Software as a Service

clearSKY project 2025

## Table of Contents

### 1. Executive Summary

* [1.1 Key Business Value Propositions](#key-business-value-propositions)
* [1.2 Technological Innovation](#technological-innovation)

### 2. Architecture Philosophy and Design Principles

* [2.1 Domain-Driven Design Implementation](#domain-driven-design-implementation)
* 2.2 Architecture Overview
* [2.3 Microservices Architecture Benefits](#microservices-architecture-benefits)

### 3. Comprehensive Service Architecture

* [3.1 Authentication Service - Identity Foundation](#authentication-service---identity-found)
* [3.2 User Management Service - Profile and Relationship Hub](#user-management-service---profile-and-r)
* [3.3 Course Management Service - Academic Structure Engine](#course-management-service---academic-st)
* [3.4 Grade Management Service - Core Academic Engine](#grade-management-service---core-academi)
* [3.5 Grade Review Service - Communication and Workflow Hub](#grade-review-service---communication-an)
* [3.6 Analytics Service - Intelligence and Insights Engine](#analytics-service---intelligence-and-in)

### 4. Technology Stack and Implementation Details

* [4.1 Frontend Architecture Philosophy](#frontend-architecture-philosophy)
* [4.2 Backend Service Implementation](#backend-service-implementation)
* [4.3 Database Design Philosophy](#database-design-philosophy)

### 5. Service Integration and Communication Patterns

* [5.1 API Gateway Implementation Strategy](#api-gateway-implementation-strategy)
* [5.2 Inter-Service Communication Design](#inter-service-communication-design)

### 6. Security Architecture and Implementation

* [6.1 Comprehensive Security Strategy](#comprehensive-security-strategy)

### 7. Performance and Scalability Architecture

* [7.1 Scalability Strategy Implementation](#scalability-strategy-implementation)

### 8. Summary Tables

* [8.1 Service Responsibility Matrix](#service-responsibility-matrix)
* [8.2 Technology Stack Summary](#technology-stack-summary)
* [8.3 Business Process Flow Matrix](#business-process-flow-matrix)
* [8.4 Data Flow Complexity Analysis](#data-flow-complexity-analysis)
* [8.5 Deployment and Operations Summary](#deployment-and-operations-summary)
  1. **Executive Summary**

ClearSKY represents a cutting-edge educational technology platform that revolutionizes the traditional grade management process through a sophisticated microservices architecture. This system addresses the complex needs of modern educational institutions by providing a comprehensive, scalable, and secure platform for managing the complete lifecycle of academic grades—from initial posting through student review processes to final grade submission.

The platform serves as a bridge between traditional academic processes and modern software engineering practices, demonstrating how educational institutions can leverage cloud-native technologies to improve efficiency, transparency, and user experience. Built as part of the NTUA Erasmus+ Software as a Service course, ClearSKY showcases advanced architectural patterns while solving real-world educational challenges.

* 1. **Key Business Value Propositions**

For Educational Institutions: ClearSKY transforms the traditionally cumbersome grade management process into a streamlined, automated workflow. Institutions benefit from reduced administrative overhead, improved transparency in grade management, and enhanced audit capabilities. The credit-based billing model provides predictable costs while the scalable architecture ensures the platform can grow with institutional needs.

For Instructors: The platform eliminates the manual complexity of managing grade reviews and provides powerful analytics tools for understanding student performance. Instructors can upload grades in familiar Excel formats while benefiting from automated validation, review workflow management, and comprehensive statistics dashboards that provide insights into class performance trends.

For Students: Students gain unprecedented transparency into their academic performance with detailed grade breakdowns, statistical comparisons, and a formal mechanism for requesting grade reviews. The platform provides clear visibility into the review process status and maintains a complete audit trail of all grade-related communications.

**Key Features**

**Educational Grade Management** - Comprehensive grade lifecycle management

**Review Workflow** - Student-instructor grade review process

**Credit-based Billing** - Institution-level subscription model

**Analytics Dashboard** - Real-time grade statistics and insights

**Multi-role Authentication** - Students, Instructors, and Institution Representatives

**Scalable Architecture** - Microservices-based for horizontal scaling

**1.2 Technological Innovation**

ClearSKY demonstrates advanced software engineering principles through its implementation of a domain-driven microservices architecture. Each service is designed around specific business capabilities, ensuring high cohesion within services and loose coupling between them. This architectural approach enables independent scaling, technology diversity, and fault isolation while maintaining system coherence through well-defined APIs and data contracts.

* 1. **Architecture Philosophy and Design Principles**

**2.1 Domain-Driven Design Implementation**

The ClearSKY architecture follows Domain-Driven Design principles, organizing the system around core business domains rather than technical concerns. Each microservice represents a bounded context within the educational grade management domain, ensuring that business logic remains coherent and maintainable.

The system identifies six primary bounded contexts: Authentication (identity and access management), User Management (profile and role administration), Course Management (academic structure), Grade Management (core grading workflows), Review Management (grade review processes), and Analytics (reporting and insights). Each context is implemented as an independent microservice with its own data store, business logic, and API surface.

**2.2 Architecture Overview**

ClearSKY implements a **Domain-Driven Microservices Architecture** with the following key architectural patterns:

**API Gateway Pattern** - Single entry point for all client requests

**Database per Service** - Each microservice maintains its own data store

**Event-Driven Communication** - Asynchronous inter-service communication

**CQRS (Command Query Responsibility Segregation)** - Separate read/write operations

**Circuit Breaker Pattern** - Fault tolerance and resilience

*High-Level Architecture Diagram*

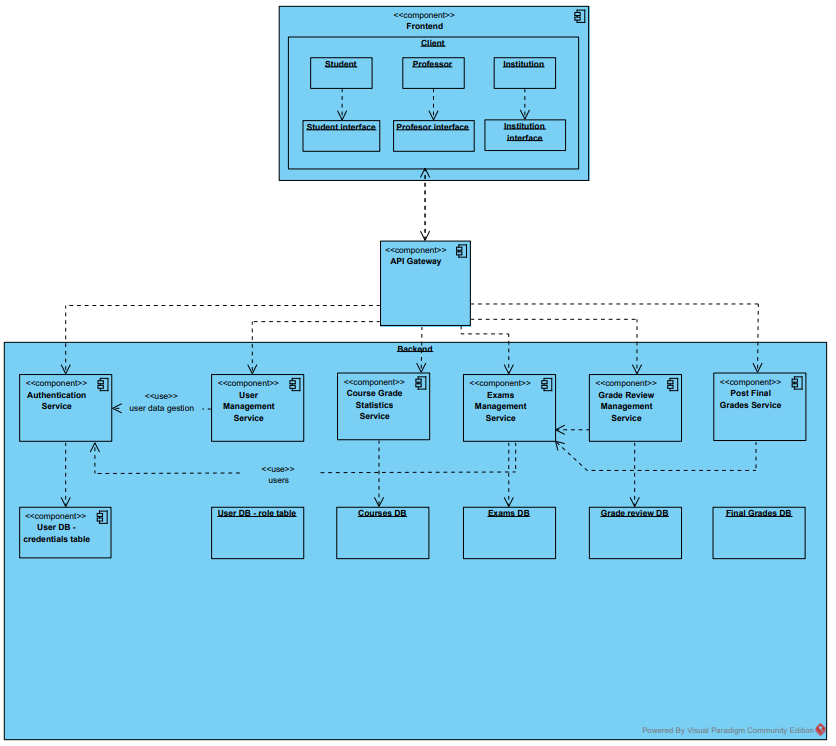
****

Fig.1: ClearSKY Microservices Architecture

**2.3 Microservices Architecture Benefits**

The microservices approach provides several critical advantages for an educational platform like ClearSKY. Service Independence allows different teams to work on different aspects of the system without coordination overhead, enabling faster development cycles and more focused expertise development. *Technology flexibility* means that each service can adopt the most appropriate technology stack for its specific requirements, from high-performance data processing to user interface optimization.

*Scalability precision* enables the system to scale individual components based on actual usage patterns rather than scaling the entire system uniformly. For example, during peak grade submission periods, only the Grade Management Service needs additional resources, while other services can maintain baseline capacity.

*Fault tolerance* ensures that issues in one service don't cascade to others, maintaining system availability even when individual components experience problems. This is particularly critical in educational environments where system availability directly impacts academic processes.

* 1. **Comprehensive Service Architecture**

**3.1 Authentication Service - Identity Foundation**

The Authentication service serves as the security foundation for the entire ClearSKY ecosystem, implementing sophisticated identity and access management capabilities that go far beyond simple username and password validation. This service handles the complex requirements of educational environments where users may have multiple roles across different institutions and courses.

The service implements JSON Web Token (JWT) based authentication with advanced security features including token rotation, blacklisting capabilities for immediate access revocation, and sophisticated session management. Password security follows industry best practices with bcrypt hashing, salt generation, and configurable complexity requirements that can be customized per institution.

Role-based access control is implemented with fine-grained permissions that can be dynamically assigned based on user context. For example, an instructor may have different permission levels for courses they teach versus courses where they serve as teaching assistants. The service maintains detailed audit logs of all authentication events, providing comprehensive security monitoring and compliance capabilities.

The service also handles password reset workflows, account lockout policies after failed attempts, and integration with external identity providers for institutions that require single sign-on capabilities. Multi-factor authentication support ensures that sensitive academic data remains protected even if primary credentials are compromised.

* 1. **User Management Service - Profile and Relationship Hub**

The User Management Service orchestrates the complex web of relationships that exist within educational institutions. Beyond basic user profile management, this service handles the intricate mappings between users and their various institutional affiliations, course enrollments, and role assignments.

The service maintains comprehensive user profiles that include not only basic demographic information but also academic history, institutional affiliations, and role progression over time. This temporal aspect is crucial in educational environments where users may transition between roles (student to teaching assistant to instructor) or maintain multiple simultaneous roles.

Institution-user relationships are managed with sophisticated hierarchical structures that can accommodate complex organizational arrangements. The service handles scenarios where users may be affiliated with multiple institutions, departments, or programs simultaneously. It maintains detailed records of enrollment periods, role effective dates, and historical changes to user status.

The Foreign Key Integration capability is particularly important for ClearSKY's Excel-based grade upload functionality. The service maintains mappings between institutional student identification numbers and internal system identifiers, enabling seamless integration with existing institutional systems while maintaining data integrity and privacy.

* 1. **Course Management Service - Academic Structure Engine**

The Course Management Service encapsulates the complex academic structures that define educational programs. This service goes beyond simple course catalogs to implement sophisticated academic organizational structures that can accommodate diverse educational models from traditional semester systems to modular program structures.

Course lifecycle management includes creation, modification, archival, and reactivation workflows that respect academic calendar constraints and institutional policies. The service maintains detailed course metadata including prerequisites, co-requisites, credit values, and academic level designations that can be used for automated validation and student guidance.

Enrollment management capabilities handle the complex business rules around course registration, capacity limits, prerequisite validation, and waitlist management. The service can enforce institutional policies such as maximum credit loads, program requirements, and academic standing requirements.

The service also manages academic periods and calendars, enabling institutions to configure complex academic structures including traditional semesters, quarters, modules, and intensive programs. This flexibility ensures that ClearSKY can accommodate diverse educational models and institutional requirements.

* 1. **Grade Management Service - Core Academic Engine**

The Grade Management Service represents the heart of the ClearSKY platform, implementing sophisticated grade processing workflows that maintain academic integrity while providing flexibility for diverse grading models. This service handles the complete grade lifecycle from initial entry through final submission and archival.

Excel file processing capabilities include advanced validation logic that can detect common data entry errors, format inconsistencies, and referential integrity violations before grades are committed to the system. The service supports multiple Excel formats and can be configured to accommodate institution-specific templates and requirements.

Grade state management implements a rigorous state machine that ensures grades progress through defined states (NULL, OPEN, FINAL) with appropriate validation and authorization checks at each transition. This prevents unauthorized modifications and maintains audit trails for all grade changes.

Statistical calculation capabilities provide real-time computation of grade distributions, class averages, standard deviations, and percentile rankings. These calculations are performed asynchronously to avoid impacting user experience while ensuring that analytics are always current.

The service also implements sophisticated conflict resolution for scenarios where grades are modified during active review processes, ensuring that all stakeholders are notified of changes and that review requests remain valid throughout the process.

* 1. **Grade Review Service - Communication and Workflow Hub**

The Grade Review Service orchestrates the complex communication workflows between students and instructors regarding grade disputes or clarifications. This service implements sophisticated workflow management that ensures fair, timely, and transparent review processes while maintaining appropriate academic protocols.

Review request processing includes validation logic that ensures requests are submitted within appropriate timeframes, contain required information, and are directed to the correct instructors. The service can enforce institutional policies regarding review deadlines, required documentation, and escalation procedures.

Communication management maintains threaded conversations between students and instructors, ensuring that all parties have access to complete communication history while maintaining appropriate privacy boundaries. The service supports file attachments for supporting documentation and integrates with notification systems to ensure timely responses.

Workflow automation capabilities can route complex review requests through multi-level approval processes, escalate overdue responses, and automatically close reviews that meet institutional criteria. These workflows can be customized per institution to accommodate different academic policies and procedures.

The service also maintains comprehensive metrics on review patterns, response times, and resolution outcomes, providing valuable insights for institutional quality assurance and continuous improvement efforts.

* 1. **Analytics Service - Intelligence and Insights Engine**

The Analytics Service transforms raw academic data into actionable insights for all stakeholders within the ClearSKY ecosystem. This service implements sophisticated data processing pipelines that can handle large volumes of grade data while providing real-time insights and historical trend analysis.

Performance analytics capabilities include student progress tracking, course difficulty analysis, instructor effectiveness metrics, and institutional benchmark comparisons. These analytics help identify at-risk students, highlight exceptional performance, and guide curriculum development decisions.

Predictive modeling capabilities use historical data patterns to identify potential issues before they become critical, such as predicting which students may benefit from additional support or identifying courses that may need curriculum adjustments based on grade distribution patterns.

Real-time dashboard generation provides stakeholders with up-to-date information presented in intuitive, actionable formats. Different user roles see different analytics dashboards tailored to their specific needs and responsibilities.

The service also implements sophisticated privacy protection measures to ensure that individual student data remains confidential while still providing valuable aggregate insights. All analytics computations respect institutional privacy policies and regulatory requirements.

* 1. **Technology Stack and Implementation Details**

**4.1 Frontend Architecture Philosophy**

The frontend implementation emphasizes user experience optimization while maintaining the flexibility to serve diverse user groups with different needs and technical proficiency levels. The React-based architecture provides a modern, responsive interface that adapts seamlessly across desktop, tablet, and mobile devices.

Component architecture follows atomic design principles, creating reusable interface elements that ensure consistency across the platform while enabling rapid development of new features. The state management approach uses Redux Toolkit for complex application state while leveraging React's built-in state management for component-level interactions.

Accessibility compliance ensures that the platform serves users with diverse abilities, implementing WCAG guidelines for screen reader compatibility, keyboard navigation, and visual accessibility features. This is particularly important in educational environments where accessibility compliance is often legally required.

Performance optimization includes code splitting for faster initial load times, lazy loading of non-critical components, and sophisticated caching strategies that balance data freshness with response time requirements.

**4.2 Backend Service Implementation**

Each microservice follows a consistent architectural pattern while being optimized for its specific responsibilities. The Node.js runtime provides excellent performance for I/O-intensive operations while maintaining developer productivity through JavaScript's flexibility and ecosystem richness.

*Express.js* framework implementation includes custom middleware for authentication, logging, error handling, and request validation. Each service implements standardized health check endpoints, metrics collection, and logging formats that enable comprehensive system monitoring and debugging.

Database abstraction layers provide flexibility for future technology changes while ensuring optimal performance for current requirements. Each service uses connection pooling, query optimization, and caching strategies appropriate to its specific data access patterns.

Error handling implementation includes circuit breaker patterns for external service dependencies, retry logic for transient failures, and comprehensive error logging that enables rapid issue identification and resolution.

**4.3 Database Design Philosophy**

The database architecture implements the "database per service" pattern, ensuring that each microservice has complete autonomy over its data while maintaining referential integrity through well-defined API contracts rather than direct database relationships.

Each service database is optimized for its specific access patterns, with appropriate indexing strategies, partitioning schemes, and backup procedures. The PostgreSQL choice provides ACID compliance for critical academic data while offering the flexibility to implement advanced features like JSON document storage for configuration data.

Data migration strategies ensure that schema changes can be deployed without system downtime, using techniques like blue-green deployments and backward-compatible schema modifications.

Privacy and security implementation includes field-level encryption for sensitive data, comprehensive audit logging, and role-based access controls that extend from the application layer down to the database level.

* 1. **Service Integration and Communication Patterns**

**5.1 API Gateway Implementation Strategy**

The *API Gateway* serves as more than just a routing layer; it implements sophisticated traffic management, security enforcement, and service composition capabilities. Request routing logic includes intelligent load balancing, circuit breaker protection, and automatic failover to backup service instances.

Authentication and authorization enforcement happens at the gateway level, providing a single point of security control while enabling fine-grained authorization decisions to be made within individual services based on user context and request parameters.

Rate limiting and throttling capabilities protect backend services from overload while ensuring fair resource allocation among different user types and usage patterns. Educational environments often experience burst traffic during grade posting periods, and the gateway's traffic shaping capabilities ensure system stability during these peak periods.

Request and response transformation capabilities enable the gateway to adapt between different API versions, aggregate responses from multiple services, and optimize data transfer by filtering unnecessary information based on client capabilities and requirements.

**5.2 Inter-Service Communication Design**

Service-to-service communication follows well-defined patterns that ensure reliability, performance, and maintainability. Synchronous communication uses RESTful APIs for real-time operations that require immediate consistency, while asynchronous communication handles background processing and event-driven workflows.

Event-driven architecture implementation enables loose coupling between services while maintaining data consistency through eventual consistency patterns. Events are used for notifications, audit logging, and triggering background processes like analytics calculations.

Service discovery mechanisms ensure that services can locate and communicate with each other regardless of deployment topology. This enables flexible deployment strategies and automatic scaling without manual configuration updates.

Retry and circuit breaker patterns protect the system from cascading failures while ensuring that transient issues don't impact user experience. These patterns are particularly important in educational environments where system reliability directly impacts academic processes.

* 1. **Security Architecture and Implementation**

**6.1 Comprehensive Security Strategy**

Security implementation goes far beyond basic authentication to encompass data protection, privacy compliance, and threat mitigation appropriate for educational environments that handle sensitive student information.

Identity and access management includes multi-factor authentication options, single sign-on integration capabilities, and sophisticated role-based access controls that can be customized per institution. The system maintains detailed audit logs of all access attempts and authorization decisions.

Data protection implementation includes encryption at rest and in transit, secure file upload handling with virus scanning, and sophisticated input validation that prevents injection attacks and data corruption. All sensitive data is encrypted using industry-standard algorithms with appropriate key management procedures.

Privacy compliance capabilities ensure that the platform can meet regulatory requirements like FERPA in the United States or GDPR in Europe. This includes data minimization principles, user consent management, and comprehensive data lifecycle management including secure deletion capabilities.

Threat mitigation includes automated monitoring for suspicious activity patterns, protection against common web application vulnerabilities, and incident response procedures that minimize impact while preserving forensic evidence.

* 1. **Performance and Scalability Architecture**

**7.1 Scalability Strategy Implementation**

The microservices architecture enables precise scaling decisions based on actual usage patterns rather than uniform scaling across the entire system. Each service can be scaled independently based on its specific performance characteristics and demand patterns.

Database optimization includes sophisticated indexing strategies, query optimization, and caching layers that minimize database load while ensuring data consistency. Read replicas are used for analytics queries that don't require real-time data, reducing load on primary database instances.

Caching implementation operates at multiple levels, from database query caching to full response caching, with sophisticated cache invalidation strategies that ensure data freshness while maximizing performance benefits.

Load balancing strategies include both horizontal scaling of service instances and intelligent request routing that can direct requests to the most appropriate service instance based on current load, geographic location, and request characteristics.

Performance monitoring includes comprehensive metrics collection, real-time alerting for performance degradation, and automated scaling triggers that can respond to demand changes without manual intervention.

* 1. **Summary Tables**

**8.1 Service Responsibility Matrix**

| **Service** | **Primary Domain** | **Key Responsibilities** | **Technology Focus** | **Scaling Characteristics** |
| --- | --- | --- | --- | --- |
| **Authentication** | Identity & Access | User login, JWT management, RBAC, session control | Security, cryptography | CPU-intensive, stateless |
| **User Management** | Profile Administration | User CRUD, role assignment, institutional relationships | Data management, validation | Database-intensive, moderate scaling |
| **Course Management** | Academic Structure | Course lifecycle, enrollment, academic calendars | Business logic, workflow | Stable load, predictable scaling |
| **Grade Management** | Core Grading | Excel processing, grade states, statistics | File processing, computation | High CPU during upload peaks |
| **Review Management** | Communication | Review workflows, student-instructor communication | Workflow orchestration | Communication-pattern dependent |
| **Analytics** | Insights & Reporting | Performance metrics, trend analysis, dashboards | Data processing, visualization | Memory-intensive, batch processing |

**8.2 Technology Stack Summary**

| **Layer** | **Component** | **Technology Choice** | **Justification** | **Scalability Impact** |
| --- | --- | --- | --- | --- |
| Frontend | UI Framework | React 18.x | Modern ecosystem, component reusability | Client-side rendering reduces server load |
| Frontend | State Management | Redux Toolkit | Predictable state updates, dev tools | Minimizes server round-trips |
| Gateway | Routing Engine | Express.js | Mature ecosystem, middleware flexibility | Horizontal scaling, stateless design |
| Services | Runtime | Node.js 18.x | JavaScript consistency, async I/O | Excellent concurrent request handling |
| Services | Framework | Express.js | Rapid development, extensive middleware | Lightweight, fast request processing |
| Database | Primary Store | PostgreSQL 13+ | ACID compliance, JSON support | Proven scaling patterns, read replicas |
| Infrastructure | Containers | Docker + Compose | Consistent environments, easy deployment | Rapid scaling, resource efficiency |
| Security | Authentication | JWT + bcrypt | Stateless tokens, secure hashing | No server-side session storage required |

**8.3 Business Process Flow Matrix**

| **Process** | **Primary Service** | **Supporting Services** | **User Roles Involved** | **Complexity Level** |
| --- | --- | --- | --- | --- |
| User Registration | User Management | Authentication, Course Management | Institution Rep, Admin | Medium |
| Grade Upload | Grade Management | Authentication, User Management, Analytics | Instructor | High |
| Grade Review Request | Review Management | Authentication, Grade Management, User Management | Student, Instructor | Medium |
| Review Response | Review Management | Grade Management, Analytics,User Management | Instructor | Medium |
| Final Grade Submission | Grade Management | Authentication, Analytics, Course Management | Instructor | High |
| Performance Analytics | Analytics | Grade Management, Course Management, User Management | All Roles | High |
| Credit Management | User Management | Authentication, Grade Management | Institution Rep | Low |

**8.4 Data Flow Complexity Analysis**

| **Integration Point** | **Data Volume** | **Frequency** | **Consistency Requirements** | **Performance Impact** |
| --- | --- | --- | --- | --- |
| Grade Upload Processing | High | Periodic (batch) | Strong consistency | High CPU, disk I/O |
| User Authentication | Low | Continuous | Immediate consistency | Low, cacheable |
| Analytics Calculation | Very High | Scheduled/triggered | Eventual consistency | High memory, CPU |
| Review Notifications | Medium | Event-driven | Eventual consistency | Low, async processing |
| Course Enrollment | Low | Periodic | Strong consistency | Low, transactional |
| Performance Dashboards | High | On-demand | Near real-time | Medium, cached responses |

**8.5 Deployment and Operations Summary**

| **Aspect** | **Implementation** | **Benefits** | **Monitoring Requirements** |
| --- | --- | --- | --- |
| Containerization | Docker per service | Consistency, isolation, portability | Resource usage, health checks |
| Orchestration | Docker Compose (dev), Kubernetes (prod) | Automated scaling, service discovery | Cluster health, service mesh metrics |
| Database Management | Per-service databases | Service autonomy, targeted optimization | Connection pools, query performance |
| Load Balancing | Nginx reverse proxy | High availability, SSL termination | Response times, error rates |
| Security | Multi-layer (gateway, service, database) | Defense in depth | Authentication failures, access patterns |
| Backup Strategy | Automated daily backups | Data protection, disaster recovery | Backup success, restore testing |

This comprehensive documentation provides a complete understanding of the ClearSKY microservices architecture, emphasizing the educational domain expertise while showcasing modern software engineering practices. The system successfully bridges traditional academic processes with cutting-edge technology solutions, providing a robust foundation for modern educational grade management.