diagram that describes the structure of the WCF classes, attributes, functions and their inputs and outputs, and the relationships among these classes is shown in Figure 3.3.

The detailed description for the WCF classes is as follow:

Database_Manager Class This class is responsible for the connection between the

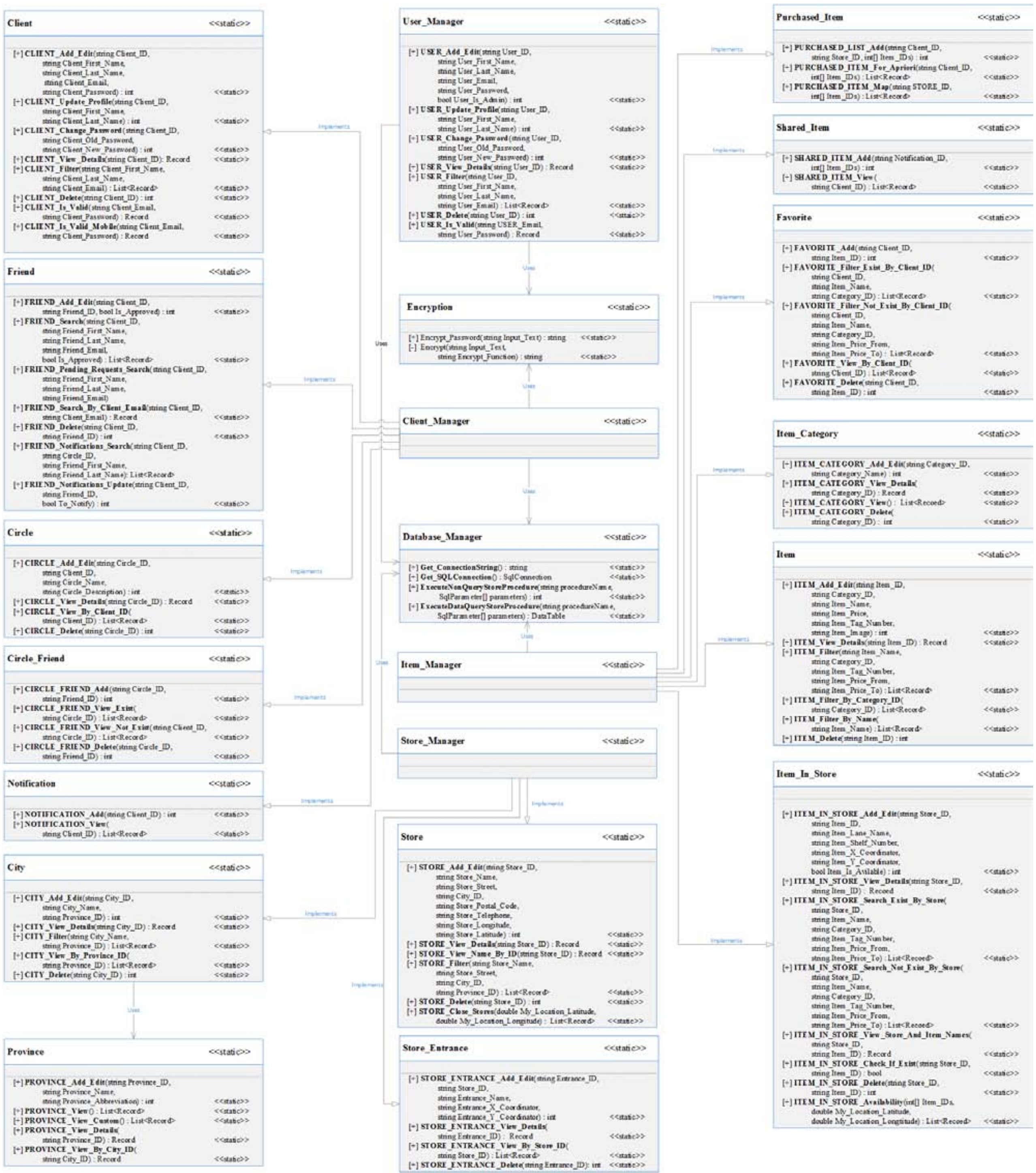

Figure 3.3: WCF Component's Class Diagram

WCF component and the framework's database. It is used to setup connections, send requests to call database's stored procedures based on the provided parameters, and receive responses from the framework's database. This class mainly includes four functions which are describe in Table 3.2.

State	**Type**	**Name**	**Description**
Public, Static	Function	Get_ConnectionString	Returns the information that require to setup connection between the WCF and the framework's database
Public, Static	Function	Get_SQLConnection	Setups a connection between the WCF and the framework's database
Public, Static	Function	ExecuteNonQueryStoreProcedure	Sends a request to the framework's database to call a stored procedure that executes catalog operations such as creating database objects, or changes the data in the database by executing UPDATE, INSERT, or DELETE statements
Public, Static	Function	ExecuteDataQueryStoreProcedure	Sends a request to the framework's database to call a stored procedure that executes SELECT statement based on the provided parameters to fill a Dataset

Table 3.2: Database_Manager Class's Functions Descriptions

User_Manager Class This class is responsible for handling the system administrator's needs such as adding new administrator, updating the administrator profile, and checking his/her authentications and authorizations. The detailed descriptions of this class's functions are described in Table 3.3.

State	Type	Name	Description
Public, Static	Function	USER_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new user or updates an existing user based on the provided parameters
Public, Static	Function	USER_Update_Profile	Sends a request to the framework's database to call a stored procedure that updates the name of an existing user based on the provided parameters
Public, Static	Function	USER_Change_Password	Sends a request to the framework's database to call a stored procedure that changes the password for a user based on the provided parameters
Public, Static	Function	USER_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves a user information and privileges based on the provided User ID
Public, Static	Function	USER_Filter	Sends a request to the framework's database to call a stored procedure that filters and retrieves users based on the provided parameters
Public, Static	Function	USER_Delete	Sends a request to the framework's database to call a stored procedure that deletes a user based on the provided User ID
Public, Static	Function	USER_Is_Valid	Sends a request to the framework's database to call a stored procedure that checks if the provided parameters are valid for an existing user

Table 3.3: User_Manager Class's Functions Descriptions

User_Manager Class

This class implements five abstract classes which are Client, Friend, Circle, Circle_Friend, and Notification. The Client_Manager class is responsible for handling the client's needs such as signing up, updating his/her profile, managing his/her friends and circles, and sending and receiving notifications. The detailed descriptions of this class's functions are described in Table 3.4.

State	Type	Name	Description
Public, Static	Function	CLIENT_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new client or updates an existing client based on the provided parameters
Public, Static	Function	CLIENT_Update_Profile	Sends a request to the framework's database to call a stored procedure that updates the name of an existing client based on the provided parameters
Public, Static	Function	CLIENT_Change_Password	Sends a request to the framework's database to call a stored procedure that changes the password for a client based on the provided parameters

Continued on next page

Table 3.4 – continued from previous page

State	Type	Name	Description
Public, Static	Function	CLIENT_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves a client information and privileges based on the provided Client ID
Public, Static	Function	CLIENT_Filter	Sends a request to the framework's database to call a stored procedure that filters and retrieves clients based on the provided parameters
Public, Static	Function	CLIENT_Delete	Sends a request to the framework's database to call a stored procedure that deletes a client based on the provided Client ID
Public, Static	Function	CLIENT_Is_Valid	Sends a request to the framework's database to call a stored procedure that checks if the provided parameters are valid for an existing client
Public, Static	Function	CLIENT_Is_Valid_Mobile	This function is for the mobile app. It sends a request to the framework's database to call a stored procedure that checks if the provided parameters are valid for an existing client

Continued on next page

Table 3.4 – continued from previous page

State	Type	Name	Description
Public, Static	Function	FRIEND_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new friend for an existing client based on the provided parameters
Public, Static	Function	FRIEND_Search	Sends a request to the framework's database to call a stored procedure that searches and retrieves client's friends information based on the provided parameters
Public, Static	Function	FRIEND_Pending_Requests_Search	Sends a request to the framework's database to call a stored procedure that searches and retrieves client's pending requests based on the provided Client ID
Public, Static	Function	FRIEND_Search_By_Client_Email	Sends a request to the framework's database to call a stored procedure that searches and retrieves client's friend information based on the provided Friend Email

Continued on next page

Table 3.4 – continued from previous page

State	Type	Name	Description
Public, Static	Function	FRIEND_Delete	Sends a request to the framework's database to call a stored procedure that deletes a client's friend based on the provided parameters
Public, Static	Function	FRIEND_Notifications_Search	Sends a request to the framework's database to call a stored procedure that searches and retrieves client's friends information who are notified by him/her based on the provided parameters
Public, Static	Function	FRIEND_Notifications_Update	Sends a request to the framework's database to call a stored procedure that update the notification's information for a client's friend based on the provided parameters
Public, Static	Function	CIRCLE_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new circle or updates an existing circle for a client based on the provided parameters

Continued on next page

Table 3.4 – continued from previous page

State	Type	Name	Description
Public, Static	Function	CIRCLE_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves a circle information based on the provided Circle ID
Public, Static	Function	CIRCLE_View_By_Client_ID	Sends a request to the framework's database to call a stored procedure that retrieves all the circles information for a client based on the provided Client ID
Public, Static	Function	CIRCLE_Delete	Sends a request to the framework's database to call a stored procedure that deletes a circle based on the provided Circle ID
Public, Static	Function	CIRCLE_FRIEND_Add	Sends a request to the framework's database to call a stored procedure that adds a client's circle based on the provided parameters
Public, Static	Function	CIRCLE_FRIEND_View_Exist	Sends a request to the framework's database to call a stored procedure that retrieves all the friends information who are exist in a client's circle based on the provided Circle ID

Continued on next page

Table 3.4 – continued from previous page

State	Type	Name	Description
Public, Static	Function	CIRCLE_FRIEND_View_Not_Exist	Sends a request to the framework's database to call a stored procedure that retrieves all the friends information who are not exist in a client's circle based on the provided Circle ID
Public, Static	Function	CIRCLE_FRIEND_Delete	Sends a request to the framework's database to call a stored procedure that deletes a client's friend from a circle based on the provided parameters
Public, Static	Function	NOTIFICATION_Add	Sends a request to the framework's database to call a stored procedure that adds notifications for the client's friends who are supposed to be notified based on the provided Client ID and the client profile
Public, Static	Function	NOTIFICATION_View	Sends a request to the framework's database to call a stored procedure that retrieves all the client's notifications that were sent to him/her based on the provided Client ID

Table 3.4: Client_Manager Class's Functions Descriptions

Store_Manager Class This class implements four abstract classes which are Store, Store_Entrance, City, and Province. The Store_Manager class is responsible for handling the store's needed information such as its name, address contact information, and entrances. The detailed descriptions of the Store_Manager class's functions are described in Table 3.5.

State	Type	Name	Description
Public, Static	Function	STORE_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new store or updates an existing store based on the provided parameters
Public, Static	Function	STORE_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves a store's information based on the provided Store ID
Public, Static	Function	STORE_View_Name_By_ID	Sends a request to the framework's database to call a stored procedure that retrieves the name of an existing store based on the provided Store ID
Public, Static	Function	STORE_Filter	Sends a request to the framework's database to call a stored procedure that filters and retrieves stores based on the provided parameters

Continued on next page

Table 3.5 – continued from previous page

State	Type	Name	Description
Public, Static	Function	STORE_Delete	Sends a request to the framework's database to call a stored procedure that deletes a store based on the provided Store ID
Public, Static	Function	STORE_Close_Stores	Sends a request to the framework's database to call a stored procedure that retrieves the closest stores based on an implemented algorithm and the provided Latitude and Longitude coordinators of the client's location
Public, Static	Function	STORE_ENTRANCE_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new entrance or updates an existing entrance for a store based on the provided parameters
Public, Static	Function	STORE_ENTRANCE_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves a store entrance's information based on the provided Entrance ID

Continued on next page

Table 3.5 – continued from previous page

State	Type	Name	Description
Public, Static	Function	STORE_ENTRANCE_View_By_Store_ID	Sends a request to the framework's database to call a stored procedure that retrieves a store entrances' information based on the provided Store ID
Public, Static	Function	STORE_ENTRANCE_Delete	Sends a request to the framework's database to call a stored procedure that deletes a store's entrance based on the provided Entrance ID
Public, Static	Function	CITY_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new city or updates an existing city based on the provided parameters
Public, Static	Function	CITY_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves city's information based on the provided City ID
Public, Static	Function	CITY_Filter	Sends a request to the framework's database to call a stored procedure that filters and retrieves cities based on the provided parameters

Continued on next page

Table 3.5 – continued from previous page

State	Type	Name	Description
Public, Static	Function	CITY_View_By_Province_ID	Sends a request to the framework's database to call a stored procedure that retrieves cities based on the provided Province ID
Public, Static	Function	CITY_Delete	Sends a request to the framework's database to call a stored procedure that deletes a city based on the provided City ID
Public, Static	Function	PROVINCE_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new province or updates an existing province based on the provided parameters
Public, Static	Function	PROVINCE_View	Sends a request to the framework's database to call a stored procedure that retrieves all the provinces information
Public, Static	Function	PROVINCE_View_Custom	Sends a request to the framework's database to call a stored procedure that retrieves some information about a province based on the provided Province ID

Continued on next page

Table 3.5 – continued from previous page

State	Type	Name	Description
Public, Static	Function	PROVINCE_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves a province's information based on the provided Province ID
Public, Static	Function	PROVINCE_View_By_City_ID	Sends a request to the framework's database to call a stored procedure that retrieves some information about a province based on the provided City ID
Public, Static	Function	PROVINCE_Delete	Sends a request to the framework's database to call a stored procedure that deletes a province based on the provided Province ID

Table 3.5: Store_Manager Class's Functions Descriptions

Item_Manager Class This class implements five abstract classes which are Item, Item_Category, Favorite, Purchased_Item, and Shared_Item. Item_Manager class is responsible for handling the item's needed information and the related information such as adding a new item, adding a new items' category, managing the favorite items, storing the client's purchasing transactions, and sharing the items with the others. The detailed descriptions of this class's functions are described in Table 3.6.

State	Type	Name	Description
Public, Static	Function	ITEM_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new item or updates an existing item based on the provided parameters
Public, Static	Function	ITEM_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves an item information based on the provided Item ID
Public, Static	Function	ITEM_Filter	Sends a request to the framework's database to call a stored procedure that filters and retrieves items based on the provided parameters
Public, Static	Function	ITEM_Filter_By_Category_ID	Sends a request to the framework's database to call a stored procedure that filters and retrieves items based on the provided Category ID
Public, Static	Function	ITEM_Filter_By_Name	Sends a request to the framework's database to call a stored procedure that filters and retrieves items based on the provided keywords

Continued on next page

Table 3.6 – continued from previous page

State	Type	Name	Description
Public, Static	Function	ITEM_Delete	Sends a request to the framework's database to call a stored procedure that deletes an item based on the provided Item ID
Public, Static	Function	ITEM_CATEGORY_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new items' category or updates an existing category based on the provided parameters
Public, Static	Function	ITEM_CATEGORY_View_ Details	Sends a request to the framework's database to call a stored procedure that retrieves items' category information based on the provided Category ID
Public, Static	Function	ITEM_CATEGORY_View	Sends a request to the framework's database to call a stored procedure that retrieves all items' categories
Public, Static	Function	ITEM_CATEGORY_Delete	Sends a request to the framework's database to call a stored procedure that deletes items' category based on the provided Category ID

Continued on next page

Table 3.6 – continued from previous page

State	Type	Name	Description
Public, Static	Function	ITEM_IN_STORE_Add_Edit	Sends a request to the framework's database to call a stored procedure that adds a new item to updates an existing item in a store based on the provided parameters
Public, Static	Function	ITEM_IN_STORE_View_Details	Sends a request to the framework's database to call a stored procedure that retrieves item's information in a store based on the provided parameters
Public, Static	Function	ITEM_IN_STORE_Search_Exist_By_Store	Sends a request to the framework's database to call a stored procedure that retrieves all the items that exist in a store based on the provided Store ID
Public, Static	Function	ITEM_IN_STORE_Search_Not_Exist_By_ Store	Sends a request to the framework's database to call a stored procedure that retrieves all the items that don't exist in a store based on the provided Store ID

Continued on next page

Table 3.6 – continued from previous page

State	Type	Name	Description
Public, Static	Function	ITEM_IN_STORE_View_Store_And_Item_Names	Sends a request to the framework's database to call a stored procedure that retrieves some information about an item and its store based on the provided parameters
Public, Static	Function	ITEM_IN_STORE_Check_If_Exist	Sends a request to the framework's database to call a stored procedure that checks if an items exists in a store based on the provided parameters
Public, Static	Function	ITEM_IN_STORE_Delete	Sends a request to the framework's database to call a stored procedure that deletes an item from a store based on the provided parameters
Public, Static	Function	ITEM_IN_STORE_Availability	Sends a request to the framework's database to call a stored procedure that checks the availability of the provided group of items in each store

Continued on next page

Table 3.6 – continued from previous page

State	Type	Name	Description
Public, Static	Function	FAVORITE_Add	Sends a request to the framework's database to call a stored procedure that adds a new items to the client favorite items list based on the provided parameters
Public, Static	Function	FAVORITE_Filter_Exist_By_Client_ID	Sends a request to the framework's database to call a stored procedure that filters and retrieves all items that exist in the client's favorite based on the provided parameters
Public, Static	Function	FAVORITE_Filter_Not_Exist_By_Client_ID	Sends a request to the framework's database to call a stored procedure that filters and retrieves all items that don't exist in the client's favorite based on the provided parameters
Public, Static	Function	FAVORITE_View_By_Client_ID	Sends a request to the framework's database to call a stored procedure that retrieves all items that exist in the client's favorite based on the provided Client ID

Continued on next page

Table 3.6 – continued from previous page

State	Type	Name	Description
Public, Static	Function	FAVORITE_Delete	Sends a request to the framework's database to call a stored procedure that deletes an item from the client's favorite based on the provided parameters
Public, Static	Function	PURCHASED_LIST_Add	Sends a request to the framework's database to call a stored procedure that adds a new transaction to the client's history which contains the purchased items based on the provided parameters
Public, Static	Function	PURCHASED_ITEM_For_Apriori	Runs the association rules maiming algorithm to extract the frequent items and proposed them to the client after sending a request to the framework's database to retrieve the needed transactions form the historical data

Continued on next page

Table 3.6 – continued from previous page

State	Type	Name	Description
Public, Static	Function	PURCHASED_ITEM_Map	Sends a request to the framework's database to call a stored procedure that runs an implemented algorithm based on the provided parameters to retrieve the map that the client should follow inside the store in order to save his/her time
Public, Static	Function	SHARED_ITEM_Add	Sends a request to the framework's database to call a stored procedure that adds a list of items that was sent by one client to another client's list to be purchased by him/her based on the provided parameters
Public, Static	Function	SHARED_ITEM_View	Sends a request to the framework's database to call a stored procedure that retrieves all the items that were shared by the client's family and friends to add them to his/her items' list based on the provided parameters

Table 3.6: Item_Manager Class's Functions Descriptions

Encryption Class This class was implemented in order to improve the security aspect in our proposed solution. It is mainly responsible for encrypting the sensitive information for the administrator and the client during their connections with the framework and before storing their information in the framework's database. Encrypting these information will help in improving our framework quality by securing the connections between the framework and its users and keeping their privacy intact.

3.2.3 Administrator Side Component

This component is responsible for administrating our framework. It is a website that can be used by the system's administrators in order to manage the framework based on their assigned privileges. The use case diagram that represents the administrator's interaction with the system is shown in Figure 3.4. A full privileges administrator will be able to access all the sub-systems that are found in this use case diagram.

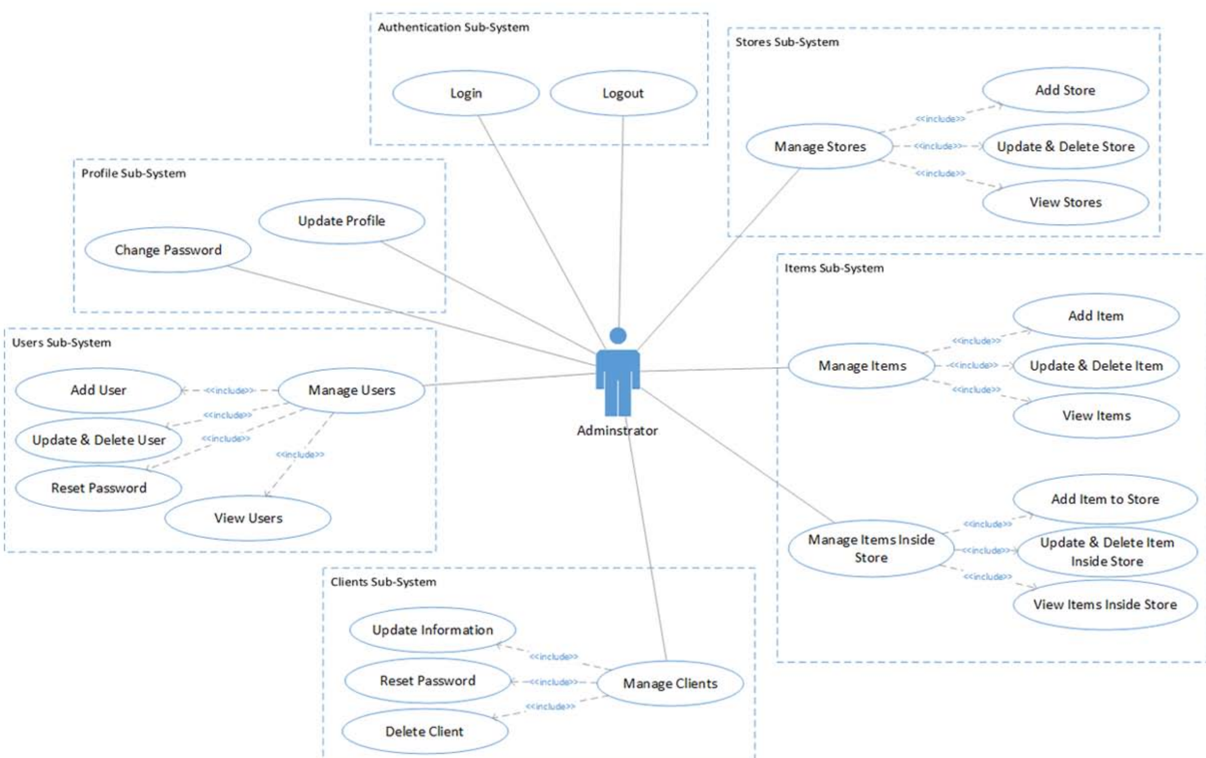

Figure 3.4: The Use Case Diagram for the Administrator Side Component

The main page for the administrator's website is shown in Figure 3.5 which contains all the functions that are required for the administrator to administrate the system.

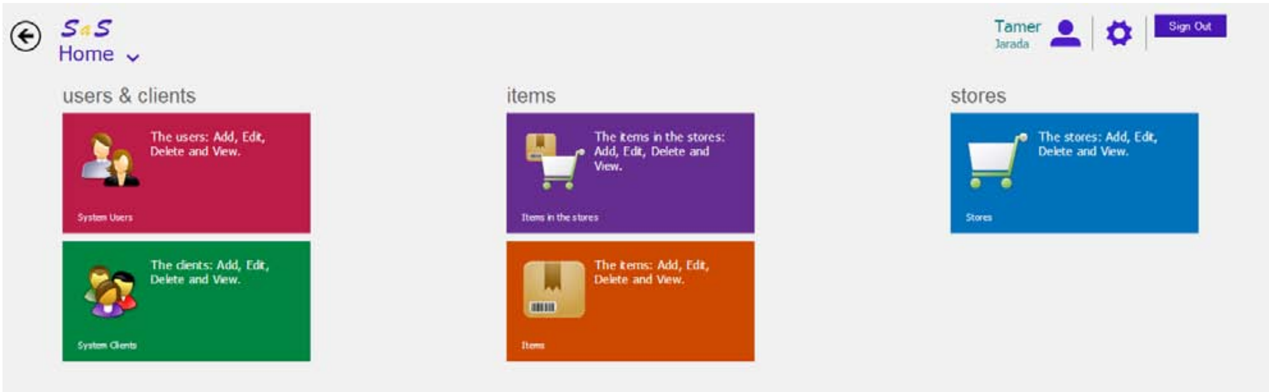

Figure 3.5: The Main Page for the Administrator Side Component

The users and clients tiles located to the left of the page will allow the administrator to manage the accounts for the users and clients in the system.

In the middle of the page, items' and items in store's tiles will allow the administrator to manage the items and the items inside each store.

The store tile located in the right of the page will allow the administrator to manage the stores that exist in the system.

As it is shown in Figure 3.4, the administrator has the ability to add a new user, update and delete an existing user or client, reset the password for an existing user or client, view existing users or clients.

Figure 3.6 shows the page that can be used to add or edit a user.

Figure 3.7 shows the page that can be used to view existing clients based on the filter parameters.

Also, the administrator will be able to add a new item, update or delete an existing item, view existing items, add a new item inside a store, update or delete an existing item inside a store, and view all existing items inside a store.

Figure 3.8 shows the page that views the existing items based on the filter parameters.

Figure 3.9 shows the page that can be used to add a new item inside a store.

Figure 3.6: The Page for Adding or Editing a User

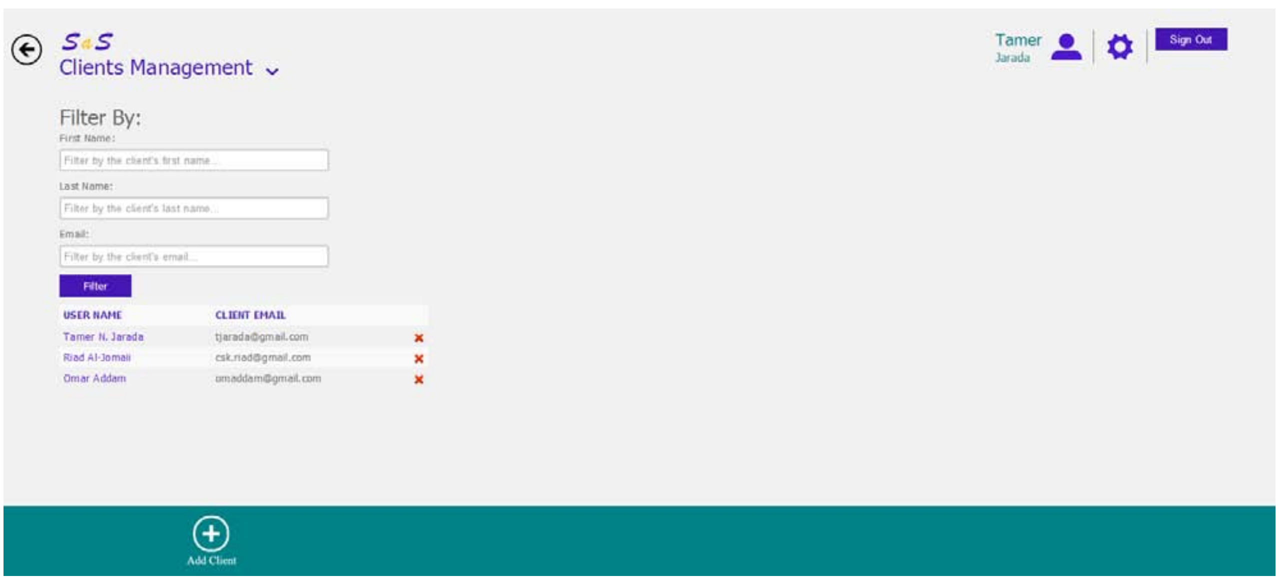

Figure 3.7: The Page for Viewing Existing Clients

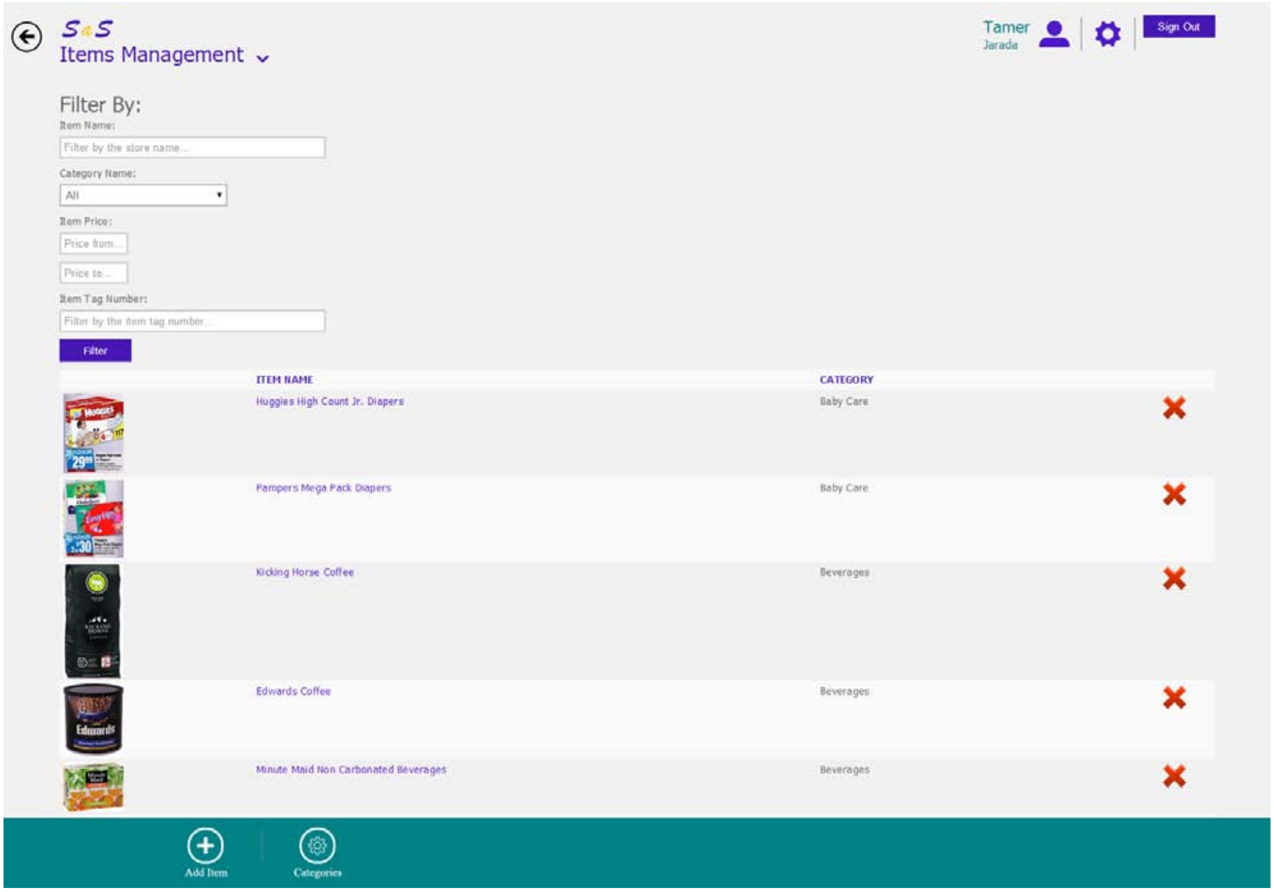

Figure 3.8: The Page for Viewing Existing Items

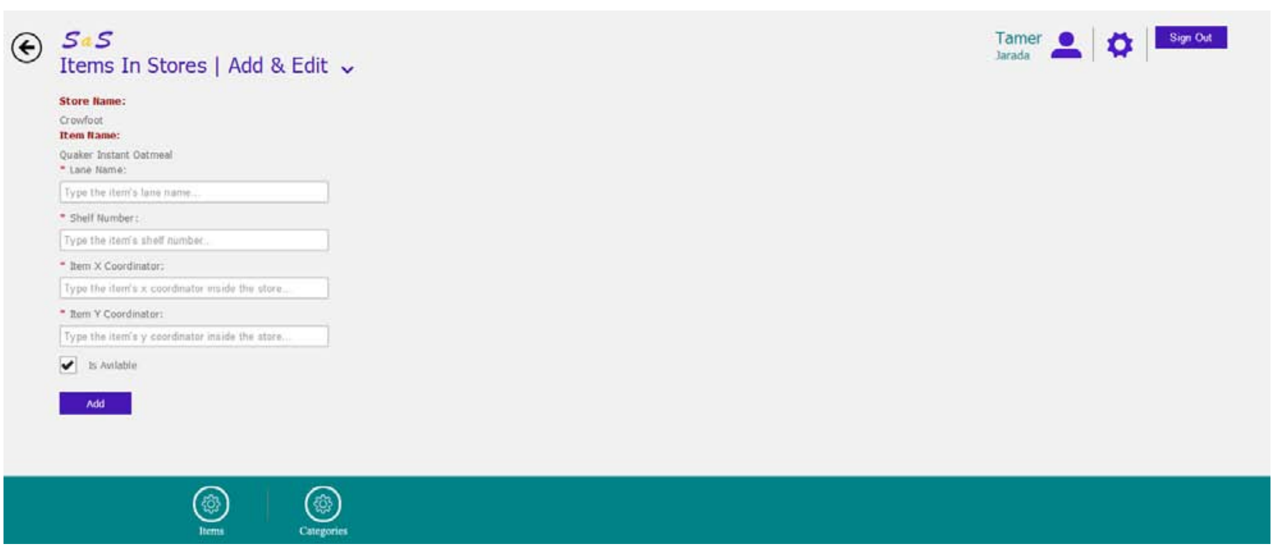

Figure 3.9: The Page for Adding a New Item inside a Store

In addition, the administrator will be able to add a new store, update or delete an existing store, view existing stores, etc.

Figure 3.10 shows the page that views the existing stores based on filtering parameters.

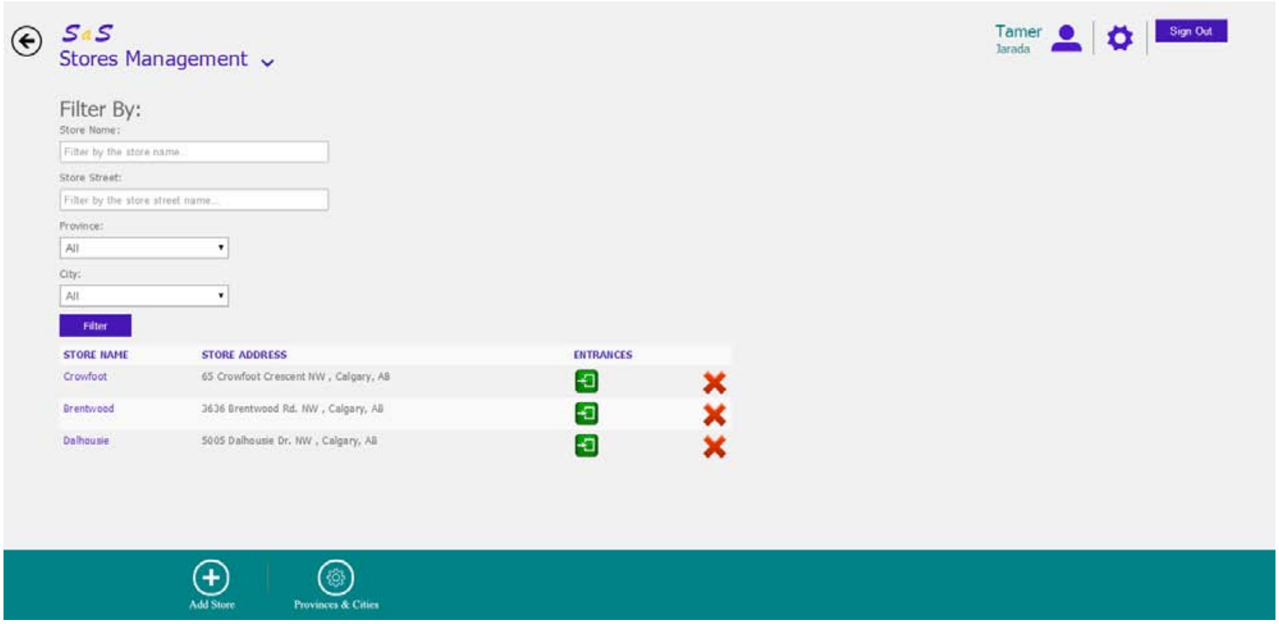

Figure 3.10: The Page for Viewing the Existing Stores

3.2.4 Client Side Component

This component is divided into two parts: website and mobile application; each part has its own functions. The detailed description for these two parts and their functions are described in the next two subsections.

3.2.4.1 Client's Website

It is a website that can be used by the system's clients in order to manage his/her favorite items list, friends and their groups, notifications center, and profiles.

The use case diagram that represents the client's interaction with the system is shown in Figure 3.11. Any client will be able to access all the sub-systems that are found in the use case diagram.

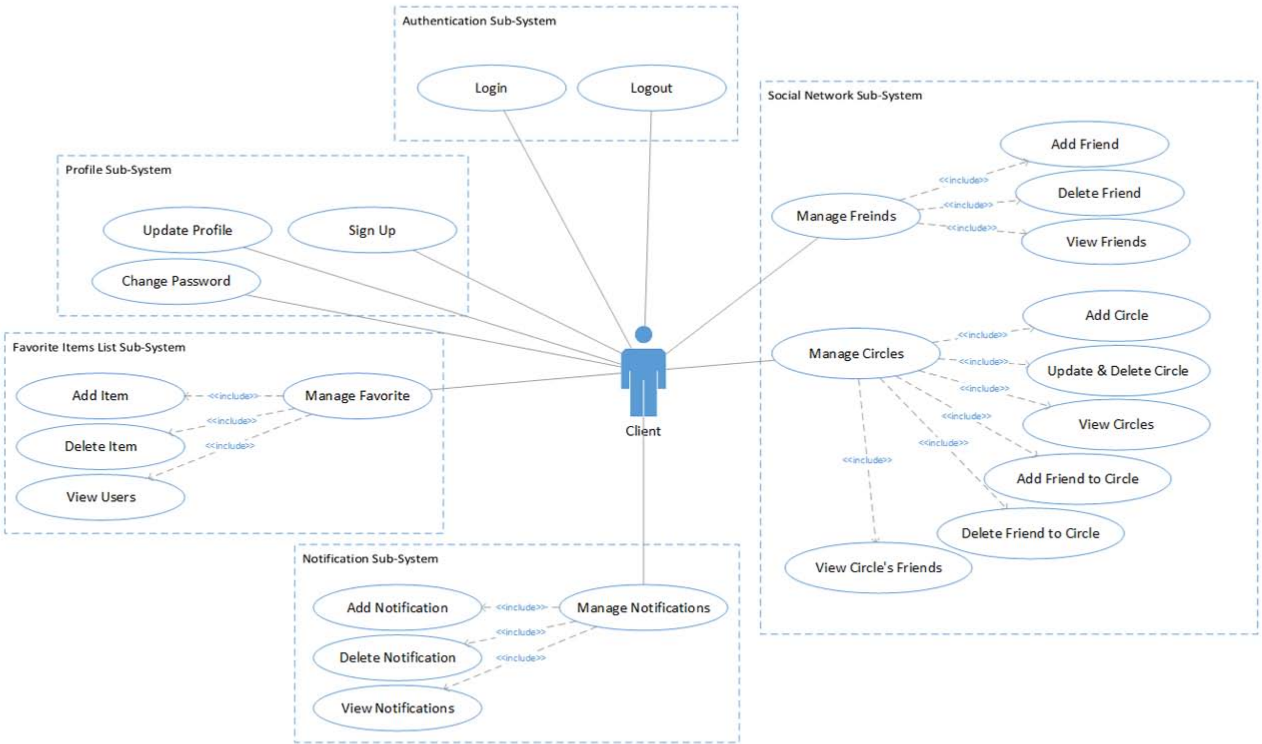

Figure 3.11: The Use Case Diagram for the Client's Website

The main page for the client's website is shown in Figure 3.12 which contains all the functions that are required for the client to manage his/her system's account.

The favorite items list tile located to the left of the page will allow the client to manage his/her favorite items that he/she used to purchase continuously.

In the middle of the page, client's friends & circles tile will allow the client to manage his/her social network.

The notifications center tile located in the right of the page will allow the client to manage the notifications for his/her family and friends who are in his/her social network.

As it is shown in Figure 3.11, the client has the ability to add a new item to his/her favorite list, delete an existing item, and view the existing items in the favorite list.

Figure 3.13 shows the page that can be used to view the client's favorite list.

Also, the client will be able to add a new friend, delete an existing friend, view existing friends, add a new circle, update and delete an existing circle, and view all existing circles.

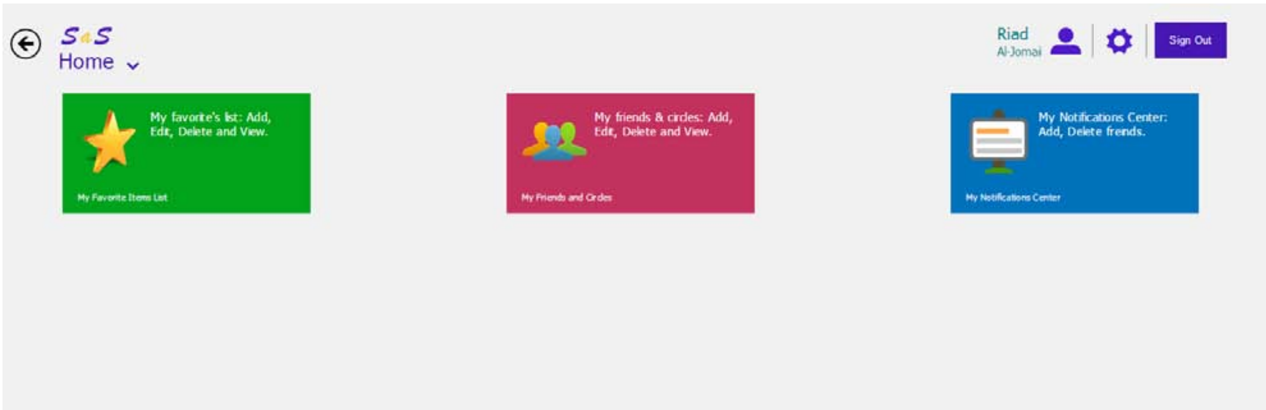

Figure 3.12: The Main Page for the Client's Website

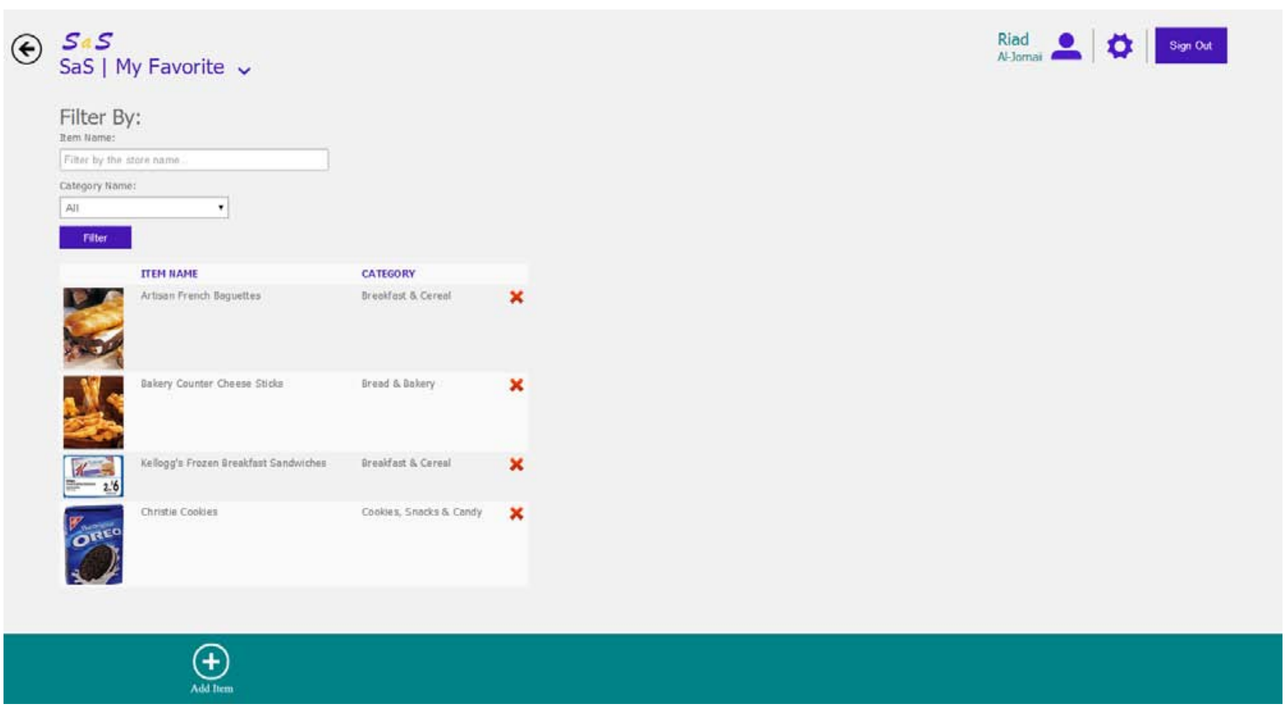

Figure 3.13: The Page for Viewing the Client's Favorite List

Figure 3.14 shows the page that views the existing friends based on the filter parameters.

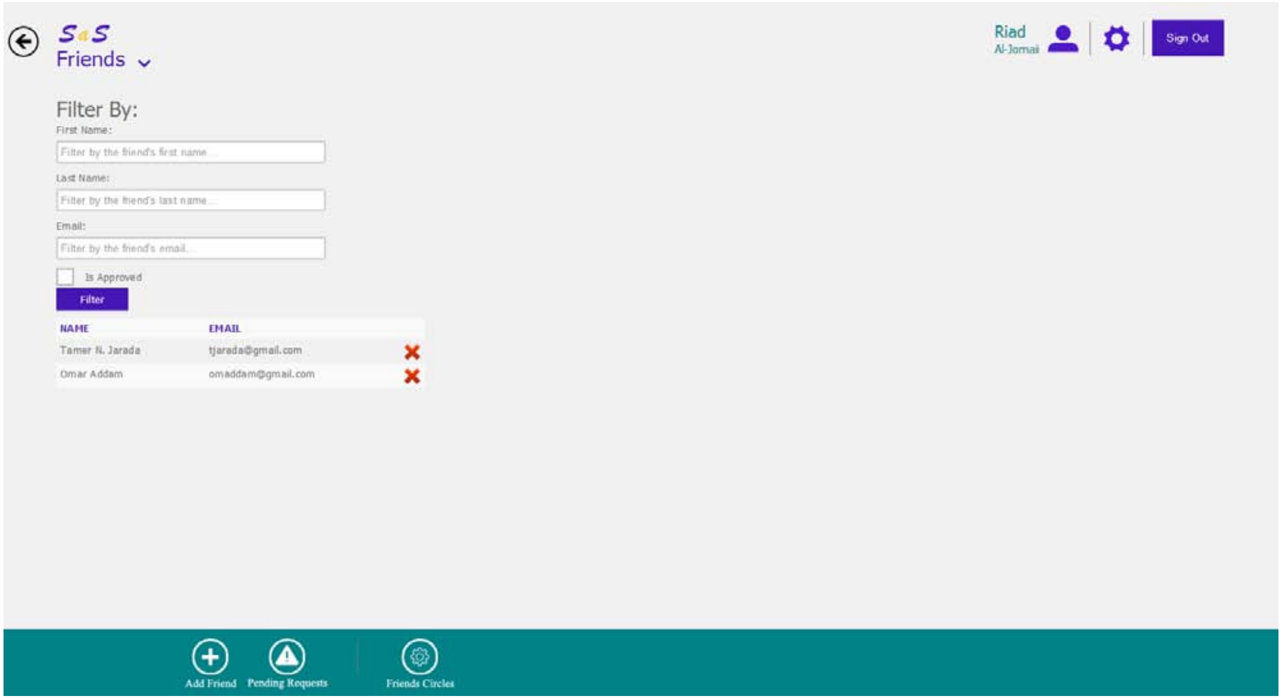

Figure 3.14: The Page for Viewing Existing Friends

In addition, the client will be able to manage the notifications center by allowing the system to notify his/her friends or denying them from his/her notifications.

Figure 3.15 shows the page that manages the notifications.

3.2.4.2 Client's Mobile Application

This is a mobile application that acts as a reminder and recommender and allows a client to socialize with his/her family and friends in order to produce a complete shopping list to speed up the shopping process, save money, and keep the environment cleaner.

SaS's mobile application will allow the client to build his/her own shopping list. Then, it will connect to the server to analyze the client's current shopping list in connection with his/her historical shopping data kept by the server to identify items which are not on his/her shopping list but he/she often purchased together with some other items on his/her list. The client may decide to add all or some of the recommended items to his/her list.

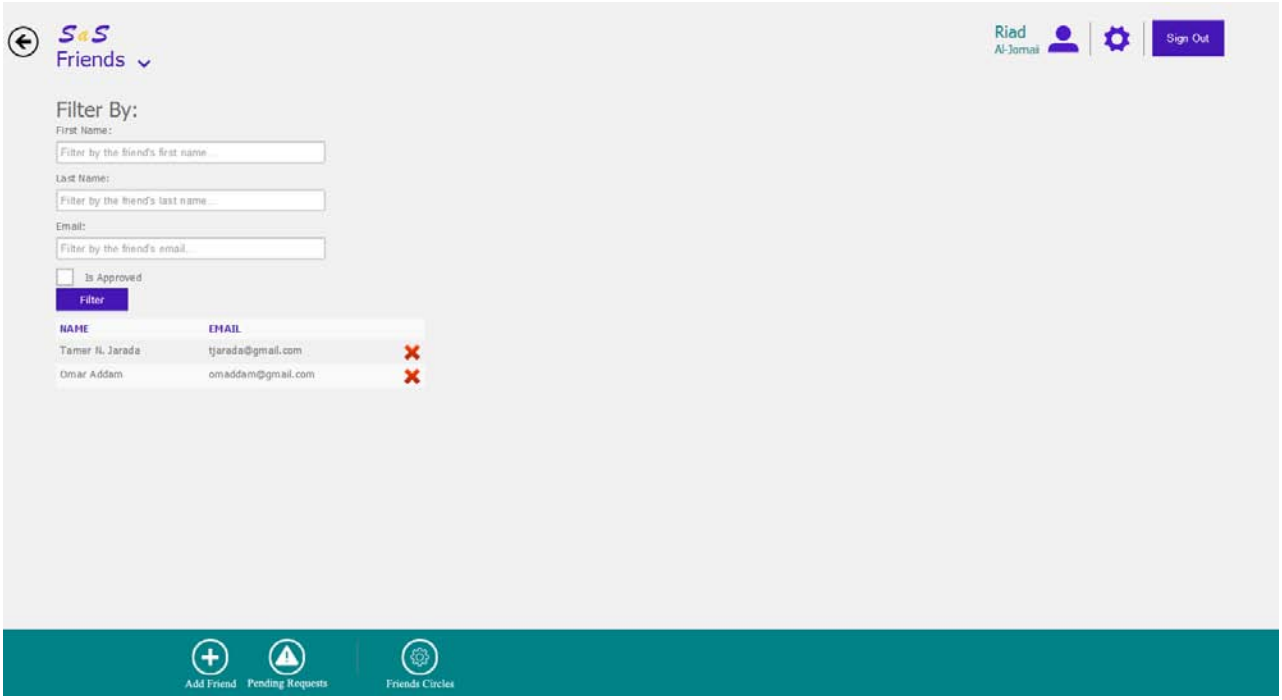

Figure 3.15: The Page for Managing the Notifications

Also, the application will notify the client's family members and friends about his/her plan to go shopping. This allows them to add some items to his/her shopping list. Also, the app will recommend to them items based on their shopping patterns.

After that, it will help the client in selecting the store that he/she should visit to purchase the selected items based on his/her location and shopping list and guiding him/her inside the store by providing a navigational map as to the locations of the items to minimize time and effort.

The use case diagram that represents the client's interaction with the mobile application is shown in Figure 3.16. Any client will be able to access all the sub-systems that are found in this use case diagram.

The main page for the client's mobile application is shown in Figure 3.17. As we see in this page, we have two parts. The upper part will lead the client to the search page to allow him/her to search for the needed items. The lower part contains five buttons which are:

- **Cart** that forwards the client to his/her cart.

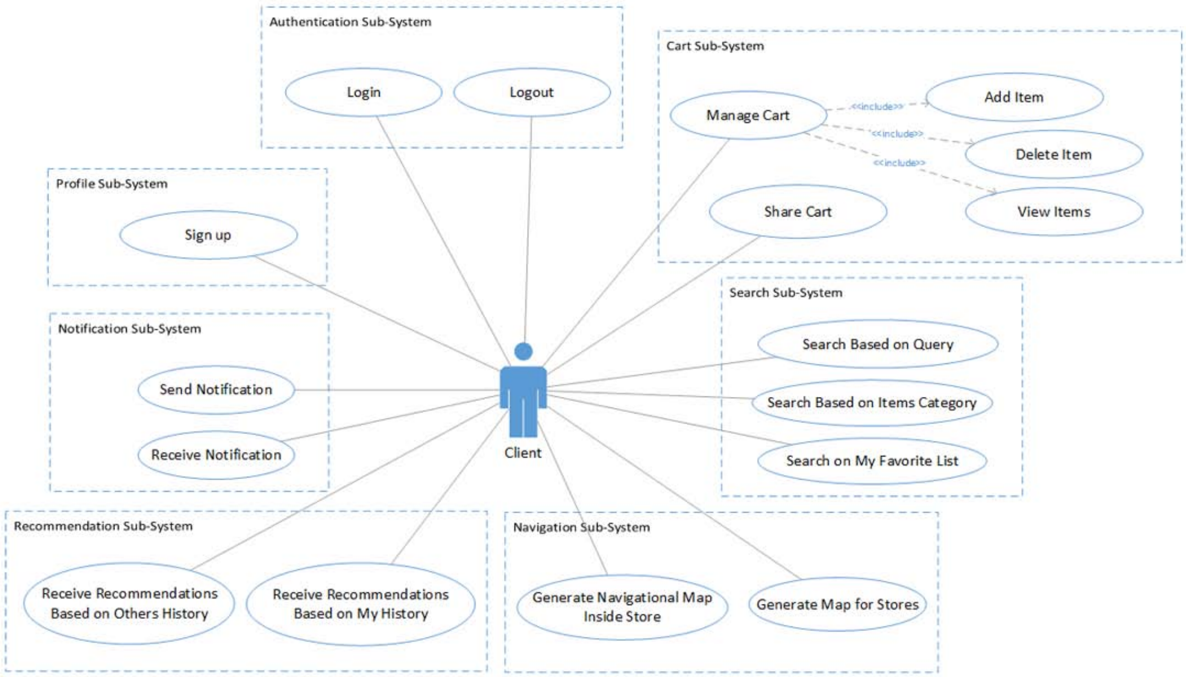

Figure 3.16: The Use Case Diagram for the Client's Mobile Application

- **Favorite** that leads the client's to his/her favorite items' list in order to add some of them to his/her cart.
- **Stores** that redirect the client to a map for the sores that are close to his/her location.
- **Notifications** that takes the client to a page to send notifications or view the received notifications.
- **Your Account** that leads the client to a page to access his account's settings.

As it is shown in Figure 3.18, the client will be able search for item by:

- **Query:** that forwards the client to his/her cart.
- **Category:** a client may select the category he/she wants and a list of its items will appear.

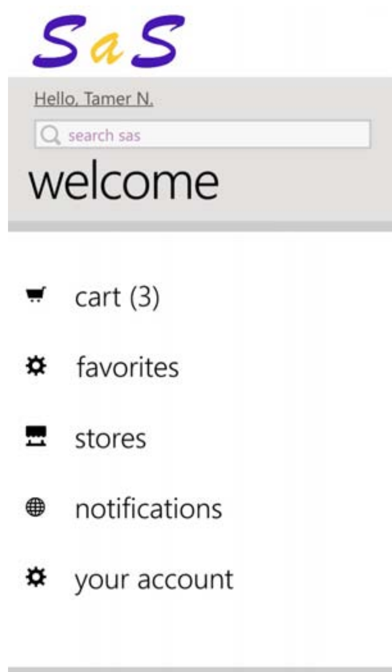

Figure 3.17: The Main Page for the Client's Mobile Application

The client may add items to his/her cart by taping on the item he/she wants to add.

Figure 3.19 shows the page that views the client's cart. In this page you two type of items:

- **Selected items:** these are the items that the client already added to his/her cart.
- **Suggested items:** these are items that were generated based on the client's selected items and history. These items are not added to the cart. So in order to add them the client has to tap on each item he/she needs to add to be added.

The client may remove any item he/she does not need any more by taping on it.

After that, the client can either share his/her cart with another client by clicking on the share button which will navigate him/her to the notification page, shown in Figure 3.20, to select the client with whom he/she wants to share his/her cart; or he/she can proceed to the next page that contains the stores' list by clicking on the checkout button.

The suggested stores page, shown on Figure 3.21, views the stores' list sorted based on the client's location and the number of non-available items in each store.

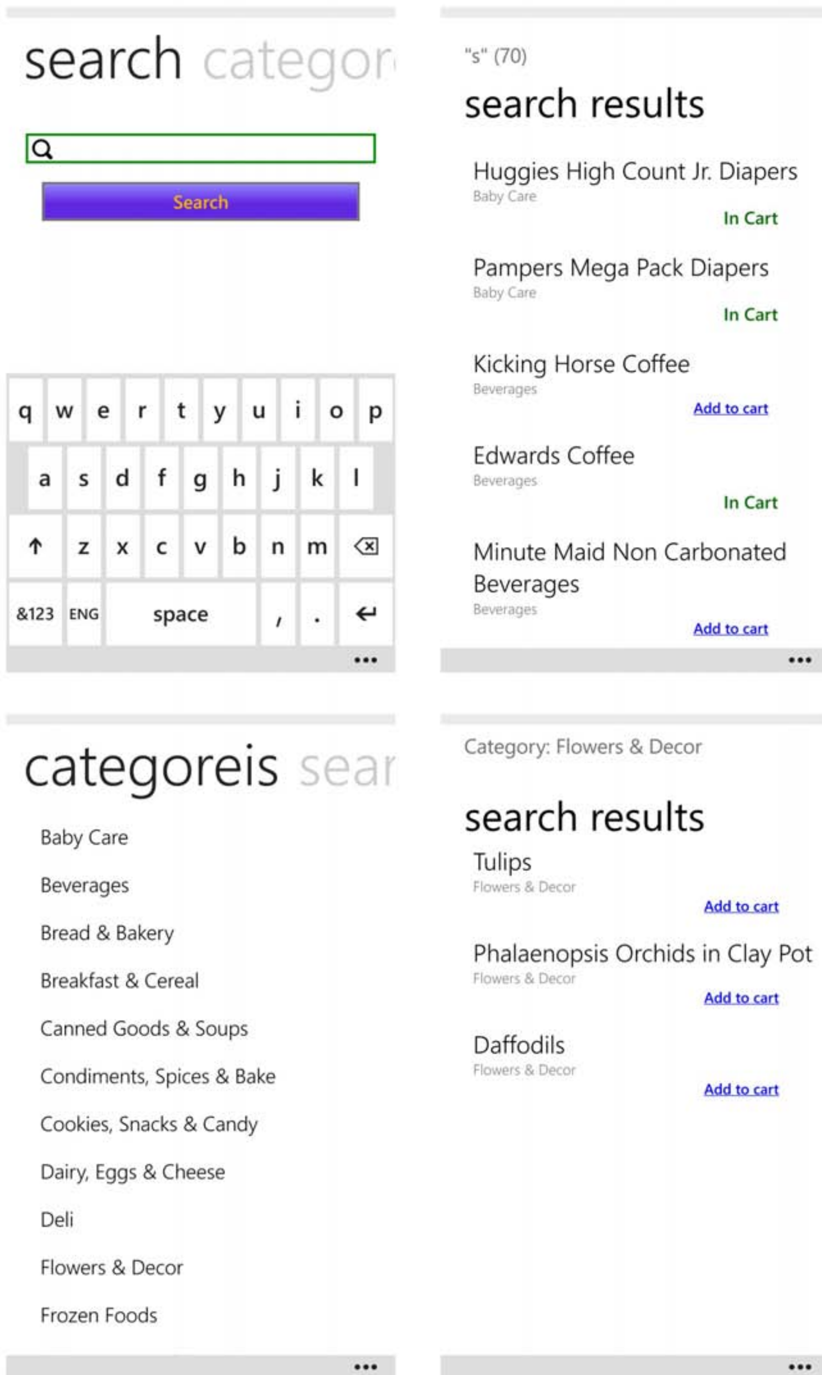

Figure 3.18: The Page that can be used to Search for Items

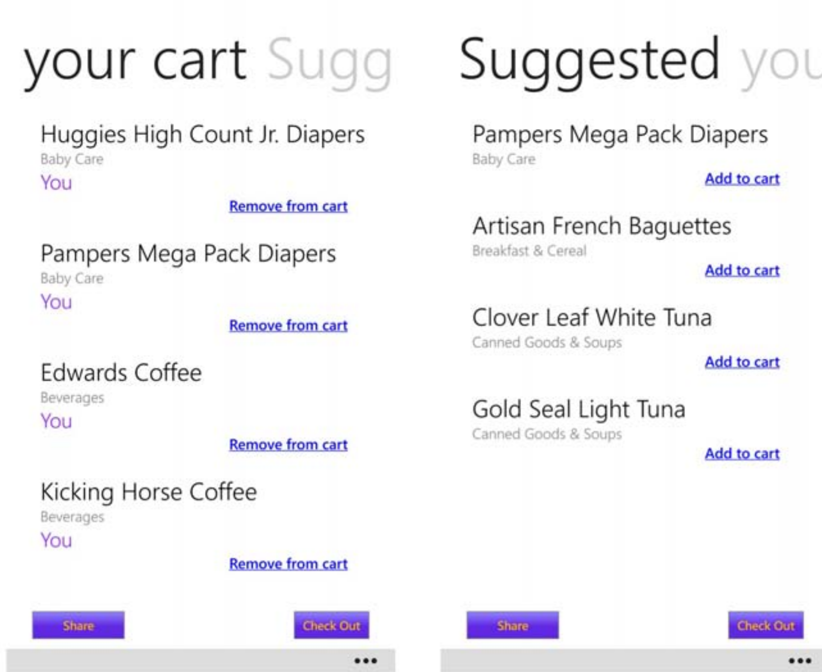

Figure 3.19: The Page that Views the Client's Cart

Figure 3.20: The Page that Views the Client's Notifications

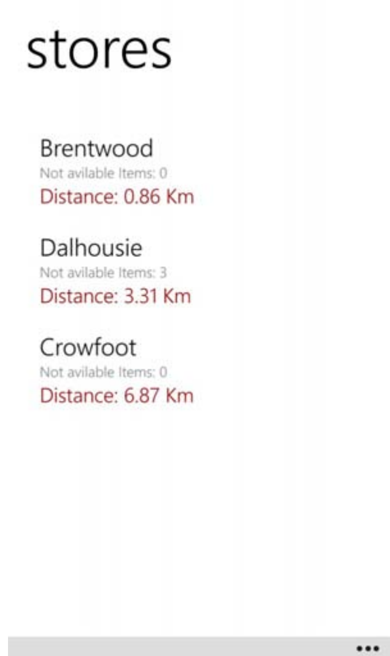

Figure 3.21: The Page that Views the Stores' List

After selecting the appropriate store, the client will be redirect to a page, shown in Figure 3.22, which contains information about the selected store and the navigation to it. Also, it will contain the navigational map inside the store to be used in picking the needed items.

In addition, the client will be able to access the favorite list page, shown in Figure 3.23, where he/she will be able to view his/her favorite items. The client can add any item from his/her favorite items to his/her cart by tapping on each item.

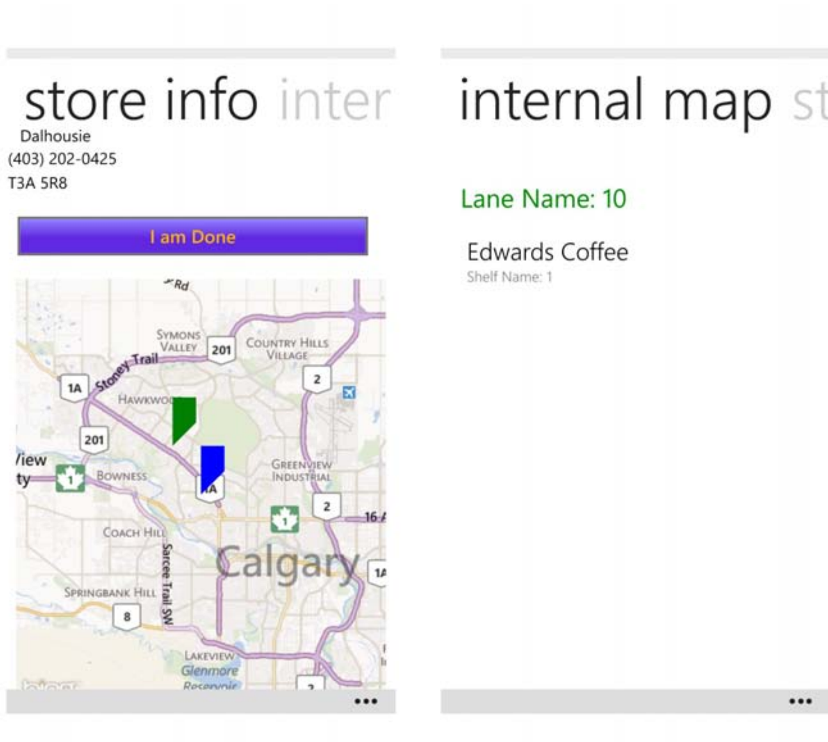

Figure 3.22: The Page that Views the Store's Maps and Information

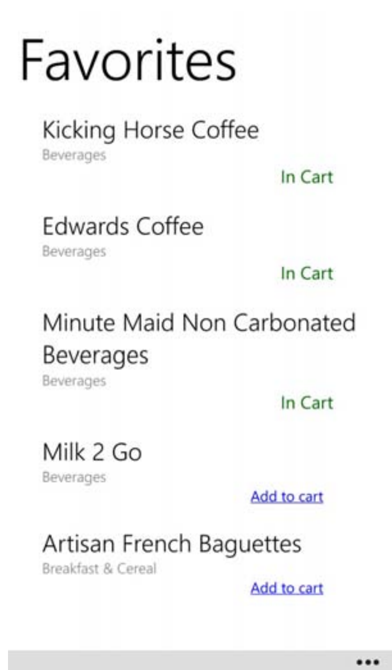

Figure 3.23: The Page that Views the Client's Favorite List

Chapter 4

Experimental Analysis

In this chapter, we propose several cases to demonstrate the correctness and the performance of our proposed solution. We first talk about the datasets and the experiments case scenarios used. Then we show and discuss the results of the proposed scenarios.

4.1 Datasets

We used synthetic data to run the experiments in order to demonstrate the correctness and the performance of our proposed solution. The synthetic data is shown in Appendix A. The data that we have generated and used was based on the nature of the real data available in stores.

4.2 Experiments Case Scenarios

In order to testify our framework, we mainly measured the correctness and the performance of the main two modules in our system which are:

- The module to generate the recommended items.
- The module to select the best store to be visited.

For our experiments, we used our synthetic data, shown in appendix A, to build three cases as shown in Tables 4.1, 4.2, and 4.3.

Item ID	Item Name
1	Milk - 4L
4	Eating Right White Bread
11	Bakery Counter Cheese Sticks
23	Bick's Pickles
44	Laughing Cow VQR
50	Hash Brown Potatoes
77	Benylin Syrup
78	Always or Tampax
80	Ivory Body Wash
87	Purina Pet Products

Table 4.1: The First Built Case to Test the Recommended Items Module

Item ID	Item Name
4	Eating Right White Bread
5	Pampers Mega Pack Diapers
10	Artisan French Baguettes
55	Del Monte Bananas
66	Open Nature Fresh Filled Pasta

Table 4.2: The Second Built Case to Test the Recommended Items Module

Item ID	Item Name
5	Pampers Mega Pack Diapers
12	Bakery Counter Chocolate Chip Cookies
77	Benylin Syrup

Table 4.3: The Third Built Case to Test the Recommended Items Module

4.3 Correctness Measurement

In this section, we will compare the system's output for the proposed scenarios with the actual output that was calculated manually to demonstrate the system correctness.

4.3.1 Correctness Measure for the Recommended Items Module

In order to test this module, we built three scenarios to demonstrate the correctness of this module. The first two scenarios will generate the recommended items based on the two cases shown in Tables 4.1 and 4.2, respectively, and specific system client's historical transactions. The third scenario will propose the suggested items based on the case shown in Table 4.3 and all system clients' historical transactions that exist in the database. The minimum support threshold that was used for the association rules mining algorithm in our experiments was 50%. Also, the value for the confidence used is 50%. These values have been arbitrarily set based on the data we have in order to get some results.

In the first scenario, we used the case shown in Table 4.1 along with the historical transactions for the system's client who has Client ID equals to 6 to generate the recommended items. After running the Apriori algorithm that was used in our system manually, we got the association rules mining that intersect with our case in the left hand side. These rules are shown in Table 4.4. The first rule in Table 4.4 means that, at least 50% from the lists purchased by our specific client contain Eating Right White Bread if they contain Milk - 4L.

Association Rule Mining
Milk - 4L ⇒ Eating Right White Bread
Eating Right White Bread ⇒ Clover Leaf White Tuna
Huggies High Count Jr. Diapers ⇒ Kicking Horse Coffee
Huggies High Count Jr. Diapers ⇒ Pampers Mega Pack Diapers

Table 4.4: The Manual Apriori Result for the First Scenario

Based on this scenario, the recommended items that were generated by our proposed solution are shown in Table 4.5.

Recommended Item
Clover Leaf White Tuna
Kicking Horse Coffee
Pampers Mega Pack Diapers

Table 4.5: The Recommended Items that Were Generated for the First Scenario

By comparing the association rules that were determined manually with the recommended items that were proposed by our system, we can say that, our system generated the recommended items as expected and the correctness is 100%.

In the second scenario, we used the case shown in Table 4.2 along with the historical transactions for the system's client who has Client ID equals to 3 to generate the recommended items. The manual result for this scenario is shown in Table 4.6.

Association Rule Mining
Eating Right White Bread ⇒ Milk - 2L
Eating Right White Bread ⇒ Pampers Mega Pack Diapers
Eating Right White Bread ⇒ Milk - 4L, Butter - Alberta Food
Pampers Mega Pack Diapers ⇒ Eating Right White Bread, Clover Leaf White Tuna
Eating Right White, Pampers Mega Pack Diapers ⇒ Clover Leaf White Tuna

Table 4.6: The Manual Apriori Result for the Second Scenario

The recommended items that were generated by our proposed solution for this scenario are shown in Table 4.7.

Based on the results in Table 4.6 and Table 4.7, the generated items from our system are the same as we determined manually and hence the correctness is 100%.

In the third scenario, we used the case shown in Table 4.3 along with all the historical transactions in the dataset in order to propose the suggested items. The manual result for this scenario is shown in Table 4.8.

Recommended Item
Milk - 4L
Butter - Alberta Food
Milk - 2L
Clover Leaf White Tuna

Table 4.7: The Recommended Items that Were Generated for the Second Scenario

Association Rule Mining
Pampers Mega Pack Diapers ⇒ Eating Right White Bread

Table 4.8: The Manual Apriori Result for the Third Scenario

After applying the Apriori algorithm by our system over all the transactions that exist in the dataset, Table 4.9 reports the suggested items that are proposed to the client based on his/her selected items shown in Table 4.3.

Recommended Item
Eating Right White Bread

Table 4.9: The Recommended Items that Were Generated for the Third Scenario

By comparing Table 4.8 with Table 4.9, we can say that the results are identical and the correctness is 100%.

4.3.2 Correctness Measure for Selecting the Best Store to be Visited Module

For this module, we built three scenarios to demonstrate the correctness of this module. We used the same three cases shown in TablesTables 4.1, 4.2, and 4.3, respectively. Each case along with longitude and latitude coordinators represents a scenario. However, each store in our system has its longitude and latitude coordinators. The stores' information are shown in Appendix A.

In the first scenario, we used the case shown in Table 4.1 along with longitude and latitude coordinators equal (51.070743, -114.055939).

We manually calculated the distance between the given location and each store in our dataset by using Haversine formula. The result is shown in Table 4.10.

Store Name	**Distance In Kilometer**
Brentwood	5.05
Dalhousie	8.11
Crowfoot	11.79

Table 4.10: The Distance between the Given Location in the First Scenario and Each Store in the Dataset

Also, we manually scanned the dataset and checked the availability of the items in this scenario in each store. The list of the stores associated with the number of available items in each store is shown in Table 4.11.

Store Name	**Number of Non Available Items**
Brentwood	0
Dalhousie	5
Crowfoot	2

Table 4.11: The Number of Non Available Items from the Given Items in the First Scenario in Each Store

Based on this scenario, the stores' list that was generated by our proposed solution is shown in Table 4.12.

Store Name	**Distance In Kilometer**	**Number of Non Available Items**
Brentwood	5.05	0
Dalhousie	8.11	5
Crowfoot	11.79	2

Table 4.12: The Stores' List that Was Generated by Our System for the First Scenario

As we have in Tables 4.10, 4.11, and 4.12, the output from our system is as we expected and the correctness is 100%.

Table 4.13 and Table 4.14 show the identical manual and systematic results for the second and the third scenarios, respectively. For the second scenario, we used the case shown in Table 4.2 along with longitude and latitude coordinators equal to (51.092742, -114.101944). For the third scenario, we used the case shown in Table 4.3 with longitude and latitude coordinators equal to (51.106971, -114.167519).

Store Name	Distance In Kilometer	Number of Non Available Items
Brentwood	1.76	0
Dalhousie	4.15	3
Crowfoot	7.78	1

Table 4.13: The Identical Manual and Systemic Stores' List for the Second Scenario

Store Name	Distance In Kilometer	Number of Non Available Items
Dalhousie	0.72	2
Crowfoot	3.08	0
Brentwood	3.88	0

Table 4.14: The Identical Manual and Systemic Stores' List for the Third Scenario

4.4 Performance Measurement

System performance measure is an important part of testing and refining any system. Performance and metrics standards help to make sure that the system meets expectations and functions properly.

In this section, we will test system performance effectiveness and check if it is providing acceptable functioning services to an end-user. For this, we built six scenarios to measure the performance for the first module and three scenarios for the second module.

4.4.1 The Experiments Environment

All the experiments have been conducted on a computer connected to the internet with the following specifications:

- **Processor:** Core i5-2400 3.1 GHz
- **Memory:** 8.00 GB
- **Operating System:** 64-bit Windows

4.4.2 Performance Measure for The Recommended Items Module

For this module, we used the cases shown in Tables 4.1, 4.2, and 4.3, respectively, to build six scenarios, two scenarios for each case.

For the first three scenarios, we will propose the suggested items based on the cases shown in Tables 4.1, 4.2, and 4.3, respectively, and all system clients' historical transactions that exist in the database. Tables 4.15, 4.16, and 4.17 show the number of the served and stacked requests per second, and the module's throughput, number of requests served per second, for each scenario with various arrival requests rates in 30 seconds. The tables are followed by their representative figures. Each table has a figure shows the module's throughput for the various arrival requests' rates over time.

Arrival Rate (Requests/Second)	**Served Requests**	**Staked Requests**	**Throughput (Requests/Second)**
5	146	4	4.87
10	291	9	9.7
20	386	214	12.87
30	386	514	12.87

Table 4.15: The Recommended Items Module's Performance Based on the First Scenario

Arrival Rate (Requests/Second)	Served Requests	Staked Requests	Throughput (Requests/Second)
5	146	4	4.87
10	291	9	9.7
20	383	217	12.77
30	382	518	12.73

Table 4.16: The Recommended Items Module's Performance Based on the Second Scenario

Arrival Rate (Requests/Second)	Served Requests	Staked Requests	Throughput (Requests/Second)
5	146	4	4.87
10	291	9	9.7
20	376	224	12.53
30	379	521	12.63

Table 4.17: The Recommended Items Module's Performance Based on the Third Scenario

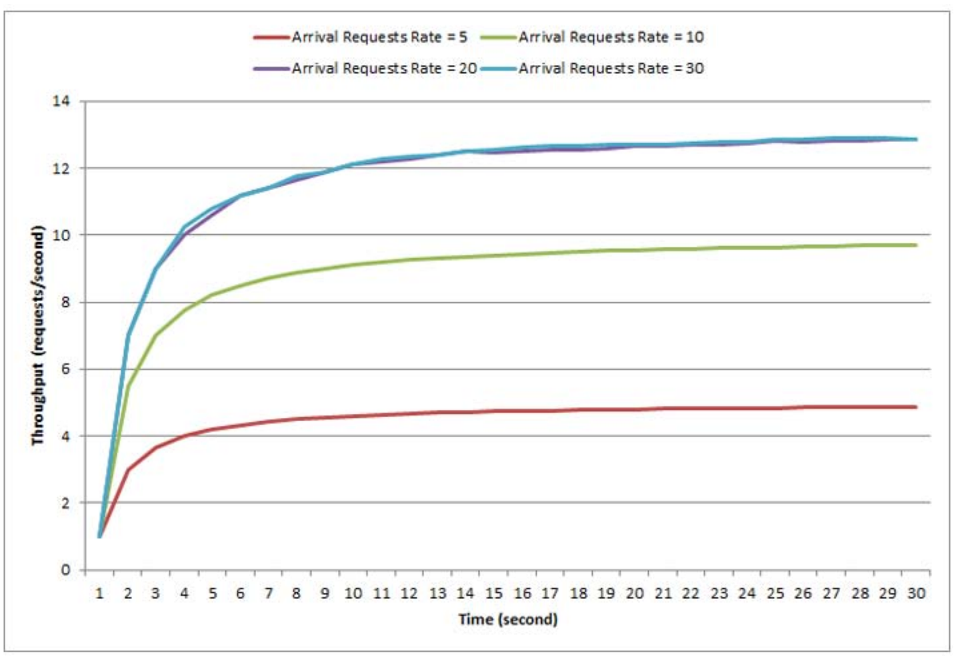

Figure 4.1: The Recommended Items Module's Throughput for the First Scenario

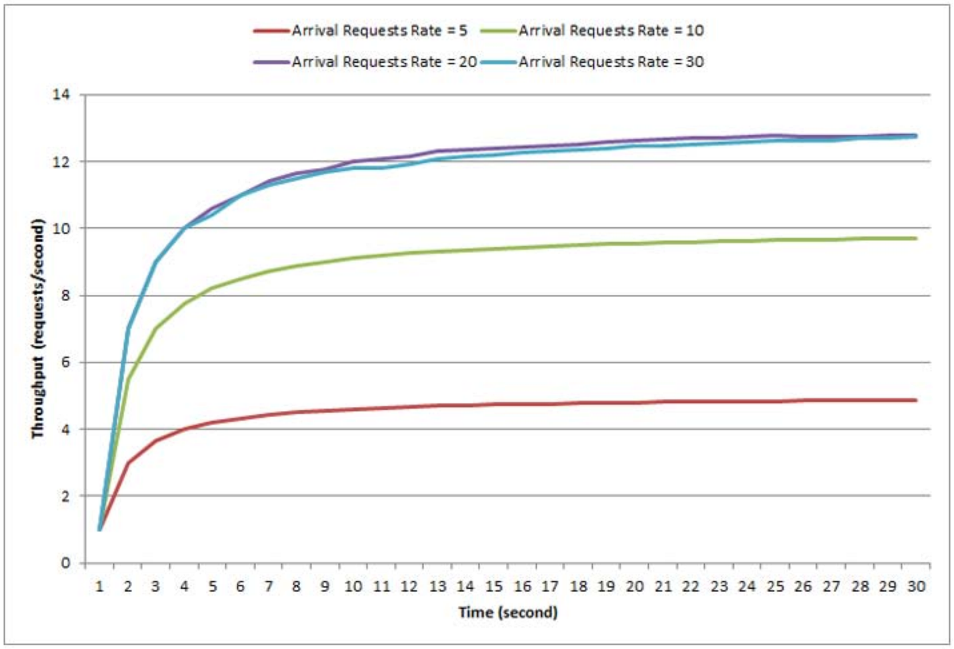

Figure 4.2: The Recommended Items Module's Throughput for the Second Scenario

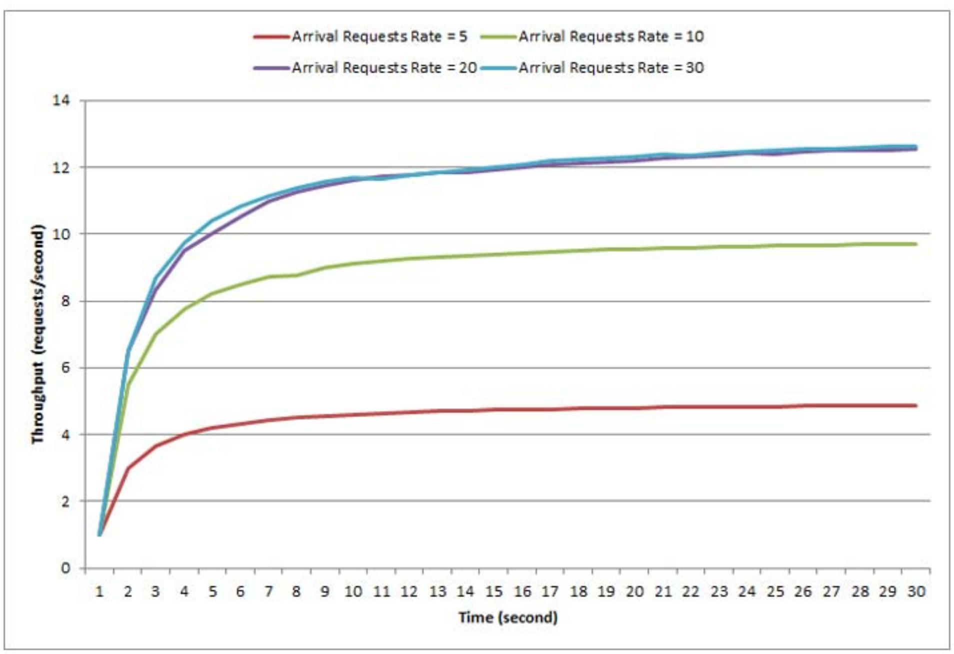

Figure 4.3: The Recommended Items Module's Throughput for the Third Scenario

For the second three scenarios, we will generate the recommended items based on the cases shown in Tables 4.1, 4.2, and 4.3, respectively, and specific system client's historical transactions which are for the client who has Client ID equals to 1.

Tables 4.18, 4.19, and 4.20 show module's throughput for each scenario with various arrival requests rates in 30 seconds. The representative figures that show the number of served and stacked requests per second and the throughput for the different arrival requests rates over time are reported.

Arrival Rate (Requests/Second)	**Served Requests**	**Staked Requests**	**Throughput (Requests/Second)**
5	146	4	4.87
10	291	9	9.7
20	357	243	11.9
30	357	543	11.9

Table 4.18: The Recommended Items Module's Performance Based on the Fourth Scenario

Arrival Rate (Requests/Second)	**Served Requests**	**Staked Requests**	**Throughput (Requests/Second)**
5	146	4	4.87
10	291	9	9.7
20	354	246	11.8
30	353	547	11.77

Table 4.19: The Recommended Items Module's Performance Based on the Fifth Scenario

After analyzing the results in Figure 4.1 to Figure 4.6 which are representing the module's throughput for each scenario with respect to time, we found out that the maximum throughput rate in the first scenarios' group, which is based on all system clients' historical transactions, is 12.87 (requests/second); this means the specific module can serve 12.87 requests every second without any delay.

Arrival Rate (Requests/Second)	Served Requests	Staked Requests	Throughput (Requests/Second)
5	146	4	4.87
10	291	9	9.7
20	355	245	11.83
30	362	538	12.07

Table 4.20: The Recommended Items Module's Performance Based on the Sixth Scenario

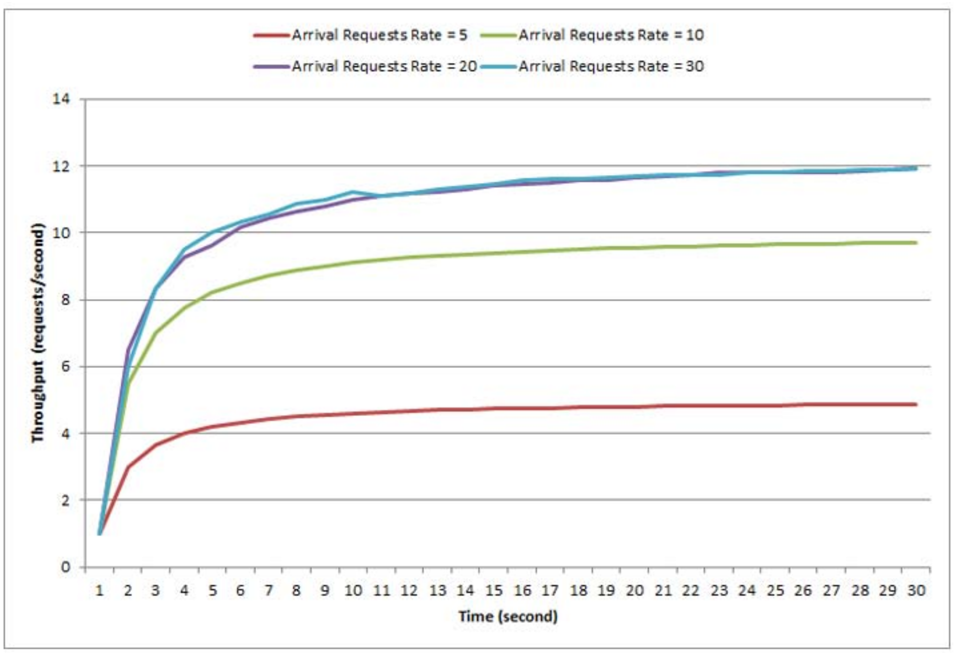

Figure 4.4: The Recommended Items Module's Throughput for the Fourth Scenario

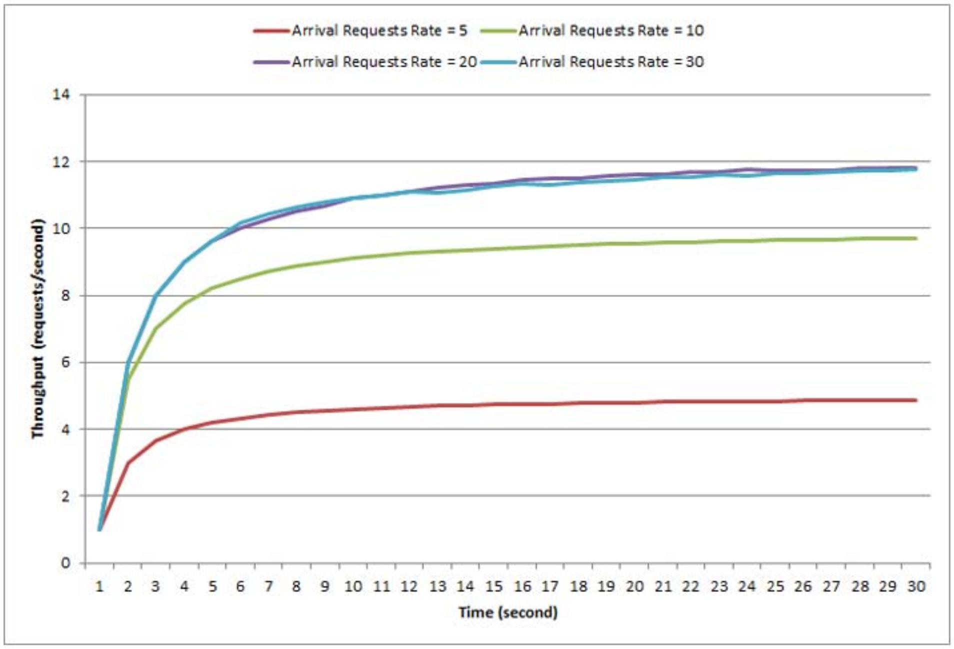

Figure 4.5: The Recommended Items Module's Throughput for the Fifth Scenario

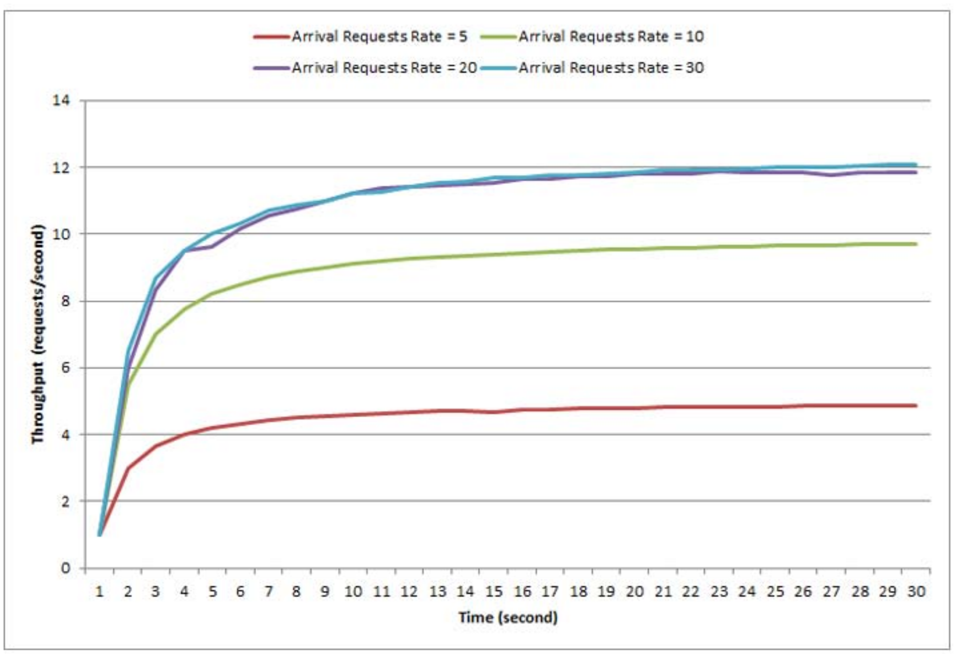

Figure 4.6: The Recommended Items Module's Throughput for the Sixth Scenario

On other hand, the maximum throughput rate for the second scenarios' group is 12.07 (requests/second).

As we notice, the throughput rate in the second group is less than the first group which can be explained by having the projection operation in the database server to select the historical transactions for a specific client.

However, we may improve the module's performance by storing the clients' association rules that are used to generate the recommended items as a precompiled data which can be fetched faster.

4.4.3 Performance Measure for Selecting the Best Store to be Visited Module

For this module, we built three scenarios to test the module's performance. We used the same three cases shown in Tables 4.1, 4.2, and 4.3, respectively. Each case along with longitude and latitude coordinators represents a scenario.

We used the case shown in Table 4.1 along with longitude and latitude coordinators equal to (51.070743, -114.055939) as the first scenario. Table 4.2 along with (51.092742, -114.101944) coordinators as the second scenario, and Table 4.3 along with (51.106971, -114.167519) coordinators as the third scenario.

The module's throughput for each scenario with various arrival requests rates in 30 seconds are shown in Tables 4.21, 4.22, and 4.23. The representative figures are Figures 4.7, 4.8, and 4.9, respectively.

Arrival Rate (Requests/Second)	**Served Requests**	**Staked Requests**	**Throughput (Requests/Second)**
5	146	4	4.87
10	291	9	9.7
20	408	192	13.6
30	406	494	13.53

Table 4.21: The Best Store to be Visited Module's Performance Based on the First Scenario

Arrival Rate (Requests/Second)	Served Requests	Staked Requests	Throughput (Requests/Second)
5	146	4	4.87
10	291	9	9.7
20	408	192	13.6
30	407	493	13.57

Table 4.22: The Best Store to be Visited Module's Performance Based on the Second Scenario

Arrival Rate (Requests/Second)	Served Requests	Staked Requests	Throughput (Requests/Second)
5	146	4	4.87
10	291	9	9.7
20	405	195	13.5
30	404	496	13.47

Table 4.23: The Best Store to be Visited Module's Performance Based on the Third Scenario

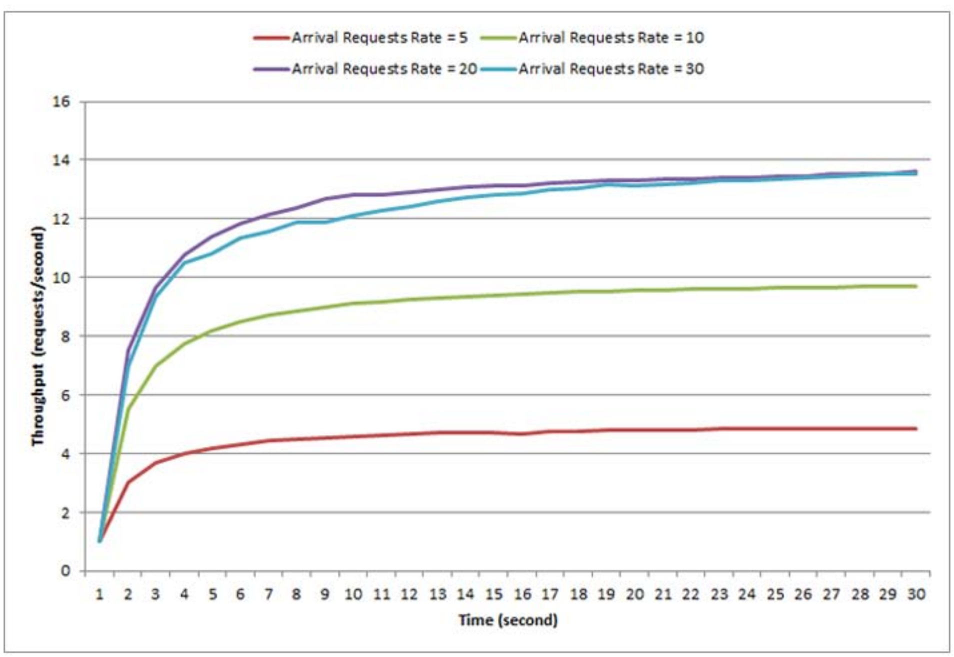

Figure 4.7: The Best Store to be Visited Module's Throughput for the First Scenario

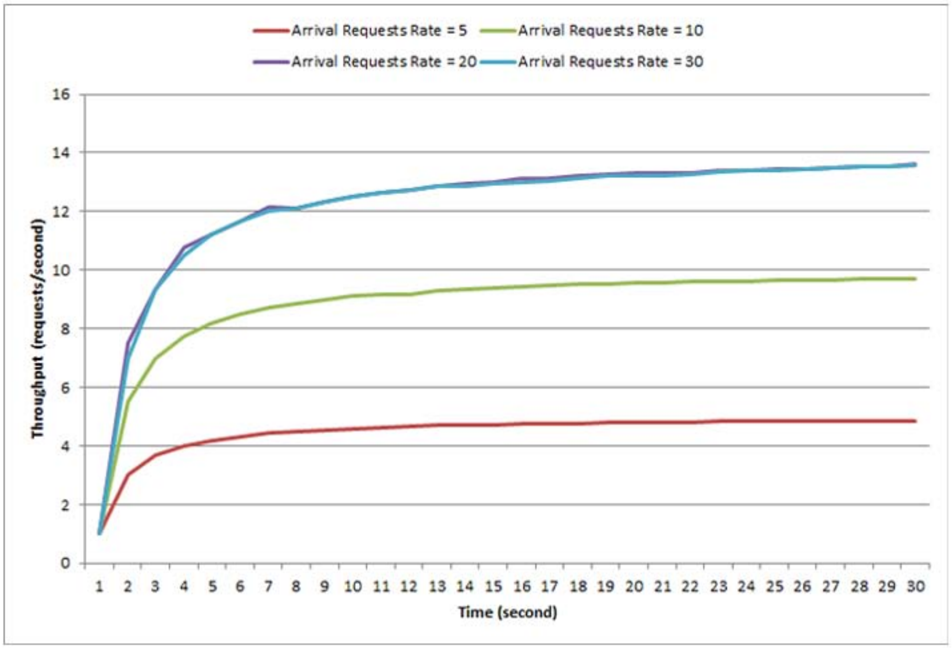

Figure 4.8: The Best Store to be Visited Module's Throughput for the Second Scenario

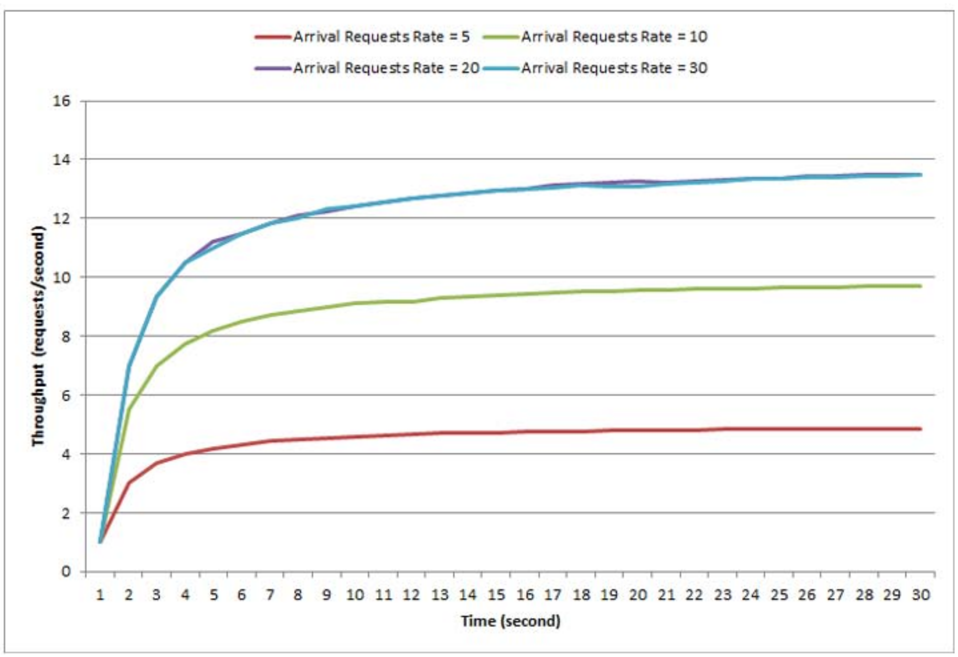

Figure 4.9: The Best Store to be Visited Module's Throughput for the Third Scenario

After analyzing the results in Figures 4.7, 4.8, and 4.9, we found out that the maximum throughput rate is 13.57 (requests/second), which means this module can serve 13.57 requests every second without any delay.

Chapter 5

Summary, Conclusions and Future Work

In this chapter, we summarize the outcome of this thesis, discuss the research limitations, and discuss several directions for the future work.

5.1 Summary and Conclusions

In this thesis, we discussed a group of issues that exist in the shopping process system and showed how data mining techniques and social network analysis can be used as modules in a framework to address these issues. We designed and implemented a framework that will turn the shopping process into an interactive process, speed up the shopping process, save money, and save the environment.

We can summarize our contributions described in this thesis as below:

1. We integrated the social network analysis concepts with the shopping process to increase the benefit from the shopping process by converting it into an interactive process, saving money, and trying to keep the environment clean.

2. We discovered frequent items that usually purchased by the consumer in order to introduce them as recommended items based on the selected items. This improvement will increase the quality of the shopping system and will minimize the number of the shopping trips.

3. We incorporate some geometry concepts with the shopping process to help in finding the nearest stores or the stores that are within a closed range in order to save time and fuel, and avoid polluting the environment as much as we can.

4. We introduced an algorithm to build a navigational map to be used by the consumer inside the store for saving his/her time and better planning to pick the target items.

5.2 Research Limitations

- This study is limited to the shopping process; but it can be adapted to work for other domains that have same kind of specifications (Transactions/Items). For instance, it can be applied in the health care system. Consider patients/health care services and lab tests/hospitals and clinics. In this model, a patient needs to have some health care services and lab tests. The system may suggest him to ask for some health care services and lab tests based on his/her history, contact his/her social network that may have some doctors or persons who have health care experiences, and suggest him the best hospital or clinic to serve him/her.
- This research is to be applied to a shopping system of a community where smartphones, GPS service, coordinators of stores, locations of items inside stores exist.

5.3 Future Work

The proposed solution introduced in this thesis has several findings that can lead to some extensions in order to increase the performance quality for the system and solve other issues in the shopping process. We plan to extend this work in a number of directions:

1. Enhance the system by allowing the consumer to enter what items he/she wants to purchase along with what they want to save, for example money, time, fuel, or any combination of the three. Then, the system will return the best list of stores in the consumer's area where the savings will be optimized.
2. Incorporate the consumer's items stock with the existing proposed solution to notify the consumer about his/her needed items before consuming them. For example, if

the consumer used to purchase 4 Liters of milk and he/she informed the system that he/she consumes 200 mL milk every day, the system will notify him/her after 20 days from his/her last purchasing that he/she needs to purchase milk.

3. Use the FP-Growth algorithm instead of the Apriori algorithm to generate the association rules. Apriori needs to scan the database for each k-candidate set produced; but FP-Growth scans the database only twice to generate the frequent patterns.

4. Due to the limitations that exist in the FP-Growth, e.g., databases keep changing over time or users may change the threshold of support which leads to repetition of the whole mining process. We will generate the frequent patterns and store them as a precompiled data in the framework's database and update them frequently in order to increase the performance of the system.

5. Consider more features in the social network module and make it more interactive, such as providing the ability to know if there is any family member or friend present in the store or the shopping center where a customer is shopping so that he/she can meet him/her.

Bibliography

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, volume 22, pages 207–216. ACM, 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Databases*, volume 1215, pages 487–499, 1994.

[3] Y. Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of topological characteristics of huge online social networking services. In *Proceedings of the 16th international conference on World Wide Web*, pages 835–844. ACM, 2007.

[4] R. Analytics. http://www.retail-analytics.com, February 2014.

[5] J. A. Barnes. Class and committees in a norwegian island parish. *Human Relations*, 7(0):39–58, 1954.

[6] S. A. Birnie and P. Horvath. Psychological predictors of internet social communication. *Journal of Computer-Mediated Communication*, 7(4):0–0, 2002.

[7] R. C. Blattberg, B.-D. Kim, and S. A. Neslin. Market basket analysis. *Database Marketing: Analyzing and Managing Customers*, pages 339–351, 2008.

[8] Canalys. http://www.canalys.com, February 2014.

[9] S. J. Cunningham and E. Frank. Market basket analysis of library circulation data. In *Proceedings of the 6th International Conference on Neural Information Processing*, volume 2, pages 825–830. IEEE, 1999.

[10] C. Dennis, D. Marsland, and T. Cockett. Data mining for shopping centres–customer knowledge-management framework. *Journal of Knowledge Management*, 5(4):368–374, 2001.

[11] J. Dewey. How we think. 1910.

[12] H. A. Edelstein. *Introduction to Data Mining and Knowledge Discovery.* Two Crows Corporation, 1999.

[13] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. Advances in knowledge discovery and data mining. 1996.

[14] Gartner. http://www.gartner.com, February 2014.

[15] M. Giering. Retail sales prediction and item recommendations using customer demographics at store level. *ACM SIGKDD Explorations Newsletter*, 10(2):84–89, 2008.

[16] D. T. Green and J. M. Pearson. Social software and cyber networks: Ties that bind or weak associations within the political organization. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, pages 117b–117b. IEEE, 2005.

[17] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *VLDB*, volume 95, pages 420–431, 1995.

[18] J. Han, M. Kamber, and J. Pei. *Data mining: concepts and techniques.* Morgan kaufmann, 2006.

[19] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *ACM SIGMOD Record*, volume 29, pages 1–12. ACM, 2000.

[20] G. Harper and S. D. Pickett. Methods for mining hts data. *Drug Discovery Today*, 11(15):694–699, 2006.

[21] D. I. Hawkins, R. J. Best, and K. A. Coney. *Consumer behavior.* Irwin, 1998.

[22] V. Jain, G. S. Narula, and M. Singh. Implementation of data mining in online shopping system using tanagra tool. *International Journal of Computer Science and Engineering*, 2013.

[23] J. K. Kim, Y. H. Cho, W. J. Kim, J. R. Kim, and J. H. Suh. A personalized recommendation procedure for internet shopping support. *Electronic Commerce Research and Applications*, 1(3):301–313, 2003.

[24] P. Kotler. *Marketing management: A south Asian perspective.* Pearson Education India, 2009.

[25] D. T. Larose. *Discovering knowledge in data: an introduction to data mining.* Wiley.com, 2005.

[26] A. H. Maslow. A theory of human motivation. *Psychological review*, 50(4):370, 1943.

[27] G. MobileCom. http://www.gs1.org, March 2014.

[28] H. Müller, T. Pun, and D. Squire. Learning from user behavior in image retrieval: Application of market basket analysis. *International Journal of Computer Vision*, 56(1-2):65–77, 2004.

[29] F. Nicosia. Consumer decision processes: marketing and advertising implications. 1966.

[30] B. of Labor Statistics. http://www.bls.gov, February 2014.

[31] B. of Labor Statistics. http://www.collierscanada.com, February 2014.

[32] T. Paper. http://www.telecompaper.com, February 2014.

[33] J. S. Park, M.-S. Chen, and P. S. Yu. *An effective hash-based algorithm for mining association rules*, volume 24. ACM, 1995.

[34] Y. Peng, G. Kou, Z. Chen, and Y. Shi. Recent trends in data mining (dm): Document clustering of dm publications. In *The 2006 International Conference on Service Systems and Service Management*, volume 2, pages 1653–1659. IEEE, 2006.

[35] B. J. Pine. *Mass customization: the new frontier in business competition.* Harvard Business Press, 1999.

[36] O. M. Planet. http://www.think.withgoogle.com/mobileplanet, February 2014.

[37] G. Prassas, K. C. Pramataris, O. Papaemmanouil, and G. Doukidis. Dynamic recommendations in internet retailing. In *ECIS*, pages 368–379, 2001.

[38] R. Rastogi and K. Shim. Scalable algorithms for mining large databases. In *The 5th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 73–140. ACM, 1999.

[39] J. Roberto and J. Bayardo. Efficiently mining long patterns from databases. In *ACM Sigmod Record*, volume 27, pages 85–93. ACM, 1998.

[40] A. Savasere, E. R. Omiecinski, and S. B. Navathe. An efficient algorithm for mining association rules in large databases. 1995.

[41] J. B. Schafer, J. Konstan, and J. Riedl. Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, pages 158–166. ACM, 1999.

[42] R. Srikant and R. Agrawal. Mining generalized association rules. In *VLDB*, volume 95, pages 407–419, 1995.

[43] R. Srikant and R. Agrawal. *Mining sequential patterns: Generalizations and performance improvements.* Springer, 1996.

[44] J. Star and J. Estes. *Geographic information systems.* prentice-Hall Englewood Cliffs, 1990.

[45] M. Type. http://www.movable-type.co.uk/, April 2014.

[46] S. Wasserman. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[47] B. Wellman. An electronic group is virtually a social network. *Culture of the Internet*, 4:179–205, 1997.

[48] Wikipedia. http://www.wikipedia.org, February 2014.

[49] G. Zaltman and M. Wallendorf. *Consumer behavior, basic findings and management implications*. Wiley, 1979.

Appendix A

Dataset

ID	Category ID	Name
1	15	Milk - 4L
2	15	Butter - Alberta Food
3	15	Milk - 2L
4	1	Eating Right White Bread
5	9	Pampers Mega Pack Diapers
6	10	Kicking Horse Coffee
7	10	Edwards Coffee
8	10	Minute Maid Non Carbonated Beverages
9	10	Milk 2 Go
10	11	Artisan French Baguettes
11	1	Bakery Counter Cheese Sticks
12	1	Bakery Counter Chocolate Chip Cookies
13	1	Bakery Counter Dutch Crunch Bread
14	1	Bakery Counter Crusty Kaiser Rolls
15	11	Kellogg's Frozen Breakfast Sandwiches
16	11	Post Cereal
17	11	Quaker Instant Oatmeal
18	12	Ocean's Albacore Tuna
19	12	Unico Tomatoes
20	12	Clover Leaf White Tuna
21	12	Gold Seal Light Tuna
22	12	Campbell's Creations Soup
23	13	Bick's Pickles
24	13	Garden Select Pasta Sauce
25	13	Open Nature Pure Sea Salt
26	13	Berio Olive Oil

Continued on next page

Table A.1 – continued from previous page

ID	Category ID	Name
27	13	California Olive Ranch, Extra Virgin Olive Oil
28	14	Old Dutch Box Potato Chips
29	14	Mars Multipack Chocolate Bars
30	14	Christie Cookies
31	14	Quaker Crispy Minis
32	14	Ruffles Potato Chips
33	14	Leclerc Cookies
34	15	Black Diamond Cheese
35	15	Danone Activa
36	15	Imperial Margarine
37	15	Philadelphia Cream Cheese
38	15	Libert Greek Yogourt
39	15	Kraft Singles Cheese Slices
40	15	Kraft Shredded Cheese
41	15	Yoplait Source Yogourt
42	16	Signature CAFE Pizzas
43	16	Signature CAFE Side Salads
44	16	Laughing Cow VQR
45	17	Tulips
46	17	Phalaenopsis Orchids in Clay Pot
47	17	Daffodils
48	18	Delissio Frozen Pizza
49	18	Open Nature Greek Frozen Yogourt
50	18	Hash Brown Potatoes
51	18	SELECT Cheese & Spinach Cannelloni
52	19	Tomatoes On the Vine
53	19	Strawberries
54	19	Whole White Mushrooms
55	19	Del Monte Bananas
56	19	Fresh Avocados

Continued on next page

Table A.1 – continued from previous page

ID	Category ID	Name
57	19	Royal Gala Apples
58	19	Produce Stand Vegetable Tray
59	19	O Organics Salads
60	19	Long English Cucumbers
61	20	Barilla Pasta
62	20	Knorr Sidekicks
63	20	Kraft Dinner Original
64	20	O Organics Rice
65	20	Olivieri Fresh Filled Pasta
66	20	Open Nature Fresh Filled Pasta
67	21	BlueWater Fillets
68	21	Fresh Atlantic Salmon Fillets
69	21	Eating Right Wild Pacific Salmon Burgers
70	21	Aqua Star Tilapia Fillets
71	22	Clorox Green Works
72	22	Finish Quantum
73	22	Lysol Disinfecting Wipes
74	22	Lysol Toilet Bowl Cleaners
75	22	Purex Bath Tissue
76	22	Sifto Crystal Plus Water Softener Salt
77	23	Benylin Syrup
78	23	Always or Tampax
79	23	Fructis Hair Care
80	23	Ivory Body Wash
81	23	L'Oreal Expertise
82	23	L'Oreal Preference
83	23	Metamucil
84	23	Schick Razors or Refills
85	23	NyQuil or DayQuil
86	23	Webber Vitamins

Continued on next page

Table A.1 – continued from previous page

ID	**Category ID**	**Name**
87	24	Purina Pet Products

Table A.1: Item Table

ID	Name
1	Bread & Bakery
9	Baby Care
10	Beverages
11	Breakfast & Cereal
12	Canned Goods & Soups
13	Condiments, Spices & Bake
14	Cookies, Snacks & Candy
15	Dairy, Eggs & Cheese
16	Deli
17	Flowers & Decor
18	Frozen Foods
19	Fruits & Vegetables
20	Grains, Pasta & Sides
21	Meat & Seafood
22	Paper, Cleaning & Home
23	Personal Care & Health
24	Pet Care

Table A.2: Item Category Table

ID	Name	Postal_Code	Telephone	Longitude	Latitude
1	Crowfoot	T3G 2L5	(403) 241-3814	-114.2	51.12548
2	Brentwood	T2L 1K8	(403) 289-1424	-114.124	51.08531
3	Dalhousie	T3A 5R8	(403) 202-0425	-114.157	51.10584

Table A.3: Store Table

ID	First_Name	Last_Name	Email	Password
1	Tamer	Jarada	tjarada@gmail.com	M40FsP+6CcI=
2	Riad	Al-Jomaii	csk.riad@gmail.com	M40FsP+6CcI=
3	Peter	Yung	peter.yung@hotmail.com	Eij/RHvezE0=
4	Omar	Addam	omaddam@gmail.com	M40FsP+6CcI=
5	John	Smith	john.smith@gmail.com	pzRB4mLO91c=
6	Amy	Novak	amynovak@gmail.com	pzRB4mLO91c=
7	James	Rock	james.roch@gmail.com	pzRB4mLO91c=

Table A.4: Client Table

List_ID	**Client_ID**	**Store_ID**	**List_Date**
1	1	1	1/1/2014
2	1	1	1/2/2014
3	1	1	2/2/2014
4	1	2	2/22/2014
5	1	2	3/10/2014
6	1	1	3/10/2014
7	1	3	3/22/2014
8	1	3	3/22/2014
9	1	2	3/22/2014
10	2	1	3/30/2014
11	2	1	3/31/2014
12	2	3	4/1/2014
13	2	2	4/2/2014

Continued on next page

Table A.5 – continued from previous page

List_ID	Client_ID	Store_ID	List_Date
14	2	1	4/3/2014
15	2	1	4/4/2014
16	2	2	4/5/2014
17	3	1	3/1/2014
18	3	2	3/2/2014
19	3	1	3/3/2014
20	4	1	3/3/2014
21	4	1	3/10/2014
22	4	2	3/15/2014
23	5	1	3/10/2014
24	5	3	3/1/2014
25	5	1	4/1/2014
26	5	1	2/1/2014
27	5	2	3/1/2014
28	6	1	3/1/2014
29	6	2	4/1/2014
30	6	1	4/2/2014
31	7	1	4/3/2014
32	7	1	3/30/2014
33	7	2	3/28/2014

Table A.5: Purchased List Table

List_ID	Item_ID
1	4
1	5
1	6
1	8

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
1	10
1	20
1	21
2	4
2	5
2	6
2	10
2	60
3	4
3	5
3	6
3	10
3	24
3	40
3	50
3	55
4	4
4	5
4	6
4	20
4	21
4	33
4	50
5	4
5	5
5	6
5	10
5	20
5	21
5	33

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
6	4
6	5
6	6
6	10
6	15
6	20
6	21
6	55
6	60
7	4
7	5
7	6
7	9
7	10
7	20
8	4
8	6
8	9
8	10
8	21
8	60
9	4
9	5
9	6
9	9
9	10
9	20
9	60
10	1
10	2

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
10	3
10	4
10	6
11	1
11	3
11	4
11	6
11	20
11	21
12	2
12	3
12	4
12	20
12	21
12	44
12	50
13	1
13	2
13	3
13	4
13	20
13	44
13	60
13	70
14	1
14	2
14	3
14	4
14	7
14	44

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
14	60
14	62
14	70
15	2
15	3
15	4
15	5
15	33
15	60
15	70
16	1
16	2
16	3
16	4
16	20
16	44
16	60
16	70
17	3
17	4
17	5
17	10
17	20
18	1
18	2
18	3
18	4
18	55
18	63
18	80

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
19	1
19	2
19	4
19	5
19	6
19	20
19	21
19	22
19	33
19	44
20	1
20	2
20	3
20	4
20	20
20	21
20	33
20	40
20	55
20	66
21	4
21	6
21	10
21	11
21	20
21	22
21	55
21	60
21	63
21	70

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
22	1
22	3
22	4
22	5
22	6
22	8
22	33
22	41
22	44
22	55
22	63
23	1
23	2
23	4
23	5
23	6
23	10
24	1
24	4
24	5
24	7
24	8
24	20
24	21
24	22
24	23
24	24
24	60
24	66
25	1

Continued on next page

Table A.6 – continued from previous page

List_ID	**Item_ID**
25	3
25	4
25	6
25	11
25	20
25	21
25	22
25	25
25	60
25	67
26	1
26	2
26	3
26	4
26	10
26	11
26	20
26	25
26	33
26	60
26	67
26	77
27	1
27	3
27	4
27	5
27	6
27	7
27	11
27	20

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
27	40
27	50
27	55
28	1
28	3
28	4
28	5
28	11
28	20
28	21
28	22
28	28
28	70
28	77
29	5
29	7
29	9
29	10
29	15
29	20
29	21
29	22
29	44
29	55
30	1
30	2
30	4
30	6
30	7
30	9

Continued on next page

Table A.6 – continued from previous page

List_ID	Item_ID
30	11
30	20
30	21
30	22
32	1
32	2
32	10
32	11
33	5
33	6
33	7
33	10
33	20
33	21
33	22
33	23
33	44
33	45

Table A.6: Purchased Item Table

Item_ID	tore_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
1	1	5	1	25	7	1
1	2	3	4	15	2	1
1	3	13	1	65	11	1
2	1	5	2	25	18	1
2	2	3	3	15	6	1
2	3	13	4	65	7	1

Continued on next page

Table A.7 – continued from previous page

Item_ID	Store_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
3	1	5	2	25	10	1
3	2	3	3	15	1	1
3	3	13	1	65	8	1
4	1	5	2	20	15	1
4	2	13	1	65	1	1
5	1	5	4	20	20	1
5	2	13	2	65	11	1
5	3	15	1	75	10	1
6	1	2	1	5	5	1
6	2	12	3	20	11	1
7	1	2	1	5	10	1
7	2	12	3	60	16	1
7	3	10	1	50	11	1
8	1	2	2	5	35	1
8	2	12	3	60	21	1
8	3	10	3	50	23	1
9	1	2	3	5	40	1
9	2	12	3	60	4	1
9	3	10	3	50	7	1
10	1	15	1	75	5	1
10	2	5	1	25	3	1
11	1	15	2	75	10	1
11	2	5	3	25	13	1
11	3	1	2	5	10	1
12	1	15	4	75	25	1
12	2	5	2	25	17	1
13	1	15	3	75	5	1
13	2	5	1	25	8	1
13	3	1	3	5	18	1
14	1	15	2	75	26	1

Continued on next page

Table A.7 – continued from previous page

Item_ID	Store_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
14	2	5	3	25	15	1
15	1	14	1	70	10	1
15	2	6	3	30	1	1
16	1	14	3	70	18	1
16	2	6	2	30	15	1
16	3	8	1	40	12	1
17	2	6	3	30	17	1
17	3	8	3	40	36	1
18	1	5	2	25	10	1
18	2	8	3	40	15	1
19	1	5	3	25	19	1
19	2	8	3	40	13	1
19	3	7	1	35	10	1
20	1	5	3	25	14	1
20	2	8	3	40	8	1
21	1	5	3	25	15	1
21	2	8	3	40	3	1
21	3	7	1	35	13	1
22	2	8	3	40	27	1
22	3	7	1	35	20	1
23	1	4	1	20	10	1
23	2	10	1	50	1	1
23	3	5	4	25	13	1
24	1	4	3	20	33	1
24	2	10	2	50	10	1
25	1	4	3	20	14	1
25	2	10	1	50	23	1
25	3	5	3	25	19	1
26	1	4	2	20	19	1
26	2	10	1	50	17	1

Continued on next page

Table A.7 – continued from previous page

Item_ID	Store_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
26	3	5	1	25	11	1
27	1	4	2	20	21	1
27	2	10	2	50	16	1
27	3	5	3	25	4	1
28	1	6	1	30	23	1
28	2	9	1	45	11	1
28	3	6	1	30	11	1
29	1	6	2	30	15	1
29	2	9	3	45	10	1
30	1	6	2	30	28	1
30	2	9	3	45	11	1
30	3	6	1	30	14	1
31	1	6	2	30	7	1
31	2	9	3	45	17	1
32	1	6	3	30	9	1
32	2	9	1	45	14	1
32	3	6	3	30	14	1
33	2	9	2	45	27	1
34	1	7	3	35	5	1
34	2	7	1	35	11	1
35	1	7	3	35	16	1
35	2	7	1	35	14	1
36	1	7	3	35	24	1
36	2	7	2	35	17	1
37	1	7	2	35	12	1
37	2	7	2	35	24	1
38	1	7	3	35	17	1
38	2	7	2	35	31	1
39	2	7	3	35	29	1
40	2	7	1	35	12	1

Continued on next page

Table A.7 – continued from previous page

Item_ID	Store_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
41	1	7	3	35	29	1
41	2	7	4	35	4	1
41	3	13	1	65	10	1
42	1	8	2	40	19	1
42	2	15	3	75	11	1
43	2	15	2	75	14	1
44	1	8	2	40	15	1
44	2	15	1	75	15	1
44	3	4	1	20	11	1
45	2	16	1	80	3	1
46	2	16	3	80	4	1
47	1	1	2	5	10	1
47	2	16	4	80	16	1
48	1	7	3	35	31	1
48	2	14	1	70	13	1
48	3	17	3	65	24	1
49	2	14	3	70	19	1
50	1	2	1	10	32	1
50	2	14	2	70	14	1
51	2	14	1	70	16	1
51	3	17	1	65	11	1
52	1	9	3	45	10	1
52	2	1	1	5	10	1
52	3	2	1	10	11	1
53	1	9	2	45	10	1
53	2	1	2	5	31	1
54	1	9	3	45	17	1
54	2	1	3	5	21	1
55	1	9	3	45	7	1
55	2	1	3	5	21	1

Continued on next page

Table A.7 – continued from previous page

Item_ID	Store_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
55	3	2	1	10	13	1
56	1	9	2	45	10	1
56	2	1	2	5	14	1
57	1	9	3	45	22	1
57	2	1	3	5	7	1
57	3	2	2	10	6	1
58	2	1	2	5	16	1
58	3	2	3	10	9	1
59	2	1	3	5	19	1
60	1	9	1	45	28	1
60	2	1	1	5	8	1
60	3	2	3	10	17	1
61	1	3	1	15	11	1
61	2	11	1	55	11	1
61	3	3	1	15	8	1
62	1	3	3	15	5	1
62	2	11	2	55	16	1
62	3	3	1	15	21	1
63	1	3	3	15	5	1
63	2	11	3	55	7	1
64	1	3	2	15	16	1
64	2	11	3	55	17	1
64	3	3	1	15	11	1
65	1	3	4	15	17	1
65	2	11	2	55	9	1
66	2	11	2	55	19	1
67	1	10	3	50	15	1
67	2	2	1	10	11	1
67	3	9	2	45	7	1
68	1	10	2	50	23	1

Continued on next page

Table A.7 – continued from previous page

Item_ID	Store_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
68	2	2	1	10	11	1
68	3	9	1	45	11	1
69	1	10	3	50	15	1
69	2	2	3	10	17	1
70	2	2	1	10	6	1
71	1	11	1	55	12	1
71	2	19	1	85	1	1
72	1	11	1	55	5	1
72	2	19	3	85	10	1
72	3	11	3	55	17	1
73	1	11	3	55	9	1
73	2	19	1	85	8	1
74	1	11	3	55	7	1
74	2	19	4	85	3	1
75	1	11	2	55	13	1
75	2	19	1	85	12	1
75	3	11	1	55	1	1
76	2	19	3	85	17	1
77	1	12	2	60	31	1
77	2	19	2	85	12	1
78	2	19	2	85	26	1
79	1	12	3	60	24	1
79	2	19	1	85	17	1
79	3	12	2	60	7	1
80	1	12	3	60	16	1
80	2	19	2	85	19	1
80	3	12	3	60	7	1
81	1	12	1	60	5	1
81	2	19	2	85	20	1
82	1	12	2	60	19	1

Continued on next page

Table A.7 – continued from previous page

Item_ID	Store_ID	Lane	Shelf	X_Coordinator	Y_Coordinator	Is_Avilable
82	2	19	2	85	2	1
83	2	19	3	85	5	1
84	1	12	1	60	18	1
84	2	19	2	85	26	1
84	3	12	3	60	17	1
85	2	19	1	85	3	1
86	1	12	1	60	3	1
86	2	19	1	85	17	1
86	3	12	1	60	11	1
87	2	20	1	100	2	1

Table A.7: Item In Store Table