

|  |
| --- |
| Business project  **Recruitment agency** |
|  |

Petar Nikić

Contents

[1 Business Description 3](#_Toc62212630)

[1.1 Business background 3](#_Toc62212631)

[1.2 Problems. Current Situation 3](#_Toc62212632)

[1.3 The benefits of implementing a database. Project Vision 3](#_Toc62212633)

[2 Model description 4](#_Toc62212634)

[2.1 Definitions & Acronyms 4](#_Toc62212635)

[2.2 Logical Scheme 4](#_Toc62212636)

[2.3 Objects 5](#_Toc62212637)

# 

# 

# Business Description

## 

## **Business background**

A recruitment agency connects job-seeking candidates with client companies. The agency publishes job listings, screens and registers candidates, and manages the hiring pipeline from application through interviews to placement. To improve candidate outcomes and generate additional revenue, the agency also offers paid services such as resume/CV writing, interview coaching, and skills development sessions delivered by agency staff.

## 

## **Problems.** **Current Situation**

Data is scattered across spreadsheets, emails, and messaging tools. Duplicate entries and inconsistent naming make candidate/job search slow and easy to make mistakes. There’s no single source to see where each application sits in the pipeline or which services a candidate purchased. Matching based on skills/experience/location is manual, and reporting takes a lot of labor.

## **the Benefits of implementing a database. Project Vision**

This database puts all the agency’s info in one clean place: candidates, jobs, applications, interviews, placements, skills, and extra services. It helps match people to the right jobs using their skills and experience, tracks each person from sign-up to hire, and shows which staff member is responsible. It also lets us sell and schedule things like CV help or coaching. With built-in rules to prevent duplicates and mistakes, data stays accurate. And we can easily see key stats like time to fill, best sourcing channels, stage-to-stage conversion, service revenue, and average number of interviews.

# Model description

## **Definitions & Acronyms**

* **3NF** - Third Normal Form
* **PK** - Primary Key
* **FK** - Foreign Key
* **UK** - Unique Key
* **CK** - Check Constraint
* **M:N** - Many-to-many relationship
* **1**:M – 1-many relationship

## **Logical Scheme**

A diagram of a computer

AI-generated content may be incorrect.

## Objects

**Table description:**

* **Company** – Client companies posting jobs and stores HQ location.
* **Location** – Reusable places (country/region/city).
* **Staff** – Agency staff (recruiters/coaches) and optional manager.
* **JobListing** – Open roles (company, location, salary range, status).
* **Candidate** – Registered people (contact, location, experience, desired salary).
* **CandidateExperience** – Work history entries per candidate.
* **Skill** – Skill list.
* **CandidateSkill** – Bridge: Candidate↔Skill with level and years.
* **JobSkill** – Bridge: JobListing↔Skill with required level and mandatory flag.
* **Application** – Candidate applies to a Job, owned by Staff, status and source.
* **Interview** – Scheduled interviews for an Application.
* **Placement** – Final outcome for an Application (offer/start).
* **Service** – Paid services offered to candidates (coaching, CV).
* **CandidateService** – A purchased/scheduled service for a candidate, delivered by Staff.

**Company:**

|  |  |  |
| --- | --- | --- |
| Int | company\_id | PK |
| string | name |  |
| string | industry |  |
| string | website |  |
| Int | hq\_location\_id | FK location.location\_id |

* **Company.hq\_location\_id → Location.location\_id (many companies can share one HQ location) - 1:N**
* **JobListing.company\_id → Company.company\_id (one company has many job listings) - 1:N**

**Example table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| company\_id | name | industry | website | hq\_location\_id |
| 12 | CompanyINC | manufacturing | companyinc.com | 10 |

**Location:**

|  |  |  |
| --- | --- | --- |
| **int** | **location\_id** | **PK** |
| **string** | **country** |  |
| **string** | **region** |  |
| **string** | **city** |  |

* **Company.hq\_location\_id → Location.location\_id (companies at a location) - 1:N**
* **JobListing.location\_id → Location.location\_id (jobs at a location) - 1:N**
* **Candidate.current\_location\_id → Location.location\_id (candidates at a location) - 1:N**

|  |  |  |  |
| --- | --- | --- | --- |
| **location\_id** | **country** | **region** | **city** |
| **1** | **Serbia** | **Eastern Serbia** | **Negotin** |

**Staff:**

|  |  |  |
| --- | --- | --- |
| int | staff\_id | PK |
| string | full\_name |  |
| string | email | UK |
| string | role |  |
| int | manager\_id | FK staff.staff\_id |

* **JobListing.posted\_by\_staff\_id → Staff.staff\_id (one staff member can post many jobs) - 1:N**
* **Application.handled\_by\_staff\_id → Staff.staff\_id (one staff member can own many applications) - 1:N**
* **CandidateService.assigned\_staff\_id → Staff.staff\_id (one staff member delivers many service sessions) - 1:N**
* **Staff.manager\_id → Staff.staff\_id (many staff report to one manager) - 1:N (self) (Optional self-reference: manager\_id → Staff.staff\_id. (makes simple org chart without seperate managers table, if someone has no boss manager\_id is NULL) )**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| staff\_id | full\_name | email | role | manager\_id |
| 5 | Petar Nikic | [petarnikic@something.com](mailto:petarnikic@something.com) | recruiter | 1 |

**Job listing:**

|  |  |  |
| --- | --- | --- |
| int | job\_id | PK |
| int | company\_id | FK company.company\_id |
| string | title |  |
| string | employment\_type | CK (full-time/part-time/intern) |
| int | location\_id | FK location.location\_id |
| decimal | salary\_min |  |
| decimal | salary\_max |  |
| int | posted\_by\_staff\_id | FK staff.staff\_id |
| string | status | CK (open/closed/hold) |

* **JobListing.company\_id → Company.company\_id** (each job belongs to one company) - **1:N**
* **JobListing.location\_id → Location.location\_id** (each job in one location) - **1:N**
* **JobListing.posted\_by\_staff\_id → Staff.staff\_id** (job posted by one staff) - **1:N**
* **Application.job\_id → JobListing.job\_id** (one job has many applications) - **1:N**
* **JobSkill.job\_id → JobListing.job\_id** (one job has many required skills) - **1:N**  
  ⇒ **JobListing ↔ Skill** via **JobSkill** is **M:N**. (many to many)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| job\_id | company\_id | title | employment\_type | location\_id | salary\_min | salary\_max | posted\_by\_staff\_id | status |
| 2 | 3 | data engineer | Full-time | 10 | 50000 | 100000 | 5 | open |

**Candidate:**

|  |  |  |
| --- | --- | --- |
| int | candidate\_id | PK |
| string | first\_name |  |
| string | last\_name |  |
| string | email | UK |
| string | phone |  |
| int | current\_location\_id | FK location.location\_id |
| int | years\_experience |  |
| decimal | desired\_salary\_min |  |

* **Candidate.current\_location\_id → Location.location\_id** (candidate lives in one location) - **1:N**
* **CandidateExperience.candidate\_id → Candidate.candidate\_id** (one candidate has many experience rows) - **1:N**
* **Application.candidate\_id → Candidate.candidate\_id** (one candidate has many applications) - **1:N**
* **CandidateSkill.candidate\_id → Candidate.candidate\_id** (one candidate has many skill rows) - **1:N**  
  ⇒ **Candidate ↔ Skill** via **CandidateSkill** is **M:N**.
* **CandidateService.candidate\_id → Candidate.candidate\_id** (one candidate buys many services) - **1:N**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| candidate\_id | first\_name | last\_name | email | phone | current\_location\_id | years\_exp | desired\_salary\_min |
| 14 | Petar | Nikic | [petarnikic@something.com](mailto:petarnikic@something.com) | +051/5535123 | Serbia | 5 | 50000 |

**Candidate experience:**

|  |  |  |
| --- | --- | --- |
| int | experience\_id | PK |
| int | candidate\_id | FK candidate.candidate\_id |
| string | company\_name |  |
| string | title |  |
| date | start\_date |  |
| date | end\_date |  |

* **CandidateExperience.candidate\_id → Candidate.candidate\_id** (many experience rows per candidate) - **1:N**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| experience\_id | candidate\_id | company\_name | title | start\_date | end\_date |
| 51 | 14 | BigCompany | data engineer | 2022-1-1 | 2024-1-1 |

**Skill:**

|  |  |  |
| --- | --- | --- |
| int | skill\_id | PK |
| string | name | UK |
| string | category |  |

* **CandidateSkill.skill\_id → Skill.skill\_id** (one skill appears on many candidates) - **1:N**
* **JobSkill.skill\_id → Skill.skill\_id** (one skill is required by many jobs) - **1:N**  
  ⇒ Supports both **M:N** bridges above.

|  |  |  |
| --- | --- | --- |
| skill\_id | name | category |
| 2 | Python | Programming |

**Candidate skill:**

|  |  |  |
| --- | --- | --- |
| int | candidate\_id | PK+FK candidate |
| int | skill\_id | PK+FK skill |
| string | level | CK (junior/medior/senior) |
| decimal | years |  |

* **CandidateSkill** has a **composite primary key** (candidate\_id, skill\_id). Each component is also a **foreign key**: candidate\_id → Candidate(candidate\_id), skill\_id → Skill(skill\_id). This prevents duplicate candidate–skill pairs
* **CandidateSkill.candidate\_id → Candidate.candidate\_id** (many rows per candidate) - **1:N**
* **CandidateSkill.skill\_id → Skill.skill\_id** (many rows per skill) - **1:N**  
  ⇒ **Candidate ↔ Skill** is **M:N** through this table (composite PK: candidate\_id + skill\_id).

|  |  |  |  |
| --- | --- | --- | --- |
| candidate\_id | skill\_id | level | years |
| 301 | 23 | junior | 4 |

**Job skill:**

|  |  |  |
| --- | --- | --- |
| int | job\_id | PK+FK JobLiting |
| int | skill\_id | PK+FK skill |
| string | required\_level | CK |
| bool | is\_mandatory |  |

* **JobSkill.job\_id → JobListing.job\_id** (many rows per job) - **1:N**
* J**obSkill.skill\_id → Skill.skill\_id** (many rows per skill) - **1:N**  
  ⇒ **JobListing ↔ Skill** is **M:N** through this table (composite PK: job\_id + skill\_id).

|  |  |  |  |
| --- | --- | --- | --- |
| job\_id | skill\_id | required\_level | is\_mandatory |
| 15 | 24 | Medior | true |

**Application:**

|  |  |  |
| --- | --- | --- |
| int | application\_id | PK |
| int | candidate\_id | FK candidate.candidate\_id |
| int | job\_id | FK joblisting.job\_id |
| datetime | applied\_at |  |
| string | source |  |
| string | status | CK (submitted, in review, etc) |
| int | handled\_by\_staff | FK staff.staff\_id |

* **Application.candidate\_id → Candidate.candidate\_id** (many applications per candidate) - **1:N**
* **Application.job\_id → JobListing.job\_id** (many applications per job) - **1:N**
* **Application.handled\_by\_staff\_id → Staff.staff\_id** (many applications per staff owner) - **1:N**
* **Interview.application\_id → Application.application\_id** (one application can have many interviews) - **1:N**
* **Placement.application\_id → Application.application\_id** (typically 0..1 placement per application) - **1:N (business rule 0..1),** Each application can result in at most one placement

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| application\_id | candidate\_id | job\_id | applied\_at | source | status | handled\_by\_staff |
| 123 | 321 | 2 | 2025-10-23 | Linkedin | Submited | 5 |

**Interview:**

|  |  |  |
| --- | --- | --- |
| int | interview\_id | PK |
| int | application\_id | FK application. |
| datetime | scheduled\_at |  |
| string | mode | CK (onsite/remote/phone) |
| string | result | CK (pass/fail/pending) |

* **Interview.application\_id → Application.application\_id** (many interviews for one application) - **1:N**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| interview\_id | application\_id | scheduled\_at | mode | result |
| 12 | 32 | 2024-12-12 12:00 | onsite | pending |

**Placement:**

|  |  |  |
| --- | --- | --- |
| int | placement\_id | PK |
| int | application\_id | FK application.application\_id |
| decimal | offer\_salary |  |
| date | start\_date |  |
| date | end\_date |  |
| string | contract\_type | CK (permanent/fixed) |

* **Permanent hires:** end\_date = NULL.
* **Placement.application\_id → Application.application\_id** (usually one placement per application) - **1:N (0..1)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| placement\_id | application\_id | offer\_salary | start\_date | end\_date | contract\_type |
| 12 | 32 | 50000.00 | 2 |  |  |

**Service:**

|  |  |  |
| --- | --- | --- |
| int | service\_id | PK |
| string | name | UK |
| decimal | base\_price |  |
| int | duration\_minutes |  |

* **CandidateService.service\_id → Service.service\_id** (one service type used in many bookings) - **1:N**

|  |  |  |  |
| --- | --- | --- | --- |
| service\_id | name | base\_price | duration\_minutes |
| 12 | Interview coaching | 100 | 60 |

**Candidate Service:**

|  |  |  |
| --- | --- | --- |
| int | candidate\_service\_id | PK |
| int | candidate\_id | FK candidate.candidate\_id |
| int | service\_id | Fk service.service\_id |
| datetime | purchased\_at |  |
| string | status | CK (scheduled/completed/canceled) |
| int | assigned\_staff\_id | FK staff.staff\_id |

* **CandidateService.candidate\_id → Candidate.candidate\_id** (many bookings per candidate) - **1:N**
* **CandidateService.service\_id → Service.service\_id** (many bookings per service type) - **1:N**
* **CandidateService.assigned\_staff\_id → Staff.staff\_id** (many bookings delivered by one staff) - **1:N**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| candidate\_service\_id | candidate\_id | service\_id | purchased\_at | status | assigned\_staff\_id |
| 123 | 32 | 41 | 2024-1-1 12:13 | completed | 15 |