02312: Opgavebeskrivelse

Christian Kalhauge: `chrg@dtu.dk`

Version 2.1

Dette er en kort beskrivelse af 3-ugers opgaven for 02312 Indledende programmering. Opgaven har til formål at teste om I har opnået kursets læringsmål; samt at give jer en smag på hvordan det er at udvikle et større stykke software. I kan læse om alle kursets læringsmål her[1].

I denne opgave er der specielt fokus på:

- Samarbejde i en projektgruppe om at designe et mindre softwaresystem ud fra en problemstilling i en bunden opgave, samt implementere og dokumentere de væsentlige dele af dette design.
- Anvende simpel tids- og aktivitetsplanlægning for et projektforløb.

Specifikt skal i

> Udvikle et Matador spil i sammenarbejde med en kunde.

I løbet af de 3 uger skal I lave en kravsanalyse og komme op med en liste af "User Stories". Implementationen af disse "User Stories" skal I planlægge, designe, implementere og validere. Opgaven har fem deadlines:

Dato	Aflevering
Mandag d 2.	G1: Gruppedannelse
Onsdag d 4.	M1: Præsentation af kravsanalyse og planlægning
Onsdag d 11.	M2: Præsentation af Minimal Viable Product (MVP)
Onsdag d 18.	F1: Aflevering af kode, og sidste præsentation.
Fredag d 20.	F2: Aflevering af rapport.

Før alle onsdage og på slutningen af fredagen, skal I aflever på DTU Learn, de præcise afleveringstidspunker vil fremgå online. Projektet skal udarbejdes i `Java` og skal bygges ved hjælp af `maven`. I er velkomne til at bruge alle open-source biblioteker (inklusiv Matador-GUI'en) I can finde på nettet så længe at I dokumentere det i både jeres source kode og i rapporten.

Evalueringen vil hovedsageligt tage udgangspunkt i rapporten, men også inkor-

[1] https://kurser.dtu.dk/course/02312

porere den indleverede kode samt jeres git-historik. Ved evaluering er der fokus på:

1. Værdi genereret for kunden,
   - Omfanget af user stories der er blevet implementeret og valideret.
2. rapportens kvalitet,
   - Forståeligheden og dokumentationen af resultatet.
3. kodens kvalitet og
   - Om der tænkt over kodens design.
   - Om koden er testet.
4. at alle har været med i udviklingen.
   - Har alle lavet commits til det fælles repository.

Rapporten er meget vigtig, så husk at skriv den lidt hen af vejen.

G1: Gruppedannelse

Der skal være mellem 4 og 6 i hver gruppe. I skal udarbejde en gruppe kontrakt, men den skal ikke afleveres. Hvis der opstår konflikt i gruppen så vil vi mediere ved hjælp af den. Her er et eksempel[2].

I løbet af dagen efter vil I få tilknyttet en kunde/hælpelære som I skal planlægge møder med hver onsdag.

M1: Kravsanalyse

Her skal I præsentere jeres kravsanalyse.

- En tabel user stories afleveres på DTU Learn dagen før i PDF format: `m1.pdf`.
- På dagen haves en 10-15 min. præsentation af user stories og en diskussion om hvad der skal afleveres ugen efter (MVP).
- Kunden kan komme med flere krav.

I skal bryde opgaven ned i en liste af user stories; samt hvad I har planer om at levere til M2 altså definere jeres MVP.

En user story har til formål at repræsentere værdi for kunden. Vi skriver dem sådan her: "Som en `<Hvem>` vil jeg gerne `<Hvad>` sådan at `<Hvorfor>`". Vi skriver `<Hvem>`, så vi ved hvem vi generer værdi for, `<Hvad>` sådan vi ved hvad der skal laves (kravet) og `<Hvorfor>` sådan at vi forstår hvorfor det vil generer værdi for kunden.

> Som en spiller, **vil jeg gerne** modtage penge når jeg passere start
> **sådan at** jeg kan have flere penge til rådighed over tid.

For hver user story, skal vi også generer en eller flere acceptance tests. Dem skriver vi: Givet `<Forudsætning>` så `<Resultat>`. Med `<Forudsætning>` mener

[2] https://docs.google.com/document/d/1zfpBltcl2HmoymUpAaNstZ3qWUEMWs40IBQzvvq-2Sk/edit#

vi en beskrivelse af hvordan verden skal se ud for at testen er i gang. Med `<Resultat>` mener vi hvad vi forventer at systemet skal gøre:

> **Givet** at en spiller passere start OG ikke er på vej i fængsel **så** skal spiller have 4000 kr mere.

Hver user story skal have et Id, så den er nem at referere til.

Id	User Story
K1	En spiller **skal** modtage penge når de passere start **sådan at** der bliver flere penge til rådighed i spillet over tid.

M2: Minimal Viable Product

Her skal I præsentere et spilbart spil, eller et MVP.

- Upload en kort status rapport dagen før i PDF format: `m2.pdf`.
- Lægge koden på Github under tag `v0.1` .
- Giv en 5 - 10 min præsentation af hvor langt I er nået.
- Demonstrer 1 - 2 accepttests med kunden køres på den version U har tag'et.
- Kunden bliver måske inspireret til at stille flere krav.

Et minimal viable product (MVP) er den mindste del af et stykke software, der sammen giver værdi til kunden. Ideen ved et MVP er at kunne involvere kunden så tidligt som muligt i udviklingen. Det er nemlig nemmere når man kan snakke ud fra et stykke virkende software.

> F.x. Ved udviklingen af en lommeregner, kan et MVP være en lommeregner der lægger tal sammen i terminalen. Funktioner som GUI og multiplikation kan tilføjes senere hvis kunden virkelig har brug for det.

Vær opmærksom på at vi forventer at MVP er et afrundet stykke software, der i princippet kunne brugertestes.

I status rapporten skal alle de user stories I har implementeret og de accepttest i har kørt være med. Her kan I med fordel følge formatet i F2. Der udover skal der også være et link til jeres GitHub repository, samt det commit hash i vil præsentere fra som er taget med `v0.1` (se i F1, hvordan i gør det).

F1: Kode

Koden skal afleveres før præsentationen Onsdag den 18.

- I skal aflevere en fil `f1.bundle` med et tag `v1.0`, og I skal aflevere en status rapport `f1.pdf`
- Lægge koden på Github under tag `v1.0`.

- Skrive commitet ned i rapporten.
- Præsentationen (10-15 min) tager udgangspunkt i de user stories I har nået og skal køres på den version af koden I har afleveret.
- Kunden vælger nogle user stories I skal demonstrere accepttest for.

Koden skal indeholde in README fil, der forklare hvordan man bygger, kører og tester programmet. Husk at teste at det hele virker på en frisk "clone" af repositoriet på en ny computer.

I afleverer ved at tag'e det commit som I vil aflevere, med navnet `v1.0`. Det gøres således:

```
$ git tag 'v1.0'
$ git push --all    # Læg tag'et og koden på github.
$ git show -q v1.0  # Print commit hashet for tag'et
                    # så I kan skrive det i rapporten.
```

I burde også kunne finde commitet online på GitHub under, hvis I bytter `<owner>/<repo>` ud med jeres repository navn.

`https://github.com/<owner>/<repo>/releases/tag/v1.0`

I skal også uploade en kopi af jeres repository til DTU learn. Det gøres således:

```
$ git bundle create f1.bundle --all
```

Dette burde lave en `f1.bundle` fil som I kan aflevere.

I kan med fordel lave jeres sidste accepttest på denne bundle; så er I sikker på at I aflevere det rigtige.

```
$ git clone -b v1.0 f1.bundle matador-final
```

Så burde I kunne køre jeres test fra projekt folderen `matador-final`.

Status rapporten skal indeholde de user-stories I har implementer med accepttest, samt github link og commit hash. Ligesom i gjorde i M2.

F2: Rapport

Rapporten skal afleveres slutningen af arbejdsdagen, Fredag den 20.

- I skal aflever en PDF fil `rapport.pdf`.

1. Gruppenummer, Navn og studienummer på alle gruppe medlemmer.
2. Et link til jeres Github page, samt commit hash på koden (vær sikker på at det er det commit `v1.0` pejer på).
3. En beskrivelse af fordelingen af arbejdet.
    - Enten laves en tabel med gruppemedlemmer som søjler og sektioner som rækker. En række for programmering bør også inkluderes. Skriv andel brugt i hver sektion:

Sektion	Per	Jytte
Indledning	0,25	0,75
Design	0,50	0,50
..	..	..
Programmering	0,50	0,50

   - Ellers, skrives bare at arbejdet er fordelt ligeligt.
4. Rapporten må ikke overskride 36 normalsider (2400 anslag), appendix og diagrammer ikke talt med.
5. Alle figurer skal have brugers til at illustrer ting som I beskriver i tekst. Figurer må derfor ikke stå alene.

Rapporten skal skrives sådan at udviklere på jeres niveau, kan læse rapporten og være i stand til at videreudvikle på jeres system. I skal komme ind på jeres kravsanalyse, planlægning, design, implementation og validering (hvordan I har testet systemet).

Rapporten skal indeholde en eller flere tabeler (gerne i appendix) med de user stories som I har planlagt og præsenteret for kunden. Tablen skal indeholde felterne:

1. id: fx. K1, US1, USO1 (User Story for oprettelse 1)
2. system (kan unlades hvis der er en tabel per system): et tag beskriver hvilken del af spillet som user storyen retter sig til. fx.
   - "oprettelse",
   - "kerne" (slå med terninger, rykke brikker, tabe/vinde),
   - "fængsel",
   - "grunde",
   - "prøv lykken",
   - og meget mere.
3. beskrivelse: fx. "Som en opretter af spillet **vil jeg gerne** kunne lave et spil med op til 6 spiller **sådan at** de kan spille sammen."
4. status ved hver præsentation (M1, M2 og F1):
   - 0 (Ikke planlagt endnu)
   - P - Planlagt (Det er estimeret hvor stor opgaven er, og en accepttest er defineret.)
   - D - Designet (Et design er klar, men er endnu ikke implementeret)
   - I - Implementeret (En implementation er blevet lavet)
   - V - Valideret (Implementationen er testet, og det virker)

Id	System	User Story	M1	M2	F1
K1	Oprettelse	Som en opretter af spillet **vil jeg gerne** kunne lave et spil med op til 6 spiller **sådan at** de kan spille sammen.	P	V	V

Rapporten skal indeholde en eller flere tabeler (gerne i appendix) med de accepttest I har for hver af de validerede user stories. **En user story kan kun være valideret (V) hvis den har en eller flere accepttest som er kørt**. Tabelel skal indeholde de følgende felter:

1. User Story id / test id
2. Testen.
3. Den sidste Virkende Version på koden (brug "tag" metoden på git)
4. Navnet på testeren.

USId/Id	Test	Virkende Version	Navn
K1/1	Givet at en opretter starter et nyt spil så har de mulighed for at taste op til 6 navne ind.	v1.0	Christian
K1/2	Givet at har tastet 4 navne, "Jeppe", "Ida", "Muhammad", "Ivan" ind så skal der være oprettet 4 spiller med de navne.	v1.0	Christian

Errata

Version 2.1

- Udybet krav til sektionsfordelingen.

Version 2.0

- Jeg skrev at en User Story skal skrives, sådan her: En spiller **skal** modtage penge når de passere start, **sådan at** der bliver flere penge til rådighed i spillet over tid. Men det er i virkeligheden er det bedre at skrive fra synspunktet af en spiller: "Som en spiller, **vil jeg gerne** modtage penge når jeg passere start **sådan at** jeg kan have flere penge til rådighed over tid." Begge dele er fint, men den sidste version er nemmere at udlede værdi

fra.

- Jeg har skrevet “acceptance test”, det danske udtryk er accepttest. Det er blevet rettet.
- Tilføjet et felt til tablen, “system”, og I kan også opdele user stories efter system.
- Tilføjet et krav om at accepttestene også skal være en del af rapporten.
- Uddybning af beskrivelsen til afleveringerne M2 og F1.