

*'In Pursuit of Technical Excellence'*

Project Report On

**“SMART NOTIFICATION SYSTEM TO REDUCE PUSH  
NOTIFICATION”**

In partial fulfilment of requirements for the degree of

Bachelor of Engineering

In

Computer science and Engineering

Submitted by

**Darshan Chobarkar (BE15F05F016)**

**Aalekh Mudiraj (BE15F05F041)**

**Aditya Ganvir (BE15F05F027)**

**Saurabh Lodha (BE15F05F061)**

Under the Esteemed Guidance of

**Prof. Arjumand Khan**



**DEPARTMENT OF COMPUTER SCIENCE AND  
ENGINEERING**

**Government College of Engineering, Aurangabad**

**(An Autonomous Institute of the Government of Maharashtra)**

**2018-19**

# CERTIFICATE

This is to certify that the project entitled “**SMART NOTIFICATION SYSTEM TO REDUCE PUSH NOTIFICATION**” which is being submitted for the final project of B.E. in “Computer Science and Engineering” of Government college of Engineering, Aurangabad.

This is the report work of the project presented by **Mr. Darshan Chobarkar, Mr. Anup Deshmukh, Mr. Aditya Ganvir & Mr. Saurabh Lodha** under my supervision and guidance during the Academic Year 2018-19. The work embodied in this report is not formed earlier for the basis of the award of degree or compatible certificate or similar title of this for any other diploma examining body of University to the best of my knowledge and belief.

**Place: Aurangabad**

**Date: \_\_\_\_/\_\_\_\_/ 2019**

**Prof. Arjumand Khan**

Project Guide

(Computer Science and Engineering)

**Dr. V. P. Kshirsagar**

H.O.D.

(Computer Science and Engineering)

**Dr. P. B. Murnal**

**Principal**

Government College of Engineering, Aurangabad.

# ACKNOWLEDGEMENT

It is great pleasure that we express our sincere and hearty gratitude to **Prof. Arjumand Khan** for their guidance and constant encouragement during the completion of whole project on “**SMART NOTIFICATION SYSTEM TO REDUCE PUSH NOTIFICATIONS**”.

The project report on “**SMART NOTIFICATION SYSTEM TO REDUCE PUSH NOTIFICATIONS**” is outcome of guidance, moral support and devotion bestowed on me throughout my work. For this we acknowledge and express our profound sense of gratitude and thanks to everybody who have been a source of inspiration during project preparation. No words could be good enough to express my deep gratitude to our respected Principal **Dr. P. B. Murnal** for his kind blessings, inspiration and providing me the necessary support.

It feels great pleasure in expressing our deepest sense of gratitude and sincere thanks to our head of department (Computer science and Engineering) **Dr. V. P. Kshirsagar** for his valuable guidance during the course of project without which it would have been very difficult task. This acknowledgement would be incomplete without expressing my special thanks to our guide for her support during the work.

We would like to thank all the teaching staff, non-teaching staff members of the department and our colleagues those who helped us directly or indirectly for completing task successfully.

Darshan Chobarkar

Anup Deshmukh

Aditya Ganvir

Saurabh Lodha

# TABLE OF CONTENTS

1 INTRODUCTION	01
1.1 Project Profile	01
1.1.1 Developed by	01
1.1.2 Hardware/Software Requirement	01
1.1.3 Development Tool	01
1.1.4 Documentation & Presentation Tool	01
1.2 System Overview	02
1.3 Scope & Objective	02
1.4 Abstract	03
2 SYSTEM DEVELOPMENT	04
2.1 Project Development Approach	04
2.1.1 Software Process Model	04
2.2 Risk Analysis	05
2.2.1 Risk Management Process	06
3 LITERATURE SURVEY	08
3.1 Recommendation System	08
3.1.1 Definition	08
3.1.2 Overview	08
3.1.3 History	09
3.2 About Push Notification	10
3.2.1 What are Push Notification	10
3.2.2 Why are They Used	10
3.2.3 How Do Push Notification Work?	11
3.3 Naïve Bayesian Algorithm	12
3.4 Rest	13
3.5 Flask	14
3.5.1 Werkzeug	14
3.5.2 Jinja	15

3.6 SQLite	15
3.7 Firebase Cloud Messaging (FCM)	16
 4 WORKING OF THE PROPOSED SYSTEM	 18
4.1 Working of the Recommendation System	18
4.2 Finding Out the Best Time of sending the Notification Based on User Response to Send Notification	21
4.2.1 Predicting Best Time of Sending Notification by Accounting User Movement and Locomotion	23
4.3 Predicting Probability of Purchase of a Product in Future	24
4.4 System Diagrams	26
4.4.1 Use Case Diagram	26
4.4.2 Data Flow Diagram	27
4.4.3 Tech Stack	27
 5 TABLES	 28
5.1 Database Design	31
5.1.1 Normalization	32
5.1.2 Data Dictionary	33
 6 OUTPUTS	 34
 7 TEST CASES	 44
 8 PROPOSED COSTING	 45
 9 CONCLUSION	 47
 10 BIBLIOGRAPHY & REFERENCES	 48

# 1. INTROUDUCTION

## 1.1 Project Profile

### 1.1.1 Developed By

This “Smart Notification System” built in Flask framework is developed by Mr. Saurabh Lodha, Mr. Anup Deshmukh, Mr. Darshan Chobarkar and Mr. Aditya Ganvir.

### 1.1.2 Hardware/Software Environment

#### *Hardware Requirements*

Processor	:	Pentium 4.0 (1.6GHz)
RAM	:	1GB Minimum
Hard Disk	:	10GB

#### *Software Requirements*

Web-Technologies	:	Flask to connect Android and SQLite3
Database	:	SQLite3 and PostgreSQL
Language	:	Python, Java

### 1.1.3 Development Tool

Android Studio  
Postman  
Spyder  
Firebase  
Heroku

### 1.1.4 Documentation and Presentation Tool

Microsoft Word 2016  
Microsoft PowerPoint 2016

## 1.2 System Overview

Smart Notification System is built in Flask framework using Python and acts as a service behind the Android application of the e-commerce website. The database is built in SQLite3 and the Push notification as a form of Cloud Messaging Service is provided by Firebase. Finally, the whole project follows REST architectural style.

The Smart Notification System is built to serve the following areas:

- To improve the quality of notifications received at the user end and deliver them at the right time.

- A Recommendation System to provide the best recommendations specific to the user interest.

- A timeline of user buying probability of products in real time and future aimed to benefit business factors.

- User response to key events and notifications as feedback or others improves the algorithm to deliver better quality of notifications.

## 1.3 Scope and Objective

- Use Flask micro framework to quickly scale projects and manage database in the simplest way by SQLite3.

- Improve the quality of notification at the user end as well leveraging the benefits of it at the business ends.

- Create a smart notification service that could be seamlessly be integrated into any e-commerce application.

- Build an automated user intent based notification delivery system that could replace the current trigger and time based notifications.

## 1.4 Abstract

The purpose of this project was getting the best out of sending notifications. Push Notifications are a special type of notifications and are used by developers to promote their content, keep their users engaged to the app, reduce app churn-out rates, send out vital app content, access user opinions and data, etc. The problem is their untimely usage and sending them without any proper strategy. Receiving unrelated and excessive notifications creates a negative image of the application and in time increase app drop-out rates. Thus the need for building a system which maps user intent in a proper way about his purchase habits, his best time of receiving notifications and others needed to be in place. This project thus focuses on building a strong Recommendation System which forms as a skeleton of finding out where and how the user spends his time, what elements of the app he interacts with and thus map the best product he is interested in. This is done by employing layers of Machine Learning models like Naive Bayes classification and Regression algorithms and built on a large dataset of products fetched from an e-commerce website. Further, his response to app notifications, his daily habits and app usage, his mobility, etc. was researched and curated into algorithms for predicting the best time of sending the notification to him. These two modules form part of building the Smart Notification System and solving the problems. Lastly, the project also contains a very interesting module which aims for predicting the probability of purchase of a particular product at a particular time in Future. This system could be employed for so many business usages for e-commerce applications. For targeting the products across a time period having a higher probability of getting sold, then leveraging them to target the suitable user base for them and then further sending them the best notification at the right time will reap benefits all around.

Also, it will lead to an increase in user's interaction with the app, increase in app-retention rates and a further increase in popularity of the application due to the dedicated customer base and their quality reviews on it. Thus the project aims to fulfil these objectives while solving the mentioned problems.



## 2 SYSTEM DEVELOPMENT

### 2.1 Project Development Approach

#### 2.1.1 Software Process Model

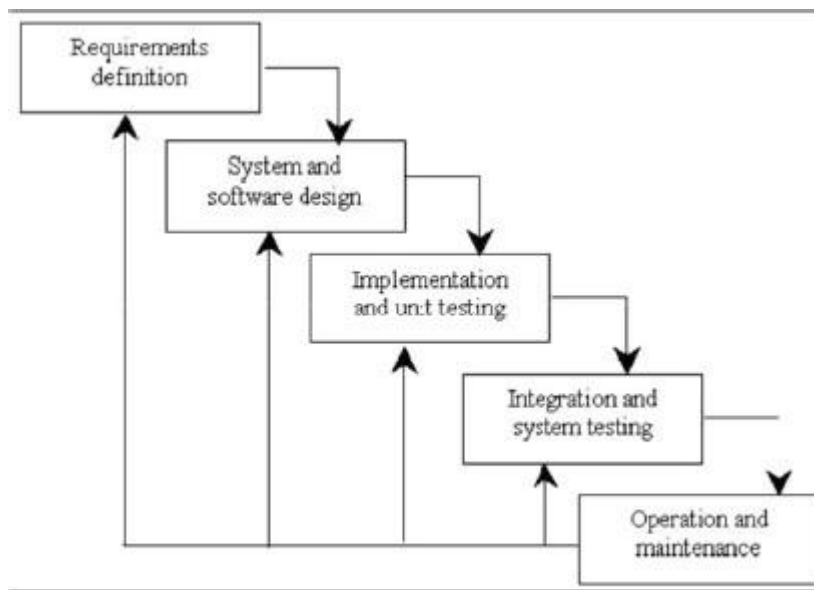
To solve actual problem in an industry, software developer and a team of developers must incorporate a development strategy that encompasses the process, methods and tools layers and generic phases. This strategy is often referred to as process model or a software developing paradigm. A process model for software developing is chosen based on nature of project and application, the methods and tools to be used, and the controls and deliverables that are required. All software development can be categorized as problem solving loop in which four distinct stages are encountered: status quo, Problem definition, technical development and solution integration. Regardless of the process model that is chosen for a software project all of the stages coexist simultaneously at some level of detail.

Our Project Follows the Waterfall Model.

#### The Waterfall Model

The steps of the typical waterfall model are:

- Requirement Definition
- System and software design
- Implementation
- Integration & System Testing
- Operation & Maintenance



*Figure 1: The Waterfall Model*

## 2.2 Risk Analysis

Risk is one of the main job of project manager. It involves anticipating risk that might affect the project schedule or quality of software being developed & taking action to avoid these risks.

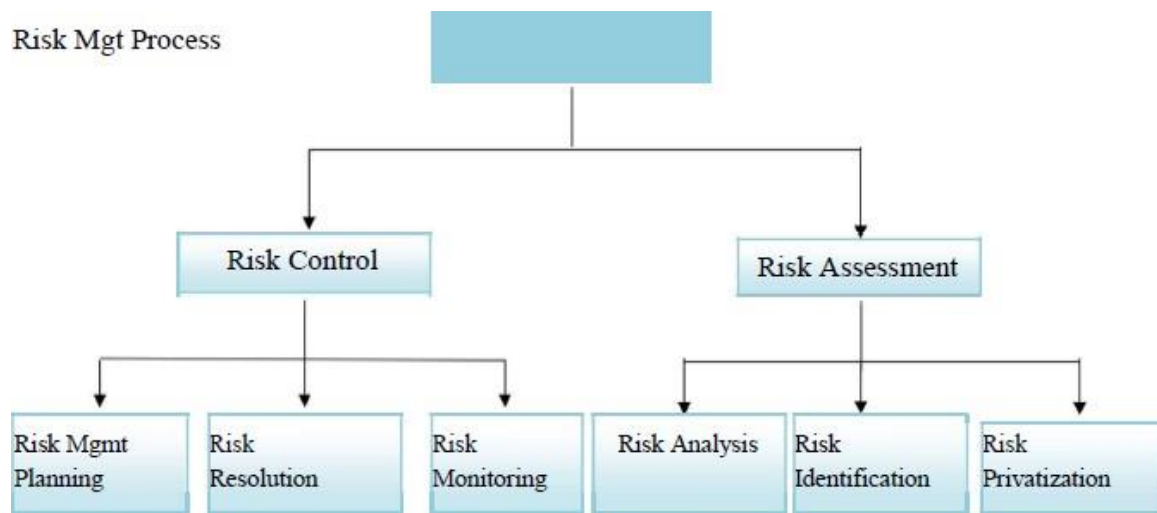
The result of the risk should document in project plan.

There are three categories of Risk:

- Project Risk  
That affects the schedule or resources
- Product Risk  
That affects the quality or performance of the software being developed.
- Business Risk  
That affects organization developing & procuring the software.

The risk that may affect the project is depends on the project & organization environment where the software is developed.

The following figure shows Risk Management activities.



## 2.2.1 Risk Management Process

Risk management is particularly important for software project because of the inherent uncertainties that most project face.

Risk occurs because loosely defined requirement problems in estimating time and resource to the project inexperienced software team etc.

You have to anticipate the risk understand the impact of the risk on the project.

You may need to draw a contingency plan so that, if risk do occur you can take immediate recovery action.

The process of risk management is illustrated in above diagram. It involves several following stages:

### 2.2.1.1 Risk Identification

Risk identification is the first step of risk management process. It concerned with discovering possible risk to the project.

### 2.2.1.2 Risk analysis

During analysis process, you have to consider each identified risk & make a judgment about probability & seriousness of it.

#### 2.2.1.3 Risk planning

The risk planning process considers each of the key risks that have been identified and identifies strategies to manage the risk.

#### 2.2.1.4 Risk monitoring

Risk monitoring involves regularly assessing each of the identified risk to decide whether or not that risk is becoming more or less probable & whether the effects of the risk has to look of other function.

## 3. Literature Survey

### 3.1 Recommendation System

#### 3.1.1 Definition

A **recommender system** or a **recommendation system** (sometimes replacing "system" with a synonym such as platform or engine) is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item.

Recommender systems are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general. There are also recommender systems for experts, collaborators, jokes, restaurants, garments, financial services, life insurance, romantic partners (online dating), and Twitter pages.

#### 3.1.2 Overview

Recommender systems typically produce a list of recommendations in one of two ways – through collaborative filtering or through content-based filtering (also known as the personality-based approach). Collaborative filtering approaches build a model from a user's past behaviour (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that the user may have an interest in. Content-based filtering approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties.

The differences between collaborative and content-based filtering can be demonstrated by comparing two popular music recommender systems – Last.fm and Pandora Radio.

- Last.fm creates a "station" of recommended songs by observing what bands and individual tracks the user has listened to on a regular basis and comparing those against the listening behavior of other users. Last.fm will play tracks that do not appear in the user's library, but are often played by other users with similar interests. As this approach leverages the behavior of users, it is an example of a collaborative filtering technique.
- Pandora uses the properties of a song or artist (a subset of the 400 attributes provided by the Music Genome Project) to seed a "station" that plays music with similar properties. User feedback is used to refine the station's results, deemphasizing certain attributes when a user "dislikes" a particular song and emphasizing other attributes when a user "likes" a song. This is an example of a content-based approach.

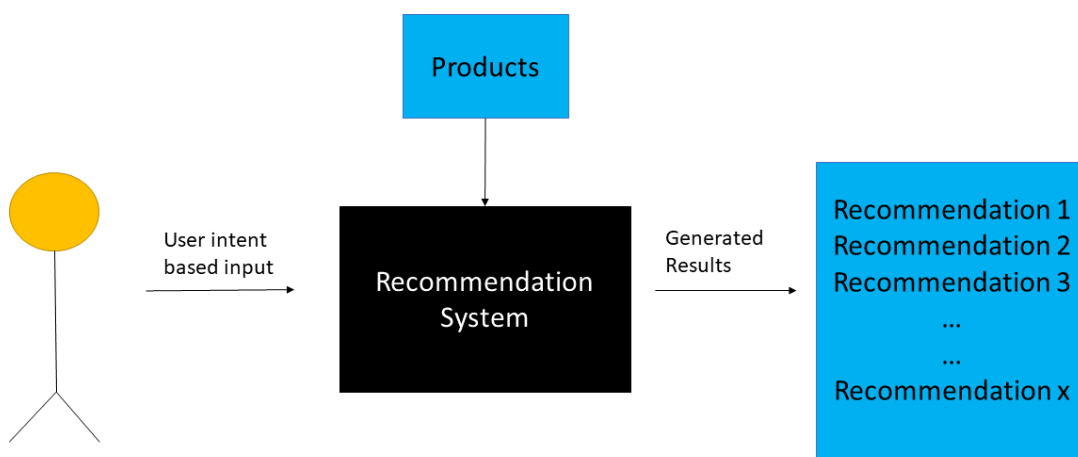
Each type of system has its strengths and weaknesses. In the above example, Last.fm requires a large amount of information about a user to make accurate recommendations. This is an example of the cold start problem, and is common in collaborative filtering systems. Whereas Pandora needs very little information to start, it is far more limited in scope (for example, it can only make recommendations that are similar to the original seed).

Recommender systems are a useful alternative to search algorithms since they help users discover items they might not have found otherwise. Of note, recommender systems are often implemented using search engines indexing non-traditional data.

### 3.1.3 History

Recommender systems were first mentioned in a technical report as a "digital bookshelf" in 1990 by Jussi Karlgren at Columbia University, and implemented at scale and worked through in technical reports and publications from 1994 onwards by Jussi Karlgren, then at SICS, and research groups led by Pattie Maes at MIT, Will Hill at Bellcore, and Paul Resnick, also at MIT whose work with GroupLens was awarded the 2010 ACM Software Systems Award.

Montaner provided the first overview of recommender systems from an intelligent agent perspective. Adomavicius provided a new, alternate overview of recommender systems. Herlocker provides an additional overview of evaluation techniques for recommender systems, and Beel et al. discussed the problems of offline evaluations. Beel et al. have also provided literature surveys on available research paper recommender systems and existing challenges.



Building of the Recommendation System

## 3.2 About Push Notifications

### 3.2.1 What are Push Notifications

A push notification is a message that pops up on a mobile device. App publishers can send them at any time; users don't have to be in the app or using their devices to receive them. They can do a lot of things; for example, they can show the latest sports scores, get a user to take an action, such as downloading a coupon, or let a user know about an event, such as a flash sale.

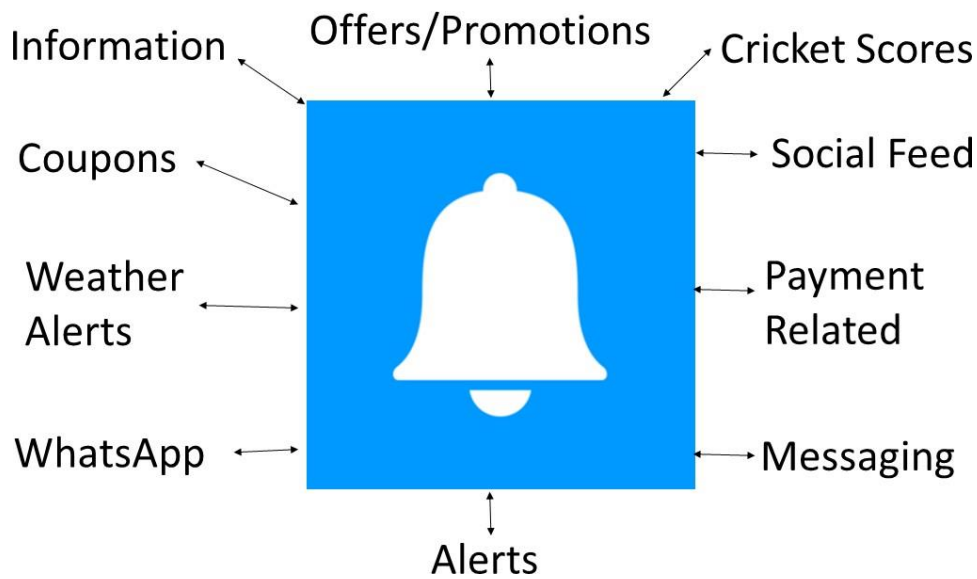
Push notifications look like SMS text messages and mobile alerts, but they only reach users who have installed your app. Each mobile platform has support for push notifications — iOS, Android, Fire OS, Windows and BlackBerry all have their own services.

### 3.2.2 Why are they used

Push notifications provide convenience and value to app users. For example, users can receive:

- Sports scores and news right on their lock screen
- Utility messages like traffic, weather and ski snow reports
- Flight check in, change, and connection information

For app publishers, push notifications are a way to speak directly to a user. They don't get caught in spam filters, or forgotten in an inbox — click-through rates can be twice as high as email. They can also remind users to use an app, whether the app is open or not.



### 3.2.3 How do push notifications work?

Some of the actors in sending a push notification include:

- **Operating system push notification service (OSPNS).** Each mobile operating system (OS), including iOS, Android, Fire OS, Windows, and BlackBerry, has its own service.
- **App publisher.** The app publisher enables their app with an OSPNS. Then, the publisher uploads the app to the app store.
- **Client app.** This is an OS-specific app, installed on a user's device. It receives incoming notifications.



### 3.3 NAÏVE BAYESIAN ALGORITHM

The Bayesian Classification represents a supervised learning method as well as a statistical method for classification. Assumes an underlying probabilistic model and it allows us to capture uncertainty about the model in a principled way by determining probabilities of the outcomes. It can solve diagnostic and predictive problems. This Classification is named after Thomas Bayes (1702-1761), who proposed the Bayes Theorem. Bayesian classification provides practical learning algorithms and prior knowledge and observed data can be combined. Bayesian Classification provides a useful perspective for understanding and evaluating many learning algorithms. It calculates explicit probabilities for hypothesis and it is robust to noise in input data.

Uses of Naive Bayes classification:

1. Naive Bayes text classification (<http://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html>). The Bayesian classification is used as a probabilistic learning method (Naive Bayes text classification). Naive Bayes classifiers are among the most successful known algorithms for learning to classify text documents.
2. Spam filtering ([http://en.wikipedia.org/wiki/Bayesian\\_spam\\_filtering](http://en.wikipedia.org/wiki/Bayesian_spam_filtering)) Spam filtering is the best known use of Naive Bayesian text classification. It makes use of a naive Bayes classifier to identify spam e-mail. Bayesian spam filtering has become a popular mechanism to distinguish illegitimate spam email from legitimate email (sometimes called "ham" or "bacn"). Many modern mail clients implement Bayesian spam filtering. Users can also install separate email filtering programs. Server-side email filters, such as DSPAM, SpamAssassin, SpamBayes, Bogofilter and ASSP, make use of Bayesian spam filtering techniques, and the functionality is sometimes embedded within mail server software itself.
3. Hybrid Recommender System Using Naive Bayes Classifier and Collaborative Filtering (<http://eprints.ecs.soton.ac.uk/18483/>) Recommender Systems apply machine learning and data mining techniques for filtering unseen information and can predict whether a user would like a given resource. It is proposed a unique switching hybrid recommendation approach by combining a Naive Bayes classification approach with the collaborative filtering. Experimental results on two different data sets, show that the proposed algorithm is scalable and provide better performance in terms of accuracy and coverage than other algorithms while at the same time eliminates some recorded problems with the recommender systems.
4. Online applications (<http://www.convo.co.uk/x02/>) This online application has been set up as a simple example of supervised machine learning and affective computing. Using a training set of examples which reflect nice, nasty or neutral sentiments, we're training Ditto to distinguish between them. Simple Emotion Modelling, combines a statistically based classifier with a dynamical model. The Naive Bayes classifier employs single words and word pairs as features. It allocates user utterances into nice, nasty and neutral classes, labelled +1, -1 and 0 respectively. This numerical

output drives a simple first-order dynamical system; whose state represents the simulated emotional state of the experiment's personification.

### 3.4 Rest

A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data.

A RESTful API, also referred to as a RESTful web service, is based on representational state transfer (REST) technology, an architectural style and approach to communications often used in web services development.

REST technology is generally preferred to the more robust Simple Object Access Protocol (SOAP) technology because REST leverages less bandwidth, making it more suitable for internet usage. An API for a website is code that allows two software programs to communicate with each another. The API spells out the proper way for a developer to write a program requesting services from an operating system or other application.

The REST used by browsers can be thought of as the language of the internet. With cloud use on the rise, APIs are emerging to expose web services. REST is a logical choice for building APIs that allow users to connect and interact with cloud services. RESTful APIs are used by such sites as Amazon, Google, LinkedIn and Twitter.

#### **How RESTful APIs work**

A RESTful API breaks down a transaction to create a series of small modules. Each module addresses a particular underlying part of the transaction. This modularity provides developers with a lot of flexibility, but it can be challenging for developers to design from scratch. Currently, the models provided by Amazon Simple Storage Services, Cloud Data Management Interface and OpenStack Swift are the most popular.

A RESTful API explicitly takes advantage of HTTP methodologies defined by the RFC 2616 protocol. They use GET to retrieve a resource; PUT to change the state of or update a resource, which can be an object, file or block; POST to create that resource; and DELETE to remove it.

With REST, networked components are a resource you request access to a black box whose implementation details are unclear. The presumption is that all calls are stateless; nothing can be retained by the RESTful service between executions.

Because the calls are stateless, REST is useful in cloud applications. Stateless components can be freely redeployed if something fails, and they can scale to accommodate load changes. This is because any request can be directed to any instance of a component; there can be nothing saved that has to be remembered by the next transaction. That makes REST preferred for web use, but the RESTful model is also helpful in cloud services because binding to a service through an API is a matter of controlling how the URL is decoded. Cloud Computing and micro services are almost certain to make RESTful API design the rule in the future.

## 3.5 Flask

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common frameworks related tools. Extensions are updated far more regularly than the core Flask program.

Applications that use the Flask framework include Pinterest, LinkedIn and the community web page for Flask itself.

The micro framework Flask is based on the *Pocoo* projects *Werkzeug* and *Jinja2*.

### 3.5.1 Werkzeug

Werkzeug is a utility library for the Python Programming Language, in other words a toolkit for Web Server Gateway Interface (WSGI) applications, and is licensed under a BSD License. Werkzeug can realize software objects for request, response, and utility functions. It can be used to build a custom software framework on top of it and supports Python 2.6, 2.7 and 3.3.

### 3.5.2 Jinja

Jinja, also by Ronacher, is a template engine for the Python programming language and is licensed under a BSD License. Similar to the Django web framework, it provides that templates are evaluated in a sandbox.

#### Features

---

- Contains development server and debugger
- Integrated support for unit testing
- RESTful request dispatching
- Uses Jinja2 templating
- Support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode -based
- Extensive documentation
- Google App Engine compatibility
- Extensions available to enhance features desired

### 3.6 SQLite

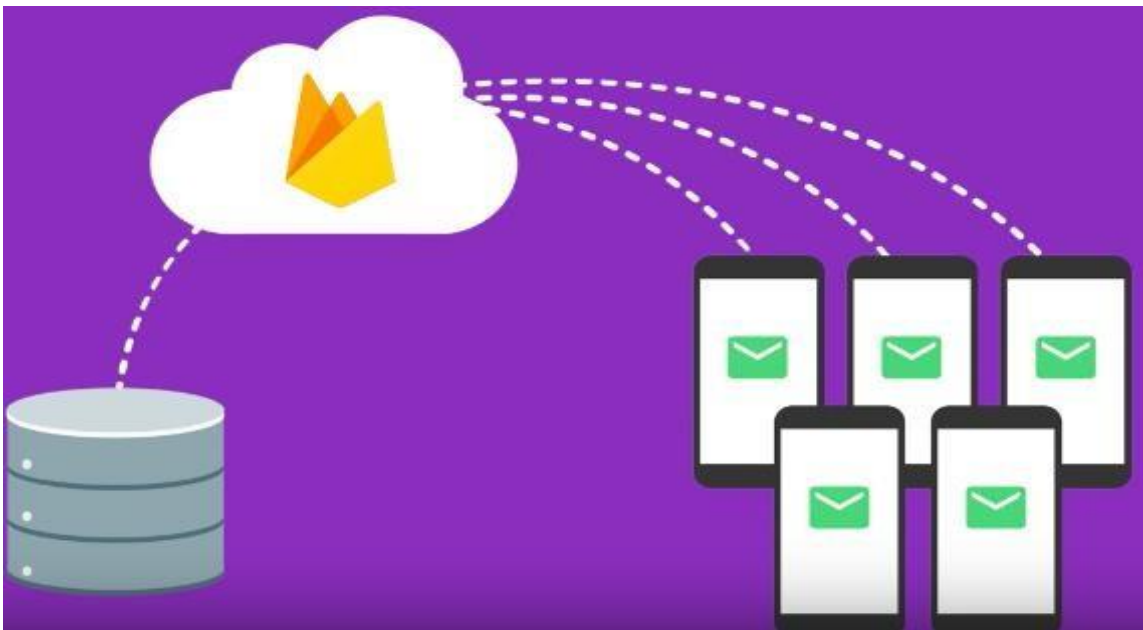
SQLite is a Relational Database Management System contained in a C Programming Library. In contrast to many other database management systems, SQLite is not a client - server database engine. Rather, it is embedded into the end program.

SQLite is ACID - compliant and implements most of the SQL standard, generally following PostgreSQL syntax. However, SQLite uses a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity. This means that one can, for example, insert a string into a column defined as an integer. SQLite will attempt to convert data between formats where appropriate, the string "123" into an integer in this case, but does not guarantee such conversions, and will store the data as-is if such a conversion is not possible.

SQLite is a popular choice as embedded database software for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsers, operating systems, and embedded systems (such as mobile phones), among others. SQLite has binding to many programming languages.



### 3.7 Firebase Cloud Messaging (FCM)



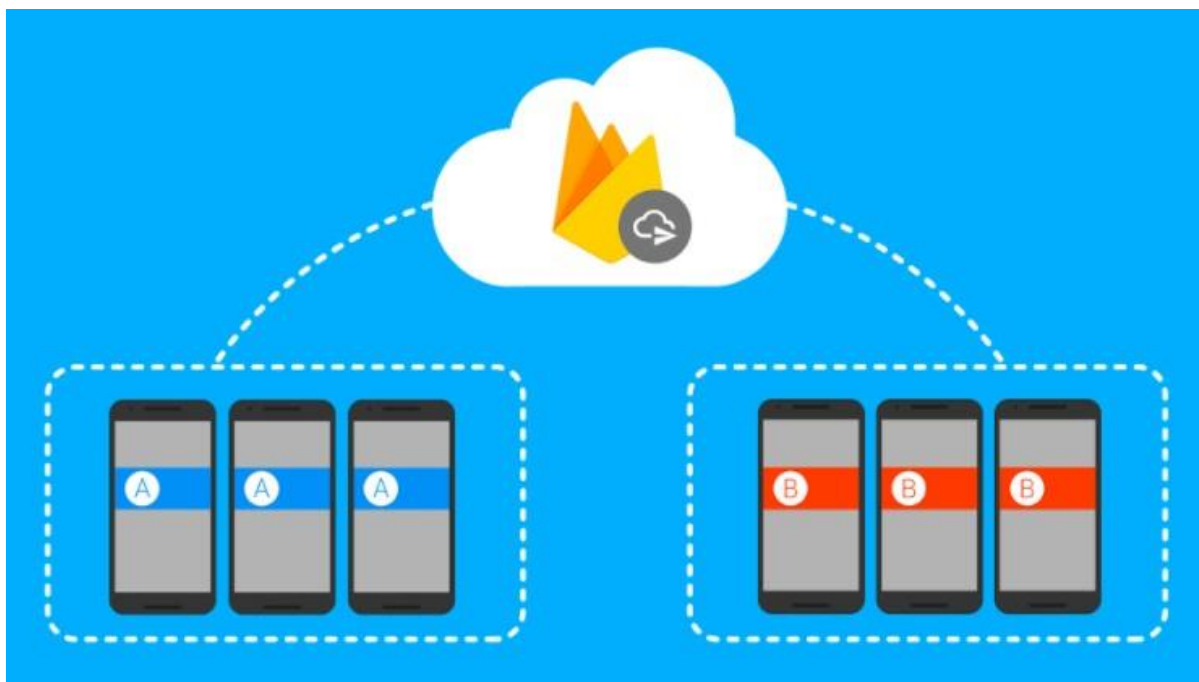
Firebase Cloud Messaging (FCM) provides a reliable and battery-efficient connection between your server and devices that allows you to deliver and receive messages and notifications on iOS, Android, and the web at no cost.

You can send notification messages (2KB limit) and data messages (4KB limit).

Using FCM, you can easily target messages using predefined segments or create your own, using demographics and behaviour. You can send messages to a group of devices that are subscribed to specific topics, or you can get as granular as a single device.

FCM can deliver messages instantly, or at a future time in the user's local time zone. You can send custom app data like setting priorities, sounds, and expiration dates, and also track custom conversion events.

The best thing about FCM is that there is hardly any coding involved! FCM is completely integrated with Firebase Analytics, giving you detailed engagement and conversion tracking.



You can also use A/B testing to try out different versions of your notification messages, and then select the one which performs best against your goals.

## 4.0 WORKING OF THE PROPOSED SYSTEM

### 4.1 Working of the Recommendation System:

The Recommendation System is the crux of the First Module. For delivering the best product notification, the one the user is interested in and is most probable to buy, we build the correlations among the different products that are present in the Database. Due to this correlation only that we can find the following useful things:

- The section the user is targeting,
- The particular product base the user is interested in and finally narrow down to
- A single best product he has the highest interest in and has the highest probability of purchasing.

So from the database of products we have the top-level categories like Electronics, Fashion, Grocery, Appliances etc. so these act as the Parent Nodes of a tree where the leaf nodes are the individual products. The layers of the tree are the further bifurcations into more specific categories which help in narrowing down to individual products. Thus every product has a unique element identification and is linked to a particular Parent id.

Now the first steps were to find out the children of a particular Parent Node. This will help to link the children nodes together as they all belong to the same parent or category.

Thus the following steps were followed:

- Importing the Products\_Database.csv file and then making a dataframe element of two columns i.e. Product id and Parent id.
- Based on these, all the database products were looped over and the Product ids of the elements having the same Parent were grouped together.
- A column was created “Children’ containing the list of all the Product ids having the same parents and were linked to that particular element. To explain it more clearly, the Category Samsung Phones had all the products of Samsung mobiles.
- For the elements, finally we had how they were related to each other and along with the parent categories.

Now there was a need of a correlation factor among all the elements, to show the strength of similarities among elements.

So the similarity index is classified as following:

Sr No.	Relationship Between Elements	Correlation Index
1	Same product ( Element to Element)	1.0
2	Element to element of same parent. ( Depth = 1)	0.6
3	Element to element of same parent ( Depth = 2)	0.25
4	Element to element with different parent of different categories	0.01

Thus this lays down the co-relations among all the products. So for a particular product, we can have the best recommended and the most significant products related to it.

To explain this more further, a product “Apple iPhone 32GB” will generally have the recommendations of phones of the same budget, same company and same variants and our algorithm predicts the same and these were the predictions for the same -

This thus forms the skeleton of the main idea. Now since the products relations are done, we need to consider the second most important factor along with it.

‘Time spent interacting with the product’

Now there are various ways of capturing the time of interaction. We have decided to capture time in the following ways and account time in units pertaining to their importance and it is represented in the following table -

Sr No.	Type of Activity Performed	Value in Units
1	Opted in for the product.	2
2	Checking out images of the product.	2
3	Checking details of the product.	3
4	Checking out the reviews of the product.	3

Thus the time and the correlation factor are set up individually and thus the main thing is to find out how searching for a particular element and seeing it impacts the complete dataset of the products. And if we assign a particular measure for this and based on this we calculate the best among the directory of dataset, we get the best product the user would be interested in.

This measure is Naive Bayes Algorithm.



We use it to find out the probability for every searched product. We consider a threshold of last 10 searched elements in History and for every element, we see how the product and time spent on it affects the catalogue of products. Also one more measure is the popularity of the product i.e how many times it has been purchased in total with respect to the total purchases made on the App. This will also have its share of influence in impacting the probability of the output. Thus combining these values to give a final sum, we get the best product for our notification.

To summarize it, these are the steps of the algorithm.

- 1) Scrap the last 10 elements of the user search history and attached with it the time spent on them.
- 2) The outside loop consists of these 10 elements and a inner loop consists of all the products in the database.
- 3) The Naive Bayes Algorithm can be represented as -

$$P(c | x) = \frac{P(x | c)P(c)}{P(x)}$$

Likelihood
Class Prior Probability  
Posterior Probability
Predictor Prior Probability

$$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$$

The Likelihood and the Predictor Probability is derived from the correlation index and the class prior probability is given by the ratio of total purchases of that product and the total purchases of all time.

- 4) We scale this value properly so that the values are of proper dimension and we get a final probability value for all the products in the database.
- 5) The product having the highest value of probability is the best product.

The Algorithm outputs the product that is best suited for the notification content and body. This is the first module and it takes in the best practices of Machine Learning and its algorithms to produce the result.

The second important module is finding the best time to send the notification.

## 4.2 Finding out the Best Time of sending the Notification based on user response to send notifications

Predicting the best time for sending the notification is built by analysing user habits of interacting with the already sent notifications. What action he chooses is very important, because based on that we can find out whether he was interested in the particular notification or not, whether he got the notification at the right time or not and finally when is the actual right time of sending the notification. Thus every notification we send has to have two main buttons - Dismiss and Opt-in where the choice of dismiss button is linked to 0 and if the user opts in for the notification, it is taken as 1 in the database. Also the time of the action like the time of response to the notification is taken into account which gives a very good view of the time the user is active and will probably respond to the notification. Finally, the number of notifications the user had on his phone prior to seeing the app notification. Generally, this too has an impact on the response to our notification as if there are more notification already, there is a high chance the user will clear it all. So this factor also needs to take care of.

Thus a dataset was built for the algorithm based on the following points and a view of it has been represented below -

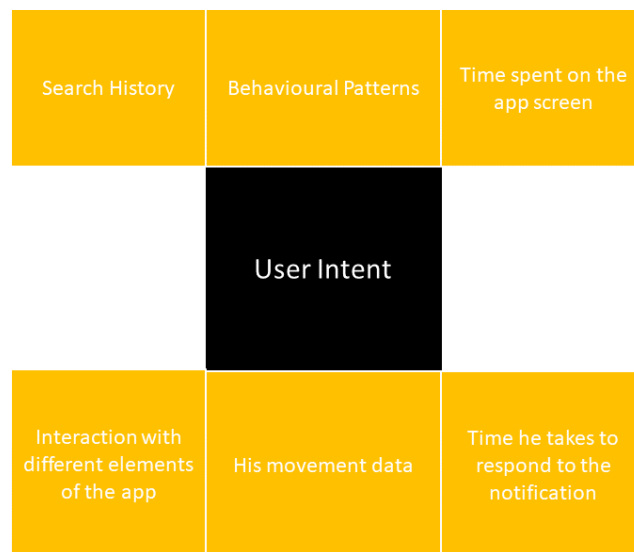
Now comes the main part, analysing this data and finding out the best time of the notification. For this the steps of the algorithm are:

- Dividing 24 hours of time into slots of 10 minutes each and then assigning them a different scaled factor i.e.

Time	Scaled Factor
00:00	0
00:10	1
00:20	2
00:30	3
00:40	4

- Next we calculate the difference between the actual time of the notification and the response time of the user in units and store it in the particular slot window. Also a factor influencing the calculation here is the number of notifications on the phone at that point of time.
- The value calculated in step 2 coupled with the number of user responded notifications in that time slot gives a final value for the particular time slot.
- The time slot with the highest value is the time slot corresponding with the best suitable time we have to send the notification at.

Based on the above dataset the algorithm after working on the steps outputs the following time. This will be the best time of the sending of the notification.



#### 4.2.1 Predicting Best Time of sending notification by accounting user movement and locomotion.

In a general case A user when moving from one point to other with certain speed and covering some distance not possible by walking, we can assume that the user is travelling. Based on many reports and analytics, users tend to see notifications when they are free and this includes when they are travelling.

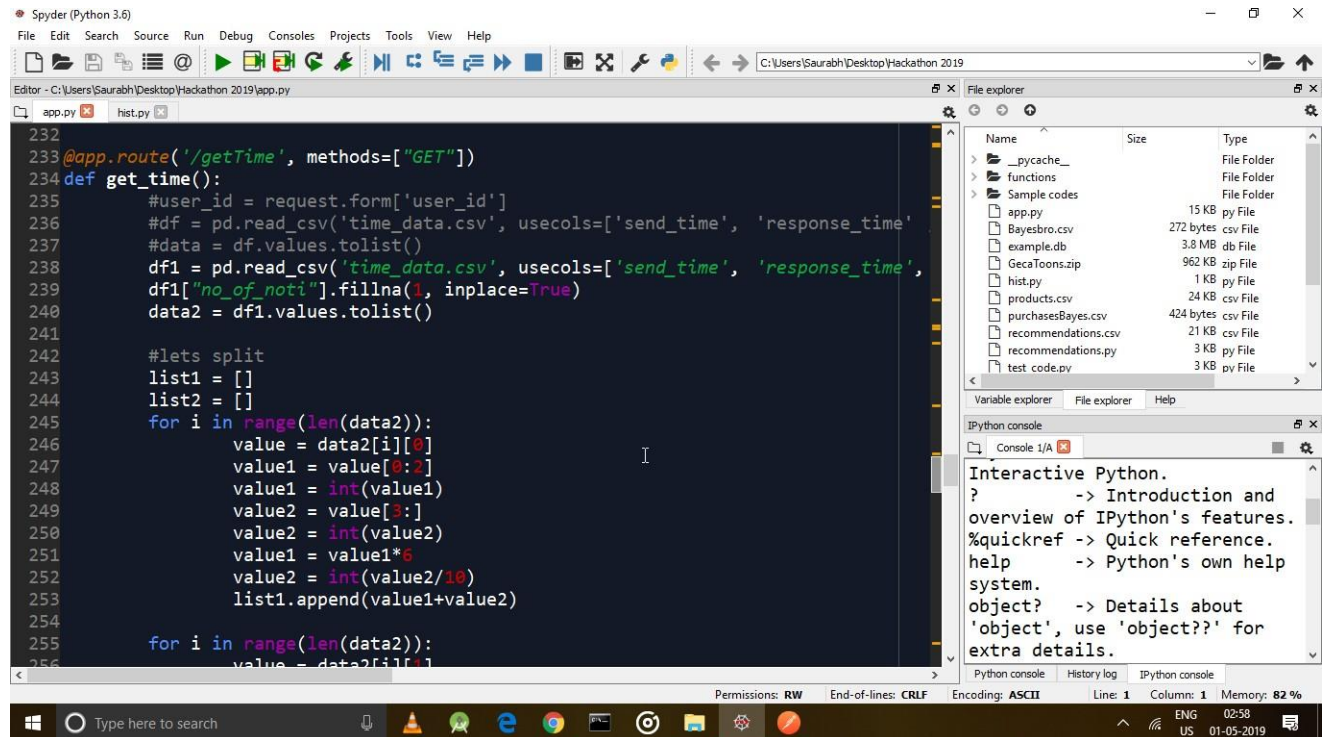
So a module is made to track coordinates of user based on his accelerometer data and when the distance crosses a certain threshold along with the speed, we can record the time slot, and can lay down a marker at them knowing that the user is travelling at that point of time.

Thus the algorithm working on this data of numerous days learns about the user activity and predicts the time of the travel of the user.

Thus these time slots coupled with the user response to the notification gives us a fair idea of the user habits of responding to the notification and his personal best time of receiving notification.

Thus Module 1 outputs the best product the user is interested in and Module 2 predicts the best time of receiving the notification. These two coupled with each other sends the user the best notification at the best time.

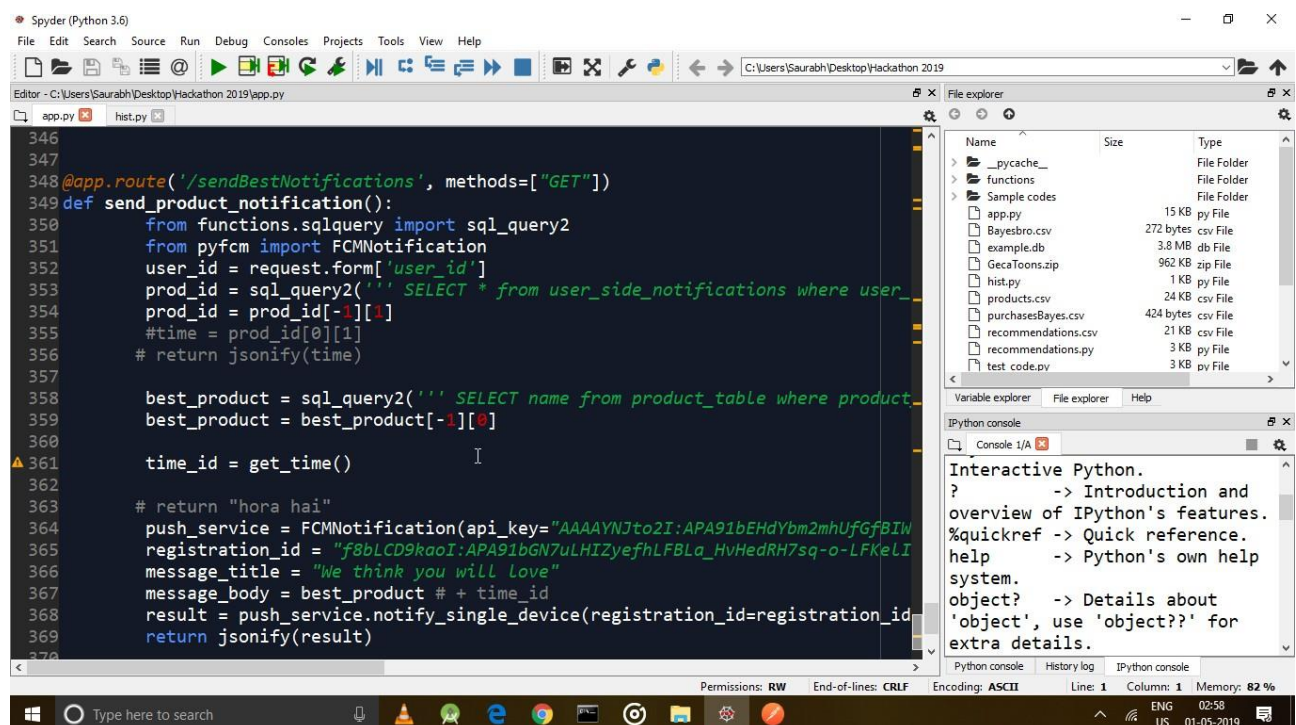
Here is a snippet of it –



```

232
233 @app.route('/getTime', methods=["GET"])
234 def get_time():
235     #user_id = request.form['user_id']
236     #df = pd.read_csv('time_data.csv', usecols=['send_time', 'response_time']
237     #data = df.values.tolist()
238     df1 = pd.read_csv('time_data.csv', usecols=['send_time', 'response_time',
239     df1["no_of_noti"].fillna(1, inplace=True)
240     data2 = df1.values.tolist()
241
242     #lets split
243     list1 = []
244     list2 = []
245     for i in range(len(data2)):
246         value = data2[i][0]
247         value1 = value[0:2]
248         value1 = int(value1)
249         value2 = value[3:]
250         value2 = int(value2)
251         value1 = value1*6
252         value2 = int(value2/10)
253         list1.append(value1+value2)
254
255     for i in range(len(data2)):
256         value = data2[i][1]

```



```

346
347
348 @app.route('/sendBestNotifications', methods=["GET"])
349 def send_product_notification():
350     from functions.sqlquery import sql_query2
351     from pyfcm import FCMNotification
352     user_id = request.form['user_id']
353     prod_id = sql_query2('' SELECT * from user_side_notifications where user_
354     prod_id = prod_id[-1][1]
355     #time = prod_id[0][1]
356     # return jsonify(time)
357
358     best_product = sql_query2('' SELECT name from product_table where product_
359     best_product = best_product[-1][0]
360
361     time_id = get_time()
362
363     # return "hora hai"
364     push_service = FCMNotification(api_key="AAAAYNJto2I:APA91bEHdYbm2mhUfGfBIW
365     registration_id = "f8bLCD9kaoI:APA91bGN7uLHIZyefhLFBLa_HvHedRH7sq-o-LFKelI
366     message_title = "We think you will Love"
367     message_body = best_product # + time_id
368     result = push_service.notify_single_device(registration_id=registration_id
369     return jsonify(result)
370

```

Thus this is the main working of the project.

Lastly the project contains a Module 3 for predicting the probability of purchase of a particular product in future.

### **4.3 Predicting probability of purchase of a product in future.**

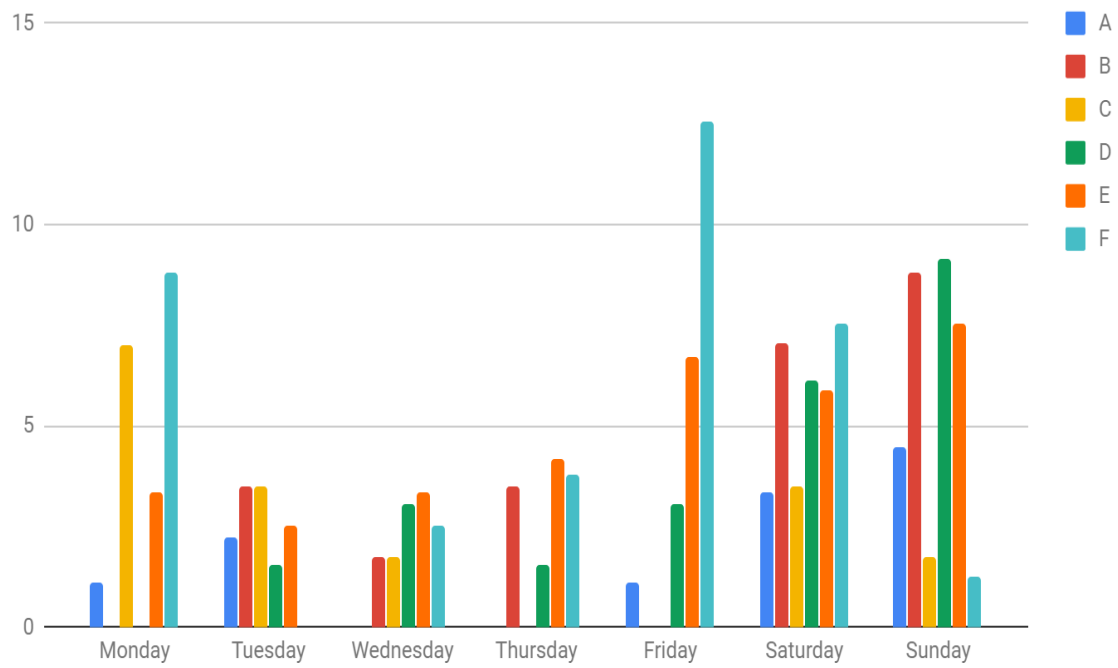
This is a very interesting and a very useful module. Its business applications are surreal. Predicting the probability of purchase of a product for the user ahead in time will be based on so many factors. The past buying pattern of the user, the category of users the particular person belongs to (the same products for the particular group) and the general purchase data of the website about the product at a particular point of time in the day.

There is this dataset showing the overall purchases made by the user up to the present on a particular day and particular time slot and also gives an idea about the total purchases made overall on the website similarly based on the particular day and particular time.

A two layer Naive Bayes Algorithm is employed here, first concerning how the purchases made by the user in a particular point of time is impacted by the total purchases made at that time on the website and second the purchases made the user on a particular day impacted by the total purchases made on that day. This is carefully followed in code and the probability of purchase of the user at a particular point in day on a particular day is obtained.

This figure after applying the above algorithm gives a view of the output of what Module 3 was built for:

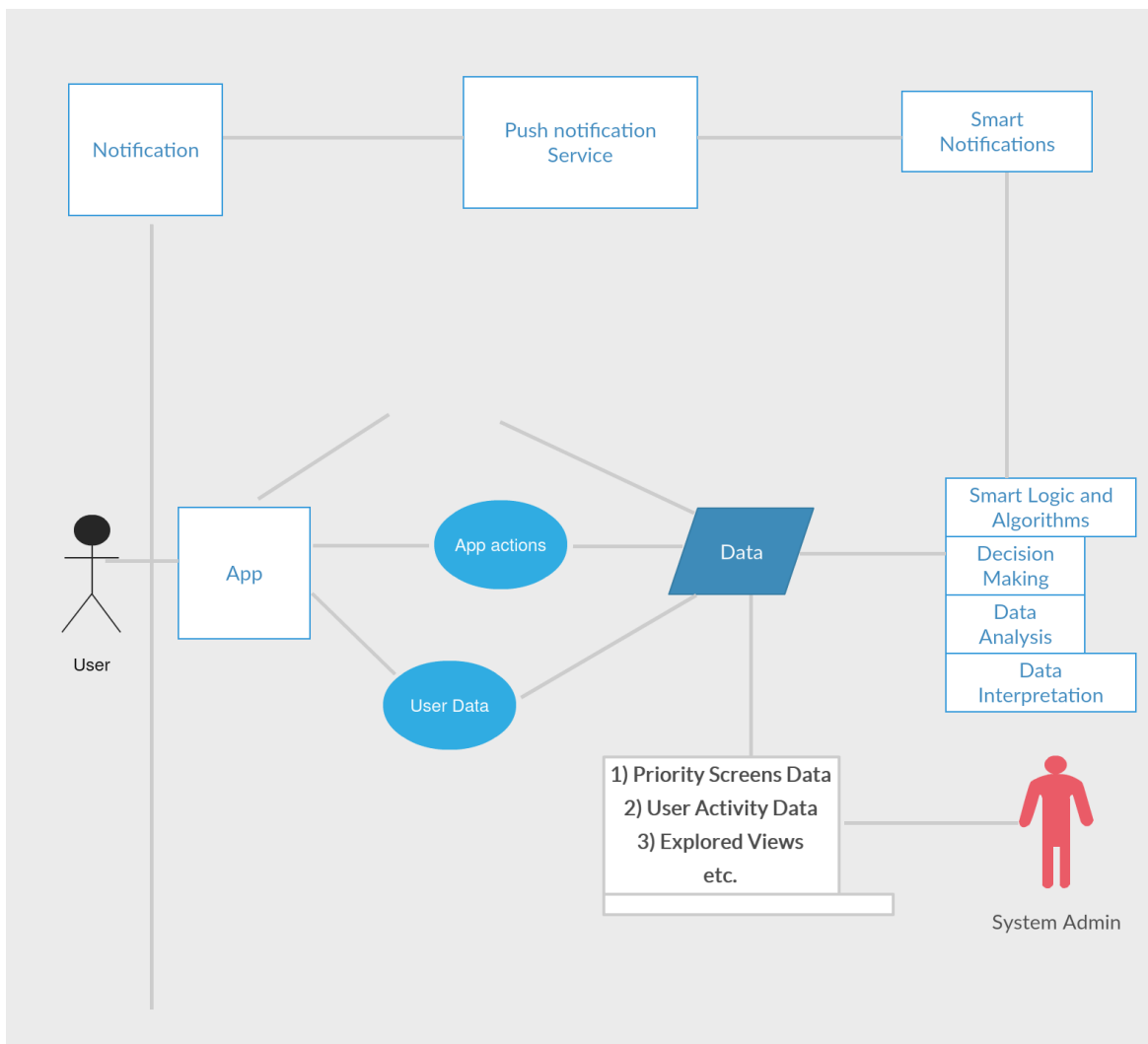
Time and Day Product Buying Probability Distribution



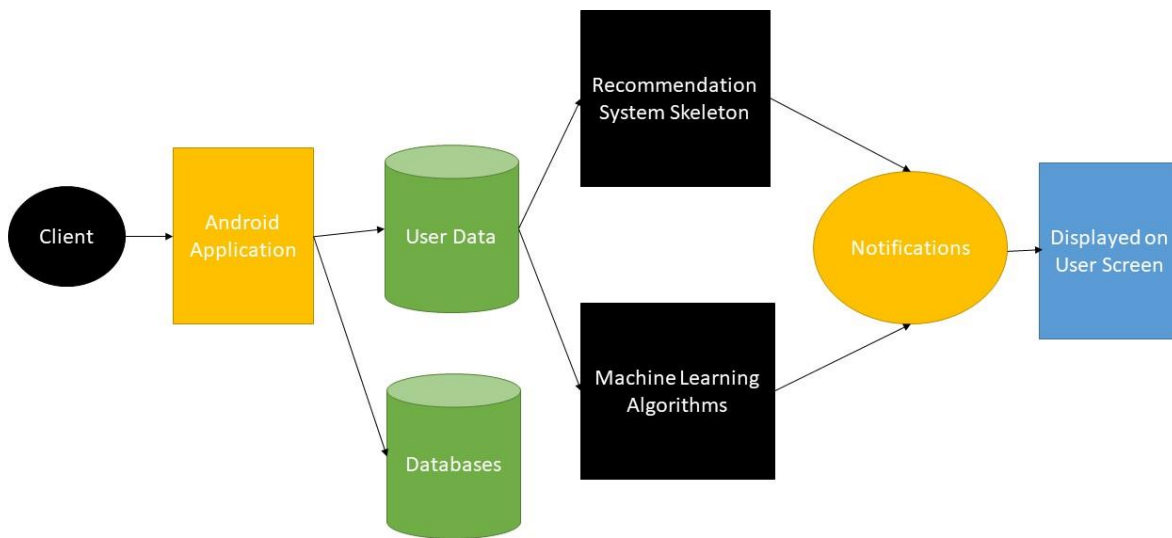
Thus was explained the working of all the modules of the project.

## 4.4 System Diagrams

### 4.4.1 Use Case Diagram –

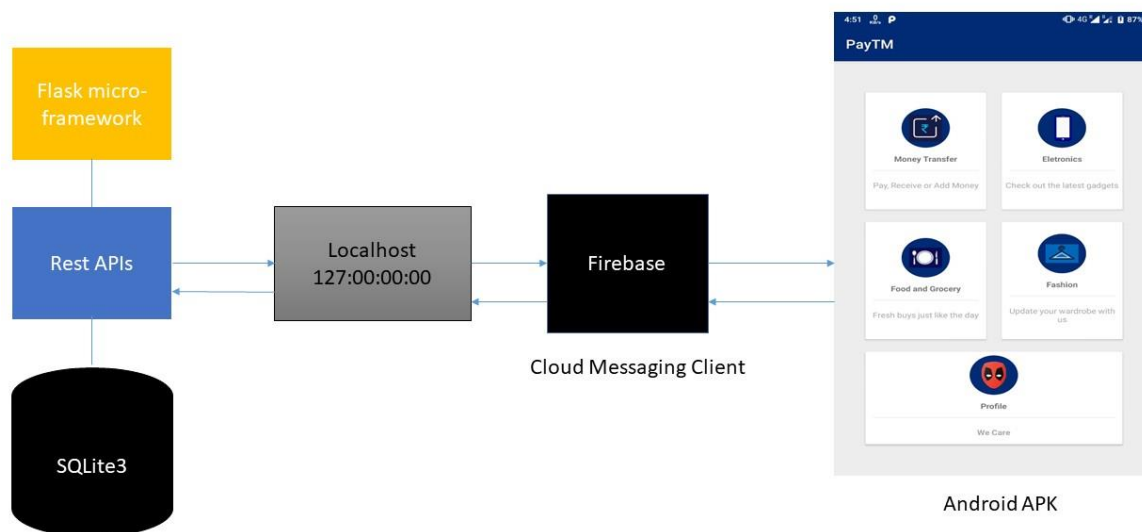


#### 4.4.2 Data Flow Diagram



#### 4.4.3 Tech Stack

### Tech Stack





## 5 Tables

Following are the main tables in the SQLite database.

Searches

Column Name	Data Type
User Id	Integer
Search Element Id	Integer
Time Spent on element	Real

User Notification Response Data

Column Name	Data Type
User Id	Integer
Product Id	Integer
Time of Sending Notification	Integer
Time of Action on Notification	Integer
Action	Integer

Correlation between products

Column Name	Data Type
Product 1 Id	Integer
Product 2 Id	Integer
Correlation factor	Real

## Product Details

Column Name	Data Type
Product Id	Integer
Number of purchases	Integer
Name	Varchar(100)
Actual Price	Integer
Original Price	Integer
Discount Percentage	Integer
Number of offers	Integer
Seller Name	Varchar(100)

Along with this, there are the databases upon which the project is based. The images of it are attached below.

	A	B	C	D	E	F	G	H	I	
1	product_id	parent_id	number_of_purchases	name	actual_price	original_price	discount_percentage	number_of_offers	seller_name	sunder
19	17	7	91	Motorola Pulse Escape Headphones (Black)	2219	6499	66	5	Motorola	
20	18	7	38317	boAt BassHeads 220 Super Extra Bass In-Ear wired Headphones with Mic	599	999	40	7	boAt	
21	19	7	10000	boAt BassHeads 220 Super Extra Bass In-Ear wired Headphones with Mic (Frosty White)	599	999	40	7	boAt	
22	20	7	785	Leaf Sport Wireless Bluetooth Earphones	999	1999	50	6	Leaf	
23	21	7	11784	Skullcandy S2DUDZ-003 In Ear Earphone (Black)	549	799	31	0	SkullCandy	
24	22	7	1430	Leaf Ear Wireless Bluetooth 4.1 Sweatproof Sports Jogger Earphones with Deep Bass and Headset Compatible With Android and iOS Devices- (Carbon Black)	1999	2999	33	6	Leaf	
25	23	8	10000	Boat Stone 1000 Portable Bluetooth Speaker ( Black )	3076	6990	56	8	boAt	
26	24	8	5387	Philips Bt50b Portable Bluetooth Speaker ( Black )	1649	1999	18	8	Philips	
27	25	8	814	HP S6500 Portable Wired (Black)	1315	2499	47	7	HP	
28	26	8	158	JBL Go 2 Bluetooth Speaker (Black)	2399	3299	27	7	JBL	
29	27	14	1146	Apple iPhone X 64 GB Silver	83618	91900	9	2	Apple	
30	28	14	3722	Apple iPhone 7 32 GB Gold	39899	52370	24	2	Apple	
31	29	13	948	Samsung Galaxy J6 2018 32GB Black	10490	11500	9	2	Samsung	
32	30	13	21	Samsung Galaxy S9 128 GB (Midnight Black)	55900	66000	15	2	Samsung	

## Main Products Database

**Travel time sheet**

File Edit View Insert Format Data Tools Add-ons Help

fx

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	num	time	distance	speed						startCo				
2	1	10:00	0	0						<a href="#">28.649767, 77.200275</a>				
3	2	10:15	0	0						<a href="#">28.649690, 77.199978</a>				
4	3	10:30	2500	10						<a href="#">28.649906, 77.200104</a>				
5	4	10:45	2000	8						<a href="#">28.650188, 77.200317</a>				
6	5	11:00	3000	12						<a href="#">28.650654, 77.200616</a>				
7	6	11:15	0	0						<a href="#">28.650778, 77.200702</a>				
8	7	11:30	0	0						<a href="#">28.650829, 77.200746</a>				
9	8	11:45	0	0						<a href="#">28.650872, 77.200781</a>				
10	9	12:00	0	0						<a href="#">28.650940, 77.200831</a>				
11	10	12:15	0	0						<a href="#">28.650990, 77.200869</a>				
12	11	12:30	0	0						<a href="#">28.651053, 77.200913</a>				
13	12	12:45	0	0						<a href="#">28.651084, 77.200934</a>				
14	13	13:00	2000	8						<a href="#">28.651113, 77.200959</a>				
15	14	13:15	1400	15.6						<a href="#">28.651171, 77.200989</a>				
16	15	13:30	0	0						<a href="#">28.651218, 77.201023</a>				
17	16	13:45	0	0						<a href="#">28.651218, 77.201023</a>				
18	17	14:00	0	0						<a href="#">28.651218, 77.201023</a>				
19	18	14:15	0	0						<a href="#">28.651218, 77.201023</a>				
20	19	14:30	0	0						<a href="#">28.651218, 77.201023</a>				
21	20	14:45	0	0						<a href="#">28.651218, 77.201023</a>				
22	21	15:00	0	0						<a href="#">28.651218, 77.201023</a>				
23	22	15:15	0	0						<a href="#">28.651574, 77.201251</a>				
24	23	15:30	0	0						<a href="#">28.651574, 77.201251</a>				
25	24	15:45	0	0						<a href="#">28.651574, 77.201251</a>				
26	25	16:00	0	0						<a href="#">28.652080, 77.201610</a>				

## Travel Time Database

**time\_database**

File Edit View Insert Format Data Tools Add-ons Help

fx

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	day_type	send_time	response_time	action	no_of_noti		Opt in - 1	Dismiss - 0							
2	normal	11:00	11:55	0	5										
3	normal	11:55	12:18	1	2										
4	normal	10:45	11:06	0	4										
5	normal	16:00	17:45	1	4										
6	special	09:00	11:12	1	0										
7	special	16:00	16:09	1	2										
8	normal	16:00	17:55	0	5										
9	normal	18:00	18:08	1	2										
10	normal	19:00	19:04	0	1										
11	special	15:00	17:00	0	3										
12	special	22:00	22:05	0	1										
13	normal	17:00	17:50	1	4										
14	normal	19:00	19:02	1	3										
15	normal	15:30	13:50	1	2										
16	normal	14:40	12:50	1	2										
17	normal	12:50	12:52	1	4										
18	normal	12:30	13:05	1	2										
19	normal	19:40	19:45	0	1										
20	normal	20:00	20:42	1	5										
21	normal	21:00	21:04	1	3										
22	normal	21:30	21:35	0	1										
23	normal	22:30	23:42	1	4										
24	normal	21:50	22:20	1	0										

## Time Database

## 5.1 DATABASE DESIGN

The data required for the project is retrieved from a database. The database system used in development of this project is SQL Server 2008R2. The document in this database presents a complete design description of the system. The model of the system is designed in such a way so as to understand the relations between the entities in the database. The data is the data objects can be integrated and shared among other objects. The table description of each table is given in the database tables. The data objects and constraints and the elements are described in the data dictionary. The process of stepwise reference is used while designing the data. A more elaborate design can be viewed in the data flow diagrams.

The major design objectives of the system are as follows

- The system has been designed to provide easy entry of data
- The data can be entered into the database table through input screen.
- The system reduces workload, inconsistency and redundancy of data.
- Based on the analysis of the system the data objects are created and designed for the purpose of data entry like storing the information.

### 5.1.1 NORMALIZATION

The process of analyzing the data to be represented and breaking it down into separate tables in accordance with the principles of relational structure.

#### 5.1.1.1 NEED FOR NORMALIZATION

Normalization reduces redundancy. Redundancy is the unnecessary repetition of the data. It can cause problems with storage and retrieval of data redundancy can lead to inconsistencies. Errors are more likely to appear when facts are repeated - update anomalies while inserting, deleting, modifying the data may cause inconsistencies. There is a high likelihood of updating or deleting data in one relations, while omitting to make corresponding changes in other relations.

During the process of normalization,

We can identify the dependence which can cause the problems while deleting or updating. Normalization also helps us to simplify the structure of tables full normalized record of a primary key that identifies that entity. A set of attributes that describes the entity.

#### 5.1.1.2 Normal Forms

Normalization results in the formation of tables that satisfy certain specifies constraints and represent certain normal forms. Normal forms are table structures with minimum redundancy.

### **First Normal Form**

A relation R is in first normal form if and only if all underlying domains contain atomic values only.

### **Second Normal Form**

A relation R is in the second normal form if and only if it is in 1st NF and every non-key attributes is fully dependent on the primary key.

### **Third Normal Form**

A relation R is in the third normal form if and only if it is in SNF and every non-key attributes is not transitively dependent on the primary key.

### **Fourth Normal Form**

A relation r is in fourth normal form if and only if whenever there exists multi-valued dependency is R, say  $A \twoheadrightarrow B$ , then attributes on other attribute is a determinant.

### **Boyce-Codd Normal Form**

A relation is in Boyce-Codd Normal form (BCNF) if and only if every determinant is a candidate key. An attribute is fully dependent on other attribute is a determinant.

### **Fifth Normal Form**

A relation is in the fifth normal form also called project-join normal form (PJNF) if and only if every join dependency in R is implied by the candidate keys.

## **5.1.2 DATA DICTIONARY**

There is a need to build some structured place to keep details of contents of data flows, data stores and process. Data dictionary is a Structured Repository of data about data. The data dictionary helps the analyst to simplify the structure for meeting the data requirements of the system.

### **5.1.2.1 Advantages of Data Dictionary**

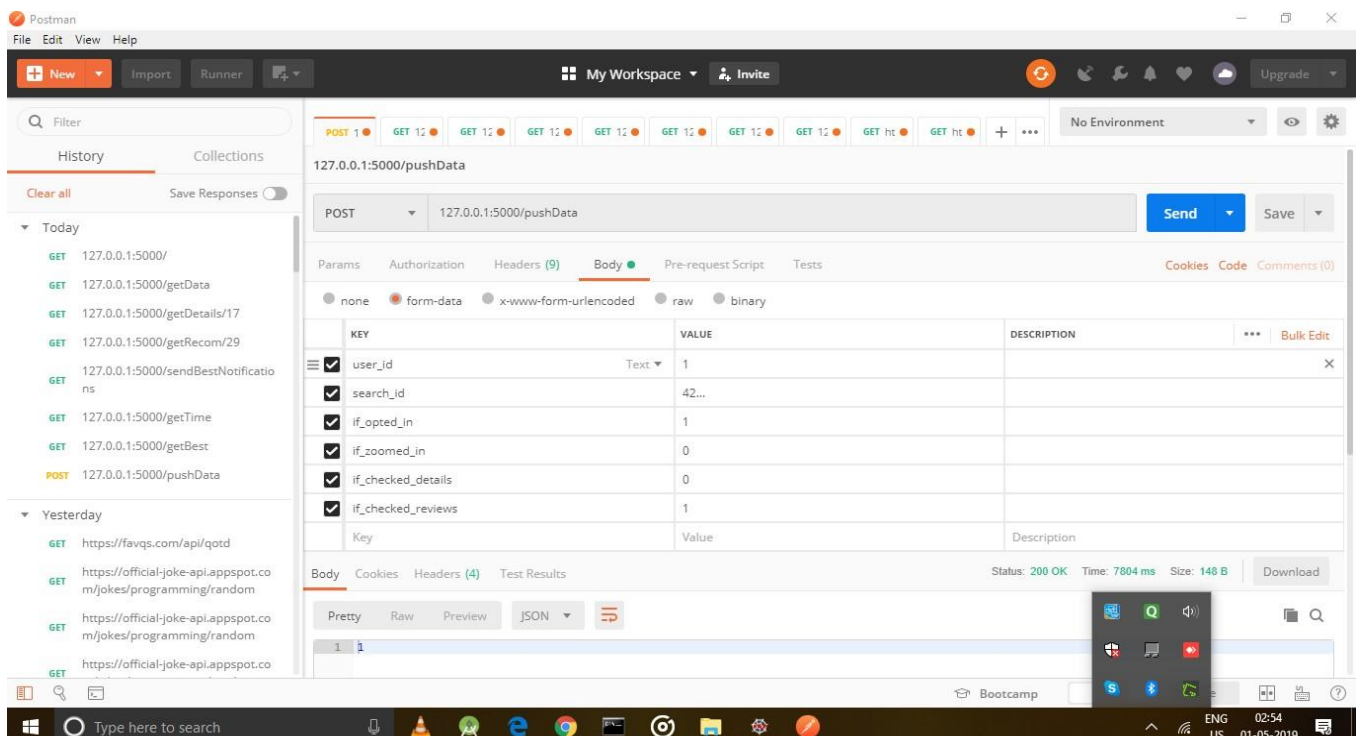
- Valuable Documentation: it is reference in any organization
- It improves analyst/user communication when establishing definitions id, various elements, terms and procedures.
- It server as command base against which programmers who are working on the system can compare their data description.
- Control information is maintained for each data element is cross-referenced in the data dictionary.
- It is important step in building the database.

# 6 OUTPUTS

```
Command Prompt - python app.py
Microsoft Windows [Version 10.0.17134.706]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Saurabh>cd Desktop\Hackathon 2019

C:\Users\Saurabh\Desktop\Hackathon 2019>python app.py
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger pin code: 133-941-182
127.0.0.1 - - [01/May/2019 02:50:31] "POST /pushData HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2019 02:51:23] "GET /getBest HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2019 02:51:23] "GET /getTime HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2019 02:51:26] "GET /sendBestNotifications HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2019 02:51:30] "GET /getRecom/29 HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2019 02:51:30] "GET /getDetails/17 HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2019 02:51:30] "GET /getData HTTP/1.1" 200 -
127.0.0.1 - - [01/May/2019 02:51:32] "GET / HTTP/1.1" 200 -
```



Postman

File Edit View Help

New Import Runner

My Workspace Invite

Filter

History Collections

Clear all Save Responses

Today

- GET 127.0.0.1:5000/
- GET 127.0.0.1:5000/getData
- GET 127.0.0.1:5000/getDetails/17
- GET 127.0.0.1:5000/getRecom/29
- GET 127.0.0.1:5000/sendBestNotifications
- GET 127.0.0.1:5000/getTime
- GET 127.0.0.1:5000/getBest
- POST 127.0.0.1:5000/pushData

Yesterday

- GET https://favqs.com/api/qotd
- GET https://official-joke-api.appspot.com/jokes/programming/random
- GET https://official-joke-api.appspot.com/jokes/programming/random
- GET https://official-joke-api.appspot.com/jokes/programming/random

127.0.0.1:5000/getBest

GET 127.0.0.1:5000/getBest

Send Save

Params Authorization Headers (9) Body Pre-request Script Tests

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 56961 ms Size: 210 B Download

Pretty Raw Preview JSON

```

1 [
2   "Oppo F9 Pro 64 GB Twilight Blue",
3   11.029886070951148
4 ]

```

Bootcamp

Type here to search

ENG US 02:54 01-05-2019

Postman

File Edit View Help

New Import Runner

My Workspace Invite

Filter

History Collections

Clear all Save Responses

Today

- GET 127.0.0.1:5000/
- GET 127.0.0.1:5000/getData
- GET 127.0.0.1:5000/getDetails/17
- GET 127.0.0.1:5000/getRecom/29
- GET 127.0.0.1:5000/sendBestNotifications
- GET 127.0.0.1:5000/getTime
- GET 127.0.0.1:5000/getBest
- POST 127.0.0.1:5000/pushData

Yesterday

- GET https://favqs.com/api/qotd
- GET https://official-joke-api.appspot.com/jokes/programming/random
- GET https://official-joke-api.appspot.com/jokes/programming/random
- GET https://official-joke-api.appspot.com/jokes/programming/random

127.0.0.1:5000/getTime

GET 127.0.0.1:5000/getTime

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 54183 ms Size: 200 B Download

Pretty Raw Preview HTML

```

1 The best time to send notification is : 17:40

```

Bootcamp

Type here to search

ENG US 02:54 01-05-2019



Postman

File Edit View Help

New Import Runner My Workspace Invite

Filter

History Collections

Clear all Save Responses

Today

- GET 127.0.0.1:5000/
- GET 127.0.0.1:5000/getData
- GET 127.0.0.1:5000/getDetails/17
- GET 127.0.0.1:5000/getRecom/29
- GET 127.0.0.1:5000/sendBestNotifications
- GET 127.0.0.1:5000/getTime
- GET 127.0.0.1:5000/getBest
- POST 127.0.0.1:5000/pushData

Yesterday

- GET https://favqs.com/api/qotd
- GET https://official-joke-api.appspot.com/jokes/programming/random
- GET https://official-joke-api.appspot.com/m/jokes/programming/random
- GET https://official-joke-api.appspot.com/m/jokes/programming/random
- GET https://official-joke-api.appspot.com/m/jokes/programming/random

127.0.0.1:5000/sendBestNotifications

GET 127.0.0.1:5000/sendBestNotifications

Send Save

Params Authorization Headers (9) Body Pre-request Script Tests

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 53829 ms Size: 378 B Download

Pretty Raw Preview JSON

```

1 {
2   "canonical_ids": 0,
3   "failure": 0,
4   "multicast_ids": [
5     9063388120191451981
6   ],
7   "results": [
8     {
9       "message_id": "0:1556659286936416%26ffff15185120eb2"
10    }
11  ],
12  "success": 1,
13  "topic_message_id": null
14 }

```

Bootcamp

Type here to search

ENG US 02:54 01-05-2019

Postman

File Edit View Help

New Import Runner My Workspace Invite

Filter

History Collections

Clear all Save Responses

Today

- GET 127.0.0.1:5000/
- GET 127.0.0.1:5000/getData
- GET 127.0.0.1:5000/getDetails/17
- GET 127.0.0.1:5000/getRecom/29
- GET 127.0.0.1:5000/sendBestNotifications
- GET 127.0.0.1:5000/getTime
- GET 127.0.0.1:5000/getBest
- POST 127.0.0.1:5000/pushData

Yesterday

- GET https://favqs.com/api/qotd
- GET https://official-joke-api.appspot.com/m/jokes/programming/random
- GET https://official-joke-api.appspot.com/m/jokes/programming/random
- GET https://official-joke-api.appspot.com/m/jokes/programming/random
- GET https://official-joke-api.appspot.com/m/jokes/programming/random

127.0.0.1:5000/getRecom/29

GET 127.0.0.1:5000/getRecom/29

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 54676 ms Size: 8.08 MB Download

Pretty Raw Preview JSON

```

1 [
2   [
3     29,
4     29,
5     948,
6     "Samsung Galaxy J6 2018 32GB Black",
7     10490,
8     11500,
9     2,
10    "Samsung"
11  ],
12  [
13    29,
14    29,
15    29,
16    29
17  ]
18 ]

```

Bootcamp

Type here to search

ENG US 02:55 01-05-2019

Postman

File Edit View Help

New Import Runner

My Workspace Invite

Filter

History Collections

Clear all Save Responses

Today

- GET 127.0.0.1:5000/
- GET 127.0.0.1:5000/getData
- GET 127.0.0.1:5000/getDetails/17
- GET 127.0.0.1:5000/getRecom/29
- GET 127.0.0.1:5000/sendBestNotifications
- GET 127.0.0.1:5000/getTime
- GET 127.0.0.1:5000/getBest
- POST 127.0.0.1:5000/pushData

Yesterday

- GET https://favqs.com/api/qotd
- GET https://official-joke-api.appspot.com/jokes/programming/random
- GET https://official-joke-api.appspot.com/jokes/programming/random
- GET https://official-joke-api.appspot.com/jokes/programming/random

127.0.0.1:5000/getDetails/17

GET 127.0.0.1:5000/getDetails/17

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

Status: 200 OK Time: 120516 ms Size: 264 B Download

Pretty Raw Preview JSON

```

1 [
2   17,
3   17,
4   91,
5   "Motorola Pulse Escape Headphones (Black)",
6   2219,
7   6499,
8   66,
9   5,
10  "Motorola"
11 ]

```

Bootcamp Build Browse

Type here to search

ENG US 02:56 01-05-2019

https://console.firebase.google.com/project/notify-me-5310b/notification

Apps Home - Quora Django: the Web fr... HTML Entities How to - Mighty Sh... Mini Project in C Fo... c - How to shrink a... generate random c... C > Games and Gra...

Firestore

Notify Me

Go to docs

Cloud Messaging

Notifications Reports

Create experiment New notification

Notification	Status	Platform	Start/Send	End	Sends	Opens
Darshan	✓ Completed		2 May 2019 18:01	—	<1,000	0%
bro	✓ Completed		2 May 2019 17:48	—	<1,000	0%
I guess	✓ Completed		2 May 2019 17:45	—	<1,000	0%
Zinda hu mai	✓ Completed		2 May 2019 17:37	—	<1,000	0%

2 May 2019 17:37

View and analyse detailed FCM message data in BigQuery Link BigQuery

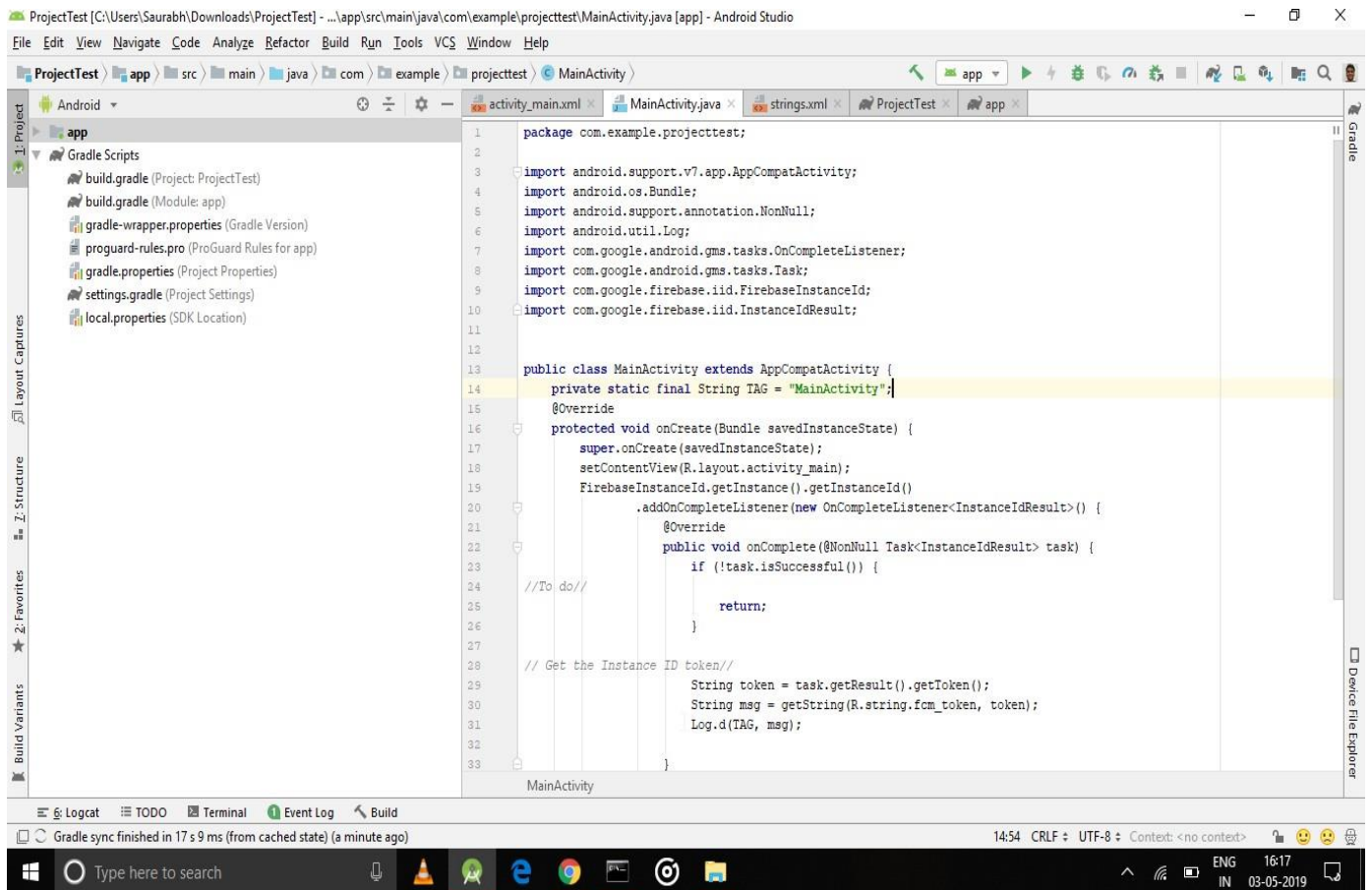
Quality Crashlytics, Performance, Test Lab

Analytics Dashboard, Events, Conversions, A...

Grow

- Predictions
- A/B Testing
- Cloud Messaging
- In-App Messaging
- Remote Config
- Dynamic Links
- AdMob

Spark Free \$0/month Upgrade



04:32

Fri 03 | May 2019

36.6KB/s 4G LTE 3G 39%



Today: 1.19GB This month: 4.71GB

**Project Test** • now ^

### We think you will love

Canon EOS 1300D Kit (EF S18-55 IS II) 18 MP DSLR Camera (Black)

**Android System**

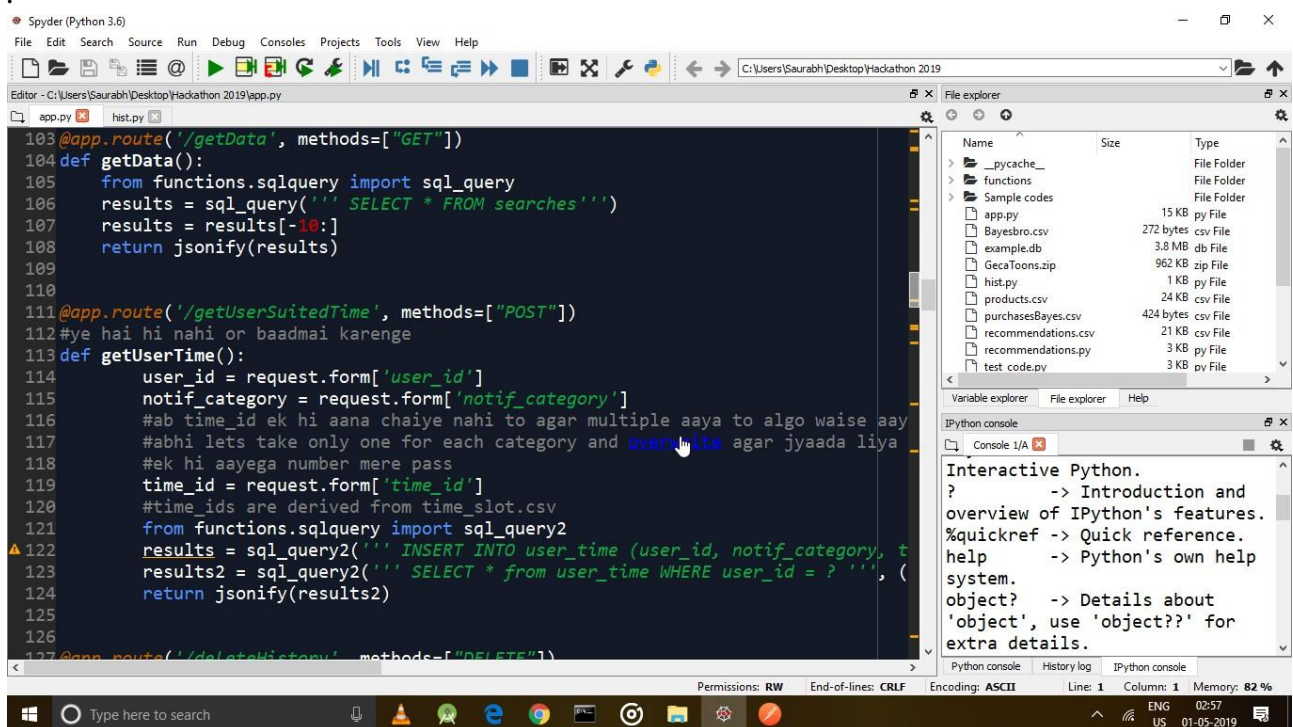
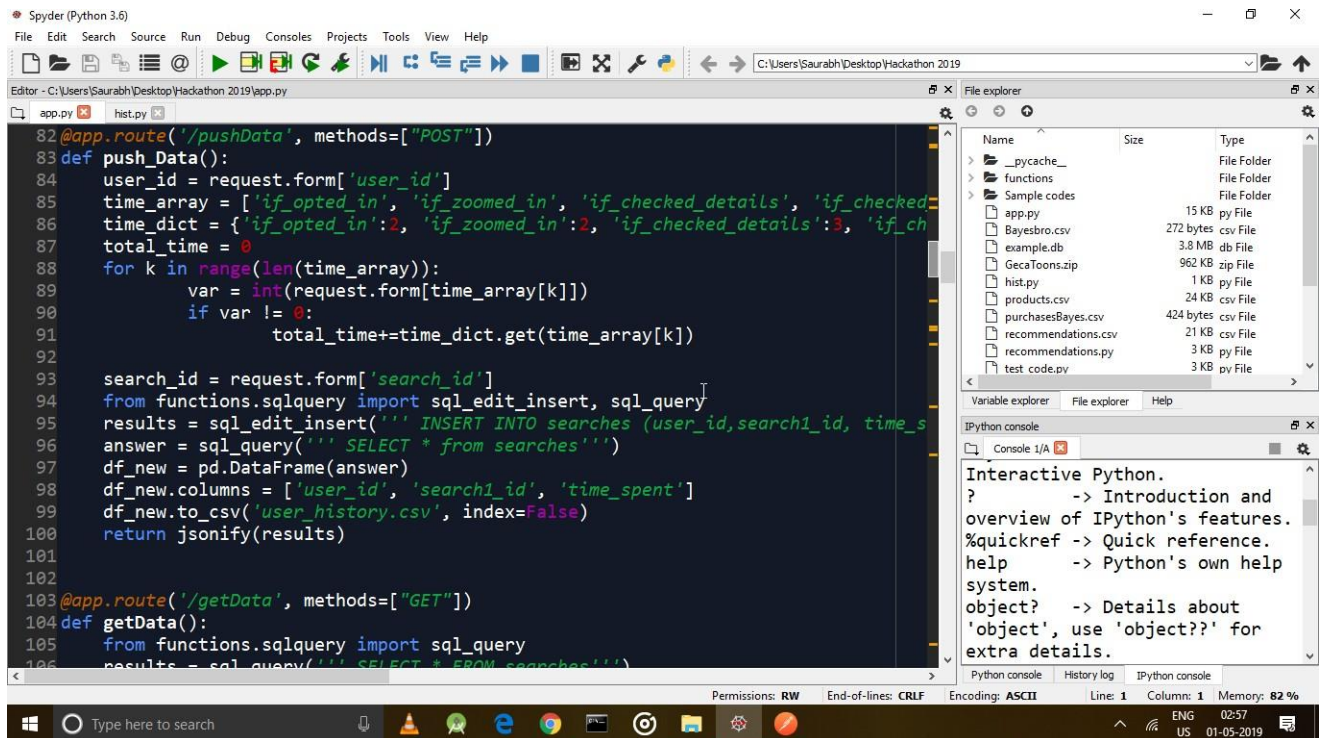
### Tethering or hotspot active

2 connected devices

Turn off







Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - C:\Users\Saurabh\Desktop\Hackathon 2019\recommendations.py

```

1 import pandas as pd
2 df = pd.read_csv('products.csv')
3 product_id = df['product_id']
4 parent_id = df['parent_id']
5 df1 = pd.read_csv('products.csv', usecols=['product_id', 'parent_id'])[:162]
6 data = df1.values.astype(int).tolist()
7 children = []
8 for i in product_id:
9     list1 = []
10    for j in range(len(data)):
11        store = data[j][0]
12        store1 = data[j][1]
13        if i == store1:
14            list1.append(store)
15    children.append(list1)
16
17 df['children'] = children
18 df.to_csv('products.csv')
19 df2 = pd.read_csv('products.csv', usecols=['product_id', 'parent_id', 'children'])
20 df2["parent_id"].fillna("0", inplace=True)
21 data2 = df2.values.tolist()
22 products = []
23 for i in range(len(data2)):
24     data2[i][1] = int(data2[i][1])

```

File explorer

Name	Size	Type
app.py	15 KB	py File
Bayesbro.csv	272 bytes	csv File
example.db	3.8 MB	db File
GecaToons.zip	962 KB	zip File
hist.py	1 KB	py File
products.csv	24 KB	csv File
purchasesBayes.csv	424 bytes	csv File
recommendations.csv	21 KB	csv File
recommendations.py	3 KB	py File
test_code.py	3 KB	py File
test_code1.py	1 KB	py File
time_data.csv	800 bytes	csv File
time_slot.csv	137 bytes	csv File

Variable explorer File explorer Help

IPython console

Console 1/A

Interactive Python.

? -> Introduction and overview of IPython's features.

%quickref -> Quick reference.

help -> Python's own help system.

object? -> Details about 'object', use 'object??' for extra details.

Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 1 Column: 1 Memory: 78 %

Spyder (Python 3.6)

File Edit Search Source Run Debug Consoles Projects Tools View Help

Editor - C:\Users\Saurabh\Desktop\Hackathon 2019\test\_code1.py

```

1 import pandas as pd
2 df = pd.read_csv('travel_time.csv', usecols=['time', 'distance', 'speed'])
3 data = df.values.tolist()
4 df1 = pd.read_csv('travel_time.csv', usecols=['distance', 'speed'])
5 data2 = df1.values.tolist()
6 threshold_distance = 3000
7 speed_threshold = 10
8
9 list1 = []
10
11 for i in range(len(data)):
12     value = data[i][0]
13     value1 = value[0:2]
14     value1 = int(value1)
15     value2 = value[3:]
16     value2 = int(value2)
17     value1 = value1*6
18     value2 = int(value2/10)
19     list1.append(value1+value2)
20
21 list2 = []
22 for i in range(len(data2)-1):
23     value = int(round(float(data2[i][0])))
24     value1 = int(round(float(data2[i+1][0])))

```

File explorer

Name	Size	Type
GecaToons.zip	962 KB	zip File
hist.py	1 KB	py File
products.csv	24 KB	csv File
purchasesBayes.csv	424 bytes	csv File
recommendations.csv	21 KB	csv File
recommendations.py	3 KB	py File
test_code.py	3 KB	py File
test_code1.py	1 KB	py File
time_data.csv	800 bytes	csv File
time_slot.csv	137 bytes	csv File
travel_time.csv	4 KB	csv File
user_history.csv	489 bytes	csv File

Variable explorer File explorer Help

IPython console

Console 1/A

Interactive Python.

? -> Introduction and overview of IPython's features.

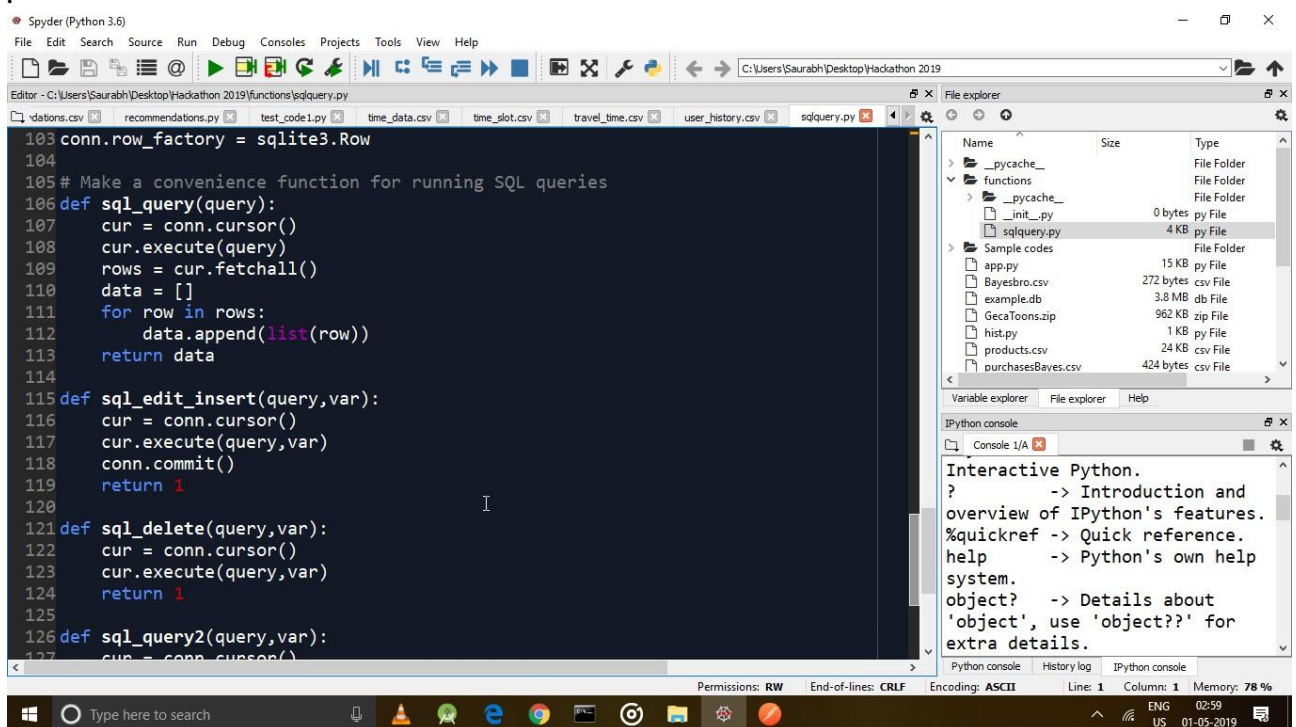
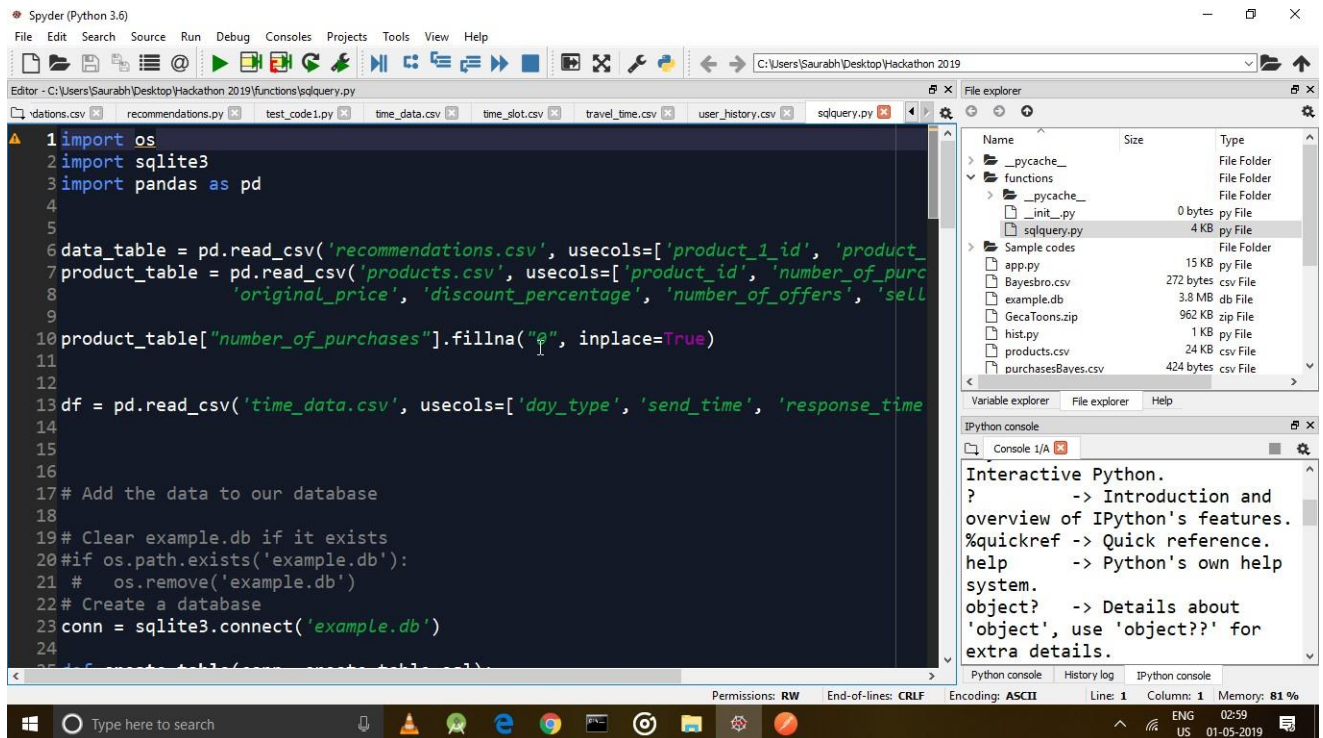
%quickref -> Quick reference.

help -> Python's own help system.

object? -> Details about 'object', use 'object??' for extra details.

Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 1 Column: 1 Memory: 78 %





## 7 Test Cases

We have verified our system using these test scenarios:

- Verify whether the system has been launched successfully or not.
- Verify whether the console for sending the API requests is working or not.
- Verify whether the code access the correct tables or not.
- Verify whether data gets updated into right columns in the specific tables.
- Verify whether the correct searches are used by the Algorithm.
- Verify whether the correct number of searches are used by the Algorithm.
- Check the connection between Flask and Firebase.
- Check the connection between SQLite and Flask.
- Check whether the data is entered in the right form in the console.
- Check whether the method (GET/PUT/POST) while sending the request is correct or not.
- Check the device token is successfully generated or not.
- Check whether the device token is correct or not.
- Check the project details on the Firebase console,
- Check whether the App correctly receives the notifications or not.
- Verify if the system is stable while the server is on for long hours.
- Verify if the android application behaves as intended.
- **Verify tap on the screen ten times at different positions, the application should work normally and not freeze.**
- **Verify when the device is suddenly powered off, the application does not corrupt data. Are the last saved data saved properly?**
- Check whether the output displayed by Postman is in the right form or not.
- Verify whether all the compulsory form data is entered in the Postman request.
- Verify whether the base and standard queries of the database work correctly and return a valid response.
- Verify a proper localhost connection is created when app.py is launched.
- Check the localhost connection.
- Check for any errors in the Urls of the code which represent the Apis of the system.
- Check whether all the necessary databases are present in the project structure.
- Check if all the necessary modules and libraries are imported for Python.



## 8 PROPOSED COSTING

For developing and deploying the proposed system, we have calculated the estimated cost by calculating the number of days required to work on each activity and also accounting for other additional costs of storage, hosting costs, etc.

ACTIVITIES	DAYS
Requirement Gathering ,Analysis and Documentation	8
System Development	20
Database Design	3
Study of Machine Learning models and libraries	5
Android Application Development	5
Hosting on Heroku	3
Upgrading SQLite to PostgreSQL	2
Usage Analytics	1
Crash Reporting	1
Testing	4

Total no. of days for designing: 5

Total no. of days for Developing: 28

Total no. of days for Other Activities: 19

Designer's rate per day: Rs. 600

Developer's rate per day: Rs. 650

Other's rate per day: Rs. 450

Total Development Cost:  $28*650+5*600+19*450 = 29750$

Firebase Cost per year=  $1500*12 = 18000$

Total cost estimate =47,750.

## 9 CONCLUSION

Smart Notification System to reduce Push Notifications using Machine Learning is developed using Flask framework, Python, Firebase and Android Studio. The developed project fully meets the objectives of the system for which it has been developed. The system has reached a steady state where all bugs have been eliminated. The main problem of reducing Push notifications has been solved and also the system can classify user intent and deliver best notifications at the right time. Also the system keeps evolving based on user's response and the algorithms and the modules involved work seamlessly and accurately. The output i.e. the actual Push notification generated has also been recorded and tested on a dummy Android application. The system was intended to solve problems specified in the requirement specification and the project has achieved all the specifications.

## 10 BIBLIOGRAPHY & REFERENCE

- <https://hackernoon.com/introduction-to-firebase>
- <https://en.wikipedia.org/wiki/Firebase>
- <https://en.wikipedia.org/wiki/Firebase>
- <https://github.com/pallets/flask>
- <https://realpython.com/tutorials/flask/>
- <https://searchmicroservices.techtarget.com/definition/RESTful-API>
- [https://en.wikipedia.org/wiki/Representational\\_state\\_transfer](https://en.wikipedia.org/wiki/Representational_state_transfer)
- [www.youtube.com](http://www.youtube.com)
- [www.wikipedia.org](http://www.wikipedia.org)
- [www.github.net](http://www.github.net)
- [www.firebase.com](http://www.firebase.com)
- [www.google.com](http://www.google.com)
- [www.android.com](http://www.android.com)