

TECHNICAL REPORT: CINEVERSE SYSTEM ARCHITECTURE AND CLOUD DEPLOYMENT

Student Names: Muhammad Hazim(BSSE23059), Umair Safdar(BSSE23063)

Project Title: CineVerse: A Scalable Full-Stack Cinema Management System

Date: January 4, 2026

TABLE OF CONTENTS

1. Introduction	1
2. System Architecture	2
3. Cloud Infrastructure & Deployment	3
○ 3.1. Database Layer (Amazon RDS)	
○ 3.2. Backend Layer (Amazon EC2)	
○ 3.3. Frontend Layer (Amazon S3)	
4. Application Features & Interface	5
○ 4.1. Authentication and Authorization	
○ 4.2. Administrative Controls	
○ 4.3. Inventory & Booking Management	
5. Conclusion	8
6. References	9

LIST OF FIGURES

- **Figure 1:** Amazon RDS PostgreSQL Instance Configuration
 - **Figure 2:** Amazon S3 Static Website Hosting Bucket
 - **Figure 3:** Amazon EC2 Backend Instance Status
 - **Figure 4:** CineVerse User Login Interface
 - **Figure 5:** Admin Panel: Hall Management System
 - **Figure 6:** Admin Panel: Movie Catalog & Show Scheduling
 - **Figure 7:** Admin Panel: Booking Management Dashboard
 - **Figure 8:** Admin Panel: Food Inventory Management
 - **Figure 9:** Food Ordering System for Customers
 - **Figure 10:** Inventory Status and Low Stock Alerts
 -
-

1. Introduction

CineVerse is a full-stack web application designed to streamline cinema operations, including movie scheduling, ticket bookings, and cafeteria management. The system is built using a decoupled architecture, ensuring high availability and scalability through Amazon Web Services (AWS).

2. System Architecture

The project follows a **Three-Tier Architecture**:

1. **Presentation Tier:** Developed in React/Vite, hosted on Amazon S3.
2. **Logic Tier:** A Node.js/Express backend running on an Amazon EC2 instance.
3. **Data Tier:** A managed PostgreSQL database hosted on Amazon RDS.

3. Cloud Infrastructure & Deployment

3.1. Database Layer (Amazon RDS)

The system utilizes **Amazon RDS (PostgreSQL)**. As shown in the configuration, the database is set to **db.t3.micro** to balance cost and performance.

The screenshot shows the AWS RDS console for the 'cinema-db' database. The main summary panel indicates the database identifier is 'cinema-db', status is 'Available', role is 'Instance', engine is 'PostgreSQL', and it is located in the 'us-east-1d' region. The 'Connectivity & security' tab is selected, displaying details about the endpoint, port (5432), networking (Availability Zone: us-east-1d, VPC: vpc-0e1ef3095e8779ec4, Subnet group: default-vpc-0e1ef3095e8779ec4), and security (VPC security groups: rds-public-access-group (sg-0175e9b72d64abe040) - Active). The database is publicly accessible (Yes) and has a certificate authority (rds-ca-rsa2048-g1) and a DB instance certificate expiration date (January 02, 2027, 21:33 (UTC+05:00)).

3.2. Backend Layer (Amazon EC2)

The backend services are deployed on an **EC2 T3.micro instance**. This instance handles API requests, business logic, and database queries.

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with options like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Capacity Manager. The main area displays a table of instances. One instance is selected: **cinema-backend** (i-016d7f73eae4ff41e). The instance is **Running**, type **t3.micro**, and has **3/3 checks passed**. It is located in the **us-east-1c** Availability Zone. The Public IP address is **52.20.176.113** and the Private IP address is **172.31.4.207**. The Public DNS name is **ec2-52-20-176-113.compute-1.amazonaws.com**.

3.3. Frontend Layer (Amazon S3)

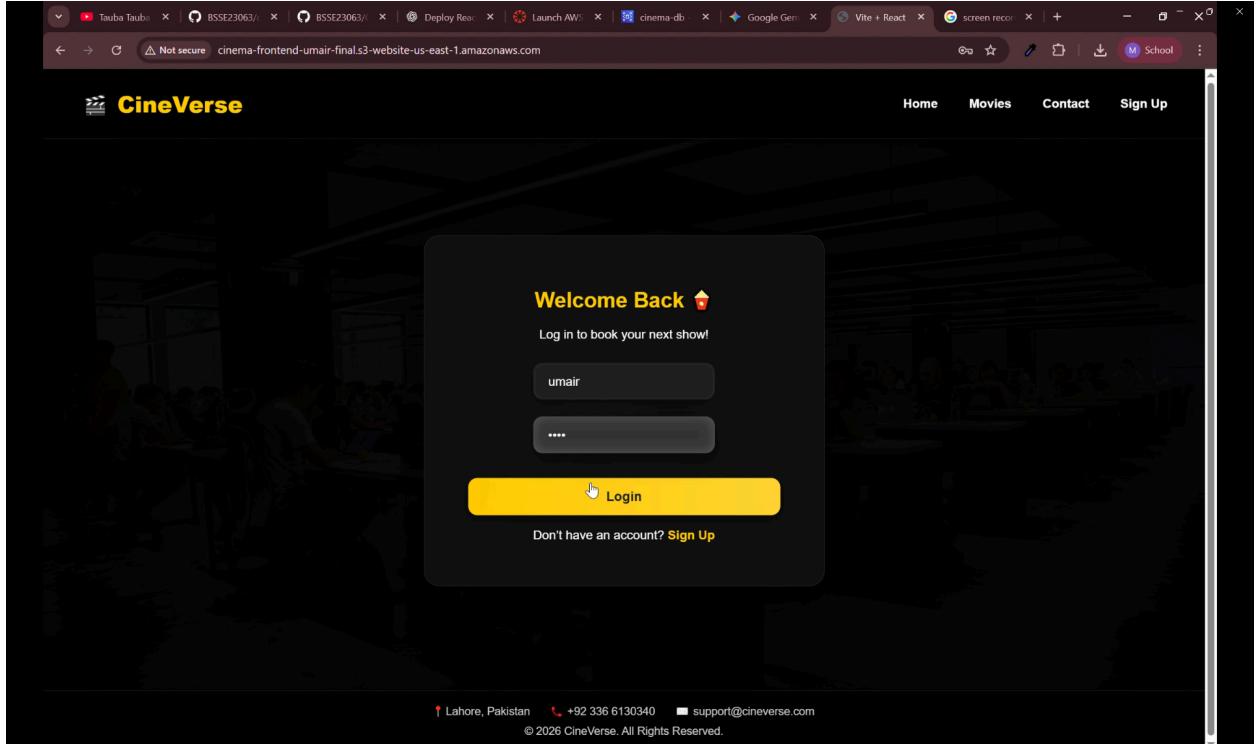
The React application is built and deployed as static files in an **S3 Bucket** configured for public website hosting.

The screenshot shows the AWS S3 Buckets page. The left sidebar includes options for Buckets (General purpose buckets, Directory buckets, Table buckets, Vector buckets), Access management and security (Access Points, Access Points for FSx, Access Grants, IAM Access Analyzer), Storage management and insights (Storage Lens, Batch Operations), Account and organization settings, and AWS Marketplace for S3. The main area shows the **cinema-frontend-umair-final** bucket. It contains two objects: **assets/** (a folder) and **index.html** (an HTML file last modified on January 3, 2026, at 15:01:55 UTC+05:00, with a size of 459.0 B and Standard storage class).

4. Application Features & Interface

4.1. Authentication

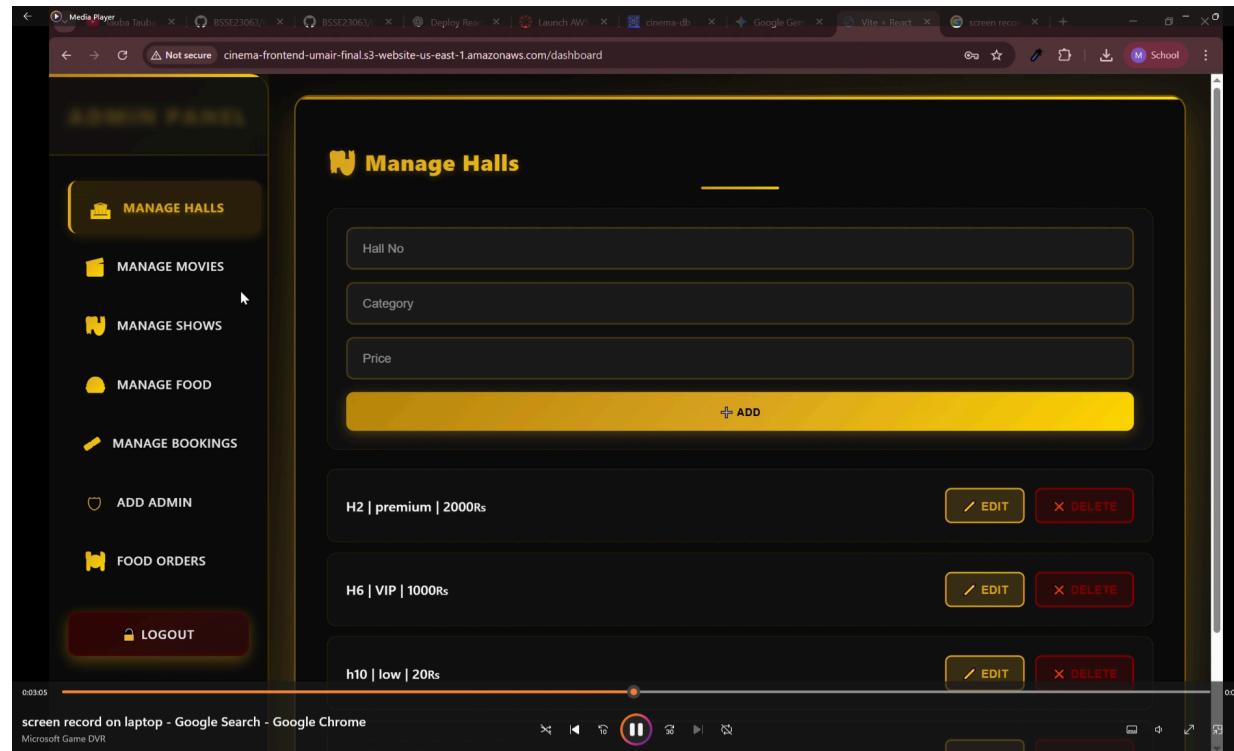
The entry point of the application is a secure login portal.



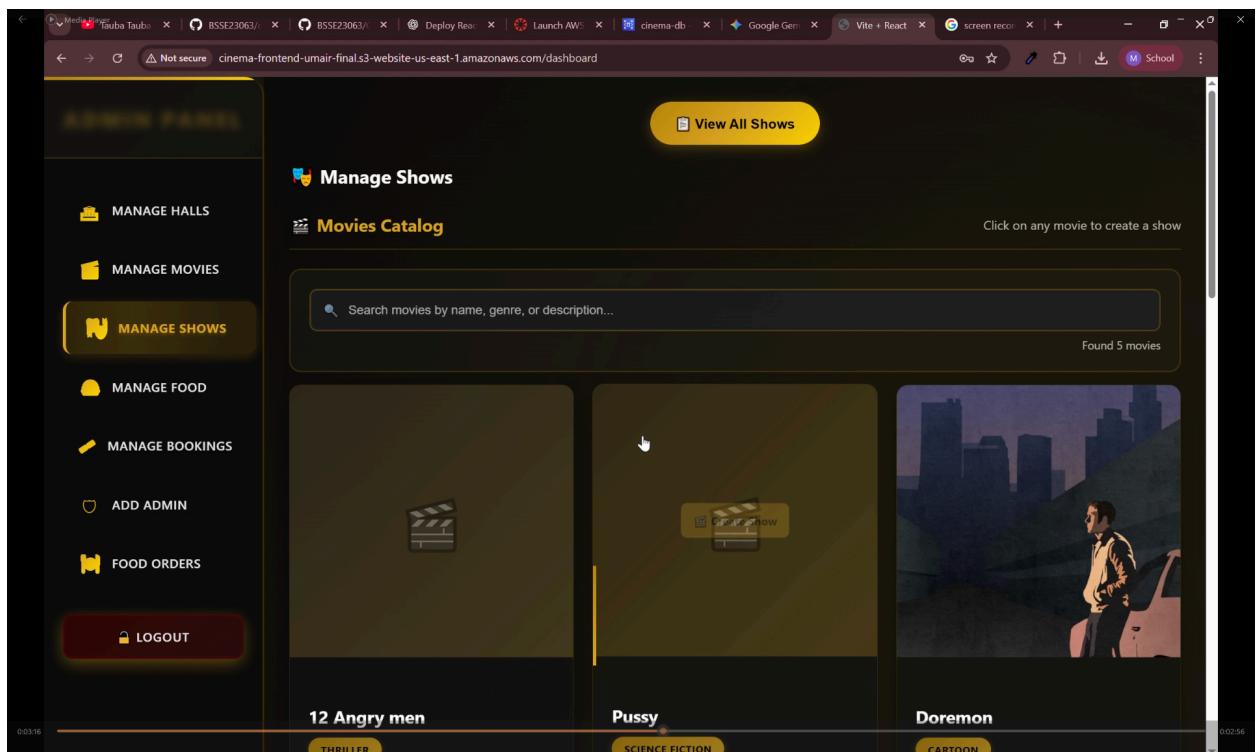
4.2. Administrative Operations

The Admin Panel allows for CRUD operations on cinema halls and shows.

- **Hall Management:** Defines hall categories (Premium/VIP) and pricing.



- **Show Scheduling:** Linking movies to specific halls and times.



4.3. Inventory and Booking

A unique feature of CineVerse is the integrated food management system, which tracks stock levels in real-time.

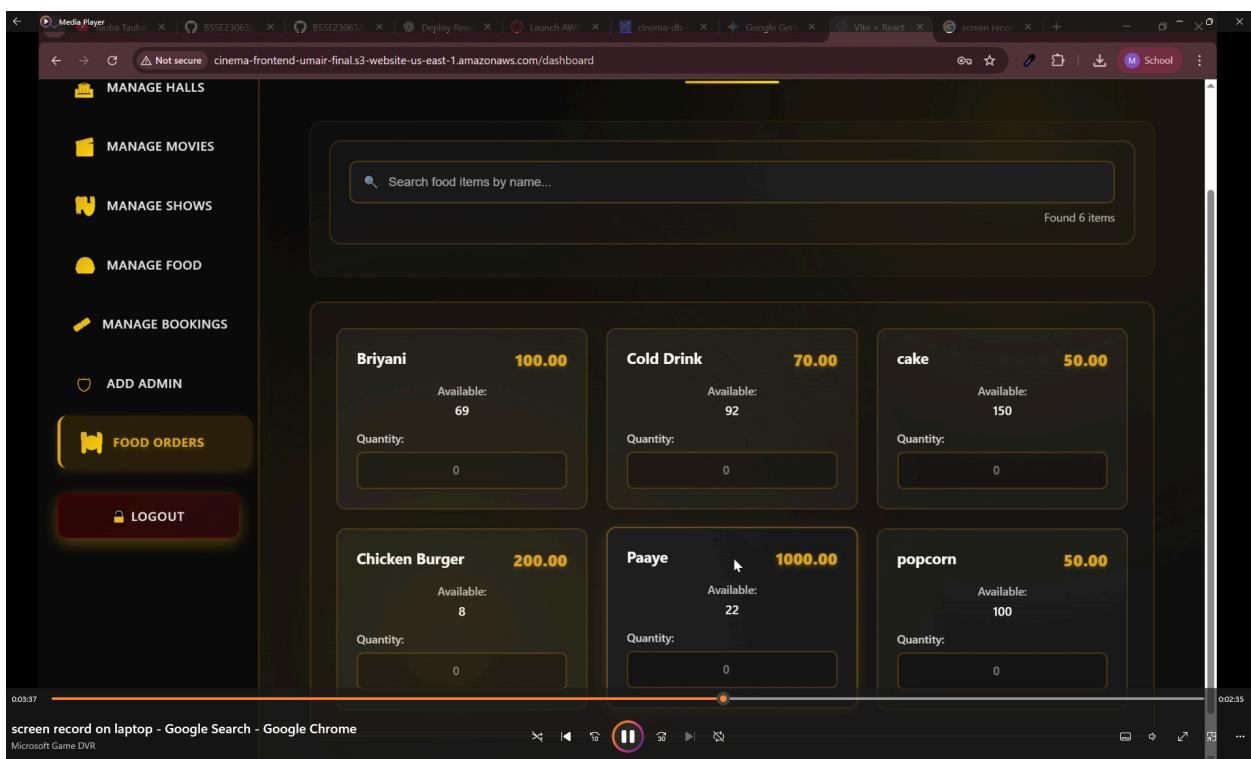
- **Food Management:** Tracking price and availability (e.g., "In Stock" vs "Low Stock").

The screenshot shows a dashboard for managing food items. There are six items listed in a grid:

- Biryani:** Quantity: 69, Price: 100.00, Status: IN STOCK
- Cold Drink:** Quantity: 92, Price: 70.00, Status: IN STOCK
- Cake:** Quantity: 150, Price: 50.00, Status: IN STOCK
- Chicken Burger:** Quantity: 8, Price: 200.00, Status: LOW STOCK
- Paaye:** Quantity: 22, Price: 1000.00, Status: IN STOCK
- Popcorn:** Quantity: 100, Price: 50.00, Status: IN STOCK

Each item card has "EDIT" and "DELETE" buttons. A "LOGOUT" button is visible on the left side of the dashboard.

- **Customer View:** A user-friendly grid for ordering items like "Biryani" or "Cold Drinks."



5. Conclusion

The CineVerse project successfully demonstrates the integration of modern web technologies with robust cloud infrastructure. By utilizing AWS RDS, EC2, and S3, the application achieves professional-grade deployment standards.

6. REFERENCES (APA)

- Amazon Web Services. (2025). *Amazon RDS User Guide for PostgreSQL*. Retrieved from <https://docs.aws.amazon.com/rds/>
- Amazon Web Services. (2025). *Hosting a static website using Amazon S3*. Retrieved from <https://docs.aws.amazon.com/S3/>
- React Documentation. (2025). *Scaling Frontend Applications with Vite*. Retrieved from <https://vitejs.dev/>