

# BANITY PLATFORM

## Technical Specification & Implementation Guide

Document Version:	1.0
Date:	December 04, 2025
Status:	Planning Phase - Ready for Development
Total Duration:	16 Weeks (4 Months)
Estimated Launch:	Q2 2026
Starting Cost:	\$0/month (Free Tier)

# EXECUTIVE SUMMARY

Banity is a creator-brand connection platform facilitating paid UGC campaigns. This document provides a comprehensive technical specification covering architecture, security, user experience, mobile integration, and development timeline. The platform leverages modern, cloud-native technologies with free-tier options allowing \$0 startup costs while maintaining enterprise-grade security and scalability to 1M+ users.

## Key Objectives:

- **Security:** JWT authentication, encrypted storage, comprehensive audit logging
- **User Experience:** Intuitive interfaces with <3 minute onboarding time
- **Scalability:** Free tier → 1M+ users without major refactoring
- **Mobile-First:** Unified API for web and iOS applications
- **Cost-Efficiency:** \$0/month for 0-1K users, \$50-100/month for 1K-10K users

# SYSTEM ARCHITECTURE

The platform follows a modern, cloud-native architecture with clear separation of concerns. Frontend (Astro.js) communicates with RESTful API (Netlify Functions), which interfaces with PostgreSQL database, Redis cache, and cloud storage (S3/Cloudinary).

Layer	Technology	Purpose
Presentation	Astro, SwiftUI	Web & mobile UIs
API Gateway	Netlify Functions	Request routing, rate limiting
Application	Node.js/JavaScript	Business logic
Data	PostgreSQL, Redis	Persistent & cache storage
Storage	Cloudinary/S3	Media files
Services	SendGrid, OneSignal	Email & push notifications

# DATABASE SCHEMA

Five primary tables with well-defined relationships:

Table	Purpose	Key Columns
creators	User accounts	id, email, social_handle, status
admins	Admin accounts	id, username, password_hash
server_messages	Homepage announcements	id, type, text, is_active
submissions	Creator media	id, creator_id, media_urls, status
message_dismissals	User preferences	id, message_id, user_identifier

# SECURITY IMPLEMENTATION

- **Authentication:** JWT with RS256 asymmetric encryption, 24-hour token expiry
- **Password Security:** bcrypt hashing (cost factor 10), never store plain text
- **Input Validation:** Server-side validation, XSS prevention, SQL injection protection
- **Rate Limiting:** IP + user-based throttling (3-60 req/min depending on endpoint)
- **Data Encryption:** TLS 1.3 in transit, database encryption at rest
- **Audit Logging:** All admin actions logged with timestamps and user IDs
- **GDPR Compliance:** Data minimization, user rights (access, delete, export)

## Rate Limiting Strategy:

Endpoint Type	Limit	Window
Public endpoints	60 req/min	Per IP
Sign-up	5 req/min	Per IP
Login	3 attempts	15 minutes
Admin actions	30 req/min	Per user
File uploads	10 req/hour	Per creator

# DEVELOPMENT TIMELINE

**Total Duration: 16 Weeks (4 Months)**

Phase	Duration	Key Deliverables
1: Foundation	Weeks 1-3	Database setup, authentication, basic pages
2: Core Features	Weeks 4-6	Admin dashboard, mailing list, server messages
3: Media & Advanced	Weeks 7-9	File uploads, creator inbox, optimizations
4: Mobile App	Weeks 10-14	iOS app development and App Store launch
5: Testing & Launch	Weeks 15-16	QA, security audit, production deployment

## Critical Milestones:

- Week 3: Authentication working, creators can sign up
- Week 6: Admin dashboard functional with all core features
- Week 9: File uploads and inbox operational
- Week 14: iOS app live on App Store
- Week 16: Platform launches in production

# TECHNOLOGY STACK & COSTS

Recommended stack with generous free tiers:

Component	Technology	Free Tier	First Upgrade
Backend	Netlify Functions	125K req/mo	\$25/mo for 2M req
Database	Supabase	500MB	\$25/mo for 8GB
File Storage	Cloudinary	25GB/mo	\$89/mo for 166 credits
Email	Resend	3K emails/mo	\$20/mo for 50K
Cache	Upstash Redis	10K cmd/day	\$0.20 per 100K
Push Notifs	OneSignal	Unlimited	\$9/mo for features

## Cost Projection:

User Scale	Monthly Cost
0 - 1,000 users	\$0/month
1,000 - 10,000 users	\$50-100/month
10,000 - 100,000 users	\$200-500/month
100,000 - 1M users	\$1,000-5,000/month

# MOBILE APPLICATION INTEGRATION

iOS application uses unified API architecture with identical endpoints as web. Tokens stored securely in iOS Keychain, biometric authentication (Face ID/Touch ID) replaces password entry. Push notifications via APNs for campaign updates.

- **Offline Mode:** Cache data locally, queue actions, sync when reconnected
- **Biometric Auth:** Face ID / Touch ID for seamless login
- **Push Notifications:** Campaign opportunities and status updates
- **Camera Integration:** Direct photo/video capture for submissions
- **Background Sync:** Automatic data refresh on app activation
- **Deep Links:** Open specific content from notifications

## iOS Technology Stack:

Component	Technology
UI Framework	SwiftUI
Networking	URLSession + Alamofire
Local Storage	Core Data
Secure Storage	Keychain Services
Authentication	LocalAuthentication (Face ID)
Push Notifications	APNs
Image Loading	Kingfisher

## API ENDPOINTS SUMMARY

Endpoint	Method	Auth	Purpose
/api/creators/signup	POST	None	Creator registration
/api/admin/login	POST	None	Admin authentication
/api/messages/active	GET	None	Get homepage messages
/api/admin/messages	POST	Admin	Create server message
/api/admin/messages/:id	DELETE	Admin	Delete message
/api/admin/subscribers	GET	Admin	Get mailing list
/api/admin/subscribers/export	GET	Admin	Export CSV
/api/submissions	POST	Creator	Submit media
/api/admin/inbox	GET	Admin	View submissions

# SCALABILITY & PERFORMANCE

Metric	Target
Homepage Load Time	<1.5 seconds
API Response Time (p95)	<200ms
Database Query Time	<50ms
File Upload Speed	>5 MB/s
Cache Hit Rate	>80%
Platform Uptime	99.9% (43min downtime/month)

## Optimization Strategies:

- **Database Indexing:** B-tree indexes on frequently queried columns
- **Connection Pooling:** Reuse connections for 5-10x faster response
- **Redis Caching:** 30s TTL for active messages, 5min for stats
- **CDN Distribution:** Serve static assets from edge locations
- **Image Optimization:** Cloudinary auto-optimization (70% size reduction)
- **Lazy Loading:** Load data on demand
- **Pagination:** 50 items per page for consistent performance

# PRE-LAUNCH CHECKLIST

- All unit and integration tests passing
- Security audit completed (zero critical vulnerabilities)
- Performance testing meets targets
- HTTPS enforced with valid SSL certificate
- Environment variables configured in production
- Database backups automated and tested
- Monitoring dashboards configured
- Rate limiting active on all endpoints
- Email delivery tested end-to-end
- Push notifications working
- File upload flow tested with max file sizes
- Admin dashboard fully functional
- Privacy policy and terms published
- User documentation completed
- Disaster recovery plan documented
- On-call schedule established for launch week

# SUCCESS CRITERIA & NEXT STEPS

## Platform is ready for launch when:

- ✓ All security audits passed with no critical findings
- ✓ Performance benchmarks meet or exceed targets
- ✓ Creator sign-up flow completes in under 3 minutes
- ✓ Admin dashboard tasks complete in under 30 seconds
- ✓ iOS app achieves 4.5+ star rating in TestFlight
- ✓ Platform handles 1,000 concurrent users in load testing
- ✓ 99.9% uptime maintained for 2 weeks in staging

## Immediate Next Steps:

- **Week 1:** Project setup, Git repository, database schema creation
- **Week 2:** Authentication system implementation (JWT, bcrypt)
- **Week 3:** Creator sign-up and admin login pages
- **Week 4:** Begin admin dashboard development
- **Ongoing:** Daily standups, weekly sprint planning, bi-weekly demos

**This platform is designed to scale from 0 to 1M+ users while starting at \$0/month. The architecture is air-tight, user-friendly, and fully mobile-ready with iOS integration. Timeline is aggressive but achievable with dedicated team focus.**

Document prepared by: Technical Team

Generated: December 04, 2025 at 06:56 PM