A REPORT

ON

Project Boxxy: Open Source GPU Remote Access & Development Suite

BY

JOEL TONY **2021A7PS2077G**

AT

Coditation Systems

A Practice School – I Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE
PILANI, GOA CAMPUS

July 2023

A REPORT

ON

Project Boxxy: Open Source GPU Remote Access & Development Suite

BY

JOEL TONY **2021A7PS2077G**

Prepared in partial fulfilment of the Practice School-I Course BITS F221

AT

Coditation Systems

A Practice School – I Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE

PILANI, GOA CAMPUS

July 2023

Abstract Sheet
BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI - (RAJASTHAN)

Practice School Division

Station: Coditation Systems	**Centre:** Pune, India
Duration: 30th May - 22nd July 2023	**Date of Submission:** 24th June 2023

Title: Project Boxxy: Open Source GPU Remote Access & Development Suite

Name of student: Joel Tony
ID Number: 2021A7PS2077G

Name of Experts	**Designation**
Mr. Shubham Kamthania	Technical Lead
Ms. Madhuri Pulipaka	Project Manager

Name of PS Faculty	**Designation**
Dr. Ravi Kadlimatti	Asst. Professor, Dept. of EEE

Keywords: Open Source, GPU, Remote, Graphics, Graphics Programming, WebGPU, WebRTC

Project Areas: Graphics Programming

Signature of Student **Signature of PS Faculty**

Acknowledgements

I would like to extend my special thanks to our mentor Dr. Ravi Kadlimatti, for being a constant support throughout the duration of the project. Also, I would like to thank Coditation Systems for providing me with the opportunity of working on an interesting project. Lastly, I would like to acknowledge the role of BITS Pilani in giving me this exposure through the Practice School-I course.

Contents

Chapter 1

Project Overview

1.1 Project Description

The project aims to develop an open-source sandbox environment for graphics payloads. It includes a WebRTC component for delivering graphics streams on the web, a native WebGPU component for real-time graphics rendering on the server, DevSync for real-time code editing, and a remote caching system for compiled resources.

1.2 Motivation Behind the Project

Project Boxxy aims to provide affordable and accessible GPU resources to developers and researchers on resource-intensive tasks. High-end GPUs are expensive and out of reach for individuals and organizations. The project aims to address that challenge by offering a sandbox environment for developing and deploying graphics payloads.

Chapter 2

Project Requirements

2.1 Functional Requirements

2.1.1 WebRTC

- Enable real-time delivery of graphics streams on the web.
- Support secure and efficient streaming of graphics data.
- Ensure compatibility with modern web browsers and platforms.

2.1.2 WebGPU

- Develop a native component for real-time graphics rendering on the server using WebGPU.
- Support efficient and optimized graphics rendering techniques.
- Handle shader languages such as GLSL or SPIR-V.

2.1.3 DevSync

- Enable real-time code editing, compilation, and hot reloading of the graphics viewport.
- Support a seamless development experience for developers working on graphics payloads.
- Integrate development tools and frameworks that facilitate live code updates and hot module replacement.

2.1.4 Compiler Infrastructure

- Implement remote caching of compiled resources to reduce the turnaround time for graphics compilation.
- Utilize distributed caching systems or cloud-based storage solutions for storing compiled resources.
- Optimize resource management and ensure efficient compilation processes.

2.1.5 Collaboration and Coherence

- Enable collaboration among developers and researchers working on the project.
- Utilize version control systems (e.g., Git) and project management tools (e.g., GitHub, Jira) for effective collaboration.
- Foster a coherent and collaborative end-to-end developer experience.

2.1.6 Open Source

- Follow open-source development practices and guidelines.
- Enable easy contribution from the community.
- Comply with open-source licenses and requirements.

2.1.7 Cloud Infrastructure

- Enable deployment of the remote access system using cloud-based infrastructure and services.
- Manage GPU resources efficiently within the cloud environment.
- Ensure scalability, reliability, and security of the cloud infrastructure.

2.1.8 Security

- Employ authentication and authorization mechanisms to control user access.
- Ensure secure remote access to the GPU-based applications.
- Protect sensitive data and prevent unauthorized access or data breaches.

2.1.9 Documentation and Testing

- Provide clear and comprehensive documentation for developers and users.
- Utilize testing frameworks and methodologies to ensure the reliability and quality of the applications.

2.2 Non-Functional Requirements

2.2.1 Performance

- Provide real-time graphics rendering and streaming with minimal latency.
- Optimize resource utilization for efficient compilation and caching processes.
- Support a scalable infrastructure to handle multiple concurrent users.

2.2.2 Usability

- Offer an intuitive and user-friendly interface for developers and researchers.
- Provide clear instructions and documentation for setup, configuration, and usage.
- Support seamless integration with popular development environments and tools.

2.2.3 Reliability

- Ensure high availability and uptime of the remote access system.
- Implement error handling and recovery mechanisms to handle exceptions and failures.
- Regularly monitor and maintain system stability and performance.

2.2.4 Security

- Implement robust security measures to protect user data and prevent unauthorized access.
- Follow security best practices and industry standards.
- Regularly update and patch the system to address security vulnerabilities.

2.2.5 Scalability

- Support a growing user base and increasing demands for GPU resources.
- Scale the infrastructure seamlessly to handle additional users and workloads.
- Optimize resource allocation and management for scalability.

2.2.6 Compatibility

- Support major web browsers and platforms for web-based delivery of graphics streams.
- Integrate with popular development tools and frameworks for code editing and compilation.
- Ensure compatibility with various operating systems and hardware configurations.

Chapter 3

Constraints

The Boxxy project requires open-source development practices and guidelines to comply with relevant open-source licenses and requirements. The project should utilize affordable infrastructure options to be cost-effective while ensuring compatibility with modern web tech and widely adopted development tools.

Chapter 4

Assumptions

4.1 Knowledge

Users should possess basic knowledge of graphics programming and web development concepts.

4.2 Resources

Sufficient GPU resources will be available for testing and development purposes.

4.3 Environment

The system will be deployed in a cloud environment with appropriate access and permissions.

Chapter 5

Risks and Mitigation Strategies

Risk	Mitigation Strategy
Insufficient community involvement and lack of contributions	• Implement effective community engagement strategies • Provide clear guidelines for contributions • Actively seek feedback from the community
Performance and scalability issues arise due to increased usage	• Regularly assess and optimize system resources • Allocate resources according to usage patterns • Introduce load balancing techniques to handle additional demands

Chapter 6

Conclusion

Project Boxxy is an open-source solution that develops a sandbox environment for GPU-based development and deployment. Its goal is to provide accessible and affordable resources that address the challenges related to high-end GPU costs and limited collaborative development experiences. Through diverse layouts and imagery, we could illustrate the project's functional, non-functional, and general requirements.