**Shoppingapplikation för GOLDSTARR TRADING**

Release 1 & 2

Kurs: OOP1

Klass: SYSM4

Termin och år: HT 2020

Kursansvarig lärare: Fredrik Jönsson

Författare: Fisnik Haliti, Johan Rova, Jony Kiiskinen, Elin Ortega, Thomas Löhr

Sammanfattning

I denna rapport så kommer vi att beskriva vårt grupparbete och arbetssätt. Vi kommer först att gå igenom vår planering och resonemang kring denna och även beskriva hur planeringen följdes och styrkor och svagheter kopplat till det. Vi kommer även gå igenom gruppdynamik och gruppens styrkor och svagheter här.

Efter det så kommer rapporten gå igenom hur själva programmeringen och utvecklingen av programmet fortlöpte. Hur vi resonerade kring implementationen av våra idéer och hur implementationen gick rent praktiskt. Här kommer vi även gå igenom problematik som uppstod under arbetet, var vi identifierade bristande kunskap, men även var vi såg styrkor och var arbetet fortlöpte oväntat lätt.

Slutligen så kommer vi att ha en del där vi går igenom våra tankar och resonemang efter arbetet i sig. Var, och hur, gruppens medlemmar behöver utveckla sig i framtiden, både när det kommer till programmering i sig och även relaterade kunskaper som behövs av en systemutvecklare.

Innehållsförteckning

[Sammanfattning 2](#_Toc54872418)

[1. Uppdragsbeskrivning 4](#_Toc54872419)

[2. Projektarbetets mål 4](#_Toc54872420)

[3. Metodbeskrivning 4](#_Toc54872421)

[4. Resultatredovisning 5](#_Toc54872422)

[5. Analys och slutsatser 6](#_Toc54872423)

[6. Bilagor 6](#_Toc54872424)

1. Uppdragsbeskrivning

Projektet går ut på att skapa ett körbart program utvecklat i C# på Universal Windows Platform(UWP). Programmet som ska utvecklas är ett lager- och affärssystem som ska kunna hantera dagliga användningsområde som kan tänkas behövas, så som lagerhantering, orderhantering samt kundlagring.

Programmet ska utöver detta även vara designat utefter objektorienterade principer, såsom inkapsling och läsbarhet. Programmet ska även kunna hantera vanliga problem som kan uppstå till följd av användning, som felaktig inmatning.

Programmet ska vara lättanvänt och förståeligt för gemene man, utan kunskaper inom programmering.

1. Projektarbetets mål

Målet med projektarbetet är att träna och utvärdera gruppens kunskap i objektorienterad programmering, objektorienterade principer och att träna på förmågan att kunna arbeta agilt och samarbeta kring problemlösning.

Utöver detta ska projektarbetet även utvärdera kunskaper som är viktiga i arbetslivet som utvecklare, som planering och social kompetens.

1. Metodbeskrivning

I början av arbetet så bestämde vi direkt att arbetet skulle genomföras baserat på agila principer. Vi implementerade en typ av planering löst baserat på scrum, där sprintarnas tid reduceras till ibland bara en dag, detta gjordes på grund av arbetets snäva tidsplan. Vi användes oss även av en work breakdown structure (bilaga 2) samt Gantt-schema (bilaga 1). Inför Release 2 gjorde vi en mer detaljerad planering och även indelning av gruppen.

Vi gjorde en planering av hur klasser skulle implementeras och deras interaktioner mellan varandra därefter påbörjades arbetet direkt enligt ”bottom-up” principer.

Vi delade in oss i roller, projektledare Jony, sekreterare Thomas, programmerare Johan, Fisnik och Elin.

För versionshantering så användes Git, där vår centrala lagring skedde på Github.

Vår utvecklingsmiljö var Microsoft Visual Studio Community.

Våra kommunikationskanaler var Microsoft Teams samt Discord.

1. Resultatredovisning

Se kod i Github. <https://github.com/JohanRova/projektlol>

För att få en bra struktur på projektet, inledde vi med att göra en WBS och ett Gantt-schema.

Release 1: Vår produktägare har tillsammans med kunden, utformat fem krav på funktionalitet för applikationen. Samtliga av dessa krav har implementerats i applikationen, arbetet och resultatet går att följa på Github. Arbetets gång kan följas via dagliga commits och resultatet i den senast upplagda versionen.

Av de första fem punkterna på kravlistan, dvs för release 1, gjorde vi två User stories:

* As a business owner I want to be able to see a list of customers so that I can choose the right customer when I want to place a new order for them.
* As a business owner I want to be able to see a list of our products and how much we have in stock, so that I can update the list when we get deliveries.

Implementeringen av funktionalitet för release 1 är felhanterad till den största delen, eftersom ytterligare funktionalitet ska implementeras, ses felhanteringen över både bredare och djupare till release 2.

Funktionaliteten är också testad och programmet fungerar enligt förväntningar och de utformade kraven för release 1.

Release 2: Vid ett första möte med produktägaren beslutades att vi skulle göra 6 punkter till på kravlistan enligt följande:

* En vy för att lägga till nya kunder
* En vy för att lägga till nya produkter
* All data ska sparas på fil och läsas in vid programstart
* En kunds ordrar ska sparas i systemet
* En vy för att se alla ordrar
* En kö för ordrar som inte kan levereras om lagret är tomt

Vi delade in oss i två grupper och började med att dela upp de första fyra punkterna, två till varje grupp, vi lade även in allt i Gantt-schemat och gjorde upp planeringen efter hur lång tid vi trodde att det skulle ta. Se bilaga 1.

Under hela processen har vi testat och felsökt programmet och även lagt in kod för felhantering.

Vi satt även lite gemensamt och tittade på punkten om att all data ska sparas i systemet, för att alla var intresserade och ville veta hur det skulle fungera. Det var dock den ena gruppen som var ytterst ansvariga för att det kunde genomföras. Sedan delade vi även upp de två sista punkterna och blev klara med dessa. Tidsmässigt gick den här biten väldigt bra, så vid möte nummer två med produktägaren blev vi tilldelade tre punkter till:

* Sortera ordrar på datum
* En lista över leverantörer
* En vy för att lägga till nya leverantörer (Vi har en funktion för att lägga till leverantörer på Stock-sidan, men då måste man även lägga till produkt och antal produkter. Efter överläggande med produktägaren beslutades pga. tidsbrist att vi inte skulle ändra på detta.)

Vi gjorde User stories för release 2:

* As a business owner I want to be able to have a view where I can add new customers, so that I can manage our customers as our business grow.
* As a business owner I want to have a view where I can add new products, so that I can manage and edit our stock as our business grows bigger.
* As a business owner I want all my data to be save so that everything is updated and ready when I start my application.
* As a business owner I want my customers orders to be saved so that I can go back to that view and see previous orders.
* As a business owner I want to be able to sort previous orders so that they are easy to overlook and manage, whenever I need to.
* As a business owner I want to be able to see a queue of orders that couldn’t be made because there wasn’t enough in stock, so that I can make that order when there is enough in stock.
* As a business owner I want to have a view where I can see a list of and add suppliers, so that I can add new suppliers as our business grow bigger.

I ett första skede tänkte vi att detta med sorteringen skulle bli lite avancerat. Vi testade att först lägga till en Reverse-metod, men insåg att det skulle bli problematiskt om man lade till fler orders efter att man hade gjort en Reverse. Därför gjorde vi om det till OrderBy och OrderByDescending, som funkar utmärkt.

1. Analys och slutsatser

Release 1

Vi jobbade oss gemensamt fram till lösningar för G-uppgifterna. Vi fastnade lite på uppgiften om en vy för att kunna lägga en ny order åt en kund. Men vi kom fram till en lösning till slut och vi är nöjda med resultatet.

I början av arbetet så gjordes större delen av arbetet kollaborativt. Gruppen satt tillsammans och diskuterade och pratade om hur koden skulle se ut konkret och en person implementerade gruppens idéer under tiden personen delade sin skärm med gruppen. Arbetet påbörjades med detta tillvägagångssätt på grund av olika anledningar, huvudsakligen så gjordes detta för att gruppen hade svårigheter att parallellisera arbetet i detta skede, men även gruppens upplevda okunskap och avsaknad av erfarenhet i UWP gjorde att detta bedömdes som det bästa sättet att

påbörja projektet.

Release 2

Här gick vi lite mer in för att dela gruppen i två och att dela upp punkterna på kravlistan mellan grupperna. Det funkade fint och vi jobbade effektivt på två olika håll, men sammanstrålade alltid ett par gånger om dagen för att uppdatera varandra och ta del av koden den andra gruppen hade skrivit. Det har funkat bra för oss att jobba så, alla ville vara lite delaktiga i alla punkter för att kunna förstå hela programmet och för att lära sig mer, även om vi hade delat upp arbetet mellan oss.

En funktion som gruppen känner var problematiskt att implementera var funktioner för att spara och ladda programdata till fil. Funktionen implementerades utan en av C# inbyggda serialiserare, istället sparades klassdata till fil med en ToString()-override, där olika properties separerades i sparfilen med speciella tecken, såsom procent tecken. Svårigheter uppstod också på grund av att laddning av filer är asynkrona anrop. Gruppen försökte implementera en funktion som först skulle kolla om det fanns filer att ladda in data ifrån, men denna funktion fungerade inte som tänkt på grund av gruppens avsaknad av kunskap gällande asynkrona metoder. Det beslutades till slut att koden i metoderna som laddar data ifrån fil skulle sättas inom try-catch block för att hantera eventuella problem som annars uppstår om applikationen inte hittade en fil.

Testning har genomförts dels enligt testscenarion och dels genom att stegvis testa funktionaliteten i varje frame i appen.

Vi började att testa navigationen i applikationen, att man kan klicka på samtliga knappar och navigerar sig fram och tillbaka utan att applikationen ger vika.

Vidare testade vi varje frame för sig för att säkerställa att funktionaliteten i framen lokalt, beter sig som tänkt. Korrekt input ska ge förväntat beteende och när felaktig input inmatas ska applikationens felhanteringsprocess informera användaren om vad som gått fel.

Till sist testade vi ett antal scenarion:

* Företaget lägger in en kundorder för en ny kund.
* Företaget behöver lägga upp en ny produkt i applikationen innan inleveransen kan bokföras.
* Företaget får in en väldigt stor order där orderkvantiteten överstiger lagersaldot, på samtliga produkter i produktkatalogen.

Att genomföra hela arbetet på distans var något vi i början kände kunde bli lite kämpigt, speciellt med tanke på våra haltande kunskaper inom Git. Men det visade sig fungera utan problem när vi väl började. Det gick galet någon gång när vi skulle göra en merge, men det löste vi ganska enkelt och efter det var det inga problem.

I det stora hela har vi inte upplevt distansen som något negativt under arbetets gång.

1. Bilagor

1. Gantt schema

2. WBS

3. UML