An essential step in the software development lifecycle is code review. Code reviews are powerful means to improve code quality, establish best practices, opportunity to learn, and knowledge sharing and mentoring, as well as promotes team cohesion.

What to look for in a code review? Try to look for things such as 𝗱𝗲𝘀𝗶𝗴𝗻 (does this integrate well with the rest of the system and are interactions of different components make sense), 𝗳𝘂𝗻𝗰𝘁𝗶𝗼𝗻𝗮𝗹𝗶𝘁𝘆 (does this change is what the developer intended), 𝗰𝗼𝗺𝗽𝗹𝗲𝘅𝗶𝘁𝘆 (is this code more complex than it should be), 𝗻𝗮𝗺𝗶𝗻𝗴 (is naming good?), 𝗲𝗻𝗴. 𝗽𝗿𝗶𝗻𝗰𝗶𝗽𝗹𝗲𝘀 (solid, kiss, dry), 𝘁𝗲𝘀𝘁𝘀 (are different kinds of tests used appropriately, code coverage), 𝘀𝘁𝘆𝗹𝗲 (does it follow style guidelines), 𝗱𝗼𝗰𝘂𝗺𝗲𝗻𝘁𝗮𝘁𝗶𝗼𝗻, etc.

Code review is a conversation, not a queue of commands.  
  
Code review checklist  
  
A checklist helps you to create a structured approach to code reviews. Also, they remind you of all the quality checks you need to perform to approve code into the codebase.  
  
  
  
𝟭. 𝗧𝗿𝘆 𝘁𝗼 𝗿𝗲𝘃𝗶𝗲𝘄 𝘆𝗼𝘂𝗿 𝗼𝘄𝗻 𝗰𝗼𝗱𝗲 𝗳𝗶𝗿𝘀𝘁  
  
Before sending a code to your colleagues, try to read and understand it first. Then, search for parts that confuse you.  
  
𝟮. 𝗪𝗿𝗶𝘁𝗲 𝗮 𝘀𝗵𝗼𝗿𝘁 𝗱𝗲𝘀𝗰𝗿𝗶𝗽𝘁𝗶𝗼𝗻 𝗼𝗳 𝘄𝗵𝗮𝘁 𝗶𝘀 𝗰𝗵𝗮𝗻𝗴𝗲𝗱  
  
It should explain what changes were at a high level and why those changes were made.  
  
𝟯. 𝗔𝘂𝘁𝗼𝗺𝗮𝘁𝗲 𝘄𝗵𝗮𝘁 𝗰𝗮𝗻 𝗯𝗲 𝗮𝘂𝘁𝗼𝗺𝗮𝘁𝗲𝗱  
  
Leave to the system everything that can be automated,   
PMD, FindBugs, and Checkstyle are the most popular open-source code analyzers,   
Jenkins for CI-CD some code smells and bugs (SonarQube).  
  
𝟰. 𝗗𝗼𝗻'𝘁 𝗿𝘂𝘀𝗵  
  
You need to understand what is changed in every line of it. Read multiple times if required, class by class.  
  
𝟱. 𝗖𝗼𝗺𝗺𝗲𝗻𝘁 𝘄𝗶𝘁𝗵 𝗸𝗶𝗻𝗱𝗻𝗲𝘀𝘀  
  
Never mention the person (you), always focus on changes as questions or suggestions and leave at least one positive comment. Explain the "why" in your comments and suggestions for improving it.  
  
𝟲. 𝗔𝗽𝗽𝗿𝗼𝘃𝗲 𝗣𝗥 𝘄𝗵𝗲𝗻 𝗶𝘁𝘀 𝗴𝗼𝗼𝗱 𝗲𝗻𝗼𝘂𝗴𝗵  
  
Don't strive for perfection but hold to high standards. Don't be a nit-picker.  
  
𝟳. 𝗠𝗮𝗸𝗲 𝗿𝗲𝘃𝗶𝗲𝘄𝘀 𝗺𝗮𝗻𝗮𝗴𝗲𝗮𝗯𝗹𝗲 𝗶𝗻 𝘀𝗶𝘇𝗲  
  
We should limit the number of lines of code for review in one sitting. Our brains cannot process so much information at once. The ideal number of LOC is 200 to 400 lines of the core at one time, which is usually 60 to 90 minutes.   
  
8) Review logic, not semicolons  
  
Automation reduces the needless checks and lets you focus on the logic behind the changes rather than syntax errors and typos.

