The Recommender

An app made for students

Minor Project Report

Submitted by:
Group 4
Malvika **Jindal** 18103040
Parijat **Garg** 18103097
Jaspreet **Singh** 18103107
Atul **Kumar** 18103122

Under the supervision of:
Prof. Rajesh Bhatia
Department of Computer Science and Engineering
Punjab Engineering College
(Deemed to be University)
Chandigarh

DECLARATION

We hereby declare that the project work entitled "**THE RECOMMENDER**" is an authentic record of our own work, carried out at **Punjab Engineering College (Deemed to be University)**, as a requirement of the **Minor Project** for the award of degree of B.Tech (Computer Science and Engineering), under the guidance of **Prof. (Dr.) RAJESH BHATIA** (Department of Computer Science and Engineering), from August 2020 to December 2020.

~BY

Malvika Jindal 18103040

Parijat Garg 18103097

Jaspreet Singh 18103107

Atul Kumar 18103122

CERTIFICATE

This is to certify that the project entitled **THE RECOMMENDER** by Malvika Jindal, Parijat Garg, Jaspreet Singh and Atul Kumar is an authentic record of our work carried out under the supervision of Prof. (Dr.) RAJESH BHATIA (Dept. of Computer Science and Engineering), Punjab Engineering College (Deemed to be University), Chandigarh in fulfilment of the requirements as a part of Minor Project for the award of 04 credits in semester 5 of the degree of Bachelor of Technology in Computer Science and Engineering.

Certified that the above statement made by the students is correct to the best of their knowledge and belief.

Prof. (Dr.) RAJESH BHATIA
(Faculty Mentor)
Department of Computer Science and Engineering
Punjab Engineering College
(Deemed to be University)
Chandigarh

MALVIKA JINDAL
18103040

PARIJAT GARG
18103097

JASPREET SINGH
18103107

ATUL KUMAR
18103122

Dated: 17 December 2020

EXPLORE INNOVATE EXCEL

ACKNOWLEDGEMENT

We have taken a lot of deliberations in this venture. But it wouldn't have been possible without the help and backing of numerous people. We want to extend our true appreciation and thank them. We take this opportunity to express our profound gratitude and deep regards to our mentor Prof. (Dr.) RAJESH BHATIA for his exemplary guidance, monitoring and constant encouragement throughout the course of this project.

This project truly wouldn’t have been possible without his mentorship.

Overview

Often the jump between the workspace of a college student and his entertainment corner is huge.

This is an effort to integrate workspace along with useful features like maintaining a schedule, organizing events along with having powerful recommenders for songs and movies. Our work has attempted to analyse the mood of the user and recommend what he/she should do like read the news, join yoga class, watch videos and in parallel to that user can also do other things as he can manage his/her assignments, notes, schedule, see events and can organize an event.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The technological world is booming, with a humongous number of application websites being released each day, making anything possible with the reach of technology. Websites which help you design websites, portals where you can draw your code, etc. exist which take the quote ‘making your life easier’ to a whole new level.

Naturally, this big boom is followed in other sectors of our life, with technology making big impacts. One such example is the entertainment sector, with powerful recommenders predicting exactly what a person wants to see and hear at what time, with very little input.

As college students, that expanse in technology is a little counterproductive, as the better the entertainment side becomes, the more we lose out on education, keeping in mind that we can either devote time to entertainment, or studies. Not only does devoting time individually pose a huge problem, but switching between one track to the other requires a massive shift in focus and frame of mind, requiring significant amounts of time to ‘get into the right mood’ to work, or play.

Our project is a minor ode in that direction, hoping to resolve the issue many professionals (targeted especially towards college goers) face, balancing their work time and free time.

The idea behind this is simply to bring both sides of the coin on the same plane, that is, to bring the productivity and entertainment tools on a common platform, so that the transition between one to the other, and subsequently, vice versa is smoothened out and easier to perform

This is an effort to integrate one’s workspace with useful features like maintaining a schedule, organizing events, chatting, etc. along with having a good entertainment side including powerful recommenders for songs and movies.

Our work has attempted to analyse the mood of the user and recommend what he/she should do like read the news, join yoga class, watch videos and in parallel to that user can also do other things as he can manage his/her assignments, notes, schedule, see events and can organize an event.

CHAPTER 2

PROBLEM STATEMENT

Formulation

We have attempted to integrate as many features as possible keeping in mind the most important ones that could lead to a better usage and understanding of our effort.

Our web application contains the following features –

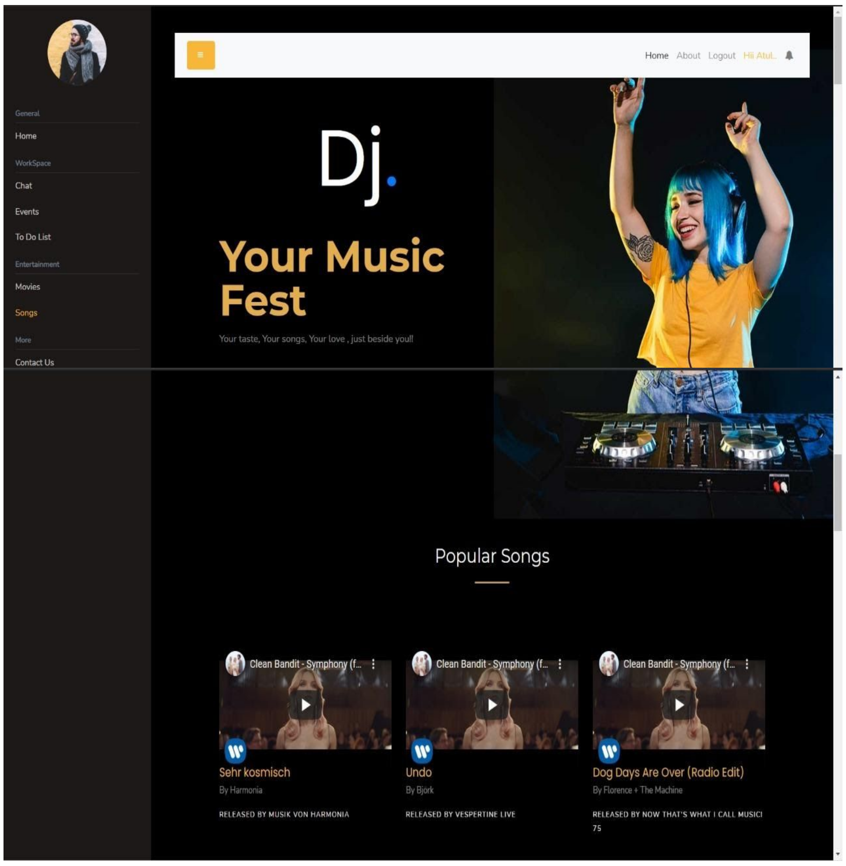
- An authentication of the users, to personalise their experience across the productivity platform (notes, chat rooms, etc.) and also on the entertainment platforms (recommenders, etc.)
- A chatting environment whereby the user can interact with his/her colleagues and continuously be updated about his work, or any other suitable things he would like to be notified upon. In addition to just one-to-one chatting, topic specific chat rooms can be opened with multiple users simultaneously conversing
- A powerful To-Do List system where any individual can schedule their tasks well and according to personal requirements, mark them with success, manipulate the completion time, etc.
- An events environment where one can organise events and notify the entire college, accessing all information through our website
- A powerful song recommender for recommending customised songs based on the user's mood and views.
- A powerful movie recommender for recommending movies depending on the different moods of the user

CHAPTER 3

PROJECT DEMONSTRATION
and
DESIGN (DIAGRAMS)

3.1 DEMONSTRATION

Music Recommendation


Home
About
Logout
Hi Atul..
General
Home
WorkSpace
Chat
Events
To Do List
Entertainment
Movies
Songs
More
Contact Us
Dj.
Your Music Fest
Your taste, Your songs, Your love , just beside you!!
Popular Songs
Clean Bandit - Symphony (f...
Sehr kosmisch
By Harmonia
RELEASED BY MUSIK VON HARMONIA
Clean Bandit - Symphony (f...
Undo
By Björk
RELEASED BY VESPERTINE LIVE
Clean Bandit - Symphony (f...
Dog Days Are Over (Radio Edit)
By Florence + The Machine
RELEASED BY NOW THAT'S WHAT I CALL MUSIC! 75

The predictions made by this recommender are based entirely upon mathematical calculations.

Our database consisted of about 2 million songs. However, we took 10,000 songs in our training set, to compensate for our limited hardware.

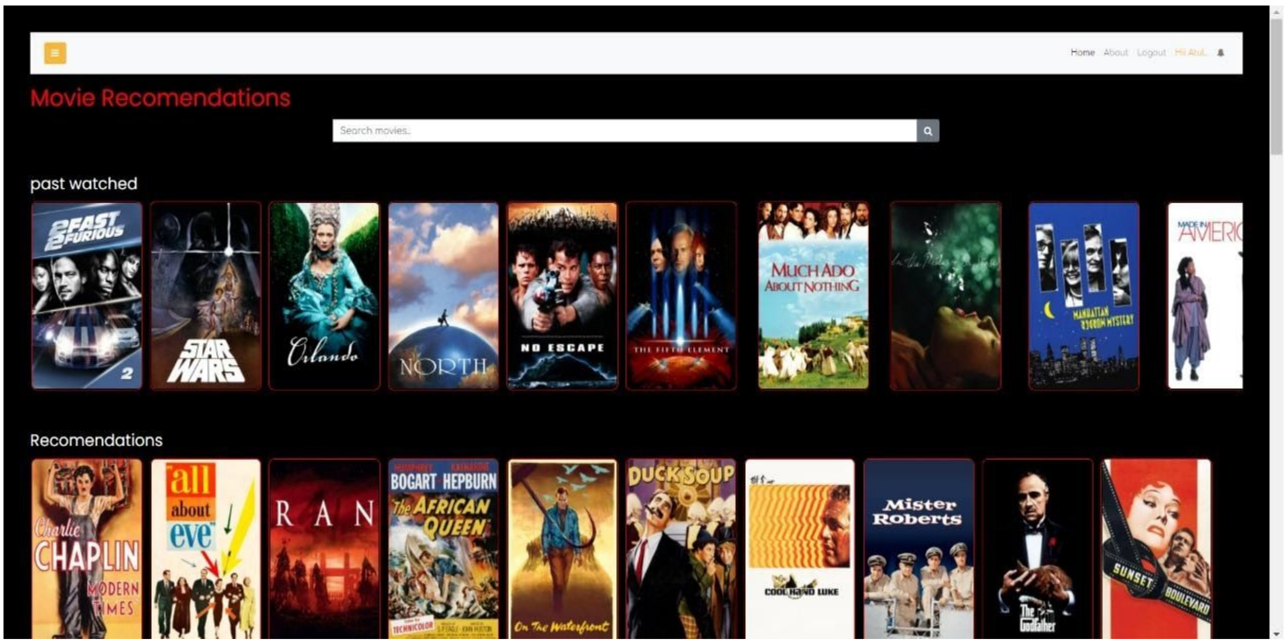
Initially, when the user logins for the first time, they are shown the top 10 most popular songs. The user can click on any of these songs and can get further recommendations which is on the basis of collaborative item filtering.

The most popular songs are obtained by calculating the percentage of likes vs dislikes, and dividing the listen_count by the sum of listen_count of all songs and multiplying by 100.

This item-item filtering approach involves defining a co-occurrence matrix based on songs a user ‘likes’.

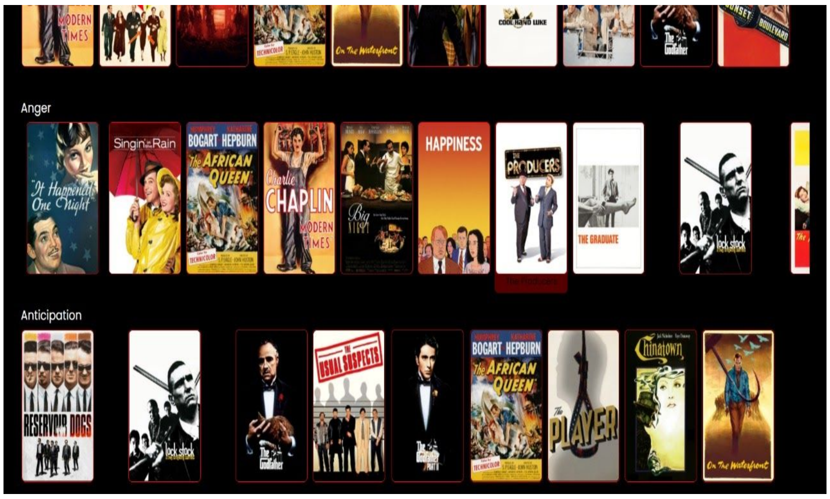
Co-occurrence is like similarity, the more two items occur together, the more they are probably related.

Movie Recommendation



Movie recommendation is a crucial part of the entertainment section.

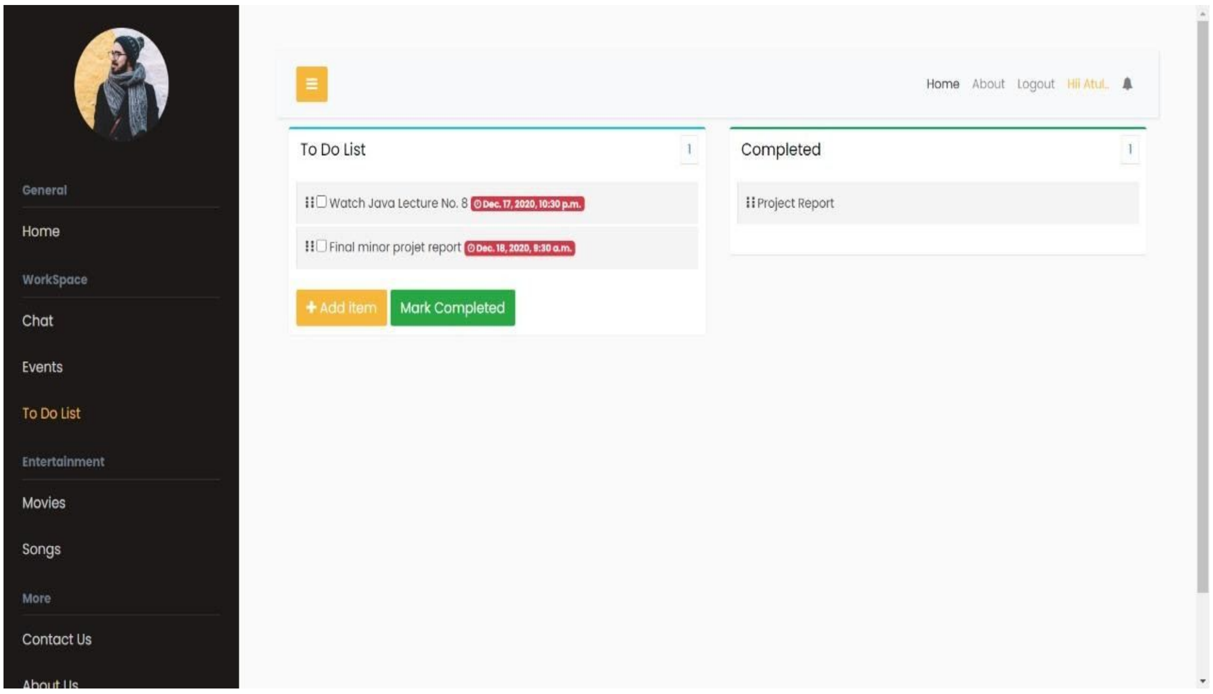
An ML model takes the user's past watched movies as an input, along with the mood/genre the user wants, to give recommendations to the user.



The Ml model is trained over the **MOVIE LENS** dataset having 100,000 ratings applied to 9,000 movies by 600 users. The **MOVIE LENS** dataset provides a key for movies which we can use to get details of movies like title, poster, release date etc. using IMDB's APIs.

To Do Lists

Tasks can be added , deleted, updated, be marked as completed, etc.



Watch Java Lecture No. 8 Dec. 17, 2020, 10:30 p.m.

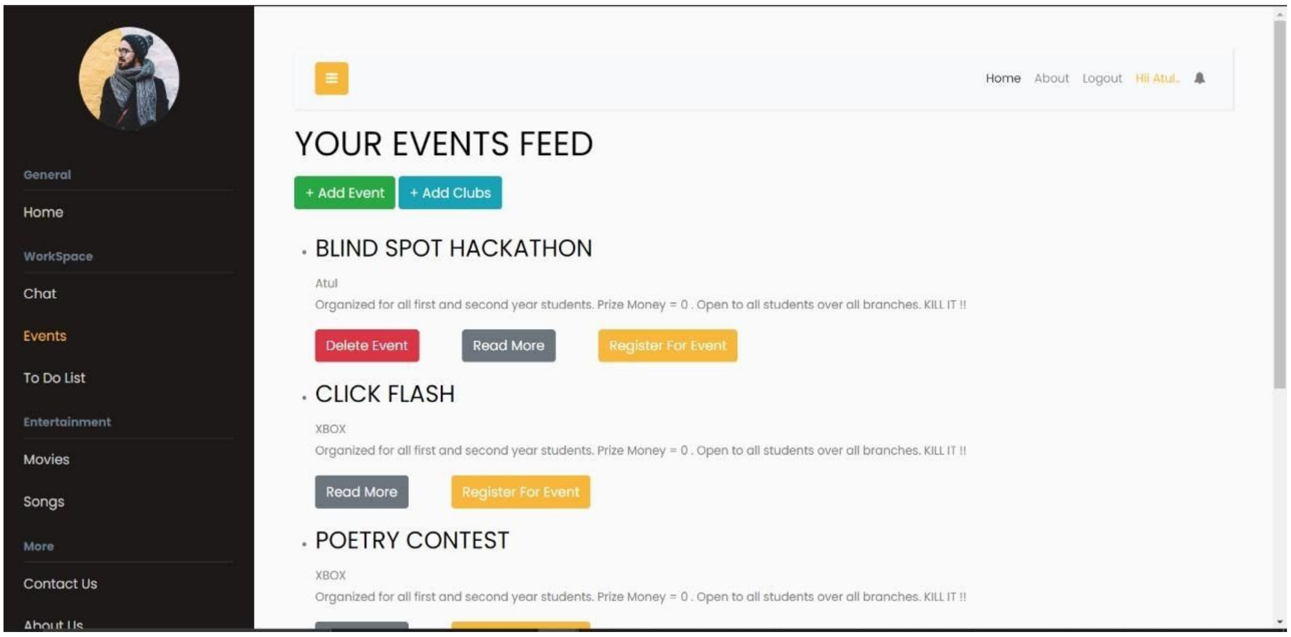
Final minor projet report Dec. 18, 2020, 9:30 a.m.

Project Report

All these functions are handled in django while the pagination is seamlessly handled by using AJAX.

This is done to improve the user experience as AJAX allows web pages to be updated asynchronously by exchanging data with a web server in javascript. This means that it is possible to update parts of a web page, without reloading the whole page, thereby reducing the load on our server.

Events



YOUR EVENTS FEED

- BLIND SPOT HACKATHON

  Atul

  Organized for all first and second year students. Prize Money = 0 . Open to all students over all branches. KILL IT !!

- CLICK FLASH

  XBOX

  Organized for all first and second year students. Prize Money = 0 . Open to all students over all branches. KILL IT !!

- POETRY CONTEST

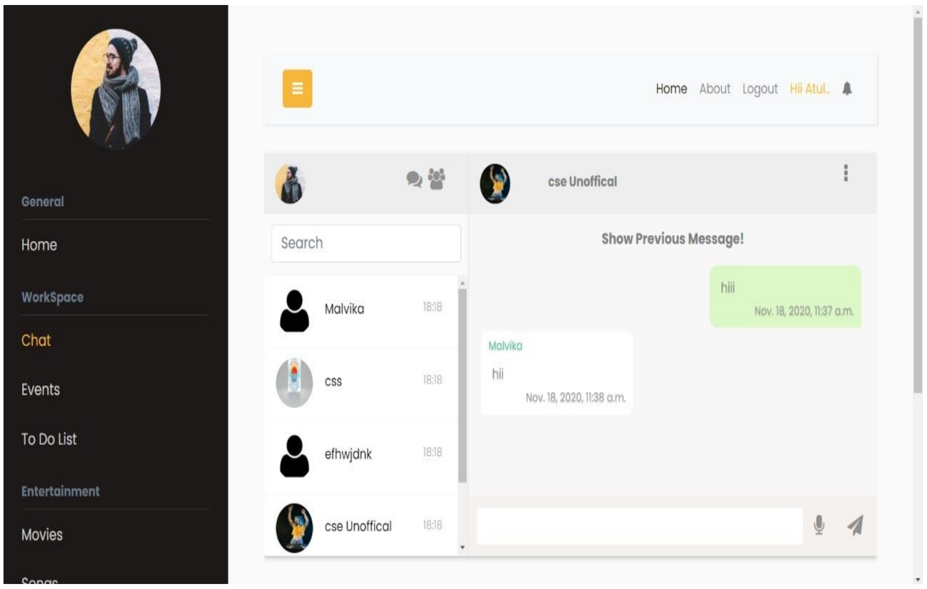
  XBOX

  Organized for all first and second year students. Prize Money = 0 . Open to all students over all branches. KILL IT !!

The users can organise events in their universities and notify everyone via our web application . Events consist of details of the CLUB organising it, the ADMIN of the event, DATE TIME VENUE, and a relevant DESCRIPTION. The user has the power to add, delete (only admin), register students, send updates regarding the events, etc.

Chatting

The users can create or join a chatting group and start sending and receiving messages. Also, the admins can remove a user from the group. This allows for chat rooms with dedicated discussion topics, on top of the regular one-to-one chatting systems.



Malvika 18:18

css 18:18

efhwjdnk 18:18

cse Unoffical 18:18

cse Unoffical

Show Previous Message!

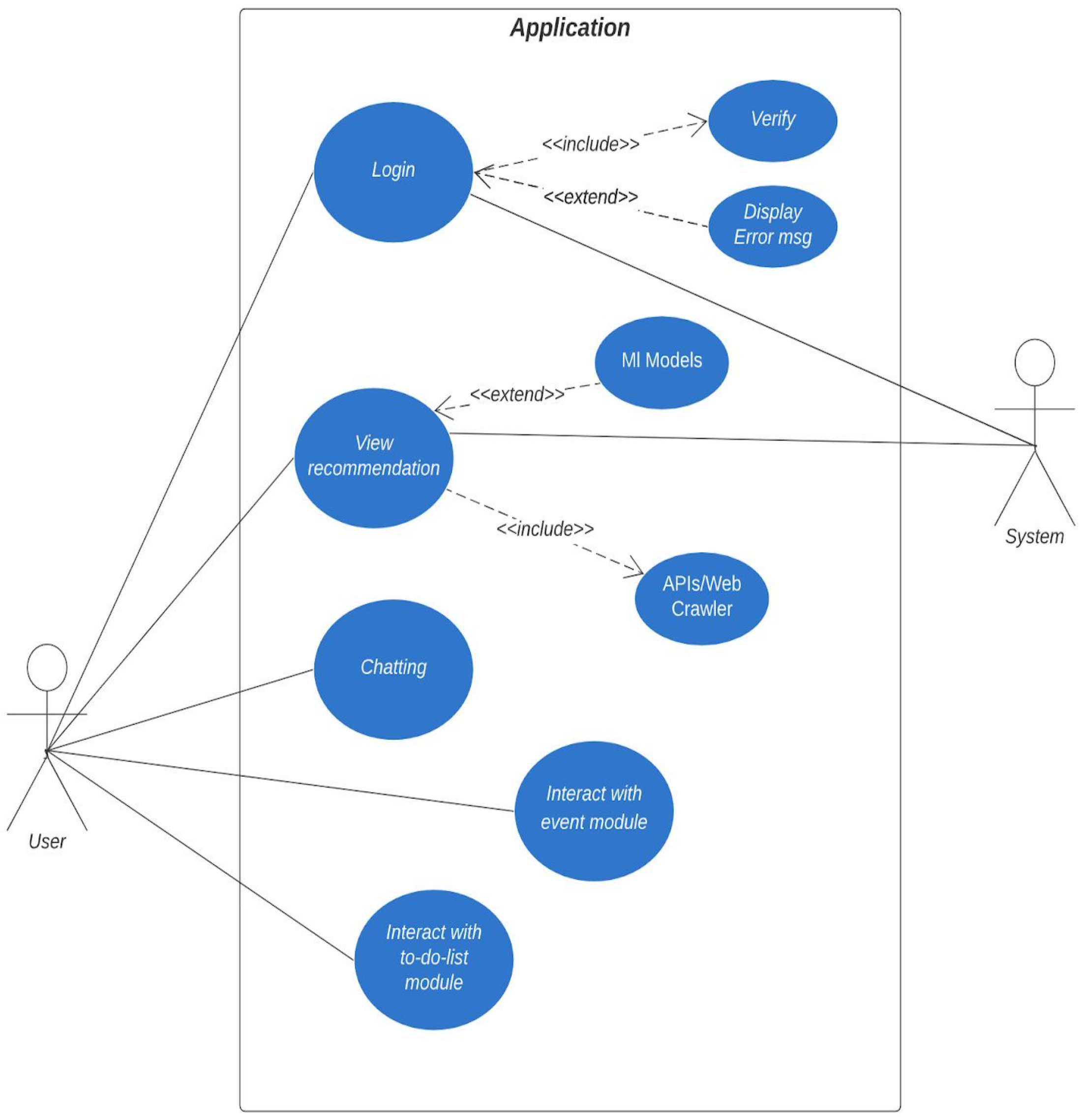
hiii
Nov. 18, 2020, 11:37 a.m.

Malvika
hii
Nov. 18, 2020, 11:38 a.m.

3.2 DESIGN

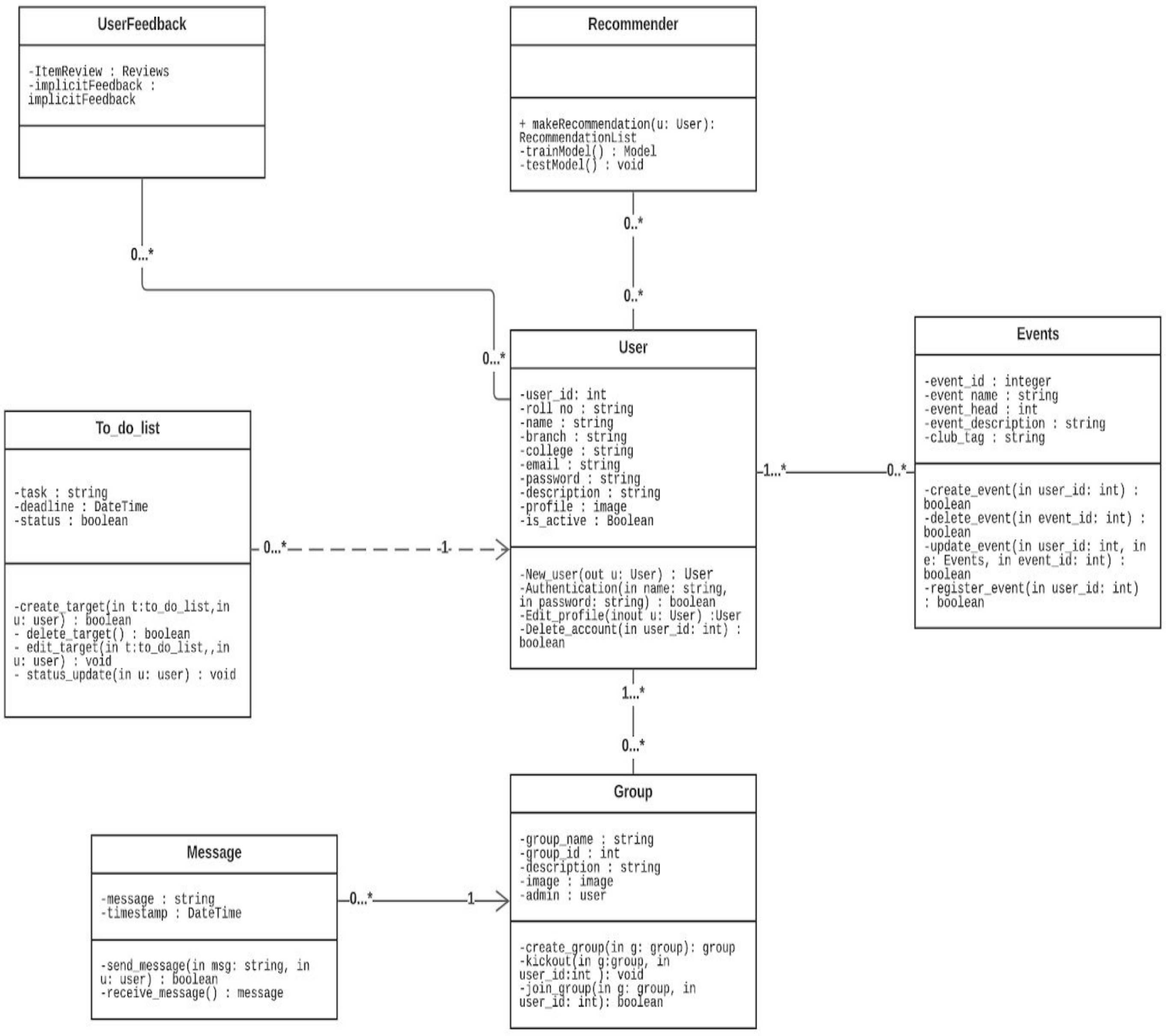
3.2.1 Use-Case Diagram

The use case diagram starts with the user applying for a sticker to be issued, followed by filling up the form. The application is then checked by the admin to make sure it is in order and the sticker is issued. To check the details of the vehicle, the sticker is scanned by the guard. The model recognizes images on entry of the vehicle in the parking.



3.2.2 Class Diagram

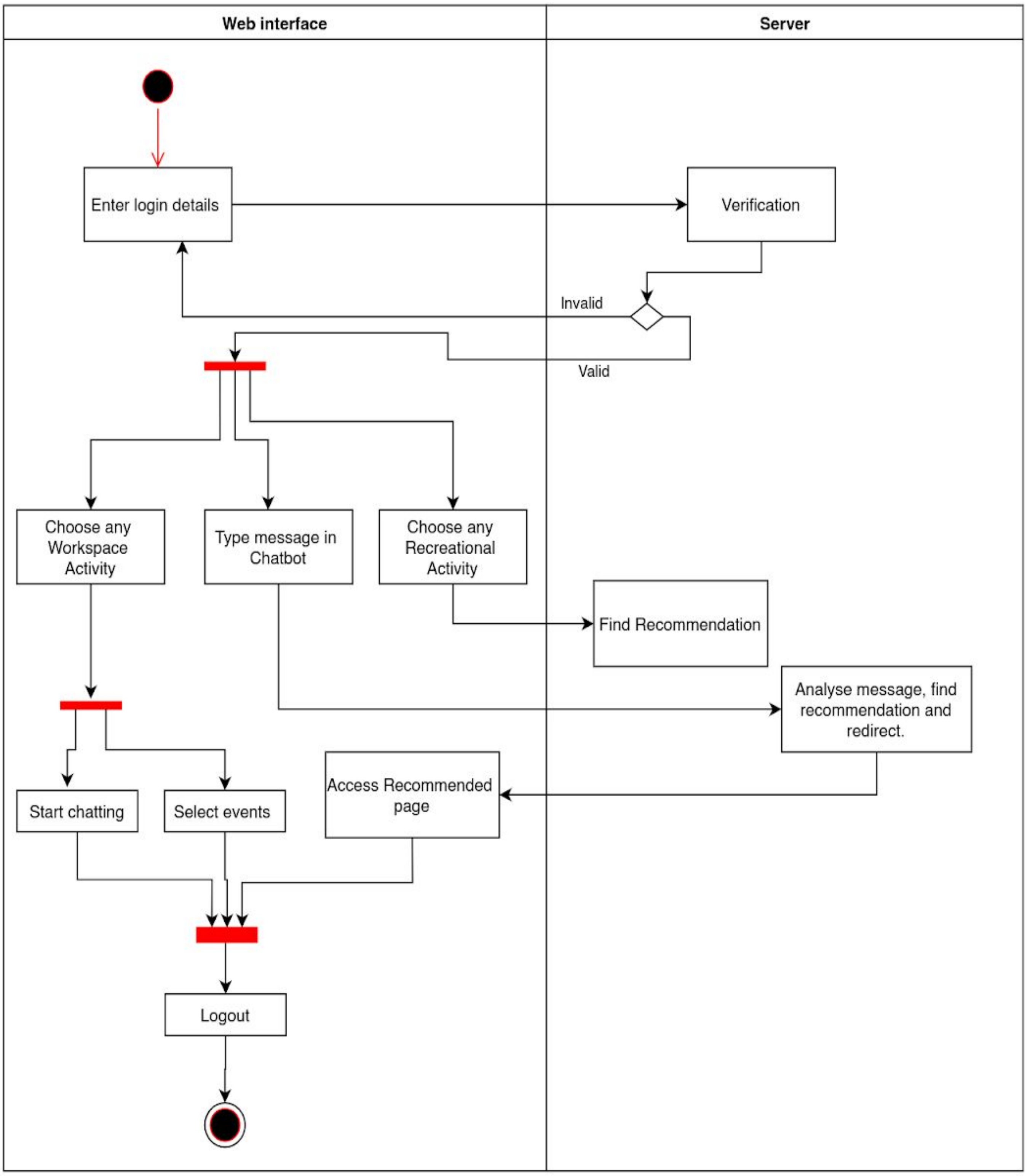
The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling.



UserFeedback
-ItemReview : Reviews
-implicitFeedback : implicitFeedback
Recommender
+ makeRecommendation(u: User): RecommendationList
-trainModel() : Model
-testModel() : void
0...*
0..*
0..*
0...*
User
-user_id: int
-roll no : string
-name : string
-branch : string
-college : string
-email : string
-password : string
-description : string
-profile : image
-is_active : Boolean
-New_user(out u: User) : User
-Authentication(in name: string, in password: string) : boolean
-Edit_profile(inout u: User) :User
-Delete_account(in user_id: int) : boolean
Events
-event_id : integer
-event name : string
-event_head : int
-event_description : string
-club_tag : string
-create_event(in user_id: int) : boolean
-delete_event(in event_id: int) : boolean
-update_event(in user_id: int, in e: Events, in event_id: int) : boolean
-register_event(in user_id: int) : boolean
1...*
0..*
To_do_list
-task : string
-deadline : DateTime
-status : boolean
-create_target(in t:to_do_list,in u: user) : boolean
- delete_target() : boolean
- edit_target(in t:to_do_list,,in u: user) : void
- status_update(in u: user) : void
0...*
1
1...*
0...*
Group
-group_name : string
-group_id : int
-description : string
-image : image
-admin : user
-create_group(in g: group): group
-kickout(in g:group, in user_id:int): void
-join_group(in g: group, in user_id: int): boolean
Message
-message : string
-timestamp : DateTime
-send_message(in msg: string, in u: user) : boolean
-receive_message() : message
0...*
1

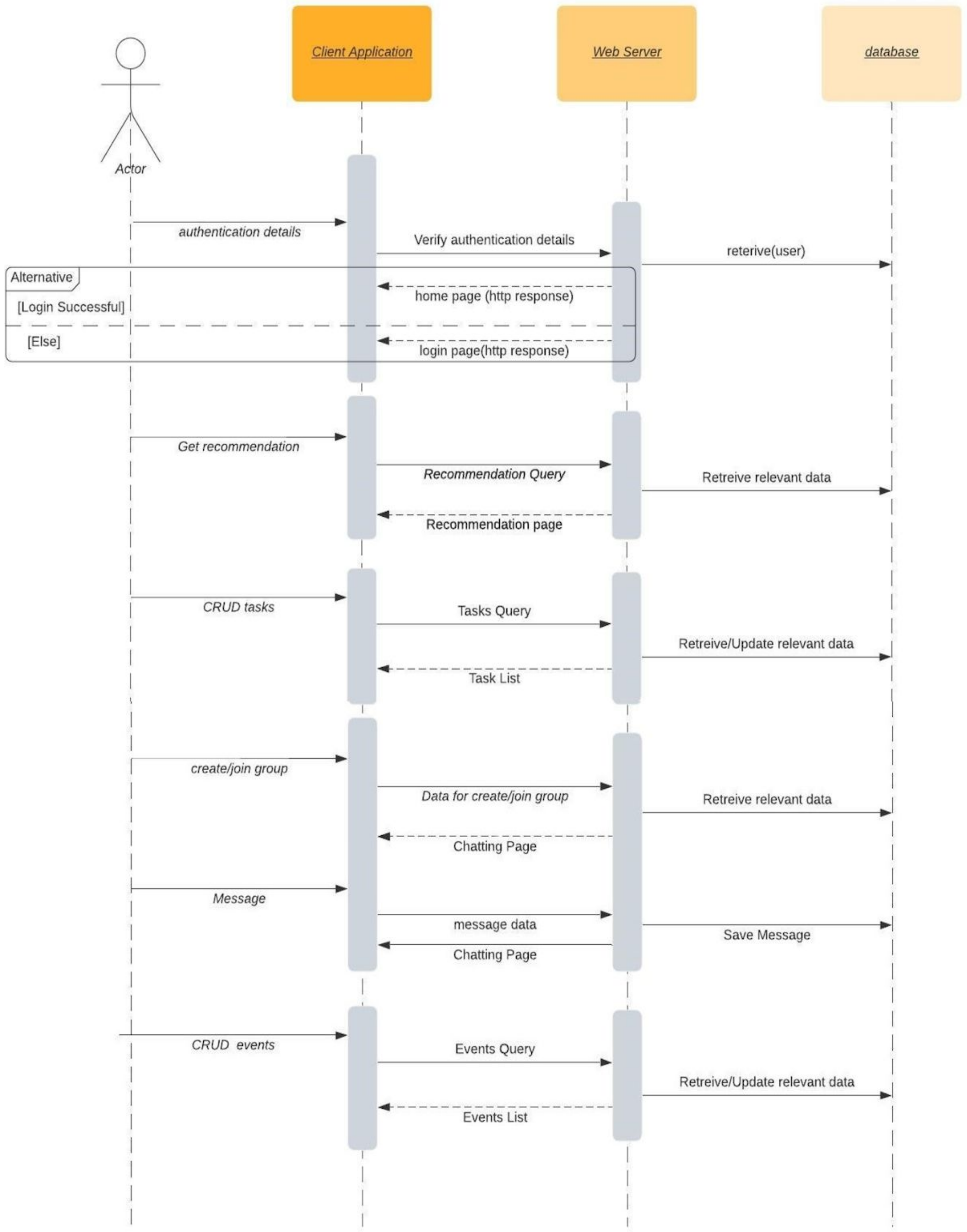
3.2.3 Activity Diagrams

Activity diagram is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flowchart that models the flow from one activity to another activity.



3.3.4 Sequence Diagrams

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.



CHAPTER 4

ALGORITHM AND APPROACH

SONG RECOMMENDER

The predictions made by this recommender are based entirely upon mathematical calculations.

Our database consisted of about 2 million songs. However, we took 10,000 songs in our training set, to compensate for our limited hardware.

Initially, when the user logins for the first time, they are shown the top 10 most popular songs. The user can click on any of these songs and can get further recommendations which is on the basis of collaborative item filtering.

The most popular songs are obtained by calculating the percentage of likes vs dislikes, and dividing the listen_count by the sum of listen_count of all songs and multiplying by 100.

This item-item filtering approach involves defining a co-occurrence matrix based on songs a user ‘likes’. Co-occurrence is like similarity, the more two items occur together, the more they are probably related.

MOVIE RECOMMENDER

Movie recommendation is one of the important parts of the entertainment section.

ML model takes the user's past watched movies as input to give recommendation to the user. Ml model is created using **SVD++** algorithm which is implemented using the **SKLEARN** python module.

SVD++ algorithm is used because iit has one of the best minimum **RMSE** (root mean square error), about 0.9002, and has a better hit ratio, which is done using the **Leaving One Out** method.

To train our ML model, we used the **Movie Lens** dataset, having about 100,000 ratings applied to 9,000 movies by 600 users, trained on a machine having 8GB ram.

Also, the Movie Lens dataset provides a key for movies which we can use to get details of movies like title, poster, release date etc. using IMDB APIs.

CHATTING ENVIRONMENT

The chatting is implemented using **Django** and **Django channels**. Since sending and receiving messages are frequent, it uses a web socket programming using django channels and **Redis** as cache for maintaining a queue of messages.

TO-DO LISTS

The to-do list is implemented using **Django** and **Ajax**. Since creating / deleting and updating the tasks are frequent requests hence it wasn’t necessary to load the whole page again and again , instead loading only that part was required which has been updated. That is why ajax was used. AJAX allows web pages to be updated asynchronously by exchanging data with a web server in javascript. This means that it is possible to update parts of a web page, without reloading the whole page.

EVENTS ENVIRONMENT

The users can organise events in their universities and notify everyone via our web application . Events consist of details about the CLUB organising it, ADMIN of event, DATE TIME VENUE and a relevant DESCRIPTION. Users have the power to add , delete(only admin) , register(students who are willing to take part in events being organised) ,for event(s).

CHAPTER 5

TECHNOLOGIES USED

5.1 PYTHON PROGRAMMING LANGUAGE

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. Complete implementation of models and generation of results is done in python.

5.2 HTML5

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look. HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance.

5.3 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. It has been used to define the front end styling of various extension components. Styling of all pages has been done with the help of CSS.

The Styling of the webpages in the project has been done using CSS.

5.4 JavaScript

JavaScript is a programming language that adds interactivity to the website. JavaScript is a full-fledged dynamic programming language that, when applied to an HTML document, can provide dynamic interactivity on websites. It is an interpreted programming language with object-oriented capabilities. JavaScript has been used for creating the page templates and for alert boxes.

5.5 AJAX

AJAX stands for Asynchronous JavaScript and XML. AJAX is a new technique for creating better, faster, and more interactive web applications with the help of XML, HTML, CSS, and Javascript. With AJAX, when you hit submit, JavaScript will make a request to the server, interpret the results, and update the current screen. In the purest sense, the user would never know that anything was even transmitted to the server.

5.6 Django

We have used Django as our framework for our web application. Django is a high-level Python Web **framework** that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel.

5.7 Django Channels

Chatting part was implemented by django channels. With WebSockets (via Django Channels) managing the communication between the client and the server, whenever a user is authenticated, an event will be broadcasted to every other connected user. Each user’s screen will change automatically, without them having to reload their browsers.

CHAPTER 6

ANALYSIS OF DATASETS

● DATASET FOR SONGS

Our Million Songs Dataset contains of two files(http://millionsongdataset.com/) :

- triplet_file
- 2) metadata_file

● DATASET FOR MOVIES

MovieLens Latest Datasets

These datasets will change over time, and are not appropriate for reporting research results. We will keep the download links stable for automated downloads. We will not archive or make available previously released versions.

100,000 ratings and 3,600 tag applications applied to 9,000 movies by 600 users. Last updated 9/2018.

README.html
ml-latest-small.zip (size: 1 MB)

CHAPTER 7

RESULTS

Looking over our progress, we have achieved all of our initially planned milestones, and, as with every project, we've had to re-evaluate and reconsider a lot of our methods and features

As of the date of the publishing of this document, we have managed to host this website on a private server with the following features:

- The ML based Movie Recommendations
- The ML based Song Recommendations
- The events organisation platform
- The to-do lists / planner
- A messaging platform with group chat functionality

If we take a snapshot of the current progress, our website is fully capable of supporting the college work flow requirements, and also host an entertainment system, achieving our initial objective of combining these two systems.

The website offers a seamless experience despite dealing in vast amounts of data, owing to the clever snips and cuts we've made through the website using AJAX and JavaScript (see above).

The only drawback turned out to be user feedback, which is crucial to any user based development.

While our initially planned milestones were achieved (with significant results), over time, we've discovered many features which could help make this application even more robust and friendly.

One of the 'futuristic' features we came up with was having a chatbot, that could detect what mood you are in, and automatically recommend certain songs and movies.

Owing to the lack of user feedback, we are also hoping to reiterate the website's development, once the pandemic is over.

While we have a fully functioning productivity and entertainment side, we realise that a lot of features could still be added to make this even more inclusive and seamless, eg. a game system, a proper streaming system, a way to give and grade quizzes and assignments, etc.

CHAPTER 8

CONCLUSION
and
FUTURE SCOPE

8.1 CONCLUSION

The Recommender is a website that aims to bridge the gap between productivity and entertainment. It achieves that goal by bringing both sides of the coin to a common plane, so that the transition between each other is smooth and easy.

A college student (the target demographic) can work on their college assignments and projects, organise events, hold discussions, chat with people, etc. i.e. everything a generic college student does, on our application (without having to switch between various applications).

On the other side, he can, within the same application, move on to watching movies and listening to songs as soon as he is done with his college work. This is supported by Machine Learning based recommender systems that can suggest movies and songs based on the user's preference.

With such a system, the user can seamlessly switch between doing work and getting entertained without having to continuously have huge shifts in focus.

8.2 FUTURE SCOPE

This application can be further expanded by including more parts of entertainment and workspace together so that a student can find all its requirements on one single platform.

We can additionally create a chatbot which can analyze our messages and hence mood thereby recommending what activity to do , for instance listen to songs, watch movies, et al .

We can distribute this application among various colleges so that students from different places can connect, conduct events together, can form groups and chat over various issues.

We can integrate a notification system including useful and relevant updates such as the admin of any event be notified about the number of registrations per events (maintaining it in a database) and updates received in the workspace.

The song recommender can be more powerful by adding a feature of likes/dislikes , comments et al.
Events can be further extended into a more interactive platform via including posts by different users.
And the list goes on…

In short, this can be turned into a powerful integration of workspace and entertainment.

CHAPTER 9

REFERENCES

Songs Dataset :

http://millionsongdataset.com/

Songs Recommender (1) :

https://towardsdatascience.com/how-to-build-a-simple-song-recommender-296fcbc8c85

Songs Recommender (2) :

https://towardsdatascience.com/create-music-recommendation-system-using-python-ce5401317159

Movie Lens Dataset :

README.html

ml-latest-small.zip

Beginner’s Guide to Creating the SVD Recommender System

https://towardsdatascience.com/beginners-guide-to-creating-an-svd-recommender-system-1fd7326d1f65

Matrix Factorization-based algorithms

https://surprise.readthedocs.io/en/stable/matrix_factorization.html