1. **Use Version Control**:
    - Implement a version control system, such as Git, to track changes in your codebase. This allows you to see who made changes, when they were made, and what specific changes were implemented.
2. **Source Code Repositories**:
    - Host your source code in a repository like GitHub, GitLab, or Bitbucket. These platforms provide collaboration features, issue tracking, and code review capabilities, enhancing traceability and collaboration.
3. **Issue Tracking System**:
    - Integrate an issue tracking system, such as Jira or Trello, to manage and track bugs, feature requests, and project tasks. Assign issues to team members, set priorities, and maintain a clear backlog.
4. **Debugging Tools in Salesforce**:
    - Utilize Salesforce's built-in debugging tools, like Debug Logs and Developer Console, to trace and troubleshoot issues. You can set log levels, monitor the execution of your code, and capture error messages.
5. **Code Testing**:
    - Implement unit testing, integration testing, and user acceptance testing to catch and fix issues at various stages of development. Salesforce provides testing frameworks like Apex and Lightning Testing Service.
6. **Continuous Integration and Continuous Deployment (CI/CD)**:
    - Set up CI/CD pipelines to automatically build, test, and deploy your application. This ensures that code changes are thoroughly tested before being deployed, improving traceability and quality.
7. **Documentation**:
    - Maintain detailed documentation for your application, including architectural diagrams, data models, and process flows. This

documentation provides insight into how the application works and can be used for traceability.

8. **Code Reviews**:
    - Enforce a code review process to have team members review each other's code. This helps identify issues early and ensures code quality.
9. **Debug Symbols and Comments**:
    - Include clear and meaningful comments in your code to explain complex logic or intentions. Use meaningful variable and function names. Utilize debug symbols and comments to facilitate debugging.
10. **Change Management and Release Planning**:
    - Implement a formal change management process to track and approve changes to the application. This includes a release planning process to ensure that changes are properly documented and tested.
11. **Regression Testing**:
    - Whenever changes are made, perform regression testing to ensure that existing functionality is not affected. Automated regression testing can help identify issues quickly.
12. **User Acceptance Testing (UAT)**:
    - Involve end-users in UAT to verify that the application meets their requirements. This provides a real-world test of the application's functionality.
13. **Monitoring and Logging**:
    - Implement application performance monitoring and logging to capture runtime errors and performance issues in production. Tools like Salesforce Event Monitoring can help with this.
14. **Compliance and Auditing**:
    - Ensure that your application adheres to any compliance requirements and auditing standards applicable to property management.

By incorporating these practices and tools into your property management application development using Salesforce, you can establish a robust system for debugging and traceability throughout the project's development phase. This helps you maintain a clear and organized development process while ensuring the quality and reliability of your application.