

**Project Title:** AutoApply App - REST API + Mobile App Development Plan

**Timeline:** 7 Days (1 Week)

**Objective:** Build a full-featured AI-powered job application system consisting of: 1. A RESTful backend API using **FastAPI** 2. A cross-platform **mobile app** using **Flutter** (Android & iOS) 3. Deployment, testing, and publishing to app stores

---

## Day 1: System Design & Backend Setup

- **Goal:** Lay the foundation of the system
  - **Tasks:**
    - Finalize feature list for both jobseekers and recruiters
    - Design REST API endpoints (OpenAPI schema)
    - Set up FastAPI project structure
    - Create GitHub repo with backend code (if separate)
    - Setup PostgreSQL (or SQLite for dev) for job & user data
    - Setup Snowflake (as data source for matched jobs)
    - Create `models.py`, `schemas.py`, `database.py`
- 

## Day 2: Build Core API Endpoints

- **Goal:** Have working endpoints to serve jobs & accept user input
  - **Tasks:**
    - Endpoint: `/jobs/matched` - Get matched jobs (Snowflake integration)
    - Endpoint: `/user/register`, `/user/login` (JWT Auth)
    - Endpoint: `/user/profile` - Upload CV, get profile
    - Endpoint: `/apply` - Auto-apply for matched jobs
    - Test endpoints with Swagger UI & Postman
- 

## Day 3: Job Scraper & AI Matching Engine

- **Goal:** Integrate data scraping and semantic CV/job matching
  - **Tasks:**
    - Set up job scraping (from Indeed, PNet, etc.) using BeautifulSoup or Selenium
    - Build AI-based matcher using `spaCy` or `sentence-transformers`
    - Save results into Snowflake / DB
    - Create script `scraper.py` to run daily
    - Test AI matching with sample CVs & job listings
-

## Day 4: Flutter Mobile App - Setup & Auth

- **Goal:** Begin mobile app build (Android & iOS)
  - **Tasks:**
    - Set up Flutter environment and folder structure
    - Create screens: Splash, Login, Signup, Dashboard
    - Connect to FastAPI endpoints for auth
    - Store JWT securely
    - Deploy auth backend to Render/Vercel/Fly.io
- 

## Day 5: Flutter App - Job Matching UI

- **Goal:** Display matched jobs and handle auto-apply
  - **Tasks:**
    - Design job cards UI
    - Connect to `/jobs/matched` API
    - Button for "Auto Apply"
    - Display response/confirmation from server
    - Add loader + error handling
- 


## Day 6: Admin Panel + Testing




- **Goal:** Add testing, admin, and polish
  - **Tasks:**
    - Add FastAPI Admin (for viewing applications, logs)
    - Write unit + integration tests
    - Setup logging and exception handling
    - Write instructions for adding new jobs manually
    - Test API security & performance (e.g. via Locust)
- 

## Day 7: Deployment & Publishing

- **Goal:** Launch!
  - **Tasks:**
    - Deploy backend to production (Render/Fly.io/DigitalOcean)
    - Set up mobile builds for Android (Play Console) and iOS (App Store Connect)
    - Create store listings: description, banner, icons
    - Submit for review
    - Monitor logs and fix final bugs
- 

## Optional (Post Week 1): Premium Features

-  Voice-to-CV builder

-  Subscription-based plans
-  AI feedback on CVs
-  Recruiter dashboard for CV filtering

---

**Success Criteria:** - Working backend API live online - Mobile app installable on Android and iPhone - All major features functional (scraping, matching, auto-apply) - Able to collect feedback from first users

---

**Tools Used:** - FastAPI, Uvicorn, PostgreSQL, Snowflake - Flutter, Dart, JWT Auth - GitHub, Codespaces, Streamlit (for prototyping) - Play Store & App Store

---

**Owner:** Makwande Gcora

**Company:** Makwande Careers

**Target:** Complete MVP & go live in 7 days