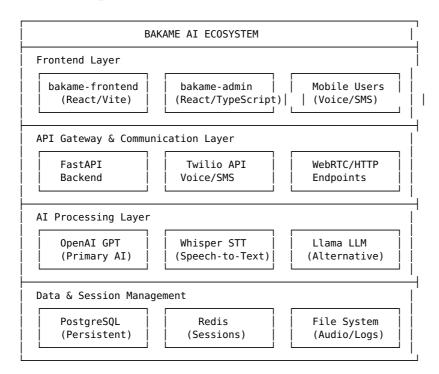
BAKAME AI - Complete System Architecture Documentation

■ System Overview

BAKAME (Building African Knowledge through Accessible Mobile Education) is a comprehensive AI-powered learning platform that delivers education through voice calls and SMS to feature phones, specifically designed for students without internet access.

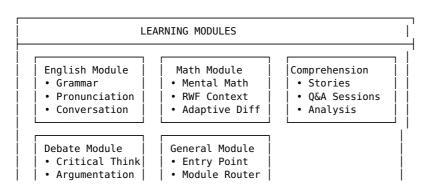
Ⅲ High-Level Architecture

Core Components

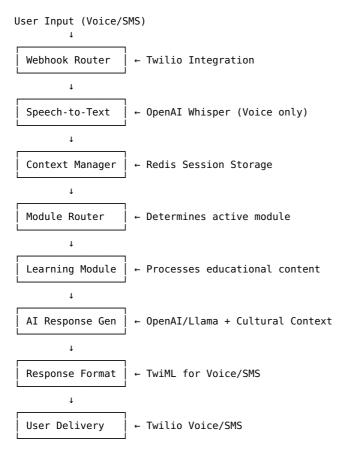


Description Learning Module Architecture

Module System Design



Module Processing Flow



↑ Technical Stack Details

Backend Architecture (FastAPI)

Core Application Structure:

```
bakame-backend/
  - app/
      - main.py
                              # FastAPI application entry point
      - config.py
                             # Environment configuration
       models/
                             # SQLAlchemy models
         database.py
         auth.py
                            # Authentication models
       routers/
         webhooks.py
                            # Twilio webhook handlers
         — admin.py
                            # Admin dashboard APIs
         auth.py
                            # Authentication endpoints
         content.py
                           # Content management
       services/
         twilio_service.py
                                   # Voice/SMS integration
         openai_service.py
                                   # AI text generation
         redis service.py
                                   # Session management
                                  # Analytics & logging
         logging_service.py

    emotional intelligence service.py # Emotion detection

         - gamification_service.py # Points & achievements
```

```
multimodal service.py
                                # Learning style detection
       wellness service.py
                                # Mental health support
      - economic_empowerment_service.py # Financial literacy
      - community_service.py # Peer learning
     — teacher service.py
                                # Educator tools
      - predictive_analytics_service.py # Learning analytics
      — adaptive_learning_service.py
                                       # Personalization
      - offline_service.py # Offline capabilities
      llama service.py
                                # Alternative LLM
      - deepgram service.py # Alternative STT
                               # Current events
      newsapi service.py
    modules/
                                # English learning logic
      - english_module.py
      math module.py
                               # Mathematics learning
      — comprehension module.py # Reading comprehension
      - debate module.py
                               # Critical thinking
                               # Entry point & routing
      general_module.py
                           # Poetry dependencies
pyproject.toml
README . md
```

Frontend Architecture

User-Facing Frontend (bakame-frontend): - **Framework:** React 18 + Vite + TypeScript - **UI Library:** Radix UI components - **Styling:** Tailwind CSS - **State Management:** React Query (TanStack) - **Routing:** React Router DOM - **Charts:** Recharts for analytics

Admin Dashboard (bakame-admin): - Framework: React 18 + Vite + TypeScript - UI Library: Radix UI + shadcn/ui components - Styling: Tailwind CSS - Features: User management, analytics, curriculum alignment

■ Data Architecture

Database Schema (PostgreSOL)

```
-- Core User Management
users (
    id, phone number, user type, name, region,
    school, grade level, is active, created at,
    last_active, total_points, current_level
)
-- Session Tracking
user sessions (
    id, phone_number, session_id, module_name,
    interaction type, user input, ai response,
    timestamp, session duration
)
-- Module Analytics
module usage (
    id, phone number, module name, usage count,
    last used, total duration
-- Community Features
peer connections (
    id, user1_phone, user2_phone, connection_type,
    status, created at, last interaction
)
learning_groups (
    id, name, description, group type, region,
    school, grade_level, subject, teacher_phone,
```

```
is active, created at, max members
group memberships (
    id, group_id, user_phone, role, joined_at, is_active
peer learning sessions (
    id, session_id, group_id, connection_id,
    module name, topic, participants, started at,
    ended at, session summary
-- Authentication
web users (
    id, email, full name, hashed password, role,
    organization, is active, created at
Session Management (Redis)
  "user_context:{phone_number}": {
    "current module": "math",
    "conversation history": [
      {
        "user": "I want to practice math",
        "ai": "Muraho! Let's practice math with RWF...",
        "timestamp": "2024-01-01T12:00:00Z"
      }
    ],
    "user_state": {
      "math level": "medium",
      "math_problems_attempted": 15,
      "math problems correct": 12,
      "current_math_problem": {
        "question": "150 + 75",
        "answer": 225,
        "operation": "+"
      "requested_module": null
    },
    "user name": "Jean",
    "phone_number": "+250781234567",
    "session start": "2024-01-01T11:45:00Z"
  }
```

Communication Flow

Voice Call Processing

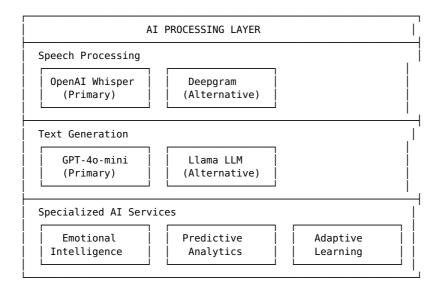
```
    User dials Twilio number
    Twilio webhook → /webhook/call
    Welcome message generation
    Speech collection (Gather)
    Audio → OpenAI Whisper → Text
    Text → Module Processing
    AI Response Generation
    TwiML Voice Response
    Audio playback to user
    Loop for continued interaction
```

SMS Processing

- 1. User sends SMS to Twilio number
- 2. Twilio webhook → /webhook/sms
- 3. Text extraction from message body
- 4. Module Processing
- 5. AI Response Generation
- 6. TwiML SMS Response
- 7. SMS delivery to user

AI Processing Pipeline

Multi-Model AI Architecture

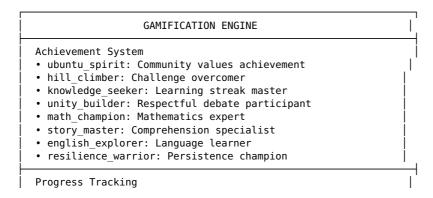


Cultural Context Integration

Rwanda-Specific AI Prompts: - English Module: "You're a friendly, encouraging English conversation partner who understands Rwandan culture deeply..." - Math Module: "You're an enthusiastic math mentor who makes numbers fun using Rwandan contexts. Use examples with Rwandan francs (RWF)..." - Comprehension: "You're an engaging storyteller who loves Rwandan culture and traditions..." - Debate: "You're a thoughtful discussion partner who understands Rwandan society and values deeply..." - General: "You're BAKAME, a warm and intelligent AI learning companion who understands Rwandan culture deeply..."

Advanced Features

Gamification System



- Points system across all modules
- Level progression (beginner → expert → master)
- Module-specific difficulty adaptation
- Cultural context rewards

Emotional Intelligence

EMOTIONAL INTELLIGENCE SYSTEM	
Emotion Detection • frustrated, confident, discouraged, motivated • confused, positive	
Cultural Response Adaptation • Kinyarwanda phrases: "Ntugire ubwoba", "Byiza cyane!" • Ubuntu philosophy integration • Rwanda resilience messaging • Community support emphasis	
Adaptive Response Generation • Emotionally-aware AI responses • Cultural sensitivity • Motivational messaging	

Community & Peer Learning

COMMUNITY FEATURES	-
Regional Learning Groups • Kigali, Northern, Southern, Eastern, Western regions • School-based groups • Grade-level cohorts	
Teacher Integration • Teacher registration and dashboard • Classroom creation and management • Student progress monitoring • Analytics and reporting	
Peer Connections • Study buddy matching • Mentor-mentee relationships • Regional peer networks • Collaborative learning sessions	

△ Security & Authentication

Multi-Layer Security

	SECURITY ARCHITECTURE	1
	Phone-Based Identity • No registration required for learners • Phone number as primary identifier • Session-based context management	
	Web Authentication • JWT-based admin authentication • Role-based access control (admin, super_admin)	

• Organization-based permissions

Data Protection

- Encrypted API communications
- Secure database connections
- Privacy-compliant data handling
- Session data TTL management

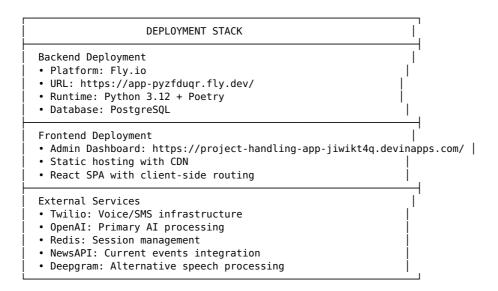
∠ Analytics & Monitoring

Comprehensive Analytics System

ANALYTICS DASHBOARD Usage Statistics • Total sessions, unique users • Module engagement metrics · Geographic usage patterns • Learning outcome tracking Predictive Analytics • Learning pattern analysis • Optimal session length detection • Engagement threshold monitoring • Mastery prediction algorithms Export & Reporting • CSV data export • Session logs for quality improvement • Performance metrics for stakeholders • Curriculum alignment reporting

№ Deployment Architecture

Production Infrastructure



Scalability & Performance

Horizontal Scaling Design

SCALABILITY FEATURES Stateless Architecture Session data in Redis (external) Stateless FastAPI application Horizontal scaling ready Performance Optimizations Sub-3-second response times Async/await throughout Connection pooling Efficient database queries Fault Tolerance Graceful fallbacks for AI services Redis memory fallback Error handling and logging Health check endpoints

1 Configuration Management

Environment Configuration

```
# Key Configuration Parameters
class Settings(BaseSettings):
    # Twilio Integration
   twilio account sid: str
   twilio auth token: str
   twilio_phone_number: str
    # AI Services
   openai api key: str
    llama api key: str
   use_llama: bool = True
   deepgram api key: str
   newsapi_key: str
    # Infrastructure
    redis url: str = "redis://localhost:6379/0"
   database_url: str
    # Application
    app_env: str = "development"
   debug: bool = True
```

Key Architectural Decisions

Design Principles

- 1. Accessibility First: No internet or smartphone required
- 2. Cultural Integration: Deep Rwanda context throughout
- 3. Scalable Design: Cloud-native, stateless architecture
- 4. Multi-Modal AI: Voice and text processing capabilities
- 5. Educational Focus: Curriculum-aligned learning modules
- 6. Community Building: Peer learning and teacher integration
- 7. **Data-Driven:** Comprehensive analytics and adaptation
- 8. Fault Tolerant: Graceful degradation and fallbacks

Technology Choices

• FastAPI: High-performance async Python framework

• React: Modern, component-based frontend development

• **PostgreSQL:** Robust relational database for complex queries

• Redis: High-performance session and cache management

• Twilio: Reliable telecommunications infrastructure

• OpenAI: State-of-the-art AI language processing

• Fly.io: Modern cloud deployment platform

Document Version: 1.0

Last Updated: September 6, 2025

Architecture Status: Production Ready MVP

Next Phase: Community features and teacher integration scaling