TestPage: [Webhallen](https://www.webhallen.com/se/)

TestTool: Cypress

GitHub: [Repository](https://github.com/RasmusIv/Automatisering_MVT22)

CI: Github Actions

Jag valde Cypress som testverktyg. Valet var för mig ganska enkelt då jag redan från start visste att jag var ute efter ett verktyg som fungerar bra med automatiserade tester mot gränssnitt i webbmiljö. Därav försvann möjligheten att jobba med testverktyg riktade mot arbete med API:er som exempelvis SoapUI. Jag kikade på både Selenium och Cypress men valde slutligen Cypress på grund av ett par saker. En sak som slog mig direkt var att jag fann det lättare att förstå webbsidan för Cypress, den innehöll även en väldigt hjälpfull guide som fick mig på rätt spår vad gäller utformandet av testerna. Jag fick även höra från en bekant i industrin att detta verktyget används i större utsträckning än Selenium. Då vi just avslutat en programmeringskurs med Python som huvudfokus kände jag mig manad att testa ett annat språk för dessa tester, Cypress körs med JavaScript. Med denna information i åtanke installerade jag Cypress och inledde skapandet av automatiserade tester.

Under loppet av min testning kom jag nästan att ångra mitt val av webbplats. Detta eftersom att flera designändringar verkställdes som orsakade att jag var tvungen att omformulera mina tester. Ett annat problem som uppstod med webbplatsen var att den ibland vägrar svara på interaktion, detta är ett problem på deras sida eftersom jag lyckats återskapa det på två olika datorer utanför och i Cypress samt i min CI Pipeline. Detta gäller specifikt testerna relaterat till “User Account manipulation” och interaktionen med logga in/skapa konto. För att undersöka och åtgärda “flakyness” i testerna som körs genom min CI Pipeline hade jag gärna använt Cypress cloud, detta är dock ett betalverktyg och inget jag kan införskaffa för denna uppgiften. Verktyget hade gjort det möjligt att visuellt följa alla teststeg i CI Pipelinen och se vad som går fel. Jag är dock övertygad om att de flesta fel beror på webbplatsens inkonsekventa gränssnitt, hade jag haft tillgång till utvecklarna hade jag förklarat att webbplatsen måste bli mer konsekvent.

Vid valet av verktyg för Continuous Integration tittade jag både på Jenkins och GitHub Actions. Anledningen att jag slutligen valde att använda mig av GitHub Actions var att det verkade enklare att implementera och att det simplifierar setupen av en VM hostad på GitHub servers som kör testerna så man slipper ha det lokalt.  
För att sätta upp GitHub Actions i mitt GitHub repo så skapade jag en mapp “.github/workflows”, i mappen skapade jag en .yml fil som jag döpte till “main.yml”. Denna filen styr alla steg min CI Pipeline följer. I filen deklarerade jag att CI ska aktiveras varje gång något pushas till mitt GitHub repo. Jag använde GitHub Actions funktion för att välja ubuntu som OS för den VM som kör testerna. Cypress har en egen fördefinierad GitHub Action som innehåller installationen och setupen av programmet som en action, jag använder mig därför av denna för att få verktyget att fungera. För att återskapa en miljö så nära som min lokala som möjligt så valde jag att tvinga Cypress att köra i Chrome. Efter att .vml filen var skapad och upsatt så testade jag att pusha ändringar till mitt repo och kontrollerade att testerna kördes.

I mina tester försökte jag att följa ett liknande format, specifikt att namnge så få produkter som möjligt och istället peka på deras position då jag tror att produktnamn/produkt har större risk att förändras än webbplatsens design.

TestArea1: User Journey from main page to checkout.

Innan varje test använde jag setup för att säkerställa att rätt webbplats kallades och även att förfrågan om cookies godkänns.

* Test search feature

För att testa sökfunktionen på webbplatsen så kontrollerade jag att sökresultatet blev detsamma som jag input i sökfältet.

* Test add to cart feature

För att testa att lägga till ett föremål i kundvagnen så använde jag sökfunktionen igen för att söka efter ett visst föremål. När föremålet var hittat så lade jag till det i kundvagnen, sedan kontrollerade så att meddelandet som informerar att föremålet har lagts till i kundvagnen var rätt.

* Test checkout feature

För att testa webbplatsens checkout funktion så använde jag de föregående stegen för att hitta ett föremål och lägga till det i kundvagnen. Därefter kontrollerade jag så att pathnamnet var korrekt för checkout för att inte riskera att ha hamnat på fel sida. Efter detta var klargjort så kontrollerade jag så alla fält gick att fylla i och att checkout knappen då blev tillgänglig att klicka på.

TestArea2: User Account manipulation.

Innan setup steget för denna testarea så använder jag Math.random för att generera ett nummer mellan 1-999 och sätta det som prefix till en standardiserad e-postadress, jag gjorde även detsamma med användarnamnet. E-postadressen och användarnamnet genereras före testerna och setupen så de finns tillgängliga för varje test utan att genereras på nytt.

Innan varje test använde jag setup för att säkerställa att rätt webbplats kallades och även att förfrågan om cookies godkänns.

* Test add user account feature

För att lägga till ett konto matar jag in de genererade variablerna tillsammans med ett testlösenord i rätt fält. När detta är gjort så kollar jag så att knappen för att skapa kontot finns och klickar på den. I slutet av det hela postar jag användarnamnet och e-postadressen till konsolen för att kunna hålla koll på vilket nummer som genererades. Detta steget krävs inte för att testet ska fungera utan är endast med för att manuellt se vad som händer.

* Test login user account feature

I detta testet använder jag samma uppgifter som genererats innan setupen och matar in i login-menyn. Efter jag klickar på “login” knappen så dubbelkollar jag att är inne på rätt konto.

* Test add user account duplicate

För att testa möjligheten att skapa dubletter av konton genomför jag samma steg som testet för att lägga till ett konto, med samma uppgifter. I slutet av testet tittar jag även efter ett felmeddelande som informerar att det inte går att skapa ett konto med samma uppgifter.