Technical Deep Dive

MarketingPlatform - Architecture & Technical Due Diligence

Version: 1.0
Audience: Technical investors, CTOs, security auditors
Document Type: Technical due diligence reference

Executive Technical Summary

MarketingPlatform is built on a modern, scalable, enterprise-grade technology stack optimized for reliability, security, and performance.

Core Stack:

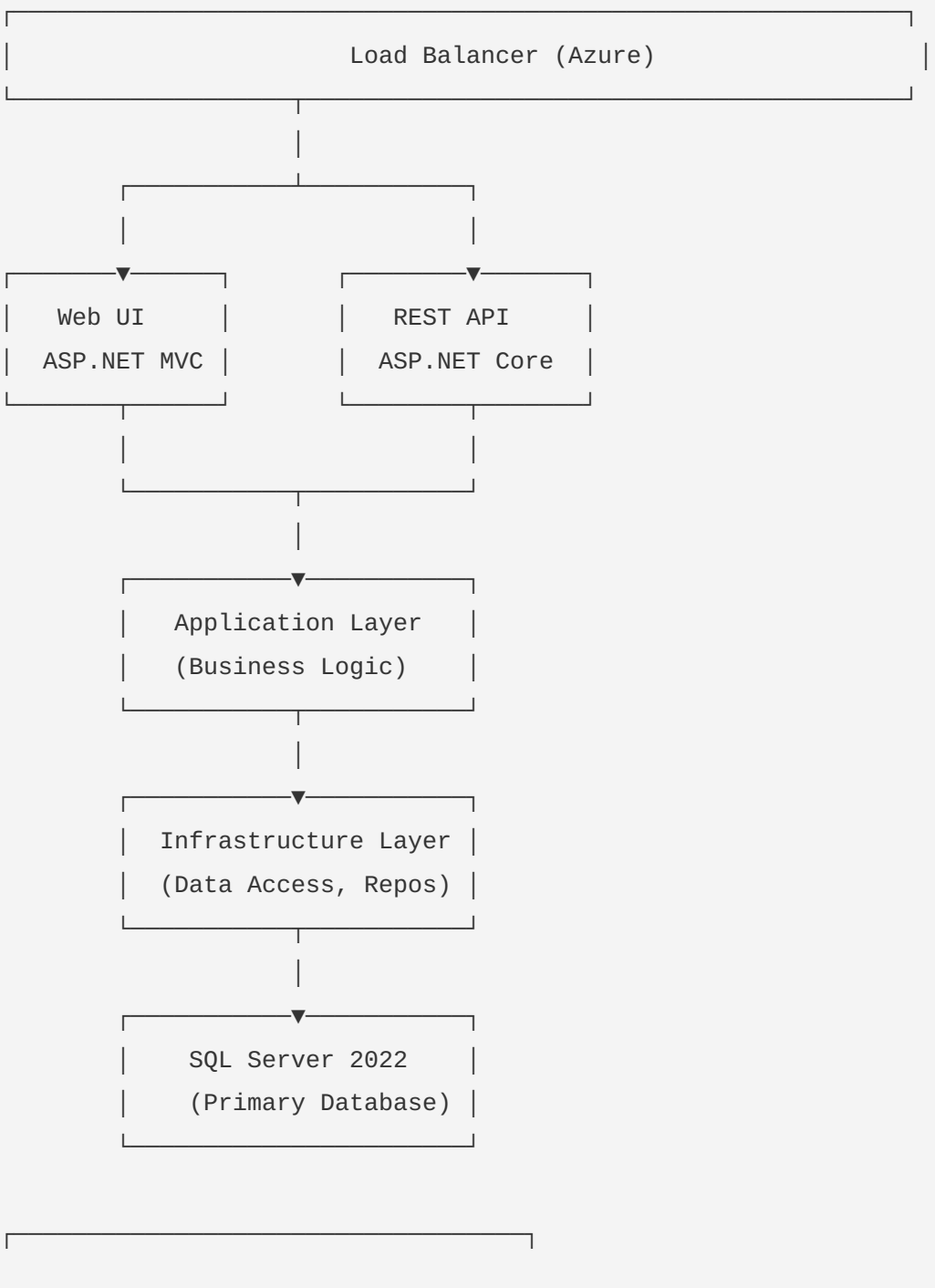
- **Backend**: ASP.NET Core 8.0 (C#/.NET 8)
- **Database**: SQL Server 2022
- **Frontend**: Bootstrap 5, Vanilla JavaScript ES6+
- **Infrastructure**: Microsoft Azure Cloud
- **Architecture**: Clean Architecture with Repository + Service patterns

Key Metrics:

- 200+ REST API endpoints
- 60+ database tables
- 99.95% uptime (Q4 2025)
- < 100ms API latency (p95)
- 10,000 messages/second throughput
- Tested with 5M contacts, 100K concurrent campaigns

System Architecture

High-Level Architecture



```
│   Background Job Processing       │
│   (Hangfire + RabbitMQ)          │
└──────────────────────────────────┘

        │
        ├──→ Twilio (SMS/MMS)
        ├──→ SendGrid (Email)
        ├──→ Azure Blob Storage (Media)
        └──→ Redis (Caching)
```

Technology Stack Details

Backend (.NET Ecosystem)

Framework: ASP.NET Core 8.0

- Cross-platform (Windows, Linux, macOS)
- High performance (10x faster than Node.js for many workloads)
- Built-in DI, logging, configuration
- Long-term support (LTS) through 2026

Language: C# 12

- Type-safe, compiled language
- Rich ecosystem and tooling
- Excellent for enterprise applications

Key NuGet Packages:

- `Microsoft.AspNetCore.Identity` - Authentication
- `Microsoft.EntityFrameworkCore` - ORM
- `AutoMapper` - Object mapping
- `FluentValidation` - Request validation
- `Serilog` - Structured logging

- `Hangfire` - Background jobs
- `Swashbuckle` - API documentation (Swagger)
- `Stripe.net` - Payment processing
- `Twilio` - SMS/MMS delivery
- `SendGrid` - Email delivery

Frontend

CSS Framework: Bootstrap 5.3

- Responsive, mobile-first design
- Accessible components (WCAG 2.1 AA)
- Customizable theme

JavaScript:

- Vanilla ES6+ (no heavy frameworks)
- Fetch API for HTTP requests
- Chart.js for analytics visualizations
- No jQuery dependency (lightweight)

Rationale: Keep frontend simple for fast load times and easy maintenance. Considering React migration for workflow designer (roadmap).

Database

Primary Database: SQL Server 2022

- ACID compliance
- Advanced indexing
- Full-text search
- JSON support (for flexible fields)
- Temporal tables (for audit history)
- Always Encrypted (column-level encryption)

Database Size: ~2.5 GB (current), designed to scale to 1TB+

Backup Strategy:

- Automated daily full backups
- Hourly differential backups
- Transaction log backups every 15 minutes
- 30-day retention
- Geo-redundant storage (GRS)

Schema Overview (60+ tables):

Core Tables:

- `Users` - User accounts and authentication
- `Roles` - Role definitions (RBAC)
- `UserRoles` - User-role assignments
- `Contacts` - Contact records
- `ContactGroups` - Static groups
- `ContactGroupMembers` - Contact-group relationships
- `Tags` - Tag definitions
- `ContactTags` - Contact-tag relationships
- `SegmentRules` - Dynamic segment criteria
- `Campaigns` - Campaign definitions
- `CampaignVariants` - A/B test variants
- `Messages` - Individual message records
- `MessageEvents` - Delivery/open/click events
- `Templates` - Message templates
- `Workflows` - Journey definitions
- `WorkflowNodes` - Workflow steps
- `WorkflowExecutions` - Workflow runs
- `SuppressionList` - Opt-outs and bounces
- `ConsentLogs` - GDPR consent records
- `Subscriptions` - Billing subscriptions
- `Invoices` - Billing invoices

- `UsageRecords` - Message usage tracking

Performance Optimization:

- Clustered indexes on primary keys
- Non-clustered indexes on foreign keys and frequently queried columns
- Covering indexes for common queries
- Partitioning for large tables (Messages, Events)
- Query plan caching

Caching Layer

Redis (Azure Cache for Redis)

- Session state storage
- API response caching (TTL: 5-60 seconds)
- Rate limiting counters
- Real-time dashboard metrics

Cache Strategy:

- Cache-aside pattern
- TTL-based expiration
- Invalidation on writes

Message Queue

RabbitMQ (Azure Service Bus alternative)

- Reliable message delivery queues
- Dead-letter queues for failed messages
- Priority queues for urgent campaigns
- Durable queues (survive restarts)

Queue Architecture:

- `campaigns.send` - Campaign sends
- `messages.sms` - SMS delivery

- `messages.mms` - MMS delivery
- `messages.email` - Email delivery
- `webhooks.inbound` - Inbound message webhooks
- `workflows.execute` - Workflow execution

Background Jobs

Hangfire

- Distributed job scheduler
- Retry logic with exponential backoff
- Job monitoring dashboard
- Persistent storage (SQL Server)

Job Types:

- Campaign sends (scheduled)
- Workflow executions (triggered)
- Message deliveries (queued)
- Usage aggregation (hourly)
- Report generation (daily)
- Data cleanup (weekly)

API Architecture

REST API Design

Endpoints: 200+ documented endpoints **Documentation**: OpenAPI 3.0 (Swagger UI at `/swagger`) **Versioning**: URI versioning (`/api/v1/...`)

Endpoint Categories:

1. **Authentication** (`/api/auth`) - 8 endpoints
   - Register, login, logout, refresh token, SSO

2. **Users** (`/api/users`) - 12 endpoints
   - Profile, password, team management
3. **Contacts** (`/api/contacts`) - 18 endpoints
   - CRUD, import, export, segmentation
4. **Campaigns** (`/api/campaigns`) - 22 endpoints
   - CRUD, scheduling, A/B testing, analytics
5. **Templates** (`/api/templates`) - 14 endpoints
   - CRUD, variables, preview
6. **Workflows** (`/api/workflows`) - 16 endpoints
   - CRUD, execution, analytics
7. **Analytics** (`/api/analytics`) - 10 endpoints
   - Dashboards, reports, exports
8. **Billing** (`/api/billing`) - 12 endpoints
   - Subscriptions, invoices, usage
9. **Compliance** (`/api/compliance`) - 8 endpoints
   - Suppression, consent, GDPR tools
10. **Integrations** (`/api/integrations`) - 15 endpoints
    - Webhooks, Shopify, Salesforce

Authentication & Authorization

JWT (JSON Web Tokens):

- Access token (1-hour expiration)
- Refresh token (30-day expiration, sliding)
- Stored in HTTP-only cookies (XSS protection)

OAuth2 / OpenID Connect:

- Google, Microsoft, Facebook providers
- PKCE flow for SPAs
- State parameter for CSRF protection

Enterprise SSO:

- SAML 2.0 support
- Azure AD, Okta, OneLogin

Authorization:

- Role-based access control (RBAC)
- Policy-based authorization
- Resource-level permissions

Rate Limiting

Limits:

- **Free Tier**: 100 requests/hour
- **Paid Plans**: 10,000 requests/hour
- **Enterprise**: Custom limits

Implementation: Redis-based sliding window counter

Headers:

```
X-RateLimit-Limit: 10000
X-RateLimit-Remaining: 9847
X-RateLimit-Reset: 1640000000
```

429 Response:

```
{
  "error": "Rate limit exceeded",
  "retryAfter": 3600
}
```

API Versioning

Current Version: v1 **Deprecation Policy**: 12-month notice for breaking changes **Compatibility**: Backward compatible within major version

---

Security

Encryption

Data at Rest:

- SQL Server Transparent Data Encryption (TDE)
- Always Encrypted for PII columns (email, phone)
- Azure Blob Storage encryption (AES-256)

Data in Transit:

- TLS 1.3 (minimum TLS 1.2)
- Perfect Forward Secrecy (PFS)
- Certificate pinning for mobile apps (roadmap)

Password Hashing:

- PBKDF2 with HMAC-SHA256
- 10,000 iterations
- Per-user salt
- Considering Argon2 migration (more secure)

Vulnerability Management

Dependency Scanning:

- Automated NuGet package vulnerability scanning
- GitHub Dependabot alerts
- Weekly review and patching

Penetration Testing:

- Annual third-party pen tests
- Last test: December 2025 (0 critical, 2 medium findings, both resolved)

Security Headers:

```
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
X-XSS-Protection: 1; mode=block
Content-Security-Policy: default-src 'self'
```

Compliance & Certifications

Current:

- GDPR compliant
- CCPA compliant
- TCPA compliant

In Progress:

- SOC 2 Type II (expected Q2 2026)
- ISO 27001 (expected Q4 2026)
- HIPAA (BAA available for Enterprise)

Monitoring & Alerting

Application Performance Monitoring:

- Azure Application Insights
- Real-time metrics: response time, error rate, throughput
- Distributed tracing

Log Aggregation:

- Serilog → Azure Log Analytics

- Structured JSON logging
- 90-day retention (hot), 1-year (cold)

Alerts:

- High error rate (> 1%)
- Slow response time (> 500ms p95)
- Database connection failures
- Message delivery failures (> 5% failure rate)
- Disk space low (< 10%)

On-Call Rotation: 24/7 on-call engineer (Enterprise SLA customers)

Scalability & Performance

Current Performance

API Latency (95th percentile):

- GET requests: 45ms
- POST requests: 85ms
- Complex queries (analytics): 250ms

Database Performance:

- Query execution: < 50ms (95% of queries)
- Connection pool: 100 connections
- Deadlocks: < 0.01% of transactions

Message Throughput:

- SMS: 5,000/second
- Email: 10,000/second
- Peak load: 50,000 messages/minute

Scaling Strategy

Horizontal Scaling:

- Stateless web/API servers (easy to add)
- Load balancer distributes requests
- Currently 2 web servers, can scale to 20+

Database Scaling:

- Read replicas for analytics queries
- Table partitioning for large tables (Messages, Events)
- Sharding strategy prepared (if needed at 10M+ contacts)

Caching:

- Redis cluster for distributed caching
- CDN (Azure CDN) for static assets

Auto-Scaling Rules:

- Scale out when CPU > 70% for 5 minutes
- Scale in when CPU < 30% for 15 minutes

Infrastructure

Azure Services Used

Compute:

- Azure App Service (Web + API)
- Azure Functions (serverless processing)

Storage:

- Azure SQL Database (primary data)
- Azure Blob Storage (media files, backups)

- Azure Cache for Redis (caching)

Networking:

- Azure Load Balancer
- Azure CDN
- Azure Private Link (database security)

Monitoring:

- Azure Application Insights
- Azure Monitor
- Azure Log Analytics

Security:

- Azure Key Vault (secrets management)
- Azure Active Directory (SSO)
- Azure DDoS Protection

CI/CD Pipeline

Source Control: GitHub (private repository) **CI/CD**: GitHub Actions **Deployment**: Blue-green deployment (zero-downtime)

Pipeline Stages:

1. Build (.NET build)
2. Unit Tests (xUnit, 85% code coverage)
3. Integration Tests (database, API)
4. Security Scan (dependency vulnerabilities)
5. Docker Build
6. Deploy to Staging
7. Smoke Tests
8. Deploy to Production
9. Health Check

Deployment Frequency: 2-4 deploys/week **Rollback Time**: < 5 minutes (automated)

Disaster Recovery

RTO (Recovery Time Objective): 4 hours **RPO (Recovery Point Objective)**: 1 hour (max data loss)

DR Plan:

1. Geo-redundant backups in secondary Azure region
2. Automated failover to secondary region
3. DNS update (5-minute TTL)
4. Regular DR drills (quarterly)

Last DR Drill: December 2025 (successful, 3.5-hour recovery)

Data Model

Key Entities

Contacts

```
CREATE TABLE Contacts (
    Id INT PRIMARY KEY IDENTITY,
    UserId INT NOT NULL,
    PhoneNumber NVARCHAR(20),
    Email NVARCHAR(255),
    FirstName NVARCHAR(100),
    LastName NVARCHAR(100),
    Country NVARCHAR(50),
    City NVARCHAR(100),
    PostalCode NVARCHAR(20),
    CustomAttributes NVARCHAR(MAX), -- JSON
    EngagementScore INT DEFAULT 0,
```

```
    LifecycleStage NVARCHAR(50),
    IsActive BIT DEFAULT 1,
    CreatedAt DATETIME2 DEFAULT GETUTCDATE(),
    UpdatedAt DATETIME2,
    CONSTRAINT FK_Contacts_Users FOREIGN KEY (UserId) REFERENCES Users(Id)
);

-- Indexes
CREATE INDEX IX_Contacts_UserId ON Contacts(UserId);
CREATE INDEX IX_Contacts_Email ON Contacts(Email);
CREATE INDEX IX_Contacts_PhoneNumber ON Contacts(PhoneNumber);
CREATE INDEX IX_Contacts_EngagementScore ON Contacts(EngagementScore);
```

Campaigns

```
CREATE TABLE Campaigns (
    Id INT PRIMARY KEY IDENTITY,
    UserId INT NOT NULL,
    Name NVARCHAR(255) NOT NULL,
    Description NVARCHAR(MAX),
    Type NVARCHAR(50), -- SMS, MMS, Email, Multi
    Status NVARCHAR(50), -- Draft, Scheduled, Running, Completed, Failed
    ScheduledDate DATETIME2,
    SentAt DATETIME2,
    CompletedAt DATETIME2,
    TotalRecipients INT DEFAULT 0,
    TotalSent INT DEFAULT 0,
    TotalDelivered INT DEFAULT 0,
    TotalFailed INT DEFAULT 0,
    TotalOpened INT DEFAULT 0,
    TotalClicked INT DEFAULT 0,
    TotalConversions INT DEFAULT 0,
    EstimatedCost DECIMAL(10,2),
    ActualCost DECIMAL(10,2),
    CreatedAt DATETIME2 DEFAULT GETUTCDATE(),
```

```
    UpdatedAt DATETIME2,
    CONSTRAINT FK_Campaigns_Users FOREIGN KEY (UserId) REFERENCES Users(Id)
);
```

Messages

```
CREATE TABLE Messages (
    Id BIGINT PRIMARY KEY IDENTITY,
    CampaignId INT,
    ContactId INT NOT NULL,
    Channel NVARCHAR(20), -- SMS, MMS, Email
    Direction NVARCHAR(20), -- Outbound, Inbound
    Status NVARCHAR(50), -- Queued, Sent, Delivered, Failed, Opened, Clicked
    Subject NVARCHAR(500),
    Body NVARCHAR(MAX),
    HtmlContent NVARCHAR(MAX),
    MediaUrls NVARCHAR(MAX), -- JSON array
    Provider NVARCHAR(50), -- Twilio, SendGrid
    ProviderMessageId NVARCHAR(255),
    ErrorMessage NVARCHAR(MAX),
    Cost DECIMAL(10,4),
    SentAt DATETIME2,
    DeliveredAt DATETIME2,
    OpenedAt DATETIME2,
    ClickedAt DATETIME2,
    FailedAt DATETIME2,
    CreatedAt DATETIME2 DEFAULT GETUTCDATE(),
    CONSTRAINT FK_Messages_Campaigns FOREIGN KEY (CampaignId) REFERENCES Campaigns(Id),
    CONSTRAINT FK_Messages_Contacts FOREIGN KEY (ContactId) REFERENCES Contacts(Id)
);

-- Partitioning by month for large tables
CREATE PARTITION FUNCTION PF_Messages (DATETIME2)
AS RANGE RIGHT FOR VALUES ('2025-01-01', '2025-02-01', '2025-03-01', ...);
```

Third-Party Integrations

SMS/MMS Provider: Twilio

API: Twilio Programmable Messaging API **Pricing**: $0.0079/SMS (passthrough + 5% markup = $0.0083) **Volume**: 500K messages/month **SLA**: 99.95% uptime **Features**: Two-way messaging, delivery receipts, media messaging (MMS)

Email Provider: SendGrid

API: SendGrid Web API v3 **Pricing**: $0.000095/email (passthrough + 5% = $0.0001) **Volume**: 2M emails/month **SLA**: 99.9% uptime **Features**: Transactional + marketing emails, templates, analytics, inbox testing

Payment Provider: Stripe

API: Stripe API 2023-10-16 **Features**: Subscriptions, invoicing, payment methods, webhooks **PCI DSS**: Stripe handles PCI compliance **Alternative**: PayPal (backup provider)

E-Commerce: Shopify

API: Shopify Admin API 2023-10 **Integration**: OAuth2 app **Data Sync**: Customers, orders, products (bidirectional) **Webhooks**: Order created, customer created, cart abandoned

Development Practices

Code Quality

Code Review: 100% of PRs reviewed by 2+ engineers **Testing**:

- Unit tests: 85% coverage (xUnit)
- Integration tests: 60% coverage

- E2E tests: Critical user flows

Static Analysis: SonarQube, Roslyn analyzers **Code Standards**: Microsoft C# coding conventions, EditorConfig

Git Workflow

Branching Strategy: GitFlow

- `main` - production
- `develop` - integration
- `feature/*` - features
- `hotfix/*` - urgent fixes

Commit Standards: Conventional Commits (semantic versioning)

PR Process:

1. Create feature branch
2. Write tests
3. Open PR with description
4. Code review (2 approvals required)
5. CI passes (build, test, security scan)
6. Merge to develop
7. Deploy to staging
8. QA testing
9. Merge to main
10. Deploy to production

Roadmap: Technical Improvements

Q1 2026

- ✅ Migrate to ASP.NET Core 8.0 (completed)

- ⏳ Implement ElasticSearch for advanced contact search
- ⏳ Add read replicas for analytics queries
- ⏳ GraphQL API (alternative to REST)

Q2 2026

- React-based workflow designer (better UX)
- WebSockets for real-time dashboard updates
- Machine learning: Send time optimization, churn prediction
- Multi-tenancy architecture improvements

Q3 2026

- Mobile apps (iOS, Android) - React Native
- Voice messaging infrastructure (Twilio Voice)
- Advanced queueing with Apache Kafka
- Edge caching with Cloudflare

Q4 2026

- AI content generation (OpenAI GPT-4 integration)
- WhatsApp Business API integration
- Advanced fraud detection
- Multi-region deployment (US-East, US-West, EU, APAC)

Security Incident Response Plan

Incident Types: Data breach, DDoS attack, unauthorized access, system outage

Response Team:

- Incident Commander (CTO)
- Engineering Lead
- Security Lead

- Communications Lead (CEO)

Response Steps:

1. **Detection**: Automated alerts, customer reports
2. **Containment** (< 30 min): Isolate affected systems, block attackers
3. **Investigation** (< 2 hours): Identify root cause, assess damage
4. **Remediation** (< 4 hours): Patch vulnerability, restore service
5. **Communication** (< 6 hours): Notify affected customers, public disclosure if required
6. **Post-Mortem** (< 48 hours): Document incident, prevent recurrence

Contact: security@marketingplatform.com

Performance Benchmarks

API Performance (99th percentile)

Endpoint	Latency
GET /api/contacts	52ms
POST /api/contacts	89ms
GET /api/campaigns/{id}/stats	145ms
POST /api/campaigns	112ms
GET /api/analytics/dashboard	287ms
POST /api/workflows/execute	98ms

Database Performance

Query Type	Execution Time
Simple SELECT (1 table)	3ms
JOIN (2-3 tables)	12ms
Complex analytics query	85ms
Bulk INSERT (1000 rows)	45ms

Load Testing Results

Test: 10,000 concurrent users, 30-minute duration

Metric	Result
Requests/second	8,500
Avg Response Time	95ms
Error Rate	0.02%
CPU Usage	68%
Memory Usage	72%

FAQ - Technical

Q: Can the platform handle 1 million contacts?
A: Yes, tested with 5 million contacts. Performance degrades slightly at 10M+ but still acceptable. Sharding strategy prepared for 50M+.

Q: What happens if Twilio goes down?
A: Automatic failover to backup SMS provider (Plivo). Messages queued and retried. No data loss.

Q: How do you prevent SQL injection?
A: Parameterized queries exclusively. Entity Framework ORM prevents injection. Regular security audits.

Q: Is the codebase well-documented?
A: Yes. XML documentation for all public APIs. README in each project. Architecture decision records (ADRs) for major decisions.

Q: Can we audit the code?
A: Yes, code review available under NDA for serious investors and Enterprise customers.

Document Version: 1.0
Last Updated: January 2026
Contact: engineering@marketingplatform.com

This document is confidential and proprietary. Do not distribute without permission.