**PROJECT DEPLOYMENT PHASE**

**DEBUGGING AND TRACEABILITY**

|  |  |
| --- | --- |
| **Date** | **03 NOVEMBER 2023** |
| **Team ID** | **NM2023TMID04681** |
| **Project name** | **BUILD AN EVENT MANAGEMENT SYSTEM** |

Debugging and traceability are critical aspects of Salesforce development to ensure your event management system is robust and reliable. Salesforce provides various tools and techniques for debugging and tracing issues in your code. Here's an overview of how to implement and test code while focusing on debugging and traceability within the Salesforce platform:

**1. Debugging in Salesforce:**

Salesforce provides tools for debugging, including debug logs and system logs.

Debug Logs: Debug logs capture detailed information about code execution, database operations, and more. To create and view debug logs:

In Salesforce, go to Setup.

Search for "Debug Logs" and set up a new trace.

Assign the trace to your user or other users who will execute the code.

Execute your code, and debug logs will record details for analysis.

System Logs: System logs are helpful for real-time debugging. You can use System.debug() statements in your Apex code to log custom messages. These logs are visible in real-time using the Developer Console in Salesforce.

**2. Traceability in Salesforce:**

Traceability ensures that you can trace code execution and changes to your data and configurations.

Version Control: Use version control systems like Git to track changes to your custom code and configurations in Salesforce. Tools like Salesforce DX make it easier to work with version control and trace changes.

Change Sets: Salesforce Change Sets allow you to bundle changes in your organization and deploy them to other environments (e.g., from a Sandbox to Production). This helps maintain traceability when migrating code and configurations.

**3. Testing in Salesforce:**

Before deploying any changes to your event management system in Salesforce, comprehensive testing is essential. Salesforce provides various types of testing:

Unit Testing: Write unit tests for your custom Apex code using Salesforce's built-in testing framework. Ensure your code behaves as expected and is free from regressions.

Integration Testing: Test how your custom code interacts with standard Salesforce functionality and external integrations. This ensures that your event management system works seamlessly within the Salesforce ecosystem.

User Acceptance Testing (UAT): Involve end users in UAT to ensure that the event management system meets their requirements and works correctly.

Regression Testing: After making changes or deploying new features, conduct regression testing to verify that existing functionality remains intact.

**4. Best Practices:**

Follow best practices for code quality and maintainability, such as adhering to Salesforce's coding standards, using bulk processing for data operations, and minimizing the use of SOQL queries and DML statements inside loops.

Keep thorough documentation to track changes, their impact, and test results.

**5. Continuous Improvement:**

Regularly review and improve your code and configurations to enhance performance, security, and scalability.

**6. Compliance and Security:**

Ensure that your event management system adheres to Salesforce security and compliance standards. This includes managing user access, data security, and compliance with Salesforce regulations like GDPR.

The actual code for your event management system in Salesforce depends on your specific requirements, and debugging and traceability will be an ongoing process. To develop and test code within Salesforce, you'll typically use the Salesforce Developer Console, which provides a comprehensive environment for writing, testing, and debugging your Apex code.

For specific code examples or debugging scenarios related to your event management system, please provide more details about the functionality you're working on, and I can offer more targeted assistance.