Candidate: Paritosh upadhyay

Client: Lirik

Panel: Missi Bawa

Date: 06-06-2024

1. Introduce yourself

2. what are the things required to make a user and profile in saleforce

To create a user and assign a profile in Salesforce, you'll need a few things:

**User Requirements:**

* **Unique Username:** This acts as the user's login ID within Salesforce and typically takes the format of an email address (e.g., [email address removed]).
* **Email Address:** This is where the user receives notifications and can be used for password resets.
* **First and Last Name:** Basic user information for identification.
* **User License:** This determines the features and data access the user will have. Different licenses offer varying levels of functionality.

**Profile Requirements:**

* **Existing Profile:** You can assign an existing profile to the user, which pre-defines access levels for objects, fields, and functionalities within Salesforce. Salesforce offers standard profiles for common roles (e.g., Sales Representative, System Administrator).
* **Custom Profile (Optional):** If the standard profiles don't meet your specific needs, you can create custom profiles tailored to user roles within your organization. This requires administrative permissions.

3. explain validation rules with examples

4. what are record types?

Salesforce Record Types are a way of grouping many records of one *type* for that object. Thesecan be applied to any standard or custom object, and allow you to have a **different page layout**, **fields**, **required fields**, and **picklist values**.

They should be used for records that have the **same concept**, but need to be **different in execution**.

As mentioned above, record types can be created on any standard or custom object in Salesforce, allowing you to configure different:

* Page layouts
* Fields
* Required fields
* Picklist values

5. explain asynchronous apex?

Asynchronous Apex is a powerful feature in Salesforce that allows developers to run Apex code asynchronously, meaning that it does not need to execute immediately and can be processed at a later time when resources are available. This is particularly useful for long-running operations, complex computations, or tasks that do not need to be completed in real-time.

**1. Future Methods**

Future methods are used to run processes in a separate thread, at a later time when system resources become available. They are ideal for operations that are not dependent on real-time processing, such as making callouts to external systems, performing resource-intensive calculations, or updating a large number of records.

### 2. ****Batch Apex****

Batch Apex is used to process large volumes of data asynchronously in chunks or batches. It is particularly useful for operations that need to handle thousands or millions of records efficiently, such as data cleansing, data migration, or complex reporting.

### 3. ****Queueable Apex****

Queueable Apex is similar to future methods but provides additional features such as job chaining and monitoring. It allows you to add jobs to the queue and chain them together, providing more control over the job execution sequence.

### 4. ****Scheduled Apex****

Scheduled Apex allows you to schedule Apex classes to run at specific times. This is useful for operations that need to be executed periodically, such as nightly data processing, weekly reporting, or monthly maintenance tasks.

6. can we call future into batch?

No

7. Can we call batch from batch in execute method?

No

8. How do you call a wire method from another wire method in LWC?

Yes we can do this, but directly it is not possible,

We can have multiple wire methods in same LWC component. One wire method may be dependent on

another. We may require one wire method response in to another.

Wire methods can call automatically via framework. We don’t have any control on that.

So directly this is not possible ,but with the help of reactive parameters we can do this.

9. what is @wire?

wire decorator is used to retrieve data from server-side controllers or Apex methods. The data

returned by the method is stored in the component’s property, and the component rerenders with the new

data

10. what wire method

11. why do we write @auraenabled

The @AuraEnabled annotation in Salesforce Apex is used to expose Apex methods and properties to Lightning components,

which includes both Aura components and Lightning Web Components (LWC). It allows these components to call server-side

Apex code from the client-side JavaScript.

12. can we call future method from batch?

No

13. explain flows , record trigger flows ( scenario)

14. write a code to update the account with the count of contacts

15. write a code to update all contact phone no with accounnt phone no

Candidate: Dinesh Parekh

Client: Lirik

Panel: Vivek Gotharwal

Date: 06-06-2024

1. Relationship and its types.

2. Explain Report type in detail where you use real time scenario

### Types of Report Types

1. **Standard Report Types**: Automatically created by Salesforce for all standard objects and many standard object relationships (e.g., Accounts with Contacts, Opportunities with Accounts).
2. **Custom Report Types**: Created by users to define custom relationships between objects, including custom objects, and to customize which fields are available for reporting.
3. Junction Object scenario and What will happen if we undelete the deleted Junction Object?
4. What is primary master

The first master-detail created on the junction object is known as the primary master. The second master-detail created on the junction object is known as the secondary master. Whosoever is the owner of the primary master will be the owner of corresponding child records.

1. Refresh Apex and it's scenario ?

**Step 1:**  In your JavaScript code, import the utility.

**import{** refreshApex **}** from '@salesforce/apex';

**Step 2:** Create a reference variable to hold the data that requires refreshing.

wiredAccountResult;

@wire**(**getAccounts**)** wiredAccounts;

**Step 3:** Develop a method that can utilize to initiate the data refresh process

refreshData**(){**

**return**refreshApex**(this**. wiredAccountResult**)**;

**}**

**Step 4:** Trigger the refresh action based on user interactions, events, or business logic requirements.

1. What is Sharing Rules, Apex sharing, Manual sharing and Automatic sharing with real time scenario

 **Sharing Rules**: Automate

d, criteria-based or ownership-based sharing.

 **Apex Sharing**: Programmatic sharing using Apex code for complex logic.

 **Manual Sharing**: User-initiated, ad-hoc sharing from the UI.

 **Automatic Sharing**: Built-in sharing through role hierarchy, manager groups, and sharing sets.

1. What is with sharing and without sharing

### with sharing

When you declare a class with sharing, the Apex code respects the sharing rules of the current

user. This means that the code will enforce the same record access permissions that the user has.

### without sharing

When you declare a class without sharing, the Apex code does not enforce the sharing rules of the current user. This means that the code can access all data irrespective of the user's sharing settings.

1. Future method call in future method possible or not? why?

No

1. Batch apex limitation

**Limitations of Batch Apex**

1. **Execution Governors and Limits**:
   * **Total Number of Batch Apex Jobs**: Only 5 active or queued batch jobs are allowed at one time. This includes start, execute, and finish methods.
   * **Number of Batch Apex Classes**: Only 100 batch classes can be defined.
   * **Total Number of Records Processed**: Each transaction in a batch job can process up to 10,000 records.
   * **Maximum Batch Size**: Each batch can process up to 2,000 records.
   * **Heap Size**: The maximum heap size is 6 MB for synchronous Apex and 12 MB for asynchronous Apex.
   * **Database Query Limits**: Each batch job execution is limited to 50,000 SOQL queries or a maximum query time of 120 seconds.
   * **DML Limits**: Each batch execution can issue up to 10,000 DML statements.

**Chaining Limitations**:

* A maximum of one execute method can call the Database.executeBatch method. You cannot chain batch jobs more than once within a single execute method.

1. Difference Future and Queueable apex with real time scenario
2. Trigger context variables

Salesforce provides several context variables that can be used within triggers to access runtime information about the DML operation.

**Summary of Context Variables Usage**

* **Trigger.isInsert**: Check if the trigger is fired due to an insert operation.
* **Trigger.isUpdate**: Check if the trigger is fired due to an update operation.
* **Trigger.isDelete**: Check if the trigger is fired due to a delete operation.
* **Trigger.isBefore**: Check if the trigger is fired before the records are saved.
* **Trigger.isAfter**: Check if the trigger is fired after the records are saved.
* **Trigger.new**: Access the new version of the sObject records being processed.
* **Trigger.old**: Access the old version of the sObject records being processed.
* **Trigger.newMap**: Access a map of IDs to the new versions of the sObject records.
* **Trigger.oldMap**: Access a map of IDs to the old versions of the sObject records.
* **Trigger.size**: Get the total number of records in a trigger invocation.
* **Trigger.isUndelete**: Check if the trigger is fired due to an undelete operation.

1. What is Dynamic form in LWC

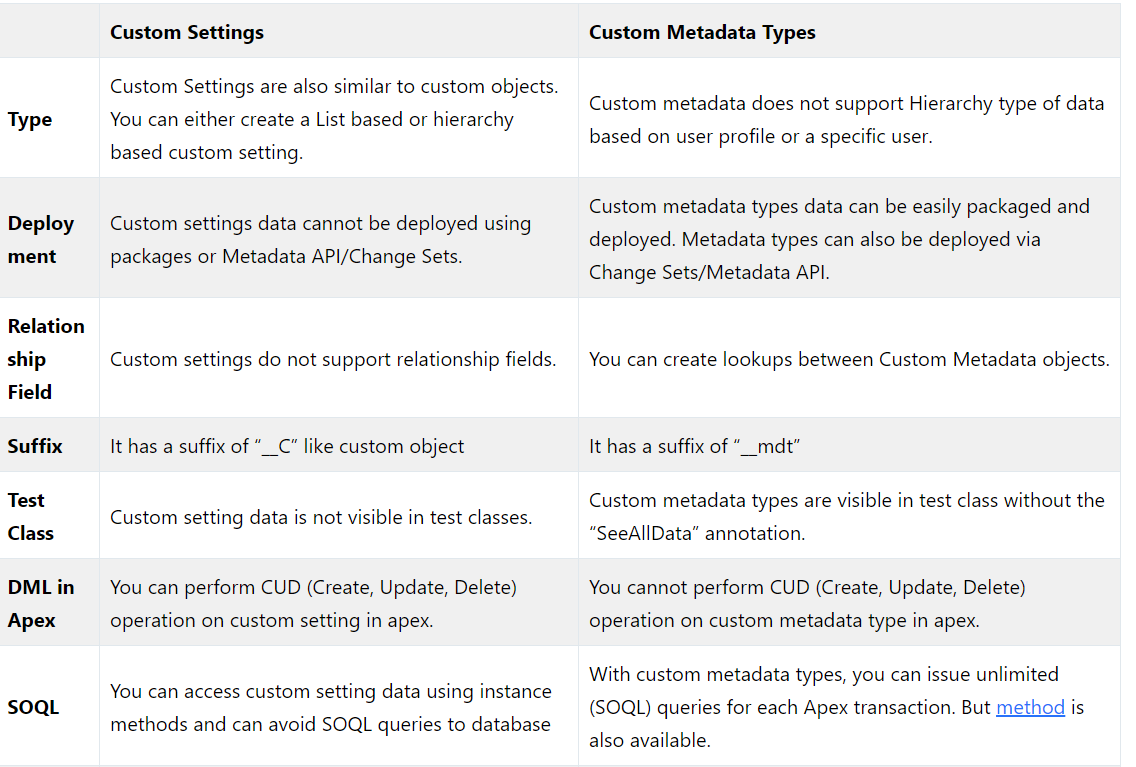
Dynamic Forms in Lightning Web Components (LWC) that allows for the creation of highly customizable and responsive record forms. These forms enable admins and developers to build forms that can dynamically adjust based on user input, field values, or other conditions.

1. Difference between Custom settings and Custom metadata

[Custom Settings](https://www.apexhours.com/salesforce-custom-settings/) are like custom objects. They are generally used to create custom sets of data and can be associated with an organization, profile or a specific user. The visibility of the custom setting can be controlled by marking it as public or protected.

The term “[Metadata](https://help.salesforce.com/s/articleView?id=sf.custommetadatatypes_overview.htm&type=5)” is nothing but data about data. In simple words, when you are creating a record within Salesforce, you are creating data as well as metadata. The data which you are trying to enter, is captured or stored in the fields which are nothing but columns of your Object.

[Custom Metadata Types](https://www.apexhours.com/custom-metadata-types/) in Salesforce are similar to custom objects. It has a suffix of “\_\_mdt” instead of “\_\_c” in the API namespace.



14. Write SOQL Query for Retrieves Account records that have associated Contacts.

15.When Account Industry field updates, Please updates its all related Contact, Contact\_Industry\_\_c from value of Industry of Account.(Codding Question)

Candidate: Nagarjuna   
Client: Lirik L1

Panel: Poornima Bharadwaj  
Date: 06-06-2024

1. Introduction

2. Explain What you have idea about Asynchronous Apex

3. Explain Batch Class

4. Can we perform 20000 records processing in single Transaction

5. Have you worked on Integration

6. What is Authentication and What was the flow you followed while you implementing Integration

7. Code a trigger for When the Status of an Account is Changed to “Active” , Update all the related contacts Status field to “Contact Updated”   
8. Why Should you used Map instead of Set in the code

you should use a Map instead of a Set when you need to store and retrieve data based on unique keys. Maps

are ideal for scenarios where you need to perform efficient lookups, associate keys with values, and work with complex data

structures. Sets, on the other hand, are best suited for ensuring uniqueness and checking the presence of elements within a

collection. By understanding the strengths and appropriate use cases for each, you can make more effective decisions in your

Apex code.

9. Have you worked LWC

10. Explain Life Cycle hooks

Candidate: Nagarjuna   
Client: Lirik L2

Panel: Hemanth Thakkar  
Date: 06-06-2024

1. Introduction

2. Object Relationships

3. Flows

4. Remote Site Setting vs Named Credentials

5. Future Vs Queueable

6. When to use @wire

7. Can we include lwc in visual pages

8. Junction Object Scenario

9. Approval Process and how many ways we can send for approval

An Approval Process in Salesforce provides a structured way to manage the review and approval of records within your

organization. You can send records for approval manually, through automation tools like Workflow Rules and Process Builder,

or programmatically using Apex code or APIs. The choice of method depends on your specific business requirements and the

complexity of your approval process.

10. I have 2 approvals I want to send one of them how

11. I have to perform record approval one after other approval user like queue how

12. Relationship bw contact and account

Standard lookup relationship

13. Lightning record form

The lightning-record-form component provides a quick and efficient way to create forms in Salesforce Lightning

Web Components. It significantly reduces the amount of code needed to create, view, and edit records. For more customized

layouts, you can use lightning-record-edit-form and lightning-input-field components. This flexibility

allows you to create both simple and complex forms to meet various business requirements.

14. Write Lightning record form code Scenario

Need to add 2 fields of account and one submit button.

15. Write a trigger for Scenario where I have 10 opportunity records has email field now and those are associated with one account object. How if email changes its related records also need to changed.

16. How to invoke apex in flow

#### Explanation

* **@InvocableMethod**: This annotation makes the calculateTotalValue method available to be called from a Flow.
* **@InvocableVariable**: This annotation defines the input variable opportunityIds which will be passed from the Flow to the Apex method.
* The method calculates the total value of the specified opportunities and returns the result as a list containing a single Decimal value

Go to flow > select flow type(screen flow and Auto-launched flow)

* In action search for apex class you created (ex: OpportunityCalculator.calculateTotalValue).

Configure the Apex Action

Use the Output in the Flow

17. Deployment tools

Changesets and workbench

18. Change Sets steps

**Prepare Changes**:

**Create a New Changeset**:

**Add Components**:

**Upload Changeset**:

**Validate Changeset**:

**Upload Test Level**:

**Deploy Changeset**:

**Monitor Deployment**:

19. How to stop recursive trigger

20. One Lwc usecase scenario on getQuerySelector() for parent child components

21. Can child obj records effected if master object record deleted in junction object

22. Refresh Apex

23. Integration knowledge

24. Promises

25. Reports and Dashboard

**Reports**:

* **Purpose**: Reports in Salesforce allow you to organize and display data from your Salesforce records. They help you analyze information, identify trends, and make data-driven decisions.
* **Types of Reports**:
  + **Tabular Reports**: Display data in rows and columns, similar to a spreadsheet.
  + **Summary Reports**: Summarize data with grouping and subtotals, providing a higher-level view of your data.
  + **Matrix Reports**: Display data in a grid format, with both rows and columns summarized.
  + **Joined Reports**: Combine multiple report blocks to display related data side by side.

 **Customization**: You can customize reports by adding filters, grouping data, creating charts and

graphs, adding formulas, and scheduling them to run at specific intervals.

 **Access**: Reports can be shared with specific users or groups, allowing teams to collaborate and analyze data together.

26. Difference bw summary reports and Matrix reports and when to choose

 **Summary Reports**:

* Choose summary reports when you want to group data hierarchically and view subtotals and totals.
* Ideal for analyzing data along a single axis (e.g., by time, geography, or category).
* Provides a straightforward, structured view of summarized data.

 **Matrix Reports**:

* Choose matrix reports when you need to analyze data across multiple dimensions simultaneously.
* Suitable for comparing data across different categories or attributes.
* Offers a more comprehensive and multidimensional view of summarized data.

Candidate: Thati Neeraja

Client: Lirik L1

Panel: Aditya saxena

Date: 06-06-2024

1. What is case assignment rule

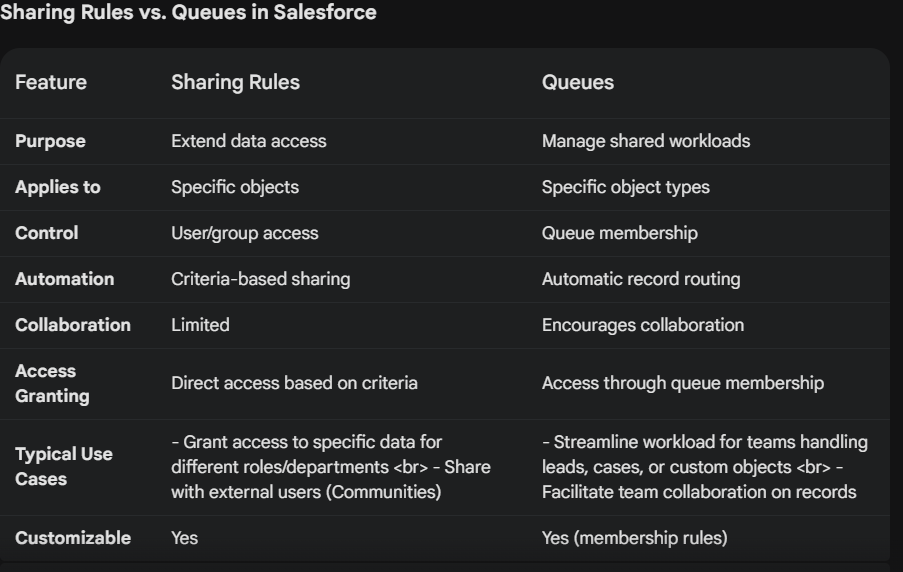
In Salesforce, a **case assignment rule** automates the process of assigning new cases to specific users or queues. It acts like a set of instructions that tells Salesforce how to distribute incoming cases based on pre-defined criteria.

2. What is standard junction object.

They are pre-built objects within Salesforce that act as connectors between two standard objects to establish a many-to-

many relationship.

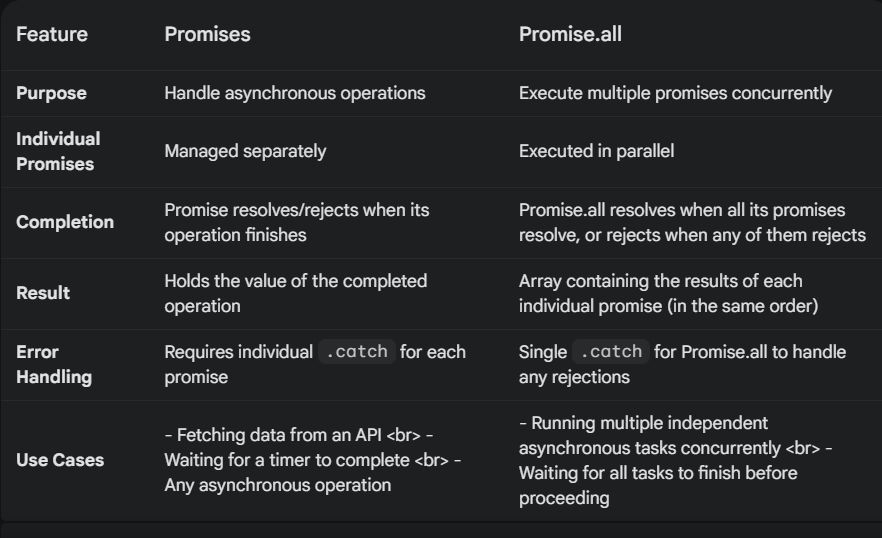
3. What is sharing rules and queues.



4. Coding Scenario - A account has multiple contacts .If Contact has a field colour and it has 2 record with one is red and green ,it does not allow those 2 colour

5. Create lightning record form, on Account object

6. Promises and promises all



7. Do you Know about uncommited pending error.

Yes, I do know about the "uncommitted work pending" error in Salesforce. It occurs when you attempt to make a

callout (an outbound communication to an external system) while there's still uncommitted data pending in the

current transaction.

8. What is the method for integration test.

Candidate: Govardhan  
Client: Lirik L1

Panel:  
Date: 10-06-2024

1. What you use for Testing Integration?

Apex Test Classes, Data Loader, Code Coverage and Quality Tools,

1. WhoId vs WhatId

**WhoId:**

* **Represents:** People or "human" objects in Salesforce, typically Contacts or Leads.
* **Label:** Often referred to as "Name" on the activity or event object where it's used.
* **Purpose:** Links activities or events (tasks, meetings, etc.) to specific individuals you're interacting with.

**WhatId:**

* **Represents:** Objects other than people, such as Accounts, Opportunities, Campaigns, Cases, or even custom objects.
* **Label:** Often referred to as "Related To" on the activity or event object where it's used.
* **Purpose:** Establishes connections between activities or events and relevant non-human objects within your Salesforce data.

1. Quote, Product Line, Opportunity Line Item, Order

**Quote**:

* The Quote object in Salesforce represents a formal offer or proposal given to a customer that includes details such as pricing, products/services, quantities, terms, and conditions.

**Product Line:**

* An optional grouping level within a Quote or Opportunity.
* It allows you to categorize products or services offered based on type, brand, or any other relevant criteria.

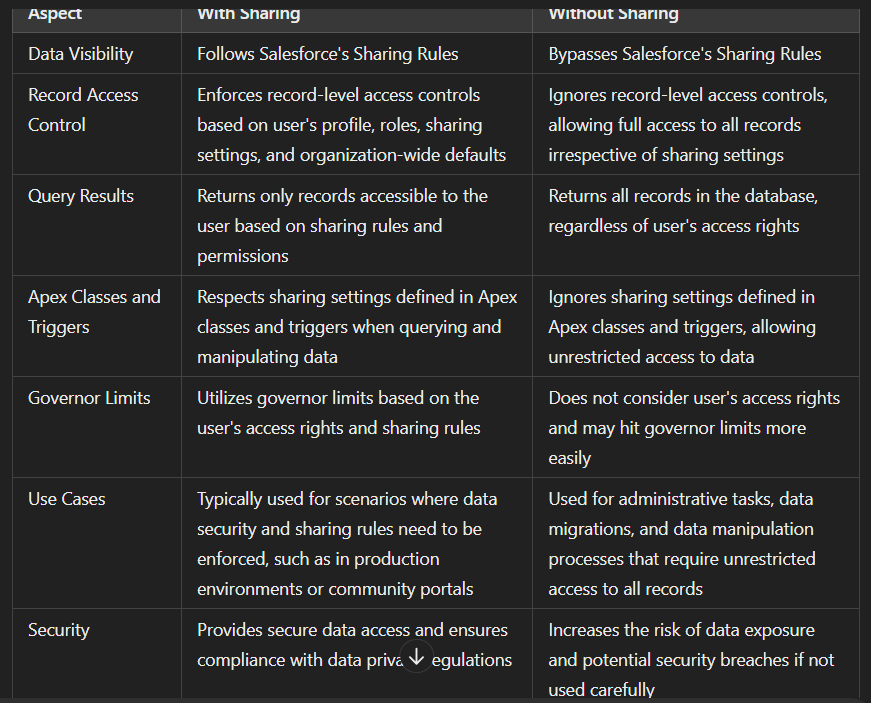
**Opportunity Line Item**:

* The Opportunity Line Item object in Salesforce represents individual line items or products/services associated with an opportunity.
* When a quote is converted to an opportunity, the quote line items are typically transferred to opportunity line items.

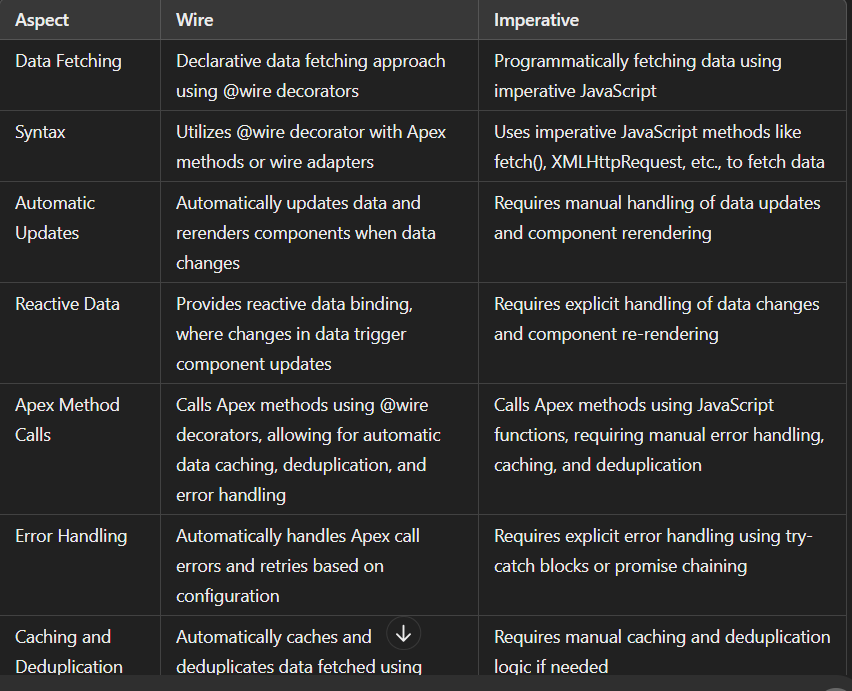
**Order**:

* The Order object in Salesforce represents a finalized agreement or purchase made by a customer for products or services.
* Orders are typically created after a quote is accepted, an opportunity is closed-won, and the customer confirms the purchase.

1. With Sharing vs Without Sharing



1. Wire vs Imperative



**Wire Service:**

* **Declarative:** You define the data you need in your component's JavaScript code using the @wire decorator.
* **Automatic Data Binding:** The Wire service automatically fetches the data from the server whenever the underlying data changes on the server-side (e.g., record updates, field changes).
* **Automatic UI Updates:** Once the data is retrieved, the Wire service automatically updates the component's UI to reflect the changes.
* **Simpler Syntax:** Generally requires less code compared to the imperative approach.
* **Limited Control:** Offers less control over the data fetching process (e.g., no custom logic during data retrieval).

**Imperative Calls:**

* **Procedural:** You write code within your component's JavaScript methods to explicitly call Apex methods and handle data retrieval.
* **Manual Data Binding:** You need to manually assign the retrieved data to the component's properties to update the UI.
* **Manual UI Updates:** You need to call methods to update the UI components based on the retrieved data.
* **More Control:** Provides more flexibility and control over the data fetching process (e.g., applying custom logic before or after data retrieval).
* **More Complex Syntax:** Typically requires more code compared to the Wire service.

1. Refresh Apex
2. Batch Apex and its Methods

Batch Apex in Salesforce is a powerful tool for processing large volumes of data asynchronously. It allows you to execute code in chunks, improving performance and avoiding governor limits that could restrict processing in a single transaction.

**Concepts:**

* **Batchable Interface:** This interface is the foundation of Batch Apex. Your Apex class implementing this interface defines how the data will be processed in batches.
* **Start method:** This method is invoked once at the beginning of the batch job execution. You can use it for any one-time setup tasks before processing the data chunks.
* **execute method:** This is the core method where you define the logic for processing each batch of data. It's called multiple times throughout the job, with each call receiving a list of records to process.
* **finish method (Optional):** This method is called once at the end of the batch job execution. You can use it for any finalization tasks, such as sending notifications or logging information

1. PMD Issues

PMD (Programming Mistake Detector) is a static code analysis tool used in Salesforce development to identify potential issues, coding best practice violations, and code smells in Apex code. Here are some common PMD issues that developers may encounter in Salesforce:

**Unused Private Method**:

**Unused local variable**

**Unused Field**

**Avoid SOQL Queries in Loops**:

**Avoid DML Statements in Loops**:

**Hardcoded IDs or URLs**:

**Empty Catch Block**:

**Avoid Using System.debug() in Production**:

**Avoid Hardcoding Field Names**:

1. SOQL Injection

SOQL Injection occurs when the input for a query comes from the user’s end and the user inputs a malicious query in the query parameter to bypass the original query restrictions.

How to prevent SOQL Injection ?

* Avoid using dynamic SOQL where possible, instead use static queries and binding variables
* If you must use dynamic SOQL, use the escapeSingleQuotes method to sanitize user-supplied input.

1. Case Assignment
2. Case Escalation
3. Named Cred vs Remote Site Setting
4. Rest Api and Its Http Methods.

Candidate: Chandrakala  
Client: Lirik L1

Panel:  
Date: 10-06-2024

1. Introduction
2. Current Project in Details
3. Integration
4. Experience in Admin, Developemnt and Integrarion
5. Security Model and Explain Mannual Sharing

 **Role Hierarchy**: Defines the organizational structure and hierarchy of roles within an organization. Users higher in the role hierarchy have access to records owned by users below them in the hierarchy.

 **Profiles**: Determine the permissions and settings assigned to a user, such as object permissions, field-level security, page layouts, record types, and app permissions. Profiles are assigned to users based on their job functions and responsibilities.

 **Permission Sets**: Provide additional permissions and access settings beyond what is defined in profiles. Permission sets are typically used to grant temporary or specific permissions to users without modifying their profiles.

 **Organization-Wide Defaults (OWD)**: Define the default level of access to records for all users within an organization. OWD settings include Private, Public Read-Only, Public Read/Write, and Public Full Access, which determine how records are shared by default.

 **Sharing Rules**: Extend sharing beyond the OWD settings by defining criteria-based rules to share records with specific groups of users or roles. Sharing rules can be based on criteria such as record ownership, fields values, or membership in public groups.

 **Manual Sharing**: Allows users with appropriate permissions to manually share individual records with other users or groups. Manual sharing is used to grant temporary or ad-hoc access to specific records that are not covered by sharing rules or OWD settings.

1. Difference bw Custom settings and Custom Metadata
2. Flows
3. When to go flows and When to go Apex

### When to Use Flows:

1. **Low to Medium Complexity Processes**:
2. **No Code/Low Code Solutions**:
3. **Integration with Lightning App Builder**:
4. **Dynamic Record Updates**:
5. **Multi-Step Processes with Screen Flows**:

### When to Use Apex:

1. **Highly Customized or Complex Logic**:
2. **Batch Processing and Scheduled Jobs**:
3. **Integration with External Systems**:
4. **Governor Limits and Bulk Processing**:
5. **Complex Business Logic with External Libraries**:
6. Rollup Summary Field
7. Best practices of Test Classes
8. After update event, When it trigger in rollup summary field
9. Why we use view all = false/ true in test class

The @isTest(SeeAllData=false) annotation (or SeeAllData=true) is used in Salesforce test classes to control

access to data within your organization during unit testing of Apex code.

**SeeAllData=false (Default):**

* **Isolation:** This is the recommended default setting for unit tests. It simulates a more realistic scenario where your code operates within the permissions assigned to the user it's running under (typically a system user with limited access).
* **Focus on Code Logic:** By restricting data access, you force your test class to focus on testing the logic of your code and how it interacts with data it can create or that's readily available through the sharing hierarchy.
* **Repeatable Tests:** Tests that don't rely on pre-existing data in the org are more reliable and can be executed consistently in any Salesforce environment (sandbox, production) without data dependencies.

**SeeAllData=true:**

* **Limited Use Cases:** This setting grants the test class access to all data in your organization, bypassing sharing rules and field-level security. It should be used cautiously and only in specific scenarios.
* **Testing Integration Points:** In some cases, you might need to test how your code interacts with data owned by other users or requires access to specific existing records. SeeAllData=true can be helpful for such scenarios.
* **Data Mocking Considerations:** Even with SeeAllData=true, keep in mind that some data, like Chatter feeds or user activity, might still be restricted. Consider using data mocking techniques to simulate the necessary data interactions.

1. When we use @testVisible

The @TestVisible annotation in Salesforce is used within Apex test classes to grant access to private

or protected members (methods, variables, inner classes) of another class during unit testing.

1. When we use trigger.new

**Using trigger.new:**

* **Insert Triggers:** trigger.new is primarily used in insert triggers to access and process the details of the newly inserted records. You can iterate through the trigger.new list and perform actions on each record's fields.
* **Update Triggers (Conditional):** In update triggers, you can use trigger.isInsert to differentiate between insert and update operations. If it's an update, you can then conditionally access and process the new values in trigger.new for the updated records.

1. What are the trigger context variable in before events

 trigger.isInsert: This flag is true only in before insert triggers, indicating an insert operation is

about to happen.

 trigger.isUpdate: This flag is true only in before update triggers, indicating an update operation

is about to happen.

 trigger.isDelete: This flag is not available in before events. It's used in before delete triggers

(which are rarely used due to limitations).

1. Batch Class
2. Soql and Sosl Injection

SOQL Injection occurs when the input for a query comes from the user’s end and the user inputs a malicious

query in the query parameter to bypass the original query restrictions.

How to prevent SOQL Injection ?

* Avoid using dynamic SOQL where possible, instead use static queries and binding variables
* If you must use dynamic SOQL, use the escapeSingleQuotes method to sanitize user-supplied input.

SOSL injection is similar to SOQL injection but occurs with SOSL queries. An attacker can manipulate the SOSL query to

return unintended results or cause errors.

1. Locker Services

Lightning Locker is a layer which sits in between your browser and DOM (document object). In other words, Lightning Locker is a virtual browser that allows only secure request to go through and have access to real DOM. This virtual browser sits in front of your unsafe real browser.

Candidate: P Nagarjuna  
Client: Lirik Final Round

Panel:Gaurav  
Date: 11-06-2024

1. What are the consideration you have worked out when working with

--> User

--> Profile

--> Role

### Users

1. **User Creation and Deactivation**:
2. **Permission Management**:

**Profile and Role Assignment**: Assign the correct Profile and Role based on the user's job function and required access levels.

**Permission Sets**: Use Permission Sets to grant additional permissions without changing the user’s Profile.

1. **Security**:

* **Password Policies**: Enforce strong password policies and regular password changes.
* **Two-Factor Authentication**: Implement two-factor authentication (2FA) to enhance login security.

1. **Record Ownership**:

### 

### Profiles

1. **Access Control**:
   * **Object and Field-Level Security**: Define object and field-level permissions to control which data users can view and edit.
   * **Record Types**: Use record types to offer different picklist values and page layouts based on the Profile.
2. **Permissions**:
   * **Standard vs. Custom Profiles**: Use standard profiles as a base but create custom profiles to meet specific needs of different user groups.
   * **Minimum Permissions**: Follow the principle of least privilege by granting the minimum necessary permissions required for users to perform their jobs.
3. **User Interface**:
   * **Page Layouts**: Assign appropriate page layouts to profiles to ensure users see relevant information and fields.
   * **Apps and Tabs**: Control the visibility of apps and tabs to simplify the user interface and improve navigation.

### Roles

1. **Data Access Hierarchy**:
   * **Role Hierarchy**: Set up the role hierarchy to reflect the organizational structure, which determines data visibility and reporting lines.
   * **Access Levels**: Ensure roles are designed to provide appropriate access levels, such as read-only, read-write, and full access based on position.
2. **Sharing Rules**:
   * **Manual Sharing**: Implement manual sharing rules to allow users to share records with others who might not have access through roles.
   * **Criteria-Based Sharing**: Use criteria-based sharing rules to grant access to records based on specific conditions.
3. **Role-Based Data Visibility**:
   * **Record Access**: Ensure users in higher roles have access to records owned by users in lower roles, following the role hierarchy.
   * **Private Sharing Model**: Consider using a private sharing model with roles and sharing rules to grant access as needed, enhancing data security.

2. Can we able to access an Opportunity records where is OWD is Set as Private

There are a few ways you might still be able to access a private Opportunity record:

* **Record Owner:** If you are the owner of the Opportunity record, you will have full access regardless of the OWD setting.
* **Sharing Rules:** Sharing rules can be configured to grant read or read/write access to specific users or groups even for private records.
* **Manual Sharing:** Individual users can be manually shared specific Opportunity records, granting them access even if they wouldn't have access based on their profile or permission sets.
* **Hierarchies:** If Opportunity sharing is set to use a hierarchy (e.g., role hierarchy), users higher in the hierarchy might have access to Opportunities owned by users below them.
* **System Administrator:** System Administrators have inherent access to all data within the Salesforce org, including private Opportunity records.

3. I have a requirement, Where i am Sending an approval for OpportunityProductLine a Child Object when ever is submitted for approval then i want send approval for Parent Object called Opportunity as Well how can we achieve it.

4. Future vs Queuable

5. FlexQueue

FlexQueue in Salesforce is a system that helps manage the execution of Apex batch jobs. It allows you to submit

more than the default limit of five batch jobs that can be queued for processing

6. How many batches we can run simultaneously.

5

 **FlexQueue Holding Capacity:** This queue can hold up to 100 batch jobs that are waiting for

resources.

 **Active Batch Limit:** Only a maximum of **five** batch jobs can be actively processed at any given time.

7. How many Jobs can we chain in Batch and Queueable

**Batch Apex Chaining:**

* **Limit:** You can chain a maximum of **five** Batch Apex jobs together.
* **Execution:** Each chained job starts only after the previous one finishes.
* **Important Note:** This chaining refers to starting a new batch job from within the finish method of another batch class.

**Queueable Apex Chaining:**

* **Limit:** There's no strict limit on chaining Queueable Apex jobs.
* **Execution:** You can chain as many Queueable jobs as needed, but be mindful of governor limits like total asynchronous executions per day.
* **Chaining Method:** Queueable jobs are typically chained by returning a new AsyncApexJob from the execute method of the current job. This new job represents the next job in the chain.

8. Can we perform Callouts with Queueable

Yes, you can absolutely perform callouts with Queueable Apex.

9. What is the governor limit of Job Chaining in queueable and Batch

 **Queueable Jobs**: You can chain only one additional queueable job from a running queueable job.

 **Batch Jobs**: You can chain up to five additional batch jobs from the finish method of a running batch

job.

10. Integration how can you authenticate and Authorize the external system

11. What are the requirements for Authentication in integration

12. Write a coding scenario, I have an array of 5 elements let say 1,2,3,4,5 now i want to pass the indexed value to next to the all elements form its index. and If i give other numbers like 6,7,8,9,10 till 100 i want to repeat the same output what we get for index 1,2,3,4,5. Write the logic in any technology alike JS. and Run in Online Compiler.

13. In LWC, What are the approaches we go thru while implementing the communication bw LWC components Child to Parent?

14. And Parent to Child

15. In point of field visibility, i have 2 fields one Country and State, Where State contain Some Capitals of india, Now with record type can we disable the some capitals values in state field if select country as India.

16. Record type?

17. What are the relationships are available in salesforce

18. Give example for Many to One Relationship.

Candidate: Thati Neeraja

Client: Lirik client round

Panel: Rohit

Date: 14-06-2024

1. Day to Day activities regarding development
2. Best Practices of triggers
3. how do you avoid recursive trgger
4. what is database.insert and insert
5. what is queuable apex?in which scenario u go for it?
6. Do u know cron expression
7. How can we store endpoints in salesforce
8. Soql injection
9. Aura vs lwc
10. 2-way binding
11. Lms
12. Wire
13. How do you handle event in lwc
14. Visual force page vs visualforce component

Candidate: Thati Neeraja

Client: Hero client round

Panel: Jai Gupta

Date: 14-06-2024

1. On which part you have worked on service cloud?
2. Have you written apex class for escalation
3. How do you achieve escalation
4. Can we write flow on delete operation after update
5. How can you set before / after scenario In RTF
6. Screen flow scenario
7. Have u works on aura and lwc
8. What are decorators
9. How many soql queries in sync and async
10. How many events in trigger
11. Limitations of queuable apex
12. How can you achieve with batch apex
13. What Is cpu limit
14. What deployement tools u used
15. Have u worked on integration
16. What is error log object
17. How do you rertrigger error in integration
18. Is metadata querying hits the goverener limits
19. Have u worked on visualforce pages
20. Scenario: a, b, c, d, lwc components and they are in hierarchy , I want to send data from d to a? how can we achieve this
21. I have account list in a table and I want to display account details if account status is active,then it should be green otherwise it should be red

Candidate: Harshini (Senior)

Client: Lirik

Panel: Hitesh Narulla

Date: 18-06-2024

1.Difference between Future & Queueable

2.Trigger.New ?How many records it hold?

3.Avoiding Recursive Trigger

4.Callout in Batch? How many

5. How to enforce security based on user

6.SOQL Injection & its prevention

7.cross site scripting(XSS)

8.Locker Services

9.Test.starttest & test.stoptest

10.@ testvisible ?

11.Refresh apex

12.Promises, callbackhell, async/await

13.communication in LWC

14. Worked on Which flows

15. Customsettings & Custom metadata

16.Is Profile mandatory?

17.Data security Model

18.Rollup summary code

Candidate: Kondal

Client: Lirik L1

Panel: Prashanth Mathur

Date: 19-06-2024

1. Introduction
2. How you will customize lightning layout and page layout
3. Dynamic Related List

Dynamic Related Lists in Salesforce Lightning Experience offer a more flexible and powerful way to display related records on

record pages. Unlike traditional related lists, Dynamic Related Lists allow you to define filter criteria directly within the

component, customize the fields displayed, and sort records without modifying page layouts.

1. There are 2 object A,B, How can you build the relationship?
2. Different Bw Future and Queueable
3. Is it possible to perform Http Callout in Future
4. Batch Apex Follows Which one of this Method overloading or over ridding in oops concept
5. If Batch Size is 100 how many times constructor,start,execute methods will run
6. How will you call lwc component through Aura
7. What is parent tab and sub tab in lwc
8. Worspace api and its uses
9. How to open aura using workspace api
10. Best practices of triggers
11. Rollup summary code
12. Difference between let, var, const
13. Parent to child and child to parent communication in lwc
14. Can you pass multiple key value pairs in detail atribute in custom events
15. Can you perform dml operations in wire yes or no why
16. Imperative apex
17. Named Credentials
18. How will you bypass remote site settings by using named credentials
19. Why you choose Named credentials instead remote site settings
20. Recent Intigration scenario
21. How do you create test class and how many methods you can create in test class
22. Deployment tools
23. How will you pass multiple parameters from js to apex

Candidate: Rashid Ahmed

Client: Lirik L1

Panel: Hitish

Date: 20-06-2024

1. Intro
2. Project Overview and Roles ans Responsibilities and Team Size
3. Deployment tools which you used
4. Agile Methodology
5. Github, GearSet
6. Can we moniter Apex job
7. What is profile
8. Data Securtiy model And Sharing rule and Manual Sharing(scenario Based)
9. OWD(scenario Based)
10. Manual Sharing How to extend(scenario Based)
11. Permission Set(scenario Based)
12. Trigger on Account and Contact(scenario Based)
13. Differentiate Custom Setting and Custom Metadata(scenario Based like what custom setting do and Metadata can’t)
14. Explain Flows where you work
15. Why we go for Apex Class instead of record trigger flow
16. How to decide whether to use flow or Apex
17. Rest Api and Methods
18. Remote Site Setting and Named Credentials
19. Connected Apps Explain & creation
20. Oauth 2.0 full Explaination
21. Serialization and De-serialization
22. Wrapper Class
23. Before Events what Context Variable we cannot used
24. Where to store Responses from Http
25. Before insert why not use newMap
26. Can we callout future method and its Drawback
27. Parent to Child Queries vice versa
28. Trigger.New (scenario Based)
29. Batch Apex(scenario Based)
30. SOQL query on Account where AccountId is not associated with contact
31. With Sharing and Without Sharing(With Security enforce & user mode
32. Dynamic SOQL queries and Injecion

Dynamic SOQL (Salesforce Object Query Language) queries are constructed and executed at runtime. While they provide flexibility, they also introduce the risk of SOQL injection, a type of attack where an attacker can manipulate the structure of the query to execute unintended commands or access unauthorized data. Preventing SOQL injection involves validating and sanitizing all user inputs that are used to construct dynamic SOQL queries.

### Understanding SOQL Injection

SOQL injection can occur when user-supplied data is directly concatenated into a query string without proper sanitization. This can lead to unauthorized access, data leaks, or other malicious activities.

1. PMD

PMD is a source-code analyzer that allows for static analysis of code written in a number of supported languages, including Java, Apex, and Visualforce. Its built-in rules detect common flaws in code, such as empty catch blocks or unused variables.

1. XSS

SS is a malicious JavaScript code injected within a web-page by the end user (hacker). This code can perform malicious activity like get browser cookies, getting user’s sessions info etc.

1. How to Test Code
2. Callout Exception

A Callout Exception in Salesforce occurs when there is an issue with an HTTP request made to an external web service. This can happen for a variety of reasons, such as network issues, incorrect endpoint URLs, authentication failures, timeouts, or server errors. Proper handling of callout exceptions is crucial to ensure robust and resilient Salesforce applications.

1. Database.AllowCallout
2. Trigger to Count all the contact and contact name with comma separated and populated this in its parent record in 2 fields Count Contacts & ContactNames\_\_c