DAY 5 INTERNSHIP REPORT SURFBOARD PAYMENTS

Today was my fifth day at Surfboard Payments. I learned about teamwork and GitHub version control, which are important for working in a team.

In the morning, koushik explained different types of team players and how their qualities affect teamwork. He divided them into three categories:

Accidental Mess Makers (Humble + Hungry) – They work hard but sometimes cause problems because they don't think strategically.

stackers(Humble + Smart) – They are reliable and trustworthy but may not take the lead.

Skillful Politicians (Hungry + Smart) – They are strategic and focused on results but may prioritize personal goals over teamwork.

This helped me understand my strengths and weaknesses in a team. I realized that a good team player should have a balance of humility, ambition, and intelligence.

In the afternoon, we had a session on GitHub. Before this, I thought GitHub was just for storing code. But I learned that it helps teams work together on software projects.

One of the key things I learned was about branches. A branch allows developers to work on a new feature separately without affecting the main project. Once the changes are tested, they can be merged into the main branch.

I also learned about pull requests. Instead of making direct changes, developers first submit a pull request. This allows teammates to review the code, suggest improvements, and approve it before merging.

I practiced some key Git commands:

git init – Creates a new Git repository.

git clone – Copies an existing project from GitHub.

git add – Stages changes before saving them.

git commit -m "message" – Saves changes with a message.

git push – Uploads changes to GitHub.

git pull – Downloads the latest updates from GitHub.

git merge – Combines changes from different branches.

I also learned about merge conflicts, which happen when two people edit the same file. GitHub provides tools to fix these conflicts and keep the project running smoothly.

A good commit message helps team members understand changes. Our instructor suggested using Conventional Commits, a structured way of writing commit messages.

We also learned about code reviews. Instead of making direct changes, developers submit a pull request, and teammates review the code before approving it. This helps prevent mistakes and improves code quality.

Today, I realized that GitHub is more than just a place to store code. It is a tool that helps teams work together efficiently without errors.