CreatorSync Technology Stack Documentation

Executive Summary

This document outlines the comprehensive technology stack for CreatorSync, a SaaS platform designed for content creators managing their presence across TikTok, Instagram Reels, and X (Twitter). The technology choices prioritize scalability, performance, security, and developer productivity to deliver a robust platform that can grow with user demand while maintaining high availability and responsiveness.

The stack is organized into frontend, backend, database, infrastructure, DevOps, security, and third-party integrations. Each component has been selected based on specific requirements, industry best practices, and alignment with the development team's expertise.

Table of Contents

Architecture Overview

CreatorSync employs a microservices architecture to enable independent scaling, development, and deployment of different system components. The overall architecture follows these key principles:

Architectural Principles

1. **Separation of Concerns**: Each microservice has a specific domain responsibility
2. **API-First Design**: All services communicate through well-defined APIs
3. **Stateless Services**: Services maintain minimal state for horizontal scalability
4. **Event-Driven Communication**: Asynchronous communication for non-critical operations
5. **Defense in Depth**: Multiple layers of security controls
6. **Observability**: Comprehensive monitoring and logging throughout the stack

High-Level Architecture Diagram

```
                         Client Layer


   Web Client         Mobile Client         API
   (React)            (React Native)       Consumers





                       API Gateway Layer
```

```
     API Gateway          Auth/Auth          Rate Limiter
     (Kong)               Service




                        Microservices Layer


     Content     Analytics   Monetization   Community
     Service      Service      Service       Service



     Growth        User       Platform      Team
     Service      Service     Connectors    Service





                          Data Layer


     PostgreSQL           MongoDB            Redis
    (Relational)         (Document)          (Cache)



     Elasticsearch        S3/Blob            Kafka
     (Search)            (Storage)           (Events)
```

Service Boundaries

1. **Content Service**: Content creation, scheduling, and management
2. **Analytics Service**: Performance metrics, reporting, and insights
3. **Monetization Service**: Revenue tracking, deals, and financial data
4. **Community Service**: Comment management, messaging, and engagement
5. **Growth Service**: Recommendations, trends, and growth strategies
6. **User Service**: User management, authentication, and preferences
7. **Platform Connectors**: Integration with social media platforms
8. **Team Service**: Collaboration, permissions, and workflow

Frontend Stack

Core Technologies

Technology	Version	Purpose
React	18.x	UI library for component-based development
Next.js	14.x	React framework for server-side rendering and routing
TypeScript	5.x	Type-safe JavaScript superset
Tailwind CSS	3.x	Utility-first CSS framework

State Management

Technology	Version	Purpose
Redux Toolkit	2.x	Global state management
React Query	5.x	Server state management and data fetching
Zustand	4.x	Lightweight state management for component-level state

UI Components

Technology	Version	Purpose
shadcn/ui	Latest	Component library built on Radix UI
Radix UI	Latest	Unstyled, accessible component primitives
Framer Motion	10.x	Animation library
Lucide Icons	Latest	Icon library

Data Visualization

Technology	Version	Purpose
Recharts	2.x	React charting library
D3.js	7.x	Data visualization library for complex visualizations
react-table	8.x	Table management

Testing

Technology	Version	Purpose
Jest	29.x	JavaScript testing framework
React Testing Library	14.x	React component testing
Cypress	13.x	End-to-end testing
Storybook	7.x	Component documentation and testing

Build Tools

Technology	Version	Purpose
Vite	5.x	Build tool and development server
ESLint	8.x	JavaScript linting
Prettier	3.x	Code formatting
PostCSS	8.x	CSS processing

Backend Stack

Core Technologies

Technology	Version	Purpose
Node.js	20.x LTS	JavaScript runtime
NestJS	10.x	TypeScript backend framework
Express	4.x	Web framework (used by NestJS)
TypeScript	5.x	Type-safe JavaScript superset

API

Technology	Version	Purpose
REST	-	Primary API architecture
GraphQL	16.x	Secondary API for complex data fetching
Apollo Server	4.x	GraphQL server implementation
Swagger/OpenAPI	3.x	API documentation

Authentication & Authorization

Technology	Version	Purpose
Passport.js	0.6.x	Authentication middleware
JWT	-	Token-based authentication
OAuth 2.0	-	Social login and platform authentication
CASL	6.x	Authorization library

Background Processing

Technology	Version	Purpose
Bull	4.x	Redis-based queue for background jobs
Node-cron	3.x	Scheduled tasks
pm2	5.x	Process manager for Node.js

Testing

Technology	Version	Purpose
Jest	29.x	Testing framework
Supertest	6.x	HTTP testing
Pactum	3.x	API testing and contract testing
k6	0.45.x	Load testing

Validation & Error Handling

Technology	Version	Purpose
class-validator	0.14.x	Input validation
Joi	17.x	Schema validation

Technology	Version	Purpose
Winston	3.x	Logging
Sentry	Latest	Error tracking

Database Stack

Primary Databases

Technology	Version	Purpose
PostgreSQL	16.x	Relational database for transactional data
MongoDB	7.x	Document database for content and analytics data

Database Access

Technology	Version	Purpose
TypeORM	0.3.x	ORM for PostgreSQL
Mongoose	8.x	MongoDB ODM
Prisma	5.x	Database toolkit and ORM

Caching

Technology	Version	Purpose
Redis	7.x	In-memory data store for caching
KeyDB	6.x	Redis-compatible cache for high throughput

Search

Technology	Version	Purpose
Elasticsearch	8.x	Search engine for content and analytics
Meilisearch	1.x	Lightweight search for specific features

Data Processing

Technology	Version	Purpose
Apache Kafka	3.x	Event streaming platform
Apache Spark	3.x	Data processing for analytics
Databricks	Latest	Managed Spark for advanced analytics

Storage

Technology	Version	Purpose
AWS S3	-	Object storage for media and assets
MinIO	Latest	S3-compatible storage for development

Infrastructure Stack

Cloud Provider

Technology	Purpose
AWS	Primary cloud provider
GCP	Secondary provider for specific services

Compute

Technology	Purpose
AWS ECS	Container orchestration
AWS Fargate	Serverless container execution
AWS Lambda	Serverless functions for specific workloads

Networking

Technology	Purpose
AWS VPC	Network isolation
AWS CloudFront	CDN for static assets
AWS Route 53	DNS management
AWS API Gateway	API management for serverless functions

Database Services

Technology	Purpose
AWS RDS	Managed PostgreSQL
MongoDB Atlas	Managed MongoDB
AWS ElastiCache	Managed Redis
AWS OpenSearch	Managed Elasticsearch

Storage Services

Technology	Purpose
AWS S3	Object storage
AWS EFS	File storage for persistent volumes

Monitoring & Observability

Technology	Purpose
AWS CloudWatch	Metrics and logging
Datadog	Comprehensive monitoring
New Relic	Application performance monitoring
Grafana	Metrics visualization
Prometheus	Metrics collection

DevOps Stack

CI/CD

Technology	Purpose
GitHub Actions	CI/CD pipeline
AWS CodePipeline	Secondary CI/CD pipeline
ArgoCD	GitOps for Kubernetes

Infrastructure as Code

Technology	Version	Purpose
Terraform	1.5.x	Infrastructure provisioning
AWS CDK	2.x	Infrastructure as code for AWS
Pulumi	3.x	Infrastructure as code for multi-cloud

Containerization

Technology	Version	Purpose
Docker	Latest	Container runtime
Docker Compose	Latest	Local development orchestration
AWS ECR	-	Container registry

Monitoring & Alerting

Technology	Purpose
Datadog	Comprehensive monitoring
PagerDuty	Incident management and alerting
Sentry	Error tracking
ELK Stack	Logging (Elasticsearch, Logstash, Kibana)

Development Tools

Technology	Purpose
Git	Version control
GitHub	Code hosting and collaboration
Jira	Project management
Confluence	Documentation

Security Stack

Authentication & Authorization

Technology	Purpose
Auth0	Identity as a service
AWS Cognito	User management and authentication

Technology	Purpose
OIDC	Open standard for authentication

Security Monitoring

Technology	Purpose
AWS GuardDuty	Threat detection
AWS Security Hub	Security posture management
Snyk	Dependency vulnerability scanning
OWASP ZAP	Security testing

Data Protection

Technology	Purpose
AWS KMS	Key management
Vault by HashiCorp	Secrets management
AWS Certificate Manager	SSL/TLS certificate management

Compliance

Technology	Purpose
AWS Artifact	Compliance reporting
AWS Config	Configuration and compliance
AWS CloudTrail	API auditing

Third-Party Integrations

Social Media Platforms

Platform	API	Purpose
TikTok	TikTok for Developers API	Content publishing, analytics
Instagram	Instagram Graph API	Content publishing, analytics
X (Twitter)	X API v2	Content publishing, analytics

Analytics Integrations

Service	Purpose
Google Analytics	Web analytics
Amplitude	Product analytics
Mixpanel	User behavior analytics
Segment	Customer data platform

Payment Processing

Service	Purpose
Stripe	Subscription billing
PayPal	Alternative payment method
Chargebee	Subscription management

Communication

Service	Purpose
SendGrid	Transactional email
Twilio	SMS notifications
Pusher	Real-time notifications
Slack	Team notifications and integrations

Content Delivery

Service	Purpose
Cloudinary	Media optimization and delivery
AWS MediaConvert	Video transcoding
Mux	Video streaming

Marketing Automation

Service	Purpose
HubSpot	Marketing automation
Customer.io	Customer messaging
Intercom	Customer engagement

Development Environment

Local Development

Technology	Purpose
Docker Desktop	Containerized development environment
VS Code	Primary IDE
ESLint/Prettier	Code quality and formatting
Husky	Git hooks for pre-commit checks
pnpm	Package manager

Development Workflow

1. **Local Development**:
   - Docker Compose for local service orchestration
   - Hot reloading for frontend and backend
   - Local database seeding
2. **Testing Environment**:
   - Automated deployment from feature branches
   - Ephemeral environments for testing

    - Synthetic data generation
3. **Staging Environment**:
    - Production-like configuration
    - Integration testing
    - Performance testing
4. **Production Environment**:
    - Blue/green deployment
    - Canary releases
    - Feature flags

Scaling Strategy

Horizontal Scaling

- **Frontend**: CDN with edge caching
- **API Layer**: Load balancing with auto-scaling
- **Microservices**: Independent scaling based on load
- **Databases**: Read replicas and sharding

Vertical Scaling

- **Database**: Instance size upgrades for write-heavy workloads
- **Compute**: Instance size optimization based on workload characteristics
- **Memory**: Cache size adjustments based on hit rates

Caching Strategy

1. **Browser Caching**: Static assets with appropriate cache headers
2. **CDN Caching**: Edge caching for static content
3. **API Caching**: Response caching for frequently accessed data
4. **Database Caching**: Query result caching with Redis
5. **Application Caching**: In-memory caching for frequently accessed data

Performance Optimization

1. **Code Optimization**: Regular performance audits and optimizations
2. **Database Optimization**: Index optimization, query tuning
3. **Asset Optimization**: Image and media compression, lazy loading
4. **Network Optimization**: HTTP/2, connection pooling, keep-alive

Technical Debt Management

Identification

- Regular code quality metrics review
- Architectural decision records (ADRs)
- Technical debt tagging in issue tracker
- Scheduled technical debt review sessions

Prioritization

- Impact assessment matrix
- Risk-based prioritization
- Technical debt interest calculation
- Regular refactoring sprints

Prevention

- Code review standards
- Automated testing requirements
- Documentation requirements
- Architecture review process

Technology Evaluation Criteria

The following criteria were used to evaluate and select technologies for the CreatorSync stack:

Functional Criteria

1. **Feature Completeness**: Ability to meet functional requirements
2. **Performance**: Speed and efficiency under expected load
3. **Scalability**: Ability to handle growth in users and data
4. **Reliability**: Stability and fault tolerance
5. **Security**: Built-in security features and community security focus

Non-Functional Criteria

1. **Developer Experience**: Ease of use, documentation quality
2. **Community Support**: Size and activity of community
3. **Maturity**: Production readiness and stability
4. **Licensing**: Compatible with commercial use
5. **Cost**: Total cost of ownership

Strategic Criteria

1. **Long-term Viability**: Project sustainability and roadmap
2. **Ecosystem Compatibility**: Integration with other selected technologies
3. **Talent Availability**: Ease of finding developers with relevant skills
4. **Vendor Lock-in Risk**: Ability to migrate if needed
5. **Innovation Pace**: Rate of improvement and feature addition

Conclusion

The CreatorSync technology stack has been carefully designed to provide a robust, scalable, and maintainable platform for content creators. By leveraging modern technologies and best practices, the stack enables rapid development while ensuring the platform can scale to meet growing demand.

Key highlights of the stack include:

1. **Modern Frontend**: React with Next.js provides an excellent developer experience and optimal performance for users.

2. **Scalable Backend**: NestJS with TypeScript offers a structured, maintainable backend architecture.

3. **Polyglot Persistence**: Multiple database technologies chosen for their specific strengths in different domains.

4. **Cloud-Native Infrastructure**: AWS-focused infrastructure with containerization for scalability and reliability.

5. **Comprehensive DevOps**: Automated CI/CD, monitoring, and infrastructure as code for operational excellence.

6. **Security by Design**: Multiple layers of security controls integrated throughout the stack.

7. **Extensible Integration Layer**: Well-defined APIs and connectors for third-party services.

This technology stack provides a solid foundation for building and scaling CreatorSync to serve content creators across TikTok, Instagram Reels, and X platforms.