The most important part of a commit message is that it should be clear and meaningful. In the long run, writing good commit messages shows how much of a collaborator you are. The benefits of writing good commit messages are not only limited to your team but indeed expand to yourself and future contributors.

Here's how you can make yours commit msg. great:

1. Specify the change in each committed files, (E.g. if there are 3 committed files there shall be a message in the body for each file)
2. Specify each msg. for each file with the type of commit:

* **Feat:** The new feature you're adding to a particular application
* **Fix:** A bug fix
* **Style:** Feature and updates related to styling
* **Test**: Everything related to testing
* **Docs**: Everything related to documentation
* **Chore:** Regular code maintenance.

1. Separate the subject from the body with a blank line
2. Your commit message should not contain any whitespace errors
3. Use the body to explain what changes you have made and why you made them.
4. Use bullets to separate each change and keep it clear
5. Do not assume the reviewer understands what the original problem was, ensure you add it.
6. Do not think your code is self-explanatory
7. Avoid any spelling mistakes
8. Capitalize each line and each paragraph

Example:

**Body**

* Doc- Added the SW\_Req.doc
* Test- Modified the PIL.slx test file
* Chore- Performed an improvement in the model (voltage\_measurment.slx) by adding the following ………and……