AMC AI Chatbot - Technical Documentation

Table of Contents

1. Introduction

1.1 Project Overview

The AMC AI Chatbot is a modern web application designed to provide users with access to Amhara Media Corporation's news and institutional information through a conversational interface. The system supports both Amharic and English languages and includes voice input capabilities.

1.2 Technical Goals

- Provide real-time news access
- Support bilingual interactions
- Ensure mobile responsiveness
- Implement voice recognition
- Maintain high performance
- Ensure scalability

1.3 Target Platforms

- Web browsers (Chrome, Firefox, Safari, Edge)
- Mobile devices (iOS, Android)
- Desktop computers (Windows, macOS, Linux)

2. System Architecture

2.1 Frontend Architecture

```
frontend/
├── src/
│   ├── components/
│   │   ├── ChatInterface.jsx (Main chat component)
│   │   ├── Layout.jsx (Page layout component)
│   │   └── VoiceInput.jsx (Voice input handler)
│   ├── config.js (Configuration settings)
│   ├── App.jsx (Root component)
│   └── styles/ (CSS and styling)
```

2.2 Backend Architecture

```
backend/
├── routes/ (API endpoints)
├── services/ (Business logic)
├── models/ (Data models)
└── server.js (Main server file)
```

2.3 Data Flow

1. User Input → Frontend
2. Frontend → API Request
3. Backend → Data Processing
4. Backend → Response Generation
5. Frontend → Display Response

3. Technical Implementation

3.1 Frontend Technologies

- **React.js**: UI framework
- **TailwindCSS**: Styling
- **Web Speech API**: Voice recognition
- **Fetch API**: HTTP requests

3.2 Key Frontend Features

- Real-time chat interface
- Voice input processing
- Language switching
- Responsive design
- Error handling
- Connection management

3.3 Component Implementation Details

ChatInterface.jsx

```
// Key features:
- Message history management
- API communication
- Error handling
- Bilingual support
- Real-time updates
```

VoiceInput.jsx

```
// Key features:
- Speech recognition
- Language detection
- Error handling
- Visual feedback
```

Layout.jsx

```
// Key features:
- Responsive design
- AMC branding
- Mobile optimization
```

4. Component Documentation

4.1 ChatInterface Component

Props

- `language` : String - Current language setting
- `onLanguageChange` : Function - Language change handler

State

- `messages` : Array - Chat history
- `input` : String - Current input
- `loading` : Boolean - Loading state
- `error` : String/null - Error state
- `isConnected` : Boolean - Connection state

Methods

- `handleSubmit()` : Message submission
- `formatNewsResults()` : News formatting

- `checkServerConnection()` : Connection check
- `scrollToBottom()` : Chat scroll handler

4.2 VoiceInput Component

Props

- `onResult` : Function - Voice result handler
- `language` : String - Current language
- `disabled` : Boolean - Disabled state

Methods

- `startListening()` : Voice recognition
- `handleError()` : Error handling

5. API Documentation

5.1 Endpoints

POST /api/ask

```
Request:
{
  "message": "string",
  "language": "string"
}

Response:
{
  "status": "string",
  "message": "string",
  "context": [
    {
      "title": "string",
      "url": "string",
      "language": "string",
      "date": "string",
      "category": "string"
    }
  ],
  "total_results": "number",
  "is_institutional": "boolean"
}
```

GET /api/health

```
Response:
{
  "status": "string",
  "message": "string"
}
```

6. Security Considerations

6.1 Frontend Security

- Input sanitization
- XSS prevention
- CORS configuration
- Secure API communication

6.2 API Security

- Rate limiting
- Request validation
- Error handling
- Secure headers

7. Performance Optimization

7.1 Frontend Optimization

- Code splitting
- Lazy loading
- Image optimization
- Caching strategies

7.2 Network Optimization

- Request batching
- Response compression
- Connection pooling
- Error recovery

8. Testing Strategy

8.1 Frontend Testing

- Unit tests for components
- Integration tests
- End-to-end testing
- Browser compatibility

- Mobile responsiveness

8.2 API Testing

- Endpoint testing
- Load testing
- Security testing
- Error handling

9. Deployment Guide

9.1 Prerequisites

- Node.js environment
- npm/yarn
- Web server
- SSL certificate

9.2 Deployment Steps

1. Build frontend
2. Configure environment
3. Deploy backend
4. Set up monitoring
5. Configure domain

9.3 Environment Variables

```
 Frontend:
REACT_APP_API_URL=http://localhost:5000

Backend:
PORT=5000
NODE_ENV=development
```

10. Maintenance and Updates

10.1 Regular Maintenance

- Dependency updates
- Security patches
- Performance monitoring
- Error logging

10.2 Update Procedure

1. Test in development

2. Stage changes
3. Deploy updates
4. Monitor performance

10.3 Monitoring

- Error tracking
- Performance metrics
- User analytics
- Server health

Appendix

A. Troubleshooting Guide

1. Connection Issues
2. Voice Input Problems
3. Language Switch Errors
4. Mobile Display Issues

B. Code Style Guide

- ESLint configuration
- Prettier settings
- Naming conventions
- Documentation standards

C. Version History

- v1.0.0: Initial release
- v1.1.0: Mobile optimization
- v1.2.0: Voice input improvements
- v1.3.0: Error handling enhancement