# Code review standard

Skapa en struktur på vad man ska kolla igenom på en code review. För att man ska hålla sig till vissa kvalitetsstandarder. Också för att sprida kunskapen om koden.

Checklist:

* Läsbarhet
  + Förstår man vad personen menar
  + Har funktionen rimlig storlek
  + Separation of concerns
* Dokumentation
  + Kort och koncis
* Variabelnamn
  + Engelska
  + Tydligt
* Endast acceptera pull request om det är en complete feature eller kritiskt för någon annan i teamet
* Kolla igenom så att alla filer som är ändrade hör till requestens syfte

## Google stuff

https://google.github.io/eng-practices/review/reviewer/

En lista på saker Google tycker man ska kolla.

* The code is well-designed.
* The functionality is good for the users of the code.
* Any UI changes are sensible and look good.
* Any parallel programming is done safely.
* The code isn’t more complex than it needs to be.
* The developer isn’t implementing things they *might* need in the future but don’t know they need now.
* Code has appropriate unit tests.
* Tests are well-designed.
* The developer used clear names for everything.
* Comments are clear and useful, and mostly explain *why* instead of *what*.
* Code is appropriately documented (generally in g3doc).
* The code conforms to our style guides.
  + <https://github.com/google/styleguide>

·

# Definition of Done:

För att den User-Story ska räknas som klar ska den fylla upp följande:

* Code reviewed
* Acceptance criteria met
* Tests passed
* Keep NFR in mind
* Product Owner accepts the User Story

Våra tankar från handledningen:

1. Merge på acceptance criteria ger tvingad merge-request.

2. Code coverage: kör testerna när man lägger till.

3. Kolla med PO att vi är det är uppfyllt.

# KPIer:

Burn down chart för att bygga ut velocity med projektion till slutet