CSN-254

SOFTWARE ENGINEERING

Project: Recommendation System for CodeForces

Group no. 15

S.NO	NAME	ENROLLMENT NO.	EMAIL
1	BALNE NITEESHA	18114017	bniteesha@cs.iitr.ac.in
2	DIKSHA	18114019	diksha@cs.iitr.ac.in 9929155847
3	KARAN SINGH	18114035	ksingh1@cs.iitr.ac.in
4	KHUSHI	18114040	khushi@cs.iitr.ac.in
5	MANASVI	18114046	mpatidar@cs.iitr.ac.in

Recommendation System For CodeForces

Requirement and Specification Document

Date: 20th Feb, 2020

Version:1.1

1.1.1 Project Abstract

Competitive Coding is an overwhelming experience that every college undergrad has to go through in order to be prepared for their job interviews. The scope of this project will be to make this experience less painful by generating just the perfect list of problems. It would involve taking into account general trends followed on CodeForces and also the most efficient methods to analyze them and obtain results from them. The challenge will be doing this programmatically and maintaining the human touch. This is where Machine Learning would come in handy. A ML based algorithm that would consider the problems being solved and the problems solved by others to generate recommendations for future problems is the expected result. Also the infrastructure to present all this would be developed. A server that keeps the logic handy and accessible but also safe from hackers. A front end that's easy to use but informative enough.

1.1.2 Conversation with Customer and Developer

College undergrads have to go through hard times in order to be prepared for their job interviews. Hence a mobile application is needed

which is handy and easy to operate with a nice GUI that makes students' jobs easy.

- It should be complementary with codeforces.
- Mobile application should be like it takes users data associated with codeforces.
- Ask for a login.
- Recommend user data according to his interests.
- Allow users to get recommendations according to other users' interests who have something common in their profiles.
- Ask for ratings of problems to improve further search results.

1.1.3 Competitive Landscape

It is tiring to search "what to solve next?" in CodeForces site, our app helps to solve their problem by recommending users in a hybrid way and there also exists a website (https://acm-recommender.herokuapp.com/#/signup) to recommend problems for CodeForces users is developed by mohabamr. This can be said as our current competitor.

Analysis:

ASSESSMENT	OUR PRODUCT	THEIR PRODUCT
UNIQUE CAPABILITIES	Dynamic hybrid of user-based and item-based collaborative filtering.	Feedback
BEST CAPABILITIES	Handy app with best featured GUI facility.	Feedback
SAME CAPABILITIES	Sign-in	Sign-in
POOR	-	GUI facilities

CAPABILITIES		

1.1.4 System Requirements

Answering the questions:

- What does the system need to do ?
- What are the functions, what business rules and logic must be implemented?
- What information does the system need to contain?
- What are the constraints under which the system must operate?
- What are major steps in registration
  - they include: we need to determine: how should the process work
  - what should the steps be?
- What information is required to make the process work effectively?
- What information is required to support each process step?

Trying to solve above questions and interacting with customers, we get that **This application is made to complement CodeForces and eliminate all its major weak points, namely bare minimal GUI and lack of user specific recommendations.**

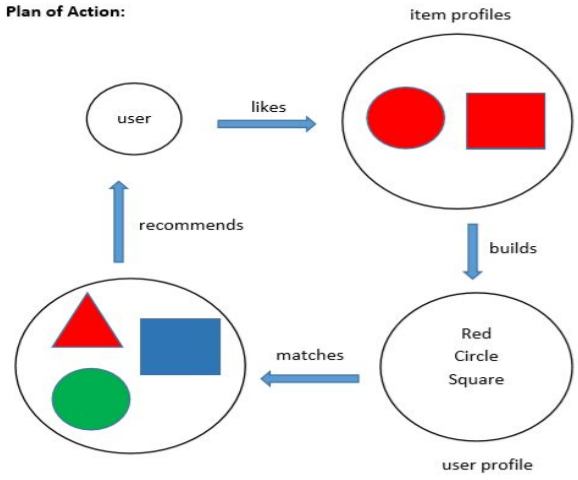
So we categorise our requirements based on following three basis:

- Functional requirements
- Technical requirements
- Performance requirements

Functional requirements

Recommendation using content or item based approach

Content-Based recommender systems assume that the users' preferences do not change significantly over time which is why these systems recommend items to target users similar to previous items rated highly or labeled as relevant by the target users. For example, in case of movies, these systems would recommend movies with the same actors, directors and so on. As for websites, blogs, accessing scientific papers and news these systems would recommend articles with similar content.



Content based recommender approaches will be of great use for devising part of our recommender model. To recommend challenges to the user, initially, the profile of the challenges and profile of the user need to be built. A challenge profile would include the category of the challenge, total accepted solutions of that challenge, total number of users attempted that challenge and other statistical information. A user profile would include the type of category challenges the user has been solving challenges from, the level of the challenges the user has been solving problems of and so on. Of course, the user has to have a significant number of submissions in order for a user profile to be built for him/her. So, content-based recommender will have difficulties

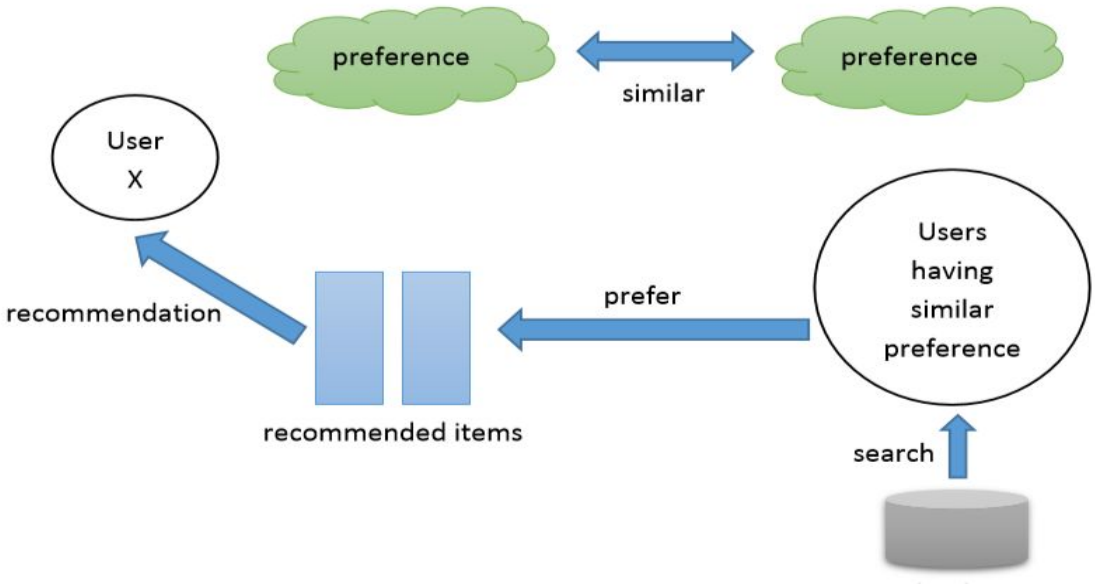
recommending challenges for new users which needs to be tackled using other methods.

Recommendation using collaborative filtering or simply users based approach

A collaborative filtering recommender system recommends items to target users based on the ratings or behavior of other users whose ratings or behaviors are similar to that of the target users. The motivation behind this recommender system is that people usually get the best recommendation from people who have similar tastes. A typical collaborative filtering recommender system will have the following

Workflow:
1) A user rates items such as books, movies, restaurants, hotels and so on. The system captures the ratings of the user. This rating is considered as an approximate representation of the user's interest in the corresponding domain.
2) The system matches this user's ratings against other users' and finds the people with most "similar" tastes.
3) The system then recommends items that the similar users have rated highly but have not yet been rated by this user.

Plan of Action:



Collaborative filtering is a good approach to suggest programming problems in an automated online judge. Majority of the users using automated online judges solve programming problems with basically the same intention, i.e, performing well in programming contests. So, for similar users who have similar submission history the system can make use of the fact that they have similar growth rate, preference towards choosing categories and problems and so on. However, this system might not perform well for users who have solved fewer problems. This is because every individual is different. Some of the users might solve problems with steady increase in difficulty levels while some might solve problems of varying difficulty haphazardly. It might happen that based on a user's similar neighbors the system is suggesting some programming problems of a particular category which the user does not have maturity and understanding to solve. In conclusion, the system should be able to recommend problems for the steady, gradual growth of the users and should not assume that the user has mastered certain topics without proper statistics.

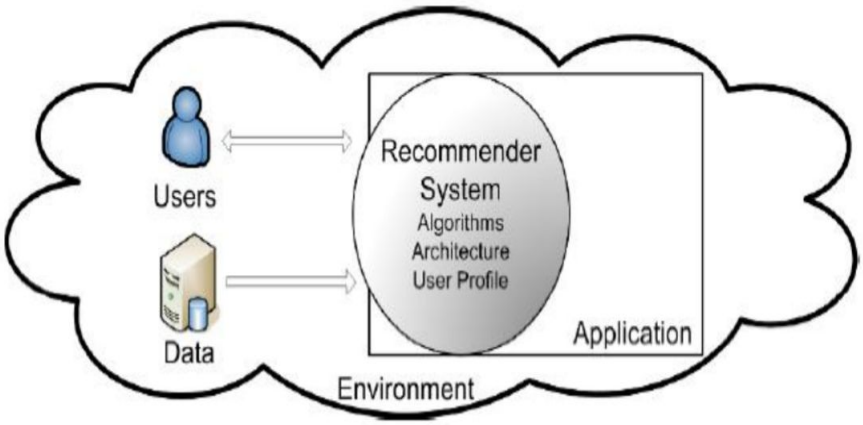
Technical requirements

The technical requirements and environment of the recommender can be studied through three dimensions:

Users: who are the users, what are their goals?

Data: what are the characteristics of the data on which recommendations are based?

Application: what is the overall application the recommender is part of?



USER MODEL

Fully understanding the user is a fundamental component to the success of any recommender system. In this section we discuss the properties on the user's side which may have an impact on the design and choices we as the designers of the recommender faced.

- Identifying those properties boils down to understanding who the users are, and what expectations and goals lie behind the users motivations to use the system the recommender supports.
- Understanding who the users are revolves around three main concerns: understanding their key identifying characteristics, their skills levels and their prior experience with similar systems.
- The most important concern in our case is: Identifying users' characteristics:

- Gathering a picture of the different user groups through both demographic information such as age and gender, job area, nationalities, spoken languages, and deep qualitative insights from user research are an important jumping off point in the development of recommender user models.
- Understanding these factors allows the development team to start to build a relationship with the users and get an appreciation of their needs.

DATA MODEL

The last point the designer should study carefully are the characteristics of the items the system will exploit and manipulate. Indeed, item descriptions generally preexist before the recommender system, and the designer of the recommender system has little possibility to influence or change them. The designer of the recommender system needs to identify the main characteristics of the data that may influence the design and the results of the recommender system.

APPLICATION

A mobile based application with following pages and features

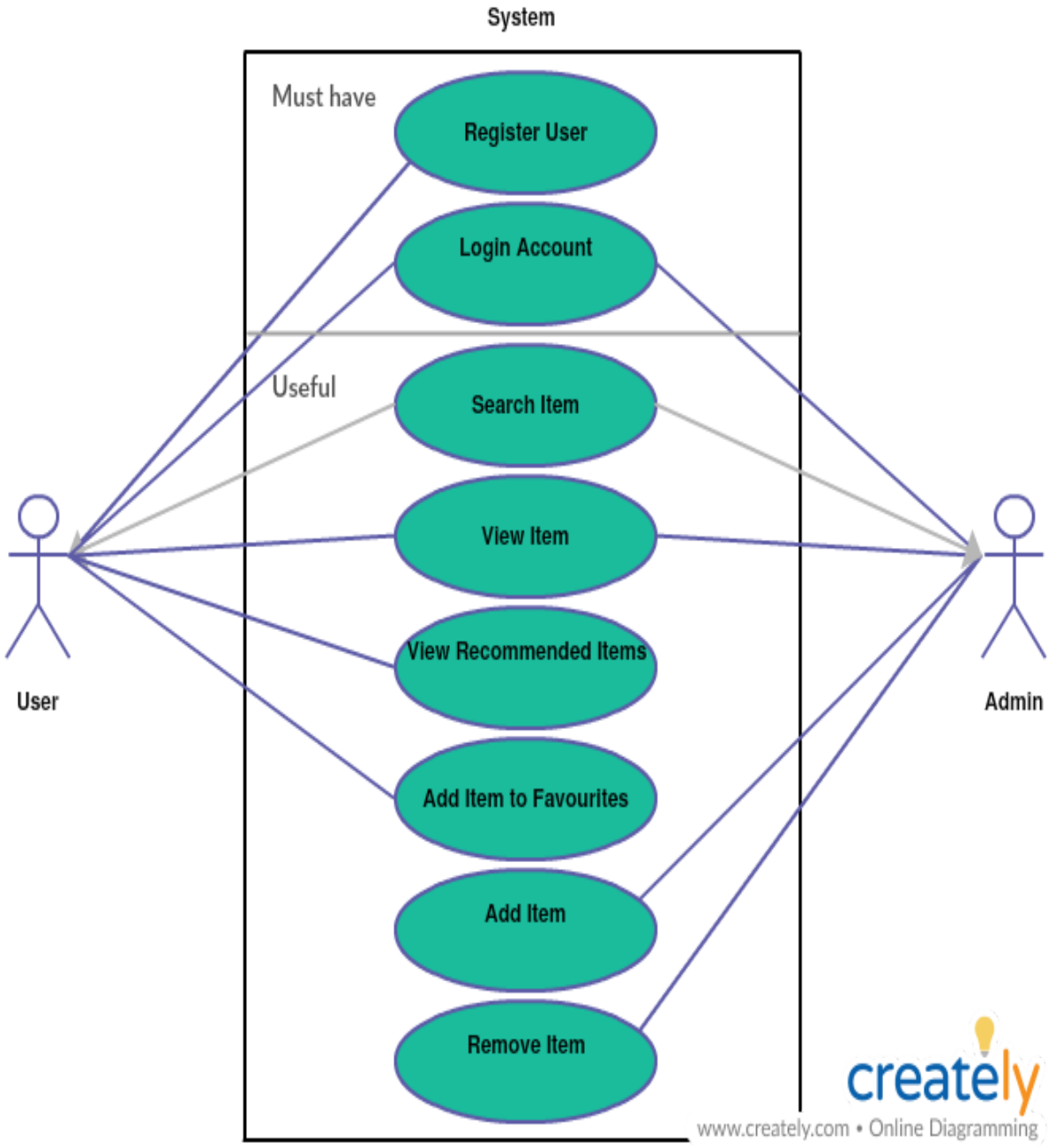
GUI:-

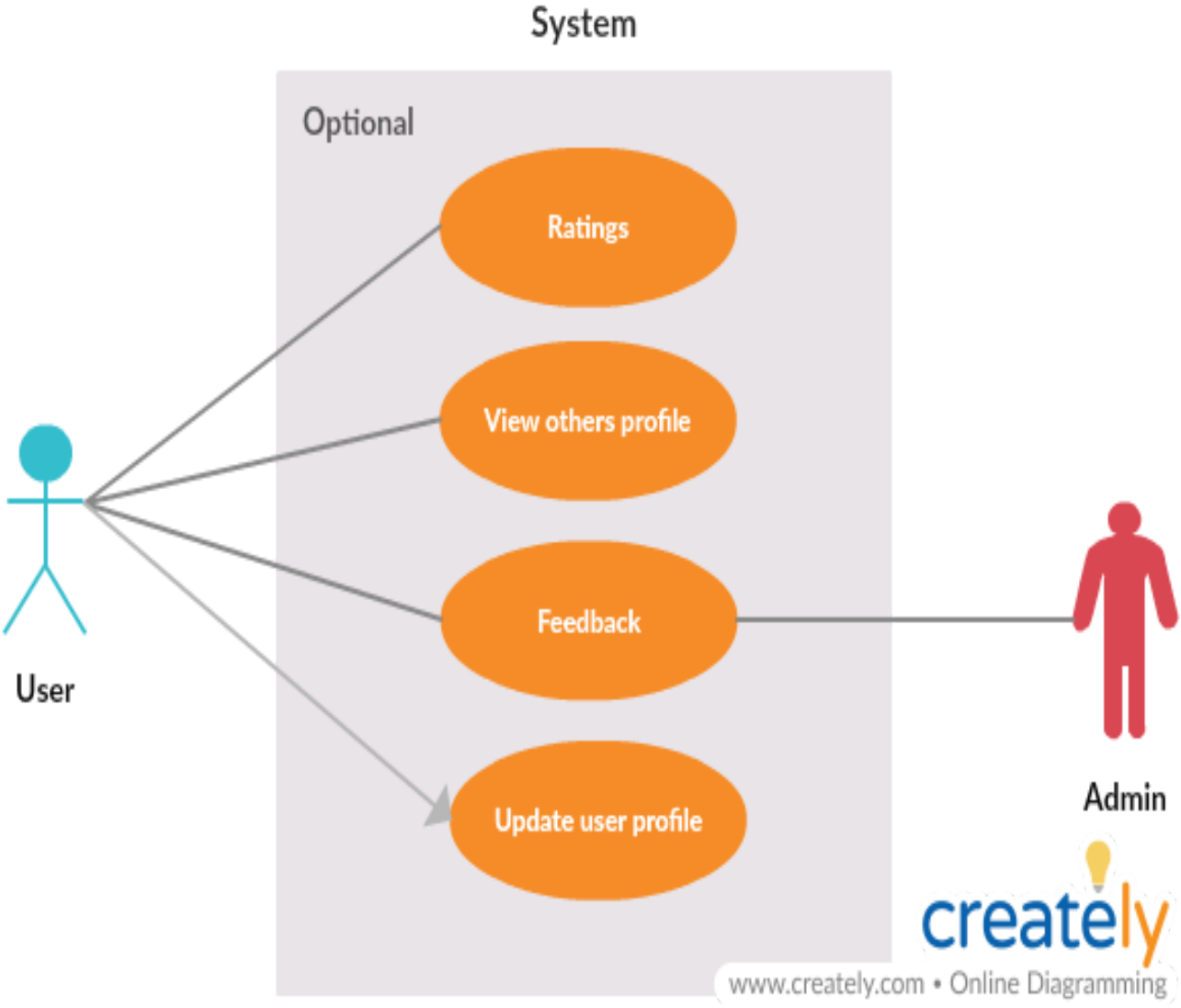
The GUI can be created with the help of Flutter, which gives great amount of packages to work with and also generates native code for both iOS and Android Applications. Because there are no Android specific features that this app will use and hence the iOS code made by Flutter SDK shall also be fully functional, and the app can be released targeting a wide area of customers.

All **optional and must have** requirements for proper GUI will be fulfilled
Within the given tabs.

- **User Sign-In.**
  This will be a one-time local sign-in process to gather information about the user and his friend list.
- **Home Page.**
  This will be the page the user is greeted with every time he/she opens the app once signed-in. It will have options to open the different suggestion list and an option to update the friend list of users.
- **List Pages.**
  These pages will provide a list of various problems of CodeForces generated by algorithms mentioned later.
- **Over the App Browser Tab.**
  On touching any of the above mentioned problems of CodeForces the app will open a browser tab to see the CodeForces page for the problem.
- **Choose category for providing recommendations**
  It may be based on user profile or
  Based on other users interests.

Needless to say that all these pages will be made according to Material Design, a major perk of using Flutter.


System
Must have
Register User
Login Account
Useful
Search Item
View Item
View Recommended Items
Add Item to Favourites
Add Item
Remove Item
User
Admin
creately
www.creately.com • Online Diagramming



Performance requirements:

1. Each click should respond in no more than 3 seconds
2. Longer search functions should not take more than 10 seconds
3. Reasonable uptime is required
4. Quick recovery if the server is ever down.
5. The application should be small i.e, not more than 10-15MB.
6. It can be a multi-user application.

1.1.5 Checklist:

In the starting of requirements section, we were having several questions that are:

- What does the system need to do ?
- What are the functions, what business rules and logic must be implemented?
- What information does the system need to contain?
- What are the constraints under which the system must operate?
- What are major steps in registration
  - they include: we need to determine: how should the process work
  - what should the steps be?
- What information is required to make the process work effectively?
- What information is required to support each process step?

Now by the end of SRS document,
We are able to answer all such questions. Hence, we are ready to move forward to the design part.

Github

App code:https://github.com/KaranKS/Project_CSN_254

Final notes

As promised, we were able to complete the most important deliverable, which was to create an app Recommendation System for CodeForces. Despite the presence of the Covid-19 pandemic and being stuck in lockdown and quarantine period from the past two months, our team was successfully able to develop the app within the time and hope so the customer will be satisfied with our work. We learned a good lot

about software development and software development life cycle (SDLC) from this project and are grateful to our Professor Dr. Sandeep Kumar Garg for giving us the opportunity to deliver our work.
Thanks and regards,

Group #15