



ESTATE BROKERS

18.12.2020

University College Lillebælt

VEADM20FD20X

Andreas Brosbøl

Andreas Brosbøl

Jesper la Cour

Jesper la Cour

Lasse Hündt Hansen

Lasse Hündt Hansen

Marc Hielscher

Marc Hielscher

Indholdsfortegnelse

Indholdsfortegnelse	1
Indledning	4
Problemstilling	4
Problemformulering	4
Tilgangsvinkel	4
Gruppekonspekt	5
Gruppedlemmer	6
Rollefordeling (SCRUM)	6
Generelle aftaler	6
Forfatter	6
Kvalitet i programmering	6
Projektetablering	7
Procesvalg	7
SCRUM	7
UP Artefakter	7
Sprint Plan (initial plan)	7
Timeboxing	8
Værktøjer	9
Visual Studios	9
Github	9
Azure DevOps	9
Draw.io	9
Teknologi	10
UML-Notation	10
Git	10
C# og Windows Forms	10
MySQL	11
Virksomhedsanalyse	11
PEST	11
Porters Five Forces	12
Porters værdikæde	13
SWOT	14
BPR	14
Systemanalyse (pre-sprint)	17
FURPS+	17
Domain diagram	18
Use Cases	19

System Sequence Diagram	22
Database	23
Product Backlog	24
Sprint 1	25
Kravs- afklaring / specifikation	25
Design	25
GUI	27
Mapning af ER diagram	27
Programmering	28
Test	29
Deploy	30
Sprint 2	32
Kravs- afklaring / specifikation	32
Design	32
ER	32
GUI	32
Programmering	33
Test	34
Deploy	34
Sprint 3	35
Kravs- afklaring / specifikation	35
Design	35
Programmering	35
Threads	36
Test	36
Deploy	37
Sprint 4	38
Kravs- afklaring / specifikation	38
Design	38
Programmering	39
Test	40
Deploy	40
Kvalitetssikring	42
Konklusion	42
Proces (SCRUM)	42
Perspektivering	43
Litteraturliste	44
Bilag	45



Indledning

Problemstilling

Virksomheden Estate Brokers Ltd har stillet os et ønske om udarbejdelse af et nyt skræddersyet informationssystem til håndtering af administrative arbejdsprocesser som er forbundet med et hussalg. Estate Brokers Ltd betegner sig selv som et *traditionelt* ejendomsmæglerfirma og må antages at være velfunderet i branchen. Der findes ikke mange oplysninger om nedskrevne arbejdsgange, men der arbejdes selvfølgelig efter nogle normer eller uskrevne *way-of-working*.

Problemformulering

Der ønskes et informationssystem som kan håndtere de daglige arbejdsopgaver for en mægler. Herunder med funktioner som kan beregne udbudspriser ud fra *bløde* faktorer for desuden at inkludere ufravigelige egenskaber ved et hus.

Til understøttelse af implementeringen vil vi desuden skulle argumentere ud fra et økonomisk perspektiv om hvorvidt investeringen i udvikling vil kunne forsvares med tanke på *Return On Investment*.

Tilgangsvinkel

Vi vil igennem den kommende opgave benytte os af faglige og relevante metoder, teknikker og principper fra områderne virksomhed, systemudvikling og programmering. Dette med henblik på, at kunne lave et veldesignet informationssystem der selvfølgelig opfylder kundens krav, men dels også viser høj kode kvalitet, brug af anerkendte mønstre og med fuld udskiftelighed.

Vores tilgang til udviklingsprocessen vil være agilt og igennem iterationer, da vi tror på, at det vil give os en højere effektivitet, mere gennemsigtighed og løbende mulighed for at tilpasse os forandringer.

Gruppekontrakt

Hvad?	Hvordan?
Forventninger for projekt og gruppearbejde	Vi i gruppen har snakket om det fælles ønske, at udarbejde en rigtig god opgave som vi alle vil være glade for, at sætte vores navn på. Vi ønsker at arbejde målrettet og fast på, at møde de krav der stilles i opgaven således at vi kan løse dem efter bedste evne.
Aftaler om møder i gruppen	Vi forholder os til SCRUM framework se punkt SCRUM under Procesvalg. <ul style="list-style-type: none"> - Sprint planning holdes fysisk hver mandag - Daily stand up holdes virtuelt (discord) Skulle man være forhindret i, at møde gives der besked til gruppen, så tidligt som muligt.
Løsning af uenighed i gruppen	Ved større uenigheder om design af programmet forholder vi os til designprincipper, anerkendte mønstre og flertallets magt i gruppen. Der argumenteres fagligt for sin sag. Ved mindre uenigheder om kode valg er det op til den ansvarlige for området at træffe beslutninger, så længe det lever op til "kvalitet i programmering" og med respekt for implementering i programmet.
Forventninger til tidsforbrug	I SCRUM planlægges der med en arbejdsindsat svarende til 40 timer ugentlig. (svarende til 5 dage x 4 gruppemedlemmer) 20 dage. Hvornår det enkelte gruppemedlem vælger at lægge sine timer er op til den enkelte. Der forventes at deadlines overholdes.
Kvalitet i programmering	Vi forpligter os til at overholde de aftaler i punktet Kvalitet i programmering (side #)
Konsekvens af manglende overholdelse af ovenstående aftaler	Vi er i gruppen enige om en politik om "three strikes and you're out". I tillæg til dette forventes der respons til gruppen på dagligt plan og mangel på dette vil medføre udtrædelse. Overordnet vil meget kunne tilgives, så længe der er kommunikation og udmelding i god tid.
Ved udtrædelse af gruppen	Det fælles fremstillede vil til enhver tid tilhøre gruppen. Ved udtrædelse får man udelukkende råderet over egenproduceret indhold. Gruppen råder fortsat over det indhold som det udtrådte medlem har produceret.

Gruppemedlemmer

Lasse Hündt	lass896a@edu.ucl.dk	51260663
Marc Hielscher	marc3411@edu.ucl.dk	31260636
Andreas Brosbøl Lauridsen	andr38d8@edu.ucl.dk	22914049
Jesper la Cour	jesp06f2@edu.ucl.dk	52501090

Rollefordeling (SCRUM)

Vi har fra start af projekt aftalt følgende rollefordeling i henhold til SCRUM. Rollerne vil varetages som "åbne" for gruppens input.

Product owner: Lasse Hündt (suppleret af vejleder)

Scrum Master : Jesper la Cour

Development Team:

Andreas Brosbøl

Marc Hielscher

Lasse Hündt

Jesper la Cour

Generelle aftaler

Forfatter

De enkelte afsnit i rapporten er angivet med pågældende forfatter. I koden angives der også med dato for oprettelse. Evt. større rettelser i kode / afsnit angives også.

Hvis der ikke er indikeret en forfatter på et afsnit er afsnittet skrevet i fællesskab og hvert gruppemedlem kan ses som ansvarlig.

I tillæg vedlægges oversigt fra GitHub med commits / bidrag.

Kvalitet i programmering

Vi vil i udviklingen arbejde med metoderne omkring Clean Code¹ ved konkret at benytte os af *Meaningful Names* ved navngivning af klasser og metoder. Vores metoder vil samtidig være så små som muligt og kun udføre én opgave.

Men vigtigst af alt vil vi minde hinanden om *The boy scout rule*.

¹ Clean Code by Robert C. Martin (Uncle Bob)

Projektetablering

Procesvalg

SCRUM

Forfatter: Jesper

Vi benytter os af Scrum som vores agile udviklingsproces. Dette til dels for at forberede os bedst muligt på en yderst anvendt proces, men også fordi det giver os en rigtig god mulighed for at optimere vores indsats og skabe et overblik igennem opgaven.

Det er besluttet at vi anvender Scrum omkring hele projektet. Dette inkludere også elementer i rapporten, så som virksomhedsanalyse, projektetablering og kravspecifikationer. Det vil sige elementer som man vil kunne argumentere for oftest vil gå forud for udviklingsprocessens start.

Vores mål med at anvende scrum er at følge den i så vid udstrækning som muligt og i dens oprindelige form. Når det er sagt, så er vi åbne for tilpasninger, så længe det ikke går direkte mod principperne for agil udvikling.

Til at styre vores Backlog, Sprints etc. benytter vi Azure DevOps da det giver gode mulighed for styring af projekt. Specielt set i lyset af, at meget af vores udvikling/projektudarbejdelse vil foregå online. Som beskrevet under punktet *Værktøjer* har vi valgt ikke, at benytte versionsstyringen i DevOps, men derimod bruger vi GitHub.

UP Artefakter

Eftersom Scrum er en ren udviklingsmetode med fokus på projektstyring, vil vi også anvende artefakter kendt fra Unified Process (UP) i forbindelse med kravspecifikation og design. Det er vigtigt at bemærke, at vi ligeledes anvender Use Case artefaktet fra UP i stedet for User Stories.

Sprint Plan (initial plan)

Forfatter: Jesper

Vi har planlagt 5 sprints startende fra uge 47. Forudsætningen for dette er, at vi har en færdiglavet virksomhedsanalyse, samt gennemført et pre-systemanalyse med overordnede use-cases, generelle valg og tanker omkring arkitektur.

Hvert sprint vil derefter indeholde følgende elementer i det omfang det er nødvendigt.

- Analyse

- Design
- Programmering
- Test
- Deployment

Timeboxing

Vi angiver vores sprint til en varighed af 5 hverdage startende hver mandag med et Sprint planning.

Event	1	2	3	4	5
Sprint planning	32	32	32	32	32
Daily Scrum	8	8	8	8	8
Sprint review	4	4	4	4	4
Sprint retrospective	4	4	4	4	4
Total Scrum event (timer	48	48	48	48	48
Timer til rådighed i teamet	160	160	160	160	160
Procentdel af tid i scrum møde	30%	30%	30%	30%	30%

**Angivet i timer og for hele teamet*



Værktøjer

I denne opgave ville der blive brugt en række værktøjer det ville hjælpe os med at få udført vores rapport og program. Der ville blive brugt flere forskellige programmer og hjælpemidler som kan hjælpe os med at få løst forskellige delopgaver, samt give os selv, og jer et visuelt syn på hvad der bliver snakket om, og hvordan opgaverne er løst.

Visual Studios

Der ville i opgaven blive gjort brug af Visual Studios som er et program med kode editor indbygget, hvor der er mulighed for at skrive i flere forskellige kodnings sprog.

Github

Github ville i denne opgave blive brugt som vores versionsstyring program som også tidligere nævnt i rapporten. Der ville blive gjort brug af Github til vores programudvikling, for at der kan arbejdes på forskellige sektioner af programmet på samme tid. Dette giver os en mulighed for at have en backup af tidligere versioner af programmet samt, pushe nye tilføjelser hvor de kan blive kigget igennem af alle gruppemedlemmer før de bliver merged sammen.

Azure DevOps

Som tidligere nævnt ville vi gøre brug af Azure til vores sprints, backlog samt holder styr på vores opgaver, og hvem der er ansvarlige for de forskellige ting, hvilket giver scrum master og teamet et meget bedre overblik over opgavefordelingen. Der ville blive sat tasks under hver sprint som har en varighed på 5 arbejdsdage. Disse tasks ville så blive delt ud på gruppens medlemmer så der er sikkerhed i at alle tasks bliver udført og hvem der står for den enkelte task. Det ville også være muligt for teamet at følge med via et *burndown* chart som giver os overblikket over om vi har nået hvad vi skulle eller er kommet foran, eller at vi er bagud i forhold til de planlagte opgaver.

Draw.io

Der ville også bliver gjort brug af draw.io hvor størstedelene af rapportens, og opgavens diagrammer ville blive opbygget i. Dette er en platform der giver mulighed for skabeloner, som vi kan bruge til at hjælpe med at lave diagrammer som vi får brug for at kunne vise visuelt. Samt for at give os selv et bedre overblik over hvad der skal laves, og hvad der skal tilføjes i forskellige program situationer samt i vores database.

Teknologi

Når vi snakker om teknologier og vores brug af de forskellige, er der tidligere blevet nævnt nogle af tingene i punktet værktøjer. Der ville i dette afsnit blive snakket mere omkring hvad de forskellige teknologier er/kan og hvordan vi kommer til at bruge dem.

UML-Notation

Der ville blive gjort brug af UML som er en notationsform. Diagrammerne har en bestemt metode de bliver lavet på. Der ville i vores opgave blive lavet diagrammer som er lavet efter principperne i en UML-Notation, dette ville vi gøre da det ville hjælpe os da vores opgave omhandler objektorienterede programmering som hænger godt sammen med de notationer der bliver brugt i UML.

Git


I vores program ville der også blive benyttet Git. Git er et versionsstyringssystem som hjælper os med at gøre vores program mere effektivt, samt give det mere hastighed. Det ville være med til at fjerne unødvendige kode og andre ting som programmet ikke behøver som gør at det ikke bliver et langsomt program, med filer der ikke bliver brugt. Når der bliver ændret i kode i projektet bliver det udført ved at være opkoblet til github, så alle i gruppen kan tjekke det nye kode ud før det bliver published.

C# og Windows Forms

Der ville i vores udførelse af et program blive gjort brug af kodnings sproget C#. C# er et objektorienteret programmeringssprog som vi ville bruge til at udføre vores version af et færdigt program, ved brugen af console applications, og brugen af Windows Forms. Vi ville bruge Windows Forms som har indbygget et grafisk klasse bibliotek til at få lavet et visuelt resultat som gerne sidste ende skulle kunne bruges af en eventuelt kunde.

MsSQL

Der ville blive gjort brug af MsSQL (Microsoft SQL Server) som er et database system. Her i ville vi gøre brug af sproget SQL til at kode i vores database som skal indeholde informationer der kan bruges i vores program. Databasen ville få en sammenhæng med vores program i Visual Studios som gør at programmet og opgave bliver en helhed. Dette



ville give os muligheden for at lagre informationer som bliver indtastet i vores Windows Forms application når kunden eller en medarbejder udføre en handling.

Virksomhedsanalyse

forfatter: Jesper

Til at skabe en større forståelse for virksomheden vil vi først skabe et overblik over den situation som Estate Brokers Ltd på nuværende tidspunkt befinder sig i. Dette med henblik på at klarlægge interne og eksterne styrker/svagheder og synliggøre, samt underbygge, den værdi udvikling af et informationssystem vil give virksomheden.

Når vi anskuer en virksomheds relationer til konkurrenter og markedet er det oftest et komplekst system og vi vil derfor benytte veldefinerede modeller til at skabe fokus. Fordelen ved at trække en model ind over denne case er, at vi som udviklere bliver tvunget til at se det mere objektivt og måske endda fra vinkler der ikke falder os naturligt.

Til at underbygge data i vores SWOT (konklusion model) vil vi derfor anvende følgende modeller med dertilhørende begrundelse.

- PEST modellen

Medtages i opgaven fordi den omfatter eksterne samfundsmæssige forhold der påvirker virksomheden. Vi har afgrænset os til modellen med kun 4 faktorer.

- Porters Five Forces

Anvendes til at analysere den pågældende branche som Estate Brokers Ltd er den del af. Specielt er det en vigtig model i forhold til branchens attraktivitet og afgørende for valg af forretningsstrategi.

- Porters værdikæde

Medtages for at vise virksomhedens evne til at skabe værdi og består af henholdsvis primære og sekundære aktiviteter.

PEST

Political

Man kan på ingen måde komme uden om at politiske beslutninger historisk set altid har haft en stor indflydelse for en ejendomsmægler. Samtidig ses der ofte referencer mellem et lands sundhed og tilbagegang/fremgang på boligmarkedet. Der kan med kort varsel indføres nye afgifter/lempelser eller strengere krav til gavn for miljø og samfund.

Economical

Som allerede antydnet ovenfor, så ses der altid stor sammenhæng mellem samfundets sundhed og situation på boligmarkedet. Er der højkonjunktur er der også oftest godt gang i boligsalget. Men det modsatte gør sig ligeledes gældende. Dette foruden at tale om historisk lave renter og øget økonomisk optimisme igennem de senere år. Skal man tro på principperne inden for konjunkturer, så kan dette både kategoriseres som en mulighed og en trussel for Estate Broker Ltd, men med afsæt i de senere år, så vurderes den økonomiske optimisme i samfundet en stor mulighed for virksomheden på nuværende tidspunkt.

Sociocultural

Nøgleordet for socio-kulturelle faktorer er uden tvivl demografi. Eftersom Estate Brokers Ltd ikke lader sig begrænse til enkelte områder, antager vi det som en mulighed for selektivt, at kunne arbejde fokuseret i de områder af landet der i højere grad oplever øget købekraft sammenlignet med andre områder.

Technological

Med let risiko for at bevæge os tilbage i socio-kulturelle faktorer, så er der i samfundet sket en væsentlig ændring i vores stræben efter at anvende teknologi i hverdagen. Næsten uanset hvad vi foretager os, så sker det med en telefon i hånden. Information kan findes med det samme, oplysninger kan deles og alt er gemt. Måske er det den største trussel for en veletableret ejendomsmægler eller i virkeligheden er det en yderst vigtigt mulighed for at differentiere sig fra sine konkurrenter.

Porters Five Forces

Kundernes- / leverandørernes forhandlingsstyrke

Det er valgt at sammenlægge disse to faktorer, da det kan være svært at skelne mellem hvad der anses som kunde og leverandør i branchen. Fremadrettet refereret til som kunde. På trods af ofte bundne aftaler mellem mægler og kunde ved ejendomssalg, så har kunden let ved at udtræde af en aftale ved manglende salg og se sig om efter en anden mægler. Dette uden de store omkostninger taget de ofte høje salgsbeløb i betragtning.

En kunde skal ligeledes sjældent lede længe efter en mægler, da der ofte bruges mange penge på marketing og synliggørelse.

Truslen fra nye indtrængere

Her ses få eller næsten ingen barrierer for en statsautoriseret ejendomsmægler der gerne vil starte som selvstændig. Specielt hvis man ser bort fra graden af rivalisering i branchen. Der stilles ligeledes ikke de store krav til tunge anlægsudgift, produktionsomkostninger samt lagerbinding. Markedsføring kan komme hen ad vejen og balanceres ved brug af ekstern bureau og løbende indtægter.

Truslen fra substituerende produkter

Hvis vi angiver den klassiske mægler-ydelse som produkt, så har der igennem tiden selvfølgelig været forsøg på at skabe et reelt substituerende service (selvsalg). Det har desværre ikke været muligt at finde decideret tal på andelen af handler via *ikke klassisk mægler-ydelse*. Men hvis det antages at information er blevet lettere tilgængelig og mange arbejdsgange inden for tinglysning og registrering af handel er blevet digitaliserede, vil det være nærliggende at konkludere eksempelvis selvsalg som en substituerende trussel.

Graden af rivalisering

Med en branche præget af flere meget dominerende mæglerkæder, så ses der også stadig nogle mindre selvstændige mæglere der søger en niche i markedet. Det kan være feriehus, landejendomme eller liebhaveri i ofte geografisk afgrænsede områder. Den udbudte service ses som homogent og forsøg på nye tiltag ses hurtigt kopieret hos konkurrenter. Differentiering er altså overladt til andre faktorer end på produktet.

Porters værdikæde

Primære aktiviteter

Logistik og produktion

Den indgående strøm af nye boliger er uden for Estate Brokers Ltd direkte indflydelse og uden garanti for kontinuerlighed. Selvfølgelig kan den påvirkes via marketing og opsøgende arbejde af kædens mæglere. Ved produktion tilføres der værdi via veludført salgsmateriale, gode billede og videomateriale.

Marketing, salg og service

En mæglers primære værdiskabende aktiviteter må antages at ligge i down-stream² forløbet af værdikæden. Der skabes værdi via diverse både digitale og trykte annonceringer, *Åbent hus* arrangementer og 1-1 fremvisninger. I tillæg tilbydes forskellige serviceydelse som dokument arbejde i forbindelse salg og overtagelse.

Sekundære aktiviteter

Virksomhedens infrastruktur

Estate Brokers Ltd viser stor interesse i udvikling af interne systemer til forbedring og digitalisering af virksomhedens forretningsgange. Det må antages at planlægning og koordinering i øjeblikket sker digitalt men decentralt. Den enkelte mægler har derfor ikke nødvendigvis adgang til hinandens aftaler og information om igangværende sager.

Menneskelige ressourcer

Vi må antage, da andet ikke er oplyst, at virksomheden har de nødvendige ressourcer/viden blandt sine medarbejdere til at løfte de arbejdsopgaver der forventes.

² Hejlsberg, Anna "*guide-til-brug-af-vaerdikaeden*", annahejlsberg.dk

Produkt- og teknologiudvikling

Virksomheden har som udgangspunkt fulgt med sine konkurrenter i forhold til at udbyde de ydelser der forventes i forbindelse med ejendomssalg. Der oplyses dog ikke om tiltag, udvikling af ideer eller brug af ny teknologi som ligger ud over hvad der må forventes i branchen.

Estate Brokers Ltd har dog på sigt en interesse i at udvide internationalt og eventuelt via samarbejde med anden mægler.

Indkøb / forsyninger

Der er ikke nævnt nogen fast aftale om eksempelvis projektsalg, så det må antages at virksomheden er meget afhængige af løbende opsøgende mæglerarbejde for at sikre et flow af indgående boligere.

SWOT

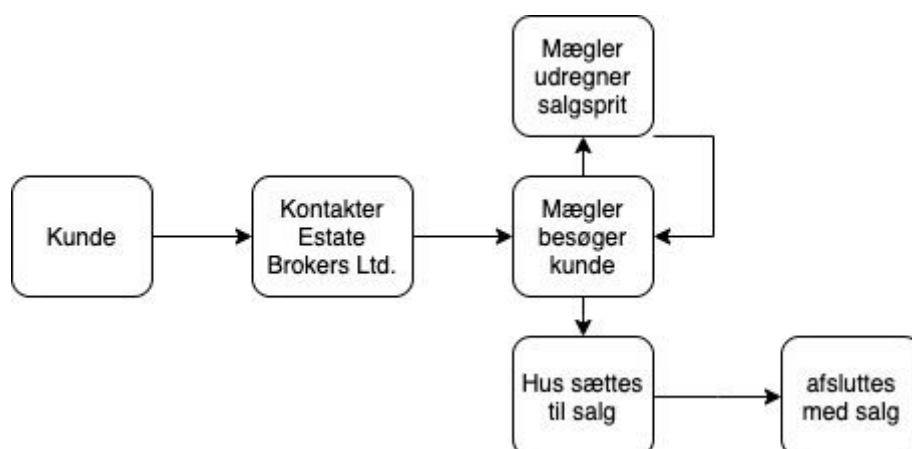
<p>Strengths</p> <ul style="list-style-type: none"> • Værditilførelse i down-stream • Kan varetage alt i relation til ejendomssalg • Nødvendige ressourcer/viden blandt medarbejdere 	<p>Weaknesses</p> <ul style="list-style-type: none"> • Mindre indflydelse på nye boliger (indgående) • Ingen aftale om projektsalg • Ingen eksisterende forretningsgange
<p>Opportunities</p> <ul style="list-style-type: none"> • Økonomisk optimisme i samfundet • Øget købekraft i udvalgte områder • Teknologisk udvikling og brugen af digitale løsninger 	<p>Threats</p> <ul style="list-style-type: none"> • Fremtidige politiske beslutninger • Kunden har gode muligheder for valg af mægler • Få tekniske barrierer for nye ejendomsmæglere. • Høj rivalisering i branchen

BPR

Workflow

Hvis vi tager udgangspunkt i nuværende forventede workflow (som skitseret nedenfor), så ser vi flere steder, hvor en *reengineering* vil være anbefalelsesværdigt.

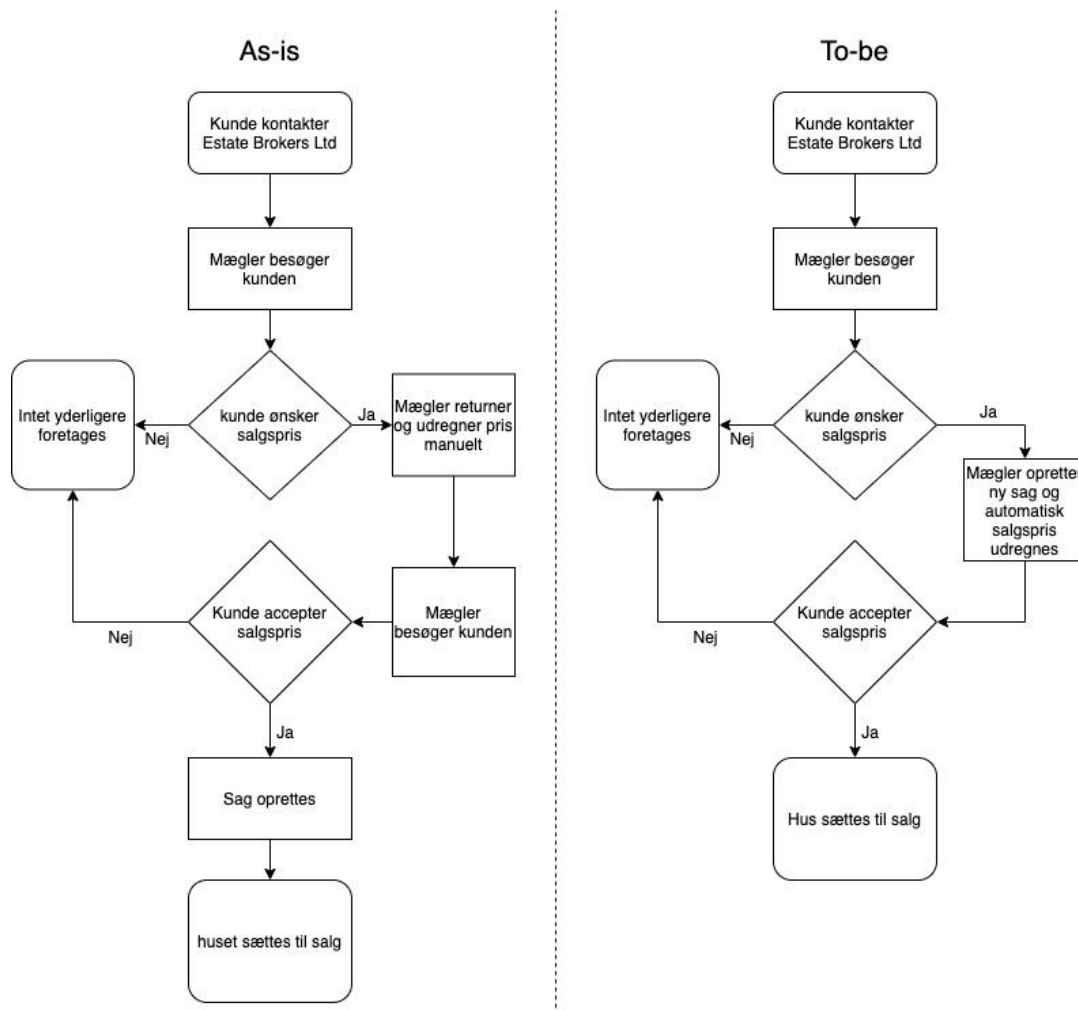
Eksisterende salgsproces som håndteres manuelt



Først og fremmest ser vi ikke nogle ensrettede arbejdsgange, hvilket der også udredes for i virksomhedsanalysen. En irrelevant eller ikke eksisterende proces er oftest en forudsætning for *reengineering*. Samtidig vurderer vi at implementering af et informationssystem vil kunne have en høj impact på virksomhedens strategisk mål og vil være afgørende for forbedringer i produktivitet.

Nuværende manuelle arbejdsproces må samtidig antages for at være under *best-in-class* for branchen og dermed ser vi ikke en *automation* eller *streamlining* som tilstrækkelige til at opfylde virksomhedens krav.

As-is, To-be



Investering

Ved fuld implementering af et digitalt informationssystem ser vi derfor gode muligheder for *return on investment (ROI)* på flere område.

Af eksterne faktorer ser vi stadig stigende optimisme hos kunderne og dermed ingen umiddelbare trussel mod virksomheder i mæglerbranchen. Det er en branche som selvfølgelig har en høj rivalisering, men samtidig ser vi mulighed for de mæglere, der omstiller sig digitalt.

Internt vil det lette arbejdsbyrden ved den enkelte medarbejder og dermed frigøre flere ressourcer til opsøgende mæglerarbejde og værdiskabende aktiviteter hos kunden. Netop det opsøgende arbejde vil kunne modvirke manglende på et kontinuerligt flow af ejendomme og de værdiskabende aktiviteter (down-stream i værdikæden) ses som en styrke hos virksomheden.

Systemanalyse (pre-sprint)

Forfatter: Marc, Jesper og Andreas

FURPS+

Functionality

- Programmet skal kunne håndtere samtlige CRUD opgaver forbundet med salg af en ejendom.
- Der forventes mulighed for at kunne udskrive salgsstatistikker på månedsniveau.
- ydermere skal programmet kunne beregne salgsspriser ud fra algoritmer på antal kvm, beliggenhed, byggeår mv. Dette for at kan gøre daglige arbejdsgange lettere for brugeren(ne)

Usability

- I henhold til brugervenlighed skal brugergrænsefladen være let forståeligt, nem at navigere i og benytte et veludviklet UI. Medarbejderne vil kunne benytte sig af programmet uden større problemer i hverdagen, for at kunne optimere tid og effektiviteten i deres opgaver.
- I forhold til support vil programmet blive udviklet således at brugeren får information igennem brugergrænsefladen. Der vil være hjælp i forhold til hvor brugeren befinder sig i programmet, såsom at hvis cursoren holdes over et felt vil der komme en dialog op der giver hjælp i forhold til givent felt.

Reliability

- Systemet forventes at kunne etablere forbindelse til andre systemer og database, hvori vi derfor lægger vægt på minimal downtime. Systemet baserer sig dog på en ekstern hosted database, som gør datatab i vores program mindre sandsynligt ved nedbrud.

Performance

- Når man ser på performance i forhold til programmet er det vigtigt at der er hurtig responstid. Dette gøres ved at programmet har clean kode, og er optimeret så godt som muligt. Ved at udnytte disse ting skulle man mindske down times i systemet, og det ville gøre at programmet kører mere optimalt. Disse ting ville også gøre at Mæglerne får mulighed for at bruge programmet med hurtigere responstid, så de ikke bliver udsat for et langsomt system når de tilgår de forskellige funktioner.

Supportability

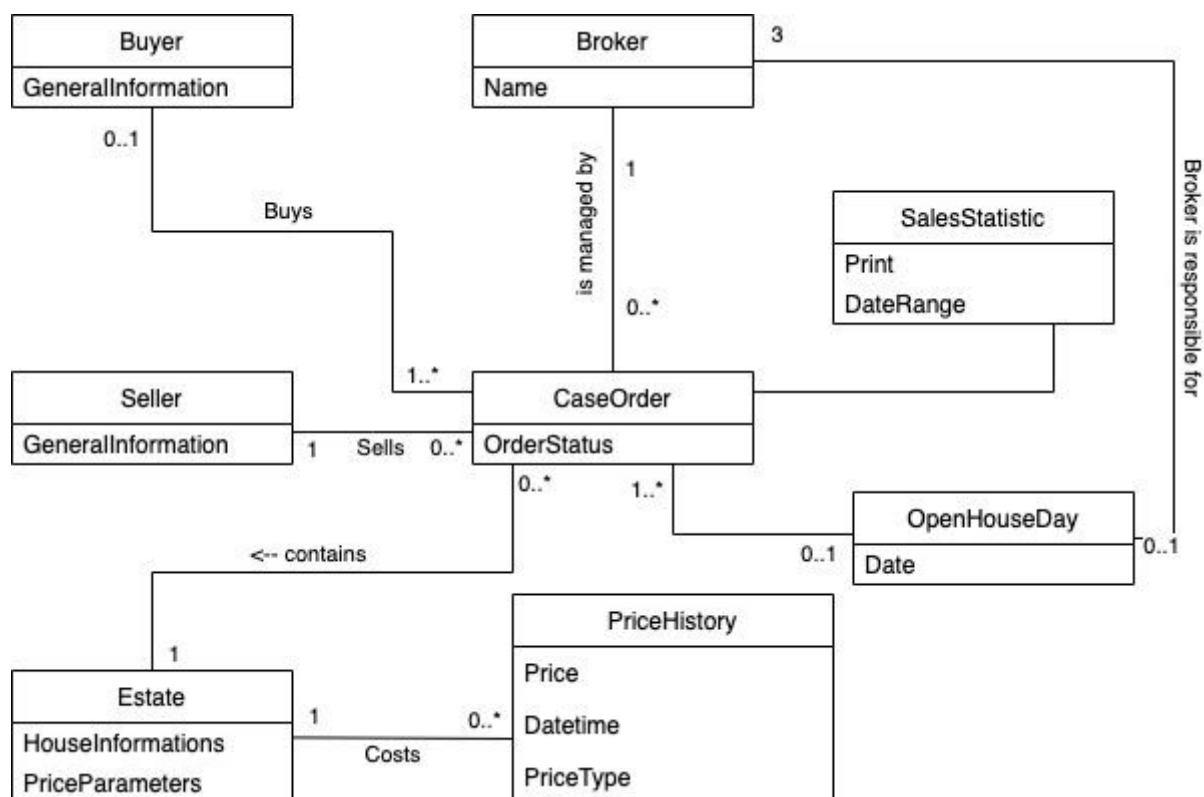
- Programmet designes med fokus på afkobling og med mulighed for etablering til eksterne systemer. Samtidig vil vi sikre udskiftelighed af de enkelte elementer i programmet. Herunder også UI. Programmet bliver udviklet på engelsk for at kunne

understøtte både fremtidige udviklere uden for Danmark og hvis virksomheden vil ekspandere til udlandet.

Plus

- Systemet udvikles således at der ved implementering ikke stilles store krav til hardware. Vi vurderer at nuværende udstyr hos Estate Brokers Ltd vil kunne bruges.
- Der vil være kobling til eksterne systemer og dette designes med mulighed for udskiftning og afkobling.

Domain diagram



Use Cases

UC1 Administrere kunde

Actor: Mægler

Preconditions:

Database med fungerende tabeller, hvor det er muligt at hente information eller der kan tilføjes nye informationer til.

Main success scenario:

1. Mægler tilgår administrer kunde funktionen i menuen.
2. Mægler vælger hvilken mulighed der ønskes.
 - Mægler har mulighed for at oprette en ny kunde.
 - Mægler har mulighed for at ændre i eksisterende kunde.
 - Mægler har mulighed for at slette en kunde.
3. Mægler udføre handling af valgte mulighed
4. Der bekræftes en oprettelse, ændring eller sletning på skærmen

Post Condition:

Mægler har nu enten fået oprettet en kunde, ændret i en allerede eksisterende kunde, eller fået slettet en kunde fra systemet.

UC2 Administrere Hus

Actor: Mægler

Preconditions:

Database med data omkring husene

Main success scenario:

1. Mægler vælger administrer huse funktionen.
2. Mægler tilgår ønsket mulighed i menuen.
 - Mægler har mulighed for at oprette et nyt hus.
 - Mægler har mulighed for at ændre i et oprettet hus.
 - Mægler har mulighed for at slette et hus.
3. Mægler udføre handling af valgte mulighed.
4. Der bekræftes en oprettelse, ændring eller sletning på skærmen

Post Condition:

Mægler har tilgået systemet hvor der enten er blevet oprettet et nyt hus, ændret i et allerede eksisterende hus, eller slettet et hus som allerede er oprettet.

UC3 Administrere salgssager (CRUD)

Actor: Mægler

Pre-conditions:

Databaseadgang med informationer omkring kunde og hus

Main success scenario:

1. Mægler åbner programmet
2. Mægler tilgår salgssags menu
3. Mægler tilgår ønsket mulighed i menuen
 - Mægler har mulighed for at oprette en ny sag.
 - Mægler har mulighed for at ændre i en oprettede sag.
 - Mægler har mulighed for at slette en sag
4. Mægler udføre handling af valgte mulighed
5. Mægler afslutter programmet

Post condition:

Der er nu blevet oprettede en ny sag, ændret i en allerede eksisterende sag, eller slettet en eksisterende sag i programmet.

UC4 Beregne udbudspris

Actor: Mægler

Pre-conditions:

Database med historik over tidligere salg.

Main Success Scenario:

1. Mægler indtaster data over ejendom som der ønskes prisberegning på
2. Prisberegning henter data fra database filtreret på en række faktorer såsom postnummer, alder, stand etc.
3. Mægler får en beregnet udbudspris

Post Condition:

Beregnet udbudspris som er vejledende for valgte bolig

UC5 Arrangere åbent hus

Primary Actor: Ledelse

Secondary Actor: Mægler

Pre-Conditions:

Database med data til at uddele huse og en algoritme der kan tildele husene efter prisleje.

Main success scenario:

1. Ledelse/Mægler vælger antal mæglere
2. Ledelse/Mægler vælger antal huse
3. Der bliver uddelt huse på tur efter højest pris til hver mægler
4. Husene samt oplysninger omkring omfordeling pr. mægler vises på skærmen

Post Condition:

Der bliver vist en lige fordeling i forhold til prisleje mellem alle mæglere på skærmen, med relevante oplysninger omkring fordelingen.

UC6 Udskrive salgsstatistikker

Actor: Mægler

Preconditions:

Database har tidligere gemte salg i system

Main success scenario

1. Mægler åbner salgsstatistikker i system
2. mægler kan gennemgå salgsstatistikker
 - sælger kan finde pris pr kvm for tidligere huse i given måned
3. sælger printer data i txt fil

Post Condition:

Sælger kan finde data og printe pris pr kvm i text fill.

UC7 Ændre salgsstatus

Actor: Mægler

Preconditions:

At en given sag er oprettet i systemet og mulig at tilgå.

Main success scenario:

1. Mægler vælger ændre salgsstatus funktionen i menuen.
2. Mægler vælger ønsket hus der skal ændres status på.
3. Mægler vælger ønsket status for valgte hus.
 - Mægler har mulighed for at sætte status til aftale oplæg.
 - Mægler har mulighed for at sætte status som aktiv på markedet.
 - Mægler har mulighed for at sætte status til ikke aktiv
 - Mægler har mulighed for at sætte status som solgt.
4. Valgte status gemmes i databasen.
5. Mægler afslutter og lukker program.

Post Condition:

Mægler har valgt den sag i systemet som der skal ændres status på, og sat det ønskede status på den valgte sag.

UC8 Tilpasse salgspris

Actor: Mægler

Preconditions:

En sag er oprettet i systemet og ligger på databasen med informationer om sagens pris.

Main success scenario:

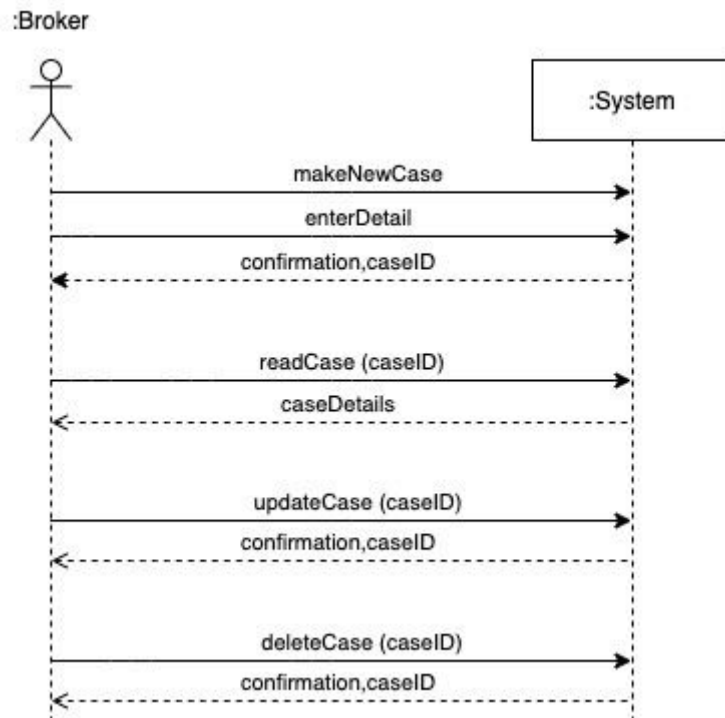
1. Mægler tilgår funktionen i systemet med hvor muligheden for tilpasning af salgspris er.
2. Mægler vælger den sag der ønskes tilpasset.
3. Sagen med den tilpassede pris ændres og gemmes på databasen.
4. Mægler afslutter og lukker programmet.

Post Condition:

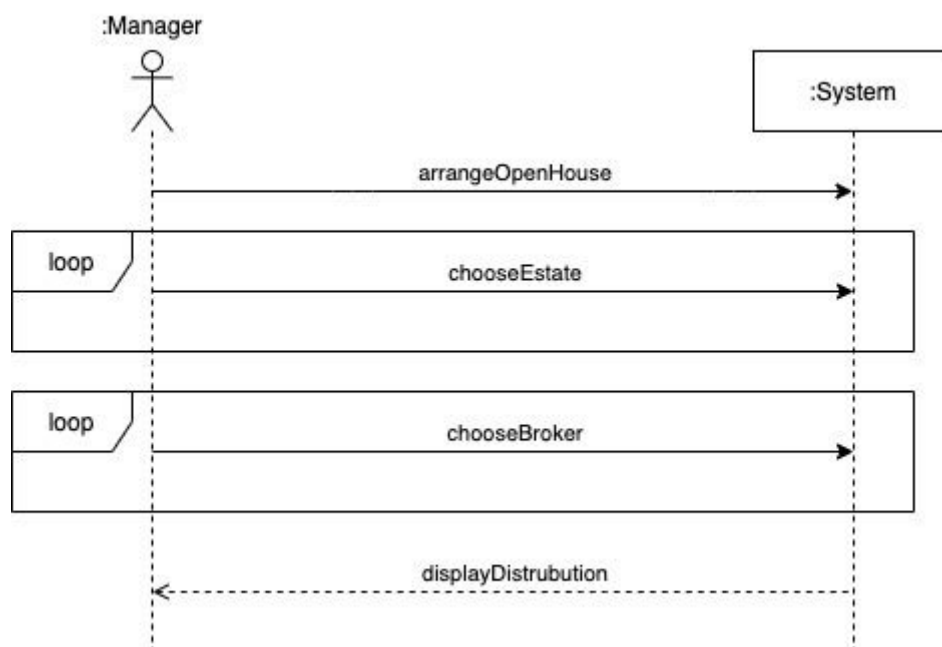
Mægleren har nu tilpasset prisen på en given sag, og ændringerne er nu gemt på databasen.

System Sequence Diagram

UC3 - Administrere salgssager



UC5 - Arrangere åbent hus





Database

Forfatter: Jesper og Marc

Normalform

I forhold til vores database er vi gået efter en database på 3. normalform. Da vi føler at dette er muligt i forhold til kravene i opgaven. Vi vurderer ikke at det vil give mening at holde den på 2. normalform i forhold til hastighed, da vores database ikke er stor nok til at det vil have betydning.

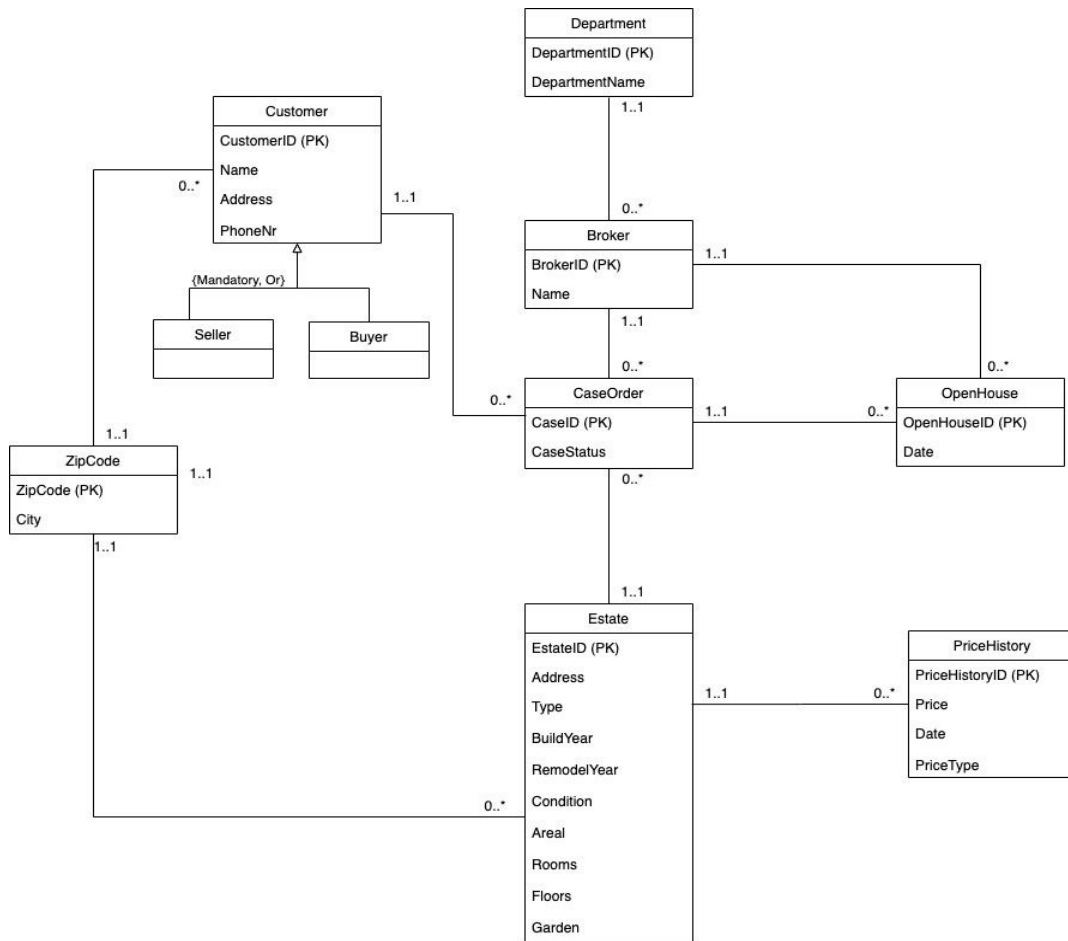
Transaktionstype

Vores database er default med transaktionstype Read Committed og det er per default det som Entity Framework benytter sig af. Vi har derfor ikke set nogen grund til at ændre dette.

I vores overvejelser har vi vægtet risiko for at flere retter i samme data på samme tid (meget usandsynligt) og at der er ikke som udgangspunkt slettes data.

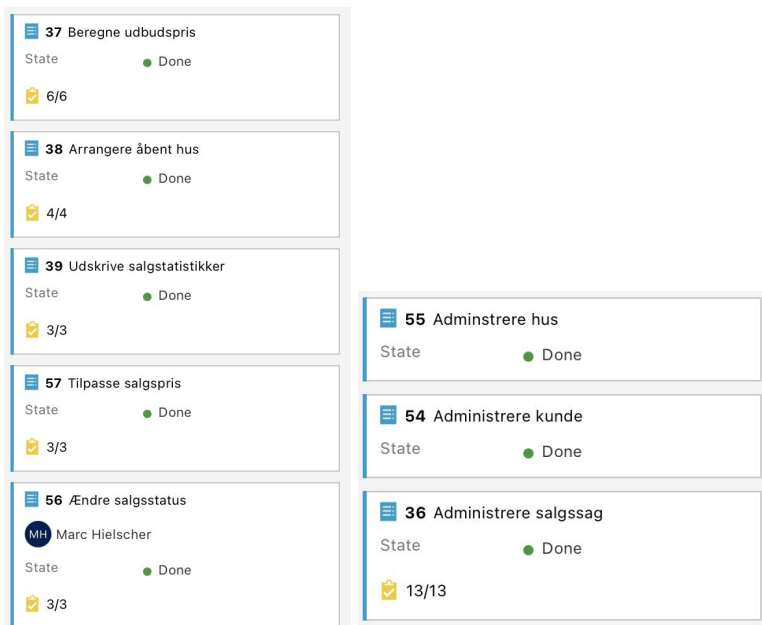
ER

Vi har forud for første sprint udarbejdet et foreløbig ER diagram. Tilrettelser vil blive dokumenteret via bilag og henvises til i de enkelte sprints.



Product Backlog

Oversigt fra Azure DevOps over vores product Backlog



Sprint 1

Kravs- afklaring / specifikation

Det er med *product owner* afstemt at systemet ikke vil have et salgsmodule, men at dette vil kunne tilføjes senere, hvis det ønskes.

Det er vurderet at Use Case 1,2 og 3 er højeste prioritet for firmaet og nødvendig for at kunne vise funktionaliteter, da disse ligger fundamentet for hele systemets funktionalitet. Disse use cases indebærer oprette, ændre og slette kunder, boliger og sager, i henholdsvis hver af deres kategorier.

Det er bestemt at update delen af vores Use Cases adskilles og placeres i egne administrations-menuer. Dette med tanke på at det vil kunne låses, således at det kun vil kunne ændres af en evt. superbruger (forskell i brugerniveauer er ikke implementeret). Disse administrations-menuer vil have en høj kobling til vores database på trods af vores ønske om lav kobling og høj udskiftelighed. Vi har noteret dette som et punkt for fremtidig forbedringer i programmet.

Design

forfatter: Jesper

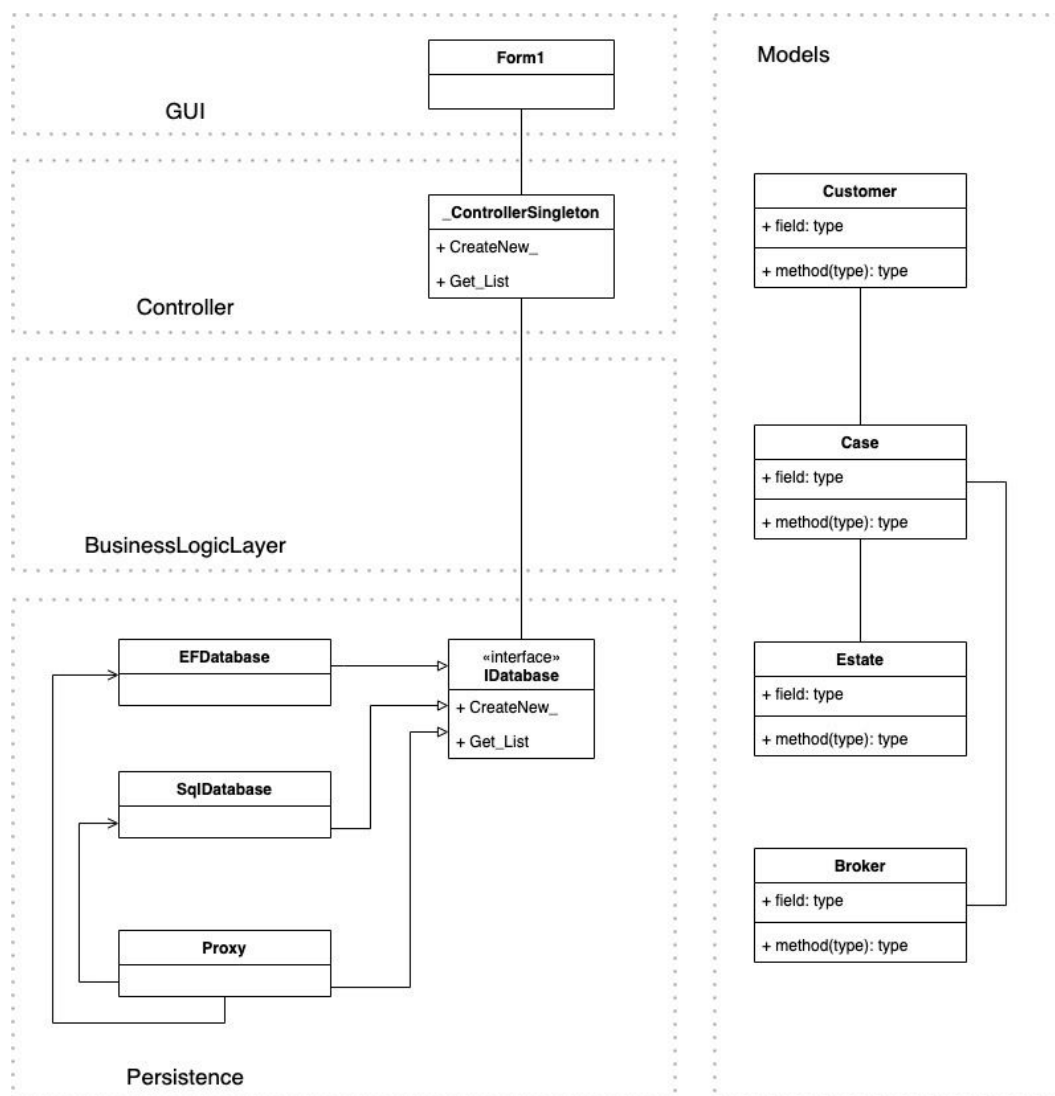
I vores opbygning af multilayer model var vores udgangspunkt et meget stringent anvendelse af lagdeling³ (se oprindelig DCD i bilag 1), hvilket viste sig at give rigtig mange metoder i vores *logic layer*, uden anden funktion end at kalde metoder i vores *Persistence layer*. Med andre ord ville vi få flere klasser og metoder i *logic layer* som ikke tilførte værdi/indeholdte logik.

Derfor har vi for vores CRUD use cases valgt at lade vores *Controller-Layer* have en direkte reference til vores *Persistence layer*. Vi er klar over betydning i forhold til afkobling, men vurderer fordelene større end ulemperne.

Overordnet design Class diagram for Use Case 1,2 og 3. Bemærk samtlige lag har reference til *Models*.

For oversigt over det endelige diagram se bilag 10.

³ Bilag 1: Oprindelig Design Klassediagram for UC 1,2 og 3




Der benyttes Singleton til *Controller-layer* fordi vi til enhver tid altid kun vil have en instans af vores controller. Der implementeres en controller for hver af vores use cases, således vi holder os til et overskuelig antal metoder i hver controller.

I vores *Persistence layer* benytter vi et *interface pattern* til at give os mulighed for på sigt at udskifte måden vi tilgår vores database. Det kan også være relevant hvis vi på sigt vil benytte anden teknologi end MsSql. På nuværende tidspunkt har vi to klasser til at tilgå vores MsSql database og valg af tilgang styres via et *Proxy pattern*, som primært vil forsøge på forbindelse via *Entity Framework Core* klassen.

Måden vi anvender *Proxy pattern* på er måske lidt "fordi vi kan" og vi er helt bevidst om, at det vil give mere mening hvis vores *catch* benyttede en anden persistence type end til database. F.eks. gemme data lokalt.

Entity Framework Core



Vores tilgang er *Database First* og vores *Models layer* er opbygget via reverse engineering med *scaffold-DbContext* kommandoen. Vi er fuld ud opmærksomme på, at vores properties i *models* er public på både *get* og *set*. Dette er af hensyn til funktionaliteterne der følger med entity framework, hvilket blandet andet er binding til og mulighed for at *update* via et DataGridView.

GUI

Vi har skitseret et design som gerne skulle lede tankerne hen på et klassisk arkiveringssystem⁴. (Se bilag 2). Derudover så er hjertet i vores GUI *CaseOrder* siden, hvor man både kan finde eksisterende og oprette nye sager.

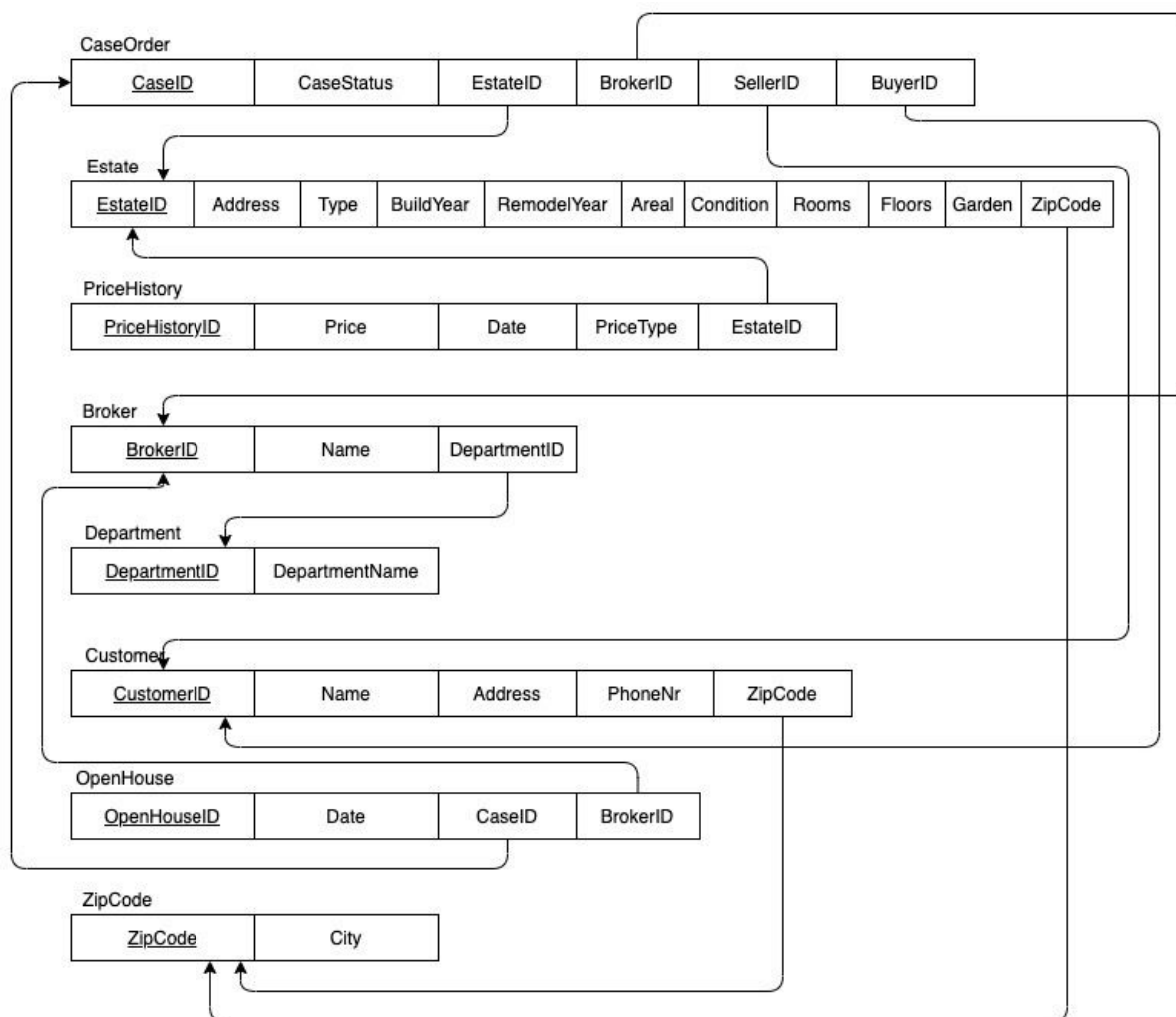
Mapning af ER diagram

forfatter: Jesper

Vi har valgt at mappe vores specialisering i ER diagrammet via metode 3 fordi *buyer/seller* ikke har nogen individuelle attributter og en *Seller* i mange tilfælde også er en *Buyer* (samtidig omvendt). Risiko ved kun at lave en tabel for *buyer/seller* vil være mange *null* attributter, hvilket ikke bliver et problem i vores tilfælde. Samtidig har det også været en del af vores refleksion, at vi gerne vil undgå redundans i vores database.

Det ses dog i vores *CaseOrder* at der ses forskelligt på *seller/buyer* og begge er med en foreign key reference til *Customer*.

⁴ Bilag 2: GUI - første udkast til brugerflade



Programmering

forfatter: Andreas

Der er i dette sprint blevet arbejdet på vores gui hvor der er blevet arbejdet på 3 funktionaliteter i vores program. I dette sprint er der i teamet blevet arbejdet på vores 3 administrere funktioner, UC1, UC2 og UC3. Der er blevet lavet kode så det er muligt for vores product owner at kunne bruge de 3 forskellige administrere funktioner, så der kan ændres direkte i kunder, huse og sager. Herunder ses et udkast af hvordan alle huse hentes ned med de informationer der er ønske, til datagridviewet.

```

1 reference | Dasse4, 12 days ago | 2 authors, 7 changes | 1 work item
private void bt_GetAllEstates_Click(object sender, EventArgs e)
{
    dataGridView_AllEstates.DataSource = EstateControllerSingleton.Instance().GetGridEstateData("").Estates.Local.ToBindingList();

    dataGridView_AllEstates.Columns["EstateId"].HeaderText = "EjendomsId";
    dataGridView_AllEstates.Columns["Address"].HeaderText = "Adresse";
    dataGridView_AllEstates.Columns["BuildYear"].HeaderText = "HusetsByggelsesår";
    dataGridView_AllEstates.Columns["RemodelYear"].HeaderText = "RenoveringsÅr";
    dataGridView_AllEstates.Columns["Condition"].HeaderText = "Stand";
    dataGridView_AllEstates.Columns["Areal"].HeaderText = "Areal";
    dataGridView_AllEstates.Columns["Rooms"].HeaderText = "Rum";
    dataGridView_AllEstates.Columns["Floors"].HeaderText = "Etager";
    dataGridView_AllEstates.Columns["Garden"].HeaderText = "Have";
    dataGridView_AllEstates.Columns["ZipCode"].HeaderText = "Postnummer";
    dataGridView_AllEstates.Columns["TypeId"].HeaderText = "TypeId";
    dataGridView_AllEstates.Columns["Type"].HeaderText = "Type";
    dataGridView_AllEstates.Columns["ZipCodeNavigation"].HeaderText = "PostnummerNavigation";
    dataGridView_AllEstates.Columns["CaseOrders"].HeaderText = "SagsOndre";
    dataGridView_AllEstates.Columns["PriceHistories"].HeaderText = "PrisHistorik";

    this.dataGridView_AllEstates.Columns["ZipCodeNavigation"].Visible = false;
    this.dataGridView_AllEstates.Columns["Type"].Visible = false;
    this.dataGridView_AllEstates.Columns["CaseOrders"].Visible = false;
    this.dataGridView_AllEstates.Columns["PriceHistories"].Visible = false;
}

```

```

IDatabase db;
2 references | Jesper la Cour, 13 days ago | 1 author, 1 change
public EstateBrokersContext GetGridEstateData(string name)
{
    db = new DatabaseProxy();
    return db.GetGridEstateData(name);
}

```

Test

Forfatter: Marc

Vi har benyttet Unit tests til at teste både vores Sql og Entity Framework CRUD metoder for at sikre at de virker som de skal, og de returnere de ønskede data fra databasen. Dette billede viser vores fremgang for nogle af de tests vi har lavet for sql metoderne:

```

public void GetEstateByEstateIdExpectVillaType()
{
    //Arrange
    IDatabase testDb = new SqlDatabase();
    //Act
    string actual = testDb.GetEstate(1).Type;
    string expected = "Villa";
    //Assert
    Assert.AreEqual(expected, actual);
}

[TestMethod]
0 references | Marc3411, 18 minutes ago | 1 author, 1 change
public void GetAllEstates()
{
    //arrange
    IDatabase testDB = new SqlDatabase();

    //Act
    int actual = testDB.GetAllEstates(address: "").Count;
    int expected = 2;

    //Assert
    Assert.AreEqual(expected, actual);
}

```

Deploy

Forfatter: Andreas

Der er i dette sprint blevet udarbejdet og deployet de første dele af vores program som er vores administrere funktioner. Vores product owner har ønsket muligheden for at kunne ændre direkte i de huse, kunder og sager der nu er oprettet i systemet. Dette er der mulighed for nu, og der er derfor 3 funktionelle use cases. Se udkast af de 3 sider nedenunder.

Form1

Salgsstatus

Salgsstatistik

Abent Hus

Odbudspris Beregner

Administrere Mægler

Administrere Hus

Administrere Kunde

Administrere Sag

Hent Alle

Gem

EjendomsId	Adresse	HusetsByggeset	RenoveringsAr	Stand	Areal	Rum	Etager	Have	Postnummer	Typeld
1	Grenningen 4	1965	2019	8	148	4	1	True	7100	1
2	Ryttervej 32	1987		5	178	6	1	False	8000	2
3	Borgervænget 14	2018		10	136	3	1	True	7100	1
4	Kobbevangenget ...	1992	2009	3	165	4	2	True	7100	1
5	Rung Alle 197 ...	1934	2018	10	425	12	1	False	900	3
6	Birke Alle 12 B	1967	2008	9	198	3	2	True	7100	1
7	Smaragdvvej 7, ...	2018	2018	10	107	3	1	False	7100	4
8	Skem Vælborps ...	1973	2019	8	140	4	2	True	7100	1
9	Lærketofte 3	2014	2015	9	177	3	1	True	7100	1
10	Flegborg 9 A, 4...	1939	2012	7	72	3	1	False	7100	4
11	Skytthugsgade ...	1890	2016	4	95	3	1	False	7100	4
12	Søndermarkvej...	1887	2019	10	126	3	2	True	7100	1
13	Mylius Erichsen...	1973	2002	5	172	5	1	True	7100	1
14	Sindballevej 2, 1	1909	2005	5	151	7	1	False	7100	4
15	Ribe Landevej 3...	1844	1994	4	245	4	2	True	7100	1
16	Krogagervej 37	1963	2000	6	115	3	1	True	7100	1
17	Haraldsvænget ...	2017	2017	7	185	5	1	True	7100	1
18	Rådhuset 4 ...	1930	2019	9	224	6	1	False	7100	4
19	Sternevej 17, 1...	1969	2018	1	91	4	1	False	7100	4
20	Østengårdvej 31	2010	2010	10	225	4	2	True	7100	1
21	Grejsdalvej 32...	1877	2002	4	115	3	2	False	7100	4
22	Vestparken 24	2002	2002	6	168	4	2	True	7100	1
23	Svendsgade 135	1951	2019	9	80	3	2	True	7100	1

Form1

Salgsstatus Salgsstatistik Abent Hus Odbudspris Beregner Administrere Mægler Administrere Hus Administrere Kunde Administrere Sag

Hent alle		Gem					
Kundenummer	Navn	Adresse	Telefonnummer	Postnummer			
1	Marc Madsen	Vesterbrogade	31606036	7100			
2	Lars Seier	Øksenej 2	12345678	900			
3	Eric Dembela	Janusvej 1	54655465	6700			
4	Ole Larsen	Svinget 4	62554433	7100			
5	Frank Onyeka	Patricius Alle 4	67628222	8000			
6	Bent Madsen	Byttervej 32	76762345	8000			
7	Rune Schmidt	Vestergade 72	34674232	8700			
9	Peter Nielsen	Smaragdvvej 7, 3	13669911	7100			
10	Kirsten Jensen	Birke Alle 12 B	63134114	7100			
11	Michael Hansen	Skem Vælborps Vej 54	63561511	7100			
12	Hanne Pedersen	Lærketofte 3	67215511	7100			
13	Thomas Andersen	Flegborg 9 A, 4. th	70967519	7100			
14	Anna Christensen	Skytthugsgade 3, 1. th.	69569000	7100			
15	Søren Larsen	Søndermarkvej 43	69076158	7100			
16	Lene Sørensen	Mylius Erichsenvej 4	49510040	7100			
17	Jan Rasmussen	Sindballevej 2, 1.	22518871	7100			
18	Marianne Jørgensen	Ribe Landevej 373	24995288	7100			
19	Niels Petersen	Krogagervej 37	91666673	7100			
20	Camilla	Haraldsvænget 9	90527896	7100			
21	Morten Kristensen	Rådhuset 4, 3. tv.	36804914	7100			
22	Louise Olse	Sternevej 17, 1. tv.	39703130	7100			
23	Charlotte Thomsen	Østengårdvej 31	67127574	7100			
24	Hans Christensen	Grejsdalvej 326 D	15739821	7100			
25	Tina Rasmussen	Vestparken 24	82535478	7100			
26	Per Johansen	Svendsgade 135	95130073	7100			
27	Inge Mettler	Fredericiagade 40A, 3.	48108708	7100			
28	Ole Mortensen	Munkvej 8	47346827	7100			
29	Karen Rasmussen	Aagade 28 B	82129952	7100			
30	Andreas Koefoed-Kron	Gammelengen 4	43763421	7100			

Form1

Salgsstatus Salgsstatistik Abent Hus Odbudspris Beregner Administrere Mægler Administrere Hus Administrere Kunde Administrere Sag

Sagsnummer: 3

Salger information:	Køber information:	Mægler information:
Kundenummer: 2	Kundenummer: 4	MæglerId: 1
Navn: Lars Seier	Navn: Ole Larsen	Navn: Lars Larsen
Adresse: Øksenej 2	Adresse: Svinget 4	AfdelingsId: 1
Tlf. nr.: 12345678	Tlf. nr.: 62554433	Afdeling: EstateBrokers Vejle
Postnummer: 900	Postnummer: 7100	
By: København C	By: Vejle	Abenthuss dato: label30

Rus information:

EjendomsId	Adresse	HusetsByg	Renoverings	Stand	Areal	Rum	Etager	Have	Postnumme	Typeld
2	Ryttervej 32	1987		5	178	6	1	False	8000	2

Pris historik:

PrisHistorik	Pris	Dato	EjendomsId	PriceType	PrisType
1	1449000	20-11-2020	2		
2	1500000	20-11-2020	2		

Sprint 2

Kravs- afklaring / specifikation

Forfatter: Andreas

Der er i sprint 2 blevet aftalt med product owner at vi fokusere på Use case 4, 7 og 8. Efter aftale med product owner er Use case 8 blevet integreret i programmet som en dialogboks.

Det ville give mulighed for at tilgå funktionen fra 2 sider, både Caseorder siden, samt sagsstatus ændres til solgt.

Efter krav fra product owner er det nu muligt i vores Use Case 4 "Beregn udbudspris" at beregne en udbudspris ud fra nogle ønskede faktorer.

- hustype
- postnummer
- Kvm
- stand
- renoverings år
- antal badeværelser

Design

Forfatter: Jesper

ER

Vi har af hensyn til ønskede funktionalitet opdateret og lavet tilrettelser af vores ER diagram⁵⁶. Disse tilrettelser indebærer en tabel over *HouseTypes*, *PriceTypes* og *CaseStatus*. Vi har valgt at implementere dette som tabeller i vores database i stedet for at hardcoded dem i systemet. Ved at gøre det på denne måde tillader det at der senere hen vil kunne udvides med flere mulighed under hver enkelt type/status,

GUI

Skitse for *UC7 Ændre Sagsstatus* er vedlagt som Bilag 5.

Skitse for *UC4 Beregne udbudspris* er vedlagt som Bilag 6

⁵ Bilag 3: ER diagram sprint 2

⁶ Bilag 4: Mapning af ER sprint 2

Vi har talt meget om hvordan vi skulle integrere *UC8 Tilpasse salgspris* i systemet og de første tanker gik på at implementere det som sin egen side (på samme måde som UC7). Men vi er endt med en løsning som vi mener er mere brugervenlig og holder systemet så enkelt som muligt. UC8 integreres som en dialogboks, der blandt andet kan tilgås fra CaseOrder siden og automatisk kommer frem når sagsstatus ændres til solgt.

Programmering

Vi har i dette sprint arbejdet med at finde en løsning på at kunne sende data/informationer mellem vores Forms (user controls). Dette specielt når vi gerne ville bruge *søge-forms* til at vælge hus, kunde eller sag. Løsningen blev at instantiere vores *søge-forms* og kalde den frem ved at bruge ShowDialog() metode. Ved afslutning af *søge-forms* blev der sendt en *dialogResult.OK* retur og vi kunne opdatere aktuelle tekstboks med valgte værdier. Se gerne nedenstående kode eksempel.

```
private void btn_SearchCase_Click(object sender, EventArgs e)
{
    SagsSøgningsForm ssf = new SagsSøgningsForm();
    if (ssf.ShowDialog() == DialogResult.OK)
    {
        txt_caseOrderID.Text = caseId;
        UpdateAdminCase();
        UpdateBrokerInfo();
    }
}

private void btn_choose_Click(object sender, EventArgs e)
{
    if (dataGridView_caseOrder.SelectedRows.Count > 0)
    {
        UC3AdministrereSag.caseId = dataGridView_caseOrder.SelectedCells[0].Value.ToString();
        UC7Salgsstatus.CaseId = dataGridView_caseOrder.SelectedCells[0].Value.ToString();
        DialogResult = DialogResult.OK;
        this.Close();
    }
    else
    {
        MessageBox.Show("Ingen sag valgt");
    }
}
```

Test

[TestMethod]

0 references | 0 changes | 0 authors, 0 changes

public void GetEstatesToListingCalculation()

{

//arrange

IDatabase testDB = new EFDatabase();

//Act

int actual = testDB.GetEstatesBasedOn_ZipCodeAndHousetype(7100, new HouseType(1).TypeId, 2000).Count()

int expected = 1;

//Assert

Assert.AreEqual(expected, actual);

}

Deploy

I dette sprint har vi udviklet og deployet vores Salgsstatus og Udbudsprisberegner, så de er fuldt funktionelle og dermed har vi nogle funktionelle use cases der virker til vores produkt owner. Se nedenstående skærbillede for et eksempel på et af dem:

Beregn udbudspris

Husoplysninger

HusType: Villa, Postnummer: 7100, Evm: 156, Stand: 1, Renoverings år: 2005, Antal badeværelser: 2

Udbudspris i kr. 1993052

Opret ny sag

Huse i området

Ejendomslid	Adresse	HusetsByggeselsk	RenoveringsÅr	Stand	Areal	Ru
4	Kolbervænget ...	1992	2005	3	165	4
6	Birke Allé 12 B	1967	2008	9	198	3
13	Mylius Erichsens...	1973	2002	5	172	5
16	Krogagervej 37	1963	2000	6	115	3
20	Østengårdvej 31	2010	2010	10	225	4
22	Vestparken 24	2002	2002	6	168	4
25	Merkurvej 8	1959	2003	5	102	3

Prishistorik

PrisHistorikId	PrisHistorikId	PrisHistorikId	Ejendomslid	PrisTypeId	PrisType
13	2500000	27-11-2020	22	1	
14	2000000	27-11-2020	22	1	

Sprint 3

Kravs- afklaring / specifikation

forfatter: Lasse

I sprint 3 arbejder vi på at implementere vores Use Case 5 "Arrangere åbent hus". kravene til denne funktion i systemet dikterer, at der skal kunne fordele op til 18 huse på 3 ejendomsmæglere således, at de kan hver især lave åbent hus arrangementer. Fordelingen skal ske således at de hver får et af de dyreste 3 huse osv., efter tur, således at de har 6 huse hver med nogenlunde lige fordelt salgspriser. Relevante informationer omkring fordelingen vises også på skærmen.

Design

forfatter: Andreas

I starten af dette sprint er der blevet arbejdet på Use case 5 (Arrangere åbent hus) GUI og funktionalitet. Der er blevet snakket i teamet om hvordan vi ville lave vores gui ud fra de krav vores product owner har stillet os. Der er efterfølgende blevet lavet nogle ideer, hvor der så er fundet frem til en løsning som vi mener er den bedste til at opfylde disse krav. (Se bilag 7) Der er blevet sat 2 gridviews op med de informationer der skal bruges til at kunne oprette et åbent hus arrangement, samt mulighed for at tilføje de 3 mæglere der ønskes at skulle stå for dette. Der er også udviklet en ekstra Windows Form som åbner når man ønsker at se en allerede sket tildeling, eller den nye oprettede som man lige har lavet.

Skitse for *UC5 Arrangere åbent hus* er vedlagt som bilag 7

Programmering

forfatter: Andreas

Der blev i starten af dette sprint efter vi var færdig med gui til åbent hus arrangement, lavet en del kode som har fået vores 2 datagridviews til at snakke sammen, som gør det muligt at sætte mæglere på til de kommende åbent hus arrangementer. Derefter er der blevet tilføjet funktionalitet som gør at vi får de ønskede data med over i vores nye Windows Forms. Kode eksemplerne herunder er vores metode til at vælge et hus og føre det over i vores nye datagridview.

```

1 reference | Jesper la Cour, 5 days ago | 1 author, 2 changes | 2 work items
private void btn_ChooseHouse_Click(object sender, EventArgs e)
{
    if (dataGridView_AllCaseOrders.SelectedRows.Count != 0)
    {
        dataGridView_selectedCaseOrders.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
        dataGridView_selectedCaseOrders.DataSource = null;
        SelectedCaseOrders.Add(CaseOrderControllerSingleton.Instance().GetCaseOrder(Convert.ToInt32(dataGridView_AllCaseOrders.SelectedCells[0].Value)));
        dataGridView_selectedCaseOrders.DataSource = SelectedCaseOrders;

        ActiveCaseOrders.RemoveAt(dataGridView_AllCaseOrders.SelectedCells[0].RowIndex);
        dataGridView_AllCaseOrders.DataSource = null;
        dataGridView_AllCaseOrders.DataSource = ActiveCaseOrders;

        SetSelectedColumns();
    }
}

```

```

5 references | Jesper la Cour, 13 days ago | 1 author, 1 change
public CaseOrder GetCaseOrder(int caseOrderID)
{
    IDatabase db = new DatabaseProxy();
    return db.GetCaseOrder(caseOrderID);
}

```

Threads

forfatter: Jesper

I opgaven var der i tillæg et ønske om, at der blev benyttet flere tråde i programmet. Vi har valgt at implementere baggrunds tråde til opdatering af vores combo boxes. Vores combo boxes opdateres fra databasen, hvorfor det gav god mening at dette kunne ske i baggrunden og uden at *forsænke* resten af programmet.

```

1 reference | Jesper la Cour, 5 days ago | 3 authors, 3 changes
public UC7Salgsstatus()
{
    InitializeComponent();
    Thread thread = new Thread(new ThreadStart(UpdateDropdown_CaseStatus));
    thread.IsBackground = true;
    thread.Priority = ThreadPriority.Lowest;
    thread.Start();
}

```

Test

Forfatter: Marc

Vi har testet vores GetOpenHouses metode for at se at den returnere det korrekte antal open houses på den valgte dato se kode eksempel under

```
[TestMethod]
✓ | 0 references | 0 changes | 0 authors, 0 changes
public void GetOpenHousesByDate()
{
    //Arrange
    IDatabase testDB = new EFDatabase();
    DateTime expectedDay = new DateTime(year: 2020, month: 12, day: 17);

    //Act
    var actual:int = testDB.GetOpenHouses(expectedDay).Count();
    var expected = 4;

    //Assert
    Assert.AreEqual(expected, actual);
}
```

Deploy

forfatter: Andreas

Der er i dette sprint blevet udarbejdet og deployet programmets åbent hus funktion, som opfylder de krav der var stillet fra vores product owner, som gør at denne use case er fuldt ud funktionel. Der ville derfor være mulighed for at bruge denne funktion med fuld funktionalitet efter dette sprint. Der kan ses et program eksempel af hvordan denne side ser ud når man tilgår den.

The screenshot shows a web application titled "Arrangere åbenthus". At the top, there is a navigation bar with several menu items: "Salgsstatus", "Salgst Statistik", "Åbent Hus", "Odbudspris Beregner", "Administrere Mægler", "Administrere Hus", "Administrere Kunde", and "Administrere Sag". The main content area is divided into three sections. On the left, there is a "Se eksisterende" section with an "Adresse:" label and a "Søg" button. In the center, there is a "Valgte huse" section with a large grey placeholder box and two buttons: "Tilføj" and "Fjern". On the right, there is a "Meglere" section with a dropdown menu showing "Lars Larsen" and a "Vælg" button. Below this is another large grey placeholder box and a "Ryd liste" button. At the bottom right, there is an "Åbenhuss dato:" section with a date picker showing "1. december 2020" and a "Tilføj" button.

OpenhouseOverview

Overblik over åbenthus dage

Dato:

	OpenHouseld	Date	BrokerId	Caseld	Broker	Case
▶	23	11-02-2021	1	13	Lars Larsen	
	24	11-02-2021	3	18	Lotte Petersen	
	25	11-02-2021	1	14	Lars Larsen	
	26	11-02-2021	3	10	Lotte Petersen	
	27	11-02-2021	1	11	Lars Larsen	
	28	11-02-2021	3	17	Lotte Petersen	

Sprint 4

Kravs- afklaring / specifikation

forfatter: Jesper

I Use Case 6 "Udskrive salgsstatistikker" har vi et systemkrav som hedder, at vores system skal kunne printe relevante statistikker på en .txt fil. Disse statistikker dækker overordnet solgte ejendomme og hvor mange af disse der er solgt i diverse områder, til hvilken pris, dato på salg samt hvilken mægler der stod for salget.

Vi har i samråd med *product owner* besluttet de følgende funktioner i statistikken

- Mulighed for filtrering på fra og til dato.
- Mulighed for filtrering på postnummer. (udeladt postnummer viser alle solgte boliger i valgte periode)
- Direkte visning af salgsstatistik i programmet.
- Mulighed for at gemme som .txt fil.

Design

forfatter: Andreas og Jesper

Der er i dette sprint blevet arbejdet på Use Case 6 (Udskrive salgsstatistik) GUI og funktionalitet. Der er blevet snakket om hvordan vi ønsker at vores GUI skulle se ud og

hvilke funktioner vi skal have tilføjet til den. I de første overvejelser blev der snakket om, at vi ville bruge nogle datetimepickers og søgefunktioner via dropdown/combo boxes. Men efter der er blevet snakket i teamet og ændret på designet, er det endeligt blevet til datetimepickers, en tekst-søgefunktion samt et datagridview. Dette ville give et overblik over vores salgsstatistikker ud fra filtrering, og derefter giver det os muligheden for at udprinte det til et .txt fil.

Der er også blevet tilføjet en ny *ControllerSingleton*, ny klasse i vores *logic layer* og en klasse i vores *model layer*.

Vores *Logic layer* håndtere den logik som står for at bygge vores liste af solgte huse, således vi igen sikrer at GUI er udskiftelige. Vi genbruger tidligere anvendte metoder i vores *persistence layer*.

Vi har ligeledes valgt at placere vores *PrintSalesStatistic()* i vores *logic layer* selvom man kan argumentere for at den skulle være placeret i vores *persistence layer*. Til grund for vores beslutning gik overvejelserne hovedsagelig på principperne omkring *information expert* fra GRASP.

Programmering

forfatter: Andreas

Der er i dette sprint blevet arbejdet på hvordan vi kan hente nogle informationer fra vores gridview ved at søge på nogle parametre. Dette gør at vi kan gemme vores søgte informationer over på txt.doc via en streamwriter. Som der har været en ønske funktion i programmet fra vores product owner, se kode eksempel af print funktionen under.

```
1 reference | Jesper la Cour, 7 minutes ago | 1 author, 1 change | 1 work item
public void PrintSalesStatistic(string fileName)
{
    StreamWriter writer = new StreamWriter(fileName);

    SoldHouse house = new SoldHouse();
    foreach (PropertyInfo property1 in house.GetType().GetProperties())
    {
        writer.Write(property1.Name.ToString().PadRight(15).Substring(0, 15) + " ");
    }
    writer.WriteLine();
    foreach (object item in soldHouses)
    {
        foreach (PropertyInfo property in item.GetType().GetProperties())
        {
            writer.Write(property.GetValue(item, null).ToString().PadRight(15).Substring(0, 15) + " ");
        }
        writer.WriteLine();
    }
    writer.Close();
}
```


Test

Forfatter: Marc

Vi har i dette sprint testet vores GetSoldHouses metode som bruges i vores SalesStatistics for at være sikker på at den returnere det korrekte antal huse der står som solgte, inden for en fra og til dato og et postnummer. Se kode eksempel under

```
[TestMethod]
0 references | 0 changes | 0 authors, 0 changes
public void GetSoldHousesByZipCodeAndDates()
{
    //Arrange
    SalesStatisticControllerSingleton
    testScs = SalesStatisticControllerSingleton.Instance();
    DateTime expectedDateFrom = new DateTime(year: 2020, month: 11, day: 12);
    DateTime expectedDateTo = new DateTime(year: 2020, month: 12, day: 7);

    //Act
    var actual:int = testScs.GetSoldHouses(zipCode: 7100, expectedDateFrom, expectedDateTo).Count();
    var expected = 4;

    //Assert
    Assert.AreEqual(expected: actual, actual: expected);
}
```

Deploy

forfatter: Andreas

Der er i dette sprint blevet udarbejdet og deployet vores salgsstatistik funktion, så vi har en færdig use case, som vores product owner ville have mulighed for at bruge efter dette sprint. Der ville være et udkast af hvordan denne use cases side ville se ud her under. Det første udkast ville være uden information i postnummer, og i andet eksempel ville der være udfyldt information hvor postnummer er udfyldt.

Samtidig har vi implementeret ToolTip i Udbuds Prisberegner.

Beregn udbudspris

Husoplysninger

HusType	Postnummer	Kvm	Stand	Renoverings år	Antal badeværelser	
Villa	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Beregn

Indtast postnummer for huset der skal beregnes udbudspris på

Udbudspris i kr.

Udbudspris

Form1

Salgsstatus Salgsstatistik Åbent Hus Dødbudspris Beregner Administrere Møbler Administrere Hus Administrere Runde Administrere Sag

Salgsstatistik oversigt

Fra: 10. november 2020 Til: 7. december 2020 Postnummer:

Print salgsstatistik Vis Salgsstatistik

	CaseOrderId	Estateld	Address	ZipCode	SquareMeters	SalesPrice	SquareMeterPrice	MyProperty	SalesDate	BrokerName
▶	2	1	Grønningen 4	7100	148	2020000	13649	0	24-11-2020	Lars Larsen
	3	2	Ryttervej 32	8000	178	15000000	84270	0	20-11-2020	Lars Larsen
	15	1	Grønningen 4	7100	148	2020000	13649	0	24-11-2020	Lotte Petersen
	16	11	Skyttehusgade 3.	7100	95	1799000	18937	0	27-11-2020	Lars Larsen
	18	14	Sindballevej 2, 1.	7100	151	2799000	18536	0	01-12-2020	Peter Jensen

Form1

Salgsstatus Salgsstatistik Åbent Hus Dødbudspris Beregner Administrere Møbler Administrere Hus Administrere Runde Administrere Sag

Salgsstatistik oversigt

Fra: 1. november 2020 Til: 7. december 2020 Postnummer: 7100

Print salgsstatistik Vis Salgsstatistik

	CaseOrderId	Estateld	Address	ZipCode	SquareMeters	SalesPrice	SquareMeterPrice	MyProperty	SalesDate	BrokerName
▶	2	1	Grønningen 4	7100	148	2020000	13649	0	24-11-2020	Lars Larsen
	15	1	Grønningen 4	7100	148	2020000	13649	0	24-11-2020	Lotte Petersen
	16	11	Skyttehusgade 3.	7100	95	1799000	18937	0	27-11-2020	Lars Larsen
	18	14	Sindballevej 2, 1.	7100	151	2799000	18536	0	01-12-2020	Peter Jensen

Kvalitetssikring

forfatter: jesper

Vi har i alle vores sprints arbejdet med at teste udvalgte metoder via Unit Testing. Dette for at sikre mod logiske fejl, som kan være svære at spotte. Samt ville dette sikre vores product owner mod mulige fejl når de sidder i en kunde situation, og skal levere en ydelse.

Samtidig har vi holdt hinanden op på de aftaler vi fra start har sat i forhold til *clean code*. Herunder sigende metode navne, single responsibilities og vigtigst af alt - *the boy scout rule*.

Vi har i hvert sprint arbejdet med at tage vores nye viden med videre til næste sprint. Herunder blevet klogere på hvilket begrænsninger vi mødte, specielt med henblik på samspil med database. Erfaringer som gjorde at vi både i sprint 2 og 3 måtte tilpasse tabellerne i databasen og reintegrere entity framework i programmet. Al

I forhold til vores user experience (GUI) har vi tegnet og drøftet den bedste og mest brugervenlige løsning i hvert sprint. Samtidig har vi lagt hjælp ind via pop-up dialog labels når cursoren holdes stille (via ToolTip).

Konklusion

Forfatter: Marc


I vores problemformulering har vi beskrevet de ønskede krav til programmet såsom et informationssystem som kan håndtere de daglige arbejdsprocesser ved et hussalg. Vi kan konkludere at vores system lever op til de krav der stilles i forhold til opgaven. Der er udviklet funktionalitet til alle ønskede use cases. Vi har benyttet os af faglige relevante metoder, teknikker og principper vi har lært, samtidig med at vi har brugt en agile process.

Proces (SCRUM)

Forfatter: Marc

Vi startede med et *pre-sprint* for at få alle de analytiske ting på plads inden vi startede på sprint 1, for at få styr på alle opgaverne der ikke er en del af et sprint og de opgaver der ligger til grund for at vi kan udfører sprintende, såsom use cases. Derefter gik vi i gang med sprint 1, hvor vi greb det lidt forkert an til at starte med og kom ud i en form for vandfaldsmodel i stedet for en agile proces som SCRUM jo gerne skulle føre til. Dette fik vi dog styr på efter første sprint review, hvor vi derefter fik rettet det til at være en agile proces.

Herefter fik vi lavet flere use cases som kunne ligge til grund for vores forskellige sprints og begyndte derefter at gå i gang med sprint 1. Vi har haft 4 sprints i det hele. Hver af vores sprints havde en varighed på en uge, hvor vi mandag morgen mødes fysisk til vores sprint



planning, og så var vi vores deploy om fredagen hvor vi havde en færdig del af produktet som havde fuld funktionalitet, og kunne vises til product owner.

Vi har benyttet Azure DevOps til vores scrum, hvor vi har lagt de forskellige sprints op og under dem har vi de forskellige features(use cases) som indeholder tasks, så vi havde styr på hvilke der var to do, in progress og done. Vi har haft den samme scrum master under hele projektet, da det virkede naturligt for os.

Perspektivering

Forfatter: Jesper

Siden starten af projektet er der i teamet blevet arbejdet ud fra vores aftaler i gruppe kontrakten og vi har alle været ansvarlige for at holde os til det aftalte. Processen med at lave kontrakten har dog nok været mest interessant, da det tvang os til at drøfte emner, der ikke nødvendigvis var de nemmeste at tale om. Herunder *hvad hvis*-scenarier og forventninger.

Vi startede projektet ud med at en ide om at arbejde delvis online og kun være fysisk til stede ved sprint events. Ret tidligt i projektet valgte vi dog, at lægge kursen om og ændrede det til en langt større fysisk tilstedeværelse for at komme godt i gang. Men vi fandt faktisk ud af, at der var rigtig mange fordele ved at sidde sammen og noget af det som fungerede rigtig godt var muligheden for at sidde og par-programmere og hurtigere kunne gøre tanke til handling. Så den langt større fysiske tilstedeværelse har vi egentlig fortsat hele projektet igennem.

Noget af det som i projektet har vist sig svært for os har været projektets opdeling i tasks og estimering af opgavernes omfang, hvilket klart er noget som vil være nemmere des større erfaring vi får med programmering.

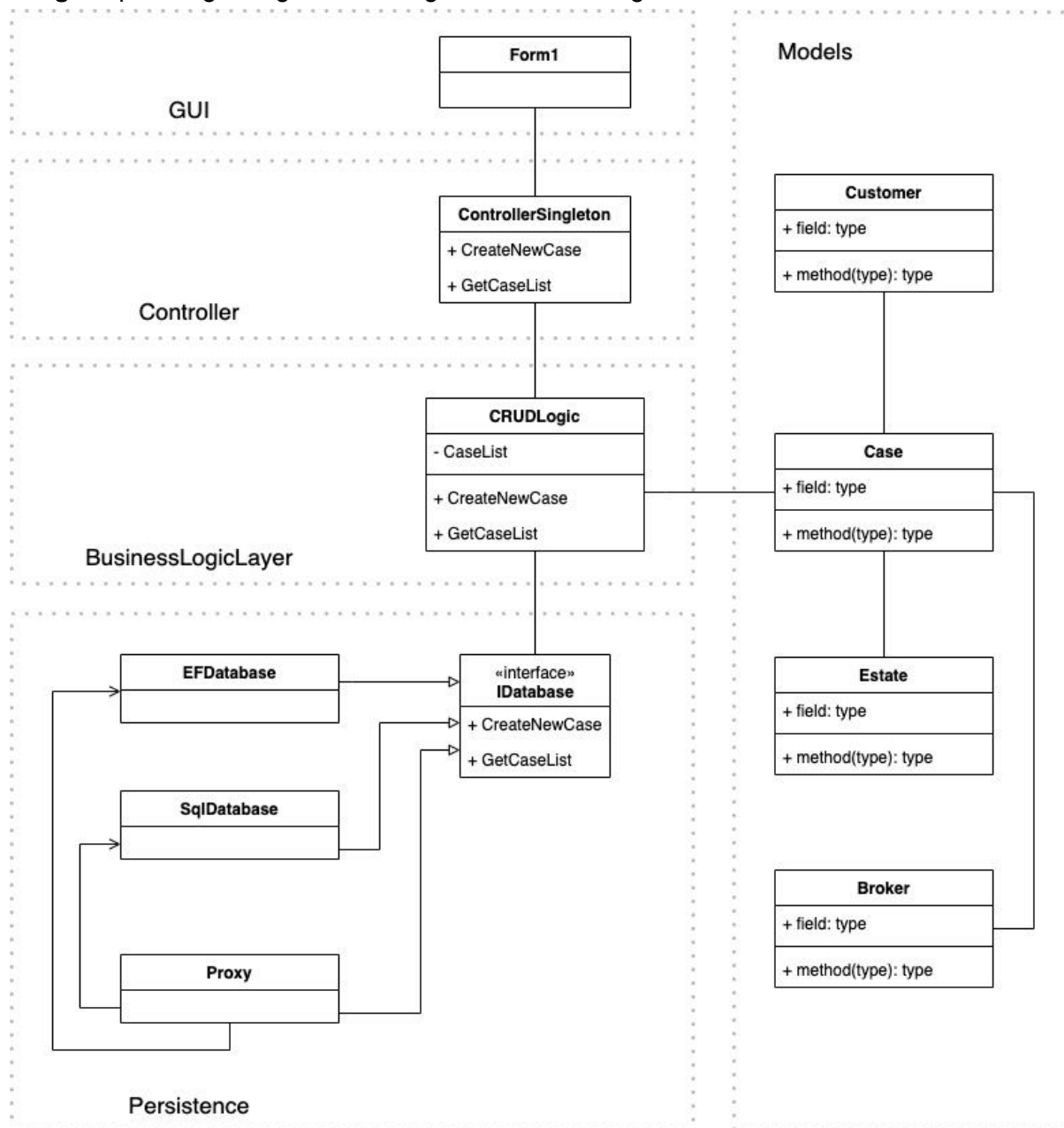
Så noget af det vigtigste vi vil tage med os er klart det positive ved at arbejde sammen med mulighed for at lære af hinanden. Derudover at SCRUM processen langt hen ad vejen har fungeret efter hensigten.

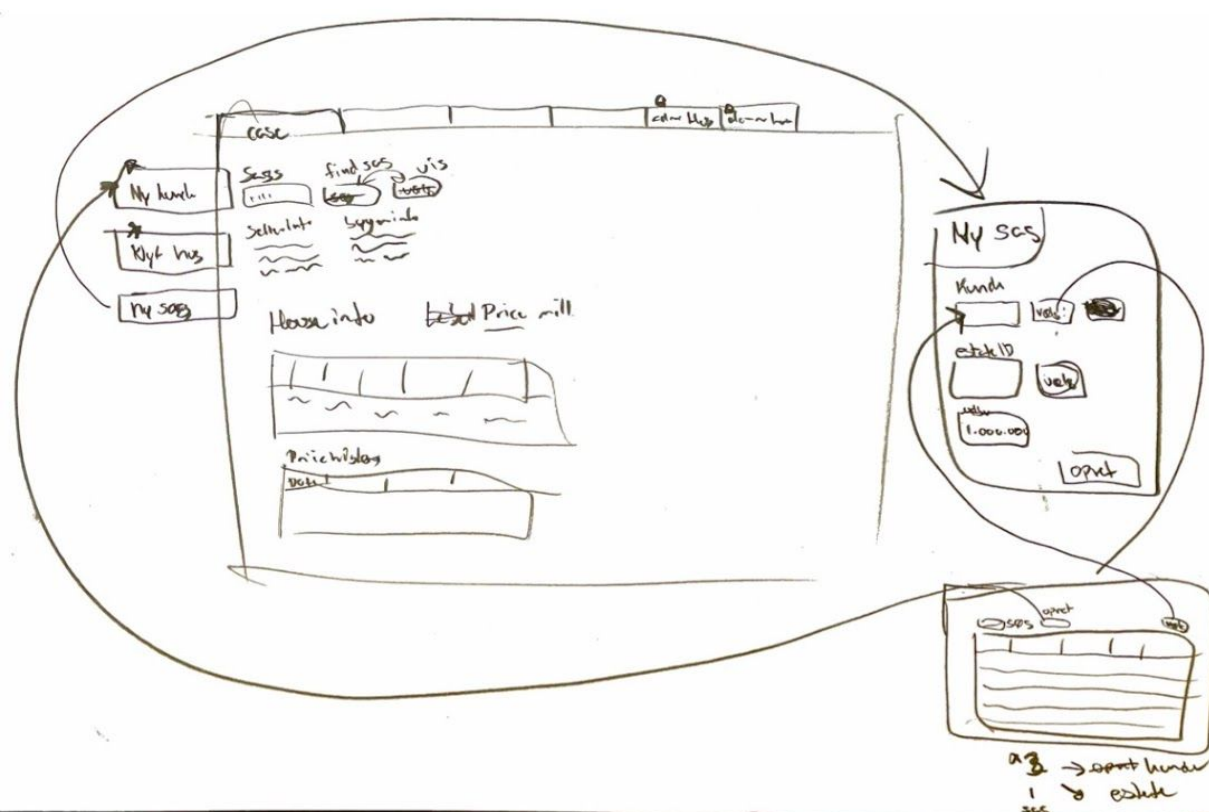
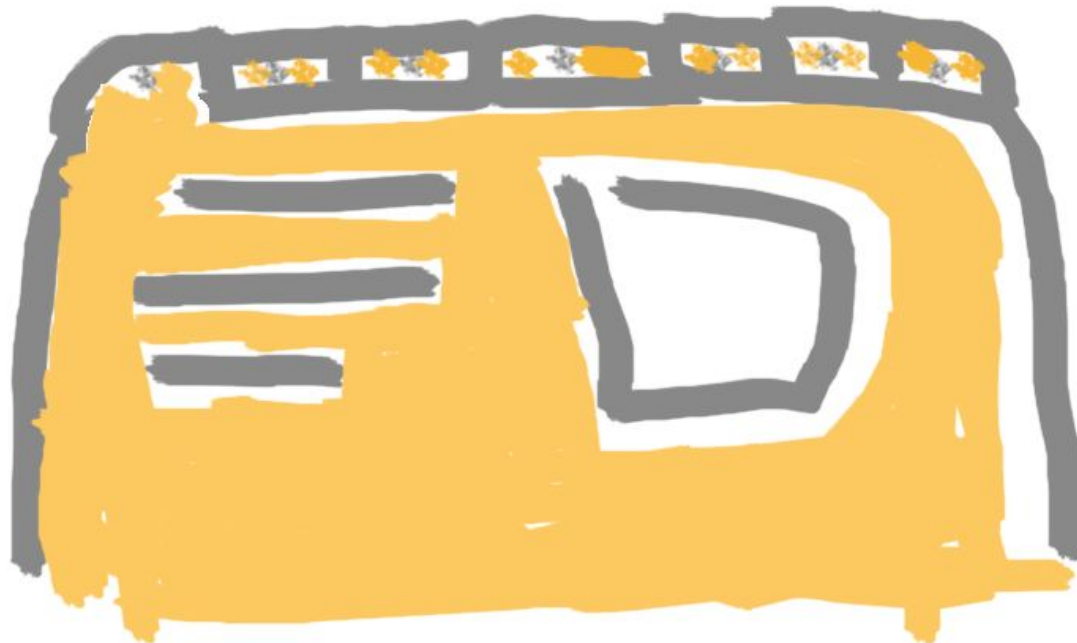
Litteraturliste

- Anna Hejlsberg, "*guide-til-brug-af-vaerdikaeden*", annahejlsberg.dk
<https://annahejlsberg.dk/guide-til-brug-af-vaerdikaeden/> 10/10/2020
- Craig Larman, "*Applying UML and Patterns*", 3. udgave.
- Robert C Martin, "*The Clean Coder*",
- Flyger, Hedegaard og Melander, "*Økonomisk Virksomhedsbeskrivelse*", 3. udgave

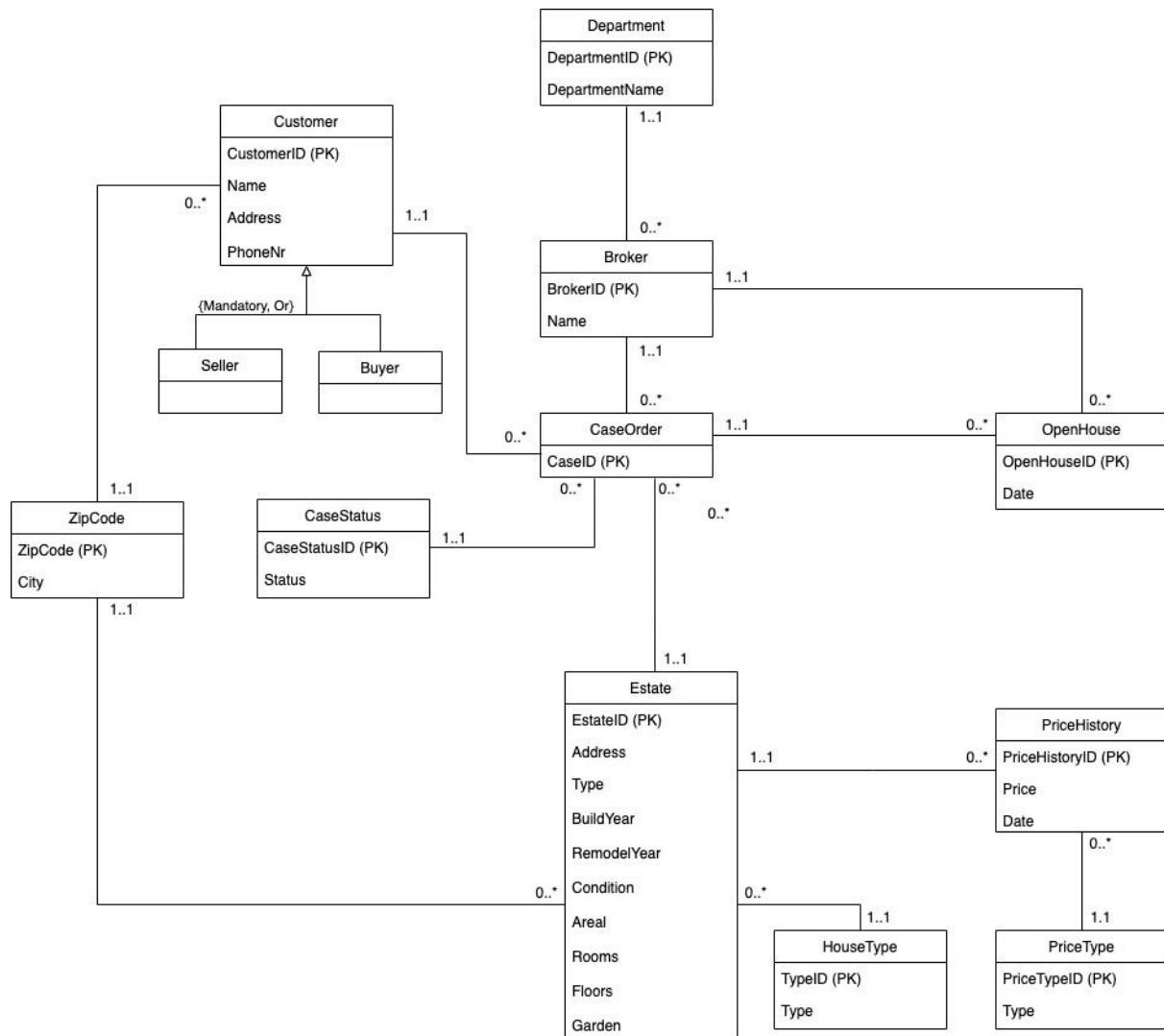
Bilag

Bilag 1 Oprindelig Design Klassediagram for UC 1,2 og 3

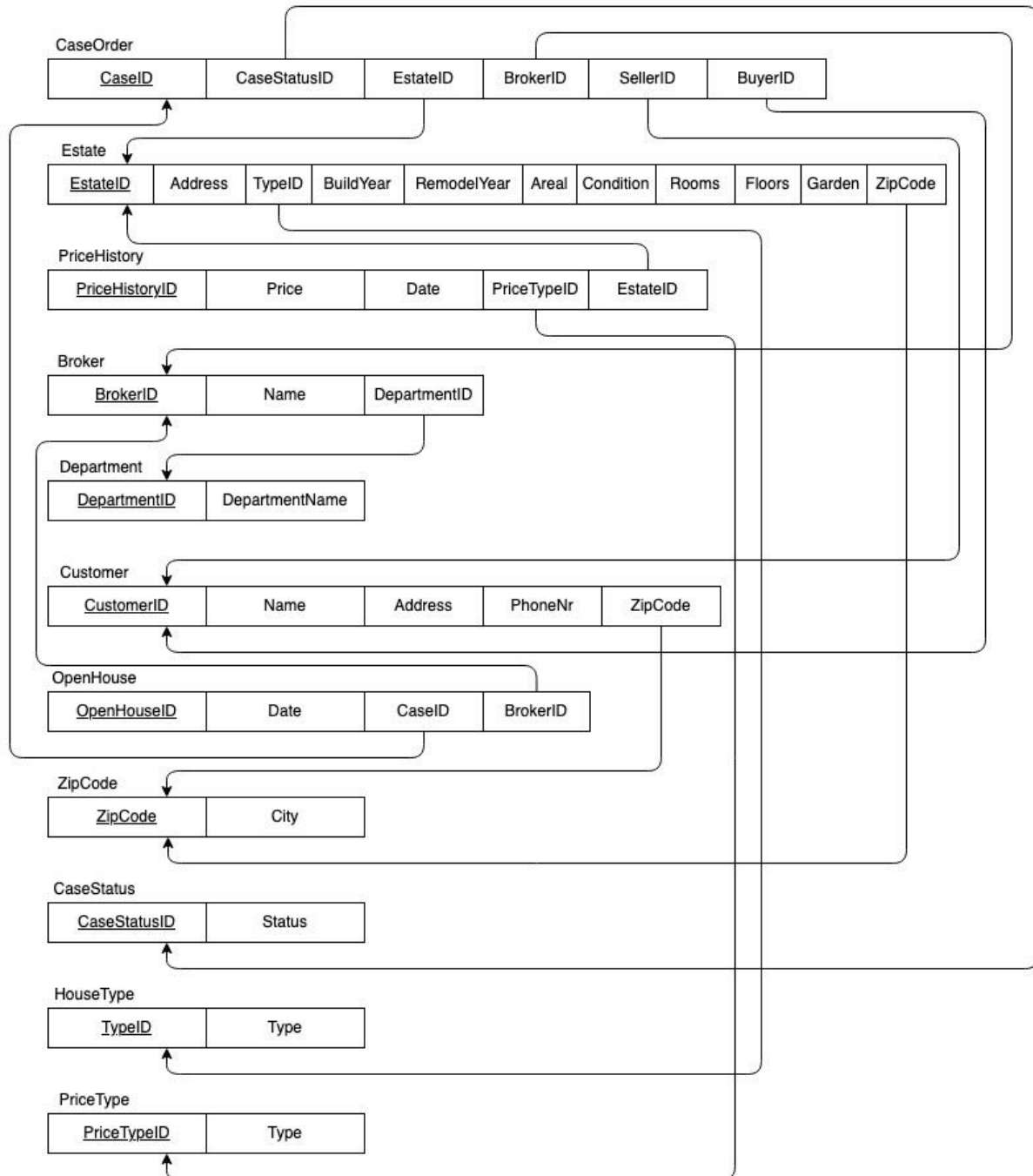


Bilag 2 GUI - første udkast til brugerflade

Bilag 3 ER diagram sprint 2



Bilag 4 Mapping af ER sprint 2



Bilag 5 UC7 Ændre Sagsstatus skitse.

Ændre sagsstatus

Sagshammer

VIS VIS alle

Sag

Vælg

Sags information:

Sags nr.

Status

Sender

Mægler id

Ejendom id

Køber id

Gem

Bilag 6 UC4 Beregne udbudspris

Beregn udbudspris

Hus oplysninger

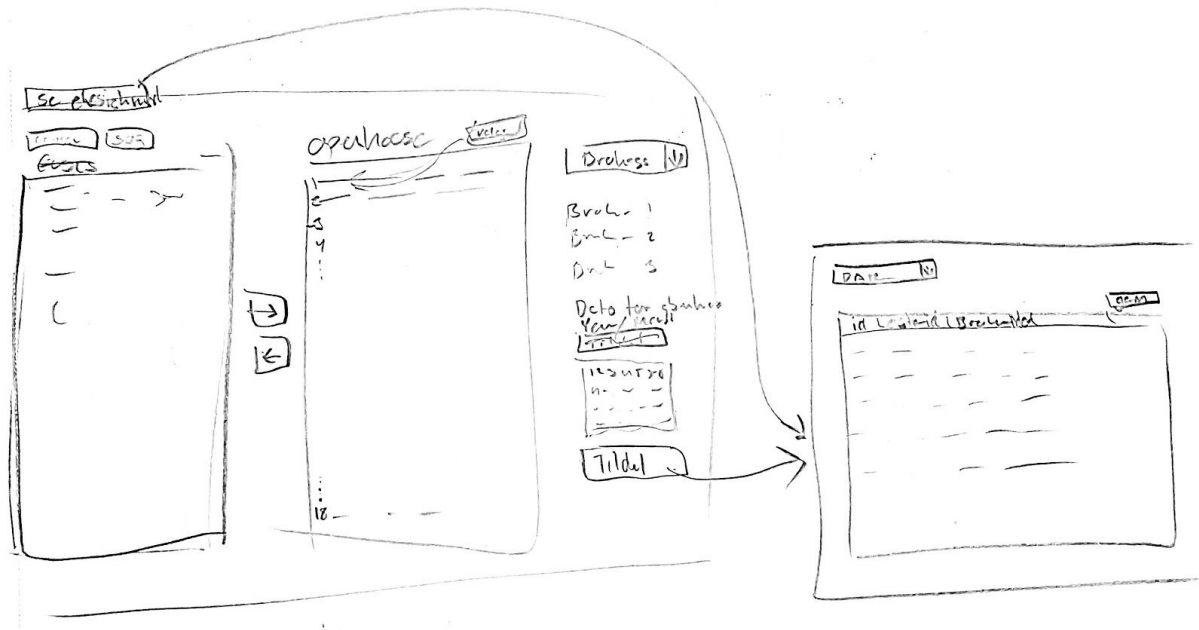
Hus type	Post nr.	Komm.	Stad.	Investerings d.	Renoverings år	Antal Bed.
			-10%	+10%		

P

opret hus

Sammenlignelse hus

Bilag 7 UC5 Arrangere åbent hus



Bilag 8 Uddrag af sql kode

```
create table Department
(
  DepartmentID int identity primary key,
  DepartmentName varchar(55)
)

create table Broker
(
  BrokerID int identity primary key,
  Name varchar(45),
  DepartmentID int foreign key references Department(DepartmentID)
)

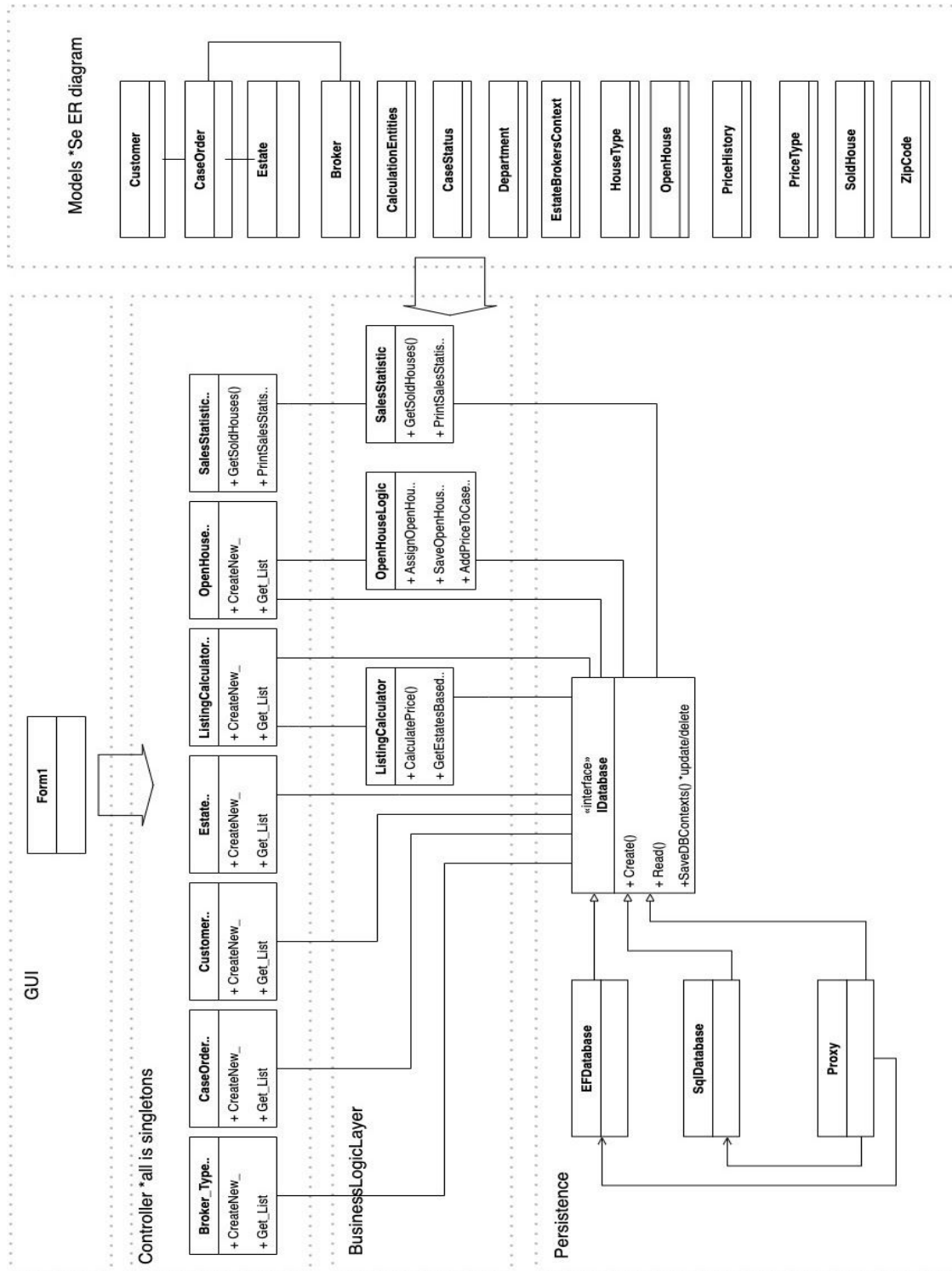
create table ZipCode
(
  ZipCode int primary key,
  City varchar(45)
)

create table Customer
(
  CustomerID int identity primary key,
  Name varchar(45),
  Address varchar(55),
  PhoneNr int,
  ZipCode int foreign key references ZipCode(ZipCode)
)
```

Bilag 9 Oversigt over Pull Request fra GitHub

- se separat PDF (Pull requests · EstateBrokers overview)

Bilag 10 Endeligt design class diagram (DCD)



Bilag 11 Endeligt ER diagram

