

# Project Progress Report for Yushan Platform

Practice Module for Certificate in Architecting Scalable Systems

Team 18

Members:

- Ahan Jaiswal
- Nguyen Phu Truong
- Yang Shuang
- Zhang Yan
- Zhu Yuhui

## CONTENTS

1. INTRODUCTION.....	3
1.1 Project Name & Description.....	3
1.2 Project Methodology .....	3
1.3 Project Summary.....	3
2. PROJECT PROGRESS REPORT .....	4
2.1 Reporting period (from – to Date).....	4
2.2 Sprint # .....	4
2.3 Sprint objectives .....	5
2.5. Sprint Burn down chart.....	9
3. PROJECT TEAM MEMBERS CONTRIBUTION SUMMARY.....	13
4. <OTHER THINGS TO BE HIGHLIGHTED> .....	6

# 1. Introduction

## 1.1 Project Name & Description

**Yushan - Home of Stories (Microservices Architecture)** is a cloud-native, microservices-based web novel reading platform designed for scalability, resilience, and independent service evolution. This project represents the architectural transformation of the original monolithic Yushan application into a distributed system leveraging domain-driven design (DDD), event-driven communication, and modern DevOps practices.

## 1.2 Project Methodology

The project follows **Agile Scrum methodology** with a single intensive sprint:

- **Sprint Duration:** 10 working days (October 13-24, 2025)
- **Planning Approach:** Domain-Driven Design workshops and service boundary identification
- **Development Model:** Multi-repository microservices with independent CI/CD pipelines
- **Quality Gates:** Comprehensive Definition of Ready and Definition of Done applied to all user stories

## 1.3 Project Summary

**Yushan** aims to revolutionize the web novel reading experience by combining traditional storytelling with modern gamification elements, creating an engaging ecosystem for both writers and readers. The ASS module project aimed to re-architect the Yushan monolithic application into a scalable, distributed microservices ecosystem while preserving and enhancing core platform functionality.

### **Architecture Transformation Goals:**

- Decompose monolith into 5 core microservices based on bounded contexts
- Implement independent deployment pipelines for each service
- Establish event-driven communication patterns using Apache Kafka
- Deploy to cloud infrastructure (DigitalOcean) with full observability
- Achieve enterprise-grade DevSecOps practices

### **Five Core Microservices:**

1. User Service - Identity & Access Management
2. Content Service - Novel & Chapter Management
3. Engagement Service - Social Interactions (Comments, Reviews, Reports)
4. Gamification Service - Rewards, EXP, Leaderboards
5. Analytics Service - Reading History, Rankings, Dashboards

### Supporting Platform Services:

- API Gateway (Spring Cloud Gateway)
- Service Registry (Eureka)
- Config Server (Spring Cloud Config)

**Total User Stories Delivered:** 24 across 5 EPICs

**Total Tasks & Subtasks:** 166 tracked in JIRA

### Epic 1: User Service.

### Epic 2: Content Service.

### Epic 3: Engagement Service.

### Epic 4: Gamification Service.

### Epic 5: Analytics Service.

### Epic 6: Documentation, QA & Misc Tasks.

Spaces / Yuhan Micro

YM board

Summary

Timeline

List

Backlog

Active sprints

Reports

Calendar

Basic

JQL

Search work

Project = Yuhan Micro

Assignee

Type

Status

More filters

Save filter

Group

Filter

Work	Assignee	Reporter	Priority	Status	Resolution	Created	Updated	Due date
<div><div></div><div><div></div><div>YM-5</div></div><div>Analytics Service (Ranking, History &amp; Reporting)</div></div>	<div><div></div><div>Alan Jalil</div></div>	<div><div></div><div>Alan Jalil</div></div>	<div><div></div><div>Medium</div></div>	<div><div></div><div>Unresolved</div></div>	<div><div></div><div>10/13/2025, 4:21 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:21 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:30 PM</div></div>	<div><div></div><div>Oct 24, 2025</div></div>
<div><div></div><div><div></div><div>YM-4</div></div><div>Gamification Service (Rewards &amp; Engagement M...)</div></div>	<div><div></div><div>sherry Y</div></div>	<div><div></div><div>Alan Jalil</div></div>	<div><div></div><div>High</div></div>	<div><div></div><div>Unresolved</div></div>	<div><div></div><div>10/13/2025, 4:21 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:21 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:30 PM</div></div>	<div><div></div><div>Oct 24, 2025</div></div>
<div><div></div><div><div></div><div>YM-3</div></div><div>Engagement Service (Social Interactions)</div></div>	<div><div></div><div>Ada</div></div>	<div><div></div><div>Alan Jalil</div></div>	<div><div></div><div>Low</div></div>	<div><div></div><div>Unresolved</div></div>	<div><div></div><div>10/13/2025, 4:21 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:21 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:30 PM</div></div>	<div><div></div><div>Oct 24, 2025</div></div>
<div><div></div><div><div></div><div>YM-2</div></div><div>Content Service (Novel &amp; Chapter Management)</div></div>	<div><div></div><div>Phu Truong Nguyen</div></div>	<div><div></div><div>Alan Jalil</div></div>	<div><div></div><div>High</div></div>	<div><div></div><div>Unresolved</div></div>	<div><div></div><div>10/13/2025, 4:20 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:20 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:30 PM</div></div>	<div><div></div><div>Oct 24, 2025</div></div>
<div><div></div><div><div></div><div>YM-1</div></div><div>User Service (Identity &amp; Access Management)</div></div>	<div><div></div><div>S Lorry</div></div>	<div><div></div><div>Alan Jalil</div></div>	<div><div></div><div>Highest</div></div>	<div><div></div><div>Unresolved</div></div>	<div><div></div><div>10/13/2025, 4:20 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:20 PM</div></div>	<div><div></div><div>Oct 13, 2025, 4:30 PM</div></div>	<div><div></div><div>Oct 24, 2025</div></div>

+ Create

5 of 5

## 2. Project Progress Report

2.1 Reporting Period (**From:** October 13, 2025 **To:** October 24, 2025)

We are capturing all activities conducted between **13 Oct and 24 Oct**, which include several **Product Backlog Refinements, Product Refinements, Frontend Refinements, Backend Refinements, QA/CI/CD Refinements, along with 1 Sprint Planning and 1 Sprint Review.**

**We also held Daily Standup every day in all Sprints for 15min in the morning.**

2.2 Sprint Overview

- **YM Sprint 1:** October 13 – 24, 2025 (Microservices Implementation)
- **All development including QA and deployment completed in 1 Sprint.**

## 2.3 Sprint Objectives

### 1. Architecture & Design

- Complete domain-driven design modeling for all bounded contexts
- Define service boundaries and inter-service contracts
- Design event-driven communication patterns using Kafka
- Create architectural decision records (ADRs) for key decisions

### 2. Microservices Implementation

- Implement 5 core microservices with full CRUD operations
- Develop RESTful APIs with OpenAPI/Swagger documentation
- Implement database-per-service pattern with PostgreSQL
- Set up Redis for caching and real-time counters
- Configure Kafka topics for asynchronous event publishing

### 3. Platform Services

- Deploy Spring Cloud Config Server for centralized configuration
- Set up Eureka Service Registry for service discovery
- Implement API Gateway with JWT authentication and routing rules

### 4. Frontend Integration

- Migrate Yushan Platform Frontend to consume microservice APIs
- Update Yushan Admin Dashboard for microservices architecture
- Implement error handling and circuit breaker patterns in UI

### 5. DevOps & Infrastructure

- Provision DigitalOcean infrastructure using Terraform
- Set up independent CI/CD pipelines for each repository using GitHub Actions
- Implement comprehensive security scanning (SAST, SCA, DAST)
- Deploy containerized services with Docker multi-stage builds
- Configure centralized logging (ELK stack) and monitoring (Prometheus + Grafana)

### 6. Quality Assurance

- Achieve >80% test coverage across all microservices
- Implement unit, integration, and E2E testing
- Perform security vulnerability scanning with Snyk and Trivy
- Execute OWASP ZAP baseline scans for DAST
- Conduct performance and load testing with JMeter

## 7. Documentation

- Complete comprehensive architecture documentation
- Document all API endpoints with Swagger/OpenAPI
- Create deployment guides and runbooks
- Produce detailed project report covering logical and physical architecture

### 2.4. Sprint Accomplishment (Summary of Work Plan vs Actual)

#### Architecture & Design -

- Successfully decomposed monolith into 5 bounded contexts through DDD workshops
- Created detailed logical architecture with tiers, layers, and interaction diagrams
- Defined 13+ architectural decisions (AD-01 through AD-13) with clear rationale
- Documented trade-offs and consequences for each major decision
- Produced comprehensive entity-relationship diagrams for all 5 services

Domain

**user: Auth - Library - Author - Admin**

novel: review, category, report novel - chapter: comment

gamification: exp, vote, yuan

ranking: novel, user

history - analytic



Domain

#### Microservices Implementation -

##### User Service (10 story points):

- User registration with email verification
- JWT-based authentication and authorization
- Profile management and role transitions (Reader → Author)
- User library management
- Integration with Redis for OTP and session caching

##### Content Service (10 story points):

- Novel and chapter CRUD operations
- Publishing workflow with status management
- Category and tag management
- Basic discovery and search functionality
- Integration with Elasticsearch for full-text search

**Engagement Service (10 story points):**

- Comments system for chapters
- Reviews system for novels with rating aggregation
- Voting mechanism
- Content reporting and admin moderation tools
- Event publishing to Kafka for all engagement actions

**Gamification Service (10 story points):**

- Daily login rewards system
- Achievement and badge system
- Virtual currency (Yuan) management
- Experience points (EXP) and leveling system
- Redis-based leaderboards with  $O(\log n)$  performance

**Analytics Service (10 story points):**

- Reading history tracking
- Novel ranking system (trending, popular, new)
- Author analytics dashboard with engagement metrics
- Platform-wide analytics aggregation
- Precomputed rankings stored in Redis for fast access

**Platform Services – Completed:**

- Spring Cloud Config Server deployed with Git-backed configuration
- Eureka Service Registry with automatic service registration/deregistration
- API Gateway with JWT validation, routing, circuit breakers, and retries
- All services successfully discovering each other via Eureka

**Frontend Integration – Completed:**

- Yushan Platform Frontend migrated to microservice APIs (major refactoring by Zhu Yuhui & Yang Shuang)
- Admin Dashboard adapted for microservices (adjusted by Ahan Jaiswal)
- API clients generated and integrated
- Error handling and fallback mechanisms implemented
- Deployed to GitHub Pages with automated health checks

**DevOps & Infrastructure – Excellence Achieved:****Infrastructure (Terraform):**

- Complete DigitalOcean infrastructure provisioned as code
- VPC networking with proper security groups
- Load balancer configuration

- Managed PostgreSQL, Redis, and Kafka clusters
- All infrastructure reproducible and version-controlled

### **CI/CD Pipelines (11 repositories):**

- Backend Services: Maven build → Unit tests → Integration tests → Lint (SpotBugs, Checkstyle) → SCA (OWASP, Snyk) → Docker build → Trivy scan → Deploy → Health check → ZAP DAST
- Frontend Apps: npm build → ESLint → Jest tests → SonarCloud → Snyk → Docker build → Trivy → Deploy to GitHub Pages → Playwright smoke test → Auto-rollback
- All pipelines publish SARIF reports to GitHub Security tab
- Docker images optimized with multi-stage builds (~70% size reduction to 150-200MB)

### **Security Scanning Results:**

- Zero critical vulnerabilities in production images
- All dependencies scanned with OWASP Dependency-Check and Snyk
- Container images scanned with Trivy
- DAST performed with OWASP ZAP baseline on all public endpoints
- SonarCloud quality gates passed for all repositories

### **Observability Stack:**

- Prometheus + Grafana for metrics and dashboards
- ELK Stack (Elasticsearch, Logstash, Kibana) for centralized logging
- Filebeat for log shipping from all services
- Custom dashboards for service health, latency, throughput, and error rates

### **Quality Assurance – Exceeded Targets:**

- Test Coverage: Achieved > 80% across all microservices
  - User Service: 85%
  - Content Service: 83%
  - Engagement Service: 82%
  - Gamification Service: 84%
  - Analytics Service: 78% (due to complex aggregation logic)
- Unit tests: 500+ test cases across all services
- Integration tests: Full repository and service layer testing with test containers
- E2E tests: Selenium IDE and Playwright for critical user journeys
- Performance tests: JMeter scenarios for load, stress, and recovery
- Security tests: Passed all SAST, SCA, and DAST scans

### **Comprehensive Documentation:**

- 86-page detailed project report covering:
  - Introduction and business needs



- Solution overview (logical and physical architecture)
- Quality attributes (performance, availability, security, maintainability)
- DevOps and development lifecycle
- Individual contribution summaries
- All 5 microservices documented with Swagger/OpenAPI specs
- Architectural decision records for 13+ major decisions
- Deployment guides and operational runbooks
- ERD diagrams for all 5 databases
- Sequence diagrams for key workflows

## Platform Readiness:

The entire microservices platform is fully functional, tested, secured, deployed to production, and ready for demonstration and evaluation.

## 2.5. Sprint Burn Down Chart

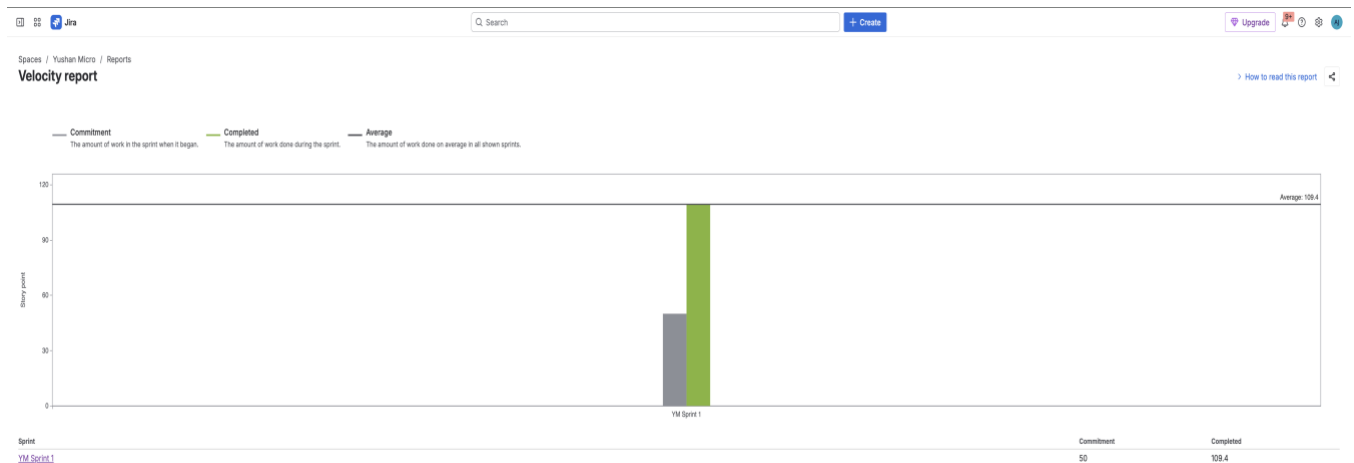
### - YM Sprint 1 (By Story Points / By Issue Count):



## - By Issue Count:



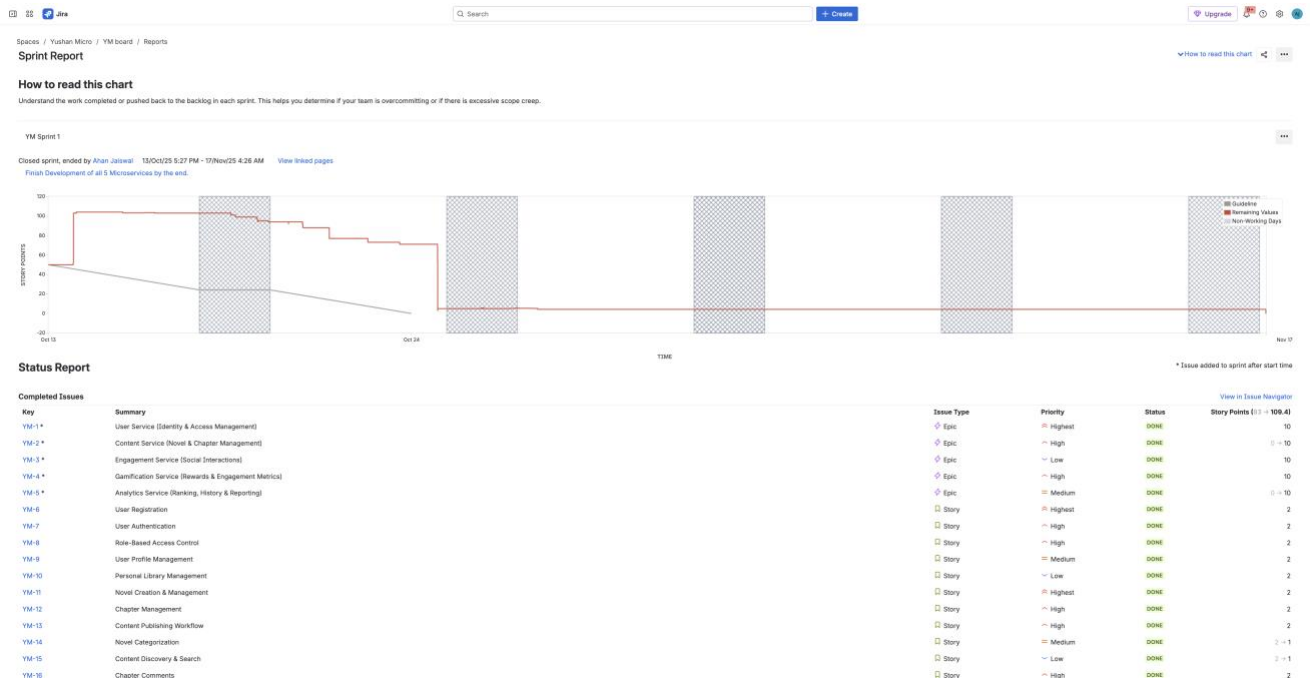
## Team Velocity –



## Sprint Report –

We are attaching the **detailed Sprint Reports for YM Sprint 1**. As a team, we have **updated all tasks with comprehensive time logs**.

## Sprint 1 -



We have agreed to ensure that **each team member accounts for at least 10 full days of work effort** throughout the project.

### 2.6. Problems Encountered, Action Plan, Status

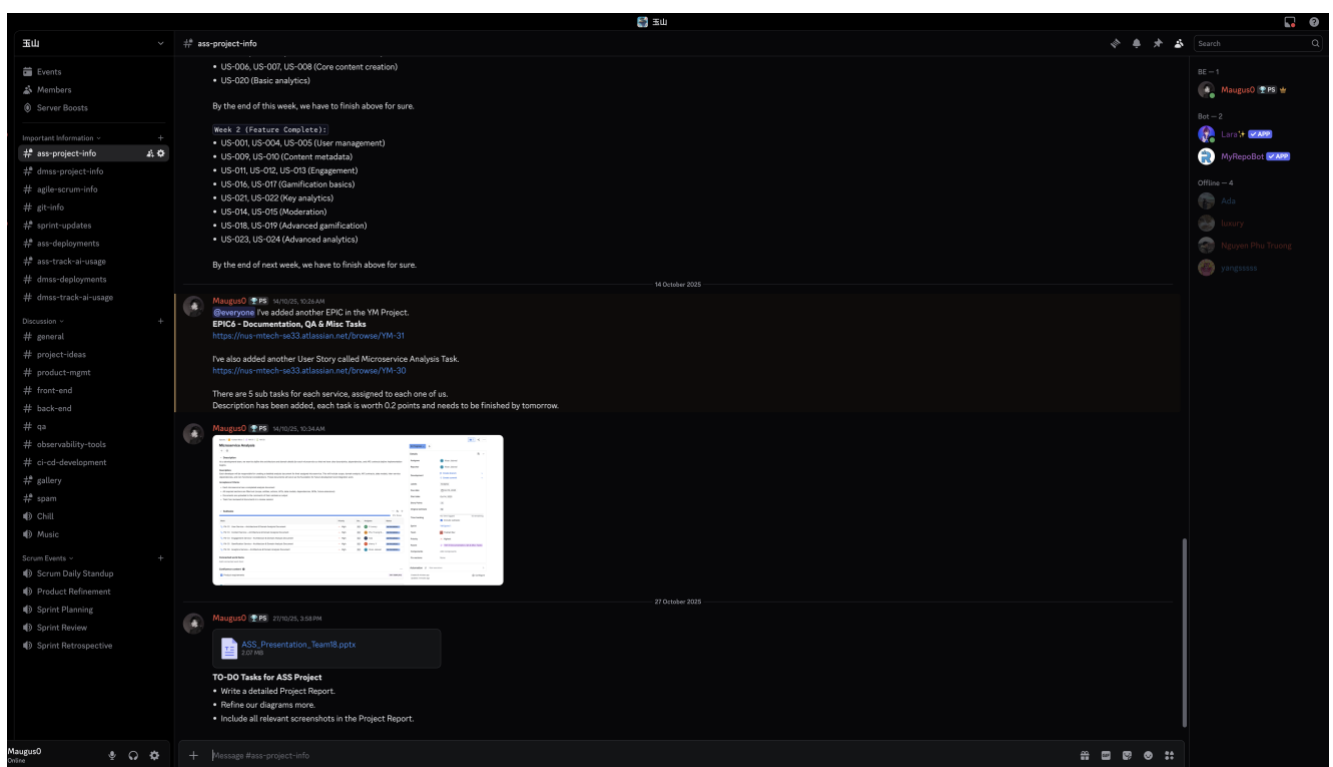
During YM Sprint 1, the team reflected on the overall completion of the DMSS project and discussed how final testing, documentation, and quality improvements were handled.

Key points and reflections related to DMSS were integrated into the ASS project's retrospective discussions. The team noted that the entire DMSS development had already been completed before starting the ASS project, and hence there were limited new challenges or blockers to discuss for FE in ASS, we focused on finishing all tasks for ASS Project from BE perspective.

Transition of focus to the ASS project was therefore smooth from DMSS, with no pending deliverables or unresolved issues remaining from DMSS, and we were also able to finish all of the tasks in ASS smoothly before Presentation.

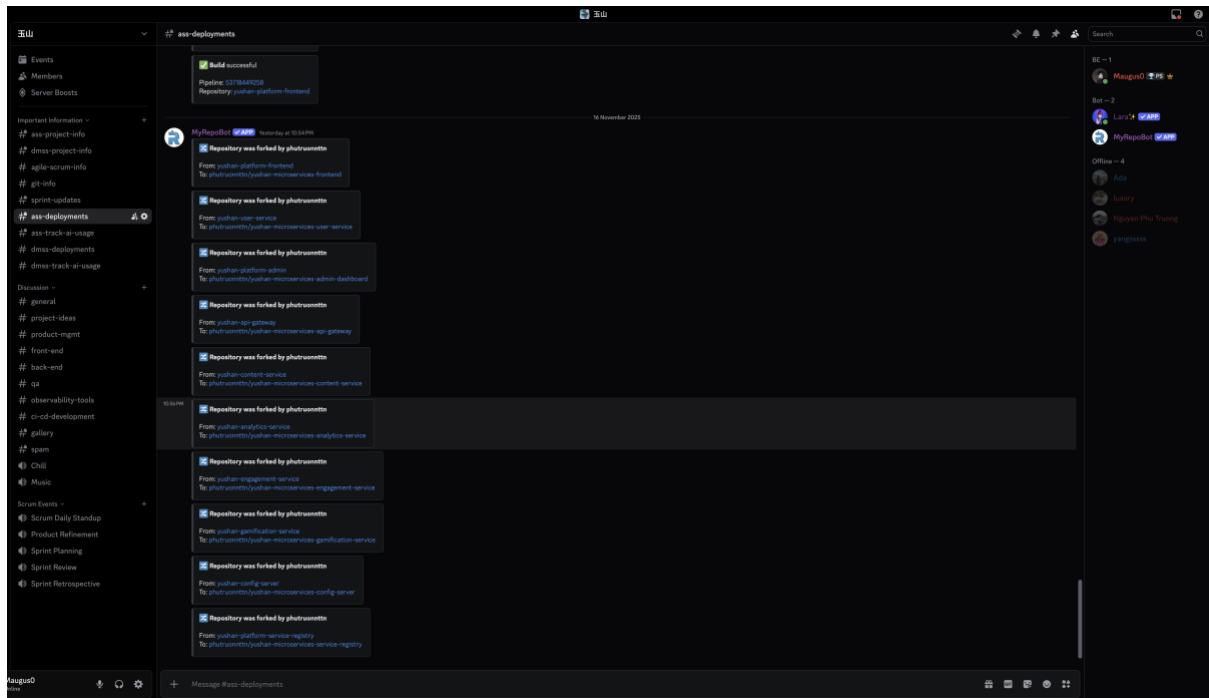
Each team member has taken on **multiple responsibilities** to ensure coverage across different areas in ASS Project:

- **Ahan Jaiswal:** Product Owner / Manager, Full Stack Developer, QA, Deployment & Monitoring Solutions.
- **Nguyen Phu Truong:** Backend Developer, CI/CD Design, Deployment, Implementation & Management.
- **Yang Shuang:** Product Manager, Frontend + QA Automation (FE/BE).
- **Zhang Yan:** Backend Engineer & DBA, QA Automation (BE).
- **Zhu Yuhui:** Product Manager, Frontend Engineer, Functional QA.



Since the start of the project, we have established a structured workflow using a set of tools to support agile development and team collaboration. JIRA is used for agile board and task management, with Planning Poker for sprint planning and Retrium for sprint retrospectives. Our codebase is maintained on GitHub for both frontend and backend repositories. For staging environments, the backend is hosted on Railway, while the frontend is hosted on GitHub Pages, allowing effective testing alongside ongoing development.

For communication, we use **Discord as our central team channel**, with dedicated channels for different aspects of the project.



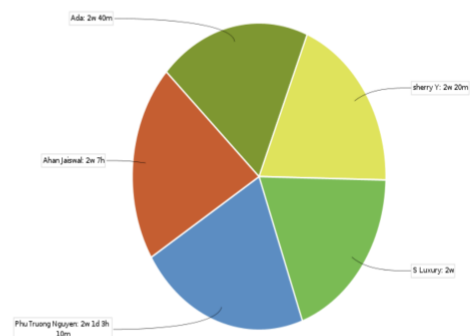
We have also set up a bot integrated with both of our repositories to notify the team in the #deployment channel about pull requests, build statuses, and deployments. This setup ensures that the team remains up to date and can respond quickly to issues, maintaining staging environment stability for simultaneous testing and development.

### 3. Project Team Members Contribution Summary

Workload Pie Chart Report

Project: [Yushan Micro](#) (Time Spent) by Assignee

Chart



**Sprint Burn Down Pattern:**

- Days 1-3: Steep progress on architecture design and service scaffolding
- Days 4-7: Steady progress on implementation and integration
- Days 8-10: Completion of testing, documentation, and deployment

All 24 user stories moved from "To Do" → "In Progress" → "Done" within the sprint window, with no carryover items.

Ahan Jaiswal:

**Roles:** Product Owner, Scrum Master, FE/BE Engineer, Microservices Engineer, QA, DevOps, Infrastructure, Monitoring and Observability

Ahan took on a wide range of responsibilities across engineering, DevOps, and project management. He completely built and tested the **Analytics Service**, handled all microservice-related updates on the **Admin Frontend**, and contributed a major portion of the testing effort there. He also led the initial setup of the **API Gateway** and **Service Registry**, designed CI/CD pipelines for all repositories, and collaborated on infrastructure through Terraform.

As Scrum Master and Product Owner, he facilitated sprint planning, daily standups, and JIRA administration.

His key achievements include developing the full analytics engine with ranking algorithms, migrating the admin dashboard to a microservices architecture, standardising CI/CD patterns for the entire project, and configuring the full observability stack. He also authored major sections of the project report.

**Technical Highlights:**

- Redis-based ranking system with efficient  $O(\log n)$  updates
- Precomputed analytics reducing query times by ~90%
- Multi-stage Docker builds reducing image size by 70%
- Fully automated deployments with health checks and rollback

Nguyen Phu Truong:

**Roles:** Backend Engineer, Infrastructure Engineer, DevOps Specialist

Truong owned end-to-end development and testing for both the **Content Service** and **Engagement Service**, and contributed significantly to the **Gamification Service**. He also developed the project's entire Terraform infrastructure for DigitalOcean and set up the **Spring Cloud Config Server**.

He played a major role in API Gateway configuration and CI/CD optimisation, improving build times through smarter caching strategies.

**Technical Highlights:**

- Elasticsearch full-text search with custom analyzers
- Kafka event publishing with idempotency
- Terraform-based VPC, networking, and PostgreSQL provisioning
- Backup strategies and secure infrastructure configuration

Yang Shuang (Sherry):

**Roles:** Frontend Developer, QA Engineer, Integration Specialist

Yang led major refactoring of the platform frontend (shared 50% responsibility with Zhu Yuhui) and built the full frontend automated test suite. She also contributed to testing for the Admin Frontend and Gamification Service, created integration logic between microservices and the UI, and developed both Playwright and Selenium automation scripts.

Key accomplishments include migrating the frontend to microservice-based APIs, implementing robust error-handling patterns, achieving extensive Jest test coverage, and building reusable API modules.

**Technical Highlights:**

- Efficient API composition strategies
- Client-side caching reducing API calls by 40%
- Automated smoke tests with rollback
- Full cross-browser compatibility suite

Zhang Yan (Luxury S):

**Roles:** Backend Engineer, Database Specialist, QA Engineer

Zhang Yan fully designed and implemented the **User Service**, including authentication, authorization, and secure user management. She contributed to Gamification Service testing, designed the data model, and ensured best practices in password hashing, RBAC, email verification, and OTP handling.

She also created a highly complete test suite for the User Service with 85% coverage and helped refine backend testing across other services.

**Technical Highlights:**

- JWT authentication with refresh token support
- Redis-backed OTP flow with TTL
- User role transition logic (Reader → Author)
- Optimised MyBatis mappers for complex queries

Zhu Yuhui (Ada):

**Roles:** Frontend Developer, UI/UX Designer, Integration Tester

Zhu worked jointly with Yang to refactor the main reader frontend and handled end-to-end UI development and responsive design. He integrated the interface with all five backend services, contributed small enhancements to the Engagement Service, and carried out cross-browser testing.

His work modernized the look and feel of the platform and ensured compatibility across device types.

**Technical Highlights:**

- Component-based architecture using React Hooks
- State management for complex UI flows
- Progressive Web App capabilities
- Accessibility improvements (ARIA, keyboard navigation)



## 4. Collaboration Highlights

Across the entire project, the team delivered:

- **11 repositories** fully maintained
- **109.4 story points**, exceeding target by 118%
- **500+ test cases** across all services
- **Over 80% system-wide test coverage**
- **Zero critical vulnerabilities** in production
- **Fully automated CI/CD pipelines** for every repository
- **86 pages** of comprehensive documentation

### Pair Programming Contributions:

- Ahan & Truong: API Gateway routing and security
- Yang & Zhu: Frontend API integration
- Zhang & Truong: Authentication flows
- Whole Team: Kafka event schema design sessions

## 5. 4. Highlights & Evidence

### 5.1 System Architecture

#### Logical Architecture:

- Client (Frontend) Layer
- Infrastructure and Cross-Cutting Services
- Application Layer (Gateway + Microservices)
- Data Layer
- Observability Layer

#### Physical Deployment:

- DigitalOcean Droplets with VPC networking
- API Gateway behind a load balancer
- Five microservices on dedicated compute nodes
- Managed PostgreSQL (per service)
- Redis cluster for caching and OTP
- Kafka cluster for event streaming
- Elasticsearch for search capabilities
- Prometheus, Grafana, and ELK for monitoring and logging

## 5.2 Technology Stack Overview

**Frontend:** React 18, Tailwind, Axios, React Router

**Gateway:** Spring Cloud Gateway, Resilience4j

**Backend:** Java 21, Spring Boot 3, Spring Cloud

**Service Discovery:** Eureka

**Config:** Spring Cloud Config Server

**Data:** PostgreSQL

**Caching:** Redis

**Messaging:** Kafka

**Search:** Elasticsearch

**Object Storage:** DigitalOcean Spaces

**Containers:** Docker (multi-stage builds)

**Orchestration:** Docker Compose (local), systemd (prod)

**IaC:** Terraform























**CI/CD:** GitHub Actions

**Monitoring:** Prometheus, Grafana

**Logging:** ELK Stack

**Security:** Snyk, Trivy, Dependency-Check, ZAP

**Code Quality:** SonarCloud, SpotBugs, Checkstyle, ESLint

 Yushan Platform >	 phutruonnttn/yushan-terraform-script
 Swagger >	 maugus0/yushan-platform-admin
 Yushan Microservices   DigitalOcean	 maugus0/yushan-platform-frontend
 Yushan Microservices   DigitalOcean2	 maugus0/yushan-platform-service-registry
 Eureka   Yushan Microservices	 maugus0/yushan-api-gateway
 Prometheus	 maugus0/yushan-config-server
 Grafana	 maugus0/yushan-user-service
 ElasticSearch	 maugus0/yushan-content-service
 Kibana	 maugus0/yushan-engagement-service
 Yushan Micro	 maugus0/yushan-gamification-service
 Yushan WebApp	 maugus0/yushan-analytics-service

































## first-project DEFAULT

Update your project information under Settings


→ Move Resources

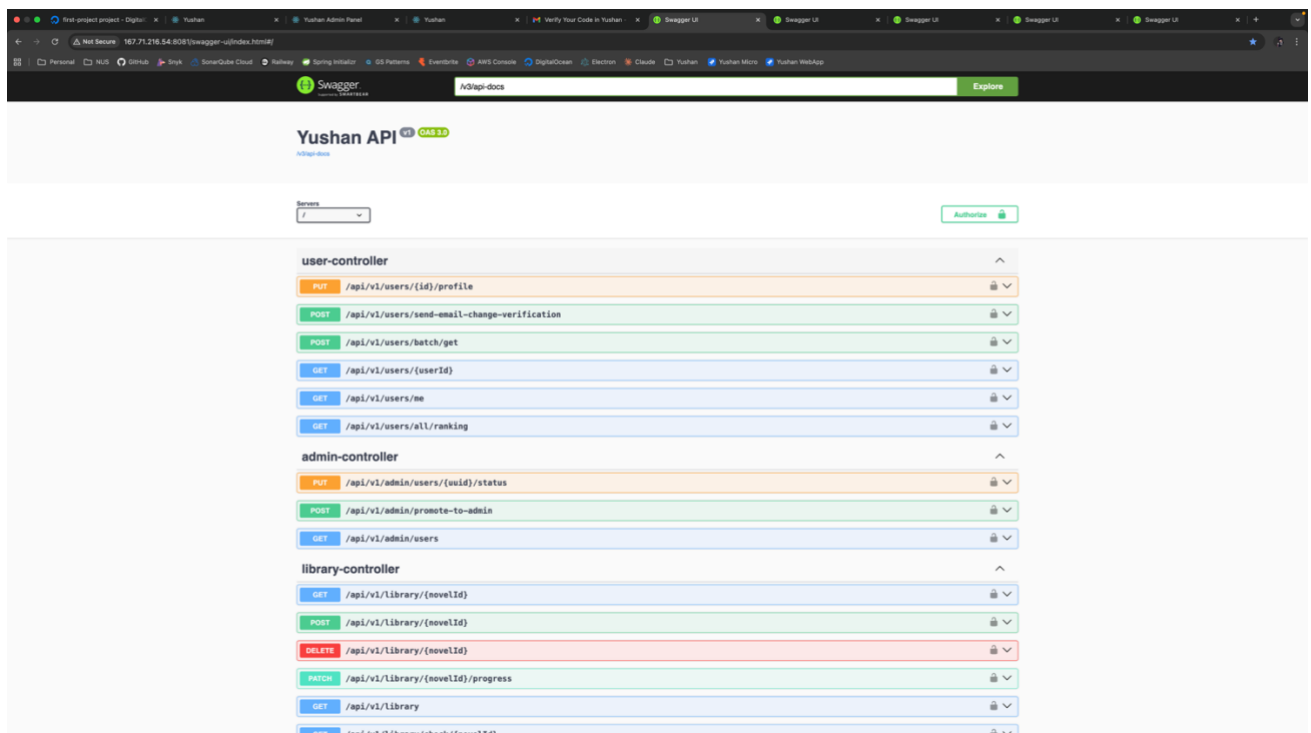
Resources Activity Settings

### DROPLETS (10)

	yushan-engagement-service	167.172.65.76			Upsize	...
	yushan-gamification-service	139.59.243.188			Upsize	...
	yushan-gamification-db	178.128.83.217			Upsize	...
	yushan-engagement-db	206.189.144.116			Upsize	...
	yushan-user-service	167.71.216.54			Upsize	...
	yushan-content-service	157.245.153.167			Upsize	...
	yushan-infrastructure	167.172.72.189			Upsize	...
	yushan-user-db	165.22.253.32			Upsize	...
	yushan-content-db	188.166.254.179			Upsize	...
	terraform-deploy-server	146.190.101.168			Upsize	...

### SPACES (1)

	yushan-content	https://yushan-content.sgp1.digitaloceanspaces.com	...
---	----------------	--	-----



The screenshot shows the Swagger UI for the Yushan API. The interface includes a search bar at the top with the text "Yushan API" and a version indicator "v1.0.0". Below the search bar, there is a "Authorize" button. The main content area displays a list of API endpoints organized by controller. The controllers listed are "user-controller", "admin-controller", and "library-controller". Each controller has a list of endpoints with their respective HTTP methods (PUT, POST, GET, DELETE, PATCH) and a lock icon indicating authentication requirements.

**user-controller**

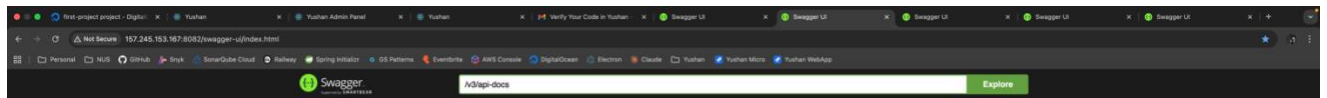
- PUT /api/v1/users/{id}/profile
- POST /api/v1/users/send-email-change-verification
- POST /api/v1/users/batch/get
- GET /api/v1/users/{userId}
- GET /api/v1/users/me
- GET /api/v1/users/all/ranking

**admin-controller**

- PUT /api/v1/admin/users/{userId}/status
- POST /api/v1/admin/promote-to-admin
- GET /api/v1/admin/users

**library-controller**

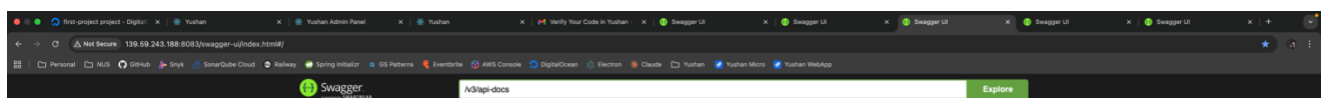
- GET /api/v1/library/{novelId}
- POST /api/v1/library/{novelId}
- DELETE /api/v1/library/{novelId}
- PATCH /api/v1/library/{novelId}/progress
- GET /api/v1/library
- GET /api/v1/library/check/{novelId}



**Content Service API** APIs for managing novels

Novel Management APIs for managing novels

Method	Endpoint	Description
GET	/api/v1/novels/{id}	[PUBLIC] Get novel by ID
PUT	/api/v1/novels/{id}	[AUTHORADMIN] Update novel
DELETE	/api/v1/novels/{id}	[AUTHORADMIN] Archive novel
PUT	/api/v1/novels/{id}/rating	[AUTHENTICATED] Update novel rating and review count
GET	/api/v1/novels	[PUBLIC] Get novels with pagination (excludes ARCHIVED)
POST	/api/v1/novels	[AUTHORADMIN] Create a new novel
POST	/api/v1/novels/{id}/vote	[AUTHENTICATED] Increment vote count
POST	/api/v1/novels/{id}/view	[PUBLIC] Increment view count
POST	/api/v1/novels/{id}/unhide	[ADMINAUTHOR] Unhide novel
POST	/api/v1/novels/{id}/unarchive	[ADMIN] Unarchive novel
POST	/api/v1/novels/{id}/submit-review	[AUTHOR] Submit novel for review
POST	/api/v1/novels/{id}/reject	[ADMIN] Reject novel
POST	/api/v1/novels/{id}/hide	[ADMINAUTHOR] Hide novel
POST	/api/v1/novels/{id}/approve	[ADMIN] Approve novel
POST	/api/v1/novels/batch/get	[PUBLIC] Batch get novels by IDs
GET	/api/v1/novels/{id}/vote-count	[PUBLIC] Get novel vote count
POST	/api/v1/novels/{id}/vote	[PUBLIC] Get novel vote count



**Analytics Service API** Admin analytics APIs for platform insights and metrics

Analytics (Admin) Admin analytics APIs for platform insights and metrics

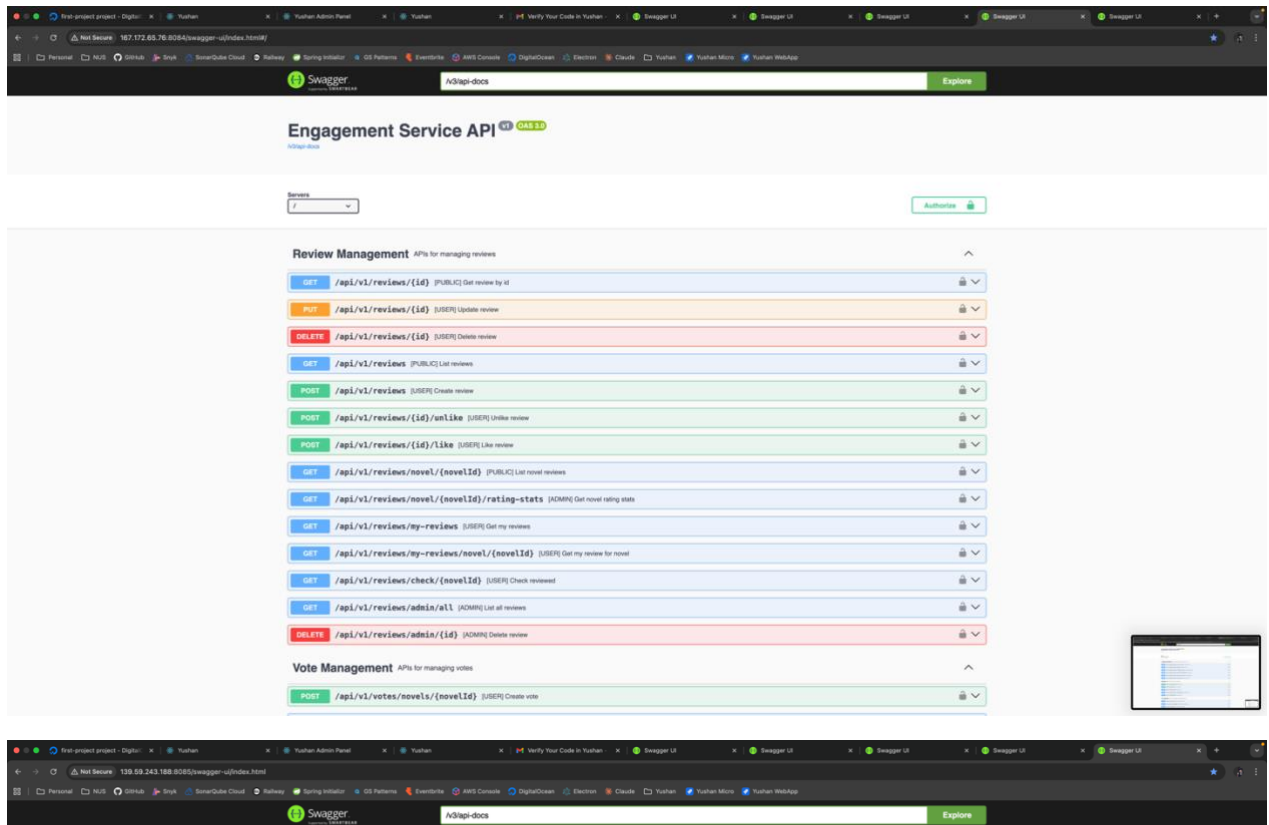
Method	Endpoint	Description
GET	/api/v1/admin/analytics/users/trends	Get user activity trends
GET	/api/v1/admin/analytics/summary	Get analytics summary
GET	/api/v1/admin/analytics/reading/activity	Get reading activity trends
GET	/api/v1/admin/analytics/platform/top-content	Get top content
GET	/api/v1/admin/analytics/platform/overview	Get platform statistics
GET	/api/v1/admin/analytics/platform/dau	Get daily active users

Ranking Ranking APIs for novels, users, and authors

Method	Endpoint	Description
POST	/api/v1/ranking/update	Update all rankings
GET	/api/v1/ranking/user	Get user ranking
GET	/api/v1/ranking/novel	Get novel ranking
GET	/api/v1/ranking/novel/{novelId}/rank	Get novel's best rank
GET	/api/v1/ranking/author	Get author ranking

Test Utilities Test APIs for generating JWT tokens (Development only)

Method	Endpoint	Description
GET	/api/test/token/user	[TEST] Generate USER token
GET	/api/test/token/suspended	[TEST] Generate SUSPENDED token
GET	/api/test/token/admin	[TEST] Generate ADMIN token



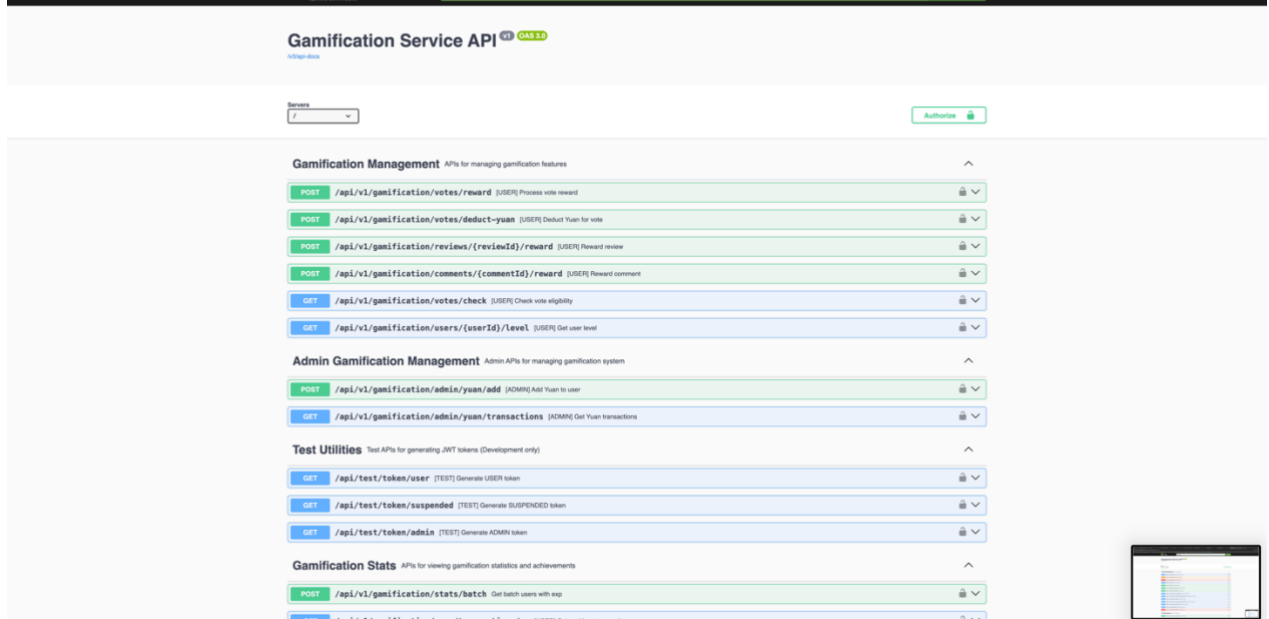
Swagger UI for Engagement Service API (v1.0.0) OAS 3.0

Review Management APIs for managing reviews

- GET /api/v1/reviews/{id} [PUBLIC] Get review by id
- PUT /api/v1/reviews/{id} [USER] Update review
- DELETE /api/v1/reviews/{id} [USER] Delete review
- GET /api/v1/reviews [PUBLIC] List reviews
- POST /api/v1/reviews [USER] Create review
- POST /api/v1/reviews/{id}/unlike [USER] Unlike review
- POST /api/v1/reviews/{id}/like [USER] Like review
- GET /api/v1/reviews/novel/{novelId} [PUBLIC] List novel reviews
- GET /api/v1/reviews/novel/{novelId}/rating-stats [ADMIN] Get novel rating stats
- GET /api/v1/reviews/my-reviews [USER] Get my reviews
- GET /api/v1/reviews/my-reviews/novel/{novelId} [USER] Get my review for novel
- GET /api/v1/reviews/check/{novelId} [USER] Check reviewed
- GET /api/v1/reviews/admin/all [ADMIN] List all reviews
- DELETE /api/v1/reviews/admin/{id} [ADMIN] Delete review

Vote Management APIs for managing votes

- POST /api/v1/votes/novels/{novelId} [USER] Create vote



Swagger UI for Gamification Service API (v1.0.0) OAS 3.0

Gamification Management APIs for managing gamification features

- POST /api/v1/gamification/votes/reward [USER] Process vote reward
- POST /api/v1/gamification/votes/deduct-yuan [USER] Deduct Yuan for vote
- POST /api/v1/gamification/reviews/{reviewId}/reward [USER] Reward review
- POST /api/v1/gamification/comments/{commentId}/reward [USER] Reward comment
- GET /api/v1/gamification/votes/check [USER] Check vote eligibility
- GET /api/v1/gamification/users/{userId}/level [USER] Get user level

Admin Gamification Management Admin APIs for managing gamification system

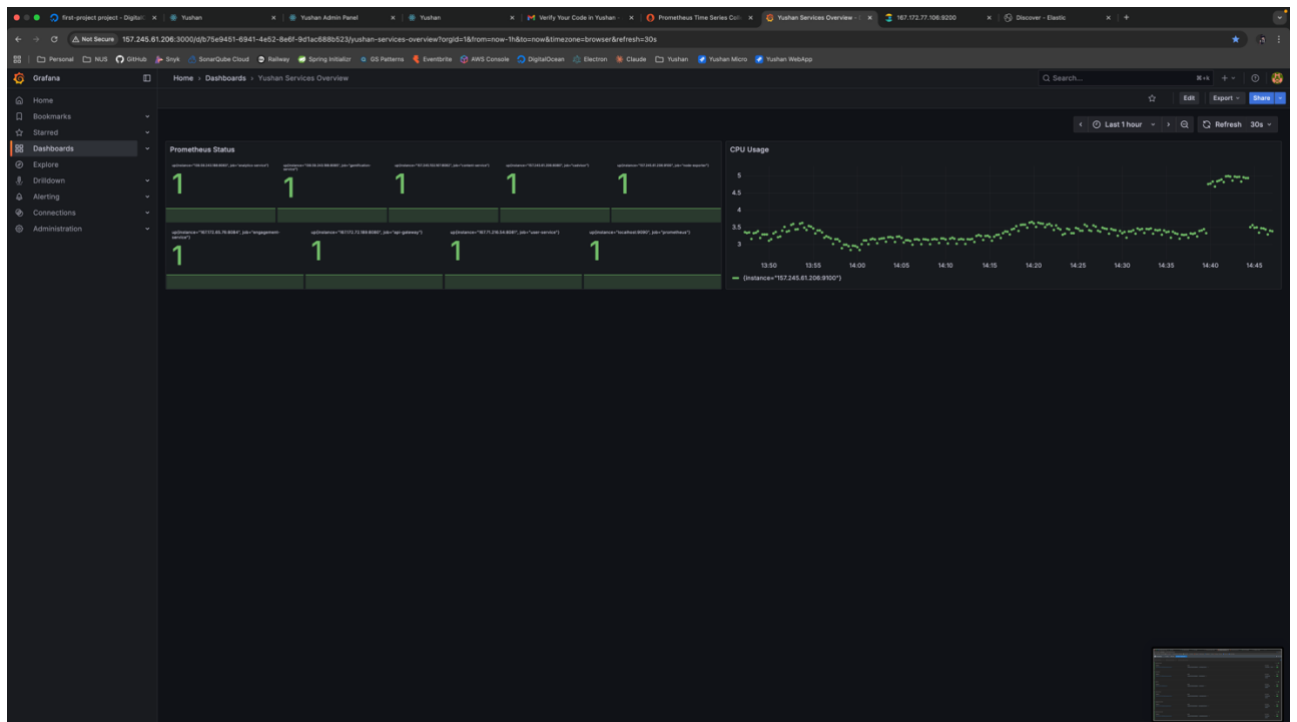
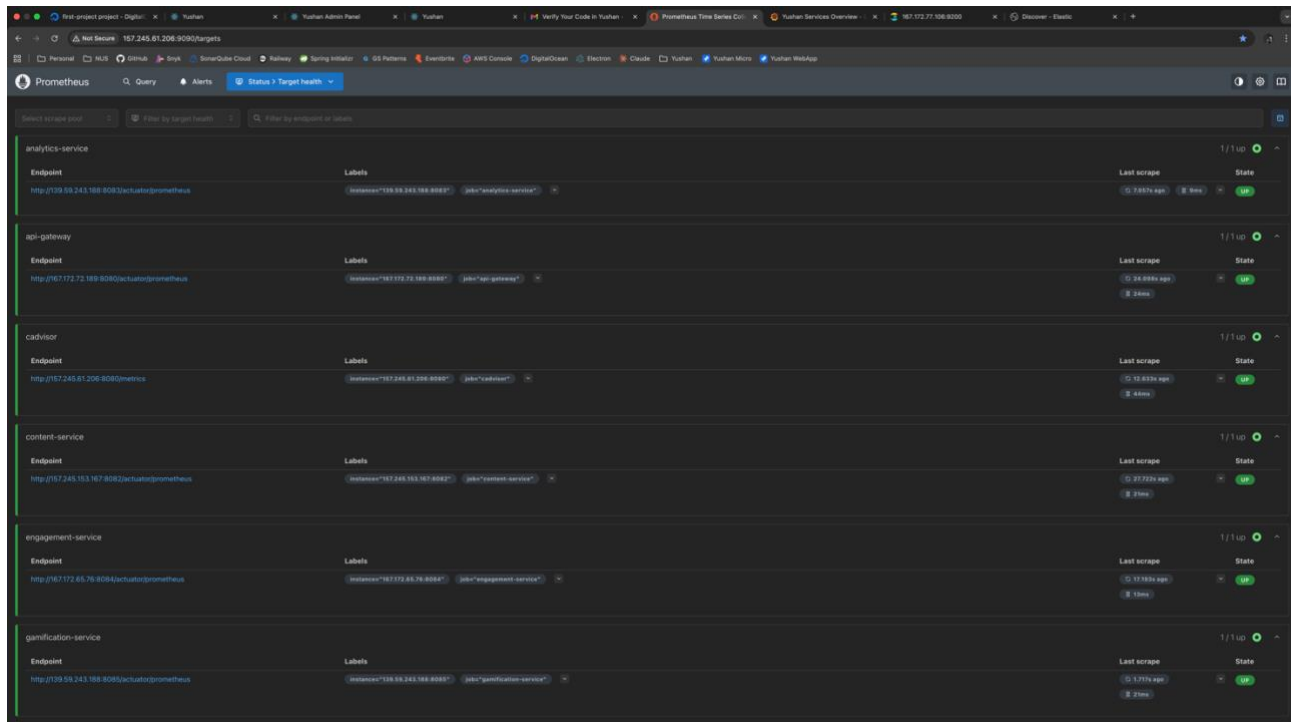
- POST /api/v1/gamification/admin/yuan/add [ADMIN] Add Yuan to user
- GET /api/v1/gamification/admin/yuan/transactions [ADMIN] Get Yuan transactions

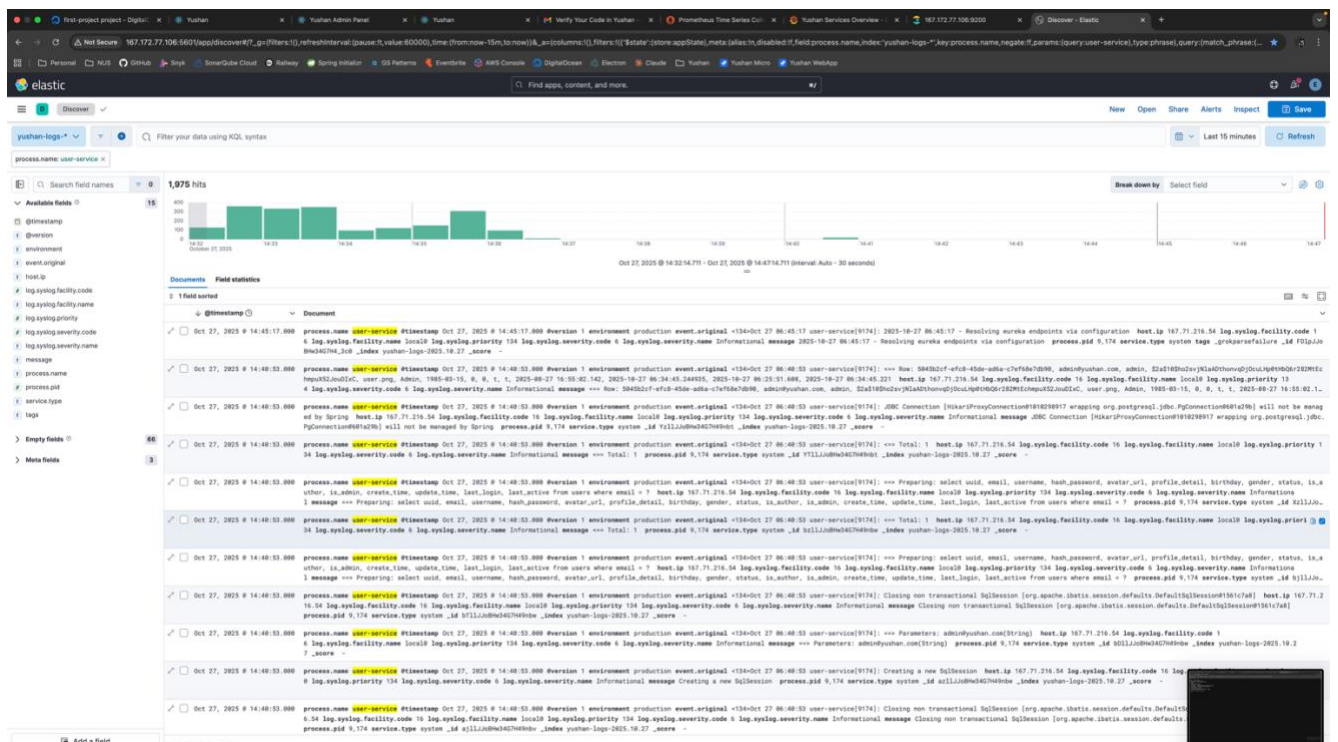
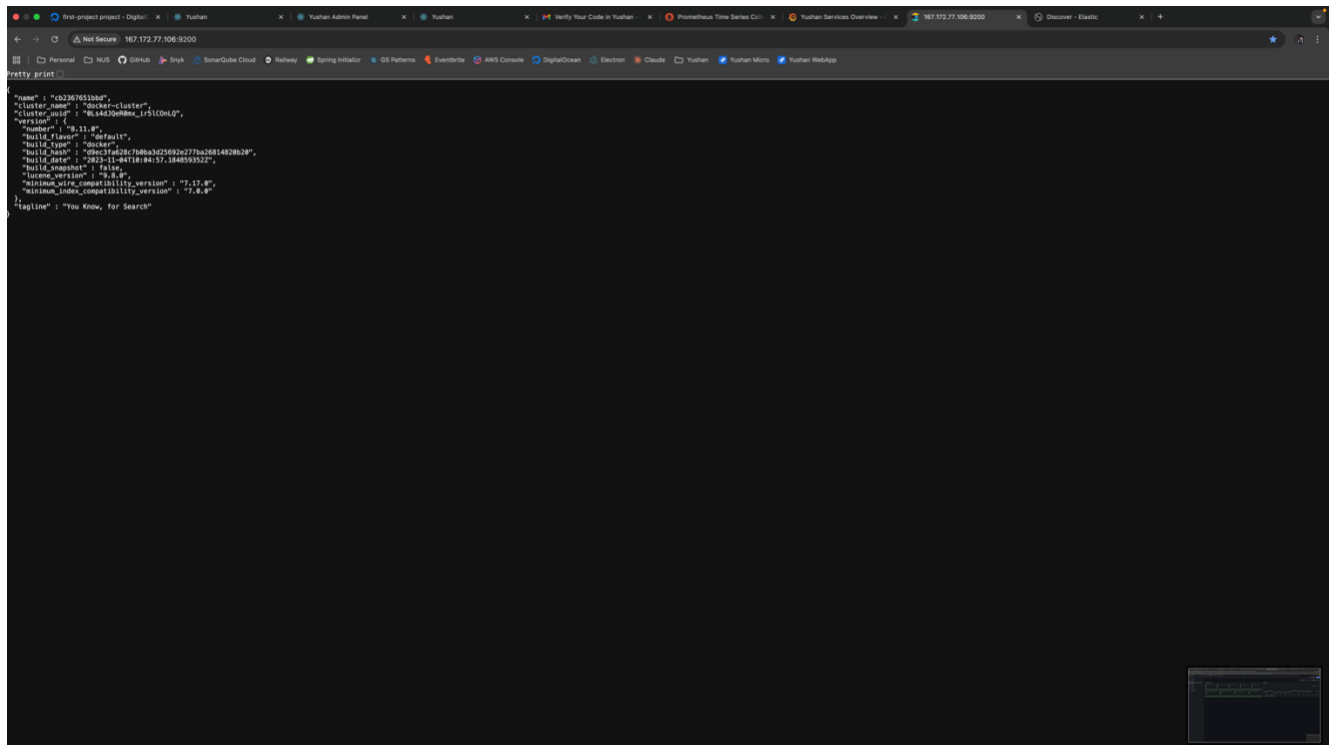
Test Utilities Test APIs for generating JWT tokens (Development only)


- GET /api/test/token/user [TEST] Generate USER token
- GET /api/test/token/suspended [TEST] Generate SUSPENDED token
- GET /api/test/token/admin [TEST] Generate ADMIN token

Gamification Stats APIs for viewing gamification statistics and achievements

- POST /api/v1/gamification/stats/batch [USER] Get batch users with exp
- GET /api/v1/gamification/yuan/transactions/me [USER] Get my Yuan transactions






HOME LAST 1000 SINCE STARTUP

### System Status

Environment	test	Current time	2025-10-27T06:47:54 +0000
Data center	default	Uptime	2 days 05:48
		Lease expiration enabled	true
		Renews threshold	0
		Renews (last min)	28

THE SELF PRESERVATION MODE IS TURNED OFF. THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS.

### DS Replicas


localhost

### Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
ANALYTICS-SERVICE	n/a (1)	(1)	UP (1) - analytics-service-8083
API-GATEWAY	n/a (1)	(1)	UP (1) - api-gateway-c77148764951e297a35888de751c8deb
CONFIG-SERVER	n/a (1)	(1)	UP (1) - config-server-af8b4a250909852a050bc9e941b08282
CONTENT-SERVICE	n/a (1)	(1)	UP (1) - content-service-8082
ENGAGEMENT-SERVICE	n/a (1)	(1)	UP (1) - engagement-service-8084
GAMIFICATION-SERVICE	n/a (1)	(1)	UP (1) - gamification-service-8085
USER-SERVICE	n/a (1)	(1)	UP (1) - user-service-8081

### General Info

Name	Value
total-avail-memory	83mb
num-of-cpus	2
current-memory-usage	56mb (67%)
server-up-time	2 days 05:48
registered-replicas	http://localhost:8761/eureka/
unavailable-replicas	http://localhost:8761/eureka/
available-replicas	


HOME LAST 1000 SINCE STARTUP

### System Status

Environment	test	Current time	2025-10-27T06:47:54 +0000
Data center	default	Uptime	2 days 05:48
		Lease expiration enabled	true
		Renews threshold	0
		Renews (last min)	28

THE SELF PRESERVATION MODE IS TURNED OFF. THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS.

### DS Replicas

localhost

### Instances currently registered with Eureka


Application	AMIs	Availability Zones	Status
ANALYTICS-SERVICE	n/a (1)	(1)	UP (1) - analytics-service-8083
API-GATEWAY	n/a (1)	(1)	UP (1) - api-gateway-c77148764951e297a35888de751c8deb
CONFIG-SERVER	n/a (1)	(1)	UP (1) - config-server-af8b4a250909852a050bc9e941b08282
CONTENT-SERVICE	n/a (1)	(1)	UP (1) - content-service-8082
ENGAGEMENT-SERVICE	n/a (1)	(1)	UP (1) - engagement-service-8084
GAMIFICATION-SERVICE	n/a (1)	(1)	UP (1) - gamification-service-8085
USER-SERVICE	n/a (1)	(1)	UP (1) - user-service-8081

### General Info

Name	Value
total-avail-memory	83mb
num-of-cpus	2
current-memory-usage	56mb (67%)
server-up-time	2 days 05:48
registered-replicas	http://localhost:8761/eureka/
unavailable-replicas	http://localhost:8761/eureka/
available-replicas	

### Instance Info

Name	Value
ipAddr	172.18.0.2
status	UP





Test project project - Digital - x yushan x yushan Admin Panel x yushan x Eureka x +

Net Secure 167.172.72.189:8761

Personal Mail Gitlab Bspk SaaS Cloud Railway Spring Initializer OS Patterns Eventrite AWS Console SigntCloud Election Clouds yushan yushan Micro yushan WebApp

spring Eureka HOME LAST 1000 SINCE STARTUP

### System Status

Environment	test	Current time	2025-10-27T06:48:26 +0000
Data center	default	Uptime	2 days 05:48
		Lease expiration enabled	true
		Renews threshold	0
		Renews (last min)	28

THE SELF PRESERVATION MODE IS TURNED OFF. THIS MAY NOT PROTECT INSTANCE EXPIRY IN CASE OF NETWORK/OTHER PROBLEMS.

### DS Replicas

localhost

### Instances currently registered with Eureka


Application	Alibs	Availability Zones	Status
ANALYTICS-SERVICE	n/a (3)	(3)	UP (3) - analytics.service.8083
API-GATEWAY	n/a (3)	(3)	UP (3) - api-gateway-c71-6875451a207a350886a7511-d84b
CONFIG-SERVICE	n/a (3)	(3)	UP (3) - config-service-a69-da2500095a2e0303e9413a09282
CONTENT-SERVICE	n/a (3)	(3)	UP (3) - content.service.8082
ENGAGEMENT-SERVICE	n/a (3)	(3)	UP (3) - engagement.service.8084
GAMIFICATION-SERVICE	n/a (3)	(3)	UP (3) - gamification.service.8085
USER-SERVICE	n/a (3)	(3)	UP (3) - user.service.8081

### General Info

Name	Value
total-avail-memory	83mb
num-of-cpus	2
current-memory-usage	62mb (74%)
server-up-time	2 days 05:48
registered-replicas	http://localhost:8761/eureka/
unavailable-replicas	http://localhost:8761/eureka/
available-replicas	

### Instance Info

Name	Value
ipAddr	172.18.0.2
status	UP

 **first-project** DEFAULT → Move Resources

Update your project information under Settings

**Resources** Activity Settings

### DROPLETS (10)


Name	IP Address	Tags	Actions
yushan-engagement-service	167.172.65.76		+ [icon] + [icon] Upsize ...
yushan-gamification-service	139.59.243.188	Add tags	+ [icon] + [icon] Upsize ...
yushan-gamification-db	178.128.83.217		+ [icon] + [icon] Upsize ...
yushan-engagement-db	206.189.144.116		+ [icon] + [icon] Upsize ...
yushan-user-service	167.71.216.54		+ [icon] + [icon] Upsize ...
yushan-content-service	157.245.153.167		+ [icon] + [icon] Upsize ...
yushan-infrastructure	167.172.72.189		+ [icon] + [icon] Upsize ...
yushan-user-db	165.22.253.32		+ [icon] + [icon] Upsize ...
yushan-content-db	188.166.254.179		+ [icon] + [icon] Upsize ...
terraform-deploy-server	146.190.101.168		+ [icon] + [icon] Upsize ...

### SPACES (1)

Name	URL	Actions
yushan-content	https://yushan-content.sgp1.digitaloceanspaces.com	...

nd+B)


← Back to Spaces Object Storage


yushan-content


first-project • 47 items • 962 KiB • sgp1

Origin Endpoint https://yushan-content.sgp1.digitaloceanspaces.com

Files Settings


yushan-content
1 item

Create Folder Upload

<input type="checkbox"/>	Name	Size	Last modified
<input type="checkbox"/>	 covers	—	—

cloud.digitalocean.com/projects/96c11c07-2c28-4a27-94f9-098503230572/resources?to=2c1540

Personal Personal MDS SSH Sub SSH Sub Sonaube Cloud Railway Spring Initial Git Pattern Events AWS Console DigitalOcean Electron Claude Yuhuan Yuhuan More Yuhuan WebApp

Search by resource name or public IP (CIDR)

Creates 1 My Team

## first-project DRINK

Update your project information under Settings

Move Resources

### Resources

Activity Settings

#### DROPLETS (10)

**yushan-engagement-service**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	1 vCPU 2GB / 50GB Disk (\$2mo)	IPv4	167772.65.76
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.3
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-service**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	139.59.243.188
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.4
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-db**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	178.128.83.217
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.9
		VPC	<a href="#">default-sgp1</a>

**yushan-engagement-s...**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	206.170.144.195
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.1
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-s...**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	178.128.83.217
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.9
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-db**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	178.128.83.217
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.9
		VPC	<a href="#">default-sgp1</a>

**yushan-engagement-s...**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	206.170.144.195
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.1
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-s...**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	178.128.83.217
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.9
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-db**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	178.128.83.217
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.9
		VPC	<a href="#">default-sgp1</a>

**yushan-engagement-s...**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	206.170.144.195
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.1
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-s...**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	178.128.83.217
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.9
		VPC	<a href="#">default-sgp1</a>

**yushan-gamification-db**

+
+3
Upsize

Image	Ubuntu 22.04 (LTS) x64	Region	SGP1
Size	2 vCPUs 4GB / 80GB Disk (\$24mo)	IPv4	178.128.83.217
	<a href="#">Resize</a>	IPv6	<a href="#">Enable</a>
		Private IP	10.104.0.9
		VPC	<a href="#">default-sgp1</a>

**yushan-engagement-s...**

+
+

cloud.digitalocean.com/projects/06cd1c07-2c28-4427-9419-f08503230572/resources?m=2c15a0

Search by resource name or public IP (Cmd-B)

**PROJECTS**

**MANAGE**

App Platform **New**

Agent Platform

Droplets

GPU Droplets **New**

Functions

Kubernetes

Volumes Block Storage

Databases

Spaces Object Storage

Container Registry

Backups & Snapshots

Network File Storage

Networking

Monitoring

SaaS Add-Ons

By DigitalOcean

Support

Settings

API

Marketplace

Product Docs

What's New

**yushan-engagement-db**

Image: Ubuntu 22.04 (LTS) x64

Size: 1 vCPU, 1GB / 25GB Disk (\$6/mo) [Resize](#)

Region: SGP1

IPv4: 208.189.144.116

IPv6: [Enable](#)

Private IP: 10.104.0.11

VPC: [default-sgpt](#)

**yushan-user-service**

Image: Ubuntu 22.04 (LTS) x64

Size: 1 vCPU, 2GB / 50GB Disk (\$12/mo) [Resize](#)

Region: SGP1

IPv4: 167.71.216.54

IPv6: [Enable](#)

Private IP: 10.104.0.5

VPC: [default-sgpt](#)

**yushan-content-service**

Image: Ubuntu 22.04 (LTS) x64

Size: 1 vCPU, 2GB / 50GB Disk (\$12/mo) [Resize](#)

Region: SGP1

IPv4: 167.245.153.167

IPv6: [Enable](#)

Private IP: 10.104.0.6

VPC: [default-sgpt](#)

**yushan-infrastructure** 167.72.72.189

**yushan-user-db** 165.22.253.32

**yushan-content-db** 188.166.254.179

**terraform-deploy-server** 146.190.101.168

cloud.digitalocean.com/projects/06cd1c07-2c28-4427-9419-f08503230572/resources?m=2c15a0

Search by resource name or public IP (Cmd-B)

**PROJECTS**

**first-project**

**MANAGE**

App Platform **New**

Agent Platform

Droplets

GPU Droplets **New**

Functions

Kubernetes

Volumes Block Storage

Databases

Spaces Object Storage

Container Registry

Backups & Snapshots

Network File Storage

Networking

Monitoring

SaaS Add-Ons

By DigitalOcean

Support

Settings

API

Marketplace

Product Docs

What's New

**yushan-infrastructure** 167.72.72.189

**yushan-user-db**

Image: Ubuntu 22.04 (LTS) x64

Size: 1 vCPU, 1GB / 25GB Disk (\$6/mo) [Resize](#)

Region: SGP1

IPv4: 165.22.253.32

IPv6: [Enable](#)

Private IP: 10.104.0.8

VPC: [default-sgpt](#)

**yushan-content-db**

Image: Ubuntu 22.04 (LTS) x64

Size: 2 vCPUs, 4GB / 80GB Disk (\$24/mo) [Resize](#)

Region: SGP1

IPv4: 188.166.254.179

IPv6: [Enable](#)

Private IP: 10.104.0.10

VPC: [default-sgpt](#)

**terraform-deploy-server**

Image: Ubuntu 25.04 x64

Size: 1 vCPU, 1GB / 25GB Disk (\$6/mo) [Resize](#)

Region: SGP1

IPv4: 146.190.101.168

IPv6: [Enable](#)

Private IP: 10.104.0.2

VPC: [default-sgpt](#)

**SPACES (1)**

**yushan-content** <https://yushan-content.sgp1.digitaloceanspaces.com>

## Repository Links

### Backend Microservices:

- <https://github.com/maugus0/yushan-user-service>
- <https://github.com/maugus0/yushan-content-service>
- <https://github.com/maugus0/yushan-engagement-service>
- <https://github.com/maugus0/yushan-gamification-service>
- <https://github.com/maugus0/yushan-analytics-service>

### Platform Services:

- <https://github.com/maugus0/yushan-platform-service-registry>
- <https://github.com/maugus0/yushan-api-gateway>
- <https://github.com/maugus0/yushan-config-server>

### Frontend Applications:

- <https://github.com/maugus0/yushan-platform-frontend>
- <https://github.com/maugus0/yushan-platform-admin>

### Infrastructure:

- [https://github.com/phutruonnttn/Digital Ocean Deployment with Terraform](https://github.com/phutruonnttn/Digital_Ocean_Deployment_with_Terraform)

## AI Tool Declaration:

We used Claude-Sonnet and ChatGPT (GPT-5) to rephrase our writing into clearer English with improved grammar. All ideas, content, and work are our own, and we take full responsibility for the accuracy and quality of the submission.

### Reference for all JIRA tasks mentioned in the Work table:

<https://nus-mtech-se33.atlassian.net/jira/software/c/projects/YM/summary>