**BIL 476 Data Mining Project Final Report**

**Recommendation System With Collaborative Filtering**

<https://github.com/Rau7/476_573Recommender>

Kasım Alp TOKER Ayça GÜRLEYİK Orhan ONAR

131101058 151101032 131101063

**ABSTRACT**

In this project, our goal is to enhance a Book Recommendation System with using Collaborative Filtering.In this way, we will equip the local website to demo this system which consist recommendation system with Collaborative Filtering.

**1. INTRODUCTION**

At the present time, people may use the internet to access the alternative of their favourite objects quite easily. However, in the acquiring of alternatives, people generally choose alternatives which are popular. Because of this choice, the person misses a lot of alternatives that fit him/her more. For instance, the person took the book which is not genre of that person likes because that person saw it on the list of popular boks that day and, he could not read that book. Even when people start reading a new genre, they ask the question of where to start. They find what they want to read after they read many books which are not good choices for that person. We intend to integrate a Recommendation System which is created using Collaborative Filtering to shorten the duration of the “long-running adventure”.

**2. RECOMMENDATION SYSTEMS**

[1] Recommendation System or otherwise known as Recommendation Engines is a subclass of the Information Filtering Systems. It is often used in advertising (commercial) applications. Major companies such as Netflix, Youtube and Spotify use the Recommendation System. There are many Recommendation System approaches. Some of these approaches are Collaborative Filtering, Content-Based Filtering, Multi-Criteria Systems.

In this project, we will use Collaborative Filtering which is an approach of the Recommendation System.

**2.1 COLLABORATIVE FILTERING**

[2] Collaborative Filtering is divided into two approaches. These approaches are Narrow and General.

The Narrow Approach which is the newest attempts to predict a user’s interest by taking the idea of many users.

In general, Collaborative Filtering is a filtering process. Informations and patterns are used during this process.

The Collaborative Filtering assumption is that if person A agreed with Person B on a topic in the past, person A is close to agreement on a topic in the future according to a random person.

Collaborative Filtering basically consists of two phases;

* Scan for other users who agree with the user.
* Create an estimate for the user by using the information of the users who agree.

**3. IMPLEMENTATION**

Our implementation will mainly be on the website. We generate a site for trial rather than a real site because our goal is to have more Collaborative Filtering implementation. Technologies and descriptions are located in the 4. Technologies section. Implementation images are located in the 5. Appendix section.

**3.1 IMPLEMENTATION ABSTRACT**

We do not plan to use any “Register Section”, user information is added to the database manually, then user starts adding the books which are read by user to user’s profile. Important case in our project is that when adding one’s book “Genre” means adding the type of the book. We aimed to use the “Genre” attribute in our project instead of the “Rating” used in common Collaborative Filtering.

We thought of using “Rating” to constitute a Recommendation System according to the “Genre” attribute to prevent people from missing out on books which are liked by them, as described in the abstract section. When the user clicks on any books that he/she has added after his/her books, the books of the closest type are listed and user can add the books from that list to his/her list. The user’s books are included in our Recommendation System is detailed in the section 3.2. Algorithm.

Collaborative Filtering needs ratings so we have changed “Genre” Filtering to regular “Rating” Filtering. Genre could not give measurable data!

So, users can no more click their books and see recommended books. We are now using Recommendation data instead of Future Reading data.

**3.2 ALGORITHM**

First of all, our algorithm expects user inputs. As mentioned in the 3.1 Implementation Abstract section, the user must add the book “X” his/her profile then click on the book “X” to see the books which are closest to the book “X” in the “Genre”. The information of book “X” is taken when user clicks into this book. (Book name,author,year,type etc.). Then, to make a search faster, instead of iterating through the whole books dataset, we narrow the dataset to the type of book “X”. When sorting the books closest to the book “X”, if the user selects the by author option, searches for the type of the “X” book + the author of the “X” book will be the first option.The next books will be the ones that do not correlate. In the same way, the person can perform this search according to the year of the book. For example, if the user adds the book “Y” from this list to his/her list, the name of this list will be “Read List”. When the user clicks on the “X” book , the “Y” book will appear on the list until the “Y” book enters the “Read Books” list.

After Mid-Report,

We have seen that we need to use similarity for Collaborative Filtering and after we are clear about our dataset (with rating) we can use all users’ ratings for given user. With ratings we can recommend books for given user.

After regular Collaborative Filtering steps, we extended our recommended data. Regular Collaborative Filtering gives us 10 books with their similarity score. We constructed two new arrays, then if first data has books which has similarity score above 8, we pushed it array1. For those which are below 7, we look author similarity and also genre similarity and pushed it array2. Then we merged these two arrays.

We have now recommended books.

We used the Pearson Correlation Similarity Score when calculating the similarity ratio for recommending books.

**3.3 DATASET**

Collaborative Filtering works more effectively in fairly large datasets. Accordingly, we first wanted to use the dataset of the [3] Good Books 10K project in the [4] Kaggle, but we had some problems with the “Tags”, that is the “Genre” in this dataset Because the Good Books 10K project is a Recommendation System created according to “Rating”, the dataset has not been an appropriate set for us. The different datasets that we found were too large, the DBMS that we used were insufficient. So, we decided to take care of other issues instead of dataset which includes “Genre” until the interim report is finished. For now, we will work on a trial dataset.

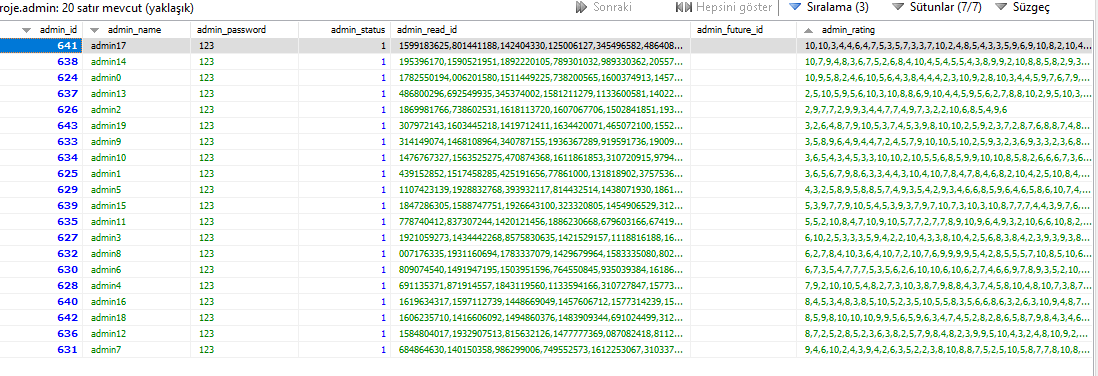
After all works on finding dataset, we found a dataset which we are looking for.

<http://www2.informatik.uni-freiburg.de/~cziegler/BX/>

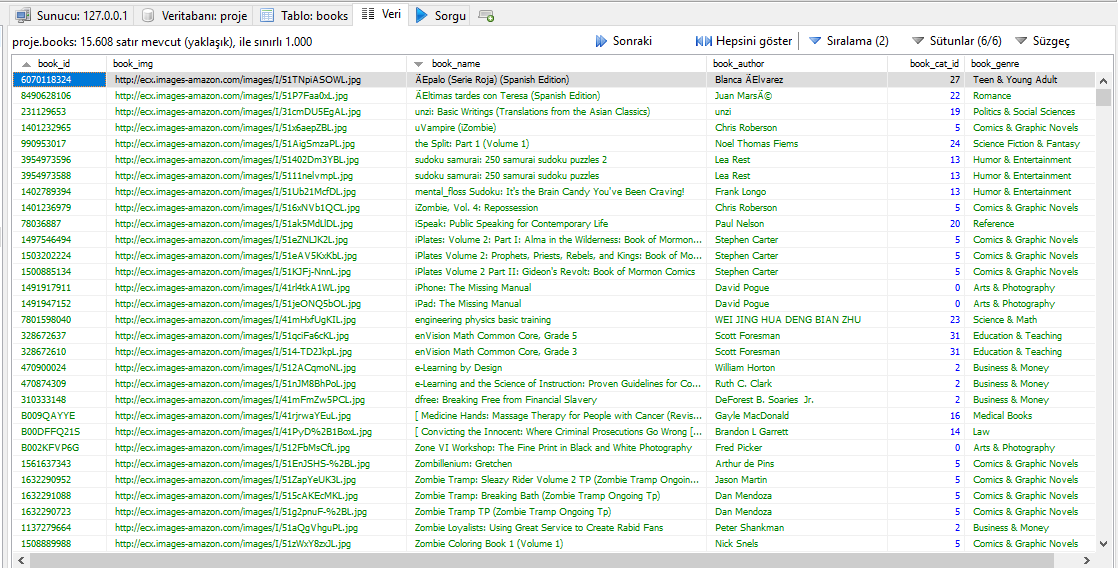
But for implementing Collaborative Filtering, this dataset needs user ratings. Each user should give rating to own readed books. We constructed random 100 users and give them random books and random ratings.

We have constructed ratings also.

User database:



Book database:



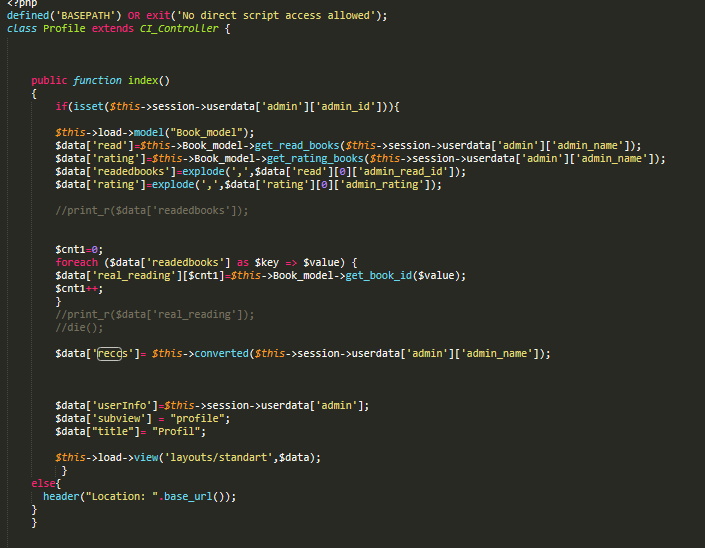
**3.6 FUNCTIONS**

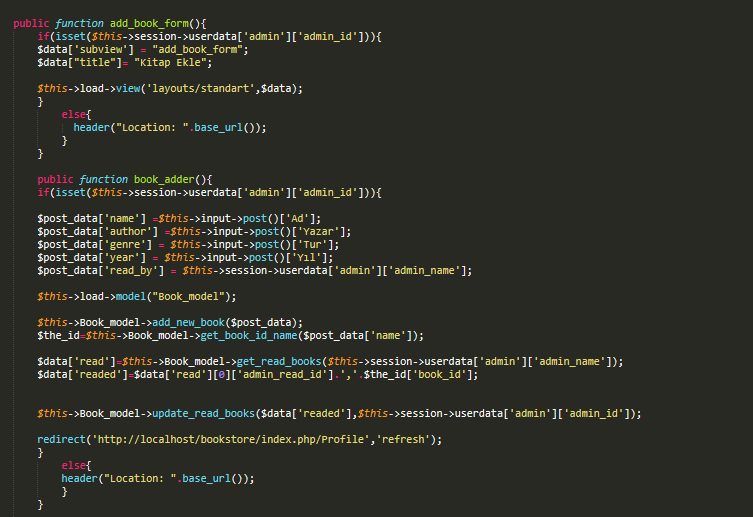
Book functions:

****

****

Profile functions:



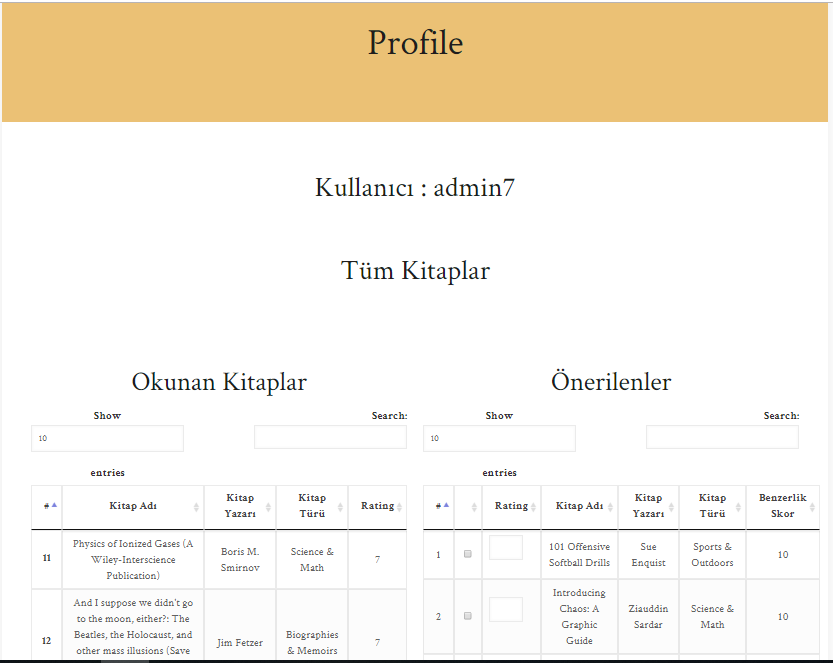


**3.5 USER MODULES**

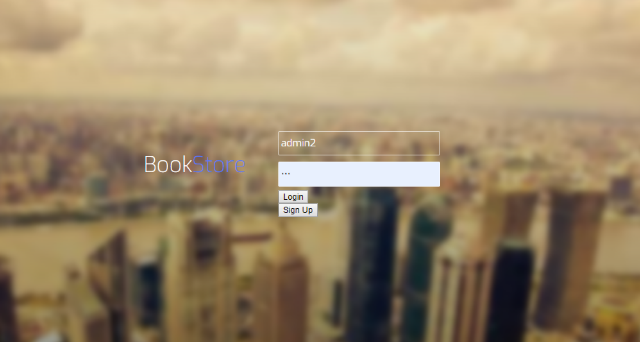
Registration / Login: User or customer should register with basic details to make an account for recommended and read the books online. Registered User can login into the system with valid username and password.  
View All Books: After Successful Login user can see all the books . The book gallery author,rating and name . Recommendations: When User come to website again the system will show the recommendation of books using extended collaborative filtering.   
ReadBooks: All read books are listed for user.  
Log out: User can easily log out from website and can sign in with an other account after clicking log out button.

**3.6 USER INTERFACES**

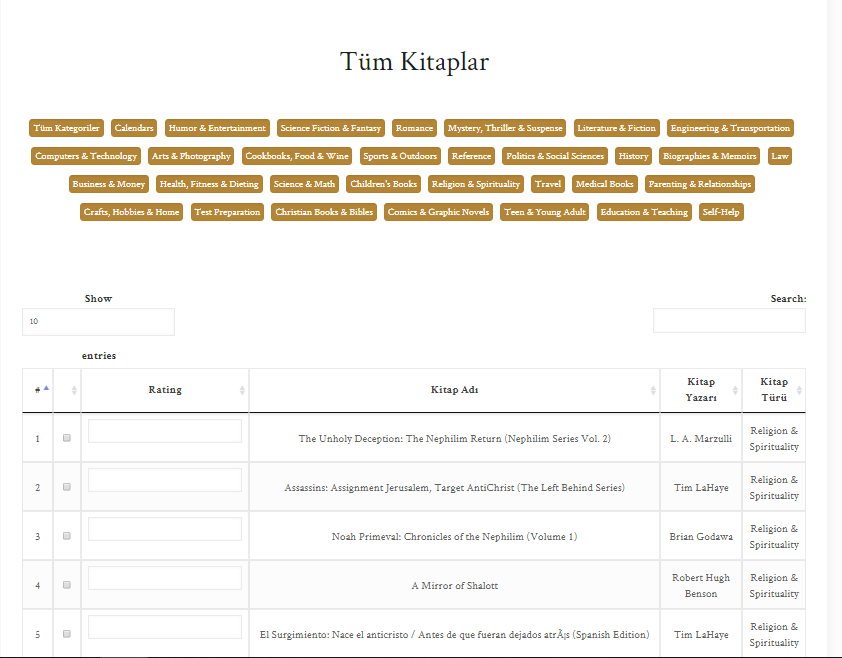
User profile:

****

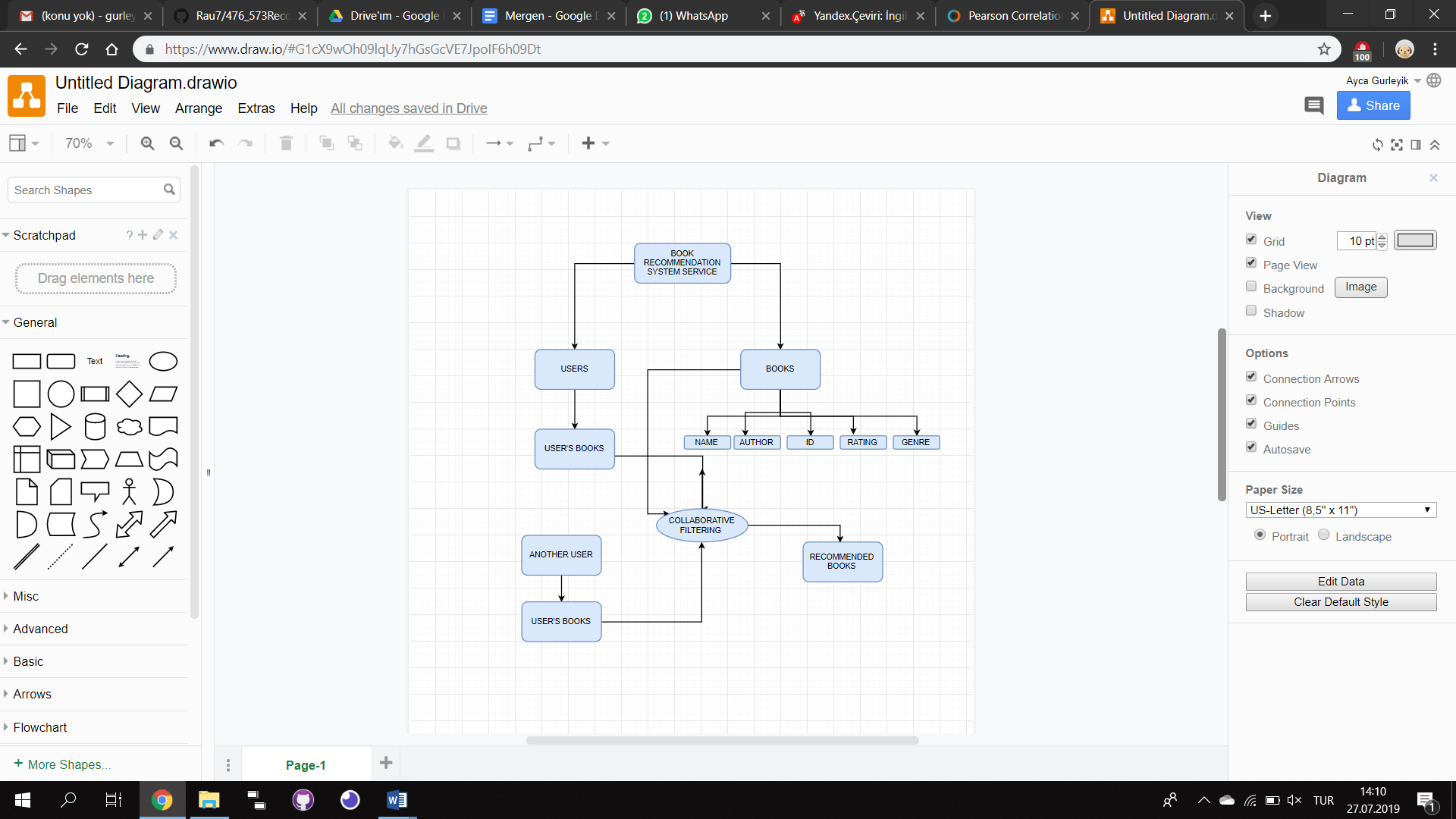
Login page:

****

Books pages:

****

**3.7 SYSTEM FLOW DIAGRAM**

****

**3.8 PERFORMANCE (ACCURACY)**

We calculated accuracy with java by using little test. This test can be found at our GitHub repository.

Mean absolute error (MAE)

The MAE measures the average magnitude of the errors in a set of forecasts, without considering their direction. It measures accuracy for continuous variables. The equation is given in the library references. Expressed in words, the MAE is the average over the verification sample of the absolute values of the differences between forecast and the corresponding observation. The MAE is a linear score which means that all the individual differences are weighted equally in the average.

Root mean squared error (RMSE)

The RMSE is a quadratic scoring rule which measures the average magnitude of the error. The equation for the RMSE is given in both of the references. Expressing the formula in words, the difference between forecast and corresponding observed values are each squared and then averaged over the sample. Finally, the square root of the average is taken. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE is most useful when large errors are particularly undesirable.

The MAE and the RMSE can be used together to diagnose the variation in the errors in a set of forecasts. The RMSE will always be larger or equal to the MAE; the greater difference between them, the greater the variance in the individual errors in the sample. If the RMSE=MAE, then all the errors are of the same magnitude

Both the MAE and RMSE can range from 0 to ∞. They are negatively-oriented scores: Lower values are better.

**3.8 SYSTEM ADVANTAGES & DISADVANTAGES**

Advantages:

This system saves the precious time of customer and very efficient to use.  
Provides large number of choices for books & also recommend for books.  
The system recommending algorithm scale well with co-rated items.

Disadvantages:

Dependent on human ratings for books  
Need user data for better recommendation.

**4. TECHNOLOGIES**

For this project PHP, HTML, SQL, CSS As the framework [5]CodeIgniter (Model View Controller structure similar to framework RubyOnRails)

For Database Management System (DBMS) [6]HeidiSQL we are planning to use. To be able to work in local, we will use [7]Xampp.

**5. CONCLUSION**

In this project,First,we had to create a website for our project.In this process,,for our modules on website, we had too many problems and  solved most of them with  improving our php,html and implementing algorithm skills.We implemented the collaborative filtering and furthermore we extended it for better recommendation system.On extension process,we  integrate our new algorithm with collaborative filtering algorithm which we developed.All the group members learned how to    realize and  implement   recommendation algorithm on our website.When we had a problem about project we solved the problems  as group and  we think that our teamwork skill is highly improved.

**6. APPENDIX**

Project photographs (especially codes) are on our GitHub repository.

**7. REFERENCES**

[1] : Reccomendation Systems <https://en.wikipedia.org/wiki/Recommender_system>

Francesco Ricci and Lior Rokach and Bracha Shapira, Introduction to Recommender Systems Handbook, Recommender Systems Handbook, Springer, 2011, pp. 1-35

[2] : Collaborative Filtering <https://en.wikipedia.org/wiki/Collaborative_filtering>

[3] : Kaggle [https://www.kaggle.com](https://www.kaggle.com/)

[4] : Goodbooks10K <https://www.kaggle.com/zygmunt/goodbooks-10k>

[5] : Codeigniter <https://www.codeigniter.com/>

[6] : HeidiSQL <https://www.heidisql.com/>

[7] : Xampp <https://www.apachefriends.org/tr/index.html>