

TEAM NORDIC

STOP MADSPILD



Håndtering af madspild hos erhvervsdrivende virksomheder

Afgangsprojekt 2020 af
Christian A. Nicolau

Copenhagen Business

Academy

Afgangsprojekt i Udvikling

AU I IT

Informationer			
Startdato	31-8-2020	Termin: Uge 50	
Slutdato	25-11-2020	Bedømmelsesform	Dansk 7-trinskala
Eksamensform	Mundtlig prøve	ECTS:	10
Intern bedømmer	Kristian Langborg-Hansen		
	Deltager		
Navn	Christian A. Nicolau		
Kandidatnr:	cn228		
CPHBusiness ID:	cph-cn228		
Vejleder/underviser	Kristian Langborg-Hansen		
Titel:	Live tilbud		
Anslag:	59.701		
Må besvarelsen gøres til genstand for udlån	ja		
Indeholder besvarelsen fortroligt materiale?	Nej		

Titelblad

Opgavetitel:	Live tilbud
Omfang:	59.701 tegn / næsten 25 sider
Afleveringsdato:	25 november 2020
Vejleder/underviser:	Kristian Langborg-Hansen
Fag:	Afgangsprojekt for Akademiuddannelse i IT - udvikling
Uddannelsessted:	Smartlearning.dk del af erhvervsakademi CphBusiness
Forfatter:	Christian A. Nicolau

RESUMÉ

I denne rapport vil jeg undersøge hvordan vi kan IT-sikre Team Nordics software samt hjemmeside, for at forhindre madspild. Hvilke fordele der opnås ved at benytte vores platform og hvad der ligger til grund for, at virksomheder som Team Nordic skal vælge vores løsning. Team Nordic er en nystartede virksomhed, der har efterspurgt vores bud på live tilbud. Vi er et IT-konsulent firma, der hjælper virksomheder som Team Nordic med deres it-løsninger.

I denne rapport vil jeg som it-konsulent analysere hvordan live tilbud ideen bliver lavet, fra idefase til produkt og hvilke muligheder der kan opnås, ved brug af produktet. Hertil vil jeg også præsentere min prototype, for at give et bedre indblik i dens funktioner.

Igennem hele rapporten, vil jeg analysere fra it-konsulent perspektiv. Det omhandler jeg betragter mig som "vi". Hvor Team Nordic er vores kunder.

Livetilbud er en it-løsning, som giver virksomhederne muligheden for, at reducere deres madspild ved at tilbyde dem vores platform, hvor de kan sælge deres overskydende varer, som ellers ville overskride sidste udløbsdato, og derved gå til spilde

Dette vil forårsage madspild som kan forhindres ved, at butikkerne sætter deres valgte varer ind i vores program og kunden derefter kan købe varerne, inden lukketid til en favorabel pris.

FORORD

Denne rapport har til formål at besvare problemstillingens hovedspørgsmål, samt underspørgsmål til afgangsprojektet for Smartlearning (Akademiuddannelse i IT med retning mod udvikling).

Uddannelsen er taget ved smartlearning.dk, som er en del af CPH Business School.

Rapporten er opbygget på en indledning, en problemstilling, en problemformulering hvor jeg herunder vil komme ind på mine hoved- og underspørgsmål, metodevalg, herunder fordele og ulemper, projektplan, valg af teori samt afgrænsning. Afslutningsvis der er en konklusion, perspektivering samt hvilke udviklingsmuligheder der er.

Rapporten er afleveret D. 25. november 2020

Indholdsfortegnelse

RESUMÉ.....	3
FORORD.....	4
1.0 INDLEDNING	8
1.1 PROBLEMSTILLING	8
1.2 PROBLEMFORMULERING	9
1.3 AFGRÆNSNING.....	9
1.4 METODE	10
2.0 MADSPILD.....	12
3.0 AUTHENTICATION	17
4.0 RISK TIER.....	17
5.0 GOVERNANCE	18
6.0 RISIKOANALYSE - SOFTWARE	18
7.0 RISIKOANALYSE - HJEMMESIDE.....	20
8.0 FORSLAG TIL SIKKERHEDEN	22
8.1 ROI BEREGNING	23
8.2 CLOUD	23
8.3 LOCKS	24
8.4 CONNECTION STRINGS.....	24
8.5 IP-ADGANG	24
8.6 FORDELE OG ULEMPER VED CLOUD	25
9.0 PROJEKTPLAN	29
9.1 KONSTRUKTION SOFTWARE.....	29
9.2 LOGIN	29
9.3 SOFTWARE	30
9.3.1 TILBUD	30
9.3.2 AFHENTNING.....	31
9.3.3 DE SMÅ FUNKTIONER	31
9.3.4 DATABASE TIL SOFTWARET.....	32
9.3.5 OPDATERING AF DASHBOARD STATIKER	32
9.3.6 LOG UD.....	32
9.4 HJEMMESIDE.....	32
9.4.1 DATA.....	33
9.4.2 MENUER	33
9.4.3 HTTPS	33

9.4.4 IMPLEMENTERING AF SECURITY HEADERS	34
9.4.5 AUTHENTICATION	35
10.0 VALG AF TEORI.....	36
10.1 SYSTEM UDVIKLING.....	37
10.2 IT-SIKKERHED	37
10.3 CLOUD	38
10.4 HJEMMESIDE.....	39
10.5 SOFTWARE	40
11.0 KONKLUSION	41
12.0 PERSPEKTIVERING.....	42
12.1 BLEV IKKE LAVET	42
13.0 REFLEKSION	43
14.0 LITTERATURLISTE	44
15.0 BILAG	48
BILAG 1.0 PROBLEMOMRÅDE KUNDER	48
BILAG 1:1 PROBLEMOMRÅDE SLUTBRUGER.....	48
BILAG 1:2 PROBLEMOMRÅDE ADMIN	49
BILAG 1:3 MADSPILD.....	49
BILAG 1:4 PROCENT MADSPILD.....	49
BILAG 1:5 LOGIN	50
BILAG 1:6 SOFTWARE.....	51
BILAG 1:7 TILBUD	51
BILAG 1:7 USE CASE HJEMMESIDE	52
BILAG 1:8 USE CASE SOFTWARE.....	52
BILAG 1.9 PROJEKT PLAN	53
BILAG 2:0 VALIDERING AF SQL INJECTION	53
BILAG 2:1 SECURITYHEADERS	54
BILAG 2:2 SECURITYHEADERS A	54
BILAG 2:3 HTTPS.....	54
BILAG 2:4 CONNECTION STRING	55
BILAG 2:5 PHISHING	55
BILAG 3:0 LOGIN KODE.....	56
BILAG 3:2 OPRET KODE TILBUD.....	57
BILAG 3:3 REDIGERER TILBUD	59
BILAG 3:4 REDIGERE KODE.....	59
BILAG 3:5 SLET FUNKTION KODE.....	60

BILAG 3:6 AFHENTNING KODE FUNKTION	60
BILAG 3:7 NULSTILLE TEKSTBOKSENE	61
BILAG 3:8 INPUT TIL REDIGERING	61
BILAG 3:9 LUK/MINIMERER PROGRAMMET	61
BILAG 4:0 MENUER	62
BILAG 4:1 FORBINDELSE AF DATABASE TIL DATAGRID	63
BILAG 4:2 DATABASE QUERIES USER	63
BILAG 4.3 DATABASE QUERIES TILBUD	63
BILAG 4.4 "INCREMENTATION" VED DESIGN PATTERNS	64
BILAG 4.5 "INCREMENTATION" AF TILBUD	64
BILAG 4.6 LOGUD	64
BILAG 4:7 HJEMMESIDE MENUER.....	64
BILAG 4:8 SECURITY HEADERS	65
BILAG 4.9 AUTHENTICATION.....	66
BILAG 4.10 MODEL OG CONTROLLER.....	67
BILAG 5:0 DATABASE ROLLES	68
BILAG 5:1 SQL INJECTION SÅRBARHED	68
BILAG 5:2 ANTI SQL INJECTION	68
BILAG 5.4 FULLY DRESSED USE CASE SQL INJECTION.....	70
BILAG 5.5 FULLY DRESSED USE CASE CLICKJACKING	71
BILAG 5.6 HJEMMESIDEN.....	72
BILAG 5.7 AFHENTNING	72

1.0 INDLEDNING

På det globale plan udgør madspild 1.3 milliarder tons om året, hvilket svarer til at brødføde 3 milliarder mennesker verden over. I Danmark smider vi lige omkring 700.000 tons mad ud om året, hvor størstedelen af maden er frugt som udgør 41%, kød og kødprodukter 24%, brød og kager 20%. (Miljøministeriet (2015) *Regeringens Strategi for affaldsforebyggelse "Danmark uden affald II"*. Lokaliseret den 13-11-2020 på [linket](#))

Madspild har enorme store konsekvenser på det miljø- og klimamæssige plan og er noget vi alle skal være med til at bekæmpe og mange virksomheder er med i kampen om madspild. I 2018 lykkes det eksempelvis Q8 at redde over 69.000 måltider fra at gå til spilde, hvilket svarer til 175 tons CO₂. (Q8: Pressemeddelelse (25-09-2019) Over 69.000 måltider reddet: Q8 og kunderne bekæmper madspild. Lokaliseret 10-11-2020 på [linket](#))

Med vores IT-løsning kan vi være med til at bidrage til, at formindske madspildet i Danmark og gøre en indsats for reducere af madspild. Se hele statikker fordeling i (Se [Bilag 1.3](#)) og (Se [bilag 1.4](#))

1.1 PROBLEMSTILLING

I 2016 startede en af Team Nordic iværksætter med, at arbejde på tankstationen Circle K. Her stod han for kassebetjening og salg. Hver aften når han lukkede butikken, skulle han sørge for alt mad der ikke blev solgt på dagen, blev smidt ud i deres madcontainer.

Han undrede sig, hvorfor Circle K ikke havde en løsning på deres madspild. Da de er de førende skandinaviske leverandører af brændstof til transport, hvor de kun i Danmark har 220 bemandede benzinstationer (Se link [Circle K – Om os](#))

Hvis hver tankstation smed samme mængder mad ud, som han gjorde, ville der være tons af mad om året der vil gå til spilde, kun for stationerne i Danmark.

Med denne store undren, valgte han at starte virksomheden Team Nordic, for bekæmpelse af madspild. Eftersom Team Nordic ingen IT-erfaring havde, kontaktede de os, om vi kunne lave en IT-løsning der kunne bekæmpe madspild på det danske marked, som f.eks. tankstationer, restauranter

butikker mm. For Team Nordic var det essentielle at finde ud af, hvordan de kunne være med til at bekæmpe madspild, ved brug af de IT teknologier som vi har i dag. Kan man med Team Nordics IT-løsning være med til at bekæmpe madspild og kan man sprede budskabet og hermed gøre andre beviste om problemet?

1.2 PROBLEMFORMULERING

Problemformulering er baseret på hovedspørgsmål samt underspørgsmål, som vil blive analyseret og undersøgt.

Hovedspørgsmål

- Hvordan reducerer virksomhederne madspild?
- Hvordan imødekommer Team Nordics deres behov og sikre den bedste løsning?

Underspørgsmål

- Hvordan sikre vi os, at hjemmesiden er it-sikker?
- Hvordan laver vi et software til madspild?
 - o Hvordan sikre vi os at programmet er it-sikkert?
 - o Hvordan kan softwaret være mere brugervenligt?
- Hvordan sikre vi os clouden er sikker?
 - o Hvilken fordele/ulemper er der ved cloud?
 - o Hvilken fordele er der ved at hoste løsningen på cloud?

1.3 AFGRÆNSNING

Da mit fokusområde er problemformulerings hoved-underspørgsmål, har jeg derfor valgt at afgrænse mit område, da nedenstående punkter er meget omfattende. Jeg ville derfor afgrænse mig til at undersøge og lave de nedenstående punkter og derfor ikke redegøre/undersøge følgende.

1. Systemudvikling Unified Process Construction samt Transition fasen
2. Software beregning af CO2, købte tilbud

3. Software dekryptering af identity format
4. Database inner join for at kombinere 2 tabeller
5. Hjemmeside: Oprettelse af pris samt tilbud
6. Hjemmeside: Download knappen til at download programmet
7. Hjemmeside: Betalings system
8. *Security headers X-Content-type-Options*
9. *Security headers Permission-Policy*

1.4 METODE

Til at afdække problemformulerings hovedspørgsmål samt underspørgsmål, om hvordan vi reducerer madspild, benyttes der viden fra tidligere fag på Smartlearning og viden fra internetkilder. Med denne viden giver det os muligheden, for at danne os et overblik over dele af fagene, vi skal bruge til at løse problemet.

Til vores It-løsning vil jeg gennemgå nedenstående underpunkter og kort forklare deres betydning til projektet.

1. SYSTEMUDVIKLING

Ved brug af systemudvikling er der stærkt fokus på Team Nordic virksomhed, da det er det første indtryk man får af dem. Systemudvikling skaber rammer og guidelines for Team Nordic på baggrund af problemstillingen.

Idet vi har nøglekrav som skal overholdes, er det vigtigt at analysere Team Nordics behov, hvilket er problemformulering hovedspørgsmål.

2. SIKKERHEDEN

For at opretholde sikkerheden på hjemmesiden, vil der blive lavet en risikoanalyse, som sikrer Team Nordic mod angreb som f.eks. "clickjacking, sql injection og phishing". Derudover vil der også blive lavet en analyse over sikkerheden på softwaret, cloud og hjemmesiden.

3. SOFTWARE

Hele løsning kommer til at have et software, hvor slutbruger kan logge sig på og

oprette et tilbud på de madvarer de vil sælge. På baggrund af viden fra tidligere fag, er der brugt C# til softwaret. I C# kan man nemt lave et brugervenligt interface, så det er mere overskueligt for slutbrugeren at navigere rundt i. Slutbrugeren er de virksomheder der kommer til at have gavn af Team Nordics ide, hvor slutbrugeren er dem, der gør gavn af at købe madvarer.

4. HJEMMESIDEN

Når slutbrugeren har oprettet tilbuddet på softwaret, vil de efterfølgende komme ind på hjemmesiden, hvor brugeren kan se det.

Når tilbuddet er oprettet på hjemmesiden, kan de efterfølgende købes af brugerne. Hjemmesiden bliver lavet med asp.net, da både programmet og hjemmesiden fungerer bedst med hinanden. Hele beskrivelsen af projektplanen kan ses i [9.0 Projektplan](#)

5. CLOUD

Hele løsning kommer til at være hostet på cloud. Da cloududbyderen har værktøjer, der hjælper med it-sikkerheden.

Til dette projekt har vi valgt Microsoft Azure cloud som deres løsning.

Dette bruger vi til databehandling samt hosting for Teamnordics IT-løsning.

Microsoft Azure har den fordel at sikkerhedsindstillinger er it-sikret og man kan begrænse medarbejders, fejl hvis de f.eks. ved en fejl kommer til at slette en tabel.

Til sidst vil vi fokusere på sikkerheden som en helhed hvorefter vi undersøger og giver en dybere besvarelse på spørgsmålet i problemformulering. Vi vil efterfølgere også gå i dybden med alle metoder der er nævnt og undersøge hvilken fordele hver funktion/metode har til fordele for Team Nordics behov.

6. DATABASEN

Efter slutbrugeren har oprettet et tilbud, vil de blive gemt i databasen som befinder sig i clouden. Udover den nævnte database, har vi også valgt at have en "test-database" som er lokal.

1:5 OPSUMMERING

Med denne viden fra fagene kan vi imødekomme problemformulering og begynde, at danne os et overblik over løsninger der kan medføre besvarelse af hovedspørgsmål samt underspørgsmål. Til opbygning af software, har vi valgt at bruge c#, da viden fra de lærte fag udgør en komplet forståelse for, at imødekomme problemet. Fordele ved c# er nemlig at asp.net og c# går hånd i hånd. Dette er en fordel fordi vi skal ikke bruge ekstra tid og udviklerne skal ikke bruge tid på at udvikle hjemmesiden.

Til oprettelse af database samt tabel til både softwaret og hjemmesiden bruger vi *Microsoft sql server management*. Dette gør, at vi nemt, kan teste vores "on fly" data og validere vores "Query string" for at se om alt virker. Vi vil efterfølgende begrunde vores valg af teori (se punktet for [valg af teori](#))

2.0 MADSPILD

På baggrund af problemstillingen og problemformuleringen, har vi valgt at bruge Unified Process til at analysere hvordan vi reducere madspild samt underspørgsmålene. Unified Process modellen fokuserer på nøglekravene (Ivar, Craig, Kendall (4 november) Unified Process. Lokaliseret 05-10-2020 på [Wikipedia Unified Process](#)) i problemformulering som dækker Team Nordics behov.

Følgende faser er

1. *Forberedelse*
2. *Etablering*

1. FORBEREDELSE

Forberedelsesfasens hovedpunkt er at forstå kravspecifikationerne. Disse kravspecifikationer analyseres på baggrund af forberedelsesfasen. For at stifte bekendtskab til problemområdet (Ove Thomsen (03-03-2019) Unified Process. Lokaliseret d. 06-03-2020 [uge 2 Unified Process](#)) vil vi først analysere hjemmesiden og derefter softwaret.

Følgende punkter viser først fase i forberedelsen

Problemområde analyse
Anvendelsesområde analyse

- *Brugersystemet*
- *Betalingssystemet*
- *Tilbudssystemet*

PROBLEMOMRÅDEANALYSE

Problemområdet er som nævnt for at stifte bekendtskab til kunderne, systemet og slutbrugeren.

Dette gøres ved at analysere hvilken "Behaviour" samt "properties" Team Nordics kunder har.

Vi vil som det første analysere kunderne, som benytter og køber tilbuddene på hjemmesiden. (se

Bilag [1.0 problemområde kunder](#))

Vi kender nu til problemområdet på Team Nordics kunder og hvilken "Behaviour" de har på

hjemmesiden. Problemområdets slutbruger er blevet analyseret i (Se Bilag [1.1 Slutbruger](#)

[Problemområde](#))

Vi kan nu drage en konklusion som viser forskellen på kunderne og slutbrugeren og hvilken fordele og "Behaviours" de har. Efterfølgende vil vi lave den samme analyse til "Admin" på hjemmesiden, da det er en del af sikkerheden på hjemmesiden. Følgende figur ses på [1.2](#).

ANVENDELSESOMRÅDEANALYSE

De centrale spørgsmål omkring IT-systemets brug er": (Lars Mathiassen- OOA&D side 122)

1. Hvem skal bruge systemet?
2. Hvordan skal system bruges?

De nævnte punkter besvares i form af aktører som er slutbrugeren, kunderne og brugsmønstrene.

Da aktørerne allerede er blevet analyseret i problemområdet, med spørgsmålet om hvordan de skal bruge systemet og hvilken opførelse de har. Vil vi gå dybere ind i analysen, for at se hvordan tilbuddene virker for kunderne og hvilken roller de har i forhold til systemet og hjemmesiden.

Følgende brugsmønstre viser interaktion mellem aktører og systemet som ses i nedenstående figur.

Brugsmønstre	Kunder (Aktører)	Slutbruger (aktører)	Admin
Opret tilbud	x	✓	✓
Slet tilbud	x	✓	✓
Redigere tilbud	x	✓	✓
Download software	x	✓	✓

Figur 1.3 brugerinteraktion.

De centrale spørgsmål indebærer, også analyse indenfor betalingssystemet. Da betalingssystemet også er en stor del af hjemmesiden og har en interaktion mellem aktører og systemet som ses i nedenstående figur.

Brugsmønstre	Kunder (Aktører)	Slutbruger (aktører)	Admin
Månedlig abonnement	x	✓	✓
Annullere abonnement	x	✓	✓
Betaling af tilbuddene	x	✓	✓
Køb af tilbud	✓	✓	✓
Køb af pakker	x	✓	✓

Figur 1.4 betalingssystemet

Til sidst vil vi analysere tilbudssystemet som er hovedformålet. Tilbudssystemet har til formål at reducere madspild, ved at oprette tilbud. Nedenstående figur viser hvordan tilbuddene hænger sammen med hjemmesiden og software, samt hvilke brugsmønstre der hører til de forskellige systemer.

Brugsmønstre	Software (Aktører)	Hjemmeside(aktører)
Opret tilbud	✓	x
Slet tilbud	✓	x
Redigere tilbud	✓	x
Download software	x	✓
Køb tilbud	x	✓
Kategori valg af tilbud	x	✓

Figur 1.5 Tilbud.

2. ETABLERING

Idet vi kender brugsmønstret og hvordan systemet interagerer med aktøren og systemet, kan vi gå videre med kravspecifikationerne. Kravspecifikationerne bliver en del af "use cases" som vil blive brugt til, at vise nøglefunktionerne i systemet. Der vil efterfølgende blive lavet "fully dressed use cases" for at analysere systemet svaghed, systemets stærke sider samt logikken bagved systemet.

1. *Phishing* (Se [Bilag 5.3 Fully Dressed use case](#))
2. *Sql injection* (Se [Bilag 5.4 Fully Dressed use case](#))
3. *Clickjacking* (Se [Bilag 5.5 Fully Dressed use case](#))

Risikodrevet

"Risici identificeres tidligt i forløbet og valget af, hvilke use cases man skal koncentrere sig om i starten, foretages udefra, hvor højt de scorer i risikovurdering (eliminering af de største risici først)" (Ivar, Craig, Kendall (4 november) Unified Process. Lokaliseret 05-10-2020 på [Wikipedia Unified Process](#)).

For at imødekomme de risikodrevende use cases, har vi valgt at analysere 3 angreb på begge aktører og en enkel på en medarbejder. Følgende risikovurdering beskrives og besvares længere nede i [5.0 GOVERNANCE](#)

Dette imødekommer også problemformulerings underspørgsmål, om hvordan vi sikrer os den mest it-sikre løsning til software samt hjemmeside. Som første del af etableringsfasen, er det vigtigt at vi designer use cases som nævnt. Eftersom vi har analyseret Team Nordics trusler, kan vi nu gå dybere ind i systemet og lave "fully dressed use cases".

Vi vil derefter teste hvordan systemet vil reagere på de følgende angreb. Efterfulgt af en dybere analyse om hvordan vi kan sikre os den bedste løsning.

Besvarelsen af de 3 overstående fully dressed use cases, giver os en fuld forståelse for, hvad der helt præcis sker, når en hacker angriber systemet. Grunden til de overstående angreb, er fordi de er nogle af de mest almindelige angreb ifølge (Darius S. (29-03-2018) Website Hacking – The Most Common Techniques. Lokaliseret d. 09-11-2020 på [linket](#))

For at besvare og begrunde de overstående angreb, vil vi tage udgangspunkt i "security headers". "Securityheaders" er en hjemmeside, hvor man kan se hvor sikker ens hjemmeside er. Denne funktion uddyber svaghederne, hjemmesiden måtte have. Team Nordics hjemmeside fik "F" i Securityheaders scan, som ses i [bilag 2.1 Securityheaders](#) hvilket er det ringeste på skalaen. Vi vil danne os et overblik over, hvordan Team Nordics hjemmeside bliver mest IT-sikret, både for deres renommé, men også for deres kunders sikkerhed. For at gøre deres hjemmeside mere IT-sikker, vil vi fokusere på understående punkter

1. *Content Security Policy*
2. *X-Frame-Options*
3. *Referer-Policy*

Disse punkter er fatale for Team Nordics hjemmeside, da de er sårbare overfor angreb og derfor vil vi forklare hvorfor og hvordan det kan være skadeligt for hjemmesiden.

1) *Content Security Policy* er et beskyttelseslag der forhindrer angreb som XSS. XSS er et hul i systemet der giver hackeren adgang til at indsprøjte, virus i klientsiden og giver hermed hackeren mulighed for at undgå systemers alarmer.

I Security Header (Se dataindsamling [9.4.4 bilag for Securityheaders](#)) vil der være en dataindsamlingskode, der forklarer og besvarer hvordan vi forhindre *Content Security Policy* på Team Nordic hjemmeside. (Scott Helme (27-11-2014). Content Security Policy - An Introduction. Lokaliseret den 30-10-2020 på [linket](#))

2) *X-Frame-Options* er en vigtig del af Team Nordics renommé, da renomméet kan tage skade af X-frame-Options. X-Frame-Option er et angreb på hjemmesiden der kopierer layoutet og efterfølgende manipulere alt, fra knappen til personlige informationer såsom, køn, adresse, brugernavn m.m. også kaldt "Clickjacking".

Efterfølgere vil der blive lavet en risikoanalyse, om hvor meget skade et "Clickjacking" angreb vil forårsage på Team Nordics hjemmeside. Derefter vil vi lave endnu en risikoanalyse, hvor vi undersøger hvor meget 3 clickjacking angreb, vil koste Team Nordic (Scott H. (24-03-2015).

Hardening your HTTP response headers. Lokaliseret 30-10-2020 på [linket](#))

3) Referer-Policy I dag er reklamer på de sociale medier en stor del af markedsføring. Når Administratoren af siden, ser statistikkerne fra reklamen, kan de se hvor kunderne kommer fra. Refererer- Policy er den mægte data af personlige informationer Admin får lov til at se (Scott H. (17-02-2017). A new security header: Referrer Policy. Lokaliseret d 30-10-2020 på [linket](#))

3.0 AUTHENTICATION

Som en del af problemformulerings underspørgsmål om IT-sikkerhed på hjemmesiden, har vi implementeret et rollesystem hvor lederen af de forskellige afdelinger, kan tildele dem rettigheder til deres medarbejder, for at sikre de har den adgang de skal bruge. Formål med dette er, at i IT-sikkerhedens verden er det menneskerne, der er den største fare. Da vores authentication system giver lederne begrænset adgang og har herved har de mulighed for, at slette følsomme data. (se [9.4.5 Authentication](#))

4.0 RISK TIER

Risk Tier er en vejledede guideline, der vejleder Team Nordic, hvad de skal være opmærksom på. Når vi undersøger hvilken Risk Tier, Team Nordic hører under, er det vigtigt at kigge på deres sårbarhed. Idet vi analyserer Risk tier, er det vigtigt vi fokuserer på følgende 3 krav (**CIS-RAM-Version-.1.0. 2015 s.12**)

- A. *Nist Tier*
- B. *Ekspertise*
- C. *Tid*

Nist Tier blev analyseret på baggrund af Team Nordic, som er blevet vurderet til Risk tier 1, fordi de ikke koordinere deres sikkerhedsplaner og krav. Team Nordic har på nuværende tidspunkt erfaring nok til at genkende de store trusler, men ingen ekspertise omkring hvordan man behandler dem. De har den tid det kræves, for at evaluere informationsrisici ved hjemmesiden. Vi vil derfor

undersøge følgende 3 angreb for at belyse hvordan vores løsning er sikker mod evt. angreb.

Løsningerne findes i risikoanalyse.

1. *Sql injection (software)*
2. *Clickjacking (web)*
3. *Phishing (psykisk)*

5.0 GOVERNANCE

IT-sikkerhed er en stor del af alle it-løsninger, derfor er det meget betydningsfuldt for virksomheden, at de har styr på sikkerheden. På baggrund af problemformulering vil vi analysere, hvordan Team Nordic bedst sikre sig mod nævnte angreb og forbereder sig bedst mulig på angreb. I denne sektion vil vi gå i dybden med hovedspørgsmålene og underspørgsmålene om IT-sikkerhed på hjemmeside, software og remote. Samt hvordan vi rent teknisk løser overstående trusler hos Team Nordic.

For at belyse vigtigheden i IT-sikkerheden, har vi valgt at lave en risikoanalyse, om hvor omfattende de 3 forskellige angreb kan påvirke Team Nordic, samt hvor meget det kan ramme deres renommé. I denne rapport vil følgende punkter vurderes, for at undersøge hvor meget skade de kunne have.

1. *Impact*
2. *Likelihood*
3. *Risk Threshold*

6.0 RISIKOANALYSE - SOFTWARE

For at demonstrere hvor vigtigt IT-sikkerheden er indenfor problemformulering, har vi valgt at lave en risikoanalyse, der undersøger hvor meget skade der kan blive forårsaget, samt hvor hårdt Team Nordic kan blive ramt, hvis de ikke følger vores anbefaling om sikkerhed i deres it-løsning.

Vi vil efterfølgende risikoanalysere 3 angreb af Sql injection, Clickjacking og Phishing.

som et eksempel tager vi udgangs punkt i at Team Nordic bliver angrebet 3 gange på et år.

I figur 1.0 nedenfor demonstrer hvor stor impacten er.

Da vi på nuværende tidspunkt ved at, Team Nordics virksomhed er tildelt tier 1, kan vi bruge figur

1.0 til at videreanalysere i cis ram workbook (Corden Pharma (2015). CIS-RAM-Workbook-Version-1.0. Center for Internet security. Lokaliseret d. 25-02-2019 på [linket](#))

Impact score	Impact score defineret
1	Ingen eller meget lidt skade i virksomheden
2	Skaden kan ikke accepteres
3	Skaden er svær at genskabe

Figur 1.0

Følgende rapport vil også vise hvor meget 3 SQL injection angreb har på Team Nordic og hvor fatalt deres renommé kan blive ramt. (Figur 1.1.)

Samtidig vil vi analysere hvor stor "likelihood" er, for at de 3 angreb gentager sig på Team Nordics softwaret.

Likelihood Score	Likelihood defineret
1	Angrebet kommer ikke til at ske
2	Angrebet kunne ske, men virksomheden forventer det ikke
3	Forventet, men ikke almindeligt
4	Almindeligt
5	Kan ske nu

1.1 Figur

SQL INJECTION er et angreb fra hackeren, hvor de i logintekstboksen skriver databaseforespørgsler, om at udprinte data f.eks. brugernavn, kodeord m.m. Dette forårsager dårligt renommé og kan give et stort impact på Team Nordic.

Vi vil analysere hvor meget impact og Likelihood Sql injection kan have i figur 1.4

Overblik over truslen	Vores impact	Likelihood score	Impact score
Sql injection	2	3	6

1.4 Figur

Impact Threshold	x	Likelihood Threshold	=	Risk Threshold
2	x	3	=	6
<p>Følgende analyse om sql injection blev vurderet til 6. Da alle de 3 angreb kan risikere at miste både økonomi, renommé, og kunder, kan dette blive fatalt for Team Nordic, hvis de ikke har værktøjer til at genoprette databasen hurtigt muligt. Det tekniske resultat 10.4 hjemmesiden</p>				
Acceptable Risk			<	6

1.5 Figur

7.0 RISIKOANALYSE - HJEMMESIDE

CLICKJACKING er et angreb på en hjemmeside, hvor hackeren kopierer hjemmesidelayout og laver sine egen funktioner på hjemmesiden. Dette kan resultere i, at hackeren selv kan manipulere betaling, personlige informationer og kan forårsage stor skade på Team Nordic. På baggrund af den nuværende viden fra "[2.0 Madspild Security Headers Frame-Options](#)" ved vi hvordan vi håndterer denne trussel og hvor højt den vægter på risikoanalyseskala samt hvordan de bedst kan være forbedret på sådan en trussel. (Se figur 1.6)

Overblik over truslen	Vores impact	Likelihood score	Impact score
Clickjacking	2	3	5

1.6 Figur

Impact Threshold	x	Likelihood Threshold	=	Risk Threshold
2	x	4	=	8
Følgende analyse om Clickjacking blev vurderet til 8. Da hackeren har muligheden for at stjæle Team Nordics kunder personlig informationer, hvilket gør det fatalt for Team Nordic. Samtidig er det forbudt, ikke at beskytte deres kunders informationer.				
Acceptable Risk			<	8

1.7 Figur

PHISHING er et angreb hvor hackeren sender mails rundt i håbet om, at få en til at downloade en fil, eller klikke på et link der efterfølgende kan medføre, at hackeren får adgang til din computer. I vores tilfælde kan det være at hackeren udgiver sig for at være chefen, hvor han spørger om du kan sende midler fra virksomhedens konto til hackeren. For at understrege hvor hurtigt det kan tage fart, vil vi analysere hvordan Team Nordic kan reducere dette og hvor højt på risikoskalaen det vil skade dem, hvis de blev udsat for det.

Følgende risikoanalyse tager udgangspunkt i dette eksempel, hvorpå hackeren udgiver sig for at være chefen der beder en medarbejder om at sende penge. (Se [bilag 2:5 Phishing](#))

For at belyse dette problem, vil vi gå dybere ned i analysen og redegøre for, hvad de kan gøre for at reducere denne type angreb, som ses i nedenstående figur 1.8

Impact Threshold	x	Likelihood Threshold	=	Risk Threshold
2	x	4	=	8
Følgende risikorapport viser hvor meget skade, 3 angreb fra Phishing kunne have på Team Nordic. Impact threshold blev vurderet til 2, da det faktisk kan gå udover deres renommé men også deres samarbejdes med andre virksomheder.				
Acceptable Risk			<	8

1.8 Figur

For at analysere hvordan Team Nordic kan reducere de 3 nævnte angreb, vil vi give forslag til hvordan vi løser dette. På baggrund af (Corden Pharma (2015). CIS-Controls-Measures-and-Metrics-for-Version-7-cc-FINAL. Center for Internet security. Lokaliseret d. 25-10-2020 på [linket](#)) vil vi komme med 3 forslag til de 3 nævnte angreb i (figur 1.9-2.0 og 2.1)

8.0 FORSLAG TIL SIKKERHEDEN

Summary	Forslag til SQL injection er følgende blevet vurderet til CIS kontrol 18.1 hvor udvikleren skal sikre, kodningsmetoden, som både skal være sikker og passe til det programmeringssprog der brugs til at udvikle.
Date Completed	09/11/2020
Acceptable Risk Score is less than	2

Figur 1.9

Summary	Forslag til Clickjacking er følgende blevet vurderet til CIS kontrol 18.7, da det er vigtigt at udvikleren overholder sikkerhedskodning til internetapplikationer og anvender statistiske og dynamiske analyseværktøjer.
Date Completed	10/11/2020
Acceptable Risk Score is less than	1

Figur 2.0

Summary	Forslag til Phishing er følgende blevet vurderet til CIS kontrol 17.3, da Team Nordic skal implementere sikkerhedskurser til deres medarbejder, så de får den nødvendige viden om Phishing og andre sikkerhedsrisikoer.
Date Completed	12/11/2020
Acceptable Risk Score is less than	3

Figur 2.1

8.1 ROI BEREGNING

Med en “return of Investment (ROI)” beregning, vil implementering af nye løsninger og omkostningerne, være synligt for Team Nordic. Det vil give et præcist indblik i, hvor meget det kan komme til at koste dem. Som hovedfokus, vil vi lave en ROI beregning på Phishing. Som tidligere nævnt i [bilag 2:5](#) tager vi udgangspunkt i et virkeligt eksempel, som skete for en medarbejder i et shippingfirma. Denne hændelse ville have kostet shippingfirmaet 266.000 DKK, for et Phishing angreb. Men da medarbejder fandt ud af, det var Phishing angreb, valgte hun derfor ikke at besvare hackeren, eller lave bankoverførslen. De mange virksomheder er ikke klar over, at deres medarbejder bliver udsat for phishing tit, og de ved heller ikke hvordan de imødekommer disse udfordringer. På baggrund af analysen, vil vi demonstrere via en ROI beregning, hvor meget de kan spare årligt på, at implementere de nævnte forslag til reducere af Phishing.

Team Nordic regner med at få lige omkring 5 angreb om året. Med det nævnte eksempel kender vi prisen på et eventuelt Phishing angreb, hvilket i dette tilfælde er 266.000 kr.

Med vores forslag, vil vi kunne reducere Phishing angreb med 80% (A CISOs Guide to Bolstering Cybersecurity Posture, 2015, s. 8).

1. 5 angreb om året
2. 5 angreb x 266.000 kr. = 1.330.000 kr.
3. Reducering på 80%

Vi har valgt Smartlearning it-sikkerheds fag (Smartlearning (2015) IT-sikkerhed. Lokaliseret d. 04.10.2020 på [linket](#)) for at uddanne deres medarbejder. Prisen for et IT-sikkerhedsfag koster 3.990 kr.

Følgende beregning viser hvor meget Team Nordic kan spare ved at uddanne sin medarbejder og herved hvor stor besparelse de kan få på et år.

$$\begin{aligned}5 \times 266.000 \times 0,80 &= 1.064.000 \\(1.064.000 - 3.990) \div 3.990 &= 265,66 \text{ ROI} \\3.990 \times 265,66 &= 1.059.983 \text{ besparelse om året.}\end{aligned}$$

8.2 CLOUD

For at belyse problemformuleringen, vil vi undersøge hvordan clouden er med til at gøre hjemmesiden samt databasen mere sikker og hvilke fordele/ulempes der kan være, ved at bruge nedenstående cloud indstillinger.

1. Locks
2. Connections string
3. Whitelist / blacklisting
4. Serverless

8.3 LOCKS

Locks er en lås der sikre, at databaserne ikke bliver slettet. Et f.eks. kunne være, hvis en medarbejder hos Team Nordic fik besked på at slette en database, men kommer ved en fejl til at slette den forkerte. Hvis dette skete, vil medarbejderen slette alle deres primære data, med alle deres nuværende kunder, hvilket vil være et enormt stort tab for virksomheden, og hjemmesiden vil herved også være ude af funktion. Med "Locks" kan du låse din database, så man ikke kan slette den. (Morten Christian Empeno Kristensen (2018) Videre IT-sikkerhed. Part 4. Lokaliseret den 06-11-2020 på [linket](#))

8.4 CONNECTION STRINGS

Connection string er en tråd man bruger, til at forbinde hjemmesiden med databasen. Denne tråd plejer at stå i en folder, der hedder app settings. I nogle tilfælde kommer den til at stå i deres kode, hvilket resulterer i, at angriberen kan se denne connection string. Dette vil forårsage fatale skader på både databasen og Team Nordics renommé, da hackeren har adgang til deres database.

For at undgå dette problem har man muligheden for at gemme sin connection string i sin app services, så den netop ikke bliver synlige for hackeren. Dette ses i ([Bilag 2.4 Connection string](#))

8.5 IP-ADGANG

Som en del af Azure cloud, har de en indstilling hvor man kan oprette adgang fra egen IP til databasen. Dette er en fordel, da man på forhånd ved at det kun er den valgte IP, der kan få adgang til databasen. Udover at man skal whitelist sin IP, skal man også kende brugernavnet og kodeordet, før man kan få adgang til databasen.

8.6 FORDELE OG ULEMPER VED CLOUD

WHITELIST & BLACKLISTING

WHITELIST

Clouden er en platform med mange personlige oplysninger og informationer. Man skal være opmærksom på hvem der får adgang til hvilken informationer. Whitelisting er hvor man godkender en IP og i Team Nordics tilfælde, vil den IT-ansvarlig få adgang, da ingen fra Team Nordic har it-erfaring. Dette gør også at udvikleren kan arbejde remote da clouden tillader udviklerens IP. (Karen Mesoznik (1-02-2019). cloud-security-ip-whitelisting. Lokaliseret den 04-10-2020 på [linket](#))

BLACKLISTING

Blacklisting er derimod, hvor du allerede kender IP-adressen fra en usædvanlig bruger, der har prøvet at angribe serveren. Allerede der ved man, at angriberens IP skal være blokeret, så man undgår at det sker igen.

SERVERLESS

Team Nordic har hverken fysiske servere eller cloud og derfor vil vi analysere hvilke fordele og ulemper der vil være, hvis de bruger cloud fremfor fysiske servere.

FORDELE

- Serverless funktioner
 - i. Sikkerhed
 - ii. Reaktionsid
 - iii. Agile udvikling
- Pris
- Server trafik
- Cloud storage
 - i. Adgang til data overalt
 - ii. Pris
- Integration fra monolitiske infrastruktur til Microservices

ULEMPE

- Afhængig af tredjepartsudbyderen
- Langsigtede applikationer
- Integration fra monolitisk infrastruktur til Microservices
- Cloud kontrakter

FORDELE

SERVERLESS FUNKTIONER

i. Sikkerhed

Sikkerheden er en stor del af cloud, da det bl.a. kan være med at reducere angrebene på serveren. Dette kan f.eks. ske hvis en it-ansvarlig glemmer at opdatere firewallen til en nyere version og den eksisterende version har en bug. Det vil resultere i at, hackeren vil kunne komme ind i systemet. Fordelen ved at bruge cloud i Team Nordics tilfælde er, at den opdaterer selv systemerne, til den nyere version uden at Team Nordic skal gøre noget. (Xperience (2018). Cloud Vs On Premise Software: Which is Best For Your Business. Lokaliseret 05-11-2020 på [linket](#))

ii. REAKTIONSTID

En reaktionstid er den tid det tager, for at vise indholdet på f.eks. en hjemmesiden/serveren. Hvis reaktionstiden er for lang, kan Team Nordic hurtigt miste kunder, da kundeoplevelse bliver forsinket. Team Nordic kommer til at have deres server i Vesteuropa, da reaktionstid i Danmark er bedst fra Vesteuropa. Fordelen er at man selv vælger hvor i verden, man vil have ens server i clouden. Hvis Team Nordic vælger at udgive i flere lande, kan de have servere, der har den bedste reaktionstid i de udvalgte lande.

(Morten Christian Empeno Kristensen (23-10-2019) Smartlearning cloud: Opret Function [Video] uge 2)

iii. AGILE UDVIKLING

For udviklerne er teamwork en stor del af deres hverdagen. En virksomhed vil typisk have flere udviklere, hvor hver udvikler sidder med noget forskelligt i samme projekt. Med Serverless kodning kan hver udvikler udgive deres egen kode til løsningen, som kan komme online med et klik.

CLOUD PRIS

Prisen er en stor fordel ved cloud. Faktisk så stor at de fleste vælger clouden frem for andre fysiske servere, netop på grund af den lave pris.

Hvis Team Nordic eksempelvis skal have database, vil det koste omkring et engangsbeløb på 35.000

kr. og 3 ugers ventetid. Hvis de derimod vælger en cloud platform som deres løsning, vil det tage alt i alt ca. 2 min inkl. oprettelse og koster 36 kr. månedligt for en lille database.

SERVER TRAFIK

En af fordelene ved cloud er scaling funktionen. Scaling funktionen er opdelt i 2 punkter som er scaling up og scaling out. (Francis Laurence 27-01-2020. Cloud Scalability: Scale Up vs Scale Out. Lokaliseret den 05-11-2020 på [linket](#))



10:12 figur

Scale up betyder at hvis trafikken er lav på serveren, scaler den ned. Hvorimod hvis trafikken er høj, så scaler den op. Teknisk set betyder det at hvis den scaler op bruger den mere CPU/RAM/DISK og så opfylder den de krav, for den høje trafik. Vertikal up er ofte brugt til små og mellem store virksomheder.

Scale out betyder at den scaler på serverne. Dette indebærer tilføjelse af flere behandlingenheder som f.eks. CPU eller database til serveren.

Scaling out reducere den høje mængde data og spreder det ud på de forskellige noder. Så hver node har lige meget at arbejde med.

CLOUD STORAGE

Når man bruger Cloud opbevaring, har storage reduceret risikoen for systemfejl. Alle de gemte data er sikkerhedskopieret på eksterne enheder hos cloududbyderen, oftest langt væk fra ens lokation. Cloududbyderne bliver per automatisk sikkerhedskopieret. (Securestorageservices 05-11-2020. Pros and Cons of Cloud Storage. Lokaliseret den 05-11-2020 på [linket](#))

i. ADGANG TIL DATA OVERALT

Da ens data gemmes eksternt, kan man få adgang til dem, uanset hvor man er. Dette er især nyttigt hvis Team Nordic har planer om at rejse til flere lande, for at udvide konceptet eller hvis de får flere medarbejdere, der skal arbejde på tværs af lande. (Securestorageservices 05-11-2020. Pros and Cons of Cloud Storage. Lokaliseret den 05-11-2020 på [linket](#))

ii. PRIS PÅ STORAGE

Cloud Storage eliminerer behovet for at betale softwarelicenser og opdateringer, da de fleste Cloududbydere har en månedlig pris, hvilket er en fordel.

INTEGRATION FRA MONOLITISK INFRASTRUKTUR TIL MICROSERVICES

En monolit er det vi i dag kalder, en klassisk it-løsning, hvor alle dele af softwaret er en samlet service. Når der foretages forbedringer og ændringer eller opdateringer af en monolit arkitektur, opdateres hele systemet på en gang. Hvorimod Microservices er delt op i flere forskellige små services, som hver især har sin funktion. Dette minimerer risikoen for nedetid og reducere fejlfinding. Med Microservices kan Team Nordic vælge de værktøjer der passer til deres nuværende behov til de enkelte opgaver. Det resulterer i, at Team Nordic kan levere nye features og opdatering til slutbrugeren. (Magnus Boye (16-12-2016). Træt af it-monolitten? Prøv en microservices. Lokaliseret den 06-11-2020 på [linket](#))

ULEMPER

AFHÆNGIG AF TREDJEPARTSUDBYDER

I de fleste tilfælde vil virksomheder som Team Nordic være afhængig af en tredjepartsudbyder, da de fleste cloududbydere ikke har alt på deres platform. Det kan f.eks. være at Team Nordic har brug for en API til at gemme nogle data, men dette kræver en tredjepartsudbyder.

LANGSIGTEDE APPLIKATIONER

I nogle tilfælde kan applikationerne være mere bekostelige på cloud, end hvis man kører det på en fysiske server. Da clouden tilbyder services til en dyre plan, hvilket giver større omkostninger.

INTEGRATION FRA MONOLITISK INFRASTRUKTUR TIL MICROSERVICES

Monolitisk arkitektur er en ulempe for virksomheder, hvis de vælger at bruge cloud, da de skal integrere hele deres arkitektur i Microservices. I mange virksomheder har de systemer der er 20 år gamle, som aldrig bliver lavet til microservices (Magnus Boye (16-12-2016). Træt af it-monolitten? Prøv en microservice. Lokaliseret den 13-11-2020 på [linket](#))

CLOUD KONTRAKTER

I visse tilfælde er man bundet til en kontrakt, hvilket kan være en ulempe, da man ikke kan vide hvor meget lagerplads man får brugt. Derudover har en storage, også en tendens til at stige i pris løbende (Drew Robb (10-10-2018) Cloud Storage Pros and Cons. Lokaliseret den 12-10-2020 på [linket](#))

9.0 PROJEKTPLAN

For at belyse hver Fully dressed use case, har vi valgt at dele det op i iterationer. Hver iteration omhandler fully dressed use case, som tidligere nævnt i rapporten.

Idet vi har elimineret risikoerne for fully dressed use cases og besvarede dem, er det nu muligt at begynde på dem. Følgende projektplan viser netop hvordan, hver iteration er delt op i en tidsplan. Tidsplanen er med til at sætte en deadline på hver iteration.

Følgende projektplan ses i [Bilag 1.8 Use case software](#)

9.1 KONSTRUKTION SOFTWARE

Som en del af "Unified Process" fasen, har vi nu samlet alle de nøglekrav samt use cases.

Til opbygning af konstruktionsfasen, vil der blive udviklet og designet et software som vil være brugervenligheden. Vi vil efterfølgende beskrive hver funktion i software hvor vi til sidst, vil fremstille koden efterfulgt af analyse af designet.

9.2 LOGIN

Til at starte med vil vi, analysere loginet som er en del af softwaret (Se [Bilag 1.6 Software overblik](#))

Login har til formål at gøre det let for slutbrugeren at logge ind. Vi har efterfølgende lavet et panel

hvor vi har taget to labels til lukning og minimeret softwaret. Der er både en knap for login og en knap for nulstilling af login. Følgende dataindsamling for loginet ses ([bilag 2.0 validering/login af sql injection](#))

9.3 SOFTWARE

På baggrund af problemformulerings underspørgsmål, har vi valgt at analysere og redegøre for hvorfor vi har valgt dette design. I softwaren har vi særlig fokus på, at gøre det let for slutbrugeren at navigere rundt i. Derfor har vi lavet store ikoner og menuer. Menuerne er følgende:

- I. *Dashboard*
- II. *Tilbud*
- III. *Afhentning*
- IV. *Betalingsindstillinger*
- V. *Log ud*

Vi vil stadigvæk holde os til det samme design som loginet, da hele software skal passe med de samme farver. I dashbordet har vi givet mulighed for at slutbrugeren kan danne sig et overblik over, de nedenstående punkter som er:

- I. Antal tilbud solgte
- II. Antal kg madspild
- III. Antal CO2 besparet
- IV. Antal tilbud oprettet

På baggrund af brugervenligheden synes vi, det er betydeligt at slutbrugeren har alle informationer ved sin hånd. Længere nede i menuen "dashboard", har vi lavet et datagridview for at få overblik over alle de tilbud, de nuværende slutbrugere har oprettet. Statistikkerne har til formål at give slutbrugeren et nemt overblik, over overstående punkter. (Se bilag [1.6 Software](#))

9.3.1 TILBUD

Idet Team Nordics ide er "livetilbud", har vi særligt fokus på brugervenligheden i menuen "tilbud". Her har vi tilføjet "opret", "redigere", "slet tilbud" tydeligt for slutbrugeren. Som nævnt bruger vi også datagridview til at fremhæve tilbuddene i Dashboard. (Se i bilag [1.7 Tilbud](#))

I tilbudsmenuen har man 3 funktioner som er:

- I. [Opret tilbud kildekoden](#)

II. [Redigere tilbud kildekoden](#)

III. [Slet tilbud kildeorden](#)

9.3.2 AFHENTNING

I denne menu oplyser slutbrugerne hvor brugerne kan afhente deres tilbud, slutbrugerens adresse og virksomhedens navn, som tydeligt vil fremgå i softwaret oversigt. Designet for Afhentning ses i ([bilag 5.7 AFHENTNING](#))

Koden for afhentning kan ses på ([bilag 3.6 Afhentning kode funktion](#))

9.3.3 DE SMÅ FUNKTIONER

For at gøre det mere brugervenligt for slutbruger, har vi lavet følgende funktioner:

- I. *Opdatering af tabeller*
- II. *Nulstilling af felter*
- III. *Input til redigering*
- IV. *Luk programmet/ Minimere programmet*
- V. *Menuer*
- VI. *Binder database til datagridview*
- VII. *Log ud*

- I. Opdatering af tabellerne sker når slutbrugere har enten redigeret, oprettet eller slettet et tilbud. Opdaterings funktionen står i sin klasse for sig selv.
Se koden for opdatering af tabellen ([bilag 3.2 Opret kode tilbud](#))
- II. Slutbruger vil altid have muligheden for at nulstille tekstboksene. ([bilag 3.7 tekstbokserne](#))
- III. Idet slutbruger klicker på en række i datagridviewet, bliver rækken sendt til redigeringssiden, så slutbruger har muligheden for at redigere den valgte række. (Bilag [3.8 input til redigering](#))
- IV. Luk/minimere programmet. (Bilag [3.9 luk/minimerer programmet](#))
- V. Menuer er med til at gøre det nemt, for slutbruger at navigere rundt i softwaren. ([Bilag 4.0 menuer](#))

VI. Vores database forbundet til datagridview. (bilag [4.1 Forbindelse af database til datagrid](#))

9.3.4 DATABASE TIL SOFTWARET

Idet vi har testet om vores data virker, bliver der lavet en database på Azure cloud. Databasen har til formål at forbinde softwaret og dataene sammen. Vi har to tabeller, en til vores slutbruger ([bilag 4:2 database queries User](#)) og en til vores tilbud ([Bilag 4.3 database queries tilbud](#))

9.3.5 OPDATERING AF DASHBOARD STATIKERNE

For at opdatere dashboardets statikkerne, har vi brugt en increment funktion. For hvert tilbud der bliver oprettet, bliver statikkerne incremented. For at opnå dette, har vi brugt design pattens som singleton, som gør det let at få adgang til klassen. ([Bilag 4.4 "inkrementering" ved design patterns](#)) ([Bilag 4.5 "incrementation" af tilbud](#))

9.3.6 LOG UD

I softwaret er der også mulighed for at logge ud ([Bilag 4.6 logud](#))

9.4 HJEMMESIDE

Når slutbrugere opretter et tilbud via vores software, bliver der også oprettet et tilbuddet på hjemmesiden.

Når slutbrugeren downloader softwaret og opretter tilbud, bliver tilbuddet vist på hjemmesiden (Se [bilag 5.6 HJEMMESIDEN](#)). For at få et overblik over hjemmesidens menuer, er de listet nedenfor:

- I. *Tilbud menu*
- II. *Priser*
- III. *Login*
 - I. *Login*
 - II. *Opret bruger*
 - III. *Redigering af personlige data*

IV. Admin

I. Authentication

Til underbygning af hjemmesidens IT-sikkerhed, er der blevet lavet en sikkerheds- og risikoanalyse. Til besvarelse af underspørgsmålet i problemformulering om IT-sikkerhed på hjemmesiden, er der desuden blevet lavet en analyse på hjemmesiden, for at demonstrere hvordan Team Nordic håndterer IT-trusler.

Se ([Punkt Clickjacking](#))

9.4.1 DATA

Med "On fly" data er det mest sikkert at have to databaser. En database til lokale test af diverse input- og output og en online database. Formålet med at have to databaser er, at man kan teste diverse data, uden det har nogen effekt på systemet.

(Morten C.E. Kristensen (2018). Part 7. Lokaliseret den 06-11-2020 på [linket](#))

9.4.2 MENUER

For at følge op på brugervenligheden, er hjemmesiden lavet i asp.core.

Asp.core regenerer vores menuer som login, oprettelse af brugere og hvor brugeren kan redigere deres personlige informationer. De resterende menuer har til formål at vise dataene, samt gøre sikkerheden bedre, da den IT-ansvarlig selv kan tildele adgang til kunderne.

([Bilag 4:7 Hjemmeside Menuer](#))

9.4.3 HTTPS

Da https er en del af sikkerheden på hjemmesiden, har vi valgt at implementeret det.

Https er med til at gøre Team Nordics kunder mere sikre, når de besøger hjemmesiden. Hvis man bruger google Chrome som sin standard browser, vil Chrome også indikere med en lås, at hjemmesiden bruger https. Hvis hjemmesiden er http, vil Chrome indikere at siden ikke bruger https og kan forårsage tab af kunder, da det virker til at være en usikker hjemmeside for brugeren, når de besøger hjemmesiden. De største fordele ved at bruge https over http er, kryptering af personlige data som f.eks. cpr, navn, adresse køn m.m.

Følgende lovgivning om databeskyttelse

" Myndigheder, organisationer og virksomheder, som indsamler personlige oplysninger, skal fra 25. maj 2018 rette sig efter den nye lov. For eksempel skal visse virksomheder, der indsamler personlige oplysninger, bygge databeskyttelse ind i deres it-løsning. Nogle virksomheder skal også ansætte en "databeskyttelsesrådgiver", der skal stå for, at virksomheden indsamler og behandler private oplysninger korrekt efter den nye lov. Datatilsynet fører kontrol med virksomheder og myndigheder for at sikre, at de lever op til de nye regler. Gør de ikke det, kan der gives bøder."

(Astrid Olstrup (24-04-2019. GDPR: Få overblik over databeskyttelsesloven. Lokaliseret den 27.10-2020 på [linket](#)).

Da Team Nordic allerede har med personlig informationer at gøre som Navn og alder, er det vigtigt de implementere forslaget om HTTPS.

For at implementere https er der indbygget https løsning i Azure, hvor man kan indstille hjemmesiden til kun at køre på https (Se ([Bilag 2.3 https](#)))

9.4.4 IMPLEMENTERING AF SECURITY HEADERS

Som en stor del af sikkerheden på hjemmesiden, har vi på nuværende tidspunkt lavet en Securityheaders test, for at se hvilken bogstav Team Nordic fik.

Som tidligere nævnt fik Team Nordic karakteren **F** ([Bilag 2:1 Securityheaders](#)). Efter implementering af koden i vores startup (Anumpam Mati (17-08-2020). Secure Web Application Using HTTP Security Headers In ASP.NET Core. Lokaliseret d. 27-10-2020 på [linket](#)) blev der efterfølgende lavet en scanning af vores hjemmeside, hvor Team Nordic fik karakteren **A**, hvilket er det næst bedste for Securityheaders

(Se [Bilag 2:2 Securityheaders A](#))

Til underbygning af Securityheaders er der lavet følgende:

- I. X-XSS-Protection Header
- II. Content-Security-Policy Header
- III. X-Frame-Options Header
- IV. Strict-Transport-Security Header
- V. Referrer Policy Header
- VI. Securityheaders koden se ([Bilag 4.8 Security headers](#))

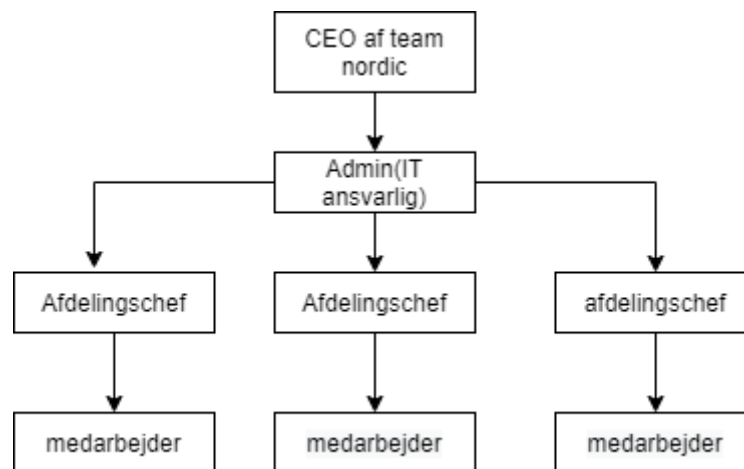
Som beskrevet i [9.4.4 Implementering af Securityheaders](#), har vi nu implementeret de nævnte headers og er derfor klar til at færdiggøre hjemmesiden.

9.4.5 AUTHENTICATION

Som en ekstra sikkerhedsforanstaltning er der som tidligere nævnt, også lavet et virksomhedshierarki. (Se [3.0 Authentication](#))

Authentication er med til at øge sikkerheden i virksomheden.

Dette vises i figur 6.7 nedenfor, hvor man kan se deres ledelsesstruktur for de forskellige afdelinger og hvilket ansvarsområder de forskellige har, når det omhandler hjemmesides IT-sikkerhed.



Figur 6.7 Authentication

Udover at hver afdelingschef skal have rettigheder til noget forskelligt, er det kun Admin der kan give adgang til dem.

Følgende kode demonstrer koden bagved authentication funktionen ([Bilag 4.9 authentication](#)).

Hovedfunktionerne er som følgende

- A. Models
 - a. AdminResponsModel
- B. Admin controller
- C. Views
 - a. Index

Følgende hovedfunktioner er den bagvedliggende kode ved authentication funktionen.

([Bilag 4.10 Model og controller](#)).

10.0 VALG AF TEORI

Følgende kapitel vil danne et grundlag for den valgte teori. Jeg vil komme ind på følgende:

- i. Systemudvikling*
 - I. Unified Process*
 - II. Use cases*
 - III. Fully dressed use cases*
 - IV. Anvendelsesområde*
 - V. Problemområde*
- ii. IT-sikkerhed*
 - i. Governance*
 - ii. ROI beregning*
- iii. Cloud*
 - i. Locks*
 - ii. Connection string*
 - iii. Hosting på Azure*
- iv. Database*
 - i. On fly data*
 - ii. Databasen i cloud*
- v. Hjemmeside*
 - i. Securityheaders*
 - ii. Azure*
 - iii. Asp.core*
 - iv. Software pattens*
- vi. Programmering*
 - i. Validering af kode*

- ii. *UI*
- iii. *Test*
- iv. *Software patterns*
- v. *Forhindring af sql injection*

10.1 SYSTEM UDVIKLING

HVORFOR HAR JEG VALGT UNIFIED PROCESS

Begrundelse for den valgte metode er, at vi vil fokusere på nøglekravene i den tidligere fase, som i vores tilfælde er problemformulering, da Team Nordic både har krav om hjemmeside, software og sikkerheden i projektet.

Unified Process fokuserer på, risikovurdering i starten af hele processen. På nuværende tidspunkt er Covid-19 pandemien stadigvæk i gang, derfor er der mange der arbejder hjemmefra, for at undgå smitterisiko.

Idet vi bruger Unified Process, kan hver udvikler derfor løse deres del af fasen, med skarpt fokus omkring målet og Team Nordics krav.

Som en del af systemudvikling definerer vi tidligt i fasen, kundernes opførelse, klasse og objekter som gør gavn for hele systemets opbygning.

(Ivar, Craig, Kendall (17-06-2018) Unified Process. Lokaliseret den 06-11-2020 på [linket](#))

HVORFOR IKKE WATERFALL UDVIKLING

Med Waterfall udvikling er ulempen, hvis Team Nordic ombestemmer sig, at det er svært at lave om på kravene. Med denne udviklingsmetode kan deadline også kan være længere.

(Pal Kienitz (2017) Business Software Engineering. Lokaliseret den 27-02-2017 på [linket](#))

10.2 IT-SIKKERHED

HVORFOR HAR VI VALGT GOVERNANCE

Begrundelsen for Governance analysen er, at vi kan vise hvor meget skade et angreb kan have på Team Nordic og hvor vigtigt det er, at løse disse problemer hurtigst muligt.

(CIS controller (2019). CIS RAM Express Edition Version 1.0. Lokaliseret den 06-11-2020 på [linket](#))

ROI BEREGNING: Med ROI beregning bliver der analyseret, hvor meget skade et angreb kan have på Team Nordic. Men hvor meget kommer det egentlig til at koste dem, hvis de bliver udsat for sådan et angreb? Begrundelsen for brug af ROI beregning er at sætte det i forretning perspektiv, så Team Nordics ledelse også får en klar forståelse for dette.

CIS CONTROLLER: CIS Controller er en stor del af IT-sikkerhedens analyse. CIS controller er en vejledning for virksomheder, som belyser problemet samt viser hvilke metoder eller hardware, virksomheden kan implementere, for at opnå det bedste resultat.

10.3 CLOUD

HVORFOR HAR VI VALGT CLOUD

Begrundelse for at vælge cloud er sikkerheden, prisen og simpliciteten. Sikkerheden er en del af underspørgsmålene som indebærer følgende:

- i. *Locks*
- ii. *Monitorering*
- iii. *Automatisk opdatering af sikkerheden*
- iv. *Connection string*
- v. *Hosting på Azure*

De overstående punkter er med til at gøre clouden en favorit, da alt fra monitorering, til at gemme sine vigtige "connection string" som forbinder hjemmesiden med databasen. Fordelen er nemlig man har alt det værktøj, et enkelt klik væk. De andre fordele ved clouden er, at man kan tilføje og slette servers som man vil og at prisforskellen mellem fysiske servere og cloud er stor.

(Se emne [8.6 Fordele og ulemper ved cloud scale out og in](#))

HVORFOR IKKE FYSISKE SERVERS

Ulempen ved fysiske servere er, at investering i infrastruktur og hardwaren er meget omkostningsfuldt og pladsen det kræver, at have disse stående. En stor ulempe er også, at man ikke ved hvor meget data man mister, hvis et angreb skulle ske.

(Blog Posts (20-05-2018) THE PROS AND CONS OF CLOUD VS ON PREMISE SERVERS. Lokaliseret den 16-11-2020 på [linket](#))

10.4 HJEMMESIDE

HVORFOR HAR VI VALGT AT BRUGE SECURITYHEADERS

Securityheaders er en god start, på en god IT-sikkerhed. Man kan scanne sin egen hjemmeside på Securityheaders og derefter få et resultat på, hvad præcis hjemmesiden er sårbar overfor.

Securityheaders hjælper nemlig virksomhederne med hvilket sikkerhed der er brug for, for at hjemmesiden bliver sikker.

HVORFOR HAR VI VALGT ASP.CORE.

Begrundelse for at vælge asp.core er, at den mulighed for at lave moderate ui implementering. I dag er design det første man får øje på, når man besøger en hjemmeside.

Asp. core er cross-platform, dvs. enheder som smartphones, tablets, Mac computer m.m.

Dette er en stor fordel for Team Nordic, da både designet af vores to løsninger samt brugervenligheden er et krav.

HVORFOR BRUG AF MVC PATTEN

Model view controller (MVC) er et patten, der hjælper udviklere med at udvikle hurtigere webapplikationer. MVC er med til at gøre udviklingsmetoden lettere for udvikleren, da en udvikler arbejder med f.eks. "view" og en anden kan arbejde med logikken af systemet.

I følgende statikkerne kan MVC patten hjælpe virksomhederne til, at udvikle deres webapplikationer, tre gange hurtigere. Hvis man kigger på Team Nordics brugere, vil de hurtigt opleve, at url af Team Nordics hjemmeside er brugervenlig. En url kan f.eks. være teamnordic.dk/bruger/indhold.

Rent teknisk er fejlfinding af kode gjort nemt at finde. En af de sidste fordele er nemlig, at hvis man som udvikler skifter en del af koden, vil det ikke påvirke resten af hjemmesiden.

(David Mburu (15-11-2017). What are the pros and cons of the MVC pattern? Lokaliseret d. 02-11-2020 på [linket](#))

HVORFOR BRUG AF ASP.NET CORE IDENTITY

Vi bruger vores authentication til at give Team Nordic adgang til vores funktioner, ved at sætte virksomheden op i et hierarki. Når man gør dette, opretter den automatisk også databasen for

vores bruger ([Bilag 5.0 Database rolles](#)). Følgende asp.core Identity bruger PBKDF2 til at "hashed" kodeord i databasen. Dette er et lovkrav (Deloitte (2019). GDPR Top 10: #8 Pseudonymisering og dets anvendelse ved profilering_. Lokaliseret den 06-11-2020 på [linket](#)).

Vi har derfor brugt "identity" framework" til at oprette databasen med "hashed" password, roller, og personlige informationer.

10.5 SOFTWARE

HVORFOR HAR VI VALGT VALIDERING AF KODE

Grunden til vi har valgt validering af vores kode er, for at undgå at programmet ikke lukker ned, efter en kunde f.eks. ikke har indtastet det rigtige format af den valgte tekstboks.

Med vores labels har vi indikeret hvad kunden har skrevet forkert.

HVORFOR HAR VI VALGT BUNIFU (UI)

Grunden til jeg valgte Bunifu ui er, fordi det gør Team Nordics ui mere brugervenligt, som er en del af problemformulerings underspørgsmål og her er Bunifu perfekt til dette.

HVORDAN HAR VI TESTET

Vi har brugt fully dressed use cases til at danne et overblik over, hvordan vores system skal reagere. Vi har derefter gjort dette ved at "debugge" og teste negative og positive data på hjemmesiden og programmet, for at opnå det bedste resultat indenfor testning af de nævnte dele af løsningen.

HVILKEN SOFTWARE PATTERNS HAR JEG VALGT OG HVORFOR

Jeg har valgt at bruge Singleton til løsning af inkrementering. Hver gang slutbrugeren opretter tilbud, tæller funktionen det og sætter tallet ind i dashboard statikkerne. Begrundelse for at bruge Singleton er også fordi, den tillader kun en instans af en givende klasse og den er nem at få adgang til.

HVORDAN HAR VI FORHINDRET SQL INJECTION

Vi har forhindret sql injection ved ikke at sætte vores tekstbokse ind i sql query, men tildel dem et navn og validere dem efterfølgende. Følgende eksempel demonstrerer hvordan en sql query bliver

sårbar over for sql injection ([Bilag 5:1 sql injection sårbarhed](#)). (Paul Litwin (10-11-2019) Stop SQL Injection Attacks Before They Stop You. Lokaliseret den 06-10-2020 på [linket](#))

Hvorimod programmet "connection string" er uden tekstbokse i sql query som ses i ([Bilag 5:2 anti sql injection](#))

11.0 KONKLUSION

Målet med mit projekt var at finde ud af om man kunne bruge IT som et værktøj til at bekæmpe madspild, med en løsning der indebærer en hjemmeside og et program. For at nå målet har det være nødvendigt at beskrive hele processen fra ide til produkt, ved hjælp af den viden der er beskrevet i bilagene samt viden fra tidligere fag. Som et formål analyserede jeg først problemformulering, ved hjælp af viden fra systemudvikling, hvorpå jeg kom med analysering af underspørgsmålene, som indebar sikkerheden i løsning. Ved hjælp af CIS kontrollerne og Governance kunne jeg beskrive og besvare problemformulerings spørgsmål om hvordan jeg bedst it-sikre min løsning på både hjemmesiden og softwaret.

Jeg har på baggrund af dette, kommet med en prototype løsning til reducere af madspild. Derfor kan jeg nu konkludere, at løsning vil komme Team Nordics kunder til gode og give dem en brugervenlig løsning, som alle har mulighed for at bruge.

Team Nordic har nemlig en sikker og brugervenlig løsning, der vil gavne deres renommé. Udover brugervenligheden og renomméet, har Team Nordic også en komplet løsning til et stort problem. Jeg mener det er vigtigt at skabe sikkerhedsrammer for ideen og evt. komme med nogle eksempler fra den virkelige verden.

Med fokus på problemformulering hovedspørgsmål kan jeg konkludere at vores it-løsning til Team Nordic kan komme andre virksomheder til gode, ved at benytte denne løsning til reducere af madspild. Da den fokuserer på brugervenligheden og er it-sikker for alle typer virksomhed. Herefter vil jeg konkludere problemformulerings underspørgsmål.

Følgende hjemmeside vil reducere madspildet, ved at brugeren går ind på hjemmesiden og køber tilbuddene. Jeg har også sikret hjemmesiden ved hjælp af securityheaders og ved hjælp af rolle funktion, der hjælper Team Nordic med at begrænse adgangen, til deres kommende medarbejder. Derefter blev der oprettet software med fokus på brugervenligheden og sikkerheden. Sikkerheden i

software er validering af tekstbokse samt slutbrugerendes input, for at reducere fejl og herved gøre at softwaret ikke lukker ned ved forkert indtastning.

Til sidst vil jeg konkludere, at jeg har beskrevet og analyseret hvordan vi som konsulentfirma benytter cloudens sikkerhedsværktøj, som connection string, https, locks m.m. til at it-sikre Team Nordics løsning på clouden, samt fordele og ulemperne ved brug af cloud. Herfra fordele ved hosting på clouden.

Udover problemformulerings spørgsmål mener jeg også, det er vigtigt at beskrive et detaljeret cloudforslag for Team Nordic, for netop at belyse enkelheden ved dette, men også for at belyse prisen og sikkerheden bagved clouden, da de ingen it-erfaring har.

12.0 PERSPEKTIVERING

Denne opgave bør ses som en opfordring til alle de butikker, restauranter og tankstationer der har med madspild at gøre. Emnet madspild får mere og mere fokus samt hvordan hver virksomhed kan være med til at reducere deres madspild.

For at afklare hvilken udviklingsmuligheder der er, har jeg valgt at beskriv følgende punkter

- i. Kategorier
- ii. Application til Crossplatform
- iii. Beregninger

På hjemmesiden ville det være et nøglekrav, at kategorierne er blevet lavet, da det er vigtigt at kunne klikke på de valgte kategorier, som brugerne søger i. I dag bliver smartphones brugt til alt. Det vil derfor være vigtigt at implementere crossplatform i web funktion og samtidig lave en evt. app til programmet. Så kan alle virksomheder der har en løsningen fra Team Nordics, bruge andre enheder og ikke nødvendigvis skal have en computer for at oprette tilbuddet.

Kategorierne i softwaret vil også implementeres, da der er forskel på produkterne, som folk opretter og dette kan også bruges til beregning af statikerne på softwares dashboard.

Beregning er en ekstra menu på hjemmesiden, hvor brugeren kan beregne hvor meget madspild de har opnået. Denne beregning har til formål at give dem resultater efter hvilken kategori af madspild de har.

12.1 BLEV IKKE LAVET

Følgende ting som ikke blevet lavet

- i. Software statikerne
- ii. Hjemmeside: Visning af tilbud
- iii. Hjemmeside: Pris siden
- iv. Hjemmeside: Download af softwaret
- v. Design til hjemmeside

13.0 REFLEKSION

Jeg synes at hele rejsen om at skabe et produkt for madspild, har været meget tankevækkende. At sætte sig ind i analyser og statikerne for at opnå et resultat, hvorpå man finder sårbarheden i produktet og prøver at eliminere eller formindske den, har været meget interessant.

Jeg synes også at viden fra de tidligere fag har gjort mig rustet til at analysere og grave dybere ned i teorien, for at kunne besvare de konflikter der opstod.

Jeg synes på nuværende tidspunkt min opgaveprioritering ikke har haft indflydelse på kvaliteten af slutproduktet, da jeg mener at jeg har prioriteret tingene efter min egne behov. Derimod har tidsstyring haft en positiv på mit slutprodukt, da flere ideer hurtigt kom til.

14.0 LITTERATURLISTE

Wikipedia Unified Process 2019[Online]

Tilgængelig på: https://da.wikipedia.org/wiki/Unified_Process

[Senest hentet eller vist den 05-10-2020].

Unified Process Smartlearning 2017[Online]

Tilgængelig på

https://studie.smartlearning.dk/pluginfile.php/613076/mod_resource/content/6/2%20UP%20Inception.pdf

[Senest hentet eller vist den 03-03-2019].

Anvendelsesområde OOA&D 122S. [Bog]

Website Hacking – The Most Common Techniques 2018[Online]

Tilgængelig på: <https://blog.threatpress.com/website-hacking-common-techniques/>

[Senest hentet eller vist den 09-11-2020].

Content Security Policy - An Introduction 2014 [Online]

Tilgængelig på: <https://scotthelme.co.uk/content-security-policy-an-introduction/>

[Senest hentet eller vist den 30-10-2020].

Hardening your HTTP response headers 2015 [Online]

Tilgængelig på: <https://scotthelme.co.uk/hardening-your-http-response-headers/#x-frame-options>

[Senest hentet eller vist den 30-10-2020].

A new security header: Referrer Policy 2017 [Online]

Tilgængelig på: <https://scotthelme.co.uk/a-new-security-header-referrer-policy/>

[Senest hentet eller vist den 30-10-2020].

CIS-RAM-Version-.1.0 2015 s.12 2015 (pdf)

Tilgængelig på: <https://learn.cisecurity.org/cis-ram>

[Senest hentet eller vist den 30-10-2020].

CIS-RAM-Workbook-Version-1.0 2015 [pdf]

Tilgængelig på: <https://learn.cisecurity.org/cis-ram>

[Senest hentet eller vist 24-10-2020].

CIS-Controls-Measures-and-Metrics-for-Version-7-cc-FINAL 2015 [pdf]

Tilgængelig på: <https://learn.cisecurity.org/cis-ram>

[Senest hentet eller vist den 25-10-2020].

A CISOs Guide to Bolstering Cybersecurity Posture s8 2015 [pdf]

Tilgængelig på: <https://learn.cisecurity.org/cis-ram>

[Senest hentet eller vist den 24-10-2020].

Videre IT-sikkerhed: Part 4 2018 [Online]

Tilgængelig på: <https://studie.smartlearning.dk/mod/resource/view.php?id=435423>

[Senest hentet eller vist den 06-11-2020].

Cloud-security-ip-whitelisting 2019 [Online]

Tilgængelig på: <https://www.perimeter81.com/blog/cloud/cloud-security-ip-whitelisting/>

[Senest hentet eller vist den 04-10-2020].

Cloud Vs On Premise Software: Which is Best For Your Business 2018 [Online]

Tilgængelig på: <https://www.xperience-group.com/cloud-vs-on-premise-software/>

[Senest hentet eller vist den 05-11-2020].

Smartlearning cloud : Opret Function [Online Video] uge 2

Tilgængelig på: <https://studie.smartlearning.dk/mod/resource/view.php?id=386487>

[Senest hentet eller vist den 23-10-2019].

Cloud Scalability: Scale Up vs Scale Out 2020 [Online]

Tilgængelig på: <https://blog.turbonomic.com/blog/on-technology/cloud-scalability-scale-vs-scale>

[Senest hentet eller vist den 05-11-2020].

Pros and Cons of Cloud Storage. 2020 [Online]

Tilgængelig på: <https://www.securestorageservices.co.uk/article/11/pros-and-cons-of-cloud-storage>

[Senest hentet eller vist den 05-11-2020].

Cloud Storage Pros and Cons 2018 [Online]

Tilgængelig på: <https://www.enterprisestorageforum.com/cloud-storage/cloud-storage-pros-and-cons.html>

[Senest hentet eller vist den 12-10-2020]

6 Pros and Cons of Cloud Storage for Business. [Online]

Tilgængelig på: <https://www.comparethecloud.net/articles/6-pros-and-cons-of-cloud-storage-for-business/>

[Senest hentet eller vist den 27-10-2020].

Træt af it-monolitten? Prøv en microservices.[Online]

Tilgængelig på: <https://www.version2.dk/artikel/traet-it-monolitten-proev-microservice-1070559>

[Senest hentet eller vist den 06-11-2020].

Morten C.E. Kristensen. Part 7 2018 [Online]

Tilgængelig på: <https://studie.smartlearning.dk/mod/resource/view.php?id=435445>

[Senest hentet eller vist den 06-11-2020].

GDPR: Få overblik over databeskyttelsesloven.2019 [Online]

Tilgængelig på: <https://taenk.dk/test-og-forbrugerliv/digitale-tjenester/hvad-er-gdpr-databeskyttelsesloven>

[Senest hentet eller vist den 27.10-2020].

Secure Web Application Using HTTP Security Headers In ASP.NET Core

Tilgængelig på: <https://www.c-sharpcorner.com/article/secure-web-application-using-http-security-headers-in-asp-net-core/>

[Senest hentet eller vist den 27-10-2020]

CIS RAM Express Edition Version 1.0 2019 (Online)

Tilgængelig på: <https://www.cisecurity.org/controls/>

[Senest hentet eller vist den 28 04 2020].

Business Software Engineering. 2017 (Online)

Tilgængelig på: <https://www.dcssoftware.com/pros-cons-waterfall-software-development/>

[Senest hentet eller vist den 27-02-2017].

THE PROS AND CONS OF CLOUD VS ON PREMISE SERVERS 2018 (Online)

Tilgængelig på: <https://www.commeg.com/2018/05/20/the-pros-and-cons-of-cloud-vs-in-house-servers/>

[Senest hentet eller vist den 16-11-2020].

What are the pros and cons of the MVC pattern? 2017 (Online)

Tilgængelig på: <https://www.quora.com/What-are-the-pros-and-cons-of-the-MVC-pattern>

[Senest hentet eller vist den 02-11-2020].

Pseudonymisering og dets anvendelse ved profilering_. Deloitte 2019 (Online)

Tilgængelig på: <https://cyber.deloitte.dk/artikler/gdpr-top-10-8-pseudonymisering-og-dets-anvendelse-ved-profilering/>

[Senest hentet eller vist den 06-11-2020].

15.0 BILAG

BILAG 1.0 PROBLEMOMRÅDE KUNDER

Objekter

Properties

- Christian Nicolau
- Kodeord123
- Christian.a.nicolau1@hotmail.com
- Mand

Behaviour

- Log ind
- Log ud
- Køb tilbud
- Afhent tilbuddet

BILAG 1:1 PROBLEMOMRÅDE SLUTBRUGER

Objekter

Properties

- e. Netto
- f. Søborg hovedgade 56
- g. NettoMadspild@netto.dk
- h. NettosKodeord123

Behaviour

- i. Køb af pris pakker
- j. Log ud / log ind
- k. Download software

BILAG 1:2 PROBLEMOMRÅDE ADMIN

Objekter

Properties

- Admin
- oOrxO\$#est98
- Christian.a.nicolau1@hotmail.com

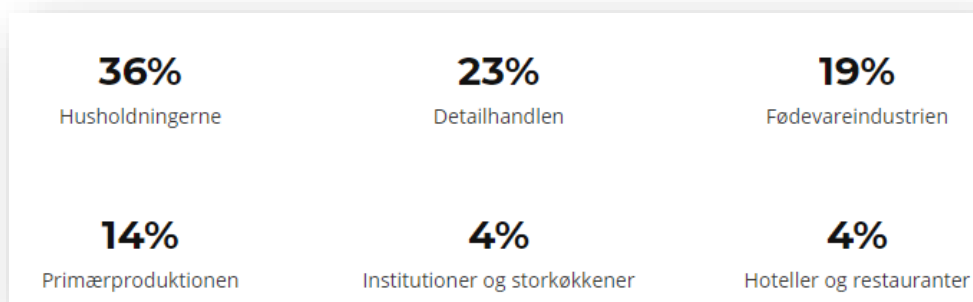
Behaviour

- Log ind
- Log ud
- Rolle-base hierarki

BILAG 1:3 MADSPILD

Husholdningernes	260.000 tons om året
Servicesektoren	227.000 tons om året
Detailhandlen	163.000 tons om året
Hoteller & restauranter	29.000 tons om året
Primærproduktionen	100.000 tons om året
Fødevarerindustrien	133.000 tons om året

BILAG 1:4 PROCENT MADSPILD



BILAG 1:5 LOGIN



— X

TEAM NORDIC

**STOP
MADSPILD**

admin

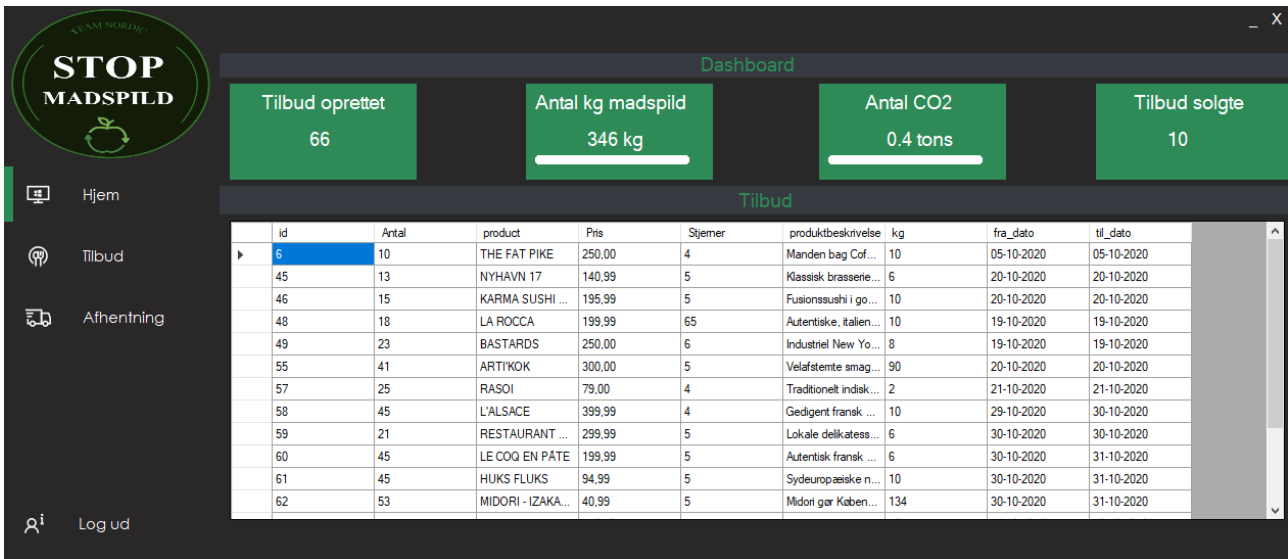
.....

Glemte din kode?

Log ind

Nulstill

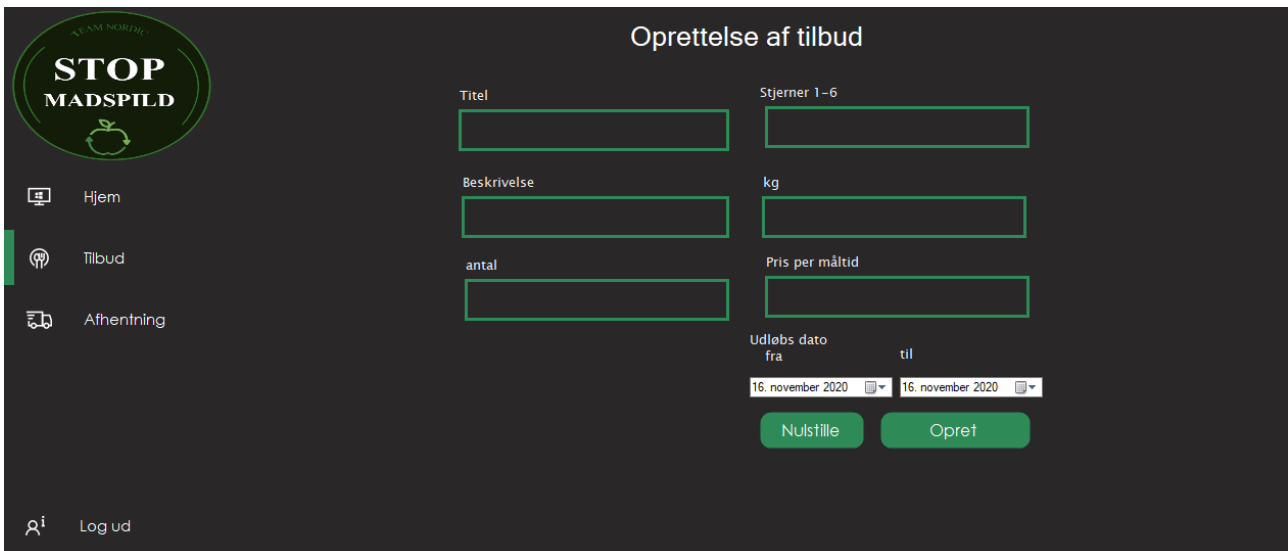
BILAG 1:6 SOFTWARE



The screenshot shows the 'STOP MADSPILD' dashboard. On the left is a sidebar with navigation links: 'Hjem', 'Tilbud', and 'Afhentning'. The main area is titled 'Dashboard' and features four green summary cards: 'Tilbud oprettet: 66', 'Antal kg madspild: 346 kg' (with a progress bar), 'Antal CO2: 0.4 tons' (with a progress bar), and 'Tilbud solgte: 10'. Below these is a 'Tilbud' table with columns for id, Antal, product, Pris, Stjerner, produktbeskrivelse, kg, fra_dato, and til_dato. The table lists various food items and their details.

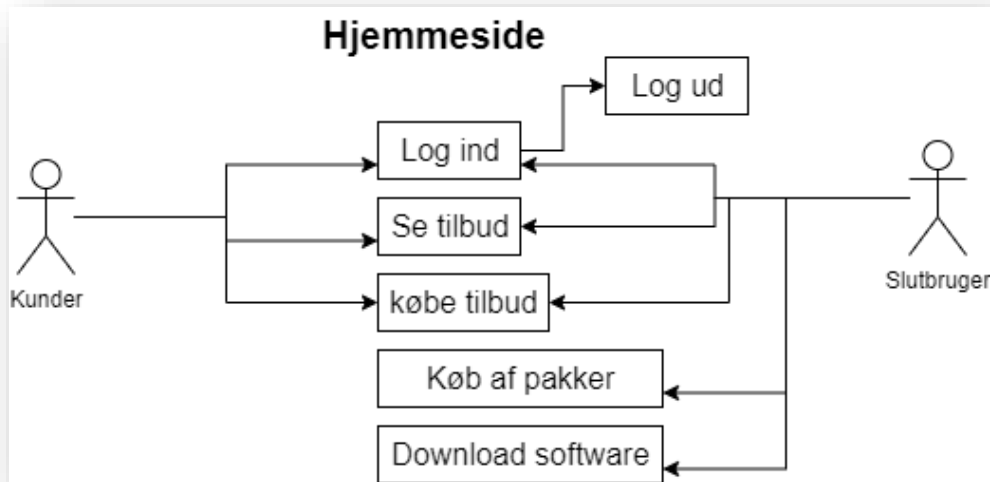
	id	Antal	product	Pris	Stjerner	produktbeskrivelse	kg	fra_dato	til_dato
▶	6	10	THE FAT PIKE	250.00	4	Manden bag Cof...	10	05-10-2020	05-10-2020
	45	13	NYHAVN 17	140.99	5	Klassisk brasserie...	6	20-10-2020	20-10-2020
	46	15	KARMA SUSHI ...	195.99	5	Fusionssushi i go...	10	20-10-2020	20-10-2020
	48	18	LA ROCCA	199.99	65	Autentiske, italien...	10	19-10-2020	19-10-2020
	49	23	BASTARDS	250.00	6	Industriel New Yo...	8	19-10-2020	19-10-2020
	55	41	ARTIKOK	300.00	5	Velafstemte smag...	90	20-10-2020	20-10-2020
	57	25	RASOI	79.00	4	Traditionelt indisk...	2	21-10-2020	21-10-2020
	58	45	L'ALSACE	399.99	4	Gedigent fransk ...	10	29-10-2020	30-10-2020
	59	21	RESTAURANT ...	299.99	5	Lokale delikatess...	6	30-10-2020	30-10-2020
	60	45	LE COQ EN PATE	199.99	5	Autentisk fransk ...	6	30-10-2020	31-10-2020
	61	45	HUKS FLUKS	94.99	5	Sydeuropæiske n...	10	30-10-2020	31-10-2020
	62	53	MIDORI - IZAKA...	40.99	5	Midori gør Køben...	134	30-10-2020	31-10-2020

BILAG 1:7 TILBUD

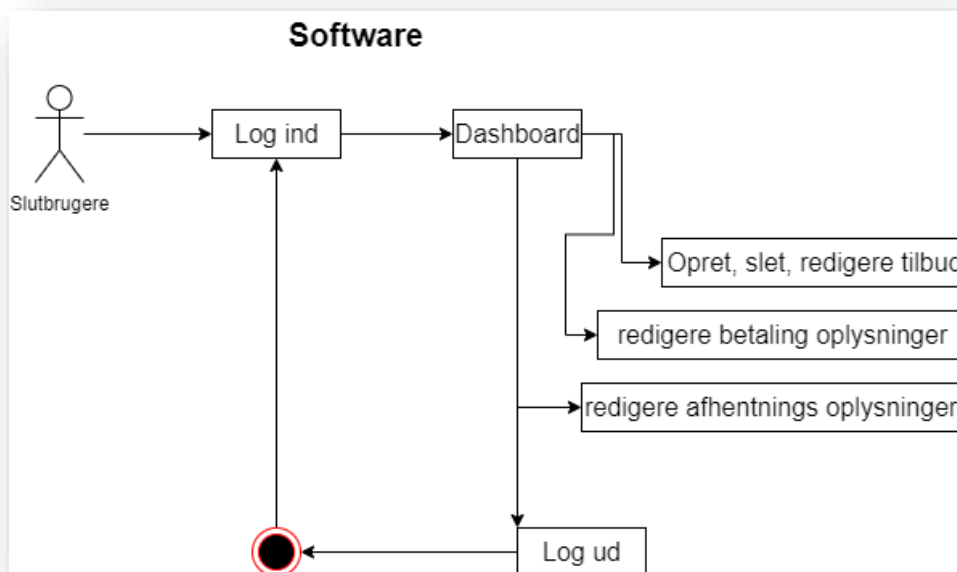


The screenshot shows the 'Oprettelse af tilbud' (Create Offer) form. It includes input fields for 'Titel', 'Beskrivelse', 'antal', 'Stjerner 1-6', 'kg', and 'Pris per måltid'. There are also date pickers for 'Udløbs dato fra' and 'til', both set to '16. november 2020'. At the bottom are 'Nulstil' and 'Opret' buttons.

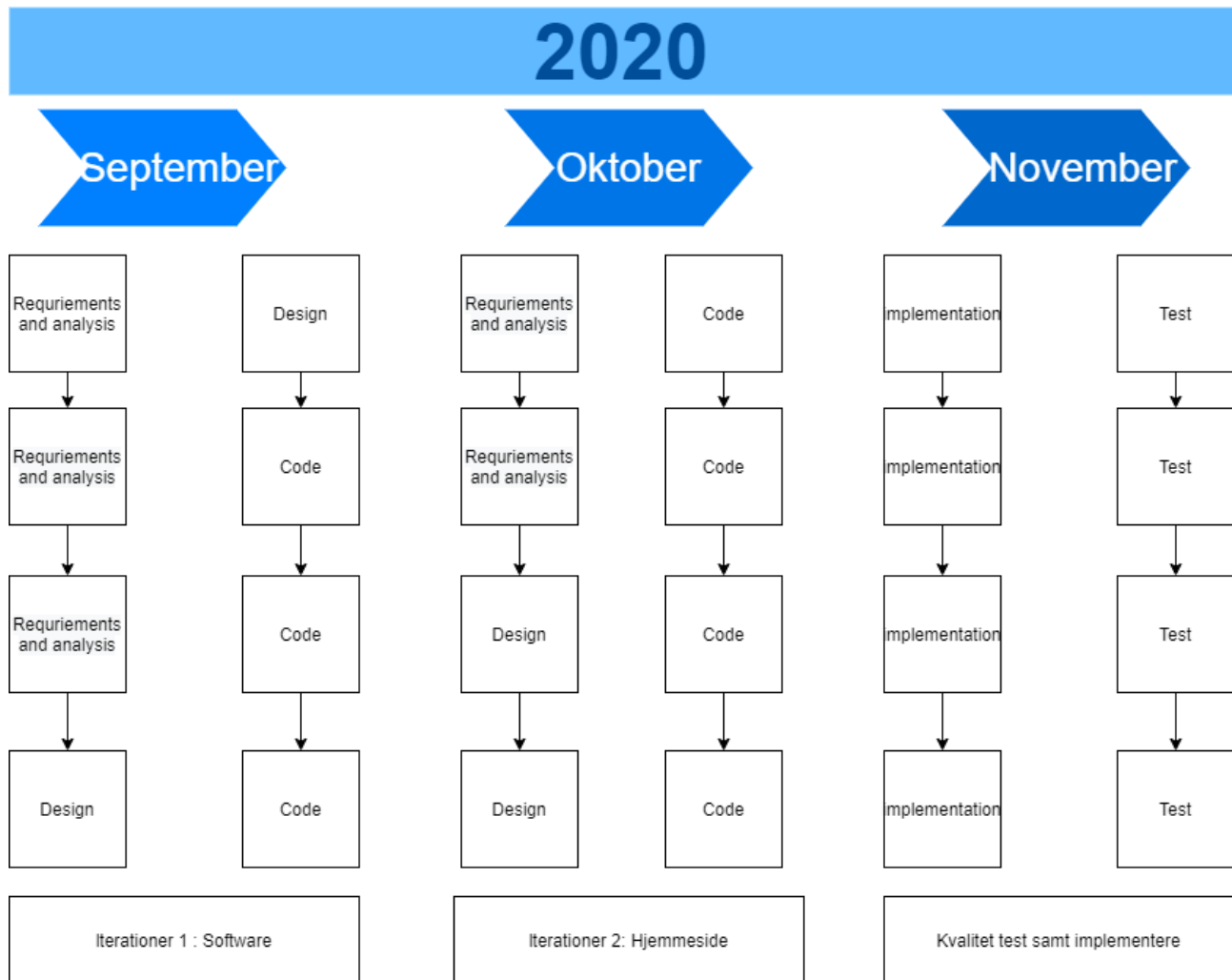
BILAG 1:7 USE CASE HJEMMESIDE



BILAG 1:8 USE CASE SOFTWARE



BILAG 1.9 PROJEKT PLAN




BILAG 2:0 VALIDERING/LOGIN AF SQL INJECTION

```
SqlConnection myConnection = default(SqlConnection);
myConnection = new SqlConnection(cs);

SqlCommand myCommand = default(SqlCommand);
//Sql queries
myCommand = new SqlCommand("SELECT LoginName,FirstName FROM [User] WHERE LoginName = @Username AND FirstName = @Password", myConnection);
//Tildeler værdien til en variable for at undgå sql injection
SqlParameter uName = new SqlParameter("@Username", SqlDbType.VarChar);
SqlParameter uPassword = new SqlParameter("@Password", SqlDbType.VarChar);
//Tildeler variablerne til tekstbokserne
uName.Value = account_txt.Text;
uPassword.Value = password_txt.Text;
myCommand.Parameters.Add(uName);
myCommand.Parameters.Add(uPassword);
```


BILAG 2:1 SECURITYHEADERS



Site:	http://nordicstream.dk/ - (Scan again over https)
IP Address:	2a02:2350:5:106:80fd:6453:49d7:6c50
Report Time:	23 Sep 2020 09:52:47 UTC
Headers:	<div>✗ Content-Security-Policy ✗ X-Frame-Options ✗ X-Content-Type-Options ✗ Referrer-Policy ✗ Permissions-Policy</div>
Warning:	Grade capped at A, please see warnings below.


BILAG 2:2 SECURITYHEADERS A

Security Report Summary



Site:	https://afgangsprojektweb.azurewebsites.net/
IP Address:	23.97.214.177
Report Time:	27 Oct 2020 11:21:38 UTC
Headers:	<div>✔ Content-Security-Policy ✔ Strict-Transport-Security ✔ X-Content-Type-Options ✔ Referrer-Policy ✔ X-Frame-Options ✗ Permissions-Policy</div>

BILAG 2:3 HTTPS



Protocol Settings

Protocol settings are global and apply to all bindings defined by your app.

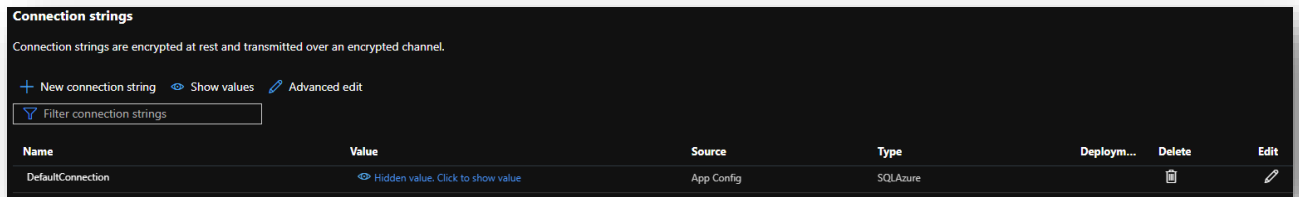
HTTPS Only: ⓘ

Off On

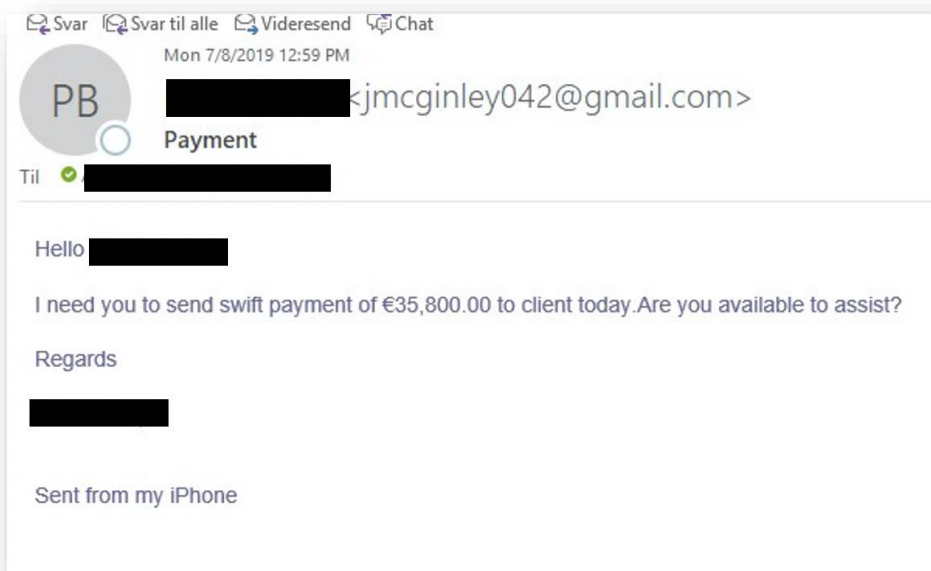
Minimum TLS Version ⓘ

1.0 1.1 1.2

BILAG 2:4 CONNECTION STRING



BILAG 2:5 PHISHING



BILAG 3:0 LOGIN KODE

```
1 reference
private void login_btn_Click_1(object sender, EventArgs e)
{
    //validering om slutbrugeren skriver eller ej i tekstboksene
    if (account_txt.Text == "")
    {
        MessageBox.Show("Please enter user name", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        account_txt.Focus();
        return;
    }
    if (password_txt.Text == "")
    {
        MessageBox.Show("Please enter password", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
        password_txt.Focus();
        return;
    }
    //database string / forbinder
    SqlConnection myConnection =
    default(SqlConnection);
    myConnection = new SqlConnection(cs);
    SqlCommand myCommand =
    default(SqlCommand);
    myCommand = new SqlCommand("SELECT LoginName,FirstName FROM [User] WHERE LoginName = @Username AND FirstName = @Password", myConnection);
    SqlParameter uName = new SqlParameter("@Username", SqlDbType.VarChar);
    SqlParameter uPassword = new SqlParameter("@Password", SqlDbType.VarChar);
    //Forbinder data typen med acc & kodeord tekstboks
    uName.Value = account_txt.Text;
    uPassword.Value = password_txt.Text;
    myCommand.Parameters.Add(uName);
    myCommand.Parameters.Add(uPassword);
    myCommand.Connection.Open();
    SqlDataReader myReader = myCommand.ExecuteReader(CommandBehavior.CloseConnection);
    if (myReader.Read() == true)
    {
        //valider om slutbrugeren har indtastet de rigtigt infomationer
        MessageBox.Show("You have logged in successfully " + account_txt.Text);
        //skjule login form
        Form1 frm1 = new Form1();
        main me = new main();
        this.Hide();
        me.ShowDialog();
        this.Close();
        myCommand.Connection.Close();
    }
    else
    {
        MessageBox.Show("Login Failed...Try again !", "Login Denied", MessageBoxButtons.OK, MessageBoxIcon.Error);
        account_txt.Text = "";
        password_txt.Text = "";
        password_txt.Text = "";
        account_txt.Focus();
    }
}
```

BILAG 3:2 OPRET KODE TILBUD

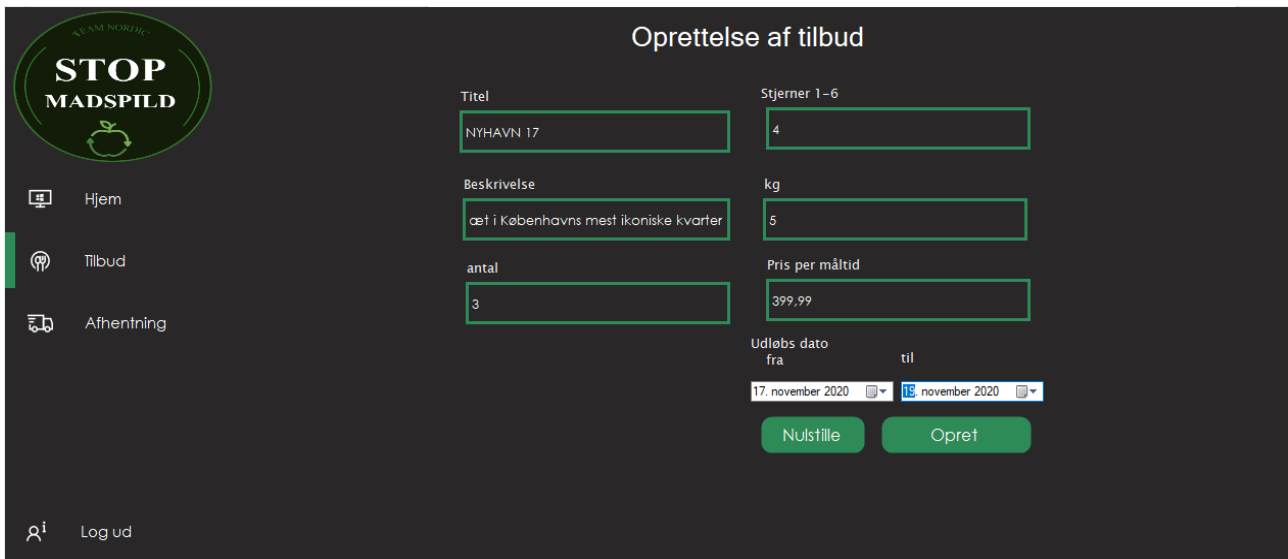
```
private void Opret_tilbud_btn_Click(object sender, EventArgs e)
{
    SqlConnection con = new SqlConnection("Server=tcp:afgangsserver.database.windows.net,1433;Initial " +
    "Catalog=afgangsprojekt;Persist Security Info=False;User ID=Chris;Password=Kaffe123;MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;");
    SqlCommand cmd = new SqlCommand("INSERT INTO produkt (product, produktbeskrivelse, quantity, price, stars, offerstats, kg, " +
    "fra_dato, til_dato, Billede) VALUES (@title, @Beskrivelse, @Antal, @Pris, @stars, @offerS, @kg, @datofra, @datotil, @billede)", con);
    //Validere om slutbrugeren skriver noget eller e
    if (string.IsNullOrEmpty(tilbud_titel.Text))
    {
        Error_titel.Visible = true;
        Error_titel.Text = "Skriv venligt en titel";
        pro.tilbud_titel = "";
    }
    else
    {
        Error_titel.Visible = false;
        cmd.Parameters.AddWithValue("@title", tilbud_titel.Text);
    }
    if (string.IsNullOrEmpty(tilbud_beskrivelse.Text))
    {
        error_beskrivelse.Visible = true;
        error_beskrivelse.Text = "Skriv venligt en beskrivelse";
        tilbud_beskrivelse.Text = "";
    }
    else
    {
        error_beskrivelse.Visible = false;
        cmd.Parameters.AddWithValue("@Beskrivelse", tilbud_beskrivelse.Text);
    }
    int i;
    //Validere om antal er 0 eller om antal er over 0
    if (int.TryParse(tilbud_antal.Text, out i))
    {
        pro.tilbud_antalID = Convert.ToInt32(tilbud_antal.Text);
        cmd.Parameters.AddWithValue("@Antal", (pro.tilbud_antalID));
        error_Antal.Visible = false;
    }
    else
    {
        error_Antal.Visible = true;
        error_Antal.Text = "Skriv venligt et tal antallet af måltider";
    }
    //Validere om Pris er 0 eller om pris er over 0
    if (int.TryParse(tilbud_pris.Text, out i))
    {
        pro.tilbud_pris = Convert.ToInt32(tilbud_pris.Text);
        cmd.Parameters.AddWithValue("@Pris", (pro.tilbud_pris));
        error_pris.Visible = false;
    }
    else
    {
        error_pris.Visible = true;
        error_pris.Text = "Skriv venligt et tal antallet af måltider";
    }
    //Validere om Pris er 0 eller om pris er over 0
    if (int.TryParse(tilbud_kg.Text, out i))
    {
        pro.tilbud_kg = Convert.ToInt32(tilbud_kg.Text);
        cmd.Parameters.AddWithValue("@kg", (pro.tilbud_kg));
        error_pris.Visible = false;
    }
    else
    {
        error_pris.Visible = true;
        error_pris.Text = "Skriv venligt et tal antallet af måltider";
    }
    #region Validerer Dato fra
    if (DateTime.Parse(tilbud_fradato.Text) == null)
    {
        error_fradato.Visible = true;
        error_fradato.Text = "Skriv venligt et tal antallet af måltider";
    }
    else
    {
        pro.tilbud_fradato = Convert.ToDateTime(tilbud_fradato.Text);
        cmd.Parameters.AddWithValue("@datotil", (pro.tilbud_fradato));
        error_fradato.Visible = false;
    }
    if (DateTime.Parse(tilbud_tildato.Text) == null)
    {
        error_tildato.Visible = true;
        error_tildato.Text = "Skriv venligt et tal antallet af måltider";
    }
}
```

```
if (int.TryParse(tilbud_pris.Text, out i))
{
    pro.tilbud_pris = Convert.ToInt32(tilbud_pris.Text);
    cmd.Parameters.AddWithValue("@Pris", (pro.tilbud_pris));
    error_pris.Visible = false;
}
else
{
    error_pris.Visible = true;
    error_pris.Text = "Skriv venligt en pris";
}
//Validere om Pris er 0 eller om pris er over 0
if (int.TryParse(tilbud_kg.Text, out i))
{
    pro.tilbud_kg = Convert.ToInt32(tilbud_kg.Text);
    cmd.Parameters.AddWithValue("@kg", (pro.tilbud_kg));
    error_pris.Visible = false;
}
else
{
    error_pris.Visible = true;
    error_pris.Text = "Skriv venligt et antal kg";
}
#region Validerer Dato fra
if (DateTime.Parse(tilbud_fradato.Text) == null)
{
    error_fradato.Visible = true;
    error_fradato.Text = "Vælg en dato fra";
}
else
{
    pro.tilbud_fradato = Convert.ToDateTime(tilbud_fradato.Text);
    cmd.Parameters.AddWithValue("@datotil", (pro.tilbud_fradato));
    error_fradato.Visible = false;
}
if (DateTime.Parse(tilbud_tildato.Text) == null)
{
    error_tildato.Visible = true;
    error_tildato.Text = "Vælg en dato til";
}
else
{
    pro.tilbud_datetil = Convert.ToDateTime(tilbud_tildato.Text);
    cmd.Parameters.AddWithValue("@datofra", (pro.tilbud_datetil));
    error_tildato.Visible = false;
}
}
if (int.TryParse(tilbud_stjerner.Text, out i))
{
    pro.tilbud_stjerner = Convert.ToInt32(tilbud_stjerner.Text);
    cmd.Parameters.AddWithValue("@stars", (pro.tilbud_stjerner));

    int x = 0;

    cmd.Parameters.AddWithValue("@offerS", (x++));
    error_stjerner.Visible = false;
}
else
{
    error_stjerner.Text = "Bedøm din venligst dit mad";
}
con.Open();
int y = cmd.ExecuteNonQuery();
con.Close();
if (y != 0)
{
    MessageBox.Show(i + "Dit tilbud blev oprettet");
    tabControl1.SelectedTab = tabPage1;
}
}
```

BILAG 3:3 REDIGERER TILBUD



STOP MADSPILD

Oprettelse af tilbud

Titel: NYHAVN 17

Beskrivelse: æt i Københavns mest ikoniske kvarter

antal: 3

Stjerner 1-6: 4

kg: 5

Pris per måltid: 399,99

Udløbs dato fra: 17. november 2020 til: 18. november 2020

Nulstil Opret

Hjem Tilbud Afhentning

Log ud

BILAG 3:4 REDIGERE KODE

```
1 reference
private void redigere_gem_Click(object sender, EventArgs e)
{
    if (redigere_titel.Text != "" && redigere_beskriv.Text != "")
    {
        cmd = new SqlCommand("UPDATE produkt set product = @title, produktbeskrivelse = @Beskrivelse," +
            " quantity = @Antal, price = @Pris, stars = @stars, kg = @kg where ID=@id", con);
        con.Open();
        cmd.Parameters.AddWithValue("@id", ID);
        cmd.Parameters.AddWithValue("@title", redigere_titel.Text);
        cmd.Parameters.AddWithValue("@Beskrivelse", redigere_beskriv.Text);
        cmd.Parameters.AddWithValue("@Antal", redigere_antal.Text);
        cmd.Parameters.Add("@Pris", SqlDbType.Decimal).Value = decimal.Parse(redigere_pris.Text);
        cmd.Parameters.AddWithValue("@stars", redigere_stjerner.Text);
        cmd.Parameters.AddWithValue("@kg", redigere_kg.Text);
        cmd.ExecuteNonQuery();
        MessageBox.Show("Redigere gennemført");
        con.Close();
        DisplayData();
        tabControl1.SelectedTab = tabPage1;
        ClearData();
    }
    else
    {
        MessageBox.Show("Vælg venligst en række at redigere");
    }
}
```

BILAG 3:5 SLET FUNKTION KODE

```
1 reference
private void slet_overblik_btn_Click(object sender, EventArgs e)
{
    if (ID != 0)
    {
        cmd = new SqlCommand("delete produkt where id=@id", con);
        con.Open();
        cmd.Parameters.AddWithValue("@id", ID);
        cmd.ExecuteNonQuery();
        con.Close();
        MessageBox.Show("Du har nu slettet dit tilbud");
        DisplayData();
    }
    else
    {
        MessageBox.Show("Vælg venligst en række at slette");
    }
}
```

BILAG 3:6 AFHENTNING KODE FUNKTION

```
1 reference
private void lebtn_opret_Click(object sender, EventArgs e)
{
    if (letxt_brugernavn.Text != "" && letxt_navn.Text != "")
    {
        cmd = new SqlCommand("UPDATE [User] set FirstName = @Name, LastName = @lastname," +
            " adresse = @Adresse, afhenting = @afhent, Mail = @Mail WHERE UserID = 1", con);
        con.Open();
        cmd.Parameters.AddWithValue("@Name", letxt_navn.Text);
        cmd.Parameters.AddWithValue("@lastname", leEfternavn_txt.Text);
        cmd.Parameters.AddWithValue("@Adresse", letxt_adresse.Text);
        cmd.Parameters.AddWithValue("@Mail", letxt_mail.Text);
        if (letxt_ja.Checked)
        {
            cmd.Parameters.AddWithValue("@afhent", 1);
        }
        else if (letxt_nej.Checked)
        {
            cmd.Parameters.AddWithValue("@afhent", 0);
        }
        cmd.ExecuteNonQuery();
        MessageBox.Show("Record Updated Successfully");
        con.Close();
        DisplayData();
        tabControl1.SelectedTab = tabPage1;
    }
    else
    {
        MessageBox.Show("Please Select Record to Update");
    }
}
```

BILAG 3:7 NULSTILLE TEKSTBOKSENE

```
1 reference
private void ClearData()
{
    redigere_titel.Text = "";
    redigere_beskriv.Text = "";
    redigere_antal.Text = "";
    redigere_pris.Text = "";
    redigere_stjerner.Text = "";
    redigere_kg.Text = "";
}
```

BILAG 3:8 INPUT TIL REDIGERING

```
1 reference
private void dataGridView2_RowHeaderMouseClick(object sender, DataGridViewCellEventArgs e)
{
    ID = Convert.ToInt32(dataGridView2.Rows[e.RowIndex].Cells[0].Value.ToString());
    redigere_titel.Text = dataGridView2.Rows[e.RowIndex].Cells[1].Value.ToString();
    redigere_beskriv.Text = dataGridView2.Rows[e.RowIndex].Cells[2].Value.ToString();
    redigere_antal.Text = dataGridView2.Rows[e.RowIndex].Cells[4].Value.ToString();
    redigere_kg.Text = dataGridView2.Rows[e.RowIndex].Cells[7].Value.ToString();
    redigere_stjerner.Text = dataGridView2.Rows[e.RowIndex].Cells[6].Value.ToString();
    redigere_pris.Text = dataGridView2.Rows[e.RowIndex].Cells[5].Value.ToString();
}
```

BILAG 3:9 LUK/MINIMERER PROGRAMMET

```
1 reference
private void App_Close_Click(object sender, EventArgs e)
{
    this.Close();
}
1 reference
private void App_mini_Click(object sender, EventArgs e)
{
    this.WindowState = FormWindowState.Minimized;
}
```

BILAG 4:0 MENUER

```
1 reference
private void Home_btn_Click(object sender, EventArgs e)
{
    tabControl1.SelectedTab = tabPage1;
    Sp1.Visible = true;
    Sp2.Visible = false;
    Sp4.Visible = false;
}

1 reference
private void offer_btn_Click(object sender, EventArgs e)
{
    tabControl1.SelectedTab = tabPage2;
    Sp1.Visible = false;
    Sp2.Visible = true;
    Sp4.Visible = false;
}

1 reference
private void delivery_btn_Click(object sender, EventArgs e)
{
    tabControl1.SelectedTab = tabPage5;
    Sp1.Visible = false;
    Sp2.Visible = false;
    Sp4.Visible = true;

    SqlConnection con1 = new SqlConnection(cs);
    DataTable dt = new DataTable();
    con1.Open();
    SqlDataReader myReader = null;
    SqlCommand myCommand = new SqlCommand("select LoginName,FirstName,LastName,adresse," +
        "afhentning,Mail from [User] where UserID = 1 ", con1);

    myReader = myCommand.ExecuteReader();

    while (myReader.Read())
    {
        letxt_brugernavn.Text = (myReader["LoginName"].ToString());
        letxt_navn.Text = (myReader["FirstName"].ToString());
        leEfternavn_txt.Text = (myReader["LastName"].ToString());
        letxt_mail.Text = (myReader["adresse"].ToString());
        letxt_adresse.Text = (myReader["Mail"].ToString());
        //and whatever you have to retrieve
        bool checkvalue = Convert.ToBoolean(myReader["afhentning"]);
        if(checkvalue == true)
        {
            letxt_ja.Checked = true;
        }
        else if(checkvalue == false)
        {
            letxt_nej.Checked = false;
        }
    }
    con1.Close();
}
```


BILAG 4:1 FORBINDELSE AF DATABASE TIL DATAGRID

```
1 reference
private void main_Load(object sender, EventArgs e)
{
    // TODO: This line of code loads data into the 'afgangsprojektDataSet1.produkt' table. You can move, or remove it, as needed.
    this.produktTableAdapter1.Fill(this.afgangsprojektDataSet1.produkt);
    // TODO: This line of code loads data into the 'afgangsprojektDataSet.produkt' table. You can move, or remove it, as needed.
    this.produktTableAdapter.Fill(this.afgangsprojektDataSet.produkt);
}
```

BILAG 4:2 DATABASE QUERIES USER

```
CREATE TABLE [User] (
    UserID int IDENTITY(1,1) PRIMARY KEY,
    LoginName nvarchar(255) NOT NULL,
    PasswordHash binary(64) not null,
    FirstName nvarchar(40),
    LastName nvarchar(255),
    adresse nvarchar(255),
    afhenting bit,
    Mail nvarchar(255),
);
```

BILAG 4.3 DATABASE QUERIES TILBUD

```
CREATE TABLE produkt (
    id int IDENTITY(1,1) PRIMARY KEY,
    Product nvarchar(255) NOT NULL,
    produktbeskrivelse nvarchar(255) not null,
    quantity int,
    price decimal(18,2),
    stars int,
    offerstats int,
    kg int,
    fra_dato date,
    til_dato date,
);
```


BILAG 4.4 "INCREMENTATION" VED DESIGN PATTERNS

```
class Offerstat
{
    private static int ID = ID;
    private int myId = 0;

    0 references
    public int MyId
    {
        get { return myId; }
    }

    0 references
    public Offerstat()
    {
        ID++;
        this.myId = ID;
    }
}
```

BILAG 4.5 "INCREMENTATION" AF TILBUD

```
//Inkrementer med et for hver tilbud slutkunden laver
Offerstat offer = new Offerstat();
cmd.Parameters.AddWithValue("@offerS", (offer.MyId));
```

BILAG 4.6 LOGUD

```
private void logout_main_Click(object sender, EventArgs e)
{
    MessageBox.Show("Du er nu logget ud ");
    //Skjul formen
    Form1 frm1 = new Form1();
    main me = new main();
    this.Hide();
    frm1.ShowDialog();
    this.Close();
}
```

BILAG 4:7 HJEMMESIDE MENUER

Madspild Tilbud Priser

Register Login

BILAG 4:8 SECURITY HEADERS

```
services.AddCors(options =>
{
    options.AddPolicy("CorsPolicy",
        builder => builder
            .AllowAnyMethod()
            .AllowCredentials()
            .SetIsOriginAllowed((host) => true)
            .AllowAnyHeader());
});

//Strict-Transport-Security Header
services.AddHsts(options =>
{
    options.IncludeSubDomains = true;
    options.MaxAge = TimeSpan.FromDays(365);
});

services.AddRazorPages();
}

//Content security policy implementing
app.Use(async (context, next) =>
{
    context.Response.Headers.Add(
        "Content-Security-Policy",
        "default-src 'self';" +
        "img-src 'self'" +
        "font-src 'self'" +
        "style-src 'self'" +
        "script-src 'self'" +
        "frame-src 'self'" +
        "connect-src 'self'"
    );
    context.Response.Headers.Add("Header-Name", "Header-Value");

    await next();
});

app.UseHttpsRedirection();
app.UseStaticFiles();
// Security - Registered before static files to always set header
app.UseHsts(hsts => hsts.MaxAge(365));
app.UseXContentTypeOptions();
app.UseReferrerPolicy(opts => opts.NoReferrer());

app.UseStaticFiles();

// Security - Registered after static files, for dynamic content.
app.UseXfo(xfo => xfo.Deny());
app.UseRedirectValidation();
app.UseRouting();
```

BILAG 4.9 AUTHENTICATION

```
1 reference
public static class RolesData
{
    private static readonly string[] Roles = new string[] { "Administrator", "økonomi", "Reklame", "slutbruger" };

    1 reference
    public static async Task SeedRoles(IServiceProvider serviceProvider)
    {
        using (var serviceScope = serviceProvider.GetRequiredService<IServiceScopeFactory>().CreateScope())
        {
            var dbContext = serviceScope.ServiceProvider.GetService<Data.DbContext>();

            await dbContext.Database.MigrateAsync();

            var roleManager = serviceScope.ServiceProvider.GetRequiredService<RoleManager<IdentityRole>>();
            var userManager = serviceScope.ServiceProvider.GetRequiredService<UserManager<IdentityUser>>();
            foreach (var role in Roles)
            {
                if (!await roleManager.RoleExistsAsync(role))
                {
                    await roleManager.CreateAsync(new IdentityRole(role));
                }
            }
            //Tilføjer vores rollebeskyttelse og deres email
            var christian = await userManager.FindByEmailAsync("Chris@live.dk");
            var reklamechef = await userManager.FindByEmailAsync("økonomiChef@TeamNordic.dk");
            var medarbejder = await userManager.FindByEmailAsync("ReklameChef@TeamNordic.dk");
            if (christian != null)
            {
                await userManager.AddToRoleAsync(christian, "Administrator");
                await userManager.AddToRoleAsync(reklamechef, "økonomi");
                await userManager.AddToRoleAsync(medarbejder, "Reklame");
            }
        }
    }
}
```

BILAG 4.10 MODEL OG CONTROLLER

```
0 references
public class AdminResponsModel
{
    0 references
    public List<UserViewModel> UserViewModels { get; set; }
}

43
[Authorize(Roles = "Administrator")]
1 reference
public class AdminController : Controller
{
    private readonly DbContext applicationDbContext;
    private readonly RoleManager<IdentityRole> roleManager;
    private readonly UserManager<IdentityUser> userManager;

    0 references
    public AdminController(DbContext applicationDbContext, RoleManager<IdentityRole> roleManager, UserManager<IdentityUser> userManager)
    {
        this.applicationDbContext = applicationDbContext;
        this.roleManager = roleManager;
        this.userManager = userManager;
    }

    0 references
    public async Task<IActionResult> IndexAsync()
    {
        List<UserViewModel> users = await GetAllUsersWithRoles();
        return View(users);
    }

    0 references
    public async Task<IActionResult> Delete(string id)
    {
        IdentityUser identityUser = await userManager.FindByIdAsync(id);

        await userManager.DeleteAsync(identityUser);









        return RedirectToAction("Index");
    }

    0 references
    public async Task<IActionResult> Update(string id)
    {
        IdentityUser identityUser = await userManager.FindByIdAsync(id);

        await userManager.AddToRoleAsync(identityUser, "User");

        return RedirectToAction("Index");
    }
}
```

BILAG 5.0 DATABASE ROLLES

- +  `dbo.__EFMigrationsHistory`
- +  `dbo.AspNetRoleClaims`
- +  `dbo.AspNetRoles`
- +  `dbo.AspNetUserClaims`
- +  `dbo.AspNetUserLogins`
- +  `dbo.AspNetUserRoles`
- +  `dbo.AspNetUsers`
- +  `dbo.AspNetUserTokens`

```
1 reference
public async Task<List<UserViewModel>> GetAllUsersWithRoles()
{
    List<UserViewModel> allUsersWithRoles = new List<UserViewModel>();
    List<IdentityUser> users = userManager.Users.ToList();

    foreach (IdentityUser identityUser in users)
    {
        UserViewModel userViewModel = new UserViewModel();
        userViewModel.IdentityRoles = new List<IdentityRole>();
        userViewModel.IdentityUser = identityUser;

        IList<string> userRoleNames = await userManager.GetRolesAsync(identityUser);

        foreach (string userRoleId in userRoleNames)
        {
            IdentityRole role = roleManager.Roles.Where(r => r.Name == userRoleId).FirstOrDefault();
            userViewModel.IdentityRoles.Add(role);
        }

        allUsersWithRoles.Add(userViewModel);
    }

    return allUsersWithRoles;
}
```

BILAG 5:1 SQL INJECTION SÅRBARHED

```
string strQry = "SELECT Count(*) FROM Users WHERE UserName='" + txtUser.Text + "' AND Password='" + txtPassword.Text + "
```

BILAG 5:2 ANTI SQL INJECTION

```
myCommand = new SqlCommand("SELECT LoginName,FirstName FROM [User] WHERE LoginName = @Username AND FirstName = @Password", myConnection);
SqlParameter uName = new SqlParameter("@Username", SqlDbType.VarChar);
SqlParameter uPassword = new SqlParameter("@Password", SqlDbType.VarChar);
uName.Value = account_txt.Text;
uPassword.Value = password_txt.Text;
myCommand.Parameters.Add(uName);
myCommand.Parameters.Add(uPassword);
```

BILAG 5.3 FULLY DRESSED USE CASE PHISHING

Titel	Hackeren sender en Phishing mail, til en af Team Nordics medarbejder
Beskrivelse	En medarbejder fra Team Nordic tjekker sin mail. Idet hun ser en mail fra hendes chef, hvor der står følgende at hun skal sende 36.000 euro
Primary Actor	Medarbejder
Stakeholders	Hackeren
Main: Negativt scenarie	<ol style="list-style-type: none"> 1. Medarbejder: Åbner sin mailboks for at tjekke mails. Ser efterfølgende en mail fra chefen af firmaet, har skrevet til hende. I mailen står der, at hun skal sende 36.000 euro. 2. Medarbejder: Sender pengene til hackeren med det samme, uden nogle yderligere godkendelser, da mailen kommer direkte fra virksomhedens leder. 3. Hackeren: Har nu modtaget pengene
Titel: Positivt test af Click jacking Extensions 1	<ol style="list-style-type: none"> 1. Medarbejder: Åbner sin mailboks for at tjekke mails. Ser efterfølgende en mail fra chefen af firmaet, har skrevet til hende. I mailen står der, at hun skal sende 36.000 euro. 2. Medarbejder: Har fået øje på at mailen ser lusket ud, vælger derfor at ringe til hendes virksomhedens leder, for at eller be- eller afkræfte mailens originalitet. 3. Hackeren: Hackeren får ikke noget ud af det. da medarbejderen allerede ved det er en Phishing mail.

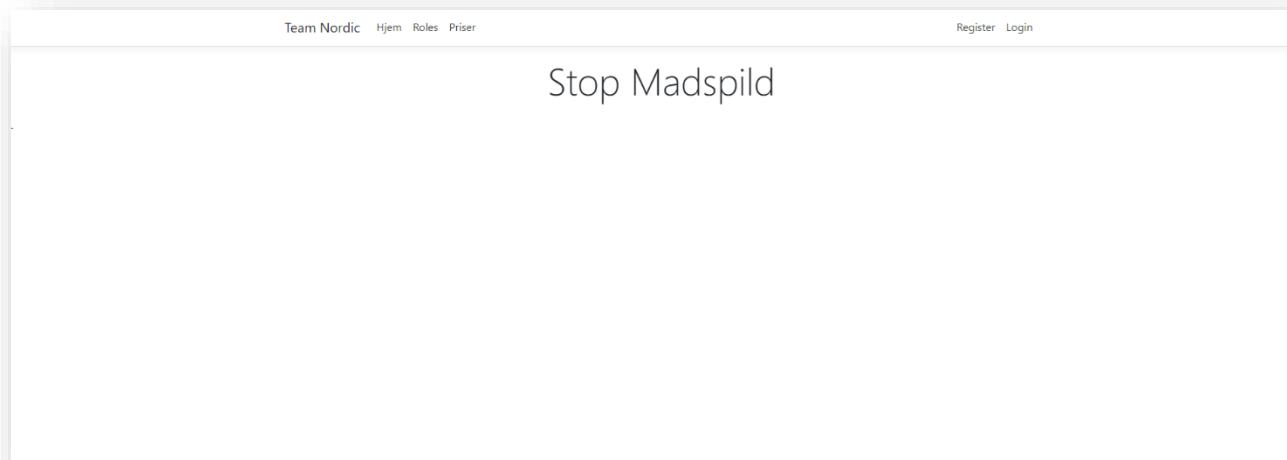
BILAG 5.4 FULLY DRESSED USE CASE SQL INJECTION

Titel	Hackeren vil gerne have <u>udprinte</u> Team Nordics login data ud.
Beskrivelse	Hackeren er inde på login siden i softwaren og er i gang med at udrette en sql injection på login tabel
Primary Actor	Systemet
Stakeholders	Hackeren
Preconditions	Hackeren har allerede adgang til software
Main: Negativt scenarie	<ol style="list-style-type: none"> 1. Hackeren: Udfylder et sql injektion kode i loginboksen, der har til formål at få login informationer fra databasen. 2. Hackeren: Idet hackeren trykker login på softwaret, sker der en <u>udprintning</u> af logininformationerne. 3. System: Alle login informationer ud printes fra databasen. 4. Hackeren: Har nu adgang til slutbrugeren og alle de personlige informationer
Titel: Positivt test af sql injection Extensions 1	<ol style="list-style-type: none"> 1. Hackeren: Udfylder et stykke kode i logintekstboksen, som har til formål at få brugernavnet og kodeordet til en slutbruger. 2. Hackeren: Idet hackeren trykker login på softwaret, sker <u>udprintningen</u>. 3. System: Sker der en validering af brugernavnet og kodeordet. 4. System: Validering blev gemmen ført og login blev annulleret.

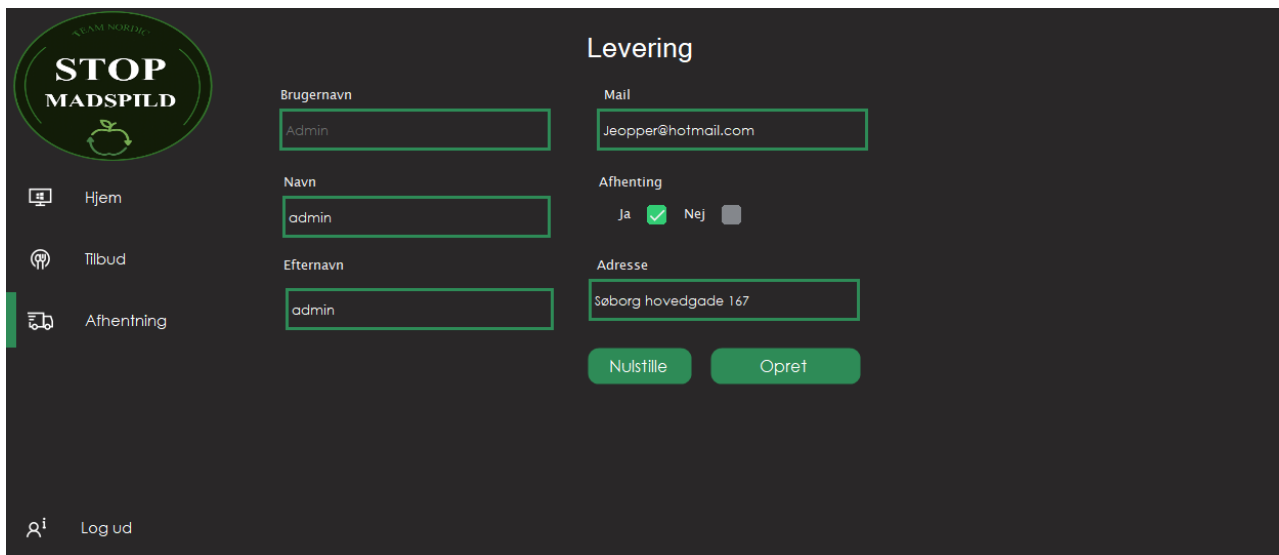
BILAG 5.5 FULLY DRESSED USE CASE CLICKJACKING

Titel	Hackeren vil bruge click jacking til at få brugeren til at logge ind.
Beskrivelse	En kunde af Team Nordic, har gemt sit brugernavn og kodeord i "Google Chrome" browseren og logger ind automatisk på hjemmesiden, uden at skulle udfylde logininformationen.
Primary Actor	Systemet
Stakeholders	Hackeren
Preconditions	Hackeren kender til svagheden på Team Nordics hjemmeside og har <u>click jacking</u> siden, for at lokke brugeren ind.
Main: Negativt scenarie	<ol style="list-style-type: none"> 4. Hackeren: Udfylder et stykke kode i logintekstboksen, der har til formål, at få brugernavnet og kodeordet til en slutbruger. 5. Hackeren: Idet hackeren trykker login på softwaret. 6. System: Sker en udprintning af den nuværende slutbruger i tekstboksen. 7. Hackeren: Har nu adgang til slutbrugeren og alle de personlige informationer.
Titel: Positivt test af Click jacking Extensions 1	<ol style="list-style-type: none"> 4. Hackeren: Udfylder et stykke kode i logintekstboksen der har til formål, at få brugernavnet og kodeord til en slutbruger. 5. Hackeren: Idet hackeren trykker login på softwaret. 6. System: Sker der en validering af brugernavnet og kodeordet. 7. System: Validering blev gemmen ført og login blev annulleret.

BILAG 5.6 HJEMMESIDEN



BILAG 5.7 AFHENTNING

The screenshot shows the 'Levering' (Delivery) registration form on the Stop Madspild website. On the left is a dark sidebar with the 'STOP MADSPILD' logo and a menu with icons and labels: 'Hjem', 'Tilbud', and 'Afhentning' (which is highlighted). At the bottom of the sidebar is a 'Log ud' (Log out) link. The main form area has a title 'Levering' and several input fields: 'Brugernavn' (username) with 'Admin', 'Mail' with 'Jeopper@hotmail.com', 'Navn' (name) with 'admin', 'Efternavn' (surname) with 'admin', and 'Adresse' (address) with 'Søborg hovedgade 167'. There is also a checkbox for 'Afhentning' (pickup) with 'Ja' (checked) and 'Nej' (unchecked). At the bottom of the form are two buttons: 'Nulstil' (Reset) and 'Opret' (Create).