

**K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

**University of Mumbai**

**AMIGOS - Event Recommendation System**

Submitted at the end of semester VIII in partial fulfilment of  
requirements

For the degree of

**Bachelors in Technology**

By

**Vikram Bhavsar**

**Roll No: 1821003**

**Mohammed Anzal Shaikh**

**Roll No: 1821008**

**Vishvesh Pandey**

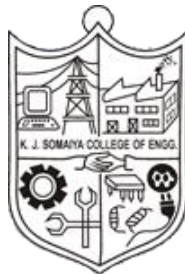
**Roll No: 1821009**

**Samiksha Gupta**

**Roll No: 1821010**

Project Guide

**Prof. Rohini Nair**



**Department of Computer Engineering**

**K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

**Batch 2017 -2021**

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

## **Certificate**

This is to certify that the dissertation report entitled **AMIGO - Event Recommendation System** submitted by **Vikram Bhavsar, Vishvesh Pandey, Samiksha Gupta and Mohammed Anzal Shaikh** at the end of semester VIII of LY B. Tech is a bona fide record for partial fulfillment of requirements for the degree of Bachelors in Technology in Computer Engineering of University of Mumbai

*Rohini Nair*

---

**Guide**

---

**Head of the Department**

---

**Principal**

**Date: 20th May, 2021**

**Place: Mumbai-77**

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

## **Certificate of Approval of Examiners**

We certify that this dissertation report entitled **AMIGOS - Event Recommendation System** is a bona fide record of project work done by **Vikram Bhavsar, Vishvesh Pandey, Samiksha Gupta and Mohammed Anzal Shaikh** during semester VII. This project work is submitted at the end of semester VII in partial fulfilment of requirements for the degree of Bachelors in Technology in Computer Engineering of University of Mumbai.

*Rohini Nair*

---

**Internal Examiners**

*S. Ashwin*

---

**External/Internal Examiners**

**Date: 20th May, 2021**

**Place: Mumbai-77**

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

## **DECLARATION**

We declare that this written report submission represents the work done based on our and / or others' ideas with adequately cited and referenced the original source. We also declare that we have adhered to all principles of intellectual property, academic honesty and integrity as we have not misinterpreted or fabricated or falsified any idea/data/fact/source/original work/ matter in my submission. We understand that any violation of the above will be cause for disciplinary action by the college and may evoke the penal action from the sources which have not been properly cited or from whom proper permission is not sought.

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)



**Signature of the Student**

**Roll No.**

1821003



**Signature of the Student**

**Roll No.**

1821008



**Signature of the Student**

**Roll No.**

1821009



**Signature of the Student**

**Roll No.**

1821010

**Date:** 20th May,2021

**Place:** Mumbai-77

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

## **Table of Contents**

1.1 Problem definition	1
1.2 Scope	1
1.3 Hardware and software requirements	1
1.3.1 Software-	1
1.3.2 Hardware Requirements-(Minimum)	1
2.1 BookMyShow website/app-	2
2.2 MagicPin app-	2
2.3 NearBuy App-	3
2.4 Papers referred-	3
3.1 Proposed System model	4
3.1.1 Event Handler Side	4
Figure (3.1.1) : Keywords segregation	4
3.1.2 User's side	5
3.1.3 User's Web History Data Fetching	6
3.2 Software Project Management Plan	7
3.2.1-Introduction	7
<b>I. Project overview</b>	7
<b>II. Project Deliverables</b>	8
<b>III. Schedule</b>	8
3.2.2-Project organization	9
3.2.3-Managerial process	10
3.2.4-Technical process	10
3.3 Software Requirement Specification Document	11
3.3.1. Introduction	11
3.3.2. Overall Description	11
3.3.3. System Features and Requirements	12
3.4 Software Design Document ( All applicable UML diagrams)	13
4.1 Structure of the project	15
4.1.1 File Structure	15
4.1.2 How to run the project	15
4.1.3 General Walkthrough of the project and basic functionality.	17

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

12) There's a button on the left hand side which says Get Local Events. When clicked the browser shares the location with the server.	23
4.2 Recommendations based on trending events	27
4.2.1 Architecture and logic	27
4.2.2 Database Structure	28
Figure (4.2.2) : Database structure	28
4.2.3 Code	28
4.2.4 Output	30
4.3 Extracting keywords from event's description using TF*IDF Algorithm	31
4.3.1 Architecture and logic	31
4.3.2 Database Schema	31
Figure (4.3.2) : Database Schema	31
4.3.3 Code	32
4.3.4 Output	34
4.4 Calculating Similarity of events	35
4.4.1 Architecture and logic	35
Figure (4.4.1) : Calculating Similarity of Events	35
4.4.2 Database Schema	35
4.4.3 Code	36
4.4.4 Output	38
4.5 Recommendation of events based on past Intra-website search	39
4.5.1 Architecture and logic	39
Figure (4.5.1.a) : Architecture of Recommendation Based on Past Intra-Website Search	39
4.5.2 Database Schema	40
4.5.3 Code	41
4.5.4 Output	44
4.6 Recommendation of events based on intra-website browsing patterns	45
4.6.1 Architecture and logic	45
Figure (4.6.1.a) : Architecture of Recommendation based on Past Browsing History	45
Figure (4.6.1.b) : Types of Event	46
4.6.2 Database Schema	48
4.6.3 Code	49

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

4.6.4 Output	54
4.7 Extracting User's Web browsing history	56
4.7.1 Architecture and logic	56
Figure (4.7.1) : Architecture of Extracting User's Web History	57
4.7.2 Code	58
4.8 Calculating similarity between User's web browsing history and events and recommendations	61
4.8.1 Architecture and logic	62
Figure (4.8.1) : Architecture for Calculating similarity between user's web history and events	62
4.8.2 Database Schema	64
Figure (4.8.2) : Database Schema	64
4.8.3 Code	64
4.8.4 Output	67
4.9 Recommending local events only.	68
4.9.1 Architecture and logic	68
Figure (4.9.1) : Architecture for recommending local events	69
4.9.2 Code	69
4.9.3 output	72
4.10 Testing	74
4.10.1 Black Box Testing	74
4.10.2 White Box Testing	85
5.1 Conclusion	93
5.2 Scope for further work	93
5.2.1 Friends Concept	93
5.2.2 Intra-Somaiya Network	94
5.2.3 Large scale Implementation	94
5.2.4 Adding events live to the website	94
5.3 References	94
ACKNOWLEDGEMENT	96
PLAGIARISM REPORT	97



## **Chapter 1 : Introduction**

### **1.1 Problem definition**

To create a Personalized Event Recommendation System that recommends the events that are sorted according to the user based on his/her interests and also the events that are the most trending. This system would intend to save time for the user and also help the event organizers to boost their events and increase their reach.

### **1.2 Scope**

- To suggest events to the users with as little interaction with the system as possible.
- To recommend events to the user which are highly personalized and based on the user's Interest
- To ensure only interesting events are recommended to the user.
- Also to recommend new trending events to the user that are out of the user's interest.
- Capturing implicit user behavior within the website and using that data to provide better recommendations

### **1.3 Hardware and software requirements**

#### **1.3.1 Software-**

Python, Django, Google Chrome.

#### **1.3.2 Hardware Requirements-(Minimum)**

4GB-RAM.

I3-8th generation-Processor.

512 GB -Hard disk.

256GB -SSD.

## **Chapter 2 : Literature Survey**

In research, we have found 3 different applications which are used for recommending events.

### **2.1 BookMyShow website/app-**

- It is majorly used for booking movie tickets and passes to various events. It uses - the famous Amazon model that is based on a method called Collaborative Filtering.
- It takes items such as movies, books, and products that were rated highly by a set of users and recommends them to other users who also gave them high ratings. This method works well in domains where explicit ratings or implicit user actions can be gathered and analyzed.
- In our system, we focus on the implicit data such as what events are registered through the system, and use that to understand the interest of the user.

### **2.2 MagicPin app-**

- MagicPin is more focused on inviting friends and getting discounts to visit different places based on the user's bills and locations visited by the user. Following is directly quoted from the app description of MagicPin on the play store-
- “Use the magic pin feed to find out what places & brands are trending among your friends & peers. Follow & interact with them by liking, commenting, & bookmarking their recommendations!”.
- Hence, MagicPin is heavily based on recommending events and places based on what other people and friends are visiting. On the other hand, our system makes suggestions based on what’s trending as well as the ones that the user is interested in.

### **2.3 NearBuy App-**

- The NearBuy app is similar to MagicPin in a way that It suggests various Shops and restaurants nearby based on the user's preference.
- There is a tab inside the app which provides suggestions based on what the user searches.
- But, these searches are from within the application whereas, in our case, the important information is taken from web history.

### **2.4 Papers referred-**

- Users may often search for cars, so there must exist a spread of the keyword car based on time. From this, we can assume that since the user is searching for cars often then the user is interested in cars and we can recommend events related to cars to that user.
- This part of the module is heavily based on the paper **Mining User Interests from Web History by Saurabh Kumar**. The paper contains more details on the implementation of the above-mentioned.

## Chapter 3 : Project Design

### 3.1 Proposed System model

#### 3.1.1 Event Handler Side

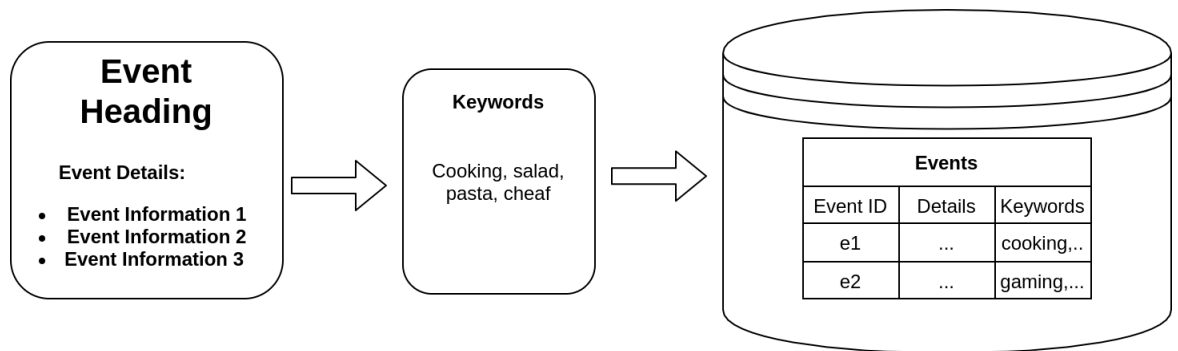


Figure (3.1.1) : Keywords segregation

The above-proposed system depicts the Events Handler part where events details are specified as given above which displays heading, details, and event information. By using NLP's TF-IDF algorithm the important keywords are fetched or the data is cleaned and is stored in the database with the given above format. It will Extract information such as Nouns, and Named Entity using NLP.

### 3.1.2 User's side

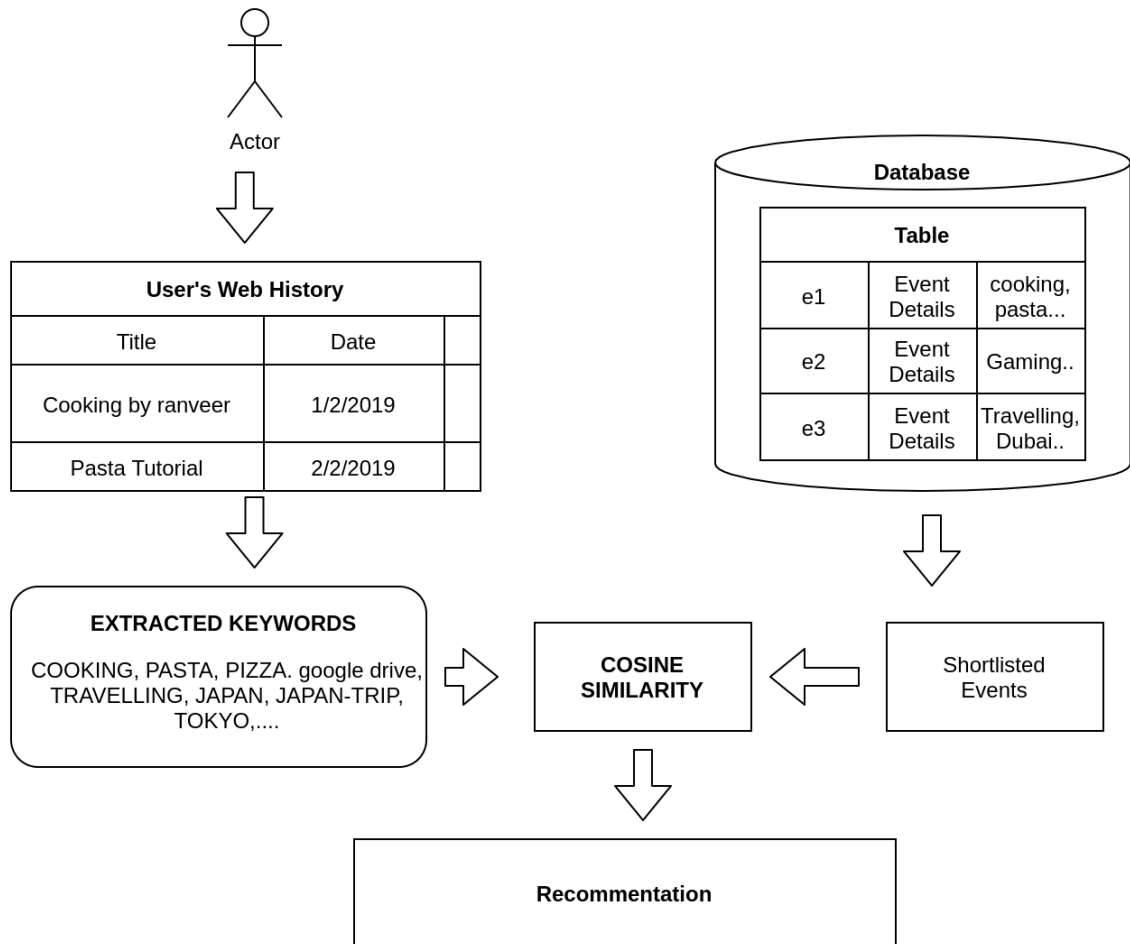
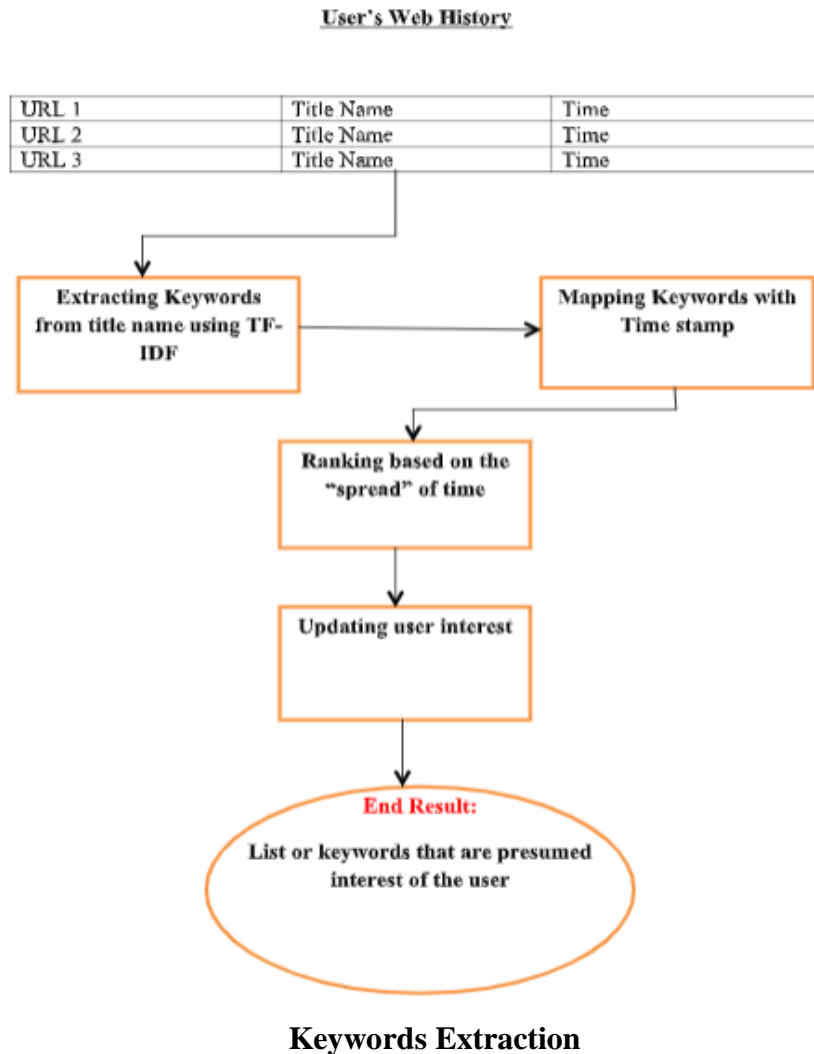


Figure (3.1.2) : Recommendation based on user's web history

At the user's side, it will fetch the user's web history automatically using a web browser extension, on and based on the title and date from the history generated format it will apply the TF-IDF algorithm for extracting nouns, named entities, etc. which are our basic keywords required for our project. After fetching the important keywords from the user as well as from the event handler side our system will check using cosine similarity algorithm for identifying the same keywords and frequency of those words that occurred and similarly, it will recommend to the user based on his/her interest respectively as shown in the above figure.

### 3.1.3 User's Web History Data Fetching



**Figure (3.1.3) : User's web history data fetching and keyword extraction**

The above-proposed model depicts the basic workflow of how the keywords will be extracted from the user's web history using an algorithm.

## **3.2 Software Project Management Plan**

### **3.2.1-Introduction**

We aim to create a Personalized Event Recommendation System that recommends the events that are sorted according to the user based on his/her interests and also the events that are the most trending.

This system would intend to save time for the user and also helps the event organizers to boost their events and increase their reach and ensure that the user does not get recommended any events that he/she is not going to attend but, only the events according to his/her domain-interest along with a list of Trending Events. The system that we plan on developing is a hybrid recommendation system. This system will have two participants-Event Organizers and Users.

For Example: If a user on youtube is searching for Cake Baking videos it would be noted in the history. The system would then extract the title of that video and time and prepare a user profile based on similar searches. In the future when a new event is registered and that happens to be a workshop on cake baking then this event would be recommended to the user since that's one of his/her interests.

### **I. Project overview**

Many times, we receive a large number of notifications about various events, exhibitions, and meetups happening all around us that are irrelevant to us because we simply are not interested in them. Various people have their importance of things that they are interested in and be notified of all these events and from them searching for something that might interest them will take a lot of time and sometimes does not provide any meaningful information.

In today's world, there is no such existing facility that notifies us about the various events that are tailored to our interest. Many popular Recommendation systems show

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

their users various events and exhibitions based on what's trending and what their friends are likely interested in. Although various recommender systems such as google news provide users personalized news that they are interested in. There simply exists no system that focuses on the user's preference and gives personalized suggestions for various events.

## **II. Project Deliverables**

- 1: Preliminary Project Plan
- 2: Requirement Specification
- 3: Timeline
- 4: Architecture Specification
- 5: Source Code
- 6: Test Plan
- 7: Final Product Submission

## **III. Schedule**

**September 15** - System Planning, System Flowchart, Approval Phase

**September 30** - Technology Learning - Django, ER Diagram, Designing System Architecture

**October 15** - Detailed Study on recommendation Systems, Database Design

**October 30** - Technology Learning NLP

Gathering User's Data

**November 15** - Implementation of Dummy user profiling and suggestion of trending events, Implementation of content Recommendation System.

**January 30** - Detailed Study of collaborative Recommendation System and implementation

**February 15** - UI Designing and UI implementation

**February 30** - Final Implementation of knowledge-based being event recommendation System

**March 15** - UAT testing, Alpha testing

**March 30** - Change implementation



# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

**April 15** - UAT testing

**April 30** - Final Changes and Production deployment

## **3.2.2-Project organization**

### **I. Process model**

Our project is going to use the Agile process model because it is iterative and incremental. Our requirements and risks are not concrete and are changing to a small extent as we go on developing the project and hence agile process model is best for that matter.

### **II. Organizational structure**

#### **Team Members –**

1821008 - Mohammed Anzal Shaikh

1821009 - Vishvesh Pandey

1821010 - Samiksha Gupta

1821003 - Vikram Bhavsar

### **III. Project responsibilities**

- Project Plan – Entire Team
- Analysis - Entire Team
- Diagrams and figures - Vishvesh Pandey
- Backend Intra-website search recommendation - Anzal Shaikh and Vikram Bhavsar
- Backend intra-website browsing pattern recommendation - Vishvesh Pandey and Samiksha Gupta
- Chrome web browser extension - Anzal Shaikh and Samiksha Gupta
- Local events and user web history recommendation - Vikram Bhavsar
- Front End UI - Vishvesh Pandey and Samiksha Gupta
- Populating user history data - Entire Team

## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

- Populating Event data - Entire Team
- Testing - Entire Team
- Documentation - Entire Team

### **3.2.3-Managerial process**

#### **I. Management objectives and priorities**

The management objective is to deliver the product in time and of high quality. The PM and QA work together to achieve this by respectively checking that progress is made as planned and monitoring the quality of the product at various stages.

#### **II. Potential Risks**

Potential Risks are mentioned below

##### **i. Technology risk**

Risk of unavailable libraries for other languages other than python such as javascript

##### **ii. Structure/process risk**

Since the project is covered with research as well as development, there's a risk of a lot of time to be taken by the project for implementation.

### **3.2.4-Technical process**

#### **I. Methods, tools, and techniques**

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

The project will be implemented using agile methodology and tools such as Github, Python, Javascript, HTML, CSS, Django, Postgresql, Visual Studio Code Jupyter Notebook.

## **3.3 Software Requirement Specification Document**

### **3.3.1. Introduction**

#### **I. Intended Audience**

This project is intended for the audience who want to explore the events of their interest all at one place.

#### **II. Intended Use**

Event organizers can use this project to post various events and increase their reach, whereas the users can get access to all the events of their interest which will save the user's time.

### **3.3.2. Overall Description**

#### **I. User Needs**

This system will have two participants:

##### **Event Organizers**

We plan on making a website where both event organizers and users have to participate. If an organization's person or any party wants to organize an event then they have to register this event through a portal. The organizer would put in various information about the events such as what the event is about, what is the date, venue, and other information which they seem to fit.

Example of a typical event: If a Treasure hunt event is going to be organized by some organization then they'll put information about this event on our web portal.

# **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

## **Users**

We aim to suggest events to the users with as little interaction with the system as possible. Each user can log into the website and see the top suggested events for them. These suggested events will be the result of multiple algorithms. We plan on taking implicit data from the user and suggest them based on their likings. We plan on limiting the interaction between the user and the system as much as possible when it comes to learning about interest from the user and hence data from the user would be taken implicitly.

## **II. Assumptions and Dependencies**

- This project assumes that it will be able to fulfill and suggest the events of the user's choice based on general user profiling.
- Also, it will be dependent on the user's browsing history such as keywords, titles, etc. for user profiling.
- This system also assumes that a user may be using a Web browser to browse the web and various things and events of his interest
- The Event data should be submitted by Event organizers to the website. The event data is not web scrapped from some third-party website or database.
- History Extractor extension is pre-installed when browsing the website.

## **3.3.3. System Features and Requirements**

### **I. Functional Requirements**

A fully functional laptop/PC with an internet connection and user's prior web history.

### **II. Non-functional Requirements**

**Performance** - The process of getting a user's history and extracting important keywords and ranking them. Sending them to the server and matching the keywords with the keywords from the events should be fast and fluent. Should not take more time and make the user wait.

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

**Safety** - Since the browser extension takes user history data user browser history data must be kept safe from various sniffing attacks to keep user privacy.

**Security** - Proper Security mechanism to ensure user data is not leaked.

**Quality** - Recommending only those events which can be of interest to users and saving time of the user by limiting irrelevant interaction between user and the proposed system.

### 3.4 Software Design Document ( All applicable UML diagrams)

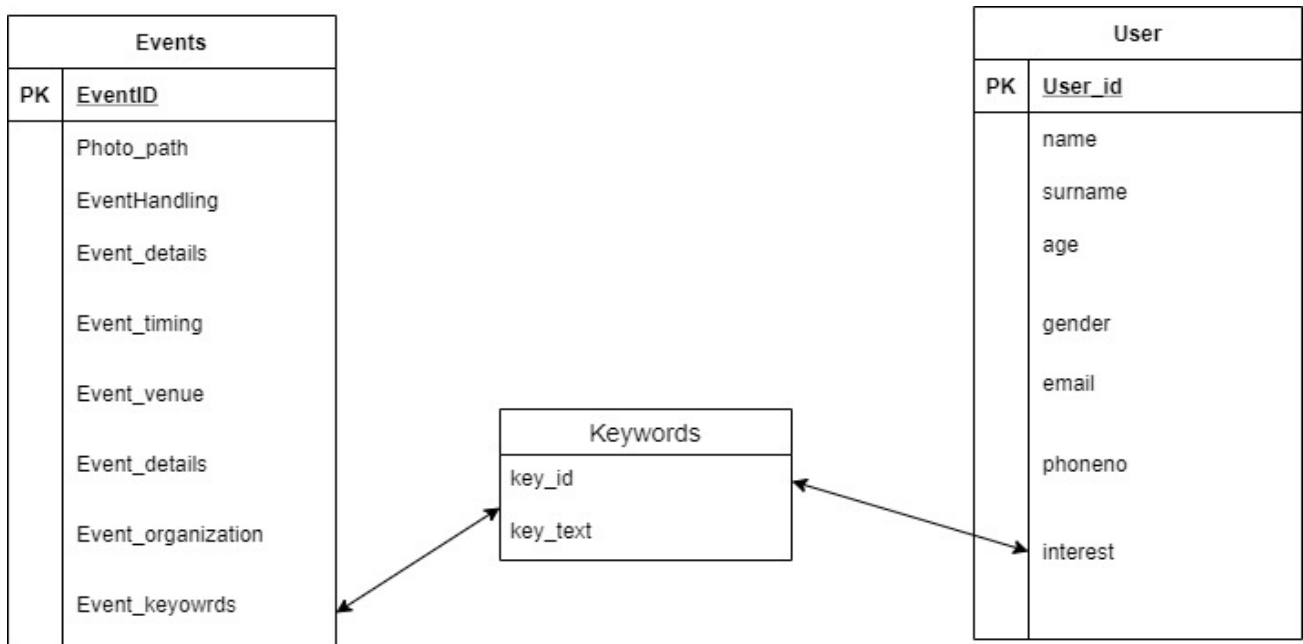


Figure (3.4.1) : User's and Events data mapping based on keywords

Above is the schema of the proposed model. Here the Event and User table has many to many relationships with the keyword table.

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

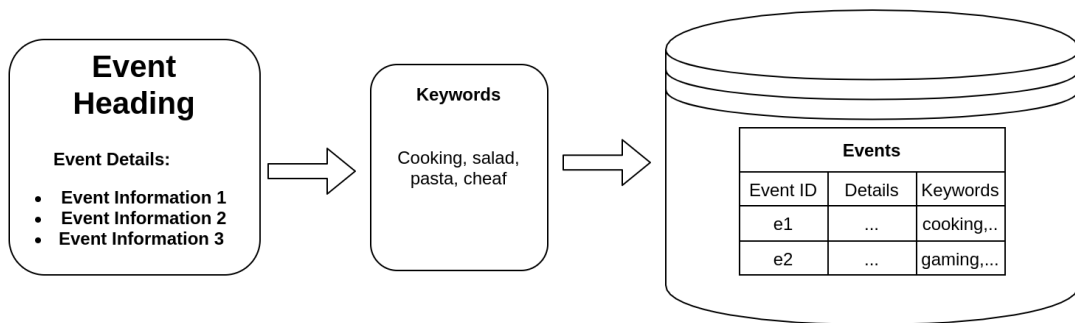


Figure (3.4.2) : Keyword extraction from events and storing it into the database

## Implementation at Event Organizers side

Event Details is a useful source for extracting keywords. When these event data are stored in the database, keywords related to those events would also be stored along.

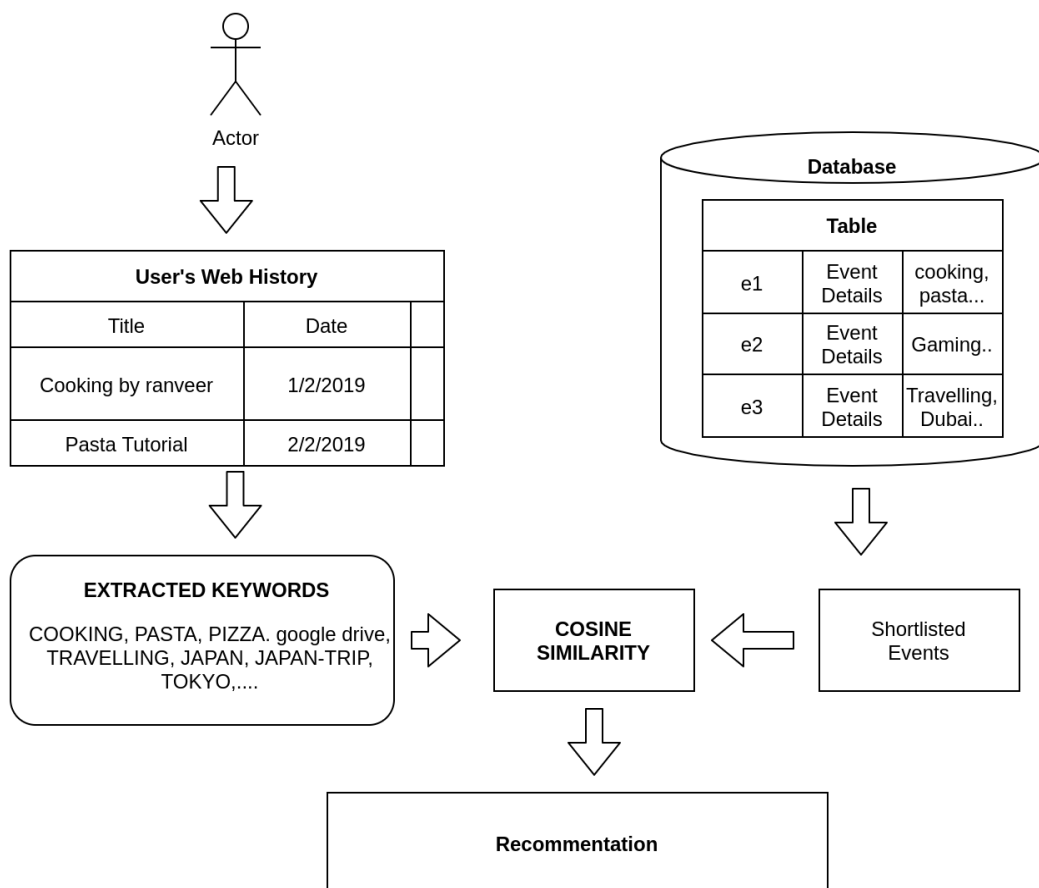


Figure (3.4.3) : Recommendation based on Event Organizer's side

## **Chapter 4 : Implementation and Experimentation of Prototype Model**

### **4.1 Structure of the project**

#### **4.1.1 File Structure**

The main folder is Event\_recommender\_system

- **history\_extractor\_extension** - this folder contains the browser extension.
- **recommender** - This part of the code is responsible for providing recommendation based on the user's web browsing history
- **Collector** - this folder is responsible for handling the implicit data that is gathered from the front-end side. Javascript makes ajax calls back to the server.
- **static** - folder contains static files such as image, CSS, javascript.
- **Template** - contains template code.
- **HelperPack** - contains code related to stop words and calculating TD\*IDF and cosine.
- **Events** - main driver code. All requests are handled here.

#### **4.1.2 How to run the project**

Download the project using the below command:

Git clone <https://github.com/vikramBhavsar/Event-Recommendation-System.git>

Change directory using:

cd Event-Recommendation-System

Find out all the branches of the repo:

git branch -a

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Checkout to the remote master branch using the following command:

```
git checkout remotes/origin/master
```

The project needs Django==3.0.7 (or above) and Python 3.5 and above.

It is recommended that the user runs the project in a virtualenv.

Check out the link below on setting up virtualenv using python

<https://uoa-eresearch.github.io/eresearch-cookbook/recipe/2014/11/26/python-virtual-env/>

Following are the list of dependencies that can be installed using **pip install**

pip install pandas - (for handling CV data)

pip install sklearn - (for training and keyword extraction)

pip install geopandas - (for recommendation of local events)

pip install geopy - (for recommendation of local events)

pip install folium - (mostly not required)

Once all the dependencies are installed make sure the database is incorrect to state using the following commands:

```
python manage.py makemigrations
```

```
python manage.py migrate
```

The GitHub repository contains pre-calculated similarity of events, keywords, etc. If in case a new CSV is provided or to be used then few scripts are needed to run before the execution of the project. These scripts calculate the similarities between events, extract keywords, etc and fill in the models with those data. Following is the order in which scripts need to be run. Also, make sure that the user is in virtualenv created using python.

```
python add_events_data_to_models_from_csv.py
```

```
python add_categories_to_events.py
```

```
python add_keywords_to_events.py
```

```
python add_similarity_to_events.py
```



## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

Once everything is set up, the user can run the project using the following command:

`Python manage.py runserver <port-number>`

Here port-number is optional. (Since browser extension uses port 8000 it is advised to use that port otherwise that link needs to be changed in background.js in extension.)

For personalized Recommendations, Recommendation install History Extractor Extension on our google chrome browser. Chromium browser does not work with handling location.

To install the google chrome extension follow the below steps:

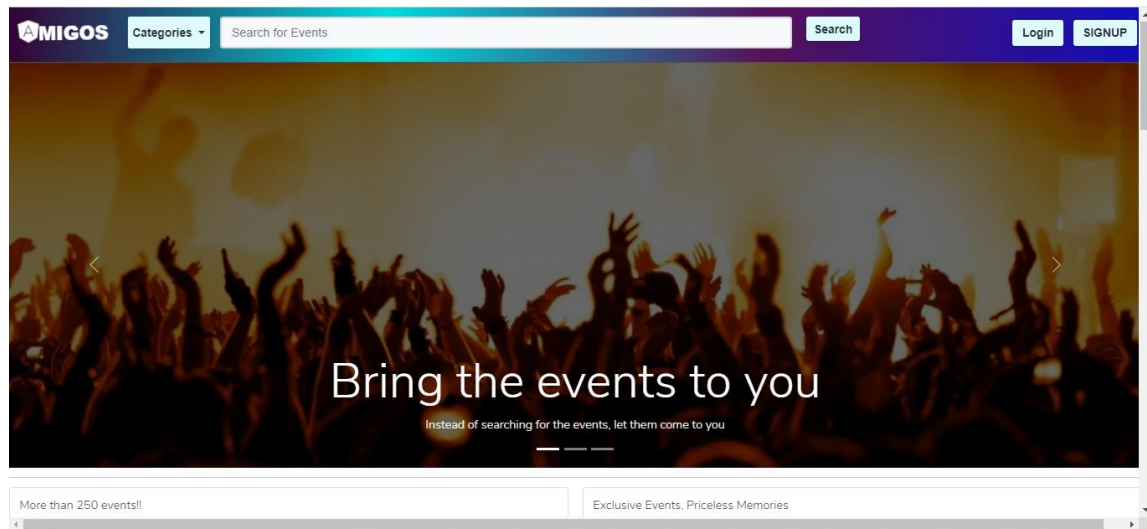
- Open chrome
- Click on the 3 dots on the top right-hand side and open settings
- Open Extensions from the left-hand menu.
- Activate the developer mode option on the top right side.
- Click on Load Unpacked.
- Go to our **Event-Recommendation-System** folder and select the folder which is named **history\_extractor\_extension**
- The browser extension will successfully install itself.

### **4.1.3 General Walkthrough of the project and basic functionality.**

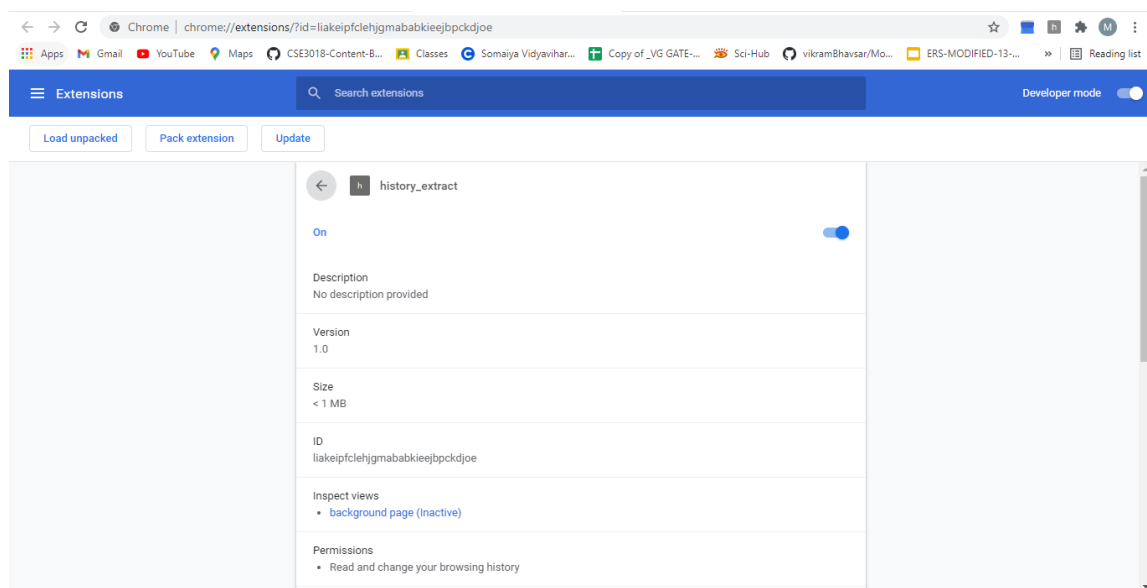
- 1) When a new user comes to our website Home page is displayed along with all categories of Events

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)



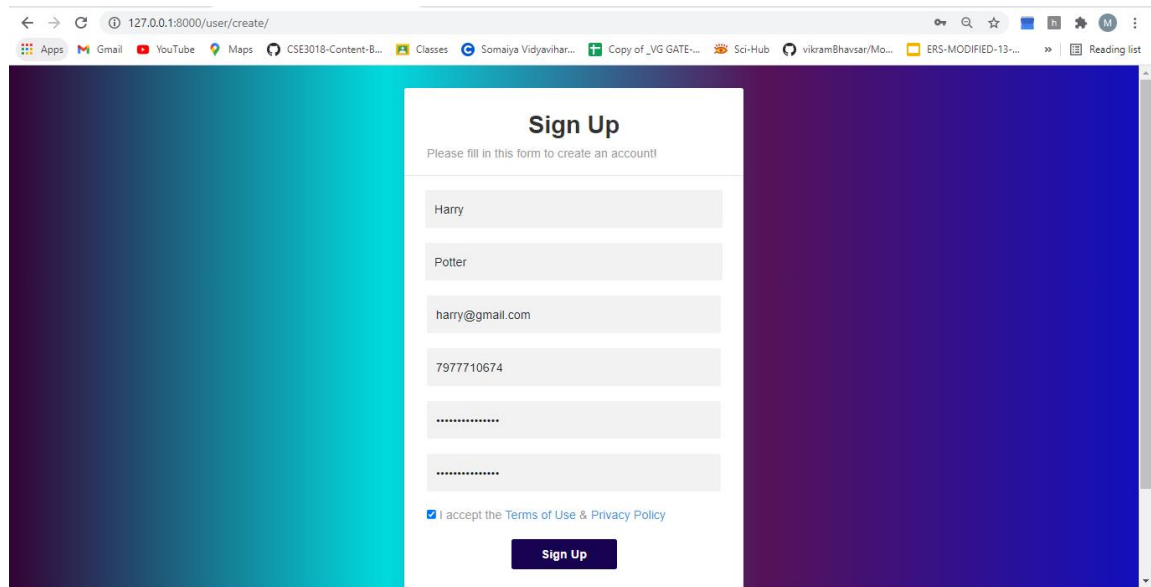
- 2) Then User needs to add an extension to the chrome browser for extracting personal history



- 3) New User needs to SIGN UP first before moving ahead with events details and description

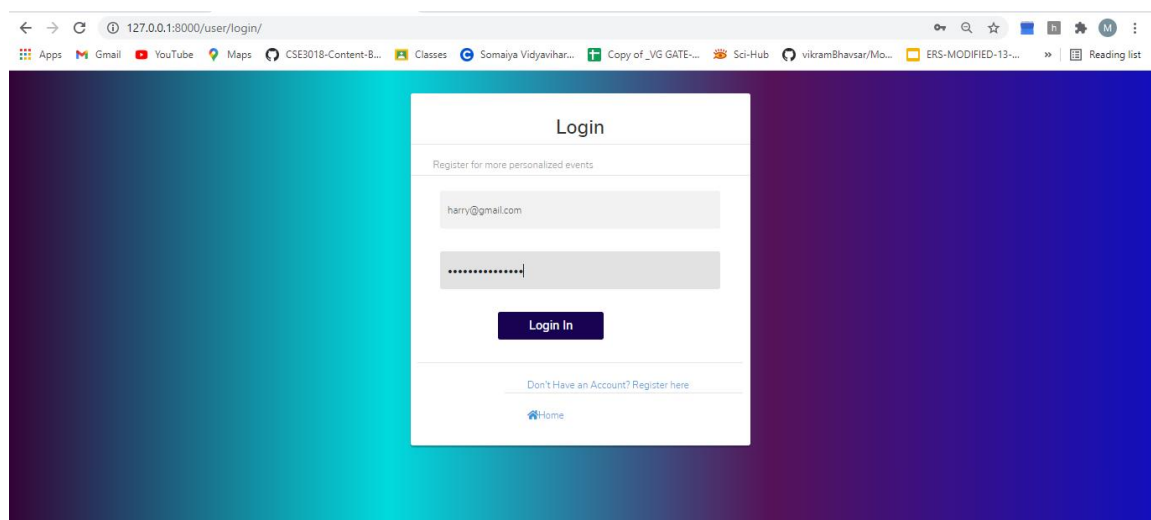
# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/user/create/". The browser's tab bar includes "Apps", "Gmail", "YouTube", "Maps", "CSE3018-Content-B...", "Classes", "Somaiya Vidyavihar...", "Copy of \_VG GATE...", "Sci-Hub", "vikramBhavsar/Mo...", "ERS-MODIFIED-13-...", and "Reading list". The main content area features a "Sign Up" form with the heading "Sign Up" and the instruction "Please fill in this form to create an account!". The form contains input fields for "Name" (filled with "Harry"), "Email" (filled with "Potter"), "Phone" (filled with "harry@gmail.com"), "Password" (filled with "7977710674"), and "Confirm Password" (filled with "7977710674"). A checkbox labeled "I accept the Terms of Use & Privacy Policy" is checked. A "Sign Up" button is at the bottom of the form.

4) After signing up user needs to Login with proper credentials

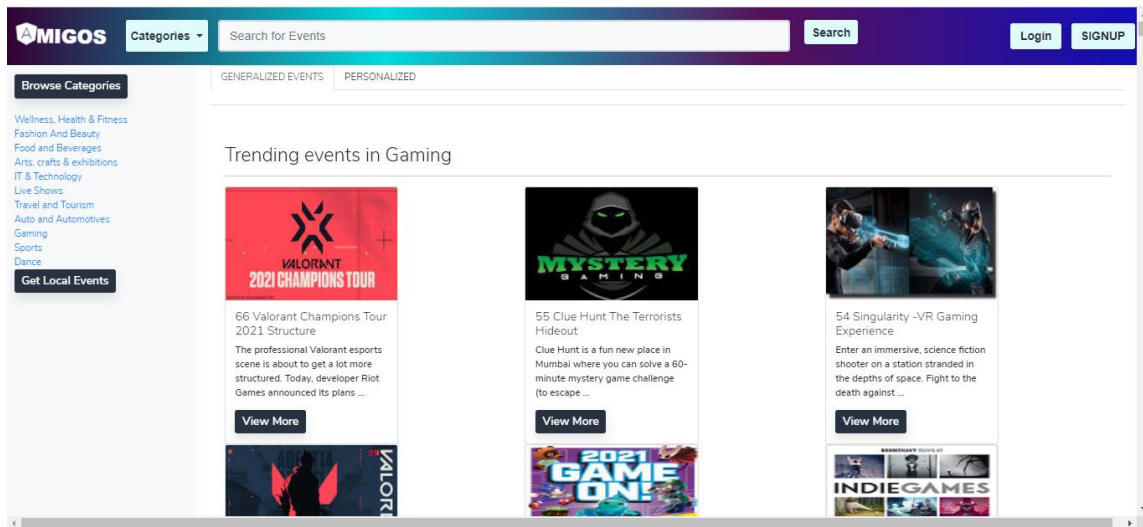


The screenshot shows a web browser window with the address bar displaying "127.0.0.1:8000/user/login/". The browser's tab bar is the same as in the previous screenshot. The main content area features a "Login" form with the heading "Login" and the instruction "Register for more personalized events". The form contains input fields for "Email" (filled with "harry@gmail.com") and "Password" (filled with "7977710674"). A "Login In" button is at the bottom of the form. Below the button, there is a link "Don't Have an Account? Register here" and a "Home" link with a house icon.

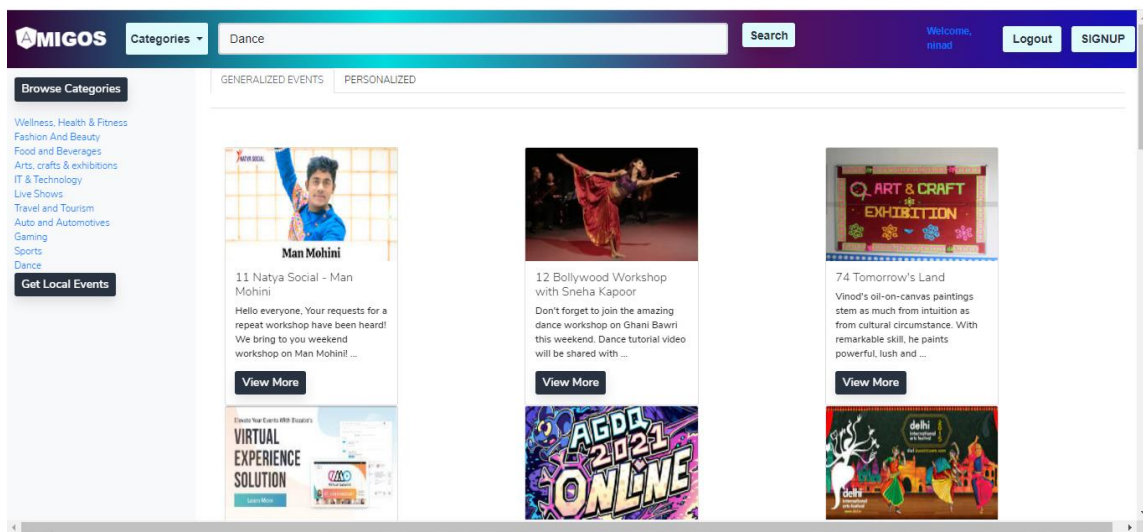
5) After logging in User gets all the trending events in each of the category mentioned on the homepage under the categories drop-down button

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)



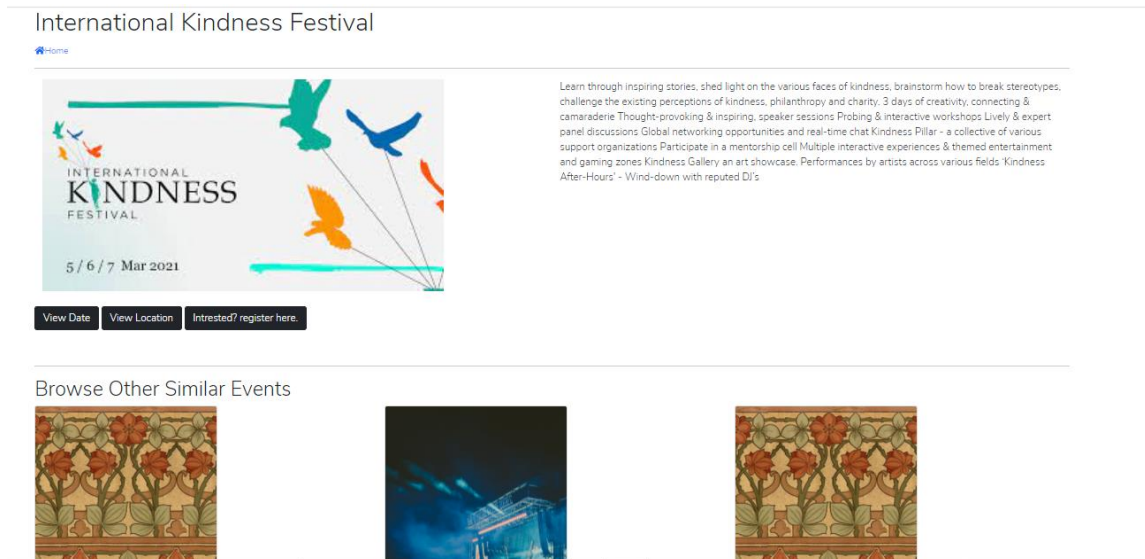
6) User can search for specific event category of his/her choice in the search bar



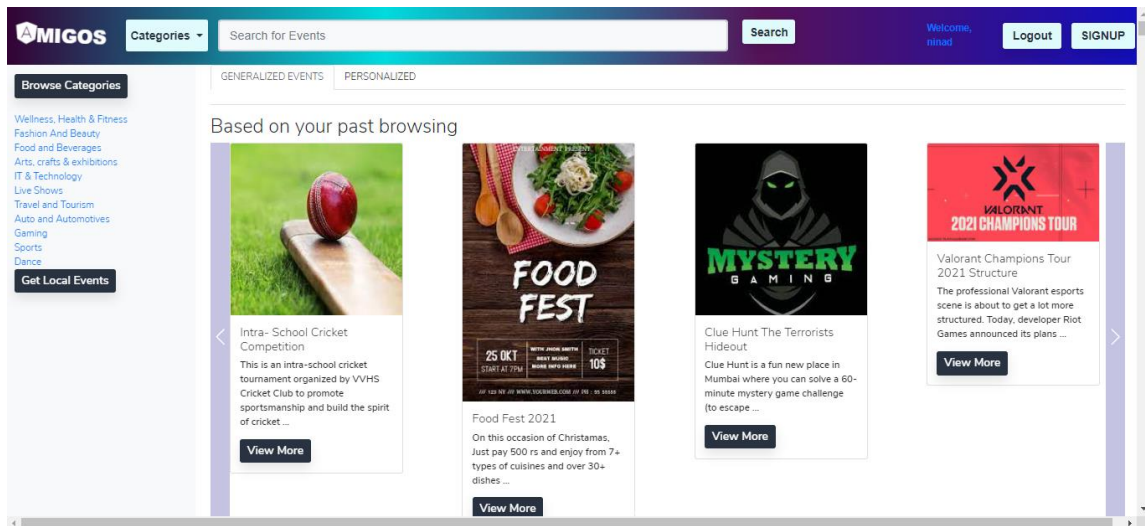
# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

- 7) By clicking on the View More option user can get a piece of detailed information about the event like date, venue and registration link, etc and also related to similar category events are displayed



- 8) User also gets similar events based on past browsing history




# K. J. Somaiya College of Engineering, Mumbai-77


(Autonomous College Affiliated to University of Mumbai)

- 9) User also gets similar events and only previous viewed event if he/she has not registered for it, into the tab of past search history


Based on your Past Search history




**Singularity -VR Gaming Experience**  
Enter an immersive, science fiction shooter on a station stranded in the depths of space. Fight to the death against ...  
[View More](#)



**2021 GAME ON!**  
GameOn 2021  
Delve into Vancouver's ultimate gamer experience for aspiring developers, students, and gaming enthusiasts. Brought to you by UBC Game Development, ...  
[View More](#)






**International Kindness Festival**  
Learn through inspiring stories, shed light on the various faces of kindness, brainstorm how to break stereotypes, challenge the existing ...  
[View More](#)




**Man Mohini**  
Natya Social - Man Mohini  
Hello everyone, Your requests for a repeat workshop have been heard! We bring to you weekend workshop on Man Mohini! ...  
[View More](#)

Trending events in Gaming




- 10) Once a user registers for a particular event then only similar events are displayed on to the past search browsing tab


Based on your Past Search history




**Bollywood Workshop with Sneha Kapoor**  
Don't forget to join the amazing dance workshop on Ghani Bawri this weekend. Dance tutorial video will be shared with ....  
[View More](#)



**ART & CRAFT EXHIBITION**  
Tomorrow's Land  
Vinod's oil-on-canvas paintings stem as much from intuition as from cultural circumstance. With remarkable skill, he paints powerful, lush and ...  
[View More](#)




**VIRTUAL EXPERIENCE SOLUTION**  
FLOOR - Solution for your Virtual events  
A recent survey showed that over 75% of event professionals are planning a virtual or hybrid event in 2020. But ...  
[View More](#)




**AGDQ 2021 ONLINE**  
Awesome Games Done Quick 2021 (Online)  
Out of an abundance of caution for the safety of the community, Games Done Quick has made the difficult decision ...  
[View More](#)


Trending events in Gaming



66 Valorant Champions Tour



55 Clue Hunt The Terrorists



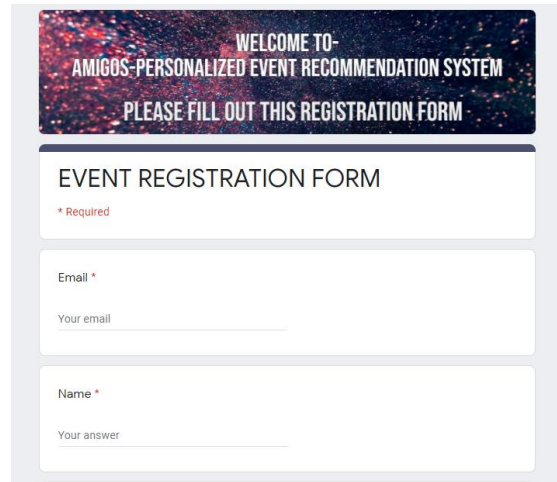
54 Singularity -VR Gaming



# K. J. Somaiya College of Engineering, Mumbai-77

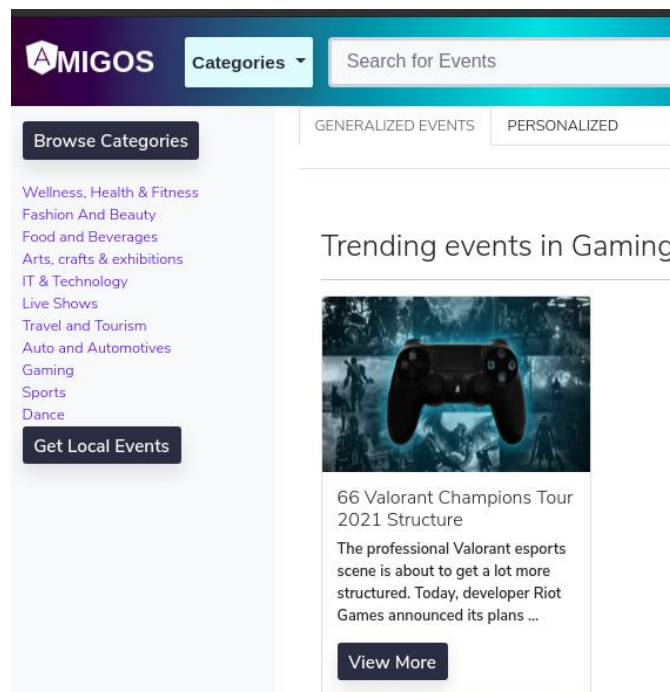
(Autonomous College Affiliated to University of Mumbai)

11) Google form is displayed for registering for a particular event



The screenshot shows a registration form titled "WELCOME TO- AMIGOS- PERSONALIZED EVENT RECOMMENDATION SYSTEM PLEASE FILL OUT THIS REGISTRATION FORM". Below the title is a section labeled "EVENT REGISTRATION FORM" with a red asterisk and the word "Required". There are two input fields: "Email \*" with a placeholder "Your email" and "Name \*" with a placeholder "Your answer".

12) There's a button on the left hand side which says Get Local Events. When clicked the browser shares the location with the server.



The screenshot shows the AMIGOS website interface. The header includes the AMIGOS logo, a "Categories" dropdown, and a "Search for Events" input field. Below the header, there are two tabs: "GENERALIZED EVENTS" and "PERSONALIZED". On the left side, there is a "Browse Categories" section with a list of categories: Wellness, Health & Fitness; Fashion And Beauty; Food and Beverages; Arts, crafts & exhibitions; IT & Technology; Live Shows; Travel and Tourism; Auto and Automotives; Gaming; Sports; and Dance. Below this list is a "Get Local Events" button. On the right side, there is a section titled "Trending events in Gaming" featuring a game controller image and a text block about the "66 Valorant Champions Tour 2021 Structure". The text block mentions that the professional Valorant esports scene is about to get a lot more structured and that developer Riot Games has announced its plans. Below the text block is a "View More" button.

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

13) Based on past search history the event **Industrial Engineering expo** is shown to the user.

The screenshot shows a recommendation section titled "Based on your Past Search history". It features three event cards with left and right navigation arrows. The first card is for the "Industrial Engineering Expo", described as "Central India's Largest SME Exhibition" and offering opportunities for businesses to showcase products and services. The second card is for the "Valorant Champions Tour 2021 Structure", mentioning the professional Valorant esports scene and Riot Games' plans. The third card is for "INDIAN UGC VALORANT SEASON 2", announcing a tournament with an entry fee of Rs200/- INR. Each card includes a "View More" button. Above the cards, a partially visible card for a cricket event also has a "View More" button. Below the cards, a section titled "Trending events in Gaming" is visible.

14) Here the venue for the event is **Vasant vihar, Delhi** but the user's location is Thane and hence it is not possible for the user to visit that event. Hence when clicked on **getting Local Events**

### Industrial Engineering Expo

[Home](#)



"Central India's Largest SME Exhibition" This Exhibition will deliver real opportunities for the businesses to showcase their products & services to key audiences, to enabling them to develop new relationships with businesses across India. It is a great business platform with enormous opportunity where participants will be rest assured will be great response at the hand.

[View Date](#) [View Location](#) [Intrested? register here.](#)

vasant vihar, delhi




# K. J. Somaiya College of Engineering, Mumbai-77


(Autonomous College Affiliated to University of Mumbai)

15) That event is no longer shown.


Based on your Past Search history




Singularity -VR Gaming Experience  
Enter an immersive, science fiction shooter on a station stranded in the depths of space. Fight to the death against ...  
[View More](#)



GameOn 2021  
Delve into Vancouver's ultimate gamer experience for aspiring developers, students, and gaming enthusiasts. Brought to you by UBC Game Development, ...  
[View More](#)





International Kindness Festival  
Learn through inspiring stories, shed light on the various faces of kindness, brainstorm how to break stereotypes, challenge the existing ...  
[View More](#)




Natya Social - Man Mohini  
Hello everyone. Your requests for a repeat workshop have been heard! We bring to you weekend workshop on Man Mohini! ...  
[View More](#)

Trending events in Gaming








16) Personalized Tab shows all the recommendation based on user' web history

**MIGOS** Categories Search for Events Search Welcome, ninaad Logout SIGNUP


Browse Categories  
Wellness, Health & Fitness  
Fashion And Beauty  
Food and Beverages  
Arts, crafts & exhibitions  
IT & Technology  
Live Shows  
Travel and Tourism  
Auto and Automotives  
Gaming  
Sports  
Dance  
[Get Local Events](#)

GENERALIZED EVENTS PERSONALIZED


PERSONALIZED EVENTS




3 International Salon + Spa Expo  
International Beaut  
[View More](#)




4 BEASA INTERNATIONAL BEAUTY SALON TRADE SHOW  
The International Be  
[View More](#)




10 IOT Training  
IoT Training is an o  
[View More](#)



12 Bollywood Workshop with Sneha Kapoor



17 Central Florida International Auto Show



29 STACK

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

## Organizer Side:

- 1) Creating/adding new events to our website from organizer's side

127.0.0.1:8000/organizer/new

Apps Gmail YouTube Maps CSE3018-Content-B... Classes Somaiya Vidyavihar... Copy of \_VG GATE-...

### Enter Event Details

EVENT NAME: Indian Art Festival

EVENT DESCRIPTION: Great Exhibition based on

EVENT GUEST: Uddhav Thackrey

EVENT LOCATION: Bandra

EVENT CATEGORY: Arts craft and exhibition

Submit

VIEW EVENTS

- 2) Organizer can view all events posted by him/her in View Events section

127.0.0.1:8000/organizer/show

EVENT NAME	EVENT DESCRIPTION	EVENT GUEST	EVENT LOCATION	EVENT CATEGORY	Actions
Al Gazibo	Food Fest	Mr Gazibo	Ghatkopar, Mumbai	Food and Beverages	<a href="#">Edit</a> <a href="#">Delete</a>
Indian Art Festival	Great Exhibition based on famous arts monuments	Uddhav Thackrey	Bandra	Arts craft and exhibition	<a href="#">Edit</a> <a href="#">Delete</a>

Add New Record

- 3) Organizer can edit as well as delete particular event from the list

127.0.0.1:8000/organizer/edit/6

Apps Gmail YouTube Maps CSE3018-Content-B... Classes Sc

### Update Event Details

Event Name: Indian Art Festival

Event Description: Great Exhibition based on

Event Guest: Uddhav Thackrey

Event Location: Mumbai

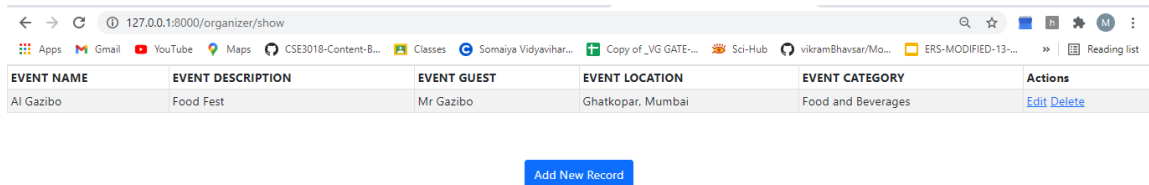
Event Category: Arts craft and exhibition

Update

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

4) After deleting recently added event



EVENT NAME	EVENT DESCRIPTION	EVENT GUEST	EVENT LOCATION	EVENT CATEGORY	Actions
Al Gazibo	Food Fest	Mr Gazibo	Ghatkopar, Mumbai	Food and Beverages	<a href="#">Edit</a> <a href="#">Delete</a>

[Add New Record](#)

## 4.2 Recommendations based on trending events

### 4.2.1 Architecture and logic

This is one of the simplest types of recommendation that is provided in this system.

All the events are categorized. Available categories are:

- Wellness, Health & Fitness
- Fashion And Beauty
- Food and Beverages
- Arts, Crafts & exhibitions
- IT & Technology
- Live Shows
- Travel and Tourism
- Auto and Automotives
- Gaming
- Sports
- Dance

To provide this recommendation the system takes into account which active event has the largest number of registered users.

### 4.2.2 Database Structure

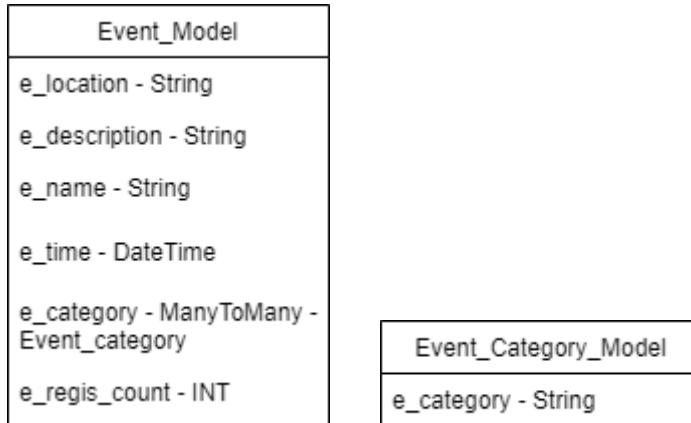


Figure (4.2.2) : Database structure

Above is the schema of how the tables are organized.

### 4.2.3 Code

Models created in Django

Code in [/events/models.py](#)

#### Event Category Model

```
class Event_category_model(models.Model):
    e_category = models.CharField(max_length=60)

    def __str__(self):
        return str(self.id) + " " + self.e_category
```

#### Event Model

# Create your models here.

```
class Events_model(models.Model):
    e_name = models.CharField(max_length=200)
    e_description = models.TextField()
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
e_guest = models.CharField()
e_location = models.CharField()
e_time = models.DateTimeField()
e_category = models.ManyToManyField(Event_category_model)
e_regis_count = models.IntegerField(default=0)
```

### Recommendation code:

Code in [events/views.py](#)

```
        if 'category_pk' not in self.kwargs and 'search_q' not in self.request.GET:
            for cat in Event_category_model.objects.all():
                context[str(cat.e_category).replace("
", "_").replace("&", "and").replace(", ", "")] =
Events_model.objects.filter(e_category=cat).order_by("-e_regis_count")
```

The above code first ensures that the user has not specifically requested any category and then it checks if the user has not provided any search query. If the user provides search query then we don't want to show trending events, we just want to show the search results. We loop each category from **Event\_Category\_Model** and get every event from that category by sorting it based on the registration count.

If in case the user is browsing any specific category then we show all trending events from that specific category. The following code is used:

Code in [events/views.py](#)

```
        if 'category_pk' in self.kwargs:
            category_pk = self.kwargs["category_pk"]
```

# K. J. Somaiya College of Engineering, Mumbai-77


(Autonomous College Affiliated to University of Mumbai)

return

```
Events_model.objects.filter(e_category=Event_category_model.objects.get(pk=category_pk)).order_by("-e_regis_count")
```

## 4.2.4 Output


Trending events in Gaming



66 Valorant Champions Tour 2021 Structure

The professional Valorant esports scene is about to get a lot more structured. Today, developer Riot Games announced its plans ...


[View More](#)



55 Clue Hunt The Terrorists Hideout

Clue Hunt is a fun new place in Mumbai where you can solve a 60-minute mystery game challenge (to escape ...


[View More](#)



54 Singularity -VR Gaming Experience


Enter an immersive, science fiction shooter on a station stranded in the depths of space. Fight to the death against ...

[View More](#)




67 INDIAN UGC VALORANT SEASON 2

We have organized valorant tournament with a entry fee of Rs200/- INR . Join our discord to



125 GameOn 2021

Delve into Vancouver's ultimate gamer experience for aspiring developers, students, and gaming enthusiasts. Brought to you by




19 Indie Games Expo 2020

Don't miss the 2020 Virtual Indie Games Expo (VIGE 2020) which will take place this December 19th, 2020. The main ...

Showing trending events from the gaming category in the **generalized tabs**.


Trending events in Sports



21 Intra- School Cricket Competition

This is an intra-school cricket tournament organized by VVHS Cricket Club to promote sportsmanship and build the spirit of cricket ...


[View More](#)



23 Street Football Tournament

If you are a local in the Mumbai-Thane area and have kids looking to ball up, make sure to make ...


[View More](#)



36 Tata Mumbai Marathon


The Tata Mumbai Marathon (previously known as Standard Chartered Mumbai Marathon) is an annual international marathon held in Mumbai, India, ...

[View More](#)




234 Tennisball cricket tournament

Tennis ball cricket tournament was held under the ST BODITTO



22 Delhi International Triathlon 2021

We are back, and now throwing Open the Gates of FIT 2021 and it



35 India Open (badminton)

The 2019 India Open was the eighth tournament of the 2019 BWF World Tour and also part of

Showing trending events from the sports category in the generalized tabs.

## 4.3 Extracting keywords from event's description using TF\*IDF

### Algorithm

#### 4.3.1 Architecture and logic

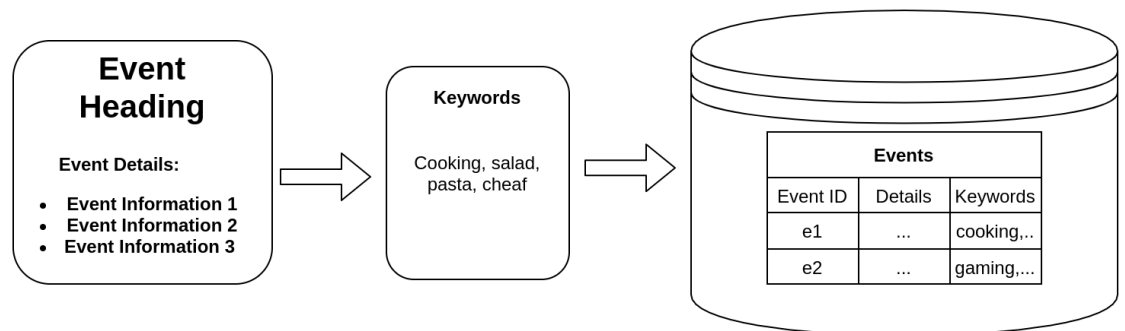


Figure (4.3.1) : Keywords storing based on particular events

Here, Each event has useful information from which keywords can be extracted and its stored in a separated model known as **Event\_keywords\_model**.

#### 4.3.2 Database Schema

Event_Keyword_Model
e_keyword - String
e_score - float
e_event - ForeignKey - Events_model

FIgure (4.3.2) : Database Schema

The column **e\_event** is a foreign key to the **Events\_model** table.




## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

The column **e\_keyword** is the keyword and **e\_score** is the TF\*IDF score of that keyword.

Example snapshot of the database looks like this:

Change event\_keywords\_model

E keyword:	<input type="text" value="mumbai"/>
E score:	<input type="text" value="0.06823233129080249"/>
E event:	<div>234 Tennisball cricket tournament   </div>
<div>Delete</div>	

The image is the single entry inside the database. Here the keyword “Mumbai” and its score is given and also to which event this keyword belongs to.

### 4.3.3 Code

Model Code - Event\_keywords\_model

Code in [events/models.py](#)

```
class Event_keywords_model(models.Model):
    e_keyword = models.CharField(max_length=50)
    e_score = models.FloatField()
    e_event = models.ForeignKey(Events_model)
```

Following is the python script which generates keywords based on stored events inside the database and stores the updated keywords in **Event\_keywords\_model**.

Code in [add\\_keywords\\_to\\_events.py](#)



## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

Following is the class which was created:

```
class KeywordsExtraction:
    def get_record_split(self,sentence):
        rege = re.compile(r'[a-z0-9]{2,}')
        # returns list of words by removing any special characters from the string.
        return rege.findall(sentence)

    def getProcessedEventData(self,stop_words):
        # gets event details after pre-processing by removing stop words and combining
        event
        # title and event Description together.
        return processed_events_list

    def tf_idf_get_keywords_list(self,event_data_list):
        # a list of keywords for a particular event that will added to the main data frame as a
        sentence
        # and will be used during cosine similarity between events and interest extracted.

    def updateKeywordsAndScoreToDatabase(self,stop_words):
        # this method is responsible to update the keywords and their respected scores into
        the database
```

Below is the driver code.

```
# Getting stop words for extraction
from HelperPack import help
stop_words = help.getStopWords()

extractioner = KeywordsExtraction()
extractioner.updateKeywordsAndScoreToDatabase(stop_words)
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Here HelperPack is a library from which help is imported. We need stopwords in order to calculate the keywords.

The first Method that is called is **updateKeywordsAndScoreToDatabase**.

**updateKeywordsAndScoreToDatabase** - it calls **getProcessedEventData** and gives that data to **tf\_idf\_get\_keywords\_list** which returns a tuple. Which is a keywords\_list, scores\_list.

Those two lists are then added as columns into the database.

**getProcessedEventData** - removes stop words and alpha numeric values from the description and title and then returns the list back to the caller method.

**Tf\_idf\_get\_keywords\_list** - uses the **TfidfVectorizer()** to generate a list of keywords and their corresponding scores of each keyword.

### 4.3.4 Output

<input type="checkbox"/>	EVENT_KEYWORDS_MODEL
<input type="checkbox"/>	16042 mumbai 0.06823233129080249
<input type="checkbox"/>	16041 teams 0.09142783312914754
<input type="checkbox"/>	16040 school 0.09142783312914754
<input type="checkbox"/>	16039 hands 0.09455026916896536
<input type="checkbox"/>	16038 participated 0.1085904780081596
<input type="checkbox"/>	16037 league 0.1085904780081596
<input type="checkbox"/>	16036 match 0.1085904780081596
<input type="checkbox"/>	16035 prizes 0.1085904780081596
<input type="checkbox"/>	16034 finally 0.1085904780081596
<input type="checkbox"/>	16033 chennai 0.1085904780081596

A table is generated after the script has been executed.

## 4.4 Calculating Similarity of events

### 4.4.1 Architecture and logic

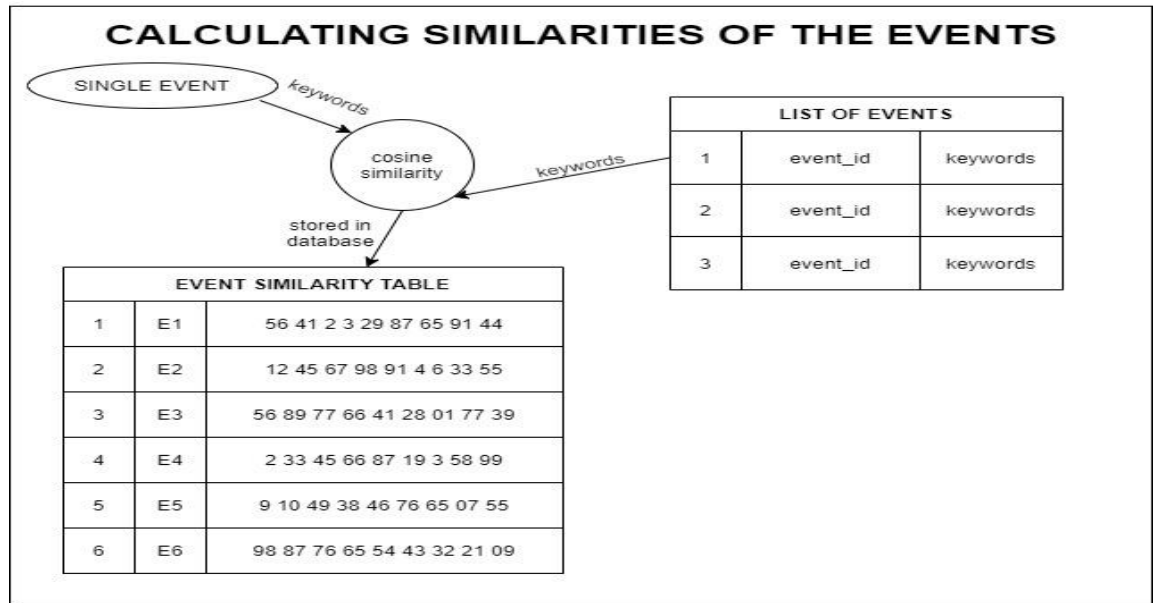


Figure (4.4.1) : Calculating Similarity of Events

When a user views any event then the detail page of the website shows another 5 events which are similar to the event. This is because similarities between events are calculated and stored inside a table.

Calculated similarities are stored in the SimilarEvents model.

### 4.4.2 Database Schema

SimilarEvents
event-oneTOOne-Events_model
similar_events= String

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

As shown in the above schema. The column **event** is oneToOne field to the **Events\_model** and column **similar\_events** is a string that contains space-separated ids of events in descending order of their similarities.

Example:

Change similar events

Event:	233 Globe-Tech Engineering Expo	▼ ✎ +
Similar events:	233 228 223 38 227 45 46 230 222 148 3 221	
<div>Delete</div>		

Here the 233 Globe-Tech Engineering Expo is the event and the list of similar events i.e 233 228 223 are other events that are most similar to that event. Also, client-side events happened more similar to 233 as compared to 223. It's in descending order of their similarity which makes sense as we only need the top 5 similar events.

### 4.4.3 Code

Model Code. Code in [recommender/models.py](#)

```
# Create your models here.
```

```
class SimilarEvents(models.Model):
```

```
    event = models.OneToOneField(Event, on_delete=models.CASCADE, primary_key=True)
```

```
    similar_events = models.CharField(max_length=200)
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Following is the code to get similar events based on the currently viewed event's primary key. The primary key of the event is taken from the URL.

Code in [events/views.py](#)

```
def get_context_data(self, **kwargs):

    cur_even_pk = self.kwargs['pk']

    similar_events_id = SimilarEvents.objects.get(pk=cur_even_pk)
    similar_events_id = similar_events_id.similar_events.split(' ')

    # converting ids to numeric

    # similar_events =
    Events_model.objects.filter(pk__in=[similar_events_id_int[1:min(6,len(similar_e
vents_id_int))]])
    similar_events =
    Events_model.objects.filter(pk__in=similar_events_id_int[1:min(6,len(similar_e
vents_id_int))]])
    context['similar_events'] = similar_events
    return context
```

Once the context has been made and passed to the response then all the events are displayed on the client side using template code as follows:

Code in [templates/events/events\\_details.html](#)

```
<div class="row">
    {% for sim_evn in similar_events%}
        <h5 class="card-title">{{ sim_evn.pk }} {{ sim_evn.e_name}}</h5>
        <p class="card-
text">{{ sim_evn.e_description|truncatewords:20 }}</p>
```

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
{% if user.is_authenticated %}
```

```
<a href="{% url 'event_detail' sim_evn.pk %}"
```

```
onclick="sendEvidenceToCollector({{user.id}},{{sim_evn.pk}},4)">View
```

```
more</a>
```

```
{% else %}
```

```
<a href="{% url 'event_detail' sim_evn.pk %}">View more</a>
```

```
{% endif %}
```

## 4.4.4 Output

### Singularity -VR Gaming Experience

[Home](#)



Enter an immersive, science fiction shooter on a station stranded in the depths of space. Fight to the death against killer robots as you explore narrow corridors, treacherous lifts, and zero-gravity environments. Zero Latency is one of the world's best free-roam VR gaming experiences where you can roam around in an area of 2000 sq. feet without wires! Featuring state-of-the-art, award-winning technology from one of the pioneers in VR entertainment, players can move around freely, as well as talk, interact and strategize with their team-mates in real-time. Choose from a variety of exciting multiplayer games ranging from cooperative like games surviving the zombie apocalypse or rescuing a space station from robots. "Step into the world of Zero Latency where Your body is the controller and your mind believes it is real" #EXPLORE #PLAY.

[View Date](#)

[View Location](#)

[Interested? register here.](#)

### Browse Other Similar Events



19 Indie Games Expo 2020  
Don't miss the 2020 Virtual Indie



124 OrcaCon 2021  
OrcaCon 2021 is an inclusive



The user is currently looking up an event on VR gaming experience and below the user can see other similar events which are Indie Games Expo, OrcaCon, etc.

## **4.5 Recommendation of events based on past Intra-website search**

### **4.5.1 Architecture and logic**

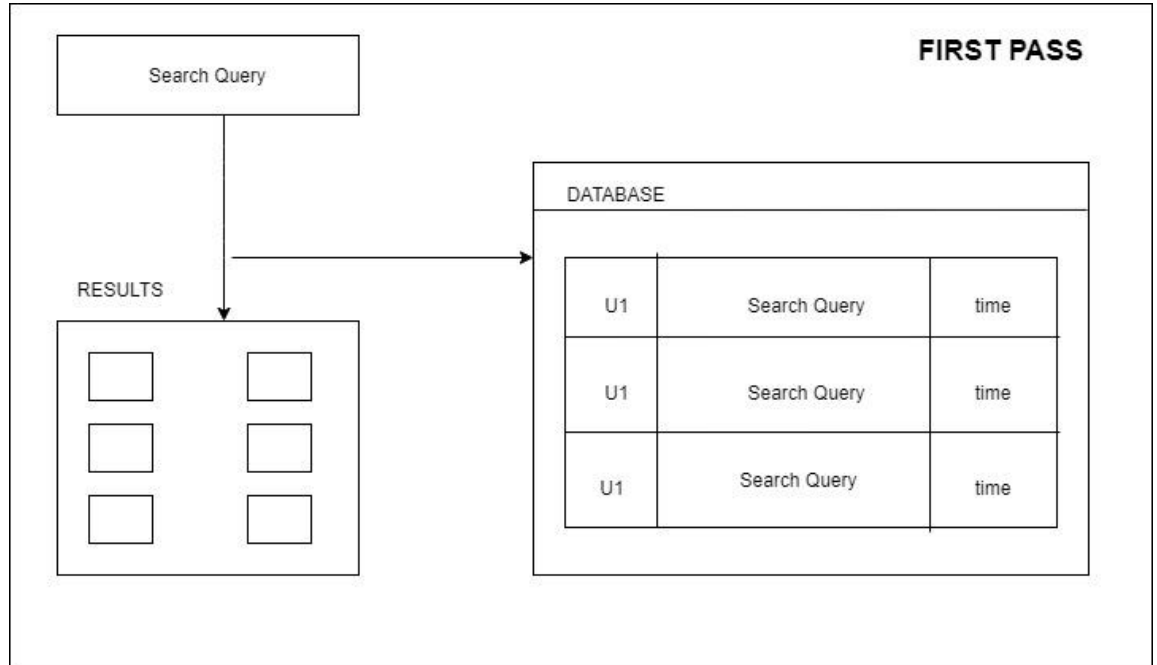


Figure (4.5.1.a) : Architecture of Recommendation Based on Past Intra-Website Search

In Order to provide powerful recommendations to the user, this website takes into account what the user has searched within the website as well. If the user logs in and searches for any specific event then it is very evident that the user is interested in that search query. Hence these search queries are used to provide the results but these are stored in the database as well. As seen in the above diagram whenever the user searches for a keyword two things happens:

- Results of that search query is shown which is normal and expected
- The search query is stored inside a table in the database.

We have named this process as “First Pass” because to provide a recommendation based on an intra-website search query requires two passes.

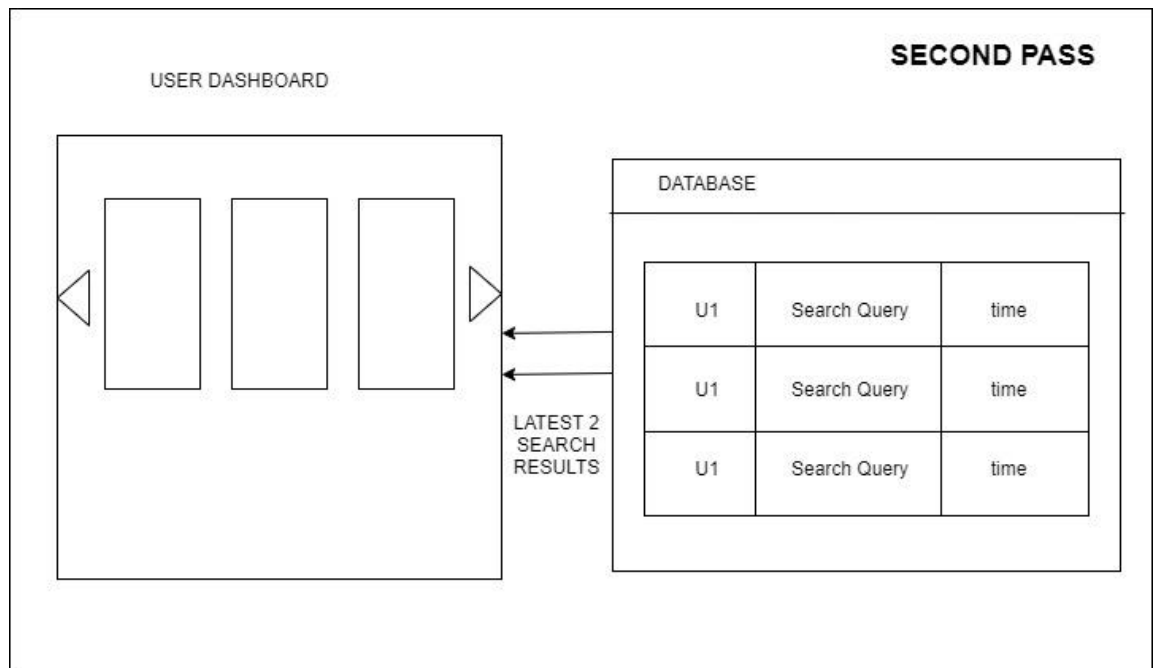


Figure (4.5.1.b) : Based on past search query results from the database

Here in the second pass, The server checks whether the logged-in user has any intra-website search query stored inside the table. If there isn't then no results are shown specifically to this. If there are records then the server gets the latest two search queries and displays them back to the user. This is part two.

#### 4.5.2 Database Schema

UserSearch
user- foreignkey to User
search_term-String
timeDetails-DateTime

Above is the schema of the table. All the UserSearch is stored inside this table. The user is a foreign key to the main user model. It's not one to one Field since there will be multiple records of a single user. **search\_term** is to store the actual term used by the user and time details is DateTimeField. Here, that is taken as default and the user does not have control over it. This field is required because we needed the latest two search results.



### 4.5.3 Code

**Database Model Code:**

Code in user\_app/models.py

```
class UserSearch(models.Model):
    user = models.ForeignKey(User)
    search_term = models.TextField()
    time_details = models.DateTimeField(default=now)
```

**Code required for the first pass:**

Code in events/views.py

Inside EventsListView -> get\_queryset() method.

```
# checking if user has searched for any events
if 'search_q' in self.request.GET:
    search_term = self.request.GET["search_q"]

    if self.request.user.is_authenticated:
        cu_user = User.get(pk=self.request.user.id)
        new_search =
        UserSearch(user=cu_user,search_term=search_term,time_details=datetime.now()
        )

        new_search.save()

        return Events_model.objects.filter(Q(e_name__contains=search_term)|
        Q(e_description__contains=search_term))

    else:
        print("Normal search is taking place.")
        return super().get_queryset()
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Here the code checks if the user is authenticated, we get the user ID and search term from the request and then store it into the UserSearch model.

We then return the search result regardless if the user is logged in or not. If the user is not logged in then we just return the search results and if the user is logged in then we first save the search query and then return the results anyway.

### Code for Pass two:

```
# getting results based on past search history
cu_user = User.objects.get(pk=self.request.user.id)
User_browse_recommends = UserSearch.objects.filter(user=cu_user).order_by("-time_details")

    # to get only last 3 searches
    search_count = 0
    for user_search in user_browse_recommends:
        search_count += 1
        # taking only last two search count to show.
        if search_count >= 4:
            break

    search_events = Events_model.objects.filter(
Q(e_name__contains=user_search.search_term)|
Q(e_description__contains=user_search.search_term))

## CODE TO HANDLE USER'S LOCATION REMOVED FOR SIMPLICITY.
More on this in "Recommending local events only"

# RETURNING THE SEARCH EVENTS
```

Here if the user is logged in we get the user id from the request and get the user object. We then use that user object to filter **UserSearch** and order the records by

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

time. We then use the search count to only use the latest two searches to show results. The list of events is then appended to the list.

```
        browse_recs_carousal_active, browse_recs_carousal            =  
get_courousel_list_for_objects(user_browse_recommendation_list)
```

```
Context["user_browse_recommendations_active"] = browse_recs_carousal_active  
context["user_browse_recommendations"] = browse_recs_carousal
```

Since the Carousel we used in front end requires an “Active” list of events and then the rest of them we divide them in two parts and attach it to the context object. These two parts are named as **browse\_recs\_carousal\_active** and **browse\_recs\_carousal**.

Code in [templates/events/events\\_list.html](#)

```
{% for eve in user_click_recommendations_active %}  
<h5 class="card-title">  
    {{ eve.e_name }}  
</h5>  
<p> {{ eve.e_description|truncatewords:20 }} </p>  
{% if user.is_authenticated %}  
    <a href="{% url 'event_detail' eve.pk %}"  
  
    onclick="sendEvidenceToCollector({{ user.id }},{{ event.pk }},4)"            >View  
    more</a>  
{% else %}  
<a href="{% url 'event_detail' eve.pk %}">View more</a            >  
{% endif %}  
{% endfor %}
```

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Small snippet of code where the active carousel events are rendered on html

## 4.5.4 Output

Change user search

User:

ninad@gmail.com

Search term:

valorant

Time details:

Date: 2021-05-07

Today

Time: 17:58:15


Now

Note: You are 5.5 hours ahead of server time.

Delete


This is a database snapshot of user ninad. The search term he used was **valorant** which is a popular game.

Based on your Past Search history




Industrial Engineering Expo  
"Central India's Largest SME Exhibition" This Exhibition will deliver real opportunities for the businesses to showcase their products & services ...

[View More](#)



Valorant Champions Tour 2021 Structure  
The professional Valorant esports scene is about to get a lot more structured. Today, developer Riot Games announced its plans ...

[View More](#)



INDIAN UGC VALORANT SEASON 2  
We have organized valorant tournament with a entry fee of Rs200/- INR . Join our discord to complete registration, remember ...

[View More](#)

Because of the above table, during pass two ninad sees this carousel where search results for **valorant** are shown.

#### 4.6 Recommendation of events based on intra-website browsing patterns

This section is mainly focused on how to make sense of various activities that the user does and what it means. At the heart of a good recommendation system is to take implicit data from the user to provide recommendations. Whenever the user interacts with a particular section on the website, in our case any event, then we can safely assume that the user is interested in that particular section, or in our case an event. We collect data from the front end using javascript when the user interacts with various events on the website.

##### 4.6.1 Architecture and logic

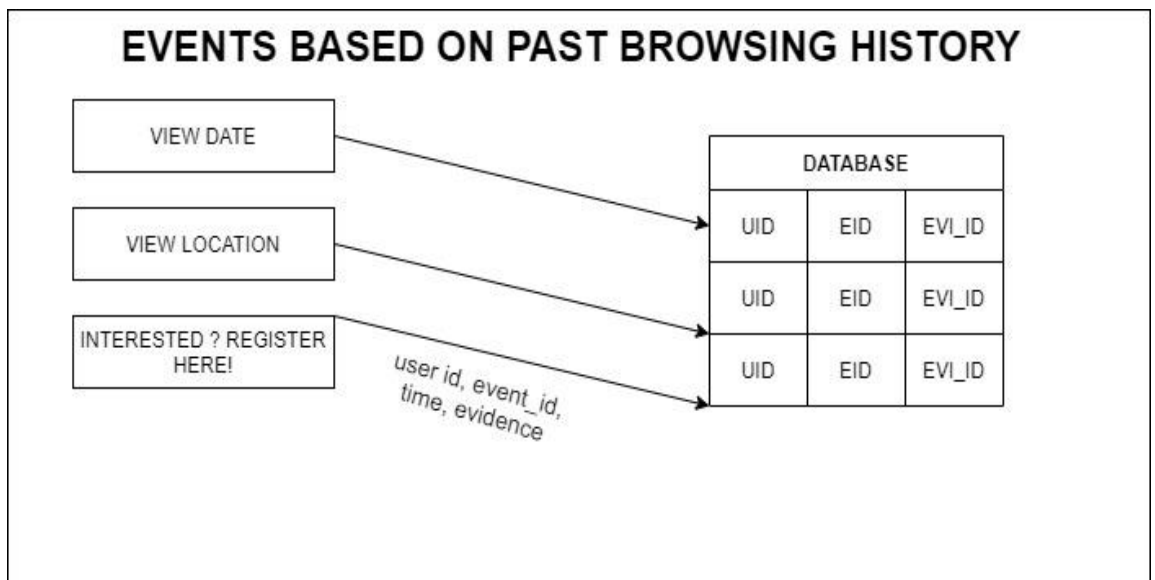


Figure (4.6.1.a) : Architecture of Recommendation based on Past Browsing History

There are a total of 4 places from where data is collected from the user's side using Javascript. Which are:

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

- When the user clicks on **view more** for an event and goes to the detail page of that event.
- When the user clicks on the **view date** of that event
- When the user clicks on the with **view location** of that event
- When the user finally registers for that event.

A table is maintained, known as **event\_user\_log** model, with 7 columns. 4 columns for 4 types of data collected from the user's side (mentioned above). And other details such as user id, event id which references the event which the user interacted with, and DateTime to order by latest events the user interacted with. Each time the user clicks on any of the four “**evidence**” i.e (View details, View date, View location, and register?) the client sends the data back to the server where it is stored inside the table.

The primary key of this table is based on User ID and Event ID together since users will interact with multiple events and each event would be interacted by multiple users.

2 TYPES OF EVENTS		
SRNO.	REGISTERED EVENTS	NON-REGSITETED EVENTS
1	REGIS=1	REGIS=0

Figure (4.6.1.b) : Types of Event

When collecting data on events there are two different types of events.

- Registered events
- Not registered events

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Registered events have the highest score, which will be explained below, but there's no point in recommending an event that has already been registered by the user. But we still take that data because now we know for sure what kind of events the user likes and hence we can show other similar events based on that registered event.

For recommending similar events refer to the section on *calculating the similarity of events*

SCORING EVENTS		
EVENT1	EVENT2	EVENT3
VIEW DATE = 1 VIEW LOCATION = 1 VIEW REGISTRATION = 1	VIEW DATE = 1 VIEW LOCATION = 1 VIEW REGISTRATION = 0	VIEW DATE = 1 VIEW LOCATION = 0 VIEW REGISTRATION = 0
SCORE = 40 + 40 + 100 = 180	SCORE = 40 + 40 + 0 = 80	SCORE = 40 + 0 + 0 = 40

Figure (4.6.1.c) : Scoring Events

### Scoring the events

All the evidence that is gathered from the front end is then used to score the events. Each piece of evidence has a weight assigned to it based on its importance. Here importance is how likely is that the user is interested. For example, if the user has registered to an event then it is very likely that the user is interested in that event but registered events cannot be recommended back to the user.

When the user reads the description and checks the date and location then it can be implied that the user is thinking of registering for that event. So evidence like **view**

## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

**date** and **view location** is also very important in terms of calculating the interest of the user. But at the same time, it is not as important as registering for the event. Similarly when the user clicks on an event to see what that event is about can be used to imply that the user is interested. But compared to other evidence types we have it is the least important of all.

Hence weight assigned to each of these pieces of evidence are:

- Is a user registered? Weight is 100
- If a user has a view date and location both? Weight is 80
- If a user has just viewed the date or location? Weight is 40
- If the user has just viewed the event? Weight is 20.

The final score of an event and user interaction is calculated by summing all of the above.

As seen in the diagram, Event 1 has a score of 180 because not only did the user inquire about the date and location together but he/she registered for the event as well. Hence  $80 (40 + 40) + 100 = 180$ .

For Event 2, the User has just viewed the date and location hence the score of that event for that particular user is  $40 + 40 + 0 = 80$ .

For Event 3, the user has just viewed the date and hence the total score is  $40 + 0 + 0 = 40$ .

Hence the recommendation of new events would be based on Event 1 and Event 2 rather than 3 since it is implied that the user is interested more in events 1 and 2 based on the evidence collected from the front end.

### **4.6.2 Database Schema**



## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

EventUserlog
user-foreignkey to user model
event-foreignkey to EventsModel
timedetails-DateTime
viewDetails-Integer
viewDate-Integer
viewLocation- Integer
viewRegistration- Integer

The table is as above schema.

### 4.6.3 Code

#### Model Code

Code in [collector/models.py](#)

```
class Event_User_log(models.Model):
    user = models.ForeignKey(User)
    event = models.ForeignKey(Events_model)
    timedetails = models.DateTimeField(default=now)
    viewDetails = models.IntegerField(default=0)
    viewDate = models.IntegerField(default=0)
    viewLocation = models.IntegerField(default=0)
    viewRegistration = models.IntegerField(default=0)
```

Firstly, the data is collected from the client-side using javascript.

Code in [static/js/customJavaScript.js](#)

```
function sendEvidenceToCollector(userid,eventid,evidenceType){
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function(){
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
}  
xhttp.open('POST','/collector/put_evidence/',true);  
  
xhttp.send(`user=${userid}&eventid=${eventid}&evidenceType=${evidenceType}`);  
}
```

This JS function is called from the HTML buttons whenever the user clicks on it.

Below code in [templates/events/events\\_list.html](#)

```
<h5>{{eve.e_name}}</h5>  
<p> {{eve.e_description|truncatewords:20}} </p>  
{% if user.is_authenticated %}  
    <a href="{% url 'event_detail' eve.pk %}"  
    onclick="sendEvidenceToCollector({{user.id}},{{event.pk}},4)">View  
more</a>  
{% else %}  
<a href="{% url 'event_detail' eve.pk %}">View more</a>  
>  
{% endif %}
```

Here user's login id and events primary key are provided by Django using template language and the evidence ID here is **4**. Which is the View details Evidence.

We also check whether the user is logged in or not because we do not want to send Evidence since we wouldn't have the user id if the user has not logged in.

Code in [templates/events/events\\_details.html](#)

```
{% if user.is_authenticated %}  
<button onClick=  
"sendEvidenceToCollector({{user.id}},{{object.pk}},3)">View Date</button>
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
<button
onClick="sendEvidenceToCollector({ {user.id} }, { {object.pk} }, 2)">View
Location</button>

<button
onClick="sendEvidenceToCollector({ {user.id} }, { {object.pk} }, 1)">Interested?
register here.</button>
{% else %}
// normal buttons
{% endif %}
```

Here the evidence id 3 , 2, existsInare, and 1 is sent from the details page which refers to view date, view location, and Registration. Whenever the button is clicked the javascript makes an ajax request back to the server.

The ajax request is made to /collector/put\_evidence/ link.

Below code in *collector/views.py*

The collector class is specifically made to collect the evidence.

Following is the post request in **CollectBrowsingData** view.

The below code is responsible for handling the post request made from the front end/client-side using javascript.

```
def post(self,request):

    userid = int(request.POST["user"])
    eventid = int(request.POST["eventid"])
    evidenceType = int(request.POST["evidenceType"])

    log_user = User.objects.get(pk=userid)
    log_event = Events_model.objects.get(pk=eventid)
    log_event.e_regis_count += 1
    log_event.save()
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
try:
# will be executed if data is already present in the database
event_log = Event_User_log.objects.get(user=log_user,event=log_event)
event_log.timedetails = datetime.now()
if evidenceType == 1:

    event_log.viewRegistration = event_log.viewRegistration + 1
// for all type of evidence
event_log.save()
return response(JsonResponse({"status":'done'}))
```

In the above code it is first checked whether the entry of a particular user and event already exists or not. If an entry exist then we update the existing entry by getting that object. If not then we create a new object and save that in the model.

At the end, we just respond with a JSON responseIn which doesn't do anything since in case an error occurs we don't really want to handle it.

### Providing recommendation based on these evidences

Below code is in events/views.py

Inside the get\_recs\_based\_on\_click\_events(user): method

```
# calculating score:
for evidence in evidences:
    score = 0
    # print("[+] Investigation %s" % evidence)
if evidence.viewRegistration > 0:
    score += 100
# checking view date and location together. (very strong chance the user is
interested)
if evidence.viewDate > 0 and evidence.viewLocation > 0:
    score += 80
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
elif evidence.viewDate > 0 or evidence.viewLocation > 0:
```

```
    score += 40
```

```
if evidence.viewDetails > 0:
```

```
    score += 20
```

```
if score > 0:
```

```
    evidences_id[evidence.event.id] = score
```

```
    # sorting the dictionary based on values.
```

```
    sorted_evidence = sorted(evidences_id.items(), key=lambda x:
x[1],reverse=True)
```

The scores are calculated and sorted in descending order since higher the score more important.

It is responsible to get the **highest\_scored\_registered\_events** and **highest\_scored\_not\_registered\_event**

We pass these two objects to the below function

```
def get_recs_from_regis_and_not_regis(high_regis_event,high_not_regis_event):
```

```
    recs_list = []
```

```
    if high_regis_event is not None and high_not_regis_event is not None:
```

```
        recs_list.append(high_not_regis_event)
```

```
    similar_list = getTopFiveSimilarEvents(high_regis_event)
```

```
    for eve in similar_list:
```

```
        recs_list.append(eve)
```

```
    similar_list = getTopFiveSimilarEvents(high_not_regis_event)
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

// FOR MORE CODE REFER THE GITHUB REPO

```
return recs_list
```

The logic here is that instead of recommending registered events again back to the user, which serves no purpose, we show other similar events to the user based on that registered user.

On the other hand, the below pieces of evidence the event which scores the highest is then again recommended to the user with the hopes that the user may register it now since they viewed it already.

### 4.6.4 Output

Action:   0 of 80 selected







<input type="checkbox"/>	EVENT_USER_LOG
<input type="checkbox"/>	harsh@gmail.com - 67 INDIAN UGC VALORANT SEASON 2
<input type="checkbox"/>	harsh@gmail.com - 28 Uttarakhand Darshan
<input type="checkbox"/>	harsh@gmail.com - 66 Valorant Champions Tour 2021 Structure
<input type="checkbox"/>	jethalal@gmail.com - 28 Uttarakhand Darshan
<input type="checkbox"/>	jethalal@gmail.com - 66 Valorant Champions Tour 2021 Structure
<input type="checkbox"/>	shamik@gmail.com - 23 Street Football Tournament
<input type="checkbox"/>	shamik@gmail.com - 28 Uttarakhand Darshan
<input type="checkbox"/>	shamik@gmail.com - 66 Valorant Champions Tour 2021 Structure
<input type="checkbox"/>	anzal@gmail.com - 147 Solo Dance Competition Online Powered By AYC
<input type="checkbox"/>	nidhi@gmail.com - 150 Groove Dance Competition

Above is the database snapshot of different users interacting with different events.

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Change event\_user\_log


User:	<input type="text" value="ninad@gmail.com"/>	 
Event:	<input type="text" value="67 INDIAN UGC VALORANT SEASON 2"/>	 
Timedetails:	<div>Date: <input type="text" value="2021-03-28"/> Today </div> <div>Time: <input type="text" value="10:53:28"/> Now </div> <div><small>Note: You are 5.5 hours ahead of server time.</small></div>	
ViewDetails:	<input type="text" value="0"/>	
ViewDate:	<input type="text" value="45"/>	
ViewLocation:	<input type="text" value="5"/>	
ViewRegistration:	<input type="text" value="1"/>	

[Delete](#)

The above is an example of user ninad interacting with the valorant season 2 event. And evidence are shown here.


[GENERALIZED EVENTS](#) [PERSONALIZED](#)

Based on your past browsing




International Conference on Nutrition & Health  
ASAR is bringing an International Conference on Nutrition & Health. This is going to be an interesting event as we ...

[View More](#)




Bollywood Workshop with Sneha Kapoor  
Don't forget to join the amazing dance workshop on Ghani Bawri this weekend. Dance tutorial video will be shared with ...

[View More](#)



Indie Games Expo 2020  
Don't miss the 2020 Virtual Indie Games Expo (VIGE 2020) which will take place this December 19th, 2020. The main ...

[View More](#)



Intra- School Cricket Competition  
This is an intra-school cricket tournament organized by VVHS Cricket Club to promote sportsmanship and build the spirit of cricket ...

[View More](#)

Based on your past browsing



Based on those data, scores are calculated and then events are recommended back to the user

## 4.7 Extracting User's Web browsing history

### 4.7.1 Architecture and logic

This is one of the most important parts of our project. We need a user's history in order to provide better recommendations based on their web browsing history.

This is a complex logic and hence we need help with a browser extension because JavaScript has more power as a browser extension compared to normal web pages.

We need access to the user's web browsing history which is simply not possible using normal JavaScript that we pass during serving requests.



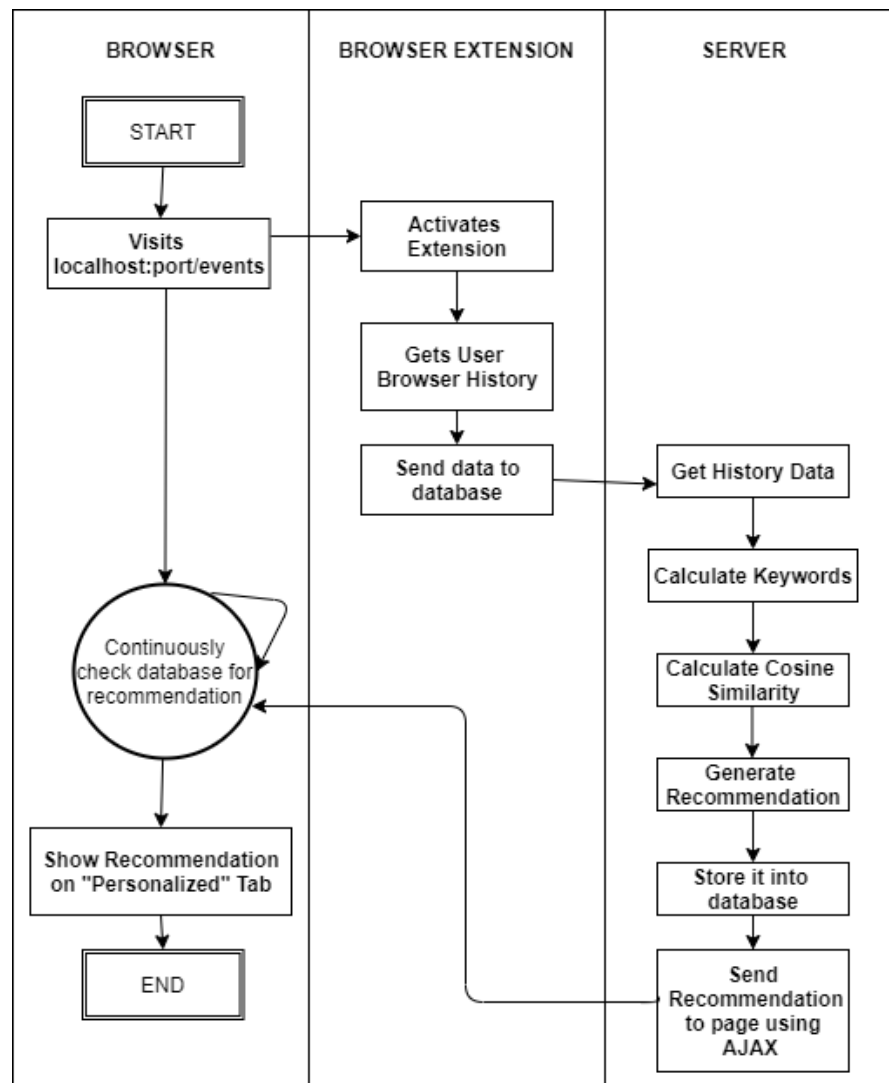


Figure (4.7.1) : Architecture of Extracting User's Web History

Here whenever the client visits the link localhost: port/events the browser extension that is installed is automatically activated.

The task of browser extension is to gather the user's browsing history into a JSON data and then send it to the server. The browser extension also extracts User ID from the page visited which is a hidden field inside in the main page. If the hidden field is missing that means the user is not logged in and hence the extension won't send any data.

## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

While at the same time the front end webpage i.e */events* continuously sends requests to the server checking if recommendations are available with respect to the user's browsing history.

When the browser extension has gathered all the History data it then sends it to the server. The Server calculates the recommendation and stores it temporarily inside the database.

When the data is ready the client side which is continuously checking the database for new records using ajax finds new records. Hence server replies with the new recommendation and the javascript on the front end render that on the client side access last access time two-three

### **4.7.2 Code**

Here is the manifest file which is required to make web browser extensions

Code in *history\_extractor\_extension/manifest.json*

```
{
  "name": "history_extract",
  "version": "1.0",
  "manifest_version": 2,

  "background": {
    "scripts": ["background.js"],
    "persistent": false
  },

  "content_scripts": [
    {
      "matches": ["http://127.0.0.1:8000/events/"],
      "js": ["content.js"]
    }
  ],
}
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
"permissions":[
  "history"
]
}
```

Here the inside the **content\_scripts** the matches is the property which says that whenever the link <http://127.0.0.1:8000/events/> is used inside the browser then execute the **content.js** file of the extension.

Below is the code responsible to check the hidden field for UserID. If the user id is not present then it means that the user has visited the link but he/she is not logged in.

Code in *history\_extractor\_extension/content.js*

```
var userHiddenField = document.getElementById("user_id_hidden_field");

if(userHiddenField.value != 'not_logged'){
  // alert(`User ID is ${userHiddenField.value}`);
  chrome.runtime.sendMessage({text:          userHiddenField.value},
function(response) {
  console.log(`${response}`) });}
```

When the user is indeed logged in then the **content.js** sends a message to the **background.js**

- Content.js - Run in the context of the web pages.
- Background.js - constantly runs in the background regardless of any website visited.

Background JS is then responsible for extracting the history.

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Code in [history\\_extractor\\_extension/background.js](#)

```
chrome.runtime.onMessage.addListener(function(msg, sender, sendResponse) {  
  var userId = msg["text"];  
  
  chrome.history.search({text: "", maxResults: 9999999, startTime: 0},  
    function(data){  
      // data is history  
      data.forEach(function(page) {  
  
        if(lastGotIT == false){  
          latestAccessedTime = page.lastVisitTime;  
          singleHistory["title"] = page.title; // other information  
  
          historyJSON.push(singleHistory);  
        }  
        moreData = {}  
        moreData["latestAccessedTime"] = latestAccessedTime;  
        historyJSON.push(moreData);  
        sendHistoryToServerUsingPOST(historyJSON);  
        sendResponse("something");  
      });  
    });  
});
```

Above is the method which gets executed when called by the content.js.

It accesses the user's browsing history and also access the **lastAccessTime** which is the date and time of the latest website visited by the user. This will be needed for calculating recommendations.

Once all the data is gathered the method internally calls **sendHistoryToServerUsingPOST(historyJSON)** where historyJSON is the web browser's history in json format.

Within the same file, following method is used for sending data back to the server.

Code in *history\_extractor\_extension/background.js*

```
function sendHistoryToServerUsingPOST(historyJSON){  
    var xhttp = new XMLHttpRequest();  
    xhttp.onreadystatechange = function(){  
        console.log(xhttp.responseText);  
    }  
    xhttp.open('POST','http://127.0.0.1:8000/recommender/index/',true);  
    xhttp.send(JSON.stringify(historyJSON));  
}
```

The next section discusses how the data is received through the POST and further calculations are done for event recommendations.

#### **4.8 Calculating similarity between User's web browsing history and events and recommendations**

Following section discusses how the recommendations are calculated based on the extracted history from the client browser.

#### 4.8.1 Architecture and logic

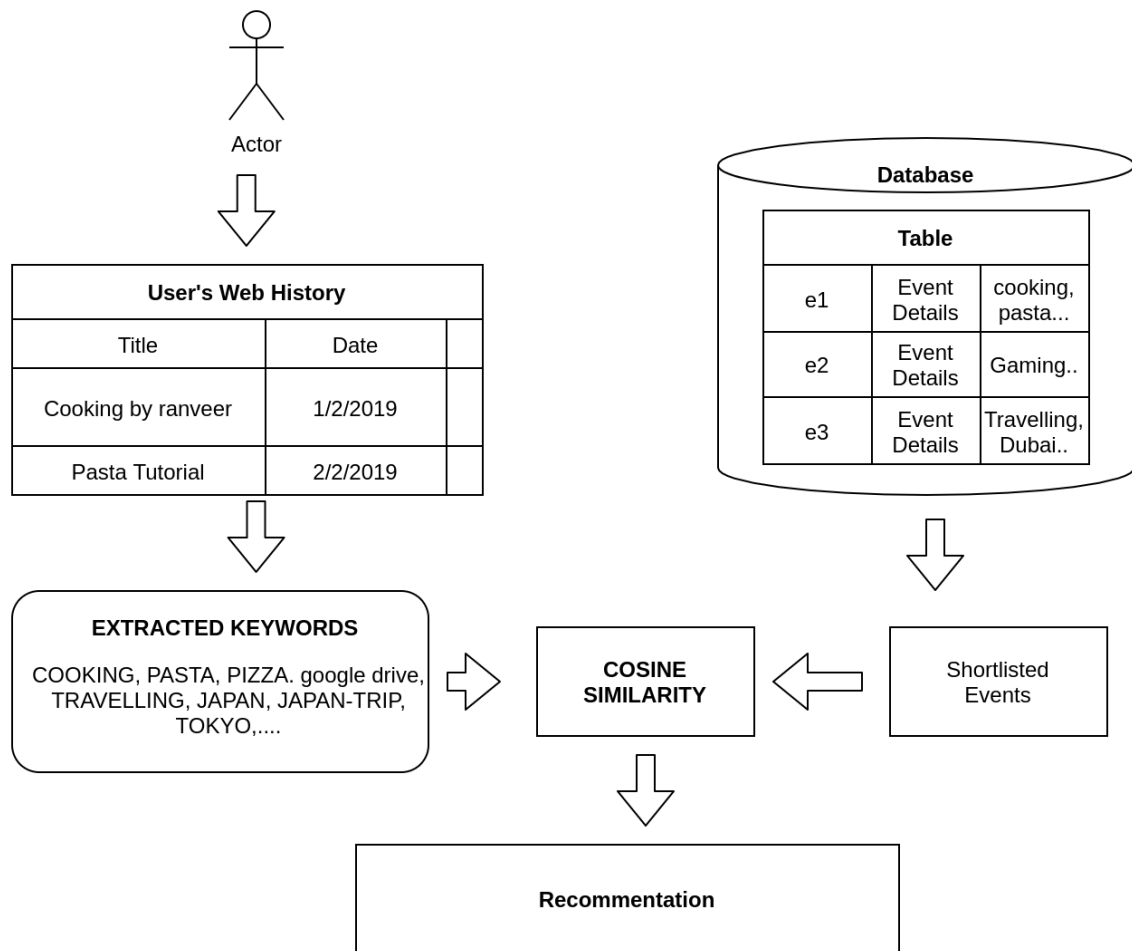
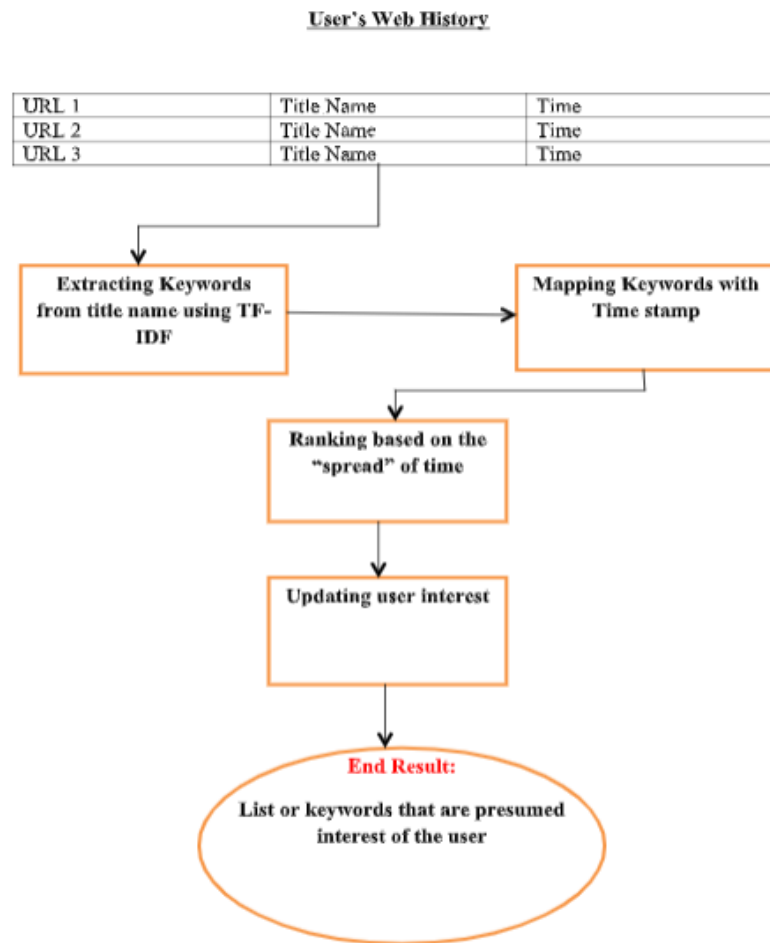


Figure (4.8.1) : Architecture for Calculating similarity between user's web history and events

There's useful information on the title of the web page that the user visits. For example, The following the JSON concerning front-end the with pieces of evidence if the user happens to visit the news section of valorant game using the link: <https://playvalorant.com/en-gb/news/> the title of that web page is "VALORANT: RIOT GAMES". Hence title is our only and a very good source of information for keywords. The keywords which are extracted from the user's history are then matched with the keywords extracted from various event's description and title. Cosine Similarity is used to calculate similarity between the two and based on that recommendations are provided.

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)



Here from the list of titles available. They are pre-processed such as removing stop words, removing numbers, the the two three, and converting every uppercase character to lowercase. Then each keyword is matched with the timestamp. The timestamp is the time when that particular keyword occurred. There may be a single keyword that would have multiple timestamps and that's the point. Keywords that are repeatedly found are of importance to us. Timestamp and time details are required to count the spread of that keyword. The understanding is that if repeated words have more spread it means that is something a user is interested in. If the spread is less but the count is more then it's just noise that the user was searching for or came across a topic for a day. For example: if a user searches about chairs for a single day but many times the count of the word chairs would be a lot but then again there isn't much "spread" of time between their access. They were

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

accessed consistently and repeatedly in a short period which does not amount to something that the user may be interested in.

On the other hand, if the user searches about valorant game one day, then after two-three days and then again after a week, here not only the individual count rise but also the “spread” is more and can be implied that since user is interested in the topic he/she is regularly invested to research about that topic.

Any person interested in dance would look up dance videos and workshops regularly. Spread here would be more.

After the spread is calculated these keywords are ranked in descending order because the higher the spread score, the more important the keyword is to the user.

To keep the privacy of the user, the history is not stored inside the database. Whenever the server receives the history immediately the recommendations are calculated and then only event ids which are the calculated recommendations are stored inside the database keeping the privacy of the user.

They calculated keywords and the spread score and keywords from events and their score are taken together and cosine similarity is calculated to recommend events to the user.

### 4.8.2 Database Schema

HistoryRecommendedEvents
user oneToOne- User_model
rec_events-String
latest_update-DataField

Figure (4.8.2) : Database Schema

Above is the schema used to store the calculated recommendations?

### 4.8.3 Code



## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

As seen in the earlier section, the browser extension makes a POST ajax call to the server giving the history data.

Below code is in *recommender/views.py*

```
elif request.method == 'POST':  
    jsonHistory = json.loads(request.body)  
  
    // For simplicity code is reduced .refer the github repo  
    putRecInDatabase(sorted_recs,user_id)  
    return HttpResponse("Received Data from server")
```

The code makes use of the below of the helper pack.

```
class HistoryRecommendation():  
  
    def get_record_split(self,sentence):  
    def main_driver(self,records,latest_access_time,  
stop_words):  
    def getSpread(self,word_counts,  
latest_access_time):
```

Here the main driver code is `main_driver` which produces **sorted\_keywords**, **keywords**,**keyword\_set**,**max\_spread**,**min\_spread** for us.

- Max\_spread is the maximum spread
- Min\_spread is the minimum spread
- Keyword\_set is just a set of keywords to check whether any keyword is available. It is used for fast checking
- Keywords are the dictionary of keywords with their individual score.
- Sorted\_keywords is the list of keywords which are sorted in descending order based on their score.

## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

Implementation of the main\_driver function is as follows:

```
def main_driver(self,records,latest_access_time,stop_words): // Code is reduced
for simplicity sake
return sorted_keywords,keywords,
keyword_set,max_spread,min_spread
```

To generate individual keywords properly below function is used which in turn makes uses of regular expressions

```
def get_record_split(self,sentence):
    rege = re.compile(r'[a-z0-9]{2,}')
    # returns list of words by removing any special characters from the string.
    return rege.findall(sentence)
```

It is responsible for splitting a sentence into words.

The main driver also uses following function to calculate the spread of the keywords:

```
def getSpread(self,word_counts,latest_access_time):
    max_spread = 0
    // code is reduced for simplicity sake. Refer to the github repo
    return keywords, keyword_set, max_spread, min_spread
```

After getting these keywords from the user's history side we calculate the cosine similarity. Cosine similarity is calculated from the below function.

Code in [recommender/views.py](#)

```
def
getCosineBetHistoryAndEvents(sorted_keywords,keywords_dictionary,user_key
words_set,max_spread,min_spread):
```

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

// for simplicity sake code is reduced. Refer to the github repo

```
return sorted_tuples
```

And then finally putting those recommendations into the database using the below function in the same file.

```
def putRecInDatabase(sorted_recs,user_id):
```

```
    cu_user = User.objects.get(id=user_id)
```

```
        // For simplicity sake code is reduces. Refer to the github repo
```

```
    reco = HistoryRecommendedEvents(user=cu_user,
```

```
    rec_events=recs)
```

```
    reco.save()
```

### 4.8.4 Output

Change history recommended events

User:

ninad@gmail.com



Rec events:

12 58 66 67 69 75 81 131 141 111 10

Latest update:

2021-05-26

Today |

Note: You are 5.5 hours ahead of server time.

Delete

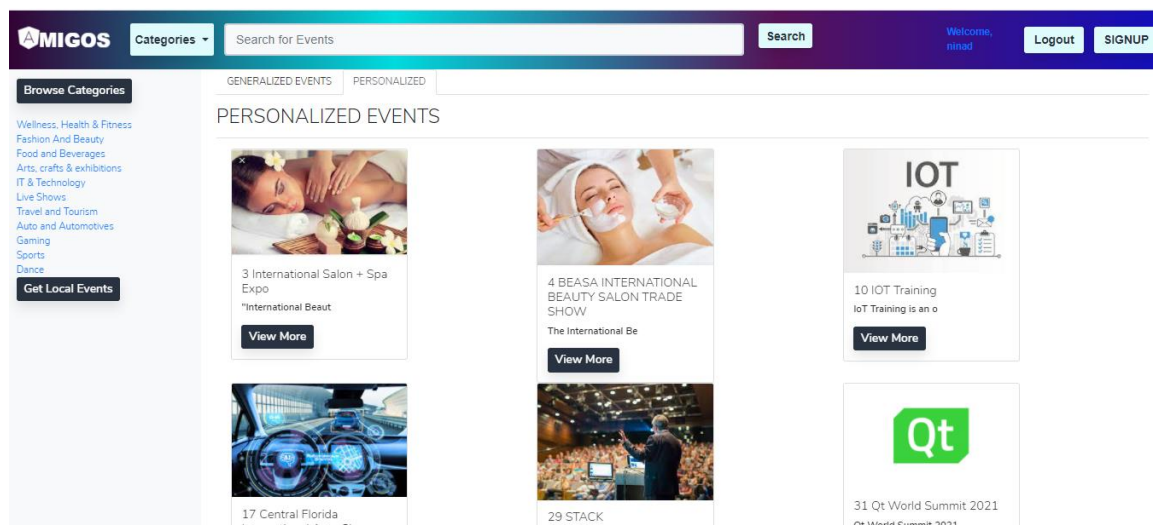
This is a snapshot of the database from Django Admin. Here the user is ninad and the calculated recommendation of the event is stored space separated as a CharField. Date is given to ensure the records are always updated.

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

<input type="checkbox"/>	1:34 AM		9 REASONS Why Your AIM SUCKS - Valorant Tips, Tricks, & Guide - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:34 AM		RETAKE // Episode 2 Cinematic - VALORANT - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:34 AM		Valorant Now   Playing Connect 4 with Subscribers Idiscord - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:34 AM		LIVE! Valorant with Ryan Higa, Fuslie, Kkatamina and Yvonne! :D - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:34 AM		VALORANT : FUNNY MOMENTS THAT MAKE YOU LAUGH SO HARD #6 - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:34 AM		Clips you would NEVER Believe existed in VALORANT! - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:34 AM		My Teammates vs YOUR Teammates #3 - VALORANT - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:33 AM		valorant - YouTube	www.youtube.com	⋮
<input type="checkbox"/>	1:33 AM		YouTube	www.youtube.com	★ ⋮

Above is the snapshot of the web history that is present in the browser. And below we can see that few valorant recommendations are provided along with others as well



The output of the personalized tab. Here the final recommendations are shown.

## 4.9 Recommending local events only.

### 4.9.1 Architecture and logic

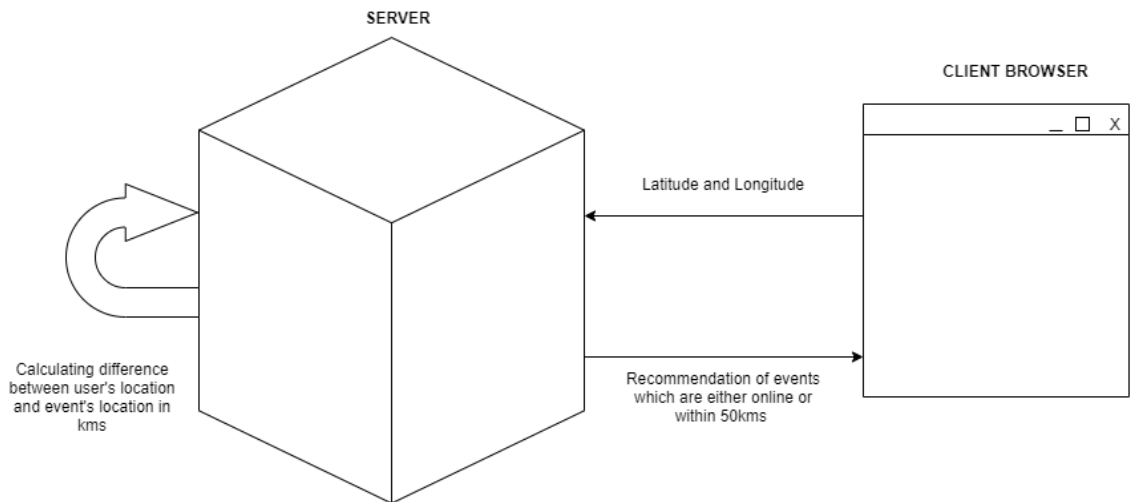


Figure (4.9.1) : Architecture for recommending local events

Here the main page allows a button and option to show local events. When clicked on local events the browser automatically shares the user's latitude and longitude coordinates to the server. The server has event data and each event data has a location field. This could be Vasant Vihar, Delhi or Manpada, Mumbai etc. Based on this information the server is able to extract the coordinates of that location. So now the server has coordinates of the event as well as coordinates of the user and hence the server can calculate the distance between the event venue and user. Using this information server can calculate the distance between them in km. So with the aim of suggesting only the local events, the server does not recommend the event to the user where the distance is more than 50kms

#### 4.9.2 Code

Following is the JavaScript code which is responsible for extracting the user's location.

Code in [static/js/customJavascript.js](#)

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
function getLocation() {  
    if (navigator.geolocation) {  
        navigator.geolocation.getCurrentPosition(showPosition,error);  
    }else{  
        alert("location not available");}}  
  
function showPosition(position) {  
    var hidden_location_field = document.getElementById("user_location");  
    hidden_location_field.value      =      `${position.coords.latitude}--  
    ${position.coords.longitude}`;
```

Code below is in *templates/events/events\_list.html*

```
<script type="text/javascript">  
    getLocation();  
</script>
```

This code is at the bottom so it is called when all the above HTMLto suggest elements are properly rendered.

There is a hidden form field inside the HTML. Code is in *templates/events/events\_list.html*

```
<form>  
    <input type="hidden" id="user_location" name="user_location" value="_">  
    <button type="submit">Get Local events</button>  
</form>
```

When the page is rendered, the Html to that left-handthatDelhi javascript function extracts the latitude and longitude and replaces the value with those coordinates.

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

```
<form>
    <input    type="hidden"    id="user_location"    name="user_location"
value="19.2214739--72.9814762">
    <button class="btn" type="submit">Get Local events</button>
</form>
```

Like this and hence when the button is clicked the value is then shared with the server.

Code in [events/views.py](#)

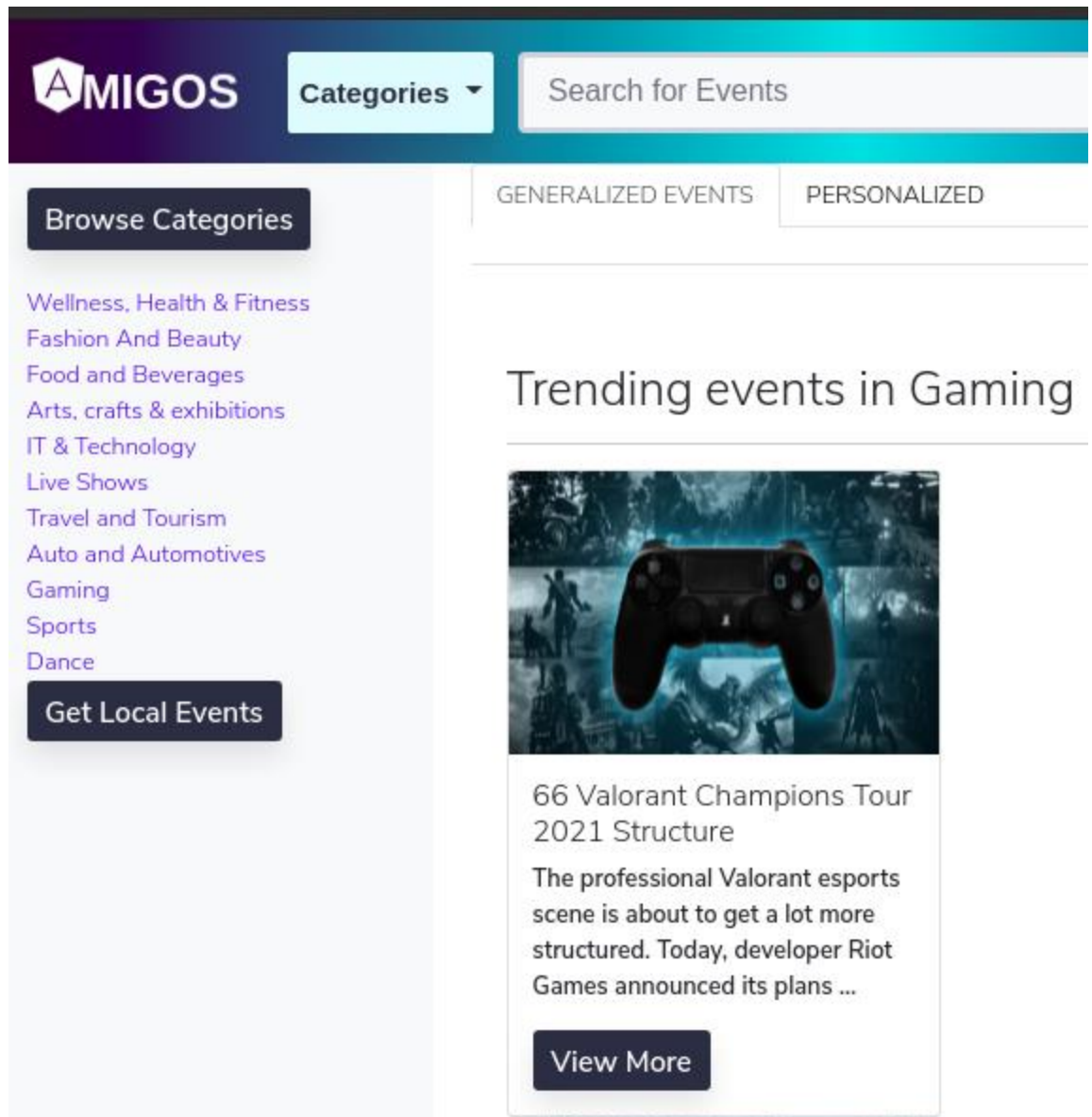
```
if 'user_location' in self.request.GET:
    // for simplicity code is reduced. Refer the github repo.
user_clicks_recommends = user_clicks_recommends_temp
```

We don't want to restrict “online” events from the user hence that is excluded from the search. The code above is extra code that is added to remove events which are far away from the user.

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

### 4.9.3 output



There's a button on the left hand side which says **Get Local Events**. When clicked the browser shares the location with the server.




# K. J. Somaiya College of Engineering, Mumbai-77


(Autonomous College Affiliated to University of Mumbai)

of cricket ...  
[View More](#)


Based on your Past Search history



Industrial Engineering Expo  
"Central India's Largest SME Exhibition" This Exhibition will deliver real opportunities for the businesses to showcase their products & services ...  
[View More](#)



Valorant Champions Tour 2021 Structure  
The professional Valorant esports scene is about to get a lot more structured. Today, developer Riot Games announced its plans ...  
[View More](#)



INDIAN UGC VALORANT SEASON 2  
We have organized valorant tournament with a entry fee of Rs200/- INR . Join our discord to complete registration, remember ...  
[View More](#)

Trending events in Gaming

Based on past search history the event **Industrial Engineering expo** is shown to the user.

## Industrial Engineering Expo

[Home](#)



[View Date](#)

[View Location](#)

[Intrested? register here.](#)

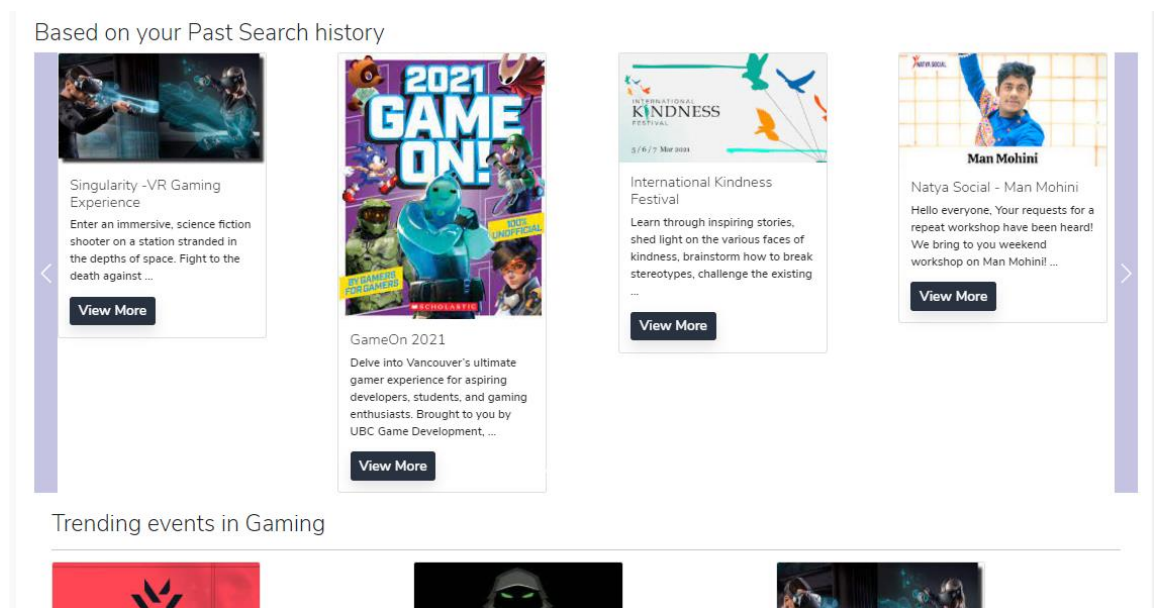
"Central India's Largest SME Exhibition" This Exhibition will deliver real opportunities for the businesses to showcase their products & services to key audiences, to enabling them to develop new relationships with businesses across India. It is a great business platform with enormous opportunity where participants will be rest assured will be great response at the hand.

vasant vihar, delhi

Here the venue for the event is **Vasant vihar, delhi** but the user's location is Thane and hence it is not possible for the user to visit that event. Hence when clicked on **getting Local Events**

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)



That event is no longer shown.

## 4.10 Testing

### 4.10.1 Black Box Testing

#### REGISTRATION OF USER AND EVENT ORGANIZER

→ Equivalence Class Partitioning to Verify UserID (only alphabets; 8-12 characters)

Valid	Invalid
a-z EC1	0-9 EC3
A-Z EC2	Special Chars EC4
-	Blank Field5 EC5

## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

### **TEST CASES-**

<b>TEST CASE ID</b>	<b>INPUT</b>	<b>EXPECTED OUTPUT</b>	<b>EQUIVALENCE CLASS-TESTED</b>
#TC01	vishvesh	Registration will proceed	EC1
#TC02	SamikshA	Registration will proceed	EC2
#TC03	Vik09ram	Registration will not proceed	EC3
#TC04	@nz@l***	Registration will not proceed	EC4
#TC05	n id hi	Registration will not proceed	EC5

### **LOGIN**

→ Equivalence Class Partitioning Verify Password (only alphabets ; 8-12 characters)

<b>Valid</b>	<b>Invalid</b>
a-z (EC1)	Blank Field (EC5)
A-Z (EC2)	
0 - 9 (EC3)	
Special Characters (EC4)	

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

### TEST CASES-

TEST CASE ID	INPUT	EXPECTED OUTPUT	EQUIVALENCE CLASS-TESTEDthat
#TC01	vishveshh	Registration will proceed	EC1
#TC02	SAmikSHA	Registration will proceed	EC2
#TC03	anzal123	Registration will proceed	EC3
#TC04	v!kram@5	Registration will proceed	EC4
#TC05	n! dh!5	Registration will not proceed	EC5

### → Boundary Value Analysis for Login

It accepts valid User Name and Password field and accepts minimum 8 characters and a maximum of 12 characters.

Valid range 8-12

Invalid range 7 or less than 7 and

Invalid range 13 or more than 13.

### (BEST CASE)

Min = 8

Min+ = 9

Nom = 10

Max = 12

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Max- = 11

N = 2 variables

$$(4n+1) = (4*2+1) = 9$$

Case	A	B	Expected Output
1	10	8	Valid
2	10	9	Valid
3	10	10	Valid
4	10	11	Valid
5	10	12	Valid
6	8	10	Valid
7	9	10	Valid
8	11	10	Valid
9	12	10	Valid

### → Decision Table for Login Screen (Username and Password)

- T – Correct username/password
- F – Wrong username/password
- E – Error message is displayed
- H – Home screen is displayed

- 1) Rule 1 – Username and password both were wrong. The user is shown an error message.

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

- 2) Rule 2 – The username of was correct, but the password was wrong. The user is shown an error message.
- 3) Rule 3 – The username was wrong, but the password was correct. The user is shown an error message.
- 4) Rule 4 – Username and password both were correct, and the user navigated to the homepage

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Username(T/F)	F	T	F	T
Password(T/F)	F	F	T	T
Actions				
Output(E/H)	E	E	E	H

## EVENT LIST

There are a total 234 events stored in the database.

[←](#) [→](#) [↻](#) [127.0.0.1:8000/events/234/](#)

[←](#) [→](#) [↻](#) [127.0.0.1:8000/events/234/](#)

Apps

Gmail

YouTube

Maps

CSE3018-Content-B...

Classes

Somaiya Vidyavihar...

Copy of \_VG GATE-...

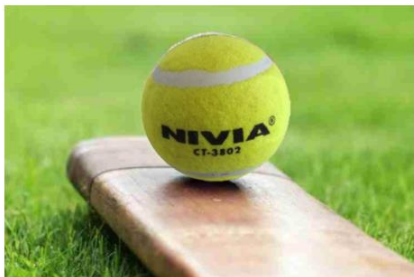
Sci-Hub

vikramBhavsar/Mo...

ERS-MODIFIED-13-...

### Tennisball cricket tournament

[home](#)



Tennis ball cricket tournament was held under the ST BRITTO CRICKET ACADEMY BANNER. Under 5 teams from within the school participated. First round was round robin league. Second round was a knock out semis, in which kolkatta knight riders def bangalore challenger and mumbai indians defeated chennai kings and entered the finals. Finals was held between them. 20 20 overs match in which mumbai indians defeated Bangalore challenger by 8 wickets. Finally the prizes were distributed under the hands of the principal and chairperson of the cricket academy

[View Date](#) [View Location](#) [Intrested? register here.](#)

## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

→ ECP to check Event List:

TEST CASE ID	INPUT	EXPECTED OUTPUT	Remark
#TC01	0	404 Error	PASS
#TC02	1	Event 1 detail will be displayed	PASS
#TC03	115	Event 115 details will be displayed	PASS
#TC04	230	Event 230 details will be displayed	PASS
#TC05	231	404 Error	PASS

→ BVA to check Event List

Min = 1

Min+ = 2

Nom = 115

Max = 230

Max- = 229

N = 1 variables

$(4n+1) = (4*1+1) = 5$  test cases

TEST CASE ID	INPUT	EXPECTED OUTPUT	Remark
#TC01	1	Event 1 detail will be	PASS

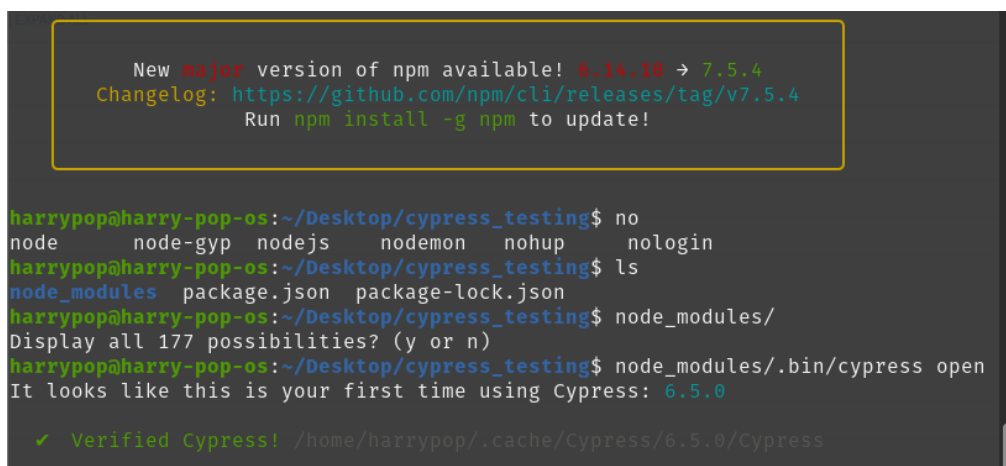
## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

		displayed	
#TC02	2	Event 2 detail will be displayed	PASS
#TC03	115	Event 115 details will be displayed	PASS
#TC04	230	Event 230 details will be displayed	PASS
#TC05	229	Event 229 details will be displayed	PASS

### BLACK BOX TESTING USING AUTOMATION TOOL

The tool used info, right **Cypress**.



```
New major version of npm available! 6.14.18 → 7.5.4
Changelog: https://github.com/npm/cli/releases/tag/v7.5.4
Run npm install -g npm to update!

harrypop@harry-pop-os:~/Desktop/cypress_testing$ no
node node-gyp nodejs nodemon nohup nologin
harrypop@harry-pop-os:~/Desktop/cypress_testing$ ls
node_modules package.json package-lock.json
harrypop@harry-pop-os:~/Desktop/cypress_testing$ node_modules/
Display all 177 possibilities? (y or n)
harrypop@harry-pop-os:~/Desktop/cypress_testing$ node_modules/.bin/cypress open
It looks like this is your first time using Cypress: 6.5.0

✓ Verified Cypress! /home/harrypop/.cache/Cypress/6.5.0/Cypress
```

#### Test Case 1:

The following test checks whether the **search query** provided in the website is reflected or not. After the search query is submitted the page is not reflected with the site. Hence the test case fails.



# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Chrome is being controlled by automated test software.

Tests: 10.33

http://localhost:8000/events/

1000 x 660 (87%)

cypress/integration/specs.js

Search Bar test

can fill the form

TEST BODY

```
1 visit http://localhost:8000/events -> 301...
2 get input[name="search-input"]
3 -type valorant
4 get #search-form
5 -submit
  (form sub) --submitting form--
  (page load) --page loaded--
  (new url) http://localhost:8000/events/?search-input=valorant
6 url
7 -assert expected http://localhost:8000/events/?search-input=valorant to equal http://localhost:8000/events/?search_query=valorant
```

AssertionError

Timed out retrying after 4000ms: expected 'http://localhost:8000/events/?search-input=valorant' to equal 'http://localhost:8000/events/?search\_query=valorant'

cypress/integration/search.js:7:18

Event Recommender System Login Categories

valorant Search

Browse Categories

Events For You

Share Export This week

Wellness, Health & Fitness  
Fashion And Beauty  
Food and Beverages  
Arts, crafts & exhibitions  
IT & Technology  
Live Shows  
Travel and Tourism  
Auto and Automotives  
Gaming  
Sports  
Dance

Trending New

1 International Conference on Nutrition & Health  
ASAR is bringing an International Conference on Nutrition & Health. This is going to be an interesting event as we ...  
View more

2 Workshop: Meditation and Mindfulness  
This workshop : Meditation and Mindfulness will focus on Meditation that helps the attendees build a relationship to a place ...  
View more

3 International Salon + Spa Expo  
"International Beauty & Cosmetic Industry Trade Fair" International Salon + Spa Expo (ISSE) is an international platform for Salon, Spa ...  
View more

4 BEASA INTERNATIONAL BEAUTY SALON TRADE SHOW  
The International Beauty Salon Trade Show is not just a major showcase ...  
View more

5 The Power of Food: Binding us to our Culture and One Another  
We are here with the National Conference on the topic : The ...  
View more

6 Indus Food  
Indus Food will gather most prominent global buyers from F&B industry to meet the selected Indian suppliers to negotiate their ...  
View more

```
(form sub) --submitting form--
(page load) --page loaded--
(new url) http://localhost:8000/events/?search-input=valorant
```

6 url

7 - assert expected http://localhost:8000/events/?search-input=valorant to equal http://localhost:8000/events/?search\_query=valorant

AssertionError

Timed out retrying after 4000ms: expected 'http://localhost:8000/events/?search-input=valorant' to equal 'http://localhost:8000/events/?search\_query=valorant'

cypress/integration/search.js:7:18

```
5 | cy.get('#search-form').submit();
6 |
> 7 | cy.url().should('eq', 'http://localhost:8000/events/?search_query=valorant')
    |                                     ^
8 |
9 |
10 |
```

View stack trace Print to console

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

## Following is the test code:

```
describe("Search Bar test",() =>{[
  it("can fill the form",() =>{
    cy.visit("http://localhost:8000/events");
    cy.get('input[name="search-input"]').type("valorant");
    cy.get('#search-form').submit();

    cy.url().should('eq','http://localhost:8000/events/?search_query=valorant');

  })
})
```

## Test Case 2:

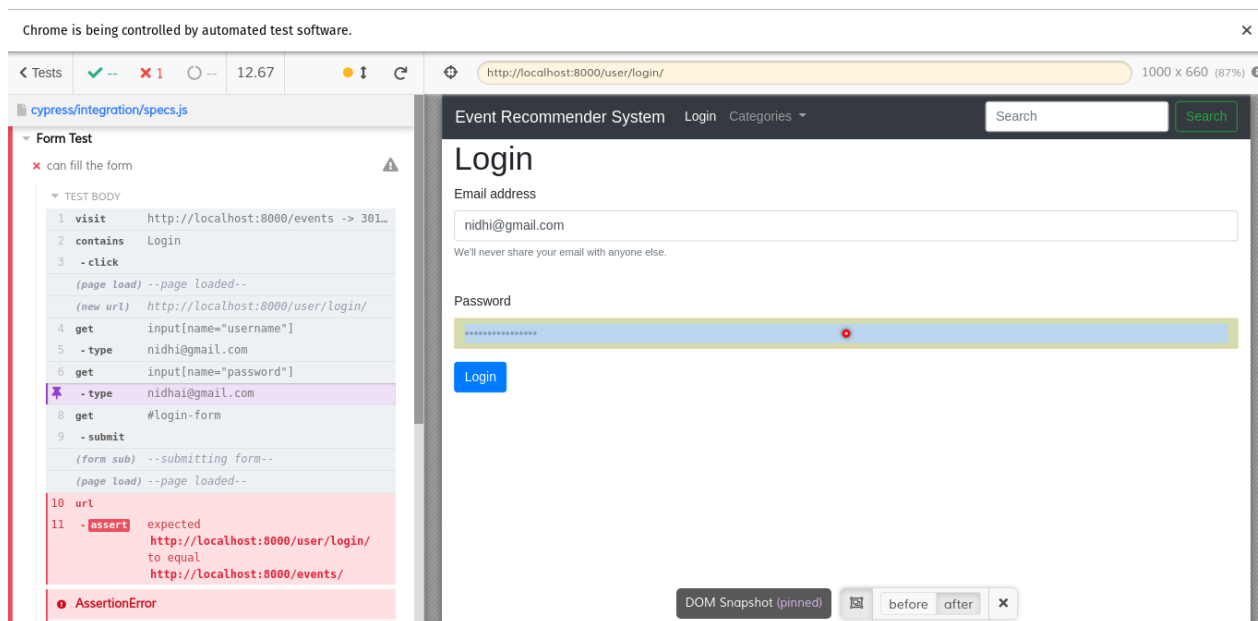
The screenshot displays the Cypress test runner interface on the left and a web application on the right. The test runner shows a test suite 'Form Test' with a failing test 'can fill the form'. The test body includes steps for visiting the URL, clicking the login button, and submitting the form. An assertion error is visible at the bottom of the test runner, indicating that the URL did not match the expected value.

The web application, titled 'Event Recommender System', shows a search bar and a list of events. The events listed are:

- 1 International Conference on Nutrition & Health
- 2 Workshop: Meditation and Mindfulness
- 3 International Salon + Spa Expo
- 4 BEASA INTERNATIONAL BEAUTY SALON TRADE SHOW
- 5 The Power of Food: Binding us to our Culture and One Another
- 6 Indus Food

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

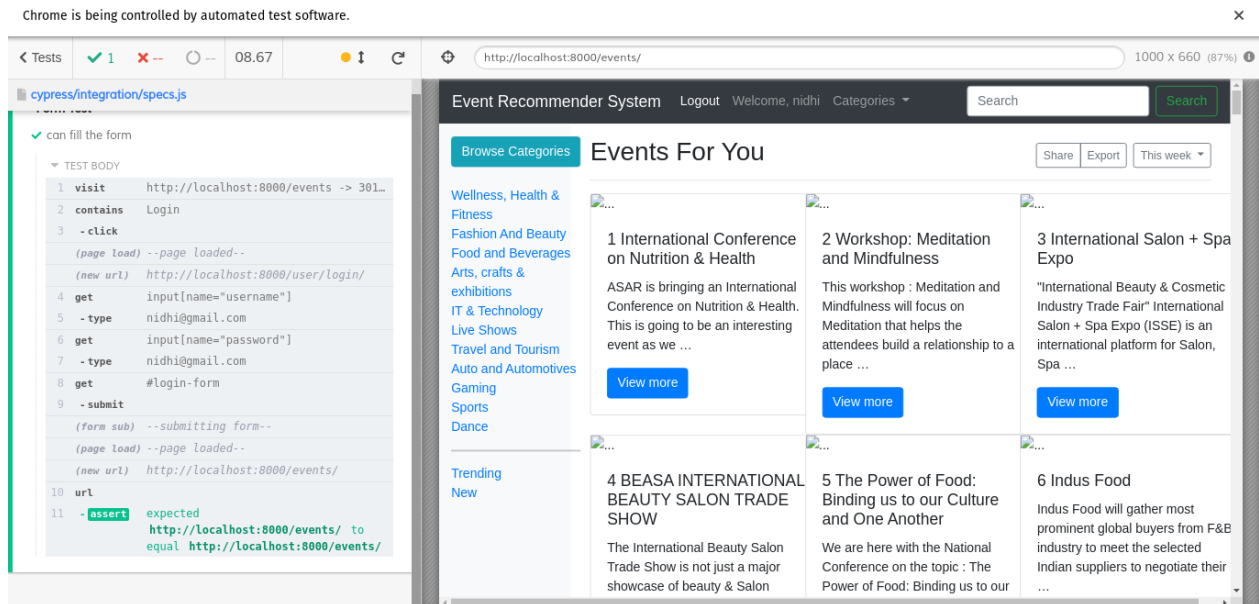


The following test case 2 is to test whether if correct credentials are provided then it's reflected back or not. The username and password entered here are [nid@gmail.com](mailto:nid@gmail.com) and [nidha@gmail.com](mailto:nidha@gmail.com). Now for the sake of learning the tool, we have purposely entered the wrong data but if it was the right data and since it does not match the expected output on the website, the test fails.

Now below is an example of test which passes

# K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)



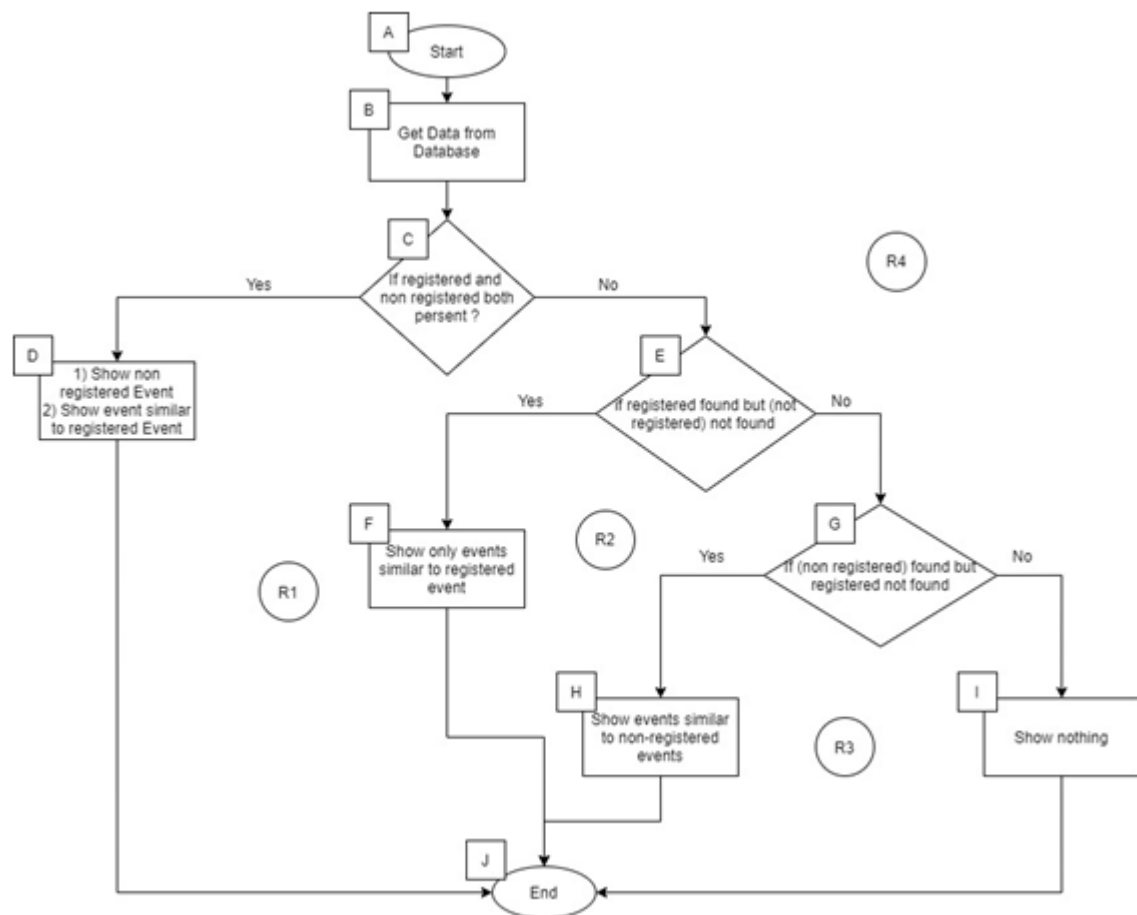
In this, when correct credentials are provided the website moves to the expected URL. Which the test recognizes and hence it shows the status that the test passed. Following is the code:

```
describe("Form Test",() =>{  
  it("can fill the form",() =>{  
    // cy.visit("http://localhost:8000/events");  
    // cy.contains('Login').click();  
  
    // cy.get('input[name="username"]').type("nidhi@gmail.com");  
    // cy.get('input[name="password"]').type("nidhai@gmail.com");  
    // cy.get('#login-form').submit();  
    // cy.url().should('eq','http://localhost:8000/events/');  
  
    cy.visit("http://localhost:8000/events");  
    cy.contains('Login').click();  
  
    cy.get('input[name="username"]').type("nidhi@gmail.com");  
    cy.get('input[name="password"]').type("nidhi@gmail.com");
```

```
cy.get('#login-form').submit();  
cy.url().should('eq', 'http://localhost:8000/events/');  
})  
}}
```

#### 4.10.2 White Box Testing

##### Recommendation based on past browsing patterns



→ **Cyclomatic Complexity:**

$$V(G) = E - N + 2 = 12 - 10 + 2 = 4$$

$$V(G) = \text{No. of Predicate Nodes} + 1 = 3 + 1 = 4$$

$$V(G) = \text{No. of Regions} = 4$$

→ **Identification Of Different Paths-**

## K. J. Somaiya College of Engineering, Mumbai-77

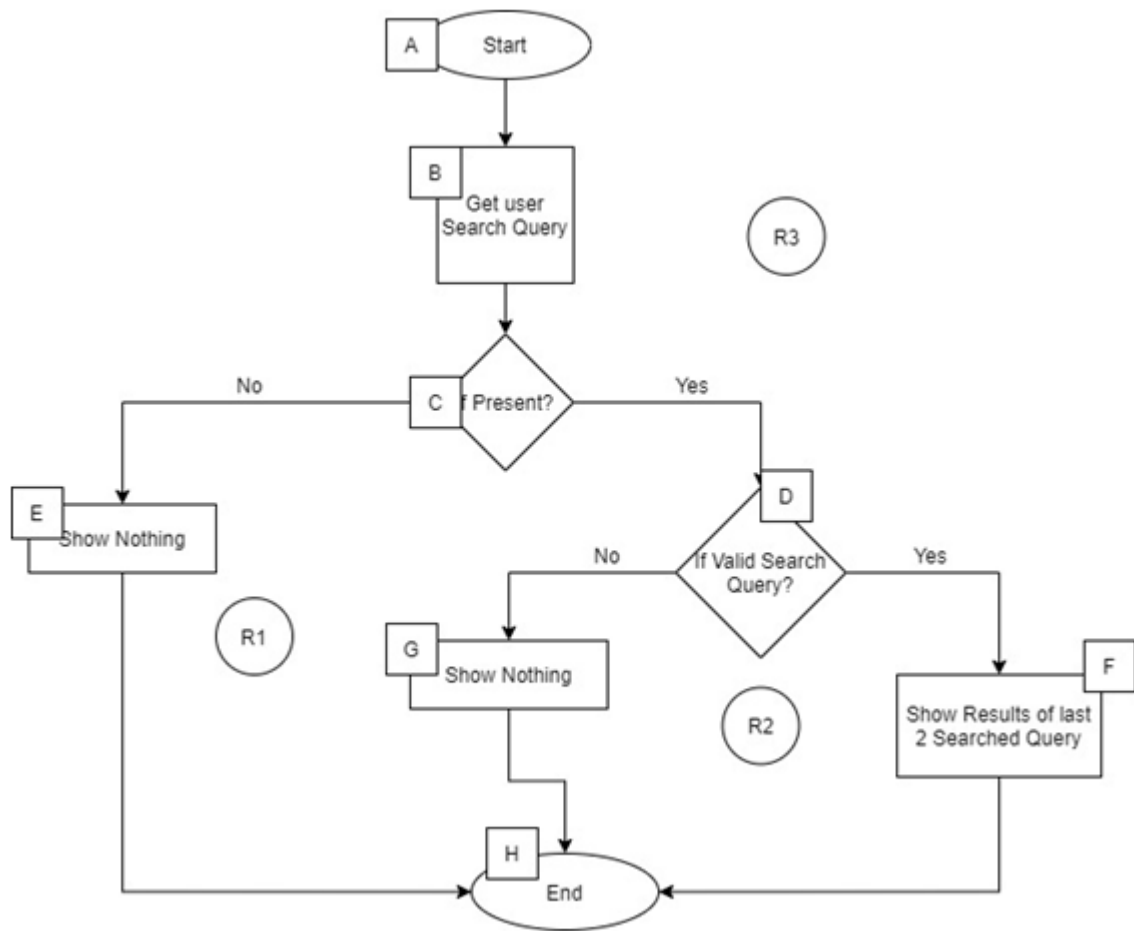
(Autonomous College Affiliated to University of Mumbai)

- A-B-C-D-J
- A-B-C-E-F-J
- A-B-C-E-G-H-J
- A-B-C-E-G-I-J

### Test Cases

ID	Input Value	Expected Output	Actual Output	Independent Path
1	List of events with values:  Regis = 1  regis = 0	Show non registered event  Show similar events based on registered event	Showed non registered event  Showed similar events tp registered event	A-B-C-D-J
2	List of events with only regis = 1	Show similar events based on registered events	Showed similar events based on registered events	A-B-C-E-F-J
3	List of events with only regis = 0	Show similar events based on non registered events	Showed similar events based on non registered events	A-B-C-E-G-H-J
4	No events	Show nothing	Showed nothing	A-B-C-E-G-I-J

**Recommendation based on past search query**



**Cyclomatic Complexity:**

$$V(G) = E - N + 2 = 9 - 8 + 2 = 3$$

$$V(G) = \text{No. of Predicate Nodes} + 1 = 2 + 1 = 3$$

$$V(G) = \text{No. of Regions} = 3$$

**Identification Of Different Paths-**

- A-B-C-E-H
- A-B-C-D-G-H
- A-B-C-D-F-H

# K. J. Somaiya College of Engineering, Mumbai-77

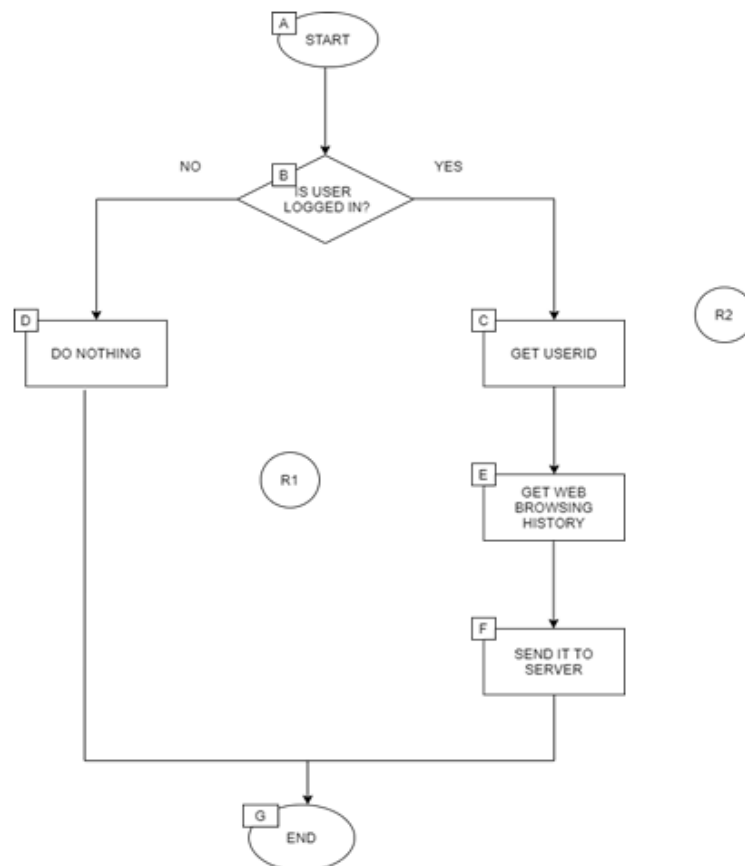
(Autonomous College Affiliated to University of Mumbai)

## Test Cases

ID	Input Value	Expected output	Actual output	Independent Path
1	No Data	Show nothing	Showed nothing	<b>A-B-C-E-H</b>
2	Invalid search query	Show nothing	Showed nothing	<b>A-B-C-D-G-H</b>
3	Valid Search query	Show results of the search query	Showed results of search query	<b>A-B-C-D-F-H</b>

## Recommendation based on user's browsing history

### Part 1





## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

### Cyclomatic Complexity:

$$V(G) = E - N + 2 = 7 - 7 + 2 = 2$$

$$V(G) = \text{No. of Predicate Nodes} + 1 = 1 + 1 = 2$$

$$V(G) = \text{No. of Regions} = 2$$

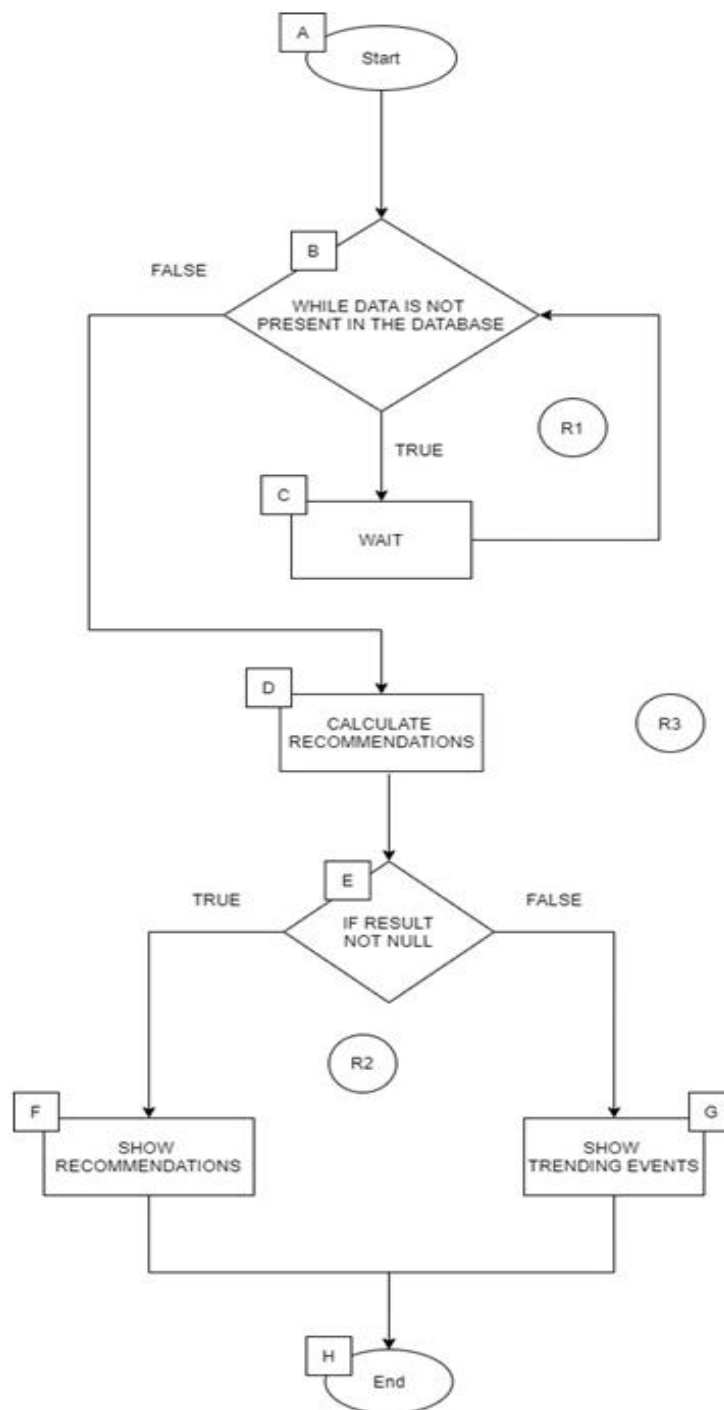
### Identification of Different Paths-

- A-B-D-G
- A-B-C-E-F-G

### Test Cases

ID	Input Value	Expected output	Actual output	Independent Path
1	Valid User credentials	Passing history data to server	Passed history data to server	<b>A-B-C-E-F-G</b>
2	Not logged in	Do Nothing	Did nothing	<b>A-B-D-G</b>

**Part 2:**



## K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

### Cyclomatic Complexity:

$$V(G) = E - N + 2 = 9 - 8 + 2 = 3$$

$$V(G) = \text{No. of Predicate Nodes} + 1 = 2 + 1 = 3$$

$$V(G) = \text{No. of Regions} = 3$$

### Identification of Different Paths-

- A-B-D-E-F-H
- A-B-D-E-G-H
- A-B-C-B-D-E-F-H
- A-B-C-B-D-E-G-H

### Test Cases -

ID	Input Values	Exptected output	Actual Output	Independent Path
1	Browsing before database is populated and no recommendation is calculated	Show nothing	Showed nothing	<b>A-B-C-B-D-E-G-H</b>
2	Browsing before database is populated and with calculated recommendation	Show nothing	Showed nothing	<b>A-B-C-B-D-E-F-H</b>
2	Browsing after data is populated but no recommendations calculated	Show nothing	Showed nothing	<b>A-B-D-E-F-H</b>

## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

3	Browsing after data is populated with calculated recommendations	Show recommendations	Showed recommendations	<b>A-B-D-E-G-H</b>
---	--	----------------------	------------------------	--------------------

## **Chapter 5 : Conclusion**

### **5.1 Conclusion**

- Amigo-Event Recommendation System is a Personalized Event Recommendation System which makes use of concepts such as Cosine Similarity, TF-IDF algorithm which is used to Mine the user's interest from their web history to recommend them the events of their interests, the web history is extracted with the help of a web browser extension and then cleaned and processed accordingly to mine a user's interest.
- By default, any user who logs in to the website is shown the list of Trending Events~ the events that are being registered the most by the current users of the System and also a list of events segregated categorically.
- A user is shown a list of Personalized Events based on his history and then if he/she is interested then a score is calculated which is further used for knowing up to which extent the user is interested in that event.
- From which the user is shown some unregistered events similar to those in which the user has explicitly shown interest.
- Apart from that, the user is also shown some events based on his search query on the website.
- An event organizer can register and post the events, which gets saved to our database from where they get into the list of events if the user is interested or if it belongs to any particular category.
- With the powerful location detection, up to the user's choice, can also view the events which are close to the user. (Within 50kms).

### **5.2 Scope for further work**

#### **5.2.1 Friends Concept**

Our Project can be tuned and merged to further include the Friend's Concept.

The friend's concept includes the logic of suggesting the event attended by your friends. If you and your friend have the same taste of events and if your cosine similarity of events

match along with the number of past events that you have attended together, then you are recommended the events attended by your friends and vice versa.

### **5.2.2 Intra-Somaiya Network**

We can extend our project to make it functional for the Intra-Somaiya network. The idea is to make this project limited to the Intra-somaiya network which includes all the colleges under the Somaiya domain, so that any tech-events, lectures, exhibitions, competitions can be reachable to students and most importantly, it would only recommend the events according to the student's interest.

### **5.2.3 Large scale Implementation**

If full-scale implementation istech events and our project is made more optimal and further performance-enhanced then, it can be used as a direct competitor to BookMyShow and MagicPin app, as it includes the main features of BookMyShow and MagicPin App along with suggesting only those events in which the user is interested in.

### **5.2.4 Adding events live to the website**

Currently, all the events are taken from the CSV file. To add events live and dynamically by having event organizers register on our website and then update their events is out of the scope of this project. The Scope of this project is mainly to provide recommendations to the user using multiple techniques. Hence in future this can be elaborated and extended by having a live page for Event Organizers to register themselves and upload events on our website.

## **5.3 References**

- [1] Saurabh Kumar, Mining User's Interest from Web History, at ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT),IEEE 2013.
- [2] Cai Zhi Liu , Yan-xiu Sheng , Zhi-qiang Wei And Yong-Quan Yang, Research of Text Classification Based on Improved TF-IDF Algorithm ,IEEE 2018.
- [3] Matching two documents based on TF-IDF and Cosine Similarity,William Scott,

## **K. J. Somaiya College of Engineering, Mumbai-77**

(Autonomous College Affiliated to University of Mumbai)

Available: <https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089>

[4] Cosine Similarity, Jiawei Han,

Available: <https://www.sciencedirect.com/topics/computer-science/cosine-similarity#:~:text=Cosine%20similarity%20measures%20the%20similarity,document%20similarity%20in%20text%20analysis>.

[5] Kim Falk, Practical recommender System, Manning Publications Co. 2019.

## **ACKNOWLEDGEMENT**

We have immense pleasure in the successful completion of this work titled.

“Amigos- Event Recommendation System” in this unprecedented time of Covid pandemic going around all over the world.

The unique virtual environment at KJSCE, which always supports academic activities, facilitated our work on this project, especially sincere thanks to our Principal, Vice Principal, and Head of the Computer Department.

We gratefully acknowledge the support, guidance, and personal encouragement extended for this successful project by our guide Prof. Rohini Nair Ma'am who responded promptly and enthusiastically to our doubts, requests, and suggestions, despite her congested schedules.

We are also appreciative to the Administrative Staff of KJSCE, who directly or indirectly have been helpful in some or another possible way.

We overwhelmingly want to thank our Parents, Classmates, and dear Friends, who encouraged us to extend our extensive reach. With their help and support, we have been capable to sufficiently complete this work well and on time.