A STUDY ON USING CONTEXTUAL INFORMATION TO MODERATE MOBILE NOTIFICATIONS

Wengappuli Arachchige Charitha Sasanka De Silva

168215H

Dissertation submitted in partial fulfilment of the requirements for the degree Master of Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa
Sri Lanka

March 2020

DECLARATION

I, declare that this is my own work and this MSc Research Project Report does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works.

....

W A C S De Silva Date

I certify that the declaration above by the candidate is true to the best of my knowledge and that this project report is acceptable for evaluation for the CS6997 MSc Research Project.

....

Dr. Indika Perera Date

ABSTRACT

At present, people use their smart devices (phones, phablets, tablets, watches, glasses etc.) for much more than mere communication. People just do not send text messages, make calls, or send e-mails. They attend to their bank activities, check the weather information, involve in the stock market trading, and even control their home lighting system using their smart devices. Hence the user experience of a smart device has become more of a personalized experience. "Push Notifications" is a method that the user communicates with smart devices. Push Notifications can range from an alarm alert to weather alert. However, with the increasing number of apps the number of notifications received by a typical user per day had increased several folds in the past few years.

In this research, how such notifications were moderated is discussed. As these notifications cannot be moderated in a generic manner, the users' requirements and preferences were considered before being moderated. How a typical user reacted to different kinds of situations was first analysed and users' preferences were considered before the moderation happened. To develop this notification system, current research regarding analysing human behaviour using contextual information gathered from smart devices along with research related to the moderation of Push Notifications were studied. Then using the current research information and the mostly used contextual information were used to create a profile for each individual user and to moderate notifications.

In summary, the research discussed how the notifications were moderated based on different criteria and how user behaviours were useful in doing so. It further discussed the performance and security issues which arose when implementing the solution and the methods used to overcome such boundaries. It also discussed on the architectural aspect of the whole solution. The solution has the capability to moderate incoming notifications based on user preferences and behaviours as well as the notification content while using the abilities of several 3rd party web services.

Keywords: Mobile, Notifications, Context

ACKNOWLEDGEMENTS

I am thankful to Dr. Indika Perera, my supervisor for the MSc Level Research and Dr. Malaka Walpola, my supervisor for the PGDip Level Research. Under Dr. Indika Perera's supervision, guidance, continuous support, I was able to make this research a success.

My sincere appreciation goes to my family and my fiancée for the support and the motivation given throughout this journey and helping out when I am in need and motivating me to complete the research.

I would also like to thank my colleagues at my workplace, IFS (Global Support Organization) for spending their valuable time with me to discuss about my research and factoring in new ideas.

Finally, I wish to thank all the academic and non-academic staff of Department of Computer Science and Engineering, University of Moratuwa and my colleagues of MSC16 batch for the support and encouragement provided throughout the last 4 years.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
AMS	Active Map Service
API	Application Programming Interface
CSV	Comma-Separated Value
GPS	Global Positioning System
HTML	Hypertext Mark-up Language
HTTP	Hyper Text Transfer Protocol
IT	Information Technology
JSON	JavaScript Object Notation
ML	Machine Learning
NFC	Near-Field Communication
NoSQL	Not only SQL
PDA	Personal Digital Assistant
POP	Personalized Ontology Profile
QR	Quick Response
REST	Representational State Transfer
RFID	Radio-Frequency Identification
RMS	Root-Mean Square
SDK	Software Development Toolkit
SMS	Short Message Service
URI	Uniform Resource Identifier
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
XML	Extensible Mark-up Language

CHAPTER 1 - INTRODUCTION

With the dawn of the mobile era, the computer has become not only a workplace equipment, it has now become a place an end user constantly interacts with, in some way or the other. From the time a person wakes up in the morning (due to an alarm set in the mobile), to check the current weather (services like AccuWeather) and traffic conditions (services like Google Maps, Bing Maps etc.) until the time the person goes to sleep, this interaction continues in one form or the other. As mobile devices support a wide range of sensors and technologies, compared to a traditional desktops or laptops, the possibilities have become endless.

From a technological point of view, the mobile shifts the paradigm of computing from a more static to a more dynamic level. Hence the new paradigm demands new architectural styles, new technological knowledge to cope up with the challenges and opportunities. The information that can be harnessed by mobile devices are higher and far more valuable than that of desktop computers due to its multiple contextual setup.

With these kind of powers at hand, the analysis of human behaviour has become a more approachable topic now. Super computers, having the power to learn the human behaviour from a complex data setup and having the ability to mimic a human behaviour albeit in a single process have become possible due to the evolution of mobile computing.

However, a simple end user might not need such sophisticated technologies in their day to day life activities, the contextual data gathered from the mobile devices can be used to make the mobile experience better. Contextual information like, location, current activity, mobile phone mode (muted, loud etc.) can be some minor details which could in turn help the end user to interact with the mobile in a much simpler manner.

1.1 - Push Notifications

The term “Push Notification” comes along with the smart device era, where a notification appears on the topmost or middle part of the mobile screen to display a small detail of information (i.e. a part of a text message, a missed call alert etc.). Nowadays, every application in all major mobile platforms (Android, iOS etc.) use Push Notifications to communicate information to the end user.

A typical Push Notification occurs when an event is triggered from an application or the mobile operating system itself. Hence the user gets notified when a specific event is triggered.

1.2 - Context Aware Notifications

Context aware notifications are notifications that triggers due to a contextual information change. The major, most widely used aspect of context aware notifications are the use of location-based weather alerts and review alerts.

Nowadays, social media sites like Google, Facebook routinely inform the user for the weather corresponding to their current location. This is to alert the user to be ready for any intermittent weather changes. This is enhancing the knowledge of the location information obtained from the mobile device to provide the user with a more proactive notification.

1.3 - Notifications Moderated Using Contextual Information, Current User Activity and Human Behaviour

Notifications moderated using both contextual information and current user activity can be classified as a different set of notifications and hence can be called as “Smart

Notifications". They can be both context aware and user interaction aware. In a scenario where a user is interacting with a mobile device and if the user prefers not to be bothered during this interaction a simple push notification would not suffice. In such a scenario, the smart device must understand the importance of the interaction and delay or hide the notification.

Furthermore, in such a scenario, where the user is expecting a notification even while attending to a separate task, then the device must understand the notification must immediately pop-up as it should. How to identify the dos and do nots is the key factor in this research.

To identify the user preferences, first the notification system needs to identify the user and profile him or her according to the preferences of the user and any basic knowledge about his or her likes and dislikes.

Hence in order to carefully moderate the notifications incoming for the user, along with the users' preferences, the current contextual and activity information needed to be checked and the moderation should happen according to the decision gathered after considering all the factors.

These are some example scenarios that can be possible with the analysis of the user behaviour using the mobile context information.

1.4 - Human Behaviour Mapping

The human behaviour is varying and could take any form within a given period. Hence mapping user behaviour and emotions is not a simple practical application. Even with using a multiple sensory setup, with hundreds of servers analysing each set of data that comes with complex algorithms and machine-learning, even one cannot simply decode

the human mind completely. However, a mere reaction to an event is something that can be identified through a simple setup. The contextual information of a mobile device would be sufficient to handle such a scenario.

There has been multiple research done in order map the human behaviour using mostly location information from Cellular towers and GPS coordinates. Furthermore, there are instances where the mobile is trained through machine learning to be smart enough to identify the human emotions. These are all possible scenarios that could be counted into mapping the user behaviour through the mobile devices' contextual information.

1.5 - Motivation

Smart devices are advancing in leaps and bounds compared to desktop computers. With smart devices comes several different capabilities that the typical desktop user couldn't do. Hence the smart device experience should also be customizable and configurable for every user and should not interfere negatively with the user mobile interaction. It should also be noted that in the event that mobile sensors are being used excessively in the background, the battery drainage would be such that the mobile would be in an always charging state. However, the system was able to cope up with such challenges as well. In that note, how to moderate notifications depending on the current context, current user activity and user preferences was studied through this research [1].

1.6 - Problem Statement

- As the usage of smart devices are on the rise and increasingly new technologies are integrated to smart devices, the need of notification moderation is necessary
- The push notification moderation should depend on the user preferences as well as information gathered from the multi-sensory setup of the smart device along

with the use of several 3rd party services or Application Programming Interfaces (APIs)

- With the use of contextual information that were gathered from the mobile devices' sensor network alongside 3rd party services the users' behavioural patterns were analysed and mapped

1.7 - Objectives

- Create a research concept and a suitable software project to moderate push notifications depending on the user preferences, current context and activity data
- Provide a mechanism to identify the user behaviours using contextual information from a smart device
- Classify the user behaviour depending on the contextual information and the current activity of the user
- Create a profile for the user depending on the information gathered

1.8 - Outcomes

A research project report along with a sample mobile application that analyse the user preferences and along with the users' current contextual information gathered from a multiple sensory setup and using 3rd party services to get the current user activity which in turn created a user profile that accommodated to provide the user the ability of moderating the incoming push notifications.

CHAPTER 2 - LITERATURE REVIEW

In the era of Ubiquitous Computing where a computer is made to appear anytime and everywhere, mobile devices are in the forefront in implementing this concept as mobile devices are vastly different to the desktop computers in ways that were remotely possible 2 - 3 decades back.

Users with desktop computers interact with it using the typical input and output devices whereas mobile devices take human computer interaction to a completely different level, using touch screen gestures, built in sensors like gyroscope, accelerometer, proximity sensor etc. Furthermore, security sensors like facial recognition, fingerprint recognition and iris scanners have highlighted this ability of mobile devices even more. Nowadays, people can configure their credit card to the mobile device and make payments without even ever touching it again using the security sensors.

Hence, the ability of mobile devices and usage of highly intelligent, highly reliable sensor network, the users can interact with their computers in a brand of ways.

With the introduction of multiple sensors built inside a mobile device, the usages of data output from such sensors have become limitless. The idea of Context Aware Computing is built upon this concept.

2.1 - Context and Context Awareness

The definition of Context is not a topic that is iron clad. Hence, many authors from different Universities and Institutes have come up with different definitions of Context. However, the most comprehensive definition of Context is defined by Schilit et al. [1] stating the following:

A context is defined through the 3 aspects:

1. Where you are?
2. Who are you with?
3. What are the resources nearby?

The definition of Context is introduced by constantly changing the execution environment. The environments can be defined as follows:

- Computing Environment where the devices, processors, networks etc. reside
- User Environment where the location of the user and the current social situation
- Physical Environment is what defined by the sensors built with such as light and noise level

Dey et al. [2] combines all the definitions of Context and comes up with a simple yet concise definition:

> *"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves."*

Furthermore, this paper discusses the categories of context that can be identified. In each scenario, the location of the user, any users nearby, the time and what other devices are nearby can be considered used to define the context. In other words, the user being in the same place in 2 different times can be considered as 2 different contexts. For example, if a person is connected to their home Wi-Fi network within a working day then it can be considered that the user is taking a leave of absence. It can be a pre-planned leave or due to a sickness where the context can be vastly different. The same scenario but a different

period would make the context normal. The way a system should act in 2 different contexts albeit in the same location with the same users can be different. This can be considered as Context Awareness.

Dey et al. [2] also identifies the main elements of Context and defines Context Awareness as the following:

> *"A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task."*

Hence, in the above-mentioned example, in the scenario where the user is connected to the home Wi-Fi network during the morning time the mobile device should be able to identify that this is a differentiation from the typical scenario and would in turn change the typical interactions with the user accordingly.

2.2 - Context Aware Systems

The first idea of Context Aware Systems by Schilit et al. [3] was put forward in 1994 in which it discusses the concept of an Active Map where a user using a mobile device can understand the changes of objects in the environment in which the user interacts with. The service which is used to understand such behaviour is the Active Map Service (AMS). The authors discuss the possibility of the mobile device being able to understand using the device location what the user is about to do and act appropriately.

However, in such cases there were limitations of the network bandwidth limit, scalability of the Active Map Service etc. Hence, the idea of Context Aware Systems was disclosed to a given environment (such as an office premises) where a mobile device can be used to locate a given person and act accordingly.

With the evolution of smart mobile devices, the Context Awareness as broken its primary concerns of scalability and network bandwidth usages. Currently there are possibilities that users can connect to each other's devices rather than a server that is far away to accomplish the same goal. Hence this would limit the network traffic coming towards and from the main server. This is possible due to the technologies such as Bluetooth, Near-Field Communication (NFC), Global Positioning System (GPS) etc.

2.3 - Applications of Context Aware Systems

2.3.1 - Location Based Applications

Want et al. [4] discusses the idea around an Active Badge System where users belonging to a certain organization using a badge that transmits signals which they carry around them wherever they go around the premises can be used to recognize their location using a network of sensors. The data gathered through the sensors is processed by a central server which can identify the exact location a person is in each moment.

This concept can be used to locate staff in a highly responsive organization such as a hospital where locating a doctor in an emergency within a large hospital can be a hassle.

2.3.2 - Healthcare Based Applications

With the evolution of smart watches and wearables there are high number of applications that can be used in the healthcare sector for Context Aware Computing. A doctor can consult the records of a patients' heart rate during active hours, inactive hours to check for any irregularities to find out whether the patient has a potential of a heart failure.

2.3.3 - Manufacturing Based Applications

In a manufacturing process there are numerous components that come together when creating one final product. These components are either created or bought from suppliers. When the process is ongoing and if one component is getting low in quantity the system can automatically be able to detect and send a notification to the relevant parties regarding this situation.

This can be used to further enhance the process as there will be no delay in the components as the personnel handling the process is aware of any delays beforehand. This can be accomplished by scanning of a Quick Response (QR) code or Radio-Frequency Identification (RFID) tag when the component is about to be used. If the used amount is nearing its limit in stock, a notification can be triggered.

Such are some applications of Context Aware Computing. Hence through the evolution of smart devices its applications are increasing rapidly.

2.4 - Using Contextual Information to Identify User Behaviour

User behaviour identification is a vast subject area due to the different ways that the user can interact with their mobile device. There is numerous research that has been done in accordance with this topic.

Casale et al. [5] in the research to identify human behaviour using the Accelerometer data in the wearable device of the user. With the emergence of smart watches which tracks the heart rate, steps taken, calories burnt in an activity as well as sleep patterns can give out several sets of data regarding the user behaviour. Hence in using a smart wearable device, we could track the users' physical activity throughout the day. In this paper the authors

have discussed five basic day to day activities that can be identified using the accelerometer alone.

They are walking, climbing stairs, standing, talking with people, and working with a computer. The research is done using a custom wearable that the user can use just when he/she changes their routine and the wearable can get the data according to the state that the user is in.

However, since some users do not like to use wearable devices causes a problem in this regard. Having to constantly interact with the wearable even if it is a mere touch will not be an accurate measure of a user behaviour.

Farrahi et al. [6] analyses user behaviour using location data gathered from cell-tower connections and proximity data gathered from Bluetooth information. It is then analysed through topic modelling to classify the user behaviour. This research covers over 68,000+ hours of user behaviour data which is then classified into datasets namely: going to work early/late, being home all day, working constantly, working sporadically and meeting at lunch time. The research is done using a Symbian Operating System driven Nokia 6600 which during the research times is a well-recognized mobile device. However, since the smart devices these days carry a set of sensors, there are obvious advantages today than what was a decade back.

Azam et al. [7] discusses this topic in a level that is closer to the current era of smart device and mobile computing. The data from the usual sensors such as Bluetooth, digital camera, GPS sensors etc. can be further enhanced using the additional sensors that is built inside a smart device. The research further addresses the fact that the data that is gathered from cell-towers and Bluetooth proximity sensors can only identify the location the user is in and it certainly cannot identify the activity the user is currently engaged in.

The research classifies the users to 2 user groups. High entropy users and low entropy users. High entropy users are the user group that randomly changes their daily routine and does not necessarily have a daily routine. A door to door salesclerk can be considered as a high entropy user, as he/she does not visit the same neighbourhood every day. A low entropy user is a person who does a day job at a specified workplace and use the same mode of transportation every day. Hence the change in the day to day behaviour is minimum.

However, this is also a relative measurement, as an elderly user who, due to an illness will do activities within a very small area while a daily worker will have a wide area of interactions and compared to an elderly user a day job user is a high entropy user.

2.5 - Using Contextual Information to Classify User Behaviour

Gathering the Contextual Data and identifying the useful information out of the whole set of data is the next challenge. In order to execute this process, a pre-defined method and goals should be set. A classification which is valid for all kinds of human behaviour should be used in this process. There are few research studies that were done regarding this area and there are several classifications that were created according to these studies.

In Casale's et al. [5] study which is based only on the Accelerometer to gather Contextual Information has a classification which depends on the accelerometer data. The Velocity and its derivations such as (Minimum, Maximum, RMS value, Mean, Standard Deviation etc.) is trained through a Random Forest data training mechanism to get the optimum classification. The Confusion Matrix created from the Random Forest data is used to classify the defined human behaviours like, taking the stairs, walking, talking, standing, and working.

Ma et al. [8] furthermore evaluates the study of classification of human behaviour by classifying human emotions using the contextual data gathered. The study addresses the issue of mental health problems that people face in day-to-day life and how mobile phone data is used to recognize such emotions. They have proposed a new framework called "MoodMiner" which uses mobile phone sensing data gathered from sensors like Accelerometer, Sound, Light and Location to learn regarding the user behaviour. "MoodMiner" combines the location data, activity data from accelerometer and the communication frequency of the user to determine the micro emotion the user is in.

Farrahi et al. [6] also uses their study to classify over 68,000+ hours of mobile phone data using Topic Models to classify the users' activity. It classifies the users' work patterns to working all day, working sporadically, leaving early or late etc.

The contextual data gathered from the sensors can be used in numerous ways to classify the user behaviour. However, classifying is subject to the application of the method that is used to. Also, the sensors that are being used to gather the data is also dependent on its application. In each of the studies mentioned above, the sensory data is used in different set of applications. The authors of the studies mention the fact of using mobile data sensors all the time will drain the battery life, hence causing the experience to fail. This would be needed to be taken into consideration to design a plan to gather data whilst not having a major effect on the mobile devices' battery life.

2.6 - Using Contextual Information to Create a User Profile

In Gao's et al. [9] study, the main objective is to find like-minded users through the whole user landscape. The methodology followed is to use the Context Data gathered through the mobile device in order to find out users with similar interests. A measuring unit called the "Trust Degree" is introduced in this study such that users with similar interests which has a similar "Trust Degree" is matched. The goal of the study is to reduce the overload

of irrelevant information which is thrust upon the users from multiple sources. If a study can identify the user interests albeit to a certain degree, then the information provided by various web content providers can differ and would enhance the user experience.

A users' profile is constructed using their short-term and long-term behaviours using a Personalized Ontology Profile (POP). Usually the first type of interests is defined from "topics of interest" which the user picks. However, such topics can be short term or long term and there is no way to determine whether it is such. Hence a 4-step knowledge structure is used to express the user interest. They are, Taxonomy, Thesaurus, WordNet, and Ontology. In a nutshell, Taxonomy defines the user interests as given by the user and the Thesaurus defines some similar terms that would denote the same. WordNet is a special type of thesaurus that is used in query expansion mechanisms. It would make sense of each word and would give a weight on each term as per the users' perspective. The ontology level defines the highest level of expressiveness where all the semantics are defined for any imaginable kind of word, hence allowing to expand a simple user interest to a deep level.

From the study mentioned it is identified that the fact that classifying user data and identifying the patterns and user interests is not a direct process. The user behaviours need to be classified and then revaluated for a period of time to check for the validity of the data. Once it is validated this information can be updated to the user profile as a valid part of information.

2.7 - Using Contextual Information to Create a System Which Provides the User with Proactive Notifications based on the User Preferences

The research study done by Mayrhofer et al. [10] uses the above-mentioned topics to create a framework which supports proactively predicting user behaviours using the past contextual data. It creates an architecture to facilitate this behaviour in a mobile device.

The major challenges mentioned in this research is to use the limited resources available in a mobile device to process the contextual data gathered through the sensors. The process is online, and the processed information is used to predict the users' behaviours in the future. As there is no learning pattern used and a non-obtrusive user interaction should be needed, the framework is created to affect the user minimally during the process. The process itself is carried out in 4 steps:

- Feature Extraction
- Classification
- Labelling
- Prediction

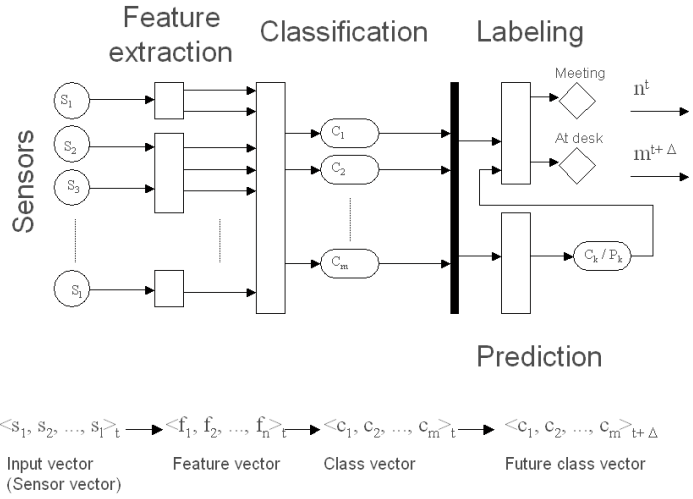

Figure 2.1 - The 4-Step Process to Predict User Behaviour

As the diagram describes, the user inputs are measured via various sensors (as the research is done from a PDA, more sophisticated sensors are not available - the basic sensors in a PDA such as microphone, Bluetooth, WLAN, brightness etc. is used) and then the data is classified into user states. After a few rounds of learning, the device can identify the state patterns the user follows usually. Hence a state S_1 can either go to state S_2 or S_3 depending on the time. The framework can then predict if the user is in state S_1, the user will most likely go to state S_2 or S_3.

The learning phase of the framework should be continuous to optimize the process further as the more data is input to the system, the more correctly the system can identify the next state of the user.

From the above research which discusses the main topic of this research as well, the main steps that needs to be carried out to create a proactive notification system is understand the user activities, classify them, and then predict on the classified information.

2.8 - Similar Systems

Context Aware notification systems have been created in various studies and it is mostly based on the location driven data. Location based reminders plays a significant part of such notification systems where once a user arrives at a location the system suggests a fact regarding that location. This information ensures that the marketing decision makers of supermarket outlets and shopping malls can prompt users regarding their latest offers and discounted items once the user arrives in a shopping location [11].

This has been significantly used by Google Maps service for quite a long time as well. Google Maps service periodically checks the location where the user is in and if a user is in a place for a significant amount of time, a suggestion notification pops up to review the

location and add any photos that were taken in the location to the Google Maps information base. This will be useful for any other user that looks up the place before visiting.

“Place-Its” is a location-based reminder study that was carried out by Sohn et al. [11] in which the application lets the user to create a note in a location and pin in such that once the user arrives or departs from that location it will trigger a reminder-based alert such that a certain activity needs to be done. The 3 components in a “Place-It” is trigger which is either departure or arrival, text which is the reminder text and location which is the location where the post is pinned.

Messaging is another factor that the user significantly uses to interact and socialize with friends, family, and co-workers. At present messaging services like WhatsApp, Viber, Google Hangouts and Skype facilitate such options. However, in a working environment e-mails are the commonly used mechanism. Buthpitiya et al. [14] in the study creates a context aware messaging assistant which addresses the problem of a flooded inbox. It will read the mail and identify the context of it and classifies it to 1 of 4 categories namely: Unimportant - Urgent, Important - Urgent, Unimportant - Unurgent and Important - Unurgent. Hence the notification will remind the user only for the Urgent mails and keep the rest for the user to investigate whenever possible.

“Cybreminder” [13] is another such study which uses contextual information including location-based data along with a Context Toolkit developed by Dey et al. [14] to determine when to best trigger reminders. The purpose of this project of abstracting the hardware dependency from the software developer. The project succeeded in creating a fully featured reminder application that uses a set of contextual information.

As mentioned in the above research studies, there are various ways to extract contextual information and use the information to provide the user with notifications. There have

been several different implementations of such systems which uses different kinds of contextual information for different kinds of applications. However, there are very less amount of studies that has been done to create a system which proactively predicts and suggests users to take actions based on their contextual information. This is the research problem that is addressed by this study.

2.9 - Research on Moderating Push Notifications

Mehrotra et. al. [17] has done several researches on this specific matter and had come up with several ideas on how to moderate Push Notifications. In the first of such research an Android Application is built to get data related to the incoming Push Notifications. Data related to the Arrival Time, Removal Time (if the notification was dismissed from the screen), Application which sends the notification, Title of the notification, Alert type, Physical location of the user, Surrounding sound etc. were taken and then a questionnaire was posted on the users who used the application. From the data taken out of the questionnaire a large data set was formed and machine learning was used to find out patterns of the data set. From the results it was deducted that even though notifications help out people to accomplish different tasks, they tend to burden the user with several unwanted notifications a single user receives per day. Furthermore, it states that the user acceptance can be predicted after the analysis of a large data set.

However, as noted this would be a generic approach to predicting user behaviour since the complete data set is analysed as a whole, rather than customizing it per user. Since, each user can act different on the same notification, there is a downside to this research.

In a similar research, Mehrotra et. al. [16] studies the receptiveness of a user to mobile notification and what are the down sides of it for a user. If a user who is performing a task and gets interrupted by an incoming notification causing the ongoing task to be stopped, this can be considered as interrupting the current task at hand. Hence, in such cases, the

disruptiveness of a notification is studied. This also considers the notification arrival time and the response time for the notification (whether it is dismissed or handled). The research concludes that most notifications are dismissed if they are usually disruptive. Furthermore, the research states that the user or application who send the notification would be something which needs to be considered. The limitations of this research include the correctness of measurements such as whether the notification is actually dismissed or not (in scenarios where a notification can be seen on multiple devices, it can be handled elsewhere while the mobile notification is dismissed).

In his attempt to create a working mobile notification system Mehrotra et. al. [19] and team creates “PerfMiner” a system (mobile application) which moderates incoming notifications and they state this as the first of its kind to perform such a task. In this research, a rule-based system is created to identify whether the notification is Acceptable or Dismissible depending on the user location, time of the day and the application which triggers the notification. Using this rule-based system, the incoming notifications are being moderated. However, Mehrotra et. al. also states that end user acceptance on Machine Learning is of paramount importance if such system is to succeed since if the user is to understand and trust the system, the understanding of Machine Learning is important. This is due to the nature of Machine Learning having a Black-Box like state and does not have the simple IF-THEN-ELSE kind of mechanism.

CHAPTER 3 - METHODOLOGY

3.1 - The Solution

The work done in this research investigated the several aspects discussed in the preceding literature and several aspects were used to conceptualize and create an even better system.

The study was done on how to moderate the users' incoming push notifications depending on their current context, activity and preferences. Hence some notifications from the same application were pushed to the user while others were hidden for later use or dismissed completely.

The main objective was to identify which notifications needed to be pushed and which notifications were to be delayed or hidden. In order to accomplish this, the following steps were carried out once the application was installed in the users' mobile device.

- Got the users' required configurations per each application installed
- Got the users' configurations for the commonly visited locations like Work, Home, Educational Institute, Religious Place, etc.
- A profile was created according to the configurations of the user and was then synchronized to the associated cloud database
- A local copy of the database was always kept in the memory for faster access and low mobile data usage (Usually synchronization happens in the users' specified time or an idle time where a strong Wi-Fi signal is available)
- Once all was setup correctly, periodically measured the users' current activity and the location information
- Once a push notification was triggered, the last known measurements were used to determine the users' requirement for the notification

- In cases where notification was to be moderated further, 3rd party services (which have machine learning (ML) capabilities) were used

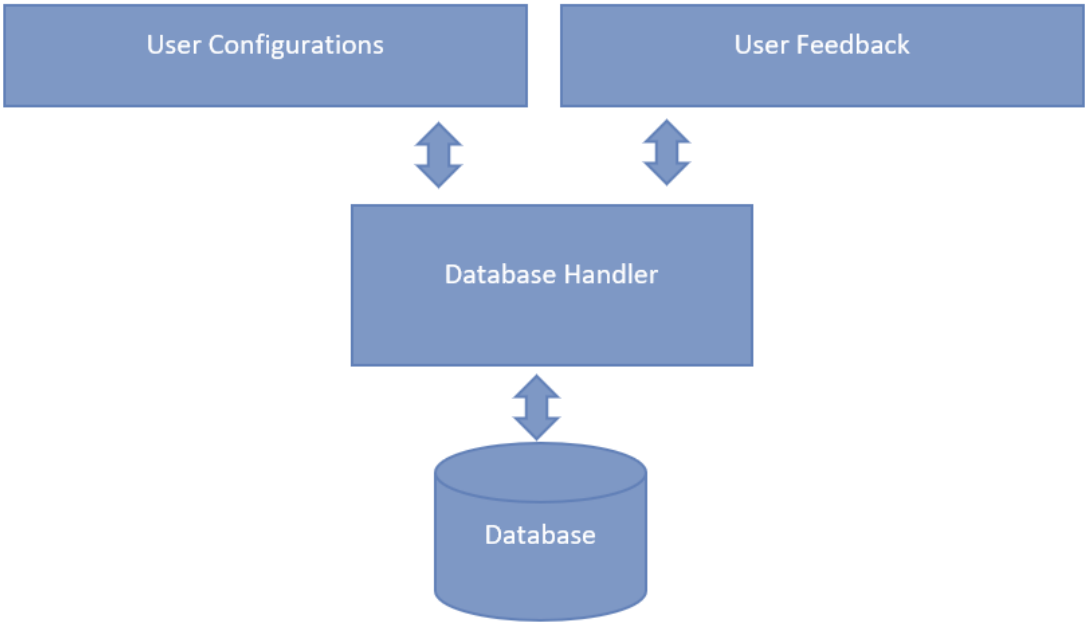

Figure 3.1 - The Process of Creating a Profile for the User

In general terms, the push notifications a single user receive can be categorized as follows:

- Notifications from the operating system - Reminding of any pending updates, security threats etc.
- Alerts from the mobile carrier - Short Message Service (SMS) alerts, missed call alerts etc.
- Alerts from utility apps - Alarms, meeting reminders, birthday reminders etc.
- Alerts from location-based apps - Google Maps, Apple Maps etc.
- Alerts from other miscellaneous apps - Games, Weather etc.

Out of these, depending on the context and the users' current activity some notifications were muted whilst others were displayed.

For example, while the user was driving, any call alerts (if required only from a specific set of contacts) were displayed since minimal mobile-user interaction should happen while driving. While the user was at work, all alerts from the mobile carrier and utility apps were displayed since it was considered as a necessity. Notifications from the operating system and any other miscellaneous apps were hidden until the user was at his or her residence such that these notifications were taken care of.

Initially, the batter power consumption was a major downside in this application. Hence some sensor data were taken periodically while other sensors were used only when the user was interacting with the device. Furthermore, some actions (like usage of 3rd party APIs) were inoperable in a network disconnected or in network environment where signal strength was low in order to prevent excessive battery usage for a simple action.

Along with the battery power usage, there was a cost when interacting with 3rd party services in order to gather the necessary activity related data. Hence such interactions were set to a minimal and if and only when the mobile detects a change in the activity or location were such interactions activated.

3.3 - High Level Architecture

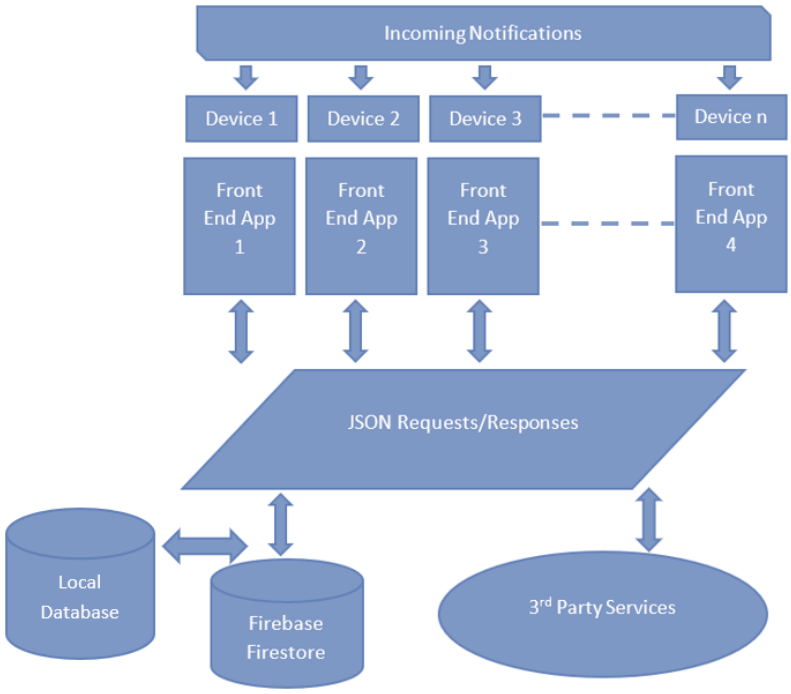

Figure 3.2 - High-Level Architecture of the Application

3.3.1 - Android Application

The main application of the system was the Android application which the user downloaded from the Google Play Store and installed it on their mobile device. The application itself needed permissions for location data in order to cater the required objective.

Furthermore, the 3rd party services used by the application handled the required data gathering in different intervals and neither the application nor the user did not have any requirement to control it.

3.3.2 - 3rd Party Services

3.3.2.1 - Google Awareness API

Google Awareness API [18] is an API which unifies several sensors signals as well as context signals in a single API and would enable the developer getting insight on several aspects of the users' current activity location etc.

Following details were gathered from the two sub APIs (Google Fence API and Google Snapshot API) under Google Awareness API:

- Users' current location
- Users' current activity
- Device specific conditions
- Proximity to nearby location beacons
- Weather conditions nearby

The information gathered from these APIs was helpful in determining the current contextual information regarding the user and was used to moderate the incoming notifications as per the latest information gathered by these APIs.

Considering the pros and cons of using these APIs, the API itself does all the heavy lifting and the users' device only need to provide the data from their sensory setup. Hence a very low amount of power was consumed from the users' device since all computations and calculations happen in the cloud with the result only being passed back to the device.

Furthermore, Google Activity Recognition API also was used to gain understanding on the users' current activity and when it is changed etc.

3.3.2.2 - Google Natural Language API

The next hurdle when analysing notifications were to analyse the text in the notifications in order to determine the importance of the text and whether to display it as it is, delay the displaying or hide it completely (even mark it as spam if required).

Such capabilities are provided by the Google Cloud Natural Language API [19] which uses the Google's inbuilt natural language capabilities along with the power of machine learning and deep learning.

Hence in this scenario, the text from the users' notification was passed on to the Google Cloud and it determined and classified the text received to several categories varying from News, Sports, Internet and Telecommunication, Computer and Electronics etc. It also pointed out any entities involved in the text such as an Organization, Event, Person etc.

With this information, according to the users' preferences for the categories that he or she was interested in, the notifications were skipped or displayed.

Using of 3rd party services can be of much importance as the computational power to learn and decode the contextual information is taken care of by the services. Furthermore, machine learning capabilities which the services possess with an everyday increasing dataset from users worldwide, the capabilities of such services are guaranteed to get better each day.

3.3.3 - Firebase

Currently, the system relies on Firebase services in 2 forms.

- Authentication
- Data Storage

The user was first authenticated by Firebase Authentication service and currently only Google Mail authentication was implemented. However, since a 3rd party service was handling the authentication functionality adding more authentication factors should be easy and reliable for the user as well as the developer.

As Figure 3.2 illustrates there is a local database for faster access, and the user settings were periodically synchronized with the cloud database from the local database. The main objective was that even if the user uses multiple devices or started to use a new device, all the settings and configurations along with the user profile was readily available in all such devices.

Furthermore, if the study is to be further enhanced there is an additional benefit of using Firebase Firestore [20] in order to harvest the 3rd party machine learning capabilities it provides. Hence in an ideal scenario, where the user provides the configurations and necessary basic information and depending on the user behaviour analysis, machine learning can be used to enhance the performance and throughput of the application itself.

3.4 - RESTful Web Services and JSON Responses

Representational State Transfer [21] (REST) is an architectural style mainly used to communicate with Web Services via Hyper Text Transfer Protocol (HTTP). This would enable the developer to use a common Uniform Resource Identifier (URI) along with a set of parameters (dynamically defined) to be sent out in order to be processed by the Web Service.

The response or payload from the Web Service is usually sent out through one of the formats like Extensible Mark-up Language (XML) or JavaScript Object Notation (JSON). In this application, the responses are sent out in the JSON format and then be converted to a corresponding response object via Deserialization.

3.4.1 - Serialization and Deserialization

When a data is stored in the memory as a custom object, and the data needs to be passed out in a uniformly readable format serialization is used. Serialization converts the custom object into an XML or a JSON type text file which can be consumed by any 3rd party service or an integrated application.

Deserialization is the opposite process of Serialization where an XML or JSON object is converted to a custom object. In responses coming from Web Services, the payload is delivered in the XML or JSON formats where after checking the flags of the response object, the response is converted to a custom object of the Response type [22].

3.5 - How Does the Application Work?

3.5.1 - The Setup

In a typical scenario, the user after downloading the application from the Google Play Store and installing it in the mobile device. Figure 3.3 shows the login screen which authenticated the user with the Firebase Cloud Authentication service.

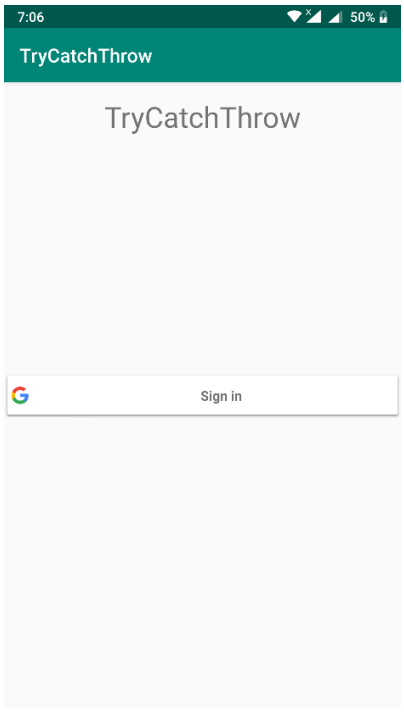

Figure 3.3 - Screenshot of the Login Page of the Application

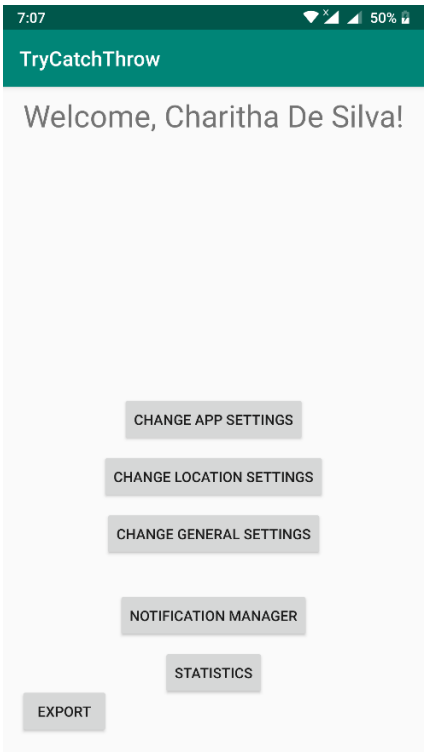

Figure 3.4 - Screenshot of the Home Page of the Application

Afterwards, the users used ‘Change App Settings’ option or ‘Change Location Settings’ option to add or change their configurations regarding incoming notifications.

In both options the user was provided with three options in order to handle the incoming notifications.

- Use System Settings - The application itself does not do any kind of moderation to the given application
- Moderate All Notifications - The application itself moderates all the notifications triggered from the given application
- Stop All Notifications - The application blocks all notifications triggered from the given application

Furthermore, another option was to combine above three settings along with the analysis of the text itself such that users provides a set of keywords to moderate the notifications. If a notification carried a list of given keywords specified by the user, such notifications can be hidden or displayed depending on the user context and configuration.

After all configurations were set up, the user can turn off the application and the notifications were moderated according to the specifications. For the analysis purpose, all the notifications the user receives were displayed under the Notification Manager option and would mention whether these notifications are moderated or not.

3.5.2 - How Notifications were Processed

Once a notification arrived in the device, first its settings and configurations were checked. If the application where the notification originated was not moderated, nothing happened to the notification, it was displayed as it should.

If the application was moderated, the configurations were considered along with the sensory data that were periodically taken in order to proceed further. If the notification was being moderated regarding the time frame and if it is not the specific time to be moderated, it was displayed as it should.

If the notification was to be moderated depending on a location or an activity, the current sensory data was considered, and the notification was displayed or hidden depending on the classification.

If the notification required analysis using the content only, the data was sent to the 3rd party services like Google Natural Language API for further analysis and once the analysis results were received, the notification was processed accordingly.

3.5.3 - Managing Outgoing Requests and Incoming Responses

When there was a requirement to get input from a 3rd party service, a RESTful Web Service request was initiated. The response would be in JSON format and it was then converted to the corresponding object for further usage.

3.5.4 - User Feedback Generation

Since the application was still on the phase of evaluation and testing, user feedback for all the received notifications were gathered. The feedback gathered was to mention whether the moderation worked as expected or not. Once the feedback was provided by the user, the notification related feedback was exported in the form of a CSV file for further evaluation.

3.6 - Evaluation

Once the development work was completed and the testing was carried out, the application was distributed among a set of people of different age demographics. Hence the configurations of each user can be different.

Afterwards, after a given time period, the results were analysed on how the application had fared over a period of time. The main calculation points gathered from the analysis are:

- How many notifications arrived (per day)?
- How many notifications were displayed?
- How many notifications were skipped (hidden)?
- What was the percentage of the application being able to moderate the notification accurately?

After the given period, the users were requested to review the notifications and provided feedback on the above-mentioned questions. Afterwards, depending on the results gathered from the users' feedback, the algorithms were adjusted further such that more accurate moderation can be done on the notifications.

A user group of 20 people were allocated to evaluate each functionality. The users provided feedback on the notifications they received for a 10-day period. Afterwards, depending on the success rate of the moderation and if there were any further modifications required on the algorithm that moderated the notifications, another 10-day period was evaluated. This was carried out 3 times whereas the algorithms were modified accordingly.

CHAPTER 4 - ARCHITECTURE AND IMPLEMENTATION

The main outcome of this research was a research concept along with a sample software product for the Android operating system which facilitated in the moderation of notifications as per the users' contextual information, activity information and preferences along with the use of the multiple sensory setup and the use of several 3rd party APIs.

4.1 - Design

The main architecture of the product was a simple 2 tier architecture where all the client code and business logic resided in the application itself while the data storage layer was divided into 2 parts.

- Local Storage was used to store all settings and configurations set up by the user along with the moderated notification information and any related Meta data
- Cloud Storage was used to synchronize with the local storage such that the user logging in from a different device would have the same configurations set up instantaneously
- Multiple sensory setups of the mobile device were used to get the necessary data for the 3rd party APIs to carry out the computations required
- 3rd party APIs provided different services to the application in order to cater the requirements.

4.1.1 - Solution Architecture

In the previous chapter, the high-level architecture of the solution was described. However, since this was an evolutionary product it was better if the product was kept modular such that each functionality could be replaced by a different module which could perform the same functionality.

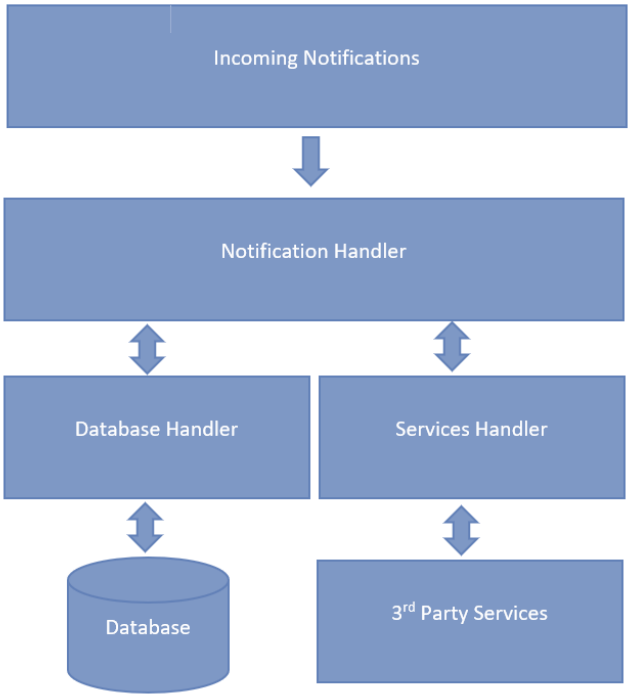

Figure 4.1 - Modular Architecture of the Solution

The modular architecture was created following industry level standards for high performance, maintainability over the course of the life cycle of the solution.

- Created different modules which handled different functionalities and had extremely low inter modular dependencies which enabled the developer to add, remove or change modules without affecting the overall functionality of the system
- De-coupled all database related activities from the functionalities such that database related activities were performed separately from the main functionalities of the application

4.2 - Implementation

4.2.1 - Adding Firebase Services to the Application

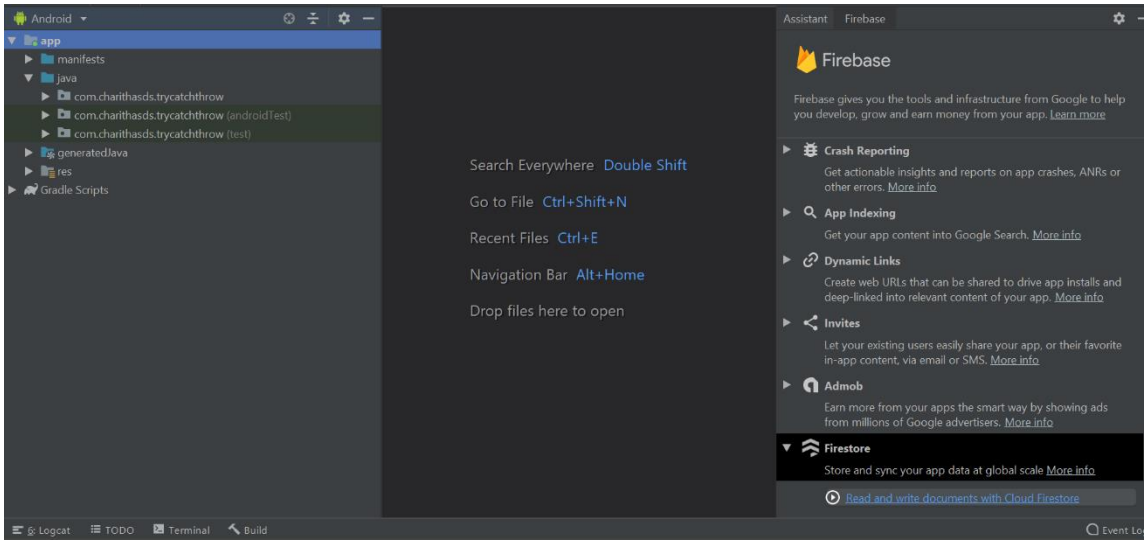

Figure 4.2 - Adding Firestore Cloud Database

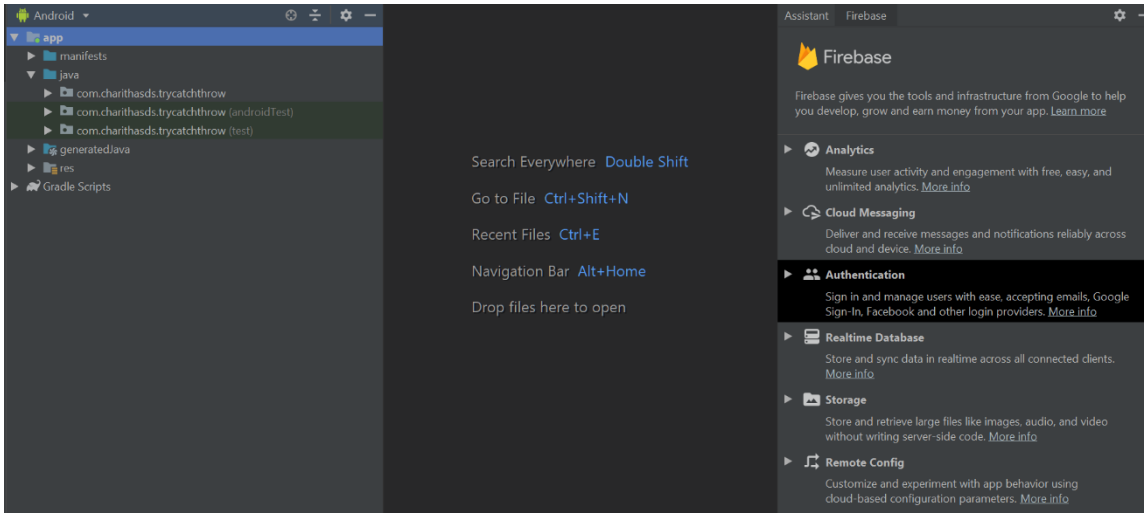

Figure 4.3 - Adding Firebase Authentication

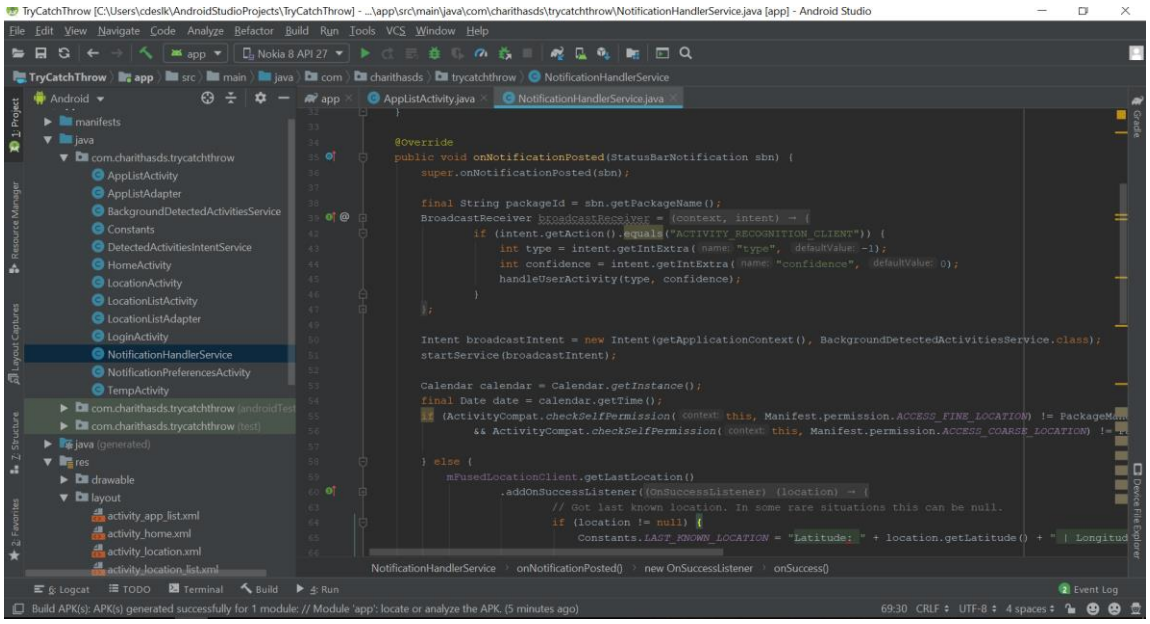

Figure 4.4 - Handling Incoming Notifications

The first step was to setup the authentication with the NoSQL Cloud Database Firebase Firestore to the application where all the cloud database related functionality would occur.

4.2.2 - Enabling APIs from Google Cloud Platform

Name	↓ Requests	Errors (%)	Latency, median (ms)	Latency, 95% (ms)
Maps SDK for Android	44	0	0	0
Identity Toolkit API	24	0	0	0
Cloud Natural Language API	8	0	157	251
Cloud Functions API	4	0	229	484

Figure 4.5 - Google API Usage During the Testing Phase

All the required 3rd party services were provided by Google Cloud Console and the APIs were enabled along with the respective API keys added to the application for usage.

Since all the operations happened at the users' device and 3rd party services, there was no requirement for a central dashboard to monitor any traffic. Any such traffic was monitored

through the Google Cloud Console tools in which the incoming traffic data, incoming requests and responses were visualized.

Furthermore, since gathering all the required data per each notification arrival drained the battery and increased the traffic on the 3rd Party APIs and the usage of network bandwidth, the user applications fetched the required data periodically. For the research purposes, it was taken every 5 minutes.

Hence for each minute, the users' activity and location data were collected from the 3rd party services and stored in the application locally. Once a notification arrived during this interval, the application moderated the notification in accordance to the data which was last retrieved. Hence it ensured that per each notification there was not a service request.

However, for the configuration where all 3 types of settings were working in parallel, meaning the application read out the notification content and categorized the notification. In such scenarios, the above-mentioned workaround did not work since it depended on the notification content itself. Hence for such situations a service request needed to be carried out.

4.2.2.1 - Working with Google Natural Language API

Google Cloud Natural Language API uses machine learning and deep learning to train a huge set of models to cater a set of services.

- Entity Analysis - Recognition of People, Organizations, Places from a sample text
- Syntax Analysis - Extract tokens and sentences, identify parts of speech and create dependency parse trees from a sample text
- Sentiment Analysis - Understand the sentiment of a sample text

- Content Classification - Classify the content of a sample text into a set of given categories

For this application, currently only the Content Classification was used and there is a possibility to use Sentiment Analysis going forward. Content Classification uses the machine learning capabilities and categorized and classified the notification content and provided a set of categories in which the content can be put into.

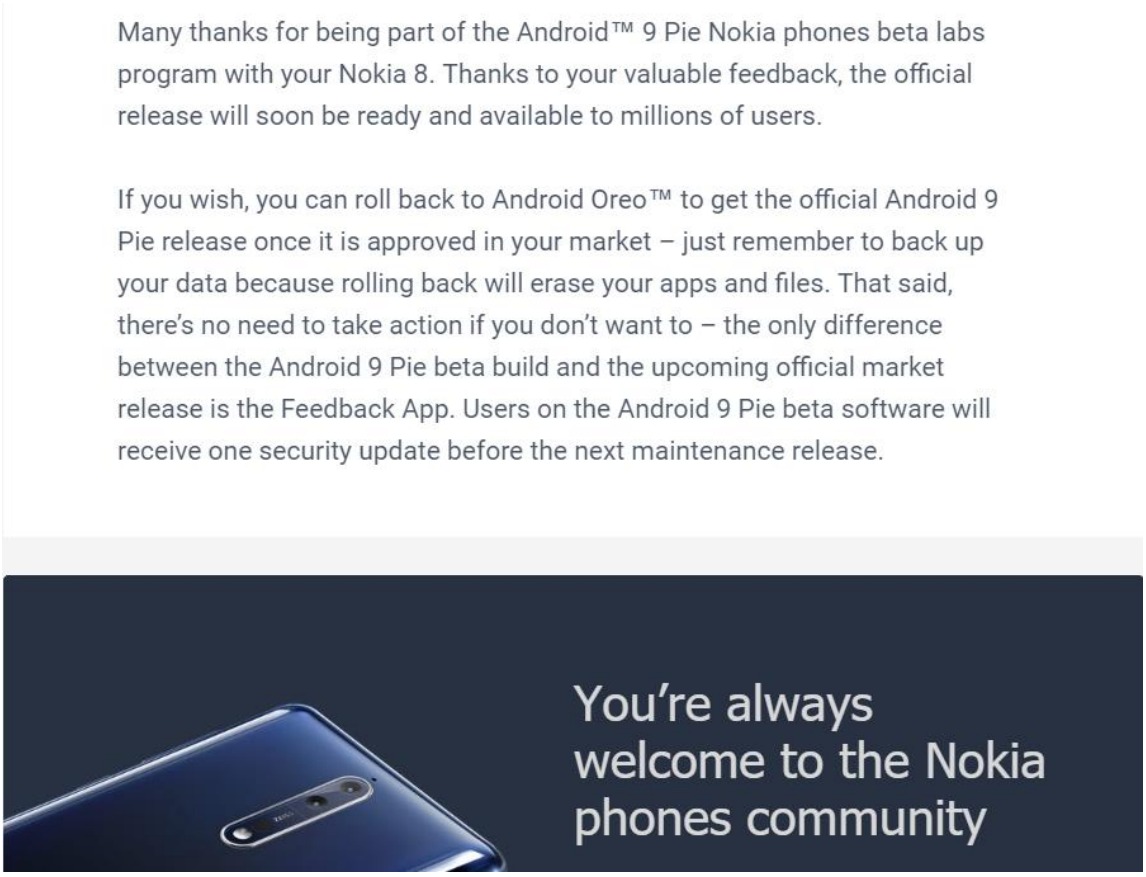

Figure 4.6 - Sample E-Mail

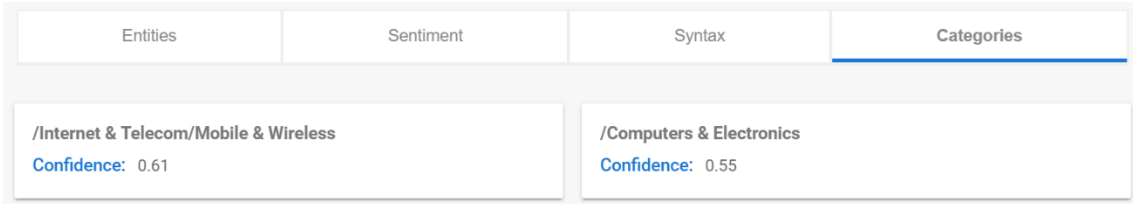

Figure 4.7 - Categorization Results

From this method, if the user had provided that he or she require, or not-require notifications related to 'Computer & Electronics' or 'Mobile & Wireless', then the moderation happened accordingly to the Confidence value of the provided category. Hence for this research the Confidence value was set to be above 0.5. Hence if there was a Confidence value above 0.5 then the category was also taken into consideration.

Finally, once all these information was processed the application decided whether or not the notification was important to the user in the current context and then made a decision whether or not to display it right-away [21].

4.3 - Sample Results

Application Package	Activity	Time	Location
com.viber.voip	Unknown	Wed Feb 20 06:59:46 GMT+05:30 2019	Latitude: 6.830039 | Longitude: 79.8679754
android	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048
com.whatsapp	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048
com.whatsapp	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048
com.whatsapp	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048
com.whatsapp	In Vehicle	Wed Feb 20 07:31:55 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379
com.whatsapp	In Vehicle	Wed Feb 20 07:31:55 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379
com.whatsapp	In Vehicle	Wed Feb 20 07:34:33 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379
com.whatsapp	In Vehicle	Wed Feb 20 07:34:33 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379
com.android.mms	In Vehicle	Wed Feb 20 07:37:12 GMT+05:30 2019	Latitude: 6.910161 | Longitude: 79.8501407
com.whatsapp	In Vehicle	Wed Feb 20 07:49:29 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017
com.whatsapp	In Vehicle	Wed Feb 20 07:49:29 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017

Figure 4.8 - Sample Result-Set from a User

Figure 4.8 was a sample result set of a user before analysing the notification moderation results. The users checked the accuracy of the moderation of the results and provided feedback. Such feedback was then taken into consideration to change the algorithm of the moderation.

CHAPTER 5 - SYSTEM EVALUATION

The main objective of this research project was to classify and manage incoming notifications. Hence, as mentioned in the previous chapters, the correctness of the management of notifications was the main evaluation point discussed. Furthermore, the performance and the effectiveness aspects were discussed in detail.

5.1 - How was the Evaluation Done?

During the testing phase, the evaluation was done in 3 steps. In each of these steps, the results were analysed depending on the user feedback and then the algorithms were further tweaked. Afterwards, the updated applications were deployed to the users' devices and the next testing period commenced.

The advantage of having such testing being done was that the application errors, performance issues and issues that were not expected previously during initial unit testing period were identified. Furthermore, as the users were actively involved during the testing period, the feedback that was gathered was invaluable to adjust the algorithms.

In the initial step, the incoming notifications were not actively moderated. In fact, it was just analysed, and the results were logged in order to gather feedback from the users. This was done by 20 users in different age demographics for a 10-day period.

Application Package	Activity	Time	Location	Status
com.viber.voip	Unknown	Wed Feb 20 06:59:46 GMT+05:30 2019	Latitude: 6.830039 | Longitude: 79.8679754	Displayed
android	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:23:05 GMT+05:30 2019	Latitude: 6.8554817 | Longitude: 79.8654048	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:31:55 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:31:55 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:34:33 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:34:33 GMT+05:30 2019	Latitude: 6.8803273 | Longitude: 79.8563379	Displayed
com.android.mms	In Vehicle	Wed Feb 20 07:37:12 GMT+05:30 2019	Latitude: 6.910161 | Longitude: 79.8501407	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:49:29 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Displayed
com.whatsapp	In Vehicle	Wed Feb 20 07:49:29 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Displayed
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Hidden
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Hidden
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Hidden
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Hidden
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Hidden
com.whatsapp	Still	Wed Feb 20 07:53:50 GMT+05:30 2019	Latitude: 6.9198636 | Longitude: 79.8573017	Hidden

Figure 5.1 - Display Status of the Notifications

Afterwards, once the users were satisfied with the performance and the stability of the application, the notification management was activated for few applications which had high incoming notifications. For example, social media applications like Facebook, WhatsApp, Viber etc. and utility applications like SMS. The users then again provided feedback for the notifications.

5.2 - Factors Evaluated

As mentioned in the Evaluation section of Chapter 3, the main factors that were evaluated; the number of notifications arrived in a given users mobile device per day and what notifications were to be displayed and what notifications were to be hidden. The accuracy of the said notifications was the main evaluation point of this research.

However, going forward through the testing phases, several more factors came up as to how the performance of the application would be when several information was to be processed, when the mobile internet connectivity was low and what were the security and privacy concerns when moderating notifications.

Furthermore, there were scenarios where the application did not have the capability of moderating each notification if they came up in a queue one after the other without any delay.

The scalability of the application was not a major concern as the main processing was done in the Cloud which had its own scalability aspects. However, parallel processing of notifications was the option when there was a queue of notifications which needed moderation as and when they arrive.

5.3 - Feedback Results

Following are cumulated results from the feedback provided by the end users after the initial phases and the subsequent iterations.

Stage	Average Number of Incoming Requests	Average Number of Apps Used	Number of Notifications Received from the Used Apps	Average Number of Notifications Displayed	Average Number of Notifications Hidden	Average Number of Notifications which Actually should be Displayed	Accuracy of Moderation
Initial	90 - 120 notifications per day	5 - 8	10 - 15	3 - 4	8 - 10	7 - 8	46.67 %
After Test Phase 1		8 - 15	25 - 40	12 - 15	10 - 14	16 - 23	69.23 %
After Test Phase 2		13 - 19	50 - 80	45 - 58	25 - 37	47 - 62	94.5 %
After Test Phase 3		No algorithmic changes - Introduction of Suggestive Notifications					

Table 5.1 - Data on Moderated Notifications

User testing was the foundation of making this application and its implementation successful. Hence a varied user base for testing was of paramount importance. The users which were chosen for testing came from different age demographics and in different work ethics and cultures.

Few users were from my workplace who were usually in the age group of 25 - 30 and were usually the most avid users of the mobile devices in their day to day activities. Some users came from the 20 - 25 age bracket where they usually were in the learning background and maybe working as trainees or interns and attending lectures in the evening into the night.

Few users were in the age bracket of 30 - 40, where they were in various fields of IT, Banking, Engineering, Supply Chain Management backgrounds. These users usually used their mobile devices for work related activities such as Outlook Mail and Skype for Business applications to attend to e-mails and meetings on the go.

Other set of users came from the age bracket of above 40 out of which some were retired after 55 years of age. The ones which were working were using their mobile devices for work related applications and ones who were retired tend to have more utility related applications installed. It was a commonly identified fact that since this age group had the greatest number of users who were not used to technology very much, they used to install a number of Spam or Junk applications into their mobiles without even knowing or maybe tricked into installing through various advertisements. Hence the number of notifications which were useless seems to be in the higher side in this age category.

Age Bracket	Average Number of Apps Installed	Average Number of Notifications per Day
20 - 25	60 - 75	100 - 120
25 - 30	50 - 70	90 - 120
30 - 40	40 - 60	80 - 100
Beyond 40	45 - 60	70 - 110

Table 5.2 - Data on Notification Received per Age Demographic

In the initial stage, there were a smaller number of applications that were used for testing since the users were not sure about the reliability of the application and its moderation process. Hence, the effectiveness of the application at this stage was not a reliable identifier.

Initially, the users only used the application on very less used or very less important applications like Games and Sports updates. Hence, it was usually, an ignorable value for the accuracy that came out of the evaluation process. Usually, the users did not want to be disturbed of games and sports updates during their work time and the time during daily driving. People who were using the daily commute or using taxis from and to work used to read the sports and news updates during the commute.

These were instances where the application had some inconsistencies due to the changes of locations and the speed accuracy. Sometimes, the application had derived that the user was driving a vehicle while the user was in a vehicle. In such scenarios, the notifications were not displayed.

The initial concerns were resolved after fine tuning the application and it was then identified that due to the inaccuracies of the application when taking the location related values, it was decided that location data, activity related data was to be taken in a periodic basis.

Hence, after the initial testing phase and the reliability of the application improved, the users agreed to use the application on several day-to-day used applications. The added applications were mostly chatting applications like WhatsApp, Viber and Facebook Messenger.

Once the evaluation period was over, the results were moderately successful and the main concern of the users was the battery life while the application was running in the background since now there were calls to several web services in order to fetch the location and activity data.

Hence, in order to optimize the process, it was decided that the period from which the values were fetched from the web services should be increased from 30 seconds to 5 minutes. It was assumed in such a way that a user will not change their activity and location drastically within the given 5-minute period. This was decided after changing the period of which the activity and location data was fetched vs. the accuracy of the notifications vs. the time taken for the battery to die out completely.

Period of Data Fetched	Accuracy of the Moderation	Lifetime of the Battery
30 seconds	68%	8 hours
1 minute	72%	9.5 hours
2 minutes	67%	13 hours
5 minutes	65%	15 hours
10 minutes	54%	18 hours

Table 5.3 - Period of Data Fetched vs. Accuracy of the Notification Moderation vs. Lifetime of the Battery

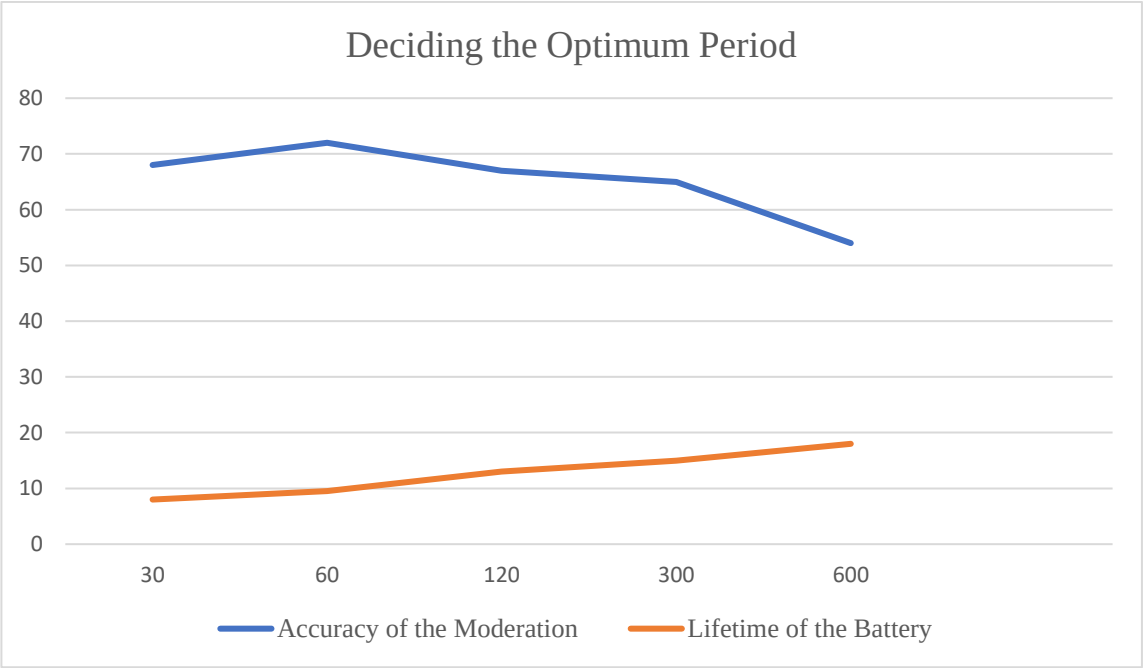

Figure 5.2 - Deciding the Optimum Period

From Figure 5.2, it was then decided, for the optimum balance between the lifetime of the battery and the accuracy of the moderation, the measurements were to be fetched at every 5 minutes.

From the results from the Table 5.1, it was observed that the optimizing of the logic had a major effect on the moderation of the notifications and the accuracy of the process has increased into a moderately successful value.

Afterwards, testing phase 2 was commenced out after the users agreed to use the mostly used applications like Text Messages, Outlook and Skype along with social media applications like Facebook and Instagram.

The next phase was the phase in which the application had to moderate a high number of notifications since most of the applications that were used by the common everyday user

was utilized for the moderation and hence most of the incoming notifications were moderated accordingly.

Due to the high number of notifications, even though there were not much of battery related issues, the users came up with performance concerns like application crashes and security vulnerabilities that all their notification content was read by the application and if required sent to the cloud-based services in order to analyse the content.

The performance related concern was when the users were receiving several parallel notifications from different applications and the application was unable to process them in an effective manner. Hence, there was a queue introduced to insert the incoming notifications and then processing was done in the arrival order. Furthermore, in scenarios, where the same application sent a batch of notifications (social media applications like WhatsApp etc.), a timer was introduced for each application. The timer would be set to 0 when a notification arrived from a certain application and then the moderation was done for that notification. Once this was done and if another notification in each 1-minute period arrived, the previous moderation result was utilized without re-moderating the notification completely. This was a highly effective way to moderate parallel notifications.

Even though the content was not stored anywhere in the cloud or locally, it was a notable security lapse and the eventual argument was to fetch the keywords out of the content of the notification and send it to the cloud rather than sending out the whole content.

However, this had an eventual downside as well since some notification content had words in non-English languages such as Sinhala, Tamil and even in Sinhalese + English combined. Hence there were some issues when extracting keywords out of such notifications. It was then decided, that in an instance where keyword count cannot be of a minimum range, the complete notification content would be sent to the cloud for analysis.

Keyword Count	Accuracy of the Moderation
< 2	< 10 %
2 - 6	17.2 %
6 - 10	48.3 %
10 - 15	62 %

Table 5.4 - Keyword Count vs. Accuracy of the Notification Moderation

Hence it was concluded that at least 10 keywords should be extracted from the notifications. In a scenario where the notification word count is less than 10, the complete notification was sent out for analysis.

In an overall analysis perspective, it was noted that the users used more location and activity related moderation considerably over others. For example, most of the users viewed the notifications much less (with more ignoring or rejecting) when the location is set to “Work” when comparing to other locations. “Home” is where most notification viewing took place. These notifications were mostly ranging from Social Media apps to miscellaneous apps. Notifications from the most important apps had been viewed while in “Work”.

When considering the Activity related moderation, it was noted that most of the viewing of notifications were done when the user was “Still”. While the user was “In a Vehicle”, most of the notifications were hidden. All such hidden notifications were viewed or rejected once the user is at “Home” and while “Still”.

The activity and location-based result summary is as follows:

Location	Viewing Percentage
Home	78.7%
Work	17.3%
Education	28.1%
Religious	45.6%
Other Locations	45.2%

Table 5.5 - Location vs. Notification Viewing Percentages

Activity	Viewing Percentage
Still	21.9%
In Vehicle	7.8%
Walking	32.4%
Running	4.1%

Table 5.6 - Activity vs. Notification Viewing Percentages

In the final phase of testing, the user behaviours were analysed and mapped. This was to be used for any "Suggestive Notifications" which was a feature that analyses the behavioural patterns from the incoming notification data and then suggests next actions proactively.

For example, in a scenario where the user always reads the daily news while on commute to work, once the user commences their commute each day, the device could initiate a notification reminding the user to start reading the daily news and if possible, provides a dynamically changing link.

This was enabled on a set of users and the objective was to create a set of rules based on patterns and then the users can enable or disable the rules which can initiate or revoke the suggestive notifications. The main concern of this was the pattern identification procedure, and it was decided that in a scenario where the system can observe the same recurring pattern for a 10-day period, a rule was created such that the user can enable it accordingly. This was an experimental feature, that some of the users liked and some of the users dismissed completely.

A sample Rule would look like the following:

1. Location - Going Away from | Coming Towards
2. Time - Between a time period of 1 hour
3. Activity - Should be consistent
4. Opening an app

Hence, from these combinations, the rules were created after a pattern was identified. Such patterns include:

{Work - Leaving; 05.00 PM - 06.00 PM; In a Vehicle; Opening Music}
{Home - Leaving; 06.30 AM - 07.30 AM; In a Vehicle; Opening News}

Overall, it was concluded that after 3 stages of testing, the application was able to moderate notifications of the given set of users with high accuracy. However, the steady algorithm tweaks were a necessity when moving forward towards a larger audience.

CHAPTER 6 - CONCLUSION

In the dawn of the age of computers, human behaviour was analysed using multiple sensory setups which costed millions and required specific hardware and software setup. Mobiles are the most used devices by most of the humans in their day to day activities. The applications which provide various capabilities are limitless and humans tend to receive a multitude of notifications each day and more than half of those are just ignored.

Users who receive a bucket load of notifications gets distracted by each and every one of it. It would have a negative impact on the concentration of the user on the tasks that they were performing. What if there was a solution that was able to identify the notifications the users WANT to get, while ignoring the ones that should be IGNORED? [21]

The main objective of this research was to identify small but highly used human behaviour patterns to moderate incoming mobile notifications.

6.1 - Research Contributions

The research had proposed an architectural setup along with a mobile application which can understand the incoming notifications from the user and using human behaviours, and configurations along with the required 3rd party web services to moderate the notifications accordingly.

This would enable any end user to download the application, setup the configurations and start managing their notifications effectively and efficiently. Once the user was happy on how the notifications were managed, he or she can let the application fully manage the notifications and does not get bothered by any kind of unwanted notifications.

Finally, there were several tests carried out to test the performance on the application on several circumstances and the algorithms were adjusted accordingly to such scenarios.

The mechanism of deriving the user acceptance or declining of notifications were based on pre-user preferences and their configurations beforehand. This were the core pillars of this application since without this foundation there was no stableness in the application and the users were not able to predict the outcome of their configurations and in turn resulting in a bad user experience.

The user activity was classified based on several data from different sensors in a typical mobile user setup. Sensors provided the data in turn used up by different 3rd party services which provided information on what the user was doing and where the user was at and what was the weather and other conditions like in each moment. This information was then used to make a sound judgment on how a notification was to be handled if and when it arrives in the mobile. Hence the sensory data was periodically measured for better accuracy as explained in the previous chapter regarding evaluation.

As mentioned in the previous chapters, the main consideration was the use of contextual data to moderate the notifications. The main contextual factors were the user activity and the location information. Users used both such configurations hand in hand to moderate the notifications of the specified applications.

From the user location related analysis, it was noted that most users do not want to get disturbed while in a place of interest such as Work location, Educational Institute location etc. In such cases, the number of hidden notifications were of higher percentage compared to locations like Home.

From the results analysed regarding user activity, most of the users does not want to be disturbed while being in a vehicle. This is a direct case where the user is driving a vehicle.

There were also cases where the users viewed the notifications while the user activity is in a vehicle. This are instances where the user is taking the public commute rather than using his or her private vehicle. Also, there were occurrences where the users viewed notifications while driving that were initiated from their close family members denoting higher importance.

Combining these patterns is not a task that a programmer or a group of programmers can do since there will be several combinations of activities and user locations that can trigger a true or false value for the notification moderation.

Furthermore, the user behavioural analysis would ensure that there are proactive user notifications generated by the application itself ensuring that the user can do their typical day to day activities with a tap of a button or maybe even a simple voice command if the application can be allowed to do so.

All the information gathered would be stored locally and any confidential information on the contents of any notifications were not stored at all considering privacy and security concerns. The configurations were time to time backed up to the cloud such that the user can move from one device to another with minimal hassle and need not to worry about configuring all the settings again from scratch.

In summary, the solution fetched user configurations, the activity and location based data from the sensory setup and information from the 3rd party services and analysed the results and provided a binary output whether the incoming notification was to be displayed or not in the given moment.

6.2 - Research Limitations

The application is still on its early stage of development and could be enhanced in several ways. The application currently uses the capability of only a handful of sensory data and uses limited number of 3rd party services. However, the possibilities are endless when using very high-end smart devices with several sensory setups.

The next limitation of the research is that this was based on a user group of 20 people living in the same area. However, when varying by Country, Region, the expectations would differ, and the algorithms would need to be tweaked accordingly.

As discussed in the evaluation section of this research, sending the complete notification content was a major concern from the users and this was somewhat moderated using extraction of keywords locally. However, in cases where there were a smaller number of keywords in the notification, or the notification contained only images or media attachments, or in a language other than English there were still issues when moderating such notifications. Optimizing the current algorithms was one step towards achieving this but there are limitations when using the mobile processing power when processing heavy algorithms. A balance must be maintained where the overall user experience would not take a hit while processing the notifications locally.

There was another concern that was not addressed by this research in what would happen to the notifications if the user were in a remote area where there is very limited mobile - internet connectivity present. Since, the communication with 3rd party services were out of the equation, the notification displaying mechanism would only be processed locally, in turn limiting the capabilities of the solution.

6.3 - Future Works

In the next phase of this research, more context related information can be used for more finer moderation. Information such as speed of the vehicle, whether conditions can be considered when moderating notifications. This information is already provided by the Google Awareness API but not used in this application.

How to integrate Artificial Intelligence (AI) can also be researched upon in multiple scenarios. If there is AI integrated, the resultant application would be having much more capabilities to understand human behaviour and act accordingly. Furthermore, it would ensure that very less amount of 3rd party services are used since AI can understand the patterns and act upon it without having to rely on inputs from 3rd party services. Usage of AI would ensure, that minimum human feedback is required for moderation and the rule-based engine used for Suggestive Notifications would be something that can be purely based on AI rather than pure programming algorithms [21].

The research, which were similar to this, were using AI in a level where data that was collected via devices but analysed separately. However, with the advent of Machine Learning toolkits available for mobile devices [21], the research could be further enhanced where analysis can be done in the device itself.

As mentioned previously since there are several patterns that emerge for a particular user, it will be a tedious task for a developer to analyse all such data. Also, the logic should also not be based on the data of particular set of users but should be valid for all such scenarios. This is another occurrence where Machine Learning can be used to analyse such patterns and then the logic would use such patterns and could make decisions based on the it.

Using machine learning inside the application would in turn reduce the network bandwidth usage of the application which was a major concern that the application came across. Since

the network bandwidth was used to communicate with 3rd party services, there were some measures taken to lessen the communication, but it still had a high impact.

Another, way to improve the user experience of the solution is to use Voice Based alerts whenever it is necessary. In a scenario where users are in a vehicle driving, rather than popping up a notification, the application can pop-up the notification and read it along and if required ask whether the user wishes to reply to this notification via a voice command. This can be accommodated using Android libraries such as Text to Speech [22].

Sentimental Analysis is another service that is provided by the Cloud Natural Language API and it can be used in a future implementation as well since it would be able to give information regarding the state of the content of an incoming notification. It would enable the user to further customize their notifications based on the sentimental value the service provides [26].

Furthermore, if there were ways to carry out functionalities locally rather than via 3rd party services, it would ensure less network bandwidth and have a positive impact on the overall user experience.

CHAPTER 7 - REFERENCES

[1] Tech by Vice, “Our Brains Are Being Overloaded With Push Notifications About Nothing,” [Online]. Available: https://www.vice.com/en_us/article/a3a848/facebook-notification-overload.

[2] B. N. Schilit, N. Adams and R. Want, “Context-Aware Computing Applications,” in *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, 1994.

[3] A. K. Dey and G. D. Abowd, “Towards a Better Understanding of Context and Context-Awareness,” *Technical report git-gvu-99-22,* pp. 304-307, 2000.

[4] B. N. Schilit and M. M. Theimer, “Disseminating Active Map Information to Mobile Hosts,” *IEEE Network,* vol. 08, no. 05, pp. 22 - 32, 1994.

[5] R. Want, A. Hopper, V. Falcão and J. Gibbons, “The Active Badge Location System,” *ACM Transactions on Information Systems,* vol. 10, no. 1, pp. 91-102, 1992.

[6] P. Casale, O. Pujol and P. Radeva, “Human Activity Recognition from Accelerometer Data Using a Wearable Device,” *IbPRIA 2011: Conference on Pattern Recognition and Image Analysis,* vol. 6669, pp. 289-296, 2011.

[7] K. Farrahi and D. Gatica-Perez, “Discovering Human Routines from Cell Phone Data with Topic Models,” *12th IEEE International Symposium on Wearable Computers, ISWC 2008,* pp. 29-32, 2008.

[8] M. A. Azam, J. Loo, S. K. A. Khan, M. Adeel and W. Ejaz, “Human Behaviour Analysis Using Data Collected from Mobile Devices,” *International Journal on Advances in Life Sciences,* pp. 1 - 10, 2012.

[9] Y. Ma, B. Xu, Y. Bai, G. Sun and R. Zhu, “Daily Mood Assessment based on Mobile Phone Sensing,” *2012 Ninth International Conference on Wearable and Implantable Body Sensor Networks (BSN),* pp. 142-147, 2012.

[10] Q. Gao, D. Fu and X. Dong, “A Context-Aware Mobile User Behavior-Based Neighbor Finding Approach for Preference Profile Construction,” *Sensors,* vol. 16, no. 8, p. 143, 2016.

[11] R. Mayrhofer, H. Radi and A. Ferscha, “Recognizing and Predicting Context by Learning from User Behavior,” *Radiomatics: Journal of Communication Engineering, special issue on Advances in Mobile Multimedia,* vol. 1, no. 1, pp. 30-42, 2004.

[12] H. Bray, “How Location-Based Apps Will Shape the Future of Shopping,” 01 05 2014. [Online]. Available: https://www.discovermagazine.com/technology/how-location-based-apps-will-shape-the-future-of-shopping.

[13] T. Sohn, K. A. Li, G. Lee, I. Smith, J. Scott and W. G. Griswold, “Place-Its: A Study of Location-Based Reminders on Mobile Phones,” *UbiComp 2005: Ubiquitous Computing.g, 7th International Conference,* vol. 3660, pp. 232-250, 2005.

[14] S. Buthpitiya, D. Madamanchi, S. Kommaraju and M. Griss, “Mobile Context-Aware Personal Messaging Assistant,” *Mobile Computing, Applications, and Services,* vol. 35, pp. 254-272, 2010.

[15] A. K. Dey and G. D. Abowd, “CybreMinder: A Context-Aware System for Supporting Reminders,” *Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing (HUC2K), Bristol, UK,* vol. 1927, pp. 172-186, 2000.

[16] A. K. Dey and G. D. Abowd, “A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications,” *Human-Computer Interaction,* vol. 16, no. 2-4, pp. 97-166, 2001.

[17] A. Mehrotra, M. Musolesi, R. Hendley and V. Pejovic, “Designing Content-driven Intelligent Notification Mechanisms for Mobile Applications,” *UbiComp '15: Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing,* pp. 813-824, 2015.

[18] A. Mehrotra, V. Pejovic, J. Vermeulen, R. Hendley and M. Musolesi, “My Phone and Me: Understanding People’s Receptivity to Mobile Notifications,” *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI ’16),* pp. 1021-1032, 2016.

[19] A. Mehrotra, R. Hendley and M. Musolesi, “Interpretable Machine Learning for Mobile Notification Management: An Overview of PrefMiner,” *GetMobile: Mobile Computing and Communications,* vol. 21, no. 2, pp. 35-38, 2017.

[20] Google, “Google Awareness API | Google Developers,” Google, [Online]. Available: https://developers.google.com/awareness.

[21] Google, “Cloud Natural Language API | Google Cloud,” Google, [Online]. Available: https://cloud.google.com/natural-language/.

[22] Google, “Cloud Firestore | Firebase,” Google, [Online]. Available: https://firebase.google.com/docs/firestore.

[23] Oracle, “What Are RESTful Web Services?,” Oracle, [Online]. Available: https://docs.oracle.com/javaee/6/tutorial/doc/gijqy.html.

[24] Geeks for Geeks, “Serialization and Deserialization in Java with Example,” Geeks for Geeks, [Online]. Available: https://www.geeksforgeeks.org/serialization-in-java/.

[25] Android Developers, “Notifications Overview | Android Developers,” [Online]. Available: https://developer.android.com/guide/topics/ui/notifiers/notifications.

[26] K. Rijensky, “To Push, or Not to Push? Designing Proactive Messages in Chatops,” 27 02 2018. [Online]. Available: https://chatbotsmagazine.com/to-push-or-not-to-push-designing-proactive-messages-in-chatops-38d61893c738.

[27] P. Kordík, “Personalized push notifications enabled by artificial intelligence,” 04 03 2018. [Online]. Available: https://medium.com/recombee-blog/personalized-push-notifications-enabled-by-artificial-intelligence-8ac057bc97ba.

[28] Android Developers, “Machine Learning | Android Developers,” [Online]. Available: https://developer.android.com/ml.

[29] Android Developers, “TextToSpeech | Android Developers,” [Online]. Available: https://developer.android.com/reference/android/speech/tts/TextToSpeech.

[30] Google, “Analyzing Sentiment | Cloud Natural Language API | Google Cloud,” Google, [Online]. Available: https://cloud.google.com/natural-language/docs/analyzing-sentiment#language-sentiment-string-java.