P2: Administration af skolebåde og bådudlån i en sejlklub.

- Programmering og modelløsning -

Projektrapport

SW2A305

Caspar Rosgaard Kuchartik

Marc Tom Thorgersen

Thomas Pilgaard Nielsen

Nikolaj Møller Smed

Søren Hvidberg Frandsen

Tristan Carl Benjamin Bendixen

Troels Beck Krøgh

AALBORG UNIVERSITET
STUDENTERRAPPORT

Første Studieår — Software
Strandvejen 12–14
9000 Aalborg
http://tnb.aau.dk

Titel:
Administration af skolebåde og bådudlån i en sejlklub.

Tema:
Programering og modelløsning.

Projektperiode:
P2 (Forårssemestret 2014)
03-02-2014 - 21-05-2014

Projektgruppe:
Gruppe SW2A305

Gruppemedlemmer:
Caspar Rosgaard Kuchartik
Marc Tom Thorgersen
Nikolaj Møller Smed
Søren Hvidberg Frandsen
Thomas Pilgaard Nielsen
Tristan Carl Benjamin Bendixen
Troels Beck Krøgh

Vejleder:
Jacob Nørbjerg

Oplagstal: 1

Rapport sideantal: 68

Appendiks sideantal: 25

Total sideantal: 99

Projekt klaret den:
20-05-2014

Ord/Tegn (Cirka): 18756 /98782

Abstract:

The following project studies the matter of creating a management system for a sailclub, with an integrated sailing school. The project first takes a wider look at the similarities of a sailclub, and other clubs used for freetime. An analysis is made of sailclubs and their needs in a management system, and from this analysis an application is designed and developed. The design part of the project adresses the problem of having persistant data, from run-time to run-time. At the end of the project it is concluded that it is possible to create a management system, which will be able to have all the functionalities required by the sailclub. Although this is true, more work needs to be done on the application created alongside the project, in order for the projectgroup, to stand by the application as a good solution to the problem.

Indholdsfortegnelse

Indledning 1

Denne rapport er den skriftlige del af et projekt, udført af en gruppe studerende på Aalborg Universitet, nærmere bestemt er det et P2 projekt på Softwareingeniørstudiet. Formålet med projektet er, ifølge projektoplægget, *"[...] at opnå færdigheder i problemorienteret projektarbejde i en gruppe samt viden om sammenhænge mellem problemdefinition, modeldannelsers rolle i forståelse og konstruktion af programmer, og programmer som løsning på et problem i en problemstilling kontekst."* [Kurt Nørmark 2014]. Helt konkret skal der produceres et program i C#, der skal virke som en hel eller delvis løsning på problemet, opstillet i det valgte emne.

Projektets emne er "Administration af skolebåde og bådudlån i en sejlklub". Dette er et meget specifikt emne, derfor vil muligheden for at udvide problemstillingen til andre sports- og fritidsklubber undersøges.

I moderne tider er computere og smartphones i højere grad blevet en integreret del af størstedelen af befolkningens hverdag. Det er ikke altid, at foreninger og fritidsklubber anvender disse nye værktøjer på en effektiv og optimal måde. Dette er det problem, som gruppen vil undersøge, samt forsøge at løse.

Rapporten består af tre dele, problemanalyse, problemløsning og refleksion. Problemanalysen har til formål at opnå et overblik over hvilke interessenter der er, deres organisation og den teknologi der allerede findes indenfor det valgte emne. Dette ender ud i en problemformulering, som leder projektet ind i løsningsdelen af rapporten. Derefter opsættes der programspecifikationer og programkrav, som vil være lavet på baggrund af analysen. Så bliver arbejdsprocessen beskrevet, og programmet dokumenteres og testes.

Dette bliver efterfulgt af diskussion, konklusion og perspektivering, i refleksionsafsnittet.

1.1 Initierende Problem

I følgende afsnit vil der konstrueres et initierende problem dannet ud fra projektoplægget givet ved semesterstart. Der vil desuden dannes nogle spørgsmål, der har til formål at hjælpe med undersøgelsen af dette problem.

Fritidsklubber benytter ofte frivillige i klubben til at hjælpe med administrativt arbejde. [Center for frivilligt socialt arbejde 2012] I en sejlklub er der meget data, som skal håndteres, i bl.a. lister over sejlture, evt. sejlerskole eller andre begivenheder, der relaterer sig til sejlklubben. Hvis dette ikke organiseres effektivt kan det medføre, at arbejdet bliver uoverskueligt, tager

længere tid end nødvendigt og risikoen for at miste information kan blive større.

Denne problemstilling er ikke unikt knyttet til sejlklubber, men er et generelt dilemma med administrativt arbejde i fritidsklubber, specielt klubber der bruger samme faciliteter til både udlejning og undervisning.

Rapportens analyse vil bygges ud fra følgende initierende problem:

> *Er det muligt at lave en softwareløsning, der kan gøre frivillige i fritidsklubbers administrative arbejde nemmere, som stadig er let at benytte uden at have meget erfaring med anvendelse af computersystemer, og i så fald hvordan?*

I analysen vil disse problemstillinger blive undersøgt:

- Hvilke fritidsklubber har administrative opgaver, såsom udlejning, skemalægning af begivenheder mm., der skal håndteres?
- Hvilken information skal diverse fritidsklubber håndtere?
- Er det muligt at organisere de frivilliges arbejde med en softwareløsning?
- Findes der andre løsninger på markedet til at løse problemet? I så fald, hvorfor benyttes de ikke?

Fritidsklubber 2

I følgende afsnit vil forskellige fritidsklubber blive undersøgt, for at afdække de respektive administrative poster, som kunne forekomme i sådanne klubber. Afsnittet er skrevet ud fra gruppens egne erfaringer, hvilket gøres, da det ikke har været muligt, at finde eksterne kilder omhandlende administrationen af de respektive typer af fritidsklubber. Det er taget i betragtning, at disse erfaringer kan være specifikke for den enkelte klub, hvor erfaringen stammer fra. Dog vurderes det, at de forekommende administrative opgaver er generelle opgaver, og altså ikke enkelttilfælde, og derfor godt kan anvendes i denne sammenhæng.

Begrebet fritidsklubber dækker over klubber, hvor voksne og unge kan bruge deres fritid, hvad end det er socialt, sportsligt eller begge. Fritidsklubberne er valgt ud fra det faktum, at de ofte er drevet af frivilligt arbejde og har faciliteter der udlejes. Denne type frivilligt arbejde er individuel fra klub til klub, hvad end det er undervisning, håndtering af det økonomiske eller noget helt tredje. Dog er der nogle administrative opgaver, som går igen blandt fritidsklubberne. Det er primært opgaver som omhandler administration af medlemmer, og deres gøremål i klubsammenhæng. De generelle administrative opgaver omkring medlemshåndtering kan være følgende:

- Kontingentbetaling
- Undervisning/træning
- Turneringsbetaling

2.1 Sportsklubber

I sportsklubber kan der forekomme mange forskellige administrative opgaver. Det er forskelligt fra sportsklub til sportsklub, hvilke opgaver der er tale om, men der er nogle fællestræk, som går igen blandt klubberne. Disse fællestræk er bl.a.:

- Udlejning/udlån af aktiver, f.eks. baner og sportsudstyr.
- Arrangering og pointtildeling ved sportsarrangementer.
- Arrangering af kørsel til og fra arrangementer.
- Kommunikation til forældre og sportsudøvere.

Sportsklubber kan også have specielle forhold, som har indflydelse på, hvilke administrative opgaver klubben har, grundet de forskellige sportsgrene de udbyder, f.eks. ved en skytteklub gælder der specielle regler for våbenlicenser og våbentransport.

2.2 Sportscentre

Sportscentre lægger lokale til mange forskellige sportsklubber. Fodboldklubber bruger omklædningen og de udendørs baner til træning. Badmintonklubber bruger de indendørs baner, som de skal dele med andre aktiviteter f.eks. håndbold, indendørs fodbold, hockey osv. Sportscentre har også ofte lokale arrangementer, f.eks. foredrag, ungdomsklub osv. Sportscentre kan også have et motionscenter, hvor idrætsklubber og almindelige personer kan købe adgang [Spøttrup Kulturhal 2014]. Eksempler på administrative opgaver kunne være:

- Administration af udlejning og omklædning.
- Information fra kiosken.
- Informationsdeling til sportsklubber og andre interesserede.

2.3 Ungdomsklub

En ungdomsklub ses som et sted, hvor unge kan bruge deres fritid på sociale aktiviteter. Det varierer, hvordan ungdomsklubber drives. Nogle drives af frivillige, hvor andre drives af skolevæsnet, f.eks. ungdomsskoler[Holstebro Ungdomsskole 2013]. Ofte holder ungdomsklubber til i skolebygninger. Hvad end de drives af frivillige eller skolevæsnet, har de stadig nogle administrative opgaver, som skal løses. Eksempler på sådanne administrative opgaver kunne være:

- Arrangementsplanlægning.
- Bestilling af varer (hvis der sælges sodavand, slik o.l.).

2.4 Sejlklub

En sejlklub kan have flere administrative opgaver. Disse opgaver kan bl.a. bestå af administration af klubbens både, herunder hvem der har reserveret en båd, hvornår båden er ledig osv. Nogle sejlklubber har også sejlundervisning, hvor der bl.a. skal kunne holdes styr på hvilke lektioner, eleverne har haft. Andre eksempler på administrative opgaver i en sejlklub kunne være:

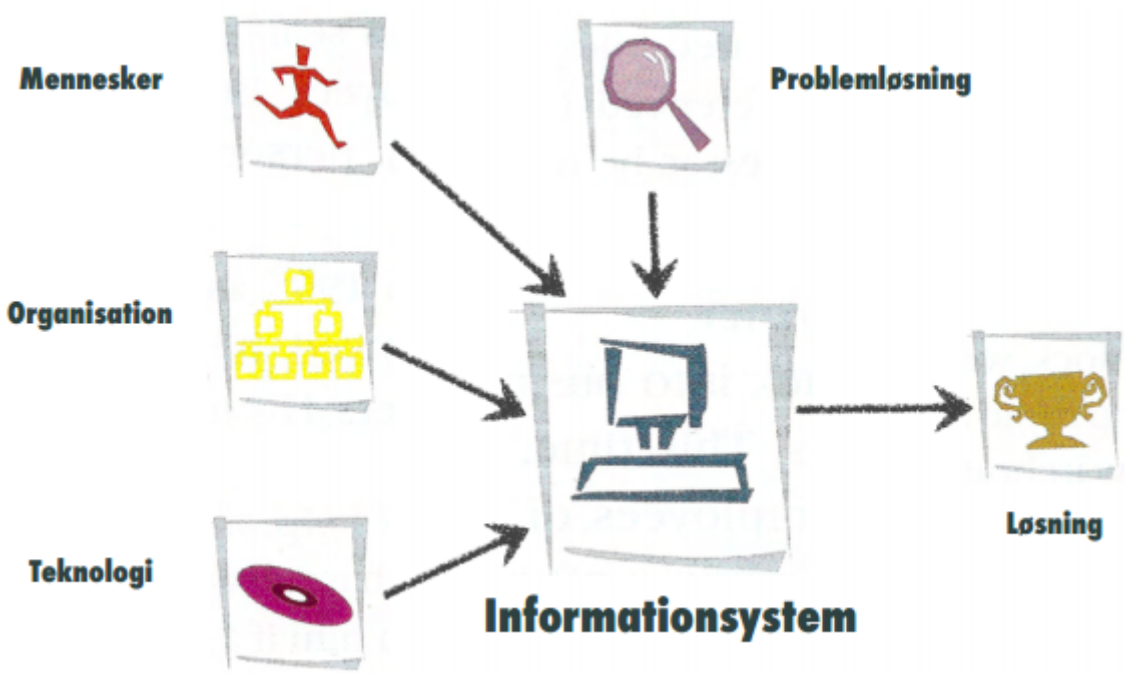
- Af- og tilmelding af undervisning.
- Reservering og udlån af både.
- Oversigt over hvem der skylder penge for lån af både.

2.5 Afgrænsning

Ud fra ovenstående fritidsklubber afgrænses der til fremover udelukkende at beskæftige sig med sejlklubber. Denne afgrænsning finder sted, da det vurderes, at der er potentiale for at gøre arbejdet i en sejlklub nemmere, grundet de administrative udfordringer. Ligeledes findes sejlklubber specielle, da nogle sejlklubber har en undervisningsfunktion, og derfor har specielle administrative funktioner.

Struktur af rapporten 3

I dette afsnit, beskrives strukturen af rapporten. Strukturen er baseret på beskrivelsen af et informationssystem fra Laudon and Laudon [1999]. Komponenterne i et informationssystem er illustreret i figur 3.1.



Figur 3.1. Illustration af elementerne i et informationssystem. Kilde: Laudon and Laudon [1999]

3.1 Informationssystem

Et informationssystem bruges til at effektivisere en arbejdsproces og hjælper med at holde fokus, så arbejdet bliver udført tilfredsstillende. For at udvikle et godt informationssystem, skal man indsamle viden om mennesker, organisation og teknologi. Når informationssystemet er udviklet, består det af tre processer: Indsamling af data, behandling af dataene og formidling af dataene. De tre elementer bruges til at undersøge, hvad der skal tages hensyn til under problemløsningsdelen, og vil blive beskrevet herefter.

3.1.1 Mennesker

Elementet *mennesker* omhandler de personer, som har en interesse i, at en given problemstilling løses. Det kan være brugeren af det program, der bliver lavet, samt andre, som får gavn af en løsning. Man undersøger bl.a. brugerens evner, da programmet skal laves på en sådan måde, at brugeren har den fornødne kunnen, til at betjene programmet. Brugerens behov undersøges også, så man får alle de funktioner med, som er nødvendige for et brugbart program.

3.1.1.1 Organisation

Elementet *organisation* undersøges hvor og hvordan problemerne opstår, derudover undersøges hvilke regler og værdier organisationen har, for at kunne tage højde for disse under problemløsningen.

3.1.2 Teknologi

Elementet *teknologi* omhandler de teknologier, som anvendes til at løse en informationssystemrelevant problemstilling. Derfor vil der blive undersøgt andre informationssystemer, der er rettet imod samme gruppe, som dette projekt.

3.1.3 Problemløsning

Ud fra de tre førnævnte elementer er der blevet indsamlet data, hvilket skal tages i betragtning i forhold til et informationssystem. Disse anvendes til at opstille krav, som problemløsningen skal tage hensyn til. Kravene vil blive brugt under resten af problemløsningen.

3.2 Rapportens opbygning

Først bliver menneskedelen behandlet, i form af en interessentanalyse, hvorefter vil organisationselementet blive berørt. Derefter bliver der skrevet om teknologier, hvorefter de tre elementer munder ud i en problemafgrænsning og en problemformulering. Informationen, indsamlet igennem problemanalysen såvel som problemformuleringen, benyttes til at opstille krav for problemløsningen, efterfulgt af en beskrivelse af informationssystemet. Rapporten afrundes med en diskussion af rapporten, samt en opsamling på informationssystemet i form af en konklusion. Desuden ses der på en eventuel videreudvikling og tilføjelser til informationssystemet såvel som andre potentielle anvendelser.

Del A

Problemanalyse

Interessenter for en sejlklub 4

I denne interessentanalyse ses der på de mennesker og grupper, der har interesse i en specifik softwareløsning, konkret til sejlklubber med sejlerskole. Der vil herudover blive set på, hvad deres interesse er i projektet, samt hvad de hver især kunne få gavn af i forhold til projektet. Det er værd at notere, at hver enkelt klub har forskellige behov, så dette er en generel forståelse af interessenternes behov. Forståelsen for de forskellige grupper af interessenter er dannet ud fra et interview med tidligere skolechef, Jacob Nørbjerg, fra Sejlklubben Sundet. En transskription af interviewet kan findes i appendiks C på side 81.

4.1 Interessenter for sejlklubbers administrative opgaver

4.1.1 Dansk Sejlunion

Dansk Sejlunion er et forbund, som blev dannet i 1913. Deres mål er at være det nationale samlingspunkt for alle sejlere. Dansk Sejlunion er tilsluttet Danmarks Idrætsforbund, International Sailing Federation og andre lignende organisationer inden for sejlsport [Dansk Sejlunion 2014]. Et af deres mål er, som forbund, at hjælpe sejlklubber med service, rådgivning mm. og derfor menes det, at Dansk Sejlunion også vil være interesseret i et system, der vil kunne hjælpe de frivilliges arbejde. De vil ikke komme til direkte at anvende løsningen, men vil stadig være relevante at tage med i projektets overvejelser.

4.1.2 Medlemmer

Hver sejlklub bestemmer selv deres organisatoriske opbygning af, hvilke type medlemmer de har i klubben. Dette vil blive undersøgt nærmere i kapitel 5 på side 11.

Det er vigtigt, at klubberne gør medlemmernes så tilfredse som muligt i deres brug af klubben, således de fortsat vil gøre brug af klubben. Et system der kunne hjælpe medlemmerne med at finde informationer, leje både og som har medlemmernes interesse i højsædet, vil være i deres interesse.

Medlemmer i en sejlklub består af et meget bredt befolkningssegment, da der normalvis accepteres medlemmer i mange aldre. Det er derfor vigtigt at lave et design, som er intuitivt i anvendelse for alle brugere.

4.1.3 Underviserne

En sejlklub med sejlerskole skal have undervisere til sejlerskolen, og disse kan også drage nytte af et system til at hjælpe med administrative opgaver. Underviserne vil ved hjælp af et system, eventuelt kunne sende afbud via e–mails eller SMS–beskeder til alle på undervisningsholdet. Systemet vil muligvis kunne gøre det lettere for underviseren at holde styr på den enkelte elevs fremskridt ved undervisningen mm., hvilket betyder, at underviserne bør have andre funktioner i et system til en sejlklub, end almindelige medlemmer ville have. Underviserne ville eventuelt kunne stå for at registrere, hvem der har et førerbevis i klubben o.l. Underviserne kan også være frivillige i sejlklubben, men de er beskrevet for sig selv, da de har mere specifikke opgaver i sejlklubben.

Det antages, at hvis et system ville kunne hjælpe undervisere med deres opgaver, så de bruger mere af deres arbejdstid på at undervise, ville det have en positiv indvirkning på undervisernes frivillige arbejde.

4.1.4 De frivillige i sejlklubben

Der er, udover undervisere i en sejlklub, også andre frivillige der arbejder i sejlklubben. Dette kan være lige fra en formand, til en sekretær eller måske endda nogen som gør rent i sejlklubben. Disse frivillige kan alle have forskellig gavn af et system, der kan håndtere deres anliggender i sejlklubben, og er derfor taget med som interessenter. Nogle af dem kan have til opgave at holde styr på ind- og udbetalinger for medlemmerne, hvilket et medlemssystem ville kunne hjælpe med. En sekretær ville kunne give nyheder eller referater videre til klubbens medlemmer igennem sådan et system. Der er mange forskellige muligheder et administrativt system ved en sejlklub ville kunne hjælpe med, og herudover også mange mennesker der ville kunne få gavn af en løsning.

4.1.5 Konklusion af interessentanalyse

Disse fire interessenter, har hver deres interesse for projektet, hvor underviserne og de andre frivillige smelter en smule sammen. Det antages, at underviserne, samt de andre frivillige, gerne vil have det gjort lettere at udføre deres frivillige arbejde. Medlemmerne vil gerne have informationer fra klubberne angående begivenheder, der måtte foregå, samt muligheder for at deltage i disse, eller leje sejlbåde. Dansk Sejlunion er en indirekte interessent, men kan dog have interesse i at hjælpe nye eller mindre sejlklubber med at opsætte et godt administrativt netværk, for at danne en velfungerende sejlklub, der giver gode oplevelser for medlemmerne. Men da dette ikke er et kommercielt projekt, vil der ikke tages kontakt til Dansk Sejlunion.

Organisation 5

I dette kapitel vil der blive set på sejlklubber som en organisation. Der vil undersøges, hvordan medlemmer kategoriseres, hvordan undervisningen organiseres, samt hvordan bådudlån og andre administrative opgaver håndteres. Undersøgelsen omhandler tre forskellige klubber: Sejlklubben Sundet og Bådklubben Valby, som begge er beliggende i København og Aalborg Sejlklub, som har til huse i Aalborg. Der vil primært blive fokuseret på Sejlklubben Sundet, da der er mest information til rådighed om dem. Dette i form af interviewet med Jacob Nørbjerg, som blev nævnt i kapitel 4 på side 9, samt en mere informativ hjemmeside, i forhold til de to andre klubber. En bredere forståelse, for den organisatoriske opbygning, kunne være opnået ved, at flere repræsentanter fra Sejlklubben Sundet blev interviewet, men dette har der ikke været mulighed for. Aalborg sejlklub og Bådklubben Valby er med i undersøgelsen for sammenligningens skyld, men har ikke været tilgængelige for interviews. Denne mangel på information om de to andre klubber, gør at betingelserne for en sammenligning ikke er optimale, men dette anses i denne sammenhæng for værende acceptabelt.

5.1 Medlemmer

Det er meget forskelligt fra sejlklub til sejlklub, hvordan medlemmer kategoriseres, hvis dette overhovedet bliver gjort. Sejlklubben Sundet kategoriserer deres medlemmer således: Voksen–, bådejer–, gaste–, mini-kølbåd–, ungdoms–, passiv– og støttemedlem. Desuden har de en særlig æresmedlemskategori. Denne status opnås ved at have ydet en mangeårig, ekstraordinær indsats for klubben og sejlsporten generelt.[Sejlklubben Sundet 2012]

Til sammenligning har Aalborg sejlklub kategoriseret deres medlemmer som følger: aktive, passive, junior– og æresmedlemmer [Aalborg Sejlklub 2014b].

Bådklubben Valby kategoriserer som A–, B– og C–medlemmer, som indikerer hvilken bådtype de har [Bådklubben Valby 2014a].

Det formodes, på baggrund af appendiks C på side 81, at fælles for medlemmerne er, at de gerne vil have informationer fra klubberne, så de ved hvilke begivenheder, der finder sted. Det kan være kapsejladser, foredrag o.l.

5.2 Sejlerskole

Sejlklubben Sundet og Aalborg Sejlklub har en sejlerskole, hvor man kan opnå et duelighedsbevis til båd. Sejlklubben Sundet kalder deres udgave af duelighedsbeviset et førerbevis og har yderligere krav. Bådklubben Valby adskiller sig fra de to andre, ved ikke at have en decideret sejlerskole, men de har kurser, man kan tage hen over vintersæsonen, for at opnå et duelighedsbevis [Bådklubben Valby 2014b].

I Sejlklubben Sundets sejlerskole er der undervisning i hverdagene fra kl. 18 til 21, i månederne maj, juni, august og september. Uddannelsen varer to år, og man lærer at sejle i bådtyperne Drabant og Gaffelrigger. I løbet af en sæson sejler man typisk 18 gange. Er man forhindret i at møde op, skal man ringe og melde afbud. Hvis man ikke møder op tre gange, uden at have meldt afbud, mister man sin plads i sejlerskolen. Undervisningsforløbet afsluttes med en praktisk prøve, hvor man opnår sit førerbevis, i så fald man består [Sejlklubben Sundet 2014a]. Aalborg sejlklubs undervisningsforløb minder om Sejlklubben Sundets [Aalborg Sejlklub 2014a].

5.3 Bådudlån og andre administrative opgaver

Man kan i Sejlklubben Sundet, leje både til sejlads, når de ikke er i brug til undervisning. Prisen for leje af bådene er højere i weekenderne end i hverdagene, men hvis en af skolens elever er med på sejladsen, er det gratis at leje en båd. For at kunne leje en båd, skal der være mindst én bådfører med. Hver gang der skal sejles ved Sejlklubben Sundet, er der meget data, der skal skrives ned, blandt andet i en logbog, af forskellige årsager. Det er vigtigt at vide, hvem der er med på båden tildels for at sikre, at besætningen er i stand til at føre båden, men også af sikkerhedsmæssige årsager, hvis der skulle ske en ulykke. En liste, over hvilken information der skal skrives ned, er udformet og kan ses i appendiks A på side 77. På Sejlklubben Sundets website ses det, at forskellige værktøjer, bl.a. en doodle, er taget i brug, i et forsøg på at øge brugervenlighed og interaktion mellem deltagere [Sejlklubben Sundet 2014b]. Som Jacob Nørbjerg fortalte i interviewet, skulle man før i tiden ned i klubhuset, for at se om der var en ledig båd, man kunne reservere, men dette kan nu gøres gennem et offentligt bookingsystem fra Supersaas [Supersaas.dk 2014]. Jacob Nørbjerg giver i interviewet også udtryk for, at det er svært at få viden omkring klubbens foretagender hjemmefra. Nuværende praksis er, at man skal ned i klubben for at se og melde sig på diverse begivenheder, såsom 24-timers sejladser, eller for at se om der er en ledig plads på en sejltur.

Håndtering af information omkring klubbens medlemmer foregår på en lokal computer i Sejlklubben Sundets klubhus. Endvidere bruges denne computer til at udprinte girokort til betaling af bådlån. Da informationerne er gemt lokalt, betyder det, at de frivillige, som er ansvarlige for udprintningen af girokort, skal ned i klubhuset for at udføre denne opgave.

5.4 Konklusion af organisationsanalyse

Sejlklubben Sundet har mange administrative opgaver, som håndteres manuelt og til dels på en ustruktureret måde. Som Jacob Nørbjerg giver udtryk for i interviewet, er det svært at holde sig opdateret med, hvad der sker i klubben. Ud fra dette kan der konkluderes at en softwareløsning, hvori der implementeres en kalender, som kan holde styr på alle

begivenhederne, vil være et godt supplement til administrationen af klubben. For at give eleverne bedre mulighed for at kunne af– og tilmelde sig, så de har mindre risiko for at miste deres plads i sejlerskolen, kunne der med fordel implementeres en sådan funktion i en softwareløsning. Det nuværende bookingsystem er åbent for offentligheden og er derfor redigerbart af alle. Dette er problematisk, derfor vil en forbedret udlånsfunktion med fordel kunne udvikles.

Da der har været flest informationer til rådighed for Sejlklubben Sundet, som dermed også har haft de bedste konkretiserede krav til funktioner, vil der fremover blive fokuseret på at lave en løsning til netop dem. Dog skal det nævnes, at systemet der udvikles til Sejlklubben Sundet, vil kunne bruges af de andre sejlklubber, enten ved at ændre eller slette funktioner.

Det lader til at f.eks. Sejlklubben Sundet har brug for det projektgruppen definerer som et management system, denne findes i kapitel 6 på side 15.

Teknologianalyse 6

I det følgende kapitel findes en definition af management systemer og en analyse af de programmer, som på nuværende tidspunkt er state of the art inden for administration af sejlklubber.

6.1 Management system

Projektgruppen definerer management system til at være et program, som håndterer og digitaliserer administrativt arbejde. Med administrativt arbejde i forbindelse med en sejlklub menes der f.eks. håndtering af reservationer, opkrævning af penge, kommunikation osv.

Eksempler på management systemer findes i afsnit 6.2.

6.2 State of the art

For at kunne udvikle et godt produkt, der skal kunne bruges i en sejlklub, er det vigtigt, at se på hvilke produkter der allerede er på markedet, altså hvad der er "state of the art". I denne forbindelse er der fundet forskellige produkter, som har nogle af de features, der efterspørges i et system til en bådklub.

BoatCloud

BoatCloud er udviklet af Anderson Software [Anderson Software 2014]. BoatCloud består af 3 applikationer, StackTrack, VesselValet og TicketTracker. StackTrack benyttes, når medlemmerne selv sejler deres både, hvorimod VesselValet, er når der skal tjenere og passagerer med ombord på bådene. BoatCloud applikationerne er derfor mere designet til en sejlklub, som passer på medlemmernes både, fremfor at klubben har både til udlejning. I applikationerne kan medlemmerne meddele, at de vil sejle på et bestemt tidspunkt. Medlemmet kan få klubben til at vaske båden, tanke den, og fylde den op med diverse snacks, alt sammen registreres igennem denne webbaserede applikation, dette foregår i TicketTracker. Applikationen tager imod alle disse bestillinger 24 timer i døgnet. Herudover bliver der sendt e-mails ud til medlemmerne, som bekræftelse når de har lavet en bestilling. Man kan logge ind som administrator, og her kan man se alle reservationer, der er lavet, samt have mulighed for at se yderligere detaljer om hver enkelt reservation.

Sailing Club Manager

En anden applikation der findes er Sailing Club Manager [SailingClubManager 2014]. Denne applikation er også webbaseret, og gør det muligt for en bådklub at tilføje både, samt hvor disse er fortøjet i havnen. Man kan i en kalender lave begivenheder og reservationer, desuden er der medlemstilmeldelse til forskellige begivenheder. Applikationen kan også bruges som kontaktmedium for klubben til deres medlemmer. Klubben kan sætte et e-mail system op, samt tilføje et template, som bliver sendt med hver enkelt e-mail. Applikationen kan endvidere holde styr på økonomien, man kan tilføje en bankkonto, samt holde styr på fakturaer, og sende dem og håndtere det online. Man kan tilføje medlemsskaber, med forskellige oplysninger omkring medlemmerne, der har netop dette medlemsskab i klubben.

ForeningLet

ForeningLet er en applikation, som adskiller sig fra de andre ved ikke at henvende sig specifikt til én type klub, desuden er det en dansk applikation. Applikationen er web-baseret og har mange nyttige funktioner for en sejlklub som Sundet. Applikationen har en medlemsdatabase, et regnskabssystem samt en kontaktfunktion, som kan benytte både e-mail og SMS. Der kan også opkræves betalinger fra medlemmerne gennem systemet. Applikationen har også en reservationsfunktion, hvilket klubber som Sejlklubben Sundet kunne anvende til at administrere udlejning af deres både. Desuden kan der oprettes begivenheder i applikationen, som medlemmerne også kan tilmelde sig gennem applikationen. ForeningLet giver også foreningen mulighed for at oprette deres egen hjemmeside samt deres egen online butik.

Ud fra undersøgelsen af disse tre programmer er flg. funktioner fundet i applikationerne:

- Medlemmer kan reservere afgange.
- Medlemmer kan melde sig på begivenheder i bådklubben.
- Medlemmer kan betale igennem hjemmesiden.
- Medlemmerne kan bede klubben om at gøre deres egen båd klar, med forskellige aftaler klubben håndterer før den aftalte tid.
- Bådklubben kan vise hvilke både der er i klubben.
- Bådklubben kan oprette begivenheder, og afsætte hvilke medlemmer der kan melde sig på begivenheden.
- Bådklubben kan oprette forskellige medlemsskaber og opkræve betalinger igennem hjemmesiden.
- Bådklubben kan sende e-mails med klubbens egen skabelon gennem hjemmesiden, herunder påmindelser om en kommende reservation.

6.3 Konklusion af teknologianalyse

Ud fra state of the art afsnittet kan der konkluderes, at der findes programmer, som administrerer udlån/udlejning af både. Specielt ForeningLet har denne funktion, hvilket netop ville være en funktion som Sejlklubben Sundet ville kunne benytte sig af. Flere af funktionerne, som findes på tværs af de tre programmer, anses for at være relevante for dette projekt, og vil derfor blive taget med videre til problemløsningsdelen. Dog er der ikke nogen af disse tre programmer, der har undervisningsfunktioner, hvilket Jacob Nørbjerg gjorde udtryk for kunne

være brugbart i appendiks C på side 81. Derfor menes der, at der her findes en mangel indenfor de produkter, der allerede findes. Netop manglen af undervisningsfunktioner kunne være en grund til, at Sejlklubben Sundet ikke anvender et af disse produkter. Et lignende produkt med undervisningsfunktioner vil derfor kunne formodes at være en bedre løsning for Sejlklubben Sundet.

Problemformulering 7

Indtil nu er problemet om fritidsklubbers administrative opgaver blevet belyst og analyseret vha. Laudon og Laudons model beskrevet i kapitel 3 på side 5. Det er nu, ud fra denne analyse, muligt at komme frem til en problemformulering, som der vil arbejdes ud fra i løsningsdelen for at finde en løsning til fritidsklubber og ikke mindst sejlklubber.

Der blev i kapitel 2 på side 3, omhandlende fritidsklubber, afgrænset til sejlklubber, da disse viste sig at have flere specifikke administrative opgaver, sammenlignet med andre fritidsklubber. Denne afgrænsning blev foretaget da et system, der kan håndtere generelle samt specifikke opgaver i en sejlklub, kan bruges af andre former for fritidsklubber ved at tilpasse specifikke funktioner. Ved at analysere videre på de direkte forhold vedrørende en sejlklub, blev der dannet en forståelse for, hvilke problemer en sejlklub med sejlerskole har, som kan løses vha. et management system.

7.1 Interessenterne for sejlklubber

Der var forskellige interessenter for sejlklubberne, og de havde varierende interesse i projektet.

Medlemmerne, underviserne og de frivillige vil alle være personer, der skal bruge det udviklede system. De har derfor en mere direkte indvirkning på, hvordan system skal opbygges. Hvis systemet ikke har de funktioner, som de tre interessenter har behov for, vil det ikke være en god løsning. Dette gælder for alle de tre nævnte interessenter. Det skal dog understreges, at de ikke vil bruge samme dele af systemet.

7.2 Organisation

I forbindelse med organisationsafsnittet blev der undersøgt, hvordan sejlklubberne håndterer forskellige opgaver i klubben, samt hvilke opgaver de beskæftiger sig med.

Det viste sig, at klubberne har individuelle medlemstyper, og det kan derfor være relevant at lade klubberne selv oprette forskellige medlemstyper i systemet. Desuden efterspørges det, at man kan tilkoble sig systemet hjemmefra, for således at kunne få informationer om begivenheder og undervisning i klubben, og måske endda tilmelde sig disse, uden at tage hen til klubben.

Grundet den store mængde af information, der skal nedskrives, i forbindelse med en sejlads,

giver det mening at hjælpe med at organisere denne opgave, samt at gøre det lettere at registrere informationerne for diverse frivillige og undervisere. Desuden kunne det hjælpe, hvis hvert medlem havde en saldo over udgifter ved sejlklubben, således det er nemmere at håndtere brugerbetaling.

Følgende er en liste over opgaver, som kan dækkes af et system for sejlklubberne:

- Tilkobling hjemmefra via internettet.
- Mulighed for at få informationer vedr. begivenheder, samt at tilmelde sig disse.
- Organisering af information der nedskrives i forbindelse med en sejlads.
- Organisering af betalinger, samlet for det enkelte medlem, samt mulighed for online betaling.
- Reservation af både.
- Administrering af undervisning.

7.3 Teknologi

Sejlklubben Sundet viste sig at have flere forhold at organisere end de andre klubber der er undersøgt, og derfor afgrænses projektet til konkret at designe et IT-system til Sejlklubben Sundet. Dette gøres, da hvis et system kan hjælpe Sejlklubben Sundet, vurderes det, at det også kan hjælpe andre sejlklubber, der har færre forhold at holde styr på. Der er blevet fundet frem til at systemet, der efterspørges, er af typen management system, som blev beskrevet i afsnit 6.1 på side 15. Der findes allerede systemer, som kan dække Sejlklubben Sundets administrative behov, bortset fra håndteringen af undervisning. Der mangler altså et program på markedet, som kan håndtere en sejlklubs sejlerskole. Heri ligger projektets eksistensberettigelse.

Problemformulering

Ud fra denne afgrænsning er flg. problemformulering formuleret:

> *Det er et problem at frivillige i fritidsklubber med specielle udlejningsmuligheder, så som Sejlklubben Sundet, benytter unødvendig arbejdskraft på fysisk dokumenthåndtering vedrørende udlånte faciliteter, undervisning og begivenhedsorganisation. Hvordan kan der udvikles et system som kan hjælpe med at danne overblik over sådanne opgaver?*

7.4 Afgrænsning for problemløsning

Da dette projekt er udarbejdet i løbet af 2. semester på Aalborg Universitet, er der en begrænset mængde tid. Der afgrænses derfor fra at udvikle hele management systemet til Sejlklubben Sundet, til i stedet at lave enkelte dele af systemet.

Grundet de manglende ressourcer og tid, vil der altså ikke blive lavet et system, der kører over internettet, men i stedet et system, som kan håndtere de forskellige emner lokalt. Denne afgrænsning finder sted, da hvis funktionerne for klubben kan fungere på en computer, skal programmet tilkobles en server, for at kunne tilgås fra flere forskellige computere. Derfor ser projektgruppen altså funktionerne for Sejlklubben Sundet som værende de vigtige ting at udvikle, selvom systemet mister noget af sin brugbarhed uden internetopkoblingen. Dette ses

som værende acceptabelt, da systemet ikke skal tages i brug af klubben, da dette er et projekt lavet i forbindelse med uddannelsen på Softwareingeniørstudiet ved Aalborg Universitet.

De resterende kapitler i rapporten, vil beskæftige sig med udviklingen af et management system, henvendt imod Sejlklubben Sundet, baseret på problemformuleringen.

Som det første vil der blive udformet en kravspecifikation på baggrund af user stories.

Del B

Problemløsning

User stories 8

Til udarbejdelse af en kravspecifikation for programmet er der blevet brugt user stories. Ud fra disse user stories dannes den egentlige kravspecifikation for programmet. Under udviklingen kan det også bruges som en checkliste, så der ikke glemmes funktionaliteter.

8.1 Medlemmer

Som **bruger** kan jeg se sejlklubbens kalender.
Som **bruger** kan jeg ændre eller slette mine reservationer, hvis jeg ombestemmer mig.
Som **bruger** vil jeg kunne logge ind i programmet.
Som **bruger** vil jeg have en personlig side med overblik over mine sejlture.
Som **bruger** vil jeg gerne kunne se hvilke andre medlemmer der er i sejlklubben, for at kunne aftale fælles sejlture med dem.
Som **administrator** vil jeg gerne være i stand til at se detaljeret information om de medlemmer, der er i sejlklubben.
Som **administrator** kan jeg importere medlemmer fra andre systemer, så de kan blive genbrugt.
Som **administrator** vil jeg have en speciel administratordel af programmet.
Som **administrator** vil jeg kunne angive rettighedsniveau for medlemmerne.

8.2 Både

Som **administrator** vil jeg være i stand til at tilføje og fjerne både fra systemet.
Som **administrator** vil jeg kunne markere både som værende operationelle.

8.3 Bådudlån

Som **bruger** kan jeg reservere både.
Som **administrator** kan jeg se hvor mange ture hver bruger har taget, så jeg kan kræve det korrekte beløb fra dem.

8.4 Undervisning

Som **elev** kan jeg se informationer omkring sejlerskolen, så jeg kan følge mine fremskridt, se fremtidige lektioner osv.
Som **elev** vil jeg kunne se hvilke lektioner jeg har deltaget i og hvilke jeg ikke har.
Som **underviser** vil jeg gerne være i stand til at kontakte de elever, som er med i sejlerskolen.
Som **underviser** kan jeg se og ændre brugers fremskridt i sejlerskolen, så brugerne bliver opdateret.
Som **underviser** vil jeg kunne se en liste over mine elever.
Som **underviser** vil jeg have en oversigt over mine lektioner samt deltagere til disse.

8.5 Begivenheder

Som **bruger** vil jeg kunne se hvilke begivenheder der er i klubben samt tilmelde mig.
Som **administrator** vil jeg være i stand til at oprette/fjerne/redigere begivenheder.

Produktkrav 9

Igennem problemanalysen blev det gjort klart, at der findes et problem ved fritidsklubber, særligt Sejlklubben Sundet, når det kommer til management af deres ressourcer samt dokumenthåndtering. Problemet kan muligvis løses ved hjælp af et management system, til at håndtere dokumentation for sejlklubben og gøre det overskueligt. I følgende afsnit undersøges der, ud fra user stories kapitel 8 på side 23, hvilke krav et management system for Sejlklubben Sundet skal have.

9.1 Funktionelle krav

Det er et krav, at systemet indholder den nødvendige dokumentation, som indebærer logbøger, bådreservationer, besætningslisten, tilmeldinger til begivenheder mm. Som grundlag for, at alt dette kan fungere, kræves der en metode til at sætte sig selv på en aktivitet. For at gøre dette muligt kræves der således to ting: Et register af medlemmer, samt en form for begivenhedskalender. Foruden dette kræves der information om de forskellige begivenheder og hvem der er tilmeldt. Der kræves også et system til udlejning af både samt undervisning.

Produktet bør have følgende funktioner i prioriteret rækkefølge:

1. Medlemshåndtering
2. Brugerlogin
3. Undervisningsorganisering
4. Bådreservation
5. Logbog
6. Begivenhedsadministration og visning.

9.2 Uddybning af funktionelle krav

Der ligger mere bag de ovenstående elementer end afsnit 9.1 giver udtryk for, disse vil her blive set nærmere på.

9.2.1 Medlemshåndtering

Det er et krav, at man kan håndtere medlemmerne i klubben, så man kan registrere hvem der er tilmeldt de forskellige reservationer eller lignende. Systemet skal kunne kende forskel

på medlemmer, så man ikke kan have flere af det samme medlem på en reservation eller en begivenhed. Forskellige informationer skal kunne findes på medlemmerne i programmet, så brugerne kan finde de rigtige medlemmer og markere, om de mødte op.

9.2.2 Brugerlogin

Det skal være muligt at registrere hvem der er logget ind i programmet, for på denne måde at give forskellig adgang til diverse funktionaliteter i programmet.

9.2.3 Undervisningsorganisering

Det skal være muligt for elever ved sejlerskolen at se, hvornår deres næste lektion finder sted. Herudover skal de også kunne se, hvor langt de er kommet i forbindelse med deres uddannelse, hvilke milepæle de har udført mm. Underviserne skal have mulighed for at oprette nye lektioner, og angive hvilke læremål eleverne udførte i den pågældende lektion. Underviserne skal også kunne oprette, slette og redigere hold på skolen.

9.2.4 Bådreservation

Medlemmerne i klubben har muligheden for at reservere klubbens både, og skal derfor kunne gøre dette i programmet. Derfor skal der være en måde hvorpå, man kan tjekke om en båd er booket i en given tidsperiode, og samtidig en måde hvorpå man kan reservere båden samt slette en reservation. Herudover skal det være muligt af afmelde sig en booking, hvis man ikke deltager alligevel, og dermed åbne op for nye besætningsmedlemmer. Administrator skal have mulighed for at oprette og redigere både.

9.2.5 Logbog

I forbindelse med en sejltur, skal der kunne gemmes en logbog over turen med informationer, som beskrevet i afsnit 5.3 på side 12. Der skal her være registreret på hvilken båd turen foregik, så de korrekte logbøger kan findes for en given båd.

9.2.6 Begivenhedsadministration og visning

Endeligt skal man kunne gemme begivenheder i klubben, med en dato og en beskrivelse af begivenheden, samt slette disse igen, hvis de skulle blive aflyst. Det skal være muligt at til- og afmelde sig begivenheden, foruden at kunne se hvilke medlemmer der er tilmeldt. Desuden skal der kunne vises en liste over begivenheder sorteret efter dato, så man kan danne et overblik over, hvornår der sker hvad i klubben.

Arbejdsproces 10

I dette afsnit vil projektgruppens arbejdsproces i forbindelse udviklingen af programmet beskrives, som baseres på et begreb kaldet “agile software development”.

10.1 Agile software development

Når agile software development benyttes i udviklingsprocessen, lægges der fokus på altid at have et fungerende program. Dette anvendes for at holde programmets omfang realistisk ved tankegangen: “Begynd ikke på noget der ikke er tid til eller kun kan udvikles delvist”. Med dette menes der, at hver enkelt funktion skal fungere fejlfrit, før arbejdet med den næste påbegyndes. Der tilstræbes altså at have et mindre og fejlfrit program, fremfor et større fejlfyldt program.

Agile software development lægger yderligere fokus på “co-location”, således at arbejde bliver udført på en arbejdsplads med ens kollegaer eller medstuderende, i stedet for at hver person sidder for sig selv, heri ligger “pair programming”. Ved denne metode programmerer to programmører på en enkelt computer, med en fører og en observatør, for øget kodekvalitet.[agilemanifesto.org 2014]

10.2 Faktiske arbejdsproces

Pair programming er delvist blevet anvendt i programmeringsprocessen, dvs. at noget af kildekoden er skrevet med denne programmeringsteknik. Således er mindre dele af kode skrevet af enkeltpersoner, som efterfølgende er gennemgået af andre gruppemedlemmer. Tankegangen med at en funktionalitet skulle være færdig før der blev startet på ny, blev overholdt i starten af udviklingsprocessen. Der gik dog ikke lang tid, før der blev fraveget fra dette, da det faldt gruppen svært at opdele de enkelte funktioner i små dele, som kunne laves af forskellige personer eller pair programming grupper. I stedet for blev arbejdet med andre funktioner påbegyndt, så alle i gruppen havde noget at lave.

Grafisk teori 11

For at kunne udvikle en brugbar grafisk brugergrænseflade er det nødvendigt at forstå teorier bag dette. Derfor vil dette blive beskrevet i følgende kapitel.

11.1 Grafisk brugergrænseflade

Centralt for systemets design, er at brugerne kan navigere rundt i det. Da Sejlklubben Sundets medlemmer kan bestå af en mangfoldig gruppe af mennesker, er det svært at indskrænke brugerne i en enkelt gruppe, dermed formodes det at IT-evnerne kan være meget forskellige.

11.1.1 Designet

Programmets brugergrænseflade vil blive designet ud fra principperne i følge Kang og Kims fortolkning. [Seonghoon Kang and Won Kim 2007]

- Minimalisme
- Konsistent design (Fra engelsk: Consistency)

Minimalisme betyder, at der skal være så få forstyrrelser på skærmbilledet som muligt. Det skal være enkelt og simpelt at navigere rundt og finde de funktioner i programmet, man skal bruge. Knapper skal gerne være holdt til et minimum og mindre brugte funktioner gemmes derfor væk i menuer eller andre vinduer. En sådan struktur skal dog ikke have et hierarki, der er dybere end tre niveauer for ikke at gemme funktioner for brugeren. Unødvendige ikoner og lange sætninger er således også kun til forvirring for brugeren og skal gerne undgås. Tekst, der forekommer på brugergrænsefladen, skal også gerne være konsistente, dette gælder hele brugergrænsefladen: Farver, funktionstyper der tilføjer eller fjerner elementer, samt teksttype og størrelse for at undgå forvirring for brugeren. Således skal den samme navigationsstruktur også være tilbagevendende igennem brugergrænsefladen.

Disse forskellige principper eller retningslinjer, er forsøgt implementeret i systemet.

11.1.2 Implementation

Menustrukturen består af tabs, som er store og lette at se. De forskellige funktioner i programmet er delt op i deres tilhørende tabs, og man kan altid gå ind i en ny tab uanset hvor,

man befinder sig i programmet. Dette betyder, at programmet ikke har et dybt vindueshierarki. Brugergrænsefladen vil blive testet i brugertesten, beskrevet i kapitel 16 på side 51.

11.2 Valg af grafikframework

Dette afsnit omhandler udvalgte valgmuligheder for udviklingen af brugergrænsefladen, samt valget af det endelige framework.

11.2.1 Windows Presentation Foundation

Windows Presentation Foundation (WPF) er en grafisk framework på Windowsbaserede applikationer. WPF gør brug af moderne teknologier til at generere grafik, bl.a. vektorbaseret og hardware accelereret rendering. Visual Studio har en indbygget designer til WPF, hvilket simplificerer udviklingsprocessen. Grafikken kan enten defineres i Extensible Application Markup Language (XAML)–formatet eller ved C#–kode. Fordele ved XAML–formatet er en adskillelse af logik og grafik. Hvis grafikken kun er defineret som C#–kode kan det være svært at bevare overblikket, når kompleksiteten øges. Selve C#-koden, den såkaldte Code–Behind, ligger i en separat fil, med samme navn som XAML-filen. Hvis der f.eks. er skabt en Button, en almindelig knap, i XAML-filen, så vil det kode, som Button'en skal udføre, blive placeret i Code-Behind-filen, som består af C#–kode, så grafikken og koden bliver hver for sig, hvilket kan give et bedre overblik [Microsoft 2014].

11.2.2 Windows Forms

Windows Forms (WinForms) er, som WPF, et grafisk framework til udvikling på Windows platformen, som en overbygning til Windows API. WinForms er forgængeren til WPF, hvilket betyder at WPF har nogle nye features, som WinForms ikke har. WinForms har ikke et tilknyttet markup sprog, så selve grafikken skal skrives sammen med selve koden. [wpf-tutorial.com 2014]

11.2.3 Website

En anden mulighed, som blev diskuteret, var at bruge en hjemmeside, som ville have den fordel, at den kan køre på stort set alle enheder, som har adgang til internettet via en browser. En hjemmeside, som gør brug af C#, vil kunne laves på en måde, som minder om den ved WPF og XAML; med HTML (HyperText Markup Language), JS (JavaScript) og CSS (Cascading Style Sheets) som front end og C#–kode som back end.

11.2.4 Afgrænsning

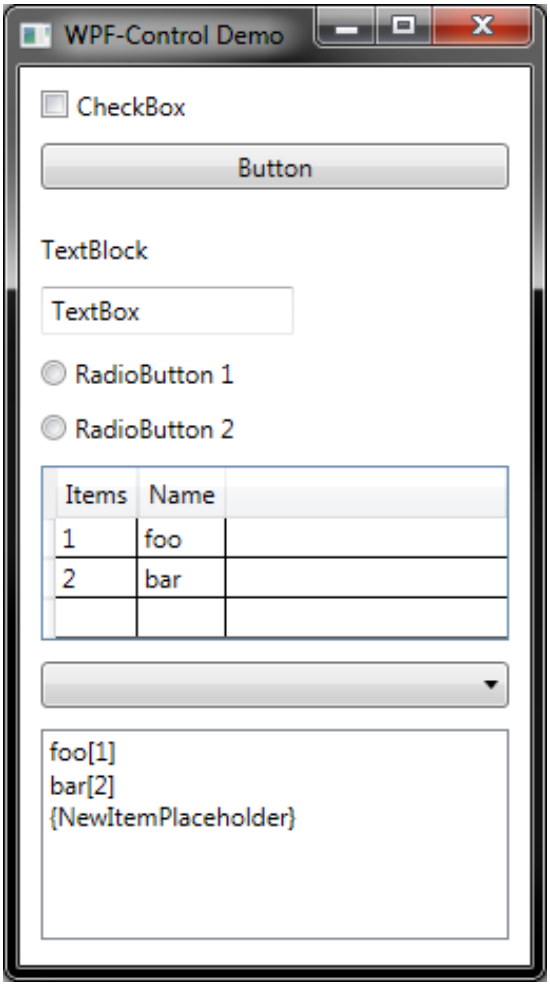
En hjemmeside blev fravalgt, da det blev vurderet at WPF var nemmere at oprette end en hjemmeside. Samtidig er fokus rettet mod objektorienterede principper fremfor udvikling af brugergrænseflade.

WinForms blev fravalgt, da det er ved at være forældet. Microsoft har informeret om, at der ikke længere tilføjes nye funktioner til WinForms, men at der udelukkende bliver lavet rettelser af fejl.[Jonathan Allen 2014]

I dette projekt bruges WPF til at skabe den grafiske brugergrænseflade. Valget faldt på WPF, da dets integration i Visual Studio simplificerer det skabelsen af en grafisk brugergrænseflade.

11.2.5 Ofte anvendte WPF-controls

I projektet anvendes bestemte WPF controls til at bygge brugergrænsefladen. De hyppigst anvendte vil her blive beskrevet. På figur 11.2.4 vises de beskrevne elementer.



Figur 11.1. Demonstration af WPFs Controls

ComboBox

I WPF er en ComboBox det element, som andre steder omtales som en dropdown menu. Dens indhold kan indstilles enten i XAML-koden eller i Code-Behind koden. Hvis dette indhold skal være dynamisk, kan man anvende Code-Behind, eksempelvis hvis man kun vil have de medlemmer af en liste, som opfylder et givent prædikat.

Button

En Button er en knap, som kalder en metode i Code-Behind, når den trykkes på.

DataGrid

Et DataGrid er et grafisk element, som kan vise data på tabelform. Ofte vil det vise en liste af objekter. Dets layout er opdelt i rækker og kolonner, hvor hver kolonne indeholder en bestemt type data, og kan sorteres ved at klikke på de forskellige headers. Det er også muligt at konstruere en søgefunktion, altså kan et DataGrid blive filtreret.

CheckBox

En CheckBox er en kvadratisk boks, som enten er “Checked” eller “UnChecked”. I code–behind kan stadiet af en CheckBox aflæses.

RadioButton

En RadioButton er en cirkelformet CheckBox. Dog vil RadioButtons ofte optræde i serie, altså to eller flere sammen, da valget af den ene ekskluderer valget af de andre. Et oplagt brug af dem er til ja, nej (og måske) situationer, hvor kun en af dem skal vælges.

TextBlock

En TextBlock bruges til visning af tekst, som ikke kan redigeres. Det vil typisk være en forklarende tekst op ad et andet grafisk element. Det er også muligt at ændre en TextBlock via Code-Behind, hvis man vil føre en dialog med brugeren, eksempelvis til fejlbeskeder.

TextBox

En TextBox er et tekstfelt, hvori brugeren kan indtaste tekst. Dette kan dermed ses som et inputfelt. Det er også muligt at sætte sådan et felt til at være skrivebeskyttet, så brugeren ikke kan redigere det, imens programmet stadig kan ændre feltets indhold.

ListBox

En ListBox er en liste, hvori en bruger kan vælge en eller flere elementer. ListBoxen er i stand til at indeholde samlinger af data på enhver form, oftest vil det være en streng, men et billede er også en mulighed. Den vil i nogle tilfælde have samme brugsscenarie som en ComboBox eller et DataGrid. Forskellen fra en ComboBox er, at der kan være flere synlige elementer i en ListBox på samme tid, samt der ikke er nogen dropdown menu.

UserControls

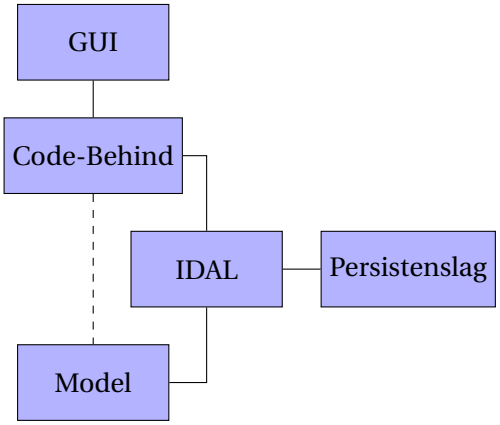
Det er muligt at konstruere sine egne grænsefladeelementer fra et eller flere af WPFs indbyggede eller 3. parts Controls. Dette kaldes en UserControl, som indeholder både en grafisk del og en Code-Behind del. Det brugerskabte element kan derefter genbruges flere steder i programmet. Dette er et eksempel på genanvendelse, hvilket kan bidrage til højere programkvalitet og større konsistens gennem programmet.

Vinduer

Et vindue er det som oftest indkapsles af kanter og knapperne minimer, maksimer og luk, som findes på Windows øverst i højre hjørne. I vinduer indsættes alle de ovenstående elementer, for at skabe og vise den grafiske brugergrænseflade.

Programopbygning 12

I dette kapitel beskrives programmets opbygning.

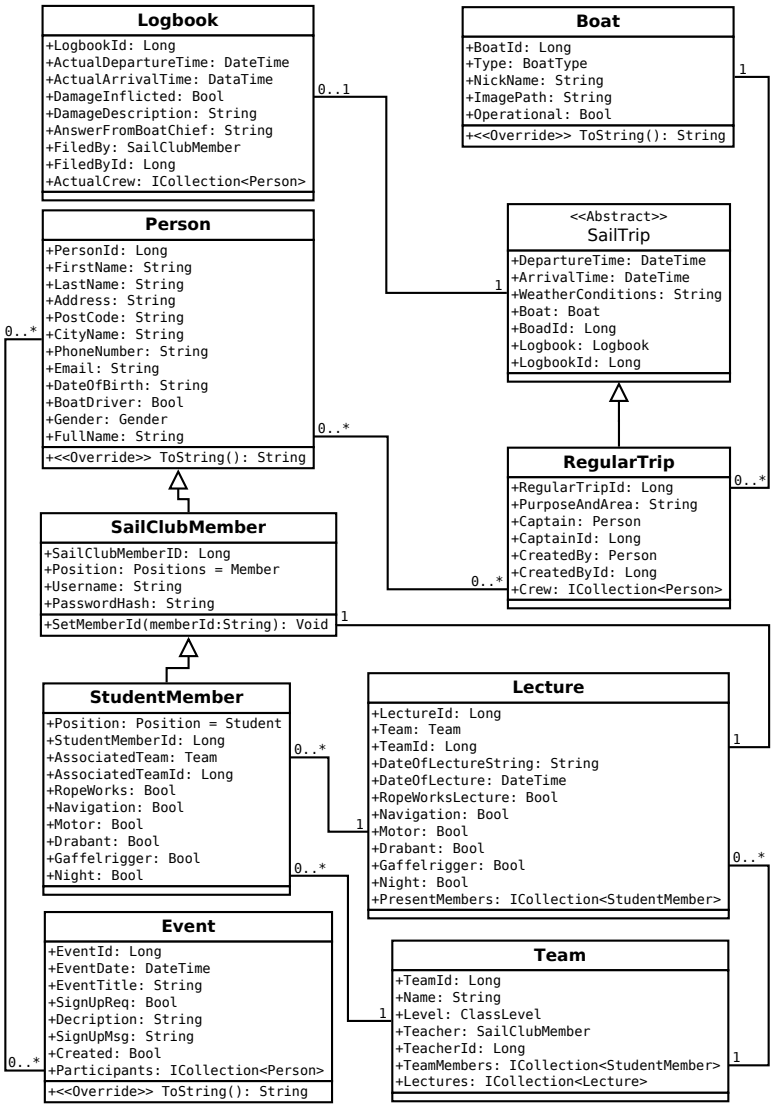


Figur 12.1. Programopbygning: Figuren viser sammenhængen mellem de forskellige komponenter i programmet.

På figur 12.1, ses den overordnede struktur af programmet. Ud fra figuren kan der ses de forskellige elementer programmet består af. Nederst på figuren ses boksen *Model*, som repræsenterer modellaget. Heri findes de modeller, som bliver brugt i programmet, de er nærmere forklaret i kapitel 13 på side 35. Modellaget har en forbindelse til det generiske Data Abstraction Layer interface (IDAL), hvilket kan ses på figuren *IDAL* boksen. Dette har en forbindelse til persistenslaget, som er nærmere forklaret i kapitel 14 på side 39. Yderligere ses en forbindelse fra boksen *IDAL* til boksen *Code-Behind*. Denne boks repræsenterer programmets bagvedliggende kode, som gennem IDAL interfacet kan tilgå persistenslaget. Det ses ydermere på figuren, at der er en forbindelse til brugergrænsefladen (GUI) repræsenteret med boksen *GUI*. Koden har yderligere kendskab til modellaget, illustreret med en stiplet linje, for at kunne instantiere modellerne så data kan håndteres. Dette er den del af programmet, som brugeren kan se, når brugeren laver en aktion i programmet kaldes Code-Behind'en for at udføre denne, heraf forbindelsen mellem de to. En nærmere forklaring af *GUI* og *Code-Behind* boksene følger i kapitel 15 på side 43.

Modeller 13

I dette kapitel vil klasserne i modellaget blive beskrevet. Den overordnede klassestruktur er beskrevet i UML–diagrammet på figur 13.1, heri kan alle de konkrete felter som findes i hver klasse ses.



Figur 13.1. UML-diagram over klasserne i programmet

Klasse: Person

Formål: Formålet med personklassen er at kunne repræsentere personer som vedrører programmet. Data om disse personer angives som felter i personklassen.

Metoder: Personklassen har overridet **ToString()**, som returnerer **FullName**.

Anvendelse: Personklassen bruges til at repræsentere alle personer i systemet. Gæster repræsenteres direkte som instanser af personklassen. Sejlklubbens medlemmer repræsenteres som subklasser af personklassen.

Klasse: SailClubMember

Formål: SailClubMember bruges til at repræsentere medlemmerne i sejlklubben, med undtagelse af de studerende som har en subklasse.

Position feltet angiver hvilken rang, og dermed rettigheder, som et sejlklubmedlem har. Dette repræsenteres ud fra en enumeration.

Anvendelse: SailClubMember bruges til at danne instanser af alle sejlklubbens medlemmer, bortset fra de studerende som er instanser af StudentMember der nedarver fra SailClubMember.

Klasse: StudentMember

Formål: Klassen StudentMember indeholder informationer om, hvilke læringsmål eleven har udført i forbindelse med sejlerskolen.

Anvendelse: Klassen StudentMember nedarver fra SailClubMember og bruges til at repræsentere eleverne i sejlklubben.

Klasse: Team

Formål: Klassen Team har til formål at repræsentere et vilkårligt sejlerskolehold. Samtidigt inderholder den både en liste over elever og lektioner.

Anvendelse: Team anvendes til at samle en gruppe elever i sejlerskolen. Denne samling gør det muligt at håndtere eleverne på holdet på samme tid.

Klasserne: SailTrip og RegularSailTrip

Formål: I en tidlig designstruktur var det tænkt at programmet skulle indeholde flere forskellige typer af sejlture, og derfor blev SailTrip-klassen skabt som værende superklasse for sejltursklasser. Senere blev ideen om at have flere typer sejlture ændret, og i stedet blev der valgt, at der kun skulle være én sejlturstype. SailTrip-klassen er dog bibeholdt i tilfælde af videre udbygning af programmet. Dog findes der enkelte felter i klasserne, som overlapper hinanden lidt, men felter er også bibeholdt i tilfælde af udbygning til flere typer sejlture.

Anvendelse: SailTrip-klassen anvendes kun som superklasse for RegularTrip-klassen. Hver reservation er en instans af klassen RegularTrip, dette gælder også reservationer for sejlerskolens lektioner.

Klasse: Event

Formål: Denne klasse repræsenterer en begivenhed i sejlklubben.

Anvendelse: En instans af eventklassen, repræsenterer en begivenhed, der ikke nødvendigvis er sejlrelateret.

Klasse: Logbook

Formål: Efter hver sejltur skal der udfyldes en logbog med informationer om sejlturen. Denne klasse repræsenterer denne logbog.

Anvendelse: En instans af klassen Logbook findes på hver sejltur. Den udfyldes af personen, som har foretaget reservationen, efter sejladsens fuldførelse.

Klasse: Lecture

Formål: Klassen Lecture indeholder information vedrørende en undervisning.

Anvendelse: Denne klasse bruges, når der reserveres en båd til sejlerskolen og samtidigt til at registrere, hvilke læringsmål som blev udført på turen.

Persistenslag 14

Dette kapitel omhandler forsøget på at opnå persistens mellem kørsler af programmet. Det ønskes, at programmet kan lagre sit data, udenfor RAM, for at kunne bruge det igen senere. Det var et fælles ønske fra gruppens side, at der blev brugt tid på at tilføje dette element til programmet.

14.1 Persistensløsninger

Der findes flere løsninger til denne problemstilling. Der kan vælges en bestemt database ud fra de eksisterende databaser, eller dataet kan gemmes på flade filer.

14.1.1 Flade filer

En simpel løsning på persistensproblemet ville være at gemme alt data på en flad fil ved programlukning, og læse filen igen ved programmets opstart. Denne løsningen bliver dog hurtig upraktisk, da flade filer ikke har samme evne til at skalere til et større system som egentlige databaser.

14.1.2 Databaser

Efter research, herunder rådgivning fra OOP lektor Rene Hansen, stod valget mellem to databaser: Entity Framework og SQLite.

Entity Framework

Den første af de to, Entity Framework (EF), er en Object-Relational Mapping (ORM) til .NET Frameworket. Et ORM skaber et billede af objekterne i hukommelsen i databasen [Wikipedia 2014b]. Herunder var rådgivningen fra Rene Hansen, at anvende Code-First udvikling. Med Code-First designes objekterne først, i form af modelklasser, hvorefter databasen og tabellerne i denne genereres ud fra sammenkoblingen mellem de forskellige modeller. Denne sammenkobling sker i såkaldte migrations, som opretter, ændrer og sletter tabeller, afhængigt af ændringerne i de benyttede modeller. Fordelen ved denne løsning er, at mange tekniske valg træffes af programmet, så programmøren ikke behøver at have et kendskab til Structured Query Language (SQL), for at kunne benytte databasen og dermed opnå persistens. Dette gør EF til en attraktiv løsning for udviklere, der gerne vil hurtigt i gang med at programmere en løsning, uden først at sætte sig ind i databaseprogrammering. [Wikipedia 2014a]

SQLite

Den anden kandidat er SQLite, som er et Relational Database Management System (RDBMS), hvilket betyder, at det benytter SQL til de forskellige former for datahåndtering, såsom INSERT til at indsætte data, og SELECT til at hente data fra databasen. SQLite er en letvægts embedded database, der er velegnet til mindre programmer, som har behov for lokal persistens. At en database er embedded betyder, at den ikke kører i et separat program, men inkluderes som et bibliotek i programmet. SQLite implementerer det meste af SQL-standarden, men udelader de dele, der af SQLite-udviklerne er bedømt til at være for tunge eller unødige til at have med. Målet med SQLite er at bibeholde letvægten, da systemet også benyttes på mange mobile enheder, såsom Android [developer.android.com 2014]. SQLite opfattes af dets udviklere, som værende den mest distribuerede SQL database i verden [sqlite.org 2014b]. [sqlite.org 2014a]

Begge databaser blev implementeret under projektarbejdet, grundet komplikationer med førstevalget af database, Entity Framework.

14.2 Database Abstraction Layer

Et Data Abstraction Layer (DAL) bruges til at separere programmet fra databasen. Dette gør det muligt at udskifte det underliggende persistenslag, samt at lave lag, der simulerer persistens, ved udelukkende at gemme data i hukommelsen, så længe programmet kører. Man udstiller et interface til programmet, som så anvendes, og derfor ikke en direkte forbindelse. Hver tabel som findes i den database programmet skal anvende, skal have sin egen DAL, som er logikken der forbinder til databasen. Der findes fire basale operationer til en database, kaldt CRUD: Create, Read, Update, Delete. Hver af disse operationer skal DAL'et udstille, og yderligere metoder kan tilføjes efter behov.

14.3 Den valgte løsning

Efter anbefaling fra OOP lektor Rene Hansen, valgte gruppen at fokusere på Entity Framework med Code-First, grundet manglende kendskab til SQL. Det blev yderligere bestemt, at der skulle designes et DAL, med henblik på at øge muligheden for at teste systemet.

14.3.1 Design af DAL

Der blev udviklet et generisk DAL-interface, som hver af de konkrete DAL-interfaces nedarver fra. Interfacet definerer otte metoder, som beskrevet i tabel 14.1 på næste side.

De konkrete interfaces defineres for hver modelklasse og implementeres i specifikke klasser, der kommunikerer med et persistenslag. For at gøre det lettere at udskifte DAL-implementation, blev en statisk hjælpeklasse, `DalLocator`, oprettet, hvis opgave består i at returnere en implementation af et DAL-interface. Hermed skal der kun foretages ændringer i en enkelt fil, hvis et nyt DAL tages i brug. På figur 12.1 på side 33 kan man se hvordan DAL indgår i programmets struktur.

Metode	Forklaring
`Create`	Opretter et eller flere nye elementer i databasen.
`Update`	Opdaterer et eller flere elementer i databasen.
`Delete`	Fjerner et eller flere elementer fra databasen.
`GetAll`	Findes i to udgaver, en som henter alle elementer, og en hvor der kan medsendes et prædikat.
`GetOne`	Henter et element i databasen ud fra det angivne itemId.
`LoadData`	Findes i to udgaver, de henter de indre referencetype objekter til elementet eller elementerne givet.

Tabel 14.1. Forklaring af hver metode i det generiske DAL interface.

14.3.2 Første implementation — Entity Framework

Entity Framework blev implementeret for modellerne, og systemet blev bygget herpå. Under programmets udvikling stod det klart, at det ikke var helt problemfrit at benytte EF.

Migrations kom i vejen for hinanden og gav konflikter, der krævede, at gruppens medlemmer regelmæssigt skulle slette hver tabel og kalde kommandoen `Update-Database`. Denne kommando får EF til at gennemløbe alle migrations og udføre handlingerne i disse.

Systemet reagerede også sporadisk, hvor det ét sted fungerede som det skulle, mens det fejlede lydløst, altså uden en fejlbesked, andre steder. Dette til trods for, at de to områder var programmeret ens.

14.3.3 Anden implementation — SQLite

Gruppen aftalte at forsøge en implementation af SQLite i stedet for. Det var på dette tidspunkt, at det udviklede Data Abstraction Layer viste sin styrke, da hele persistenslaget kunne udskiftes uden konflikter i koden.

Den implementerede løsning er relativt simpel i sin konstruktion, især i forhold til Entity Framework. Dette blev gjort for at sikre, at det ikke ville tage mere end et par dage at implementere og teste.

SQLites enkeltbruger-design har dog også givet problemer, idet databasen til tider angiver, at den er låst af en anden proces. Disse problemer kan forårsages af flere ting, og virker til at opstå tilfældigt, så det er svært at forudse, hvor og hvornår det vil ske.

14.3.4 Ekstra implementation — Mock-data

Som en backup–løsning, i tilfælde af, at SQLite pludseligt fejler, er en mock–implementation designet. Denne implementation simulerer persistens, men gemmer kun i RAM mens programmet kører, og slettes derefter.

Mock-data-persistens

Idet persistens er vigtigt for gruppen, blev der udviklet et persistenslag til mock-data-implementationen. Dette persistenslag benytter ligeledes SQLite, men gemmer kun data når programmet lukkes, og henter det igen når programmet starter op. Denne løsning er skrøbeligt

overfor eventuelle computernedbrud, idet data i så fald ikke ville blive gemt i databasen. Det ville være en mulighed at gemme data hver gang der trykkes på “Log ud”–knappen, eller måske at have en decideret “Gem”–knap.

Programmets brugergrænseflade 15

I dette kapitel vil programmets grafiske brugergrænseflade blive beskrevet.

15.1 Hovedvinduet

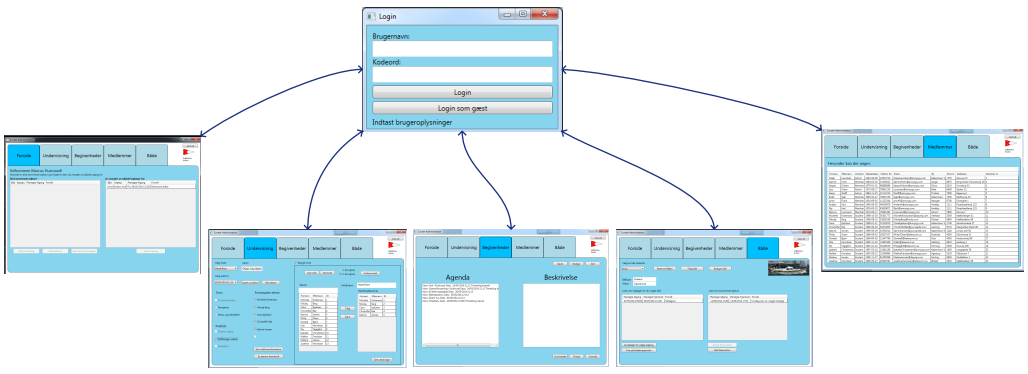
Hovedvinduet tilgås via vinduet **Login**, som åbner ved programstart.

Der findes fire udgaver af hovedvinduet, forskellen mellem dem er, hvilke tabs der er aktive. Hver tab indeholder forskellige funktioner, samlet set findes følgende tabs:

- Forside
- Undervisning
- Begivenheder
- Medlemmer
- Både.

Alle tabs kan ses på figur 15.1.

Via hver af disse tabs, vil der være adgang til programmets forskellige funktionaliteter. Programmet er lavet til at køre i opløsningen 1024x720 pixels. Denne opløsning er valgt, da stort set alle computerskærme har en opløsning større end 1024x720 pixels [w3schools.com 2014]. Programmet har et lyst farveskema; med farverne lyseblå og hvid som primære farver.



Figur 15.1. Hovedvinduets tabs.

På figur 15.1 på forrige side kan man se vinduet **Login**, der åbner hovedvinduet, som består af 5 tabs, som pilene på figuren viser.

15.2 UserControls

Der anvendes UserControls til at kode både brugergrænsefladen og den tilhørende Code-Behind. Hver usercontrol kan ses på figur 15.1 på foregående side

15.2.1 DateTimePicker

Formål: Denne UserControl er lavet, da der ikke fandtes en tilfredsstillende løsning, som gjorde det muligt at vælge både dato og tidspunkt i samme control. Det er ofte nødvendigt at vælge både dato og tidspunkt samtidigt i programmet.

Figur 15.2. DateTimePicker

BrugerGrænseflade: Den består af en DatePicker og en TimePicker, som findes i extended WPF toolkit.

Code-Behind: Der er lavet en specifik getter og setter for UserControlen, som udnytter både controllens DatePicker og TimePicker.

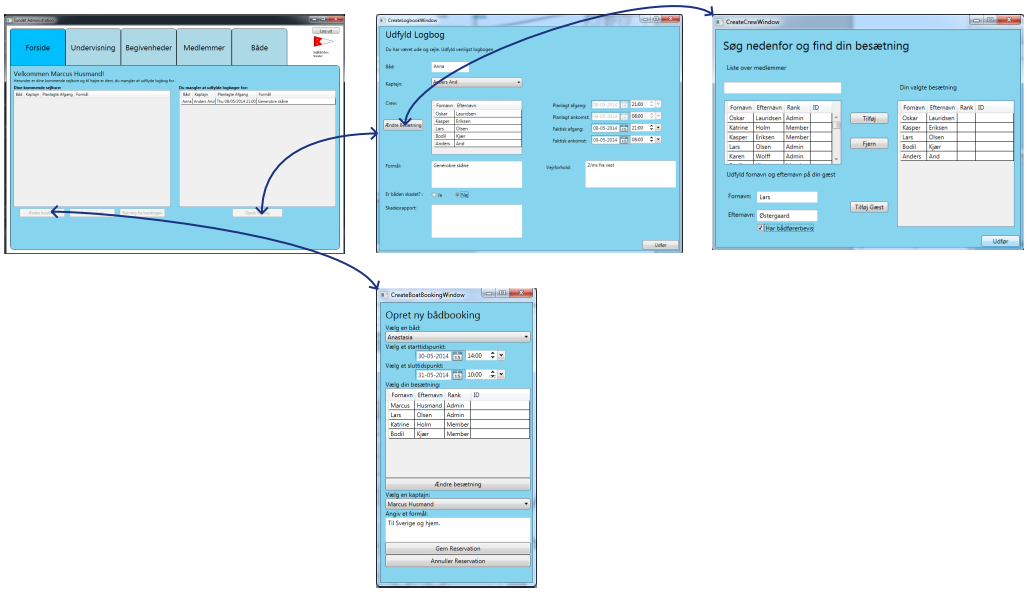
15.2.2 Forside

På figur 15.3 på modstående side ses tabben **forside**. Der vises desuden hvilke vinduer, som kan åbnes fra tabben ved et tryk på diverse knapper, som findes i tabben.

Formål: Formålet med forsiden er at vise aktuel infomation på en overskuelig måde for brugeren. Det er den første side i hovedvinduet, som man ser, med mindre man er gæst. Fra forsiden kan man ændre og slette sine reservationer, samt starte oprettelsen af en logbog.

	Gæst	Medlem	Elev	Administrator
Personlig forside		✓	✓	✓
Se begivenheder	✓	✓	✓	✓
Tilmeld begivenheder		✓	✓	✓
Opret begivenheder				✓
Se sejlture	✓	✓	✓	✓
Opret sejltur		✓	✓	✓
Se logbøger	✓	✓	✓	✓
Opret logbog		✓	✓	✓
Svar på logbog				✓
Se undervisningstimer			✓	✓
Opret undervisningstimer				✓

Tabel 15.1. Tabel over alle brugerniveauer og deres tilladte funktioner.



Figur 15.3. Forsidentabben og dens vinduer.

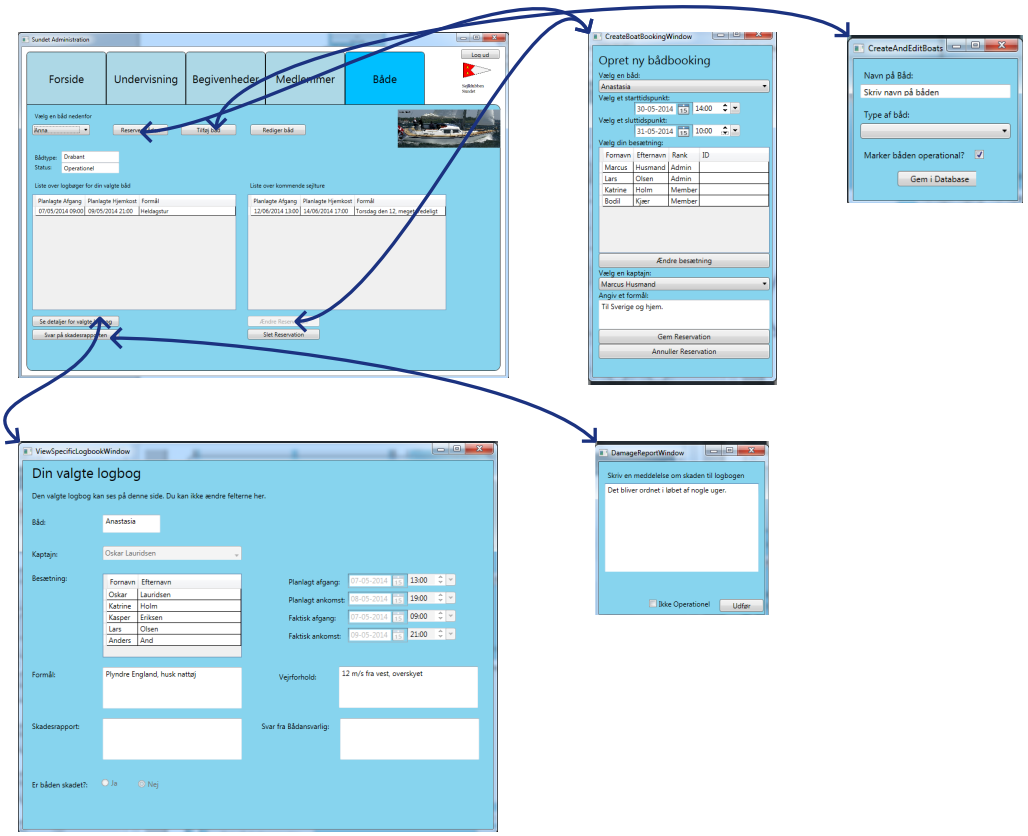
Brugergrænseflade: Brugergrænsefladen på forsiden består primært af to DataGrids: Det venstre DataGrid viser brugerens kommende sejlture, og det højre viser de sejlture brugeren mangler at udfylde logbøger for. Under dem er der knapper, som bliver aktive, efter en markering er udført ved at trykke på en af rækkerne i knappernes tilhørende DataGrid.

Code-Behind: For kun at vise de sejlture hvor personen, som er logget ind, deltager i anvendes der standard query operatorer. I listing 15.1 er der et udsnit af koden, nærmere bestemt den del som vælger de korrekte sejlture.

Begge udtryk returnerer en IEnumerable, som derefter assignes som ItemsSource til hvert DataGrid.

```
1 // Sets the trips for the person currently logged in while only
     getting the ones in the future to the DataGrid ItemsSource
2 UpcommingTripsDataGrid.ItemsSource =
3     sailTripList.Where(t => t.Crew.Select(p =>
          p.PersonId).Contains(usrId))
4          .Where(t => t.DepartureTime > DateTime.Now);
5 // Sets trips created by the current user which happened in the past
     while also missing a logbook, as the other DataGrids ItemsSource
6 LogbookDataGrid.ItemsSource =
7     sailTripList.Where(t => t.Captain.PersonId == usrId &&
          t.ArrivalTime < DateTime.Now && t.Logbook == null);
```

Listing 15.1. Forsidens Code-Behind



Figur 15.4. Bådetabben og dens vinduer.

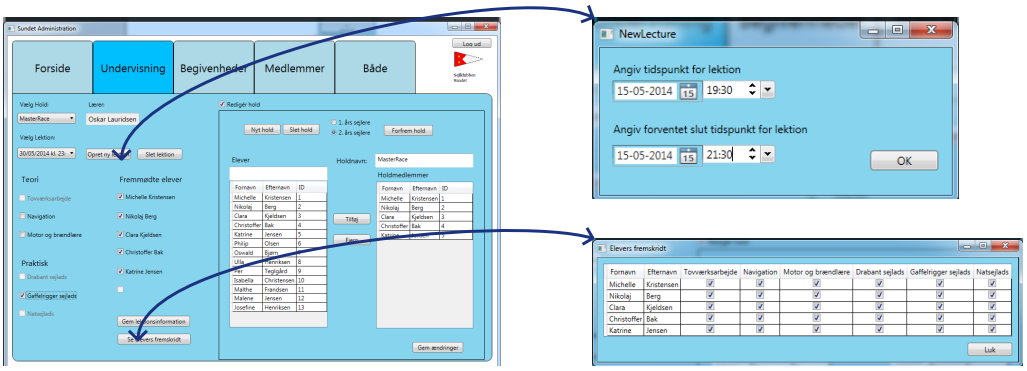
15.2.3 Boat

På figur 15.4 ses tabben **Både**. Der vises desuden hvilke vinduer, som kan åbnes fra tabben ved et tryk på diverse knapper, som findes i tabben.

Formål: I tabben **Både**, finder man UserControllen **Boat**. Her kan man få et overblik over hvilke både, der er til rådighed i sejlklubben inklusiv bådtype og status på de enkelte både. Man kan også reservere en båd, se liste over logbøger for en valgt båd, svare på skadesrapporten og ligeledes se en liste over kommende reservationer.

Brugergrænseflade: Efter valg af båd opdateres de resterende elementer i UserControllen. Her kan der ses et DataGrid med udfyldte logbøger samt et andet DataGrid med kommende reservationer på båden. Som administrator kan man også svare på logbogens skadesrapport. Foruden disse funktionaliteter kan man også herinde reservere den valgte båd. Desuden har administratorer mulighed for at tilføje og redigere både.

Code-Behind: Den bagvedliggende kode henter data fra databasen og opdaterer de grafiske elementer med dataet.



Figur 15.5. Undervisningstabben og dens vinduer.

15.2.4 StudyTeacher

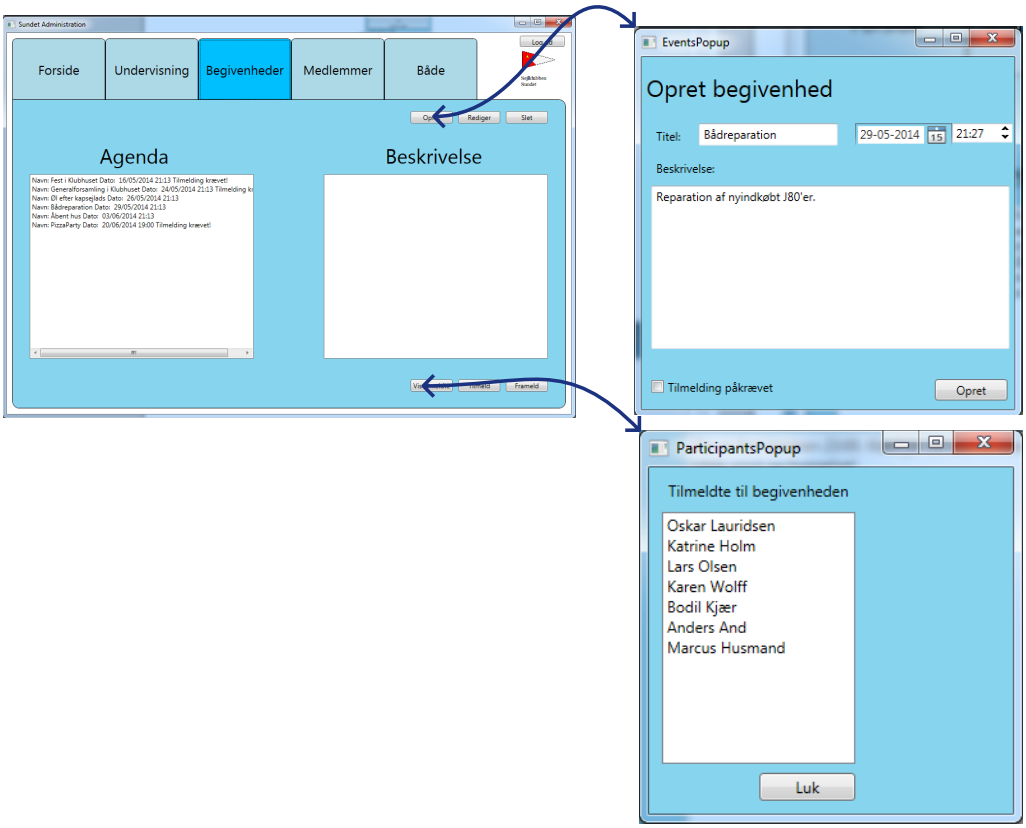
På figur 15.5 ses tabben **Undervisning**. Der vises desuden hvilke vinduer, som kan åbnes fra tabben ved et tryk på diverse knapper, som findes i tabben.

Formål: Denne UserControl bruges til give undervisere på sejlerskolen mulighed for at administrere skolehold og undervisningslektioner.

Brugergrænseflade: Når der er valgt et hold, er der forskellige controls som opdateres. Man kan se holdets underviser, elever og hvorvidt holdet er et 1. eller 2. års hold. Det er også muligt at se holdets lektioner, samt oprette og slette disse. Når der trykkes på knappen "Opret ny Lektion", åbnes et nyt vindue, hvor man skal indtaste start- og sluttidspunkt, og trykke ok, for at afslutte sin oprettelse af lektionen. Efter der er valgt en lektion, kan der afkrydses hvad der er lært på lektionen, samt hvilke elever der var fremmødte. Ved tryk på knappen "Se elevers fremskridt" åbnes et vindue, hvor man kan se hvilke læringsområder, de enkelte elever har gennemført. Når der laves et nyt hold, åbnes et vindue, hvor man blot indtaster det ønskede navn for holdet. Der kan slettes og forfremmes hold, hvilket betyder de får et bådførerbevis.

Code-Behind: Når der oprettes en lektion, bliver der reserveret den korrekte båd på det pågældende tidspunkt. Når der forsøges at forfremme et hold, tjekker Code-Behinden, om eleverne har, lært alt det de skal og ændrer herefter eleverne fra StudentMember til SailClubMember og sætter deres BoatDriver property til `true`.

Der findes også en undervisnings UserControl for elever. Denne kan ikke interageres med, men viser blot hvilke læringsområde brugeren har lært, informationer om det hold de er på, samt hvornår deres næste lektion er planlagt.



Figur 15.6. Begivenhedstabben og dens vinduer.

15.2.5 Event

På figur 15.6 ses tabben **Begivenheder**. Der vises desuden hvilke vinduer, som kan åbnes fra tabben ved et tryk på diverse knapper, som findes i tabben.

Formål: UserControllen Event bruges til at se informationer om kommende begivenheder i sejlklubben, samt til at tilmelde sig begivenhederne. Hvis man er administrator, kan man også oprette nye begivenheder, redigere og slette eksisterende begivenheder.

Brugergrænseflade: Brugergrænsefladen består af to ListBoxe og nogle knapper. I den første ListBox bliver begivenheder vist med navn, dato og om hvorvidt der kræves tilmelding. I den næste ListBox vises beskrivelsen af begivenheden, som opdateres, når der klikkes på en begivenhed i den førnævnte ListBox.

Code-Behind: Den bagvedliggende kode henter oprettede begivenheder fra persistenslaget. Ved oprettelse, redigering og sletning skrives ændringerne til persistenslaget. Ved tilmelding tilføjes personen, som er logget ind, til listen over tilmeldte personer.

15.3 Vinduer

Alle vinduer beskrevet i dette afsnit kan ses på figur 15.1 på side 43.

15.3.1 Login

Formål: Vinduet **Login** har til formål at verificere en brugers identitet. Det er det første vindue brugeren ser, når programmet åbnes, og det åbner hovedvinduet efter succesfuldt login.

Brugergrænseflade: Der er en TextBox til brugernavnet og en PasswordBox til kodeordet. En PasswordBox er en TextBox, hvor hver af de indtastede karakterer vises som en sort prik, i stedet for de skrevne tegn, for at beskytte brugeren. Knappen "Login som gæst" kræver ingen brugeroplysninger og åbner en let udgave af hovedvinduet, uden særlige tilladelser. Til sidst er en TextBlock, som fortæller brugeren, hvad og hvis, der er skrevet noget forkert.

Code-Behind: Efter der trykkes på knappen "Login", verificeres infomationen som er indtastet. TextBoxen, hvor brugernavnet indtastes, er case insensitive. PasswordTextBoxen er case sensitive. Hvis det er gyldigt, åbnes en passende udgave af hovedvinduet i hendhold til tabel 15.1 på side 44.

15.3.2 CreateBoatBookingWindow

Formål: Dette vindue opretter eller ændrer en instans af RegularTrip, det vil sige en bådreservation.

Brugergrænseflade: Først benyttes en ComboBox til at vælge hvilken båd, der skal reserveres. Herefter anvendes UserControllen **DateTimePicker** til at vælge et start- og sluttidspunkt. Der vises den nuværende besætning, samt muligheden for at ændre den ved at trykke på knappen "Ændre Besætning", som åbner **CreateCrewWindow**. Når en besætning er valgt, kan der vælges en person ud fra besætningslisten, som har førerbevis, som kaptajn. Til sidst er der en TextBox, hvori brugeren kan angive formålet med turen.

Code-Behind: I Code-Behinden er der to constructors. Den ene bruges til at oprette en ny reservation, den anden bruges når der skal ændres på en eksisterende reservation. Efter tryk på knappen "Gem Reservation", verificeres brugerens input fra alle de grafiske elementer i vinduet, og passende responsbeskeder gives, hvis noget er ugyldigt.

15.3.3 CreateCrewWindow

Formål: Dette vindue åbnes to steder i programmet: i **CreateLogbookWindow** og i **CreateBoatBookingWindow**. Det bruges, når der skal laves en besætning til en RegularTrip.

Brugergrænseflade: Der er to DataGrids, som hver indeholder en liste. Listen til venstre består af SailClubMembers, som bliver hentet ind fra persistenslaget. Listen til højre indeholder det Crew, som man er i gang med at udforme til enten RegularTrip eller Logbook. Der er også tilføjet TextBoxe, så man kan skrive navnet på en gæst, man tog med på sejlturen. Der er knapper, som tilføjer de forskellige personer til besætnings-listen. Øverst findes også et tekstfelt til at søge i listen over medlemmer. Når man har lavet sin liste, kan man trykke Udfør, for at komme tilbage til vinduet, der kaldte CreateCrewWindow.

Code-Behind: Der er blevet brugt en regular expression til at tjekke, om den string brugeren angiver i de to TextBoxe for gæstens navne, er gyldige. Det er blevet valgt, at man må bruge hele det danske alfabet samt mellemrum, så navne såsom: Lars Peter Østergaard, er mulige. Derefter tilføjes personen til listen, der vises i DataGridet til højre, og til sidst kaldes RefreshDataGrid, som man kan se på listing 15.2.

Her modtages der et DataGrid, som skal have dets Itemssource opdateret, og en ICollection<Person>, som er det data, der skal sættes ind i DataGridet. Det gøres ved at assigne dets ItemsSource til null, og derefter assigne det tilbage til den ICollection, der blev sendt med. Der kan laves en tilsvarende metode, som opdaterer andre WPF-controls, for andre ItemsSources.

```
1 private void RefreshDatagrid(DataGrid Grid, ICollection<Person> list)
2 {
3     Grid.ItemsSource = null;
4     Grid.ItemsSource = list;
5 }
```

Listing 15.2. Refresh Datagrid

15.3.4 CreateLogbookWindow

Formål: Dette vindue bruges til at udfylde logbogen for en sejltur.

Brugergrænseflade: Når vinduet åbnes, indlæses der data fra den instans af RegularTrip, som logbogen udfyldes for, hvilket bliver sendt til vinduets constructor. Som tidligere vinduer, bliver dataet afbilledet i de respektive grafiske elementer. De tomme felter skal udfyldes af brugeren.

Code-Behind: Code-Behinden verificerer dataet inden det gemmes i databasen. Herudover sættes referencen til logbogen i det pågældende RegularTrip.

Test af program 16

I dette kapitel beskrives de to typer af tests, der er udført i forbindelse med projektet. Under udviklingen af programmet har projektgruppens medlemmer testet funktionaliteterne løbende ved hjælp af blackbox testing.

16.1 Unit test

Til test af selve koden er der benyttet Unit tests.

Unit tests benyttes primært til at teste metoder, samt properties med logik. Når man har skrevet en metode, så forventer man, som regel, et bestemt resultat ved et givent input. Unit test af en metode fungerer ved, at man giver metoden, som skal testes, et input, og ser om man får det forventede resultat.

Der findes flere frameworks til unit testing. Til projektet benyttes frameworket NUnit, som er et open source unit testing framework til .NET. Det er ikke alt kode, som bliver testet, men relevante metoder og properties i udvalgte klasser. Der er bl.a. skrevet unit tests til noget af mock data DAL–implementationen, for at afprøve nogle formodninger, før hele programmet blev benyttet i testen. De valgte tests på dét område blev valgt ud fra formodningen om, at fejl i de metoder kunne vise sig som anderledes fejl andetsteds i programmet, og dermed gøre det svært at lokalisere den egentlige fejl.

Noget af det mest gennemtestede kode, baseret på dets høje code coverage, er en hjælpeklasse, der importerer medlemmer fra en XML–fil, ind i det bagvedliggende persistenslag. Code coverage er en indikator for, hvor stor en procentdel af koden der testes. Denne hjælpeklasse benyttes ikke i programmet, men er forblevet deri som kode, med henblik på, at den kunne blive nyttig i forbindelse med en udvidelse af programmet.

I listing 16.1 på næste side ses en såkaldt SetUp–metode, som køres inden hver test udføres, og dermed gør det muligt at nulstille til en bestemt tilstand, så tests kan udføres på en ensartet, systematisk måde. Til test af XML–import er der blevet benyttet et substitutionsframework, kaldet NSubstitute, der muliggør simulering af interfaces, når den bagvedliggende implementation ikke er vigtig for funktionaliteten. Her bruges den specifikt til at simulerer den del af programmets DAL, som håndterer medlemmerne i sejlklubben.

```
1   [SetUp]
2   public void Initialize()
3   {
4     _memberDalSubstitute = Substitute.For<ISailClubMemberDal>();
5     _parser = new XmlMemberParser(_memberDalSubstitute);
6     _sailClubMemberList = new List<SailClubMember>();
7
8     // React to "creation" calls
9     _memberDalSubstitute
10          .Create(Arg.Any<SailClubMember>())
11          .ReturnsForAnyArgs(true)
12          .AndDoes(x =>
                _sailClubMemberList.AddRange((SailClubMember[])x[0]));
13  }
```

Listing 16.1. Eksempel på Unit test SetUp

Når SetUp–metoden er skrevet, skrives de enkelte tests, ud fra devisen om, at hver test kun skal teste én ting. Der er flere måder at navngive tests på. Den metode, som gruppen valgte, består af tre dele, adskilt af _ (underscore):

- Første del beskriver den metode eller property, der testes.
- Anden del beskriver inputtet til metoden eller propertien.
- Tredje del beskriver det forventede udfald af testen.

Navngivningen er valgt på baggrund af, at gruppens medlemmer finder det lettere at se, hvor problemet ligger, hvis en eller flere tests fejler.

Et eksempel på en unit test kan ses på listing 16.2. Her testes metoden `ImportMembersFromXml()`, med input af et enkelt medlem, og med en forventning om, at det ene medlem bliver gemt. Først kaldes metoden `InitializeXmlStrings1()`, som genererer en tekststreng med indholdet af det, der ville stå i en XML–fil med ét medlem. Herefter kaldes `ImportMembersFromXml()` med det simulerede XML–data. Endeligt benyttes NSubstitute til at verificere, at `Create()`–metoden i det angivne DAL blev kaldt én gang, med et `SailClubMember` objekt.

```
1   [Test]
2   public void ImportMembersFromXml_InputtingOneMember_OneMemberSaved()
3   {
4     this.InitializeXmlStrings1();
5     _parser.ImportMembersFromXml(_stream);
6
7     _memberDalSubstitute.Received(1).Create(Arg.Any<SailClubMember>());
8   }
```

Listing 16.2. Eksempel på Unit test

16.2 Brugertest

I dette afsnit forklares fremgangsmåden for brugertests i forbindelse med programmets brugervenlighed. Derudover ses der nærmere på, hvordan resultaterne bruges, hvorfra der kan dannes et overblik over programmets funktionalitet.

For at teste programmet anvendes der, foruden unit tests i dette projekt, en række brugertests, som tager udgangspunkt i projektets UserStories, kapitel 8 på side 23. Testene er inspireret af user acceptance Test, men da det ikke har været muligt at finde nogle af Sejlklubben Sundets medlemmer til at teste, er det ikke en konkret user acceptance test. Testen består i at få personer, som indgår i den endelige brugers segment til at prøve programmet, og udføre nogle opgaver som simulerer den reelle brug af programmet i sejlklubben [Hambling and van Goethem 2013]. Denne type test lægger fokus på uafhængige brugeres meninger om programmets brugervenlighed. Testens form er inspireret af black-box testing, som også er blevet brugt i forbindelse med udviklingen af programmet, som normalt ellers udføres af udviklerne selv.

Da programmet er lavet med henblik på brug i en sejlklub, består målgruppen af mange forskellige befolkningssegmenter. Idet det er et computerprogram, er det eneste krav til målgruppen, at de er villige til at bruge en computer.

Ud fra programmets funktionaliteter dannes der nogle mål, som brugerne skal opnå igennem testen. Der defineres en række testscenarier med udgangspunkt i de mål, der blev sat for testen. Disse mål er:

- Følge sin undervisningsstatus.
- Se reservationer af både.
- Oprette reservation af både.
- Tilmelde sig begivenheder.
- Følge status på begivenheder og undervisning.
- Oprette logbog for fuldendt sejlads.

Ud fra disse mål er der skrevet en række korte opgaver, som berører bestemte funktionaliteter i programmet. Opgaverne kan findes i appendiks D på side 87.

Dette bilag vil blive vist til testpersoner af programmet, og de vil blive observeret under testen. Efter udførelsen af testen spørges der ind til testpersonens oplevelse af programmet med henblik på funktionalitet, brugergrænsefladens design og deres helhedsindtryk.

En opgave kunne eksempelvis være:

- Log ind på systemet med følgende brugeroplysninger:
  - Brugernavn: oskar
  - Kodeord: lauridsen
- Foretag en booking af båden: Anna til næste lørdag, med følgende andre oplysninger:
  - Starttidspunkt: 20-07-2014 13:37
  - Sluttidspunkt: 20-07-2014 20:42
  - Besætning: Jens Hansen, Anders And...

  - Kaptajn: Anders And
  - Formål: Eftermiddagssejlads
- Log ud af systemet igen

Samtidigt udfyldes et skema med tidsforbrug af hver enkelt opgave, samt kommentarer omkring brugerens færden i programmet.

I de følgende afsnit beskrives der, hvordan testens data vil blive behandlet.

16.2.1 Funktionalitet

I testområdet funktionalitet undersøges der hvorvidt opgaverne stillet er mulige at løse for testpersonerne. Dermed testes der på, om brugergrænsefladen hindrer programmets funktionalitet.

16.2.2 Effektivitet

Effektivitet anses som værende forholdet mellem en ekspert og en nybegynders tid, for at udføre en given opgave [Anton Sergeev 2014a]. Derfor tages der tid på udførelsen af, hver enkelt opgave som testpersonen udfører. Eksperten vil være et medlem af projektgruppen, som ikke var observatør i nogen af de andre testpersoners test.

16.2.3 Tilfredshed

Til sidst laves der en tilfredshedsundersøgelse for at få brugerens meninger omkring programmets design. Dette gøres igennem et spørgeskema. Undersøgelsen skal give et indtryk af, hvor godt brugeren synes programmet var i forhold til layout, placering og synlighed af de krævede elementer i programmet. Denne undersøgelse giver et indtryk af, hvor godt programmet er designet, og sværhedsgraden af forskellige funktioner. Spørgeskemaet kan ses i appendiks E på side 91. [Anton Sergeev 2014b]

16.3 Analyse af testresultater

Grundet tidspres, fra bl.a. problemer med databaseimplementationen, var der kun tid til at teste på fire personer, foruden eksperten. Personerne der udførte testene, bestod af elever på Aalborg Universitet, som ikke havde set eller på anden vis hørt om projektet før. Resultaterne kan ikke konkluderes til at give et endegyldigt svar på om programmet kan bruges i Sejlklubben Sundet, da testpersonerne ingen kendskab har til sejlklubben og dennes behov. For at få et mere brugbart resultat fra testen, skulle testpersonerne være en repræsentativ gruppe af medlemmer fra sejlklubben, eller på anden vis være involveret med en sejlklub. På denne måde, ville testpersonen have en bedre forståelse for kravene til programmets funktionalitet. Dog menes det, at testens resultater alligevel kan sige noget om brugen af programmet, både om navigering og de testede funktionaliteter.

16.3.1 TestData

På tabel 16.1 på modstående side kan der ses en tabel over tid brugt ved hver opgave for hver enkelt testperson sammenlignet med ekspertens tid.

Opgave	T1	T2	T3	T4	Gennemsnit	Ekspert	Forhold
1	00:15	00:32	00:24	00:31	00:25	00:15	1,7
2	04:42	04:00	02:04	02:27	03:18	01:35	2,1
3	01:08	02:13	01:02	01:10	01:23	00:30	2,8
4	00:49	00:32	00:32	00:55	00:42	00:29	1,4
5	00:29	00:24	00:27	00:30	00:27	00:13	2,1
6	01:16	00:57	00:57	01:02	01:03	00:50	1,3
7	02:00	00:35	00:30	00:40	00:56	00:14	4,0
8	02:02	04:20	04:57	02:10	03:22	00:56	3,6
9	00:32	00:26	00:19	00:42	00:29	00:10	3,0
SUM	11:56	13:59	11:12	10:07	11:48	05:12	2,3

Tabel 16.1. Tid brugt pr opgave under test. T[1-4] er udefrakommende testpersoner.
Forhold udregnes: $\frac{Gennemsnit}{Ekspert}$

Det lykkedes for testpersonerne at udføre alle opgaverne.

16.3.2 Diskussion af test data

Nogle opgaver var tydeligt sværere end andre for personer, som ikke havde kendskab til programmets brugergrænseflade. Særligt var opgave 2, 7, og 8 udfordrende.

Opgave 2 lød på at reservere en bestemt båd, med udleverede data. Den tiltænkte løsning af opgaven er:

1. Vælg tabben “Både”.
2. Vælg båden “Anastasia”.
3. Tryk på knappen “Reserver båden” (Dette åbner CreateBoatBookingWindow).
4. Udfyld felterne med det udleverede data.
5. Tryk på knappen “Gem reservation”.

T1 og T2 havde svært ved det 3. skridt i opgaven. Fælles var, at de havde brugt over to minutter på netop dette skridt. Efter udførelse af testen blev de bedt om at uddybe, hvad dette skyldtes. De mente begge, at der var for meget støj på tabben “Både”, og dette distraherede dem fra at finde den korrekte knap. T3 og T4 havde i kontrast ingen problemer med at gennemgå dette skridt, og brugte hver under fem sekunder på at finde den korrekte knap. Eksperten brugte væsentligt mindre tid end T1 og T2, men sammenlignet med T3 er forskellen betydeligt mindre.

Testpersonernes forslag til forbedring af funktionalitet i denne opgave:

- Ryd op i tabben “Både”.
- Herunder flyt alt logbogsrelateret til sin egen tab.
- Øg størrelsen på knappen “Reserver båden”.

I opgave 7 skulle testpersonerne forfremme et hold i sejlerskolen. Den tiltænkte løsning af opgaven er:

1. Vælg tabben “Undervisning”
2. Vælg holdet “MasterRace”
3. Tjek CheckBoxen “Rediger hold”
4. Tryk på knappen “Forfrem hold”
5. Tryk på knappen “Log ud”

T1 havde særlig svært ved regne ud, at man skal udføre det 3. skridt, før man kunne udføre det 4. skridt. T1 mente, at der var for mange elementer i tabben “Undervisning”, og at dette forvirrede i udførelsen af opgaven, da de fjernede opmærksomheden fra CheckBoxen i toppen af tabben.

Testpersonernes forslag til forbedring af funktionalitet i denne opgave:

- Flyt rediger hold funktionaliteten til sit eget vindue, ved tryk på en knap.
- Der er ingen respons fra programmet efter det 4. skridt, tilføj et popup vindue som giver besked.

I opgave 8 blev testpersonerne bedt om at udfylde logbogen for en fuldført tur, som brugeren, de var logget ind som, havde reserveret, og derfor skulle udføre logbogen for.

Den tiltænkte løsning af opgaven er:

1. Log ind
2. Marker sejlturen i det højre DataGrid på forsiden.
3. Tryk på knappen “Opret logbog”
4. Tryk på knappen “Ændre besætning”
5. Fjern “Kasper Eriksen”
6. Ændre den “Faktisk afgang” til 10:00
7. Tryk på “Ja” under “Er båden skadet?”
8. Angiv vejrforhold og skadesrapport
9. Tryk på knappen “Udfør”

Alle fire testpersoner havde svært ved skridt 2 og 3. Særligt T2 og T3 brugte lang tid på dette problem. Testpersonerne besøgte tabben “Både” for at udføre dette. Eksperten havde ingen problemer med denne opgave og gennemførte opgaven på under et minut.

Testpersonernes forslag til forbedring af funktionalitet i denne opgave:

- Giv tydelig besked og instrukser efter log in, om at en logbog mangler at blive udfyldt.
- Forøg størrelsen af den forklarende tekst på forsiden til hver af de to DataGrids.
- Øg afstanden fra velkomstbeskeden til de to DataGrids.
- Mindske højden af de to DataGrids, da de alligevel er hovedsageligt tomme.
- Dediker en tab til læsning og udfyldelse af logbøger.

16.3.3 Testpersonernes samlede vurdering

I spørgeskemaet blev testpersonerne bedt om at vurdere flere centrale skærmvinduer. De blev bedt om at vurdere, deres evne til at overskue hvert af de angivne skærmvinduers funktionaliteter. Dette skulle angives som en karakter fra et til fem, hvor et er meget uoverskueligt og fem er meget overskueligt. Dette kan ses i skemaet tabel 16.2.

Vindue	T1	T2	T3	T4	**Gennemsnit**
Forside	5	4	4	3	4,00
Undervisning	1	2	3	2	2,00
Medlemmer	4	5	4	2	3,75
Både	3	4	3	3	3,25
Begivenheder	4	4	3	2	3,25
Logbogsvinduer	3	2	2	4	2,75
CreateCrewWindow	5	5	4	3	4,25
CreateBoatBookingWindow	5	5	4	4	4,50
Samlet Vurdering	3,75	3,88	3,38	2,88	3,47

Tabel 16.2. Testpersonernes vurdering af centrale skærmvinduer

Kommentarer fra testpersonerne

Foruden tabel 16.2 argumenterede testpersonerne også for deres vurdering.

Forside: Generelt kunne testpersonerne lide formålet med forsiden, som er at give brugeren et hurtigt overblik efter login. Dog mente flere, at layoutet kunne have været udført bedre. Herunder større afstand fra velkomstteksten til de to DataGrids, større overskrifter ved hver DataGrid, og lave de to DataGrids mindre da de hovedsageligt er tomme.

Undervisning: Generel utilfredshed omkring denne UserControl. Specifikt var der klager vedrørende, brugen af CheckBoxes til at tjekke elevers fremmøde. Her mente flere, at CreateCrewWindow-vinduet kunne have været anvendt. Derudover tilføjede den højre side af vinduet, der havde et layout som lignede CreateCrewWindow-vinduet, forvirring. Der var også negative kommentarer omhandlende CheckBoxen "Rediger hold". Flere fandt denne unødvendig, mens en anden mente, at det indhold denne låste op for, burde have sit eget dedikerede popup vindue.

Medlemmer: Testpersonerne skulle ikke bruge dette vindue i opgaverne. Derfor fik de vist dette vindue efter testens gennemførsel, således de kunne kommentere tabbens indhold. Der var generelt positive kommentarer, dog mente T4, at den store mængde data virkede overvældende. T3 og T4 mente at søgebaren ikke var klart nok markeret, og foreslog at der kunne tilføjes et "forstørrelsesglas"–ikon.

Både: Flere testpersoner udtrykte, at denne UserControl var uoverskuelig, da der vises for meget information efter valg af en båd. T1, T3 og T4 forslog at flytte alt logbogsrelateret til sin egen tab. Derudover kunne størrelsen af skrift forstørres, således den ville være nemmere at læse.

Begivenheder: Kommentarer til denne tab var få, T3 foreslog at ændre overskriften "Agenda" til "Begivenheder", over ListBoxen. Flere mente at brugen af ListBoxen burde have været erstattet

med et DataGrid, som er anvendt mange andre steder i programmet.

Logbogsvinduerne: Kritikken af disse var primært rettet mod navigeringen til vinduet. Der var der uenighed imellem testpersonernes vurdering af vinduets design. Én mente at elementerne i det tospaltede layout var uoverskueligt, særligt pga. rækkefølgen af disse. En anden udtrykte positive tanker om layoutet.

CreateCrewWindow: Der var næsten kun positive kommentarer om dette vindue. Vinduet blev omtalt som værende overskueligt, dog mente de det venstre DataGrid burde være større. T3 foreslog, at dette kunne have samme størrelse som det til højre. Her blev søgefunktionen igen overset, og der blev igen foreslået tilføjelsen af et "forstørrelsesglas"–ikon.

CreateBoatBookingWindow: Der var meget få kritikpunkter til dette vindue. T4 mente at knapperne "Gem reservation" og "Annuller reservation" var for brede og burde være placeret således, de lignede dem fra Windows vinduer såsom "egenskaber". Dvs. hvor de er placeret nederst i højre hjørne, på linje med hinanden.

16.3.4 Konklusion af brugertest

Brugertesten har givet os en større indsigt i programmets layout. Målene for layoutet var minimalisme og konsistens, men dette er ikke opnået, grundet problemer i arbejdsprocessen, se afsnit 10.2 på side 27. Dette uddybes i kapitel 17 på side 61.

På baggrund af brugertesten etableres det, at nogle dele af brugergrænsefladen er bedre konstrueret end andre. De bedste vurderes til at være vinduerne: **CreateCrewWindow** og **CreateBoatBookingWindow**. Disse vinduer har begge en høj overskuelighed, dette er et af principperne fra minimalisme, altså at fjerne det unødvendige.

Ud fra testpersonernes kommentarer kunne de resterende vinduers og UserControls layout forbedres. Særligt skulle tabben "Undervisning" og tabben "Både" forbedres. Konkrete forslag til disse forbedringer er givet af testpersonerne, disse udtrykker projektgruppen enighed i.

Da projektgruppen har en begrænset mængde tid, bliver disse rettelser ikke implementeret.

Del C

Refleksion

17 Diskussion

I diskussionen vil alle elementer fra problemløsningen blive diskuteret, herunder i hvor høj grad produktkravene er opfyldt.

Medlemshåndtering og Brugerlogin

Et af kravene til produktet var medlemshåndtering, som er implementeret ved hjælp af klasserne Person, SailClubMember og StudentMember, da instanser af klasserne lagres i databasen; hvert med et unikt ID.

Medlemmerne udgør desuden grundlaget for login-systemet, da der herigennem kan bestemmes hvilken bruger der logger ind, og hvilke funktioner i programmet, de skal have lov til at tilgå. Denne løsning ses som værende tilstrækkelig, da det er muligt at registrere, hvem som er logget ind og dermed give dem adgang til de respektive funktionaliteter.

Bådreservation og logbog

Et vigtigt punkt i programmet var at sørge for, at man kunne oprette en reservation af en båd. Dette punkt er blevet gennemført, da der i programmet kan oprettes en reservation. Desuden er der indført logik, som sikrer at båden skal være ledig og ikke være skadet, før reservationen kan gennemføres. Yderligere er der til en reservation også knyttet en logbog, som skal udfyldes, efter sejladsen er gennemført. Udfyldelsen af logbøger blev beskrevet i interviewet med Jacob Nørbjerg, som værende besværligt i Sejlklubben Sundet, da deres logbøger skal udfyldes på papir. Det menes derfor, at det udviklede program kan gøre det lettere for klubben at holde styr på logbøger, ved at digitalisere dem, holde referencer til hvilken sejltur de hører til, sørge for at ikke alle kan udfylde logbøgerne og ikke mindst indeholde de krævede informationer. Dette anses for at være en god løsning, sammenlignet med at skulle finde den korrekte, fysiske logbog i klubhuset for den pågældende båd og derefter udfylde alle informationerne i hånden.

Undervisningsorganisering

Et andet vigtigt krav til programmet var at gøre det muligt at holde styr på undervisningen, både fra elevernes side og undervisernes side. I programmet er der blevet konstrueret en del, som specifikt tager sig af undervisningen, hvorfra underviserne kan notere hvilke hold og elever, der har gennemført specifikke opgaver. Tilsvarende kan eleverne se deres egne fremskridt, og

hvornår de har deres undervisningslektioner. Programmets undervisningsdel fungerer, dog er det specifikke pensum, som Sejlklubben Sundet selv står for at udvælge, ikke implementeret. Det menes, at skiftet fra det overfladiske pensum, som programmet er skabt med, til det faktiske pensum som Sejlklubben Sundet anvender, ville være relativt simpelt. Derfor kan programmet også bruges i andre sejlklubber med undervisning, som anvender et andet pensum. Det kunne dog udvides, ved også at holde styr på antallet af gange en elev har udført en given opgave, og ikke kun notere om eleven har udført opgaven før.

Begivenhedsadministration og visning

Begivenheder findes også på listen af produktkrav. Funktionaliteten for begivenhedshåndtering er blevet implementeret, og det er muligt for medlemmer at tilmelde sig begivenheder og som administrator oprette, redigere og slette begivenheder. Desuden kan en bruger, når de logger ind på systemet, se kommende begivenheder og hvilke andre medlemmer der er tilmeldt, hvis begivenheden kræver tilmelding. Dette ses som en god løsning, da medlemmer hurtigt kan få et overblik over kommende begivenheder i klubben. Dog er det ikke muligt kun at se hvilke begivenheder man selv er tilmeldt, hvilket anses for at være en mangel.

Persistenslag

Der var valgt at bruge et persistenslag til lagring af data. Først faldt valget på Entity Framework. Grundet problemer beskrevet i afsnit 14.3.2 på side 41 blev der skiftet til SQLite, som også endte med at give problemer. Hvis man tager tidsforbruget på udviklingen af persistenslagene i betragtning, ville en simplere løsning have været bedre, da der så havde været mere tid til at udvikle og teste resten af programmet. Sådan en løsning kunne have været med flade filer eller et program helt uden persistenslag, som udelukkende kørte ved brug af Mockdata.

Da der var foretaget flere databaseskift under udviklingen var det et godt valg at bruge et Data Abstraction Layer, da dette simplificerede skiftet. Uden et DAL ville dette skift have været besværligt, da ethvert databasekald skulle skrives om.

Brugergrænseflade

Ved problemløsningens start blev det besluttet, at brugergrænsefladen skulle tilstræbe principperne minimalisme og konsistens i layout. Grundet den spredte arbejdsproces, som blev beskrevet i afsnit 10.2 på side 27, blev disse principper ikke overholdt, da gruppemedlemmernes forståelse for implementationen af principperne ikke stemte overens med hinanden. Dette har haft en negativ indvirkning på programmets layout, da skærmvinduerne ikke er konsistente. Ved brugertest kom dette bl.a. til udtryk, da testpersonerne havde svært ved at finde rundt i enkelte områder af programmet og ikke ved andre. Testpersonerne bestod ikke af sejlklubmedlemmer, men det menes stadig deres testresultater og feedback kan bruges til at forbedre programmet. Dette konkluderes, da Sejlklubben Sundets medlemmer består af et bredt befolkningssegment, som også inkluderer testpersonerne. Testen viste, at der var visse mangler ved programmets layout, dog ytrede testpersonerne generelt tilfredshed med programmet. Det vurderes derfor, at der stadig skal laves ændringer til programmets brugergrænseflade i tilfælde af en publicering i kommerciel kontekst.

Modeldesign

I det udviklede modellag var der flere designfejl som vil blive uddybet nedenunder. Da der ikke var tid til at refaktorisere programmet, blev disse designfejl ikke udbedret.

- SailTrip- og RegularTripklassen
- Lectureklassen
- StudentMember booleans

SailTrip- og RegularTripklassen: Disse to klasser burde have været samlet, da indtil flere felter er placeret på subklassen i stedet for superklassen.

Lectureklassen: Denne klasse burde have en relation til SailTripklassen.

StudentMember booleans: Disse booleans burde have været af typen `int`, da elever godt kunne have lavet den samme øvelse flere gange.

Opsummering

Det er altså muligt for Sejlklubben Sundet at benytte programmet på nuværende tidspunkt, men det anbefales at testpersonernes forslag til ændringer bliver implementeret, før det tages i brug. Løsningen kan anvendes af andre sejlklubber med sejlerskole og bådudlån, som ikke har brug for flere funktionaliteter end de udviklede, efter de nødvendige rettelser er implementeret.

Det vurderes at det vil være svært at bruge programmet til andre formål end sejlklubsrelaterede. Det er grundet den tætte forbindelse mellem modellerne og selve brugergrænsefladens Code-Behind. Derfor kan programmet ikke ses som værende mulig at anvende i andre former for fritidsklubber.

Sammenlignes problemformuleringens målsætninger med den færdige løsning, kan der argumenteres for, at der er blevet udviklet et brugbart management system. Dette konkluderes, da systemet digitaliserer en stor del af den dokumenthåndtering, som foregår i klubben, såsom undervisningsfremskridt for hver elev, booking af både, udfyldelse af logbøger, oprettelser af begivenheder og fremvisning af begivenheder. Programmet er i stand til at håndtere dokumentationen, som påkræves af Sejlklubben Sundet vedrørende sejlture og undervisning. Desuden verificerer programmet brugerinput, og dette kan mindske fejlraten på informationerne.

Konklusion 18

I det initierende problem blev der undersøgt, hvorvidt der skulle laves en softwareløsning, som kan hjælpe frivillige i fritidsklubber til administration, og som er let at benytte. Der blev i afsnittet Fritidsklubber afgrænset til at arbejde med sejlklubber, fordi det blev vurderet, at hvis det var muligt at udarbejde et program til en enkelt type fritidsklub, kunne et lignende program også udvikles til andre fritidsklubber. Der blev i organisationsanalysen yderligere afgrænset til at arbejde med Sejlklubben Sundet, da der var flere informationer vedrørende Sejlklubben Sundet end andre sejlklubber. I organisationsafsnittet blev der fundet frem til hvilke funktioner, Sejlklubben Sundet søgte i et program. De centrale krav værende reservering af både, sejlerskole organisering og håndtering af logbøger. I teknologianalysen blev der fundet frem til, at der eksisterer softwareløsninger, som kan dække nogle af Sejlklubben Sundets behov, dog blev disse ikke benyttet, muligvis fordi ingen af dem havde undervisningsfunktionaliteter. Dette resulterede i en problemformulering, som fokuserede på at udarbejde en softwareløsning, som kunne dække Sejlklubben Sundets behov. Efter udviklingen af programmet blev der opstillet en række tests for at teste programmets brugervenlighed, samt for at se om programmet opfyldte de stillede krav.

På trods af, at løsningen ikke er optimal, konkluderes det hermed, at den er acceptabel. Den endelige vurdering af programmets brugbarhed kan dog kun gives af Sejlklubben Sundet. Dette ændrer ikke på, at løsningens funktionaliteter opfylder de opstillede krav fra analysen.

Perspektivering 19

I dette afsnit vil der gives en beskrivelse af forskellige funktioner, der er blevet vurderet nyttige, hvis projektet skulle videreudvikles. Der vil desuden blive diskuteret generelle overvejelser af programmet, som kunne have været udformet anderledes.

SMS og E-mail

Det har i gruppen været diskuteret, at en SMS-service ville være smart at bruge. Denne service kunne bruges til at sende notifikationer på dagen eller dagen før et medlems reservation af en båd. Det kunne også tænkes, at hvis man vil reservere en båd i et tidsrum, hvor båden allerede er reserveret, kan man placere sig selv i kø til båden. Det betyder, at hvis medlemmet, der har den reelle reservering, pludselig aflyser vil den næste i køen få en sms om, at båden nu er fri. Her kunne der evt. bruges en svarmulighed, så medlemmet, ved at svare på SMS'en, kan meddele om de vil overtage reserveringen eller ej, dvs. uden at skulle åbne systemet. Et lignende system kan konstrueres ved e-mail i stedet for SMS.

Online tilgang

Som programmet er nu, fungerer det ikke online, og det vil sige, at man skal ned i sejlklubben for at kunne reservere en båd, tilmelde sig begivenheder og generelt bruge programmet. En måde at ændre dette på er at lægge databaseforbindelsen online, så der kan laves ændringer i databasen uden at logge på i sejlklubben. En online løsning vil gøre brugen af programmet mere fleksibelt, samt hjælpe medlemmerne og gøre brugen af klubbens faciliteter lettere at tilgå.

Andre fritidsklubber

Som tidligere nævnt kan et lignende program designes så det kan anvendes i andre fritidsklubber. Et eksempel kunne være et sportscenter, som vil gøre det muligt at reservere hallen til diverse arrangementer, melde sig til begivenheder der måtte foregå o.l. Dog vil dette, som nævnt tidligere, kræve ændringer i programmets funktionaliteter, da løsningen fremstillet til dette projekt, er lavet specifikt til Sejlklubben Sundet.

Website

Der blev i starten af projektperioden diskuteret internt i gruppen, at et system som dette ville tjene sig bedre som værende et website. Grundet manglende kendskab til at udvikle et website ved hjælp af C#, blev dette dog fravalgt og en skrivebordsapplikation blev udviklet i stedet. Ved en løsning udviklet som et website, ville det være muligt at anvende mobile enheder, såsom smartphones og tablets, samt computere med andre styresystemer end Windows, hvilket anses for at være en fordel.

XML–import

Programmet har mulighed for at importere medlemmer fra en XML–fil. Det vil være muligt at eksportere medlemmer fra en Microsoft Access database, såsom den Sejlklubben Sundet bruger. Denne importfunktion skal tilpasses den konkrete database for at være brugbar.

Projektets analyse har belyst, at der ikke findes noget produkt på markedet, som opfylder de behov, som Jacob Nørbjerg har givet udtryk for under interviewet. Det viser sig dermed at være en mangelvare. Dermed kan vores analyse for området hjælpe fremtidig udvikling af management systemer med det fokusområde.

Del D

Referencer

Litteratur

Aalborg Sejlklub. Sejlerskole aalborg sejlklub, 2014a. URL http://www.aalborg-sejlklub.dk/events_category.php?id=20. *Side 12*

Aalborg Sejlklub. Vedtægter aalborg sejlklub, 2014b. URL http://aalborg-sejlklub.dk/show_event.php?id=232. *Side 11*

agilemanifesto.org. Agile software development manifest, 2014. URL http://agilemanifesto.org/iso/dk/principles.html. *Side 27*

Anderson Software. Boatcloud, 2014. URL http://www.boatcloud.com/clubhub.htm. *Side 15*

Anton Sergeev. Gui effeciency, 2014a. URL http://ui-designer.net/usability/efficiency.htm. *Side 54*

Anton Sergeev. Gui satisfaction, 2014b. URL http://ui-designer.net/usability/satisfaction.htm. *Side 54*

Bådklubben Valby. Love for bådklubben valby, 2014a. URL http://xn--bdklubbenvalby-lib.dk/index.php/love-og-regler. *Side 11*

Bådklubben Valby. Duelighedsbevis bådklubben valby, 2014b. URL http://xn--bdklubbenvalby-lib.dk/index.php/duelighedsbevis. *Side 12*

Center for frivilligt socialt arbejde. Frivilligrapporten, 2012. URL http://www.frivillighed.dk/Webnodes/da/Web/Public/Viden+%26+information/Frivilligrapporten. *Side 1*

Dansk Sejlunion. Sejlsport.dk, 2014. URL http://www.sejlsport.dk/mere/dansk-sejlunion/strategi-og-politik/vision-og-vaerdier og http://www.sejlsport.dk/mere/dansk-sejlunion/historie. *Side 9*

developer.android.com. android.database.sqlite, 2014. URL http://developer.android.com/reference/android/database/sqlite/package-summary.html. *Side 40*

Brian Hambling and Pauline van Goethem. *User Acceptance Testing - A step-by-step guide.* BCS Learning & Development Limited, 2013. ISBN 9781780171678. *Side 53*

Holstebro Ungdomsskole. Vinderup ungdomsskole, 2013. URL http://www.hk-ung.dk/vinderup/. *Side 4*

Jonathan Allen. A wpf q&a, 2014. URL http://www.infoq.com/news/2014/04/WPF-QA. *Side 30*

Kurt Nørmark. P2 projektkatalog, 2014. URL https://drive.google.com/file/d/0B3jEBF-Oqi0bV1c0bkpOWEtzWk0/edit?usp=sharing. *Side 1*

Kenneth C. Laudon and Jane P. Laudon. *Information Systems and the Internet: A Problem-Solving Approach*. The Dryden Press, Fort Worth, 4th edition, 1999. *Side 5*, *Side 5*

Microsoft. Introduction to wpf, 2014. URL http://msdn.microsoft.com/en-us/library/aa970268(v=vs.110).aspx. *Side 30*

SailingClubManager. Sailing club manager, 2014. URL http://www.sailingclubmanager.com. *Side 16*

Sejlklubben Sundet. Vedtægter for sejlklubben sundet københavn, November 2012. URL http://www.sundet.dk/vedtaegter/Vedtaegter%20for%20Sundet%2027nov12.pdf. *Side 11*

Sejlklubben Sundet. Sejlklubben sundet, 2014a. URL http://sundet.dk. *Side 12*

Sejlklubben Sundet. Sejlklubben sundet - udlån, 2014b. URL http://sundet.dk/baadudlaan/baadlaan.htm. *Side 12*

Seonghoon Kang and Won Kim. Minimalist and intuitive user interface design guidelines for consumer electronics devices, 2007. URL http://www.jot.fm/issues/issue_2007_03/column5/. *Side 29*

Spøttrup Kulturhal. Spøttrup kulturhal, 2014. URL http://www.spottrupkulturhal.dk/. *Side 4*

sqlite.org. Sqlite, 2014a. URL http://sqlite.org/. *Side 40*

sqlite.org. Sqlite — most widely deployed sql database, 2014b. URL http://sqlite.org/mostdeployed.html. *Side 40*

Supersaas.dk. Sundets Booking System, 2014. URL http://www.supersaas.dk/schedule/Sundet/Båd_booking. *Side 12*

w3schools.com. Browser display statistics, 2014. URL http://www.w3schools.com/browsers/browsers_display.asp. *Side 43*

Wikipedia. Entity framework, 2014a. URL http://en.wikipedia.org/wiki/Entity_Framework. *Side 39*

Wikipedia. Agile software development manifest, 2014b. URL http://en.wikipedia.org/wiki/Object-relational_mapping. *Side 39*

wpf-tutorial.com. Wpf vs. winforms, 2014. URL http://www.wpf-tutorial.com/about-wpf/wpf-vs-winforms/. *Side 30*

Figurer

Tabeller

Akronymer

AAU Aalborg Universitet

DAL Data Abstraction Layer

EF Entity Framework

GUI grafisk brugergrænseflade

ORM Object Relational Mapping

RDBMS Relational Database Management System

SQL Structured Query Language

UML Unified Modeling Language

WPF Windows Presentation Foundation

XAML Extensible Application Markup Language

Del E

Appendiks

Informationer påkrævet af sejlklubben Sundet A

Dette bilag har til formål, at beskrive de informationer Sejlklubben Sundet påkræver af deres medlemmer. Herudover er der også beskrevet, hvilke informationer Sejlkubben Sundets sejlerskole skal bruge om deres elever for at kunne lade dem bestå.

A.1 Data på sejladser og medlemmer

Når en skolesejlads skal foretages, skal følgende informationer skrives ned i skoleprotokollen:

- Navne på underviseren og eleverne
- Ugedag
- Dato
- Fremmøde, i form af X for mødt, A for afbud
- Navne på gæster
- Afgang og ankomsttider
- Vindforhold
- Evt. Kommentarer, normalt henvisninger til skadesrapporterne, hvis skader er opstået.

Alle disse informationer skrives for hver lektion og arkiveres i klubhuset, dvs. det er ikke elektronisk.

Når der sejles uden for skolen, dvs. fornøjelsesture eller kapsejladser, skal lignende information udfyldes. Disse informationer ses her:

- Bådnavn/id
- Navn og telefonnummer på føreren af båden.
- Navne på alle i besætningen, evt. telefonnummer.
- Startid og dato
- Forventet ankomstdato og tid
- Formål med turen med indikation af område for sejladsen.
- Reel ankomstdato og tid
- Vind og vejrforhold
- Kommentarer

Hvis der sker skader på en båd under en sejlads, skrives disse ned i skadesrapporten. Føreren skal også slå op i denne bog før en sejllads, for at sikre at man er klar over eventuelle skader på båden. I skadesrapporten noteres flg. oplysninger:

- Dato
- Tekst der beskriver uheldet, eller skaden.
- Her kan også skrives evt. svar fra bådchefen(personen der sørger for båden).

Til at holde styr på medlemmernes oplysninger, bruger Sundet et Microsoft Access–baseret system. Her er der personlige oplysninger på eleverne og medlemmerne, desuden kan der printes girokort ud til håndtering af kontingenterne. Det bruges også til at holde styr på bådene i havnen, som er ejet af nogle af klubbens medlemmer, samt deres lokationer. Dette system kan kun tilgås lokalt på computeren lokaliseret i klubhuset.

A.2 Informationer og andre events

Når Sundet skal give informationer ud til medlemmerne, sættes informationerne op på en opslagstavle. Disse informationer kan f.eks. være tilmeldinger til diverse begivenheder såsom 24-timers sejladser eller onsdagsmatcher. Det vil sige, for at kunne tilmelde sig begivenheder i klubben, skal man skrive sig på det bestemte tilmeldingsskema man finder i klubhuset. Det er ikke sikkert, at medlemmer opdager tilmeldingsfristerne på diverse begivenheder, hvilket kan resultere i et lavere fremmøde, og mindre aktivitet i klubben.

Interview spørgsmål B

I dette afsnit ses spørgsmålene fra interviewet med Jacob Nørbjerg.

- Hvordan administrerer I sejlklubben lige nu (medlemmer, fartøjer og lign.)?
- Hvilken information har I på Jeres medlemmer (kvalifikation, niveau, erfaring eller lign)?
- Hvor mange fartøjer har klubben til rådighed (sejlbåde, joller, ynglinge mm.)?
- Benytter I et EDB system til at administrere klubbens egne fartøjer?
  - Hvis ja, hvilke egenskaber har systemet og hvordan kunne det forbedres?
  - Hvis nej, hvordan administrerer I så fartøjer, og er I tilfredse med denne metode?
- Hvad er Jeres største problem mht. administration af fartøjer?
- Hvor meget tid anvendes der på administration af fartøjer (regninger, planlægning, vedligeholdelse osv.)?
- Har I selv en vision om, hvordan et system kan gøre det lettere for Jer at styre ting i klubben (medlemmer, kontingenter, arrangementer, fartøjer osv.)?
- Kan medlemmer leje/låne jeres fartøjer(joller, sejlbåde m.m.) når de ikke benyttes af sejlerskolen?
  - Hvis nej, hvorfor tilbydes denne funktion ikke, er det noget der kunne tilbydes, hvis der var et system til at hjælpe med administration?
  - Hvis ja har vi et par ekstra spørgsmål herom:
    * Hvad er kravene for at låne et fartøj?
    * Hvordan håndteres omkostninger ved udlån af et fartøj?
    * Hvordan administreres udlånte fartøjer og kunne hertil benyttes et system til hjælp? I så fald hvordan kunne et system bedst muligt være til hjælp?

Interview transskribering C

Jacob: I skal huske at det her er baseret på muligvis forældet viden.

Alle: Jaja.

Jacob: Hvad mener I med at administrere i sejlklubben? Det er jo et bredt spørgsmål.

Marc: Det er primært, hvordan I holder styr på de ting, som ligger under sig.

Jacob: Vi har et hjemmestrikket medlemssystem. Det var jo dengang ikk'. Der ligger alt det sædvanlige ikk': Navn, adresse, telefonnummer. Så ligger der fuldt medlemsnummer, og så ligger der ens uddannelse, altså sejlerskole, hvor langt man er kommet i forløbet på sejlerskolen, og om man har førerbrevet, duelighedsbevis. Så ligger der, om man er bådeejer, det koster nemlig flere penge. Der ligger vist også noget om landpladsadministration, alt hvad der hedder vand, plads i vandet, der organiserer havnen. Plads på land det arrangerer klubben. De klubber har forskellige landområder vi disponerer over. Det er så det generelle, det vi har på medlemmerne. Så kan man, f.eks. hvis man som medlem har været... Hvis man skal betale for et eller andet, hvis man har lånt en båd og skal betale for det, så er der også et skærmbillede til det. Jeg kan ikke lige huske, om man klikker direkte fra medlem, eller man skal indføre medlemsnummeret der, men der er et sted, hvor man kan gå ind og sige: "Her er en der skal betale noget. Han skal betale for så og så mange dages leje af en båd og en motor osv. osv., og så printer vi et girokort ud." Det samme typisk skærmbillede bruger vi til: "En som skal betale for en landplads og hans båd fylder så og så mange kvadratmeter dut [efterligner klikkelyd]". Så kan man skrive ind her, som jeg gjorde i weekenden, lånte en af klubbens slibemaskiner med støvsuger og det koster så og så meget pr. dag, så meget af det ligger i et pr. Access–baseret tudsegammelt hjemmestrikket system, som ligger på computeren nede i klubben.

Søren: Ligger det kun lokalt eller hvad?

Jacob: Ja, det ligger kun der lokalt, og der er forskellige record udtrækningsmuligheder os' ikk': Gi' mig en liste over alle elever på andet år, gi' mig en liste over alle bådejere eller gi' mig en liste over hvilken rækkefølge både skal sættes op i og sådan noget, så det er der også. Så har vi selvfølgelig senere, men det kan i se på nettet, så har vi jo indført forskellige... Altså meget af det information som vi kan trække, det bliver så på forskellig vis smidt op på nettet. Det jeg sagde lige før med liste over hvornår både skal op den ligger på nettet, og den kan alle og enhver gå ind og se, så i kan gå ind og se hvad min båd hedder og hvornår den skal i vandet. Det ligger bare som en PDF. Nå, hvad har i på jeres medlemmer? [oplæsning af spørgsmål, spørgsmål var

givet på forhånd]. Det er sagt, altså vi har typisk om de har eller ikke har førerbrevet, om hvor langt de er kommet i skolen, hvis vi altså husker at vedligeholde det, det er ikke altid vi gør det. Det er vel sådan cirka det.

Hvor mange både har vi til rådighed? [oplæsning af spørgsmål]

Ja det svinger meget. Den gang jeg var skolechef der havde vi to gaffelriggere, tre drabanter og en spækhugger. Nu er konfigurationen ved at ændre sig lidt så nu hedder det stadig to gaffelriggere og så hedder det vidst nok to eller tre spækhuggere, er de ved at lave det om til og så nogle nye som hedder J80, sådan nogle satans små flyvepap. Jeg tror de unge mennesker kommer til at få nogle forskrækkelser når de skal ud og sejle dem første gang hvis de ikke har sejlet før, men det er jo deres problem. Det er jo en politisk diskussion. Og så har vi en enkelt yngling. Det var en som blev til overs engang, den er bare blevet foræret til klubben. Den er til at læren til at lege i og så har faktisk en fire... men de er ikke rigtig taget med der, vi har en fire til seks Mini12'er, hedder de. Det er til handicapafdelingen. De ligner gammeldags Americas Cup både som de så ud den gang de lignede sejlbåde, skaleret ned i en to-tre meters længde, så kan man sidde nede i dem og styre med hænderne eller fødderne eller hvad man har at bruge, så den bruger handicap-afdelingen til når handicappede er ude og sejle.

Et EDB-system? [oplæsning af spørgsmål]

Ja, i den forstand at vi har overblik over hvilke både vi har. Så er der kommet det der online reservationssystem, men den situation jeg kendte, som jeg tror stadig ligger bagved, fordi der kan man kun reservere en båd, det er jo at der inde i skolestuen, dvs. det er der hvor der ligger redningsveste og grej til alle bådene og sådan, der ligger der en stor bog hvor man går ind og skriver "3. maj, sådan og sådan, Jacob, besætning, telefonnummer på Jacob, hvornår sejler vi, hvornår tror vi at vi kommer hjem, hvor tror vi nok vi tager hen" og så et eller andet sted, en kolonne som hedder: Det var her vi kom hjem, og hvis der er nogen kommentar, så det er sådan en stor bog af den tykke der, den fylder sådan her [viser hvor står bogen er]. Så er der på hver båd en havariprotokol, dvs. at hvis der er et eller andet med båden, så tager man den rigtige bog frem og skriver: "Jeg var ude og sejle med båden og jeg smadrede ind i Oslobåden" eller et eller andet, der var noget som knækkede, sejlet revnede eller hvad det nu var og hvis man ikke kan reparere det selv så skriver man det der. Hvis man føler eller ved at det er noget som bare skal repareres i en helveds fart, så er man stærkt opfodret til at ringe til skolechefen eller den der har ansvar for båden. Hver båd har en bådchef der er ansvarlig på den båd, så hvis man kan se at det her kræver mere end jeg lige kan klare, det kræver noget værktøj, reservedele, det kræver måske syn af en bådbygger, så skal man så ringe til den pågældende person som så vil sætte det i værk. Hvis man kan reparere det selv så prøver man også på at gøre det, men bare lige skriv et notat om at der var noget eller hvis det er noget som skal laves, men det kan godt sejle uden det laves så skriv det lige.

Så er der skolen: Hver båd hver aften har sådan et fortrykt ark: Fører, elever, sejladsnummer, dato, afkrydsning, hvem og hvad. . Ud fra hver, mulighed for bemærkning: Skete der et eller andet, hvad gjorde vi, er der nogen som ikke dukker op, er der gæster med så bliver de også bare skrevet på der nedenunder den faste liste, og hvis det er en anden fører skriver man også det på, så krydser man af den aften det handlede om og det bruget man tildels til at holde øje med hvem der kommer, hvem der ikke kommer, hvem som husker at melde afbud, og glemmer det, det er fyfy at glemme at melde afbud, og tælle op antal gange man har sejlet, det er jo også

vigtigt, altså i forhold til proportion på skolen, det var sejladsprotokollen til skolen.

Så har vi selvfølgelig reservationssystemet, som flytter lidt der, men der er stadig sådan et andet slags reservationssystem som handler om når skolen arrangerer noget. Når skolen siger: "Nu på første weekend i juni der er der 24-timers sejlads, det er en skide god oplevelse. Er der nogen elever der gerne vil være med?" Så hænger der gerne sådan et stykke papir nede i gangen hvor der står 24-timer sejlads, her kan førerne skrive sig på og her kan eleverne skrive sig på og så må man håbe at der er nogen som skriver sig på. Så det er sådan en: "Her er et tilbud om en ekstra tur af en eller anden slags, skriv jer lige på her." Det foregår også på papir. Så det kræver altid at man kommer der ned og gør noget, skrive sig på, sørge for overblik. Så det der *pullik 9:11* der hedder "Hvad er jeres største problem", det største problem jeg oplevede var simpelthen det her med at holde styr på... Som administrativ ansvarlig, så er der bare en fandens masse papir som man skal holde styr på. Man skal ind og tælle op på den der mødeliste, man skal... Den gang skulle man hvis nogen ringede ned og sagde kunne jeg reservere en båd på det og det tidspunkt, så skulle man ud og finde... Vi havde gerne sådan en kalender hængende nede på opslagstavlen med bådene og datoerne, så kunne man så blive krydset af, enten gjorde folk det selv eller så bad de os om at gøre det. Havde anden kalender bl.a. seddel som hang der med reservation til onsdags-kap-sejlads, sådan at folk "Jeg vil gerne ud og sejle onsdags-kap-sejladsen i næste uge" så skriver man sig lige på, "jeg har reserveret båden" og den der første når frem med en kuglepen har vundet og enten skriver man "jeg har besætning" eller "jeg har plads til to elever" eller hvad det nu er og så må folk finde ud af det på den måde. Det kræver igen at man kommer ned og kigger på opslagstavlen og forstår det der. Der var en regel omkring onsdagsmatch med at man kun kan reservere en uge ud i forvejen. Det der med at man bare lige, nogen af de gamle, der er altid nogen som sejler onsdagsmatch rundt i vores både de tager blokreservation hen over hele sommeren, nænæ. Man kan reservere en uge i forvejen. Det er jo en politik, som man har.

Det er meget papirarbejde. Når man skal finde ud af hvor meget der skal betales, så skal man ind i den store [bog] og så skal man ind og se hvem har egentlig lejet en båd og så gå igennem, og så skal man jo prøve på... Typisk så gør man det måske tre-fire gange i løbet af en sommer, maks. Dvs. at man skal finde alle de gange hvor den person har haft fat i båden, og hvad er det: En aften, en dag, en weekend, hvad fanden er det for noget?

Marc: Krydsrefereres om der er elever med?

Jacob: Ja det kan man... Det er vel den nye politik, så skal man også finde ud af om der er elever med. Det tror jeg nok vi havde en... [politik]. Så indførte vi en da det kom så indførte vi med at man lige kunne lave et kryds på besætningen og sige at man har en elev med eller skrive et sted at man har en elev med, så skal man også lige finde ud af det. Så gik vi ikke mere ned i det end at vi stolede på, at hvis man skrev at man havde en elev med, så stolede vi på det. [eksempel:] Nå det var så Anette, hun har jo sejlet der, nå er han blevet stillet ind: fint. Og så gå igennem sådan 2-3 sider, der bliver hurtigt fuldt op på sådan en side så, fandens besvær, derfor gad vi heller ikke gøre det til sidst. Og så blev der skrevet girokort ud som blev puttet i en kurvert og sendt til den pågældende. Nogen af os kunne godt finde ud af at fiske girokortet over i en PDF og så bare få det sendt på mail, men som regel blev det noget med en kuvert. "Ikke nogen elektronisk opkræven her."

Hvis vi går sådan lidt længere ud... Så omkring det hele med skolen, men det er nok lidt

voldsomt, der synes jeg at det havde været et kæmpe puslespil det med at få styr på hvilke elever skal være hvor og hvilke dage og hvordan får vi placeret dem på både osv. Men problemet var ofte lige så stort... Skyldtes lige så meget at folk bare ikke fik meldt tilbage når vi skrev: Nu må i godt lige få sendt jeres ønsker. Når vi skriver til folk: "Nu skal vi lige have af vide: Om du har tænkt dig at sejle til sommer, om du sidder over et år eller hvad du gør, hvilke dage du kan sejle." Og hvis de ikke kommer så sidder man bare der... Man kan ikke lave planen. Det synes jeg at vi brugte meget tid på, vi havde ikke rigtig noget støtte til at lave den der [plan], det blev sådan et Excel-regneark til sidst, med båd og elever og så blev det bare hængt op på opslagstavlen: Sådan her ser det ud. Det var sådan lidt... Det virkede. Som elev og som medlem synes jeg at det største var det besværlige med man kan bare ikke få overblik over en skid hjemmefra. Altså hvilke både er ledige? Det kan så se med det nye system som de har lavet, men jeg kunne virkelig godt tænke mig at sejle 24-timers sejlads. Der skal man altså have fat i *et eller andet 13:58* nede i Sundet og bede om: Er den der indkaldelse kommet op og hænge. Jeg vil gerne ud og sejle onsdagsmatch, der kan man så gå igen. Jeg vil gerne faktisk finde ud af om der er en ledig plads på en onsdagsmatchbåd. Altså, en ting er om den er reserveret, en anden ting er om der lige er en er en... "Kan jeg lige springe på som besætning her." Så den der manglende afgang til information når man sidder der hjemme og gerne vil et eller andet og så det der med at det hele er papirbaseret. Skulle gå igennem en stor protokol for at finde ud af hvad folk skal betale for et eller andet, det må da kunne gøres smartere. Min vision var den gang noget integration mellem de forskellige ting, mellem brug af både til skole, til fritidsbrug, til kapsejlads, til skolearrangeret fritidsbrug. Altså der er forskel på at jeg beslutter tage ud søndag og sejle med nogle venner og så at skolen siger at den søndag er der et arrangement som skolen gerne vil ha' at kommer ud, så der laver skolen nærmest en *for-et-eller-andet 15:17*, som så først bliver oplyst måske tre dage før at man finder ud af at der kommer noget, så bliver båden frigivet. Så der er det igen med at man som medlem kan gå ind og se: Kan man komme til der. For det er forskellig måder at reservere og bruge en båd på som godt kan være lidt indviklet at finde rundt i.

Som medlem kan man leje og låne et fartøj? [oplæsning af spørgsmål]

Ja det kan de sagtens. Det ved i godt. Kravene er førerbeviset. Omkostninger: Vi sender et girokort. Administreres: Ja det gør de jo stortset ved at vi har de der papir, protokoller hvor man skriver sig i.

Søren: Det der førerbevis, det får man bare igennem skolen på de to år, som det tager eller er det kun duelighedsbeviset?

Jacob: Altså det tager to år på skolen, det er på den måde vi organiseret det på. Det er et valg man har truffet i klubberne der nede. Jamen det tager to år og faktisk har de tre klubber en lidt forskellig policy omkring det, men i Sundet har det været sådan, *der siges et eller andet 16:35*, det var sådan at det første år sejlede man med yoghurtbærger, undskyld plastikfiberbåde og det andet år sejlede man gaffelriggere. Og det ud fra sådan rent propertion, altså glasfiber, sådan en drabant der, den skal men for det første være rigtigt ond ved den før den vælter og for det andet så er den forholdsvis nem at sejle og man får meget klar og hurtig besked hvis man gør noget forkert, båden giver meget hurtig tilbagemelding. Gaffelriggeren er svær, den er tung, den er længere om at reagere, den gir' ikke så hurtig og klar besked om hvad der foregår, dvs. at man skal være mere erfaren for at forstå hvad den fortæller en. Så den prøver vi på ikke at skræmme førsteårseleverne i. Alene mængden at torvværk det ku' godt få folk til at blive bekymrede når de sådan skal finde ud af det. Set enkelt er det en fantastisk båd at undervise i, den har et

stort åbent...[plads] Man kan gå rundt nede i den. Man sidder ikke sådan klemt sammen på smalle bænke, man kan gå omkring, fedt. Så det tager to år. Så har vi, nu glemte jeg at sige før med Spækhuggeren, jeg sagde vi havde en Spækhugger, vi har skolen 2 år. Så har vi det der hedder... Det er så blevet en kapsejladsskole, sammen med de andre klubber i havnen, så folk der er blevet uddannet fra sejlklubberne, kan så tilmelde sig kapsejladsskolen, dvs., det er mest spækhugger vi gør det i, dvs. så kommer de... Så for det noget teoriundervisning og noget praksisundervisning i kapsejlads på forskellige niveauer kulminerende med at man så kan deltage i almindelig kapsejlads. Det er ikke noget man skal, lige så snart man har førerbeviset, så kan du gå ud og sejle kapsejlads, det er slet ikke det, men der er mange der godt vil have den der uddannelse, forstå hvordan man *et eller andet med start 18:33*, hvordan lægger man taktik, hvordan får man samarbejdet til at fungere, til kapsejlads skal det gå temmelig stærkt, så det har vi så sammen med de andre, det er så os klubben, så spækhuggeren, den ene og muligvis også den næste, er mere eller mindre fastreserveret til kapsejladsskolen mandag, tirsdag og onsdag tror jeg det er. Så det er jo også en anden fast ting, altså hvor vi har... Faktisk bliver de jo... Den protokol de udfylder er den samme som skoleprotokollen, den er fortrykt med navne på, afkrydsning, du er på dette kapsejladshold, på den båd om mandagen. Så må du krydse af når du kommer. Og det her med at vi hvem der er på båden det er jo i forhold til skolen, i forhold til skolens regler for om man er dygtig nok når man har gjort nok, og det er helt generelt at vi vil vide hvem som er på båden, altid, når den er ude og sejle, det er rent sikkerhedsmæssigt. Så det har det dobbelte formål.

Søren: Ja det var...

Jacob: Har jeg fået svaret på alle jeres spørgsmål? Jeg kørte bare der ud af.

Alle: Ja det tror jeg.

Brugertests af program D

Velkommen til testen af programmet McSnttt udviklet af Software gruppen SW2A305. Programmet er designet som et management system til en sejlklub. Før du påbegynder hver opgave, bedes du gennemlæse den, når du er klar så giv venligst os besked. Dette er således, vi kan tage tid på din udførelse af opgaverne, dog skal du ikke føle dig presset til at skynde dig.

D.1 Opgave 1

Log ind med følgende brugeroplysninger:
- Brugernavn: “marcus”
- Kodeord: “husmand”

Når du har gjort dette, bedes du tilmelde dig begivenheden “PizzaParty”, som finder sted den 20/6-14 klokken 19.00.

D.2 Opgave 2

Forbliv logget ind til denne opgave. Du bedes nu booke en båd med følgende oplysninger:

- Bådnavn: Anastasia
- Afgang: “20/6-14 klokken 09:00”
- Ankomst “20/6-14 klokken 17.00”
- Besætning:
  - Marcus Husmand
  - Lars Olsen
  - Bodil Kjær
  - Anne Frank
  - Kasper Eriksen
  - Anders And
- Kaptajn: Anders And
- Formål: “Vi skal ud og fiske på havet.”

Log herefter ud af programmet, så du er klar til næste opgave.

D.3 Opgave 3

Log ind med følgende brugeroplysninger:
- Brugernavn: “oskar”
- Kodeord: “lauridsen”

Dette medlem er administrator, da medlemmet er en af sejlerskolens undervisere. Du har lige været ude og undervise eleverne og skal nu registrere, at dit hold har fuldført et nyt punkt på deres liste over mål i forbindelse med uddannelsen.

Du bedes :

- Finde holdet: “Svenskerne”
- Vælg lektion: 05/09/2014 kl. 15:00
- Afkryds at alle medlemmer var mødt op, og alle på denne undervisning, lavede Drabant sejllads.
- Gem herefter lektionsinformationerne

D.4 Opgave 4

Forbliv logget ind til denne opgave.

Gør følgende:

- Sikre du har valgt holdet: “Svenskerne”
- Opret lektion for holdet der starter kl. 19.00 og slutter kl. 21.00 den 1. august 2014

Log nu ud af programmet.

D.5 Opgave 5

Log ind med følgende brugeroplysninger:
- Brugernavn: “michelle”
- Kodeord: “kristensen”

Dette medlem er en elev. Du er færdig med de to års sejlskole og vil være sikker på at du kan få dit bådførerbevis.

Tjek derfor at alle læringsområderne er tjekket af.

Log herefter ud af programmet.

D.6 Opgave 6

Log ind med følgende brugeroplysninger:
- Brugernavn: “oskar”
- Kodeord: “lauridsen”

Nu bedes du oprette en ny begivenhed.

- Begivenheden skal foregå den 7/6-14, klokken 21:00.
- Kald begivenheden “Bålfest ved molen”.
- Beskrivelsen kan være hvad som helst.
- Afkryds “tilmelding påkrævet”.
- Tilmeld dig herefter begivenheden.

D.7 Opgave 7

Forbliv logget ind til denne opgave.

Vi skal nu forfremme et hold i sejlerskolen.

- Vælg hold: “MasterRace”.
- Forfrem holdet og tildel dem deres bådførerbevis.
- Log herefter ud af programmet.

D.8 Opgave 8

Log ind med følgende brugeroplysninger:
- Brugernavn: “marcus”
- Kodeord: “husmand”

Du er nu kommet hjem fra en sejltur, på båden “Anna”, og skal udfylde din logbog for turen.

- Åben opret logbådsvinduet for turen som forgik d. 08-05-2014, med formålet: “Generobre Skåne”.
- Fjern besætningsmedlemmet “Kasper Eriksen”
- Ændre faktiske ankomst til: 09-05-2014 kl. 10:00
- Angiv båden som værende beskadiget.
- Angiv skadesrapporten som: “Masteræb delvist flænset”
- Angiv vejrforholdene som: “2 m/s fra vest”
- Tryk “Udfør”

Forbliv logget ind.

D.9 Opgave 9

Du bedes nu tjekke, at din logbog var udført korrekt. Du skal nu finde logbogen i systemet (Husk det var båden “Anna”).

Når du har tjekket, at du havde angivet vejrforholdene korrekt (“2 m/s fra vest”), er du færdig.

Log nu ud af programmet.

D.10 Testen er nu slut

Tak fordi du ville være med i vores test, vi har nogle opfølgende spørgsmål, som vil blive stillet til dig af observatøren fra vores gruppe.

Spørgeskema E

Spørgeskema for brugertests af programmet.

På en skala fra 1-5 hvor overskueligt synes du så, programmet var at navigere rundt i?

[1 er meget uoverskueligt, og 5 er meget overskueligt.]

- Vurdering:
- Hvorfor denne holdning?
- Hvilke elementer tænker du specifikt på?

På en skala fra 1-5, hvor let var det at overskue et skærmvindues funktionaliteter?

[1 er meget svært, og 5 er meget let.]

	Vurdering	Kommentar
Forside		
Undervisning		
Medlemmer		
Både		
Begivenheder		
Logbogsvinduer		
CreateCrewWindow		
CreateBoatBookingWindow		

Var der noget ved programmet, du ikke forstod?

Var der noget, der fungerede godt?

Hvad fungerede mindre godt?

Hvis du kunne ændre noget på programmets udseende eller navigering, hvad ville det så være?

Har du nogle forslag eller tanker om programmet?