ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 6, No. 1, 2021

ONLINE VIDEO PLATFORM WITH CONTEXT-AWARE CONTENT-BASED RECOMMENDER SYSTEM

Marian Pisotskyi, Alexey Botchkaryov

Lviv Polytechnic National University, 12, Bandery Str, Lviv, 79013, Ukraine.
Authors' e-mail*: alb@lp.edu.ua*

https://doi.org/10.23939/acps2021.01.046

Submitted on 01.05.2021

***Abstract*: The problem of developing an online video platform with a context-aware content-based recommender system has been considered. Approaches to developing online video platforms have been considered. A comparison of popular online video platforms has been presented. A method of context-aware content-based recommendation of videos has been proposed. A method involves saving information about user interaction with video, obtaining and storing information about which videos the user liked, determining user context, composing a profile of user preferences, composing a profile of user preferences depending on context, determining the similarity between the video profile and a profile of user preferences (with and without context consideration), determining the relevance of the video to the context, the conclusive estimation of the relevance of the video to the user's preferences based on the proposed composite relevance indicator. The developed structure of online video platform has been presented. The algorithm of its work has been considered. The structure of the online video platform database has been proposed. Features of designing the user interface of the online video platform have been considered. The issue of testing the developed online video platform has been considered.**

***Index Terms*: online video platform, context awareness, content-based recommender system**

I. INTRODUCTION

With the development of computer and information technology, it has been possible to take an incredible step towards automation and simplification of various aspects of human life. You can do many things with your computer: search for information, structure data, perform complex calculations, and generally simplify various processes. One of the important areas of application of information technology has been the creation, storage and viewing of digital video. At the same time, the amount of video data of different content is growing rapidly. Accordingly, the need for different means of storing and viewing video data is growing. Therefore, the development and implementation of online video platforms is an increasingly important issue [1–5]. Examples of using online video platforms include: creating and maintaining personal or shared video archives, storing and viewing entertainment content, working with video data in distance learning and online education, storing video data from video surveillance systems and video recording systems, saving training video data sets for machine learning systems, etc.

Given the large number of videos stored on online video platforms, there is a problem of finding such videos that could be of interest to the user [6–8]. The main approach to solving this problem is the development and use of recommender systems [19–21]. At the same time, due to the development of mobile computing technologies, the concept of context-aware computing is becoming an increasingly popular research area [22–28] with the further prospect of developing context-aware recommender systems [29–34].

This paper presents the results of the development of an online video platform with a context-aware content-based recommender system. The main functions of the developed online video platform are the following: 1) storage of video data on the platform server; 2) viewing video data in the mode of video stream transmission from the server; 3) creation of a new channel, within which its owner can add new videos; 4) rating and commenting on the video, the ability to subscribe to the channel; 5) context-aware content-based recommendation of videos; 6) the possibility to edit the user profile; 7) the possibility to distribute video on social networks.

The online video platform is universal and can be configured to run on both local and remote servers. The client side of the online platform is implemented as a web application using the Java programming language, XML and the Hibernate framework. The server side of the online platform is implemented using MySQL Server.

II. APPROACHES TO DEVELOPING ONLINE VIDEO PLATFORMS

The problem of developing online video platforms has many aspects, including the optimization of appropriate software solutions, monitoring and load balancing, user-friendliness of the user interface, the effectiveness of the recommender system and many others. Let's consider approaches to creating online video hosting platforms on the example of some well-known platforms.

The D.Tube platform is a crypto-decentralized online video platform, built on top of the STEEM Blockchain and the IPFS peer-to-peer network. The D.Tube platform is decentralized, i.e. all information downloaded by users is stored in a blockchain, which ensures the confidentiality of the author's personal data and the preservation of his content. It should be noted that D.Tube does not contain censorship, i.e. the uploaded video is not moderated: the power to control, analyze and evaluate content is fully transferred to viewers, which is both an advantage and a disadvantage of the platform.

The Vimeo platform is designed to host video content that is directly related to art. The platform allows the user to distribute their video content to a wide audience and receive feedback in the form of comments under the video. The full functionality of the platform is paid, which can be considered a disadvantage. As well as on the D.Tube platform, there is no advertising on this platform.

The Twitch platform is designed for online video broadcasts. The main type of video content on this platform is streaming video games, including e-sports broadcasts. An additional option is online chat, which allows to communicate during the broadcast, which further attracts users to the platform. The Twitch platform contains various mechanisms for advertising and monetizing video content.

YouTube is the world's most popular online video platform. Platform users can add, view and comment on videos. Due to its simplicity and ease of use, YouTube has become one of the most popular video hosting platforms. YouTube uses artificial intelligence algorithms to recommend a video to a user according to their preferences. The advantage of YouTube is cross-platform, i.e., the ability to watch videos from almost any modern mobile device or computer.

These platforms can be compared in terms of availability, ease of use, cross-platform, versatility, reliability and efficiency. The results of the comparison on a ten-point scale are shown in Table 1.

III. CONTEXT-AWARE CONTENT-BASED RECOMMENDATION

As part of the developed video platform, a context-aware content-based recommender system has been implemented. In this case, the items recommended to the user are videos uploaded to the server and configured for viewing by other users. The set of such videos is denoted as V={v}. The set of features of a video is denoted as Q={q}. Examples of features in the set Q can be: video genre, video duration, the date and time of uploading the video to the platform, the author of the video, a set of tags that characterize the content of the video, etc. Each of the features can take values from the corresponding set of values $Y(q)=\{y_1,y_2,\ldots\}$. Therefore, each video v can be matched with its profile (item profile) in the form of a set of pairs (feature, value): VP(v)={(q,y)}.

All users registered in the platform have the opportunity to watch videos that are shared with other users. The set of users is denoted as U={u}. During user u interaction with multiple V videos (viewing, rating, commenting), the system collects information about which videos the user liked and how much. The following factors are taken into account:

p_1 – number of video v views by user u,

p_2 – time of last viewing,

p_3 – duration of one video viewing (average duration in case of several views),

p_4 – rating that the user gave the video (including no rating),

p_5 – the presence of comments about the videos added by the user, and their number,

p_6 – the size and content of comments added by the user, including their emotional color,

p_7 – time of last video comment.

Table 1

Comparative characteristics of online video platforms

Characteristic	D.Tube	Vimeo	Twitch	Youtube
Availability	8	2	4	7
Ease of use	7	8	7	10
Cross-platform	8	6	8	8
Versatility	10	4	4	10
Reliability	8	8	8	10
Efficiency	8	7	7	9

The system implements the possibility to add other factors and remove previously added ones.

For each user u, the collected information is stored as a set

$$UV(u) = \{(v_1,w_1), (v_2,w_2), \ldots\}, \quad (1)$$

where $\{v_i\}$ is the video viewed by the user, and $\{w_i\}$ is a generalized indicator of how much the user u liked the video, such as $w_i \in [0,1]$. The value of w_i will be the greater, the more a user liked the video. For each video viewed, the value of w(v) is determined using the F_W function:

$$w(v) = F_W(p_1, p_2, p_3, p_4, p_5, p_6, p_7). \quad (2)$$

Based on UV(u) and {VP(v)}, by aggregating the profiles of those videos that the user liked, a profile of the user's preferences UP(u) ={(q,y)} is formed, which has the same appearance as the profile of a video, i.e. it is a set of pairs (feature, value). To do this, we use the F_A aggregation function:

$$UP(u) = F_A(UV(u), \{VP(v)\}), \quad (3)$$

where UV(u) is information about the user's interaction with the video, {VP(v)} is the set of profiles of those videos that the user liked. The system implements the possibility to use different variants of the FA aggregation function. By default, for each feature q the aggregation function calculates its average value based on the values of {q} in the profiles of those videos that the user liked, and the values of {w} from UV(u), which are used as weights.

Based on the idea of content-based recommendation, the measure of similarity F_S is used to determine which videos the user might like. Using F_S we can find videos whose profiles VP(v) in the parametric space are closest to the user profile UP(u) of the user u. That is, F_S

is used to find among all videos {V} those that are closest in their features {(q,y)} to the user's preferences. Thus, for each video v, one can calculate the similarity of its profile VP(v) to the preferences of the user UP(u):

$$D_S(v) = F_S(VP(v), UP(u)). \tag{4}$$

As a measure of similarity in our recommender system we use cosine similarity [21]:

$$D_s = \frac{\sum_{i=1}^{n}[q_i(v) \cdot q_i(u)]}{\sqrt{\sum_{i=1}^{n} q_i(v)^2} \cdot \sqrt{\sum_{i=1}^{n} q_i(u)^2}}, \tag{5}$$

where $D_S \in [-1,+1]$ is similarity value; $q_i(v)$ is the feature of the video v from its profile VP(v); $q_i(u)$ is the corresponding feature from the profile of user's preferences UP(u); n is the number of features of the video. The smaller the D_S value, the more the video profile differs from the user's preferences, and vice versa, the larger it is, the more the video profile is similar to the user's preferences.

The context-awareness of the recommendation is provided on the basis of the concept of context-aware computations [22–28] and the ideas of context-aware recommendation [29–34]. The current context determines the situation in which the user views the video, in its various aspects, which are reflected in the context model. The context model is given by a set of context parameters C={c}. The system supports the following context parameters:

c_1 – location of the user at the time of watching the video,

c_2 – current time,

c_3 – the total time, during which the user watches videos on the platform,

c_4 – the type of device on which a user views the video,

c_5 – the type of network connection between the user's device and the online video platform.

The system implements the possibility to add other context parameters and remove previously added ones.

Each context parameter can take values from the corresponding set of values $X(c) = \{x_1,x_2,...\}$. A separate instance of the context s (hereinafter simply the context) of the user is given by a set of pairs (parameter, value): $s=UC(u)=\{(c,x)\}$, $s \in S$, where S is the space of all possible contexts. At one point in time, user u is in some context s=UC(u). The current context is determined by the corresponding context definition module (Fig. 1).

Context-awareness in the system is considered in the following two aspects.

1. The relevance of the video to the context without taking into account the user's preference profile. For each video, information is stored about the context in which it was viewed and how many times (regardless of which user viewed the video). That is, if the video was viewed in the context s, the corresponding counter will be increased by one. As a result, for each video v and each instance of context s, the index of the relevance of video to the context $G(v,s) \in [-1,+1]$ is calculated. The more often the video v is viewed in the context s, the greater the value of G(v,s) is and vice versa.

2. The relevance of the user's preference profile to the context. For each instance of context s we store and use its own version of information about which videos the user liked when watching them in context s. That is, in addition to the general set UP(u), the system stores the set UP(u,s) for each instance of the context s. Accordingly, the F_A aggregation function determines the user's preference profile in the context s:

$$UP(u,s) = F_A(UV(u,s), \{VP(v)\}), \tag{6}$$

where {VP(v)} is the set of profiles of those videos that the user liked when watching them in the context s. Thus, for each video v one can calculate the similarity of its profile VP(v) to the preferences of the user UP (u,s) in the context s:

$$D_S(v,s) = F_S(VP(v), UP(u,s)). \tag{7}$$

The conclusive composite indicator P(v,u,s) of relevance of the video v to the user u preferences taking into account the context s is defined as the weighted sum of three terms:

1) similarity of the video profile VP(v) to the profile of preferences of the user UP(u);

2) relevance of the video to the context without taking into account the user's preference profile;

3) relevance of the video profile VP(v) to the profile of the user's preference UP(u,s) in the context s.

Namely,

$$P(v,u,s) = k_D \times D_S(v) + k_C \times G(v,s) + k_S \times D_S(v,s), \tag{8}$$

$$k_D + k_C + k_S = 1, \tag{9}$$

where $k_D,k_C,k_S \in [0,1]$ determine the weight of the corresponding terms. The larger the value of P(v,u,s) is, the more video v corresponds to the preferences of user u, taking into account the context s. The larger the values of the weight k_C and k_S are, the more the context-awareness is taken into account.

The procedure for getting a context-aware recommendation of the video is as follows:

1. Save information about user interaction with the video.

2. Get and save information about which videos the user liked UV(u).

3. Determine the current context s.

4. Create a profile of user preferences UP(u).

5. Create a profile of user preferences UP(u,s) in the context s.

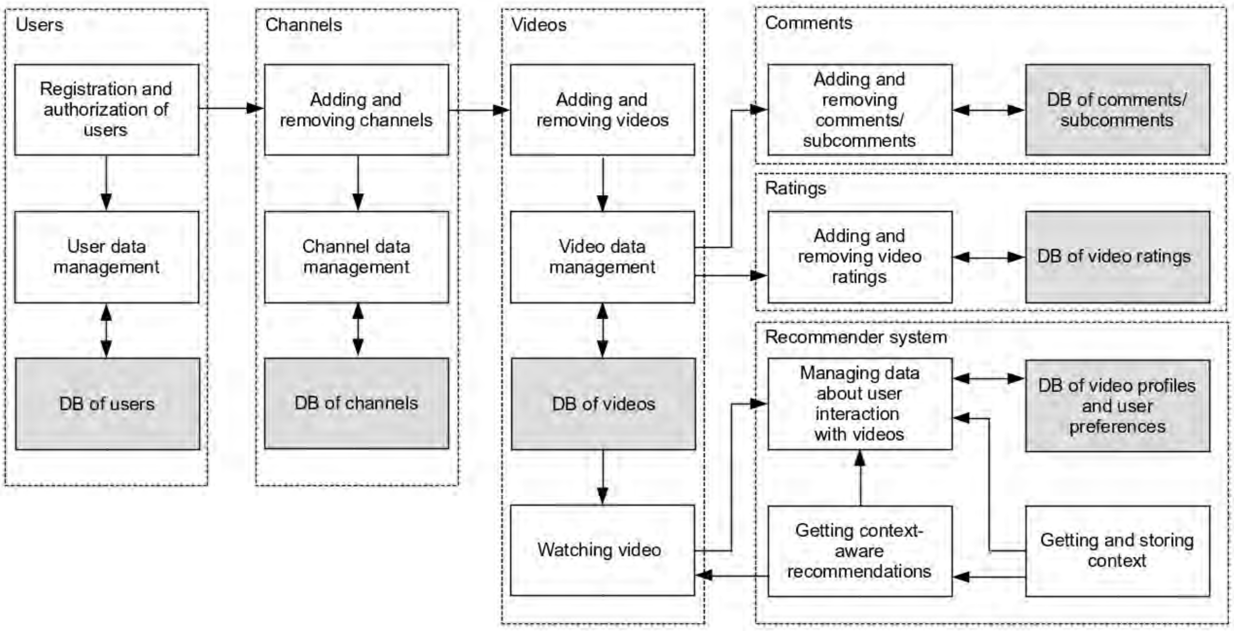

Users
Registration and authorization of users
User data management
DB of users
Channels
Adding and removing channels
Channel data management
DB of channels
Videos
Adding and removing videos
Video data management
DB of videos
Watching video
Comments
Adding and removing comments/ subcomments
DB of comments/ subcomments
Ratings
Adding and removing video ratings
DB of video ratings
Recommender system
Managing data about user interaction with videos
DB of video profiles and user preferences
Getting context-aware recommendations
Getting and storing context

Fig. 1. The outline of the general structure of the online video platform with a context-aware content-based recommender system.

6. Determine the similarity of the profile of each video to the preferences of the user: $D_S(v)$.

7. Determine the similarity of the profile of each video to the profile of user preferences in the current context: $D_S(v,s)$.

8. Determine the relevance of each video to the context G(v,s).

9. Determine for each video its composite relevance indicator P(v,u,s).

10. Recommend to the user the first N videos from the list, sorted in descending order of the value of the composite relevance indicator P(v,u,s).

IV. THE STRUCTURE OF THE ONLINE VIDEO PLATFORM

The online video platform is based on a client-server model. The main functionality is implemented in the following modules (Fig. 1): users, channels, videos, comments, video ratings and recommendation system. Working with users includes user registration and authorization, as well as user data management. Working with channels includes adding and deleting channels, as well as managing channel data. Modules for adding and deleting video comments and ratings are also implemented in the structure. Working with video includes adding and deleting video, managing video data and watching videos (encoding, decoding, streaming video). The recommendation system includes modules that implement the context-aware content-based recommendation method discussed above. The modules of the recommendation system directly interact with the modules of the video viewing subsystem (Fig. 2).

ALGORITHM OF OPERATION OF THE ONLINE VIDEO PLATFORM

The algorithm of the online video platform includes the algorithm of the server component and the algorithm of the client component. The algorithm of the server component contains the necessary sequence of operations to work with users, channels, videos, comments, video ratings and a sub-algorithm of the recommender system. The algorithm of the client component describes how the user has access to the main functionality of the system implemented in the server component. In particular, the client algorithm contains the necessary sequence of operations for authorization and registration of users, entering user login and password, checking the correctness of user input, creating a new channel, viewing the contents of an existing channel, adding new video to the channel, watching videos, adding comments to the video, adding video ratings, subscribing to the channel, etc.

V. THE STRUCTURE OF THE ONLINE VIDEO PLATFORM DATABASE

The database of the online video platform is implemented on the basis of MySQL Server. The database structure implements all the necessary tables, fields and relationships between tables to store platform data. The main part of the database structure (Fig. 3) contains the following tables: the user table is used to store information about users of the service; the channel_subscriber table stores data about the channel and subscribers, i.e. it is an intermediate table for imple-

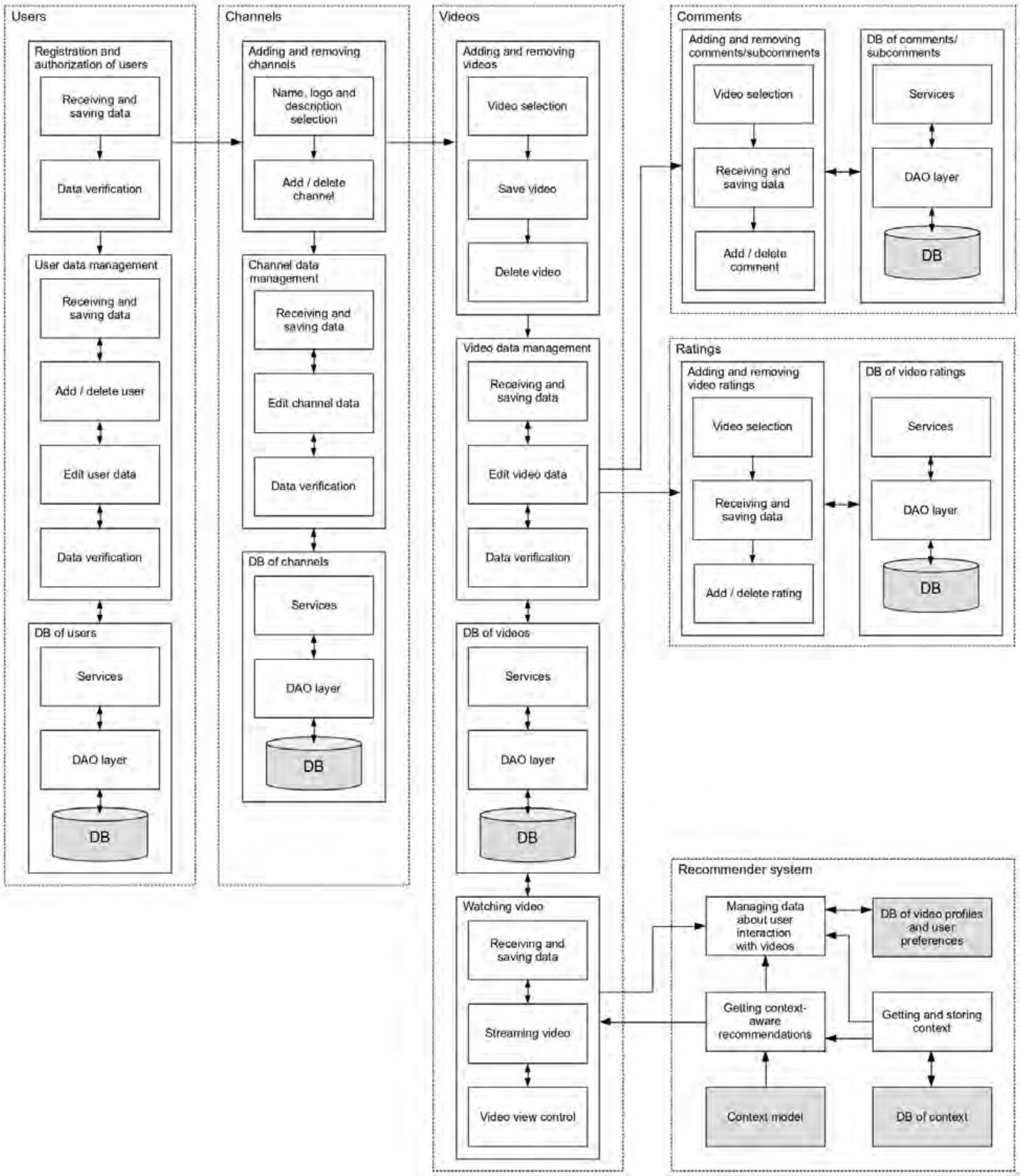


Fig. 2. The outline of the detailed structure of the online video platform with a context-aware content-based recommender system.

menting a many-to-many relationship between the ***channelpage*** and ***user*** tables; the ***channelpage*** table contains information about the channels created by the users; the *comment* table contains data about the comments left by users under the video; the *innercomment* table stores information about nested comments (subcomments) within one (main) comment; the *mylike* table contains data about the ratings that users make videos; the video table stores information about the video uploaded by users.

VI. THE USER INTERFACE OF THE ONLINE VIDEO PLATFORM

The client component of the online video platform is implemented in the form of a web application. Accordingly, the user interface looks like a multi-page website. Java and JavaScript programming languages were used in the development of the user interface, as well as the XML and the Hibernate framework. The user interface implements two independent sections: the platform user interface and the platform administrator interface. The platform user interface contains a window for user authorization, the main working window, a window for watching videos, a window for downloading videos, a window for creating a channel, a window for viewing created channels, a window for viewing user profile information with the possibility to edit it (Fig. 4) and others. The platform administrator interface provides basic functions for managing the platform, user data and service data.

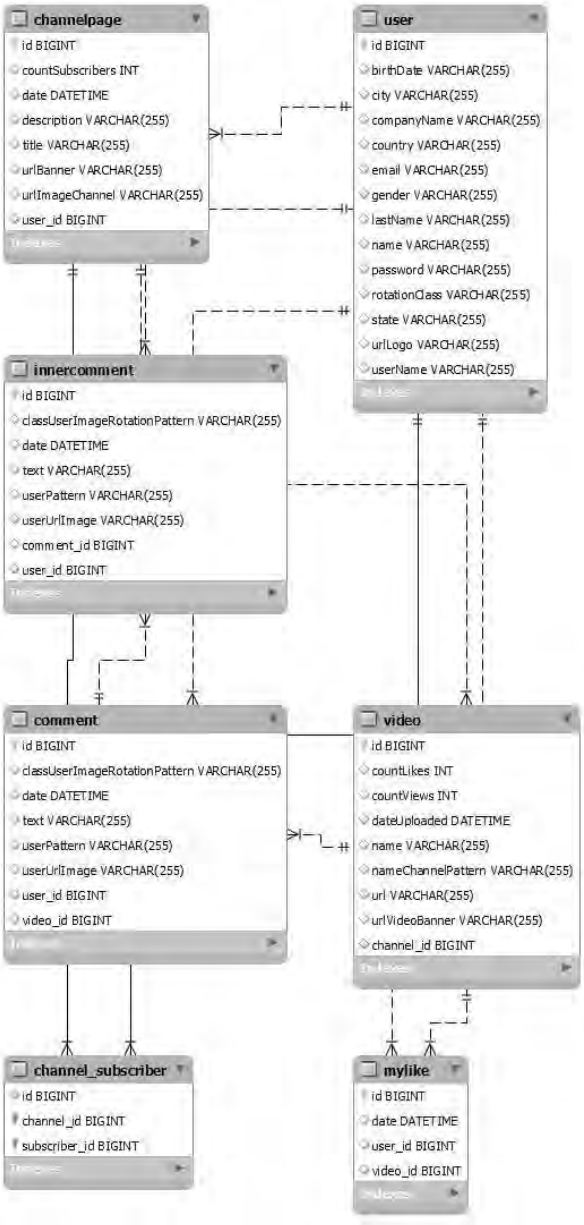
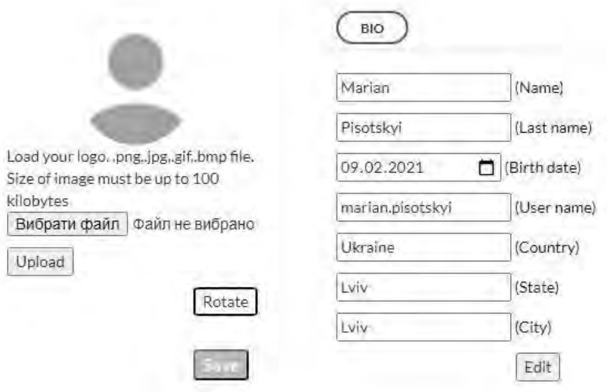


Fig. 3. The outline of the Database structure of the online video platform.

VII. TESTING THE ONLINE VIDEO PLATFORM

The operation of the online video platform was tested in three aspects: the correctness and efficiency of the server side (including load testing), the correctness of the client side, the efficiency of the recommender system. The testing confirmed the absence of critical errors in the operation of the online video platform. During the testing of the efficiency of the server side, the maximum trouble-free server load was determined (Fig. 5) using the Apache JMeter tool. Testing the client side for errors helped to significantly improve the performance of platform algorithms and improve the functionality of the client side. Testing was performed modularly. Two methods were used during testing: black-box testing and white-box testing.


Load your logo. .png,.jpg,.gif,.bmp file. Size of image must be up to 100 kilobytes

Marian (Name)

Pisotskyi (Last name)

09.02.2021 (Birth date)

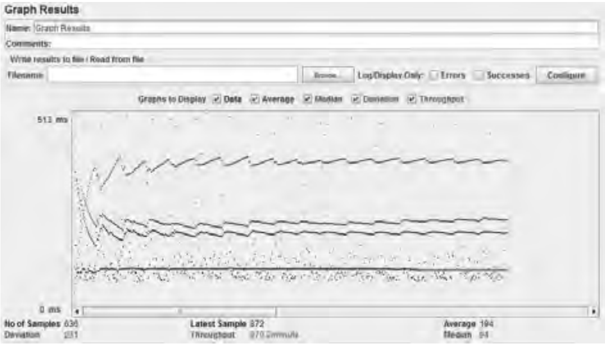
marian.pisotskyi (User name)

Ukraine (Country)

Lviv (State)

Lviv (City)

Fig. 4. Example of the client interface: user profile.


Graph Results

Name: Graph Results

Comments:

Write results to file / Read from file

Filename

Log/Display Only: □ Errors □ Successes

Graphs to Display ☑ Data ☑ Average ☑ Median ☑ Deviation ☑ Throughput

No of Samples 636 Latest Sample 372 Average 194

Deviation 231 Throughput 370/minute Median 94

Fig. 5. The results of load testing of the server side of the online video platform.

VIII. CONCLUSION

The problem of developing an online video platform with a context-aware content-based recommender system has been considered. Approaches to developing online video platforms have been considered. A comparison of popular online video platforms has been presented. A method of context-aware content-based recommendation of videos has been proposed. A method involves saving information about user interaction with video, obtaining and storing information about which videos the user liked, determining user context, composing a profile of user preferences, composing a profile of user preferences depending on context, determining the similarity between the video profile and the profile of user preferences (with and without context consideration), determining the relevance of the video to the context, the conclusive estimation of the relevance of the video to the user's preferences based on the proposed composite relevance indicator.

The developed structure of the online video platform was presented. The algorithm of its work was considered. The structure of the online video platform database was proposed. Features of designing the user interface of the online video platform were regarded. The issue of testing the developed online video platform was tackled. The testing confirmed the absence of critical errors in the operation of the online video platform.

REFERENCES

[1] Lee, J. (2005). *Scalable Continuous Media Streaming Systems: Architecture, Design, Analysis and Implementation*. Wiley. 394 p.

[2] Ce Zhu, Yuenan Li, Xiamu Niu (2010). *Streaming Media Architectures, Techniques, and Applications: Recent Advances*. IGI Global. 502 p.

[3] Dang Nam Chi Nguyen (2006). *Scalable and Cost-EffectiveFramework for ContinuousMedia-On-Demand*, Ph.D. Thesis, University of Technology Sydney. 137 p.

[4] Parthasarathy Ranganathan et al. (2021). Warehouse-scale video acceleration: co-design and deployment in the wild. In: Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 600–615.

[5] Li, H. and Liu, J. (2012). Video Sharing in Online Social Network: Measurement and Analysis. In: *Proceedings of ACM NOSSDAV'12*. Toronto, Canada, pp. 83–88.

[6] Davidson, J., Liebald, B., Liu, J. and Nandy, P. (2010). The YouTube video recommendation system. In: *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010*. Barcelona, Spain, pp. 293–296.

[7] Zhe Zhao et al. (2019). Recommending what video to watch next: a multitask ranking system. In: *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys'19)*, pp.43–51.

[8] Cheuque, G., Guzmán, J. and Parra, D. (2019). Recommender Systems for Online Video Game Platforms: the Case of STEAM. In: *Proceedings of The 2019 World Wide Web Conference*, pp. 763–771.

[9] Ricci, F., Rokach, L., Shapira, B. and Kantor, P. (eds.) (2015). *Recommender Systems Handbook*. 2nd ed., Springer. 1020 p.

[10] Aggarwal, C. (2016). *Recommender Systems: The Textbook*. Springer. 519 p.

[11] Schrage, M. (2020). *Recommendation Engines*. The MIT Press. 296 p.

[12] Falk, K. (2019). *Practical Recommender Systems*. Manning Publications. 432 p.

[13] Robillard, M., Maalej, W., Walker, R. and Zimmermann, T. (eds.) (2014). *Recommendation Systems in Software Engineering*. Springer-Verlag Berlin Heidelberg. 560 p.

[14] Jannach, D. (2010). *Recommender Systems: An Introduction*. Cambridge University Press. 352 p.

[15] Jie Lu, Qian Zhang, Guangquan Zhang (2020). *Recommender Systems: Advanced Developments*. WSPC. 362 p.

[16] Suresh Kumar Gorakala (2017). *Building Recommendation Engines*. Packt Publishing. 357 p.

[17] Neumann, A. (2009). *Recommender Systems for Information Providers: Designing Customer Centric Paths to Information*. Physica-Verlag Heidelberg. 158 p.

[18] Isinkayea, F., Folajimib, Y. and Ojokohc, B. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, Volume 16, Issue 3, November, pp. 261–273.

[19] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang and Guangquan Zhang (2015). Recommender system application developments: A survey. *Decision Support Systems*, Vol. 74, pp. 12-–32.

[20] Leskovec, J., Rajaraman, A., Ullman, J. (2020). *Mining of Massive Datasets*. 3rd ed. Cambridge University Press 565 p.

[21] Connor, R. (2016). A Tale of Four Metrics. In: Amsaleg, L., Houle, M., Schubert, E. (eds.) *Similarity Search and Applications. SISAP 2016. Lecture Notes in Computer Science*, vol. 9939. Springer, pp. 210–217.

[22] Schilit, B., Adams, N. and Want, R. (1994) Context-aware computing applications. In: *Proceedings of the IEEE Workshop on "Mobile Computing Systems and Applications"*, IEEE Computer Society, pp. 85–90.

[23] Abowd, G., Dey, A., Brown, P., Davies, N., Smith, M. and Steggles, P. (1999) Towards a Better Understanding of Context and Context-Awareness. In: Gellersen, H. (ed.) *Handheld and Ubiquitous Computing. Lecture Notes in Computer Science*, vol 1707. Springer, Berlin, Heidelberg. pp. 304–307

[24] Bolchini, C., Curino, C., Quintarelli, E., Schreiber, F. and Tanca L. (2007). A data-oriented survey of context models. *ACM SIGMOD Record*, 36, 4, pp. 19–26

[25] Perera, C., Zaslavsky, A., Christen, P. and Georgakopoulos, D. (2014). Context Aware Computing for The Internet of Things: A Survey. *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, First Quarter, pp. 414–454.

[26] Grifoni, P., D'Ulizia, A., and Ferri, F. (2018). Context-Awareness in Location Based Services in the Big Data Era, In: Skourletopoulos, G., Mastorakis, G., Mavromoustakis, C., Dobre C. and Pallis, E. (eds.) *Mobile Big Data. Lecture Notes on Data Engineering and Communications Technologies*, Springer, vol. 10, pp. 85–127.

[27] Capurso, N., Bo Mei, Tianyi Song and Xiuzhen Cheng (2018). A survey on key fields of context awareness for mobile devices. *Journal of Network and Computer Applications*, Vol. 118, pp. 44–60.

[28] Botchkaryov, A. (2018). Context-Aware Task Sequence Planning for Autonomous Intelligent Systems. *Advances in Cyber-Physical Systems*, Lviv, Vol. 3, Nr 2, pp. 60–66.

[29] Adomavicius G. and Tuzhilin A. (2011) Context-Aware Recommender Systems. In: *Recommender Systems Handbook*, ed. by Francesco Ricci et al., Springer, pp. 217–253.

[30] Adomavicius, G., Mobasher, B., Ricci F. and Tuzhilin A. (2011). Context-Aware Recommender Systems. *Ai Magazine*, 32(3), pp. 67–80.

[31] Abbar, S., Bouzeghoub, M., Lopez, S. (2009). Context-Aware Recommender Systems: A Service-Oriented Approach. In: *Proceedings of the 3rd International Workshop on Personalized Access, Profile Management and Context Awareness in Databases (PersDB)*. Lyon, France.

[32] Shaina Raza and Chen Ding (2019) Progress in context-aware recommender systems: An overview. *Computer Science Review*, Vol. 31, pp. 84–97.

[33] Nawrocki, P., Śnieżyński, B. and Czyżewski, J. (2016). Learning Agent for a Service-Oriented Context-Aware Recommender System in a Heterogeneous Environment, *Computing and Informatics*, Vol. 35, pp. 1005–1026.

[34] Bouneffouf, D. (2012). Following the User's Interests in Mobile Context-Aware Recommender Systems: The Hybrid-e-greedy Algorithm. In: *Proceedings of the 2012 26th International Conference on Advanced Information Networking and Applications Workshops*, Lecture Notes in Computer Science, IEEE Computer Society, pp. 657–662.

Marian Pisotskyi is a student of the Computer Engineering Department at Lviv Polytechnic National University. He has been doing scientific and research work since 2020. He has professional experience working in IT as a Software Developer in TechMagic (Lviv, Ukraine) since 2020. His research interests include web-technologies, recommender systems and machine learning.

Alexey Botchkaryov was born in 1975 in Lviv, Ukraine. He received B.S. and M.S. degrees in computer engineering at Lviv Polytechnic National University, in 1998 and Ph.D. degree in computer systems and components at Lviv Polytechnic National University in 2019. He has been doing scientific and research work since 1994. Currently, he is an associate professor with the Computer Engineering Department, Lviv Polytechnic National University. His research interests include self-organization in complex systems, structural adaptation, intelligent information-gathering agents and multi-agent systems.