# Lex Assist Backend Implementation Plan

## Overview

This document outlines the comprehensive implementation plan for the Lex Assist backend, focusing on role-based access control, Supabase integration, and AI components including InLegalBERT integration.

## 1. Role-Based Access Structure

### 1.1 User Roles

* **Super Admin**
  + Full access to ALL features and system settings
  + Ability to configure subscription tier rates
  + User management across all roles
  + System configuration and monitoring
* **Admin**
  + Access to ALL app features across all subscription tiers
  + No access to system settings or configuration
  + Limited user management capabilities
  + Analytics and reporting access
* **Regular Users**
  + Access based on subscription tier (Free, Pro, Enterprise)
  + Feature limitations based on tier
  + Personal profile and case management

### 1.2 Access Control Implementation

* JWT-based authentication with role claims
* Server-side validation of role permissions
* Frontend conditional rendering based on user role
* API endpoint protection with middleware
* Database-level access controls

## 2. Supabase Integration

### 2.1 Authentication

* Email/password authentication
* Mobile OTP verification
* JWT token management
* Session handling and refresh tokens
* Password reset and account recovery

### 2.2 Database Schema

* **Users Table**
  + User ID (primary key)
  + Email, name, mobile number
  + Role (super\_admin, admin, user)
  + Account status
  + Created/updated timestamps
* **Subscriptions Table**
  + Subscription ID (primary key)
  + User ID (foreign key)
  + Tier (free, pro, enterprise)
  + Payment status
  + Start/end dates
  + Billing information
* **Case Briefs Table**
  + Brief ID (primary key)
  + User ID (foreign key)
  + Brief content
  + Created/updated timestamps
  + Status
* **Analysis Results Table**
  + Result ID (primary key)
  + Brief ID (foreign key)
  + Law sections (JSON)
  + Case histories (JSON)
  + Analysis content
  + Created timestamp
* **Subscription Tiers Table**
  + Tier ID (primary key)
  + Name (free, pro, enterprise)
  + Price
  + Features (JSON)
  + Active status

### 2.3 Supabase Row-Level Security

* Implement RLS policies for each table
* User-specific data access controls
* Role-based read/write permissions

## 3. Backend API Structure

### 3.1 Authentication Endpoints

* /api/auth/register - User registration
* /api/auth/login - Email/password login
* /api/auth/otp/request - Request OTP
* /api/auth/otp/verify - Verify OTP
* /api/auth/refresh - Refresh authentication token

### 3.2 User Management Endpoints

* /api/users - User CRUD operations
* /api/users/profile - User profile management
* /api/users/role - Role management (Super Admin only)

### 3.3 Subscription Endpoints

* /api/subscriptions - Subscription management
* /api/subscriptions/tiers - Tier configuration (Super Admin only)
* /api/subscriptions/payment - Payment processing

### 3.4 Legal Analysis Endpoints

* /api/briefs - Case brief CRUD operations
* /api/briefs/analyze - Analyze case brief
* /api/briefs/draft - Draft case file
* /api/briefs/export - Export analysis results

## 4. AI Integration Architecture

### 4.1 Model Context Protocol

* Standardized input/output format for all AI models
* Context management for legal domain knowledge
* Prompt engineering templates for different legal tasks
* Version control for models and prompts

### 4.2 InLegalBERT Integration

* API wrapper for InLegalBERT model
* Fine-tuning pipeline for Indian legal context
* Embedding generation for semantic search
* Model performance monitoring

### 4.3 AI Agent Implementation

* Orchestration layer between user and models
* Task routing based on user request
* Multi-step reasoning for complex legal analysis
* Result formatting and explanation generation

### 4.4 Legal Database Integration

* API connections to Indian legal databases
* Caching mechanism for frequently accessed data
* Data preprocessing for model consumption
* Citation formatting and verification

## 5. Implementation Phases

### Phase 1: Core Infrastructure (Weeks 1-2)

* Set up Supabase project
* Implement authentication system
* Create database schema with RLS
* Develop basic API endpoints

### Phase 2: Role-Based Access Control (Weeks 3-4)

* Implement user roles and permissions
* Develop admin and super admin interfaces
* Create subscription tier management
* Set up access control middleware

### Phase 3: AI Integration (Weeks 5-7)

* Integrate InLegalBERT model
* Develop Model Context Protocol
* Implement AI Agent architecture
* Connect to legal databases

### Phase 4: Frontend Integration (Weeks 8-9)

* Connect frontend to backend APIs
* Implement conditional UI based on roles
* Develop subscription management UI
* Create analysis visualization components

### Phase 5: Testing and Optimization (Weeks 10-12)

* Comprehensive testing of all components
* Performance optimization
* Security auditing
* Documentation finalization

## 6. Technical Stack

* **Backend Framework**: Flask/FastAPI (Python)
* **Database**: PostgreSQL (via Supabase)
* **Authentication**: Supabase Auth + JWT
* **AI Models**: InLegalBERT, custom fine-tuned models
* **API Documentation**: Swagger/OpenAPI
* **Deployment**: Netlify (frontend), Supabase Functions (backend)
* **Monitoring**: Sentry, Supabase Realtime

## 7. Security Considerations

* Data encryption at rest and in transit
* Regular security audits
* Input validation and sanitization
* Rate limiting and DDoS protection
* Compliance with Indian data protection regulations

## 8. Scalability Planning

* Horizontal scaling for API servers
* Database read replicas for high traffic
* Caching strategy for frequent queries
* Batch processing for heavy AI workloads
* CDN integration for static assets

## Next Steps

1. Set up Supabase project and configure authentication
2. Create initial database schema with role-based security
3. Develop core API endpoints for user management
4. Implement role-based access control system
5. Begin integration with InLegalBERT model