PopCareers MVP
Development Checklist & Task Assignment

Team Members: Mohamed Afif & Ridouane Lachguer

January 8, 2026

Project Status

Overall Progress: 86% Complete
Backend: Fully Operational (PHP 8.5.1, Laravel, PostgreSQL, Groq API)
Frontend: In Development (React 19, TypeScript, Vite)
Database: PostgreSQL 13+ with pgAdmin

Contents

1 Feature Overview

1.1 Project Tech Stack

- **Frontend:** React 19 + TypeScript + Vite + TailwindCSS
- **Backend:** Laravel 10 + PHP 8.5.1
- **Database:** PostgreSQL 13+
- **AI Integration:** Groq API (Llama 3.3 70B) for CV Parsing
- **PDF Processing:** Python + PyMuPDF
- **Authentication:** JWT (Sanctum)

2 Feature List & Task Assignment

2.1 Feature 1: AI-Powered Pro Tips in Student Dashboard

Feature Details

Priority: HIGH **Status:** IN PROGRESS

Description: Integrate AI-powered CV analysis section in student profile displaying strengths and improvement areas.

2.1.1 Requirements

- ☐ Create new "CV Analysis" section in student profile
- ☐ Call Groq API to analyze CV content
- ☐ Display strengths (points forts) from AI analysis
- ☐ Display improvement areas (axes d'amélioration)
- ☐ Format and structure AI response properly
- ☐ Add loading states during analysis
- ☐ Add error handling for API failures
- ☐ Cache analysis results in database

2.1.2 Backend Tasks

- ☐ Create `CVAnalysis` model to store results
- ☐ Create API endpoint: `GET /api/cv-analysis`
- ☐ Create API endpoint: `POST /api/cv-analysis/refresh`
- ☐ Database migration for cv_analyses table

2.1.3 Frontend Tasks

- ☐ Create `CVAnalysisSection` component
- ☐ Integrate into Profile page
- ☐ Add UI for strengths display
- ☐ Add UI for improvement areas
- ☐ Add refresh button

Assigned to: _____ **Due Date:** _____

2.2 Feature 2: Update Quick Actions in Student Dashboard

Feature Details

Priority: HIGH **Status:** TODO

Description: Remove "Public Profile" card from Quick Actions in student dashboard.

2.2.1 Requirements

- ☐ Locate QuickActions component in StudentDashboard
- ☐ Remove PublicProfile card
- ☐ Keep: Edit Profile, My QR Card, Applications (*will be removed in Feature 3*)
- ☐ Test navigation still works

2.2.2 Frontend Tasks

- ☐ Edit `StudentDashboard.tsx` or QuickActions component
- ☐ Remove PublicProfile card JSX
- ☐ Verify layout adjusts properly

Assigned to: _____ **Due Date:** _____

2.3 Feature 3: Remove Applications Page from Student Features

Feature Details

Priority: HIGH **Status:** TODO

Description: Applications page is only static data. Remove completely from student features; implement proper backend logic later if needed for recruiters only.

2.3.1 Requirements

- ☐ Remove "Applications" from student navbar
- ☐ Remove Applications.tsx page component
- ☐ Remove any student application routes
- ☐ Remove Applications menu item from navigation
- ☐ Update sidebar/navigation logic

2.3.2 Frontend Tasks

- ☐ Delete `src/pages/Applications.tsx`
- ☐ Remove route from `App.tsx`
- ☐ Remove from Navbar component
- ☐ Test student nav still works

Assigned to: _____ **Due Date:** _____

2.4 Feature 4: Recruiter Profile Page - Adapt for Recruiter Fields

Feature Details

Priority: HIGH **Status:** TODO

Description: Currently showing student fields (Experience, Skills, Education). Create recruiter-specific profile page.

2.4.1 Requirements

- ☐ Remove Experience, Skills, Education tabs from recruiter profile
- ☐ Add recruiter-specific fields:
  - Company Name
  - Company Description
  - Industry/Sector
  - Company Size
  - Website URL
  - Company Logo upload
  - Hiring preferences
  - Job posting permissions
- ☐ Create separate ProfileRecruiter component
- ☐ Update backend API for recruiter profile

2.4.2 Backend Tasks

- ☐ Update `Recruiter` model with new fields
- ☐ Database migration to add recruiter fields
- ☐ Update API: `GET /api/profile` (detect role)
- ☐ Update API: `PUT /api/profile` (handle recruiter data)
- ☐ Add file upload handling for company logo

2.4.3 Frontend Tasks

- ☐ Create `ProfileRecruiter.tsx` component
- ☐ Remove student-specific tabs from recruiter view
- ☐ Add recruiter form fields
- ☐ Add company logo upload component
- ☐ Update Profile page routing to show correct component

Assigned to: _____ **Due Date:** _____

2.5 Feature 5: Recruiter Shortlist - Export & Skill Normalization

Feature Details

Priority: MEDIUM **Status:** TODO

Description: Enable export button functionality and normalize skill names in filter dropdown.

2.5.1 Requirements - Export Feature

- ☐ Implement export to CSV format
- ☐ Export to Excel (.xlsx) format
- ☐ Export to PDF format (optional)
- ☐ Include: Name, Email, Skills, Education, Phone
- ☐ Allow export of selected or all candidates
- ☐ Add download button functionality

2.5.2 Requirements - Skill Normalization

- ☐ Create skill normalization mapping
- ☐ Examples: "Machine Learning" = "ML", "JS" = "JavaScript"
- ☐ Update filter dropdown to show normalized skills
- ☐ Remove duplicate/variation skills from filter
- ☐ Backend should normalize skills when storing/filtering

2.5.3 Backend Tasks

- ☐ Create `SkillNormalization` model/table
- ☐ Add skill alias mappings to database
- ☐ Create API: `POST /api/shortlist/export` (return file)
- ☐ Create normalization function
- ☐ Update skill filtering logic

2.5.4 Frontend Tasks

- ☐ Make Export button functional
- ☐ Add export format selection dropdown
- ☐ Handle file download in browser
- ☐ Update skill filter to use normalized names
- ☐ Add loading state during export

Assigned to: _____ **Due Date:** _____

2.6 Feature 6: Recruiter Smart Student Search

Feature Details

Priority: MEDIUM **Status:** TODO

Description: Add advanced natural language search for recruiters to find students by prompt/query in database.

2.6.1 Requirements

- ☐ Create advanced search bar in Scanner/Search page
- ☐ Allow natural language prompts (e.g., "Find Python developers from ENSAM")
- ☐ Real-time search suggestions
- ☐ Filter by multiple criteria simultaneously:
  - Skills (normalized)
  - University
  - Degree/Field
  - Location
  - Experience level
- ☐ Return matching student profiles
- ☐ Highlight matching criteria

2.6.2 Backend Tasks

- ☐ Create API: `POST /api/students/search`
- ☐ Parse natural language query (or use structured params)
- ☐ Build complex database query with filters
- ☐ Return paginated results
- ☐ Add search analytics/logging

2.6.3 Frontend Tasks

- ☐ Create `AdvancedSearch` component
- ☐ Add prompt input field
- ☐ Add multi-select filters
- ☐ Display search results
- ☐ Add pagination
- ☐ Highlight matching skills/criteria

Optional: Integrate with Groq API for AI-powered intelligent matching
Assigned to: _____ **Due Date:** _____
—

2.7 Feature 7: UI/UX Enhancement - Professional Polish

Feature Details

Priority: HIGH **Status:** IN PROGRESS

Description: Comprehensive UI/UX refinement for professional, polished appearance across all pages.

2.7.1 Requirements - Design System

- ☐ Consistent spacing/padding across all pages
- ☐ Unified typography (font sizes, weights)
- ☐ Consistent color palette (brand colors)
- ☐ Button styling standardization
- ☐ Card component styling
- ☐ Modal/dialog styling
- ☐ Form input styling

2.7.2 Requirements - User Experience

- ☐ Loading states (spinners, skeletons)
- ☐ Smooth animations/transitions
- ☐ Improved form validation feedback
- ☐ Better error messages
- ☐ Success notifications/toasts
- ☐ Hover states on interactive elements
- ☐ Focus states for accessibility

2.7.3 Requirements - Responsiveness

- ☐ Mobile-first design approach
- ☐ Test on multiple screen sizes
- ☐ Tablet layout optimization
- ☐ Desktop layout optimization
- ☐ Touch-friendly button sizes

2.7.4 Requirements - Accessibility

- ☐ WCAG 2.1 AA compliance
- ☐ Proper heading hierarchy
- ☐ Alt text for images
- ☐ Keyboard navigation support
- ☐ Screen reader compatibility
- ☐ Color contrast ratios
- ☐ ARIA labels where needed

2.7.5 Requirements - Additional

- ☐ Professional icon set
- ☐ Consistent micro-interactions
- ☐ Dark mode support (optional)
- ☐ Loading animations
- ☐ Empty states design
- ☐ Error page design

2.7.6 Frontend Tasks (Distributed)

- ☐ Review TailwindCSS config
- ☐ Create reusable component library
- ☐ Update all existing pages with new design
- ☐ Add animation library (Framer Motion)
- ☐ Implement toast notifications
- ☐ Add loading states to all API calls
- ☐ Test on multiple browsers/devices

Assigned to: _____ **Due Date:** _____

3 Task Assignment Matrix

Feature	Assigned to	Priority	Status	Est. Hours	
Feature 1: AI Pro Tips		HIGH	TODO	16	
Feature 2: Quick Actions		HIGH	TODO	4	
Feature 3: Remove Applications		HIGH	TODO	6	
Feature 4: Recruiter Profile		HIGH	TODO	20	
Feature 5: Export & Skills		MEDIUM	TODO	24	
Feature 6: Smart Search		MEDIUM	TODO	28	
Feature 7: UI/UX Polish		HIGH	IN PROGRESS	40	
TOTAL				**138 hours**	

4 Implementation Guidelines

4.1 Code Quality Standards

- Use TypeScript strictly (no `any` types)
- Follow ESLint rules configured in project
- Write proper error handling
- Add unit tests for critical functions
- Comment complex logic

4.2 Backend Development

- Create migrations for database changes
- Use Laravel best practices (Models, Controllers, Routes)
- Add proper validation on all endpoints
- Use JWT token authentication
- Log all important actions
- Add error responses with proper HTTP status codes

4.3 Frontend Development

- Create reusable components
- Use React hooks properly
- Manage state with context API or state management
- Test components in isolation
- Use TailwindCSS utility classes
- Optimize performance (lazy loading, memoization)

4.4 Git Workflow

- Create feature branches: `feature/feature-name`
- Use descriptive commit messages
- Create pull requests before merging
- Review code with partner
- Merge to main only when complete & tested

5 Testing Checklist

Before Merging to Main

- ☐ Functionality tested locally
- ☐ No console errors
- ☐ API endpoints responding correctly
- ☐ Database migrations successful
- ☐ UI looks good on desktop/mobile
- ☐ Forms validate properly
- ☐ Error handling works
- ☐ Tested with real API keys
- ☐ No hardcoded values

6 Timeline

Phase	Duration	Target Date	
Phase 1 (Features 1-3)	1-2 weeks	_____	
Phase 2 (Features 4-5)	2 weeks	_____	
Phase 3 (Feature 6)	1-2 weeks	_____	
Phase 4 (Feature 7 - Final Polish)	1 week	_____	
Testing & Deployment	1 week	_____	

7 Important Notes

Key Points

- **Database Backups:** Always backup database before running migrations
- **API Keys:** Keep Groq API key secure, never commit to repository
- **Testing:** Test thoroughly before marking as complete
- **Communication:** Update this checklist as you progress
- **Dependencies:** Keep packages updated, check for security vulnerabilities

8 Contacts & References

Team Members:

- You: _____
- Ridouane Lachguer: _____

Project Resources:

- Backend: `c:PopCareers`
- Frontend: `c:PopCareers`
- Database: PostgreSQL (localhost:5432) via pgAdmin
- API Documentation: See README.md in backend folder

Document Information

Created: January 8, 2026
Version: 1.0
Last Updated: January 8, 2026
Status: Active Development