Design and Implementation of a Web-Controlled Real-Time 3D Visualization System Using Firebase and Unreal Engine

Muhammad Abdullah Ejaz¹, ᵃ⁾, Kanesaraj Ramasamy¹, ᵇ⁾, and Mirza Zaib Hassan², ᶜ⁾

¹ Multimedia University, Persiaran Multimedia, Cyberjaya, Malaysia

² Riphah International University, Sector I-14, Hajj Complex, Islamabad, Pakistan

*ᵃ⁾ Corresponding author: muhammad.abdullah.ejaz@student.mmy.edu.my*

*ᵇ⁾ r.kanesaraj@mmu.edu.my*

*ᶜ⁾ zaib.hassan@riphah.edu.pk*

**Abstract** The 3D real-time visualization systems are increasingly vital across diverse domains including media reporting, disaster management, crisis reporting and public information dissemination. However, controlling and updating these complex visualizations often requires specialized technical expertise. This study presents the design and implementation of novel web-based control system that enable non-technical users to manage photorealistic 3D data visualizations rendered in Unreal Engine. Utilizing a layered architecture, the system integrates a React JS web interface with a Node.js backend and Firebase real-time Database. Users can create and update data through intuitive web interface with changes seamlessly reflected in the Unreal Engine visualization in real-time. The core innovation of this study is the seamless integration of standard web technologies with an advanced rendering engine, enabling real-time broadcast quality visual updates to be controlled entirely through an accessible web-based front-end.

# Introduction

Real-time 3D visualization systems have become increasingly vital across various domains a wide range of fields, such as journalism, disaster response, infrastructure monitoring, and education. These systems offer dynamic rendering of data, and users receive instantaneous insights, thus improving the decision-making process. The use of web technologies and cloud platforms has helped in moving away from classic desktop-native visualization software towards browser-managed, distributed visualization systems.

Unreal Engine was first initially intended for the gaming domain but is now being used in non-gaming domains that include simulation, digital twins, and highly immersive data visualization [1]. Its high-fidelity rendering capability makes it highly suitable to work with real-time 3D environments, making some reasonable obstacles for controlling Unreal Engine using web interfaces in a modular and user-friendly way. Firebase Realtime Database has proven to be a light, cloud-native synchronization layer for live data streaming to many clients at low latency. Combining Firebase with Unreal Engine creates new horizons for browser-based control of intricate 3D worlds without requiring users to directly interact with the rendering backend.

Whereas past endeavors have merged web interfaces in real-time with 3D engines, no full modular system designs exist to let remote users interact and modify 3D visualizations from a web dashboard without the need for local control or platform dependency. This paper introduces the design and implementation of a new web-driven 3D visualization system, with input data from a browser interface synchronized through Firebase and displayed in real-time in Unreal Engine.

# Related Work

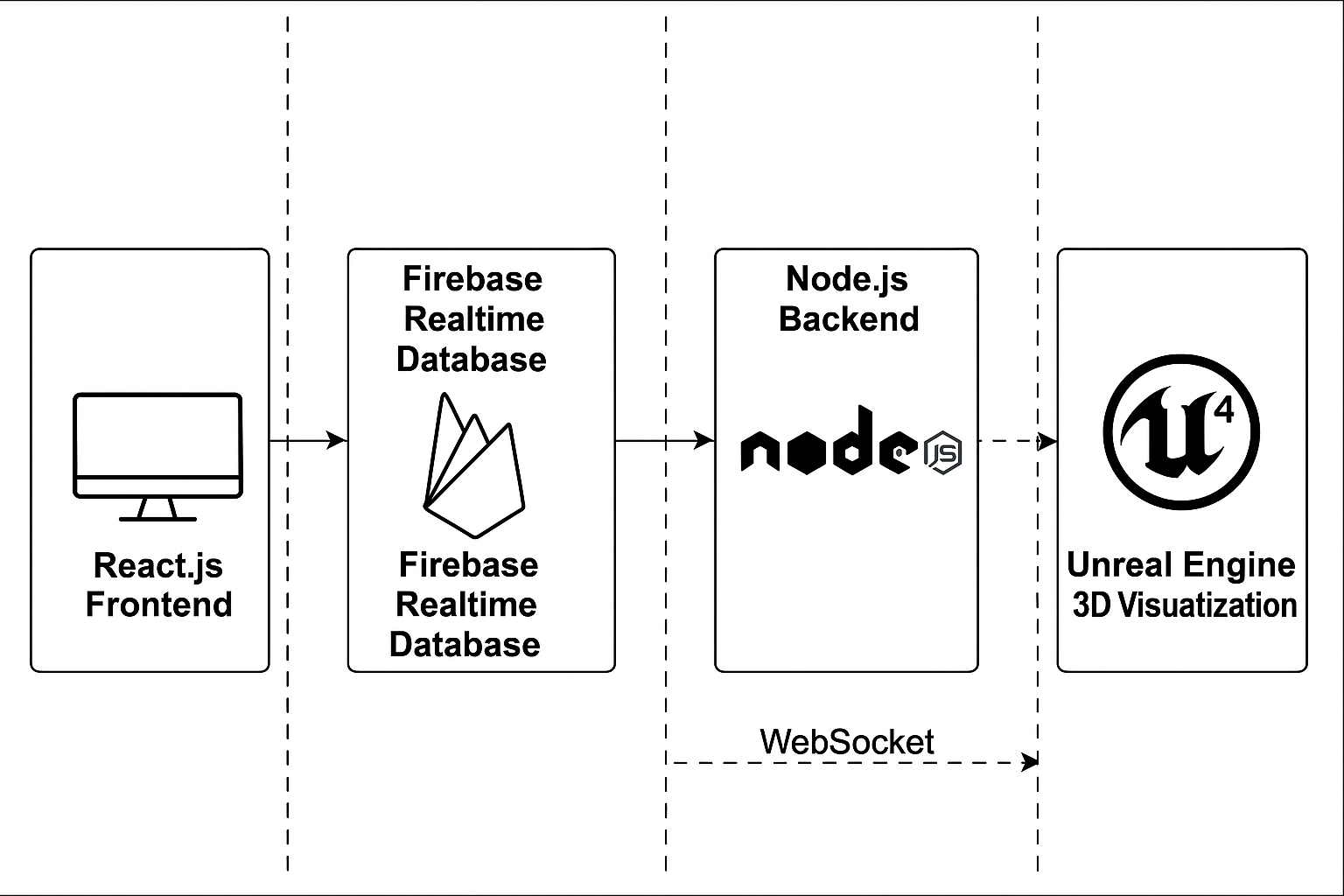
Several studies have explored the integration of web technologies with 3D rendering engines for real-time visualization Lv (2024) proposed a digital twin highway infrastructure system based on combining Unreal Engine with a Vue.js-based web interface using WebRTC and Node.js as real-time communication mediums. The system architecture they provide can handle dynamic interaction with geospatial models, proving the feasibility of web-based controls within immersive 3D worlds [2]. In the area of infrastructure observation, Liu (2021) presented a WebGL-based visualization of airport pavement condition based on Building Information Modeling (BIM) data. In this research, the feasibility of real-time rendering of complex 3D infrastructure models in an ordinary web browser was successfully presented  [3]. In areas of public data and emergency management, Onorati (2016) provided a real-time dashboard that treats Twitter data through ontology-based filtering. The system shows geotagged and grouped tweets through a real-time web interface, enhancing situational awareness in crisis incidents  [4]. For civic engagement and storytelling purposes, Sampat (2016) showed an interactive election data visualization application that integrates narrative aspects with interactive maps and charts. Though their framework is not focused on real-time synchronization or intensive rendering, it prioritizes accessibility and open communication in election data reporting  [5].

Nguyen (2025) developed an air quality monitoring system that utilizes deep learning models within a Node.js server. The platform provides forecasted and real-time environmental information through geospatial visualizations and time-series charts, demonstrating the integration of AI and web-based platforms for adaptive dashboards [6]. Zhang (2019) proposed a distributed telemedicine system for real-time sharing of medical images and collaborative diagnosis. Their solution is based on a web interface with both 2D and 3D visualization capabilities, satisfying the requirement for low-latency interaction in healthcare settings  [7].

Together, these works present the development of real-time data visualization environments based on web and immersive technology. None of them specifically address the exact issue of 3D, browser-managed visualization for live election data via a Firebase Unreal Engine pipeline, however. This is the new contribution of the current research.

# Methodology

The Proposed system is designed in a layered architecture [8] that make system acquire , synchronize and render real-time data into in 3d designed visualizations and mixed reality environments .Architecture of the system consist of four primary components (i) React.Js based frontend for CRUD operations (ii) Firebase Realtime Database (iii) Node.js backend server as a middleware between Unreal Engine and Firebase Realtime Database. Complete architecture is shown in Figure1.

****

**FIGURE 1.** System architecture

The system follows public-relay-render workflow. Frontend server as a control panel to create, read, update and delete data from firebase. Firebase serve as database and handling real-time events. Trigging data updated for all type of CRUD operations their events are immediately detected by react frontend and Node.js backend. Node.JS serve as middleware between unreal and firebase upon receiving data from firebase events Node.js construct and reorganize the data according to unreal engine readable formats and emit data using WebSocket to unreal engine. unreal engine uses same Node.js backend and uses unreal.js plugin to subscribe to Node.js WebSocket events. this data is sent to rendering engine to convert that Json based data into 3D visualization and displaying them into mixed reality environment in real-time using blueprint scripting. The layered architecture of the system makes each layer, independent, scalable, consistent, secure and suitable for low latency operations.

Data flow pipeline work in real-time by synchronization of user input from react based frontend to visual output from Unreal Engine. the sequence of the data pipeline is below

1. A user submits or modifies election data using the web interface shown in Figure 2.
2. The frontend writes this data to a designated path in Firebase Realtime Database.
3. The Node.js backend listens for updates using the Firebase Admin SDK.
4. When triggered, the backend formats the updated data into a JSON object and transmits it via WebSocket to the Unreal Engine client.
5. Unreal Engine parses the incoming data and modifies the active scene elements accordingly.

A screenshot of a voting application

Description automatically generated

**FIGURE 2.** Web based interface

All four technologies in the proposed system stack are proven to be a viable choice for real-time system , Reacts is popular choice in real-time systems due to its Component-Based Structure, Virtual DOM and One-way Data Binding [9].Firebase Realtime database offers Real-time synchronization is Cloud-hosted and scalable and offer JSON data format making data format suitable to parse and work in all major programming languages including C++ and unreal engine blueprints .NodeJS is carefully selected due to its event-driven architecture. Node.js uses an event loop to handle multiple concurrent requests efficiently making it a first choice of real-time applications like chat, gaming servers, collaborative and real-time environments [10]. Unreal Engine is used in this system due to its real-time rendering capabilities, ease of creating mixed reality by visualizing 3d virtual elements into camera recorded real environments [1].

All technologies are selected for this system create a balance of real-time performance, accessibility, and extensibility. A summary of the implementation stack is provided in Table 1.

**TABLE 1**. Implementation stack

|  |  |
| --- | --- |
| **Layer** | **Technology Used** |
| Frontend | React.js, Bootstrap |
| Synchronization | Firebase Realtime Database |
| Backend | Node.js, Firebase Admin SDK, ws (WebSocket) |
| Visualization | Unreal Engine 4.27+, Blueprints |

# CONCLUSION AND Future Work

Although the outlined system illustrates a working architecture for real-time 3D visualization via web-based control, several caveats still exist. To begin with, no formal performance benchmarking (e.g., latency, load capacity) has been incorporated in the current implementation, which is a critical step towards assessing scalability within high-traffic settings such as national elections. Second, integration with real-time data feeds (e.g., electoral APIs or newsroom CMS) has been simulated rather than field-tested. Future development will be geared towards improving the system with strong analytics, real-time data pipelines, and multi-user control capability, along with increased compatibility with mobile devices and VR/AR platforms.

# References

1. M. Krüger, D. Gilbert, T. W. Kuhlen, and T. Gerrits, “Game Engines for Immersive Visualization: Using Unreal Engine Beyond Entertainment,” PRESENCE: Virtual and Augmented Reality, vol. 33, pp. 31–55, Jul. 2024, doi: 10.1162/pres\_a\_00416.
2. “Data Management Framework for Highways: An Unreal Engine-Based Digital Sandbox Platform.” Accessed: May 05, 2025. [Online]. Available: https://www.mdpi.com/2075-5309/14/7/1961
3. “Integrating three-dimensional road design and pavement structure analysis based on BIM - ScienceDirect.” Accessed: May 05, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S092658051930768X
4. T. Onorati and P. Díaz, “Giving meaning to tweets in emergency situations: a semantic approach for filtering and visualizing social data,” Springerplus, vol. 5, no. 1, p. 1782, Oct. 2016, doi: 10.1186/s40064-016-3384-x.
5. “Visualization of election data: Using interaction design and visual discovery for communicating complex insights | JeDEM - eJournal of eDemocracy and Open Government.” Accessed: May 05, 2025. [Online]. Available: https://jedem.org/index.php/jedem/article/view/422
6. H. A. D. Nguyen et al., “A Deep-Learning-Based Visualization Tool for Air Pollution Forecasting,” IEEE Software, vol. 42, no. 02, pp. 47–56, Mar. 2025, doi: 10.1109/MS.2024.3496663.
7. “(PDF) Web-based medical data visualization and information sharing towards application in distributed diagnosis.” Accessed: May 05, 2025. [Online]. Available: https://www.researchgate.net/publication/328911885\_Web-based\_medical\_data\_visualization\_and\_information\_sharing\_towards\_application\_in\_distributed\_diagnosis
8. “1. Layered Architecture - Software Architecture Patterns [Book].” Accessed: May 01, 2025. [Online]. Available: https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html
9. J. Tong, R. R. Jikson, and A. A. S. Gunawan, “Comparative Performance Analysis of Javascript Frontend Web Frameworks,” in 2023 3rd International Conference on Electronic and Electrical Engineering and Intelligent System (ICE3IS), Aug. 2023, pp. 81–86. doi: 10.1109/ICE3IS59323.2023.10335250.
10. S. Tilkov and S. Vinoski, “Node.js: Using JavaScript to Build High-Performance Network Programs,” IEEE Internet Computing, vol. 14, no. 6, pp. 80–83, Nov. 2010, doi: 10.1109/MIC.2010.145.