



Channel Islands

CALIFORNIA STATE UNIVERSITY

*Helping Sapiens: A Geo-based mobile
Application to help people in the vicinity*

A Thesis Presented to
The Faculty of Computer Science Department

In (Partial) Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science

by

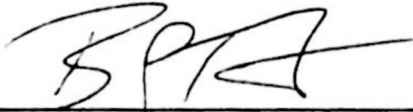
Student Name:
Simrandeep Singh

Advisor
Dr. Brian Thoms

December 2019

Copyright 2019

APPROVED FOR MS IN COMPUTER SCIENCE



12-16-19

ADVISOR: DR. BRIAN THOMS

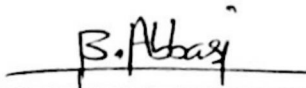
DATE



12-16-19

DR. JASON ISSACS

DATE



12 - 17 - 19

DR. ABBASI BAHAREH

DATE

APPROVED FOR THE UNIVERSITY

DR. OSMAN OZTURGUT

DATE

Non-Exclusive Distribution License

In order for California State University Channel Islands (CSUCI) to reproduce, translate and distribute your submission worldwide through the CSUCI Institutional Repository, your agreement to the following terms is necessary. The author(s) retain any copyright currently on the item as well as the ability to submit the item to publishers or other repositories.

By signing and submitting this license, you (the author(s) or copyright owner) grants to CSUCI the nonexclusive right to reproduce, translate (as defined below), and/or distribute your submission (including the abstract) worldwide in print and electronic format and in any medium, including but not limited to audio or video.

You agree that CSUCI may, without changing the content, translate the submission to any medium or format for the purpose of preservation.

You also agree that CSUCI may keep more than one copy of this submission for purposes of security, backup and preservation.

You represent that the submission is your original work, and that you have the right to grant the rights contained in this license. You also represent that your submission does not, to the best of your knowledge infringe upon anyone's copyright. You also represent and warrant that the submission contains no libelous or other unlawful matter and makes no improper invasion of the privacy of any other person.

If the submission contains material for which you do not hold copyright, you represent that you have obtained the unrestricted permission of the copyright owner to grant CSUCI the rights required by this license, and that such third party owned material is clearly identified and acknowledged within the text or content of the submission. You take full responsibility to obtain permission to use any material that is not your own. This permission must be granted to you before you sign this form.

IF THE SUBMISSION IS BASED UPON WORK THAT HAS BEEN SPONSORED OR SUPPORTED BY AN AGENCY OR ORGANIZATION OTHER THAN CSUCI, YOU REPRESENT THAT YOU HAVE FULFILLED ANY RIGHT OF REVIEW OR OTHER OBLIGATIONS REQUIRED BY SUCH CONTRACT OR AGREEMENT.

The CSUCI Institutional Repository will clearly identify your name(s) as the author(s) or owner(s) of the submission, and will not make any alteration, other than as allowed by this license, to your submission.

Helping Sapiens: Geo Based Mobile application to help people

Title of Item

Reputation System, Recommendation and Rating System

3 to 5 keywords or phrases to describe the item

SIMRANDEEP SINGH

Author(s) Name (Print)

Deep Singh

Author(s) Signature

12-18-19

Date

“A person’s most useful asset is not a head full of knowledge and pocket full of dollars....

... But a heart full of love and a hand willing to help others....”

- Simrandeep Singh

*Helping Sapiens: A Geo-based mobile
Application to help people in the vicinity*

CHAPTER 1. INTRODUCTION

A quote from Dr. Loretta Scott [29] states that “We can’t help everyone, but everyone can help someone.” Bible puts it this way “Practice giving and people will give to you with the measure that you are measuring out, they will measure out to you in return”- **Luke 6:38**.

1. Introduction

An interesting study by Felix Warneken and Michael Tomasello [44] with title "Altruistic Helping in Human Infants and Young Chimpanzees" discusses the basic nature of human beings to help each other to achieve their goals even without immediate benefits. This study show that human children as young as 18 months readily help others to achieve their goals. Helping each other is a continuous process of evaluation that helps peers to learn from each other [30]. With recent advances in technical systems, trust has emerged as important concepts in not only social science but computer science as well. Within the field of computer science, researchers use this concept to build application frameworks for recommendation and reputation based system [31]. For the working of any regulated system, appropriate components are required for helpers (help providers) and seekers (help seekers). In many instances, people can find it difficult to seek help when they need it and in other instances, people ready to provide help have no channel for them to do so. The overarching theme of this thesis is to provide a common platform for each of these cases. The proposed framework is designed to match help seekers with appropriate help providers.

With the adoption of social media applications such as Facebook, Twitter and YouTube, many research studies explore location-based social services. This thesis adopts geo-based networking to connect the users who are seeking help with those who are ready to provide help. We also use the concept of reputation and recommendation system to build trust amongst users.

The following provides a listing of some famous applications that use the concept of location based networking. Some of the concepts from these apps are also included in our project. We are mentioning the name of each application (app) here; the detail about each can be found in next chapter. Examples of these apps: Askalo, Badoo, Block Chalk, Buddy Cloud, Carticipate, Check points and Weneat.

2. Motivations

We have always been taught from our childhood time that helping people is good. It not only makes the life of the ones we help a bit easier but also gives us inner peace and satisfaction [5]. Overall et al. [45] discusses the effect on personality and social growth of a person if she helps other. That help can be for self-improvement and relationship quality. The life experiences of the author also acted as a motivating factor to work on this platform. Some of the motivational factors are listed below, some are taken from the literature [39] and some are taken from personal experience.

Helping others can help us to kill our anxiety and depression [48]: Sedentarily lifestyle has given us not only relaxed and comfortable lifestyle, but adverse effects also include anxiety, depression and other mental ill effects. Helping others gives us a positive vibe to overcome these things. *What goes around comes around [47]:* It's about karma but the point that I have

observed here is not only karma. As we help others repeatedly you will see something amazing start to happen in your life as well.

A study by Oxford and Green [46] shows the importance of help and a case study the used *Language learning histories*: Learners and teachers helping each other understand learning styles and strategies. *Meet the unexpected*: Many people I have met through the intuition of help and in unforeseen circumstances have become an important part of my life and have helped me to grow in my personal life [49]. *We rise by lifting others*: I believe that the world is one big family and we need to help each other that can help in raising others.

As a computer science student, this thesis tries to combine all these things in the form of a framework that can help the society. The next section introduces the concept of reputation and recommendation system that acts as the basic elements for building our project.

3. Reputation System

An interesting study by Thoms [50] for social interaction and learning in higher education discusses the design and implementation of a feedback mechanism for social software. A reputation system provides individuals with the ability to rate others. Apart from this, the other main module of our work is Rating system that is discussed in next section. These two components are related to each other as Rating system would provide the necessary data for Reputation system. The developed application helps in building a reputation-based relationship amongst strangers who interact with each other and during this interaction they also rate each other in an online environment. While designing the system, we need to keep in mind the biased ratings. Let it be an unfairly positive rating or unfairly negative ratings. Reputation system can

be supposed to consist of three components namely: an information gathering system, a reputation building system and a penalizing-rewarding system.

As stated in the literature, a reputation system can act as a mechanism for individuals to achieve a common task, the rating system can act as a motivator for active participation [52, 53].

Reputation serves as a symbol of status in a system for a user. Many hypotheses can be drawn based on the reputation of the user. We are making an assumption here that as the user's reputation grows it becomes a hard-earned asset to the user that he wants to protect from degrading [1].

Design of reputation system: The basic idea of Reputation system is to provide a simple way to allow individuals to give feedback to others. There are many Reputation system available online, but this thesis work is motivated from Amazon mobile app interface. Depending upon the requirements of other modules, different components were added. Similar to websites such as Amazon, ratings are on a scale of 1-5 stars. It is to be noted that 1 star, represents a low rating and 5 stars represents a high rating. Upon hovering over a star, a pop-up will indicate the number of stars out of 5 as shown in figure 1.1



Figure 1.1. Rating system star rating for Amazon

4. Recommendation System

Recommendation system consists of recommending a right entity to the user who is seeking help. Our system focuses on two things while making recommendations: a) based on preference and user reputation b) based on location of the user [4] for example AirBnB helps the other users to find homestay for vacation. A recommendation system takes Rating system as an input.

Many existing Rating systems are used for recommendation – percentage based, binary, and textual reviews based . A confidence value is always used with the rating that specifies the recommender's own confidence. A pseudo identity is added with each user for hiding the user's real world identity. Reputation and recommendation are close to each other as the reputation of the user grows with the recommendation of other users. In real life, users do not strictly act

according to a behavioral pattern. So a confidence value is used to vary the reputation value. Many online shopping websites like amazon, and flipkart uses these concepts to recommend the products to other users. Helping Sapiens will also use this type of user recommendation to help other users.

Design of Recommendations system: The proposed recommender system uses rating system to recommend a user for helping in a particular category based on the previous ratings. We have used collaborative model for the implementation. This is the most widely used recommendation system and used by most of the leading online portals like Amazon, Netflix, and YouTube .

Collaborative filtering selects the best user from the pool of help providers who have already provided help in a given category [54]. Collaborative-based recommender system is depicted in [55] also shown in figure 1.2. It shows help customers (help seekers who were served by the helpers) providing information on the application. Subsequently, computer algorithms calculate and present other help seekers with a list of recommendations based on the category of help they have provided, and also the rating given to helpers. This is done by using various backend system components such as a rating database, and a correlation database.

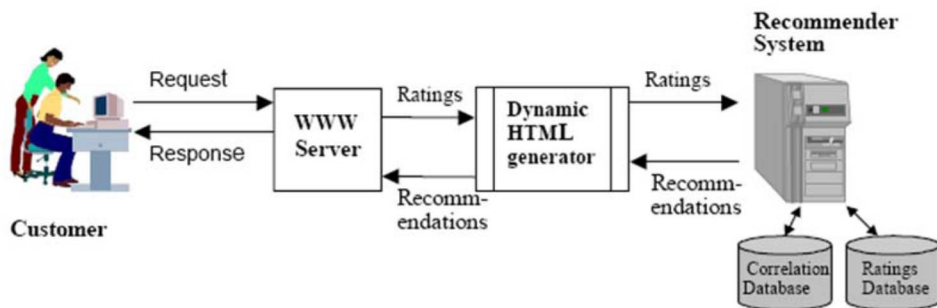


Figure 1.2. Collaborative-based Recommender System [55]

5. Model of trust

From a sociological perspective, trust can be considered as a collective unit not of isolated individuals which denotes the mutual “faithfulness” which is the base of all social relationships. Consider two independent entities, A and B. Without any prior experience between the two, there will also be no prior levels of trust between A towards B or B towards A. One might argue that this lack of experience results in a lack of trust. Consequently, different levels of trust are inherent across social constructs. Within the framework of this thesis, I will construct a set of trust categories along with a corresponding confidence vector. Trust values are taken in the range of [0,1]. While 0 indicates no previous experience or bad experience with the particular entity in the given category, 1 means highest possible trust. Initial trust value is 0 for each entity and each category.

The confidence vector is used to decide how the value of trust will vary. It contains all the meta information regarding an entity. It includes the number of experiences with any other entities in any categories [8]. A concept of sustainable online learning community based on knowledge sharing and trust is presented in [55]. They use the theories of social learning and networking to assess trust in communities in practice

6. Organization of thesis

The remaining portion of the thesis is organized in the following manner.

In chapter 2, we have summarized relevant literature and tabular comparison of existing reputation and recommendation systems. We also present the detail of each paper that is used for thesis. Chapter 3 provides a brief description of the architecture and various components that are used in the overall framework. Also, complete details of implementation procedure and working model of the proposed architecture is provided in chapter 4 which explains detail of each and every module through screenshot and flowcharts. Chapter 6 discusses the outcomes of the research work and various conclusion drawn. It also discusses the future work. We also provide a proof of concept in this chapter.

CHAPTER 2. LITERATURE REVIEW

2.1. Introduction

This chapter summarizes relevant literature. The literature has been categorized into tables and a comparative analysis of these studies is also presented. Our developed system will be

referred as “Helping Sapiens” throughout the thesis now. This title clearly depicts the working and motive of the developed application.

2.2. Reputation Systems

Reputation system have emerged as a method for simulating adherence to electronic contracts and for fostering trust amongst strangers in online transactions [40]. More details about the reputation system can be found on [41] which discusses various design issues for P2P networks.

Lappas, T. *et al.* [1] proposes a theoretical model to study the association between the reputation of a user with his ability of being a risk taker (answering difficult questions), his performance (measured through quality of his answers) and topics of questions which he chooses to answer. These associations are studied through StackOverflow.com dataset. This study overcomes the existing shortcoming i.e. user’s reputation only depends upon the volume of responses submitted by the user. The limitation of the study is, it is difficult to have the user who take risks i.e. submit responses of difficult questions. The various shortcoming acted as the problems I might get during the implementation phase of our study. From this study various general questions related to implementation were answered automatically.

Howley, Iris, *et al.* [2] Investigates the effect of various reputation system features like up or downvoting, badges (rewards credited to students) and helper expertise on student help selection in MOOC discussions. It uses Quick Helper application to connect the Help seekers to the help providers involved in discussions and Expectancy value Theory to find out whether help source provides expected help. The experiment results show that upvoting or downvoting has negative

effect on student help seeking in MOOC discussions, which can be eliminated through badges. It uses specific selected features of Quick Helper system, which may not be applicable outside proposed helper system. The basic architecture of this study is used in our work as well. In this study student was interacting with the MOOC courses for help seeking while in my system seekers will be connected with help providers or helpers.

Dijkmans *et al.* [3] examined whether corporate reputation is achieved through online activities of the company, to acquire engaged customers. International airline consumer's data is used to perform the experiment. The experiment results show, consumer's social media usage leads to his engagement in airlines social media activities, which in turn leads to corporate reputation. The part of this study related to reputation was used in my thesis work.

Whitby, A. *et al.* [9] proposed a statistical filtering mechanism to remove the unfair ratings, which can affect the reputation system. The proposed mechanism is applicable for both unfair positive and unfair negative ratings given by a rater in Bayesian reputation system. It assumes whenever a rater changes his ratings, it can affect the ratings of the other raters who communicate with him. Filtering effectiveness is measured by two parameters – Proportion of unfair buyers (should be low) and probability that unfair buyers would rate unfairly (should be low). The proposed method works well when less than 30% raters rate unfairly. The limitation is, to satisfy the requirement (unfair raters rate unfairly in consistent manner) in real systems. The work of filtering to remove unfair rating is a concept that comes under the future area of my thesis but some basic concept like rating number has been used in my work as for now.

An interesting patent [11] proposes a personal reputation system that depends upon user's behavior and his activities and transfer of data in one or more networks. Each user in a social network has a personal profile, which contains personal reputation index to represent the reputation of that user and personal tags associated with each profile. These personal tags are voted to measure their accuracy. The profile management flows of our work is motivated from this patent work.

Tang et al. [27] explained the importance of a business's reputations in social media, the way the measurement of reputation system is done, and the way the reputations are valued by the contributors. Based on contributor's decisions for content-contributions, they have developed a dynamic-structural model to recognize utility function of content contributor. Contributor's desire for reputation drives the social media's contribution to content. Dynamic, forward-looking decision making can help in explaining the content contribution better. The proposed dynamic structure model provides the benefit of the evaluation of parameter as well as a structured framework for decision making. It also states that due to competition among the contributors, it is feasible to measure reputation in relative performance terms than in actual achievement.

If one contributor fetches more subscribers (not the viewers) than other, then his reputation increases. Hence, a contributor with high subscriber rating has higher reputation than the viewer rating. Even if a web site does not offer sharing of revenue, still it is attracted by contributors seeking for platform for reputation and recognition. Comparing the contributors on the basis of interests of content and popularity can help in increasing the reputation.

2.3. Location-based Social Media

The location based social networks can shorten the distance between people, make new way of the communication, enhance communication between people and change the way people live. With the development of internet and smartphone era, people can communicate with each other at any time and place [5]. Helping sapiens also used this location based social media concept for interaction between the users who are present in the vicinity of each other. Helpers and seekers need to be close to each other so that they can interact with each other online as well as offline.

Bao J. *et al.* [4] proposes a location and preference based recommender system to provide set of venues to a user in local range. It considers user preferences as well as social opinions for recommendation. To perform the experiment, tips from various users from Foursquare, is used as. It considers five data structures- user, check-in, venue, category hierarchy and user location history. Comparison is performed among four recommender systems – Most preferred category based, Location-based collaborative filtering, Preference-based collaborative filtering, and proposed approach, proposed method performs better. The proposed method overcomes data sparsity problem (when user visits only limited number of places).

Briones *et al.* [5] explained the role of social media for communication among people. In this paper, Facebook and Twitter are mainly used for communication by the American Red Cross organization. Two-way communication is used, so that quality of communication can be improved through user's feedback. As in this paper, American Red Cross organization is used, this study becomes specific to this organization and cannot be generalized.

Assuncao *et al.* [6] proposed a method to evaluate the reputation of a social networking system's user. It analyses the attributes related to the profile of social network user, and based on that analysis rank or rate is assigned to a particular reputation category. Social networking systems are used to create teams, knowledge sharing and analyzing complex networks. The major problem faced in reputation systems is accuracy and validity of information. This paper provides methods to measure the reputation so that trusted and accurate reputation system can be developed. Network Measure of Influence (ratio of influential people that have influential network) is used for social network analysis, person's network authentication score to authenticate the person and Network Diversity profiles and metrics to authenticate person's network.

Doytsher *et al.* [7] proposed a framework socio-spatial network algebra (SSNA), that uses socio-spatial graph concept in which users are connected to geographical entities through life-pattern (association of users to places, which they frequently visit) edges. It combines social information with spatial information. To store the data, relational database system and graph database system are used and comparison is performed among these two implementations. It also presents various operators to form the queries, to extract the integrated data.

Saarinen *et al.* [23] have created a social networking system known as TWIN which exploits the connectivity between people. Depending on the location of the person, the TWIN system creates a locally present people's community. Like recently used social networking applications, TWIN can also be used to share files, do private and public chats, share experiences, photos, videos and audios, managing old friends and creating new ones within the locally present people.

The TWIN system executes on ad-hoc WLAN network (hence no infrastructure needs to be developed). The TWIN system was first implemented on Nokia N900 device. It is purely based on Peer-to-peer (P2P) networking. The support of multi-hop transfer of data provides facilities such as management of community, chatting and data transfer, and file transfer of more capacity than single-hop data transfer.

The system is linux based and written in Python language. The experimentation process was named as pilot and was tested with users around the TUT university. Somehow, the system didn't prove to be reliable in some cases. The experiments indicated that the TWIN system did not prove to be better than the existing systems but rather complements them.

2.3. Trust in Reputation Systems

Trust is defined in [43] as “the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party.”

Bhuiyan, T. *et al.* [8] proposed a model, which combines the trust among various users and reputation of items, for recommending the items to network user. It mainly focuses on major issue i.e. to maintain good relationship among users of a network and quality of information shared among these users, so that trust can be maintained. It assigns reputation scores to each item, based on these scores items are recommended to the network user. The same reputation based system will be used in our framework so that we can find user who is most suitable to provide help to a user.

Resnick et al. [15] described the importance of reputation systems. Reputation systems can act as a trust building tool between the strangers. Many websites such as e-bay, askme.com, epinions.com, etc. attribute their success to allowing consumers to rate products. Using the example of long-term relationships, authors explain the reason behind the reputation system's role in building the trust among the strangers. Reputation systems depend highly on the past experiences of the user. Reputation system helps in getting good prices to high quality sellers whereas, low prices to low quality sellers. In order to depict the quality, ratings and using user's actual name (rather than the pseudo names) prove to be highly beneficial.

Thoms et al. [42] discuss a trust based knowledge sharing (KS) system. They use theories of social networking and social learning to guide the process. A case study of graduate students to assess their perceptions of LS and trust in communities of practice is presented. **Kinater and Pearson** [10] propose a distributed reputation system that has improved privacy and security. It uses trust mechanism to form and gather the sensitive information. Privacy and Security is achieved through Trusted Computing Platform Alliance (TCPA) technology. The proposed model consists of three entities- recommender, requester and accumulator. An entity seeking help about a target uses agent on its site to form the query, this agent returns the locally available set of recommendations. The weights are assigned to these recommendations based on the reputation of recommender and suitable recommendations are presented to the entity (that seeks help). Same set of entities is used in helping sapiens as well where two main entities- help seeker and help providers are presented.

Angermeyer *et al.* [12] propose a help seeking system for psychiatric disorders. It uses ranking and rating approaches to know the sentiments of people about mental disorders. This paper concludes human attitude and belief system have great impact on help seeking process.

Terveen *et al.* [13] use People Helping One another Know Stuff (PHOAKS), which uses collaborative filtering (recommending resources to one another) to identify the appropriate recommendations of web resources. The users help each other based on the previous comments from other users. These recommendations are filtered from Usenet messages (source of web resources recommendations). PHOAKS has two features- role specialization and reuse (uses information from existing online discussions). Role specialization deals with how a single user can have different roles in a single system. PHOAKS system works in three phases – Search, Categorization and Disposition. The number of distinct recommenders of a resource act as a measure of resource quality. The paper suggests, generic filtering can be applied to extract the information from electronic messages, which can be used in intranet and education applications.

Lewis, J.D. *et al.* [14] proposes social concepts of trust, to bring social concepts to bear psychological and political studies of trust and review the studies which explain social nature of trust. In this paper, three dimensions are defined to describe social nature of trust – Cognitive, Behavioural and emotional.

2.4. Recommendation System

In order to run effectively, the following properties need to be incorporated by a reputation system [15]:

- Entities should be long-lived so that an inspiration for expectation of further interaction can be developed. The users who are dealing with the system should use it quite often so that other users have an enhanced trust in them.
- Feedbacks about ongoing interactions must be captured and spread. As helping sapiens will also be using feedback from other users as a primary working module, so it is required that all the feedbacks are genuine and visible to others.
- Feedback be used in order to provide guidance to trust decisions. Helping sapiens will use the feedback to recommend help providers to help seekers depending upon what they have done in past. This type of model will be beneficial for building the trust amongst the users.

The main operating phases of reputation systems are eliciting, distributing, and aggregating feedback. Each phase has some challenges associated with it.

Scellato *et al.* [22] explained six techniques which can be used by collaborative filtering recommender to know about new users. Collaborative filtering is a technique used in recommendation to allow users to give ratings and feedbacks for the items/products used by them already. They defined the new user problem for the recommender system as when system knows nothing about the new user. The system solves it by gaining some knowledge about the user by asking for rating some products. Determining what to ask to a new user so that it does not seem boring to him is the main task of the recommender system. Following techniques have been explained in the paper for presenting products to the new users

- Random strategies (Choosing items randomly)

- Popularity (Presenting in descending rating order)
- Pure entropy (Equality in hatred and liking of the product)
- Balanced strategies (Obtain maximum information about a product)
- Personalized (using item-item personalization)
- Asking questions based on attributes.

Jensen *et al.* [17] explain different types of reputation systems, which are grouped as ranking systems, rating systems and collaborative filtering systems based on generation of rating and information's nature. It also explains two additional types of reputation systems implicit peer-based systems (user's friend's data for recommendations) and explicit peer-based systems (ratings are generated by the assessment made by the user chosen friends group). Different types of experiments have been detailed and conducted on different types of reputation systems in order to analyze which is more liked by the users.

It was hypothesized that the personally relevant and peer based systems are more preferred by the users in social environments, whereas rest are preferred in less social tasks. It suggests that studying the way the systems address users' needs and the way users value and use reputation information can be helpful for future work.

McDonald [18] explains the importance of social networks for building reputation systems. Social networks can be visualized as a tool to see group behaviors and links between the groups. The social network of MSC taken into example has been visualized in two graphs: The Work Group Graph (WGG), and SPS Social Network (Sociability of Individuals). It is found that a

large overlap exists between SPS and WGG networks. It explains the technique known as Expertise Recommender to find the expert of a particular domain which can help in building a better recommendation system.

Users of the reputation system generally spot a trade-off between finding a knowledgeable person and an interactive person. User's expectation for reputation system to naturally expand and assist, instead of replacing; their attitude to not accept the network depicted by the system as they are well versed with their social network; and dynamism in the social networks of groups can be a hurdle to the system.

Marti *et al.* [19] identify the three basic components of the reputation system i.e. Information Gathering, Scoring and Ranking, and Response. In order to explain the peer-to-peer reputation systems various terms such as Transactions (interactions between parties), Cooperate/Defect (transactions carried correctly by decent parties), Structured and Unstructured (categories of P2P architectures), Strangers (new parties to the system), Adversary (harmful peers) were introduced. The information gathering systems must have some of the following properties in order to gain information about a trust worthy user:

- Adversary
- Spoof-resistant
- Unforgeable

The information collection source, information agents quantity and the way information is combined by different sources by the parties has been briefly described. A general reputation

score function is calculated to assist the agent to decide the service provider with whom it should transact. The system can influence the parties to contribute to the network or punish the harmful peers by providing incentives or punishment respectively.

Chen *et al.* [20] describe four algorithms i.e. Content Matching Algorithm, Content-plus-Link (CplusL) Algorithm, Friend-of-Friend (FoF) algorithm and SONAR algorithm. The proposed algorithms proved to be effective in making recommendations for people and increasing subsequently the number of friends of a user on a site. These four algorithms can be categorized into two categories:

- Based on social relationship information (SONAR and FoF)
- Based on content similarity (Content and CplusL)

Somehow, it has been found that the relationship categories proved to be better in terms of user response than that of content similarity. It has also been observed that content similarity algorithms were better in finding new friends whereas relationship-based algorithms are better at finding known contacts. It also suggests a better approach to combine the two categories by initially taking hold of relationship based algorithms to build up a network as soon as possible by finding known people offline and, as the network expands, accompany them with content similarity based algorithms.

Scellato *et al.* [22] presents a novel approach for geographic network analysis and defines metrics for deciding if an individual has short range or long range social bonds. It explains the geographic social network as a graph with nodes connected through the links, where the nodes

represent the users and the links represents whether there is a connection between the users or not (can be directed or undirected).

Jamali *et al.* [23] describe the importance of Recommender systems due to large growing amount of information available on WWW as a tool that helps the users to easily select relevant information online. Due to the great advancement in social networks, recommendation systems based on social networks approach have emerged. A model-based approach for a recommendation in social networks has been explained that employs the various matrix factorization techniques. Here cold users mean those who have expressed only a few ratings for other users. A method of collaborative filtering is not effective for cold users while the new approach of social networks to recommendations seems to be very useful for such kind of users. A model of trust propagation has also been incorporated into the current model which seems to be very significant in increasing the accuracy in recommendation systems, especially for cold start users. The whole study is carried to two real life data sets from Epinions and Flixster. Flixster is a social networking service in which a user can rate movie and create a social network.

Zhang *et al.* [26] went in great depth to show geo-social influence on people's real life. Today Geo location services are widespread; depending on the location of a person the event at user level can be quantified. LBSN (Location Based Social Network) does allow a person to maintain cyber link between a mutual friends but also allow a person to share content among their mutual friend. This geolocation correlation among the person and their mutual gives intuition how might people react to their surroundings. The situation can be leveraged just by

using targeted camping, advertisement and viral marketing to name a few. Geo social based applications are gaining popularity as the targeted user can be found easily.

Above framework uses penalized hitting time (PHT) to figure out the social proximity between a User and his mutual. Penalized hitting time (PHT) consists of two, one is hitting time and other is hitting time with penalty. GEO - Location of a user and his mutual are drawn on a geo map hereinafter referred as node and the proximity between nodes are called hitting time. But there are some drawbacks, so they tend to lean towards a more mature version of its which has a nice property that the path weight is exponentially penalized by nodes proximity.

Atele-Williams *et al.* [28] due to short term interactions between people, lays emphasis on the improvement of reputation data based recommendation system. It provides a systematic literature review on the already existing methods for reputation based recommendation system. It describes various recommendation system principles, reputation system principles, and how the reputation is related to recommendation system. Using a hybridization method, they have categorised approaches into 6 different categories:

- Weighted (Outputs combined on the basis of weighting factor)
- Switching (Use of reputation system when suggestions are not available)
- Rec-rep-cascade (Output refined by reputation system)
- Mixed (Combining both outputs of recommendation and reputation system)
- Rep-rec-cascade (Input of recommendation system first filtered by reputation system)

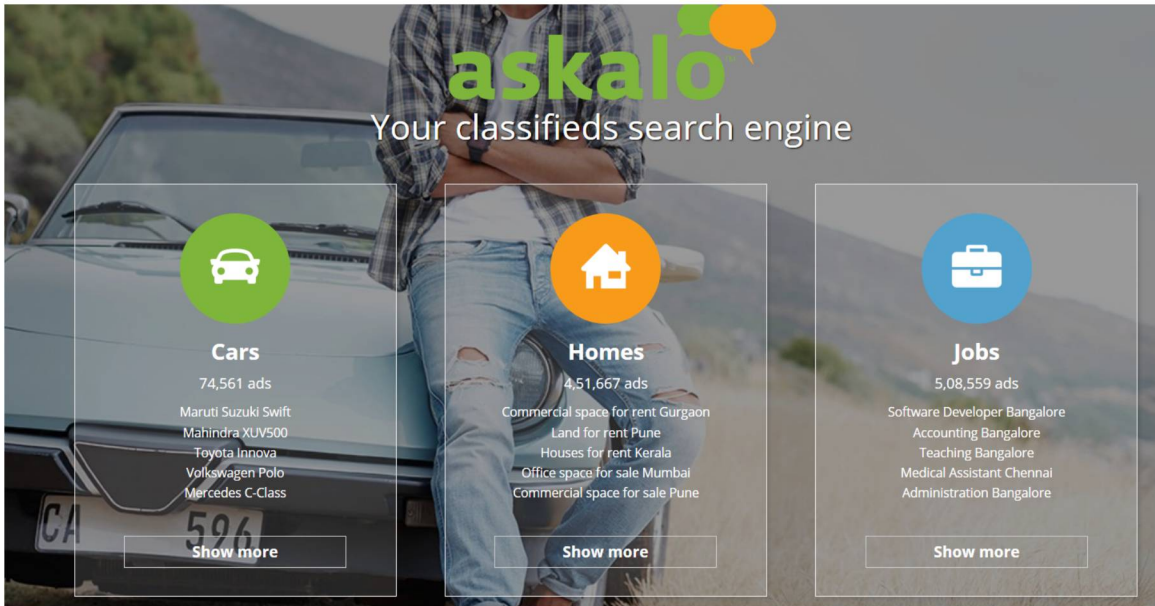
- Augmentation (Data of reputation system used directly in the recommender system calculations)

Further on the basis of the recommendation system and reputation system's connection, databases are categorized into two categories: IR (for feedback related to item), and UR (feedback related to user). The enhanced reputation based recommendation system has been compared with already existing recommendation approaches. It states that the contrary of their approach i.e. the way in which recommendation system can be used to improve the rating system, can be reviewed.

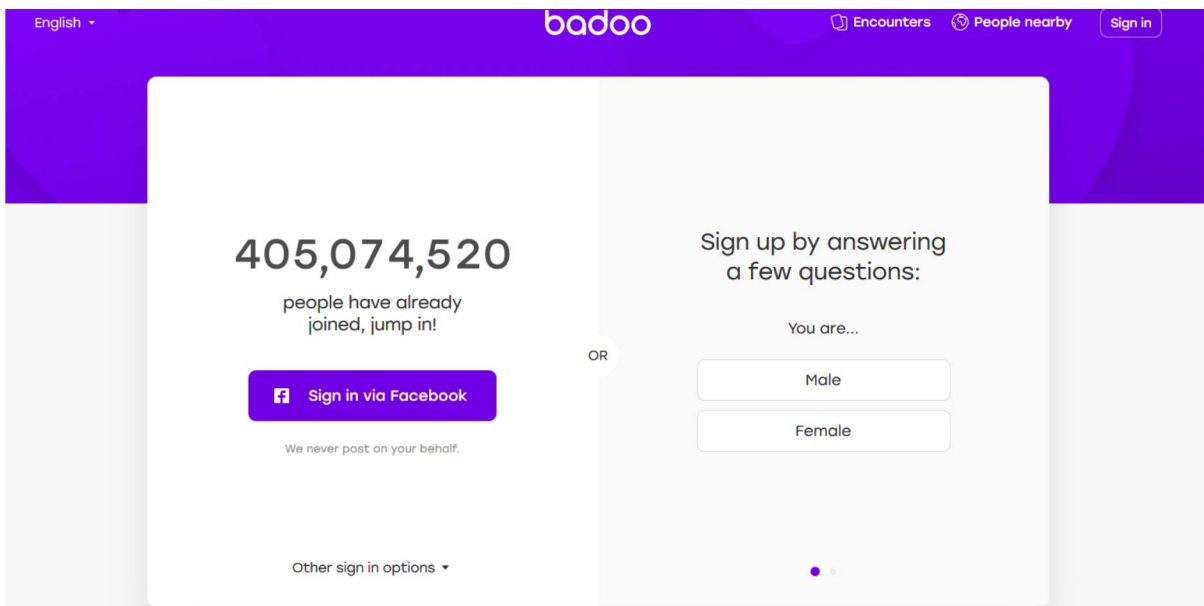
2.5. Existing Apps that motivated our work

The following provides a listing of some famous applications that use the concept of location based networking. Some of the concepts from these apps are also included in our project.

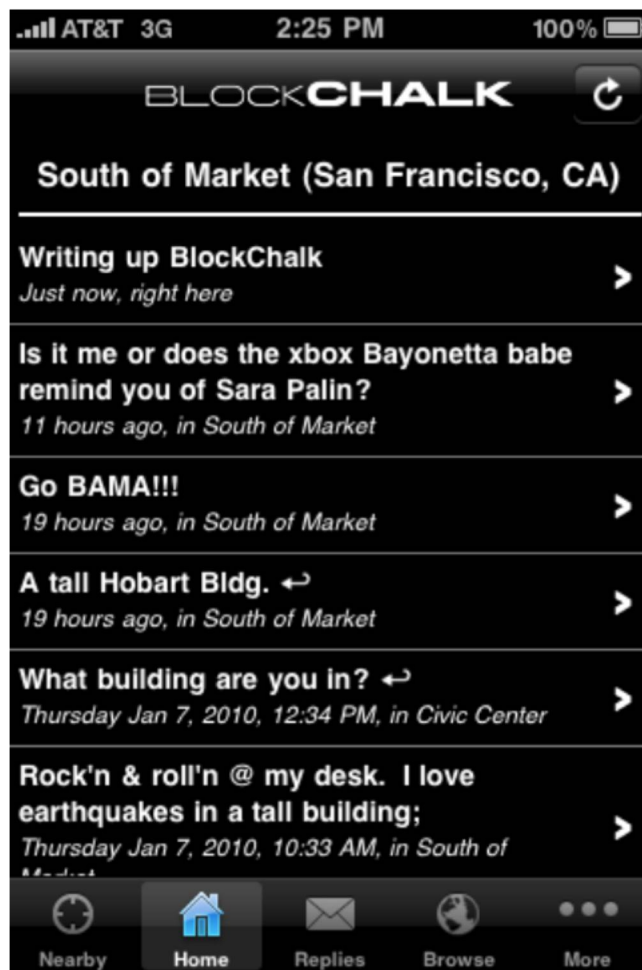
1. **Askalo- *your classifieds search engine*** [32]: This is a famous website to discover your city in USA. This app helps the user to discover the city of interest. A user can also share her experiences and what's hot in the city. Motivated by their location-based service, we add geo-based feature to our proposed app. Google API is used by them.



2. **Badoo- Meet new people! [33]:** This website is for chatting, flirting, socializing and to have fun with the new people around. It gives the user an option to choose the radius in which she wants to find any other person to hangout with. Our developed framework also uses the concept of distance to find the helper in a region.

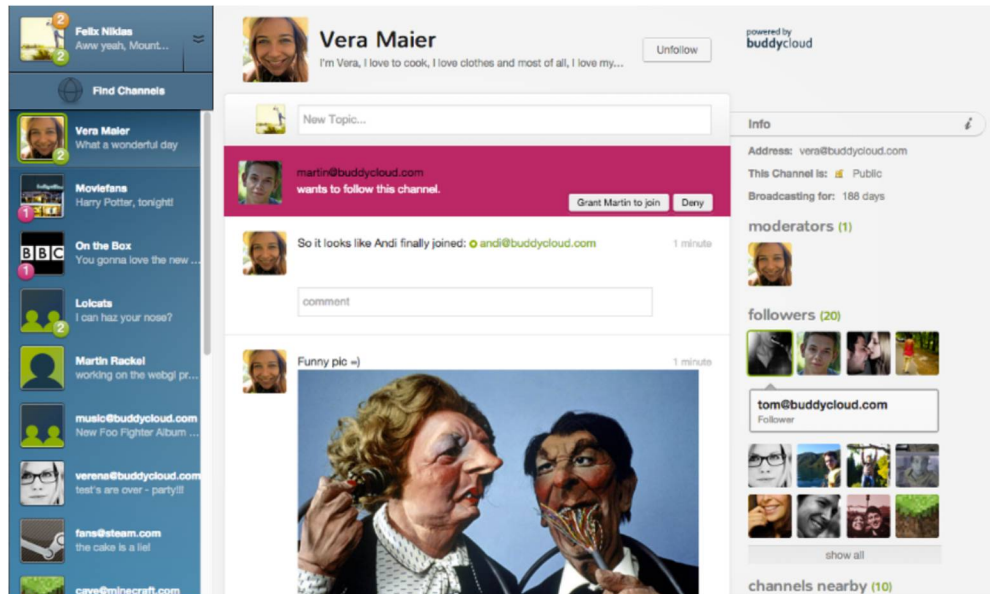


3. **Block Chalk- *Location based neighborhood interaction app* [34]:** This is a famous app for mobilizing your neighborhood. This app lets you leave a message in your neighborhood and see what your neighbors are saying. People use it to ask, answer, praise, borrow, sell and much more. It is easy and free and you can use it without sign up. Our framework is inspired by this app also, as we also let the people in vicinity of each other.

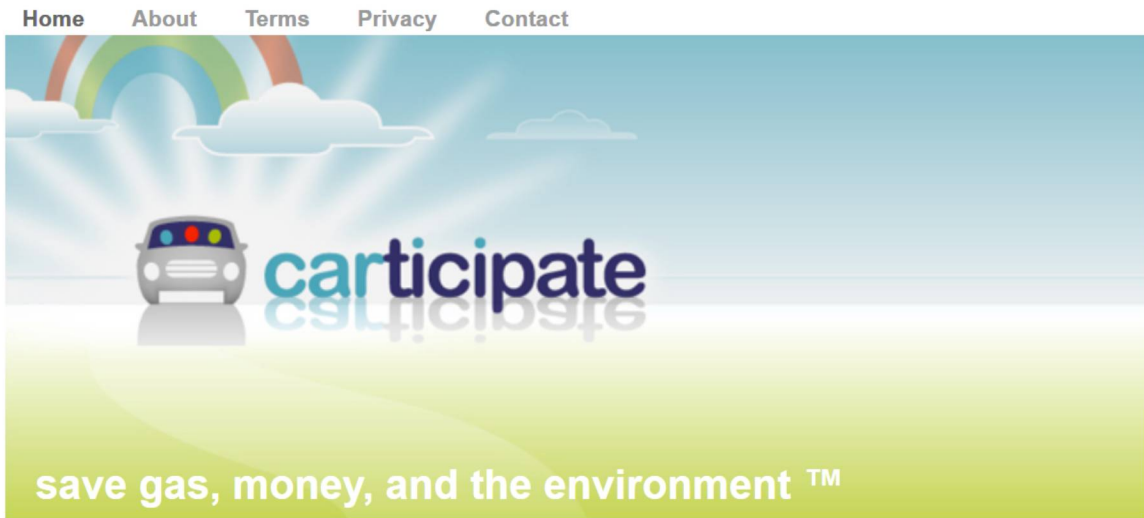


4. **BuddyCloud- *A location based social network* [35]:** It's a famous mobile phone app that uses social location based platform. It lets your friends know what you are doing and

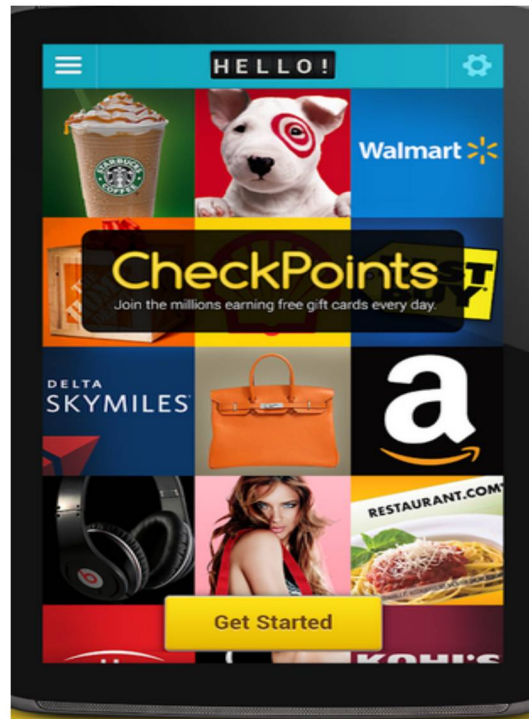
where you are. It adds context to the places you visit in real world and makes it fun and easy to share these on other websites like twitter and Facebook. It also allows you to share short status messages on twitter. We can also attach our framework with other social network sites in the future.



5. **Carticipate- A location based social transportation** [36]: It is an experiment in social transportation to share rides based on location aware mobile platform. It states to help the environment by saving fuels. It uses coordinates of the drivers ride to indicate where and when she is going. It will match her local carticipants going in the way. In our framework we use coordinates of the user to find the help provider who is within the preferred radius to provide the help.



6. **CheckPoints- Location based loyalty programs [37]:** This website lets you check out the products over a million stores. If you are going out for grocery shopping, book store . She can collect points for every purchase and can redeem them later.



7. **Wenear- Location based social network framework [38]:** It's a location aware mobile application that will keep the user connected to your favorite people around you. This app delivers your messages to nearby people and lets you grab the opportunities present near you.



The basic concept of these apps is same as they all used location based networking. We also used the same concept and added our concept of providing and taking help from others.

The abuses of the proposed system are discussed in [9] which deals with unfair ratings. As these ratings act as input to the reputation system, false rating may adversely affect the reputation as well as recommendation. In helping sapiens work we have not dealt with any such abuses but as a future work this proposed system can be improved to handle false rating and users.

Some of the papers that motivated my work are summarized in below tables i.e. table 2.1 and 2.2. Table 2.1 compares the various studies on different parameters like framework used, domain and framework, data set used, and evaluation measure. We also discuss key features and shortcomings of each study in detail. In table 2.2 various features including whether it has recommender, reputation and rating system are presented. We also pointed out whether the study has used social linkage and Geo based linkage.

All the paper that are discussed in chapter 2 are presented in tabular form in table 2.1.

Table 2.1 summarizes the key findings of the papers in one place so that a reader can get work from a single location. The table 2.1 also present the future directions of research which can help the researchers in finding the area in which they can work.

The reader can also get an idea of the framework or model that can be used in a particular scenario. Further the domain in which the study has focused is also presented in the table. If a researcher wants to work in the same domain as that of the study showed, the dataset that may be used for the work is also presented in table 2.1. The various metrics of evaluation are also given in the table that may help the user in choosing the right evaluation measure to compare her work with the actual work.

In table 2.2 the above-selected studies are compared based on the various common features. There can be other features, but only those were chosen that are related to our proposed work. These features set includes following:

Recommender: If the study is related to the recommender system defined above, we have pasted yes in the column otherwise no.

Reputation: In the similar way if the selected study is related to reputation-based system yes is present in the column.

Rating system: If the study includes the concept of rating-based system for other used a yes is there in the column otherwise no.

Social linkage: If there is a social linkage between different nodes or users corresponding yes is there otherwise no.

Geo Based: If the study uses any geo based social networking, we used yes in the column otherwise no.

Table 2.1. Comparative analysis of various studies

Reference	Title of paper	Key Findings	Shortcomings/ Future work	Framework/ Model	Domain/ Basis for example	Dataset	Evaluation measure
[1].	Reputation and Contribution In Online Question-Answering Communities	Proposes a theoretical model to study the association between reputation of a user with his ability of being a risk taker (answering difficult questions), his performance (measured through quality of his answers) and topics of questions which he chooses to answer.	Difficult to have users that take risks i.e. answer difficult questions.	NA	Online Question Answering communities	Stack Overflow dataset	Reputation, user's ability, and Risk taking
[2].	Reputation Systems' Impact on Help Seeking in MOOC Discussion Forums	Investigates effect of various reputation system features like up or downvoting, badges and helper expertise on student help selection in MOOC discussions.	Uses specific selected features of Quick Helper system Quick helper system did not address whether help seekers get desired help.	NA	MOOC discussion forum	NA	NA
[3].	A stage to engage: Social media use and corporate reputation	Examines whether corporate reputation is achieved through online activities of the company, to acquire engaged customers.	Study uses KLM airlines company which is more engaged in social media, therefore may attract customers due to its popularity. It uses basic concept of engagement i.e. only being familiar with company's social media activities.	NA	Travel or Tourism	KLM Royal Dutch Airlines consumer's data	Corporate reputation, intensity of social media use, and engagement in social media activities of company
[4].	Location-based and Preference-Aware Recommendation Using Sparse Geo-Social Networking Data	Proposes a location and preference based recommender system to provide set of venues to a user.	Temporal and weather features can be considered in recommendation system.	NA	Restaurants	Tips from users in restaurants	Recommendation effectiveness and recommendation efficiency
[5].	Keeping up with the digital age: How the American Red Cross uses social media to build relationships	Explains the role and importance of social media for communication among people.	In this paper, American Red Cross organization is used, this study becomes specific to this organization and cannot be generalized.	NA	NA	NA	NA

[6].	Method and system for reputation evaluation of online users in a social networking scheme	Proposes a method to evaluate the reputation of a social networking system's user.		NA	NA	NA	Network Measure of Influence , person's network authentication score and Network Diversity profiles and metrics
[7].	Querying Geo-social Data by Bridging Spatial Networks and Social Networks	Proposes a framework, which uses socio-spatial graph concept in which users are connected to geographical entities through life-pattern edges.	Aggregation functions can be introduced to form the query	NA	NA	Real data of Haifa city	Evaluation time
[8].	Integrating Trust with Public Reputation in Location-based Social Networks For Recommendation Making	Proposes a model, which combines the trust among various users and reputation of items, for recommending the items to network user.	Requires a method to prove that proposed method shows improvement in terms of item selection.	NA	NA	NA	NA
[9].	Filtering Out Unfair Ratings in Bayesian Reputation Systems	Proposes a statistical filtering mechanism to remove the unfair ratings, which can affect the reputation system.	Difficult to satisfy the requirement (unfair raters rate unfairly in consistent manner) in real systems	NA	Market simulation (Sellers and Buyers)	NA	Reputation score
[10].	A Privacy-Enhanced Peer-to-Peer Reputation System	Proposes a reputation system that has improved privacy and security. It also considers trust mechanism to collect and from sensitive recommendations.		Trusted Computing Platform Alliance (TCPA) technology	NA	NA	Trust values and Confidence vector
[11].	Personal Reputation System based on Social networking	Proposes a personal reputation system which depends upon user's behaviour, his activities and transfer of data in one or more networks.		NA	NA	NA	NA

[12].	Whom to ask for help in case of a mental disorder? Preferences of the lay public	Proposes a help seeking system for psychiatric disorders.	Very less efforts are made to discuss mental health issues in public	NA	Medical	NA	NA
[13].	PHOAKS: a system for sharing recommendations	Uses People Helping one another Know Stuff (PHOAKS), which uses collaborative filtering to identify the appropriate recommendations of web resources.	Generic filtering can be applied to extract the information from electronic messages, which can be used in intranet and education applications.	PHOAKS	Newsgroup data and FAQ content	Usenet news messages	Number of distinct recommenders
[14].	Trust as a Social Reality	Proposes social concepts of trust, to bring social concepts to bear psychological and political studies of trust and review the studies which explains social nature of trust.		NA	NA	NA	NA
[15].	Reputation Systems	Details about the reputation system, its major properties, how trust can be established between strangers and importance of feedback.	NA	NA	eBay, www.askme.com, www.epinions.com, iExchange.com	NA	NA
[16].	Getting to Know You: Learning New User Preferences in Recommender Systems	Explains six techniques which can be used by collaborative filtering recommender to know about new users.	A more in-depth analysis of the system's needs for diverse ratings across all items needs to be done.	NA	MovieLens movie recommender	7.5 million-rating MovieLens dataset	Mean Absolute Error (<1.2), User Effort, Accuracy
[17].	Finding others online: reputation systems for social online spaces	Explains different types of reputation systems, grouped as ranking systems, rating systems and collaborative filtering systems based on generation of rating and information's nature.	Studying the way the systems address users' needs and the way users value and use reputation information can be helpful for future work.	NA	Online Lab	Online Lab	Standard Deviation (SD) of importance

[18].	Recommending Collaboration with Social Networks: A Comparative Evaluation	Explains the importance of social networks for building reputation systems. Also explains Expertise Recommender System.	The application of social networks by implementers and groupware designers is yet to be completed.	Expertise Locating System (ER)	NetScan Project, Conversation map, Referral Web	Medical Software Company (MSC)	NA
[19].	Taxonomy of Trust: Categorizing P2P Reputation Systems	Identifies the three basic components of the reputation system i.e. Information Gathering, Scoring and Ranking, and Response.	Developing a reliable, functional and trusted system in an anonymous peer-to-peer network.	P2P reputation system	Peers research group, Stanford	NA	NA
[21].	Make new friends, but keep the old	Describes four algorithms for making recommendations for people and increasing subsequently the number of friends of a user on a site	Develop better recommendation system algorithms and addressing the social networking issues.	People recommender system for Beehive solutions.	NA	Beehive	Connection action rate, Post-hoc comparison
[22].	Distance Matters: Geo-social Metrics for Online Social Networks	Explains how geo-spatial properties can help us to understand the ties among people and their preferences which further benefit many large-scale product recommendation systems.	The approach is still naive. The dataset can be expanded, and the features can be clustered using many advance techniques like k-means, Euclidean distance to name a few.	NA	BrightKite, FourSquare, and Twitter	BrightKite, FourSquare, LiveJournal, Twitter	Complementary Cumulative Distribution function and Cumulative Distribution Function.
[23].	A matrix factorization technique with trust propagation for recommendation in social networks	A model-based approach for recommendation in social networks is used that involves the employment of matrix factorization techniques	Can be extended to handle negative trust relations, Automatic Tuning of λT can be investigated. Cold start items can also be addressed	SOCIALMF Model	Epinons.com and Flixster.com	Epinons and Flixster datasets	RMSE
[24].	User Experience of Social Ad Hoc Networking: Findings from a Large-Scale Field Trial of TWIN	Explains a social networking system known as TWIN which exploits the connectivity between people. Depending on the location of the person, the TWIN system creates a locally present people's community.	The multi-hop system did not prove to be 100% accurate.	TWIN	NA	Pilot system around TUT campus	Usefulness, fun and Delightfulness

[25].	Teaching, Learning, and Sharing : How Today's Higher Education Faculty Use Social Media	Describes the impact of social media sites on personal, professional, and instructional use by higher education faculty Members have been examined. Also their levels of awareness are also probed.	It can be carried further to explore more about Higher education's ability to take advantage of social media for promoting professional development, broadening institutional reach, and increasing student's success.	A multiple-stage Selection Process.	Facebook, Twitter, Myspace, LinkedIn, SlideShare, and Flickr	Market Data Retrieval.	NA
[26].	Evaluating Geo-Social Influence in Location-Based Social Network	Explains how location Based social networking can influence people's behaviour and affect their life	Power law distribution is powerful metric, still other metrics are yet to be explored which could be very interesting.	PHT(penalized hitting time)	Livejournal Active user base	LiveJournal	Convergence and Scalability
[27].	Content Contribution for Revenue Sharing and Reputation in Social Media: A Dynamic Structural Model	Explains the importance of a business's reputations in social media, the way the measurement of reputation system is done, and the way the reputations are valued by the contributors.	The contribution decision can be complex in reality and is explained as binary one (Contributing or not).	Dynamic structure model	YouTube	YouTube	NA

Table 2.2. Feature based comparison of studies

	FEATURES				
Referenc e	Recommender	Reputatio n	Rating System	Social Linkage	Geo Based
[1]	No	Yes	Yes	Yes	No
[2]	Yes	Yes	No	Yes	No
[3]	No	Yes	No	Yes	No
[4]	Yes	No	No	Yes	Yes
[5]	No	No	No	Yes	No
[6]	No	Yes	No	Yes	No
[7]	No	No	No	Yes	Yes
[8]	Yes	Yes	No	Yes	Yes
[9]	No	Yes	Yes	Yes	No
[10]	Yes	Yes	Yes	No	No
[11]	No	Yes	No	Yes	No
[12]	Yes	No	Yes	Yes	No
[13]	Yes	No	No	Yes	No
[14]	No	No	No	No	No
[15]	No	Yes	Yes	Yes	No
[16]	Yes	No	Yes	No	No
[17]	Yes	Yes	Yes	Yes	No
[18]	Yes	Yes	No	Yes	No
[19]	No	Yes	No	Yes	No
[21]	Yes	No	Yes	Yes	No
[22]	Yes	No	No	Yes	Yes
[23]	Yes	No	Yes	Yes	No
[24]	Yes	No	No	Yes	Yes
[25]	No	No	No	Yes	No

[26]	Yes	No	No	Yes	Yes
[27]	No	Yes	Yes	Yes	Yes
[28]	Yes	Yes	Yes	Yes	No

Table 2.2 can help the researchers to determine the features may coexist in a study. Also the correlation between these features can be determined by referring to the corresponding study in detail. As it is clear from the table that the feature of social linkage is prominent in most of the studies. Further very few studies use Geo based feature but in their future aspect all the studies suggested to include this feature. We are also using the feature of Geo location to match help seeker and providers.

CHAPTER 3. PROBLEM FORMULATION

This chapter discusses the research gaps identified in the literature, states the problem and objectives of the proposed work.

3.1. Research Gaps

After a detailed analysis of literature review as shown in chapter 2, various research gaps were found. These research gaps can be inferred directly from table 2.1 and table 2.2. Work done by various researchers motivated me to extend that work in this shape. These research gaps act as open issues for our problem. More research gaps can be inferred from the literature review chapter, but we are only mentioning those which are related to our work.

- There is no application or framework available that can connect help seeker user with help providers in the vicinity of each other.
- Geo-based tagging is used extensively in social media, but it is never used in such applications.
- It is challenging for people who are seeking help to connect with those who are ready to help and vice-versa.
- There are many reputations and recommendation-based systems available in literature but none of them is used in our proposed scenario.

3.2. Problem Statement

The problem statement for this thesis is to design and develop a recommendation-based framework for connecting people seeking help with those who are ready to help. The developed framework is titled, Helping Sapiens, which acts to serve as a platform for connecting specific types of users.

3.3. Research Objectives

Helping Sapiens is built considering the basic idea of human nature i.e. Helping each other.

The main objectives of my work can be summarized as:

- Investigate the body of literature on rating-based frameworks.
- Design and develop a Geo-based application framework for connecting helpers and seekers based on their location.
- To introduce the concept of rating in the above-developed framework.
- To validate the proposed work in real-world scenario.

CHAPTER 4. PROPOSED FRAMEWORK DESIGN

This chapter presents the design proposed for Helping Sapiens. These diagrams aim to depict user requirements, technical requirements and the overall architecture for the system. This chapter discusses the system design for reputation and recommendation system as well. We used android studio to implement the Helping Sapiens. This chapter also presents some of the basics diagrams that are used for the development of system. Entity-Relationship diagrams present two main entities help seekers and helpers along with various attributes. The class diagram is included that presents the main classes that are used for the implementation of the proposed work. In workflow section, various workflows like login workflow, home screen workflow, workflow for request user, user profile workflow and helper user workflow are presented.

4.1. Entity-Relationship Diagram

An Entity relationship diagram represents various entities and their attributes. Various types of attributes like multivalued, prime attributes are also presented here. As entities interact with each other, the interaction is shown by the relationship in ER diagrams. Figure 4.1 represents the basic ER diagram for the proposed framework. Real-world things can be taken as an entity, so we have chosen help seeker and helper as two entities. Various attributes of the two entities are also presented. In help seeker contact number, fields of choice are multivalued attributes. Also in helper entity unique ID assigned to the user act as the primary key.

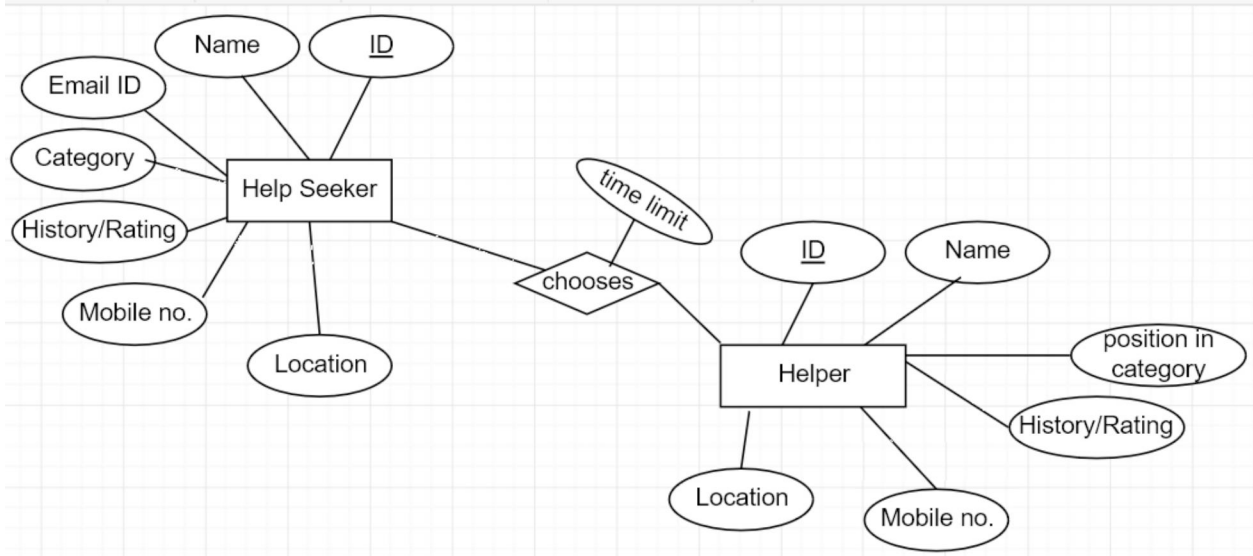


Figure 4.1. ER diagram representing two entities and interaction between them

4.3. Flow diagrams

4.3.1 Login Workflow

Figure 4.2 represents the starting login page of the framework, where the user has option to sign up using email id or social media. Once the user has successfully registered then the flow presented in next diagram is taken. If the user is old she can use sign in option and directly go to her homepage. When user log in with social media, app checks if the user already exists in the database . If user already exist, it will be directly signed in otherwise new profile will be created .

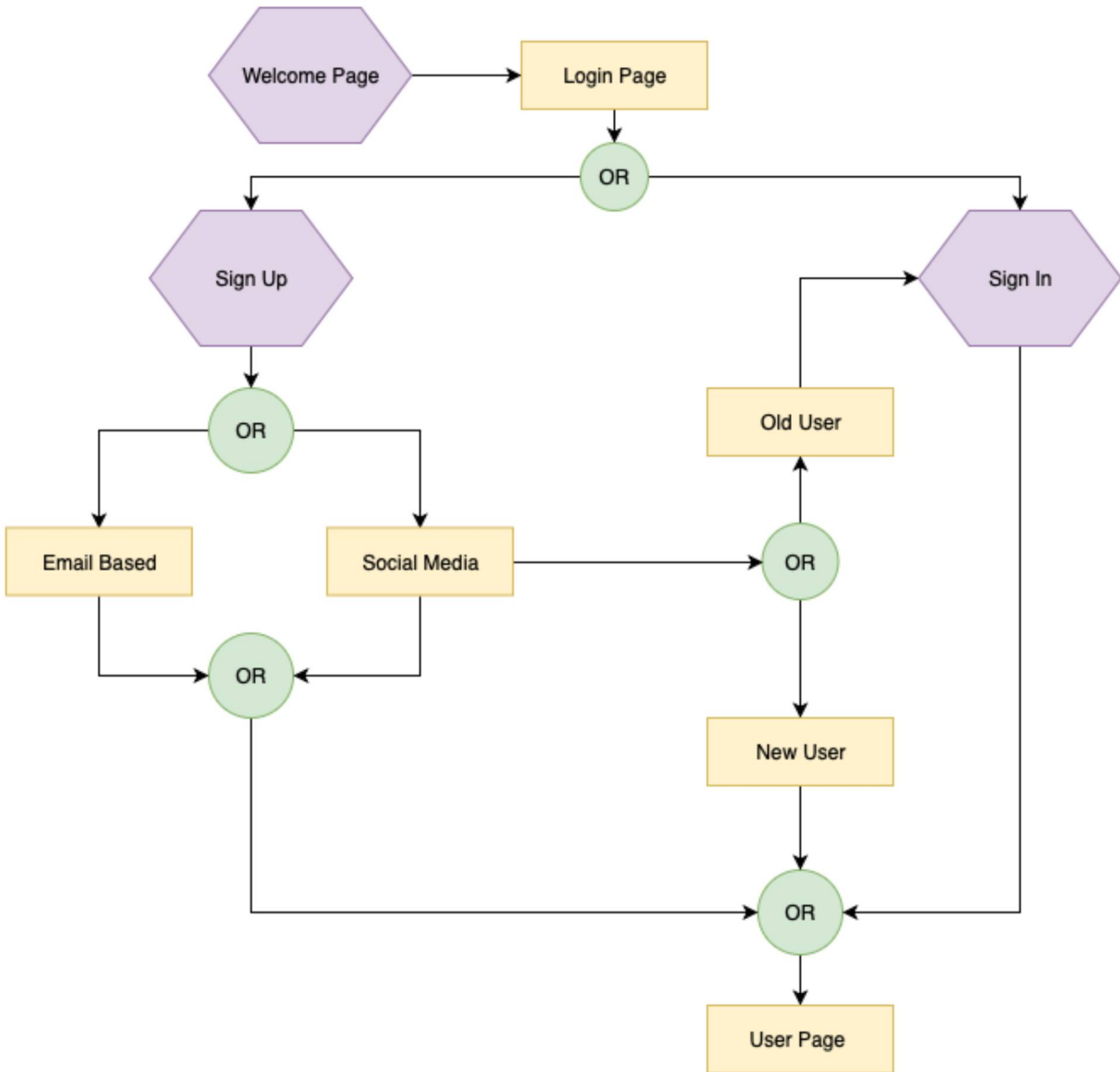


Figure 4.2. Flow diagram for getting login credential in the framework

4.3.2 Homescreen Workflow

Once the user has logged in, homepage is presented to the user where he/she has the option to go into any of the three modules. The framework is divided into these three modules to make the system manageable. Depending on the role of the user, she can choose any module for

the further flow of system. Once the chosen module ends, user gets the option to log out from the system. The described things are presented below in figure 4.3.

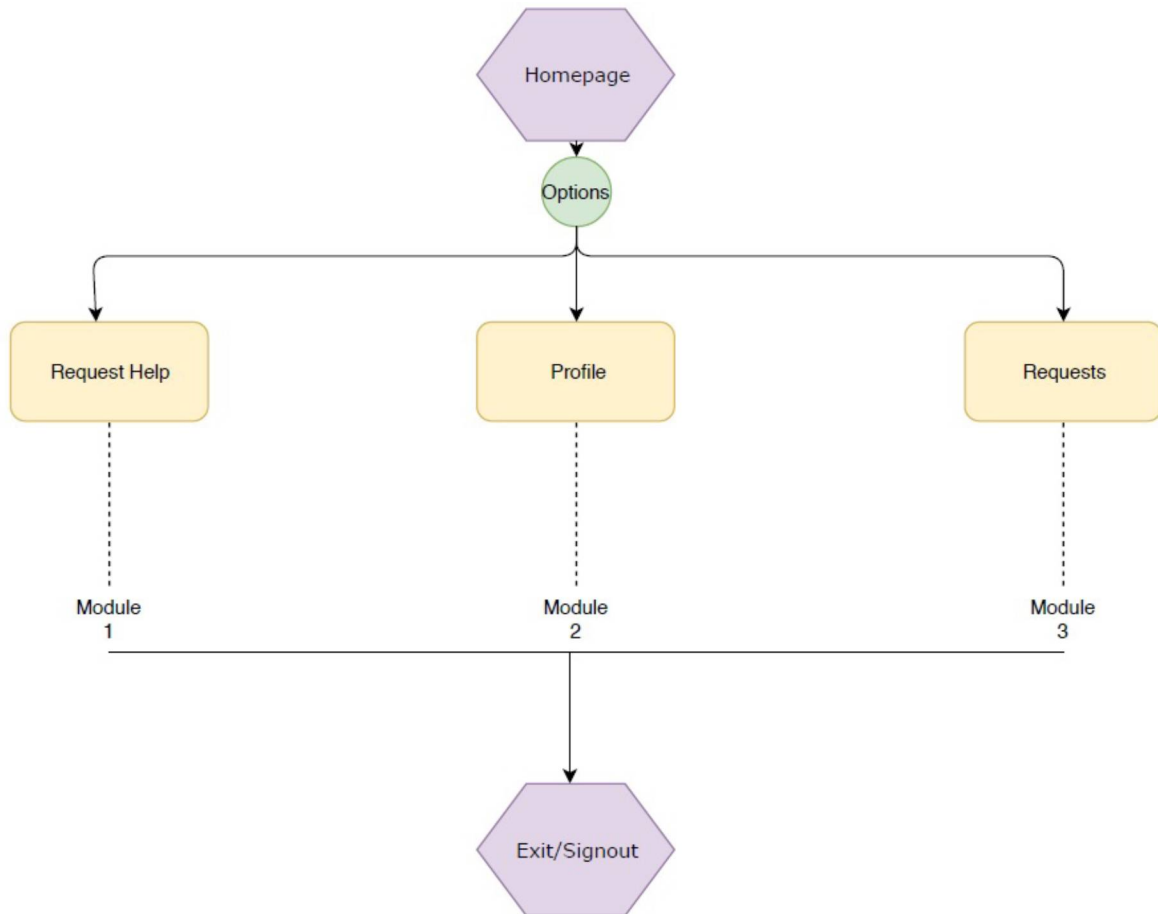


Figure 4.3. Control flow option in different modules after the user has logged-in successfully

4.3.3 Request User

The next three flow diagrams represent the flow of shown modules 1, 2 and 3 respectively. As shown in figure 4.4. When a person is on request-page in which she can post the request or can check the status of the already posted request. As shown in figure 4.5 if she is posting a new request, she needs to choose the category which best suits the type of request like

tuition, grocery etc. It is to be noted that each request will be posted publically which is shown to all users.

The user when posting the request can choose the desired radius of distance in which she needs the user who can help her. Also, she can specify the time frame in which help is required like within a day or in 24 hours. A timer is associated with each request that deletes the request automatically after the timer expires. Each request has associated details and requirements with it.

As depicted in figure 4.4 the user can get the person who is ready to provide help. Depending upon the list of users who are ready to help, the help seeker may choose anyone to get help. As both users get ready for help, the contact details of the users are shared on registered Email Ids.

There is also an option to track the position of the user in real time. After the help seeker has got the desired help from the helper, she can give a rating to the help provider. This rating can help the other users in the future to get the desired person in the category for which they are seeking help.

The user also has the option to choose from the previously saved or pre-drafted requests, which can be then be completed.

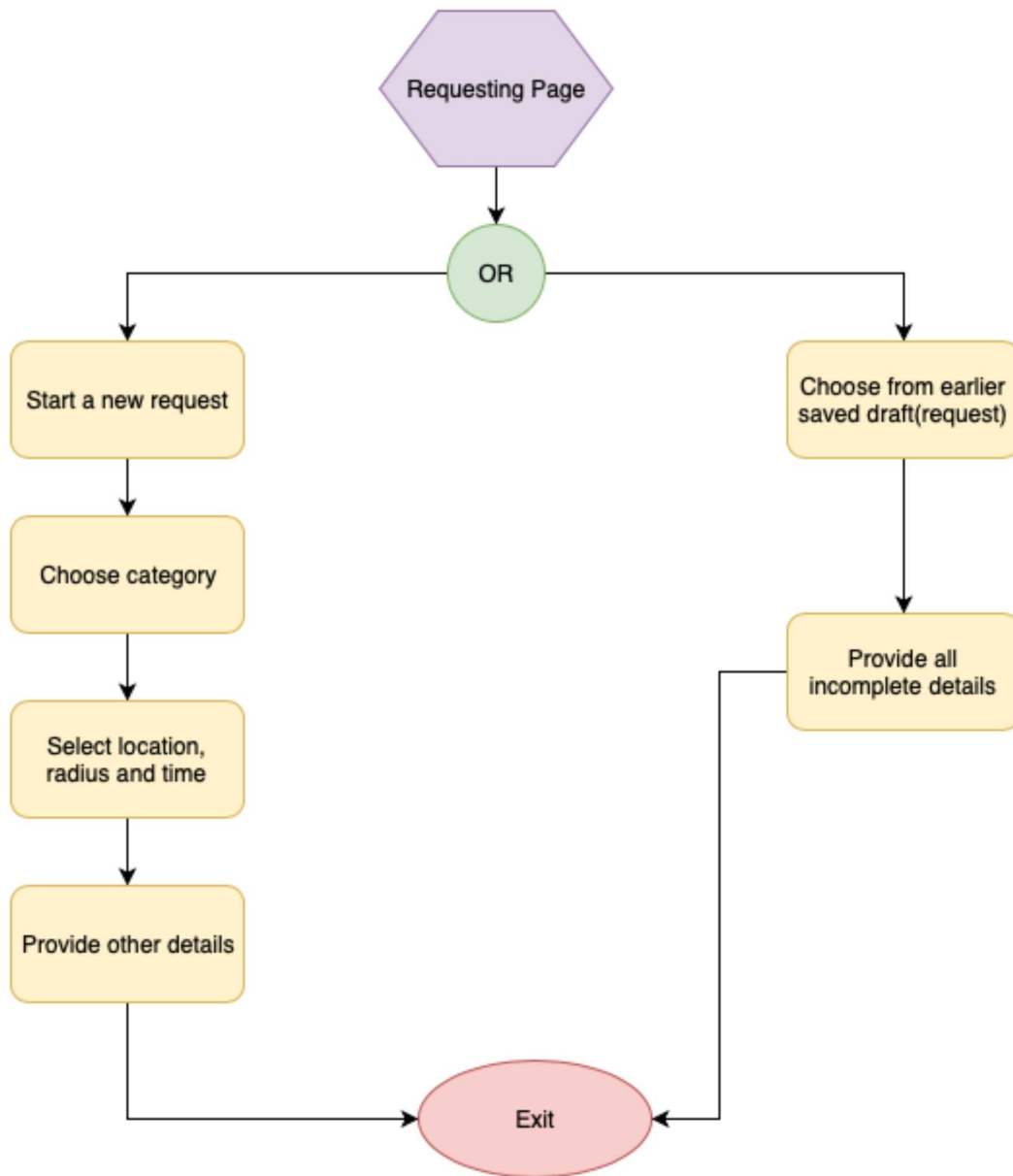


Figure 4.4. Module 1: Control flow for the user who is seeking help

4.3.4 User Profile

The second module of figure 4.5 deals with profiles of users as shown in figure 4.6. Any user can play role of both help seeker and help provider. The user can edit her profile that

includes privacy, contact info and other details related to the user. A user can also view the user ratings, location, and other public information. It is to be noted that only public information is visible for any user, once the users have agreed on providing the help their contact information is emailed to each other.

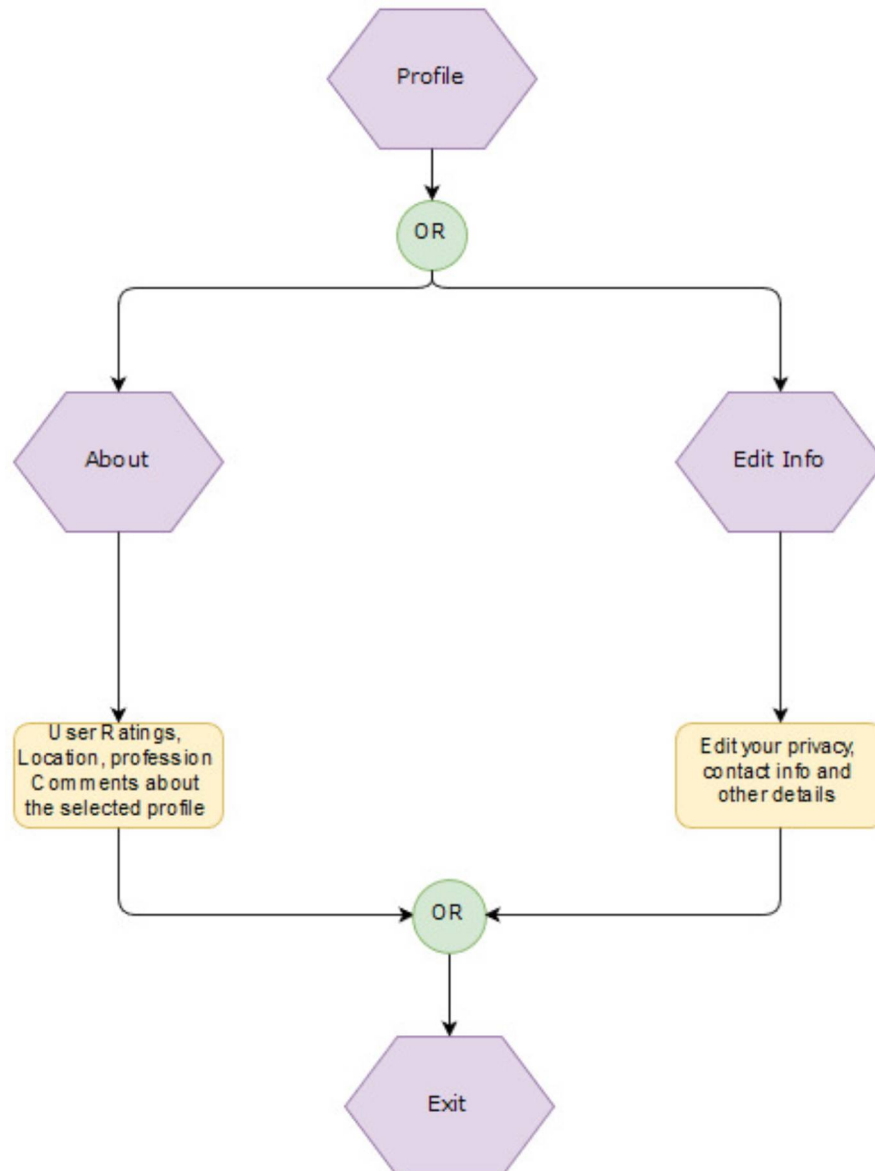


Figure 4.5. Module 2: Control flow for managing user information

4.3.5 Helper User

The third module of figure 4.3 presents the user who is ready to provide help, i.e. helper. Any helper as shown in figure 4.6 can choose the request she wants to fulfill from the list of open requests. Once the helper is clicked on open help request, both helper and help seeker get the Email containing the contact information of others.

Once the user has provided the help, they need to provide ratings to each other that can help the other users in the future. Even the user can ignore the request and move to other open requests.

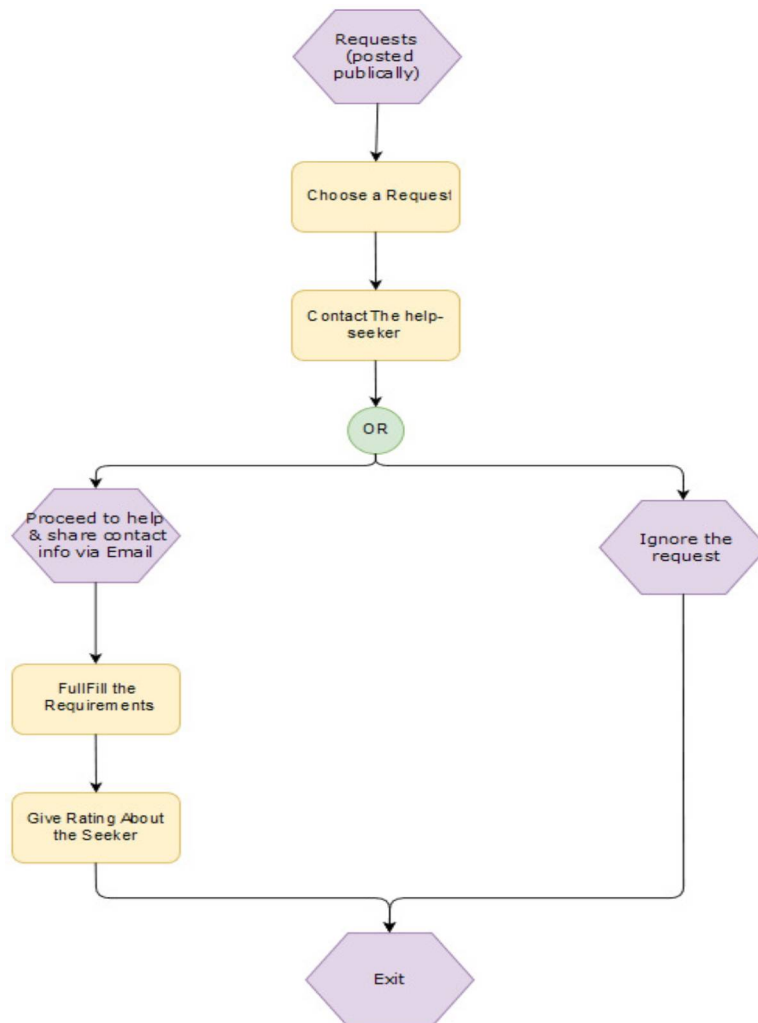


Figure 4.6. Module 3: Control flow for the user who is providing the help i.e. helper

4.4. Rating and Recommendation modules

Rating Module: As shown in figure 4.6, once the help has been provided the user can give rating, this module is provided in detail in figure 4.7. Once the help is completed, user is provided with a rating dialogue display where users can choose star ratings ranging from 1 to 5 stars. Once the rating is provided the average rating of all the categories is calculated. This rating is saved in the database along with the UserID of helper and current timestamp. Using the UserID the previous ratings of helper as well as UserID of seekers that gave rating and the time of rating are extracted in the array. Then we compare the Current seeker UserID with the Seeker UserID in the array, if it turns out to be false we directly update the rating in the database. However if the two come out to be true then we compare current time with rating time. We further compare time difference with the month and check if seekerUIDcount(Count of total unique ids of seekers that rated the same helper) is greater than 3. If any of the two conditions become false we update the rating in the database. Further, if both conditions become true oldest entry of rating gets deleted and all entries are pushed back using FIFO concept. These updated entries are updated back in the database.

So here we are implementing a decay factor that uses two main factors. Firstly, if rating is 1 year old we remove it so that we get an updated version of rating always. Secondly, we wanted to avoid *bias rating*. It checks if the same seeker is rating same helper every time. So it is like if user A rate user B more than 3 times in a month only recent three ratings will be stored. The

oldest one of the same month will be removed and the new one will be stored. These two things are shown in figure 4.7, in blue diamond decision making blocks.

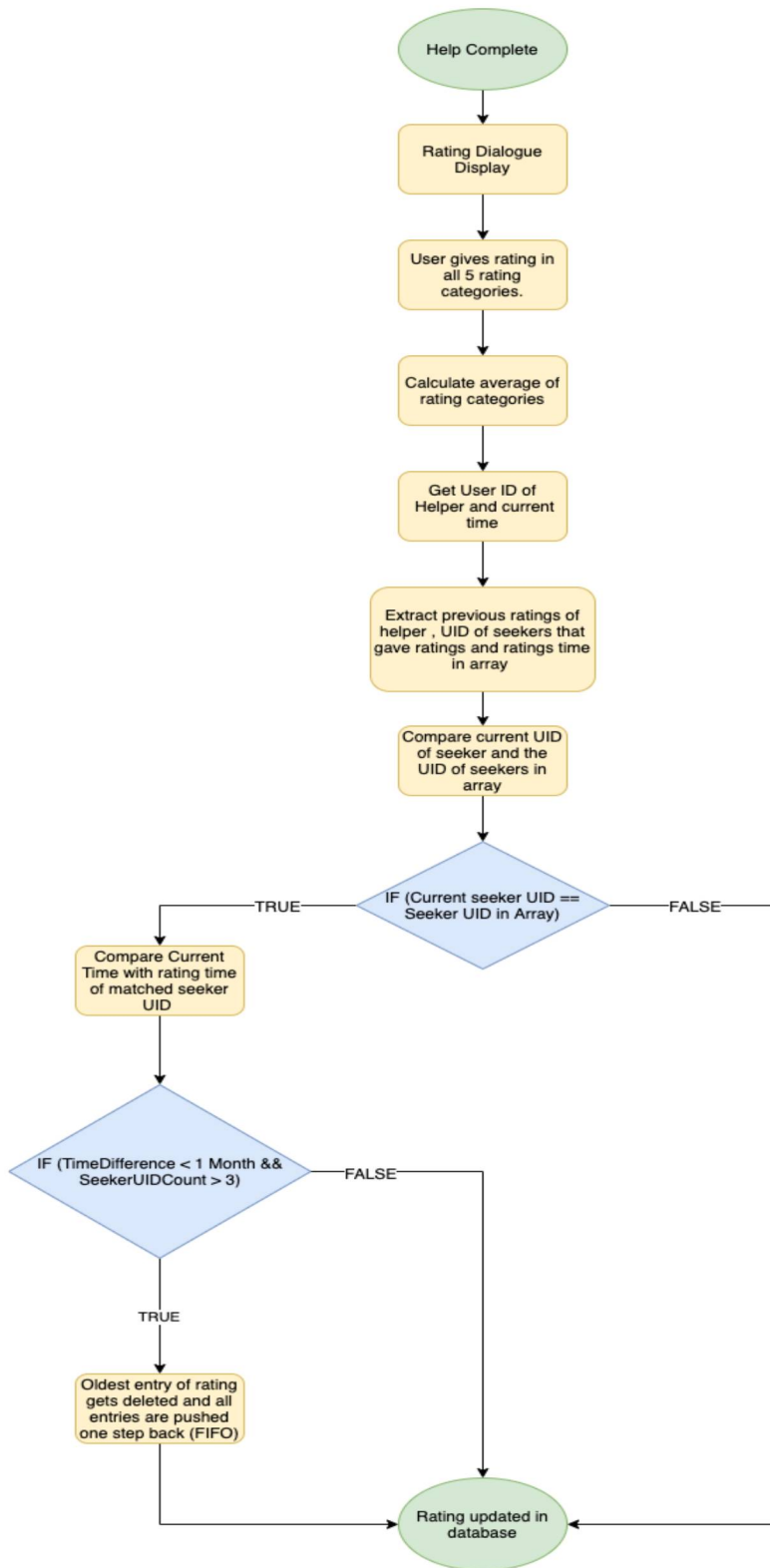


Figure 4.7. Module for calculating the rating of the user

Recommendation Module: This module is used to calculate and display the recommendation to a user who is seeking help in a particular domain. As shown in figure 4.8 depending on the type of help, the user has selected all the profiles of that type will be extracted from the database. These profiles are filtered based on the radius of the help seeker. This provides the help seeker with the profiles which are in the vicinity of the user. Along with the profiles the ratings and time of rating are also extracted.

In the next step, to recommend the profiles extracted in the previous step, we compare the ratings of the profile with the current time and if the time difference is greater than 1 month we remove such rating with the database. This is done so as to keep only fresh ratings in the database. If more than one profile is found with rating then we sort those profiles in descending order of rating and show the recommendation to the help seeker.

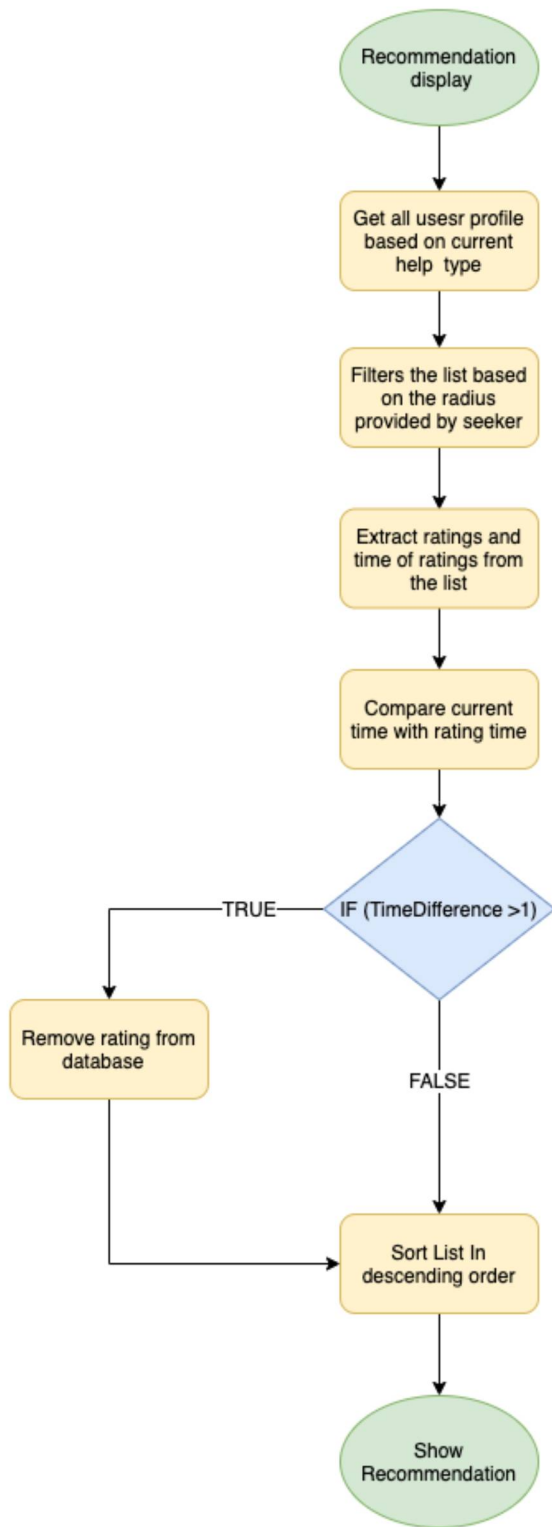


Figure 4.8. Recommendation module for the user

Detailed Explanation of Recommendation system:

This section presents the detailed explanation of the recommendation system that is used in our application. Our Recommendation system is the combination of three main criteria namely:

- i) Recommendation
- ii) Rating
- iii) Decay Factors

These factor in combination determines the overall rating of any users and based on these ratings, user gets the recommendations. Further, the decay factor is added to keep the user rating upto date. So, these three factors are closely connected. Next, we will explain the rating and recommendation module in detail.

i) Recommendation: Following steps are followed for the recommendation module

Step 1: When user opens “help page”, the module check if the user has the same id from which the help was posted

```
if (CurrentUserId == SeekerUserId)      ....(1)
```

Step 2: If the `UserId` is same then get the current help type (i.e. condition 1 is true)

```
currentHelpType = getCurrentHelpType ( )
```

Step 3: Using current help type, query database to get all the user profiles from database with the same help category as current help type.

```
UserRef = FirebaseDatabase.getInstance( ).getReference( ). child(“currentHelpType”)
...(2)
```

Step 4: Filter the `UserRef` list obtained in equation 2 based on radius provided by seeker. This filtration is done by comparing the location of seeker's help and locations of helper's in the list `UserRef` which gives us Aerial distance between them.

If the Aerial distance is greater than the radius provided by seeker the user profile gets rejected else, it gets stored in an array as shown below

```
int radius = getHelpRadius();
arialDistance = getAerialDistanceByGeocodingApi();
if (radius > arialDstance){
    reject user profile;
}
else {
    Array ReccomendedUser[ ] = Users from UserRef .. (3)
}
```

Decay Factor used bellow which rejects any rating which is older than 1 year.

Step 5. Extract all the user's ratings of user from `RecommendedUserArray[]` as calculated in equation 3. Compare the current date and time with the user's rating date and time if the difference is greater than year reject it else take the average of the ratings and store in an array.

```
If (CurrentDateTime - RatingsArray.DateTime < 1 year) {
    AverageRattingArray = getAverarageRating( );
}
```

Step 6: Now we sort the `AverageRatingsArray` in descending order with `usersProfiles` and display the helper's profiles on the recycler view as recommendation.

Above algorithm will recommend helpers to seekers using different factors like rating, decay factor by time, radius and type of help.

ii) Ratings: This subsection discusses the rating module. We use the concept of bias rating that is explained in detail below.

Step 1: After help is complete, the user gets a rating dialogue. Here help seeker can rates the helper in 5 different categories: communication, courteous, generosity, punctuality and skill . We take the average of 5 and store as one rating.

$$\text{helperRating} = (\text{communicationRating} + \text{CourteousRating} + \text{GenerosityRating} + \text{PunctualityRating} + \text{SkillRating})/5$$

Step 2: It gets the helper profile and the previous ratings and compares the date and time of each individual ratings with current date and time. If the difference is less than a month that particular rating gets stored in different array `BiasedratingsArray`.

```
if (TimeDifference < 1 month) {  
    BiasRatingArray[ ] = rating  
}
```

Step 3: After getting all the ratings of current month in `BiasedratingsArray`. Compare last 3 `ratingIDs` with `seekersId`. If the value is same the older rating is deleted and `currentRating` is Added in last position.

```
If (RatingUserId = seekerUserId)  
{  
    Delete the old value of rating and add the new rating in BiasRatingArray[ ]  
}
```

Step 4: Rating are updated in the backend database.

4.5. Data Flow Diagrams (DFDs)

As data flow diagrams represent, the flow of data in the system. It describes the data inputs and outputs, data stores and various sub-processes through which the data moves. We will start with

the DFD level 0 in which only one process "Helping Sapiens" is presented. Help seeker and help provider are the two entities which interact. The flow of help requests is presented in figure 4.9. Details of the user (help seeker/provider) can also be fetched. Rating is also used in the system for making it easy for future users. As shown in below figure two type of user help seeker and help providers can interact with the app. As shown by the edged arrow, a help seeker can request for any type of help and the application in return will provide the detail of the person who is going to serve her (arrow shown in the direction app to help seeker). Further the help provider will get detail of the person who is seeking help. Also, she will confirm in return that she is ready to help.



Figure 4.9. DFD level 0 for the proposed framework

CHAPTER 5. IMPLEMENTATION DETAILS

In this chapter we discuss the implementation details for the mobile app along with the implementation of reputation and recommendation system which was presented in figure 4.8 and 4.9 is given. Implementation is done on Intel Xenon hexacore processor E5620, clock cycle 2.40 GHz with 8 GB RAM running Windows server 2012 R2 standard. Android Studio version 3.0 is used for the implementation as shown in figure 5.1.



Figure 5.1. Android studio vs 3.0 is used for implementation

Some of the basic commands and code in accordance with the diagram of chapter 4 are shown below. The database that is used is Firebase. The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

Instead of typical HTTP requests, the Firebase Realtime Database uses data synchronization—every time data changes, any connected device receives that update within milliseconds. Provide collaborative and immersive experiences without thinking about the networking code.

In this chapter we discuss the implementation details for the mobile app alongwith the implementation of reputation and recommnedation system which was presented in figure 4.8 and 4.9 is given. Implementaion is done on Intel Xenon hexacore processor E5620, clock cycle 2.40 GHz with 8 GB RAM running Windows server 2012 R2 standard. Android Studio version 3.0 is used for the implementation as shown in figure 5.1.



Figure 5.1. Android studio vs 3.0 is used for implementation

Some of the basic commands and code in accordance with the diagram of chapter 4 are shown below. The database that is used is Firebase. The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

Instead of typical HTTP requests, the Firebase Realtime Database uses data synchronization—every time data changes, any connected device receives that update within milliseconds. Provide collaborative and immersive experiences without thinking about the networking code.



Figure 5.2. Firebase is used for the database implementation

Firebase apps remain responsive even when offline because the Firebase Realtime Database SDK persists your data to disk. Once connectivity is reestablished, the client device receives any changes it missed, synchronizing it with the current server state.

The Firebase Realtime Database can be accessed directly from a mobile device or web browser; there's no need for an application server. Security and data validation are available through the

Firebase Realtime Database Security Rules, expression-based rules that are executed when data is read or written.

Code snippets of important modules are presented in the appendix A, reader may refer them to get an insight of the implementation.

The next chapter i.e. chapter 6 presents the results and screenshots of the developed app. Also, future work for the developed project is presented in the next chapter

CHAPTER 6. RESULTS AND FUTURE SCOPE

This chapter shows the results in form of screenshot of Helping Sapiens and also how the values in the database are updated when a new user or requested is generated by the user. This chapter also presents the future scope of the project.

Although the interface of the developed app is self-explanatory here I will explain step by step procedure of using the app.

Step 1: The user gets the first interface of either login if she has an account else she needs to signup for the application as shown in figure 6.1. The new user can signup using Google.

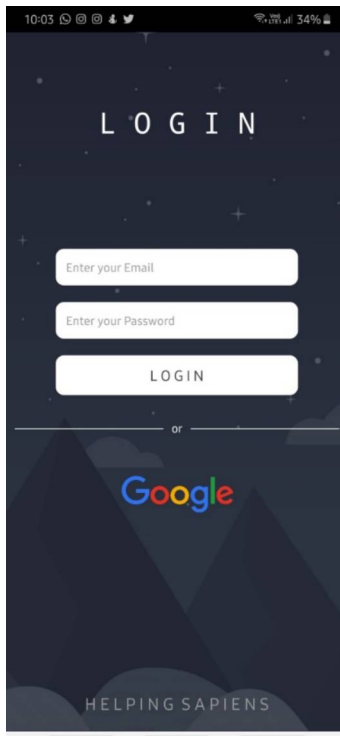


Figure 6.1. The first screen of login for existing user or signup for new users

Step 2: If the user has clicked on signup then the app will ask the user to enter some details as shown in figure 6.2. The details that I am collecting right now is username that will be a unique field. When two user-helper and seeker has agree to meet their corresponding details i.e. email ID and phone number will be shared with the other one. As discussed in the last chapter all the checks corresponding to the email ID and password filed are applied.

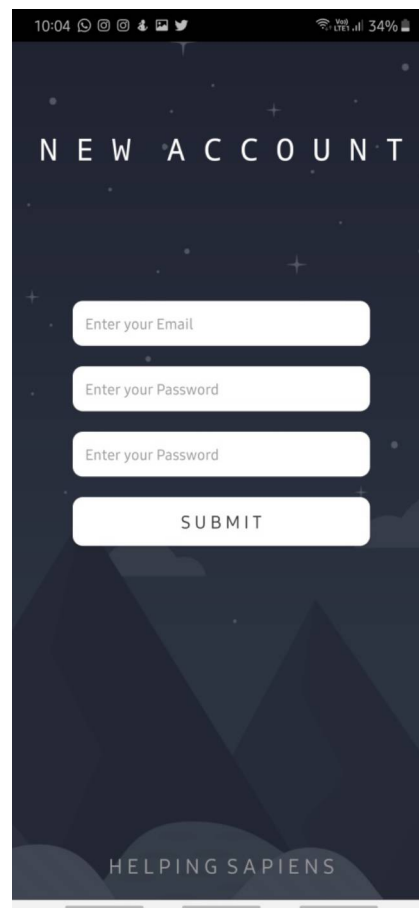


Figure 6.2. Signup data required for a user

Step 3: If the user is already registered she can directly login as shown in figure 6.3.

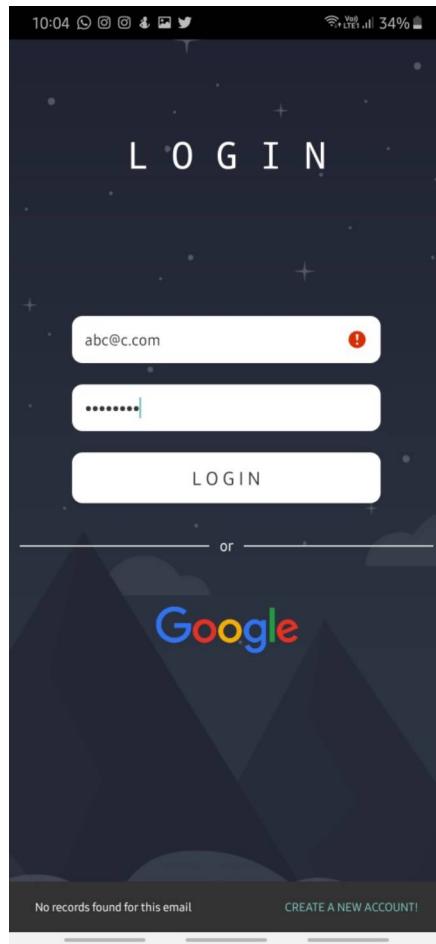


Figure 6.3. An already register user can directly login

Step 4: After a user has successfully logged in she by default gets the role of help seeker. As shown in figure 6.4 we ask the user to enter her name to be displayed and contact number. I have fixed some of the categories like buying groceries, Tuitions, traveling etc.

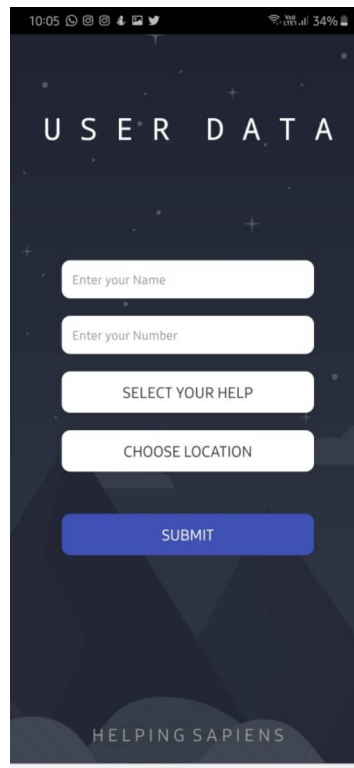
The image shows a mobile application interface for user registration. At the top, the status bar shows the time 10:05, signal strength, Wi-Fi, and 34% battery. The main title 'USER DATA' is centered in a white, spaced-out font. Below the title are two white input fields with the placeholder text 'Enter your Name' and 'Enter your Number'. Underneath these are three white buttons with rounded corners: 'SELECT YOUR HELP', 'CHOOSE LOCATION', and a larger blue button labeled 'SUBMIT'. At the bottom of the screen, the text 'HELPING SAPIENS' is displayed in a light gray font. The background is a dark blue gradient with faint white stars and geometric shapes.

Figure 6.4. The information user is asked for alongwith the location and category

Step 5: figure 6.5 various categories that the user can choose from. These categories are not provided as a radio button but as checkboxes that means a single user can choose multiple categories at once. Figure 6.6 depicts how a user can choose the location in the vicinity of which she is seeking help.

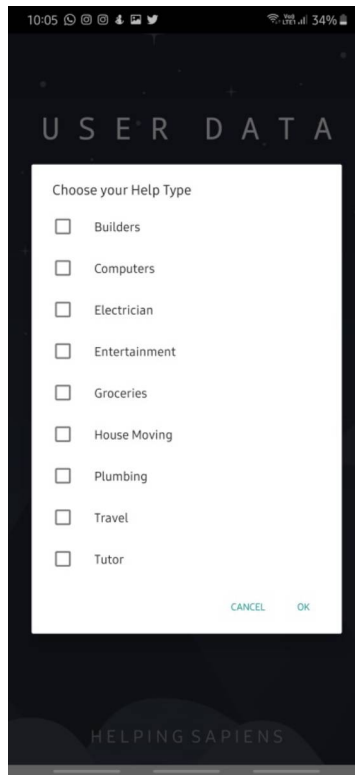


Figure 6.5. Various categories that the user is provided with



Figure 6.6. The user can also choose the location on map

Step 6: Once the user has chosen one or more categories in the previous step, we execute our recommendation module as discussed in previous chapter and all the users who are around the help seeker are displayed. Only the users who have provided help in the above-chosen category are displayed as shown in figure 6.7.

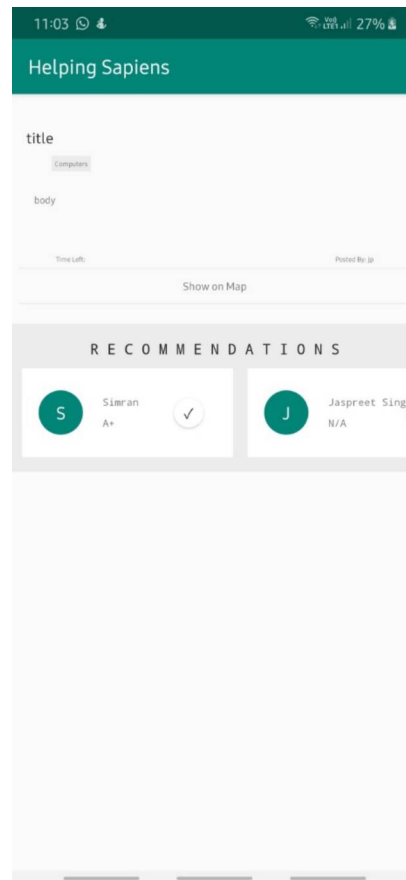


Figure 6.7. Recommendation module

Step 7: As shown in figure 6.8, a sorted list of users is provided to the help seeker. The rating is used as a sorting criterion.

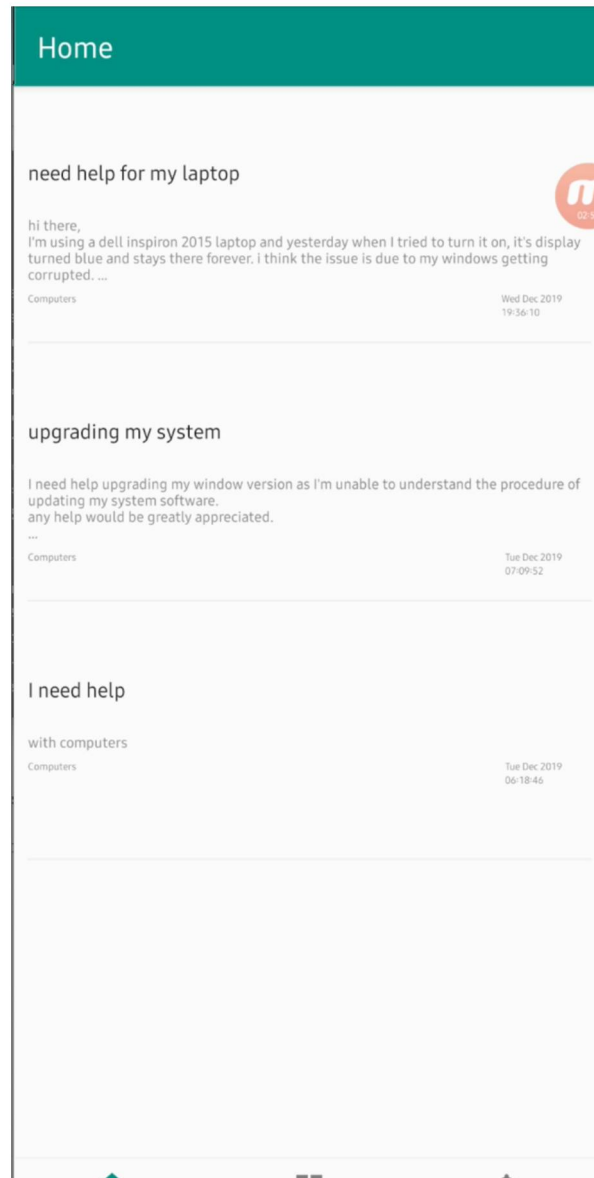


Figure 6.8. A sorted list of the users is presented to help seeker.

Step 8: After the interaction is completed, the seeker will get a screen to give a rating to the provider as shown in figure 6.9. There are various criteria based on which user can be rated as shown below.

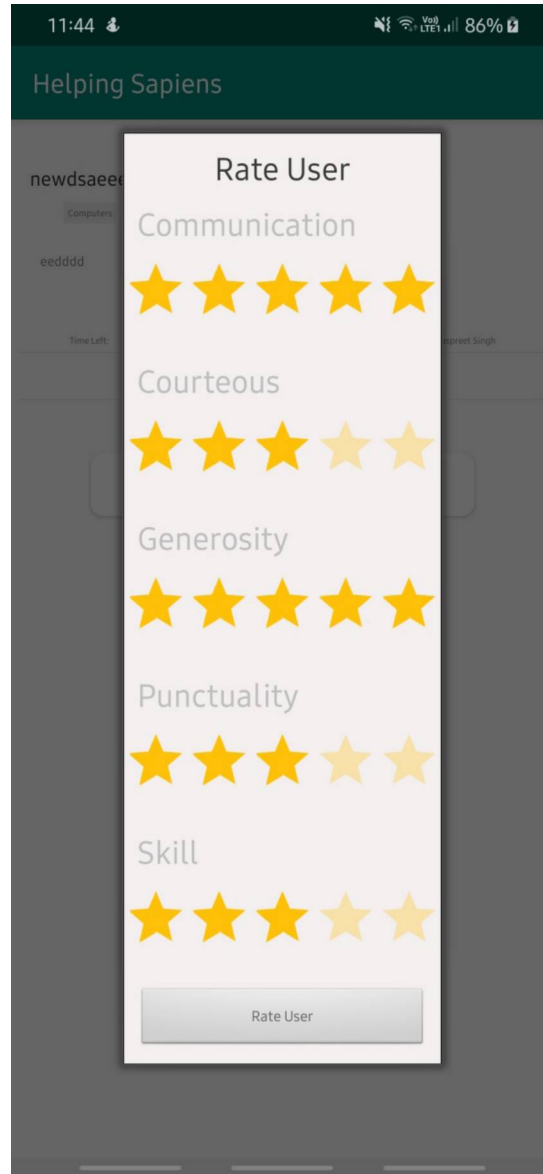


Figure 6.9. The rating option on which a help seeker can rate help provider on various parameters

Future Scope:

The future scope of the application is summarized in this section. The recommendation system can be improved further to get exactly the same type of users.

1. More categories can be added so that category-based recommendations can be provided to the user.
2. Live tracking of the user can be added to get the exact location of the users on both sides.
3. Sentiment Analysis can be used to find the level of help needed by user.
4. AI can be used for recommendation.
5. IOS version of the app can be developed for iPhone users.

Proof of concept: As per the literature found over the Internet this term appears first in 1967 [58]. Proof of concept can be defined as the phase in development in which experimental product is constructed and tested to explore and demonstrate the feasibility of a new concept [59]. The proof of concept is used in many areas and in engineering and technology, a rough prototype of new idea is constructed as proof of concept. Here we are also using the same definition and will apply this to our area of thesis. Although this is not something which is done in the end we are just putting in the last chapter as it supports the full development of the Helping Sapiens. The following steps will be used in the proof of concept in terms of Helping Sapiens app.

- 1. Prove the need:** This step involves the reason for the app that is to be developed. In our case as we have discussed various motivations in chapter 1. We have always been taught from our childhood time that helping people is good. It not only makes the life of the ones we help a bit easier but also gives us inner peace and satisfaction [5]. Overall et al. [45] discusses the effect on personality and social growth of a person if she helps others. That help can be for love partners, self-improvement, and relationship quality. The life experiences of author also acted as a motivating factor to work on this platform. Some of the motivational factors are listed below, some are taken from the literature [39] and some are included from personal experience. This motivated us to build an app that can be used to provide help to users and can connect help seekers and providers.
- 2. Brainstorming and finding the feasible solutions:** After finding the motivation and problem statement we had several meetings amongst the research group to find out what can be done to connect helpers and seekers. Most people always carry a mobile phone so we jointly came up with the idea of a mobile app. There were other solutions as well but this solution was the prominent one.
- 3. Prototype the solution and test:** To get a detailed insight into the solution that was finalized in the last step, detailed analysis was done. Various figures and flow charts of chapter 4 were drawn. Further we added the concept of rating and recommendation in the app, the rating concept helps the user to give feedback to the system on various parameters. This rating is can be used by other users of the app for recommendation.

When the user chooses some category based on that we can give him the suggestion of help providers in that category.

- 4. Create a Minimum Viable solution:** A minimum viable solution of the above-discussed problem was created in the form of app Helping Sapiens. This app has the minimum functionality to connect help seekers with the help providers. In chapter 6 above various screenshots of the developed app are presented. The module of rating and recommendation is also included in the developed solution.
- 5. Roadmap for the actual product:** Developing this app was a learning and challenging task. It helped me in the understanding of various technologies that can be used further in many fields. This app is currently being used by my close friends and family members, I am working on their feedback and will keep on updating the things as per their feedback. As discussed above in the future scope section, more functionality can be added to make it more user effective.

Next we will discuss some cases where the user have actually used this app. This can be considered as the pseudo experiment to show the above discussed proof of concept.

As shown in figure 6.10 user who seeks help fills all the details. There is a requirement of heading of the help and detail of the help that she is asking for, is also filled. The help seeker can also specify the time for which this request will remain active.

Ask For Help

need computer repair

i have a dell all in one desktop which is not working. When i click the On button, the Fan starts however nothing gets displayed.
It would be great if someone could help me repair it.
any help would be appreciated.

Computers | Map

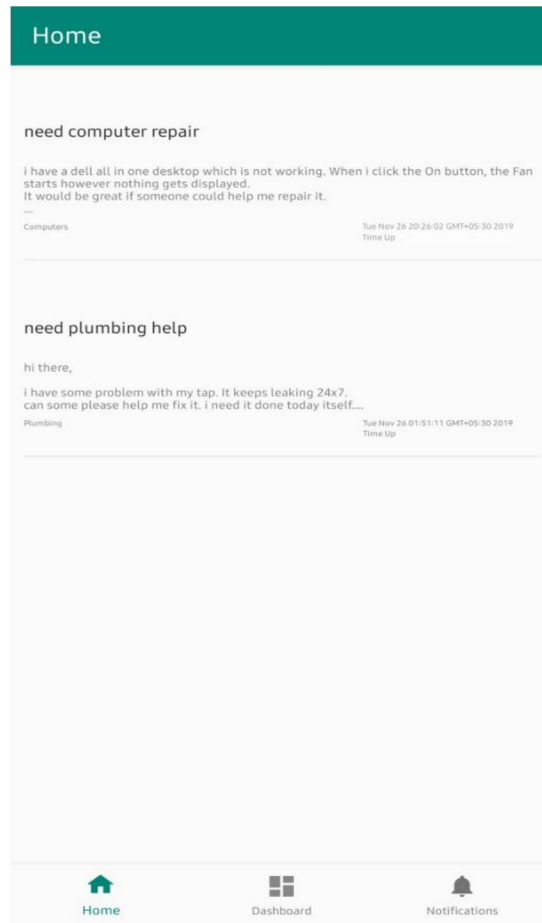
Time Allotted:

31	24
0	0
1	1
Days	Hours

SUBMIT

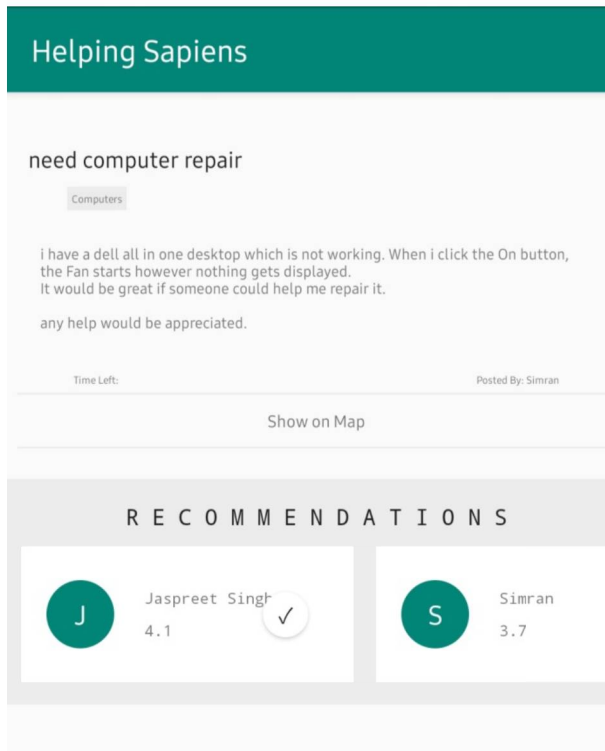
Figure 6.10. A user posted help for computer repair

The help is posted on the timeline and is seen by every user within the seeker radius. As shown in figure 6.11, the help providers can see all the helps that are posted in the vicinity. As shown in below figure there are two open requests for computer repair and plumbing request.



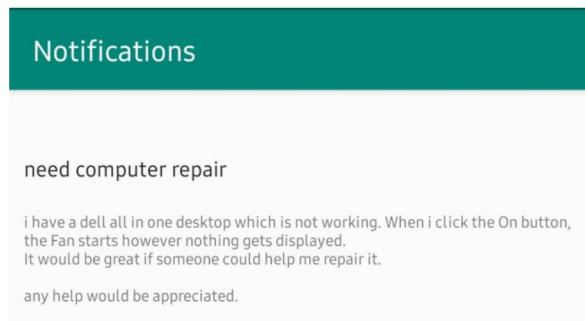
6.11. All the requests that are open in the vicinity.

Further, to help the seeker in the best possible way our recommendation algorithm gives her some suggestion of the help provides. The seeker gets recommendations of various users based on the help that they have provided in the past as shown in figure 6.12.



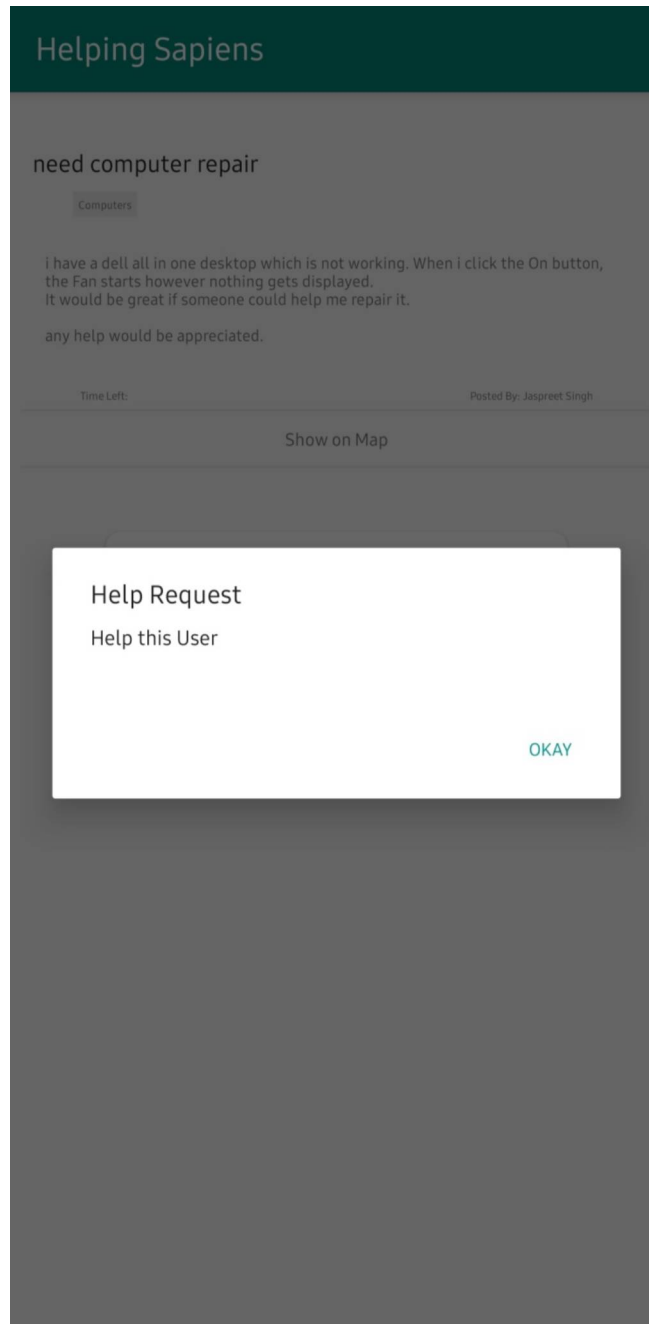
6.12. Recommendation shows the users based on their past rating

If help seeker clicks on the tick, present on the right of recommendations card the user gets a notification as shown below in figure 6.13.



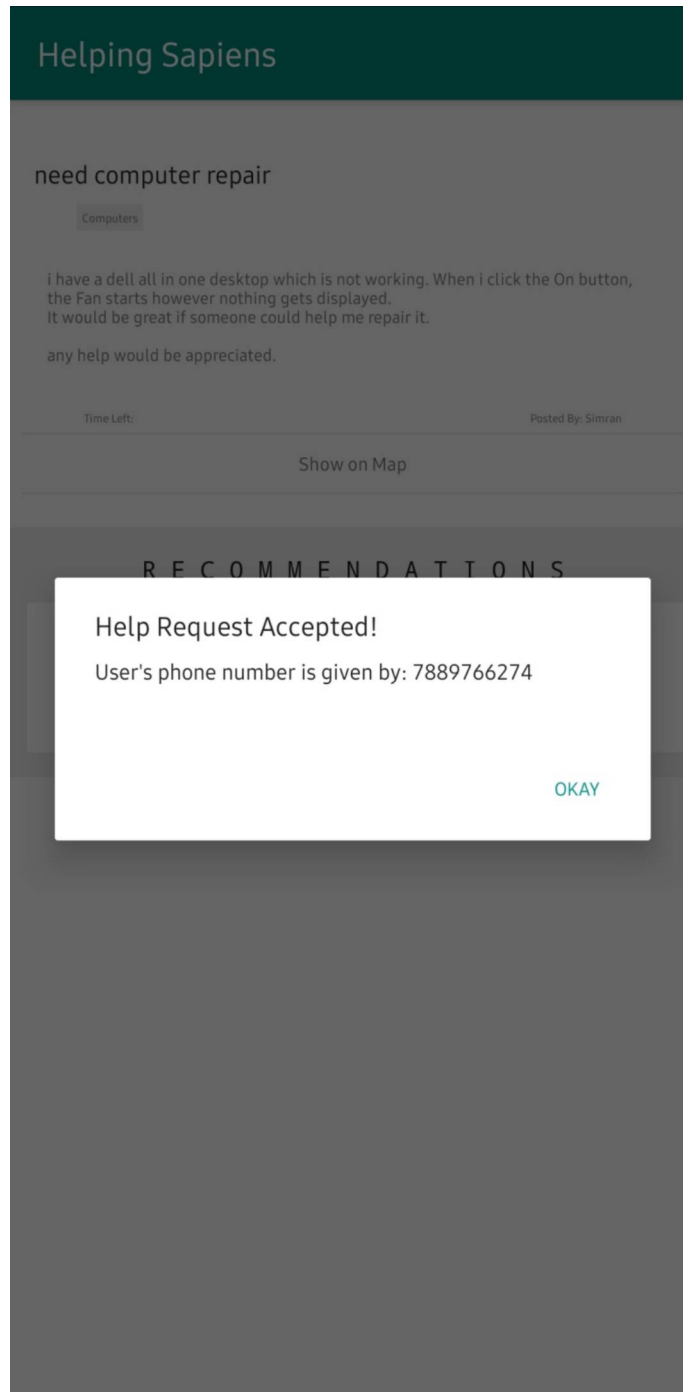
6.13. As the user click on the right symbol, user gets the notification

As shown in figure 6.14 when user clicks on the notification, he gets a dialog asking for approval, where user can click on okay to send her consent to the help seeker.



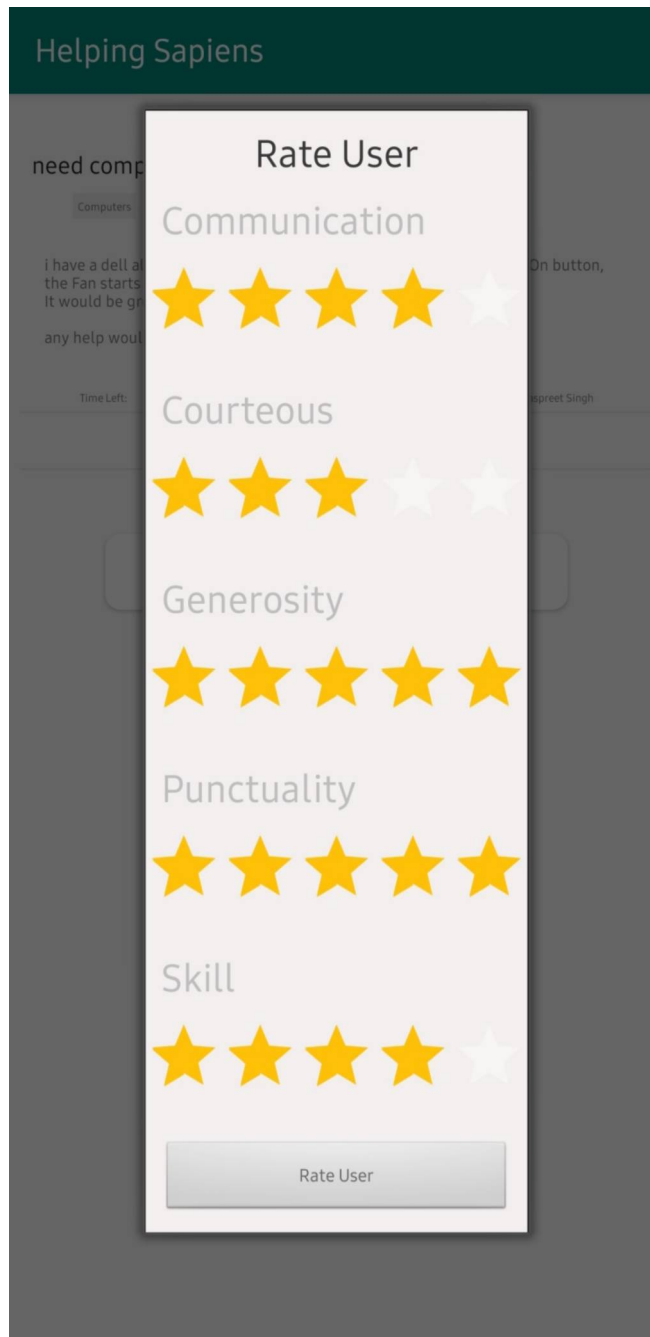
6.14. The user gets the notification where she can okay to approve the request.

If user clicks on Okay Seeker gets helpers Phone number as shown below in figure 6.15. This way the help seeker and help providers can be connected.



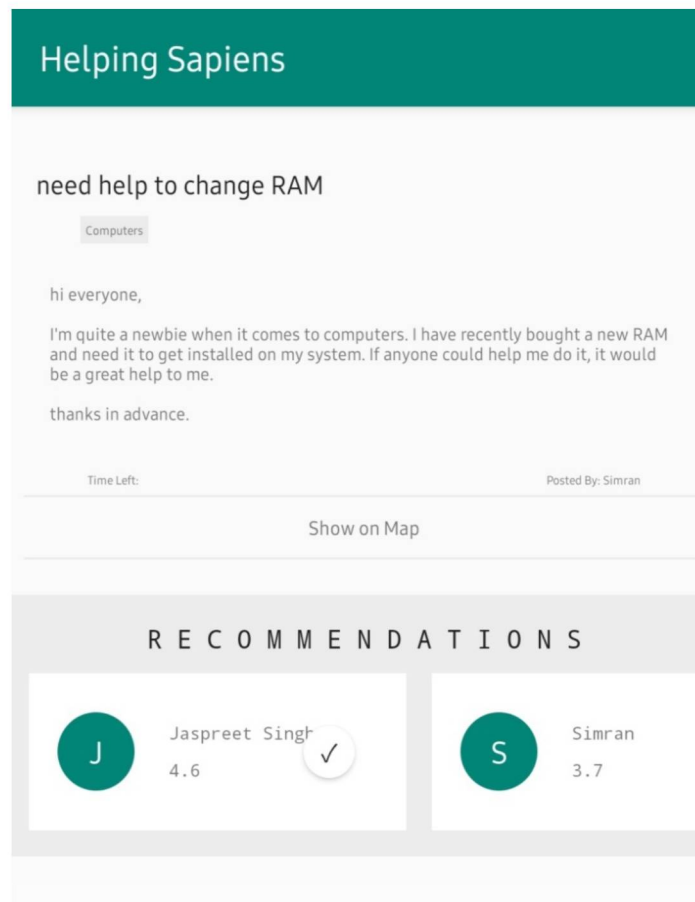
6.14. The user gets the notification where she can “okay” to approve the request.

When the Help is complete, the seeker gets a rating dialog as shown below in figure 6.15 to rate the user. This rating is used for our recommendation module as explained in last chapter. The various features on which user can be rated are communication, courteous, generosity, punctuality, skill.



6.15. After the help has been provided the user get a prompt to rate the help provider

When a new User posts a new help. He sees recommendations with updated ratings from previous user as shown in figure 6.16.



6.16. The rating gets updated after the help provider has provided the rating to the seeker.

This clearly shows that the app behaved in a way which satisfy all the objectives that were defined in the starting of the chapter. Further, other cases were also tried (screenshot not attached here) that shows the success of the application build.

CHAPTER 7. Experimental Design

After building this app the next step is to validate the features and determining the points of improvement. This section discusses the methods of the survey that may be followed to determine what features and elements need upgrading or do not work correctly. Experimental questions based on the technology acceptance model (TAM), as proposed by [60], will be used to measure:

* Perceived usefulness (PU) or the degree to which a person believes that the system will perform the desired capability and

* Perceived ease-of-use (PEOU) or the degree to which a person believes that a system is simple to use

Target Sample size: We will target a total of 260 users with varying age groups and professions.

<i>Age group</i>	<i>Number of users</i>	<i>Profession</i>	<i>Gender</i>
10-20	30	Students	Male (15), Female(15)
20-30	30	Students	M (15), F(15)
20-35	30	Working professional	M(15), F(15)
35-50	30	Working professional	M(15), F(15)
35-50	15	House makers	F(20)

50+	30	Working professional	M(15), F(15)
50+	30	Retired	M(15), F(15)

The above table represents the target users of the app; further different age group and size is chosen to avoid any bias.

Survey Research

Survey research is an effective way to assess TAM constructs. In essence, survey research allows participants to report directly on their own thoughts, feelings, and behaviors regarding the system (Jhangiani et al, 2017) [60]. Close-ended survey TAM questions adopt likert-style choices on a sliding scale (e.g. 1 through 5 where 1=Strongly Agree, 3=Neutral and 5=Strongly Disagree). These constructs would gauge users' perceptions of Helping Sapiens across the following areas:

- * Quantity of available matches
- * Quality of available matches

The survey for our app will have two types of questions that is Open-Ended and Close-Ended Questions. Open-ended questions are useful for qualitative feedback and to get exploratory input from the users. They are also an excellent way to dig deeper into a specific area when coupled with a particular question. These types of questions are time-consuming both for users and for us to go them individually. The following types of open-ended question be added:

1. How effective was Helping Sapiens in discovering helpers?
2. What features, if any, would you like to see added to the system?
3. What features, if any, would you like to see removed from the system?

Close-ended questions are useful for quantitative feedback and to answer specific questions you might have. This quantifiable data shows you the “big picture”, and is an excellent way to reveal trends and patterns in your app’s use.

1. Does this app helped you to connect with the users you were seeking for help?
2. Should the distance be increased or decreased to get more matches with the user of interest.
3. Does the category in which help seeker posted the request matches with the actual help they were looking for?

User experience should be measured as well in order to understand which features of the system were effective and which should be redesigned.

Rating and Recommendation Module : As rating and recommendation modules act as the backbone of our developed app, to get the feedback on these two modules, following questions can be included in the survey. Sample survey questions may include the following.

1. Does the person who was matched based on the rating of other users justifies the actual behavior of the user?

Importance : Based on the feedback from the user we can modify our rating module to make it more reliable to the user.

2. The criteria for evaluating helpers was appropriate.

Importance: We can further add more features in the app so that the rating module can cover overall qualities of help provider.

3. Do you trust the recommendations made by the system.

Importance: This open ended question can help us in improving the trust factor that act the basic of rating and recommendation module.

4. Collaborative filtering helped me to discover new helpers.

Importance: As our recommendation module uses the concept of collaborative filtering, if the users are not matched with the same category of helpers alternative recommendation modules maybe tested.

5. Discarding recommendations after a certain time-period seems like a good approach to keeping recommendations relevant.

Importance: As **decay factor** plays an importance role in determining the recommendation to the user. We can improve our decay factor formula based on the feedback from the user.

6. Do we need to add more categories in the recommendation module so that the user can be matched directly to that particular type of category helper.

Importance: Adding more category can help the user to match directly in that category.

As decay factor plays an important role in deciding the rating which indirectly determines the recommendation. So above factors also represent the decay factor indirectly.

CHAPTER 8. CONCLUSION

This chapter represents the final “take away” of the thesis. In chapter 1, we started with the motivation behind building this app called "Helping Sapiens". We also discussed the rating and recommendation systems that act as the basic building blocks of our app. We discussed the various studies that have used this concept in the literature.

In chapter 2, we provided various studies related to the field of reputation system and recommendation system. All these studies are briefly described so that reader can get an insight of these works. Apart from this location based social media related studies are also discussed. As Helping Sapiens also uses the concept of location-based feature to connect help seekers with help providers. The concept of trust that is used for rating system is also discussed in chapter 2. Various existing apps that motivated our work are presented in the same chapter. These apps were studied to get an idea of basic working concept of such apps.

The shortcomings of such apps were also studied and improved in Helping Sapiens. Various studies are compared in tabular form with the key findings, shortcomings and dataset. The features of various studies were presented in the next table, so that reader can understand which are the important features that needs to be included when building such apps.

Chapter 3 presented the problem statement and research gaps identified from various studies in chapter 2. The objectives of building Helping Sapiens are also given in this chapter. All the objectives presented in this chapter are getting satisfied and we are still verifying this app in real world.

Chapter 4 discussed the framework design for the proposed work. Various flow diagrams of the modules along with the explanation was presented in this chapter. The details of rating and recommendation module along with the flow was discussed.

Chapter 5 represented the implementation details along with the code of the main modules. Results and working of the app are described in chapter 6 along with the future scope. Various limitation of the work on which any researcher can work and improve the working of the app is also discussed.

An experiment is proposed in chapter 7 to evaluate the system which can be used to improve the work in future.

References

- [1] T. Lappas, C. Dellarocas, and N. Derakhshani, “Reputation and Contribution in Online Question-Answering Communities,” *SSRN Electron. J.*, no. Ericson, pp. 1–44, 2017.
- [2] I. Howley, G. 1S. Tomar, O. Ferschke, and C. P. Rose, “Reputation Systems Impact on Help Seeking in MOOC Discussion Forums,” *IEEE Trans. Learn. Technol.*, vol. 1382, no. c, pp. 1–14, 2017.
- [3] C. Dijkmans, P. Kerkhof, and C. J. Beukeboom, “A stage to engage: Social media use and corporate reputation,” *Tour. Manag.*, vol. 47, pp. 58–67, 2015.
- [4] J. Bao, Y. Zheng, and M. F. Mokbel, “Location-based and preference-aware recommendation using sparse geo-social networking data,” *Proc. 20th Int. Conf. Adv. Geogr. Inf. Syst. - SIGSPATIAL ’12*, no. c, p. 199, 2012.
- [5] R. L. Briones, B. Kuch, B. F. Liu, and Y. Jin, “Keeping up with the digital age: How the American Red Cross uses social media to build relationships,” *Public Relat. Rev.*, vol. 37, no. 1, pp. 37–43, 2011.
- [6] M. D. Assuncao et al., “Method and system for reputation evaluation of online users in a social networking scheme,” *U.S. Pat. No. 8,010,460*, 2011.
- [7] Y. Doytsher, B. Galon, and Y. Kanza, “Querying geo-social data by bridging spatial

- networks and social networks,” 2010 Int. Work. Locat. Based Soc. Networks, LBSN 2010, pp. 39–46, 2010.
- [8] T. Bhuiyan, Y. Xu, and A. Josang, “Integrating trust with public reputation in location-based social networks for recommendation making,” Proc. - 2008 IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol. - Work. WI-IAT Work. 2008, pp. 107–110, 2008.
- [9] A. Whitby, A. Jøsang, and J. Indulska, “Filtering out unfair ratings in bayesian reputation systems,” Chem. Biodivers., vol. 1, no. 11, pp. 1829–1841, 2005.
- [10] M. Kinaterder and S. Pearson, “A privacy-enhanced peer-to-peer reputation system,” Int. Conf. Electron. Commer. Web Technol., pp. 206–215, 2003.
- [11] J. D. Petty, J. N. Huckins, and A. David, “(12) Patent Application Publication (10) Pub . No .: US 2002/0187020 A1,” vol. 1, no. 19, 2002.
- [12] M. C. Angermeyer and H. Matschinger, “Whom to ask for help in case of a mental Disorder?,” Soc. Psychiatry Psychiatr. Epidemiol., vol. 34, no. 4, pp. 202–210, 1999.
- [13] L. Terveen et al., “PHOAKS: a system for sharing recommendations,” Commun. ACM, vol. 40, no. 3, pp. 59–62, 1997.
- [14] J. D. and W. Lewis A., “Trust as a social reality,” Soc. forces, vol. 63, no. 1, pp. 967–985, 1985.
- [15] P. Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara, “Reputation Systems,” Commun. ACM December 2000/Vol. 43, No. 12, vol. 43, no. 12, pp. 45–48, 2000.
- [16] A. M. Rashid et al., “Getting to Know You: Learning New User Preferences in

- Recommender Systems,” *Int. Conf. Intell. User Interfaces, IUI 2002*, pp. 127–134, 2002.
- [17] C. Jensen, J. Davis, and S. Farnham, “Finding others online: reputation systems for social online spaces,” *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, no. 1, pp. 447–454, 2002.
- [18] D. McDonald, “Recommending collaboration with social networks: a comparative evaluation,” *CHI ’03 Proc. SIGCHI Conf. Hum. factors Comput. Syst.*, no. 5, pp. 593–600, 2003.
- [19] S. Marti and H. Garcia-Molina, “Taxonomy of trust: Categorizing P2P reputation systems,” *Comput. Networks*, vol. 50, no. 4, pp. 472–484, 2006.
- [20] J. Audun, I. Roslan, and C. a. Boyd, “A Survey of Trust and Reputation Systems for Online Service Provision,” *Decis. Support Syst.*, vol. 43, no. 2, pp. 618–644, 2007.
- [21] J. Chen, W. Geyer, C. Dugan, M. Muller, and I. Guy, “Make new friends, but keep the old,” *Proc. 27th Int. Conf. Hum. factors Comput. Syst. - CHI 09*, no. May 2016, p. 201, 2009.
- [22] S. Scellato, C. Mascolo, M. Musolesi, and V. Latora, “Distance Matters: Geo-social Metrics for Online Social Networks,” *Proc. 3rd Conf. Online Soc. Networks*, pp. 1–8, 2010.
- [23] M. Jamali and M. Ester, “A matrix factorization technique with trust propagation for recommendation in social networks,” *Proc. fourth ACM Conf. Recomm. Syst. - RecSys ’10*, p. 135, 2010.

- [24] K. Väänänen-Vainio-Mattila, P. Saarinen, M. Wäljas, M. Hännikäinen, H. Orsila, and N. Kiukkonen, "User experience of social ad hoc networking: findings from a large-scale field trial of {TWIN}," Proc. 9th Int. Conf. Mob. Ubiquitous Multimed., pp. 1–10, 2010.
- [25] M. Moran, J. Seaman, and H. Tinti-Kane, "Teaching, Learning, and Sharing: How Today's Higher Education Faculty Use Social Media.," Babson Surv. Res. Gr., no. April, pp. 1–16, 2011.
- [26] C. Zhang, L. Shou, K. Chen, G. Chen, and Y. Bei, "Evaluating Geo-Social Influence in Location-Based Social Networks," 2012.
- [27] Q. Tang, B. Gu, and A. B. Whinston, "Content Contribution for Revenue Sharing and Reputation in Social Media: A Dynamic Structural Model," J. Manag. Inf. Syst., vol. 29, no. 2, pp. 41–76, 2012.
- [28] T. Atele-Williams and S. Marsh, "Information trust," IFIP Adv. Inf. Commun. Technol., vol. 505, pp. 221–222, 2017.
- [29] <http://www.quoteland.com/author/Dr-Loretta-Scott-Quotes/7735/>
- [30] Glynn, Liam G., Anne MacFarlane, Maureen Kelly, Peter Cantillon, and Andrew W. Murphy. "Helping each other to learn—a process evaluation of peer assisted learning." *BMC medical education* 6, no. 1 (2006): 18.
- [31] Resnick, Paul, and Richard Zeckhauser. "Trust among strangers in Internet transactions: Empirical analysis of eBay's reputation system." In *The Economics of the Internet and E-commerce*, pp. 127-157. Emerald Group Publishing Limited, 2002.

[32] <https://www.askalo.com/>

[33] <https://badoo.com/>

[34] <https://techcrunch.com/2010/01/08/blockchalk-location/>

[35] <http://buddycloud.com/>

[36] <http://www.carticipate.com/Home.html>

[37] <https://www.checkpoints.com/>

[38] www.wenear.com

[39] Anxiety and Depression Association of America. <https://adaa.org/understanding-anxiety/suicide>

[40] M.A.Patton and A. Jøsang. Technologies for Trust in E-Commerce. In Proceedings of the IFIP working conference on E-Commerce, Salzburg, Austria, June 2001[41] Gupta, Minaxi, Paul Judge, and Mostafa Ammar. "A reputation system for peer-to-peer networks." In *Proceedings of the 13th international workshop on Network and operating systems support for digital audio and video*, pp. 144-152. ACM, 2003.

[42] Thoms, Brian, Nathan Garrett, Jesus Canelon Herrera, and Terry Ryan. "Understanding the roles of knowledge sharing and trust in online learning communities." In Hawaii International Conference on System Sciences, Proceedings of the 41st Annual, pp. 3-3. IEEE, 2008.

- [43] Mayer, R. C., J. H. Davis, and F. D. Schoorman (1995) "An Integrative Model of Organizational Trust," *The Academy of Management Review* (20) 3, pp. 709-734.
- [44] Warneken, F., & Tomasello, M. (2006). Altruistic helping in human infants and young chimpanzees. *science*, 311(5765), pp1301-1303.
- [45] Overall, N. C., Fletcher, G. J., & Simpson, J. A. (2010). Helping each other grow: Romantic partner support, self-improvement, and relationship quality. *Personality and Social Psychology Bulletin*, 36(11), 1496-1513.
- [46] Oxford, R. L., & Green, J. M. (1996). Language Learning Histories: Learners and Teachers Helping Each Other Understand Learning Styles and Strategies. *TESOL journal*, 6(1), 20-23.
- [47] Trivedi, J. K. (2000). Relevance of ancient Indian knowledge to modern psychiatry. *Indian journal of psychiatry*, 42(4), 325.
- [48] Gilbert, P., Price, J., & Allan, S. (1995). Social comparison, social attractiveness and evolution: How might they be related?. *New ideas in Psychology*, 13(2), 149-165.
- [49] Isen, A. M., & Levin, P. F. (1972). Effect of feeling good on helping: cookies and kindness. *Journal of personality and social psychology*, 21(3), 384.
- [50] B. Thoms "A Dynamic Social Feedback System to Support Learning and Social Interaction in Higher Education," *IEEE Transactions on Learning Technologies*, Issue: 99, Mar. 2011.

- [51] Donath, J. S. "Identity and Deception in the Virtual Community," in *Communities in Cyberspace*, M. A. Smith and P. Kollock (Eds.), Routledge, New York, 1999, pp. 29-59.
- Goffin, RD and Anderson, DW (2007). "The self-rater's personality and self-other disagreement in multi-source performance ratings Is disagreement healthy?" *Journal of Managerial Psychology*, Vol. 22 No. 3, 2007 pp. 271-289, Emerald Group Publishing Limited.
- [52] Jones, C., Hesterly, W. S., and Borgatti, S. P. (1997). "A General Theory of Network Governance: Exchange Conditions and Social Mechanisms," *Academy of Management Review*, Vol. 22 No. 4, pp. 911-945.
- [53] Lo, S. and Lin, C. (2006). "WMR--A Graph-Based Algorithm for Friend Recommendation," *IEEE/WIC/ACM 2006 International Conference on Web Intelligence (WI'06)*, WI, pp. 121-128, 2006.
- [54] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J (2000). "Application of dimensionality reduction in recommender systems: A Case Study." *Proceedings of the WebKDD Workshop at the ACM SIGKDD*, Boston.
- [55] B. Thoms, N. Garrett, JC Herrera and T. Ryan, "Understanding the Roles of Knowledge Sharing and Trust in Online Learning Communities," *Hawaiian International Conference on System Sciences (HICSS 41)* Jan., 2008, Waikoloa, HI. (Best Paper Nomination)
- [56] Abdel-Hafez, A., Tang, X., Tian, N., & Xu, Y. (2014, December). A reputation-enhanced recommender system. In *International Conference on Advanced Data Mining and Applications* (pp. 185-198). Springer, Cham.

- [57] Jøsang, A., Haller, J.: Dirichlet Reputation Systems. Proceedings of the Second International. In: Proceedings of Conference on Availability, Reliability and Security, 112-119 (2007)
- [58] Schmidt, Bernd (2006). "Proof of principle studies". Epilepsy Research. 68 (1): 48–52. doi:10.1016/j.eplepsyres.2005.09.019. PMID 16377153.
- [59] Aeronautical Research : hearings before the United States House Committee on Science and Astronautics, Subcommittee on Advanced Research and Technology, Ninety-First Congress, first session. Washington, D.C. , USA: United States Senate, Ninetieth Congress, first session. 1969. December, 1, 2, 4, 8-11, 1969,
- [60] Davis, F. D. (1989), "Perceived usefulness, perceived ease of use, and user acceptance of information technology", MIS Quarterly, 13 (3): 319–340, doi:10.2307/249008, JSTOR 249008
- [61] Jhangiani, RS, Chiang, IA, Price, PC (2017). Research Methods in Psychology - 2nd Canadian Edition, BCcampus.

Appendix A: Code snippet for the application

A screenshot of the code showing the usage of firebase is attached below.

```
mAuth.signInWithEmailAndPassword(email, password)
    .addOnCompleteListener((new OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                // Sign in success, update UI with the signed-in user's
information
                Log.d(TAG, "signInWithEmail:success");
                FirebaseUser user = mAuth.getCurrentUser();
```

```

        progressDialog.dismiss();
        //updateUI(user);
    } else {
        // If sign in fails, display a message to the user.
        Log.w(TAG, "signInWithEmail:failure", task.getException());
        progressDialog.dismiss();

        if (String.valueOf(task.getException()).equals(error)) {

            emailEditText.setError("Invalid Email");
            Snackbar snackbar = Snackbar.make(findViewById(R.id.Login),
"No records found for this email",
            Snackbar.LENGTH_INDEFINITE).setAction("Create a new
account!", new
            View.OnClickListener() {
                @Override
                public void onClick(View view) {
                    Intent intent = new
Intent(getApplicationContext(), SignIn.class);
                    startActivity(intent);
                    finish();
                }
            });

            snackbar.show();
        } else if (String.valueOf(task.getException()).equals(PassError)) {

            passEditText.setError("Invalid Password!");

        } else {
            Snackbar.make(findViewById(R.id.Login),
String.valueOf(task.getException()),
            Snackbar.LENGTH_LONG).show();
        }
    }
    });
});

```

Figure A.1 Database used for helping sapiens is firebase (Login page)

The code corresponding to login page is given in figure A.1 where Email ID and password are used as the login credentials. The user is also provided with the option to create a new account of login using Google credentials. The option of login with Google credentials is also shown in figure A.2.

```

1. google.setOnClickListner(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        // Configure Google Sign In
        GoogleSignInOptions gso = new
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
            .requestIdToken(getString(R.string.default_web_client_id))
            .requestEmail()
            .build();
    }
});

```

```

        mGoogleSignInClient = GoogleSignIn.getClient(getApplicationContext(),
gso);

        signIn();
    }
});
2. private void signIn() {
    Intent signInIntent = mGoogleSignInClient.getSignInIntent();
    startActivityForResult(signInIntent, RC_SIGN_IN);
}

@Override
public void onStart() {
    super.onStart();
    // Check if user is signed in (non-null) and update UI accordingly.
    FirebaseUser currentUser = mAuth.getCurrentUser();
    updateUI(currentUser);
}

private void firebaseAuthWithGoogle(GoogleSignInAccount acct) {
    Log.d(TAG, "firebaseAuthWithGoogle:" + acct.getId());

    AuthCredential credential =
GoogleAuthProvider.getCredential(acct.getIdToken(), null);
    mAuth.signInWithCredential(credential)
        .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) {
                // Sign in success, update UI with the signed-in
user's information

                Log.d(TAG, "signInWithCredential:success");
                FirebaseUser user = mAuth.getCurrentUser();

                String Uid = user.getId();

                // Snackbar.make(findViewById(R.id.Login), "Success
with UID: " + Uid, Snackbar.LENGTH_LONG).show();

                Intent intent = new
Intent(getApplicationContext(), UserInfo.class);
                startActivity(intent);
                finish();
                // updateUI(user);
            } else {
                // If sign in fails, display a message to the
user.

                Log.w(TAG, "signInWithCredential:failure",
task.getException());

                Snackbar.make(findViewById(R.id.Login),
String.valueOf(task.getException()), Snackbar.LENGTH_SHORT).show();
                updateUI(null);
            }

            // ...
        }
    });
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data)

```

```

{
    super.onActivityResult(requestCode, resultCode, data);

    // Result returned from launching the Intent from
    GoogleSignInApi.getSignInIntent(...);
    if (requestCode == RC_SIGN_IN) {
        Task<GoogleSignInAccount> task =
        GoogleSignIn.getSignedInAccountFromIntent(data);
        try {
            // Google Sign In was successful, authenticate with Firebase
            GoogleSignInAccount account =
            task.getResult(ApiException.class);
            firebaseAuthWithGoogle(account);
        } catch (ApiException e) {
            // Google Sign In failed, update UI appropriately
            Log.w(TAG, "Google sign in failed", e);
            // ...
        }
    }
}
}

```

Figure A.2 Login with Google credential

If the user credentials are not found in the database, then user is also given the option to create a new account as shown in figure A.3. Here the user is asked to enter the email ID and password and retype the password. The required checks are performed for the valid email ID. Also some checks for the password is also applied, these checks can be altered further as per the requirement of the application and developer.

```

3. Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_sign_in);

    emailEditText = (EditText)findViewById(R.id.SigninEmail);
    PassEditText = (EditText)findViewById(R.id.SigninPass);
    ConfPass = (EditText)findViewById(R.id.SigninConfPass);

    mAuth = FirebaseAuth.getInstance();

    Submit = (Button)findViewById(R.id.SignInButton);
    Submit.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {

            email = emailEditText.getText().toString().trim();
            pass = PassEditText.getText().toString().trim();
            confPass = ConfPass.getText().toString().trim();

            if (!email.isEmpty()) {
                if (!email.matches("[a-zA-Z0-9._-]+@[a-zA-Z]+.[a-z]+")) {
                    emailEditText.setError("Invalid format");
                } else {

```



```

        if (!pass.isEmpty()) {
            if (!confPass.isEmpty()) {
                Validate();
            } else {
                ConfPass.setError("Password cannot be empty!");
            }
        } else {
            PassEditText.setError("Password cannot be empty!");
        }
    }
} else {
    emailEditText.setError("Email cannot be empty");
}
});
}

public void Validate() {

    email = emailEditText.getText().toString().trim();
    pass = PassEditText.getText().toString().trim();
    confPass = ConfPass.getText().toString().trim();

    if (pass.length() > 6){
        if (pass.equals(confPass)){

            final ProgressDialog progressDialog = new ProgressDialog(this);
            progressDialog.setMessage("Please wait...");
            progressDialog.setCanceledOnTouchOutside(false);
            progressDialog.setCancelable(false);
            progressDialog.show();

            mAuth.createUserWithEmailAndPassword(email, pass)
                .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task)
{
                            if (task.isSuccessful()) {
                                // Sign in success, update UI with the signed-
in user's information

                                FirebaseUser user = mAuth.getCurrentUser();
                                progressDialog.dismiss();
                                Intent intent = new
Intent(getApplicationContext(), UserInfo.class);
                                startActivity(intent);
                                finish();

                            } else {
                                // If sign in fails, display a message to the
user.

                                progressDialog.dismiss();
                                Log.w(TAG, "createUserWithEmail:failure",
task.getException());

                                Snackbar.make(findViewById(R.id.SignIn), String.valueOf(task.getException()), Sn
ackbar.LENGTH_LONG).show();

                            }
                        }
                    });
        } else {
            ConfPass.setError("Passwords do not match");
        }
    }
}
}

```

```

    }
    } else {
        PassEditText.setError("Increase password length");
    }
}

```

Figure A.3 *Creating a new user account*

The below figure A.4 the code for updating the various fields for a new user is presented. Here a unique firebase token for each user is generated. Also, whenever a user is asking for some help her latitude and longitude are extracted so that other users (help providers) in the vicinity can be informed.

```

4.     @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_user_info);

            username = (EditText) findViewById(R.id.UserName);
            usernumber = (EditText) findViewById(R.id.UserNumber);
            submit = (Button) findViewById(R.id.UserSubmit);
            LocationText = (TextView) findViewById(R.id.SelectedLocationText);
            edit1 = (ImageView) findViewById(R.id.edit1);
            UserMap = (Button) findViewById(R.id.UserMap);
            helpText = (TextView) findViewById(R.id.SelectedHelpText);
            edit = (ImageView) findViewById(R.id.edit);
            userHelp = (Button) findViewById(R.id.UserHelp);

            databaseReference =
            FirebaseDatabase.getInstance().getReference().child("UserInfo");
            currentUser = FirebaseAuth.getInstance().getCurrentUser();
            Uid = currentUser.getUid();

5.
6.     //to generate a unique firebase token for each user
7.

            FirebaseInstanceId.getInstance().getInstanceId().addOnCompleteListener(new
            OnCompleteListener<InstanceIdResult>() {
                @Override
                public void onComplete(@NonNull Task<InstanceIdResult> task) {
                    if (!task.isSuccessful()) {
                        Log.d("TokenErrorIs:
", String.valueOf(task.getException()));
                    }

                    token = task.getResult().getToken();
                }
            });

            Intent intent = getIntent();

            if (intent.hasExtra("Name") || intent.hasExtra("Number") ||
            intent.hasExtra("LatLong") || intent.hasExtra("Selections")) {

                getName = getIntent().getExtras().getString("Name");
                getNumber = getIntent().getExtras().getString("Number");

                LatLong = getIntent().getExtras().getString("LatLong");

```

```

        getSelections = getIntent().getExtras().getString("Selections");

        replaced = LatLong.replaceAll("[Latlng: (,)]", "");

        if (!getNumber.equals("0")) {
            username.setText(getName);
        }

        if (!getNumber.equals("0")) {
            usernumber.setText(getNumber);
        }

        if (!getSelections.equals("0")) {
            helpText.setText(getSelections);
        }
8.
9. //Getting the current location of user

        String[] cordinates = replaced.split("\\s+");
        String lat1 = cordinates[0];
        String long1 = cordinates[1];

        latitude = Double.parseDouble(lat1);
        longitude = Double.parseDouble(long1);

        geocoder = new Geocoder(this, Locale.getDefault());
        try {
            addresses = geocoder.getFromLocation(latitude, longitude, 1);

            address = addresses.get(0).getAddressLine(0);
            LocationText.setText(address);
            LocationText.setVisibility(View.VISIBLE);
            UserMap.setVisibility(View.INVISIBLE);
            edit1.setVisibility(View.VISIBLE);

        } catch (IOException e) {
            e.printStackTrace();
        }

    }

    edit.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            selectedHelp.clear();
            userHelp.setVisibility(View.VISIBLE);
            helpText.setVisibility(View.INVISIBLE);
            edit.setVisibility(View.INVISIBLE);
        }
    });

    UserMap.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            name = username.getText().toString().trim();
            number = usernumber.getText().toString().trim();
            selections = helpText.getText().toString().trim();

            Intent intent = new Intent(getApplicationContext(),
Maps.class);

            intent.putExtra("Name", name);
            intent.putExtra("Number", number);
            intent.putExtra("Selections", selections);
            startActivity(intent);

```

```

        finish();
    }
});

userHelp.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        AlertDialog.Builder builder = new
AlertDialog.Builder(UserInfo.this, R.style.MyDialogTheme);

        builder.setTitle("Choose your Help Type");

        final String[] helpType = {"Builders", "Computers",
"Electrician", "Entertainment", "Groceries", "House Moving",
        "Plumbing", "Travel", "Tutor"};

        boolean[] checkedItems = { false, false, false, false, false,
false, false, false, false};

        builder.setMultiChoiceItems(helpType, checkedItems, new
DialogInterface.OnMultiChoiceClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int
i, boolean b) {

                if (b){
                    selectedHelp.add(helpType[i]);
                } else if (selectedHelp.contains(helpType[i])){
                    selectedHelp.remove(helpType[i]);
                }

            }

        });

        builder.setPositiveButton("OK", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int
i) {

                userHelp.setVisibility(View.INVISIBLE);
                helpText.setVisibility(View.VISIBLE);
                edit.setVisibility(View.VISIBLE);
                String data =
String.valueOf(selectedHelp).replaceAll("\\[", "").replaceAll("\\]",
                "");
                helpText.setText(String.valueOf(data));

                dialogInterface.dismiss();

            }

        });

        builder.setNegativeButton("Cancel", null);

        AlertDialog dialog = builder.create();
        dialog.show();

    }

});
10.
11. //Saving the values in the database

        submit.setOnClickListener(new View.OnClickListener() {

```

```

@Override
public void onClick(View view) {

    if (validate()){

        String Name = username.getText().toString().trim();
        String Number = usernumber.getText().toString().trim();
        String LatLon = String.valueOf(replaced);

        int size = selectedHelp.size() - 1;

        for (int i = 0;i < selectedHelp.size();i++){

            dbRef =
databaseReference.child(selectedHelp.get(i)).child(Uid);

            dbRef.child("Name").setValue(Name);
            dbRef.child("Number").setValue(Number);
            dbRef.child("LatLon").setValue(LatLon);
            dbRef.child("HelType").setValue(selectedHelp.get(i));
            dbRef.child("Token").setValue(token);

            if (i == size){

                SharedPreferences.Editor editor =
getSharedPreferences("MY_PREFS",MODE_PRIVATE).edit();

                editor.putString("ChildNode",String.valueOf(selectedHelp.get(i)));
                editor.apply();

                Intent intent1 = new
Intent(getApplicationContext(),MainActivity.class);
                startActivity(intent1);
                finish();

            }

        }

    }

});

edit1.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
    UserMap.setVisibility(View.VISIBLE);
    edit1.setVisibility(View.INVISIBLE);
    LocationText.setVisibility(View.INVISIBLE);
    LocationText.setText("0");
}
});

}

public boolean validate() {

    boolean result = false;

    String n , no;

    n = username.getText().toString().trim();
    no = usernumber.getText().toString().trim();

    if (!n.isEmpty()){
        if (!no.isEmpty()){
            if (!(LocationText.getText().toString().trim()).equals("0")){
                if (!selectedHelp.isEmpty()){

```

```

        result = true;
    } else {
        Snackbar.make(findViewById(R.id.UserInfo), "Please
Select your Help type", Snackbar.LENGTH_LONG).show();
    }
    } else {
        Snackbar.make(findViewById(R.id.UserInfo), "Please choose
your Location", Snackbar.LENGTH_LONG).show();
    }
    } else {
        usernumber.setError("Please fill your number");
    }
} else {
    username.setError("Please fill your full name");
}
return result;
}
}
}

```

Figure A.4 Updating user information in the database

Figure A.5 represents the various categories: Builders, Computers, Electrician, Entertainment, Groceries, House Moving, Plumbing, Travel, and Tutor. The categories can be further added or removed depending on the requirement. The help seeker can choose any of the above category and the user will be provided the list providers in the vicinity.

```

userRef.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        for (DataSnapshot snapshot : dataSnapshot.getChildren()) {

            MainPageAdapter items = snapshot.getValue(MainPageAdapter.class);

            recommendations.add(items.getName() + "#" + items.getRating() + "#" +
items.getUid() + "#" + items.getLatlon());

            loop = true;
        }

        if (loop){

            for (int i = 0; i < recommendations.size(); i++) {
                String all = recommendations.get(i);
                String[] SplitAll = all.split("#");

                String newLatlon = SplitAll[3];

                String SplitOldLatlon[] = latlong.split(" ");

                Double OldLat = Double.parseDouble(SplitOldLatlon[0]);
                Double OldLon = Double.parseDouble(SplitOldLatlon[1]);

                String SplitNewLatlon[] = newLatlon.split(" ");

                Double NewLat = Double.parseDouble(SplitNewLatlon[0]);
                Double NewLon = Double.parseDouble(SplitNewLatlon[1]);

                double Theta = NewLon - OldLon;
            }
        }
    }
}

```

```

        double dist = Math.sin(deg2rad(NewLat))
            * Math.sin(deg2rad(OldLat))
            + Math.cos(deg2rad(NewLat))
            * Math.cos(deg2rad(OldLat))
            * Math.cos(deg2rad(Theta));

        dist = Math.acos(dist);
        dist = rad2deg(dist);
        dist = dist * 60 * 1.1515;

        if (dist < UserRadius){

            recommendation.add(recommendations.get(i));

        }

        loop2 = true;

    }

} if (loop2){

    listString = String.valueOf(recommendations);

    listString = listString.replaceAll("\\\\[", "\\").replaceAll("\\\\]", "");

    String[] SplitList = listString.split(",");

    for (int i = 0; i < SplitList.length; i++){

        // Log.i("Data1: ", SplitList[i]);

        //dummyData.add(SplitList[i]);

        String[] SplitDummy = SplitList[i].split("#");

        String ReqData = SplitDummy[1];
        Log.i("RequiredData ", ReqData);

        String[] SplitReqData = ReqData.split("/");

        for (int j = 0; j < SplitReqData.length; j++){

            String[] SplitFinal = SplitReqData[j].split("@");

            Log.i("FinalValues: ", SplitFinal[0]);
            sum = sum + Double.parseDouble(SplitFinal[0]);

        }

        sum = sum / Double.parseDouble(String.valueOf(SplitReqData.length));

        String Sum = String.format("%.1f", sum);

        dummyData.add(String.valueOf(Sum));
        sum = 0;
    }
}

```

```

        loop3 = true;
    }

    if (loop3) {

        for (int i = 0; i < dummyData.size() - 1; i++) {

            int j = i + 1;

            if (Double.parseDouble(dummyData.get(i)) <
Double.parseDouble(dummyData.get(j))) {

                temp = recommendations.get(i);
                recommendations.set(i, recommendations.get(j));
                recommendations.set(j, temp);

            }

        }

        Log.i("dummyData ", String.valueOf(dummyData));
        Log.i("Recommendations", String.valueOf(recommendations));

    }

    Recommendations adapter = new Recommendations(getApplicationContext(),
recommendations, Key, title.getText().toString().trim(),
                body.getText().toString().trim(), typeH);

    recyclerView = (RecyclerView) findViewById(R.id.MainPageRecyclerView);
    RecyclerView.LayoutManager layoutManager = new
LinearLayoutManager(getApplicationContext(), LinearLayoutManager.HORIZONTAL, false);
    recyclerView.setLayoutManager(layoutManager);
    recyclerView.setItemAnimator(new DefaultItemAnimator());
    recyclerView.setHasFixedSize(true);
    recyclerView.setAdapter(adapter);

    }
}

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {

}

});

```

Figure A.5 Various categories of users in help providers role

The below figure A.6 represents the code for filtering the homepage based on the location of the user who is seeking help.

```

final DatabaseReference databaseReference =
FirebaseDatabase.getInstance().getReference().child("Questions");

databaseReference.addValueEventListener(new ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
        for (DataSnapshot snapshot : dataSnapshot.getChildren()){
            MainConstructor items = snapshot.getValue(MainConstructor.class);

            allList.add(items.getTitle()+"#" +items.getBody()
+"#" +items.getDate()+"#" +items.getDays()+"#" +items.getHelpType()
            +"#" +items.getHour()+"#" +items.getKey()
+"#" +items.getLatLng()+"#" +items.getUID());

            loop = true;
        }

        if (loop = true){

            for (int i = 0; i < allList.size(); i++) {
                String all = allList.get(i);
                String[] SplitAll = all.split("#");

                String newLatlon = SplitAll[7];

                String SplitOldLatlon[] = LatLong.split(" ");

                Double OldLat = Double.parseDouble(SplitOldLatlon[0]);
                Double OldLon = Double.parseDouble(SplitOldLatlon[1]);

                String SplitNewLatlon[] = newLatlon.split(" ");

                Double NewLat = Double.parseDouble(SplitNewLatlon[0]);
                Double NewLon = Double.parseDouble(SplitNewLatlon[1]);

                double Theta = NewLon - OldLon;
                double dist = Math.sin(deg2rad(NewLat))
                    * Math.sin(deg2rad(OldLat))
                    + Math.cos(deg2rad(NewLat))
                    * Math.cos(deg2rad(OldLat))
                    * Math.cos(deg2rad(Theta));

                dist = Math.acos(dist);
                dist = rad2deg(dist);
                dist = dist * 60 * 1.1515;

                if (dist < HelpLocation){

                    finalList.add(allList.get(i));

                }

                loop2 = true;

            }

            if (loop2 = true){

                Collections.reverse(finalList);
            }
        }
    }
});

```

```

        homeAdapter adapter = new homeAdapter(getContext(),
finalList);

        RecyclerView recyclerView =
(RecyclerView) view.findViewById(R.id.HomeRecyclerView);
        LinearLayoutManager layoutManager1 = new
LinearLayoutManager(getContext());
        recyclerView.setLayoutManager(layoutManager1);
        recyclerView.setItemAnimator(new DefaultItemAnimator());
        recyclerView.setHasFixedSize(true);
        recyclerView.setAdapter(adapter);
        recyclerView.getViewTreeObserver().addOnGlobalLayoutListener(new
ViewTreeObserver.OnGlobalLayoutListener() {
            @Override
            public void onGlobalLayout() {
                progressDialog.dismiss();

                //
recyclerView.getViewTreeObserver().removeOnGlobalLayoutListener((ViewTreeObserver.OnGl
obalLayoutListener) getContext());
            }
        });
    }
}

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
}
});

}

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
}
});
});

```

Figure A.6 The list is filtered again using location which user provides for help

As shown below in figure A.7, whenever any help is done, the user who posted the help gets to rate the user who helped. This is done by storing the rating in helping users profile on database. Every new element is stored as a new entry in the array. Whenever a user's rating is to be shown, the sum of the entire array is calculated and divided by the size of the array giving out the average rating of the user.

Due to the fact that for each help type user has a different profile. This ensures that the rating is different for the same user working under different help types. This will be helpful for the help seeker in getting the best match for the area in which she is seeking help.

```

final Dialog rankDialog = new Dialog(MainPage.this, R.style.dialog);
rankDialog.setContentView(R.layout.rank_dialog);
rankDialog.setCancelable(true);
RatingBar Communication = (RatingBar)rankDialog.findViewById(R.id.dialog_ratingBar1);
Communication.setRating(0);
Communication.setNumStars(5);
Communication.setMax(5);

Communication.setOnRatingBarChangeListener(new RatingBar.OnRatingBarChangeListener() {
    @Override
    public void onRatingChanged(RatingBar ratingBar, float v, boolean b) {
        Rating1 = String.valueOf(v);
    }
});

RatingBar Generosity = (RatingBar)rankDialog.findViewById(R.id.dialog_ratingBar2);
Generosity.setRating(0);
Generosity.setNumStars(5);
Generosity.setMax(5);

Generosity.setOnRatingBarChangeListener(new RatingBar.OnRatingBarChangeListener() {
    @Override
    public void onRatingChanged(RatingBar ratingBar, float v, boolean b) {
        Rating2 = String.valueOf(v);
    }
});

RatingBar Punctuality = (RatingBar)rankDialog.findViewById(R.id.dialog_ratingBar3);
Punctuality.setRating(0);
Punctuality.setNumStars(5);
Punctuality.setMax(5);

Punctuality.setOnRatingBarChangeListener(new RatingBar.OnRatingBarChangeListener() {
    @Override
    public void onRatingChanged(RatingBar ratingBar, float v, boolean b) {
        Rating3 = String.valueOf(v);
    }
});

RatingBar Courteous = (RatingBar)rankDialog.findViewById(R.id.dialog_ratingBar4);
Courteous.setRating(0);
Courteous.setNumStars(5);
Courteous.setMax(5);

Courteous.setOnRatingBarChangeListener(new RatingBar.OnRatingBarChangeListener() {
    @Override
    public void onRatingChanged(RatingBar ratingBar, float v, boolean b) {
        Rating4 = String.valueOf(v);
    }
});

RatingBar Skill = (RatingBar)rankDialog.findViewById(R.id.dialog_ratingBar);
Skill.setRating(0);
Skill.setNumStars(5);
Skill.setMax(5);

Skill.setOnRatingBarChangeListener(new RatingBar.OnRatingBarChangeListener() {
    @Override
    public void onRatingChanged(RatingBar ratingBar, float v, boolean b) {
        Rating5 = String.valueOf(v);
    }
});

```

```

Button submit = (Button)rankDialog.findViewById(R.id.dialog_submit);
submit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        Double rate = Double.parseDouble(Rating1) + Double.parseDouble(Rating2) +
Double.parseDouble(Rating3)
            + Double.parseDouble(Rating4) + Double.parseDouble(Rating5);

        rate = rate/5;

        Rating = String.valueOf(rate);

        rankDialog.dismiss();

        postRef =
FirebaseDatabase.getInstance().getReference().child("UserInfo").child(typeH).child(req
Uid);

        postRef.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                if (dataSnapshot.hasChild("Rating")){
                    MainConstructor item =
dataSnapshot.getValue(MainConstructor.class);

                    String R = item.getRating();

                    Date currentDate = Calendar.getInstance().getTime();
                    String Date = String.valueOf(currentDate);

                    String CurrentRating = Rating + "@" + dataUID + "@" + Date;

                    CurrentRating = R + "/" + CurrentRating;

                    postRef.child("Rating").setValue(CurrentRating);

                } else {

                    Date currentDate = Calendar.getInstance().getTime();
                    String Date = String.valueOf(currentDate);

                    postRef.child("Rating").setValue(Rating+"@"+dataUID+"@"+Date);
                }
            }

            @Override
            public void onCancelled(@NonNull DatabaseError databaseError) {

            }
        });
    }
});
rankDialog.show();

```

Figure A.7 *The module for providing a rating after the help has been provided*

Future Contribution: <https://github.com/simrandeep10/HelpingSapiensAndroid> This link can be used for future contribution and improvement of the work.