Project Development Phase-**||**

**Debugging & Traceability**

|  |  |
| --- | --- |
| NAME | NAVEEN. V |
| NM ID | C259747173E9454BEE44375156B7964E |
| PROJECT | SEARCH ENGINE OPTIMIZATION |

Debugging:

1. Logging: Implement comprehensive logging throughout your application. Use structured logs that capture relevant information about each operation, including timestamps, error details, and the user responsible. Logging can be useful for auditing and debugging.

2. Debugging Tools: Utilize integrated development environments (IDEs) and debugging tools provided by your chosen programming language. Tools like breakpoints, watches, and stack traces are valuable for isolating and fixing issues.

3. Exception Handling: Properly handle exceptions, including capturing exceptions in a structured way and providing clear error messages. Handle exceptions gracefully to prevent crashes and data corruption.

4. Unit Testing: Write unit tests to validate the functionality of individual components. This helps catch errors early in the development process and provides a safety net for future changes.

5. Integration Testing: Conduct integration tests to verify that different parts of your system work together as expected. This helps identify issues that may arise when components interact.

6. Version Control: Use version control systems (e.g., Git) to track code changes. Version control enables you to pinpoint when and why a change was made and helps with identifying the source of issues.

7. Code Review: Implement a code review process to have team members review each other's code. Code reviews can help catch logic errors, security vulnerabilities, and best practices violations.

Traceability:

1. Unique Identifiers: Assign unique identifiers to campaigns, ad groups, ads, keywords, and other elements in your system. This helps trace actions and changes back to specific entities.

2. Audit Trails: Maintain an audit trail of significant events, including who initiated the event, when it occurred, and the nature of the action. This can be useful for tracking changes and identifying accountability.

3. Data Flow Diagrams: Create data flow diagrams that illustrate how data moves through your system. This can help visualize the path of data and actions, making it easier to identify issues and dependencies.

4. Change Management: Implement a change management process to track and document changes to your system. This includes change requests, approvals, and the deployment of new versions.

5. User Activity Logs: Record user activities, such as logins, campaign creation, ad modifications, and settings changes. This traceability can help in identifying issues, auditing user actions, and ensuring accountability.

6. Request Tracing: Implement request tracing mechanisms that track the journey of a user's request through the system. This can help pinpoint bottlenecks and issues in request processing.

7. Custom Reporting: Develop custom reporting tools that allow you to retrieve historical data and track the performance of campaigns over time.

8. Integration with Monitoring Tools: Integrate your application with monitoring and observability tools (e.g., Prometheus, Grafana) to track system performance and receive alerts when anomalies occur.