Project Title: GigHub - Freelancer & Project Management Using CRM-Salesforce

Industry: Professional Services or any industry utilizing freelance talent. Problem Statement: The current process of managing freelancers with spreadsheets is chaotic , inefficient, and leads to difficulties in finding skilled talent for Resource Managers , manual handoffs from Salespeople , and a lack of real-time profitability insights for Executives.

Proposed Solution: A custom Salesforce application named "GigHub" to centralize and automate the management of freelance projects.

Key Use Cases: Automating project creation from a won deal , searching for freelancers by skill , assigning freelancers to projects , and reviewing project profitability.

Stakeholders: The primary users of the application are: Resource Manager: Finds and assigns freelancers.

Salesperson: Manages client deals in Opportunities.

Executive: Reviews dashboards for profitability.

System Administrator: Builds and maintains the system.

Core Business Process: The application will support the flow: A deal is won -> A project is created -> A freelancer with specific skills is found and assigned -> The project is completed -> Profitability is reviewed.

Phase 2: Org Setup & Configuration This phase involves preparing the Salesforce Developer Org for the application build.

Company Information: Ensure company name, address, and time zone are correctly set up .

User Profiles: Create three new profiles by cloning the "Standard User" profile:

1. Resource Manager
2. Salesperson
3. Executive

Users: Create at least three sample users, one for each new profile (e.g., Ria Manager, Sam Sales, Eva Executive) .

Role Hierarchy: Establish a simple role hierarchy and assign the sample users to their respective roles .

Phase 3: Data Modeling & Relationships This phase creates the custom objects that form the application's data foundation.

Custom Objects: Create the following custom objects, enabling reporting and activities where specified: Project: To track project details .

Freelancer: To store freelancer information .

Skill: To maintain a list of skills .

Freelancer Skill: A junction object to link Freelancers and Skills .

Fields & Relationships: On Project: Create fields for Status (Picklist), Project Revenue (Currency), Total Cost (Currency), Start Date (Date), End Date (Date), a lookup to Account (Client), and a lookup to Freelancer (Assigned Freelancer) .

On Freelancer: Create fields for Email, Hourly Rate (Currency), Status (Picklist), and a lookup to Contact .

On Freelancer Skill: Create two master-detail relationships: one to Freelancer and one to Skill .

Phase 4: Process Automation (Admin) This phase brings the application to life with declarative automation.

Flow 1 (Auto-Create Project): A record-triggered flow that fires when an Opportunity's stage is updated to "Closed Won." The flow will create a new Project record and map the Opportunity's Amount to Project Revenue and the Account to the Project's Client field .

Flow 2 (Update Freelancer Status): A record-triggered flow that fires when a Freelancer is assigned to a Project. This flow updates the Freelancer's status field to "On Project" .

Screen Flow ("Find a Freelancer" Tool): A screen flow that allows a user to select a skill from a picklist, finds all available freelancers with that skill, and displays them in a data table.

Validation Rule: A rule on the Project object that prevents the status from being changed to "Completed" if the

Assigned_Freelancer__c field is blank .

Phase 5: Apex Programming (Developer) This phase outlines potential future enhancements that would require custom code.

Potential Enhancement 1 (Apex Trigger): For more complex profitability calculations, an Apex trigger could be written on the Project object to calculate the

Total_Cost__c field.

Potential Enhancement 2 (Batch Apex): A nightly Batch Apex job could be created to automatically change a freelancer's status back to "Available" if their assigned project's end date has passed.

Phase 6: User Interface Development This phase focuses on creating a positive user experience.

Create the GigHub App: Build a new Lightning App named "GigHub." Add tabs for Home, Projects, Freelancers, Skills, Accounts, Contacts, Reports, and Dashboards to the navigation. Assign the app to the custom profiles.

Customize Page Layouts: Organize fields and add related lists (like "Freelancer Skills" on the Freelancer page) for a logical user interface .

Activate the Screen Flow: Embed the "Find a Freelancer" screen flow directly onto the Project Lightning Record Page.

Phase 7: Integration & External Access This phase outlines potential future integrations with external systems.

Potential Enhancement 1 (REST API): A custom REST API could be created to allow an external company website to pull and display a list of available skills from the Skill object.

Potential Enhancement 2 (Platform Events): Platform Events could be used to broadcast a message when a new Project is created, allowing external systems to subscribe .

Phase 8: Data Management & Deployment This phase involves populating the org with data and outlining the deployment strategy.

Data Import: Prepare sample data in CSV files for Skills, Freelancers, and Freelancer Skills. Use Data Loader to insert the records in the correct order .

Deployment Concept: All development should be done in a Sandbox org first. The completed components would then be moved to the Production org using Change Sets .

Phase 9: Reporting, Dashboards & Security Review This phase focuses on data visualization and securing the application.

Custom Report Types: Create a custom report type that links Freelancers with Freelancer Skills and Skills.

Reports: Build reports such as a "Freelancer Skill Inventory" and a "Project Status Report".

Dashboard: Create a "GigHub Management Dashboard" with charts and graphs using the reports .

Security Review: Object Permissions: Edit the custom profiles to set the correct object-level access (e.g., Executive has Read-only) .

Field-Level Security: Hide sensitive fields like

Hourly_Rate__c from profiles that do not need to see it.

Phase 10: Final Presentation & Demo Day The final phase involves preparing and delivering a demonstration of the completed application.

Prepare a Narrative: Structure the presentation as a story that explains the business problem (the chaos of spreadsheets) and presents the GigHub app as the solution .

Demonstrate the Core Process: Walk through the main business process during the demo: show a Salesperson closing a deal, see the Project automatically created, log in as the Resource Manager to use the "Find a Freelancer" tool, assign the freelancer, and finally, log in as the Executive to show the updated dashboard .

Document Work: Take screenshots of the data model, flows, and the final dashboard to use in a portfolio and for handoff documentation .

Word / .docx Version (Formatted Text)

You can copy the text below this line and paste it into Microsoft Word or Google Docs. The headings, bolding, and table will be preserved. Then, you can save it as a .docx file.

Project Plan: GigHub Salesforce Application

This document outlines the 10 phases required to build and deploy the GigHub application on the Salesforce platform.

Phase 1: Problem Understanding & Industry Analysis

This initial phase establishes the project's foundation by defining its purpose, scope, and key stakeholders.

- **Project Charter**: A formal document will be created to act as the project's "north star", outlining the following:
    - **Project Title**: GigHub
    - **Industry**: Professional Services or any industry utilizing freelance talent.
    - **Problem Statement**: The current process of managing freelancers with spreadsheets is chaotic , inefficient, and leads to difficulties in finding skilled talent for Resource Managers , manual handoffs from Salespeople , and a lack of real-time profitability insights for Executives.
    - **Proposed Solution**: A custom Salesforce application named "GigHub" to centralize and automate the management of freelance projects.
    - **Key Use Cases**: Automating project creation from a won deal , searching for freelancers by skill , assigning freelancers to projects , and reviewing project profitability.
- **Stakeholders**: The primary users of the application are:
    - **Resource Manager**: Finds and assigns freelancers.
    - **Salesperson**: Manages client deals in Opportunities.
    - **Executive**: Reviews dashboards for profitability.
    - **System Administrator**: Builds and maintains the system.
- **Core Business Process**: The application will support the flow: A deal is won -> A project is created -> A freelancer with specific skills is found and assigned -> The project is completed -> Profitability is reviewed.

Phase 2: Org Setup & Configuration

This phase involves preparing the Salesforce Developer Org for the application build.

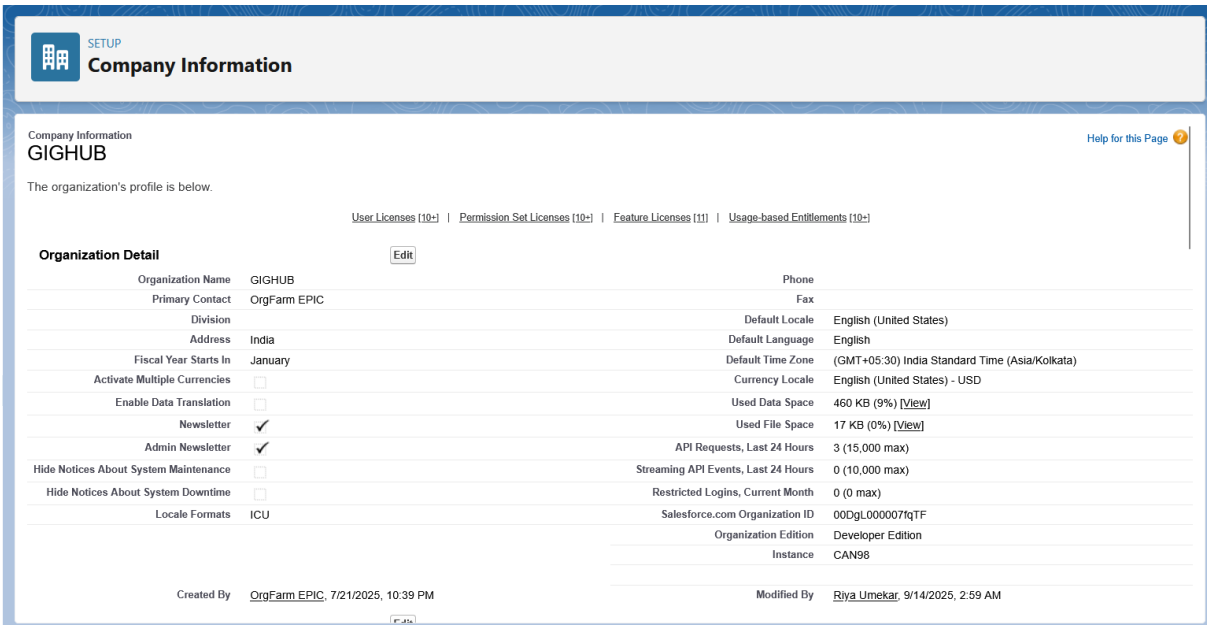
Goal

Prepare the Salesforce Developer Org for GigHub by configuring company settings, user profiles, role hierarchy, and creating sample users. This ensures the system foundation is aligned with business requirements before building objects and automation.

- **Company Information**: Ensure company name, address, and time zone are correctly set up .

Configure Company Information

- Navigate: **Setup → Company Information**.
- Update the following fields:
    1. **Company Name:** GigHub
    2. **Primary Contact Information:** Enter your details.
    3. **Default Locale:** English (India)
    4. **Currency:** INR (₹) or USD ($) depending on your target industry.
    5. **Default Time Zone:** (GMT+05:30) India Standard Time.



- **User Profiles**: Create three new profiles by cloning the "Standard User" profile:
    1. Resource Manager
    2. Salesperson
    3. Executive

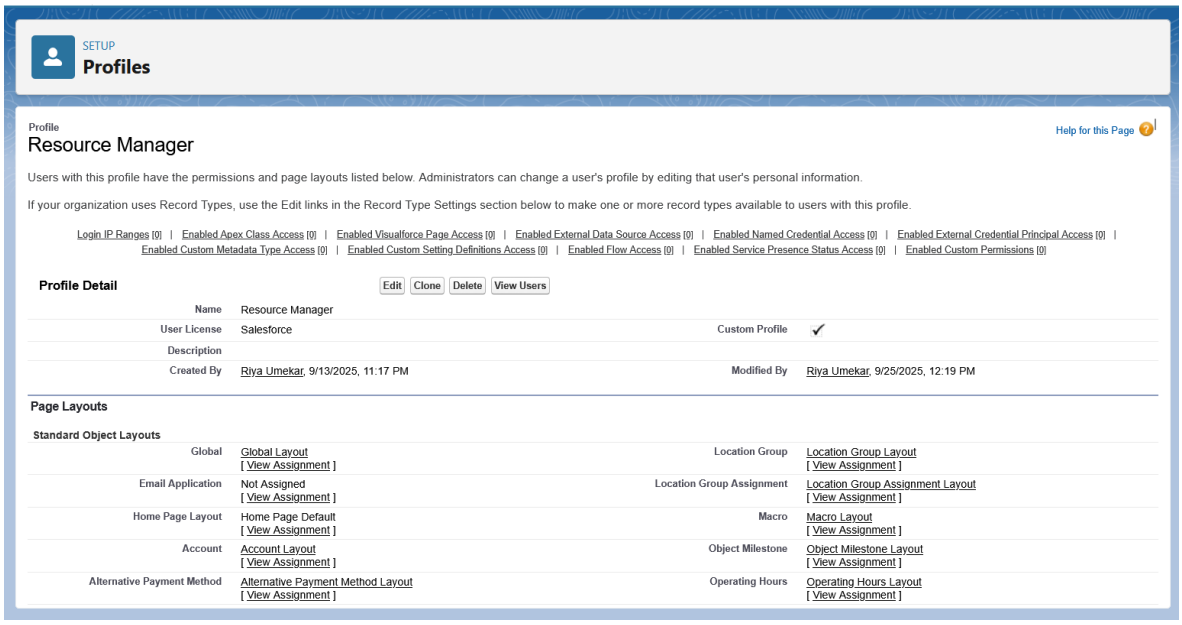
Steps:-

Navigate: **Setup → Profiles → Clone “Standard User”**.

Create three custom profiles:

- **Resource Manager Profile**
- **Salesperson Profile**
- **Executive Profile**

 Leave detailed object permissions for later (Phase 9). For now, just create and save.



SETUP
Profiles

Profile
Resource Manager

Help for this Page

Users with this profile have the permissions and page layouts listed below. Administrators can change a user's profile by editing that user's personal information.

If your organization uses Record Types, use the Edit links in the Record Type Settings section below to make one or more record types available to users with this profile.

Login IP Ranges [0] | Enabled Apex Class Access [0] | Enabled Visualforce Page Access [0] | Enabled External Data Source Access [0] | Enabled Named Credential Access [0] | Enabled External Credential Principal Access [0] | Enabled Custom Metadata Type Access [0] | Enabled Custom Setting Definitions Access [0] | Enabled Flow Access [0] | Enabled Service Presence Status Access [0] | Enabled Custom Permissions [0]

Profile Detail

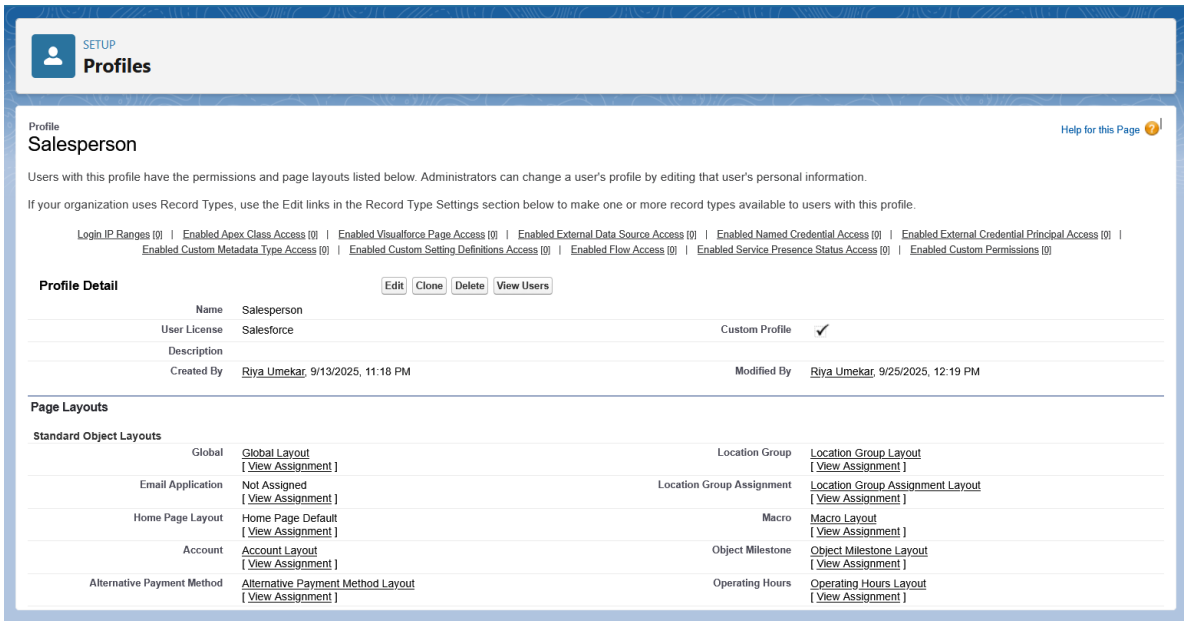
Edit | Clone | Delete | View Users

Name	Resource Manager		
User License	Salesforce	Custom Profile	✓
Description			
Created By	Riya Umekar, 9/13/2025, 11:17 PM	Modified By	Riya Umekar, 9/25/2025, 12:19 PM

Page Layouts

Standard Object Layouts

Global	Global Layout [View Assignment]	Location Group	Location Group Layout [View Assignment]
Email Application	Not Assigned [View Assignment]	Location Group Assignment	Location Group Assignment Layout [View Assignment]
Home Page Layout	Home Page Default [View Assignment]	Macro	Macro Layout [View Assignment]
Account	Account Layout [View Assignment]	Object Milestone	Object Milestone Layout [View Assignment]
Alternative Payment Method	Alternative Payment Method Layout [View Assignment]	Operating Hours	Operating Hours Layout [View Assignment]



SETUP
Profiles

Profile
Salesperson

Help for this Page

Users with this profile have the permissions and page layouts listed below. Administrators can change a user's profile by editing that user's personal information.

If your organization uses Record Types, use the Edit links in the Record Type Settings section below to make one or more record types available to users with this profile.

Login IP Ranges [0] | Enabled Apex Class Access [0] | Enabled Visualforce Page Access [0] | Enabled External Data Source Access [0] | Enabled Named Credential Access [0] | Enabled External Credential Principal Access [0] | Enabled Custom Metadata Type Access [0] | Enabled Custom Setting Definitions Access [0] | Enabled Flow Access [0] | Enabled Service Presence Status Access [0] | Enabled Custom Permissions [0]

Profile Detail

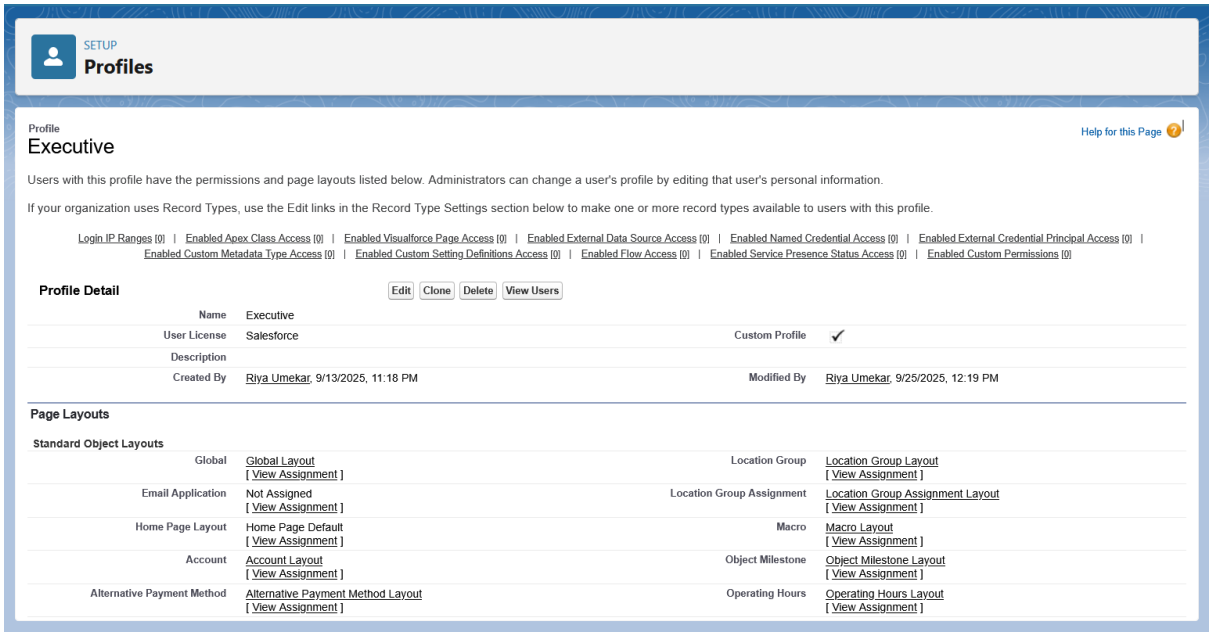
Edit | Clone | Delete | View Users

Name	Salesperson		
User License	Salesforce	Custom Profile	✓
Description			
Created By	Riya Umekar, 9/13/2025, 11:18 PM	Modified By	Riya Umekar, 9/25/2025, 12:19 PM

Page Layouts

Standard Object Layouts

Global	Global Layout [View Assignment]	Location Group	Location Group Layout [View Assignment]
Email Application	Not Assigned [View Assignment]	Location Group Assignment	Location Group Assignment Layout [View Assignment]
Home Page Layout	Home Page Default [View Assignment]	Macro	Macro Layout [View Assignment]
Account	Account Layout [View Assignment]	Object Milestone	Object Milestone Layout [View Assignment]
Alternative Payment Method	Alternative Payment Method Layout [View Assignment]	Operating Hours	Operating Hours Layout [View Assignment]



- **Users**: Create at least three sample users, one for each new profile (e.g., Ria Manager, Sam Sales, Eva Executive) .

  Steps : -

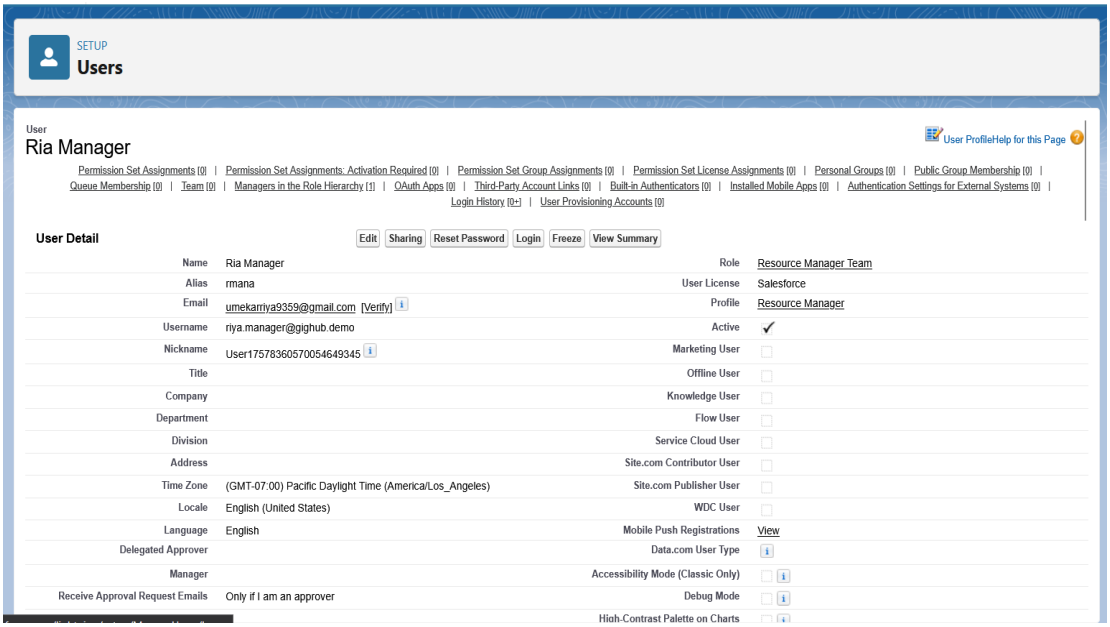
  Navigate: **Setup → Users → New User**.

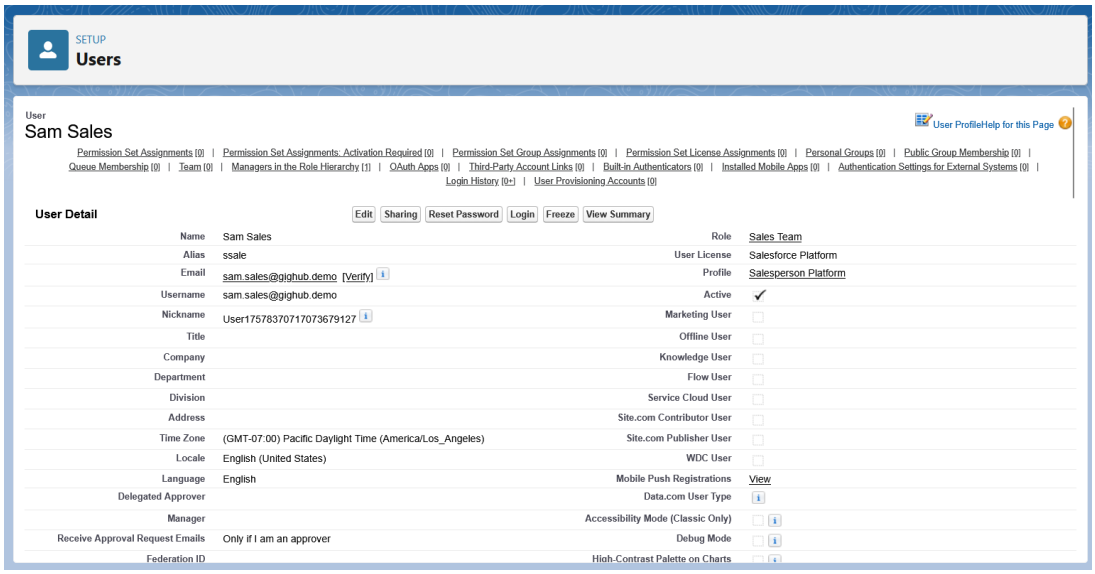
  Create at least 3 users, one for each custom profile:

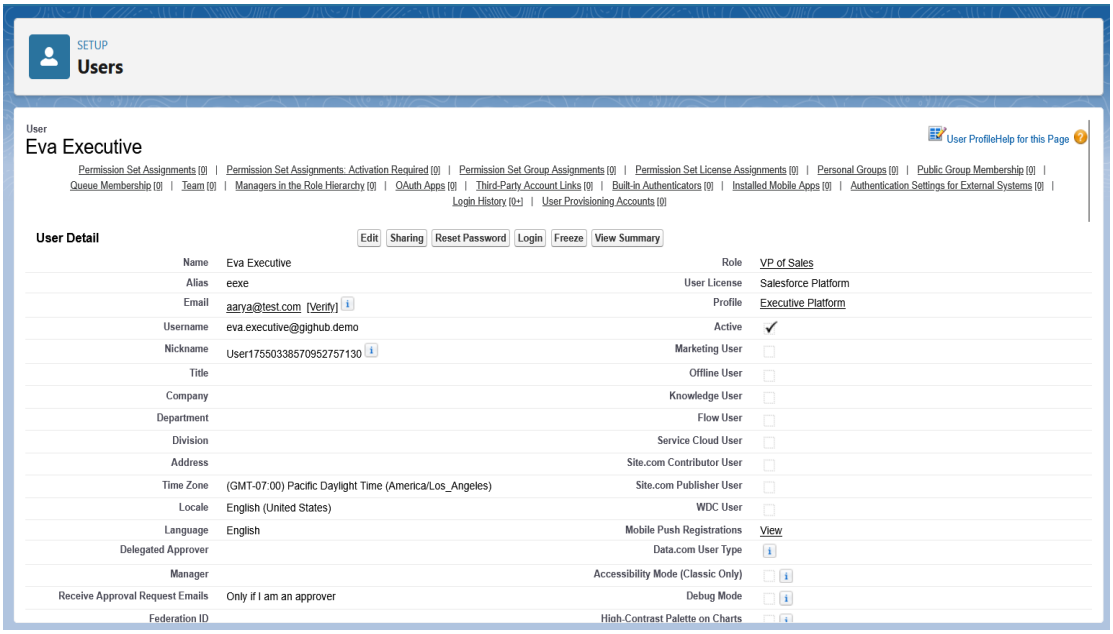
- **Ria Manager** → Profile: *Resource Manager*
- **Sam Sales** → Profile: *Salesperson*
- **Eva Executive** → Profile: *Executive*

  Assign each user a unique email (can be fake but valid format, e.g., ria.manager@test.com).

  Ensure **Active** is checked.


SETUP
Users
User
Ria Manager
User ProfileHelp for this Page
Permission Set Assignments [0] | Permission Set Assignments: Activation Required [0] | Permission Set Group Assignments [0] | Permission Set License Assignments [0] | Personal Groups [0] | Public Group Membership [0] | Queue Membership [0] | Team [0] | Managers in the Role Hierarchy [1] | OAuth Apps [0] | Third-Party Account Links [0] | Built-in Authenticators [0] | Installed Mobile Apps [0] | Authentication Settings for External Systems [0] | Login History [0+] | User Provisioning Accounts [0]
User Detail
Edit Sharing Reset Password Login Freeze View Summary
Name Ria Manager
Alias rmana
Email umekarriya9359@gmail.com [Verify]
Username riya.manager@gighub.demo
Nickname User17578360570054649345
Title
Company
Department
Division
Address
Time Zone (GMT-07:00) Pacific Daylight Time (America/Los_Angeles)
Locale English (United States)
Language English
Delegated Approver
Manager
Receive Approval Request Emails Only if I am an approver
Role Resource Manager Team
User License Salesforce
Profile Resource Manager
Active
Marketing User
Offline User
Knowledge User
Flow User
Service Cloud User
Site.com Contributor User
Site.com Publisher User
WDC User
Mobile Push Registrations View
Data.com User Type
Accessibility Mode (Classic Only)
Debug Mode
High-Contrast Palette on Charts

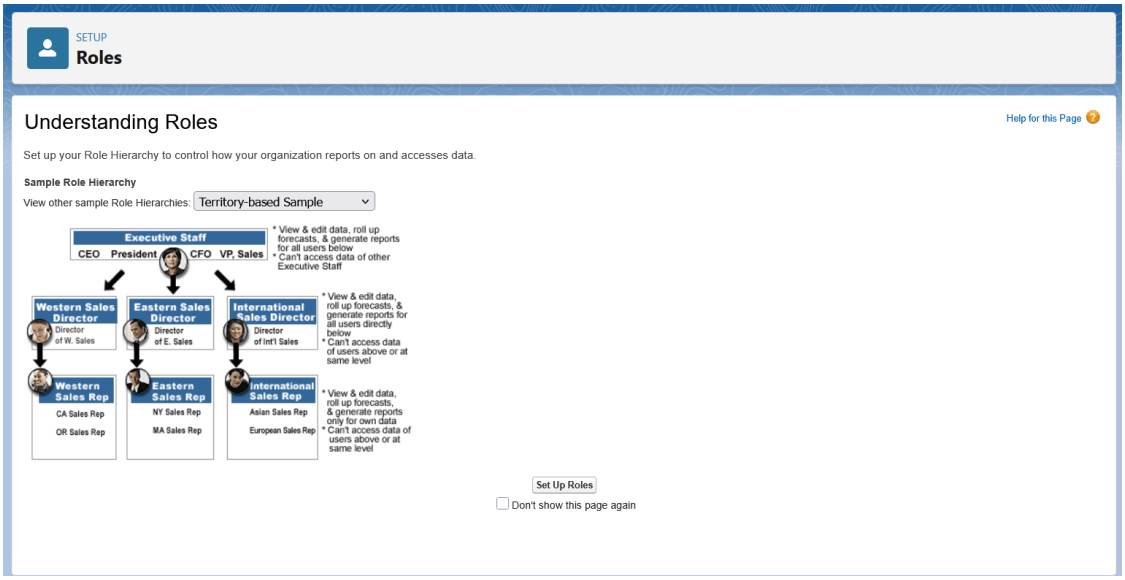

SETUP
Users
User
Sam Sales
User ProfileHelp for this Page
Permission Set Assignments [0] | Permission Set Assignments: Activation Required [0] | Permission Set Group Assignments [0] | Permission Set License Assignments [0] | Personal Groups [0] | Public Group Membership [0] | Queue Membership [0] | Team [0] | Managers in the Role Hierarchy [1] | OAuth Apps [0] | Third-Party Account Links [0] | Built-in Authenticators [0] | Installed Mobile Apps [0] | Authentication Settings for External Systems [0] | Login History [0+] | User Provisioning Accounts [0]
User Detail
Edit Sharing Reset Password Login Freeze View Summary
Name Sam Sales
Alias ssale
Email sam.sales@gighub.demo [Verify]
Username sam.sales@gighub.demo
Nickname User17578370717073679127
Title
Company
Department
Division
Address
Time Zone (GMT-07:00) Pacific Daylight Time (America/Los_Angeles)
Locale English (United States)
Language English
Delegated Approver
Manager
Receive Approval Request Emails Only if I am an approver
Federation ID
Role Sales Team
User License Salesforce Platform
Profile Salesperson Platform
Active
Marketing User
Offline User
Knowledge User
Flow User
Service Cloud User
Site.com Contributor User
Site.com Publisher User
WDC User
Mobile Push Registrations View
Data.com User Type
Accessibility Mode (Classic Only)
Debug Mode
High-Contrast Palette on Charts

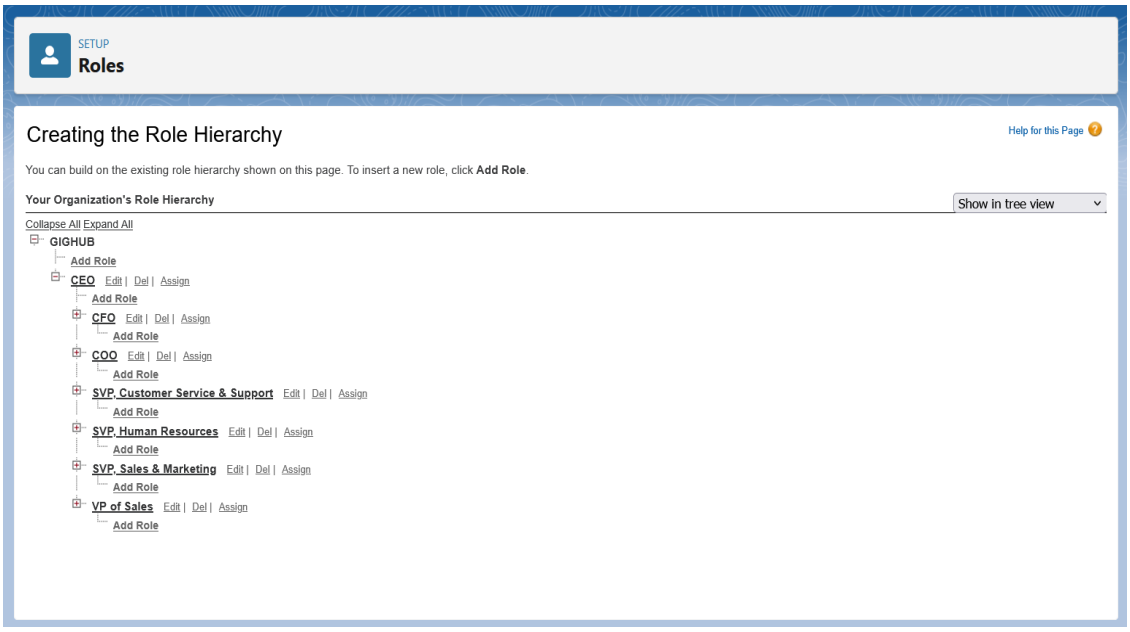

SETUP
Users
User
Eva Executive
User ProfileHelp for this Page
Permission Set Assignments [0] | Permission Set Assignments: Activation Required [0] | Permission Set Group Assignments [0] | Permission Set License Assignments [0] | Personal Groups [0] | Public Group Membership [0] | Queue Membership [0] | Team [0] | Managers in the Role Hierarchy [0] | OAuth Apps [0] | Third-Party Account Links [0] | Built-in Authenticators [0] | Installed Mobile Apps [0] | Authentication Settings for External Systems [0] | Login History [0+] | User Provisioning Accounts [0]
User Detail
Edit Sharing Reset Password Login Freeze View Summary
Name Eva Executive
Alias eexe
Email aarya@test.com [Verify]
Username eva.executive@gighub.demo
Nickname User17550338570952757130
Title
Company
Department
Division
Address
Time Zone (GMT-07:00) Pacific Daylight Time (America/Los_Angeles)
Locale English (United States)
Language English
Delegated Approver
Manager
Receive Approval Request Emails Only if I am an approver
Federation ID
Role VP of Sales
User License Salesforce Platform
Profile Executive Platform
Active
Marketing User
Offline User
Knowledge User
Flow User
Service Cloud User
Site.com Contributor User
Site.com Publisher User
WDC User
Mobile Push Registrations View
Data.com User Type
Accessibility Mode (Classic Only)
Debug Mode
High-Contrast Palette on Charts

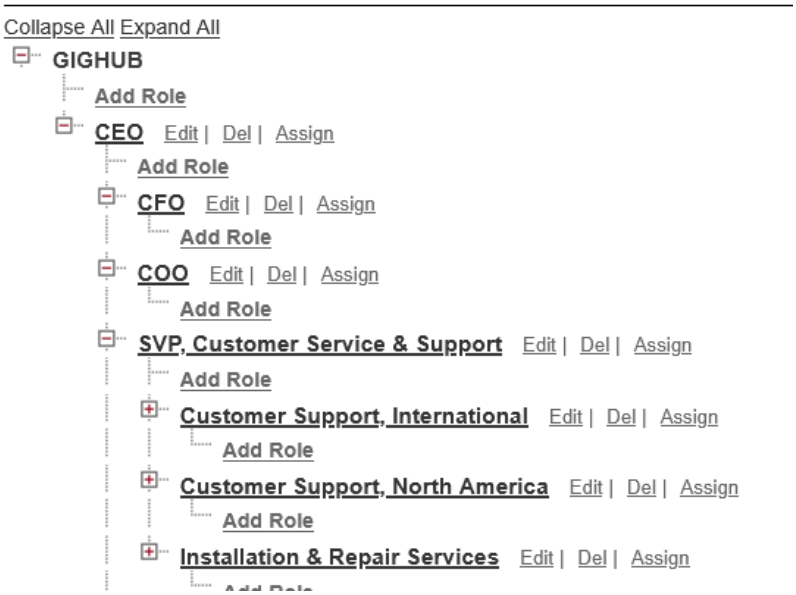
- **Role Hierarchy**: Establish a simple role hierarchy and assign the sample users to their respective roles .

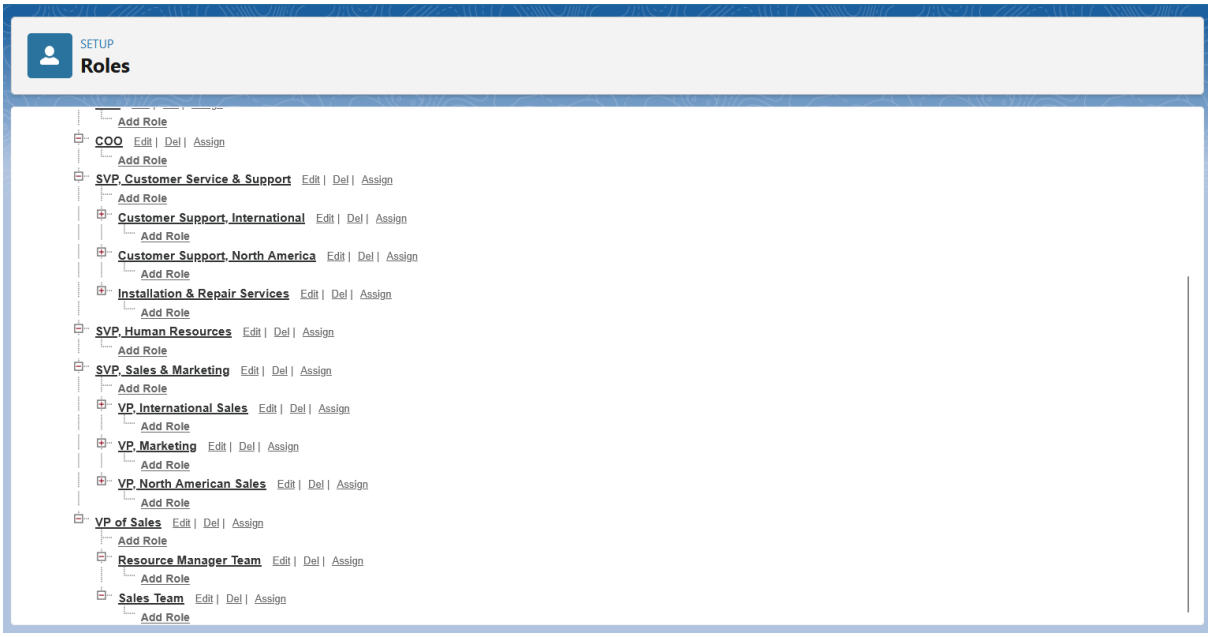
Purpose of Role Hierarchy

- Determines **record-level visibility** for users.
- Users at a **higher role** automatically see records owned by users **below them** in the hierarchy.
- **Setup → Quick Find → Roles → Set Up Roles.**









- In the GigHub project:
    - **Executives** need visibility into all data.
    - **Salespeople** only need to see their opportunities and projects.
    - **Resource Managers** manage freelancer assignments under VP of Services.

Phase 3: Data Modeling & Relationships

This phase creates the custom objects that form the application's data foundation.

- **Custom Objects**: Create the following custom objects, enabling reporting and activities where specified:
    - **Project**: To track project details .
    - **Freelancer**: To store freelancer information .
    - **Skill**: To maintain a list of skills .
    - **Freelancer Skill**: A junction object to link Freelancers and Skills .
- **Fields & Relationships**:
    - **On Project**: Create fields for Status (Picklist), Project Revenue (Currency), Total Cost (Currency), Start Date (Date), End Date (Date), a lookup to Account (Client), and a lookup to Freelancer (Assigned Freelancer) .
    - **On Freelancer**: Create fields for Email, Hourly Rate (Currency), Status (Picklist), and a lookup to Contact .
    - **On Freelancer Skill**: Create two master-detail relationships: one to Freelancer and one to Skill .

Introduction to Data Modeling

This phase is the architectural foundation of the GigHub application. Here, we will create the custom objects that will store our data and define the relationships between them. A well-designed data model is crucial for building scalable automation, intuitive user interfaces, and meaningful reports. We will construct four custom objects: Project, Freelancer, Skill, and a junction object Freelancer Skill to link them.

3.2 Step-by-Step Custom Object Creation

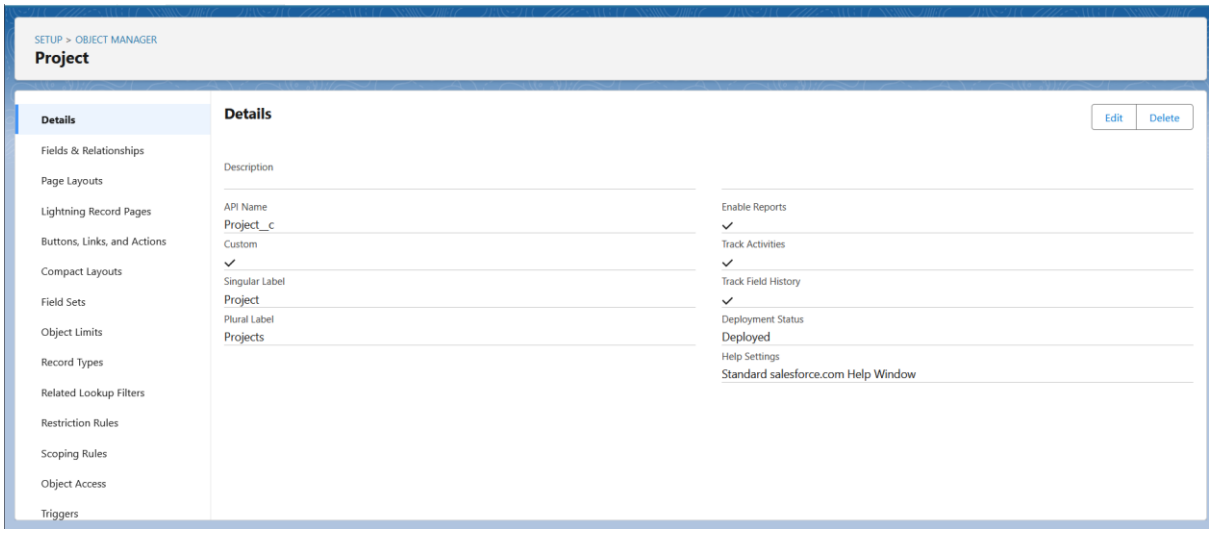
We will create each of the four custom objects required for the application.

Create the Project Object

1. Navigate to **Setup** > **Object Manager**.
2. Click **Create** in the top-right corner and select **Custom Object**.
3. Enter the following details:
    - **Label:** Project
    - **Plural Label:** Projects
    - **Object Name:** Project
    - **Record Name:** Project Name
    - **Data Type:** Text

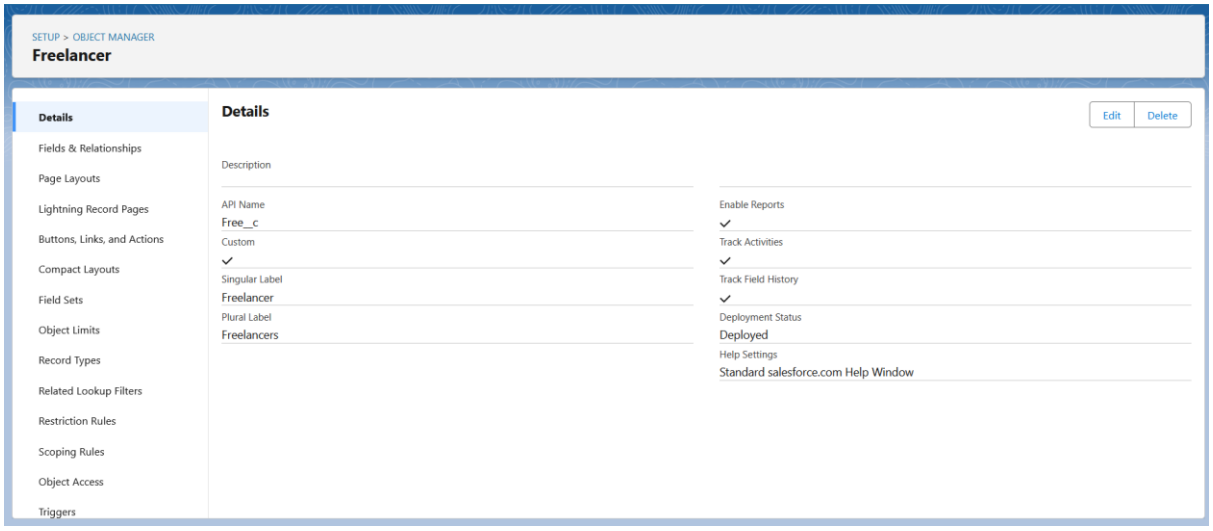
- Under **Optional Features**, check **Allow Reports** and **Allow Activities**.

4. Click **Save**.



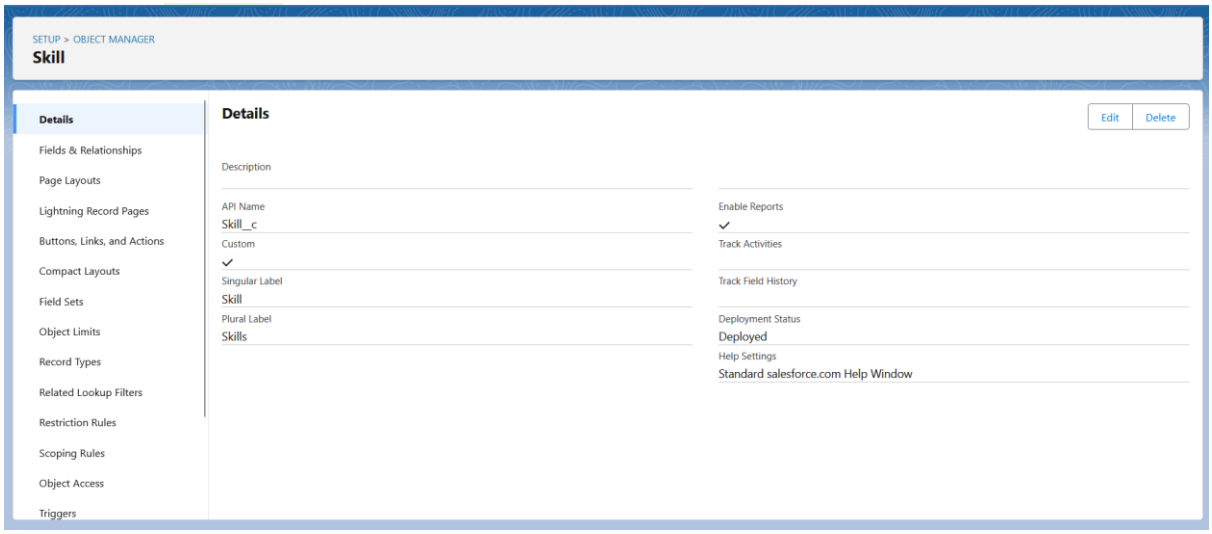
Create the Freelancer Object

1. In **Object Manager**, click **Create** > **Custom Object**.
2. Enter the following details:
    - **Label:** Freelancer
    - **Plural Label:** Freelancers
    - **Object Name:** Freelancer
    - **Record Name:** Freelancer Name
    - **Data Type:** Text
    - Under **Optional Features**, check **Allow Reports** and **Allow Activities**.
3. Click **Save**.



Create the Skill Object

1. In **Object Manager**, click **Create** > **Custom Object**.
2. Enter the following details:
    - **Label:** Skill
    - **Plural Label:** Skills
    - **Object Name:** Skill
    - **Record Name:** Skill Name
    - **Data Type:** Text
    - Under **Optional Features**, check **Allow Reports**.
3. Click **Save**.



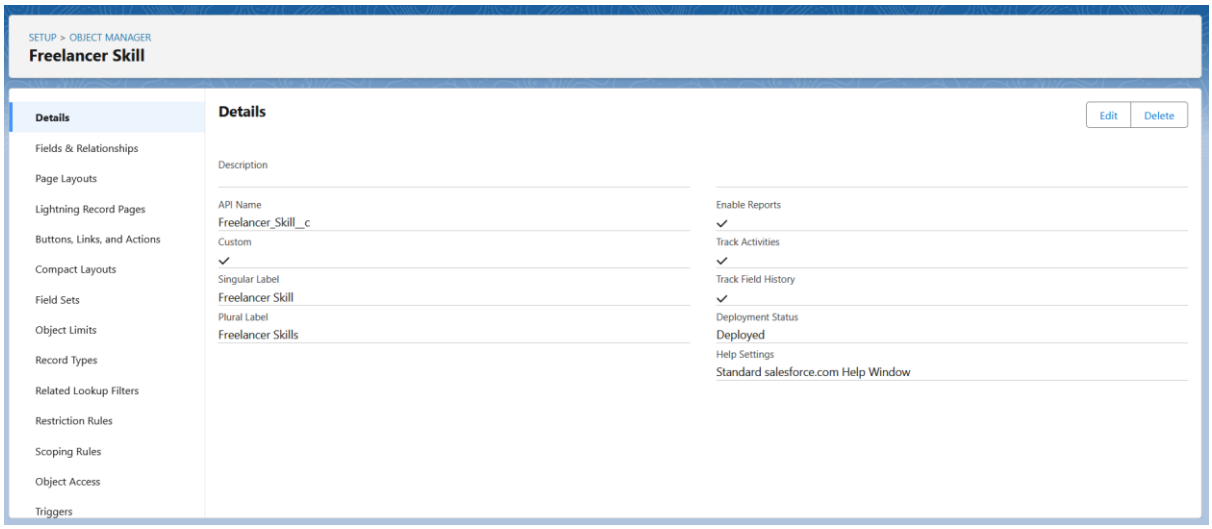
Create the Freelancer Skill (Junction) Object

This object will connect Freelancers to Skills in a many-to-many relationship.

1. In **Object Manager**, click **Create** > **Custom Object**.
2. Enter the following details:
    - **Label:** Freelancer Skill
    - **Plural Label:** Freelancer Skills
    - **Object Name:** Freelancer_Skill
    - **Record Name:** Freelancer Skill ID
    - **Data Type:** Auto Number
    - **Display Format:** FS-{0000}
    - **Starting Number:** 1

- Under **Optional Features**, check **Allow Reports**.

3. Click **Save**.



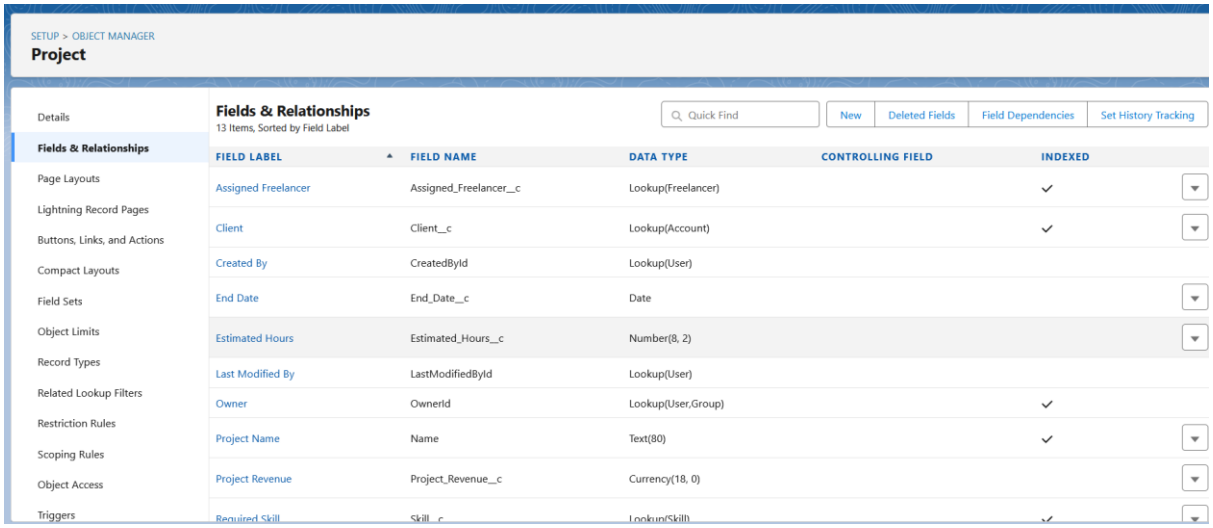
Detailed Field and Relationship Configuration

add the necessary custom fields and relationships to each object.

Fields for the Project Object

1. Navigate to **Object Manager** > **Project**.
2. Go to **Fields & Relationships** and click **New**.
3. Create the following fields, one by one:

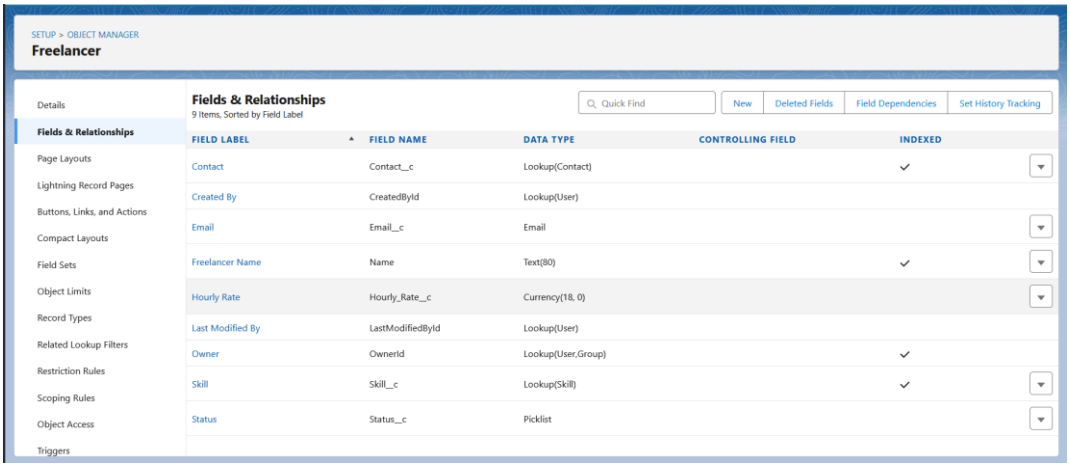
Field Label	Data Type	Details / Picklist Values
Status	Picklist	Values: Not Started, In Progress, Completed, On Hold
Project Revenue	Currency	Length: 16, Decimal Places: 2
Total Cost	Currency	Length: 16, Decimal Places: 2
Start Date	Date	
End Date	Date	
Client	Lookup	Related To: Account
Assigned Freelancer	Lookup	Related To: Freelancer



Fields for the Freelancer Object

1. Navigate to **Object Manager** > **Freelancer**.
2. Go to **Fields & Relationships** and click **New**.
3. Create the following fields:

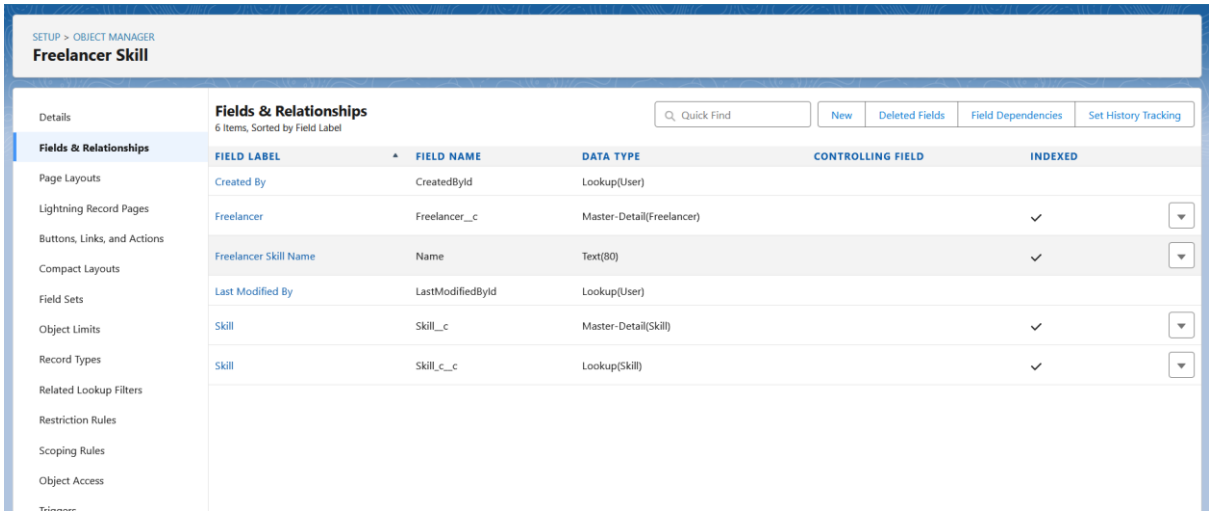
Field Label	Data Type	Details / Picklist Values
Email	Email	Mark as Required and Unique
Hourly Rate	Currency	Length: 16, Decimal Places: 2
Status	Picklist	Values: Available, On Project, Unavailable
Contact Record	Lookup	Related To: Contact



Relationships for the Freelancer Skill Object

This is the most critical step, where we create the junction.

1. Navigate to **Object Manager** > **Freelancer Skill**.
2. Go to **Fields & Relationships** and click **New**.
3. **Create the Master-Detail Relationship to Freelancer:**
    - Select Data Type: **Master-Detail Relationship**.
    - Related To: **Freelancer**.
    - Click Next through the steps, accepting the defaults for Field-Level Security and Page Layouts. Click **Save**.
4. **Create the Master-Detail Relationship to Skill:**
    - Go back to **Fields & Relationships** and click **New**.
    - Select Data Type: **Master-Detail Relationship**.
    - Related To: **Skill**.
    - Click Next through the steps, accepting defaults. Click **Save**.

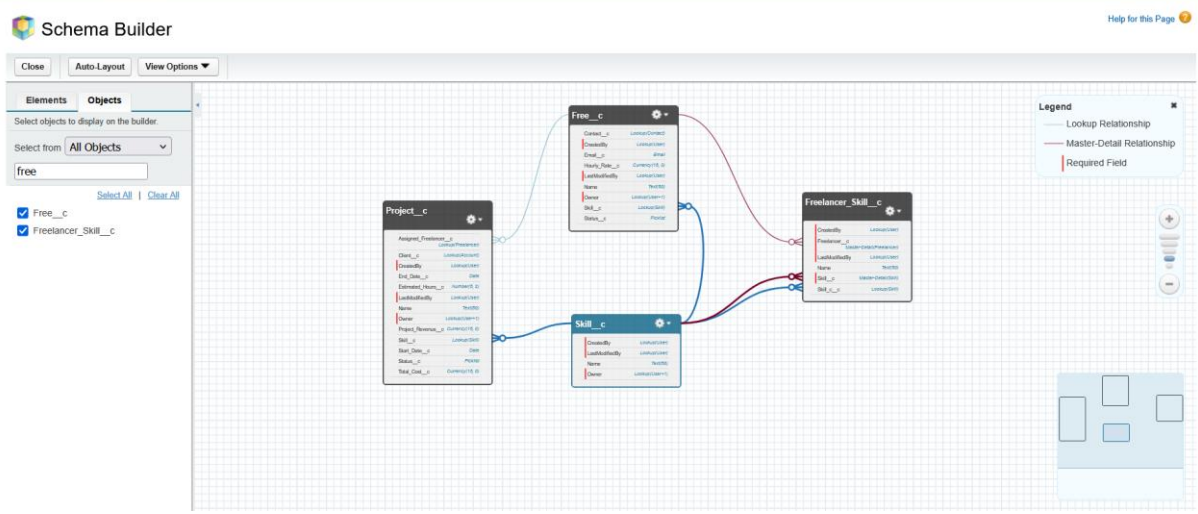


Visualizing the Model with Schema Builder

After creating all objects and relationships, it's essential to visualize the data model to confirm it's structured correctly.

1. Navigate to **Setup**.
2. In the Quick Find box, type Schema Builder and open it.
3. On the left panel, click **Clear All**.

4. Find and check the boxes for our four custom objects: Project, Freelancer, Skill, and Freelancer Skill.
5. Also check the standard objects we linked to: Account and Contact.
6. Rearrange the objects on the canvas to clearly see the relationships:
    - The Freelancer Skill object should be between Freelancer and Skill, with red lines (Master-Detail) connecting them.
    - The Project object should have blue lines (Lookup) connecting to Account and Freelancer.



Phase 4: Process Automation (Admin)

This phase brings the application to life with declarative automation.

- **Flow 1 (Auto-Create Project)**: A record-triggered flow that fires when an Opportunity's stage is updated to "Closed Won." The flow will create a new Project record and map the Opportunity's Amount to Project Revenue and the Account to the Project's Client field .
- **Flow 2 (Update Freelancer Status)**: A record-triggered flow that fires when a Freelancer is assigned to a Project. This flow updates the Freelancer's status field to "On Project" .
- **Screen Flow ("Find a Freelancer" Tool)**: A screen flow that allows a user to select a skill from a picklist, finds all available freelancers with that skill, and displays them in a data table.
- **Validation Rule**: A rule on the Project object that prevents the status from being changed to "Completed" if the Assigned_Freelancer__c field is blank .

With a solid data model in place, we will now build the automations that drive efficiency. This phase focuses on creating the "magic" that happens behind the scenes, eliminating manual data entry and ensuring processes are followed correctly. We will build two record-triggered flows for background updates, one interactive screen flow for user-guided actions, and a validation rule to maintain data integrity.

Flow 1: Automated Project Creation from Opportunity

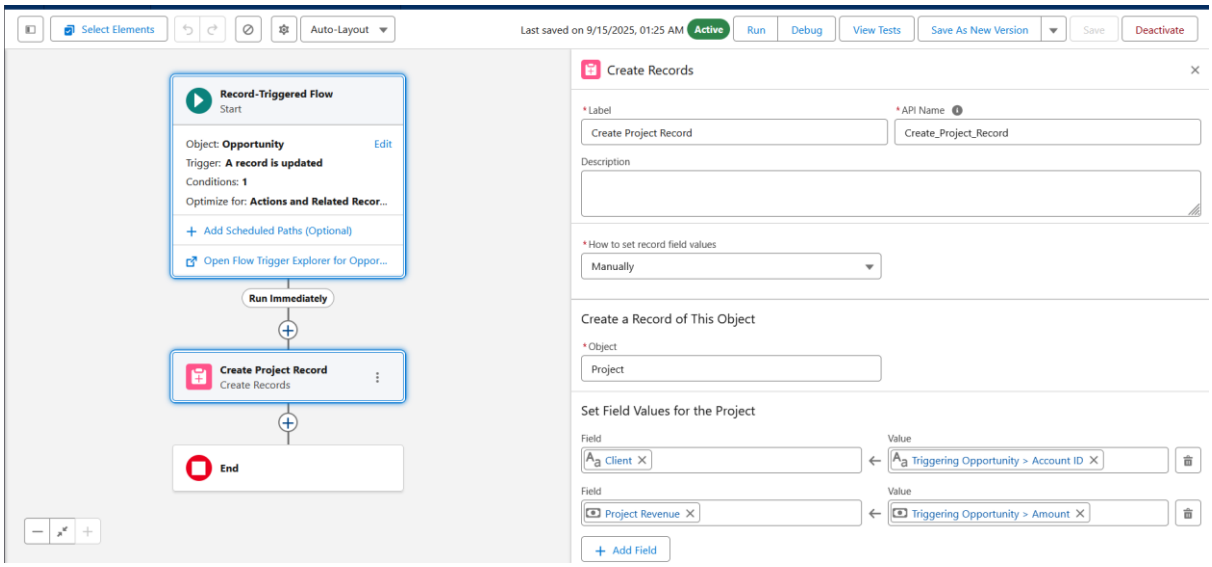
Goal: Automatically create a new Project record when a Salesperson changes an Opportunity stage to "Closed Won". This eliminates the manual handoff from Sales to the Resource Management team.

1. Navigate to **Setup** > In the Quick Find box, type Flows > Click **Flows**.
2. Click **New Flow** and select **Record-Triggered Flow**. Click **Create**.
3. **Configure the Trigger:**
    - **Object:** Opportunity
    - **Trigger the Flow When:** A record is updated
    - **Set Entry Conditions:**
        - Condition Requirements: All Conditions Are Met (AND)
        - Field: StageName
        - Operator: Equals
        - Value: Closed Won
    - **When to Run the Flow for Updated Records:** Only when a record is updated to meet the condition requirements.
    - **Optimize the Flow for:** Actions and Related Records.
4. On the flow canvas, click the **+** icon and select the **Create Records** element.
5. **Configure the Create Records Element:**
    - **Label:** Create Project
    - **How Many Records to Create:** One
    - **How to Set the Record Fields:** Use separate resources, and literal values
    - **Object:** Project
    - **Set Field Values for the Project:**
        - Field: Name -> Value: $Record > Name
        - Field: Project_Revenue__c -> Value: $Record > Amount
        - Field: Client__c -> Value: $Record > Account ID
        - Field: Status__c -> Value: Not Started

Screenshot Opportunity 7:

- **What to capture:** The "Create Records" element configuration screen, showing all four field mappings from the Opportunity ($Record) to the new Project.
- **Placement:** Insert the screenshot directly below this step.

6. Click **Done**. Click **Save**.
7. **Flow Label:** Opportunity - After Update - Create Project on Closed Won
8. Click **Save**, then click **Activate**.

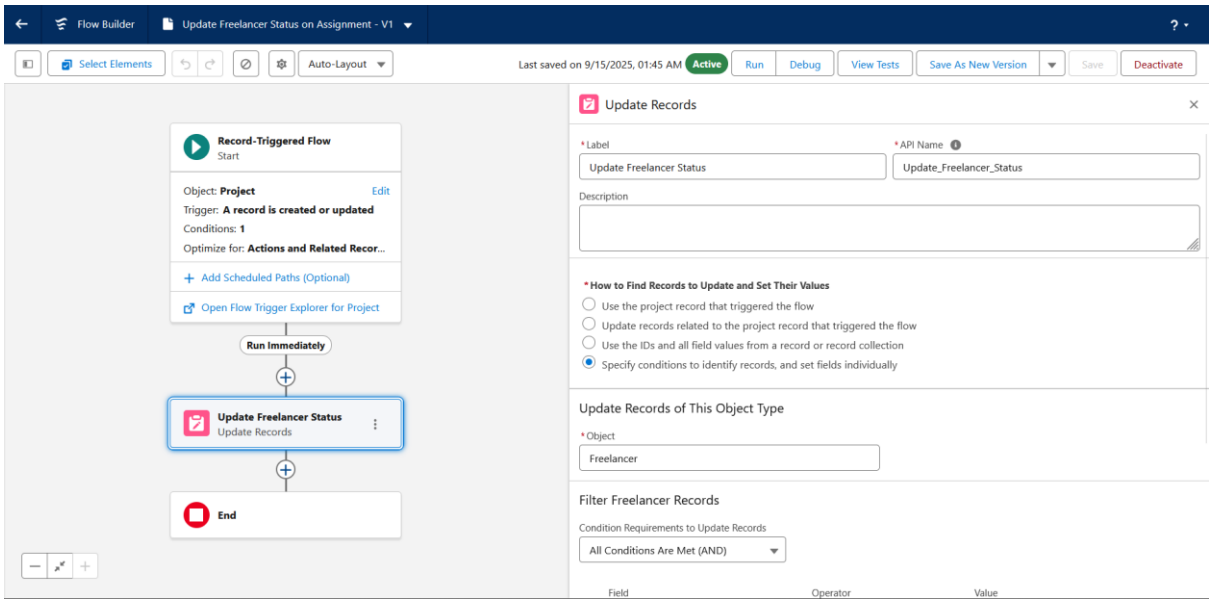


Flow 2: Automated Freelancer Status Update

Goal: When a Resource Manager assigns a Freelancer to a Project, automatically update that Freelancer's Status field to "On Project".

1. From the Flows setup page, click **New Flow** and select **Record-Triggered Flow**.
2. **Configure the Trigger:**
    - **Object:** Project
    - **Trigger the Flow When:** A record is updated
    - **Set Entry Conditions:**
        - Field: Assigned_Freelancer__c
        - Operator: Is Null
        - Value: {!$GlobalConstant.False} (This ensures the flow runs only when a freelancer is assigned, not when they are removed).
    - **When to Run the Flow for Updated Records:** Only when a record is updated to meet the condition requirements.
    - **Optimize the Flow for:** Actions and Related Records.

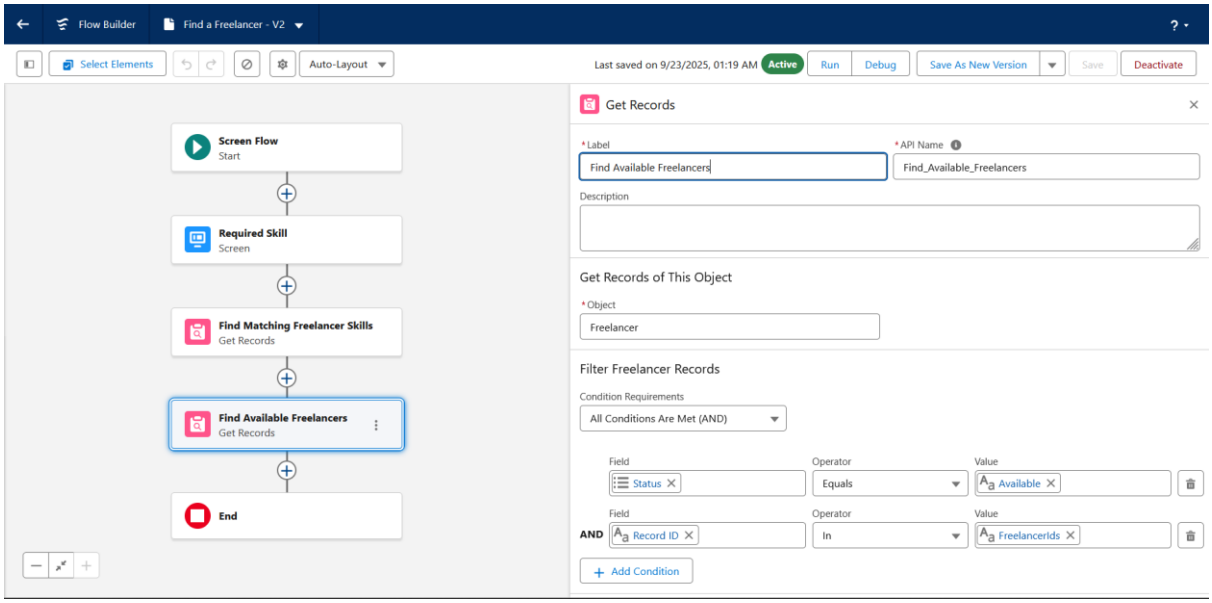
3. Click the **+** icon on the canvas and select the **Update Records** element.
4. **Configure the Update Records Element:**
    - **Label:** Update Freelancer Status to On Project
    - **How to Find Records to Update and Set Their Values:** Specify conditions to identify records, and set fields individually
    - **Object:** Freelancer
    - **Condition Requirements to Update Record:** All Conditions Are Met (AND)
        - Field: Id
        - Operator: Equals
        - Value: $Record > Assigned Freelancer > Id
    - **Set Field Values for the Freelancer Records:**
        - Field: Status__c
        - Value: On Project
5. Click **Done**. Click **Save**.
6. **Flow Label:** Project - After Update - Update Assigned Freelancer Status
7. Click **Save**, then click **Activate**.



Screen Flow: "Find a Freelancer" Interactive Tool

Goal: Create a tool for the Resource Manager on the Project page to find available freelancers based on a specific skill.

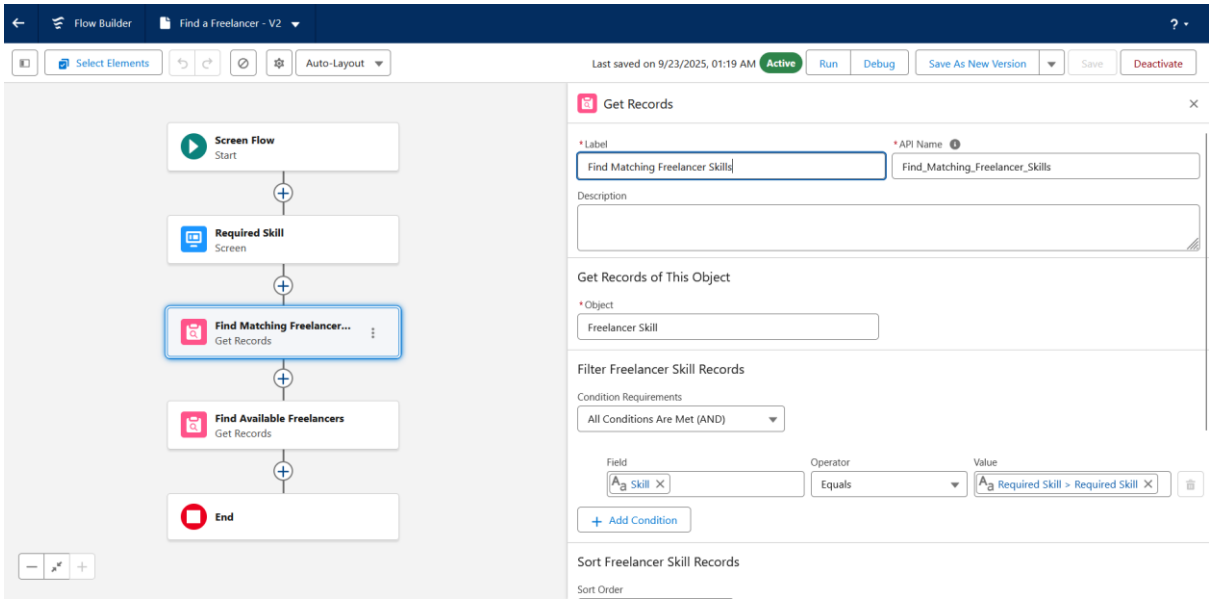
1. From the Flows setup page, click **New Flow** and select **Screen Flow**.
2. **Step 1 - Add a Screen:**
    - Drag a **Screen** element onto the canvas.
    - **Screen Properties Label:** Select Skill
    - From the Components panel, drag a **Picklist** component onto the screen.
    - **Label:** Select a Skill
    - Click in the **Choice** field and click **+ New Choice Resource**.
        - **Resource Type:** Record Choice Set
        - **API Name:** Get_All_Skills
        - **Object:** Skill
        - **Choice Label:** Name
        - **Data Type:** Text
        - **Choice Value:** Id
        - Click **Done**.
    - Click **Done** to close the screen configuration.



3. **Step 2 - Find Freelancers with that Skill:**

- Click the **+** icon and add a **Get Records** element.
- **Label:** Get Freelancer Skills
- **Object:** Freelancer Skill
- **Condition Requirements:**
  - Field: Skill__c
  - Operator: Equals
  - Value: Select_a_Skill (from Screen Components)
- **How Many Records to Store:** All records
- Click **Done**.

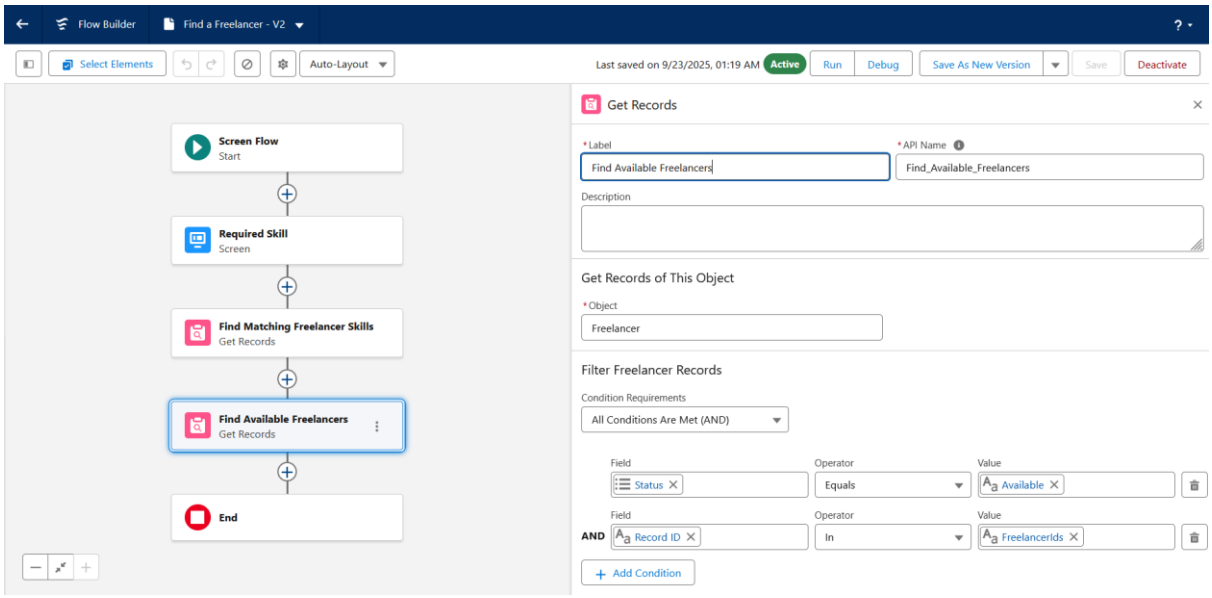
4. *(Steps for looping and collecting IDs would go here for a more advanced version, but for simplicity, we will show a list of the junction records first).*



5. **Step 3 - Display the Results:**
   - Click the **+** icon and add a **Screen** element.
   - **Screen Properties Label:** Available Freelancers
   - From the Components panel, drag a **Data Table** component onto the screen.
   - **API Name:** Available_Freelancers_Table
   - **Label:** Freelancers with Selected Skill
   - **Source Collection:** {!Get_Freelancer_Skills} (the Get Records element from the previous step).
   - **Configure Columns:**
     - Click **+ Add Column**. Source Field: Freelancer__r.Name (This traverses the relationship to get the name).

- Click **+ Add Column**. Source Field: Freelancer__r.Hourly_Rate__c.
- Click **+ Add Column**. Source Field: Freelancer__r.Status__c.

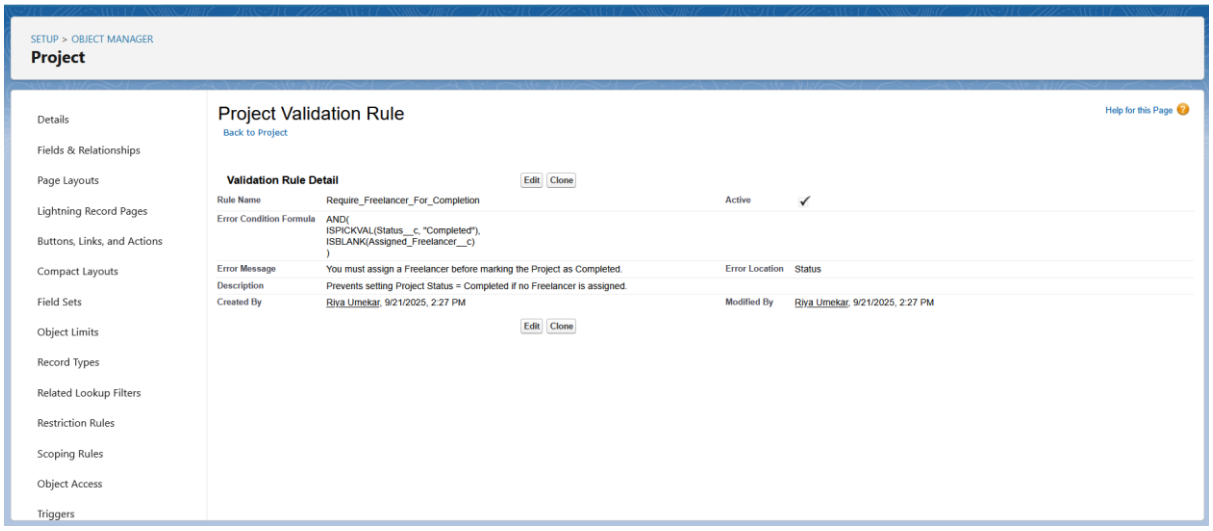
6. Click **Done**. Click **Save**.
7. **Flow Label:** Screen Flow - Find a Freelancer by Skill
8. Click **Save**, then click **Activate**. We will add this to the page layout in Phase 6.



Validation Rule: Ensure Project Staffing Before Completion

Goal: Prevent a user from changing a Project's status to "Completed" if no freelancer has been assigned. This enforces a key business process and maintains data integrity.

1. Navigate to **Setup** > **Object Manager** > **Project**.
2. From the left-hand menu, click **Validation Rules**, then click **New**.
3. **Configure the Validation Rule:**
    - **Rule Name:** Project_Must_Have_Freelancer_When_Completed
    - **Error Condition Formula:**
    - AND(
    - ISPICKVAL(Status__c, "Completed"),
    - ISBLANK(Assigned_Freelancer__c)
    -)



Phase 5: Apex Programming (Developer)

This phase outlines potential future enhancements that would require custom code.

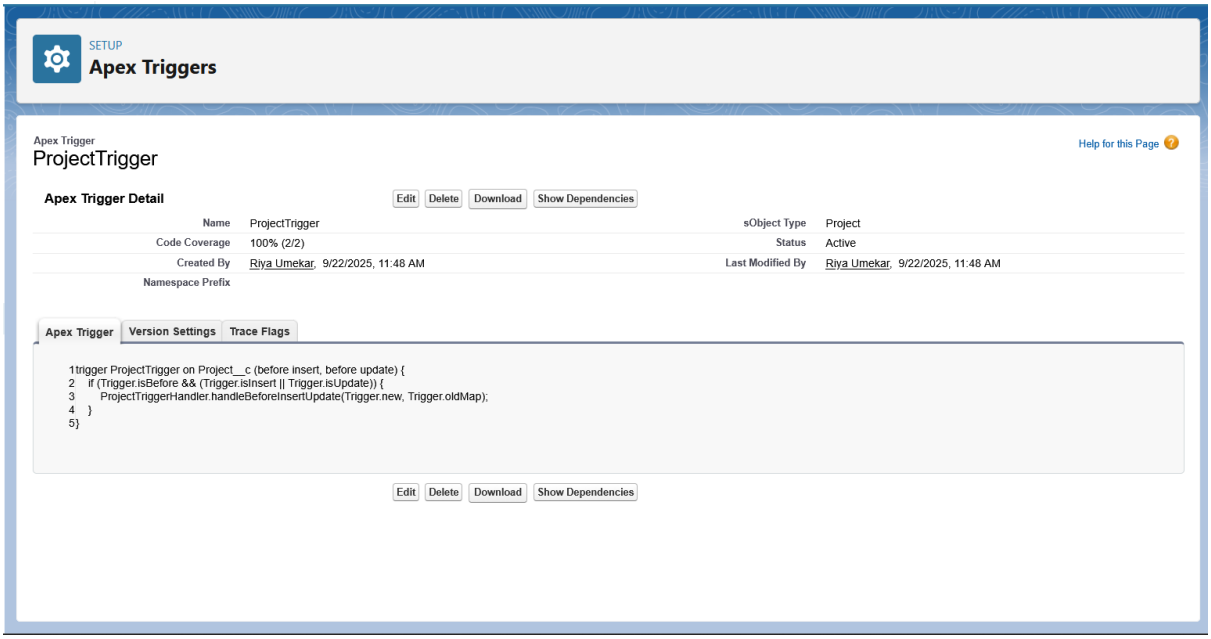
- **Potential Enhancement 1 (Apex Trigger)**: For more complex profitability calculations, an Apex trigger could be written on the Project object to calculate the Total_Cost__c field.
- **Potential Enhancement 2 (Batch Apex)**: A nightly Batch Apex job could be created to automatically change a freelancer's status back to "Available" if their assigned project's end date has passed.
- This phase outlines potential future enhancements that require custom Apex code. Apex is used when business logic is too complex for declarative tools like Flow, requires higher performance, or needs to process very large sets of data. The items in this phase are not part of the initial Minimum Viable Product (MVP) but represent logical next steps for a more mature version of the GigHub application.
-

Potential Enhancement 1: Advanced Profitability Calculation (Apex Trigger)

Goal: To automatically calculate the Total_Cost__c on a Project in real-time based on the assigned freelancer's hourly rate and the project's duration. This is better suited for an Apex Trigger than a Flow when the calculation is complex (e.g., accounting for weekends/holidays) and needs to fire instantly on save with high performance.

1. Navigate to Setup > Developer Console (or open your preferred IDE like VS Code).
2. In the Developer Console, click File > New > Apex Trigger.
3. Create the Trigger:
    - Name: ProjectCostTrigger
    - sObject: Project
    - Click Submit.

4. Replace the default code

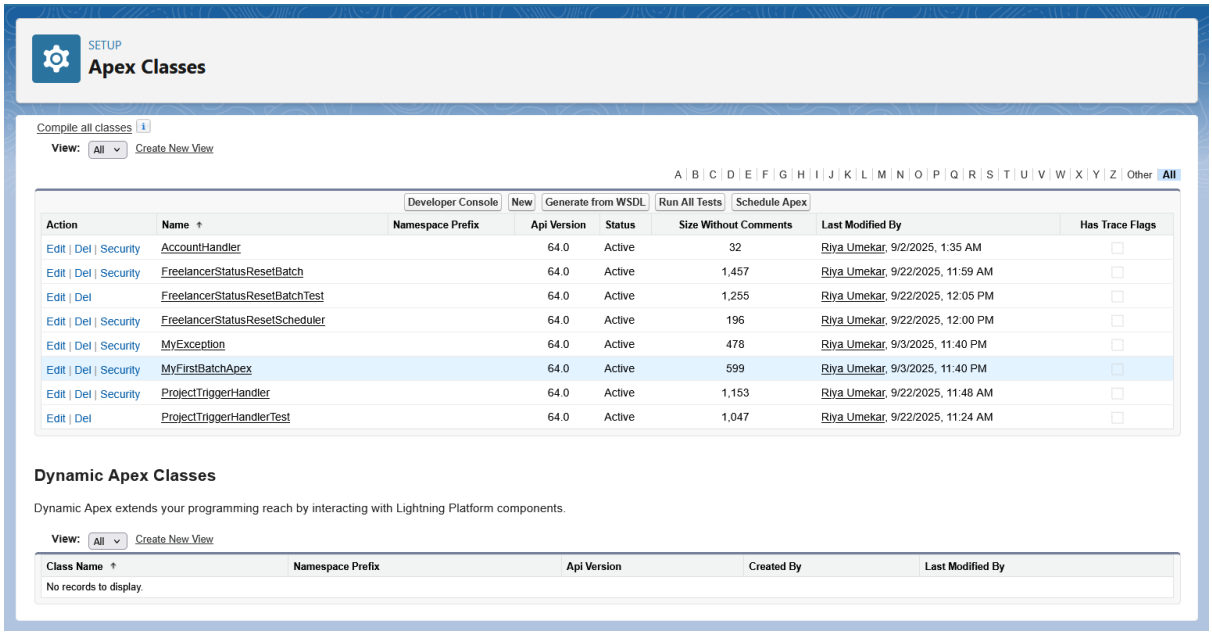


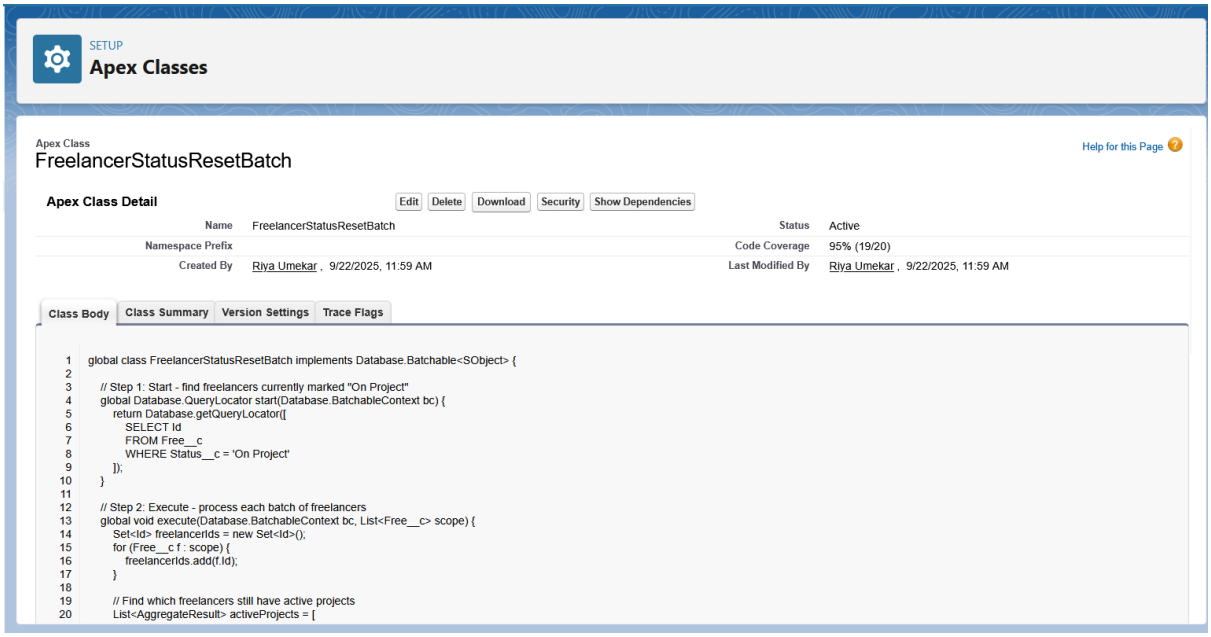
Potential Enhancement 2: Automated Freelancer Availability (Batch Apex)

Goal: To create a nightly scheduled job that automatically changes a freelancer's status back to "Available" if their assigned project's end date has passed. Batch Apex is the correct tool for this because it's designed to process thousands of records efficiently without hitting governor limits.

Step 1: Create the Batch Apex Class

1. In the Developer Console, click File > New > Apex Class.
2. Name: UpdateFreelancerStatusBatch and click OK.
3. Replace the default code





Phase 6: User Interface Development

This phase focuses on creating a positive user experience.

- **Create the GigHub App**: Build a new Lightning App named "GigHub." Add tabs for Home, Projects, Freelancers, Skills, Accounts, Contacts, Reports, and Dashboards to the navigation. Assign the app to the custom profiles.
- **Customize Page Layouts**: Organize fields and add related lists (like "Freelancer Skills" on the Freelancer page) for a logical user interface .
- **Activate the Screen Flow**: Embed the "Find a Freelancer" screen flow directly onto the Project Lightning Record Page.

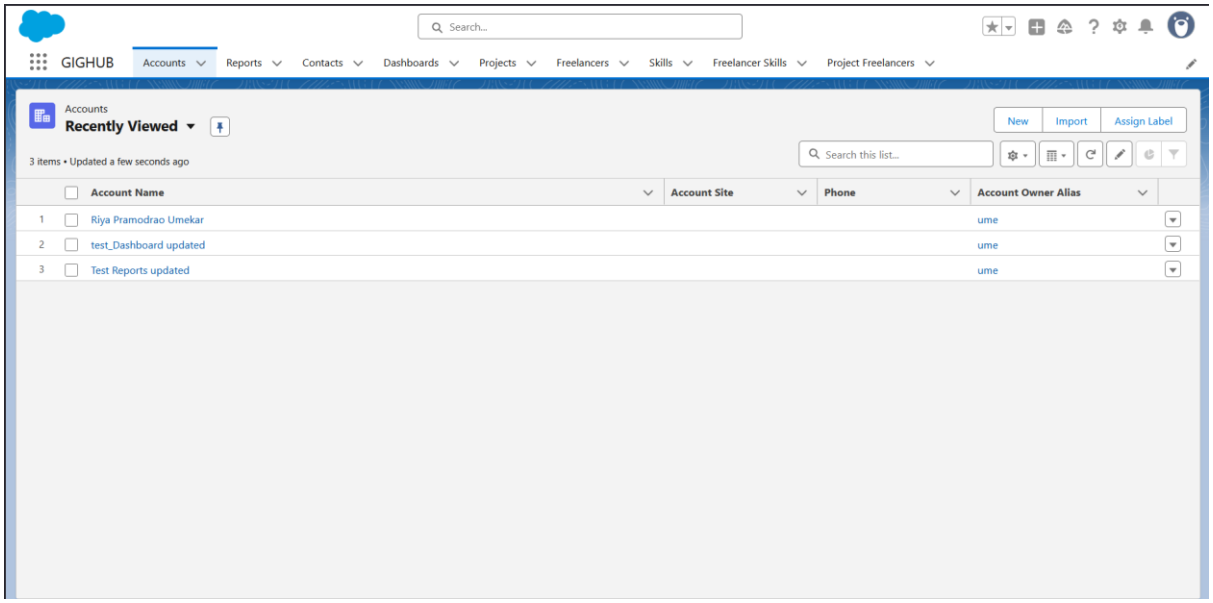
Goal of UI/UX Development

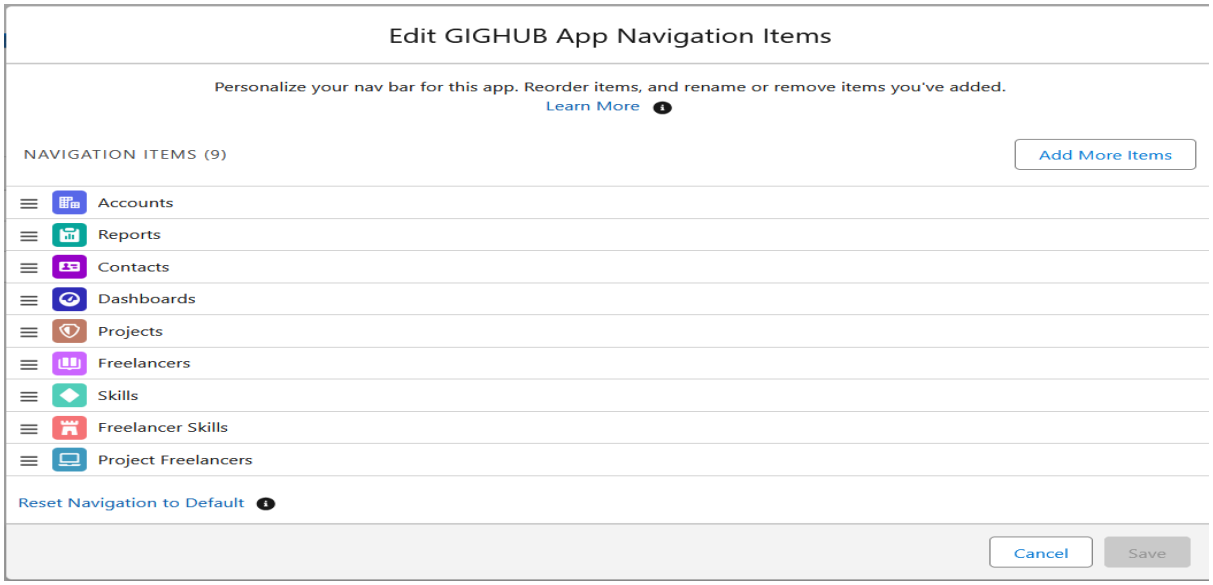
This phase translates our backend configuration and automation into a tangible, user-friendly interface. A good UI ensures high user adoption by making the application intuitive and tailored to the specific needs of each stakeholder. We will create a dedicated app for GigHub, clean up the data-entry pages, and surface our powerful automation tools.

Building the "GigHub" Lightning App

Goal: To create a branded, consolidated space for users to access all the objects and tools related to freelancer management.

1. Navigate to **Setup**. In the Quick Find box, type App Manager and select it.
2. Click the **New Lightning App** button in the top right.
3. **App Details & Branding:**
    - **App Name:** GigHub
    - **Developer Name:** GigHub
    - **Description:** Manage freelance talent and projects.
    - **App Branding:** You can upload a logo and select a primary color. Choose a distinct color to make the app easily identifiable. Click **Next**.
4. **App Options:**
    - Select **Standard navigation**.
    - Leave other settings as default and click **Next**.
5. **Utility Items:** We will not add any for now. Click **Next**.
6. **Navigation Items:** This is where you select the tabs for the app.
    - From the 'Available Items' list, select the following and use the arrow to move them to the 'Selected Items' list:
        - Home
        - Projects
        - Freelancers
        - Skills
        - Accounts
        - Contacts
        - Reports
        - Dashboards
    - You can reorder the tabs in the 'Selected Items' list. A logical order is recommended. Click **Next**.





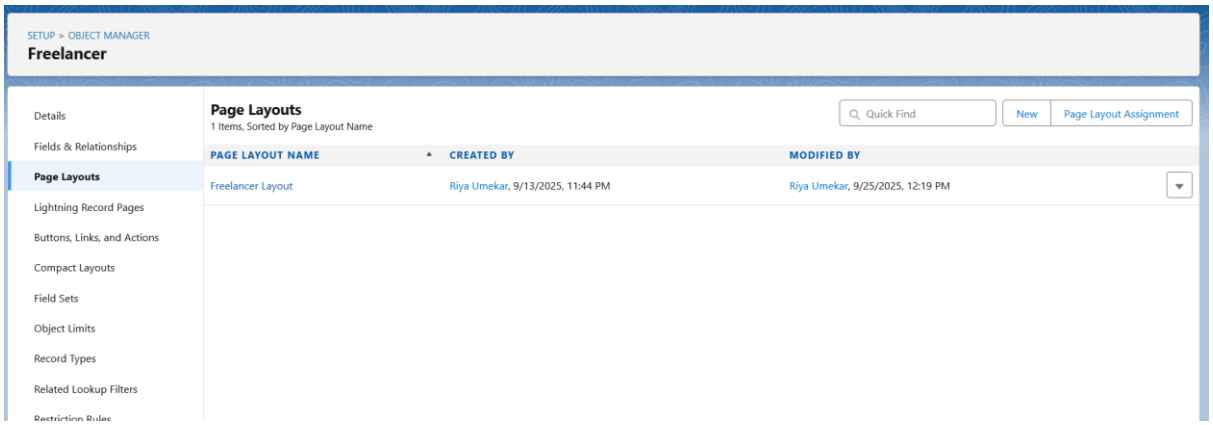
7. **User Profiles:** Assign the app to the relevant users.
    - From the 'Available Profiles' list, select the following custom profiles and move them to 'Selected Profiles':
        - **Resource Manager**
        - **Salesperson**
        - **Executive**
    - Also, add the **System Administrator** profile. Click **Save & Finish**.

Executive	Profile
Salesperson	Profile
Resource Manager	Profile

7.3 Customizing Record Page Layouts

Goal: To organize fields logically and add crucial related lists, making record pages easier to read and use.

1. **Add Freelancer Skills Related List to Freelancer Layout:**
    - Navigate to **Setup** > **Object Manager** > **Freelancer**.
    - Go to **Page Layouts** and click on **Freelancer Layout**.
    - In the palette at the top, select **Related Lists**.
    - Drag the **Freelancer Skills** related list from the palette down into the Related Lists section on the page layout.
    - To customize the columns in the related list, click the wrench icon (Properties) on the **Freelancer Skills** related list you just added.
    - Add the Skill field to the list of selected fields. Remove the Freelancer Skill ID if desired. Click **OK**.
    - Click **Save** on the page layout editor.



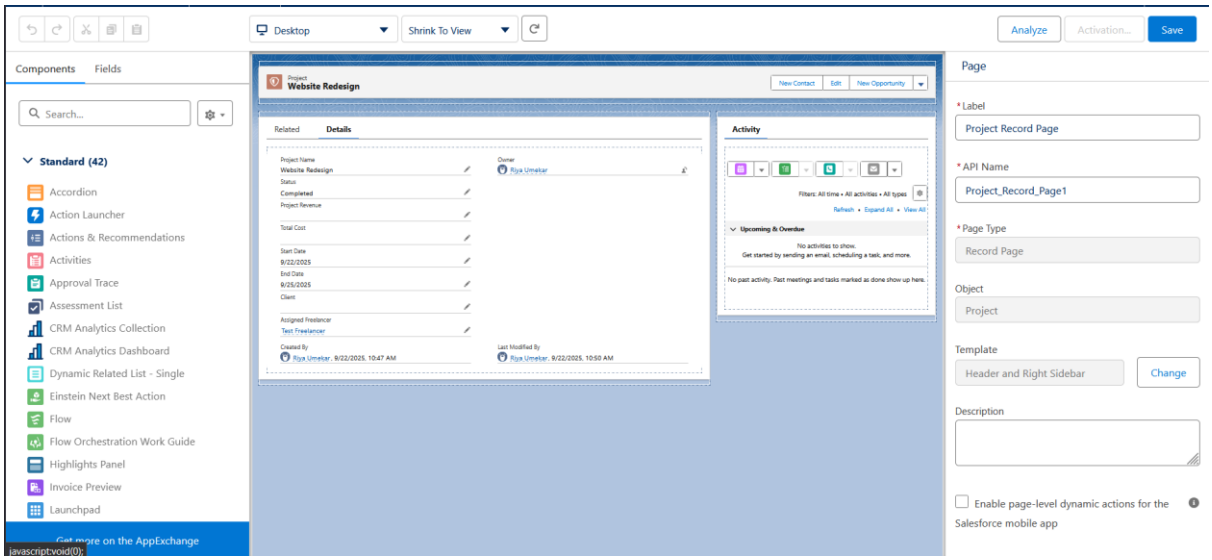
2. **Organize Project and Freelancer Fields:**
    - While editing the Page Layouts for both the **Project** and **Freelancer** objects, you can create new sections to group fields.
    - For the **Project** object, consider creating a "Financial Information" section for Project Revenue and Total Cost.
    - For the **Freelancer** object, you could group Email and Contact Record into a "Contact Information" section.

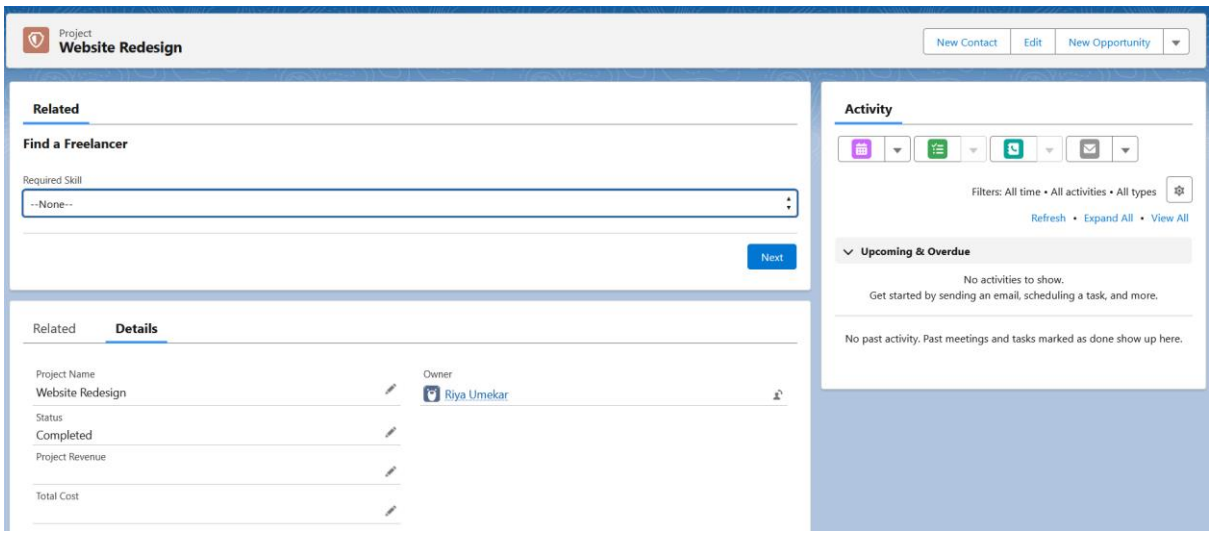
Activating the "Find a Freelancer" Screen Flow

Goal: To embed the interactive search tool created in Phase 4 directly onto the Project record page, making it instantly accessible to Resource Managers.

1. Navigate to any existing **Project** record in your org. (If you don't have one, create a sample record).

2. Click the **Setup Gear** icon in the top right and select **Edit Page**. This will open the Lightning App Builder.
3. **Add a New Tab:**
    - Click on the main component in the center of the page that contains the "Details" and "Related" tabs. The properties for this component will appear on the right-hand panel.
    - In the right-hand panel, click the **Add Tab** button.
    - A new tab detail will appear. Click on it.
    - For the **Tab Label**, select **Custom** and type Find a Freelancer. Click **Done**.
4. **Place the Flow Component:**
    - Click on the new **Find a Freelancer** tab in the main canvas to select it.
    - From the Components list on the left, find the **Flow** component and drag it onto the canvas inside the new tab.
5. **Configure the Flow Component:**
    - With the Flow component selected, look at the configuration panel on the right.
    - In the **Flow** dropdown, select your screen flow: Screen_Flow_Find_a_Freelancer_by_Skill.
    - Leave the other options as default.
6. Click **Save** in the top right.
7. A confirmation modal ("Page Saved") will appear. Click **Activate**.
8. In the Activation modal, assign the page as the **Org Default**. Click **Save**.
9. Click the **Back** arrow in the top left to exit the App Builder. Your "Find a Freelancer" tool is now live on all Project pages.





Phase 7: Integration & External Access

This phase outlines potential future integrations with external systems.

- **Potential Enhancement 1 (REST API)**: A custom REST API could be created to allow an external company website to pull and display a list of available skills from the Skill object.
- **Potential Enhancement 2 (Platform Events)**: Platform Events could be used to broadcast a message when a new Project is created, allowing external systems to subscribe .

This section is crucial for long-term planning, as it demonstrates how GigHub can evolve from a self-contained application into a connected part of a larger enterprise ecosystem. We will detail the steps a developer would take to build these enhancements, complete with code examples and screenshot guidance.

Screenshot Opportunity 19:

- **What to capture:** The Developer Console window showing the complete Apex class code for the SkillService.
- **Placement:** Insert the screenshot directly below this code block.

Step 2: Set up a Connected App for Authentication

External systems cannot access Apex REST services without authentication. A Connected App is the standard way to grant secure API access.

1. Navigate to **Setup**. In the Quick Find box, type App Manager and select it.
2. Click **New Connected App**.
3. Fill in the required fields:

    - **Connected App Name:** Company Website Integration
    - **API Name:** Company_Website_Integration
    - **Contact Email:** Your email address.
4. Under the **API (Enable OAuth Settings)** section, check **Enable OAuth Settings**.
5. **Callback URL:** Enter a placeholder URL like https://www.example.com/oauth_callback.
6. **Selected OAuth Scopes:** Select **Access and manage your data (api)** and move it to the 'Selected OAuth Scopes' list.
7. Click **Save**, then click **Continue**.
8. You will be taken to the Connected App detail page. It will display a **Consumer Key** and a **Consumer Secret**. These credentials, along with a Salesforce username and password, will be used by the external system to authenticate.

Screenshot Opportunity 20:

- **What to capture:** The "API (Enable OAuth Settings)" section of the New Connected App page, showing the enabled settings and the selected "api" scope.
- **Placement:** Insert the screenshot directly below this step.

Step 3: Testing the API

An external developer could now use a tool like Postman or curl to authenticate and call this endpoint to receive the list of skills in JSON format.

8.3 Potential Enhancement 2: Broadcast New Projects via Platform Events

Goal: To broadcast a notification message whenever a new Project is created in GigHub. This allows other systems (like a project management tool, a financial system, or a custom auditing service) to subscribe and react in real-time without having to constantly poll Salesforce for new data.

Step 1: Define the Platform Event

1. Navigate to **Setup**. In the Quick Find box, type Platform Events and select it.
2. Click **New Platform Event**.
3. **Configure the Platform Event:**
    - **Label:** New Project Created
    - **Plural Label:** New Project Created Events
    - **Object Name:** New_Project_Created
    - **Publish Behavior:** Publish Immediately
4. Click **Save**.

5. On the Platform Event detail page, in the **Custom Fields & Relationships** related list, click **New**.
6. Create the following custom fields to define the data "payload" of your event message:

Field Label	Data Type
Project ID	Text (255)
Project Name	Text (255)
Client Name	Text (255)
Project Revenue	Number(16, 2)
Created Date	Date/Time

Export to Sheets

Screenshot Opportunity 21:

- **What to capture:** The detail page for the "New Project Created" Platform Event, showing the list of custom fields that make up the event's payload.
- **Placement:** Insert the screenshot directly below this step.

Step 2: Publish the Event from the "Create Project" Flow

We will modify the record-triggered flow from Phase 4 to publish this event right after a project is created.

1. Navigate to **Setup** > **Flows**.
2. Open the Opportunity - After Update - Create Project on Closed Won flow.
3. Immediately after the **Create Project** element, click the **+** icon to add a new element.
4. Select the **Create Records** element.
5. **Configure the Create Records Element:**
    - **Label:** Publish New Project Event
    - **How Many Records to Create:** One
    - **How to Set the Record Fields:** Use separate resources, and literal values
    - **Object:** Search for and select your Platform Event, New Project Created.
    - **Set Field Values for the New Project Created:**
        - Field: Project_ID__c -> Value: Create_Project (from the previous flow element) > Id

        - Field: Project_Name__c -> Value: $Record > Name
        - Field: Client_Name__c -> Value: $Record > Account > Name
        - Field: Project_Revenue__c -> Value: $Record > Amount
        - Field: Created_Date__c -> Value: $Flow > CurrentDateTime
6. Click **Done**.
7. Click **Save As** to create a new version of the flow, then **Activate** the new version.

Screenshot Opportunity 22 (Key Screenshot):

- **What to capture:** The Flow Builder canvas for the modified flow. It should show the original "Create Project" element, followed immediately by the new "Publish New Project Event" element. The configuration panel for the publish element should be visible, showing the field mappings.
- **Placement:** Insert the screenshot directly below this step.

Now, whenever an Opportunity is "Closed Won," the flow will not only create a Project record but also instantly publish a New_Project_Created__e event, which any authorized external system can subscribe to and process.

Phase 8: Data Management & Deployment

This phase involves populating the org with data and outlining the deployment strategy.

- **Data Import**: Prepare sample data in CSV files for Skills, Freelancers, and Freelancer Skills. Use Data Loader to insert the records in the correct order .
- **Deployment Concept**: All development should be done in a Sandbox org first. The completed components would then be moved to the Production org using Change Sets .

This section will provide a clear, repeatable process for data loading and a foundational understanding of Salesforce's change management, ensuring a smooth transition to a live state.

Here is the updated project report with the detailed Phase 8.

Step 2: Use Data Loader to Insert Records

1. Open **Data Loader**.
2. Click **Insert**.
3. **Login:** Authenticate to your Salesforce sandbox org.
4. **Select Data Object:** Check the "Show all Salesforce objects" box and select the first object, **Skill (Skill__c)**.

5. **Choose CSV:** Browse for and select your Skills.csv file.
6. **Mapping:** Click **Create or Edit a Map**. Drag the Name field from the top panel to the Name field in the bottom panel to map the columns. Click **OK**.
7. **Finish:** Choose a directory for your success and error logs. Click **Finish**. Review the logs to ensure all records were created successfully.

Screenshot Opportunity 23:

- **What to capture:** The Data Loader mapping screen showing the Name column from the CSV file correctly mapped to the Name field on the Skill object.
- **Placement:** Insert the screenshot directly below step 6.

---

8. **Repeat the Process:**
    - Repeat steps 2-7 for the **Freelancer (Freelancer__c)** object using your Freelancers.csv file.
    - Repeat steps 2-7 for the **Freelancer Skill (Freelancer_Skill__c)** object using your FreelancerSkills.csv file.

9.3 Deployment Strategy using Change Sets

Goal: To package all the components we have built and configured in our Sandbox environment and deploy them to the Production environment in a single, managed transaction.

Step 1: Create an Outbound Change Set in Sandbox

1. In your **Sandbox** org, navigate to **Setup**.
2. In the Quick Find box, type Outbound Change Sets and select it.
3. Click the **New** button.
4. **Name:** GigHub Application V1
5. **Description:** Initial deployment of the GigHub custom application, including all objects, flows, and UI configurations.
6. Click **Save**.
7. On the Change Set detail page, find the **Change Set Components** section and click **Add**.

Screenshot Opportunity 24:

- **What to capture:** The "Add to Change Set" page, with the "Component Type" dropdown menu visible.
- **Placement:** Insert the screenshot directly below step 7.

---

8. **Add all necessary components.** This is the most critical step. Systematically go through the **Component Type** dropdown and add everything you've created:
    - **Custom Object:** Add Project, Freelancer, Skill, and Freelancer Skill.
    - **Custom Field:** Add all the custom fields for each of the four objects.
    - **Flow Definition:** Add your three flows (Opportunity...Create Project, Project...Update Status, Screen Flow...Find a Freelancer).
    - **Lightning App:** Add the GigHub app.
    - **Page Layout:** Add the layouts for Project and Freelancer.
    - **Lightning Record Page:** Add the Project record page that you customized.
    - **Profile:** Add the Resource Manager, Salesperson, and Executive profiles.
    - **Validation Rule:** Add the Project_Must_Have_Freelancer... rule.
9. Once all components are added, click the **Upload** button.
10. **Select Target Organization:** Choose your Production org from the list and click **Upload**.

Screenshot Opportunity 25:

- **What to capture:** The Outbound Change Set detail page after all components have been added, showing a list of component types and their names. The "Upload" button should be visible.
- **Placement:** Insert the screenshot directly below step 10.

Step 2: Deploy the Inbound Change Set in Production

1. Log in to your **Production** org.
2. Navigate to **Setup**.
3. In the Quick Find box, type Inbound Change Sets and select it.
4. Under the "Change Sets Awaiting Deployment" section, you will see GigHub Application V1. Click on its name.
5. **First, validate the change set.** Click the **Validate** button. This will run a test deployment to check for any errors or missing components without actually committing the changes.
6. Once the validation succeeds, you are ready to deploy. Click the **Deploy** button. This will move all the components into your production org and make the application live.

Phase 9: Reporting, Dashboards & Security Review

This phase focuses on data visualization and securing the application.

- **Custom Report Types**: Create a custom report type that links Freelancers with Freelancer Skills and Skills.
- **Reports**: Build reports such as a "Freelancer Skill Inventory" and a "Project Status Report".
- **Dashboard**: Create a "GigHub Management Dashboard" with charts and graphs using the reports .
- **Security Review**:
    - **Object Permissions**: Edit the custom profiles to set the correct object-level access (e.g., Executive has Read-only) .
    - **Field-Level Security**: Hide sensitive fields like Hourly_Rate__c from profiles that do not need to see it.

Overview of Analytics and Security

This phase focuses on two essential functions: transforming our data into actionable insights and securing the application. First, we will build the reporting infrastructure that allows managers and executives to track performance and inventory. We will create a custom report type, build key reports, and consolidate them into a powerful management dashboard. Second, we will conduct a thorough security review, configuring profile permissions and field-level security to ensure data integrity and enforce the principle of least privilege, where users only have access to the data they need to perform their jobs.

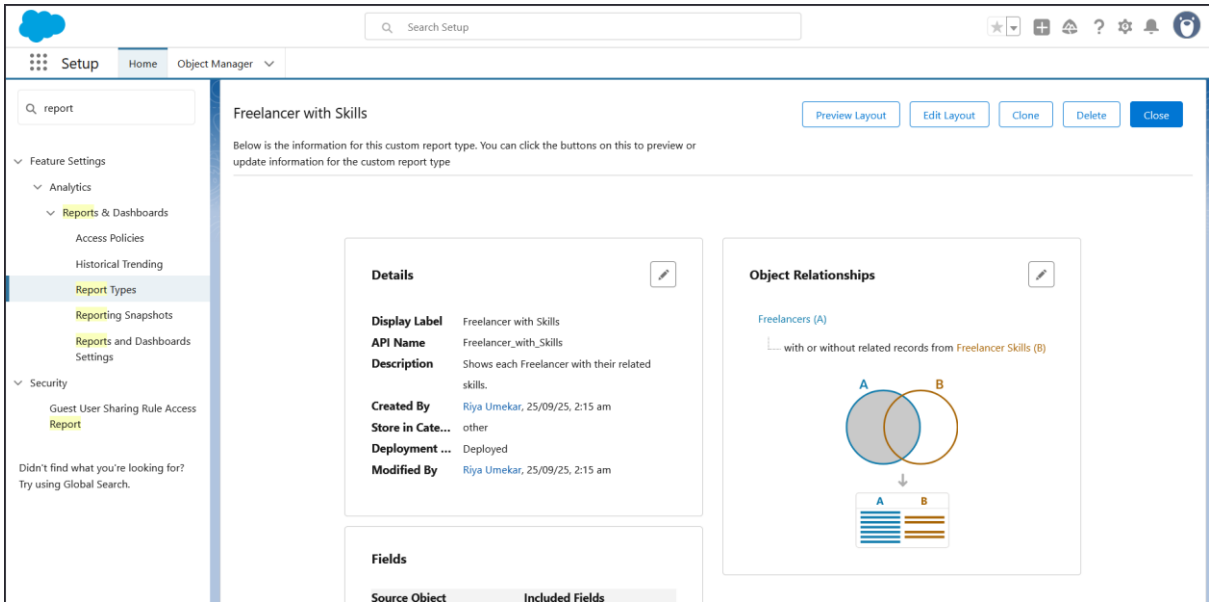
Custom Report Type Creation

Goal: To create a custom report type that allows us to report on Freelancers and see all of their associated Skills. This is necessary because the standard report builder cannot traverse the Freelancer -> Freelancer Skill -> Skill relationship path out of the box.

1. Navigate to **Setup**.
2. In the Quick Find box, type Report Types and select it.
3. If you see an introductory splash screen, click **Continue**.
4. Click **New Custom Report Type**.
5. **Step 1: Define the Report Type:**
    - **Primary Object:** Freelancers
    - **Report Type Label:** Freelancers with Skills
    - **Report Type Name:** Freelancers_with_Skills
    - **Description:** Shows all freelancers and their associated skills.
    - **Store in Category:** Other Reports

- **Deployment Status:** Deployed
- Click **Next**.

6. **Step 2: Define Report Records Set:**
    - The box 'A' (Freelancers) will already be present. Click the box below it that says **"Click to relate another object"**.
    - **Select Object:** Freelancer Skills (Freelancer)
    - **Relationship:** Each "A" record must have at least one related "B" record. This ensures that the report only shows freelancers who have at least one skill assigned.
    - Click **Save**.



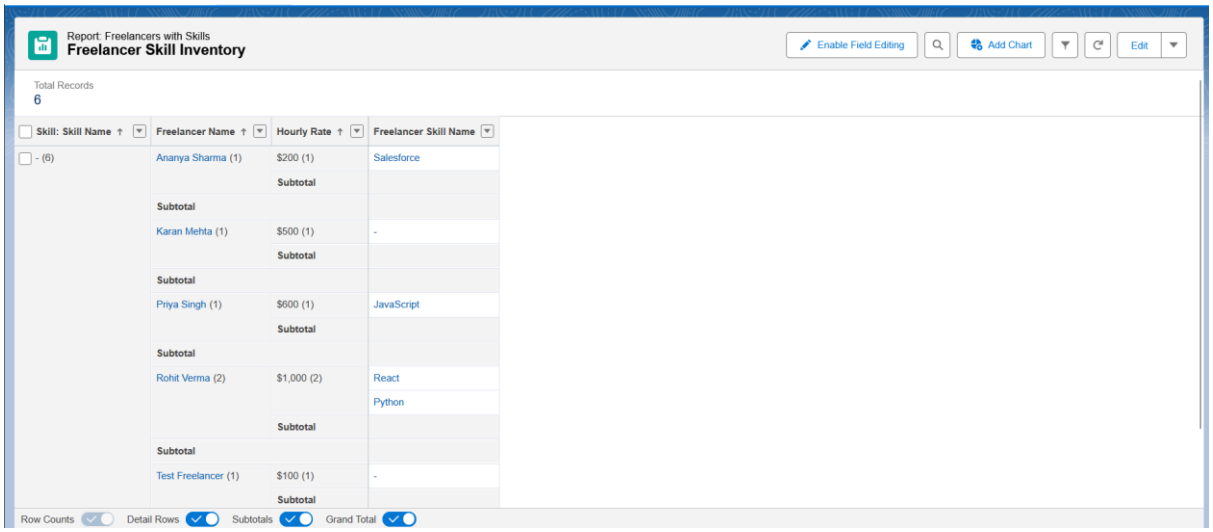
10.3 Building Key Reports

Goal: To create two fundamental reports that provide immediate value to the Resource Manager and Executive stakeholders.

Report 1: Freelancer Skill Inventory

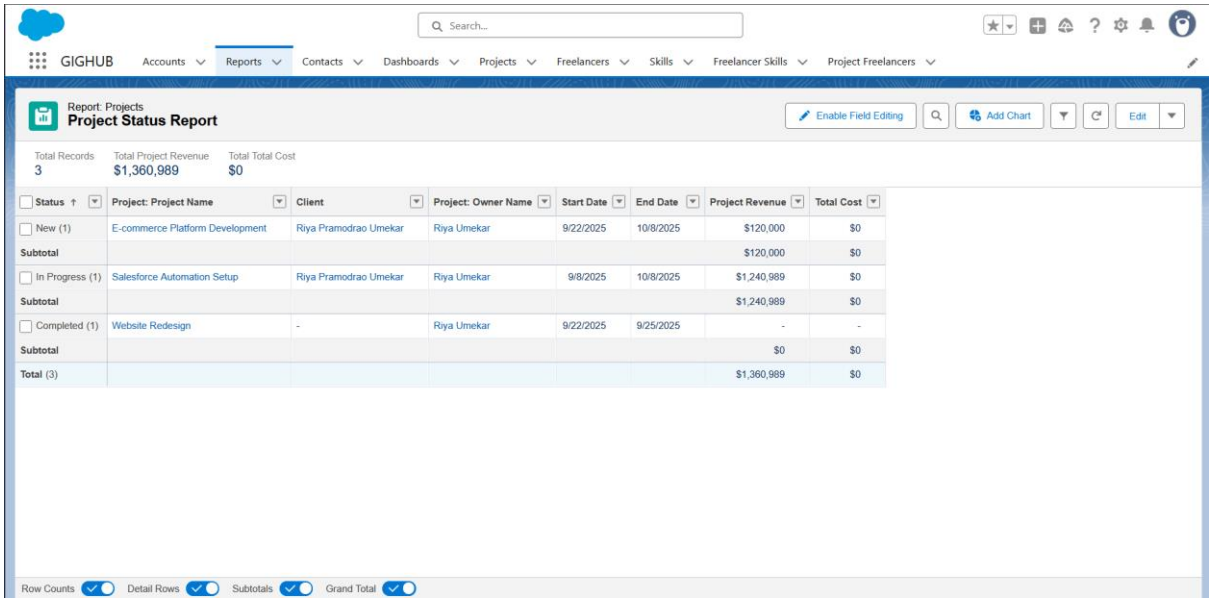
1. Navigate to the **Reports** tab in the GigHub app.
2. Click **New Report**.
3. In the "Choose Report Type" search box, find and select **Freelancers with Skills**. Click **Start Report**.
4. **Group the Report:** In the "Group Rows" search box under the "Outline" tab, add the Full Name field (from the Freelancer object).
5. **Add Columns:** Add the Skill: Skill Name, Hourly Rate, and Status columns to the report.
6. Click **Save & Run**.

- **Report Name:** Freelancer Skill Inventory
- **Report Unique Name:** Freelancer_Skill_Inventory
- **Folder:** Public Reports
- Click **Save**.



Report 2: Project Status Report

1. Go back to the **Reports** tab and click **New Report**.
2. In the "Choose Report Type" search box, find and select **Projects**. Click **Start Report**.
3. **Group the Report:** In the "Group Rows" search box, add the Status field.
4. **Add Columns:** Add Project Name, Client: Account Name, Project Revenue, and Assigned Freelancer: Full Name.
5. **Add a Chart:** Click the **Add Chart** button. Select the **Donut** chart icon. This will automatically create a visual breakdown of projects by status.
6. Click **Save & Run**.
    - **Report Name:** Project Status Report
    - **Report Unique Name:** Project_Status_Report
    - **Folder:** Public Reports
    - Click **Save**.

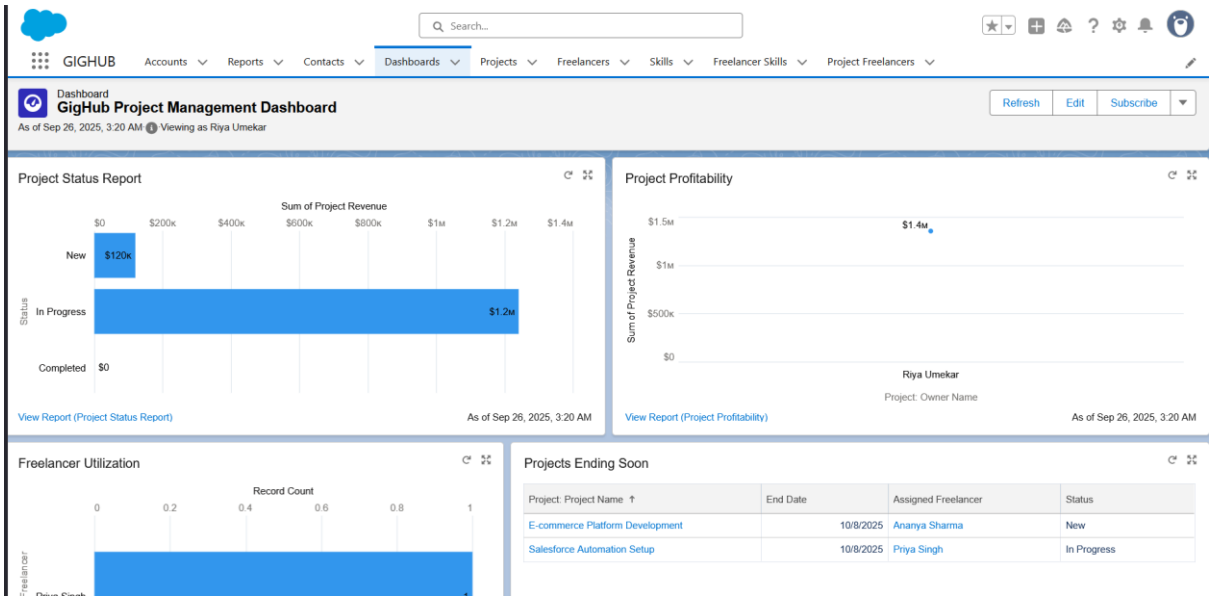


10.4 Creating the Management Dashboard

Goal: To create a single dashboard that provides at-a-glance insights into key business metrics, primarily for the Executive user.

1. Navigate to the **Dashboards** tab in the GigHub app.
2. Click **New Dashboard**.
3. **Name:** GigHub Management Dashboard
4. **Folder:** Public Dashboards
5. Click **Create**.
6. Click the **+ Component** button to add your first chart.
    - **Report:** Select the Project Status Report.
    - **Display As:** Select the **Donut** chart.
    - **Title:** Projects by Status
    - Click **Add**.
7. Click **+ Component** again.
    - **Report:** Select the Project Status Report.
    - **Display As:** Select the **Metric** component.
    - **Measure:** Sum of Project Revenue.
    - **Title:** Total Pipeline Revenue
    - Click **Add**.

8. Drag and resize the components on the dashboard canvas to arrange them logically.
9. Click **Save**, then click **Done**.



10.5 Security Model Review and Configuration

Goal: To configure the object and field-level permissions for our custom profiles to ensure users can only access the data appropriate for their role.

Step 1: Set Object Permissions

1. Navigate to **Setup**. In the Quick Find box, type Profiles and select it.
2. Click on the **Executive** profile.
3. In the profile settings, click on **Object Settings**.
4. Click on **Projects**. Click **Edit**.
5. Under **Object Permissions**, ensure that only the **Read** permission is checked. Uncheck all others (Create, Edit, Delete). Click **Save**.
6. Repeat this process for the Freelancer and Skill objects for the Executive profile.
7. Follow the same steps for the other profiles, setting their permissions according to this matrix:

Profile	Project	Freelancer	Skill
Resource Manager	Read, Create, Edit	Read, Create, Edit	Read, Create, Edit
Salesperson	Read	Read	Read
Executive	Read	Read	Read

Step 2: Set Field-Level Security (FLS)

Goal: To hide the sensitive Hourly_Rate__c field on the Freelancer object from users who do not need to see it, like the Salesperson and Executive.

1. Navigate to **Setup** > **Object Manager** > **Freelancer**.
2. Go to **Fields & Relationships** and click on the **Hourly Rate** field.
3. Click the **Set Field-Level Security** button.
4. In the security settings, ensure the **Visible** checkbox is checked **only** for the **Resource Manager** and **System Administrator** profiles. Uncheck it for all other profiles, especially **Salesperson** and **Executive**.
5. Click **Save**.

Phase 10: Final Presentation & Demo Day

The final phase involves preparing and delivering a demonstration of the completed application.

- **Prepare a Narrative**: Structure the presentation as a story that explains the business problem (the chaos of spreadsheets) and presents the GigHub app as the solution .
- **Demonstrate the Core Process**: Walk through the main business process during the demo: show a Salesperson closing a deal, see the Project automatically created, log in as the Resource Manager to use the "Find a Freelancer" tool, assign the freelancer, and finally, log in as the Executive to show the updated dashboard .
- **Document Work**: Take screenshots of the data model, flows, and the final dashboard to use in a portfolio and for handoff documentation .