**Question-3**

**From your experience, what are some practices engineering teams can use to be more effective at developing high-quality code?**

1. **Give the developer’s choice to select the task**

I would like to share development practice we followed in the Oracle and how that lead to develop and deliver quality code.

* Every morning in the coffee meeting, display all the tasks to be done for the week and allow developers to select the task on which they would like to work or would like to learn new functionality based on their interest and seniority.
* Giving team member an opportunity to be an owner of the task which itself comes with a responsibility to deliver quality output. We developer feels that this is my own work rather than an assigned task to be completed.

**How that leads to deliver quality output?**

* As I get the opportunity to choose a task, I always use to select something new that I can learn. To learn that, automatically I used to put more efforts because I want to learn it and not just want to complete the assigned task.
* We had a culture of ownership, hence every team member used to think that the work I will be doing represents me. Hence, everyone uses to take responsibilities and tried to deliver quality output all the time.

1. **Maintaining a codeBook per project**

* Each team/project should maintain a codebook as a central repository for a team where all team members will put new solutions they have developed, solutions for common issues, configurations, SQL Queries, etc.

**How that leads to deliver quality output?**

* Instead of reinventing a wheel, codebook will have tested solutions to the common problems that might occur in the future or occur periodically. A developer can search into the codebook and reuse the code. This practice will save time and testing efforts and workload of developers to a great extent.
* As all-time developers get assigned the task which was previously done by another team member hence instead of eating other developer’s time one can find solutions into the codebook.

1. **Strictly following coding standards defined by team/organization**

* As the majority of time teams within the group share the codebase. Hence, standard coding practices defined at the team level or group level must be adhered by all the developers to efficiently reuse code.
* Maintaining a change log in each code file, and adding comments give an idea to another developer what was changed last time in that code file which helps a lot in the debugging of production issues.
* Standard naming conventions- naming criteria for each type of component. For example- how algorithm file would be named? How metadata components would be named? Etc.
* I did publish a paper in the Good Practice Conference for the Oracle product on which I was working. (<https://drive.google.com/open?id=1WkJ0Pp4pyy61Y1-jtDwZDqkQtpNXoTt->)

1. **Hierarchical and Group Code reviews**

* Code review practice should be done in the hierarchy of experience level. For example- Code written by SDE1 developer will be reviewed first by SDE2 developer and so on. It will give a more refined quality code at the end of the review process.
* If possible, code review by the senior developer in front of all team members will give a better vision for all developers on what mistakes to be avoided.

1. **Architect first**

* To deliver quality code, every developer should practice architecting their code. It could be pseudo-code to be written and having flow in mind will help the developer during actual implementation.
* This practice will save a lot of efforts in bug fixing or refactoring the code in the future.

1. **Weekly, team brainstorming sessions**

* Friday evening one hour should be allotted to the team brainstorming session to share on what task everyone is working and how they are approaching and developing solutions.
* This practice will be a great learning platform for every team member which will improve everyone’s ability to learn from everyone and implement it in their work to deliver high-quality output.

1. **Iterative Coding practice**

* Don’t write all at once! Before implementing analyze and predict, what are the valuable features that you can develop with quality in the short term. Use such iterations to deploy quality updates, and not waste time and resources on unreasonable desires and sacrifice with quality.

1. **Automating unit testing and common tasks**

* Automating unit testing saves a lot of development time and developers can quickly run sanity test cases by running pre-written scripts.
* Automation should be part of team culture and a daily practice for every developer.