# Unit test

Som dokumentation af kvaliteten af Pristjek220 er Pristjek220 blevet unittestet for at sikre at klasserne i programmet lever op til de forventede krav, og kan udfører de beskrevet user stories.

Til at udfører unittestene i Pristjek220 er der blevet benyttet NUnit hvilket er et framework i C#, hvilket gør det let at lave opsætningen af testcases og tjek på at funktionerne virker. NSubstitute er også blevet benyttet til at unitteste med, Nsubstitute er også et framework hvilket gør det lettere at lave Stubs og Mocks til at tjekke tilstand og adfærd, når man tester funktionerne.

Alle funktioner i Pristjek220 er blevet unittestet med nogle få undtagelser:

* SmtpClientWrapper – Send ()
* EnterKeyPressed Funktionen
* LogInViewModel – ChangeWindow funktionerne

Disse funktioner er ikke blevet testet da der ved hvert tilfælde har været noget der har gjort vi ikke har været i stand til at teste dem.

Send funktionen har vi ikke kunne teste da den smider en exception da den bliver substitueret ud, og derfor ikke bliver instantieret, og ikke kan sende en mail.

EnterKeyPressed er ikke blevet testet da der ikke er fundet en metode at lave inputtet som er et KeyEventArgs og derfor ikke har kunne kalde funktionen.

LogInViewModel changeWindow funktionerne er ikke blevet testet da den kalder applikation, som ikke er instantieret, på grund af der igennem testene ikke køre nogle applikation.

# Integrationstest

Da unittest tester hver klasse isoleret fra resten af system, er der nogle ting som ikke kan testes igennem unittest, så som hvordan de forskellige klasser interagere med hinanden. Dette kan man så teste med integrationstest. Her bliver interfacene mellem klasserne testet da der bliver lavet funktions kald ned til klassen igennem denne. Her kan der ske fejl hvis der gennem udvikling har været tvivl om hvordan de skulle kommunikere sammen.

Til udførelse af integrationstest er der også blevet benyttet NUnit, da dette framework gør det nemt at opsætte test cases.

Integrationstestene er blevet lavet gennem en bottom-up strategi for at sikre at funktionaliteten mellem de forskellige klasser fungere helt, der er derfor skrevet test drivers til hver klasse der interagere med en anden klasse for at teste denne integrering.