Project Proposal: NUML InsightBot - One-Month Implementation Plan

Project Overview

NUML InsightBot is an AI-powered chatbot designed to provide information about the National University of Modern Languages (NUML) by leveraging scraped data from the university's website. The application will feature user authentication, protected routes, and a responsive chatbot interface with streaming responses.

Accelerated Implementation Strategy (30 Days)

Week 1: Foundation & Setup

Days 1-3: Project Initialization

- Set up React frontend with React Router DOM
- Initialize FastAPI backend structure
- Configure PostgreSQL database with basic schemas
- Implement environment configuration with .env files

Days 4-7: Authentication System

- Implement user registration and login functionality
- Create protected route middleware
- Set up JWT token management
- Design database schemas for users and conversations

Week 2: Core Functionality

Days 8-10: Web Scraping Module

- Implement Firecrawl integration for NUML website scraping
- Design data processing and cleaning pipeline
- Set up initial scraping job

Days 11-14: AI Integration & Vector Database

- Implement Ollama/Groq API connections
- Set up Gemini embeddings and Pinecone vector store
- Create basic query processing system

Week 3: Chat Interface & Additional Pages

Days 15-18: Chatbot Interface

- Build React chat components with streaming responses
- Implement conversation memory and history
- Create support page with authentication requirement

Days 19-21: Additional Pages

- Develop Home, About, Contact, and Services pages
- Implement responsive design across all pages

Week 4: Polish & Deployment

Days 22-24: Testing & Refinement

- Comprehensive testing of all functionality
- Performance optimization
- Security audit and enhancements

Days 25-28: Deployment Preparation

- Containerization with Docker
- Production environment setup
- Final documentation

Days 29-30: Deployment & Launch

- Deploy to production environment
- Monitor initial performance
- Gather user feedback

Technical Architecture Details

Database Schemas

User Schema:

```sql
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_login TIMESTAMP
);
```

Conversation Schema:

```sql
CREATE TABLE conversations (
    id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(id),
    title VARCHAR(255),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE messages (
    id SERIAL PRIMARY KEY,
    conversation_id INTEGER REFERENCES conversations(id),
    content TEXT,
    is_user BOOLEAN,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Scraping Log Schema:

```sql
CREATE TABLE scraping_logs (
    id SERIAL PRIMARY KEY,
    status VARCHAR(20),
    pages_scraped INTEGER,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Page Content Specifications

Home Page:

- Hero section with chatbot introduction
- Key features overview
- Call-to-action for authentication

Login/Signup Pages:

- Clean forms with validation
- Password recovery option
- Social authentication options (if time permits)

About Page:

- Project mission and vision
- Team information
- Technology stack overview

Contact Page:

- Contact form with validation
- University contact information
- FAQ section

Services Page:

- Detailed feature descriptions
- Usage guidelines
- Support information

Support Page (Protected):

- Chat interface with message history
- Real-time response streaming
- Conversation management options

API Endpoints

Authentication Endpoints:

- POST /api/auth/register - User registration
- POST /api/auth/login - User authentication
- POST /api/auth/refresh - Token refresh
- POST /api/auth/logout - User logout

Chat Endpoints:

- POST /api/chat/conversations - Create new conversation
- GET /api/chat/conversations - Get user conversations
- POST /api/chat/message - Send message to chatbot
- WS /api/chat/stream - WebSocket for streaming responses

Admin Endpoints:

- POST /api/admin/scrape - Initiate manual scraping
- GET /api/admin/scraping-logs - View scraping history
- GET /api/admin/usage-stats - Get usage statistics

File Structure

```
numl-insightbot/
  frontend/
    public/
    src/
      components/
        Chat/
        Auth/
        Common/
      pages/
        Login.js
        Signup.js
        Home.js
        About.js
        Contact.js
        Services.js
        Support.js
      contexts/
      hooks/
      utils/
      styles/
    package.json
  backend/
    app/
      models/
      routes/
        auth.py
        chat.py
        admin.py
      services/
        scraping.py
        embeddings.py
        llm.py
      core/
        config.py
        database.py
        security.py
        main.py
    requirements.txt
    Dockerfile
database/
    migrations/
.env
```

Risk Management

Potential Challenges:

1. **Website Structure Changes**: NUML might update their website during development
   - Mitigation: Implement adaptive scraping with regular structure checks
2. **API Rate Limiting**: LLM providers may have usage limits
   - Mitigation: Implement caching and fallback mechanisms
3. **Complex Authentication**: JWT implementation can be challenging
   - Mitigation: Use proven libraries and thorough testing

Contingency Plans:

- Ready-to-use UI component libraries for faster frontend development
- Simplified initial scraping targets with expansion planned post-MVP
- Basic streaming implementation with option to enhance later

Additional Features (If Time Permits)

Priority 1:

- File upload support for document queries
- Conversation export functionality
- Advanced admin dashboard

Priority 2:

- Multi-language support
- Voice input/output capabilities
- Integration with university calendar

Success Metrics

- User registration and retention rates
- Average response time under 2 seconds
- Scraping completion within 1 hour for initial run
- System uptime of 99.5% or higher

Conclusion

The one-month timeline for NUML InsightBot is ambitious but achievable with focused effort and proper prioritization. The plan emphasizes core functionality first with enhancements added as time permits. The application will deliver significant value to NUML students and staff by providing instant access to university information through an intuitive chat interface.