NEXT HIRE

Your best opportunity

INFO ANALYSIS & SYSTEM DESIGN

Section CS670-D

Team -03

Sacred Heart University

5151 park Avenue,

Fairfield, CT 06825

Name	Email	Student ID
Vamshi Gutha	guthav@mail.sacredheart.edu	0931569
Siddhardha Mahanthy	mahanthys@mail.sacredheart.edu	0935121
Aravind reddy chendippa	Chendippaa@mail.sacredheart.edu	0936016

Table Of Contents

List of images

List of Figures

Agenda

Next Hire is a job hiring portal designed to simplify the recruitment process and empower job seekers and recruiters to connect effectively. Our platform focuses on providing innovative tools, real-time insights, and a user-friendly experience to meet the diverse needs of modern recruitment and job searching.

1. Simplifying Job Searches

Next Hire enables job seekers to explore a wide range of opportunities with advanced filters, making it easy to find roles that match their preferences. The streamlined application process ensures quick and hassle-free submissions, while personalized job recommendations help candidates discover opportunities tailored to their profiles.

2. Facilitating Recruiter-Job Seeker Communication

Our platform prioritizes direct and efficient communication by enabling recruiters and job seekers to engage with each other via email. This focused approach fosters meaningful interactions without unnecessary distractions, ensuring clarity and professionalism throughout the hiring process.

3. Empowering Recruiters with Real-Time Insights

Recruiters benefit from powerful, Salesforce-powered tools that deliver real-time insights into candidate engagement and job posting performance. These analytics enable recruiters to make data-driven decisions, refine hiring strategies, and efficiently track applications.

4. Ensuring a Seamless User Experience

With its responsive design, Next Hire is accessible across all devices, ensuring a smooth user experience. The platform offers 24/7 support to address any technical or service-related concerns, along with robust security features to protect user data and interactions.

Our Vision

Next Hire is dedicated to bridging the gap between talent and opportunity. By combining efficiency, simplicity, and innovation, we aim to empower users to achieve their career and hiring goals.

DATA FLOW DIAGRAM

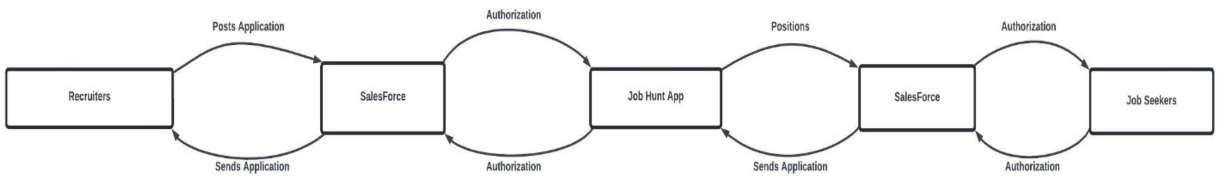


Figure 1: Data Flow Diagram

1. **Recruiters**
   - **Action:**
     Recruiters post job applications through Salesforce.
   - **Flow:**
     - Posts application → Salesforce.
2. **Salesforce**
   - **Role:**
     Acts as a central system to manage job postings and applications.
   - **Interactions:**
     - Receives job applications from Recruiters.
     - Shares available job positions with the Job Hunt App.

    - Facilitates authorization between job seekers and the system.

3. **Job Hunt App**
    - **Role:**
      Serves as an intermediary platform for job seekers to explore and apply for positions.
    - **Interactions:**
        - Receives job positions from Salesforce.
        - Sends applications from job seekers back to Salesforce.
4. **Job Seekers**
    - **Action:**
      Job seekers authorize and interact with the system to find and apply for jobs.
    - **Flow:**
        - Authorization with Salesforce.
        - Uses Job Hunt App to browse positions and apply for them.

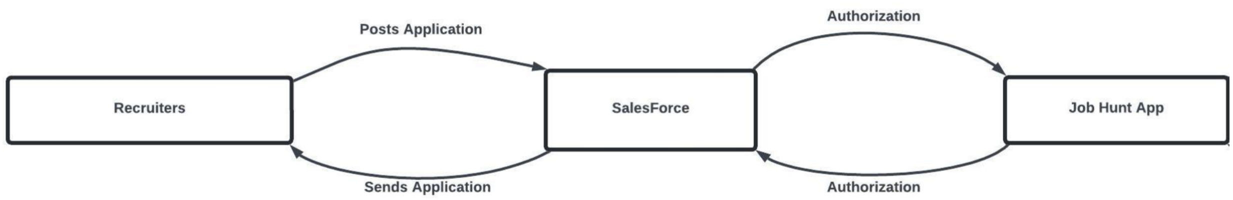


Figure 1.1: Data Flow Diagram

System Components and Workflow

1. **Recruiters**
   - **Role:**
     Responsible for posting job applications and receiving responses.
   - **Interactions:**
     - Posts job applications to Salesforce.
     - Receives applications through Salesforce.
2. **Salesforce**
   - **Role:**
     Acts as the central hub for managing job postings and facilitating communication between Recruiters and the Job Hunt App.
   - **Interactions:**
     - Receives job postings from Recruiters.
     - Shares these postings with the Job Hunt App.
     - Handles authorization requests from the Job Hunt App.

3. **Job Hunt App**

   - **Role:**
     A platform for job seekers to interact with available job opportunities.
   - **Interactions:**
     - Requests and completes authorization via Salesforce.
     - Accesses job postings from Salesforce.
     - Facilitates the submission of applications back to Salesforce.

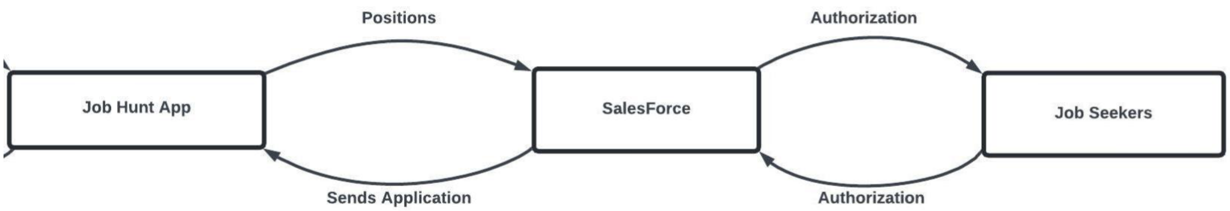


Figure 1.2: Data Flow Diagram

Job Hunt App

- **Role:**
  Acts as a platform for job seekers to browse job positions and submit applications.
- **Interactions:**
  - Receives job positions from Salesforce.
  - Sends job applications from job seekers to Salesforce.

Salesforce

- **Role:**
  Serves as the central system to manage job postings and job applications.
- **Interactions:**
  - Shares available positions with the Job Hunt App.
  - Facilitates authorization for job seekers and their interactions with the system.
  - Receives and processes applications from the Job Hunt App.

Job Seekers

- **Role:**
  Users seeking employment opportunities.
- **Interactions:**
  - Authorize their accounts through Salesforce.
  - Use the Job Hunt App to search for job positions and submit applications.

CONTEXT LEVEL DIAGRAM

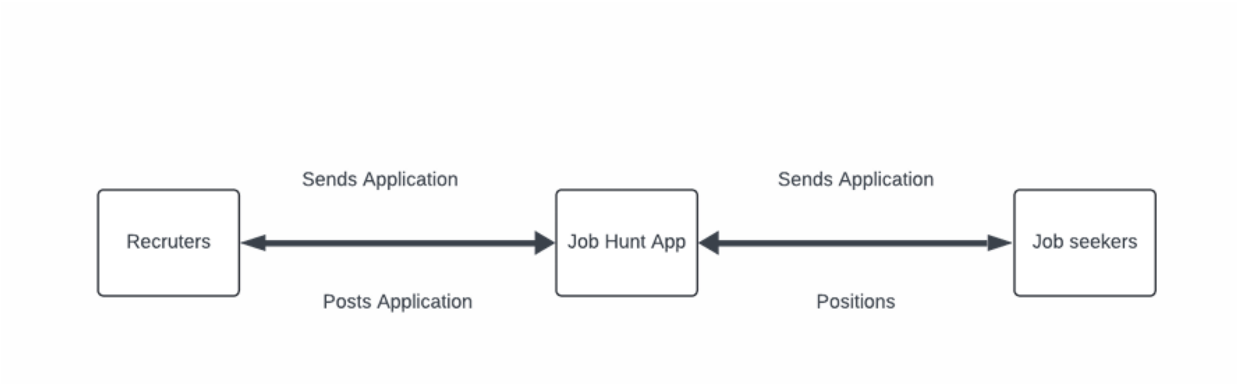


Figure 2: Context Level Diagram

1. Recruiters

- **Purpose**: Recruiters are responsible for posting job openings on the Job Hunt App and reviewing job applications submitted by job seekers.
- **Primary Functions**:
  - Post job openings to make them available to job seekers.
  - Review applications sent by job seekers through the Job Hunt App.
- **Interactions**:
  - **With Job Hunt App**: Sends job postings to the Job Hunt App and receives applications from job seekers.

2. Job Hunt App

- **Purpose**: A job search platform that allows job seekers to view and apply for job openings posted by recruiters.
    - Receives job postings from recruiters.
    - Displays available job positions to job seekers.
    - Facilitates the submission of job applications from job seekers to recruiters.
- **Data Managed**:
    - **Job Listings**: Collected from recruiters and displayed to job seekers.
    - **Job Applications**: Collected from job seekers and sent to recruiters.
- **Interactions**:
    - **With Recruiters**: Receives job postings and forwards applications.
    - **With Job Seekers**: Displays job listings and collects applications.

3. Job Seekers

- **Purpose**: Individuals seeking job opportunities who use the Job Hunt App to search for and apply to available positions.
- **Primary Functions**:
  - View available job openings on the Job Hunt App.
  - Submit applications for jobs of interest.
- **Interactions**:
  - **With NEXT HIRE App**: Searches for job listings, views available positions, and submits applications.

USE CASE DIAGRAM

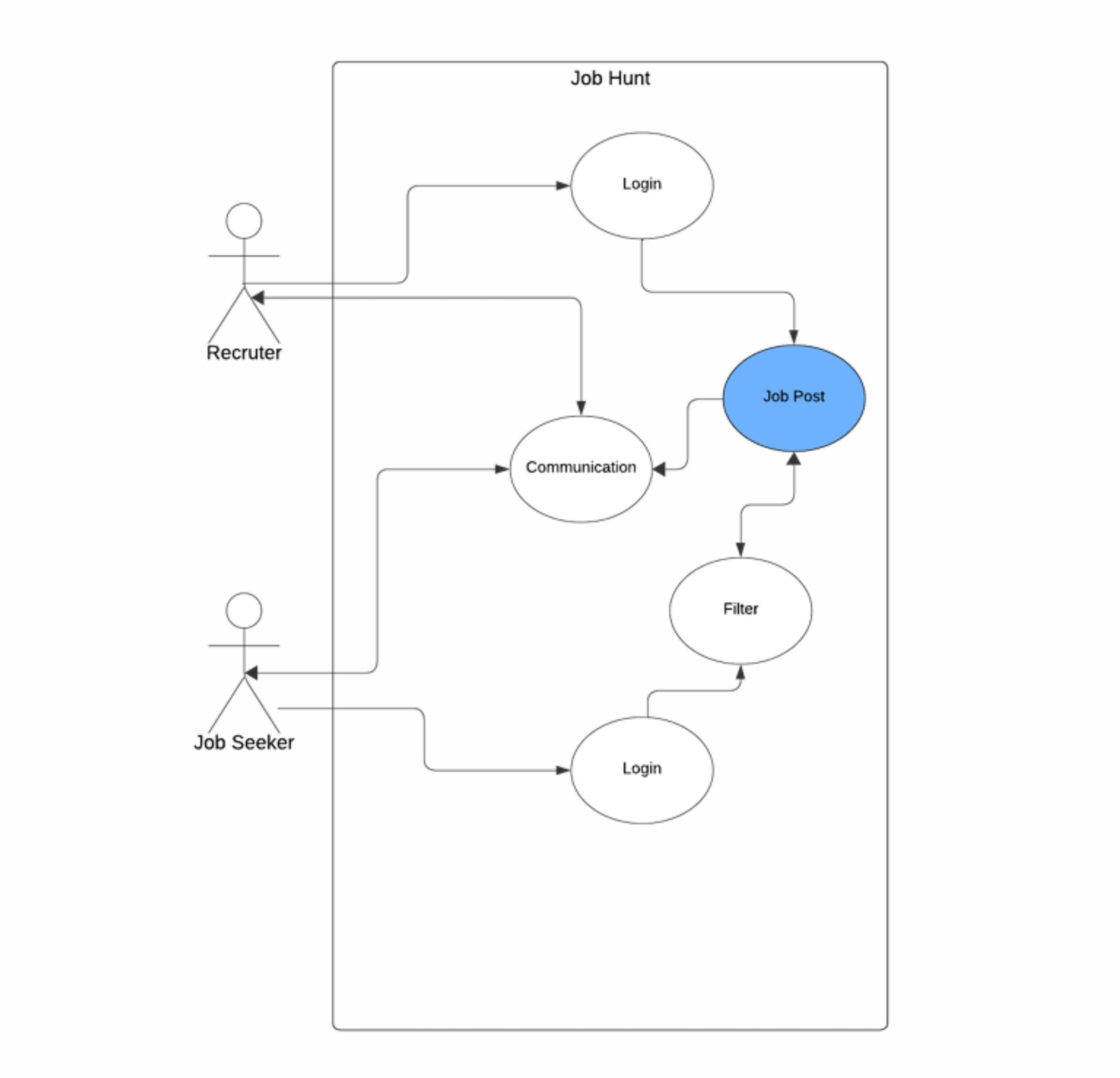


Figure 3: Use case Diagram

1. **Recruiter**

    - Represents a person or organization responsible for posting job opportunities and managing communication with job seekers.

2. **Job Seeker**
   - Represents an individual searching for job opportunities.

Use Cases:

1. **Login**
   - Both recruiters and job seekers need to log in to the system to access further functionalities.
2. **Job Post**
   - **Recruiters** can create or manage job postings that job seekers can view.
3. **Filter**
   - **Job Seekers** can filter job posts based on specific criteria (e.g., location, skills, or experience).
4. **Communication**
   - Facilitates interactions between recruiters and job seekers (e.g., inquiries, interviews, or feedback).

ACTIVITY DIAGRAM – LOGIN

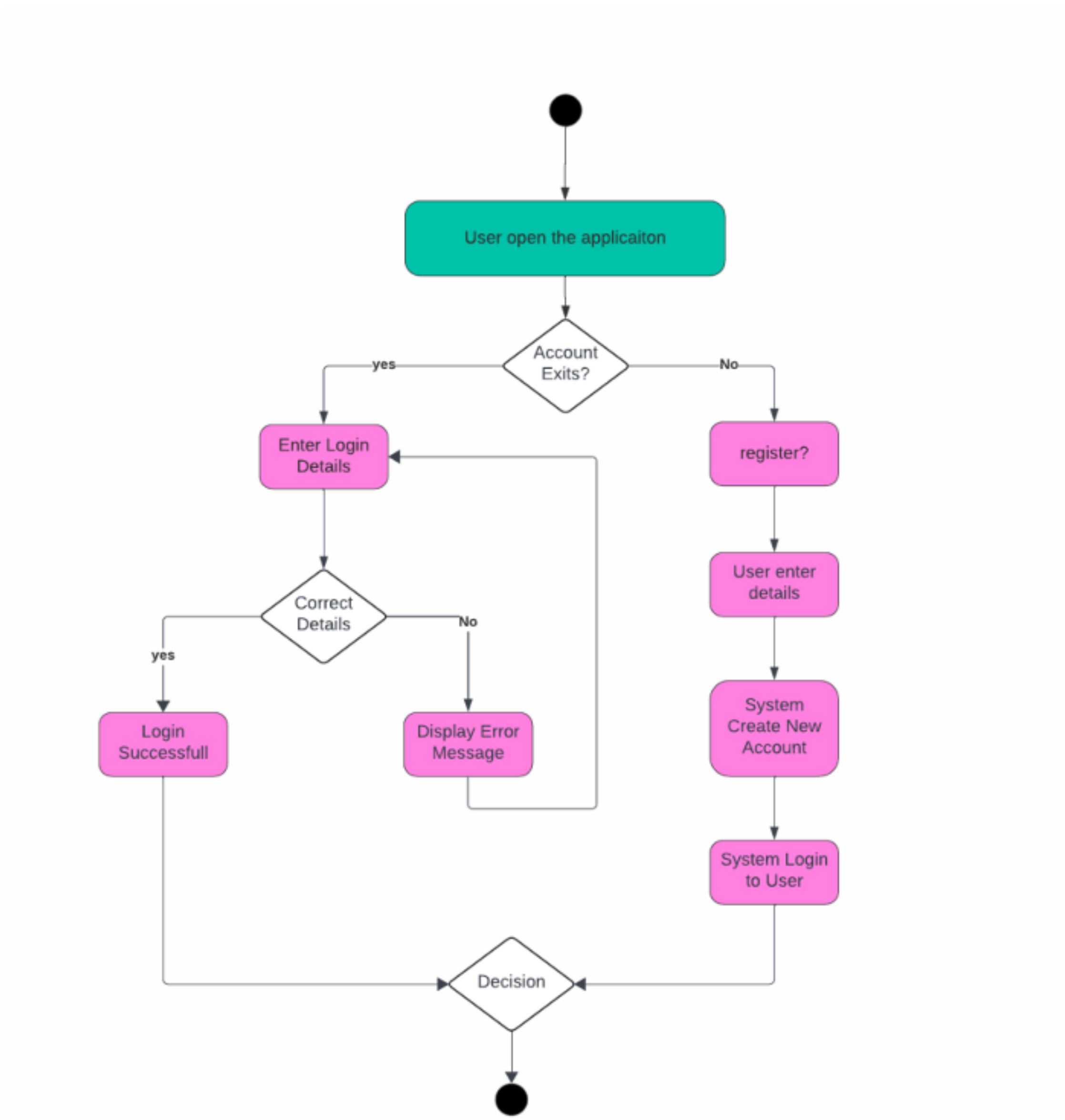


Figure 4: Activity Diagram Login

1. User Opens the Application

- The process starts when the user launches the application.

2. Check if an Account Exists

- **Decision Point:** Does the user already have an account?
  - **Yes:** Proceed to step 3.
  - **No:** Proceed to step 6.

Login Process (For Existing Users):

3. Enter Login Details

- The user inputs their login credentials (e.g., username and password).

4. Verify Login Details

- **Decision Point:** Are the credentials correct?
  - **Yes:** Proceed to step 5.
  - **No:** Display an error message and return to step 3.

5. Login Successful

- If credentials are correct, the user is granted access to the application.
- The process ends here.

Registration Process (For New Users):

6. Register a New Account

- The user is prompted to register since no account exists.

7. Enter Registration Details

- The user provides the required information (e.g., name, email, password).

8. System Creates a New Account

- The system validates the details and sets up a new account for the user.

9. Automatic Login After Registration

- Once the account is created, the system logs the user in automatically.

10. End of Process

- Whether the user logs in successfully (step 5) or completes registration and login (step 9), the process concludes here.

ACTIVITY DIAGRAM – RECRUITER

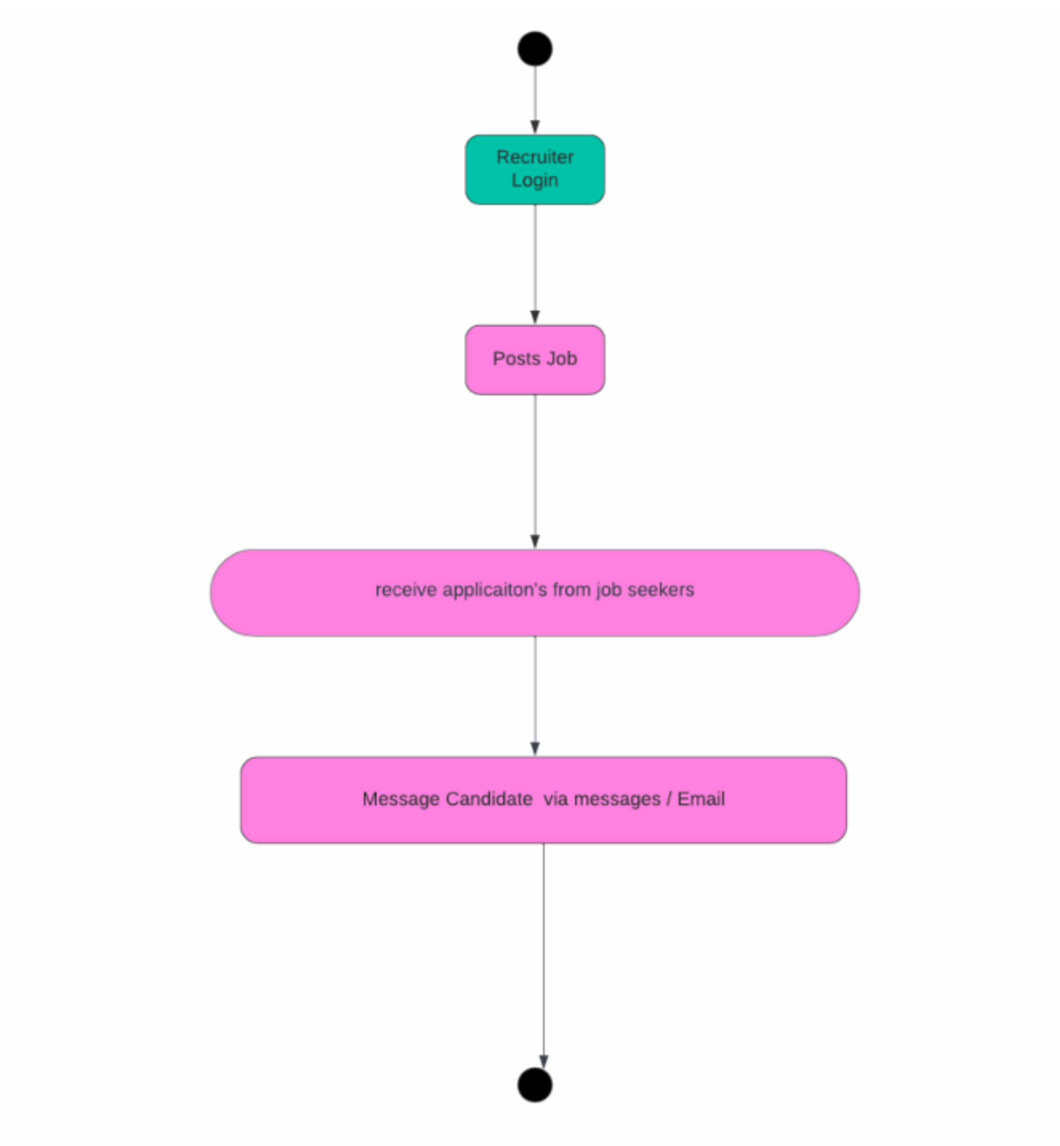


Figure 5: Activity Diagram Recruiter

1. Recruiter Login

- The process begins with the recruiter logging into the system.

- This step ensures authenticated access to the platform.

2. Post Job

- Once logged in, the recruiter creates and posts a job.
- The job post typically includes details like the job title, description, requirements, and other relevant information.

3. Receive Applications from Job Seekers

- After posting the job, the system allows job seekers to submit their applications.
- The recruiter receives applications, which may include resumes, cover letters, or other relevant documents.

4. Message Candidate

- The recruiter evaluates applications and selects suitable candidates for further communication.
- Communication can be initiated via:
  - **Messages**: Direct messaging within the platform.
  - **Email**: Contacting candidates through their provided email addresses.

5. End of Process

- The process concludes once the recruiter has interacted with the candidates.
- Further steps, such as scheduling interviews, would follow as part of the recruitment workflow (not depicted here).

DATA BASE SCHEMA – USERS

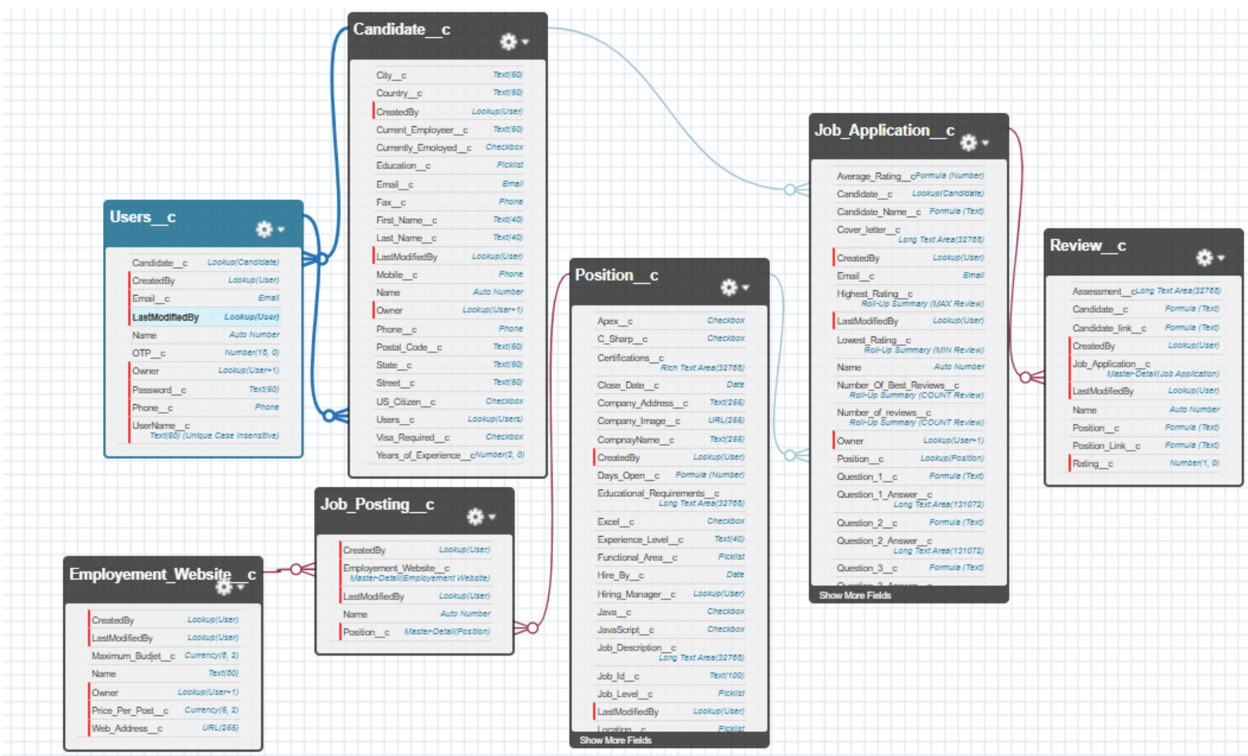


Figure 6: DATA BASE SCHEMA USERS

1. **Users_c**
    - Represents user details.
    - **Key Attributes**:
        - Candidate_c: Lookup to Candidate_c.
        - Email: Text (email field).
        - Phone_c: Text field for phone number.
        - Password_c: Text field (for storing user passwords securely).
        - LastModifiedBy: Lookup(User) to track the user making changes.

2. **Candidate_c**
    - Represents candidate profiles.
    - **Key Attributes**:
        - First_Name_c, Last_Name_c: Candidate's name.
        - Email: Text field for the candidate's email.
        - Country_c: Text field for location.
        - US_Citizen_c: Checkbox to indicate citizenship.

    - Years_of_Experience_c: Number field for professional experience.
  - **Relationships**:
    - Linked to Users_c (via lookup relationship).
    - Related to Job_Application_c (candidate applies for jobs).

3. **Position_c**
  - Represents job positions.
  - **Key Attributes**:
    - CompanyName: Name of the company.
    - Job_Description_c: Long text area for job description.
    - Days_Open_c: Number field to track how long a job has been open.
    - Experience_Level_c: Text to indicate required experience.
    - Close_Date_c: Date field indicating when the job closes.
  - **Relationships**:

        - Connected to Job_Posting_c (job postings originate from positions).
        - Related to Job_Application_c (positions are applied for).

4. **Job_Posting_c**
    - Represents postings of job positions.
    - **Key Attributes**:
        - Employment_Website_c: Master-detail relationship linking to websites where the position is posted.
        - Position_c: Master-detail relationship linking to the job position.

5. **Employment_Website_c**
    - Represents external employment websites.
    - **Key Attributes**:

- Web_Address_c: URL of the employment website.
- Price_Per_Post_c: Currency field for listing cost per job posting.
- Maximum_Budget_c: Currency field indicating the maximum budget for a website.

6. **Job_Application_c**
   - Tracks applications for positions.
   - **Key Attributes**:
     - Candidate_c: Lookup to the Candidate_c table.
     - Position_c: Master-detail to Position_c.
     - Average_Rating_c: Formula field summarizing review scores.
     - Cover_letter_c: Long text area for the application cover letter.
   - **Relationships**:
     - Related to Review_c for feedback/reviews.

7. **Review_c**

- Represents reviews for job applications.
- **Key Attributes**:
  - Rating_c: Numeric field for review score.
  - Assessment_c: Long text area for written assessment.
- **Relationships**:
  - Linked to Job_Application_c (reviews are associated with applications).
  - Related to Position_c and Candidate_c via formulas.

Creating an App and Objects

Overview: Creating an app and custom objects allows you to model your business data in Salesforce. A custom app provides a dedicated workspace for users, while custom objects let you define specific entities like products, orders, or inventory that are central to the business.

- Objective: Build a custom app that holds business-specific objects.

- **Steps:**

  1. Go to Setup > App Manager > New Lightning App.
  2. Fill in the details like app label, API name, and logo.
  3. Add Navigation Items (Objects) to your app.
  4. Save the app and assign it to profiles.

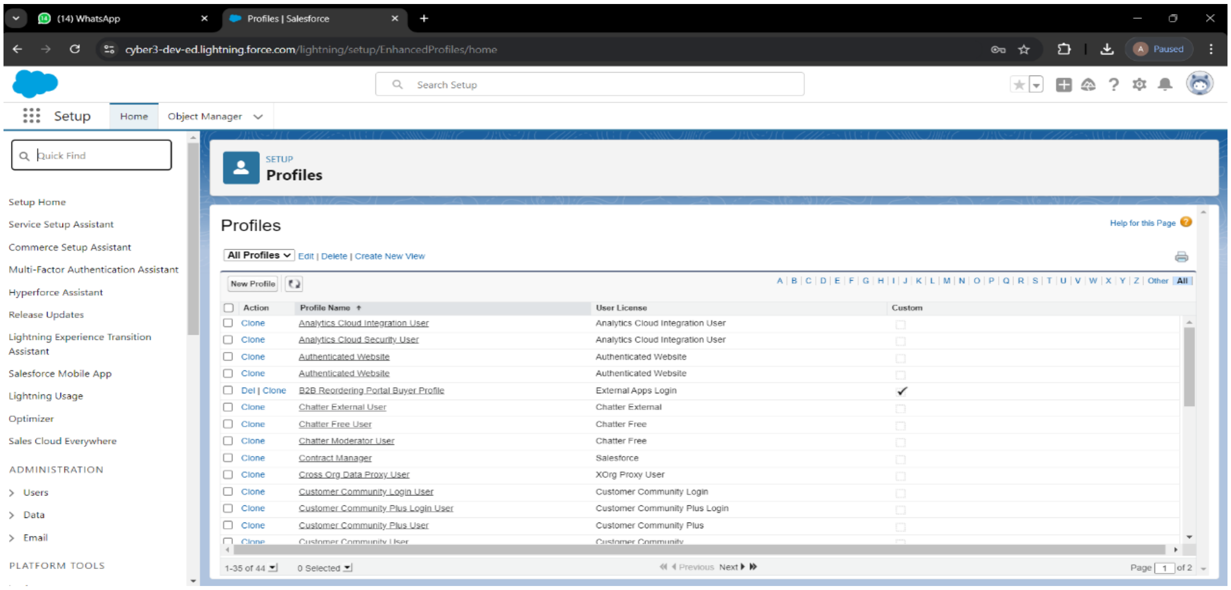


Image 1: Profiles

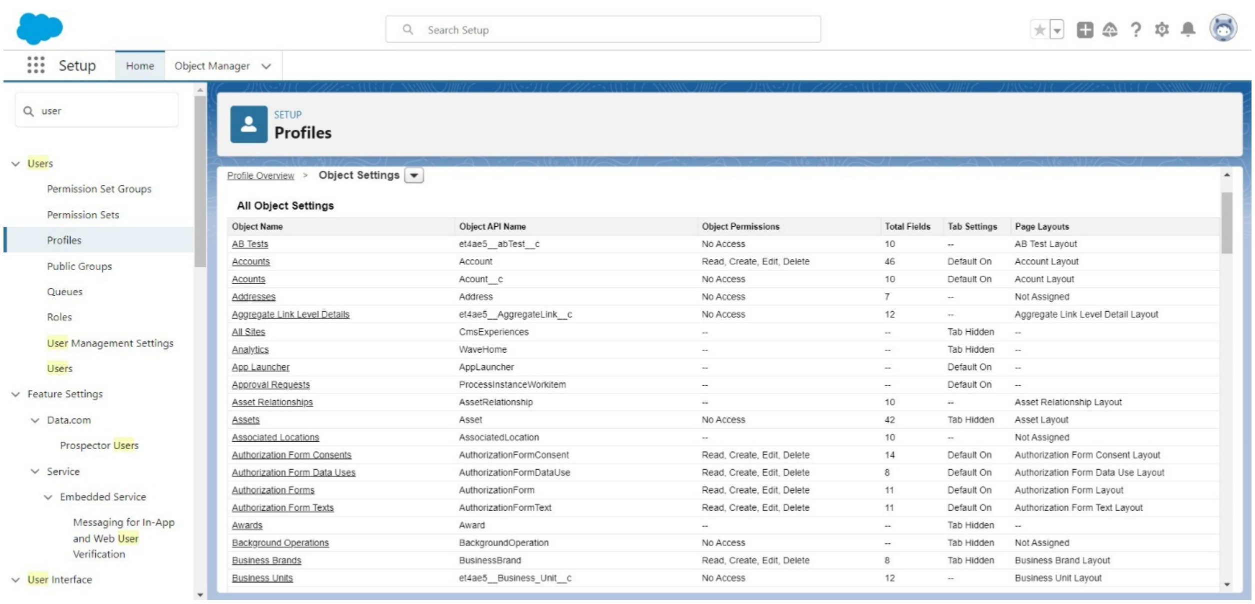


Image 1.1: Profiles

- **Custom Objects Creation:**

    1. Navigate to Object Manager.

2. Click Create > Custom Object.
3. Define the object's label, plural label, and record name.
4. Enable features like Reporting and Field History Tracking.

Creating Fields and Users

Overview: Fields allow you to capture detailed information within objects, while users are the individuals who interact with the system. This step involves adding fields that suit your business data needs and defining users with appropriate access to the system.

- Objective: Add fields to objects and set up new users in Salesforce.
- Steps for Creating Fields:
  1. Open the object in Object Manager.
  2. Navigate to Fields & Relationships > New.
  3. Select the field type (e.g., text, number, etc.) and provide a field label and field name.
  4. Set field-level security and decide which profiles can view/edit this field.

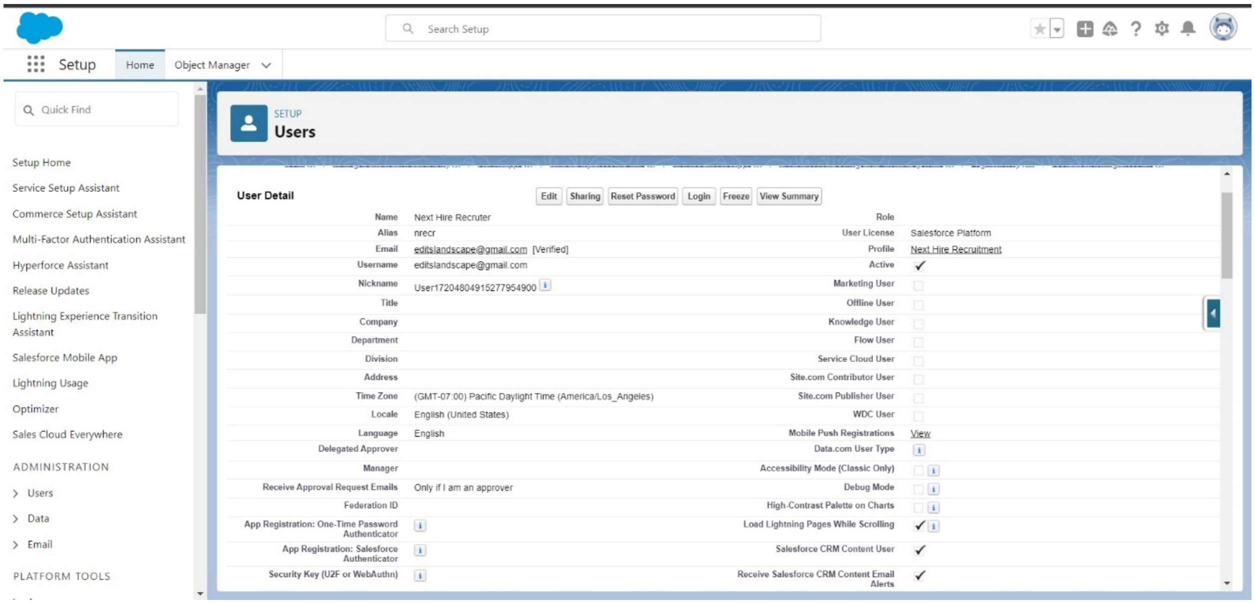
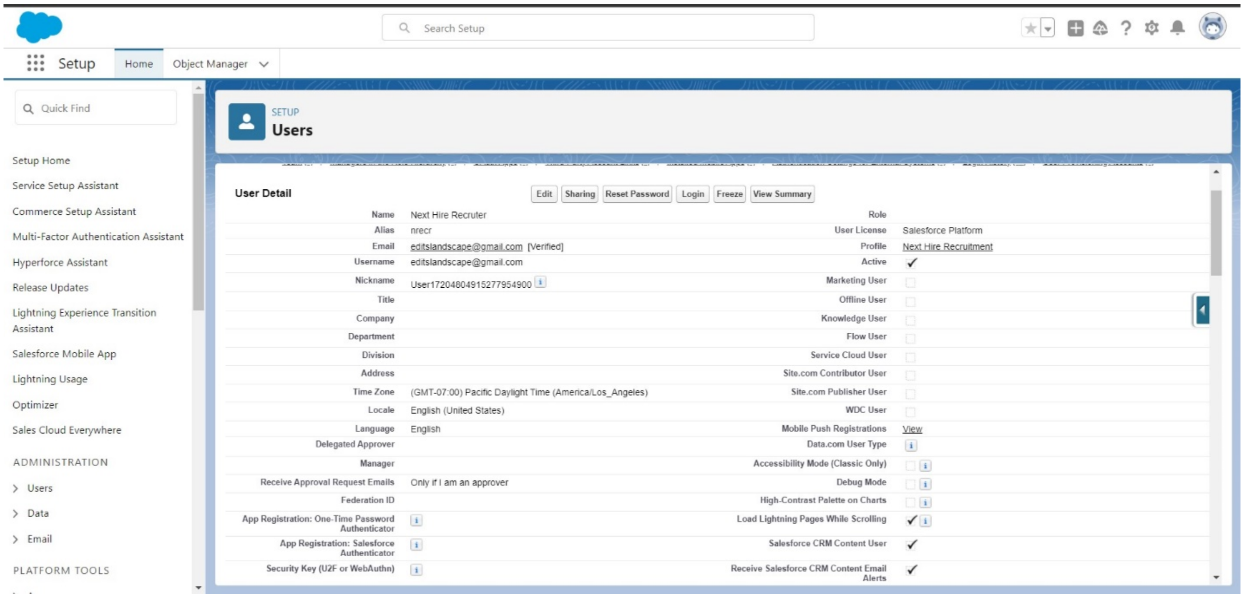


Image 2: Users

- **Steps for Creating Users:**

    1. Go to Setup > Users > New User.
    2. Fill in the required details such as name, email, role, and profile.
    3. Set user permissions and save the user.



Images 2.1: USERS

Creating Profiles

Overview: Profiles determine what a user can access within Salesforce. This step ensures that users only have access to the data and functionalities they need to perform their jobs. You can restrict or grant permissions based on their role within the organization.

- Objective: Control access for users by creating and assigning profiles.
- **Steps**:
    1. Navigate to Setup > Profiles.
    2. Click New Profile or Clone an existing one.
    3. Define object and field-level permissions for the profile.
    4. Assign the profile to the necessary users.

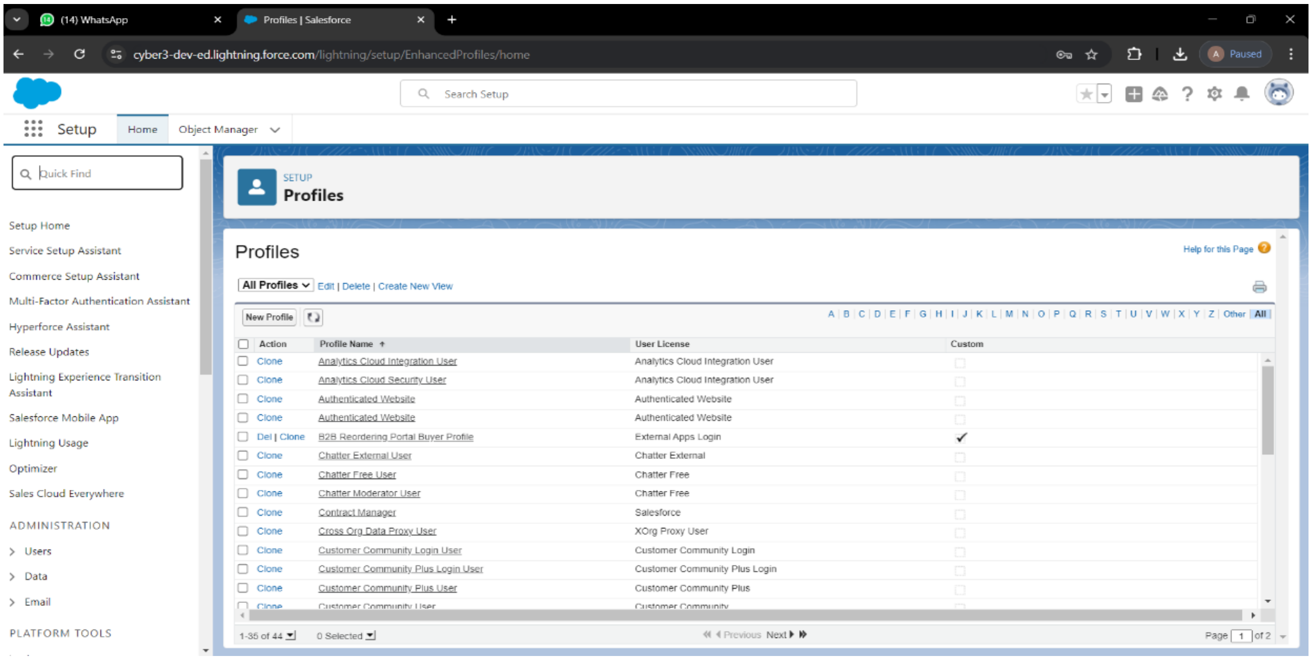


Image 3: Creating profiles

Managing Object-Level and Field-Level Permissions

Overview: This step involves managing what users can do at the object and field level, such as viewing, editing, or deleting records. Object-level permissions apply to the entire object, while field-level permissions provide more granular control over individual fields.

- Objective: Control the accessibility of objects and fields for different profiles or permission sets.
- **Steps for Object-Level Permissions:**

    1. Go to Setup > Profiles or Permission Sets.
    2. Select the profile or permission set.

3. Under Object Settings, find the desired object and level.

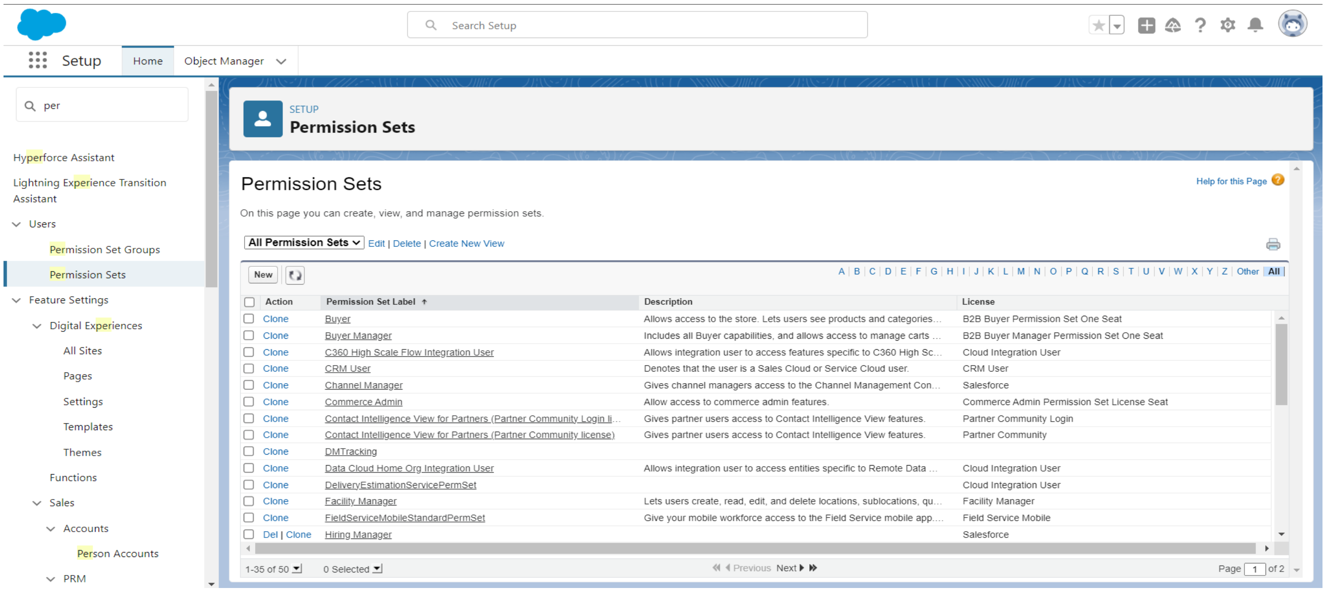


Image 4: Managing object and field

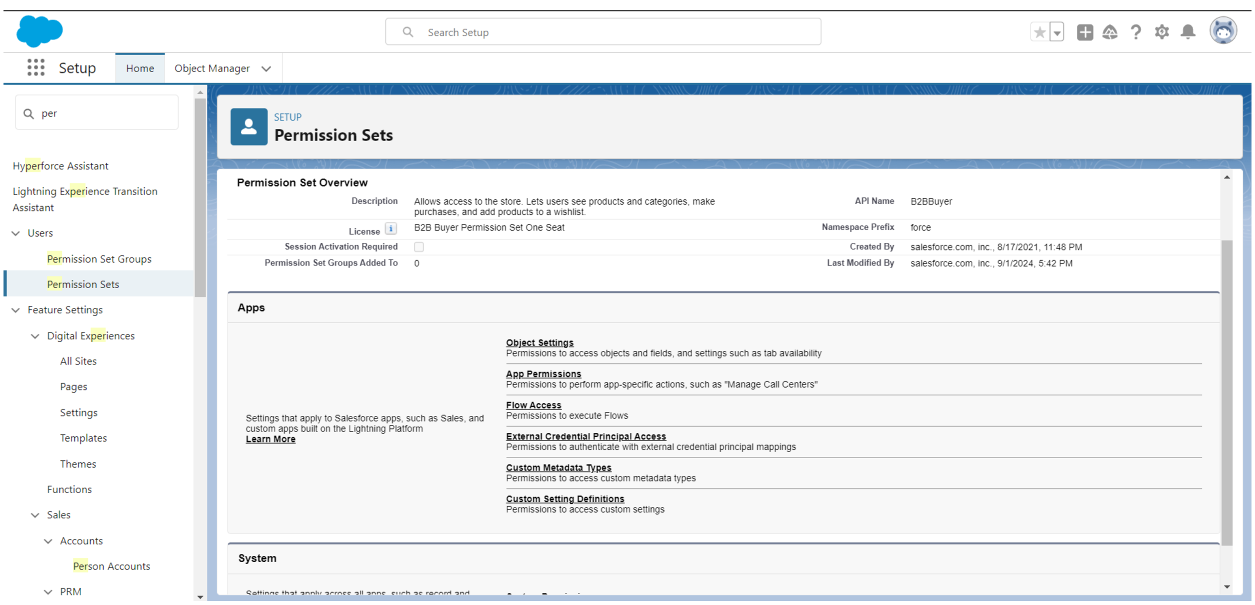


Image 4.1: Managing object and field

Working with Apex Resources and REST Resources

- Objective: Write custom business logic using Apex and expose Salesforce objects and logic as REST APIs.
- **Steps for Apex Development:**
    1. Open Developer Console and create a new Apex class.
    2. Define methods and business logic.
    3. Optionally, trigger the logic via workflows, processes, or directly from custom buttons.
- **Steps for Creating REST Resources:**
    1. Create an Apex class annotated with @RestResource to expose it as a RESTful API.
    2. Define HTTP methods like @HttpGet, @HttpPost, etc., for API actions.
    3. Test API endpoints via tools like Postman.

Creating Flows and Running Flows

- Objective: Automate processes using Salesforce Flow Builder.
- **Steps for Creating Flows:**
  1. Navigate to Setup > Flows.
  2. Select New Flow and choose the appropriate flow type (Screen Flow, Record-Triggered Flow).
  3. Drag and drop elements to define the flow logic (e.g., record updates, decisions).
  4. Activate the flow once completed.

- **Steps for Running Flows**:
  1. Flows can be triggered by user actions, record creation/update, or scheduled events.
  2. You can embed flows on Lightning pages or execute them from buttons

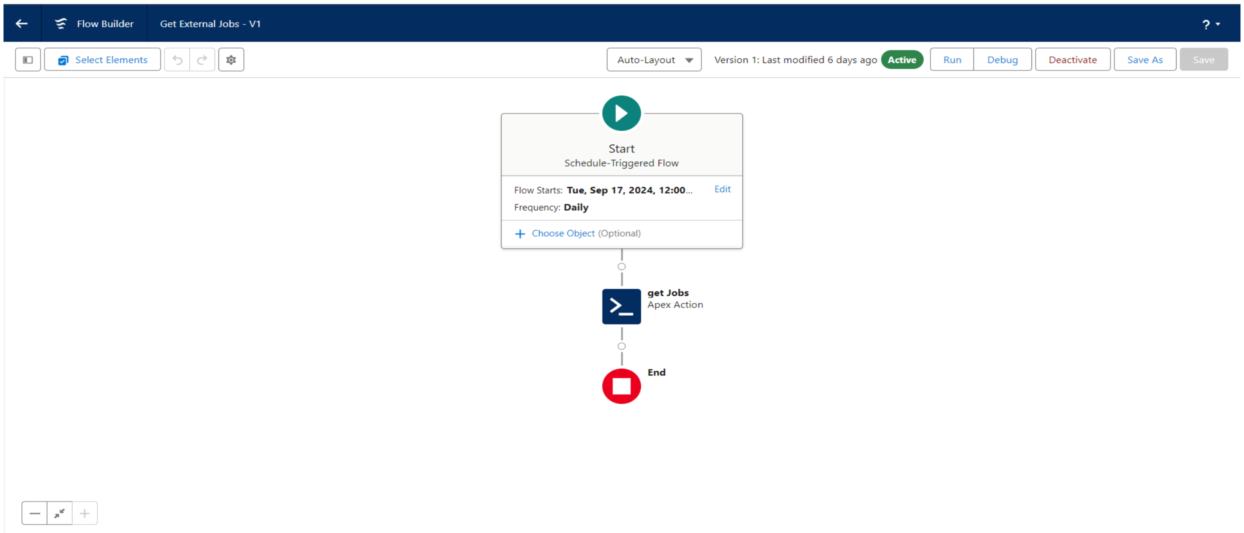


Figure 6: Creating Flows and Running Flows

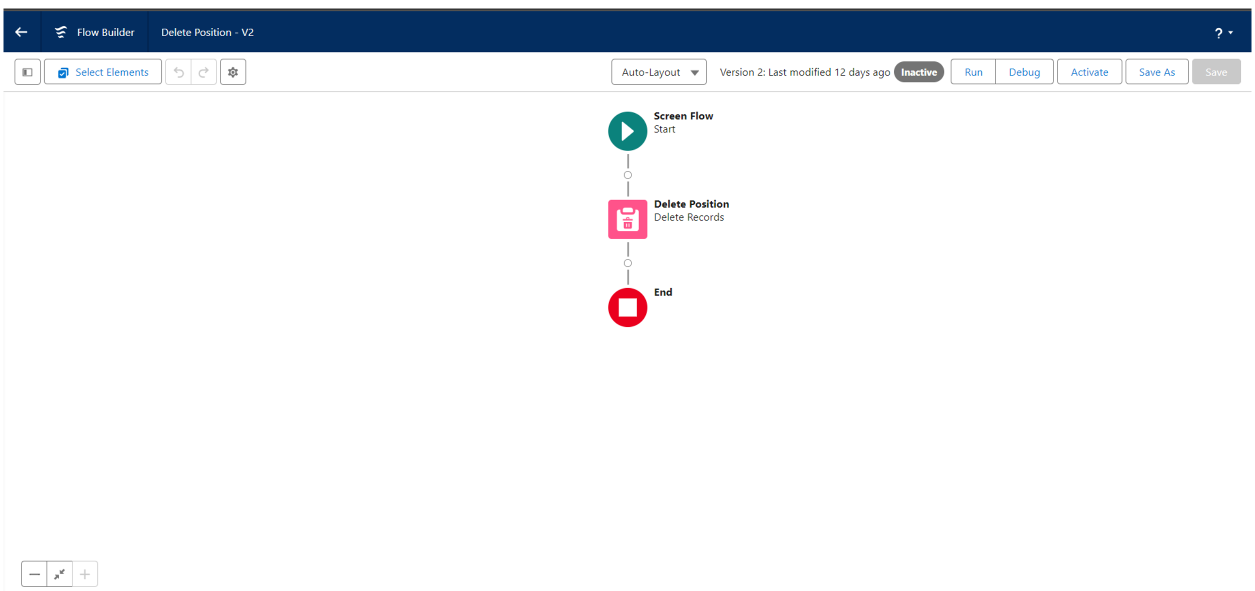


Figure 6.1: creating Flows and running flow

Website

To develop the website, we are using HTML CSS, JavaScript and Bootstrap. We are using Node Js as a backend.

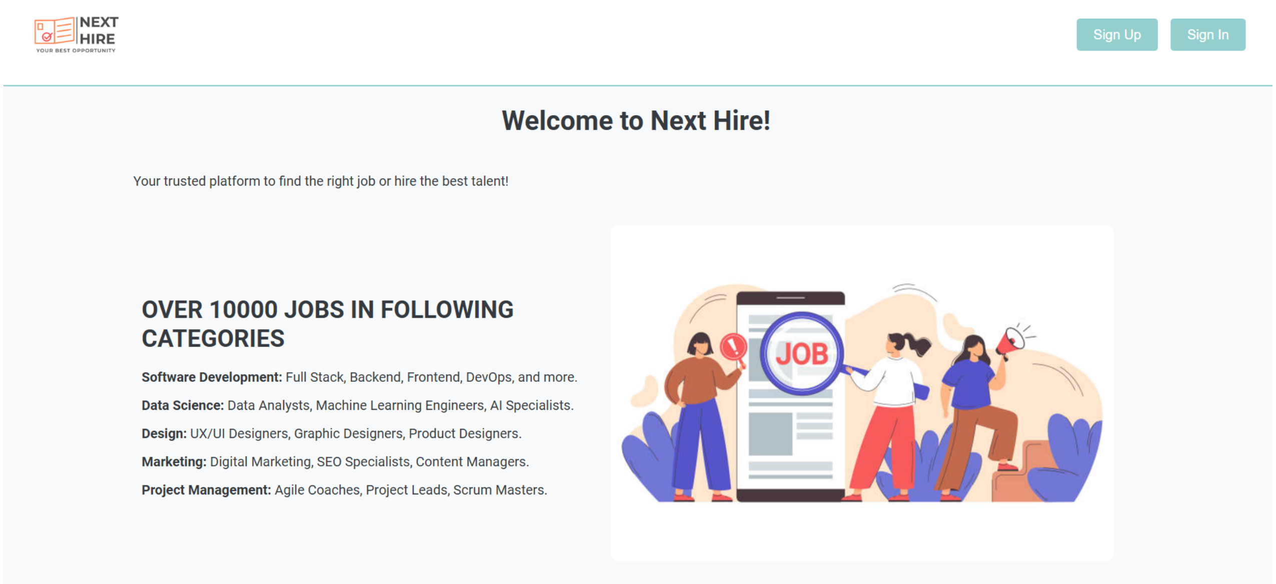


Image 5: Website

Overview:

Next Hire is a trusted platform designed to help both job seekers and employers. It provides a comprehensive list of over 10,000 job listings across various categories and aims to connect talented individuals with the right opportunities.

Key Features:

- **Job Search**: Users can search through thousands of job openings in multiple domains, including software development, data science, design, marketing, and project management.
- **Employer Hiring**: Employers can post job listings and search for skilled candidates.

How to Use:

1. **For Job Seekers**:
    - Navigate to the “Jobs” section.
    - Browse through various job categories.
    - Apply for jobs that match your qualifications.
2. **For Recruiters**:
    - Sign up as a recruiter.
    - Post job openings in relevant categories.
    - Review applications and connect with potential candidates.

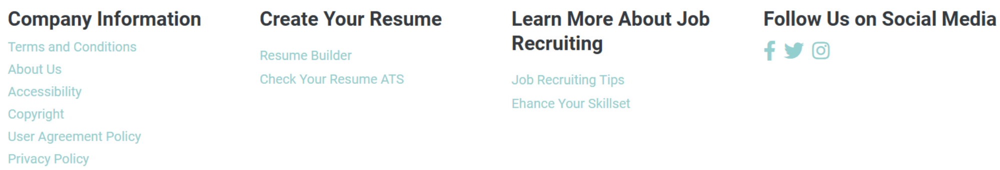


Image 5.1: Terms and conditions

1. Terms and Conditions:

- The link for "Terms and Conditions" is typically created as an anchor element, which redirects users to a separate page that outlines the platform's terms of use. This ensures users are informed of the platform's legal guidelines.
- The link can be styled using CSS to match the platform's branding. Common styling includes link color, text-decoration, and hover effects to make the link more interactive and visually consistent with the rest of the site.
- JavaScript can be used to enhance user experience by dynamically loading the Terms and Conditions page within a modal window or section of the page. This prevents users from navigating away from their current session while still providing them with the necessary legal information.

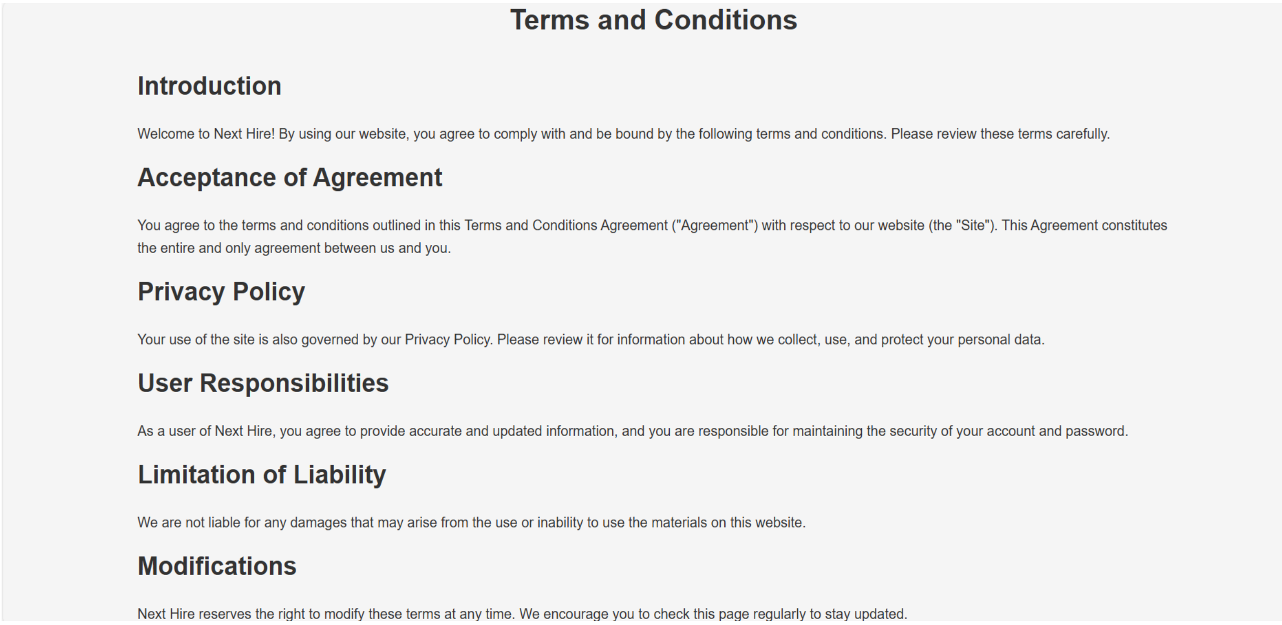
2. About Us:

- The "About Us" link is used to provide users with information about the platform’s mission, team, and services. It is embedded as a clickable link that takes the user to a separate section or page.
- CSS is applied to ensure the link is visually appealing and accessible. For example, the color and size of the text may be adjusted for readability, with hover effects added to indicate that the link is interactive.
- JavaScript can be utilized to load additional details about the company dynamically, such as team biographies or interactive content about the platform’s history. This can be done through AJAX requests or by updating a specific section of the page without needing to refresh it.

3. Accessibility:

- The "Accessibility" link is crucial for users who rely on assistive technologies. It directs users to a page or section where the platform’s accessibility features and policies are explained.
- CSS plays a significant role in ensuring accessibility compliance by maintaining high contrast between text and background colors, larger font sizes for readability, and visible focus indicators for users navigating via keyboard.
- JavaScript can improve accessibility by loading tools or providing feedback forms for users with disabilities. For instance, interactive elements, such as keyboard navigation or screen reader

compatibility options, can be dynamically enhanced to make the platform more accessible.


Terms and Conditions

Introduction

Welcome to Next Hire! By using our website, you agree to comply with and be bound by the following terms and conditions. Please review these terms carefully.

Acceptance of Agreement

You agree to the terms and conditions outlined in this Terms and Conditions Agreement ("Agreement") with respect to our website (the "Site"). This Agreement constitutes the entire and only agreement between us and you.

Privacy Policy

Your use of the site is also governed by our Privacy Policy. Please review it for information about how we collect, use, and protect your personal data.

User Responsibilities

As a user of Next Hire, you agree to provide accurate and updated information, and you are responsible for maintaining the security of your account and password.

Limitation of Liability

We are not liable for any damages that may arise from the use or inability to use the materials on this website.

Modifications

Next Hire reserves the right to modify these terms at any time. We encourage you to check this page regularly to stay updated.

Image 5.2: terms and conditions

This HTML document creates a "Terms and Conditions" page for the "Next Hire!" website. It includes a header with the site logo, navigation buttons, and the page title. The main content section covers various terms, such as "Introduction," "Privacy Policy," and "User Responsibilities," each defined with headings and paragraphs. The footer provides additional navigation links like "Terms and Conditions," "About Us," and "Accessibility."

External files are linked for styling (CSS) and functionality (JavaScript). The CSS file (nexthire.css) controls the page's layout and appearance, while the JavaScript file (terms.js) likely handles button interactivity or other dynamic features. Elements like <header>, <section>, and <footer> structure the page, and attributes such as class and id are used for targeting elements in CSS and JavaScript. This setup allows the page to be visually appealing, functional, and user-friendly.

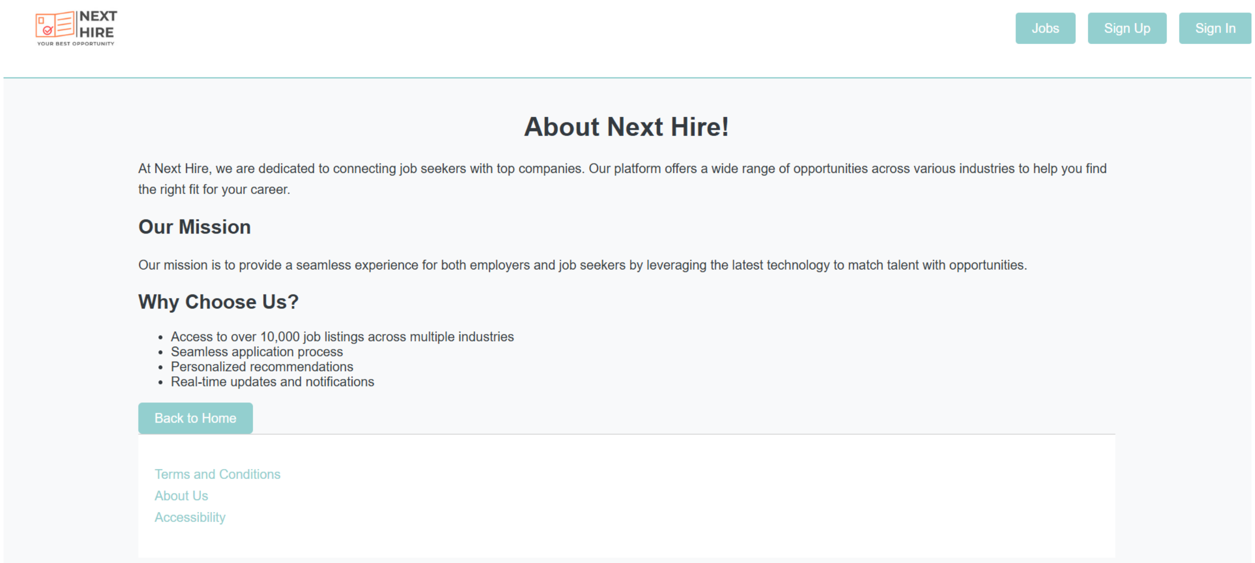


Image 5.3: About US

This HTML document creates an "About Us" page for the "Next Hire!" website. It includes a header with the company logo and navigation buttons for "Jobs," "Sign Up," and "Sign In." The content section describes the company's mission to connect job seekers with top employers using technology.

The page features a section on the platform's mission, emphasizing ease of use and efficiency in matching talent with job opportunities. It highlights key reasons to choose "Next Hire!," such as access to over 10,000 job listings, a seamless application process, personalized recommendations, and real-time updates.

The footer provides links to additional pages, including "Terms and Conditions," "About Us," and "Accessibility." External CSS styles are applied for layout and design, while JavaScript enhances interactivity. This structure ensures a user-friendly experience while offering key information about the platform.

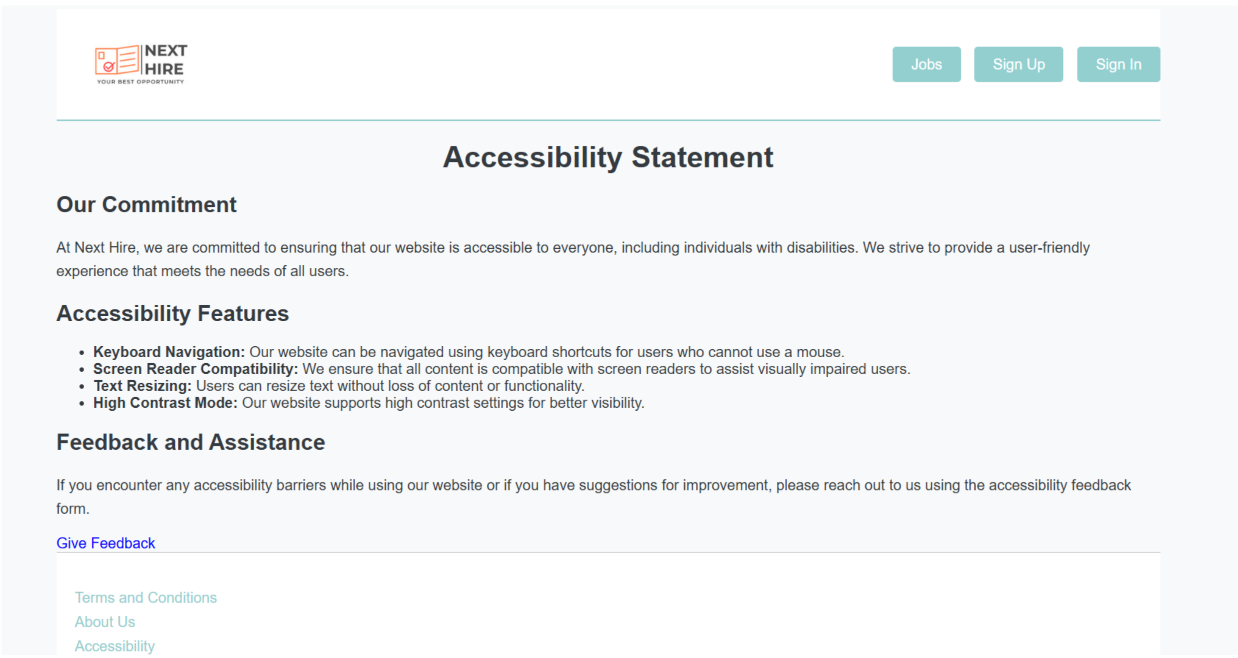


Image 5.4: Accessibility statement

This HTML document creates an "Accessibility Statement" page for the "Next Hire!" website. It includes a header featuring the company logo and navigation buttons for various sections like "Home," "Jobs," "Sign Up," and "Sign In." The main content emphasizes the company's commitment to providing an accessible, user-friendly experience for all users, including individuals with disabilities.

The page outlines key accessibility features: keyboard navigation for those who cannot use a mouse, screen reader compatibility for visually impaired users, text resizing without content loss, and high-contrast mode for better visibility. It also encourages users to provide feedback on accessibility issues via a feedback form.

The footer offers further navigation options to additional pages. External stylesheets and scripts enhance the visual design and interactivity of the page. The document ensures that all users can interact with the website effectively, regardless of their abilities.

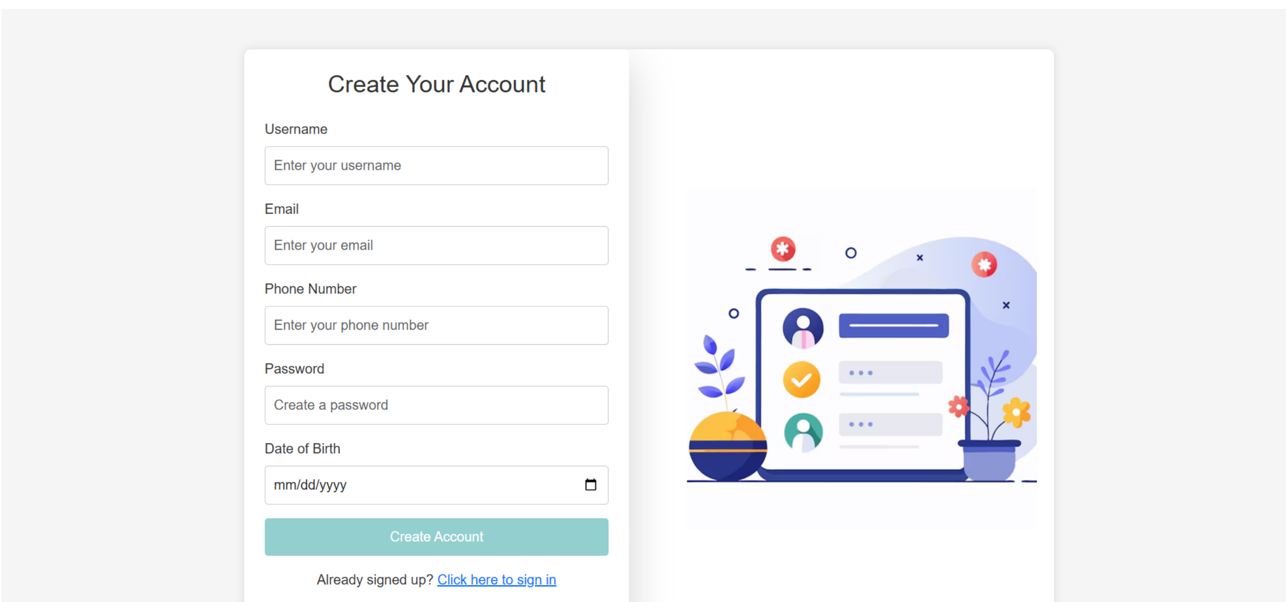


Image 5.5: Create your Account

This HTML document creates a "Sign Up" page for the "Next Hire" website. It includes a simple user registration form, allowing users to enter their first and last name, email, password, and date of birth. The form ensures that all fields are required for submission.

The page design includes a header with a logo and navigation buttons for "Jobs," "Sign Up," and "Sign In." The sign-up form is structured with input fields for user details, and a "Create Account" button submits the form. Upon submission, an alert displays the entered information, provided all fields are completed.

There is also a button to navigate back to the homepage, along with a link for existing users to sign in. JavaScript handles form validation and navigation, offering a simple and user-friendly interface for account creation. The document includes embedded CSS for styling, providing a clean and responsive layout.

Sign Up Flow:

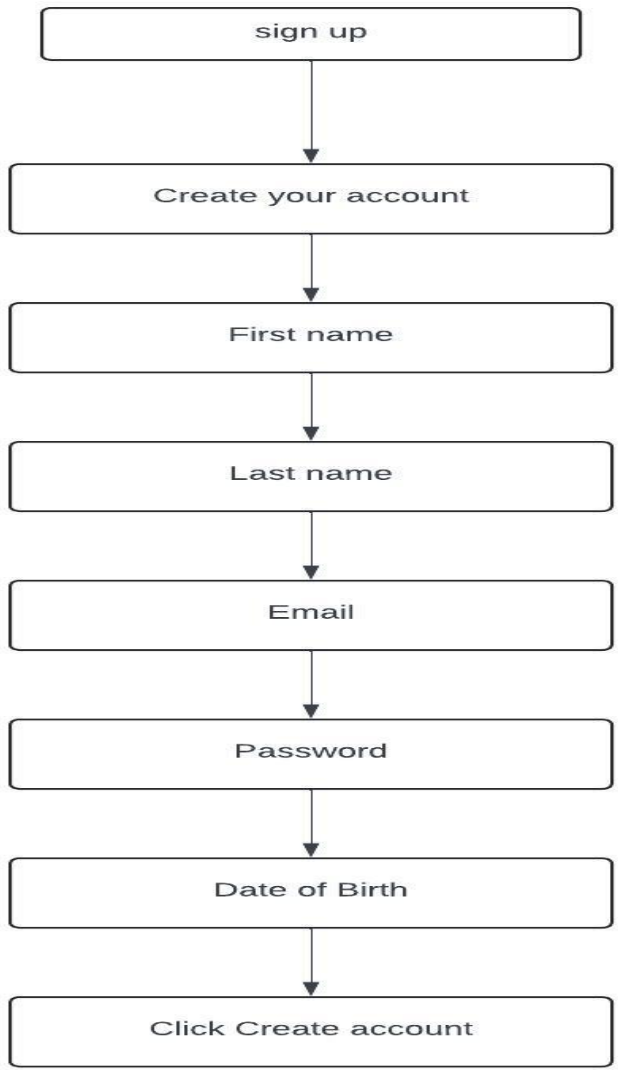


Figure 7: Sign up Flow

Steps in the Sign-Up Flow:

1. **Sign Up**
   - **Description**: The user begins the process by accessing the sign-up page. This page is typically accessible through a "Sign Up" or "Create Account" button.
   - **UI Elements**:
     - A form with fields for user input, such as personal and account information.
2. **Create Your Account**
   - **Description**: The user is prompted to enter the necessary information to create a new account.
   - **Details**:
     - Several fields are required to complete the sign-up process.
3. **First Name**
   - **Description**: The user must provide their first name.
   - **Validation**:
     - This field is mandatory.
     - No numbers or special characters should be allowed.

4. **Last Name**
    - **Description**: The user must provide their last name.
    - **Validation**:
        - This field is mandatory.
        - No numbers or special characters should be allowed.
5. **Email**
    - **Description**: The user must provide a valid email address, which will be used for communication and account verification.
    - **Validation**:
        - The system should validate the email format (e.g., "example@domain.com").
        - The email address should be unique (i.e., not already associated with an existing account).
6. **Password**
    - **Description**: The user must create a password to secure their account.
    - **Validation**:
        - Passwords should have a minimum length requirement (e.g., 8 characters).
        - Passwords should contain a mix of upper and lower-case letters, numbers, and special characters.

        - Provide real-time feedback if the password does not meet security criteria.

7. **Date of Birth**
    - **Description**: The user must provide their date of birth.
    - **Validation**:
        - This field is mandatory.
        - The system should validate that the user is of legal age (based on the platform's policy).

8. **Click Create Account**
    - **Description**: After filling out all the required information, the user must click the "Create Account" button to submit their details.
    - **Action**:
        - Clicking this button submits the data for account creation.
        - If all inputs are valid, the system creates the user account and may send a confirmation email.
        - If any input fails validation, the user will be prompted to correct the errors

Test Cases for Sign-Up Flow

Test Case 1: Verify Navigation to Sign Up Page

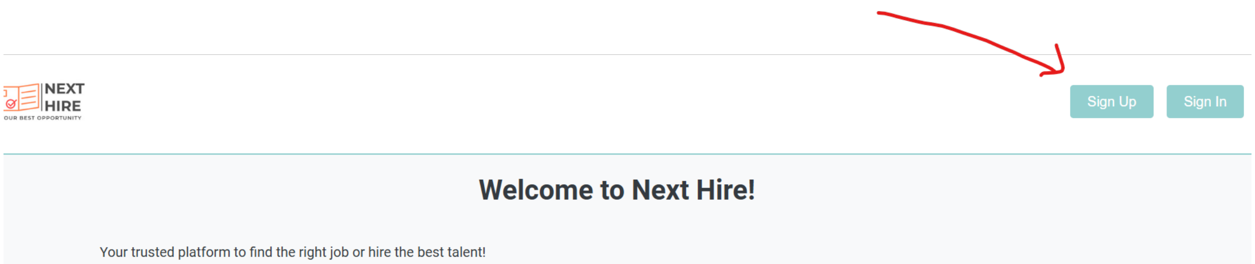


Image 6: test case signs up flow

- **Description:** Verify that the user can navigate to the "Sign Up" page.
- **Preconditions:** None
- **Steps:**
  1. Open the application.
  2. Click on the "Sign Up" button

Test Case 2: Verify username Field

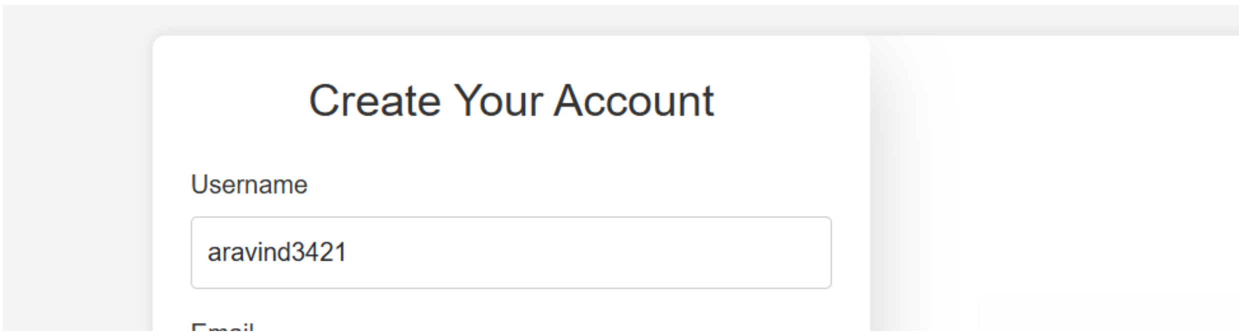


Image 6.1: verify username field

- **Description:** Verify that the user can enter their user name in the " Name" field.
- **Preconditions:** The "Sign Up" page is open.
- **Steps:**
  1. Enter a valid first name in the "First Name" field (e.g., "John").

Test Case 3: Verify Email Field with Valid Email

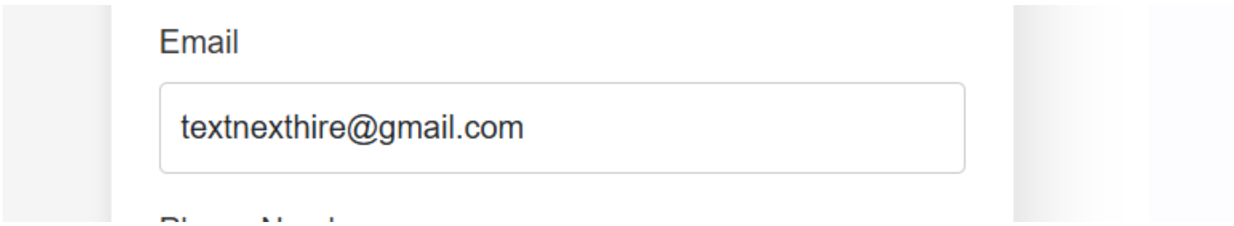


Image 6.2: verify email field with valid email

- **Description:** Verify that the user can enter a valid email address in the "Email" field.
- **Preconditions:** The "Sign Up" page is open.
- **Steps:**
    1. Enter a valid email in the "Email" field (e.g., "john.doe@example.com").

Test Case 4: Verify Email Field with Invalid Email

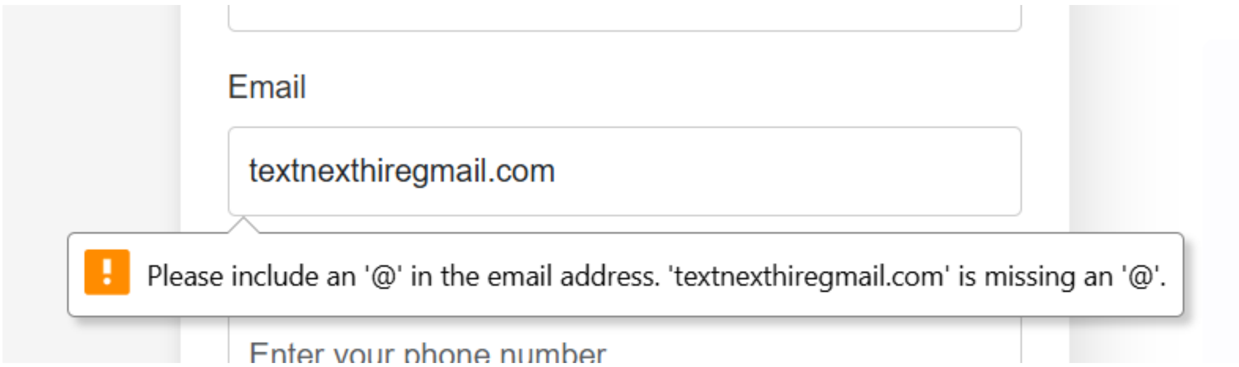


Image 6.3: verify email field with invalid email

- **Description:** Verify that an error message is displayed when an invalid email address is entered.
- **Preconditions:** The "Sign Up" page is open.
- **Steps:**
  1. Enter an invalid email in the "Email" field (e.g., "john.doeexample.com").
  2. Try to proceed to the next step.

Test Case 5: Verify phone number

Image 6.4: verify phone number

- **Description:** enter a valid phone number
- **Enter:** ten digit number

Test Case 6: Verify Password Field with Valid Password

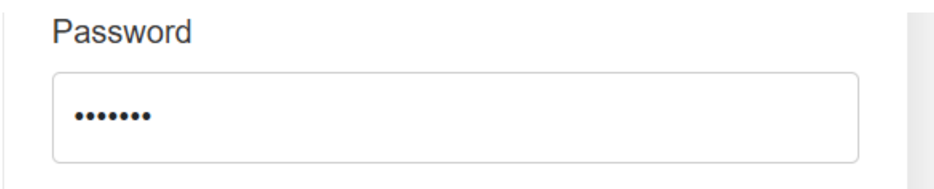


Image 6.5: verify password field

- **Description:** Verify that the user can enter a valid password in the "Password" field.
- **Preconditions:** The "Sign Up" page is open.

- **Steps:**
    1. Enter a valid password in the "Password" field (e.g., "StrongPassword123!").

Test Case 7: Verify Password Field with Weak Password

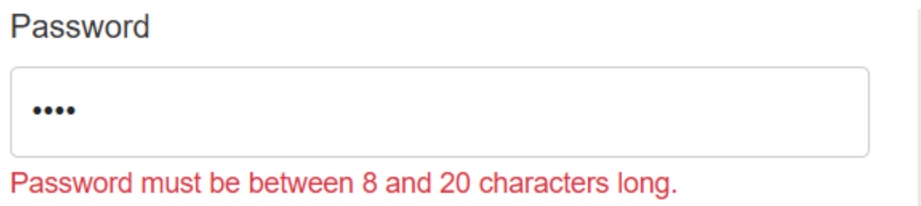


Image 6.6: verify passwords field weak password

- **Description:** Verify that an error message is displayed when a weak password is entered.
- **Preconditions:** The "Sign Up" page is open.
- **Steps:**
    1. Enter a weak password in the "Password" field (e.g., "12345").
    2. Try to proceed to the next step.

Test Case 8: Verify Date of Birth Field with Valid Date

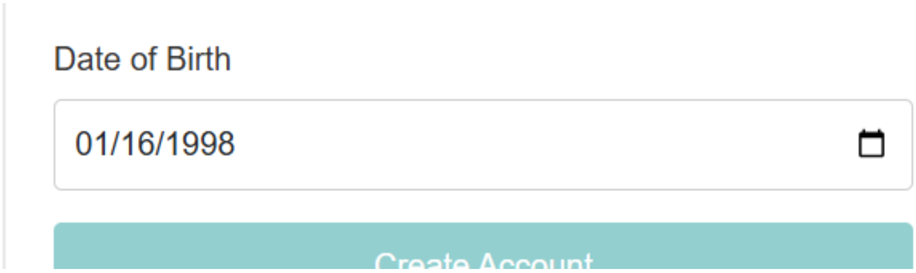


Image 6.7: verify date of birth

- **Description:** Verify that the user can enter a valid date of birth.
- **Preconditions:** The "Sign Up" page is open.
- **Steps:**
    1. Enter a valid date of birth in the "Date of Birth" field (e.g., "01/01/1990").

Test Case 9: Verify Date of Birth Field with Invalid Date

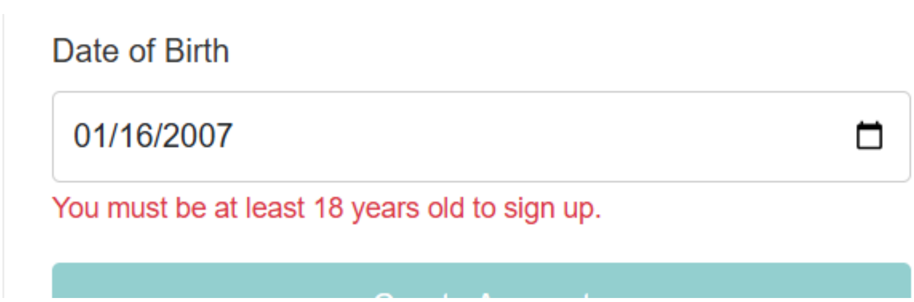


Image 6.8: verify date of birth with invalid date

- **Description:** Verify that an error message is displayed when an invalid date of birth is entered.
- **Preconditions:** The "Sign Up" page is open.
- **Steps:**
    1. Enter an invalid date of birth in the "Date of Birth" field (e.g., "32/13/2000").
    2. Try to proceed to the next step.

Test Case 10: Verify Successful Account Creation

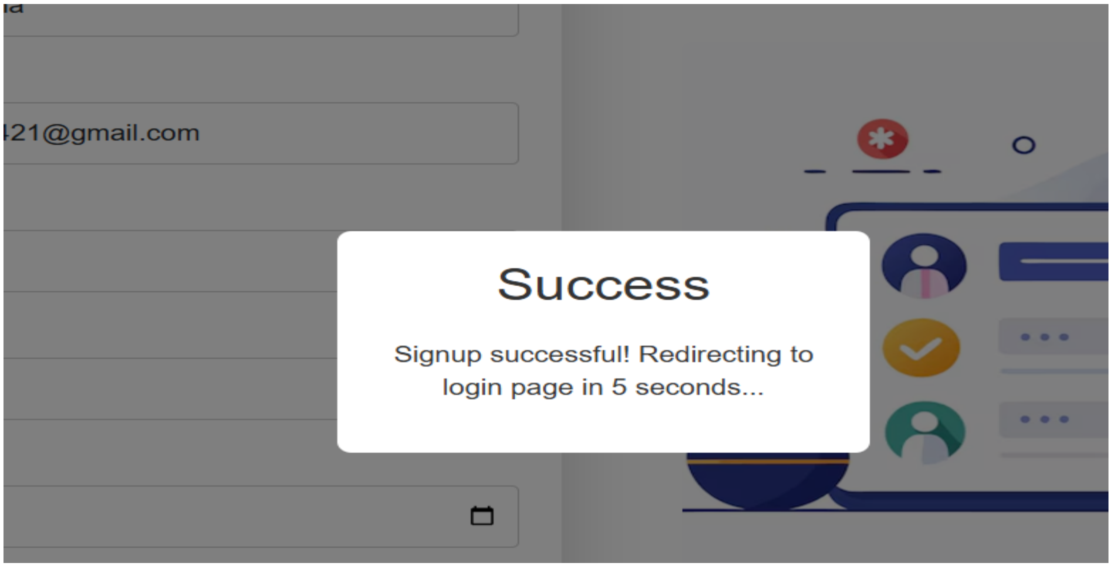


Image 6.9: verify successful Account Creation

- **Description:** Verify that the user can create an account with all valid details.

- **Preconditions:** The "Sign Up" page is open; valid input data is available.
- **Steps:**
    1. Enter a valid first name, last name, email, password, and date of birth.
    2. Click on "Create account."

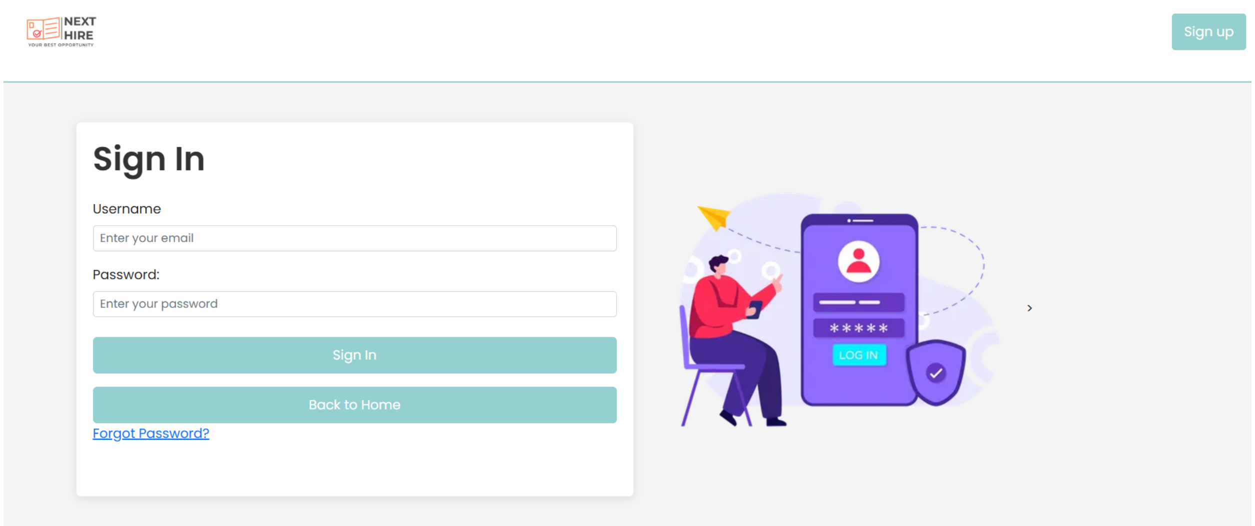


Image 7: Sign in page

This HTML document creates a "Sign In" page for the "Next Hire" website. It includes a form that allows users to log in by entering their email and password. The form has validation requirements: the email field must be filled out with a valid format, and the password must be at least six characters long.

The page includes a header with a logo and navigation buttons for "Jobs," "Sign Up," and "Sign In." Below the sign-in form, there is a "Back to Home" button to return to the homepage.

The layout is clean and minimal, using embedded CSS to style the form elements. Error messages for incorrect inputs are provided, guiding users to correct any mistakes before submitting the form. JavaScript is linked for handling form validation and user interactions.

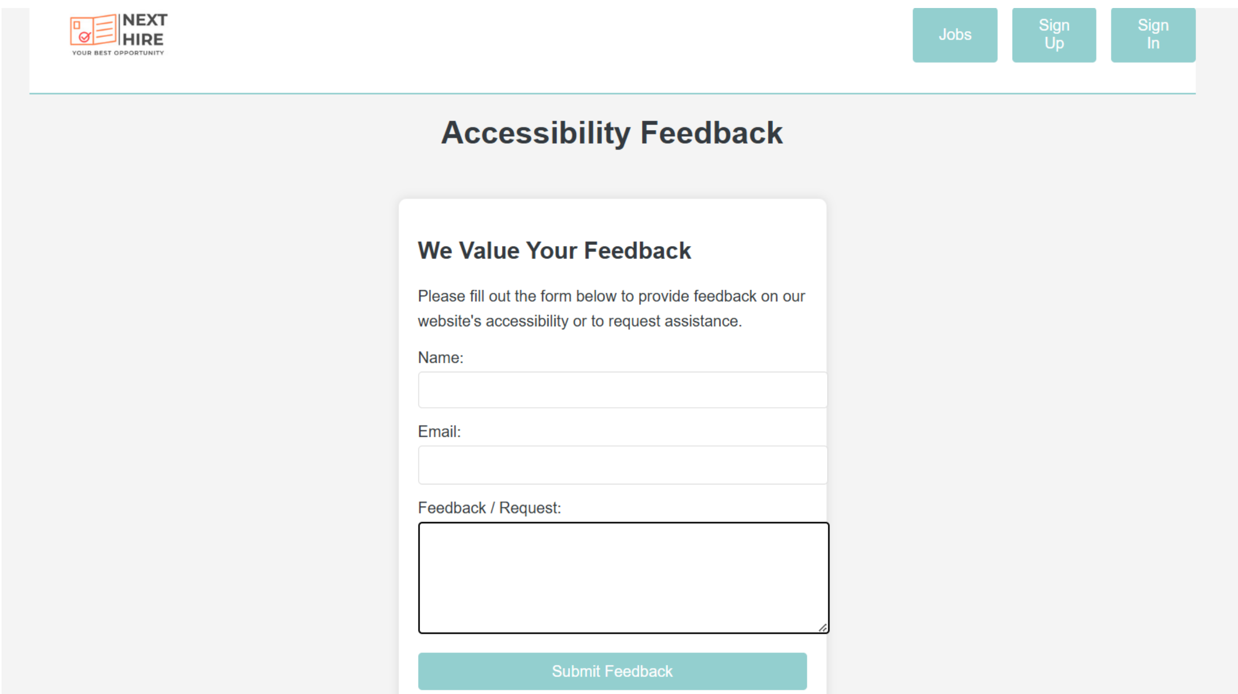


Image 7.1: Accessibility Feedback

This HTML document outlines the "Accessibility Feedback" page for the "Next Hire" website. It emphasizes the importance of user feedback regarding website accessibility and assistance requests. The page includes a header featuring the company logo and navigation buttons for "Home," "Jobs," "Sign Up," and "Sign In."

The main section contains a feedback form that requires users to provide their name, email, and specific feedback or requests related to accessibility. All fields in the form are mandatory, ensuring that users give complete information before submission.

At the bottom, a footer includes links to the terms and conditions, about us, and accessibility pages. The document is designed to be user-friendly, with a clean layout to encourage users to share their experiences and suggestions. JavaScript functionality is included for form handling and user interaction.

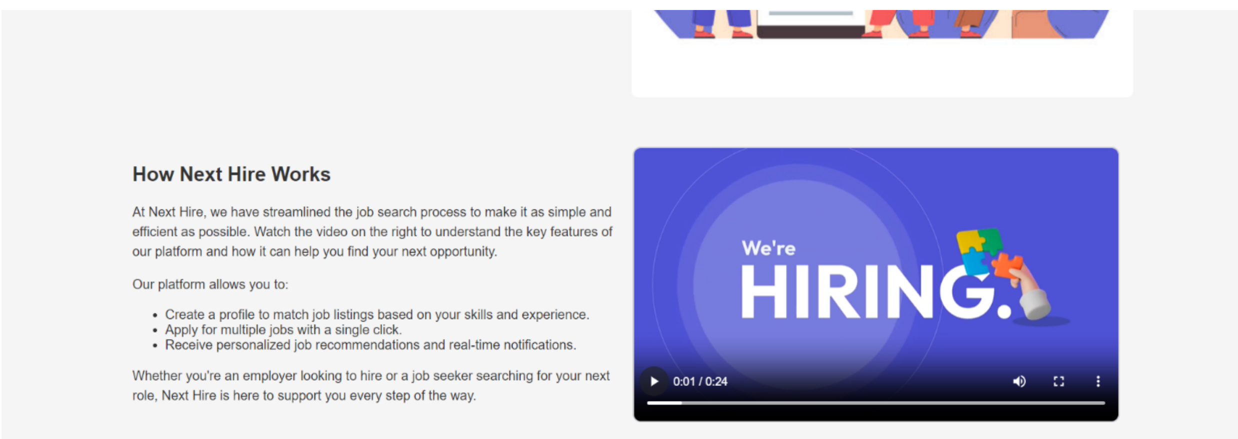


Image 8: text content and video

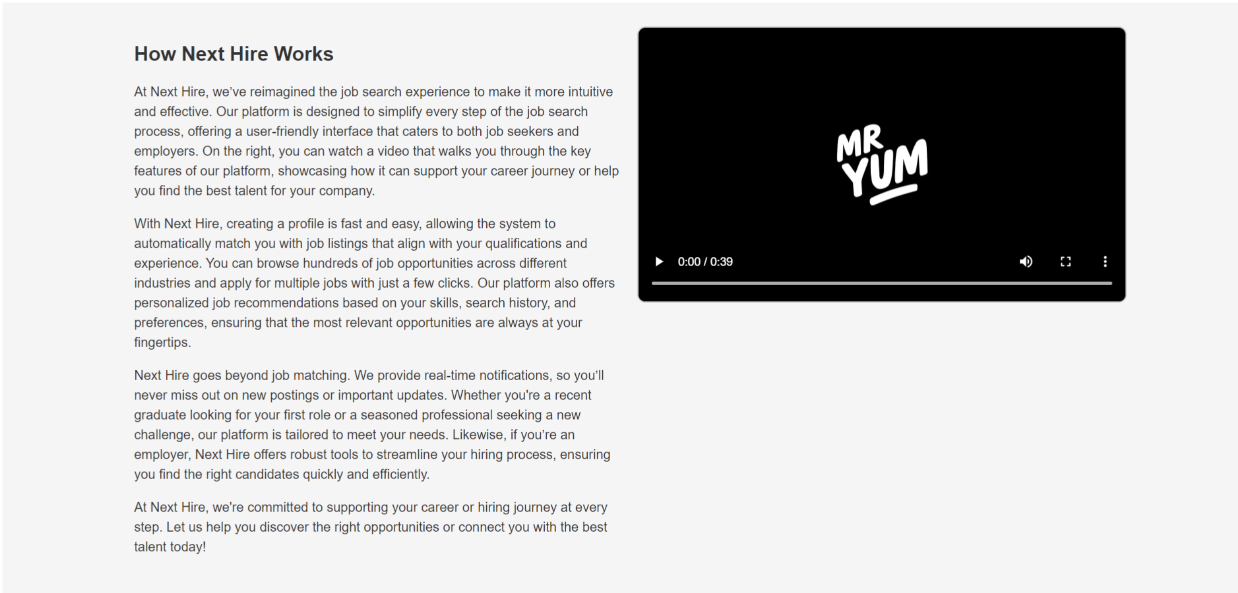


Image 8.1: Text and video

Purpose: Displays the "How Next Hire Works" section with content on the left and a video on the right.

Structure:

- section: Contains both text content and video.
- div.content: Includes a heading (h2), paragraphs, and a list of platform features.
- div.video-container: Hosts the embedded video with controls.

Animations: data-aos attributes are used for fade and zoom animations to make the content and video appear dynamically as the user scrolls.

Layout:

- .video-section uses Flexbox for side-by-side alignment of the text and video.

- content and video-container are both set to 50% width, ensuring that the content and video occupy equal space

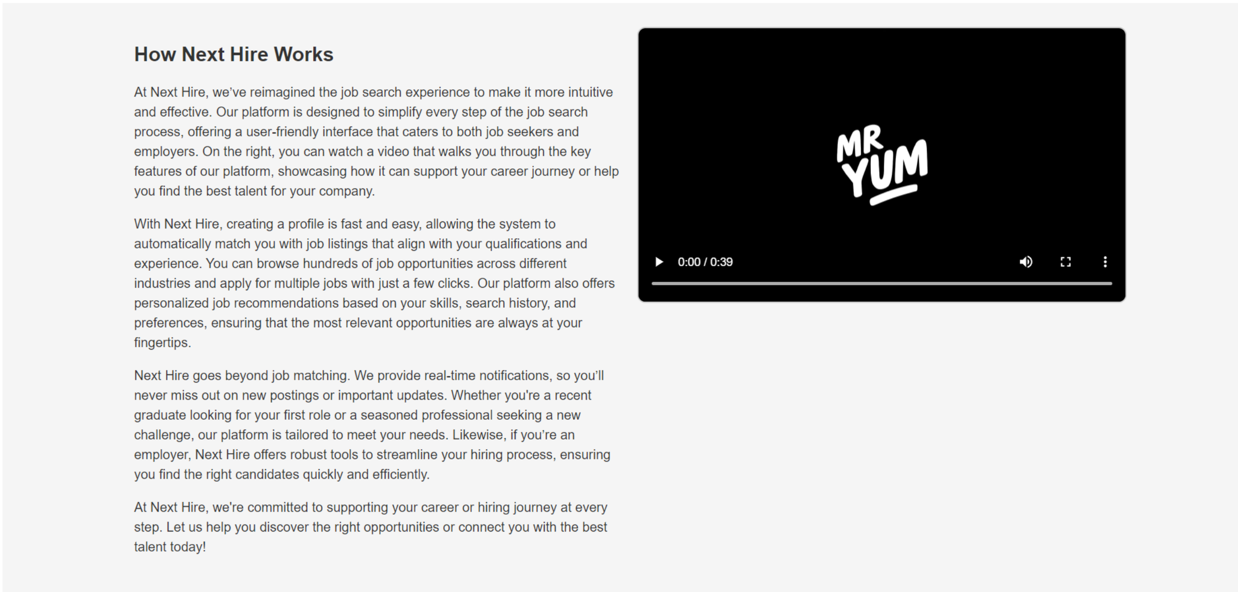


Image 8.2: how next hire works

Styling:

- Padding and margins provide spacing between the content and video.
- The video is styled with a solid border and rounded corners for an enhanced visual appearance.

Functionality:

- Adds an event listener to the **Back to Home** button, redirecting the user to the homepage when clicked.
- Logs a message to the console when the video begins to play for potential debugging or tracking user engagement

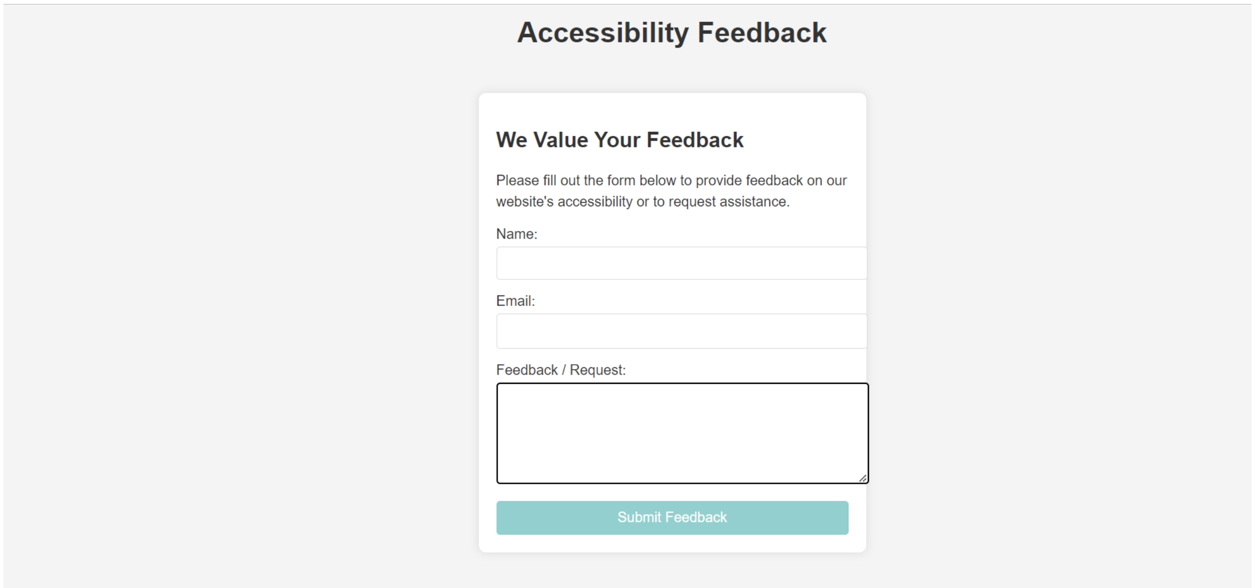


Image 9: Accessibility Feedback test case

Purpose of the Form Update:

The updated form is designed to collect user feedback regarding the accessibility of the Next Hire website, ensuring that users with accessibility concerns can easily provide input or request assistance.

Updates Implemented:

- **Form Layout**:
  The form is structured to be simple and intuitive. It consists of three input fields:

  1. **Name** (text input)

2. **Email** (email input)
3. **Feedback/Request** (textarea for detailed input)

- Each field includes descriptive labels for clarity.
- **Styling Improvements**:
  The form is styled for improved readability and user experience:
  - A **clean, modern look** is achieved using padding, border-radius, and box-shadow for the form container.
  - The form inputs and text area have a **consistent design** with padding and borders.

  A **responsive design** is ensured by setting the form's max width to 400px and centering it on the page.

User interface redesign:

- **Navigation Bar**: Redesigned the header section with clear, clickable buttons for “Jobs,” “Sign Up,” and “Sign In,” making it easy for users to access key pages. Hover effects enhance interactivity.
- **Job Categories Section**: Organized the job categories in a neatly structured list, providing a clear overview of the available opportunities. This helps users quickly identify relevant job categories.
- **Animations**: Integrated **AOS (Animate on Scroll)** library for smooth transitions and visual appeal. Elements like images, descriptions, and headings now fade in or zoom during scrolling, making the browsing experience more engaging.

- **Video Section**: Improved video responsiveness with a clean border and rounded corners. Videos explaining how the platform works now load efficiently and are seamlessly integrated into the layout.
- **Footer and Forms**: Added a clear footer with links to important sections like "Terms and Conditions" and "About Us." Sign-up and sign-in forms have enhanced validation, guiding users through the process while providing feedback for incorrect inputs.
- **Improved Readability**: Increased font sizes and padding across sections, making text easier to read, and used contrasting colors for better visibility.

Login Flow:

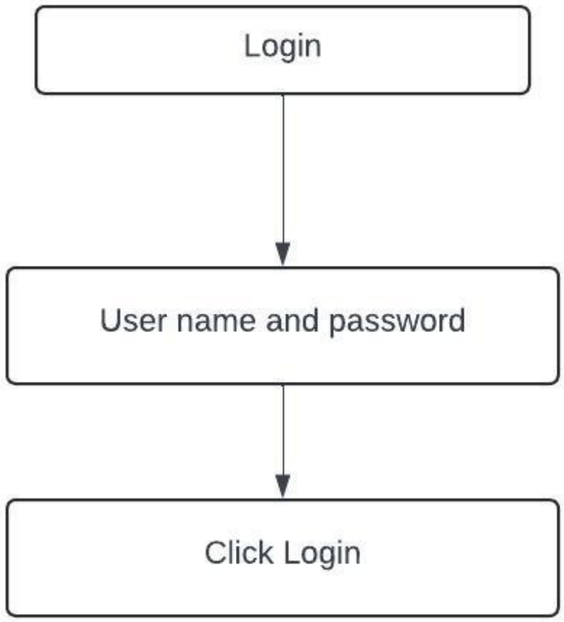


Figure 8: Login Flow

Steps in the Login Flow:

1. **Login Page**
    - **Description**: This is the initial page where users are prompted to log in to the system. The page contains fields for the username and password.
    - **UI Elements**:
        - Username input field.

        - Password input field.
        - "Login" button to submit credentials.

2. **User Inputs Credentials**
    - **Description**: The user is required to enter their unique username and corresponding password.
    - **Details**:
        - The username field is used to identify the user.
        - The password field ensures that the user's credentials match the records in the system.
    - **Validation**:
        - Both fields should be mandatory.
        - Basic validation includes checking the format of the username and ensuring that the password meets security requirements (e.g., minimum character length, inclusion of special characters, etc.).
3. **Click Login**
    - **Description**: Once the user has entered their credentials, they need to click the "Login" button to submit the information for authentication.

- **Action**:
  - Clicking the "Login" button sends a request to the authentication server to validate the entered credentials.
  - If successful, the user is redirected to the dashboard or the main area of the application.
  - If unsuccessful, an error message is displayed, prompting the user to re-enter the correct credentials.

Test case : Login Flow

Test case Steps:

1. **Navigate to the Login Page:**

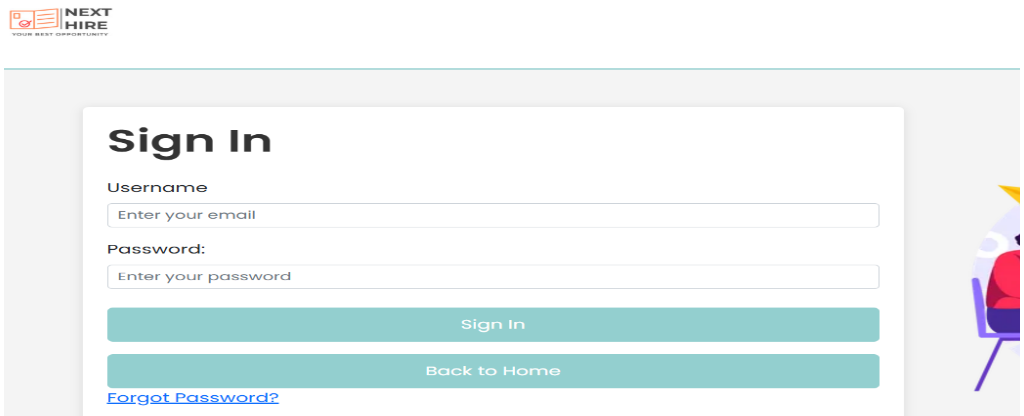


Image 10: login page

- **Action:** Open the application and click on the "Login" button or link.

2. **Login with Valid Credentials:**

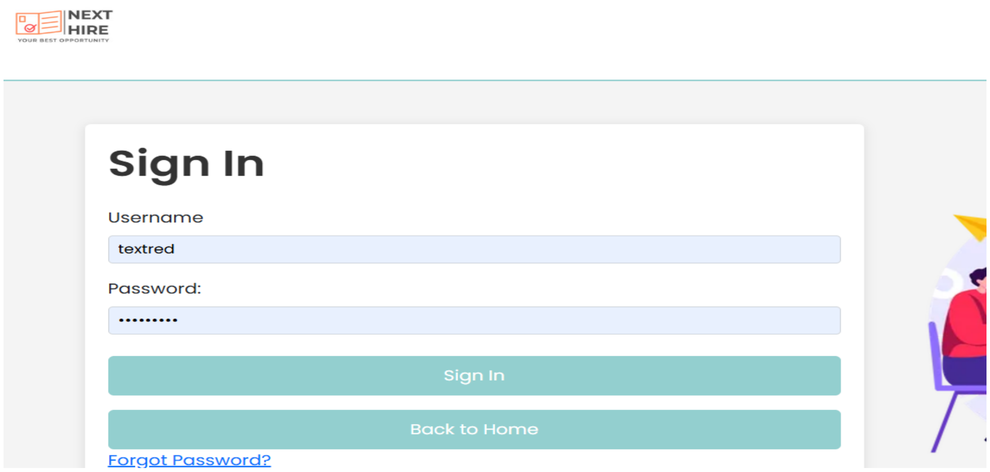


Image 10.1: Valid Credentials

- **Action:** Enter a valid username and password, then click the "Login" button.

3. **Login with Invalid Username:**

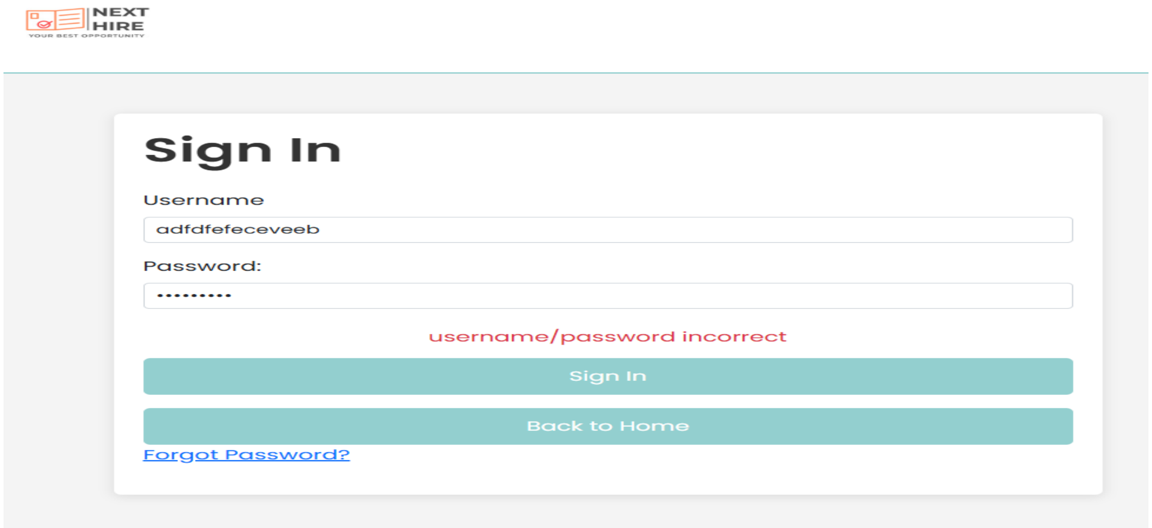


Image 10.2: Invalid Username

- **Action:** Enter an invalid username and a valid password, then click the "Login" button.

4. **Login with Invalid Password:**

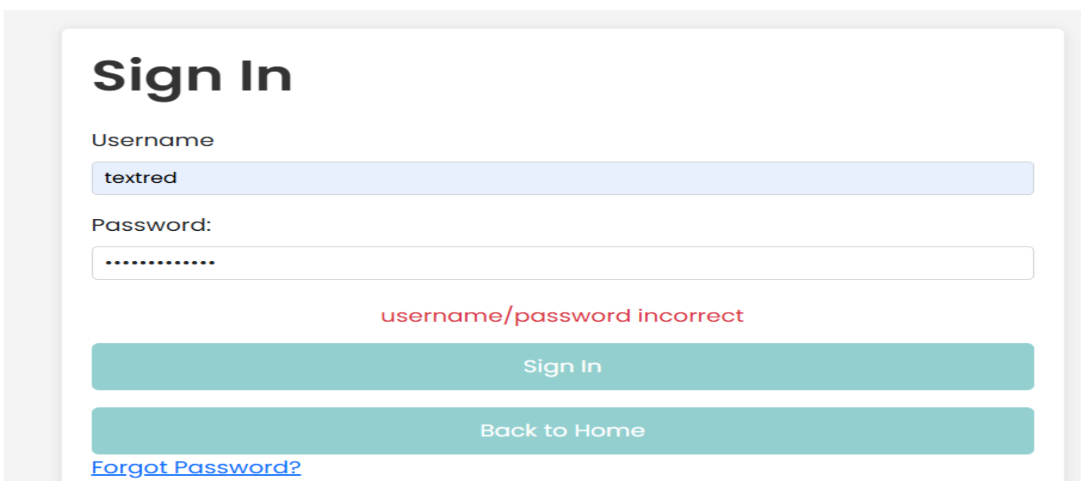


Image 10.3: Invalid Password

- **Action:** Enter a valid username and an invalid password, then click the "Login" button.

5. **Login with Both Invalid Username and Password:**

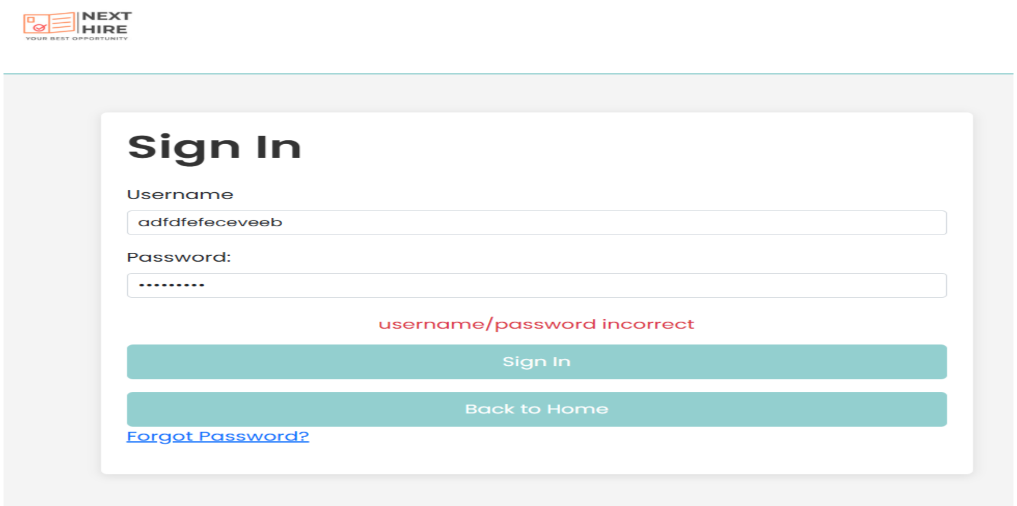


Image 10.4: Invalid Username and Password

- **Action:** Enter both an invalid username and an invalid password, then click the "Login" button.

6. **Login with Empty Username Field:**

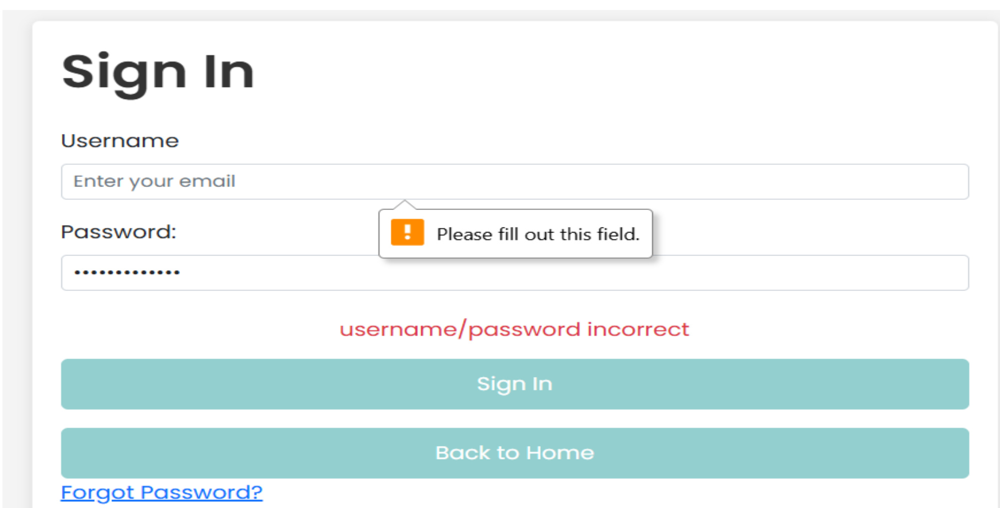


Image 10.5: Empty Username Field

- **Action:** Leave the "Username" field empty, enter a valid password, and click the "Login" button.

7. **Login with Empty Password Field:**

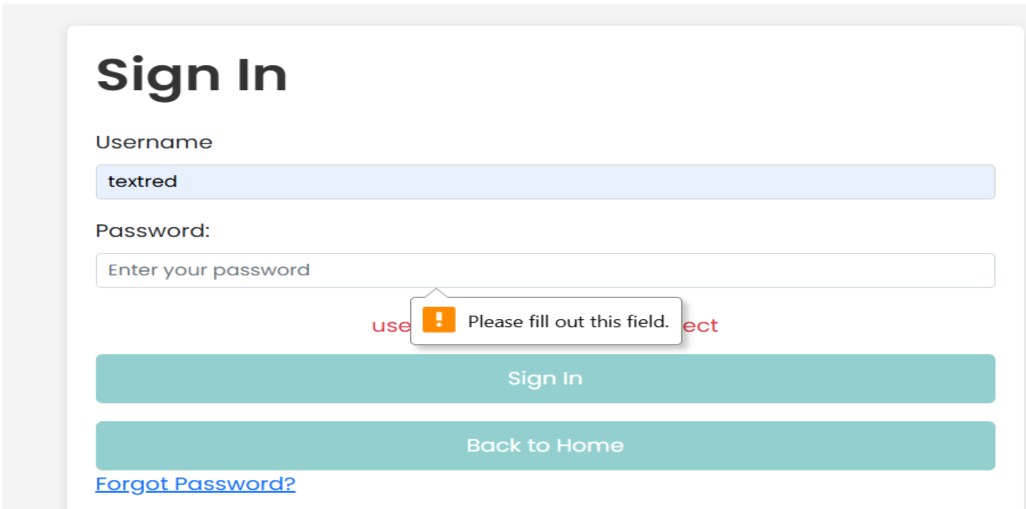


Image 10.6: Empty Password Field

- **Action:** Enter a valid username, leave the "Password" field empty, and click the "Login" button.

8. **Login with Both Empty Username and Password Fields:**

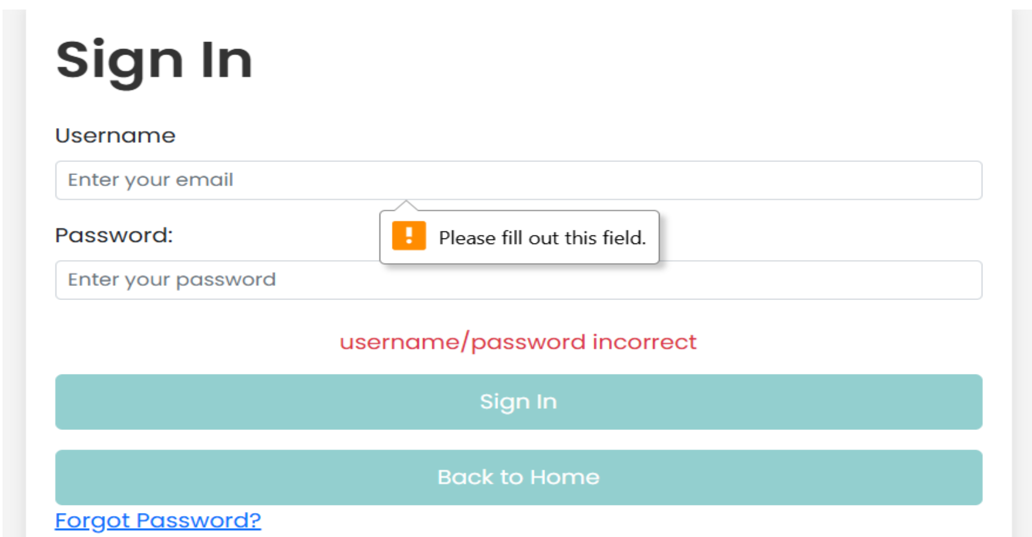


Image 10.7: Both Empty Username & Password Fields

- **Action:** Leave both the "Username" and "Password" fields empty, then click the "Login" button.

Job Details

1. Job Details Page

Job Details and Fetching Questions

- **User Interaction**:
  - On the **Job Details** page, users see details about a specific job, such as job title, description, required skills, salary, and location. This data is fetched from Salesforce using a REST resource.
  - When the user clicks the **Apply** button, the system checks for additional recruiter-posted questions. These questions, if present, are designed to gather more specific insights from applicants.
- **REST Resource (getJobDetails)**:
  - The **getJobDetails** REST resource fetches detailed job information from Salesforce. It has been enhanced to return recruiter questions along with general job details.
  - The questions are stored as custom fields in Salesforce (e.g., Question_1__c, Question_2__c, etc.). These fields are dynamically generated, meaning they can vary between job postings.

- **Dynamic Handling of Questions**:
  - The API response includes job-related questions when available, and these are dynamically rendered in the front end as form fields.
  - The front end creates the necessary input fields (text, radio buttons, etc.) based on the questions retrieved. These questions are tailored to the job and provide additional information to help recruiters assess candidates beyond their resumes.

Submitting a Job Application

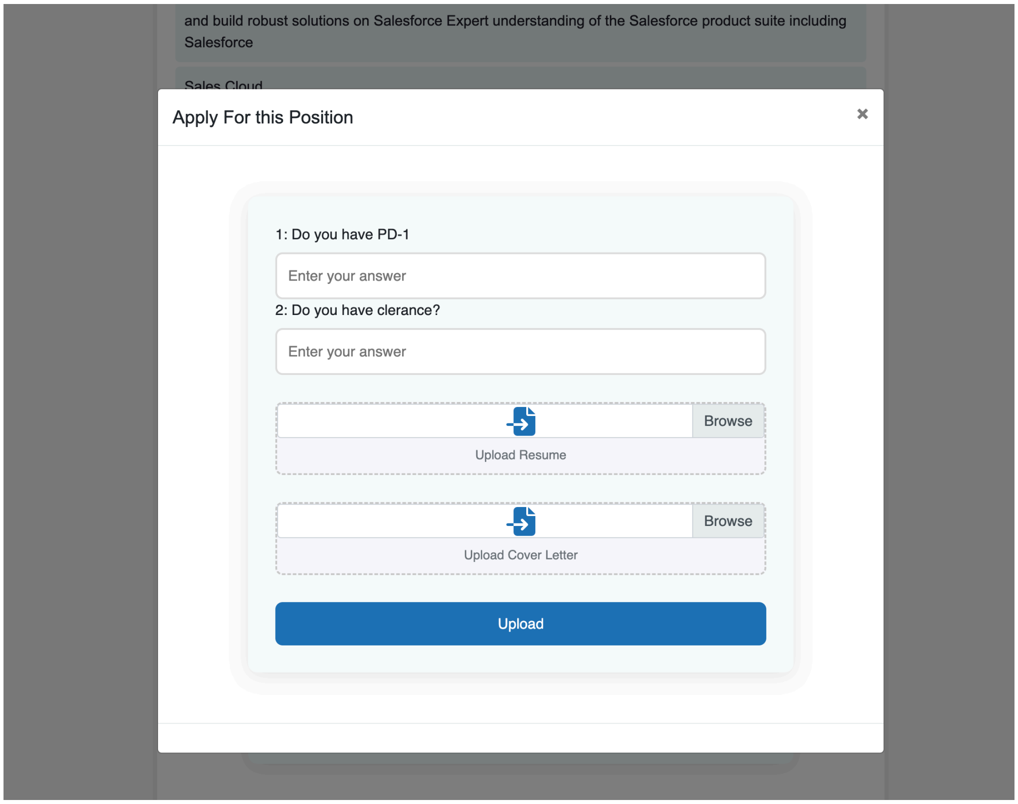


Image 11: Submitting a job application

Submitting a Job Application

- **User Input**:
    - After answering the recruiter's questions, users attach their **resume** and **CV** before submitting the form.
    - The system validates the input to ensure that required fields are filled in and that the attachments are valid (e.g., checking file type and size).

- **Data Submission**:
  - Once the user submits their application, the answers, resume, and CV are sent to the backend. The backend processes the data and communicates with Salesforce to create a new **Job Application** record.
  - The user's answers are mapped to corresponding fields in Salesforce. For example, answers to recruiter questions are saved in fields such as Question_1_Answer__c, Question_2_Answer__c, and so on.
- **Salesforce Integration**:
  - Using **jsforce** (a Salesforce API integration library for Node.js), the backend creates the application record in Salesforce. The record includes:
    - Job applicant's name and contact details.
    - Answers to the recruiter's questions.
    - Attached resume and CV files.
    - The job to which they applied (linked to the **Position** object in Salesforce).

2. Applied Jobs Page

Fetching Applied Jobs Data

- **REST Resource (Applied Jobs)**:

    - A custom **Applied Jobs** REST resource was created in Salesforce to fetch information about jobs to which the user has applied. This data includes:
        - **Job Name**: The title of the job (e.g., Software Engineer).
        - **Position Type**: The nature of the position (e.g., Full-time, Part-time, Contract).
        - **Status of Application**: The current status of the user's job application (e.g., Pending, Reviewed, Accepted, Rejected).
        - **Location**: The location of the job posting.
        - **Company Logo**: The logo of the company offering the job.
- **Data Format**:
    - The data retrieved from the **Applied Jobs** REST resource is passed to the front end, which uses it to display a list of jobs the user has applied for.
    - Each job is displayed in a **tile-based UI**, designed to present the information in a visually organized and user-friendly manner.

Tile-Based Display of Applied Jobs

- **UI Layout**:

- The job details (job name, status, position type, location, and company logo) are displayed in individual tiles on the **Applied Jobs** page.
- The **tiles** provide an overview of the application history and status, making it easy for users to track which jobs they've applied for and where their applications stand in the review process.

Job Application Flow:

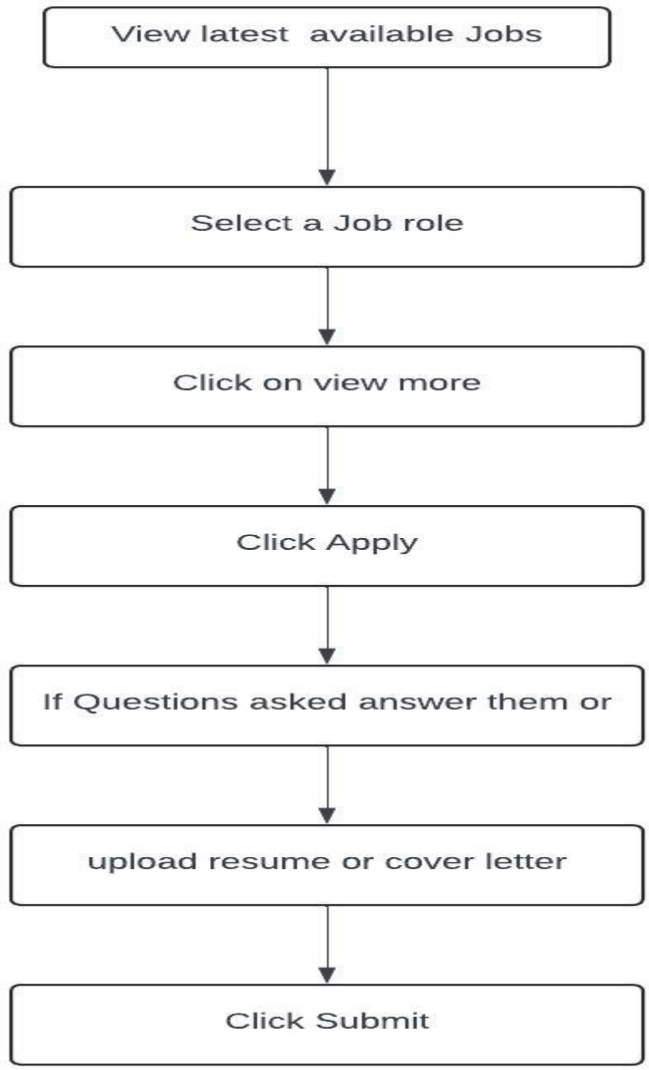


Figure 9: Job application Flow

Steps in the Job Application Flow:

1. **View Latest Available Jobs**
   - **Description**: The user begins by viewing the list of current job openings. This list can be accessed from the jobs page.
   - **UI Elements**:

        - Job listings are displayed, possibly with filters for location, industry, and job type.
    - **Action**:
        - The user scrolls through or uses filters to find a job of interest.

2. **Select a Job Role**
    - **Description**: The user selects a specific job role that matches their interest and qualifications.
    - **Action**:
        - The user clicks on the job title to see more details about the role.

3. **Click on View More**
    - **Description**: After selecting a job, the user can click the "View More" button to get detailed information about the position.
    - **Details**:
        - The job details may include job responsibilities, required qualifications, salary range, and benefits.

4. **Click Apply**
    - **Description**: Once the user has reviewed the job details and decided to apply, they click the "Apply" button to begin the application process.
    - **Action**:

        - The system navigates the user to the application form.

5. **Answer Questions or Upload Resume/Cover Letter**
    - **Description**: Depending on the employer's requirements, the user may be asked to answer specific questions about their qualifications or upload relevant documents such as a resume and/or cover letter.
    - **Action**:
        - If questions are asked, the user provides text-based answers.
        - Alternatively, the user uploads their resume and cover letter from their device.
    - **Validation**:
        - Ensure the file formats for uploads (e.g., PDF, DOC) are acceptable.
        - Required fields must be completed before proceeding.
6. **Click Submit**
    - **Description**: After providing all necessary details or documents, the user clicks the "Submit" button to finalize the job application.
    - **Action**:
        - The system processes the application and confirms that the submission was successful.
        - The user may receive a confirmation email or message.

Test cases: Job Application Process

Test cases Steps:

1. **View Latest Available Jobs:**

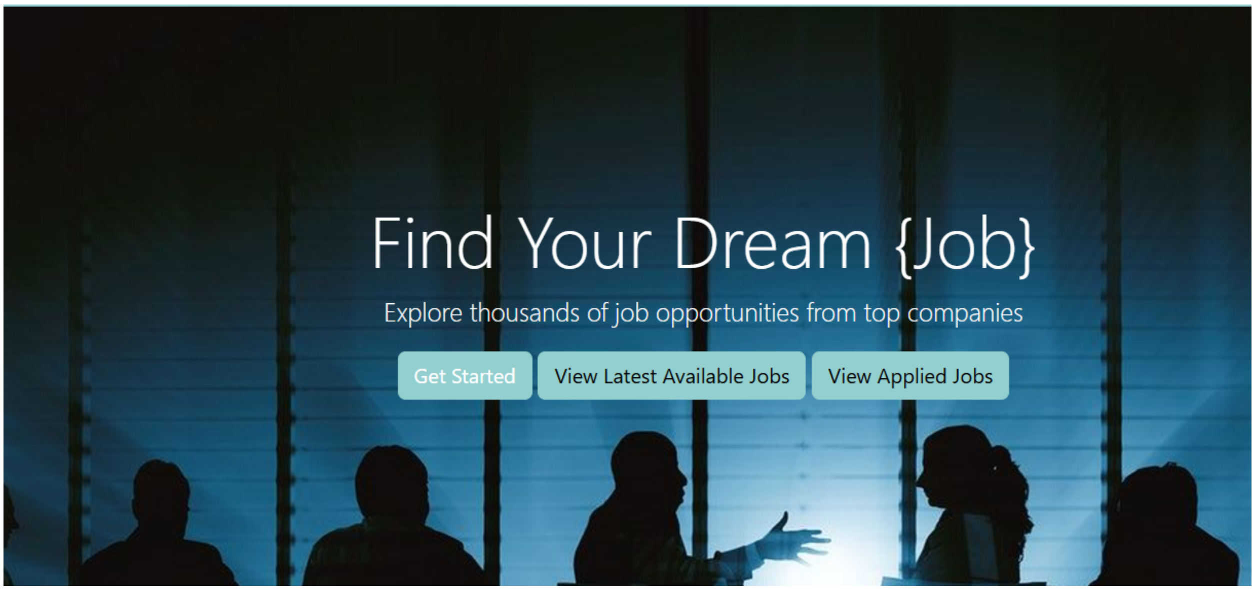


Image 12: Available Jobs

- **Action:** Navigate to the job listings page to view the latest available jobs.

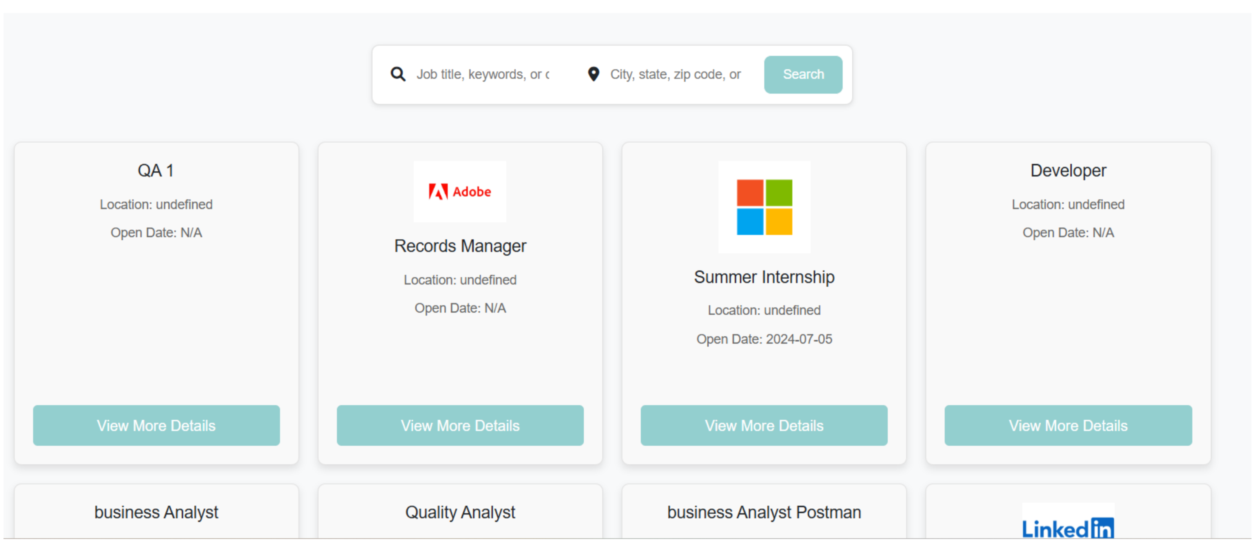


Image 12.1: Available Jobs

- A list of the latest job postings is displayed with relevant details (job title, location, posting date, etc.).

2. **Select a Job Role:**

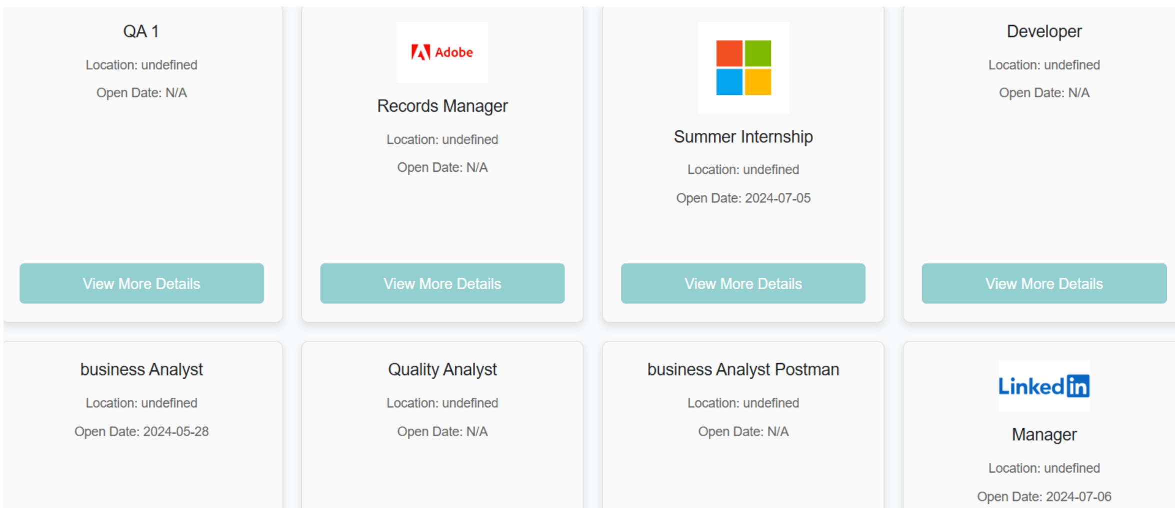


Image 12.2: Job role

- **Action:** Select a specific job role from the available list.

3. **Click on "View More":**

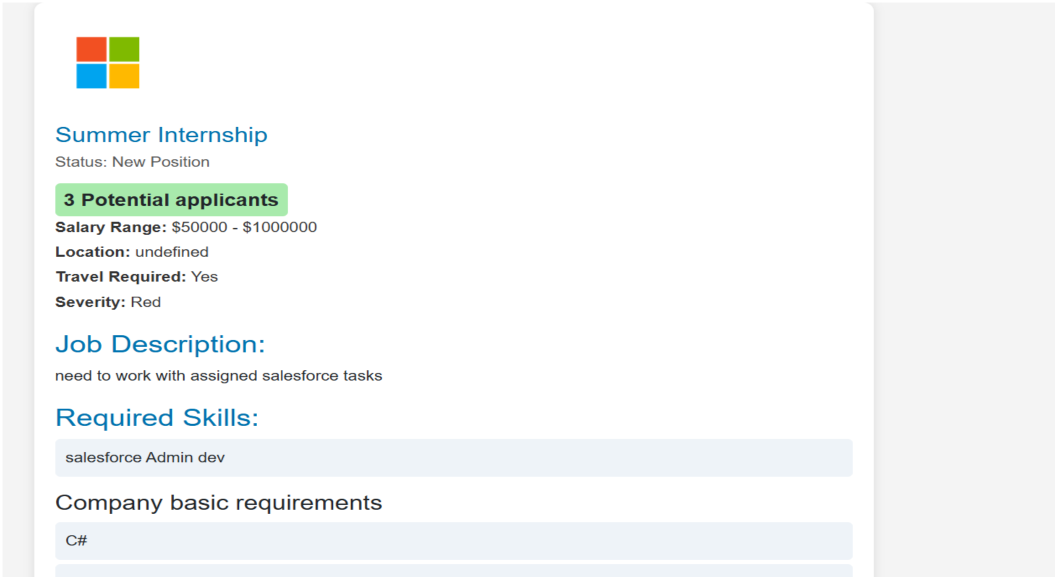


Image 12.3: View more

- **Action:** Click on the "View More" button/link for the selected job role to view additional job details.

4. **Click "Apply":**

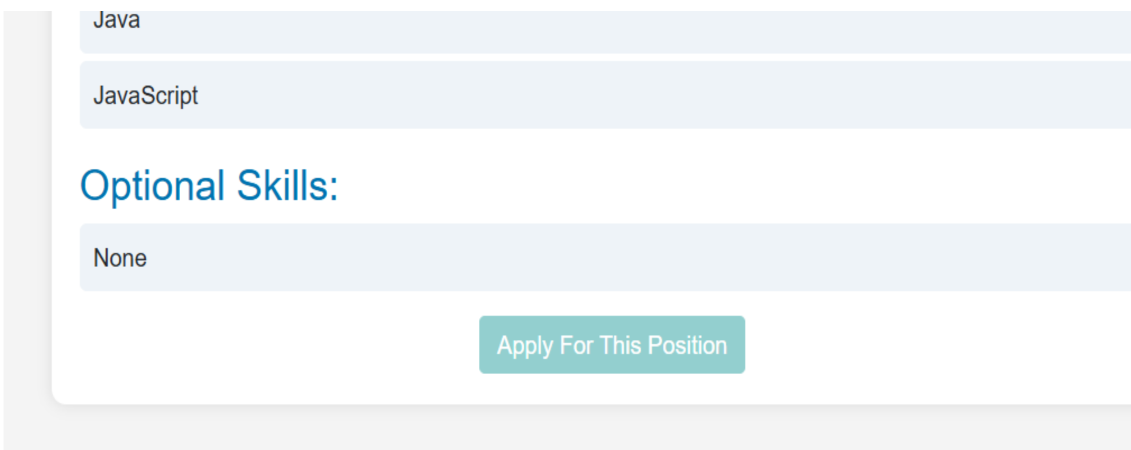


Image 12.4: Apply

- **Action:** Click on the "Apply" button for the selected job role.

5. **Answer Screening Questions (If Any):**

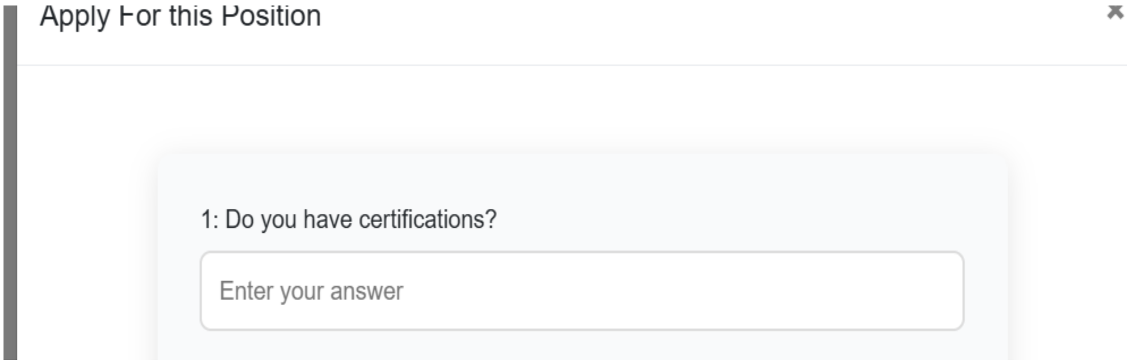


Image 12.5: Answer Screening Questions

- **Action:** If the job application requires answering specific questions, provide responses to these questions.

6. **Upload Resume or Cover Letter:**

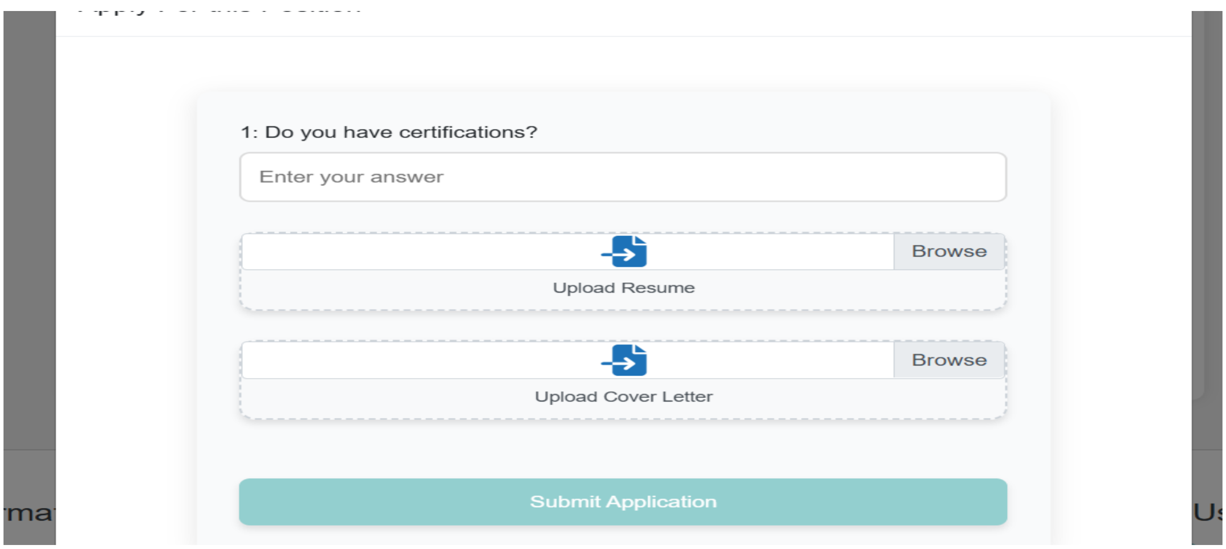


Image 12.6: Upload Resume or Cover Letter

- **Action:** Upload a resume and/or cover letter file, if required, to complete the application.

7. **Click "Submit":**

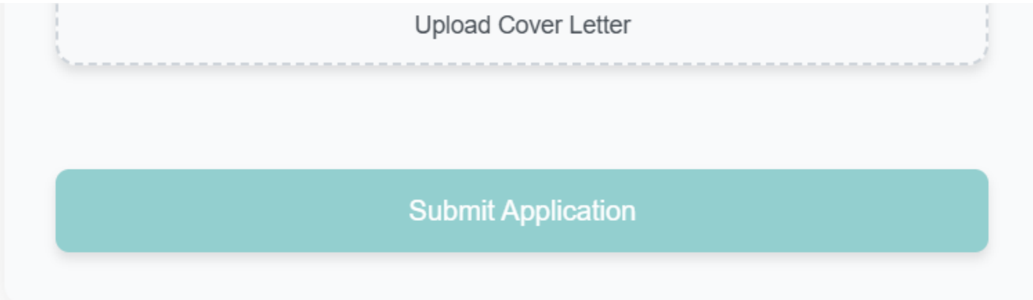


Image 12.7: Submit

- **Action:** Click on the "Submit" button to complete the job application.

Data Flow and Integration

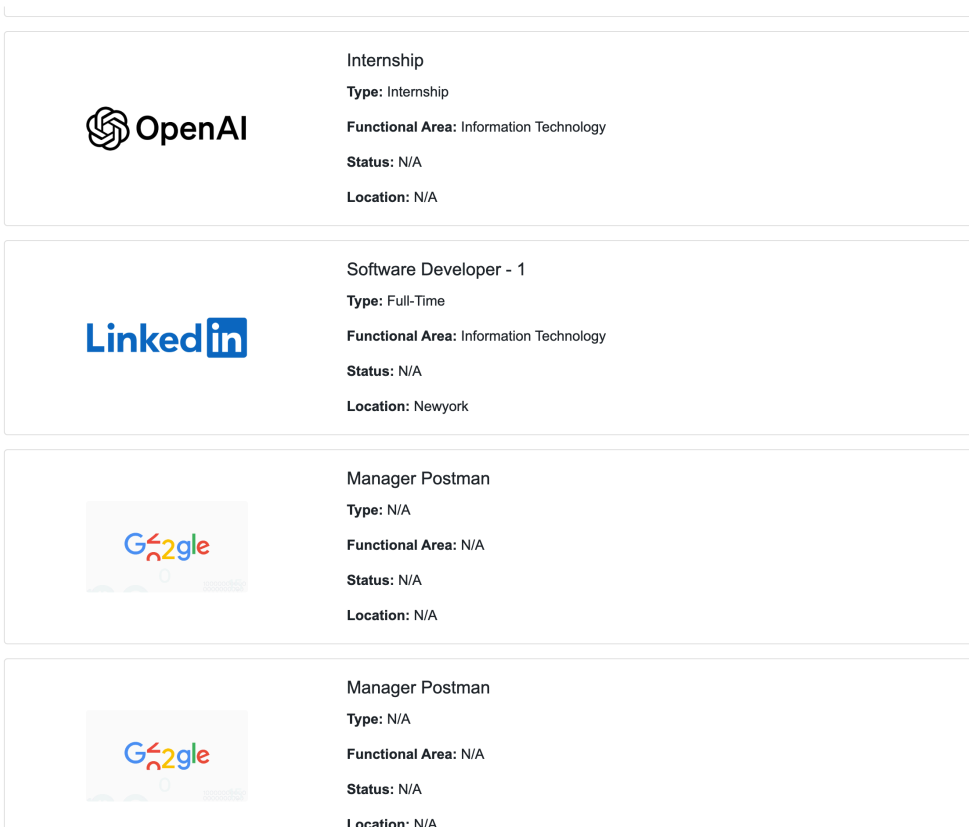


Image 13: Data flow and integration

Data Flow for Job Application

- **Step 1**:
  - When the user clicks **Apply** on the job details page, the system checks for and fetches recruiter-posted questions. These questions are displayed dynamically on the frontend.

- **Step 2**:
  - The user answers the questions, attaches their resume and CV, and submits the application.
- **Step 3**:
  - The frontend sends the form data (answers, attachments) to the backend, where it is processed and sent to Salesforce. A **Job Application** record is created, linked to the specific job and applicant.
- **Step 4**:
  - If successful, the user is notified that their application has been submitted.

Data Flow for Applied Jobs

- **Step 1**:
  - The user navigates to the **Applied Jobs** page. A call is made to the **Applied Jobs** REST resource in Salesforce to fetch all jobs the user has applied for.
- **Step 2**:
  - The REST resource returns details of each job application, including job name, position type, application status, job location, and the company's logo.

- **Step 3**:
  - The data is rendered as a set of **tiles** on the frontend, giving users a quick overview of their job applications. Each tile provides a summary of key information about the application.
- **Step 4**:
  - The user can interact with each tile to get more detailed information about the application and the current status.

Checking Any Updates or Status

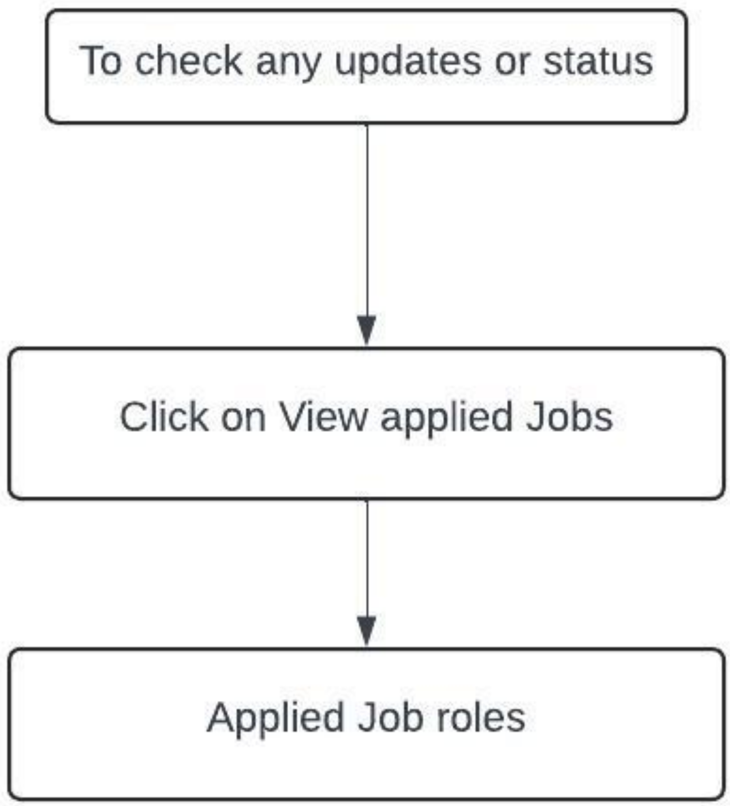


Figure 10: Checking any updates or status

1. Checking Any Updates or Status

To check for updates or the current status of any job application, start by navigating to the section that displays applied jobs.

- **Step 1**: Go to the main dashboard or the job application portal homepage.

- **Step 2**: Look for an option or menu labeled **“Applied Jobs”**, **“Job Applications”**, or something similar. This section contains all the roles you have previously applied for.

2. Viewing Applied Jobs

Once you're in the correct section of the application portal, you'll be able to see a list of jobs that you have applied for.

- **Step 3**: Click on the button labeled **“View Applied Jobs”**.
- **Step 4**: The system will display a list of all the jobs you've applied to. This includes information about the job titles and application submission dates.

3. Checking Applied Job Roles

After selecting the applied jobs, you will be able to see the details and status of each job role. This section will give specific insights into your application progress.

- **Step 5**: Review the list of jobs. For each job, you will see one of the following statuses:
    - **Pending**: Your application has been submitted and is awaiting review.
    - **In Progress**: Your application is under review or has moved to another stage in the process.
    - **Interview Scheduled**: An interview has been scheduled for this role.

- **Hired**: You have been selected for the role.
- **Rejected**: Your application has not been successful for this role.

Salesforce Object: Review

Overview

The Review object in Salesforce is used to capture and manage feedback on job applications. This object allows users to rate a job application and provide a description of their experience or opinion about the application. Each review includes a rating on a scale from 1 to 5 and a detailed description.

Fields

1. Job Application (Lookup Relationship)

- **Field Name:** Job_Application__c
- **Data Type:** Lookup (Job Application)
- **Description:** This field establishes a relationship between the Review object and a specific job application. Users select the job application they are reviewing. This helps track which application the review is related to.

2. Rating

- **Field Name:** Rating__c
- **Data Type:** Number (1,0)
- **Description:** A numerical field where the user rates the job application on a scale of 1 to 5.
- **Validation:** Must be an integer between 1 and 5. A value outside this range will not be accepted.

3. Review Description

- **Field Name:** Review_Description__c
- **Data Type:** Long Text Area
- **Description:** This field allows the user to provide a detailed description of their experience with the job application. The description can include both positive and negative feedback, or any additional insights they want to share.

4. Reviewer (Lookup Relationship)

- **Field Name:** Reviewer__c
- **Data Type:** Lookup (User)
- **Description:** This field associates the review with the user who created it. It helps in identifying which user provided the feedback on the job application.

5. Review Date

- **Field Name:** Review_Date__c
- **Data Type:** Date
- **Description:** The date when the review was submitted. This field is automatically populated when the review is created.

Key Features

- **Rating System:** Users can rate job applications between 1 (lowest) and 5 (highest).
- **Descriptive Feedback:** Users can provide detailed feedback about their experience with the job application, which can be valuable for improving hiring processes.
- **Linked to Job Application:** Each review is associated with a specific job application, allowing tracking of feedback for each application.
- **Reviewer Association:** The system tracks which user submitted the review for future reference or audits.

Business Logic and Validation Rules

1. **Rating Validation:**
   - The Rating__c field must have a value between 1 and 5.
   - If the user attempts to enter a value outside this range, they will receive an error message.
2. **Required Fields:**
   - Both Job_Application__c and Rating__c fields are required. A review cannot be created without selecting a job application and providing a rating.
   - The Review_Description__c field is optional but encouraged for detailed feedback.
3. **Default Values:**
   - The Review_Date__c field will automatically default to the current date when the review is created.

Use Case

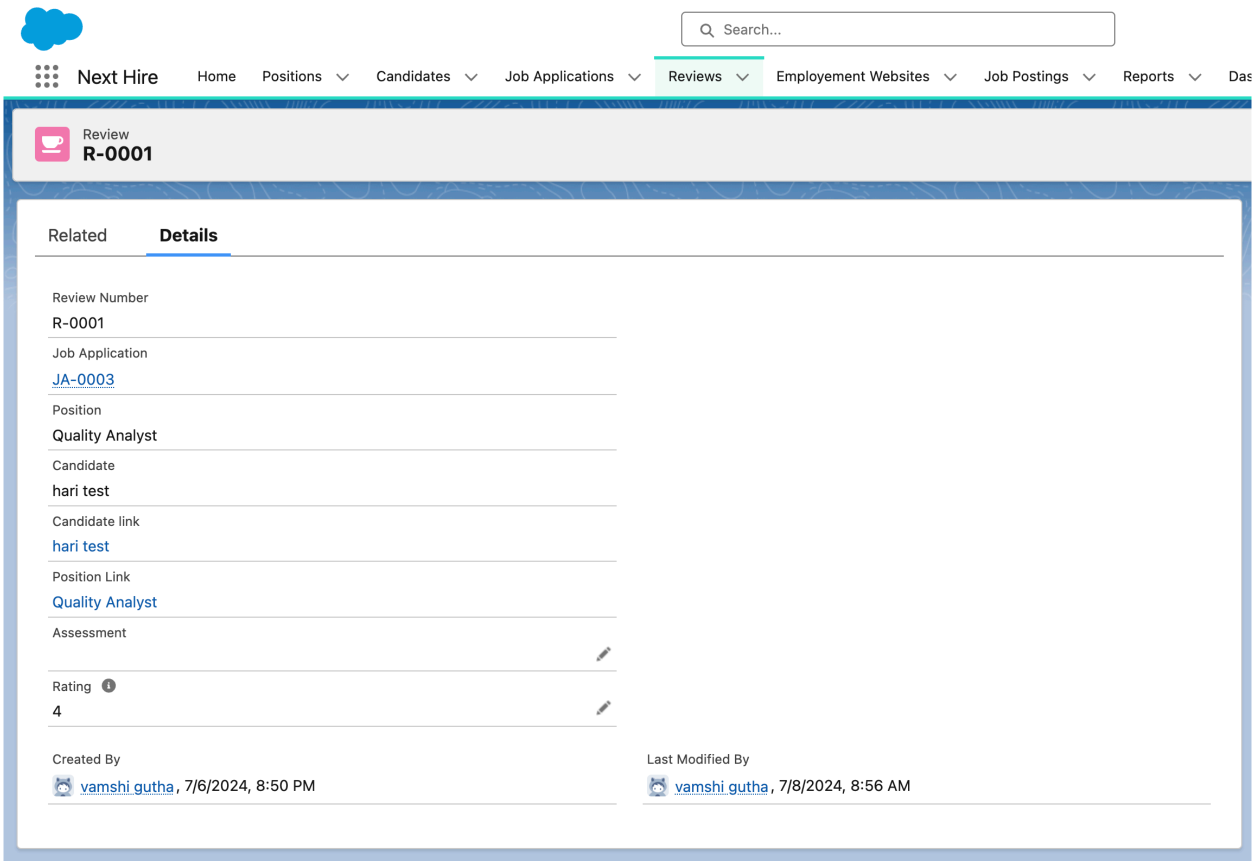


Image 14: Use case

Use Case

- **Scenario:** A user has applied for a job and after their interview, they are asked to provide feedback.
- **Steps:**
  1. The user navigates to the Job Application record they applied for.
  2. They create a new Review record.

3. They rate the job application on a scale of 1 to 5 using the Rating__c field.
4. They provide their feedback in the Review_Description__c field.
5. The review is submitted, and the Review_Date__c is populated with the current date.

Salesforce Reviews Object

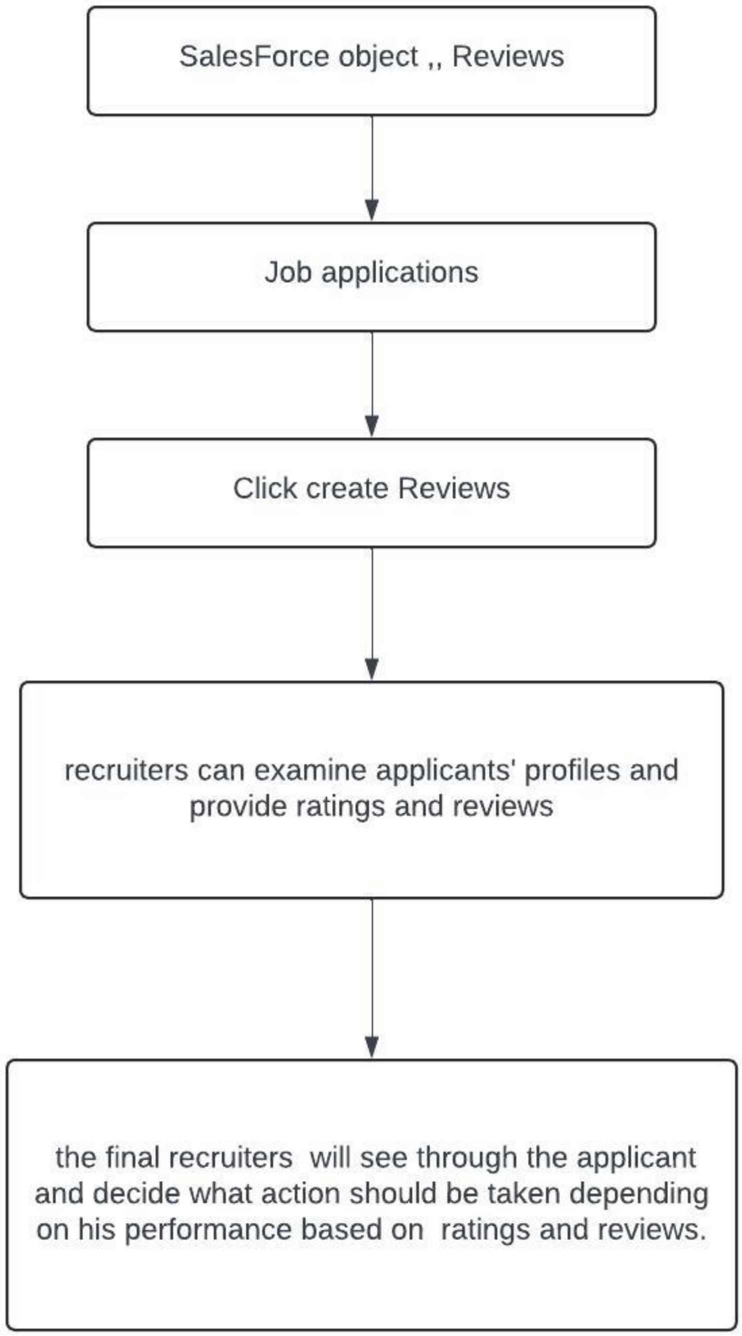


Figure 11: Salesforce reviews object

Step 1: Access Salesforce Reviews Object

- Navigate to the **Salesforce Reviews object**:
  - Purpose: To access the data and options related to applicant reviews.
  - Path: This may be available as a dedicated tab or object within Salesforce depending on your organization's customization.
  - Example: From the main dashboard, click on the **“Reviews”** tab.

Step 2: View Job Applications

- Inside the Reviews object, locate the **“Job Applications”** section:
  - Purpose: To view the list of all applicants who have applied for roles.
  - Data Displayed:
    - Job role title
    - Applicant name
    - Date of application
    - Current status of the application
  - Example: A table showing all the job applications, their associated roles, and basic applicant details.

Step 3: Click “Create Review”

- Select a specific job application and click **“Create Review”**:

    - Purpose: To begin reviewing the selected applicant.
    - Action: This button will generate a form or page where you can input your review.
    - Example: Clicking on the button brings up the applicant’s profile and a review form with rating options.

Step 4: Examine Applicant Profile

- Once inside the review section, recruiters can examine the applicant's profile:
    - Data Examined:
        - Resume or CV
        - Work experience and job history
        - Skills and qualifications
        - Any uploaded cover letter or certifications
    - Example: Reviewing the applicant's job experience and assessing whether it meets the job role's requirements.

Step 5: Provide Ratings and Reviews

- After reviewing the profile, recruiters can provide ratings and written feedback:
    - Rating Scale: Recruiters can score the applicant on key areas such as:
        - **Experience**: Does the applicant's previous work experience match the job role?

        - **Skills**: Are the skills listed relevant and strong?
        - **Qualifications**: Does the applicant possess the required education or certifications?
    - Written Review: Recruiters can leave detailed feedback to explain their ratings.
    - Example: Rating an applicant 4 out of 5 for experience but noting a lack of specific certifications.

Step 6: Final Review by Decision-Makers

- The final step is for decision-makers to review all the feedback and ratings from multiple recruiters:
    - Purpose: To make a hiring decision based on collective reviews.
    - Data Considered:
        - Cumulative ratings
        - Strength of written feedback
        - Any red flags or highly positive aspects of the applicant
    - Example: An applicant with consistently high ratings might be moved forward to the interview stage, whereas one with mixed ratings may be rejected or put on hold.

Step 7: Decision and Action

- Once reviews are complete, the final decision can be made:
    - **Possible Actions**:
        - **Accepted**: The applicant is shortlisted for an interview or hiring.
        - **Rejected**: The applicant is removed from the process.
        - **On Hold**: Further information is required, or the decision is deferred.
    - Example: The final reviewer marks an applicant as **"Interview Scheduled"** based on strong ratings and feedback.

Salesforce Job Application Object

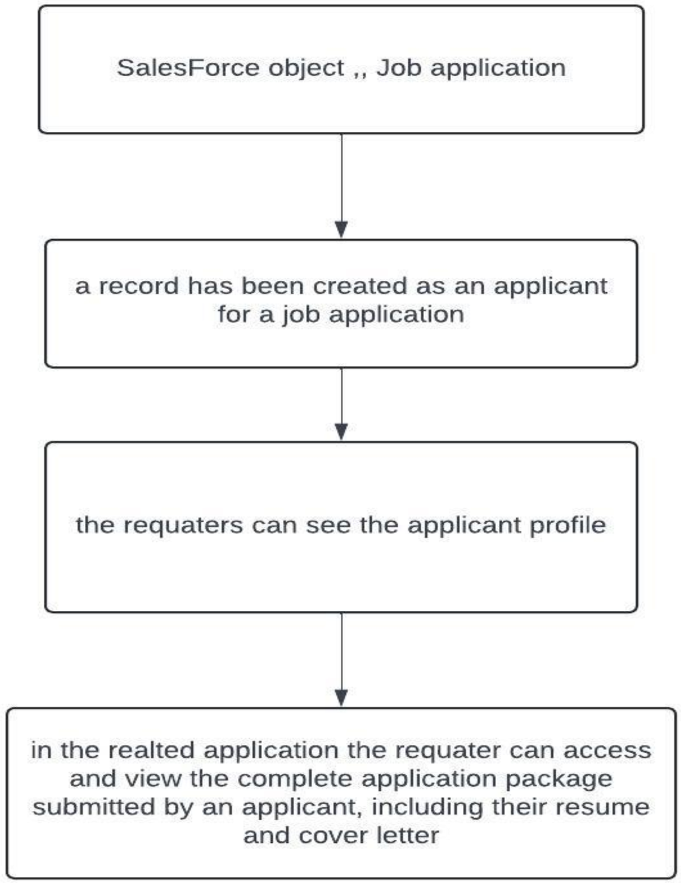


Figure 12: Salesforce Job Application Object

Step 1: Access Salesforce Job Application Object

- **Purpose**: To view and manage job applications.
- **Action**: Navigate to the **"Job Application"** object in Salesforce.

Step 2: Applicant Record Creation

- **Purpose**: When an applicant applies for a job, a new record is automatically created in the Job Application object.
- **Data Captured**:
  - Applicant's personal information (e.g., name, email, phone number)
  - The specific job the applicant has applied for
  - Submission date and status of the application (e.g., pending, under review)

Step 3: Viewing the Applicant Profile

- **Purpose**: Recruiters can access the applicant's profile to review key details such as their resume and cover letter.
- **Action**: Click on an applicant's name in the Job Application object to view their profile.
- **Data Displayed**:
  - Complete personal details (e.g., address, phone number, email)
  - Work experience, qualifications, and education history (from resume)
  - Any additional supporting documents submitted

Step 4: Accessing the Complete Application Package

- **Purpose**: In the related job application, recruiters can view the full submission package that includes all documents provided by the applicant.
- **Action**: Open the detailed view of the application by clicking on the applicant's name or related records.
- **Data Reviewed**:
    - **Resume**: A detailed summary of the applicant's work experience, skills, and education.
    - **Cover Letter**: A letter explaining the applicant's motivation for applying and suitability for the role.
    - **Additional Documents**: Certificates, references, or portfolios that were uploaded during the application process.

Matrix (Bar Chart)

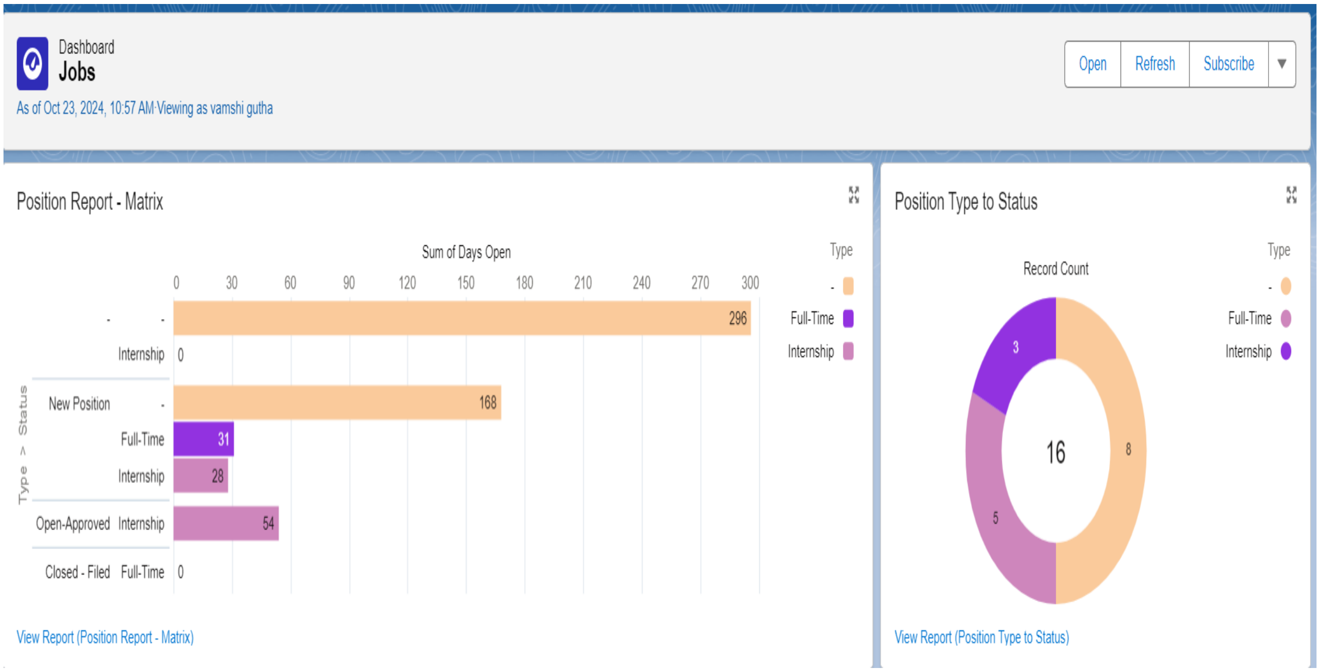


Image 15: Matrix bar chart

1. **Position Report - Matrix (Bar Chart)**

This matrix shows the number of days that various job positions have been open, categorized by job type and status. It is designed to provide insights into how long different positions have been available.

- **Columns**:
  - **Type**: This categorizes the job type into either Full-Time or Internship positions.
  - **Status**: Represents the current stage of the position, such as:

        - **New Position**: Jobs that are newly created.
        - **Open-Approved**: Positions that are approved and open for hiring.
        - **Closed-Filled**: Positions that are filled and closed.
    - **Bars**:
        - The length of each bar represents the **Sum of Days Open** for each category. The longer the bar, the longer the job has remained open.
        - Example Insights:
            - **Full-Time Closed-Filled Positions**: The chart shows these positions have been open for a total of 168 days.
            - **Internship Open-Approved Positions**: Internship positions with approved status have been open for a total of 54 days.
            - An unspecified job type (indicated as "-") has the longest open period with a total of 296 days, which may require further investigation or classification.

2. **Position Type to Status (Pie Chart)**

This section provides a high-level breakdown of job positions by type and status in the form of a pie chart, allowing for quick assessment of the distribution of job types.

- **Record Count**: Shows the number of positions in different categories.
  - **Full-Time Positions**: There are 8 Full-Time positions across all statuses.
  - **Internship Positions**: There are 5 Internship positions.
  - **Unspecified Positions**: There are 3 positions without a specified type.

The pie chart gives a visual representation of the number of positions available or filled in each category, helping HR and management teams quickly see the proportion of each job type.

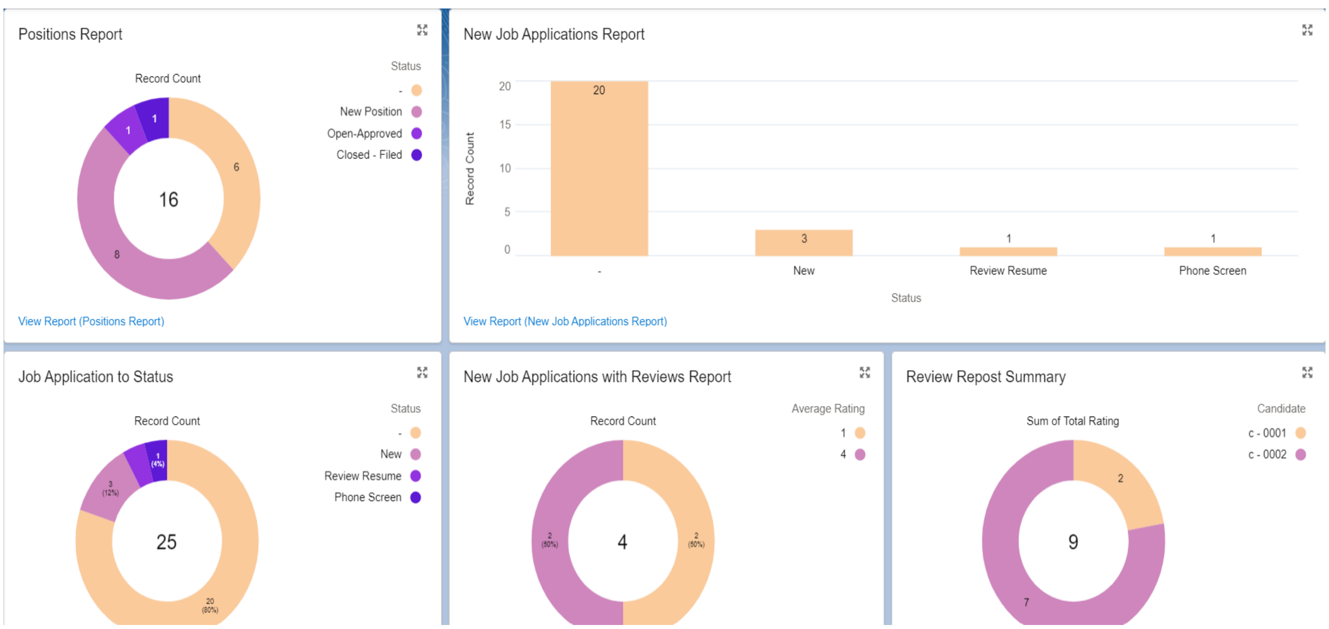


Image 15.1: Donut chart

1. **Positions Report (Donut Chart)**

This report provides a breakdown of job positions by status, offering a quick view of the distribution of jobs across different stages of recruitment.

- **Total Positions**: 16
- **Categories**:
  - **New Positions**: 6 positions (light orange)
  - **Open-Approved Positions**: 8 positions (light pink)
  - **Closed-Filled Positions**: 1 (purple)
  - **Unspecified**: 1 position (dark purple)

The donut chart helps to visualize how many positions are in each category and the status of job postings in the system.

2. **New Job Applications Report (Bar Chart)**

This report tracks the current status of new job applications across various stages of the recruitment process.

- **Total Applications**: 25 applications.
- **Statuses**:
  - **New Applications**: 20 applications (not yet processed or reviewed).
  - **Reviewed Resumes**: 3 applications.
  - **Phone Screens Completed**: 1 application.

The bar chart provides insights into where most job applications are currently standing and can help recruitment teams focus on moving candidates through the funnel.

3. **Job Application to Status (Donut Chart)**

This donut chart displays the record count of job applications across different stages. It gives a clear picture of how many applicants are in each phase of the hiring process.

- **Total Applications**: 25
- **Categories**:
  - **New**: 20 applications (80%) (light orange)
  - **Review Resume**: 3 applications (12%) (light purple)
  - **Phone Screen**: 1 application (4%) (dark purple)

The visualization is useful for tracking where the bulk of the applications lie and identifying which stages need more attention.

4. **New Job Applications with Reviews Report (Donut Chart)**

This report focuses on applications that have been reviewed and their corresponding ratings.

- **Total Reviewed Applications**: 4 applications
- **Rating Distribution**:
  - **Rating 1**: 2 applications (50%) (orange)
  - **Rating 4**: 2 applications (50%) (purple)

This chart is designed to show the number of reviewed applications along with their average ratings, helping recruiters quickly identify well-rated candidates.

5. **Review Repost Summary (Donut Chart)**

This section provides a summary of total ratings given to candidates, based on individual feedback.

- **Total Sum of Ratings**: 9
- **Candidates**:
    - **Candidate c-0001**: 2 total ratings (light orange)
    - **Candidate c-0002**: 7 total ratings (purple)

This summary helps track which candidates have been reviewed more thoroughly and may require further consideration in the hiring process.

Salesforce REST Resource: JobProfile API Documentation

Overview

The JobProfile REST resource provides a way to interact with candidate profiles in Salesforce through a custom Apex REST class. It allows external systems to retrieve and update candidate details using GET and PATCH methods. The resource is designed to be flexible and handles various scenarios such as missing or incorrect input.

Resource URL

bash

Copy code

/services/apexrest/profile

HTTP Methods

This resource supports the following HTTP methods:

- GET: Retrieve candidate details based on the CandidateId.
- PATCH: Update candidate details by sending a JSON body with fields to update.

Authentication

OAuth authentication is required to access this resource. Ensure that the integration has the necessary permissions to query and update candidate records.

GET Method: Retrieve Candidate Profile

Purpose:

Fetches details of a candidate based on their unique CandidateId. The response includes information like personal details, education, experience, and contact information.

Request Format:

- **HTTP Method**: GET
- **Endpoint**: /services/apexrest/profile?CandidateId={CandidateId}

Parameters:

- **CandidateId**: The unique Salesforce record ID of the candidate (required).

Success Response:

Returns a list containing the candidate's profile information, including fields such as name, email, phone, skills, and years of experience.

Sample Response (JSON):

json

Copy code

[

```
 {
  "Id": "a01xx000003FgTkAAK",
  "First_Name__c": "John",
  "Last_Name__c": "Doe",
  "Email__c": "john.doe@example.com",
  "Mobile__c": "+1234567890",
  "Years_of_Experience__c": 5,
  "City__c": "New York",
  "Country__c": "USA"
 }
]
```


Error Responses:

- **400 Bad Request**: Returned if the CandidateId is missing or invalid.
  - Example: Error: CandidateId is required
- **404 Not Found**: Returned if no candidate is found for the provided CandidateId.
  - Example: Error: Candidate not found
- **500 Internal Server Error**: If an unexpected error occurs during the request.
  - Example: Error: [Exception message]

PATCH Method: Update Candidate Profile

Purpose:

Updates specific fields of the candidate's profile by accepting a JSON body with the fields to be updated. This method is useful for updating dynamic fields such as phone numbers, email addresses, or years of experience.

Request Format:

- **HTTP Method**: PATCH
- **Endpoint**: /services/apexrest/profile?CandidateId={CandidateId}
- **Content-Type**: application/json

Parameters:

- **CandidateId**: The unique Salesforce record ID of the candidate (required).

Request Body:

A JSON object containing the fields to be updated. Only the fields specified in the request body will be modified.

Sample Request (JSON):

json

Copy code

```
{
  "Email__c": "new.email@example.com",
```

"Mobile__c": "+0987654321",

"Years_of_Experience__c": "6"

}

Success Response:

Returns a confirmation message if the candidate profile is successfully updated.

Sample Response (Text):

Copy code

Candidate profile updated successfully

Error Responses:

- **400 Bad Request**:
  - If CandidateId is missing: Error: CandidateId is required
  - If the request body is missing or not a valid JSON object: Error: Request body is required
  - If the request body contains invalid types: Error: Invalid type for field update
- **404 Not Found**: If the candidate with the provided CandidateId does not exist.
  - Example: Error: Candidate not found
- **500 Internal Server Error**: Returned if an error occurs while updating the record (such as a DML exception).

- Example: Error: [Exception message]

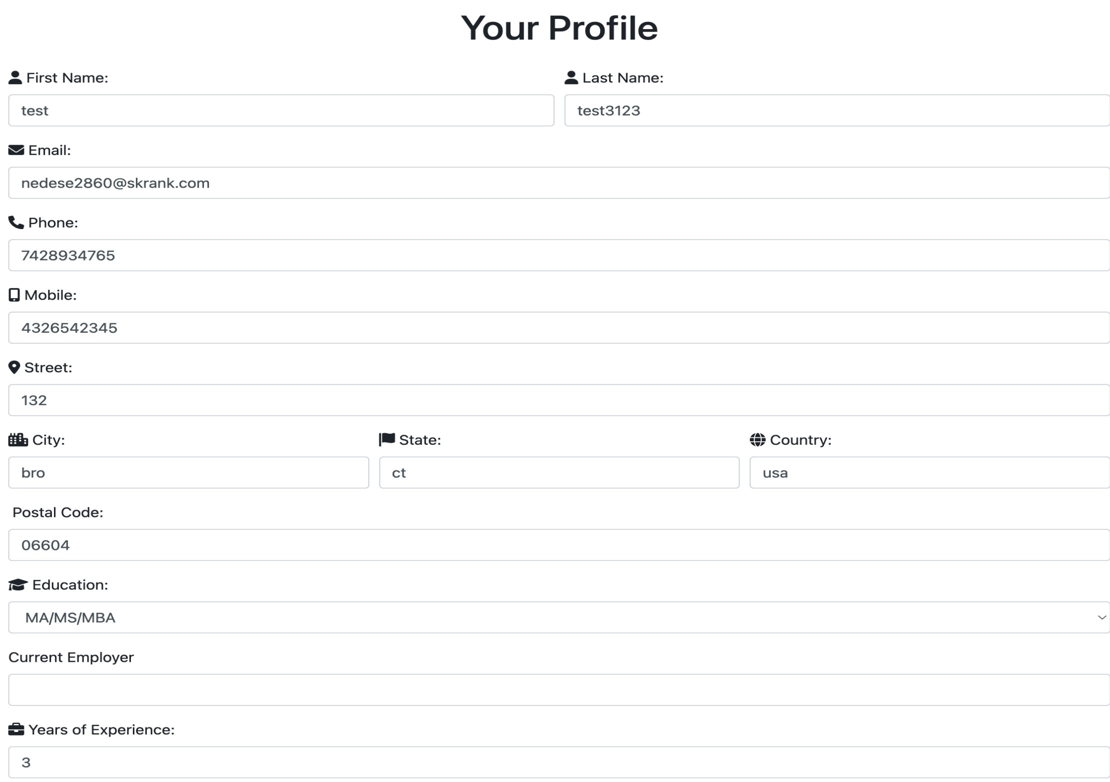


Image 16: Your profile (example)

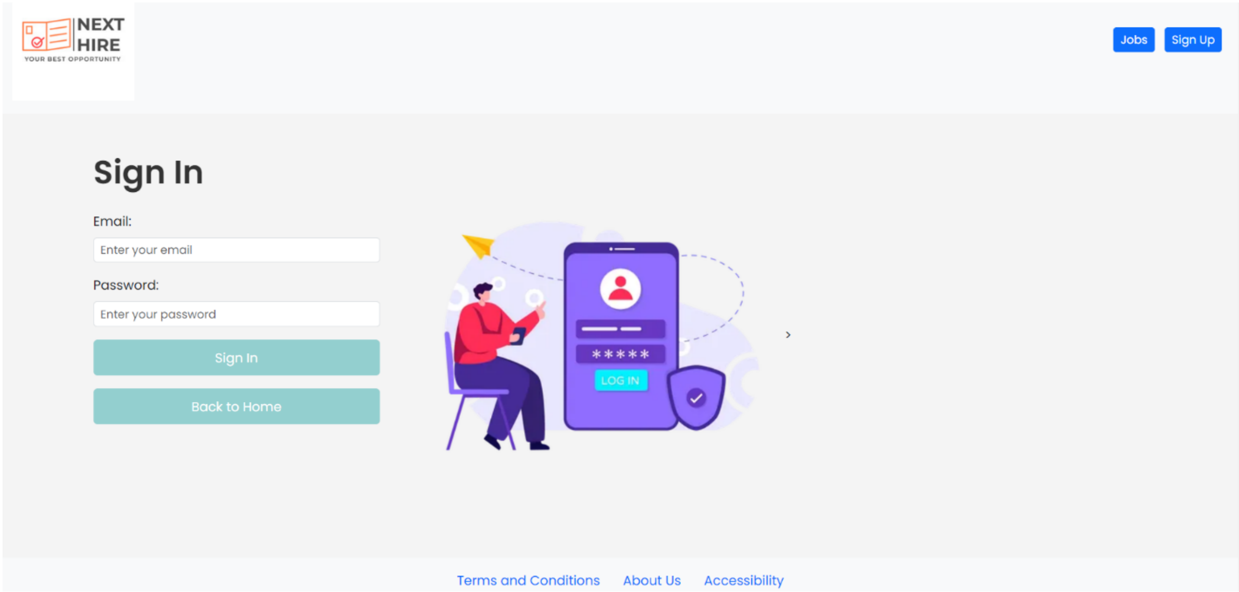


Image 17: Sign in style

Font and Styling: Implemented the Poppins font for a modern, clean aesthetic, consistent across the page.

Logo Hover Effect: Added an interactive hover effect where the logo flips (rotates) when hovered over, giving a playful yet professional touch.

Responsive Layout: Utilized flexbox for a responsive layout, ensuring that the form and illustration adjust nicely on different screen sizes.

Navigation Button Styling: Enhanced the look of the navigation buttons ("Jobs", "Sign Up") with a custom color scheme and a hover effect that changes to a darker shade.

Full-Width Form Fields: Adjusted the input fields to stretch the full width of the container for easier user interaction, improving overall readability and accessibility.

Error Message Handling: Built-in error messages for incorrect input (e.g., invalid email, short password) that only appear when triggered, keeping the interface clean.

Interactive Button Effects: All buttons, including the "Sign In" and "Back to Home" buttons, now feature hover effects that darken the button, enhancing user experience through visual feedback.

Illustration Integration: The sign-in illustration is aligned beside the form in a way that maintains responsiveness. Animation support has been added for future use.

Footer Design: The footer now mirrors the header's design, including space between footer links, a soft background color, and hover underline effects on the links.

Library Integrations: Included Bootstrap for the responsive grid and button styling, Google Fonts for the Poppins typeface, and the AOS (Animate On Scroll) library for potential future scroll-triggered animations.

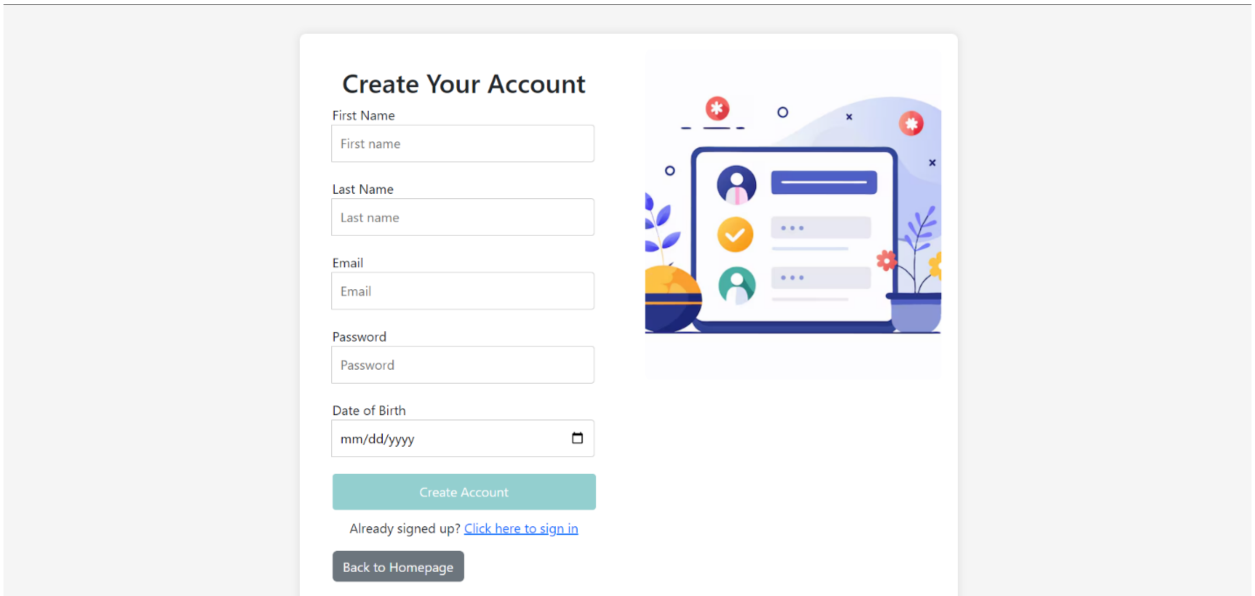


Image 18: create your Account style

Enhanced Logo Size and Placement: The logo size was increased and positioned to maintain its aspect ratio while providing more prominence in the header.

Improved Header and Navigation: The header layout was updated with more spacing between navigation buttons and a consistent design for a cleaner appearance and improved accessibility.

Refined Form Layout: The sign-up form was structured into well-spaced sections with enhanced padding to improve the user interface and make form fields more readable.

Input Validation: JavaScript validation was added for first name, last name (alphabetic characters only), password length (between 8 and 20 characters), and age verification (users must be at least 18 years old).

Visual Feedback for Errors: Error handling with alert messages was introduced for invalid inputs, such as name format, password length, and age restrictions, improving user interaction.

Illustration Integration: A visually appealing sign-up illustration was added to the right side of the form, balancing the overall layout and providing a modern look.

Mobile Responsiveness: The page was made responsive, ensuring that the layout adjusts properly for smaller screen sizes, with the form and illustration stacking vertically on mobile devices.

User-Friendly Back to Homepage Button: A "Back to Homepage" button was added below the form for easy navigation, offering a clear path back to the main site.

Consistent Button Styles and Hover Effects: Buttons across the page were standardized with consistent background colors, padding, and hover effects to provide a cohesive user experience.

Streamlined Footer: The footer design was simplified with clean navigation links, ensuring a consistent color scheme and better spacing for improved accessibility.

DOWNLOAD our App:

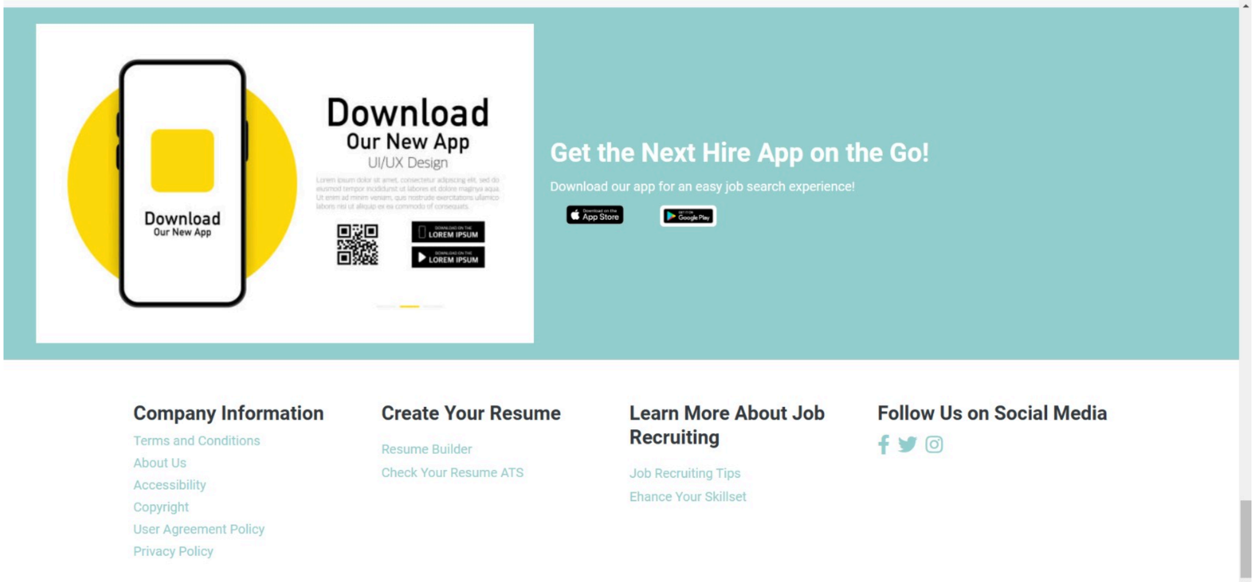


Image 19: Download our App

App Promotion Section:

- **Title**: "Download Our New App" with a subtitle mentioning "UI/UX Design."
- **Text**: Placeholder "Lorem ipsum" text describes the app's purpose or features.
- **QR Codes**: Two QR codes are present, presumably for quick downloads on different platforms.
- **Download Buttons**: Black buttons labeled with "Download on the App Store" and "Download on Google Play."

Header Section:

- Contains a call to action: "Get the Next Hire App on the Go!" and a prompt to "Download our app for an easy job search experience."

Footer Section:

- **Company Information**: Links to "Terms and Conditions," "About Us," "Accessibility," "Copyright," "User Agreement Policy," and "Privacy Policy."
- **Create Your Resume**: Links to a "Resume Builder" and an "ATS Resume Checker."
- **Learn More About Job Recruiting**: Links to "Job Recruiting Tips" and "Enhance Your Skillset."
- **Social Media Links**: Icons for Facebook, Twitter, and Instagram.

Flow Structure and Components

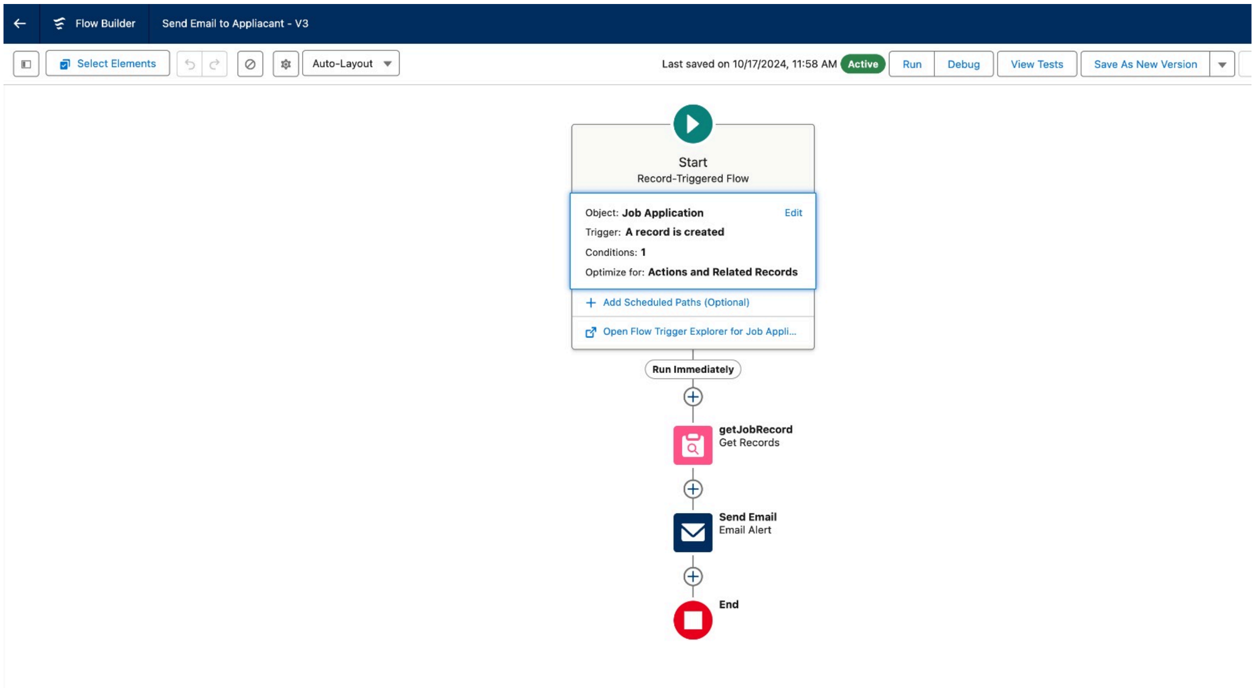


Figure 13: Flow Structure and components

1. **Start Element**
   - **Trigger**: When a Job Application record is created.
   - **Conditions**:
     - There is one specified condition (details about this condition would be set in the "Edit" settings, not shown in the image).
   - **Optimization**: Set for "Actions and Related Records" to ensure efficient execution in related records and actions.

2. **Run Immediately Path**
    - **Description**: Executes the flow immediately upon the creation of a job application record.
3. **Get Records Element: getJobRecord**
    - **Action**: Retrieves relevant records.
    - **Purpose**: The flow uses this element to gather additional information on the job for which the applicant applied. This data may include job title, department, location, etc., depending on the fields defined within the Job object.
    - **Configuration**:
        - **Object**: Job (presumably related to the Job Application object).
        - **Filter Conditions**: Retrieves only the specific job details associated with the application.
        - **Output**: Data retrieved here will be used in the email message to provide relevant information to the applicant.
4. **Send Email Element**
    - **Action**: Sends an email to the applicant.
    - **Purpose**: This element sends an automated email to the applicant acknowledging the submission of their job application.

- **Email Template**: An email template (configured outside of the flow) is used here. The template may include placeholders for the job title, applicant's name, and other relevant details, dynamically populated with data from the flow.
- **Recipient**: The applicant's email address, captured in the Job Application record.
- **Content**: Common content for this email might include a thank-you note, the next steps in the application process, and contact information for inquiries.

5. **End Element**
    - **Purpose**: Marks the completion of the flow once the email has been sent.

Figure 14: Start Element

1. **Start Element**
   - **Trigger**: When a Job Application record is updated.
   - **Conditions**: There is one specified condition (details on this condition are configured in the "Edit" settings, not visible in the image).
   - **Optimization**: Optimized for "Actions and Related Records" for efficient data processing and actions related to record updates.

2. **Run Immediately Path**
    - **Description**: Executes immediately upon the update of a job application record.

3. **Get Records Element: Get Record**
    - **Action**: Retrieves relevant records related to the updated application.
    - **Purpose**: This element fetches additional data related to the application, such as the applicant's details or job-specific information, to personalize the email message.
    - **Configuration**:
        - **Object**: Job Application or a related Job record.
        - **Filter Conditions**: Ensures that only the records associated with the updated application are retrieved.
        - **Output**: Provides data to be used in the decision and email elements.
4. **Decision Element: Status of Application**
    - **Action**: Evaluates the application status to determine which email to send.
    - **Purpose**: This element checks the status of the job application and routes the flow to send the appropriate email based on the status.

- **Outcomes**:
  - **Rejected**: Routes the flow to the path that sends a rejection email.
  - **Default Outcome (Status Not Rejected)**: Routes the flow to the path that sends a non-rejection status update email, such as "In Progress" or "Accepted."

5. **Email Alert Elements**
   - **Purpose**: Sends an email update to the applicant based on the decision outcome.
   - **Rejected Email Alert**:
     - **Action**: Sends an email notifying the applicant that their application has been rejected.
     - **Email Template**: Configured to provide a polite rejection message with any additional details as specified by the recruiting team.
     - **Recipient**: The applicant's email address, which is retrieved from the Job Application record.
   - **Status Not Rejected Email Alert**:
     - **Action**: Sends an email informing the applicant of a non-rejection status, such as "In Progress" or "Accepted."

        - **Email Template**: Configured to include details on the updated status of the application.
        - **Recipient**: The applicant's email address, retrieved from the Job Application record.

6. **End Element**
    - **Purpose**: Concludes the flow once the email has been sent.

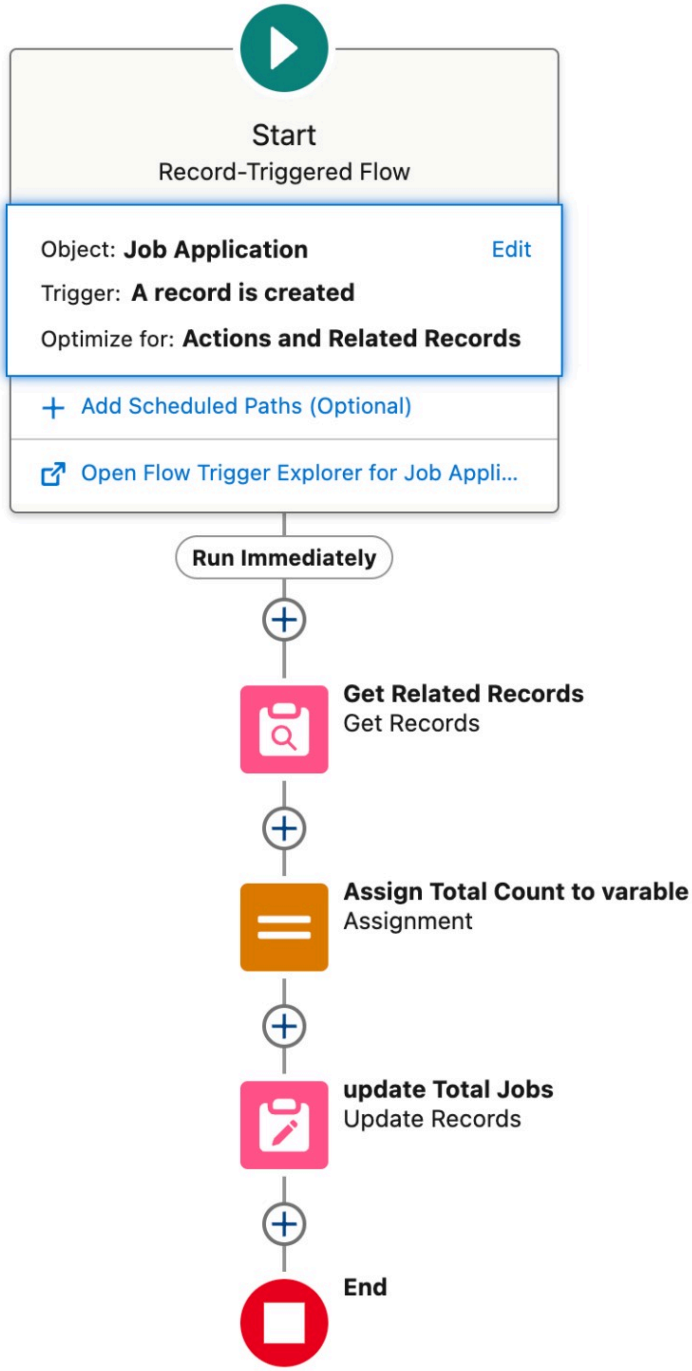


Figure 15: Start Record Triggered Flow

Start (Record-Triggered Flow)

Object: Job Application

Trigger: The flow is triggered when a job application record is created.

Optimization: Set to optimize for "Actions and Related Records" to improve performance with related records.

Get Related Records (Get Records)

Action: Retrieve the related job position records associated with the job application.

Purpose: This step fetches the job position record(s) linked to the new job application so that the total count of applications can be updated for that specific position.

Assign Total Count to Variable (Assignment)

Action: Assign the current total number of applications for the position to a variable.

Purpose: This variable will temporarily store the total number of applications before incrementing it by one.

Update Total Jobs (Update Records)

Action: Update the job position record with the new total application count.

Steps:

Increment the value of the total applications count by one.

Update the job position record with this new count.

Purpose: This step ensures that the job position record accurately reflects the new total count of applications.

End

Action: Conclude the flow.

Purpose: Successfully complete the record-triggered flow.

TOP TEN JOB ROLES

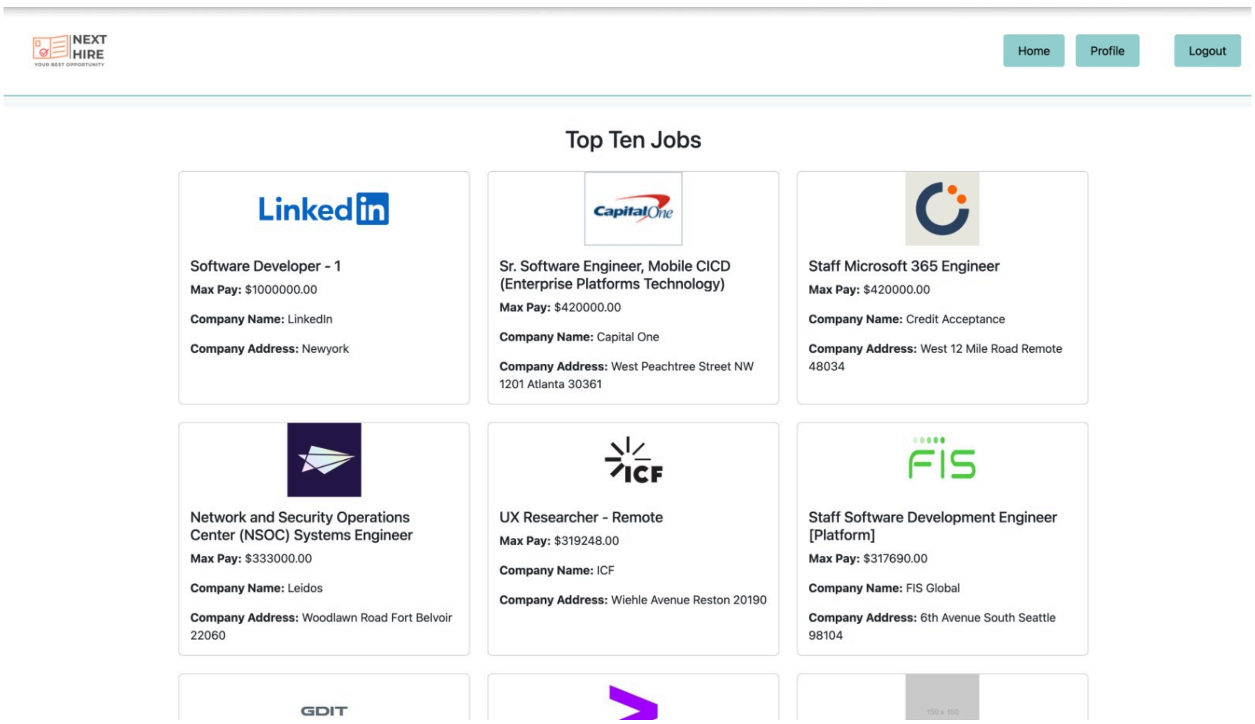


Image 20: top ten job roles

Job Title and Role: This is the main designation of the job. For example, "Software Developer - 1" and "Sr. Software Engineer, Mobile CICD."

Maximum Pay: This shows the maximum salary that could be offered for each role. The salaries vary significantly among the listings, indicating roles with high earning potential.

Company Name: This is the name of the company offering the job. Examples include LinkedIn, Capital One, and Leidos.

Company Address: This provides the location of the company's office related to the job. Some roles, like the "UX Researcher" from ICF, indicate a specific address (e.g., "Wiehle Avenue Reston"), while some mention remote work options.

TOP TEN COMPANIES

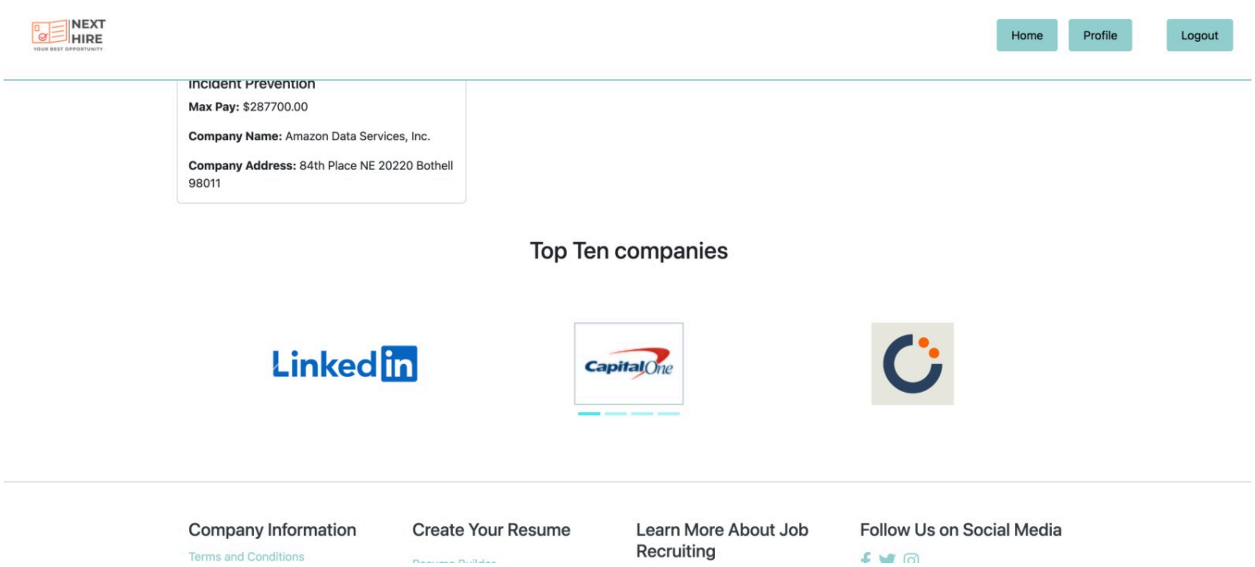


Image 21: top ten companies

Top Ten Companies: This area displays logos of notable companies, such as LinkedIn and Capital One. By showcasing these companies, the platform highlights well-known or popular employers, suggesting that they either frequently have job openings or are considered desirable workplaces. The layout uses a clean design with logos in a row, making it easy for users to recognize these companies at a glance.

Position Report – Matrix

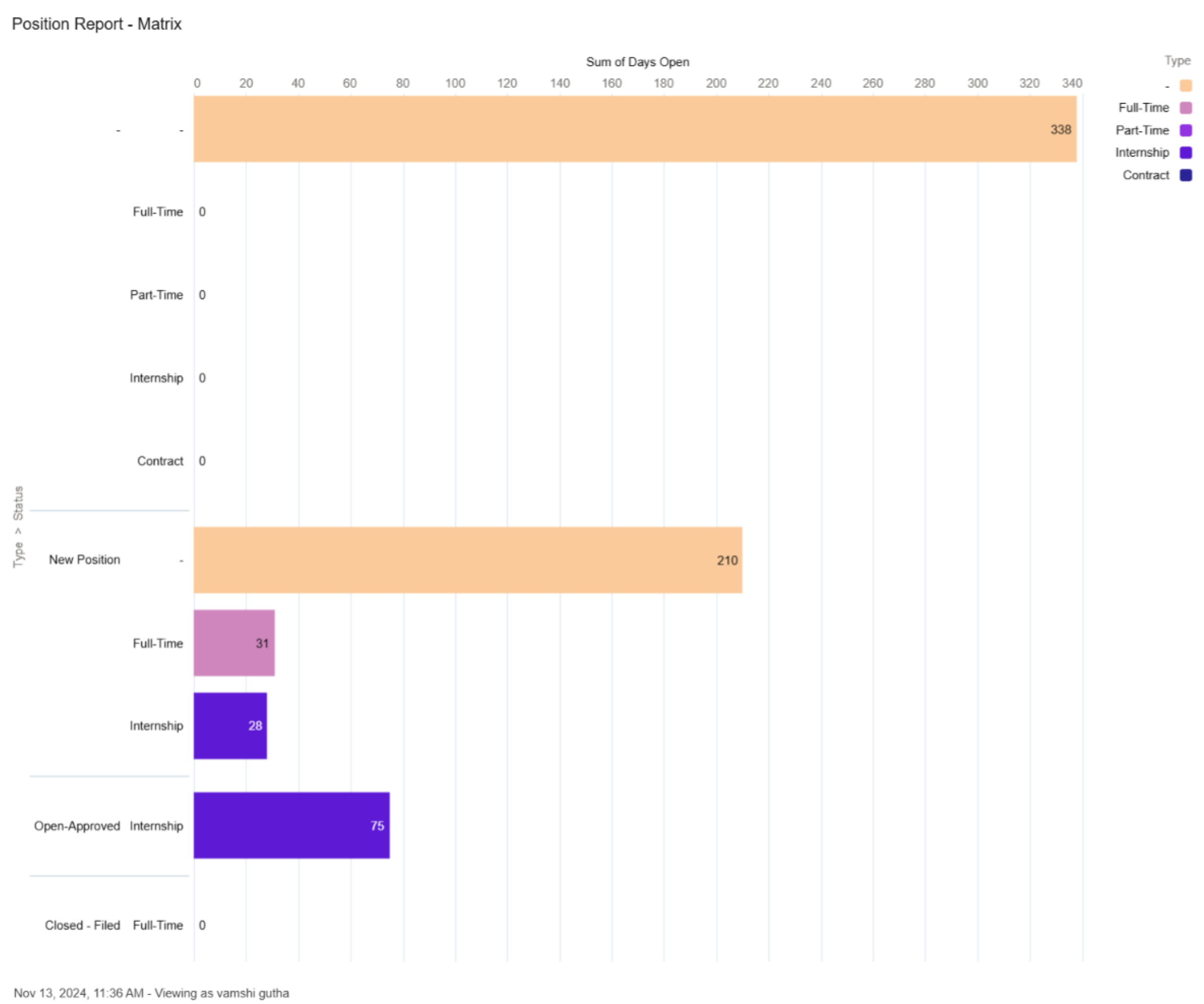


Image 22: Position report matrix

The **Position Report - Matrix** bar chart visualizes the **sum of days positions have been open** across different job types (Full-Time, Internship, Part-Time, Contract) and statuses (e.g., New Position, Open-Approved, Closed-Filled). The horizontal bars represent the total "days open" for each category, with the **X-axis** displaying the numerical count and the **Y-axis** organizing the data by job status and type.

Key Points:

1. **Dominant Category**:
    - The "New Position" status with no specific type listed ("-") has the longest duration, open for **338 days**.
2. **Internships**:
    - Under "New Position," internships have been open for **28 days**.
    - Approved internships ("Open-Approved") account for **75 days** open.
3. **Full-Time Roles**:
    - Full-time jobs under "New Position" have been open for **31 days**.
4. **Missing Data**:
    - No data is shown for part-time and contract roles, or for the "Closed-Filled" status.

Links

Git hub Front End Repo: https://github.com/guthaVamshi/Next-Hire

Training : https://github.com/guthaVamshi/Next-Hire/tree/main/Training%20Videos

Git Hub Salesforce Repo:

https://github.com/guthaVamshi/Next-Hire--Salesforce

Instructions to read Code of salesforce:

Navigate to force-app/main/default and open classes

Salesforce Login:

Username: vamshigutha@vamshi.com

Password: Nexthire@2000

To run back end code Create .env file in main folder and add these variables.

Env Variables:

SF_LOGINURL = https://login.salesforce.com

SF_USERNAME = vamshigutha@vamshi.com

SF_PASSWORD = Nexthire@2000

SF_TOKEN = BCNmofXSc9SAqjraUo3ym74v