# **Application Documentation**

## **1. Introduction**

This document explains the application’s components, functionality, and database structure. The application consists of an **Admin Panel** for managing positions and skills and a **Form Page** where users interact with the data. The backend is built with PHP and MySQL, while the frontend uses HTML, CSS, and JavaScript.

## **2. Application Components**

### **2.1 Admin Page (admin.html)**

The Admin Page allows administrators to manage job positions and their associated skills.

#### **Key Features:**

* **Sidebar Navigation:** Links to Dashboard, Candidates, Modify Skills, and Available Positions.
* **Dynamic Content Area:** Updates based on menu selection.
* **JavaScript Integration:** Fetches and displays position and skill data dynamically.

#### **Backend & Frontend Integration:**

* **PHP:** Retrieves positions and skills from the database and converts them to JSON.
* **JavaScript:** Parses JSON data, populates dropdowns, and updates the skill list dynamically.

### **2.2 Form Page (form.html / index.html)**

The form page allows users to select a **position** and view corresponding **skills**.

#### **Key Features:**

* **Dropdown for Position Selection:** Dynamically populated from the database.
* **Skills Display Section:** Updates based on the selected position.
* **JavaScript Handling:** Ensures skills are relevant to the chosen position.

#### **Data Retrieval:**

* **AJAX / Fetch API:** Calls data.php to fetch skills without a page reload.
* **JavaScript:** Updates the skills list dynamically.

## **3. Database Structure**

The application uses a **MySQL database** with three main tables:

### **3.1 Positions Table**

Stores job positions.

CREATE TABLE positions (  
 id INT PRIMARY KEY AUTO\_INCREMENT,  
 position\_name VARCHAR(255) NOT NULL  
);

**Example Data:**

| id | position\_name |
| --- | --- |
| 1 | Business Analyst |
| 2 | Graphic Designer |
| 3 | Human Resources |
| 4 | Cybersecurity |
| 5 | AI Developer |

### **3.2 Skills Table**

Stores skills.

CREATE TABLE skills (  
 id INT PRIMARY KEY AUTO\_INCREMENT,  
 skill\_name VARCHAR(255) NOT NULL  
);

**Example Data:**

| id | skill\_name |
| --- | --- |
| 109 | Effective Communication |
| 110 | Critical Thinking |
| 111 | Organizational Skills |
| 112 | Data Analysis |

### **3.3 Position\_Skills Table**

Joins positions and skills (many-to-many relationship).

CREATE TABLE position\_skills (  
 id INT PRIMARY KEY AUTO\_INCREMENT,  
 position\_id INT NOT NULL,  
 skill\_id INT NOT NULL,  
 FOREIGN KEY (position\_id) REFERENCES positions(id),  
 FOREIGN KEY (skill\_id) REFERENCES skills(id)  
);

**Example Data:**

| id | position\_id | skill\_id |
| --- | --- | --- |
| 1 | 1 | 109 |
| 2 | 1 | 112 |
| 3 | 3 | 110 |
| 4 | 4 | 135 |
| 5 | 5 | 141 |

## **4. Backend - Data Handling (PHP Scripts)**

### **4.1 data.php (Retrieves Positions & Skills)**

<?php  
include 'db.php';  
  
$positionsQuery = "SELECT id, position\_name FROM positions";  
$positionsResult = mysqli\_query($conn, $positionsQuery);  
$positions = [];  
  
while ($row = mysqli\_fetch\_assoc($positionsResult)) {  
 $positions[$row['id']] = $row['position\_name'];  
}  
  
$skillsQuery = "SELECT s.skill\_name, p.position\_name  
 FROM skills s  
 JOIN position\_skills ps ON s.id = ps.skill\_id  
 JOIN positions p ON ps.position\_id = p.id";  
  
$skillsResult = mysqli\_query($conn, $skillsQuery);  
$positionSkills = [];  
  
while ($row = mysqli\_fetch\_assoc($skillsResult)) {  
 $positionSkills[$row['position\_name']][] = $row['skill\_name'];  
}  
  
header('Content-Type: application/json');  
echo json\_encode(['positions' => $positions, 'positionSkills' => $positionSkills]);  
?>

### **How it Works:**

1. **Fetches Positions:** Queries positions table.
2. **Fetches Skills:** Joins skills and positions.
3. **Encodes Data as JSON:** Sends it to the frontend.

## **5. Frontend - JavaScript Code (Fetching & Displaying Data)**

### **5.1 script.js (Handles Position Selection & Skill Display)**

fetch("data.php")  
 .then(response => response.json())  
 .then(data => {  
 let positionDropdown = document.getElementById("position");  
 let skillsList = document.getElementById("skills-list");  
  
 Object.keys(data.positions).forEach(id => {  
 let option = document.createElement("option");  
 option.value = data.positions[id];  
 option.textContent = data.positions[id];  
 positionDropdown.appendChild(option);  
 });  
  
 positionDropdown.addEventListener("change", function() {  
 let selectedPosition = positionDropdown.value;  
 skillsList.innerHTML = "";  
   
 if (data.positionSkills[selectedPosition]) {  
 data.positionSkills[selectedPosition].forEach(skill => {  
 let listItem = document.createElement("li");  
 listItem.textContent = skill;  
 skillsList.appendChild(listItem);  
 });  
 } else {  
 skillsList.innerHTML = "<li>No skills found for this position.</li>";  
 }  
 });  
 })  
 .catch(error => console.error("Error fetching data:", error));

### **How it Works:**

1. **Fetches Data from data.php**
2. **Populates the Position Dropdown**
3. **Listens for Position Selection**
4. **Displays Relevant Skills**

## **6. Conclusion & Future Enhancements**

This application enables admins to manage job positions and skills while allowing users to view relevant skills for a selected position.

### **Potential Enhancements:**

* ✅ Add candidate registration & profile management.
* ✅ Implement search filters for positions & skills.
* ✅ Use AJAX for improved real-time updates.
* ✅ Enhance UI/UX with better styling & animations.

This document provides a clear overview of the application’s structure, database, and data flow. 🚀