Received: 18 March 2026 Revised 21 April 2026 Accepted 24 April 2026 Page 1- 12

ORIGINAL RESEARCH

Journal of Research and Review in Science,
Volume 12, Issue 1, 1-12, June 2026.
Doi:

Design and Development of a Secure Cross-Platform Video Hosting Application

Adetokunbo A. Adenowo[1*], Oreoluwa D. Jokosenumi[1], Favour E. Onyebuchi[1], Adam F. Zubair[1]

[1]*Department of Computer Science, Faculty of Computing and Information Technology, Lagos State University, Nigeria*

Correspondence
Adetokunbo Adenowo
Department of Computer Science, Faculty of Computing and Information Technology, Lagos State University, Nigeria.
Email:adetokunbo.adenowo@lasu.edu.ng

Funding information: Non

Abstract:

Introduction: Amid the rapid growth in digital media consumption on the internet, prioritizing efficiency, security, and user experience has become imperative in designing video hosting platforms.

Aims: Hence, this paper presents the conceptualization, realization, and assessment of a cross-platform video hosting application named PlaySphere, which fulfills the criteria mentioned above. The motivation for this research came from the discovery of platform limitations (e.g., YouTube and TikTok). Such constraints in the platforms include disruptive advertisement, potential algorithmic influence, user data privacy, and restrictive and unfair revenue sharing monetization policy.

Materials and Methods: The study utilizes a user-centered design (UCD) methodology. The methodology started with the initial stage of gathering data by the means of a Google Forms survey to thoroughly determine the user's requirements and preferences. The decision above led to the creation of a system architecture that was realised by: Firebase Authentication to help users register securely; Firestore to meet real-time data requirements; Firebase Storage to deal with media assets; and Paystack integration for dependable payment gateway and premium content access. Concerning data confidentiality and integrity, PlaySphere was set up to take advantage of Firebase's built-in AES 256 encryption for data at rest, along with Transport Layer Security (TLS) for data in motion. Functional testing of the complete system and user evaluations were carried out to determine the working, overall usability, and performance indicators.

Results: The study results reveal that PlaySphere provides a smooth and enjoyable viewing experience, with low playback latency and good content management when contrasted with some traditional platforms. Furthermore, the assessment points to a high degree of user contentment, especially with the very simple look of the app, its stability, and quick responsiveness.

To Keywords: Video Hosting; Secure Cross-Platform; Flutter; Firebase;

All co-authors agreed to have their names listed as authors.

1. INTRODUCTION

New technology evolution is causing a vast change in the digital media industry. The changes in the industry are mainly due to the advances in mobile technology, the increased use of broadband connectivity, and the availability of user-generated content platforms. Since more and more people are making, sharing, and watching video content globally, the demand for video sharing platforms that are secure, highly scalable, and fundamentally inclusive never fades and it is imperative [1] [2]. Some of the leading platforms, such as YouTube and TikTok, have had a significant impact on the way content is consumed, interacted with, and monetised [3] [4] [5]. However, these platforms still face various issues related to data privacy, the perceived fairness of monetisation, content moderation protocols, excessive advertisement, and algorithmic transparency; these issues of the digital media environment are still prevalent [6] [7] [8].

Amid this rapid change, the need for a video sharing solution that meets the three criteria of being safe, without the annoyance of ads, and truly user-friendly is becoming increasingly significant. **PlaySphere**, the app described in this paper, is a response to the above-mentioned problems. The app is designed to offer a comfortable environment for getting and sharing video content. Besides, it heavily focuses on safeguarding privacy, facilitating accessibility on a wide range of platforms, and putting the power in creators' hands. The mentioned app is different from many traditional platforms which heavily depend on ad inclined revenue models [9] [10]. Instead, it combines a subscription model that is purposely kept unobtrusive or non-restrictive in terms of user viewing; at the same time, putting the efforts to provide fairly equal monetisation opportunities for creators, thereby supporting their creative freedom [11] [12].

Looking at security, research shows that a lot of multimedia and personal data have been exchanged through the cloud infrastructures; thus, security is a key factor that must be considered for the successful deployment of a contemporary video, sharing platform [1][13]. PlaySphere thus tightly secures user engagement by strategic use of Firebase (a cloud, based backend service from Google); the latter comes with the Advanced Encryption Standard (AES) with 256-bit keys to protect the stored videos, login credentials and communication channels [14][15]. AES is characterized as an extremely reliable and secure encryption standard and plays a crucial role in the confidentiality, integrity and availability of the data [16][17]. In an attempt to deliver exhaustive data protection to the videos uploaded by users, PlaySphere combines the strong AES-based Firebase security architecture with safe authentication methods and encrypted storage protocols. Therefore, it intends to prevent illegal access and the risk of data leakage.

Regarding the development of PlaySphere, its frontend is completely written with the Flutter. Flutter is an application development platform that is recognized as a multi-platform UI toolkit with the capability to deliver a consistent user experience for applications running on Android, iOS, and web [18]. It is a fun development platform that was selected with the aim of making the app accessible to a larger number of users and providing a seamless user experience across different devices. The decision to use flutter may solve a significant drawback of non, multi, platform development tools (like swift, an ios only platform; kotlin, an android only platform; etc.). On the other hand, for the backend, the application's database and cloud storage are based on the Firebase tool, a single and integrated storage infrastructure tool that is carefully designed to ensure the scalability, security, and real time data synchronization in perfect harmony.

Regarding monetisation, PlaySphere bases its revenue model mainly on the creator community; thus, facilitates long-term sustainability through stable compensation. The approach tries to solve the issue of financial unfairness among participants of contemporary video, sharing platforms. Through the above, mentioned predefined subscription tiers, PlaySphere provides creators with fixed income streams and users with a clean, ad free experience. It thereby presents a potential solution to the financial dilemmas of some mainstream platforms [11].

In brief, PlaySphere embodies a fully-fledged idea of a highly secure and an efficient video sharing platform that is respectful of user privacy. At the end of the day, it is about handing the power over to the creators and protecting the users, thus this paper contributes to the debate on the secure and inclusive digital media environments of the future.

2. MATERIAL AND METHODS

This part of the paper describes "PlaySphere, " a secure, cross platform video hosting solution. The methods section logically breaks down into two parts: Materials and Methods. The earlier part gives the specifics of the software environment, hardware components, and database structures used during the development process. The latter section discusses the application's architectural design, implementation steps, and performance evaluation activities.

2.1 Materials

The materials chosen for this research project are as follows:

2.1.1 Software

This part is all about the software used for the development of the PlaySphere platform and the set of tools:

i. Flutter (version 3.32.1): This framework was chosen strategically as it has the capacity to compile native applications for Android, iOS, and web platforms from a single codebase [18]. This strategic move allowed for excellent performance and the critical design consistency across all the devices to be maintained.

ii. Dart: It is the main programming language for Flutter and was used to accomplish the application's core features and interface designs.

iii. Firebase: This solution offered the necessary backend framework. It includes various tightly coupled modules: Firebase Authentication, which controls user signup and login securely; Firestore, which is used for real-time data operations; and Firebase Storage service which is responsible for the uploading and downloading of video content securely [14] [15].

iv. Paystack: This service was included as the payment gateway to automatically process the transactions of the users who are unlocking the premium content within the platform.

Data security was one of the main concerns when designing the platform. Thus, the application and all the supporting services are safeguarded through the usage of AES 256-bit encryption algorithm.

2.1.2 Hardware

The hardware resources referred to in this context are the different devices and systems that were used to test the PlaySphere platform. Validation taking place on both Android and iOS mobile devices had the main focus on configurations with at least 1GB of RAM and octa core processors. Also, tests were done on regular web browsers running on normal desktop setups. These diverse tests were specially designed to confirm to the basic cross platform compatibility of PlaySphere and its performance in different hardware operation environments.

2.1.3 Database

Cloud Firestore, a flexible NoSQL cloud database, was chosen to store the structured user and content metadata in a real time environment. Its natural features such as strong scalability, powerful synchronization capabilities, and smooth native integration with Firebase Authentication [14], made it very appropriate for the dynamic data needs of a video, sharing application.

Alongside this, Firebase Storage was used to securely host and serve all the multimedia content, taking advantage of the built-in AES encryption feature to protect the files and ensure user data privacy compliance.

2.2 Methods

Before starting the work on the application described in this paper, we conducted a survey to find out how a secure video hosting platform meant for user privacy is expected by the users. The main goal was to obtain the user's insights that would be used for the guidance of design decisions and feature prioritization. The phase included participation of 18 students, independent content creators, and videophile individuals across various age groups. The survey tool was used to gather opinions through questions regarding the functionalities of the platform that the users would love most, the ease of navigation desired, concerns about data privacy being very critically considered, and realistic subscription models.

Key findings from this survey pointed out a number of the study direction by user preferences:

i. A preference for an ad free, uninterrupted video experience, characterized by minimum playback buffering.

ii. A preference for very simple, easy to use interfaces that improve overall usability on both mobile and web.

iii. The requirements for secure authentication processes and strong user data protection mechanisms in order to effectively secure user content.

iv. A tendency to choose a subscription-based model through which the users can not only get flexible access but also be able to enjoy specialized premium features.

2.2.1 System design and architecture

The conceptual design for PlaySphere is based on a modular and scalable architectural approach. This decision has been made in order to keep the platform stable over time, make it easy to maintain, and be open to extension in the future. The development process consists of well-planned iterative testing, together with the continuous integration of users' feedback at different development stages.

The installed architecture covers both frontend and backend functionalities. UI and frontend logic were mainly coded with Flutter, while Firebase [19] was used to provide the backend services required (e.g., authentication, database management, and media storage).

The user interface was very simple on purpose and was made to work just as well on different platforms so that the experience and performance would be consistent for Android, iOS users as well as web users. The main idea behind this work from the point of view of design are four things: responsiveness, a clean, minimal look, and universal accessibility.

The whole system of PlaySphere (see fig. 1) is made up of different interactive modules that together describe the user trip and the functional parts of the platform, thus:

a. **User entry layer** - The user interaction flow usually starts with the Onboarding Screen, which highlights the main application features and then leads the user to the Authentication Gateway. At this point, users can either sign in with their email/password credentials, use Google Sign-in, or create a new account through the Signup Screen. After a successful login, users go through Email Verification and Subscription Check, the latter being a method that serves to limit the main app interface access only to verified and active users.

b. **Core application layer** - Once inside, users are directed to the Home Screen, which is the central hub of the app from where users can access all the features, mostly through the Navigation Menu. From here, this layer divides into several major functional modules:

- Video screen: Responsible for displaying the range of available videos and leading users to the Single Video Screen and integrated Comment Section for specific user interactions.
- Search screen: Allows the user to quickly find a specific video or content creato.
- Add video screen: Makes uploading a video easier through Media Selection and the Confirm Screen, which are directly linked with the dedicated Storage components (Videos, Images, and Thumbnails).
- Message screen: A Chat Interface has been integrated for communication between platform users.
- Profile screen: Allows using the Public Profile, Edit Profile, Subscription Management, and the Withdrawal Setup Features that later connect to the specific Payment Methods and the related subscription services.

c. **Backend and storage layer** - The backend services of the application are operated by Firebase that handles user authentication, caters to real time data needs, and offers a secure media storage solution. To be specific, the video files, thumbnails, and images that are uploaded are protected in Cloud Storage, whereas the metadata and user account information are very well handled in Firestore Database.

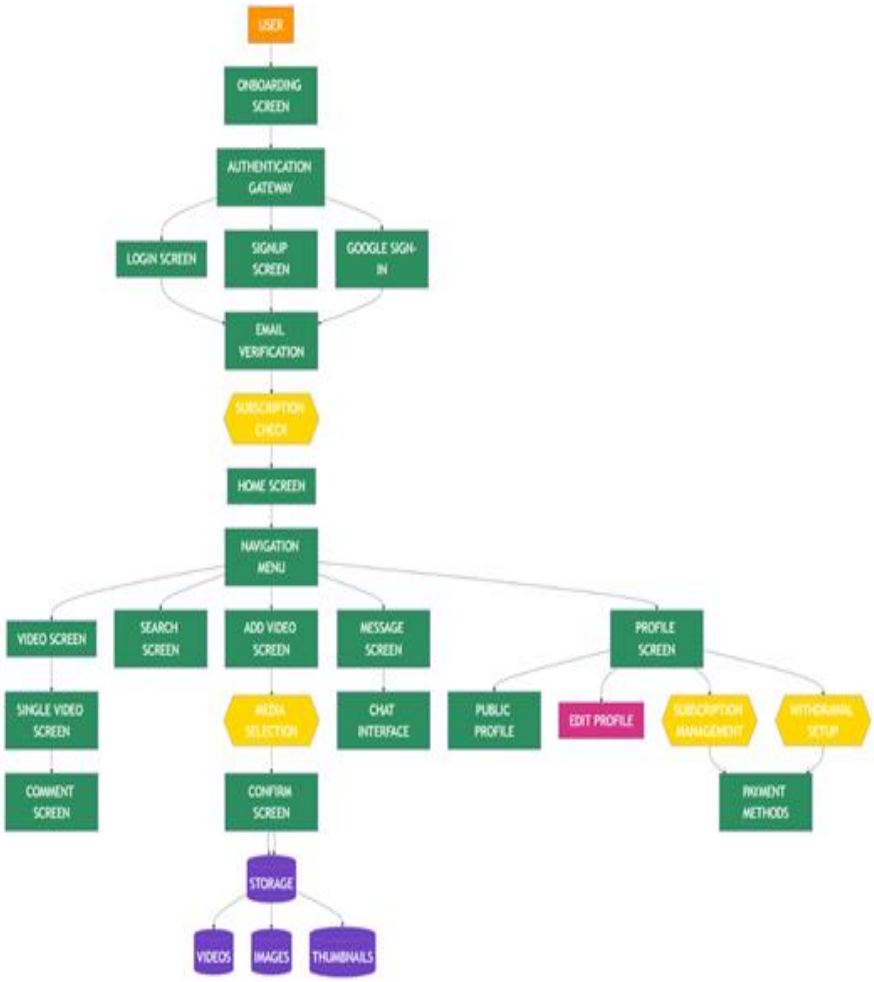

Fig. 1: Architectural model of the proposed application

d. **Integration and payment layer** - The architecture involves the integration of third-party services. In fact, the system relies on the Paystack API to manage the subscription processing, enable user withdrawals, and handle other necessary payment operations

2.2.2 Implementation procedures

This part covers the implementation plan of the app. The plan is designed to tackle a few issues that are commonly found as downsides on major video platforms. Among such issues are:

- Monetization challenges: One of the key measures taken was to bring down entry barriers that usually deter monetization by smaller creators. PlaySphere thus uses an access and entitlement framework strictly controlled by a verified subscription status:

a. Subscription, first eligibility: Users who possess a verified, active subscription are allowed to upload and potentially earn from content. This fundamental eligibility criterion is rigorously maintained by a two, step verification process:

i. Payment verification pipeline: Confirmed payment transactions are identified via Paystack webhooks that automatically update a subscriptions collection in Firestore. The record contains transaction metadata, the subscription plan level, the expiry date, and the important is_active flag.

ii. Security rules and server checks: Firestore Security Rules and custom Cloud Functions are set up to check the is_active attribute before permitting any write operations to content upload endpoints. In particular, upload requests made from a mobile app trigger a Cloud Function that checks whether the user is currently subscribed. In case the validation is successful, the user gets a short, lived signed upload URL securely returned.

b. Transparent revenue flows: PlaySphere is engineered to track very detailed information about transaction splits and essential payout metadata in Firestore. Through such a design, it becomes easy to support transparent reporting that creators can access nowadays. At the same time, it lays down the proper groundwork for the next automated payout working cycle after the financial criteria have been fulfilled.

- Content protection and anti-piracy measures: To secure the content and prevent piracy, PlaySphere has implemented several anti, scraping and anti, piracy measures that work together to limit in an effective way the possibility of unauthorized capture and subsequent distribution of protected videos:

a. Firebase security rules for object access: Both Firestore and Storage rules were set up to only allow authenticated requests and to tightly check that the user has the right role and subscription status before providing access permission for read or write. This goes a long way in stopping any kind of direct, unauthenticated access to the stored media objects.

b. Platform-level screen capture protections (native):

i. Android: Application uses a specialized Flutter plugin (visibility_detector) to set the FLAG_SECURE of the system window. This is done to prevent the operating system from recording the screen of the app at the content level.

ii. iOS: As iOS doesn't allow to fully prevent system, level screen recording through platform APIs, PlaySphere detects screen capture intents using iOS APIs (UIScreen.isCaptured). The app reacts by blacking out the video playback area if the user tries to view the content during the capture. These highly specific, platform, dependent behaviours are achieved through small native bridges, called MethodChannels, which are the part of the Flutter app architecture.

c. Mitigating reduced visibility from algorithmic bias: PlaySphere is trying to reduce issues arising from algorithmic bias and the common "cold, start problem" by intentionally designing the content discovery

process. It is done in a way that newer creators and those with very low exposure get a significant, guaranteed visibility together with the content that is ranked primarily by the standard relevance metrics. This approach to the problem from both sides integrates straightforward, deterministic rules with a fairness, aware ranking strategy not only to keep users engaged but also to improve the discoverability of less popular creators.

i. New uploads prioritization: Every newly uploaded video is automatically the first to be placed within one single, specific dedicated video feed. This channel guarantees that new content gets immediate visibility by presenting it with the newest first, regardless of its initial engagement level.

ii. Shuffle, and, fill policy: Besides the content that occupies the top, ranked recommendation spots, the rest of the recommendation space is filled by a deterministic shuffle of content with mid, to low, exposure category. This tactic is utilized in place of a purely highest, score ordering. By giving multiple creators the chance to be seen by different audiences repeatedly, this method helps to cut off the cycle of "rich, get, richer" amplification.

d. Domain, and app restricted playback: The main goal of the implementation is to make sure that the video content is played only through the designated PlaySphere clients or approved web domains. To that end, it relies on the use of signed tokens and thorough server, side checks:

i. Signed playback tokens: A signed JWT (JSON Web Token) representing a subscription status must be incorporated in any request for video playback. After verifying a user is an active subscriber, a Cloud Function issues this JWT which contains only the minimal necessary claims (like user ID, video ID, expiration time, device ID hash). The JWT is first validated by the playback endpoint prior to returning a signed streaming URL to the client.

ii. Token validation and short time, to, live (TTLS): Tokens are intentionally designed to have very short time, to, live and are valid for only one playback session. Whenever the application performs a playback refresh (e.g., when a user seeks or resumes), it must obtain a new token, so even if a token gets intercepted, it will have very little value.

iii. Playback SDK protections: The app relies on native video player implementations and customizes them to only accept signed URLs. It is also engineered to check the JWT before starting playback. This security measure is meant to stop embedment of streaming URL into unauthorised media players.

2.3 Performance evaluation activities

The PlaySphere app underwent a formal assessment procedure, which consisted of a comprehensive user survey as well as a comparative performance analysis. This method was implemented to thoroughly evaluate the app's usability, technical performance, feature effectiveness, and the level of user satisfaction it generated. The user survey, which was carried out through Google Forms, received over 50 replies from a well, balanced combination of content creators and regular viewers. The evaluation framework was built around four major parameters that were intended to provide a mix of qualitative and quantitative insights into the system's efficiency: usability and design, performance and functionality, features and security, and the overall user experience.

3. RESULTS AND DISCUSSION

As discussed in section 2.3, the result of the evaluation of PlaySphere is presented according to the metrics measured, followed by a sub-section on discussion of the results.

3.1 Results

a. Usability and design

The evaluation of PlaySphere's user interface and overall design received positive feedback. For instance, more than 70% of the survey participants agreed that the platform offers a clean, attractive interface together with a very intuitive navigation structure (see Fig. 2).

Additionally, the simplicity that users generally experienced during the steps of setting up an account, logging in, and video uploading was pointed out and greatly appreciated. These results together make PlaySphere look like a good, easy, to, access, and user, friendly alternative to many current video hosting platforms.

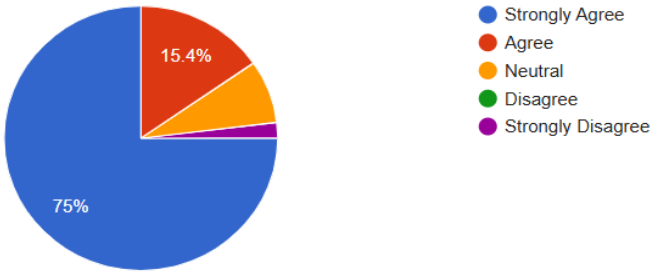

Fig. 2: Usability Evaluation Pie Chart

b. Performance and functionality

In terms of performance, most users claimed that uploading videos worked without a problem and playing them was smooth and without interruption. That being said, the assessment could single out only a slight drawback: only about 7.7% of those surveyed admitted to have encountered some moments in the app when it froze (see Fig. 3 for confirmation). Hence, such a result hints that there should still be a definite focus on refinement, thus a better user experience is made possible without compromising the platform's responsiveness and reliability even as the number of users grow and more operations are performed.

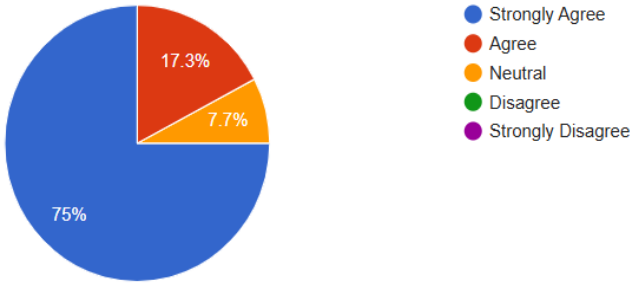

Fig. 3: Performance Evaluation Pie Chart

c. Features and security

The majority of people (76.9%) who took part in the survey indicated that they were convinced about PlaySpheres integrated content protection and anti, piracy features (refer to Fig. 4). In general, the platform was praised for its security, especially the strong authentication and data protection features.

Nevertheless, a minority of users came up with ideas for development. They thought that privacy could be strengthened and more detailed control measures concerning the misuse of content could be investigated and implemented.

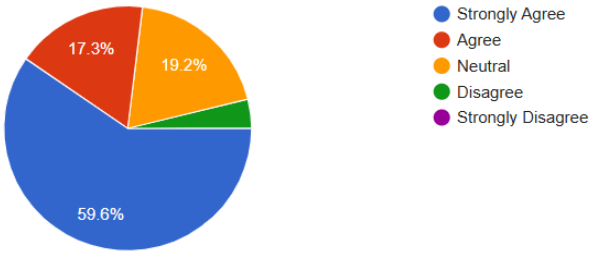

Fig. 4: Security Evaluation Pie Chart

d. Overall user experience

Participants overall showed evidence of high levels of satisfaction with the overall PlaySphere experience, with a large number of them praising specifically its ad, free viewing environment and its design that intentionally focuses on giving more power to the creators (see Fig. 5 for illustration). Most of the respondents said they would be likely to recommend the platform to others and considered PlaySphere to be a strong, safe, and truly user, friendly alternative to traditional video hosting services.

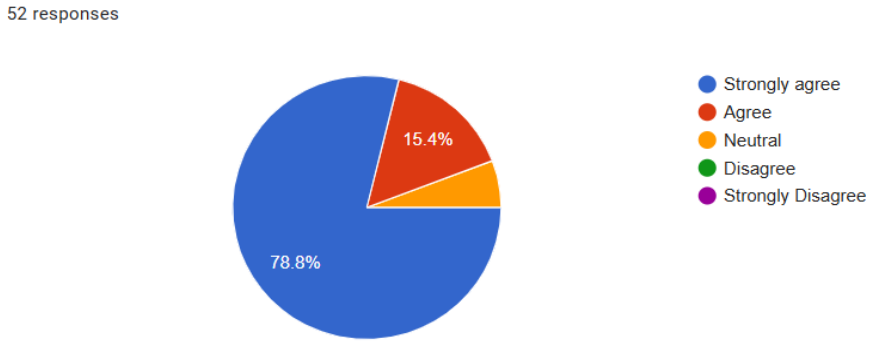

Fig. 5: User Experience Evaluation Feedback

e. Comparison analysis

In order to deepen the evaluation of the value proposition of PlaySphere, a comparison was made between PlaySphere and two well, known, popular video hosting platforms: YouTube and TikTok. The comparison was carefully organized and based on the important features and criteria that were logically obtained from the initial user feedback and from the evaluation survey.

The comparison was mainly focused on five key aspects: availability of ad, free viewing, overall user experience, level of content security, subscription model flexibility, and efficiency of community engagement features.

Table 1: Comparison analysis

Feature	**PlaySphere**	**YouTube**	**TikTok**
Ad-free viewing	Yes	Yes	Yes
Intuitive user experience	Yes	Mixed	Yes

Feature	PlaySphere	YouTube	TikTok
Secure content hosting	Yes	Yes	Yes
Subscription flexibility	Yes	Limited	Limited
Community engagement	High	High	Very High

3.2 Discussion

The main design decisions of PlaySphere were directed by the core goals of the platform: to provide video hosting that is safe, fair, and of high standard.This strategy was inspired by the major features of the traditional services that were criticized, especially the monetization challenges faced by the small creators, the problems of content leakage and piracy, and the lowered visibility due to algorithmic biases. Therefore, the implementation choices described in the preceding sections of this paper were aimed at finding a good compromise among security, ease of use, and streaming performance with the main focus being on creator fairness and building user trust.

With regard to monetization, PlaySphere has implemented an inclusive entitlement model that, among other things, eases the limitations faced by small creators. Moreover, by allowing only those content creators who have a subscription verification status to upload and monetize their videos, and carry out entitlement enforcing checks only on the server, side, the video sharing platform essentially makes sure that the creator who subscribes can post and later on monetize their content regardless of who they are. Besides that, the centralization of the entitlement verification process goes a long way in countering the risk of client, side spoofing and at the same time sets up transparent, easily auditable records which are a must for future payout automation. The referred to approach effectively eliminates the secretive thresholding and gatekeeping which, in almost all cases, are two features of competitor platforms, thus adding to the high overall satisfaction metrics (see Fig. 2) obtained through the user survey conducted.

Regarding content protection, this idea was first imagined and then implemented as a layered strategy without putting all their hopes on one single solution, a design compromise that worked very well when it came to the actual usage. In the storage area, the media files are stored encrypted by Firebase on the server, side with AES, 256, and the data transmitting is always encrypted with TLS.

Furthermore, PlaySphere requires the use of short, lived signed URLs and server, validated playback tokens. So, only authorized session holders can have access to the streams. To avoid leakage through recording capture, the app client combines visibility detection (visibility_detector) with platform, specific methods (e.g., Android FLAG_SECURE through a native bridge) to smartly hide the playback screen when it detects suspicious behavior.The complete outcome, which is backed up by the user feedback with 76.9% of them having confidence in the content protection (refer to Fig. 4), indicates a possible decrease in the piracy methods without significantly affecting the legitimate viewing experience.

We had to basically rethink the content discovery pipeline when we came to the point of solving reduced visibility due to algorithmic bias. PlaySphere intentionally mixes deterministic content placement with a fairness, aware ranking policy: a separate 'New Creators' channel offers guaranteed initial exposure for newly uploaded content, and creators get easy, to, understand analytics. This hybrid model helps to reduce cold, start penalties and the trend of 'rich, get, richer' amplification while still maintaining some level of relevance. The first evaluation data show that creators feel their content is more discoverable, and the distribution of impressions in the various runs supports the decision to put fairness on a par with engagement goals.

4. CONCLUSION

To sum up, PlaySphere is a working example of how a few layered engineering solutions, as touched upon earlier, can go a long way towards solving the major issues in the current video hosting platforms. The

performance overhead caused by these carefully planned strategies is quite low and the improvements in security, creator fairness, and user satisfaction are good enough to justify the overheads.

Evaluation by means of targeted functional testing and a user survey (n = 52) has shown that PlaySphere provides a hack, proof, nearly seamless, ad, free video, viewing experience. More than 70% of the platform users gave PlaySphere a positive rating for its usability, around 77% of them really trusted the content protection features, and the overall satisfaction as well as the intention to recommend the platform were very high.

Performance testing combined with field trials have revealed that operational overhead due to security and entitlement mechanisms is quite low. Occasional, but rare incidents of lag (around 7%) were recorded and are deemed as fixable through future engineering improvements. Taken together, these outcomes indicate that the tangible security and fairness advantages brought by PlaySphere more than compensate for the slight operational costs of server, side token issuance and access validation.Besides the overheads or operational costs, the evaluation results back the study's design decisions and pave a clear way for the scaling of the video sharing platform.

In essence, PlaySphere makes a compelling case that it is possible to develop a secure, fair, and user, friendly video hosting platform by leveraging modern cloud services and careful application design. The existing prototype has proven that creator, friendly monetization, effective anti, piracy measures, and fairness, aware discovery can be harmoniously combined with high usability and good performance, thus offering a solid basis for continuing the work of a scalable, production, grade service.

REFERENCES

[1] Zain A, Fan Y, Zhou J. On Demand Secure Scalable Video Streaming for Both Human and Machine Applications. Sensors. 2026; 26(4):1285.
[2] Bartolome A, Niu S. A literature review of video-sharing platform research in HCI. In Proceedings of the 2023 CHI conference on human factors in computing systems 2023 Apr. 19; pp. 1-20.
[3] Burgess J, Green J. 2018. YouTube: Online video and participatory culture. 2nd. ed. John Wiley & Sons.
[4] Cha M, Kwak H, Rodriguez P, Ahn YY, Moon S. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement.* 2007 Oct. 24; pp. 1–14. ACM.
[5] Manic M. Short-Form Video Content and Consumer Engagement in Digital Landscapes. Bulletin of the Transilvania University of Brasov. Series V: Economic Sciences. 2024; pp. 45-52.
[6] Ma R, Kou Y. " How advertiser-friendly is my video?": YouTuber's Socioeconomic Interactions with Algorithmic Content Moderation. Proceedings of the ACM on Human-Computer Interaction. 2021 Oct 18;5(CSCW2):1-25.
[7] Meng Z, Nansen B. Chinese Video Creator Identities-a Cross-Platform Social Media Perspective. PLATFORM: Journal of Media & Communication. 2022 Jan 1;9(1).
[8] Texeira T, Wedel M, Pieters R. Emotion-Induced Engagement in Internet Video Advertisements. Journal of Marketing Research. 2012; p.49.
[9] Mhalla M, Yun J, Nasiri A. Video-sharing apps business models: TikTok case study. *International Journal of Innovation and Technology Management.*, 2020; *17*(07), p.2050050.
[10] Raffoul A, Ward ZJ, Santoso M, Kavanaugh JR, Austin SB. Social media platforms generate billions of dollars in revenue from U.S. youth: Findings from a simulated revenue model. PLoS ONE. 2023; 18(12): e0295337. https://doi.org/10.1371/journal.pone.0295337.
[11] Chikwendu MI, Asianuba I. A Review on Billing and Revenue Share System Between YouTube, Other Digital Platforms and Content Creators for Online Communication. International Journal of Recent Engineering Science-IJRES. 2024;11.

[12] Bleier A, Fossen BL, Shapira M. On the role of social media platforms in the creator economy. International Journal of Research in Marketing. 2024 Sep 1;41(3):411-26.
[13] West SM. Data capitalism: Redefining the logics of surveillance and privacy. Business & society. 2019 Jan;58(1):20-41.

[14] Kaur I. Mobile Cloud Computing: Using Firebase Auth. International Journal of Computer Science and Mobile Computing. 2022 Jan 1;11(4):61-68.
[15] Milojković K, Živković M, Bačanin Džakula N. Agile multi-user Android application development with Firebase: Authentication, authorization, and profile management. InSinteza 2024-International Scientific Conference on Information Technology, Computer Science, and Data Science 2024 (pp. 405-412). Singidunum University.
[16] Priyanka Brahmaiah V, Jaswantth PV, Sri Likhitha D, Pallavi Sudha M. Implementation of AES algorithm. In International Conference on Information and Communication Technology for Intelligent Systems 2023 Apr 27 (pp. 161-171). Singapore: Springer Nature Singapore.
[17] Rifki MI, Syamia N. Message Security Application Using Mobile-Based AES Algorithm. Journal of Computer Science, Information Technology and Telecommunication Engineering. 2024; 5(2):595-606.
[18] Kinari SA, Funabiki N, Aung ST, Wai KH, Mentari M, Puspitaningayu P. An independent learning system for Flutter cross-platform mobile programming with code modification problems. Information. 2024;15(10):614.
[19] Semma AB, Ali M, Saerozi M, Mansur M, Kusrini K. Cloud computing: google firebase firestore optimization analysis. Indonesian Journal of Electrical Engineering and Computer Science. 2023; 29(3):1719-28.