

SkillNova AI: System Connection & Run Guide

This guide is designed to help you **quickly connect** the Frontend, Backend, and Database, and **run the code again** without confusion.

■■ Step 1: Quick Connection Checklist

Before running, verify these **4 Key Files**. If these match, your system will connect successfully.

1■■ Database Connection (Backend)

- **File:** backend/src/main/resources/application.properties
- **What to Check:** Ensure your MySQL password is correct.
- **Code Segment:**

```
properties spring.datasource.url=jdbc:mysql://localhost:3306/placement_db
spring.datasource.username=root spring.datasource.password=YOUR_PASSWORD_HERE <-- CHECK THIS
```

2■■ Allow Frontend Access (Backend)

- **File:** backend/src/main/java/com/aipsms/backend/security/WebSecurityConfig.java
- **What to Check:** Ensure "CORS" allows the Frontend URL.
- **Code Segment** (Look for corsConfigurationSource):

```
java
configuration.setAllowedOrigins(java.util.Arrays.asList( "http://localhost:5173", // <-- React
Default Port "http://localhost:3000" // <-- Alternate Port (just in case) ));
```

3■■ Point Frontend to Backend (Frontend)

- **File:** frontend/src/services/JobService.js
- **What to Check:** Ensure the API URL matches the Backend Port (8080).
- **Code Segment:**

```
javascript const API_URL = "http://localhost:8080/api/jobs"; // <-- Must match
Backend Port
```

4■■ Activate AI Features (Frontend)

- **File:** frontend/.env
 - **What to Check:** Ensure your Google Gemini API key is present.
 - **Code Segment:**

```
env VITE_GEMINI_API_KEY=AIzaSy... <-- Must be a valid key
```
-

■ Step 2: How to Run the Code

Open **Two Terminal Windows** (one for Backend, one for Frontend).

Terminal 1: Start Backend (Spring Boot)

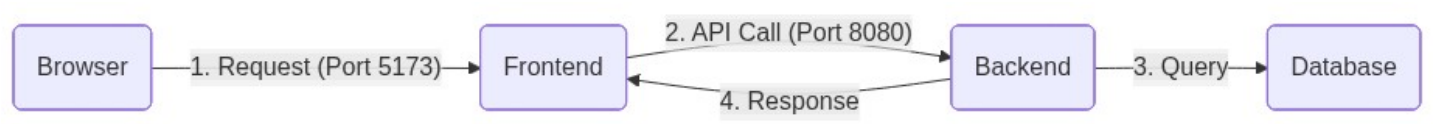
1. Navigate to the folder: `bash cd "d:\AI-Powered Student Placement System\backend"`
2. Run the server: `bash mvn spring-boot:run` *Wait until you see: Started BackendApplication in X seconds*

Terminal 2: Start Frontend (React)

1. Navigate to the folder: `bash cd "d:\AI-Powered Student Placement System\frontend"`
 2. Run the development server: `bash npm run dev`
 3. **Open in Browser:** Click the link shown (usually `http://localhost:5173`).
-

■ Visual Data Flow

This diagram shows how data moves when you click a button.



- **If Step 2 Fails:** Detailed in **Checklist Item #2** (CORS).
- **If Step 3 Fails:** Detailed in **Checklist Item #1** (Database Password).

SkillNova AI: Official Technical Documentation & Architecture Reference

Version: 1.0.0 **Date:** January 2026 **System:** AI-Powered Student Placement Management System

■ Table of Contents

- 1. [Executive Summary](#)
 - [System Architecture Blueprint](#)
 - High-Level Overview
 - Micro-Service Interaction Flow
- 3. [Technology Stack Deep Dive](#)
 - [Backend Implementation \(Spring Boot\)](#)
 - Security & Authentication Layer
 - API Controller Structure
 - Database Connectivity & ORM
 - [Frontend Implementation \(React + Vite\)](#)
 - Component Architecture
 - Service Layer Integration
 - State Management
 - [The AI Engine \(Google Gemini Integration\)](#)
 - Resume Parsing Logic
 - Career Roadmap Generation
 - Prompt Engineering & Safety Fallbacks
- 7. [Database Schema & Data Dictionary](#)
- 8. [Developer's Handbook: Configuration & Setup](#)
- 9. [Future Maintenance & Troubleshooting](#)

1. Executive Summary

SkillNova AI is a next-generation placement management platform designed to bridge the gap between academic learning and industry requirements. Unlike traditional placement portals that merely list jobs, SkillNova utilizes **Generative AI (Google Gemini)** to:

- * Analyze student resumes against real-time market demands.
- * Generate personalized, week-by-week learning roadmaps.
- * Predict future skill trends and employability scores.

This document serves as the **definitive technical manual**, detailing every architectural decision, code implementation, and integration point to ensure future maintainability and scalability.

2. System Architecture Blueprint

The system follows a **Service-Oriented Architecture (SOA)** with a decoupled Frontend and Backend, allowing for independent scaling and development.

2.1 High-Level Architecture Diagram

2.2 Data Flow Narrative

- User Interaction:** The user interacts with the React Frontend.
 - API Requests:** For transactional data (Jobs, Applications, Login), the Frontend calls the Spring Boot Backend via REST APIs.
 - AI Processing:** For Intelligence tasks (Roadmaps, Resume Analysis), the Frontend communicates **directly** with Google Gemini to ensure low latency and reduce server load.
 - Persistence:** The Backend handles all data storage ensures data integrity via ACID-compliant MySQL transactions.
-

3. Technology Stack Deep Dive

Component	Technology	Rationale
Frontend	React.js (Vite)	High performance, component-based reusability, fast build times.
Styling	Tailwind CSS	Utility-first CSS for rapid, modern UI development with "Glassmorphism" aesthetics.
Backend	Java Spring Boot	Robust, enterprise-grade security (Spring Security), and ease of REST API creation.
Database	MySQL	Reliable relational data storage for complex relationships (Users <-> Jobs <-> Applications).
AI Model	Google Gemini 1.5	Multimodal capabilities, large context window for analyzing detailed resumes, and free tier availability.
Security	JWT (JSON Web Tokens)	Stateless authentication perfect for Single Page Applications (SPAs).

4. Backend Implementation

Location: `d:\AI-Powered Student Placement System\backend\`

The backend is the "Source of Truth" for the application. It enforces business rules and security.

4.1 Security Configuration (Critical)

- **File:** `src/main/java/com/aipsms/backend/security/WebSecurityConfig.java`
- **Purpose:** Controls who can access what. It manages CORS (Cross-Origin Resource Sharing) and Password Hashing.

Key Definition (Where to Edit to allow Frontend Access):

```
@Bean public CorsConfigurationSource corsConfigurationSource() { CorsConfiguration configuration
= new CorsConfiguration(); // CRITICAL: Must match your Frontend URL
configuration.setAllowedOrigins(Arrays.asList("http://localhost:5173"));
configuration.setAllowedMethods(Arrays.asList("GET", "POST", "PUT", "DELETE")); // ... }
```

4.2 Database Connectivity

- **File:** `src/main/resources/application.properties`
- **Purpose:** Contains the "Keys to the Castle" (Database Credentials).

Configuration Segment:

```
spring.datasource.url=jdbc:mysql://localhost:3306/placement_db?useSSL=false
spring.datasource.username=root spring.datasource.password=YOUR_PASSWORD <-- Update this if DB
password changes spring.jpa.hibernate.ddl-auto=update <-- Automatically creates tables
```

5. Frontend Implementation

Location: `d:\AI-Powered Student Placement System\frontend\`

The frontend is built with distinct layers for UI components and Logic/Services.

5.1 Service Layer Pattern

Instead of hardcoding API calls in buttons, we use a **Service Layer**. This makes the code clean and reusable.

- **File:** `src/services/JobService.js`
- **Purpose:** Central hub for all Backend API calls.

Code Insight:

```
import axios from 'axios'; // The Single Point of Configuration for Backend URL
const API_URL = "http://localhost:8080/api/jobs";
const fetchJobs = async () => { // Automatically attaches the JWT Token for security
  const token = localStorage.getItem('userToken');
  return axios.get(API_URL, {
    headers: { Authorization: `Bearer ${token}` }
  });
};
```

5.2 Responsive & Modern UI

We utilize **Tailwind CSS** for the "Dark Mode" aesthetic. * **Global Styles:** `src/index.css` (Contains `@keyframes` for the marquee and custom animations). * **Layouts:** `src/components/DashboardLayout.jsx` handles the Sidebar, Header, and Navigation state.

6. The AI Engine (SkillNova Brain)

This is the core differentiator of the platform.

Location: `src/services/ResumeService.js`

6.1 Architecting the AI Bridge

We use the Google Generative AI SDK to communicate with the Gemini output.

Configuration (Environment Variable): * **File:** `.env` * **Value:** `VITE_GEMINI_API_KEY=AIzaSy...`

6.2 The "Prompt Engineering" Logic

The quality of the roadmap depends on how we ask the AI.

Prompt Structure (Inside `generateRoadmap` method):

```
const prompt = ` Act as a Senior Technical Mentor. Create a detailed 4-Week Intensive Learning Roadmap for a student who wants to become a: "${goal}". Return STRICT JSON (no markdown) with this structure: { "weeks": [ { "week": 1, "topic": "...", "resources": [...] } ] } `;
```

6.3 Safety & Fallback Mechanism

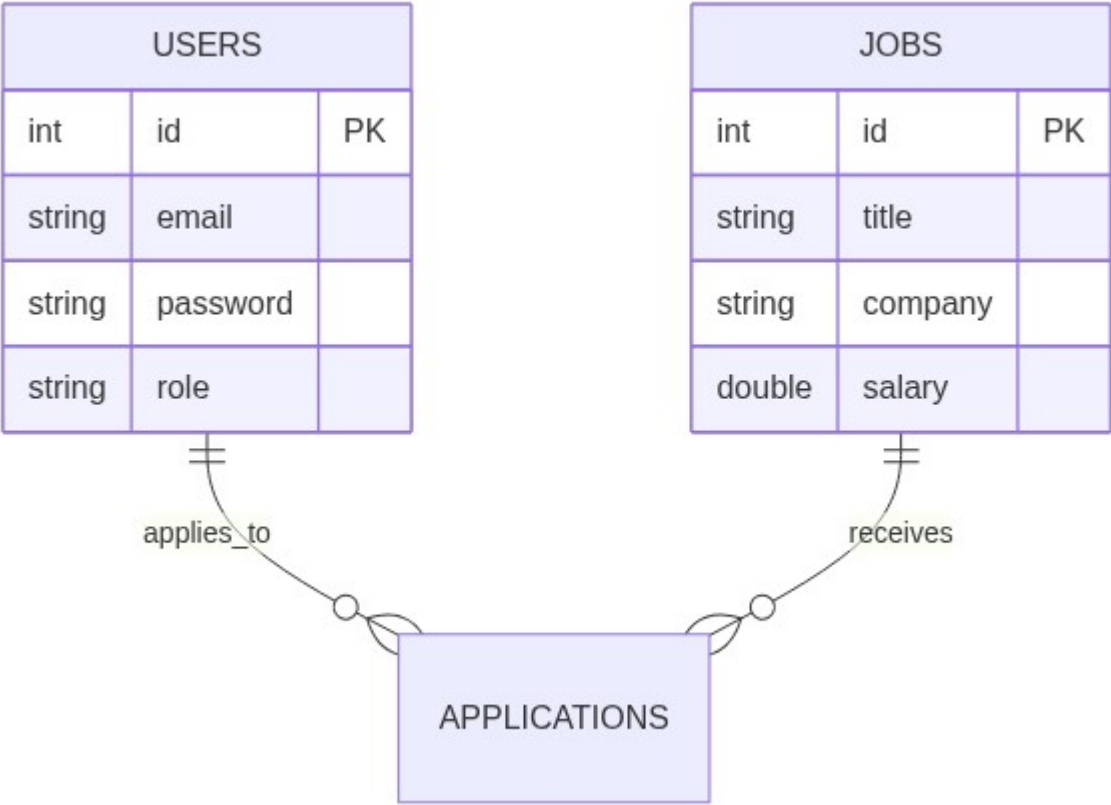
New Feature: If the Internet cuts out or the API Key expires, the system **Must Not Crash**. * **Logic:** `try { ...call API... } catch (error) { return getMockRoadmap(); }` * **Benefit:** Users always see a roadmap (Demo Mode), preserving the User Experience.

7. Database Schema & Data Dictionary

The MySQL database `placement_db` consists of these core tables:

- 1. `users`: Stores login info, role (Student/HR), and profile data.
- 2. `jobs`: Stores job postings (Title, Company, Salary).
- 3. `applications`: Linking table (User ID <-> Job ID) tracking status (Applied/Shortlisted).

ER Diagram (Entity-Relationship):



8. Developer's Handbook

How to Restart the System (Cold Boot)

If you return to this project after months, follow this exact sequence:

1. Start the Database * Ensure XAMPP/MySQL Service is Running. * Verify `placement_db` exists.

2. Revive the Backend

```
cd backend mvn clean install # Re-downloads dependencies mvn spring-boot:run
```

3. Revive the Frontend

```
cd frontend npm install # Re-downloads node_modules npm run dev
```

Common "Future You" Problems

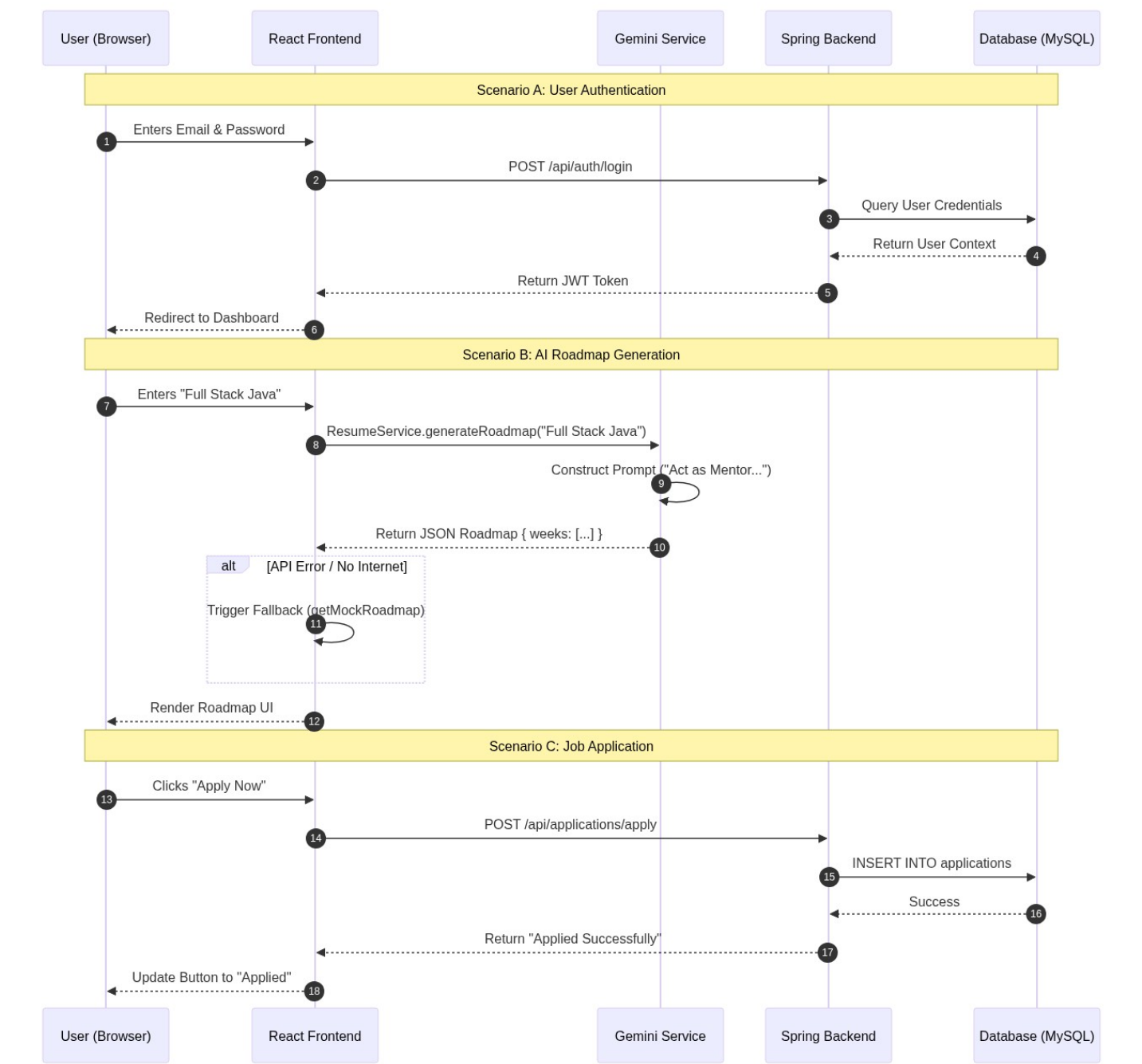
- **"API Keys Invalid"**: Google API keys often expire. Go to Google AI Studio, generate a new one, and update `frontend/.env`.
 - **"CORS Error"**: If you deploy this to a real website (e.g., Vercel), you MUST update `webSecurityConfig.java` to allow the new domain name instead of `localhost`.
-

9. Future Maintenance

- **Rebranding:** All branding strings are centralized in `Landing.jsx` and `DashboardLayout.jsx`. Search for "SkillNova AI".
 - **Updating Mock Data:** Edit `getMockRoadmap()` in `ResumeService.js` to add more modern technologies to the demo mode.
-

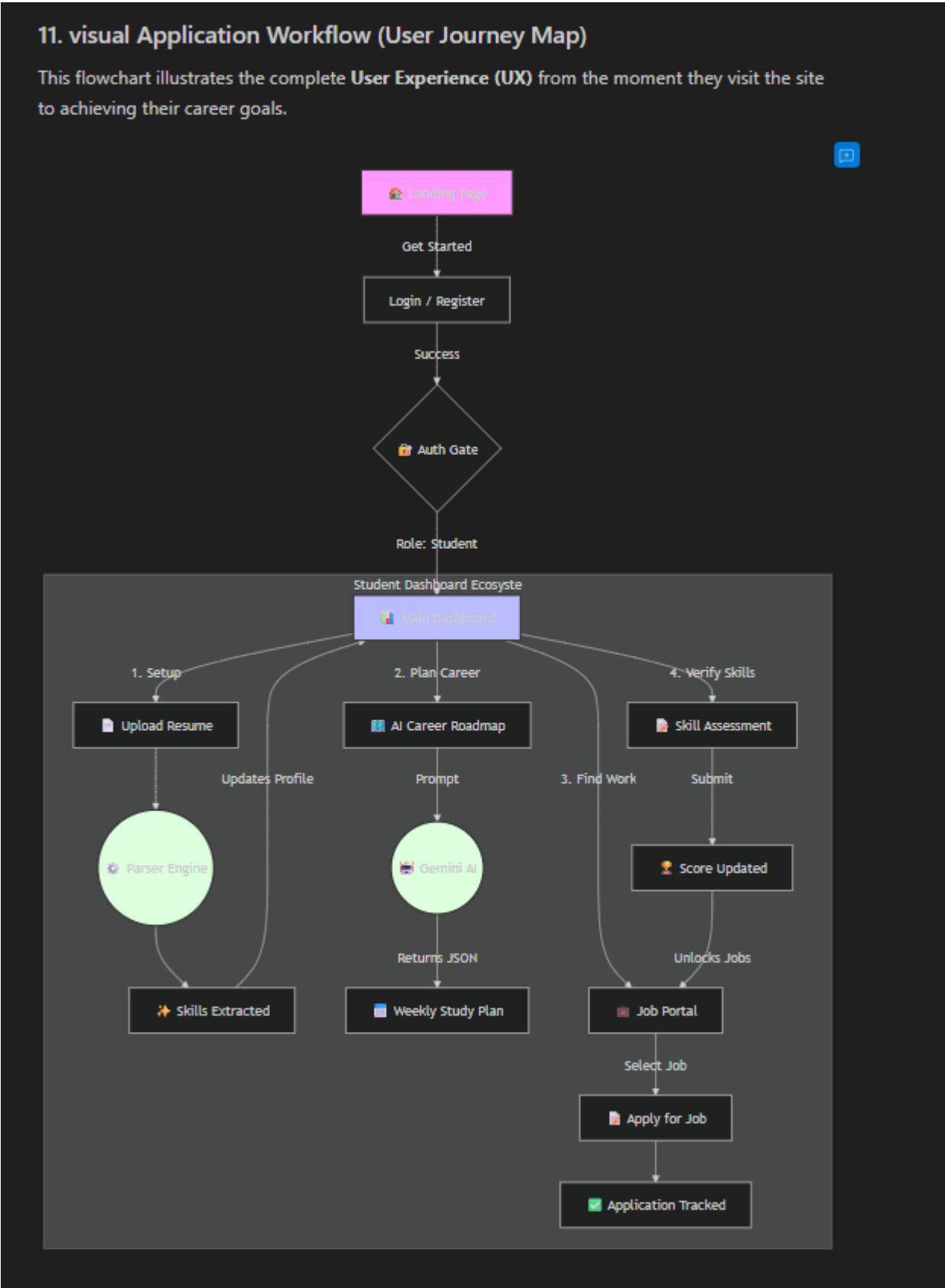
10. Visual Project Workflow (Function Interconnections)

This diagram visualizes exactly how data moves through the system when a user performs key actions like **Logging In**, **Generating a Roadmap**, and **Applying for Jobs**.



11. visual Application Workflow (User Journey Map)

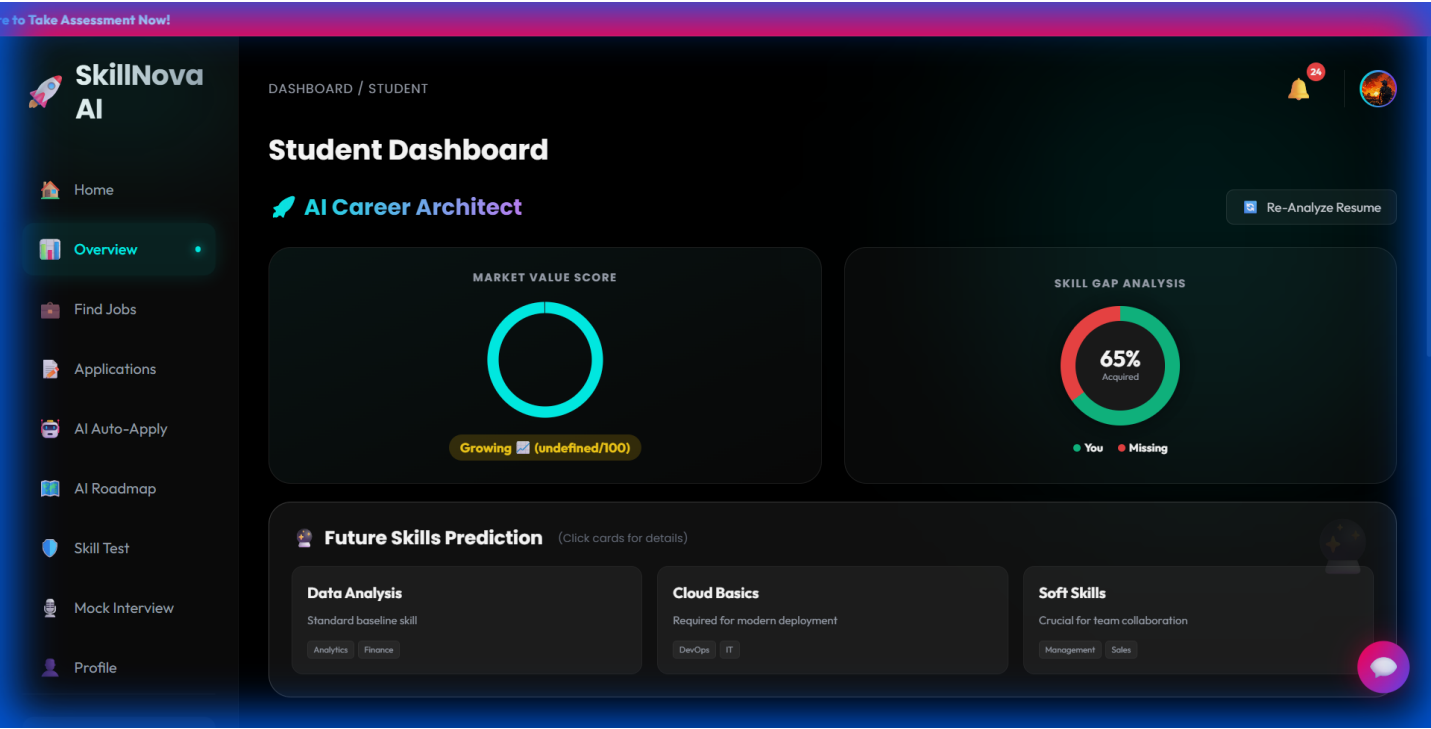
This flowchart illustrates the complete **User Experience (UX)** from the moment they visit the site to achieving their career goals.



12. Visual Interface Reference

Student Dashboard (Live)

The following screenshot demonstrates the active Student Dashboard with AI Career Architect and Job Market integration.



SkillNova AI - Confidential Technical Documentation Generated for Future Reference