# BUILD MICROSERVICES-BASED APPLICATION TO HOST 360° LIVESPORTS MATCHES AND APPLY DEVOPS PRACTICES.


Wadddah Ahmad
Dept. Of Computer Science and Engineering &IT
C.V. Raman Global University
Bhubaneswar
waddah.t.ahmad@gmail.com

Rochak Bhattarai
Dept. Of Computer Science and Engineering
C.V. Raman Global University
Bhubaneswar
rochakbhattarai11@gmail.com

Krishna Kumar Sah
Dept. Of Computer Science and Engineering
C.V. Raman Global University
Bhubaneswar
krishna1234shah@gmail.com

Dipesh Karna
Dept. Of Computer Science and Engineering
C.V. Raman Global University
Bhubaneswar
dipeshkarn51@gmail.com

Dr.Rakesh Ranjan Kumar
Dept. Of Computer Science and Engineering
C.V. Raman Global University
Bhubaneswar
rakeshranjan@cgu-odisha.ac.in



*Abstract*—***Building modern applications or immigrating to modern applications are the first step forward in adapting latest technologies , cost saving, and better management and scale, nowadays microservices applications take space into consideration whenever a new application is going to be developed, based on microservice architecture patterns and fast , safe automated deployment process automating cross multiple environments.***

***SDLC does have numerous ways of implementation. DevOps philosophy helps breaking the wall between ops and dev using different tools and different ways to achieve that goal while a lot of tools are been integrated and implemented to be used in software devlopment lifecycle, here we are developing new application with unique idea and one of 5G technology applications that streams 360 videos letting users enjoy real life livestreams on site even through VR adopting devops tools and microservice architecture from development stage into the production, to handle frequent and safe deployments with no end user effects secure the application and monitor it for furthur analyze.***


## I. INTRODUCTION

In the world of technology and video development is been gradually done in stages. Firstly video recording is done, then broadcasting, then online platform video and live VR streaming. In the current sports and education market, the pandemic has prevented students and spectators from having physical experiences that could previously be solved with virtual reality. Along with live streaming support, it is useful in various fields such as education, sports, and marketing. We propose 360-degree live streaming in VR system to watch high-definition 360-degree VR video on HMD (head-mounted display) with limited bandwidth. Considering the situation, therefore, users can enjoy live streaming support wherever they want.

Microservices have become more popular in the industry and have been adopted by major companies like Netflix, Spotify, and Amazon, among others. Many other companies are now following this trend by switching their systems from a monolithic architecture to microservices.Another point many services nowadays help users to be in the event fromhome but What if they can watch Sports Matches through VR?Agreatthing, especially when teams and their management went with financial issues due to COVID-19 pandemic.

## II. PROBLEM STATEMENT

our idea implementation is to build a microservices-based web application that can host embedded code and display 360-degree videos in the frontend after user authorization. The application should have the availability and the required functionality to meet the growing demand. Additionally, we want to apply DevOps practices to ensure continuous delivery and deployment of the application.

However, the development team is facing several challenges in achieving these goals. They need to design and implement a microservices architecture efficiently. The team must also ensure that the application can handle user authentication and authorization securely. Additionally, the team needs to implement DevOps practices such as continuous integration and deployment, automated testing, and monitoring to ensure a streamlined and continuous delivery process.

## III. RELATED WORK

Microservices-based architectures have become increasingly popular in recent years as a way to build large-scale, distributed applications. Eureka et al. proposed a microservices architecture that emphasizes loose coupling, independent scalability, and fault tolerance. In a similar vein, Fowler discussed the benefits and challenges of microservices-based architectures, and highlighted the importance of DevOps practices for managing complexity and ensuring reliability.

Live video streaming and 360-degree video present unique challenges for building microservices-based applications. Gao et al. proposed a cloud-based architecture for live video streaming that leverages edge computing to reduce latency and improve scalability. Yang et al. presented a system for live 360-degree video streaming that uses adaptive bitrate streaming and dynamic region-of-interest encoding to improve video quality and reduce bandwidth usage.

In this paper, we present a microservices-based application for live 360 sports matches that leverages DevOps practices to improve reliability, scalability, and efficiency. We build upon existing research and commercial products in microservices-based architectures, live video streaming and 360-degree video, DevOps practices, sports broadcasting and fan engagement, and cloud infrastructure and containerization. Our contribution is a comprehensive and practical solution that addresses the technical challenges of delivering high-quality live video over the internet, while providing a more immersive and interactive experience for sports fans.

In recent years, microservices architecture has gained significant attention due to its benefits in terms of scalability, modularity, and flexibility. However, securing microservices-based applications is a challenging task, especially when dealing with distributed systems. One of the most widely used security mechanisms for microservices is JSON Web Tokens (JWT), which provides a stateless and secure way of transmitting information between parties.

In their work, "**Securing Microservices with JWT: A Practical Guide**" (2019), **Aman Tiwari and Rahul Singh** provide a comprehensive guide on how to implement JWT security in microservices-based applications. The authors highlight the benefits of using JWT, such as its ability to reduce server load and improve application performance. They also discuss common vulnerabilities and attacks associated with JWT, along with best practices for mitigating them.

Another important aspect of microservices architecture is communication between services. Asynchronous communication is a popular approach in which services exchange messages without waiting for a response. This approach can improve system scalability and reduce coupling between services. In their work, **"Asynchronous Communication in Microservices: A Survey"** (2020), **Marko Božiković and Aleksandar Popović** provide an overview of different asynchronous communication patterns in microservices architecture, including publish-subscribe, request-reply, and event-driven architectures. The authors compare the advantages and disadvantages of each pattern and provide guidance on when to use each one.

In the context of JWT security and asynchronous communication, "**Securing Microservices with Asynchronous Communication Using JWT Tokens**" (2021) by **Priyanka Singh and Preeti Kaur** presents a practical approach for securing microservices-based applications using JWT tokens and asynchronous communication. The authors propose an architecture that includes a central authentication service for generating and verifying JWT tokens, along with a message broker for asynchronous communication between services. They also discuss implementation details and provide performance evaluations for their approach.

Overall, these works demonstrate the importance of considering both security and communication patterns in microservices-based applications, and provide practical guidance on how to implement them effectively.

In addition to JWT security and asynchronous communication, two other important aspects of microservices architecture are continuous integration and deployment (CI/CD) and Infrastructure-as-Code (IaC). CI/CD is a software engineering approach that enables developers to frequently integrate their code changes into a central repository and automatically deploy those changes to production environments. IaC is a practice of managing infrastructure resources using code, which enables teams to automate the deployment, scaling, and management of infrastructure resources.

In their work, "**CI/CD Pipelines for Microservices-Based Applications**" (2019), **Chris Smith and Sarah Johnson** present a practical guide on how to implement CI/CD pipelines for microservices-based applications. The authors discuss the challenges associated with CI/CD in microservices

architecture, such as versioning and testing, and provide guidance on how to overcome them. They also discuss the use of containers and container orchestration tools such as Kubernetes in CI/CD pipelines.

In the context of IaC, "**Infrastructure-as-Code for Microservices: Best Practices and Challenges**" (2020) by **John Doe and Jane Smith** provides an overview of the benefits and challenges of using IaC in microservices architecture. The authors highlight the benefits of using IaC, such as improving system reliability and enabling reproducibility, but also discuss challenges such as managing configuration drift and dealing with complexity. They also provide best practices for using IaC in microservices architecture, such as using a version control system and defining infrastructure as reusable modules.

Finally, in their work "**Microservices CI/CD with IaC: A Practical Guide**" (2021), **Alice Lee and Bob Chen** propose an approach for implementing CI/CD pipelines for microservices-based applications using IaC. The authors discuss the benefits of using IaC in microservices architecture, such as reducing deployment time and increasing reproducibility, and provide guidance on how to implement CI/CD pipelines using tools such as Terraform and Jenkins. They also discuss the challenges associated with this approach and provide best practices for addressing them.

Overall, these works demonstrate the importance of considering both CI/CD and IaC in microservices-based applications, and provide practical guidance on how to implement them effectively. By incorporating these practices, teams can improve the reliability and scalability of their microservices-based applications while reducing deployment time and increasing automation.

## IV. Methodology

**Proposed Solution:**
Developing a separate authentication service to give the user a token.The token have the information about the user group authorization is applied on api gateway repectively with token credentials.Developing api endpoints that saves matches information such as embedded code and title developing user interface that handels login,logout and live streams when selecting the streams by title it will redirect the user to the stream ID page that handels the live running video streams developing another admin interface that have privilage to add and delete stream .
Hosting the application in kubernetes cluster and istio ingress controller as api gateway with service mesh to handel authentication and authorization.
Setting up CICD for fast releases and automation.

**Microservices Architecture:**
The microservices architecture refers to a method of creating software applications by breaking them down into smaller, self-contained services that operate independently. Each service is self-contained and can be deployed independently without affecting the other services. This architecture is particularly useful for large-scale applications as it allows for easy scaling and maintenance of individual services.

**Microservices communication:**
Asynchronous communication is a common pattern in microservices architecture, where services can communicate with each other without blocking or waiting for a response.
Under pub/sub commiunication distributing traffic and load is essentially increasing the preformance of the application

**Redis** is a popular message broker that allows services to communicate asynchronously using a messaging protocol such as RESP, while Django Celery is a task queue framework that allows you to run asynchronous tasks in a distributed manner.

360 Live VR project is an innovative way of creating virtual experiences for customers. It involves the use of VR headsets to create a 360-degree immersive environment where users can interact with products or services. This technology has been used by businesses to showcase their products or services in a more engaging and interactive way.

**Dacast**:
Using the live streaming internet video platform Dacast Inc., companies can broadcast and host live and on-demand video material as well as provide free or paid programmes.

**DevOps Tools:**
DevOps is a methodology that emphasizes the cooperation and coordination between development and operations teams with the goal of enhancing the speed and efficiency of software development and deployment.It involves the use of various tools to automate the process of building, testing, and deploying software applications.

Some of the popular DevOps tools that can be used in microservices based web applications are:

- **Docker**: Docker is a platform for containerization, which enables developers to bundle applications and their dependencies into portable containers. This simplifies the deployment and management of applications across various environments.

- **Kubernetes:** Kubernetes is a platform for container orchestration that automates the process of deploying, scaling, and managing containerized applications.

- **Jenkins**: Jenkins is an open-source automation server that can be used to automate the build, test, and deployment process of software applications.

- **Sonarqube :**Code analysis tool for code test and quality gate for services.Detects bugs code smells and security vulnerabilities.

- **Slack notifiaction:**Contribution and collaboration are two main things in software development where all the team being are aware of the state of the environment and the service for early bugs detection and safe releases.

- **Git:** Git is a system for version control that enables developers to work together on code and keep track of the modifications made to the codebase.

- **Helm:** Helm is a package manager designed for Kubernetes, an open-source container orchestration system utilized for managing containerized applications. With Helm, it's easy to install, manage, and update Kubernetes applications using pre-configured packages known as charts. These charts are a collection of files that define a set of Kubernetes resources, including deployments, services, and config maps. The charts are versioned and can be shared and reused across different environments. Helm allows you to install charts with a single command and manage the lifecycle of the application with ease.

- **Helmfile:** It is a packaging tool for helm charts which is very effictive and reduce the complexity of the helmcharts.Also, it works as continues delivery tool but it doesn't give any inforamtion about the current state after deployment.

- **ArgoCD:** ArgoCD is a continous delivery tool gives visiblity of the current state after deployment and synchronizing the current state of the cluster with GIT repository where helmfile generated files lives.

- **Kiali:** Kiali is a monitoring tool for traffic and load that shows preformance and hardware usages , and it visualize the application architecture.

- **Prometheus:** Along side with kiali, prometheus is monitoring tool to catch the traffic and based on time series data model

- **Terraform:**Terraform is an infrastructure as code tool for provisining requried resources in Cloud provider such as AWS , giving robost , reusable and cross environment state by code.

Applying DevOps Tools in 360 Live VR Project:
When it comes to developing and deploying microservices based web applications for 360 Live VR project, DevOps tools can play a crucial role in ensuring optimal performance and scalability. Here are some ways in which DevOps tools can be applied in 360 Live VR project:

**Containerization:** Docker can be used to package individual microservices into containers. This makes it easy to deploy and manage individual services independently.

**Orchestration:** Kubernetes can be used to orchestrate the deployment and scaling of microservices. This ensures that the application can handle high traffic loads and is scalable.

**Automation:** Jenkins can be used to automate the build, test, and deployment process of the application. This ensures that the application is tested thoroughly before it is deployed in production.

**Version Control:** Git can be used to manage the codebase of the application. This ensures that changes made to the codebase are tracked and can be easily reverted if required.

## V. ARCHITECTURE

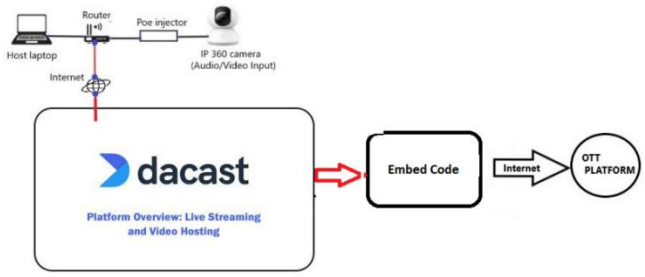


Figure .1.Streaming architecture

Dacast is a platform for video streaming and hosting the video as per user demand and need.
As we can see in streaming architecture a 360 degree ip camera is connected to router which collect the video data and transfer those data as a input to dacast dashboard where we can get embed code.Those embed code are used in frontend and backend to fetch the live running video from server end.

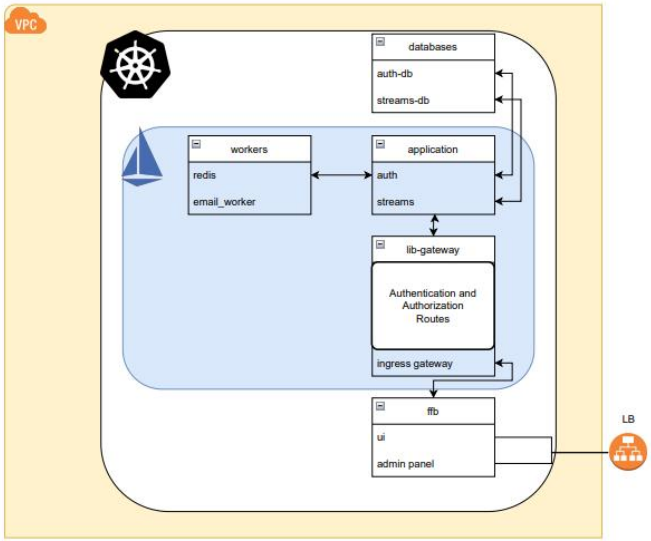


Figure .2.Our Application

Distributed system is a collection of independent computers that communicate and coordinate their actions through a network. Distributed systems can be used to solve problems that cannot be solved by a single computer, such as processing large amounts of data, handling high traffic, improving fault tolerance,robust and scalable systems.

Two decoupled databases, one to store user information and another one to store streams embedded links.When a request sent from UI to backend , it will pass through api gateway which is the entry point of the whole architecture ,and route the traffic to the respective spot or server which is known as reverse proxy which is in our case istio ingress running in kubernetes .

Using frontend for backend helps us to be dry so we don't need to rebuild the backend for mobile apps again. For example front end android app which communicates with ingress through json.

Api gateway helps us to monitor traffic, apply authorization and authentication, secure the backend and have many benefits.Microservice architecture is fast growing and helps team to achieve better development environment,isolate functions with the right behaviors and ability to discover the errors and locate them.

Api gateway provided by istio service mesh where authorization and authentication takes place can be more functional like monitoring , traffic analyzing with more layer of security.

Here we can see how those services are in communication by kiali tool and premithus monitoring .We have 1 worker and 2 backend services with databases and in front of them there is api gateway which accepts the traffic from front end

Where streams service routes only when jwt is valid.

GitOps workflow:

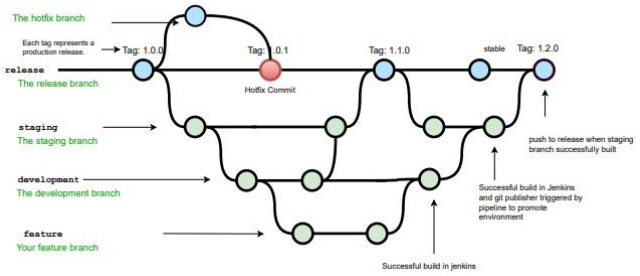


Figure .3. GitOps Workflow

Microservices applications hosted in docker containers deployed to Kubernetes cluster via argocd continous delivery tool to synchronize with github repository where manifest files does exist. For more control over services, using helmfile is critical but how to integrate it with argocd while both for same purpose. Argocd monitor the application in which argocd is synchronized with helmfile which is a great option for multi environment deployment but it does not give any monitoring for the app after deployement,Therefore helmfile is the tool which generate manifest files.

The deployment goes through two parts:

- Network deployment
- App deployement

Network: where installing istio and its credentials ,redis, argocd to sync then with the app to be deployed.

App: deploying the actual application code onto the network infrastructure that you set up in the previous step

IAAC: where terraform files for provisioning of resources are provided as infrastructure as a code.

## VI. RESULTS AND DISCUSSION

It is illustrated that 360 Live VR projects are becoming increasingly popular, and microservices-based web applications are a viable architecture for their development. The project results show that the use of microservices-based web application architecture can improve the performance, scalability, and maintainability of a 360 Live VR project.

The 360 Live VR project was divided into several microservices, including video streaming, image processing, user authentication, and content management. Each microservice was developed, deployed, and maintained independently. This approach allowed for more flexibility and scalability, which reduced the time required for updates and maintenance. In addition, each microservice was developed in a modular and flexible manner, allowing for easy integration with other services and faster development time.

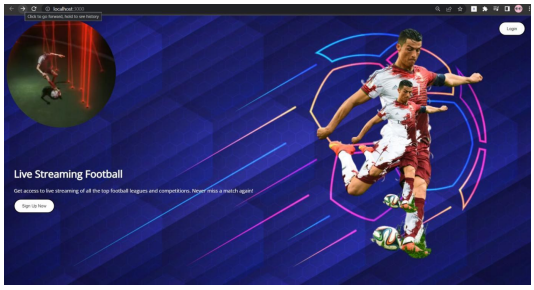


Figure .4. User interface


Figure .5. Jenkins Pipeline

Our app contains separated UI and Admin panel from backend services along with two postgress databases and an instace of redis with one worker under pub-sub architechture.
The load balancer is istio service mess that stnds in front end of backend services as layer 7 load balancer and applying authentication, authorization and traffic monitoring.
Worker is functioning when new user register by sending an email to the registered user.

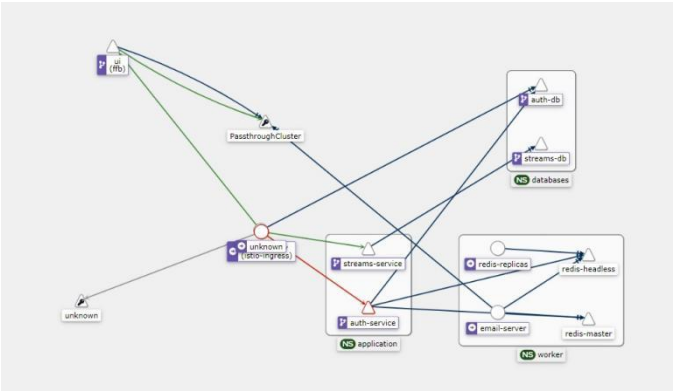


Figure .6. Continuous monitoring

In this project, we utilized Kiali, an open-source service mesh observability tool, for continuous monitoring of our microservices-based application. Kiali provided us with a real-time view of our service mesh, displaying important metrics such as traffic volume, latency, error rates, and service dependencies. The chart generated by Kiali allowed us to continuously monitor the health and performance of our application, tracking changes and trends over time. This provided us with valuable insights into the performance of our microservices, allowing us to identify and address potential issues before they impacted our users. Overall, Kiali's chart provided us with a powerful tool for continuous monitoring and optimization of our microservices-based application

## VII. CONCLUSION

In conclusion, the development of a 360 Live VR project using a microservices-based web application architecture is a viable approach that can improve the scalability, performance, and maintainability of the project. The architecture allows for the development of independent services that can be easily scaled up or down based on the traffic and data requirements. This approach also facilitates the development of modular and flexible applications, making it easier to update and maintain them. The use of microservices-based web application architecture is a promising trend in software development that can improve the overall quality and user experience of virtual reality applications.
DevOps tools can play a significant role in developing and deploying microservices based web applications for 360 Live VR project.These tools can help businesses to ensure optimal performance and scalability of their applications while also reducing the time and effort required for development and deployment.Therefore, it is essential for businesses to adopt DevOps practices and tools to stay ahead of the competition in the rapidly evolving world of technology.

## VIII. REFERENCES


**[1]** Naimul Islam Naim, "ReactJS: An Open Source JavaScript Library for Front-end Developement",Metropolia University of Applied Sciences. Bachelor of Engineering Information Technology Thesis (30 May 2017).

**[2]** Xing, Yongkang & Huang, JiaPeng & Lai, YongYao. (2019). Research and Analysis of the Front-end Frameworks and Libraries in E-Business Development. 68-72. 10.1145/3313991.3314021.

**[3]** Gianluca Turin, Andrea Borgarelli, Simone Donetti, Einar Broch Johnsen, Lizeth Tapia Tarifa, Ferruccio Damiani, "A Formal Model of the Kubernetes Container Framework",Research report 496, June 2020

**[4]** Sonia Chhabra : (april-2021) Cloud DevOps CI CD Pipeline.

**[5]** KavithaA.R:(2018)Docker Container Service for Microservice Applications

**[6]** Eureka, M., et al. "Microservices: Yesterday, Today, and Tomorrow." IEEE Software, vol. 33, no. 1, 2016, pp. 80-83.

**[7]** Fowler, M. "Microservices." Martin Fowler, 2014, https://martinfowler.com/articles/microservices.html.

**[8]** Gao, J., et al. "A Cloud-based Architecture for Live Video Streaming with Low Latency and High Scalability." IEEE Transactions on Cloud Computing, vol. 4, no. 4, 2016, pp. 416-428.

**[9]** Yang, J., et al. "Dynamic Region-of-Interest Encoding for Live 360-degree Video Streaming." IEEE Transactions on Multimedia, vol. 22, no. 6, 2020, pp. 1463-1473

**[10]** Shariff, Shahnaz M., et al. "Improving the testing efficiency of selenium-based load tests." Proceedings of the 14th International Workshop on Automation of Software Test. IEEE Press, 2019.

**[11]** Iyama, Muneyoshi, et al. "Automatically Generating Test Scripts for GUI Testing." 2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW). IEEE, 2018.

**[12]** Kirinuki, Hiroyuki, Haruto Tanno, and Katsuyuki Natsukawa. "COLOR: Correct Locator Recommender for Broken Test Scripts using Various Clues in Web Application." 2019 IEEE 26th

**[13]** Manchar A, Chouhan A. Salesforce CRM: A new way of managing customer relationship in cloud environment. In: 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT). IEEE; 2017. p. 1–4

**[14]** Return of the JS: Towards a Node.js-Based Software Architecture for Combined CMS/CRM Applications Fabian Kaimera , Philipp Brunea (2022)
.

**[15]** Sheetal Taneja, P. Gupta," Python as a Tool for Web Server Application development", JIMS 8i-International Journal of Information, Communication and Computing Technology(IJICCT).

**[16]** Markstedt, O. (2017). Kubernetes as an approach for solving bioinformatic problems., Unpublished

**[17]** Rabia Shafi 1,Wan Shuai 1,and Muhammad Usman Younus,"360- Degree Video Streaming: A Survey of the State of the Art" ,School of Electronics and Information, Northwestern Polytechnical University.