

# QuickCloud: Lightweight Cloud Infrastructure for VM Rental and Management

Asst. Prof. P. P. Nagarnaik<sup>1</sup>, Om Nasre<sup>2</sup>, Saksham Donge<sup>3</sup>, Isha Waghaye<sup>4</sup>, Mohnish Sasane<sup>5</sup>, Vaishnavi Hepat<sup>6</sup>

<sup>1,2,3,4,5,6</sup>*Artificial Intelligence, Priyadarshini JL of College Engineering*

\*\*\*

**Abstract :** The project, "QuickCloud," is designed to provide users with a web-based platform to rent and manage virtual machines (VMs) with ease. The system enables users to select and configure their desired virtual machines, access them via SSH, and pay for usage through a simple billing mechanism. The platform includes a user-friendly interface developed using Angular for the frontend, while Spring Boot and JPA handle the backend functionalities. Unlike traditional cloud services, this project focuses solely on providing EC2-like services, specifically allowing users to rent VMs with fixed configurations, without the complexities of a full-featured cloud platform. It offers essential functionalities such as user management, instance registration, billing, and instance configuration. The platform ensures efficient resource allocation and management, providing both an intuitive experience and robust technical control.

**Key Words - SSH, Spring Boot, SSH Key, Virtual Machines**

virtual machine rental with predefined configurations, streamlining the process for users seeking essential compute resources. With a frontend powered by Angular for enhanced usability and interactivity, and backend services managed by Spring Boot and JPA for secure data handling and API operations, the platform ensures a seamless user experience. Users can access VMs through SSH connections, track their usage through a simple billing system, and manage their instances efficiently. Key features include user authentication, instance management, billing integration, and resource monitoring. By combining simplicity with essential functionality, the QuickCloud provides both technical control and ease of use—targeting users who need reliable virtual machines without the overhead of a full-scale cloud infrastructure. This project is a practical solution for developers, small businesses, and individuals looking for a straightforward platform to rent and manage VMs with minimal effort.

## II. LITREATURE REVIEW

## I. Introduction

In recent years, cloud computing has revolutionized the way individuals and businesses manage their computing resources by offering scalable and on-demand infrastructure. However, many existing cloud platforms are often complex, requiring users to navigate through numerous configurations and services. To address this challenge, the "QuickCloud" project aims to provide a simplified, focused alternative—allowing users to easily rent and manage virtual machines (VMs) without the need for deep technical knowledge or navigating complex cloud environments. The QuickCloud is a web-based platform designed to deliver EC2-like services, enabling users to select, configure, and manage virtual machines through an intuitive interface. Unlike comprehensive cloud providers that offer a wide array of services, this platform focuses on

Cloud computing has significantly transformed the delivery and management of computing resources by enabling scalable, flexible, and on-demand access to virtualized infrastructures. The foundational concepts of cloud computing, including virtual storage and resource pooling, are well documented in earlier research, such as the review by El-Gazzar [1] which outlines the essential services offered by cloud platforms and emphasizes the role of virtualization in early cloud architecture development. As cloud services have evolved, the scope has expanded from basic storage to comprehensive resource management, including computing, networking, and virtualization. [2] Sharma and Gupta analyze the fundamental concepts of virtualization, discussing its significance in achieving hardware independence and scalability. [3] Zhang and Wang provide a detailed review of Virtual Machine (VM) consolidation techniques, emphasizing their impact on improving resource utilization and reducing energy consumption in cloud

data centers.

[4] Tang et al. propose a predictive and energy-aware VM consolidation strategy, demonstrating how intelligent prediction algorithms can optimize cloud infrastructure efficiency. The security aspect of virtualization has also been a crucial research focus.

[5] Patel and Gupta review various hypervisor-level security threats and countermeasures.

[6] Ksentini and Nogueira offer a classification of network slicing threats, which is critical for understanding potential vulnerabilities in multi-tenant cloud environments. The architecture and service model of commercial cloud platforms such as Amazon EC2 [7] have set benchmarks for VM provisioning and management. These services allow users to launch, configure, and manage VMs with minimal effort.

[8] Techniques like Secure Shell (SSH) for secure remote VM access are widely adopted in cloud computing platforms and are thoroughly documented by Microsoft. Building upon these foundational studies and technologies, the "QuickCloud" project aims to replicate and simplify core EC2 functionalities. Designed as a web-based platform, QuickCloud provides users with a straightforward interface to rent and manage virtual machines with predefined configurations. It is implemented using Angular for the frontend to ensure a responsive user interface and Spring Boot with JPA on the backend to facilitate efficient API integration and database management. Unlike complex and often overwhelming commercial platforms, QuickCloud is tailored for users seeking simplicity and usability. It is particularly suited for individuals and small to medium-sized enterprises (SMEs) that require essential VM services without extensive customization. By integrating intuitive features with secure SSH connectivity and automated instance management, QuickCloud reflects the practical application of cloud computing principles aimed at accessibility and user-centric design. In summary, the reviewed literature provides a solid foundation for understanding the core principles of cloud computing, VM management, security, and energy efficiency. The development of QuickCloud aligns with these principles while emphasizing a simplified and accessible approach, thereby contributing to the democratization of cloud computing services and promoting broader adoption among non-expert users.

### III. Objectives

The **frontend** of the platform serves as the user interface, providing an intuitive and user-friendly experience. Developed using HTML, CSS, and JavaScript (or optionally frameworks like React or Angular), it simplifies user interactions with the platform. The interface features a dashboard that displays critical information, such as available virtual machines, user profiles, billing details, and instance statuses. Additionally, a "Connect" button is included to trigger an automatic SSH connection to the allocated virtual machine, allowing users to manage their resources effortlessly.

The **backend** forms the core of the platform, handling APIs and logic to manage user requests and processes. Implemented in Java using Spring Boot and JPA, it supports critical functionalities like user management, virtual machine (VM) allocation, and billing management. The user management system oversees authentication, profile handling, and instance allocation, while the VM allocation APIs process user requests to allocate, connect, or terminate VMs. Furthermore, billing management logic ensures accurate tracking of VM usage and generates detailed billing information for users, ensuring the backend processes are efficient and reliable.

The **virtual machine management** component is the foundation of the cloud service, utilizing virtualization tools or manual setups across multiple CPUs. Key features include an automatic SSH connection that enables users to connect seamlessly to their VMs without the need for manual setup. Additionally, a data-wipe mechanism ensures user data is automatically erased after VM termination, maintaining privacy and security. This component empowers users to rent and interact with virtual machines while ensuring data integrity.

Lastly, the **networking and security** module focuses on establishing secure and efficient connections to the virtual machines. Using networking scripts, SSH protocols, and firewall configurations, it enables direct networking control, allowing users to manage network interfaces and IP allocations for their VMs. Port forwarding is implemented to facilitate secure access to VMs through specified ports, while SSH key management enhances security by ensuring encrypted connections. This module safeguards user data and

manages network interfaces effectively, contributing to a secure and robust cloud platform.

The QuickCloud project adopts a modular approach, emphasizing efficiency and simplicity in virtual machine (VM) management. The system is divided into four primary modules: **User Management**, **Instance Management**, **Billing**, and **Instance Configuration**, which work in unison to provide a seamless experience to end-users. Each module integrates modern tools and technologies to ensure robust functionality, scalability, and user satisfaction.

### User Management

The **User Management** module facilitates user registration, authentication, and profile management. Registration involves a simple form where users input their basic details, such as username, email, and password. The frontend, developed using Angular, ensures a responsive and intuitive user experience, while the backend, powered by Spring Boot and JPA, securely handles data storage. Passwords are hashed for security purposes.

Once registered, users can log in to access their profiles, allocate VMs, and view billing information. Authentication mechanisms like session tokens or JSON Web Tokens (JWT) provide secure communication between the client and server, safeguarding user sessions and API interactions.

### Instance Management

The **Instance Management** module lists available virtual machines and allows users to allocate them effortlessly. Instances are pre-registered in the system, with their statuses marked as either "free" or "allocated." Users can allocate a VM through a simple button click in the frontend, triggering backend logic to assign the VM to the user. This process ensures efficient VM management and real-time updates on instance availability.

### Billing

The **Billing** module tracks VM usage and calculates charges based on usage time. Each instance is associated with a unique timer that starts upon allocation and stops upon termination. The backend calculates the total usage time and applies a predefined rate (per hour) to generate the billing amount.

Billing details are automatically linked to the user's profile and stored in a relational database, ensuring easy retrieval and transparency. Users can view their updated bills at any time through the intuitive frontend interface, simplifying the billing process.

### Instance Configuration

The **Instance Configuration** module enables users to configure VMs according to their specific needs. Users can choose from predefined configurations, such as small, medium, or large instances, each mapped to distinct hardware specifications (e.g., 2 CPU cores, 4 GB RAM, 50 GB storage). This flexibility allows users to tailor the VMs to suit their requirements while maintaining a straightforward selection process.

### Tools and Platforms

The QuickCloud project leverages a comprehensive stack of modern tools and technologies to ensure a robust and scalable system:

#### 1. Frontend Technologies

- **Angular:** Utilized for creating a responsive and interactive user interface, Angular offers reusable components and handles key features like form validation, routing, and communication with backend APIs.
- **Bootstrap, CSS, or Tailwind:** Applied for styling, these tools ensure a visually appealing and responsive layout.

#### 2. Backend Technologies

- **Spring Boot:** A lightweight Java framework for developing RESTful APIs, enabling core functionalities such as user management, VM registration, and billing. It offers features like dependency injection and seamless database integration.
- **Java Persistence API (JPA):** Handles object-relational mapping and CRUD operations efficiently, ensuring smooth database interactions. Hibernate can be optionally used as the JPA implementation.

### 3. Database Management

- **MySQL/PostgreSQL:** Relational databases are employed to store user information, VM configurations, billing records, and instance logs. These systems ensure data consistency, scalability, and reliability for handling growing user data.

## IV. METHODOLOGY

The **QuickCloud** project is designed with a modular, scalable, and secure architecture for managing cloud-based virtual machines. Each module operates independently, enabling parallel development and easy maintenance. The system consists of four main modules—**User Management, Instance Management, Billing, and Administration**—which work together to provide a seamless VM provisioning experience.

---

### User Management

#### Module Objective

This module manages the complete user lifecycle, including registration, login, and profile management. Security and responsiveness are key through token-based authentication and role-based access control.

#### Registration & Authentication Workflow

Users sign up using Angular reactive forms that collect name, email, and password. Validation occurs on both frontend and backend. Passwords are securely hashed with `BCryptPasswordEncoder`. Upon successful login, the backend returns a **JWT** token that Angular stores locally. This token is automatically appended to future HTTP requests using interceptors, achieving a **stateless and secure** system.

#### Access and Profile Features

After login, users are redirected to dashboards based on their roles (USER or ADMIN). Angular uses guards and `*ngIf` directives, while the backend enforces role security using `@PreAuthorize`. Users can manage instances, view bills, and update profiles from their personalized dashboards.

### Instance Management

#### Objective

This module handles VM lifecycle operations, including allocation, running, and termination. It supports real-time updates, efficient usage of resources, and secure SSH-based access.

#### Allocation Workflow

Users browse available VMs and allocate one via the Angular UI. A POST API request allocates a VM from the pool, updates its status to “allocated,” and returns SSH credentials. Allocation time is logged, and billing is triggered. Angular reflects changes in real-time with two-way binding and toast alerts.

#### Termination and Deallocation

When users terminate VMs, the backend updates the status to “free,” logs the termination time, and stops billing. Admins can force-terminate instances in emergencies. Instance metadata includes allocation time, user ID, and status. Real-time updates are reflected in Angular through polling or WebSocket connections.

---

### Billing

#### Objective

The billing module computes charges based on how long each VM is used. It ensures transparent, accurate, and secure handling of user payments.

#### Usage Tracking and Computation

Billing starts on VM allocation and ends on termination. The duration is multiplied by a fixed rate (e.g., \$0.05/hr) to calculate the cost. Spring Boot handles logic via the service layer and updates records using transactional operations. Angular displays billing data in tabular form on the user’s dashboard.

#### Invoice Generation and History

Invoices are automatically created after each VM session. They are stored and linked to users using JPA. Each invoice includes VM name, duration, rate, and total amount. Users can view or download their bills as PDFs. Angular supports pagination and sorting for easy navigation.



## Secure Payments and Storage

A secure payment gateway is integrated into the system. On successful payment, the invoice is marked as paid and transaction data is securely stored. Atomicity ensures no partial changes in case of failure. Spring Scheduler is used to notify users of pending payments via emails or UI alerts.

## Administration and System Oversight

### Objective

This module grants administrators complete control over users, virtual machines, and billing data. It supports proactive system monitoring and critical interventions.

### Admin Panel Features

Admins can:

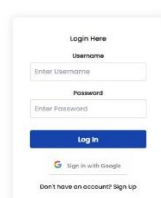
- View all users and VMs (free, allocated, or down)
- Force terminate or reassign VMs
- Review global billing logs
- Analyze system activity through filters and analytics

The admin dashboard is built with Angular and protected by method-level security in the backend. All critical actions are logged using **Spring AOP** for audit trails.

## Monitoring and Reporting

Admins monitor real-time activity via WebSockets or long-polling. They can track live sessions, flagged errors, and resource usage. Scheduled reports are automatically generated and stored to help maintain system reliability.

### 1. Login Page

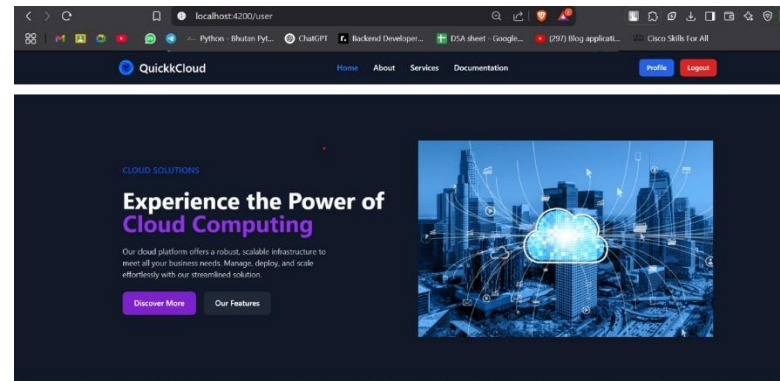


A login form with fields for Username and Password, a Log In button, and a link to sign up for new users.

Description:

*The login interface allows secure user authentication with input validation and token generation.*

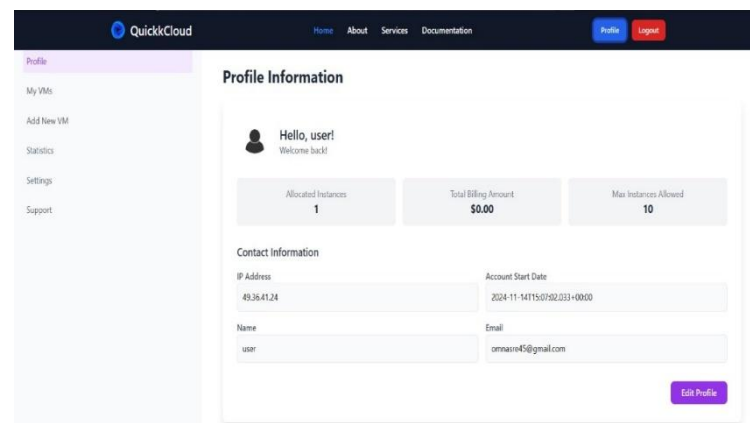
### 2. User Home Page



Description:

*This is the landing screen after login, showing summary actions and welcome info.*

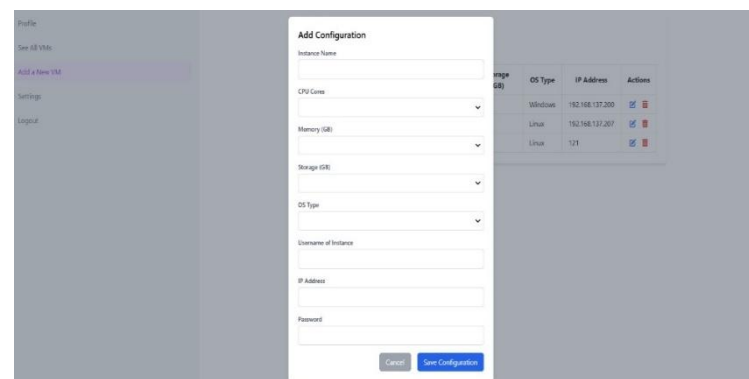
### 3. User Dashboard



Description:

*Displays VM counts, billing snapshots, and quick-access cards for configuration.*

### 4. Add New VM

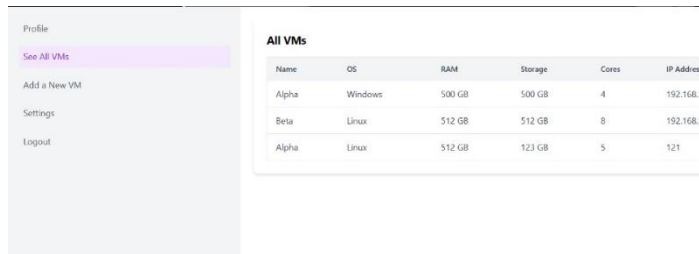


A screenshot of the 'Add New VM' form. It includes fields for Instance Name, CPU Cores, Memory (GB), Storage (GB), OS Type, Username of Instance, IP Address, and Password. There are 'Cancel' and 'Save Configuration' buttons at the bottom. In the background, a table lists existing VMs with columns for Name, OS Type, IP Address, and Actions.

Description:

UI component for VM allocation—lets users select a machine and provision it.

## 5. Admin - See All VMs



Name	OS	RAM	Storage	Cores	IP Address
Alpha	Windows	500 GB	500 GB	4	192.168.11
Beta	Linux	512 GB	512 GB	8	192.168.13
Alpha	Linux	512 GB	123 GB	5	121

Description:

Admin view listing all VMs with sorting, filters, and status indicators.

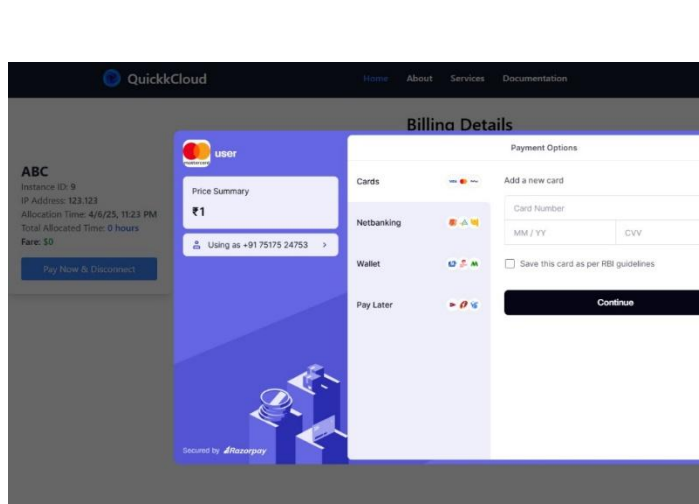
## 6. Billing Details



Description:

Shows the invoice breakdown and billing history per user session.

## 7. Payment Gateway



Description:

Secure screen where users enter card info or use e-wallets to complete payments.

## ACKNOWLEDGMENT

We want to start by sincerely thanking Er. Paritosh Nagarnaik at Priyadarshini JL College of Engineering for him superb advice and valuable suggestions that assisted us throughout this endeavor. The team members appreciate the help and materials provided by the Department of Artificial Intelligence at Priyadarshini JL College of Engineering.

## REFERENCES

- [1] R. F. El-Gazzar, "Review Paper on Cloud Computing," *L'ARCHIVE HAL*, 2014.
- [2] N. Sharma and P. Gupta, "A Study on Virtualization and Virtual Machines," *International Journal of Engineering Science and Innovation (IJESI)*, vol. 7, no. 5, pp. 51–55, May 2018.
- [3] K. Zhang and J. Wang, "A Comprehensive Review of Cloud Computing Virtual Machine Consolidation," *IEEE Access*, vol. 11, 2023.
- [4] H. Tang, H. Shao, D. Xu, and X. Li, "Research on Virtual Machine Consolidation Strategy Based on Combined Prediction and Energy-Aware in Cloud Computing Platform," *Journal of Cloud Computing*, vol. 11, article no. 50, 2022.
- [5] S. Patel and R. Gupta, "A Review Paper on Hypervisor and Virtual Machine Security," *Journal of Physics: Conference Series*, vol. 1950, *International Conference on Mechatronics and Artificial Intelligence (ICMAI)*, 27 Feb. 2021, Gurgaon, India.
- [6] A. Ksentini and M. Nogueira, "Classification of Network Slicing Threats Based on Slicing Enablers: A Survey," *International Journal of Intelligent Networks*, vol. 4, pp. 103–112, 2023.
- [7] Amazon Web Services, "Amazon EC2," [Online]. Available: <https://aws.amazon.com/ec2/>
- [8] Microsoft Docs, "SSH Connection Techniques," [Online]. Available: <https://learn.microsoft.com/en-us/windows/terminal/tutorials/ssh>