**1. Entity Relationship Diagram (ERD) for Job Website**

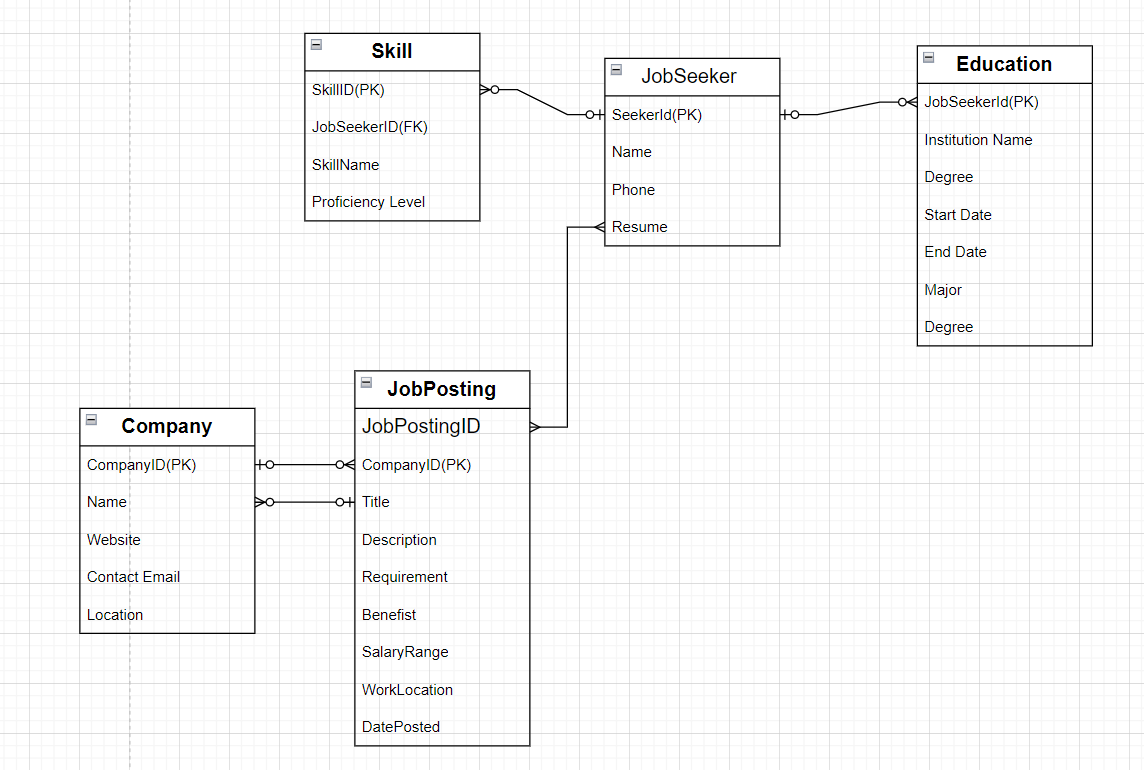
**Entities:**

* **JobSeeker:** (Primary Key: JobSeekerID)
  + Name
  + Email
  + Phone Number
  + Resume (Optional: File path)
* **Education:** (Primary Key: EducationID)
  + JobSeekerID (Foreign Key references JobSeeker.JobSeekerID)
  + Institution Name
  + Degree
  + Start Date
  + End Date
  + (Optional) Major
  + (Optional) Description
* **Skill:** (Primary Key: SkillID)
  + JobSeekerID (Foreign Key references JobSeeker.JobSeekerID)
  + Skill Name
  + Proficiency Level (e.g., Beginner, Intermediate, Advanced)
* **Company:** (Primary Key: CompanyID)
  + Name
  + Website (Optional)
  + Contact Email
  + Location (Address or City/State)
* **JobPosting:** (Primary Key: JobPostingID)
  + CompanyID (Foreign Key references Company.CompanyID)
  + Title
  + Description
  + Requirements (Text or List)
  + Benefits (Text or List)
  + Salary Range
  + Work Location (Address or City/State)
  + Date Posted

**Relationships:**

* One JobSeeker can have many Educations (One-to-Many)
* One JobSeeker can have many Skills (One-to-Many)
* One Company can have many JobPostings (One-to-Many)
* One JobPosting belongs to one Company (Many-to-One)
* A JobSeeker can apply for many JobPostings (Many-to-Many) (Optional: This requires an additional table, JobApplication, with JobSeekerID and JobPostingID as Foreign Keys)

**ERD Logical Data Model:**



**2. Project Process and Git Branching**

* **Branches in Git:** The number of branches used depends on project complexity and workflow. Here's a common approach:
  + **Master Branch:** This holds the stable and production-ready code.
  + **Develop Branch:** Used for ongoing development, feature implementation, and bug fixes.
  + **Feature Branches:** Created from Develop for isolated development of specific features. Merged back into Develop when complete.
  + **Hotfix Branches:** Created from Master for urgent bug fixes. Merged back into Master and Develop for deployment.
  + **Release Branches:** (Optional) Used for creating a pre-release staging environment for testing before pushing to Master.
* **Developing a New Module:**
  + **Planning:** Define requirements, user stories, and test cases.
  + **Implementation:** Code the module in a feature branch, following coding standards.
  + **Unit Testing:** Test individual code units (functions, classes) in isolation.
  + **Code Review:** Have peers review code for quality and adherence to best practices.
  + **Integration Testing:** Test module interactions with other parts of the codebase in the Develop branch.
  + **System Testing:** (Optional) Test the entire system functionality after integrating the module into the overall application.
  + **Deployment:** Once all testing passes, merge the feature branch into Develop and deploy to production (from Master) if applicable.

**3. Testing Strategies**

* **Unit Testing:** Tests individual units of code (functions, classes) in isolation to ensure they work as expected.
* **Integration Testing:** Tests how modules interact with each other to verify data flow and system functionality.
* **System Testing:** Tests the entire system end-to-end, simulating real-world user scenarios.
* **Regression Testing:** Re-runs existing tests after changes to ensure functionality hasn't regressed.
* **Usability Testing:** Evaluates the user interface and user experience for ease of use.
* **Performance Testing:** Assesses the application's performance under load to identify bottlenecks and ensure acceptable response times.
* **Security Testing:** Identifies and addresses security vulnerabilities to protect user data and system integrity.

**Tester's Approach to New Module Testing:**

1. **Understand Requirements:** Clearly understand the new module's functionalities and user stories.
2. **Review Test Cases:** Analyze existing and create new test cases for the new module.
3. **Unit Testing:** Collaborate with developers during unit testing and provide feedback.
4. **Integration Testing:** Test the module's interactions with other parts of the system.
5. **System Testing:** Participate in system testing to verify overall functionality.
6. **Reporting:** Document test results, identify bugs, and