

Forside

Studienummer og navn	S170852 Claus Tetens
Modulnavn	62T04 Webteknologier F17
Underviser	Birger Andersen
Antal anslag incl. mellemrum	57.842
Dato for aflevering	31. maj 2017

Filnavn S170852 Claus Tetens.pdf

voresjazzklub.dk

en studieopgave i webteknologier

Indholdsfortegnelse

Voresjazzklub.dk	1
Målgruppe	1
Indledning.....	1
Problemformulering	1
Afgrænsning	1
Teknologivalg.....	1
Teori	2
MVC som design pattern.....	2
Betragtninger omkring sikkerhed	3
Behov for hashing af passwords	5
Praktisk opgave	6
Et dynamisk website	7
Databasemodel	7
I gang med dette konkrete setup.....	11
Migrering af adgangskontrol fra SQLserver til MySQL	14
Beskyttelse af password.....	15
Bygning af Voresjazzklub.dk – de første skridt	16
Fri leg.....	17
Test/Validering.....	18
Det videre forløb	19
Konklusion	20
Litterarurliste.....	21
Appendix	22
Offentlig og privat nøgle	22
Wireshark output	24
Lidt om, hvordan programkoden bliver til	26
Noget hurtig PHP	27
Noget hurtig ASP.NET	31
Script til migrering af adgangskontrol.....	32
Lidt om Reflection	33
Projektet som Pixiebog	35
Siderne i Voresjazzklub	51
Walk through – a day at Voresjazzklub.....	52

A great many thanks to

google

w3schools

tutorialsteacher

microsoft.

Without these, this wouldn't have happened.

Forord

Engang i en fjern fortid, da jeg var på sidste år i gymnasiet, stod der et fjernsyn og et skrivemaskinetastatur på fysikgangen. Det viste sig at være en adgang til noget edb; ikke noget for mig. Min daværende interesse for elektronik var primært ovre i den analoge afdeling. Det ændredes hurtigt, og blot to år senere havde jeg en hjemmedatamat – sådan hed det altså i gamle dage. Det var en CP/M-maskine, en NewBrain.

Operativsystemet var så begrænset i omfang, at det både kunne læses og forstås. Al kodning var i assembler; og maskinkode, når assembleren ikke kunne følge med længere. Det var sjovt at pille i operativsystemet. Alt var småt, nemt og simpelt.

At begynde fra scratch, hver gang, virkede naturligt. Henad vejen indså jeg fordelene ved genbrug. Genbrug kræver dog, at koden er velfungerende, gennemtest og modstandsdygtig overfor fejlagtig anvendelse.

Men Verden er ikke konstant. Specielt er IT-verdenen ikke konstant.

Jeg tror på, at ting skal testes, og, at det skal være nemt at teste.

Jeg tror på, at det er værd at bygge videre på, hvad andre har lavet.

Jeg tro på, at it-systemer skal bygges, for at blive forandret.

Nye udviklingsværktøjer, gør det muligt at komme i gang på ingen tid. Det går næsten hurtigere at fremstille en "Hallo World"-webapplikation, end at udtales det.

Gode, stabile IDE'er, færdige komponenter og frameworks gør it-udvikling nemmere og hurtigere. Ulempen er, at der er så meget mere, der skal læres, før man rigtigt mestrer produkterne. Tillige skal der stadigt tilegnes ny viden inden for samme rammer, da produkterne udvikles hen over tid, og der kommer nye svar på eksisterende problemstillinger.

Baggrunden for dette projekt er netop, at der er kommet nye produkter og andre er forandrede, og at jeg ikke helt har fulgt med.

Her er en, der ikke har fulgt med tiden. Don't be him!



Claus Tetens, Hillerød, maj 2017.

Voresjazzklub.dk

Målgruppe

Interesserede, der har et beskeden teknisk kendskab til almindelige it-relatedede emner. Det forventes, at udbredte termer som f.eks. krypteret forbindelse og DDoS kendes i et sådant omfang, at dagligdags problemstillinger, der er beskrevet ved hjælp af disse, kan forstås.

Indledning

Blandt en gruppe mennesker, der alle havde lyst til at ses til noget spisning og musik i byen, har der været et behov for at kunne foreslå arrangementer, og for at kunne melde sig til de enkelte begivenheder.

Det har affødt en ide om at konstruere et mindre website til netop det formål.

Projektet er rent faktisk påbegyndt i starten af 2010, men på grund af noget teknisk bøvl, for lidt tid og anvendelse af en alternativ løsning [1], blev projektet droppet efter kun et par dages arbejde.

Siden 2010 er der gået syv år og i IT-sammenhæng er det lang tid, bl.a. eksisterede HTML5 ikke [2], så de tanker, der blev gjort dengang er kasseret og skal up to date.

Projektet er i en størrelse, hvor der kan laves en demo inden for web-teknologier og databaser, men næppe et helt færdigt website.

Problemformulering

Der skal fremstilles et website til administration af arrangementstilmeldinger blandt personer, der er tilmeldt som bruger.

Der skal kunne oprettes arrangementer, og, efter afholdelse, skal det være muligt at uploadne billeder med tilhørende tekst.

Brugerne skal selv kunne tilmelde sig og kunne administrere deres konti.

En administrator skal have udvidet adgang til brugerdata og skal bl.a. kunne nulstille passwords.

Brugeradgange og administratorbeføjelser skal opbevares i en database.

Der skal anvendes nyere teknologier i det omfang det er praktisk muligt.

Det er ikke målet at levere et flyvefærdigt produkt, men et walking skeleton [22].

Afgrænsning

Sikkerhed er en væsentlig del af nutidigt web. Derfor ville det være på sin plads at gennemføre den praktiske implementering underanvendelse af sikret tilgang via HTTPS (SSL eller TLS), men da ikke er gratis [3] at få lagt SSL på sitet, vil den type sikkerhed kun berøres teoretisk indtil en eventuel go-live. Det vil ikke have nogen implikationer for koden på server-siden, da kryptering foregår på socket-niveau

Der er på forhånd valgt et hostingfirma, web10.dk, og en pakke (basic med windowsserver) [6]. Det giver nogle afgrænsninger af de muligheder, der er for at vælge programmeringssprog mv.

Da der er tale om en studieopgave, er processen og læring væsentligere end produktet. En afdækning af, hvordan et delproblem løses vægtere end at få en lækker applikation.

Teknologivalg

Som et forprojekt, skulle det undersøges, hvilke teknologier, der skulle anvendes under fremstilling af websitet.

Forfatteren har allerede arbejdet med Java/JSP og Java/Wicket, så der ville ikke være meget sport i at anvende disse i forbindelse med en studieopgave.

Tilbage stod så PHP, ASP.NET og ASP.NET MVC som de mest oplagte kandidater.

PHP er relativt let tilgængeligt og med en pakke som xampp [15] får man et fuldt integreret og kørende miljø. Det er nemt at komme i gang med, og det tager kun få minutter at lave en dynamisk udgave af "Hallo World". Det er illustreret i appendix "Noget hurtig PHP"

Et af webhostingsfirmaets begrænsninger er, at man kan enten vælge Linux/PHP eller Windows/ASP.NET. Valget var i forvejen faldet på Windows, så PHP er ude.

Forfatteren har et beskeden kendskab til asp.net fra tidligere; dog må det konstateres, at der er sket en udvikling de sidste omrent syv år, hvor asp.net sidst blev anvendt.

MVC som design-pattern er kendt, men ASP.NET MVC som værktøj er nyt. Da en væsentlig del af projektet er læring, vil det være naturligt at kaste sig over dette framework. Første version kom frem for otte, ni år siden, og har undergået en del forandring undervejs.

Google er en kammerat, når det drejer sig om it-udvikling. Der er altid svar på problemstillinger, men i forbindelse med ASP.NET MVC har der været mange fejlskud, fordi produktet har undergået store forandringer gennem årene. Der har flere gange været henvisninger til metodikker, der ganske simpelt ikke anvendes længere, eller ikke kan anvendes mere.

Betydningen af denne udvikling er, at der har været brug for mange små eksperimenter og undersøgelse af, hvad koden rent faktisk gør, for at kunne fremstille en brugbar kode.

Undervejs i projektet har det været nødvendigt at skifte maskine, og da det oprindelige projekt, på grund af de mange eksperimenter, var blevet noget saneringsmodent, var det oplagt at begynde forfra. I rapporten er der både henvisninger til det gamle projekt – VoresJazzklubSomMvc – og det ny, med navnet Voresjazzklub.

Teori

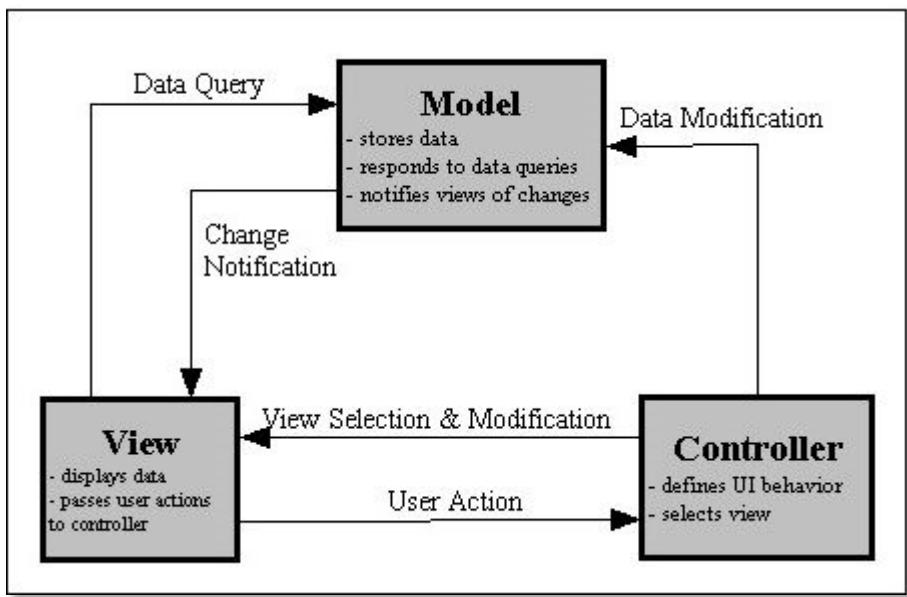
MVC som design pattern

Design patterns er mønstre, der opdages i gode løsninger af forskellige problemstillinger. Hvis man i flere IT-systemer har fundet frem til samme mønster, og det fungerer godt, så kunne der være en anledning til, at det næste projekt, med samme type opgave, også skal bygges efter det fundne mønster.

Et af de udbredte mønstre er MVC. Model – View – Control. Det har bl.a. til formål at dele applikationen i tre områder og afkoble dem dvs. af de har få indbyrdes bindinger.

Model holder de data, der skal arbejdes med. View er til brugerinteraktion, og Control er forretningslogikken. MVC er ikke udelukkende anvendeligt i forbindelse med personer, men kan også fint bruges ved kommunikation mellem maskiner. I stedet for at hænge en webbrowser på view-delen, kunne man udstille dataudvekslingen som webservice (stikord: SOAP/WSDL, REST).

Herunder er det skitseret, hvordan sammenhængen mellem de tre dele af MVC kan opfattes.



[18]

"A controller is the means by which the user interacts with the application." (ootips.org [34])

Når brugeren f.eks. taster noget ind i en textbox og trykker på submit, aktiveres den tilhørende metode i Controlleren. De indtastede data ligger i Model og kan herefter arbejdes videre med. Når Model er opdateret, sendes den tilbage til Viewet, så brugeren kan se resultatet af sine handlinger.

I ASP.NET MVC opnås sammenhængen ved at navnet på view'et og navnet

på metoden knyttes sammen, således, at hvis der f.eks. er en Controller med en metode, der er navngivet Edit, så er der et tilhørende View med navnet Edit. Kildekoden til view'et lægges så i en mappe, der er navngivet efter Controlleren. Der er dog mulighed for at anvende andre navne til de returnerede Views.

Det vil, for Microsoft, være en oplagt vej at vælge reflection til at håndtere den konkrete sammenknytning, når der i koden står "return View()". Hvilket View? Det finder man ud af, ud fra klassens og metodens navn.

I appendix er Reflection nævnt, og Dependency Injection berøres også, da det er et andet særdeles udbred design pattern, der anvendes til afkobling af klassers indbyrdes afhængigheder.

Betrægninger omkring sikkerhed

Internet er vokset op i et beskyttet værksted, i universitetsverdenen, hvor der var en naiv indstilling om, at der kun var whitehats. Det viser sig jo at ved kommercialisering af anvendelsen af nettet, er der kommet stadigt flere blackhats, og stadigt større dele af vores handel og pengestrømme kommer via nettet.

Det betyder, at forhold omkring sikkerhed er blevet stadigt mere relevante. Det er et stort område at begive sig ud i. Formålet med sikkerhed er sørge for, at data, der lagres og sendes til/fra en bruger, er korrekte, at der hverken er tilføjet, fjernet eller ændret noget. Ligeledes skal data ikke havne hos andre, end de tiltænkte. Endvidere skal der tænkes på, at den ydelse, der ønske, bliver leveret, altså at oppetid er 100% eller tæt herpå, samt at f.eks. DDoS-angreb ikke får nogen betydning. Disse kan hjælpes med dublerede systemer med automatisk failover, hhv. brug af ydelser fra f.eks. Cloudflare [36], der gemmer et internetbaseret system bag en indgang med meget stor båndbredde. På Cloudflares website kan ses, at firmaet har afværget 400Gbps angreb.

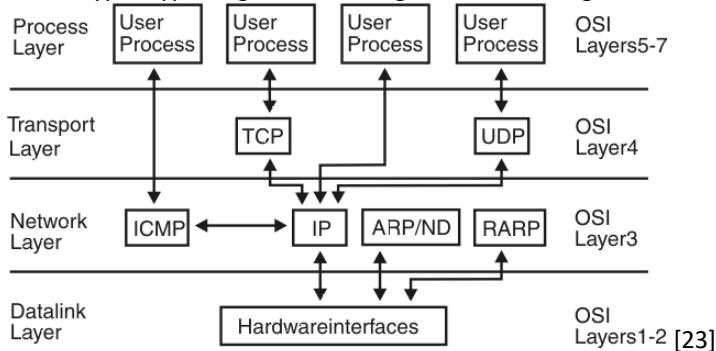
Blandt tiltag, der kan hjælpe på sikkerheden, er:

- Firewall, der kan filtrere på IP-adresser og porte, og monitorere indholdet i trafikken.
- Kryptering af payload, så trafik, der opsnappes, kun vanskeligt vil kunne afkodes.
- Kryptering af header og payload som det anvendes i forbindelse med VPN og TOR.
- Penetreringstest, for at finde hullerne før en blackhat gør det.
- Authentication, kun registrerede brugere må få adgang.
- Hashing af passwords, så de kun har ringe værdi for en tylv.

Kryptering vil typisk gøre brug af en offentlig og en privat nøgle, der skal anvendes i par. Afsenderen krypterer med modtagerens offentlige nøgle. Meddelelsen kan kun dekrypteres med den tilhørende private nøgle, og da den er privat, og dermed kun kendt af modtageren, er det kun modtageren, der kan læse indholdet. (Wie geht das? Se f.eks. appendix under "Offentlig og privat nøgle") [37]

Hvis afsender krypterer med sin private nøgle, kan alle læse indholdet ved at dekryptere med afsenders offentlige nøgle. Hvis det kan lade sig gøre, vides det, at det kun kan være den forventede afsender, der har krypteret indholdet.

Denne type kryptering vil f.eks. foregå med TLS, som gennemføres i transportlaget



VPN krypterer hele pakken, dvs. header og payload. Den arbejder på linklaget og gør, at der skal være en opkobling til en bestemt maskine.

Det er en almindelig løsning til at koble hjemmearbejdspladser til en virksomhed. Datatransporten kan kun være til en bestemt maskine, der så står i firmaet. Indholdet at kommunikationen kan hverken inficeres eller afkodes.

Man kan stedfæste en maskine, ret geografisk, ud fra IP-adressen. Ved at køre gennem en VPN-maskine ud på nettet, vil man pludseligt være et andet sted verden.

<http://whatismyipaddress.com/> fortæller at maskinen har adressen 87.59.103.16 og hører til i Hillerød.

En forbindelse til <http://freevpnaccess.com/> giver en længere liste, hvorfra UK – London er valgt

Location	VPN Server	Protocol	Username	Password
US - Chicago	67.212.175.123	PPTP	freevpnaccess.com	4850
UK - London	83.170.84.216	PPTP	freevpnaccess.com	5550
Canada - Montreal	184.107.164.107	PPTP	freevpnaccess.com	3080

Som forventet ud fra tabellen, er adressen nu ændret til 83.170.84.216, og den bliver vist som værende i London. Hvis man vil have google til at vise andre reklamer, end dem man er vant til, kan det anbefales at flytte sin virtuelle lokation.

Et opslag på dr.dk uden VPN viste Wireshark anvendelse af forskellige protokoller (bl.a. DSN og TCP) og IP-adresser. I næste session blev VPN anvendt. Der var et par ARP requests, men ellers var al trafikken mellem pc'en (192.168.0.11) og VPN-severen (83.170.84.216), og protokollerne afslørede ikke meget.

VPN gør at man er lidt sværere at spore og finde fysisk placering på. Hvis man går helt i paranoia-modus, så er det nok TOR [24] man skal ty til.

The Onion Router er bygget for at sløre forbindelsen mellem to maskiner, der kommunikere; f.eks. en webbrowser og en webserver. TOR er opbygget med et antal entry-nodes og et antal exit-nodes. Mellem dem er der et antal nodes til at sende trafikken i forskellige retninger.

Mellem klient og entry-node henholdsvis mellem exit-node og næste server, anvendes standard TCP-kommunikation. Inden i netværket anvendes en "TOR-protokol", der er lavet til netop det formål. Mellem TOR-browseren og entry-node1, og i resten af TOR-nettet, anvendes TLS, men hvis ikke destinations-web-serveren bruger TLS/SSL, så ligger store dele af kommunikationen alligevel åben; dog kan browserens adresse ikke umiddelbart findes, da exit-noden sender request, og dermed står som modtager.

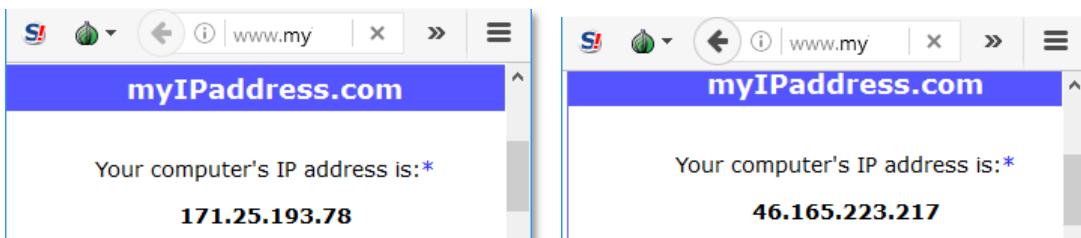
Trafikken er krypteret ved anvendelse af symetrisk nøglepar. Kommunikationen indledes ved, at klienten vælger en vej rundt i TOR-netværket. Herefter krypteres data med hver enkelte hub's offentlige nøgle. I krypteringen indgår også adressen på hub'ens efterfølger. [25] (afs. 5.1.3)

Der er nogle faste holdepunkter, hvorfra der kan hentes lister over de realy-punkter, der indgår i TOR-netværket. Den initiale liste findes på TOR-projektets web-side. For at kommunikere med en maskine på den anden side af TOR-nettet, bygger TOR-browseren først et circuit, bestående af en entry node, et antal mellem-nodes og en exit-node. Under afprøvningerne af TOR-browseren, var der konsekvent tre nodes mellem browser og destinationen.

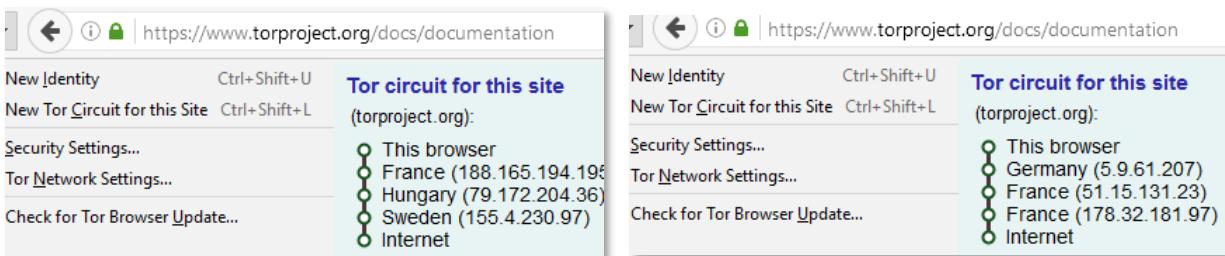
En given node kender således tre maskine i nettet. For at kunne sløre kommunikationsvejen, skal der indgå en maskine mere. Det er så TOR-browseren i dette tilfælde. I afsnittet ovenfor, blev fundet netop den kombination.

Et circuit opbygges ved at sende en Create-kommando til første node. Herefter bygges én nøgle, der skal anvendes til kryptering på første delstrækning. Efterfølgende sendes en Extend-kommando til sidste node i det partielt byggede circuit. Denne node vil så sende en Create til næste node.

For hvert HTTP-request bygges et nyt circuit, hvilket hæmmer mulighederne for at følge trafikken. Det eneste der er sket mellem disse billeder, er en refresh (Ctrl-R)



På samme måde er her to circuits for at ramme Torprojects dokumentation to gange med refresh.



Data sendes til første node, hvor data bliver dekrypteret. Herefter kan næste node's adresse findes og data sendes dertil. En periodisk oprydning på de enkelte hubs fjerner alle data omkring transmissionerne.

Alt i alt bliver det svært at se forbindelsen mellem browser og server.

I dokumentationen for TOR kan læses at TOR kan bruges til andet en browsing. Der kan bl.a. udveksles information v.hj.a. Rendevouz-points [26], hvor to personer anvender et forudaftalt ressourcenummer, der anvendes til at finde en fil eller andet. Ellers er skemaet som ved browsing.

Behov for hashing af passwords

Mange gang skal der logges på et web-site for at kunne gøre brug af ydelserne. Det sker for at sikre, at der ikke er en person eller maskine, der agerer på en anden persons vegne, om det så er debatindlæg eller pengeoverførslер.

I password input text feltet i på websiden vises det indtastede password som stjerner eller cirkler, men de opträder eller i klartekst. I nedenstående billede har Wireshark opsnappet en logon-dialog over en ikke-krypteret forbindelse.

Kunne man så ikke blot forvanske passwordet på browsersiden? Jo, det kan man godt, men det vil kun gøre livet marginalt vanskeligere for en blackhat. Det forvanskede password kan jo opsnappes, og v.hj.a. en slags proxy kan det så indsættes i stedet for det, der bliver tastet ind. Yderligere diskussion i [27].

✓ Form item: "UserID" = "JegErBruger"						
Key: UserID						
Value: JegErBruger						
✓ Form item: "Password" = "OgHarEtPassword"						
Key: Password						
Value: OgHarEtPassword						
> Form item: "Løsen" = "Løsen"						
0320 77 59 25 32 42 79 77 70 5a 76 63 35 30 34 64 6e	wY%2Bywp Zvc504dn					
0330 4c 77 76 77 4b 78 45 48 6e 77 46 33 50 56 66 65	LwvwKxEH nwF3PVfe					
0340 50 59 25 32 42 6f 69 38 77 30 44 66 55 6a 79 6f	PY%2Boi8 w0DfUjyo					
0350 38 71 30 56 43 76 50 52 62 26 55 73 65 72 49 44	8q0VCvPR b&UserID					
0360 3d 4a 65 67 45 72 42 72 75 67 65 72 26 50 61 73	=JegErBr uger&Pas					
0370 73 77 6f 72 64 3d 4f 67 48 61 72 45 74 50 61 73	sword=Og HarEtPas					
0380 73 77 6f 72 64 26 4c 6f 67 6f 6e 3d 4c 6f 67 6f	sword&Lo gon=Logo					
0390 6e	n					

Der kun én ting, der gøre kommunikationen mere sikker, og det er kryptering af forbindelsen. I.flg. INFORSECURITY.DK [28] er halvdelen af alle webadresser forsynet med kryptering. De øvrige steder må man altså være mere forsiktig.

På webserveren skal passwordet sammenlignes med en kopi gemt i f.eks. en database. Hvis de er ens, man gå ud fra, at det er den

rigtige person, der logger ind. Hvis passwordet lagres i klartekst, har man gjort livet for sine brugere mere usikkert.

Ud over at en blackhat kan misbruge sitet, hvorfra data er hentet, kan samme password forventes anvendt flere andre steder: "55 procent af befolkningen bruger den samme adgangskode flere forskellige steder på nettet.", står der at læse i computerworld [29]. Selvom der er kommet mere fokus på det problem og artiklen er ca. 1½ år gammel, så eksisterer det sandsynligvis stadigt.

For at sikre brugerne bedre, kan man forvanske passwordet før det bliver persistenter, og foretage samme forvanskning før sammenligning. Flere steder (bl.a. [30]) argumenteres for, at der skal anvendes hashing, der er irreversibel funktion, i modsætning kryptering, der er reversibel. Det vil kunne åbne nogle døre for en blackhat, og det er ikke ønskeligt for brugerne.

Når nu passwords bliver hashed er sikkerheden for brugeren lidt større, men da mange stadigt anvender passwords, der er lette at huske, optræder de hyppigt, og dermed er der ikke rigtigt nogen sikkerhed alligevel. Flere steder på nettet (se f.eks. [31]) er der lækkede passwords i klartekst og deres tilhørende hashes. I referencen til det Urainske site, anvendes MD5. Den bør man nok afholde sig fra at anvende. Dels er der fejl i den og dels tager det ikke lang tid at hashe med algoritmen.

På youtube [32] vises, hvordan HashCat på kort tid kan finde ud af, hvilken algoritme, der anvendes til hashing og i en anden youtube video [33] fortæller Dr Mike Pound, hvordan en maskine, der er "tre-fire gange så stor som en almindelig desktop", kan generere 40×10^9 MD5-hashes pr. sekund. Ud fra ordbøger og crackede passwords i lister, viser han, hvor let det er at komme i besiddelse af password i klartekst. Han anbefaler SHA512 frem for MD5

Owasp.org [35] har mange gode beskrivelser af, hvad der er af sikkerhedsproblemer og, hvordan man omgår dem. I afsnittet "Use a cryptographically strong credential-specific salt" forklares, at man genererer en stor tilfældig værdi (kaldet salt) som password forlænges med. Herefter hashes den sammensatte størrelse og gemmes i databasen sammen med salt. Derved bliver ens passwords lagret med forskellige værdier, hvorved forudberegnede lister ikke længere har nogen værdi.

Afsnippetet "Leverage an adaptive one-way function" lister forskellige hash-funktioner, hvor Argon2 anbefales, og SHA512 ikke er med. Argumentationen for valgene er bl.a., at det skal tage lang tid at generere en hash, for bedst mulig beskyttelse.

Det stadigt muligt at afprøve kendte passwords, og brute force, hvor alle tegnsekvenser løbes igennem, men det tager længere tid, da der skal beregnes en hash for hvert forsøg. For yderligere at besværliggøre angrebene, kan der tilføres et konstant salt, der gemmes i maskinens filesystem. Den værdi vil typisk være sværere at få fat i. Microsoft [38] anbefaler adgangskontrol til filesystemet på platformsniveau.

Praktisk opgave

Projektet er offentligt tilgængeligt på github [7]

Et dynamisk website

Det påtænkte website vil have en fastlagt struktur, men indholdet vil variere ud fra da lagrede data; herunder vil visse data og funktioner kun være tilgængelige for personer, der er registreret som administrator.

Databasemodel

Attributter

	Anvendelse
brugerId	brugerens eget valg af identifikation. Skal være unik.
createTs	tidspunkt for, hvornår rækken er oprettet eller ændret
password	brugerens password i hashed udgave
salt	det salt, der skal gøre hash'en forskellig fra andre hashes af samme password
email	email-adresse
prim	markering af den foretrukne email-adresser
erAdmin	om brugeren har administrative rettigheder
lastLogon	hvornår har brugeren sidste logget sig på systemet
logonCnt	hvor mange gange har brugeren været på
arrangementBeskrivelse	en kort forklaring på, hvad arrangement går ud på
arrangementWeb	link til web-siden, hvis en sådan findes
arrangementDt	hvilken dato afholdes arrangementet
arrangementTid	hvor tid på dagen indledes arrangementet
erTilmeldt	om bruger er tilmeldt arrangement
spiseSted	hvis der er spisning inkluderet, så navn og adresse på spisested
spiseTid	tidspunkt for spisning under antagelse af, at det er samme dato som arrangementet
spiser	om bruger deltager i spisning
billedAdresse	path/filename.extension for uploadedede billeder fra arrangementet
kommentar	en optionel tekst om billedet

Password og salt gemmes i et antal versioner for at brugeren ikke skal kunne genanvende et password før der er gået et antal generationer.

Da email-adresser er universelt unikke, vil det være en oplagt mulighed at lade adressen indgå som en del af nøglen, men det er valgt, at email-adressen skal være optionel, og det skal være muligt at ændre den.

I nogle RDBMS'er eksisterer en auto-increment værdi, så hver gang der indsættes en række tælles op. Microsoft SQLserver [8] har en værdi, der tager udgangspunkt i timestamp, men er garanteret unik. Disse er naturlige valg til identifikation af en række i en tabel.

Hvis ikke der er en datatype med automatisk optælling, kan man konstruere sig frem til en sådan, under forudsætning af, at operationen kan køre atomisk. Altså at der ikke er flere, der kan indsætte eller opdatere samtidigt på samme tabel. Konstruktionen vil se ud som følgende:

Der er en kolonne med navn serialNumber, der indeholder et stigende rækkenummer for tabellen.

```
Insert into tabelNavn (serialNumber, ....) values((select max(serialNumber)+1, ...);
```

I en miniature-applikation som nærværende vil det ikke være noget problem at have en lås på hele tabellen, så kun en af gangen kan opdatere i den. I lidt mere anvendte applikationer, vil det normalt ikke være acceptabelt, at låse hele tabellen. Her vil man skulle nøjes med en rækkelås, eller lås den page, der er underopdatering, og så vil ovenstående SQL ikke kunnet garantere en unik værdi.

MySQL har timestampværdier med et-sekunds opløsning som standard. Der er så alligevel mulighed for at oprette op til seks cifre efter som en brøkdel af sekunder; altså en opløsning i mikrosekunder [9].

Andre RDBMS'er anvender sekunder, millisekunder eller mikrosekunder. Nogle gange er også 100 nanosekunder set [10]. I denne webapplikation, er alle indsætninger og opdateringer på tabellerne initieret manuelt. Det vil derfor ikke være forventeligt, at en bruger foretager to ændringer inden for et sekund. BrugerId er unikt og oprettelsestidspunktet (createTs) vil være unikt pr. bruger, når det nu antages, at brugeren ikke kan foretage sig flere

handleringer inden timestamp er ændret. Det vil derfor være en mulighed at anvende brugerId og createTs som sammensat nøgle for alle data.

Hvis man forsøger at indsætte to rækker i en tabel, med samme nøgle, opstår der en constraint violation; man kan ikke anvende en given nøgle mere end en gang. Det skal så i givet fald en fejtekst i webbrowseren, så brugeren kan se, at der skal forsøges en gang mere. En mere brugervenlig udgave vil selvfølgeligt være at applikationen i første omgang selv håndterer situationen og f.eks. læser current timestamp nogle gange, og, når timestamp har ændret værdi, indsætter/opdaterer en gang mere.

Attributterne har hver netop en type. Der er ikke nogle af felterne, der skal opdeles, så de er atomiske. Det samme som, at der ikke er nogle repeterende felter i ovenstående, og ved anvendelse af brugerId og createTs som sammensat nøgle, vil alle kolonnerne i en tabel være på 1NF. Det ses dog hurtigt, at der ikke er fuld funktionel afhængighed for alle øvrige kolonner. F.eks. er arrangementTid en attribut, der fortæller om selve arrangementet, men ikke fortæller noget om personen, der har oprettet arrangementet eller har tilmeldt sig. Dermed er der ikke en funktionelt afhængighed til nøglen, hvilket er et krav på 2NF.

Generelt kan det siges, at 1NF kræver atomiske værdier og ikke-repeterende grupper. Desuden skal der være en eller flere attributter, der tilsammen entydigt kan identificere en tuple.

2NF kræver, at alle attributter, der ikke indgår i nøglen, skal være fuld funktionelt afhængige af nøglen.

3NF siger, at der ikke må være nogen transitive afhængigheder.

4NF kræver, at der er max en flerværdiet afhængighed i hver relation.

Hvis der er en relation med bilmodeller, motorstørrelser, farver og priser, og vi antager, at der udelukkende er metalfarver. R(model, motor, farve, pris). Det viser sig, at pris ikke varierer med farven, men kun model og motor. For at komme på 2NF skal R splittes i R1(model, motor, farve) og R2(model, motor, pris).

3NF kan illustreres med R(navn, adresse, postnummer, by), hvor by er transitivt afhængig af postnummer. Derfor splittes R til R1(navn, adresse, postnummer) og R2(postnummer, by)

Til 4NF kan vi se på bil-relationerne igen. Der er flere modeller, flere motorer og flere farver. Motor og farve har ingen indbyrdes afhængighed, så derfor bliver R1(model, motor, farve) opdelt i R1a(model, motor) og R1b(model, farve).

Til praktisk anvendelse, er 3NF normalt helt fin; dog finder 4NF også nogle gange god anvendelse, specielt hvis der er tale om større datamængder. Yderligere normalisering er vist mest en teoretisk øvelse.

Ovenstående diskussion udmønter sig i at der skal laves fem tabeller, og at der skal tilføjes et entydigt arrangementId, som vil blive automatisk optalt nummer eller, hvis det ikke findes i pågældende RDBMS's, current timestamp.

Logon		User							
brugerId	lastLogon	logonCnt	brugerId	createTs	password	salt	email	prim	erAdmin
Arrangement									
arrangementId	arrangementBeskrivelse		createTs	arrangementWeb	arrangementDt	arrangementTid			
spiseSted	spiseTid								
Tilmeldinger									
arrangementId	brugerId	createTs	erTilmeldt	spiser					
Billeder									
billedAdresse	arrangementId	brugerId	createTs	kommentar					

Ovenstående konstruktion kan også give nogle ganske "krøllede" SQL'er. Hvis man f.eks. skal finde ud af, hvem der deltager i spisning til et givet arrangement, skal man finde den nyeste række for et givet arrangement pr. bruger. Der er mange dialekter af SQL. En af dem er DB2 for zOS og her ville forespørgslen kunnet formuleres som følgende:

```
select brugerId
from tilmeldinger
where (arrangementId, brugerId, createTs) in (
    select arrangementId, brugerId, max(createTs)
    from tilmeldinger
    where arrangementId = :hvadErDetForEtArrangementDerSpørgesPå
    group by brugerId
)
and spiser = 'True'
```

Det er dog ikke alle SQL dialekter, der tillader sådan en konstruktion med flere attributter i where-in-clause. Som en omgåelse kunne man konatenere arrangementId, brugerId og createTs til en tekststreng både i where-clausen og i subselecten. Hvis det viser sig, at der ikke er en enkelt konstruktion, der kan løse opgaven, kan man oprette en temporær tabel og anvende den i et select-statement.

```
create table sel (arrangementId char(...), brugerId char(...), createTs Timestamp);

insert into sel
select arrangementId, brugerId, max(createTs)
from tilmeldinger
where arrangementId = :hvadErDetForEtArrangementDerSpørgesPå
group by brugerId
;

select brugerId
from tilmelding t, sel s
where t.arrangementId = s.arrangementId
and t.brugerId = s.brugerId
and t.createTs = s.createTs
and t.spiser = 'True'
;

drop table sel;
```

Det bliver til en længere historie; dog har flere SQL varianter andre muligheder, der gør forespørgslen mindre klodset. Den dyreste mulighed er at tage resultatet af subselecten ind i et array i programmet, og så derefter slå op i tabellen tilmelding med hver enkelt række i arrayet. Der bliver meget transport og initiering af forespørgsel i databaseserveren.

Man kan argumentere for en yderligere normalisering, da man kan risikere redundans. F.eks. er User-tabellen ikke i 4NF. Det medfører, at en ændring i f.eks. email-adressen giver en ekstra række i tabellen, men med de samme data som i den nyeste række, bortset altså fra email-adressen.

Der eksisterer utallige beskrivelser af normalisering, men nogle af dem er [11], [12] pp387 og [14].

Opsplitningen vil give tre tabeller, hvor password og salt hører sammen i den ene tabel, email-adressen hhv. erAdmin kommer i to andre tabeller. Hvis ikke der var interesse for historikken på Admin-tabellen, så ville createTs være overflødig. Dermed ville der være brugerId som nøgle. Da erAdmin er en boolsk værdi, ville det ikke være nødvendigt at have den som selvstændig attribut, og derfor ville Admin-tabellen kun bestå af brugerId.

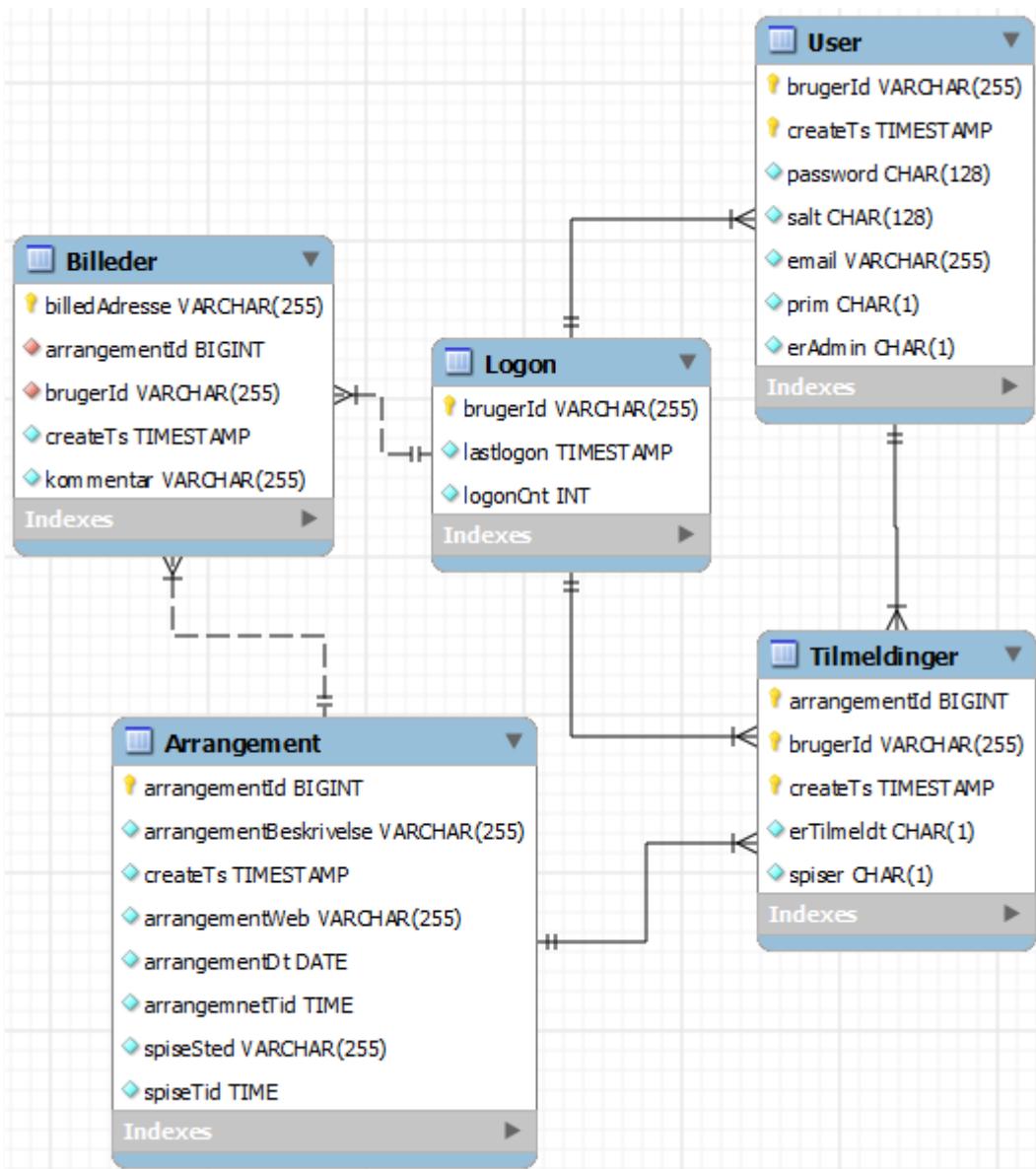
UserPassword				Email			Admin		
brugerId	createTs	password	salt	brugerId	createTs	email	brugerId	createTs	erAdmin

Af praktiske grunde accepteres redundans, og normalisering ud over 3NF (og evt. Boyce Codd) vil ikke blive gennemført.

Auto-increment konstruktionen blev fundet på mysql.com [13]. Den varierer meget mellem de forskellige database systemer. Samme sted blev også fundet en boolean datatype, men den gav syntaxfejl ved import i MySQL-Workbench; derfor blev datatypen ændret til char(1), der så f.eks. kan antage værdierne 'T' og 'F'. Det vil blive styret i programmet, der skal læse og skrive på tabellerne, ved hjælp af to konstanter. I C# bliver det f.eks. til:

```
public const char booleanTrue='T';
public const char booleanFalse='F';
```

Herunder er så en MySQL flavoured DDL for oprettelse af en database med de fem tabeller, og et EER-diagram importeret i MySQL-Workbench.



```
create database VoresJazzklub;
use VoresJazzklub;

create table Logon(brugerId varchar(255) not null, lastlogon timestamp not null, logonCnt integer not null, primary key (brugerId));

create table User(brugerId varchar(255) not null, createTs timestamp(6) not null, password char(128) not null, salt char(128) not null, email varchar(255) not null, prim char(1) not null, erAdmin char(1) not null, constraint pk_user primary key (brugerId, createTs));

create table Billeder(billedAdresse VARCHAR(255), arrangementId BIGINT, brugerId VARCHAR(255), createTs TIMESTAMP, kommentar VARCHAR(255));

create table Arrangement(arrangementId BIGINT, arrangementBeskrivelse VARCHAR(255), createTs TIMESTAMP, arrangementWeb VARCHAR(255), arrangementDt DATE, arrangementNetId TIME, spiseSted VARCHAR(255), spiseTid TIME);

create table Tilmeldinger(arrangementId BIGINT, brugerId VARCHAR(255), createTs TIMESTAMP, erTilmeldt CHAR(1), spiser CHAR(1));
```

```

create table Arrangement(arrangementId BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,
arrangementBeskrivelse varchar(255) not null, createTs timestamp(6) not null, arrangementWeb
varchar(255) not null, arrangementDt date not null, arrangementNetTid time not null, spiseSted
varchar(255) not null, spiseTid time not null, primary key(arrangementId));

create table Tilmeldinger(arrangementId BIGINT not null, brugerId varchar(255) not null, createTs
timestamp(6) not null, erTilmeldt char(1) not null, spiser char(1) not null, constraint
pk_tilmeldinger primary key(arrangementId, brugerId, createTs));

create table Billeder(billedAdresse varchar(255) not null, arrangementId BIGINT not null,
brugerId varchar(255) not null, createTs timestamp(6) not null, kommentar varchar(255) not null,
primary key(billedAdresse));

```

I gang med dette konkrete setup

En del af opgaven er at undersøge, hvordan ASP.NET MVC fungerer, og holde det op mod andre måder at løse samme slags web-kommunikation på. Derfor bliver der oprettet nogle små projekter, der på ingen måde vil kunne anvendes til noget praktisk, men kun er til for at afsløre, hvordan centrale problemstillinger bliver adresseret.

Når man opretter et projekt i Visual Studio og vælger ASP.NET MVC med tilhørende testprojekt, får man en kørende applikation og tre fungerende test i en struktur som vist her; dog er mappen Models i testprojektet tilføjet efterfølgende.

Der er små 600 filer og over 800 mapper. Om det er en fordel eller ulempe, at få så meget, komme nok an på, hvor godt man kender IDE'et og den generelle tankegang i MVC-modellen.

I første omgang kan man nøjes med at koncentrere sig om mapperne Controllers, Models og Views i projektet, og de tilsvarende mapper for controllers og views i testprojektet. Der er ikke umiddelbart nogen test af webgrænsefladen, men i betalingsudgaven af Visual Studio er der mulighed for at få startet IE og der igennem optage nogle handlinger mod applikationen. Handlingerne kan så afspilles igen, uden anvendelse af en browser. Det er så kommunikationen mellem browseren og webserveren, der undersøges. Det er også muligt at tilføje web-tests, ved at skrive dem selv efter en optagelse [20]. Alternativt kan Selenium anvendes, som er et produkt til test af web-sider [21]. Det kræver så mere installation og applikationen, der skal testes, skal køre på en server uden for IDE'et

Det skulle selvfølgeligt prøves, men der kom flere udfordringer ud af det. Google kunne ikke umiddelbart komme med nogen brugbar forklaring, så tilbage er kun at grave sig ned i problemet, hvilket formentlig er ganske tidskrævende.

Fejteksten er ikke særligt sigende, og det er en fejl, der opstår i en komponent, der ikke er beskrevet detaljeret nok, til at fejlen umiddelbart kan afsløres.

For komme i gang, oprettes først en controller-klasse med en public metode. Som udgangspunkt rammer man metoden ved at anvende en URL på følgende form: <domænenavn>/<controller>/<metode>. Det kan man konfigurere sig ud af, hvis der ønskes en anden logisk struktur set fra brugerens side.

Controlleren kan for så vidt godt anvendes alene, hvis metoden f.eks. returnerer en tekststreng eventuelt med et stykke html, men ellers kan browseren godt vise ren tekst uden markups. Det bliver dog nogle noget fattige websider, som kan laves på den måde, med mindre man koder sig frem til en website med alt indhold; men det er absolut ikke at anbefale, da det er svært at overskue og hurtigt kan blive et meget omfattende stykke arbejde at vedligeholde koden.

Dernæst laves et view. Da controlleren blev oprettet, blev der dannet to mapper. En i controller-mappen og en i view-mappen. På den måde bliver controller og view knyttet sammen. De enkelte metoder, der skal returnere et view, skal have et view med samme navn som metoden. Én metode returnerer ét view.

Der er et master-view, som man kan lade de øvrige views arve fra. På den måde kan man opnå ensartethed på tværs af siderne i applikationen.

Den tredje komponent er modellen. Her holdes de data, der skal vises på glaspladen subsidiært persisteres i databasen. Ved oprettelse af en controller blev der samtidigt oprettet en mappe til de tilhørende views. Det sker så ikke i relation til modellerne. Der er ikke den samme en-til-en relation, da man sagtens kan forestille sig, at en given model anvendes i flere sammenhænge. F.eks. kunne der være en side, som kun viser data, og en anden side, hvor data kan oprettes, vises, ændres og slettes. Den første side kan alle tilgå, og den anden side kan kun privilegerede få fat på. Det er de samme data, der skal anvendes på de to sider, og dermed samme model.

Modellen vil have adgang til forskellige metoder til læsning og skrivning på en tabel og i filsystemet. Her skal man så være opmærksom på konfigurationsfilerne. Til projektet er det Web.config og til test-projektet er det App.config. Her lægges connection strings til databaserne. Alt andet, der skiller test fra produktion, bør på lignende måde placeres i konfigurationsfiler.

En sidste ting, der kan være relevant på den funktionelle side af web-serveren, er RouteConfig.cs. Her kan man bestemme, hvor i applikationen man kommer hen ved at angive en sti efter domænenavnet i browserens URL input box.

I web.config filen er tilføjet referencer til tre af de mulige http-fejl. 400 optræder f.eks., hvis der forespørges på et bestemt element, men der ikke er medsendt identifikation af elementet. 403 forekommer, når der forsøges tilgået en ressource, der ikke er autorisation til. Endeligt er der 404, som betyder at ressourcen ikke eksisterer.

```
<httpErrors errorMode="Custom" existingResponse="Replace">
  <remove statusCode="400"/>
  <error statusCode="400" responseMode="ExecuteURL" path="/Error/BadRequest"/>
  <remove statusCode="403"/>
  <error statusCode="403" responseMode="ExecuteURL" path="/Error/Forbidden"/>
  <remove statusCode="404"/>
  <error statusCode="404" responseMode="ExecuteURL" path="/Error/NotFound"/>
  <!-- viser kun den genererede side - overrides customErrors i system.web -->
</httpErrors>
```

Det er også i web.config, at der skal forklares, hvordan MySQL skal tilgås. Navnet anvendes så i koden.

```
<connectionStrings>
  <add name="MySQLConnection" connectionString="Server=localhost; user id=root; password=;
persist security info=True; database=voresjazzklub" providerName="System.Data.SqlClient"/>
</connectionStrings>
```

Visual Studio kan afvikle programkoden i debug-modus med eller uden de indsatte breakpoints, og i release-modus. Det var et ønske at kunne se, om app'en køres i debug-modus, da der er indbygget direkte adgang til databasen.

Den første test gik helt galt da testprojektet ikke kunne finde MySQLConnection. I Testprojektets App.config indsættes connection strings fra hovedprojektets Web.config, hvorefter testen bliver grøn.

Det er afprøvet i både Release og Debug modus. Fra tid til anden skal projektet Cleanes og Rebuildes, når der bliver skiftet mellem Release og Debug, da der er cachet information, der får testprojektet til at køre mod en forkert version.

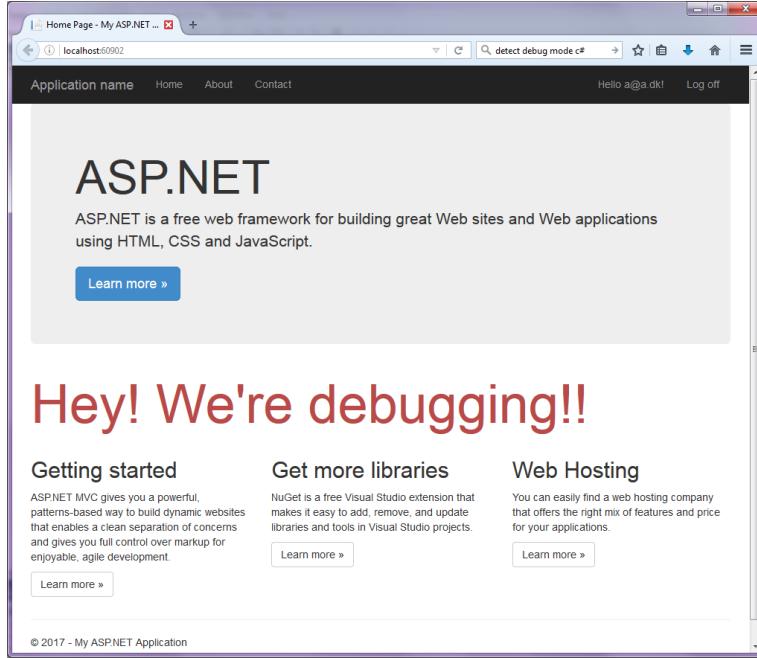
DEBUG er en variable, som VS sætter ud fra om den arbejder i Debug eller Release modus. Den kan så anvendes af C#'s precompiler til at inkludere eller ekskludere kode.

Her sættes en tekststreg i ViewBag, som er en strukturert samling objekter, der kan sendes til View'et

```
public class HomeController : Controller {
  public ActionResult Index() {
    #if DEBUG
      ViewBag.isDebugging = "Hey! We're debugging!!";
    #endif
    string title = ViewBag.Title;
    return View();
  }
}
```

I view koden undersøges, om der er en værdi med navn isDebugging og hvis det er tilfældet, bliver den vist på web-siden. Teksten bliver styret ud fra

```
@{ if(ViewBag.isDebugging != null) {
    <a href = "~/Views/Home/Index.cshtml" > ~/ Views / Home / Index.cshtml </ a >
    <w class="text-danger large-text">@ViewBag.isDebugging</w>
}
```



to classes. Der anvendes to styles sheets, det ene indeholder styling af alle almindelige elementer og det andet bruges til ekstra styling og/eller til at override eksisterende styles. Her anvendes text-danger, som er med i projektet fra starten. Den giver den fode farve, og large-text er ny, da der ikke var andre styles, der virkede naturlige at anvende. Det er nemt at lave sådan en klasse, men man skal jo også have øje for, at der ikke er flere klasser, end at det er til at vedligeholde.

Som nævnt er der direkte adgang til databasen. Her er kørsel af forespørgsel i Debug Configuration hhv. en kørsel af forespørgsel i Release Configuration. I sidstnævnte tilfælde sker der intet.

Øverst i IDE'et vindue står der enten Debug eller Release. Ved Release anvendes to tomme metoder. Den øvrige kode er grået ud. Ved debug er det lige omvendt.

I øverste venstre hjørne står "Debug" og i kode testes på om DEBUG er sat. Derfor kan SQL-scriptet køres

The screenshot shows two windows side-by-side:

- Browser Window:** Shows a page titled "SQLView". It contains a text area with the result of a SQL query: "0" and "select * from users". Below it, a message says "Ingen rækker fundet i selectScript()". At the bottom is a button labeled "Send forespørgsel".
- IDE Window:** Shows the Visual Studio IDE with the "SQLmodel.cs" file open. The code contains a try-catch block for handling MySQL exceptions. Lines 41 through 59 are visible, showing the connection setup, command execution, and exception handling logic.

Migrering af adgangskontrol fra SQLserver til MySQL

Med det projekt, der bliver genereret af Visual Studio, kommer også et sikkerhedsframework. Da det foregår i en Microsoftverden, er det naturligt, at der anvendes Microsoftprodukter. Således bruges SQLserver, men da det er MySQL, som skal holde alle data i Voresjazzklub, skal der foretages nogle ændringer i koden.

Der er nogle klasser i Microsoft.AspNet.Identity.EntityFramework, der tilsammen håndterer brugeradgangen. Forventningen var, at der skulle skiftes ud i nogle metoder og andre skulle overrides, og at connection string skulle ændres fra SQLserver til MySQL.

Efter at have boret i problematikken et par timer, måtte det konstateres, at det nok kunne lade sig gøre, men at det samtidigt ville være en større operation.

Hvis projektopgaven var fokuseret på sikkerhed, ville det ganske givet være en interessant øvelse at ændre i frameworket. Her anvendes så i stedet dele fra en opskrift fundet på nettet. Det, der ikke anvendes, er hosting i Azure. [16] Der skal hentes et projekt [17] bestående af ti klasser. Det skal bygges og i Voresjazzklub skal der så være en reference til det. Derudover et par mindre justeringer i Voresjazzklub.

Download AspNet.Identity.MySQL, og build AspNet.Identity.MySQL.sln

Højre klik Voresjazzklub | Add | Reference

Browse til AspNet.Identity.MySQL..dll

Der står ganske vist noget helt andet i opskriften, men den er datostemplet 22/5/2015 – og det er lang tid i den verden.

Der skal ændres i tre filer. Microsoft.AspNet.Identity.EntityFramework skal skiftes ud med AspNet.Identity.MySQL + et par småjusteringer i to metoder. I web.config skal DefaultConnection flyttes til at pege på MySQL

IdentityModels.cs:

```
//using Microsoft.AspNet.Identity.EntityFramework;
using AspNet.Identity.MySQL;

public class ApplicationDbContext : MySQLDatabase {
    public ApplicationDbContext(string connectionName)
        : base(connectionName) {
    }

    public static ApplicationDbContext Create() {
        return new ApplicationDbContext("DefaultConnection");
    }
}
```

IdentityConfig.cs:

```
//using Microsoft.AspNet.Identity.EntityFramework;
using AspNet.Identity.MySQL;

var manager = new ApplicationUserManager(
    new UserStore<ApplicationUser>(
        context.Get<DbContext>() as MySQLDatabase));
```

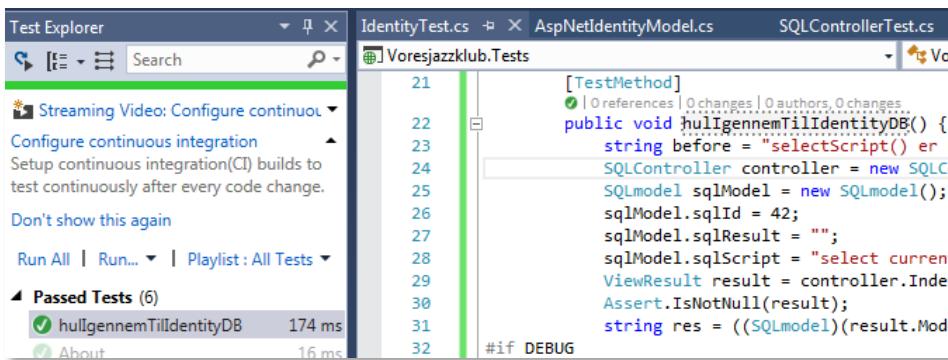
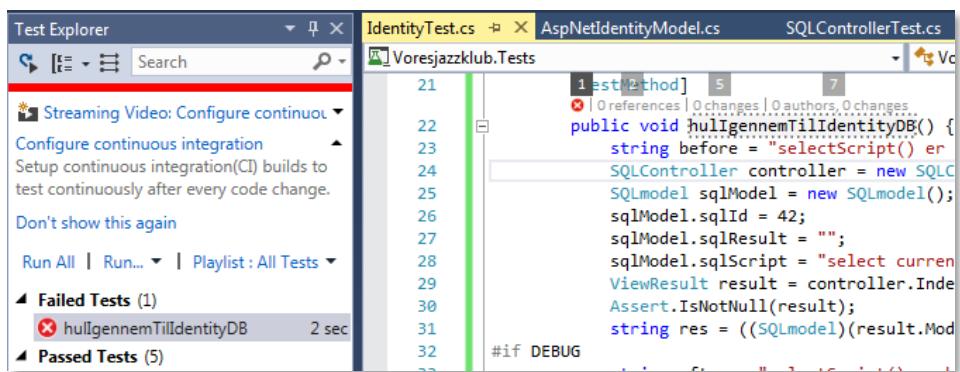
web.config:

Connection string change to MySQL

En sidste lille ting, der skal gøres før Voresjazzklub igen kan komme op at køre, er, at der skal oprettes fem tabeller i MySQL (se Appendix: "Script til migrering af adgangskontrol"). Da tabellerne allerede ligger separeret fra de øvrige tabeller, er det valgt, at fortsætte sådan.

De tabeller, der er oprettet specifikt til Voresjazzklub ligger i databasen Voresjazzklub, og tabeller til adgangskontrol ligger i databasen MySQLIdentity. Når tiden er moden til en ændring af login mv. så det ligner de oprindelige tanker, er det nemmere at skille tingene ad, fordi der skal fjerne en database og en reference til et andet projekt.

Når der bliver lavet noget nyt, skal der også være kørt en test. Første gang den køres, forventes det, at den fejler, og når der er kodet tilpas meget, og testen køres igen, skal den blive grøn. Det er naturligvis også tilfælde her, hvor det tjekkes om der er forbindelse til den nye database.



Downside ved at anvende Microsofts identitests framework er, at det kompromitterer den valgte datamodel. Det betyder så også, at applikationens funktionalitet bliver lidt anderledes en oprindeligt tænkt.

Beskyttelse af password

.Net frameworket indeholder meget funktionalitet, således også hashing og generering af tilfældige tal.

Der er fremstillet to metoder til brug, hvor brugerens password skal håndteres. Det er ved oprettelse, ændring og tjek af validitet. Den ene får et password i klartekst ind og afleverer hash og salt. Som det ses, er hashen genereret ud fra en konstant værdi, en tilfældig værdi og password. Disse tre værdier er tekststrenge, der konkateneres.

```

public void getHashedPasswordAndSalt(string password, out string salt, out string hash) {
    salt = getSalt();
    hash = getHashSha512(getConstantSalt() + salt + password);
}

```

Den anden metode tager password i klartekst og det salt, der er persisteret i databasen, og returnerer samme værdi som i metoden ovenfor.

```

public string getHashedPassword(string password, string salt) {
    return getHashSha512(getConstantSalt() + salt + password);
}

```

Selve hashingen står .Net for i klassen SHA512Managed i metoden ComputeHash. Teksten konverteres til bytes, og outputtet konverteres til en hexadecimal streng.

```

string getHashSha512(string stringToBeHashed) {
    StringBuilder hashedString = new StringBuilder();
    foreach(byte hashedByte in new SHA512Managed().ComputeHash(Encoding.Unicode.GetBytes(stringToBeHashed)))
        hashedString.Append(String.Format("{0:x2}", hashedByte));
    return hashedString.ToString();
}

```

Salt fremstilles også i .Net med klassen RNGCryptoServiceProvider, ligeledes som hexadecimal streng. De 64 bytes i mellemresultatet matcher 512 bit i hashen. 64 byte à 8 bit: $2^6 * 2^3 = 2^{(6+3)} = 2^9 = 512$.

Den sidste metode er til det konstante salt. Det bliver gemt i maskinens filesystem. Fordelen er, at salt kan beskyttes, så det kun er en administrator, der har adgang til det. Der er ingen garantier mod tyveri, men det er i hvert fald gjort besværligt.

```

string getSalt() {
    StringBuilder salt = new StringBuilder();
    byte[] bytes = new byte[64];
    new RNGCryptoServiceProvider("").GetBytes(bytes);
    foreach(byte b in bytes) {
        salt.Append(String.Format("{0:x2}", b));
    }
    return salt.ToString();
}

```

der rammes rigtigt. Det er 2^{128} , eller ca.
 $3 \cdot 10^{38}$

Hvis nu Dr Mike Pounds maskine kunne klare $40 \cdot 10^9$ SHA512 hashinger pr. sekund, ville vi skulle vente $(3 \cdot 10^{38} / 40 \cdot 10^9 \sim 10^{28}$ sekunder) i noget der ligner $3 \cdot 10^{21}$ år, dvs. det kommer aldrig til at ske.

Tilbage er kun at generere nyt salt og nulstille samtlige passwords. Meningen var at beskytte brugeren, ikke at blotlægge alle konti, så konstant salt er en vigtig ressource, der skal beskyttes.

```

public string getConstantSalt() { // anvendes i test, derfor public
    string constantSaltFilename = "...\\App_Data\\ConstantSalt";
    try {
        return System.IO.File.ReadAllText(constantSaltFilename);
    } // så må vi håbe det ikke er fordi nogen har slettet den og
    // ellers må du hellere finde din backup frem
    catch (FileNotFoundException) {
        string salt = getSalt();
        System.IO.File.WriteAllText(constantSaltFilename, salt);
        return salt;
    }
}

```

Bygning af Voresjazzklub.dk – de første skridt

Den første rudimentære udgave af modellen for tabellen for brugere: Users.

Der er to forskellige udgaver af prim og erAdmin, da datatypen i databasen er tekst, men rent logisk er det en boolsk værdi. C# kan ikke blande datatyper ved get/set, så derfor er der to properties for hver værdi. Disse properties bliver synkroniseret ved set.

Der er oprettet en metode, hentBruger, men som det fremgår, er den aldeles triviel og ganske værdiløs; bortset fra, at den er nødvendig for at kunne konstruere en test.

```

public class UsersTableModel {
    private bool _prim, _erAdmin;
    private char _primChar, _erAdminChar;

    ...
    public bool prim {
        get { return _prim; }
        set { _prim = value; _primChar=(_prim)?'T':'F'; }
    }
    public char primChar {
        get { return _primChar; }
        set { _primChar = value; _prim = _primChar == 'T'; }
    }
    ...
    public void hentBruger(string brugerId) {
    }
}

```

Tanken med klassen er, at når metoden hentBruger bliver kaldt, skal den finde brugeren på MySQL-databasen i tabellen voresjazzklub.Users og sætte klassens properties i overensstemmelse hermed.

Scriptet kunne også køres i phpmyadmin web-siden ixampp mod MySQL

Nu er tiden så inden til at fremstille den første del, der skal anvende i det færdige produkt. Det bliver model og controller til håndtering af brugere.

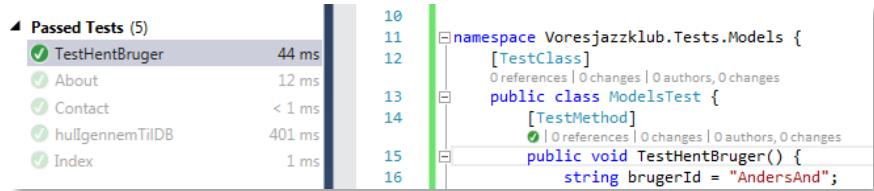
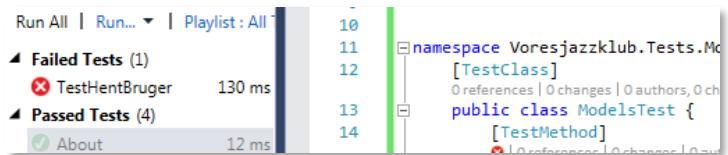
En testklasse oprettes til at undersøge, om en given person rent faktisk kan findes. I metoden TestHentBruger skrives navnet på en bruger, der skal kunne findes. Der oprettes en instans af klassen, der skal testes, og herefter kaldes den metode, der skal testes. Til sidst sammenlignes brugerId i testklassen med brugerId i den testede klasse.

I nedenstående skærmdump ses tydeligt, at testen fejlede. Ikke at det kommer som en overraskelse, da der jo netop ikke er nogen kode til at udføre det ønskede arbejde. Hvis det havde været en stor, kompleks applikation med

Sidste step er at checke sourcen incl test ind i ens versionsstyringssystem, hvortil der knyttet et Continuous Integration (CI) set-up [19], hvilket betyder, at der bliver kørt et komplet build og alle test bliver kørt. Testene skal så gerne være grønne alle sammen. Det step må gemmes til en anden gang, da det medfører tidskrævende setup, og tid er en begrænset ressource.

dependency injection i configuration filer og andre sløringer af, hvor data kommer fra, ville situationen kunne se anderledes ud. Derfor er det væsentligt, at der testes så tidligt som muligt, og at testen bliver rød.

Man starter med testmetoden. IDE'et vil ikke kunne kompilere testmetoden, da der hverken er en klasse eller metode IDE'et kan finde referencer til. Så laves det minimum, der skal til at kunne kompilere, hvorefter testen køres. De næste trin er at tilføre og ændre koden, i den testede metode, indtil testen bliver grøn. Alle test, til de berørte komponenter, skal også køres igen, for at se, om man har fået ødelagt noget.



Til et miniature skole/hobby-projekt, virker CI som overkill, men tanken om at indlede med en test, og herefter ændre koden og teste indtil tingen virker, kan man sagtens gennemføre på udviklingsmaskinen, og så en gang imellem køre samtlige test.

Efter lidt tilføjelse af kode og variable, blev testen grøn og glad. Resultatet er ganske vist ikke kønt at se på, der mangler noget kodning, og fejlhåndteringen skal ændres. Det skal også overvejes, hvor `connection.Close();` skal indsættes; men hovedformålet, at få en grøn test, blev nået.

I koden refereres klassens properties prefixed med `this.` for ikke at ramme metodens variable i stedet. Ser man godt efter, konstateres to forskellige farver på `this.` Det er fordi det kun er ved `this.brugerdId`, hvor det er nødvendigt, men hellere skrive det nogle gange fremfor at bugge tid på fejlfinding.

Det har vist sig at være en særdeles dårlig ide at anvende tekst som nøgle i database. Ikke fordi der var problemer i forhold til MySQL, men navnene på billedeerne kan indeholde både punktum, bindestreg og skråstreg. Det fungerer blot ikke med ASP.NET MVC, da link til detaljer og redigering indeholder teksten præcis som den ser ud i databasen, og det giver problemer i forhold til routing.

Nogle forsøg på at ændre RouteConfig, til at acceptere "besynderlige" tekster som nøgle, hjalp ikke i tilstrækkeligt omfang. Tilbage var så kun at ændre i tabellen og de tilhørende web-sider, Controller og Model. Helt konkret blev der oprettet en ny tabel med en automatisk nummerering, der så anvendes som nøgle:

```
create table Billeder2(billedId BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,
billedAdresse varchar(255) not null, arrangementId BIGINT not null, brugerdId varchar(255) not null, createTs timestamp(6) not null, kommentar varchar(255) not null, primary key(billedId));
```

Fri leg

Et af målene med opgaven er at lære om HTML5 og CSS3. Derfor er der flere stede indlagt netop brug af disse samt JavaScript. På siden "Baggrund" er en liste over de væsentligste web-sites, der er anvendt i opgaven. De er arrangeret

som to menuer. Den ene åbner en ny fane og den anden skifter indholdet på den eksisterende. Desuden er der hint ved mouseover, som er fremstillet med

```
<abbr title="hint"><input type="button" class="styling"...>
  onClick="window.open('http://... ')...">
</abbr>
```

Når det er samme fane, der skal vise nyt indhold anvendes
onClick="parent.location = 'https://w...'"

Stylingen har været en mange forskellige versioner. Forskellige faconer på links, der bliver vist som knapper; men og uden farveskift ved mouseover; vandrette og lodrette menuer.

Andre steder er andre nye tags som `<dl>`, `<dt>` og `<dd>` brugt til adresseliste.

Test/Validering

Et af målene med opgaven er, at gennemføre udviklingen som TDD – Test Driven Development. I bogen Clean Code [5] pp 122 beskrives tre regler for TDD:

1. Der må ikke skrives produktionskode, før der er skrevet en fejlende test
2. Der må ikke skrives mere i testen, end nødvendigt for, at den kan fejle
3. Der må ikke skrives mere produktionskode, end nødvendigt for succes ved testen

Herefter argumenteres for, at man kan få dynger af tests, der skal vedligeholdes og hvor mange af dem med tiden bliver overflødige.

Noget af det første, der blev skrevet til Voresjazzklub efter TDD-principperne, var en test af, om der var hul igennem til den database, der holder information om brugerne (MySQLIdentity). Hvis det nu var sådan, at tabellerne skulle flyttes til en anden database, skal testen justeres til den nye database før den eksisterer. Herefter vil testen fejle og gå godt, når databasen er oprettet.

Når projektet er kommet så langt, at det kunne være relevant at ændre på MySQLIdentity-databasen, vil der være adskille andre test, der tilgår databasen i forbindelse med oprettelse af brugere. De test vil så overlappet hullgennemTillIdentityDB, hvorfor den så er overflødig.

Der vil ikke være nogen grund til at have overflødige test i projektet, derfor skal de fjernes, når man er sikker på at testen bliver dække af en eller flere andre test.

Med en passende code-coverage vil TDD gøre, at man kan være rimeligt sikker på at være på rette vej med de ting, der bliver udviklet og ændret.

Hver gang der er skrevet en mængde kode, der synes at være i orden, køres samtlige test igen. I et lidt større setup, hvor flere personer skal arbejde på samme projekt, vil det være på sin plads at anvende en maskine kun til byg, som bygger i Release/Produktion-modus og kører alle test.

Databaseaktiviteterne er blevet testet med et antal oprettelser af rækker, læsning af én række, læsning af flere rækker, ændring af en værdi, og sletning af en/flere rækker.

Et testprojekt kører med sine egen konfigurationsfil, hvilket muliggør anvendelse af en dedikeret testdatabase. I alle test fjernes først alle rækker og derefter indsættes kendte værdier, som efterfølgende skal kunne genfindes, ændres og slettes. Det skal undersøges om handlingerne rent faktisk bliver gennemført som forventet.

I Visual Studio opretter man en testklasse, der annoteres [`TestClass`] og der skrives et antal testmetoder i den klasse. De annoteres [`TestMethod`]. Undersøgelsen af validiteten gøres med klassen `Assert`, der indeholder et antal metoder som f.eks. `IsTrue` og `IsFalse`. Disse sammenligner to værdier og smider en exception, hvis ikke betingelsen er opfyldt.

Her er en af de underrutiner, der anvendes som et af skridtene i test af alle CRUD-operationerne. Først tømmes tabellen og det tjekkes om den er tom. Herefter oprettes fire rækker i tabellen, og det tjekkes om der så er fire elementer i listen, der hentes i databasen.

```

private void createFourEntries() {
    emptyTable();
    isUserListEmpty();

    rip.createUser();
    rap.createUser();
    rup.createUser();
    andersAnd.createUser();

    UsersTableModel usersTableModel = new UsersTableModel();
    List<UsersTableModel> utms = usersTableModel.getUserList();
    Assert.IsTrue(utms.Count == 4, "Der skulle være 4 personer i Users. Der er " + utms.Count);
}

```

Her en af de testmetoder, der gennemløber alle de mulige CRUD-operationer. Først oprettes fire rækker i tabellen med ovenstående metode. Derefter forsøges det at læse én række og en hel liste. Umiddelbart kan det måske synes lidt overflødig at hente en liste, for det er jo allerede gjort i ovenstående metode, men readList() er lidt grundigere. Den ser specifikt efter tre af rækkerne. Så opdateres hhv. slettes en række. Hver gang tjekkes med en `Assert`.

```

[TestMethod]
public void TestUsersCRUD() {
    createFourEntries();
    readEntry();
    readList();
    updateEmail();
    deleteEntry();
}

```

Selenium kom ikke til at fungere. Derfor var det nødvendigt at afprøve websiderne manuelt. I en så lille applikation, er det heller ikke noget større problem, men hvis der havde være flere personer om at bygge app'en, og hvis den var større, ville det være svært at overskue alle konsekvenser af ændringer, hvilket så medfører et tidskrævende testarbejde.

noget. Det tæller som fejl.

Statisk test bliver udført af IDE'et. Her er en metode, der ikke altid returnerer

```

private int manglerReturnValue() {
    int a=12;
    if(a==42) {
        return a
    }
}

```

'LogonModelsController.manglerReturnValue()': not all code paths return a value

```

private int manglerReturnValue() {
}

```

'LogonModelsController.manglerReturnValue()': not all code paths return a value

```

private int manglerReturnValue() {
    int a;
    if(a==42) {
    }
    reti
}

```

Use of unassigned local variable 'a'

```

0 references | 0 changes | 0 authors, 0 changes
private int ingenReferencer() {
    return 42;
}

```

Nu returneres der rent faktisk noget, men variablen, der opereres på, har ikke fået tildelt en værdi

Dette er så endelig en metode, der ikke får røde bølgelinjer, men så ikke har nogle referencer. Det er ikke umiddelbart en fejl, men der kommer en lille reminder. På de public members, der aktiveres via URL'en, vil der formentlig også stå "0 references", da metoderne ikke kaldes direkte, men derimod via de sammenhænge, der er skabt i RouteConfig.cs.

Der er en mere dybdegående statisk analyse, som giver en længere række warnings. I en produktionsversion af produktet er hverken fejl eller advarsler acceptable. Der er to muligheder: Enten at rette i koden ud fra beskrivelsen, eller at konfigurere analyseværktøjet til ikke give bestemte advarsler. Det sidste skal naturligvis kun ske, hvis der kan argumenteres forsvarligt for det.

Det videre forløb

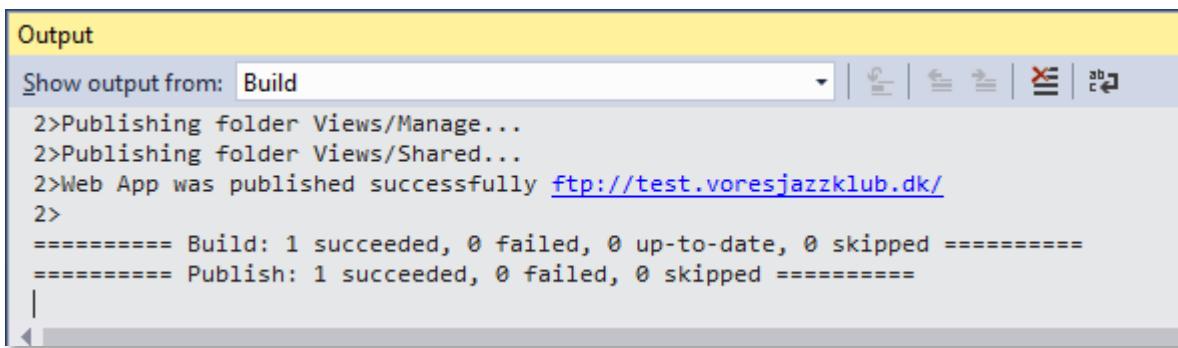
Der er flere områder, der skal arbejdes videre på. For at kunne køre TDD hele vejen, skal Selenium anvendes, så der kan testes helt ude fra glaspladen. Pt. er der et problem med at kunne få frameworkt til at fungere sammen med nærværende web-projekt, men da det er open source, vil der være en principiel mulighed for selv at foretage de

ændringer, der er nødvendige. Håbet er dog, at det er noget, som kan klares med noget konfigurering eller på anden nem vis.

Microsofts Entity Framework skal også lære at arbejde med MySQL. Alternativt kunne man muligvis finde et andet framework på nettet eller bygge et selv. Det vil være en noget omfattende opgave, men absolut ikke umulig. Ved hjælp af reflection, kan man finde en klasses properties og deres datatyper. Ud fra dem kan man "nemt" generere det DDL, der skal anvendes for at oprette tilhørende tabeller, og det SQL, der skal til for CRUD-operationerne på tabellerne.

Det er upraktisk at have to løsninger til adgangskontrol. Da Microsofts løsning rimeligvis er gennemprøvet og blevet sikker og stabil, vil det være at foretrække at anvende den løsning. Det kræver dog, at løsningen kan justeres til ønskerne til app'en.

Deployment gik tilsyneladende godt, men applikationen kunne ikke køre på test.voresjazzklub.dk, kun lokalt på udviklingsmaskinen. Det skal undersøges, hvorfor der er forskellige opfattelser.



The screenshot shows the 'Output' window in Visual Studio. The 'Show output from' dropdown is set to 'Build'. The log output is as follows:

```
2>Publishing folder Views/Manage...
2>Publishing folder Views/Shared...
2>Web App was published successfully ftp://test.voresjazzklub.dk/
2>
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
```

Konklusion

Der er fremstillet en web-applikation, der kan håndtere for emnet relevante data. Data persisteres i en database og der er frembragt kode til oprettelse, ændring og sletning af data på individniveau. Der er kode til læsning af enkeltelementer og, hvor det har været gavnligt, som lister.

Applikationen er afprøvet delvist maskinelt og manuelt. Funktionaliteten, der skal anvendes, er til stede og virker. Hovedformålene (fra afsnitte Problemformulering, Afgrænsning og Teknologivalg) med opgaven, at lære MVC-frameworket at kende, at anvende nyere teknologier som HTML5 og CSS3 og at fremstille en dynamiske web-applikation på walking skeleton-niveau, er nået.

Der er givetvis flere forhold i applikationen, der med fordel kan ændres. F.eks. er betjeningen ikke nødvendigvis særlig UX-venlig. Det skønnes, at der er plads til forbedringer, selvom der også er lækkre ting imellem, som f.eks. listen af små billeder, hvor de enkelte billeder kan forstørres ved blot at holde musen hen over dem.

Datamodellen ville være anderledes, hvis Identity framework var kendt på forhånd. Alternativt skulle der have været brugt en del tid på at justere i frameworket for at akkommoderer den valgte datamodel.

ASP.NET MVC kan meget, så det er ikke et framework man sætter sig ind i på en weekend. MVC fungerer godt til at afkoble de enkelte dele. Frontend og backend kender ikke hinanden, og kommunikerer via en datastruktur, som Controller flytter rundt mellem dem.

Microsoft har så fået indført andre bindinger og begrænsninger. En Controller skal navngives med Controller som de sidste bogstaver i navnet, men referencer til den er uden Controller til sidst. Parameternavnene i Controllerens metoder, skal matche dem, der anføres i RouteConfig. Det er noget forskellig fra normalen. Her er parameternavne udelukkende kendt inden for metoden.

Alt i alt virker ASP.NET MVC som et godt værktøj at bygge web-applikationer i, når man har vænnet sig til arbejdsgangene.

Litterarurliste

- [1] <http://hvemkommer.dk/> ejet af Billetto, sidst besøgt 1. marts 2017
- [2] <https://da.wikipedia.org/wiki/HTML5> Wikipedia, blev offentliggjort oktober 2014, siden besøgt 1. marts 2017, hvor sidste opdatering var: "Denne side blev senest ændret den 21. oktober 2016 kl. 16:07"
- [3] http://www.web10.dk/v6_information/pricelist, talkactive, det koster 8 kr. pr. md. og det kan en fattig studerende ikke klare.WEB10.dk er hostingselskabet som lægger severe til sitet. Prisen er sidst set 1. marts 2017
- [4] <https://msdn.microsoft.com/en-us/library/system.security.cryptography.sha512managed%28v=vs.110%29.aspx>, Microsoft, set 16. marts 2017
- [5] Clean Code ISBN 978-0-13-235088-4, Robert C. Martin, Prentice Hall 2013
- [6] http://www.web10.dk/v5_order/packs_basic, talkactive, sidst besøgt 1. marts 2017
- [7] <https://github.com/ClausTetens/WEB01Projekt> Github, indeholder sourcen til web-sitet
- [8] <https://msdn.microsoft.com/en-us/library/ms182776.aspx>, Microsoft, set 16. marts 2017
- [9] <https://dev.mysql.com/doc/refman/5.7/en/fractional-seconds.html>, Oracle, set 16. marts 2017
- [10] https://www.w3schools.com/sql/sql_datatypes.asp, Refsnes Data, set 16. marts 2017
- [11] <https://www.ischool.utexas.edu/~wyllys/DMPAMaterials/normstep.html>, University of Texas at Austin, set 16. marts 2017
- [12] Fundamentals of Database Systems, ISBN 0-201-53090-2, Elmasri/Navathe, The Benjamin/Cummings Publishing Company Inc, 1989
- [13] <https://dev.mysql.com/doc/refman/5.7/en/numeric-type-overview.html>, set 16. marts 2017
- [14] <http://stackoverflow.com/questions/8437957/difference-between-3nf-and-bcnf-in-simple-terms-must-be-able-to-explain-to-an-8>, Joel Spolsky, Jeff Atwood, Stack Overflow ,set 16. marts 2017
- [15] <https://www.apachefriends.org/index.html>, BitRock ,besøgt 15. april 2017
- [16] <https://docs.microsoft.com/en-us/aspnet/identity/overview/extensibility/implementing-a-custom-mysql-aspnet-identity-storage-provider>, Microsoft, set 15. april 2017
- [17] <https://aspnet.codeplex.com/SourceControl/latest#Samples/Identity/AspNet.Identity.MySQL/>, Microsoft, set 15. april 2017
- [18] <https://www.ics.com/files/docs/qicstable/2.4/mvc.jpg>, ICS, set 1. april 2017 og indgår på siden <https://www.ics.com/files/docs/qicstable/1.x/arch.html>
- [19] <https://martinfowler.com/articles/continuousIntegration.html>, Martin Fowler, set 16. april 2017
- [20] <https://msdn.microsoft.com/en-us/library/ms182538%28v=vs.90%29.aspx>, Microsoft, 16. april 2017
- [21] <https://blogs.msdn.microsoft.com/visualstudioalm/2016/01/27/getting-started-with-selenium-testing-in-a-continuous-integration-pipeline-with-visual-studio/>, Microsoft, set 16. April 2017
- [22] <http://wiki.c2.com/?WalkingSkeleton>, Ward Cunningham , set 13. maj 2017
- [23] https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.2.0/com.ibm.zos.v2r2.hala001/itctcpipcon.htm, IBM, set 13. maj 2017
- [24] <https://www.torproject.org/>, The Tor Project Inc., set 13. maj 2017
- [25] <https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt>, The Tor Project Inc., set 16. maj 2017
- [26] <https://www.torproject.org/docs/hidden-services.html.en>, The Tor Project Inc., set 16. maj 2017
- [27] <https://stackoverflow.com/questions/4121629/password-encryption-at-client-side>, indlæg nummer 91 af Gareth, 8. nov. 2010
- [28] <http://www.infosecurity.dk/newsitem/21400>, INFORSECURITY.DK, set 16. maj 2017
- [29] <https://www.computerworld.dk/art/235216/danskerne-sloeser-med-sikkerheden-vi-genbruger-passwords-alt-for-mange-steder>, Joachim Kühlmann Selliken, 15. okt. 2015
- [30] <http://www.darkreading.com/safely-storing-user-passwords-hashing-vs-encrypting/a/d-id/1269374>, Michael Coates, 4. juni 2014
- [31] <http://www.passwordrandom.com/most-popular-passwords>, Koshevoy Dmitry, muligvis i 2014 (set 16. maj 2017)
- [32] <https://www.youtube.com/watch?v=fCNOIheD-Is>, ???, 10. juli 2013
- [33] <https://www.youtube.com/watch?v=7U-RbOKanYs>, Dr Mike Pound, 13. juli 2016
- [34] <http://ootips.org/mvc-pattern.html>, citater fra Dean Helman, set 20. maj 2016
- [35] https://www.owasp.org/index.php>Password_Storage_Cheat_Sheet, The Open Web Application Security Project, set 20. maj 2017
- [36] <https://www.cloudflare.com/ddos/>, Cloudflare, set 21. Maj 2017
- [37] <https://de.wikipedia.org/wiki/Diffie-Hellman-Schl%C3%BCsselauttausch>, wikipedia, set 21. maj 2017

[38] <https://msdn.microsoft.com/en-us/library/ff648641.aspx>, Microsoft, set 21. maj 2017

Følgende henvisninger er til sider, der har bidraget med afklaringer og produkter, men som ikke direkte indgår i rapporten.

- [100] <https://dev.mysql.com/doc/connector-net/en/connector-net-installation-binary-windows-installer.html>, set 16. april 2017
- [101] <http://machinesaredigging.com/2013/10/29/how-does-a-web-session-work/>, set 16. april 2017
- [102] https://en.wikipedia.org/wiki/List_of_HTTP_status_codes, set 16. april 2017
- [103] http://www.prideparrot.com/blog/archive/2012/7/understanding_routing, set 16. april 2017
- [104] <https://www.asp.net/identity>, set 16. april 2017
- [105] <https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/>, set 16. april 2017
- [106] <https://www.apachefriends.org/index.html>, set 16. april 2017
- [107] <https://www.asp.net/downloads>, set 16. april 2017
- [108] <https://dev.mysql.com/downloads/connector/net/5.2.html>, set 1. april 2017
- [109] <https://www.codeproject.com/Articles/822392/Connecting-to-MySQL-from-ASP-NET-MVC-using-Visual>, set 1. april 2017
- [110] <https://www.mysql.com/why-mysql/windows/visualstudio/>, set 1. april 2017

Appendix

Offentlig og privat nøgle

Dette afsnit er kopieret fra
<http://www.onebigfluke.com/2013/11/public-key-crypto-math-explained.html>
dog er ikke hele web-siden med her

One Big Fluke

24 November 2013

@ [12:02](#)

[Simplest explanation of the math behind Public Key Cryptography](#)

I was trying to explain [public key cryptography](#) today and totally failed at it. Here are notes to myself based on various Wikipedia pages. There's a lot more to it than this (like [padding](#)) but this is the gist of it.

1. Pick two prime numbers:

```
p = 7  
q = 13
```

2. Multiply them together:

```
n = p * q  
n = 7 * 13  
n = 91
```

3. Compute Euler's "[totient](#)" function of n :

```
 $\phi(n) = (p - 1) * (q - 1)$   
 $\phi(91) = (7 - 1) * (13 - 1)$   
 $\phi(91) = 6 * 12$   
 $\phi(91) = 72$ 
```

4. Pick a random number e such that:

```
1 < e < φ(n)
1 < e < φ(91)
1 < e < 72
```

and e is "[coprime](#)" with $\varphi(n)$, meaning it has no common factors.

```
e = 23 (because I said so)
```

5. Compute d , the [modular multiplicative inverse](#) of $e \pmod{\varphi(n)}$:

```
e^-1 = d (mod φ(n))
23^-1 = d (mod φ(91))
23^-1 = d (mod 72)
23 * d = 1 (mod 72)
23 * 47 = 1 (mod 72)
d = 47
```

Now you have all the magic numbers you need:

```
public key = (n = 91, e = 23)
private key = (n = 91, d = 47)
```

Secret messages to you

1. Have someone else encrypt a message m using your public key (n, e) :

```
m = 60
c(m) = m^e mod n
c(60) = 60^23 mod 91
c(60) = 44
c = 44 (they send this to you)
```

2. Decrypt the message c using your private key (n, d) :

```
c = 44
m(c) = c^d mod n
m(44) = 44^47 mod 91
m(44) = 60
m = 60 (now you have the secret)
```

Prove you wrote something

1. [Hash](#) the message to broadcast (this is Knuth's insecure hash function):

```
broadcast = 102
hash(broadcast) = broadcast * K mod 2^32
hash(102) = 102 * 2654435761 mod 2^32
hash(102) = 169507974
```

2. Compute the signature with your private key (n, d) :

```

signature = hash(broadcast) ^ d mod n
signature = hash(102) ^ 47 mod 91
signature = 169507974 ^ 47 mod 91
signature = 90

```

3. People who receive the broadcast message (102) and the signature (90) can confirm with your public key (n , e):

```

broadcast = 102
hash(broadcast) = broadcast * K mod 2^32
hash(102) = 102 * 2654435761 mod 2^32
hash(102) = 169507974
confirmation = hash(broadcast)^e mod n
confirmation = 169507974^23 mod 91
confirmation = 90 (matches your signature)

```

Why is this safe?

In this simple example it's totally not. You can trivially take $n = 91$ and guess p and q with simple [factorization](#):

```
>>> set((91 / i) for i in xrange(1, 91) if 91 % i == 0)
set([91, 13, 7])
```

Redo all the math above and you have the private key. *Doh~*

Now imagine if n was such a large number that it took a very long time to guess p and q :

```
>>> set((12031294782491844L / i) for i in xrange(1, 12031294782491844L) if
12031294782491844L % i == 0)
```

This takes a really long time. Even fancy solutions on the fastest computer on Earth would take until the end of the universe. That's why transmitting secrets with public key cryptography is safe.

© 2009-2017 Brett Slatkin

Wireshark output

Et opslag på dr.dk uden VPN viste Wireshark sådan med forskellige protokoller og IP-adresser:

No.	Time	Source	Destination	Protocol	Length	Info
781	4.617037	193.162.153.164	192.168.0.11	DNS	388	Standard query response 0x4a55 A e4578.g.akamaiedge.net A 2.16.19.169 NS n0g.akamaiedge.net
782	4.621526	192.168.0.11	195.137.195.8	HTTP	771	GET /download/Forside/2016/Koncerthuset/Blaamarts/940/index.aframe HTTP/1.1
783	4.636167	195.137.195.8	192.168.0.11	HTTP	1348	HTTP/1.1 200 OK (text/html)
784	4.653749	192.168.0.11	159.20.6.22	TCP	54	53546 + 80 [ACK] Seq=15628 Ack=58896 Win=42240 Len=0
785	4.666666	192.168.0.11	193.162.153.164	DNS	82	Standard query 0xa2dc AAAA e4578.g.akamaiedge.net
786	4.669693	192.168.0.11	194.239.134.83	DNS	73	Standard query 0x49f0 A pcastol.dr.dk
787	4.677451	193.162.153.164	192.168.0.11	DNS	143	Standard query response 0xa2dc AAAA e4578.g.akamaiedge.net SOA n0g.akamaiedge.net
788	4.683688	194.239.134.83	192.168.0.11	DNS	219	Standard query response 0x49f0 A pcastol.dr.dk A 195.137.195.8 NS dns101.telia.com
789	4.686763	192.168.0.11	195.137.195.8	TCP	54	53568 + 80 [ACK] Seq=718 Ack=1295 Win=15104 Len=0
790	4.695128	192.168.0.11	104.77.230.232	TCP	66	53574 + 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
791	4.695389	192.168.0.11	195.137.195.8	TCP	66	53575 + 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
792	4.706036	104.77.230.232	192.168.0.11	TCP	66	80 + 53574 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
793	4.706185	192.168.0.11	104.77.230.232	TCP	54	53574 + 80 [ACK] Seq=1 Ack=1 Win=16384 Len=0
794	4.715333	195.137.195.8	192.168.0.11	TCP	66	80 + 53575 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
> Frame 783: 1348 bytes on wire (10784 bits), 1348 bytes captured (10784 bits) on interface 0						
> Ethernet II, Src: Netgear_6a:fe:22 (30:46:9a:6a:fe:22), Dst: WistronN_28:f1:00 (90:a4:de:28:f1:00)						
> Internet Protocol Version 4, Src: 195.137.195.8, Dst: 192.168.0.11						
0100 = Version: 4						
.... 0101 = Header Length: 20 bytes (5)						
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)						
Total Length: 1334						
Identification: 0x699c (27036)						
> Flags: 0x02 (Don't Fragment)						
Fragment offset: 0						
Time to live: 122						
Protocol: TCP (6)						
Header checksum: 0xae0 [validation disabled]						
[Header checksum status: Unverified]						
Source: 195.137.195.8						
Destination: 192.168.0.11						
0000	90 a4 de 28 f1 00	30 46 9a 6a fe 22	08 00 45 00	...	(..0F .j."..E.	
0010	05 36 69 9c 40 00	7a 06 4a e0 c3 89	c3 08 c0 a8	.6i.@.z. J.....		
0020	00 0b 00 50 d1 40	d4 ef bc ad dc 22	b7 23 50 18	...P@.#P.		
0030	01 00 00 77 00 00	48 54 54 50 2f 31	2e 31 20 32	...w..HT TP/1.1 2		
0040	30 30 20 4f 4b 0d	0a 43 6f 6e 74 65	66 74 2d 54	00 OK..C ontent-T		
0050	79 70 65 3a 20 74	65 78 74 2f 68 74	6d 6c 0d 0a	ype: tex t/html..		
0060	4c 61 73 74 2d 4d	6f 64 69 66 69 65	64 3a 20 54	Last-Mod ified: T		
0070	75 65 2c 20 30 37	20 4d 61 72 20 32	30 31 37 20	ue, 07 M ar 2017		
0080	30 39 3a 35 34 3a	30 32 20 47 4d 54	00 0a 41 63	09:54:02 GMT..Ac		
0090	63 65 70 74 2d 52	61 6e 67 65 73 3a	20 62 79 74	cept-Ran ges: byt		
00a0	65 73 0d 0a 45 54	61 67 3a 20 22 30	31 39 64 34	es..ETag : "019d4		
00b0	62 66 32 38 39 37	64 32 31 3a 30 22	0d 0a 53 65	bf2897d2 1:0" ..Se		

I denne session blev VPN anvendt. Der var et par ARP requests, men ellers var al trafikken mellem pc'en (192.168.0.11) og VPN-severen (83.170.84.216), og protokollerne ikke afslørede meget.

Apply a display filter ... <Ctrl-/>						
No.	Time	Source	Destination	Protocol	Length	Info
3793	150.787842	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3794	150.842789	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP
3795	150.940386	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP
3796	151.388933	83.170.84.216	192.168.0.11	PPP Co...	316	Compressed data
3797	151.393680	83.170.84.216	192.168.0.11	PPP Co...	316	Compressed data
3798	151.393907	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3799	151.471140	83.170.84.216	192.168.0.11	PPP Co...	318	Compressed data
3800	151.488390	192.168.0.11	83.170.84.216	GRE	46	Encapsulated PPP
3801	151.517513	83.170.84.216	192.168.0.11	PPP Co...	322	Compressed data
3802	151.517824	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3803	151.673372	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP
3804	151.917886	83.170.84.216	192.168.0.11	PPP Co...	316	Compressed data
3805	151.968034	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3806	152.121352	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP
> Frame 3803: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface 0						
> Ethernet II, Src: Netgear_6a:fe:22 (30:46:9a:6a:fe:22), Dst: WistronN_28:f1:00 (90:a4:de:28:f1:00)						
< Internet Protocol Version 4, Src: 83.170.84.216, Dst: 192.168.0.11						
0100 = Version: 4						
.... 0101 = Header Length: 20 bytes (5)						
> Differentiated Services Field: 0x08 (DSCP: Unknown, ECN: Not-ECT)						
Total Length: 32						
Identification: 0xe440 (58432)						
Flags: 0x02 (Don't Fragment)						
Fragment offset: 0						
Time to live: 50						
Protocol: Generic Routing Encapsulation (47)						
Header checksum: 0xfb30 [validation disabled]						
[Header checksum status: Unverified]						
Source: 83.170.84.216						
Destination: 192.168.0.11						
0000	90 a4 de 28 f1 00 30 46	9a 6a fe 22 08 00 45 08(0F .j."..E.		
0010	00 20 e4 40 40 00 32 2f	fb 30 53 aa 54 d8 c0 a8	.. .@0.2/.05.T...			
0020	00 0b 20 81 88 0b 00 00	c9 e3 00 00 08 a8 00 00	...[REDACTED]..			
0030	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00			

I nedenstående billede har Wireshark opsnippet en logon-dialog over en ikke-krypteret forbindelse.

Kunne man så ikke blot forvanske passwordet på browsersiden? Jo, det kan man godt, men det vil kun gøre livet marginalt vanskeligere for en blackhat. Det forvanskede password kan jo opsnappes

The screenshot shows a Wireshark capture of a logon dialog. The packet list shows three TCP packets: one from 195.128.175.65 to 192.168.0.11 (TCP port 53858), one from 192.168.0.11 to 195.128.175.65 (TCP port 80), and one from 192.168.0.11 to 195.128.175.65 (HTTP POST /Login.a). The details pane shows the form items: UserID = JegErBruger, Password = OgHarEtPassword, and Logon = Logon. The bytes pane shows the raw hex and ASCII data, with the password field highlighted.

Lidt om, hvordan programkoden bliver til

Til projektet er der ikke kopieret kode direkte, men der er ofte kun en måde, når der skal vælge fra best practice. Ofte kan man finde code sippets og anbefalinger på nettet. I nærværende projekt er det typisk hjælpesteder fra Microsoft, der er leverandør af bl.a. C# og .Net, og uafhængige sites som Stackoverflow.

F.eks. er koden til SHA512-hashing delvis hentet hos Microsoft [4] og ser sådan ud:

```
byte[] data = new byte[DATA_SIZE];
byte[] result;

SHA512 shaM = new SHA512Managed();
result = shaM.ComputeHash(data);
```

Der var et ønske om af opbevare forvanske data i datatypen string. Best practice for konvertering af et array af bytes, er, at løbe arrayet gennem med foreach og opbygge en streng med StringBuilder. Med tanker om læsbarhed og vedligeholdelsesvenlighed hentet i f.eks. Clean Code bogen [5], er rammerne relativt faste. Derfor koden snublende nær:

```
string getHashSha512(string stringToBeHashed) {
    byte[] bytesToBeHashed = Encoding.Unicode.GetBytes(stringToBeHashed);
    SHA512 sha512 = new SHA512Managed();
    byte[] hashedBytes = sha512.ComputeHash(bytesToBeHashed);
    StringBuilder hashedString = new StringBuilder();
    foreach(byte hashedByte in hashedBytes) {
        hashedString.Append(String.Format("{0:x2}", hashedByte));
    }
    return hashedString.ToString();
}
```

En lidt mere kompakt udgave kunne se sådan ud:

```
string getHashSha512(string stringToBeHashed) {
    const int bitsToCharShiftCount = 2; // 8 bits/byte and 2 characters/byte
    SHA512 sha512 = new SHA512Managed();
    StringBuilder hashedString = new StringBuilder(sh512.HashSize >> bitsToCharShiftCount);
```

```

        foreach(byte hashedByte in sha512.ComputeHash(Encoding.Unicode.GetBytes(stringToBeHashed))) {
            hashedString.Append(String.Format("{0:x2}", hashedByte));
        }
        return hashedString.ToString();
    }
}

```

eller sådan:

```

string getHashSha512(string stringToBeHashed) {
    StringBuilder hashedString = new StringBuilder();
    foreach(byte hashedByte in new SHA512Managed().ComputeHash(Encoding.Unicode.GetBytes(stringToBeHashed))) {
        hashedString.Append(String.Format("{0:x2}", hashedByte));
    }
    return hashedString.ToString();
}

```

Der er ikke gjort nogle forsøg på at finde en tilsvarende kodestumper på nettet, men der er en forventning om, at det kan lade sig gøre.

Visual Studio 2015, som er Microsofts IDE til bl.a. C# .Net, indeholder flere templates, herunder et komplet, fungerende web-site. Det kunne med fordel vælges, hvis man skal hurtigt i gang, idet der er anvendt gennemtænkte løsninger til de forskellige funktioner. Ulempen er så, at det det er noget omfattende og måske kan være for omfattende at finde rundt i, hvilket kan lede til fejtagelser og være tidskrævende.

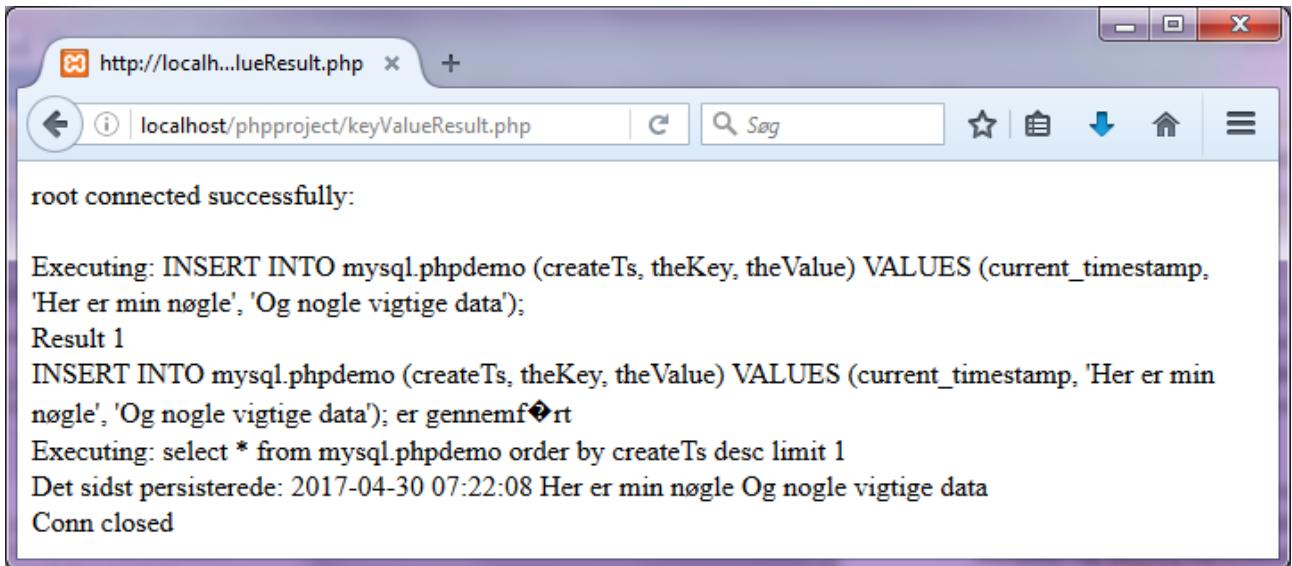
Noget hurtig PHP

Nedenstående PHP-scripts er lidt noget rod at se på og det eksisterer da også kun for at kunne se, hvordan PHP og processen med at få en LAMP/WAMP-applikation til at køre i praksis fungerer. Det hele (incl. oprettelse af tabellen) har taget en lille halv time at fremstille og deployment består også i en meget simpel operation: Kopier PHP-filen til den folder, Apache skal læse den fra.

Nedenstående to skærmdumps viser resultatet af de to PHP-scripts, der er gengivet derunder. En HTML form tager to tekst-input og submitter til et modtager-script. Det persisterer de indtastede data sammen med current_timestamp. Hvorefter nyeste række læses og returneres.

Man kommer hurtigt fra nul til noget anvendeligt. Hvis det er meningen, at det også skal kunnet vedligeholdes, bør man nok tænke struktur og genbrug. Hvis der ligefrem også er krav om, at det skal se pænt ud og sikret mod fejl, så medgår der en del mere tid på styling og validering af data.

The screenshot shows a web browser window with the URL `localhost/phpproject/keyupvalueform.php`. The page contains a simple form with two input fields: 'Key' and 'Value'. The 'Key' field contains the value 'Her er min nøgle' and the 'Value' field contains 'Og nogle vigtige data'. Below the inputs is a button labeled 'Send forespørgsel' (Send query). The browser interface includes standard navigation buttons, a search bar, and a menu bar.



KeyValueform.php

```
<html><body>
<form method="post" action="/phpproject/keyValueResult.php">
<dl>
<dt>Key</dt>
<dd><input name="akey" type="text"></dd>
<dt>Value</dt>
<dd><input name="avalue" type="text"></dd>
</dl>
<input type="submit">
</form>
</body></html>
```

KeyValueResult.php

```
<html><body>
<?php
$servername = "localhost";
$username = "root";
$password = "";

$conn = new mysqli($servername, $username, $password);
mysqli_select_db($conn, 'mysql');

if (!$conn) {
    die("Connection failed: " . $conn->connect_error);
} else
    echo $username . " connected successfully: " . $conn->connect_error . "<br/><br/>";

$akey = $_POST["akey"];
$avalue = $_POST["avalue"];

$sql="INSERT INTO mysql.phpdemo (createTs, theKey, theValue) VALUES (current_timestamp,
'$akey', '$avalue');";

echo "Executing: " . $sql;
printf("<br/>");

$result=$conn->query($sql);
```

```

printf("Result %s<br/>", $result);
if($result) { printf("%s er gennemført<br/>", $sql); }
else { printf("%s fejlede<br/>", $sql); }

$sql="select * from mysql.phpdemo where createTs=(select max(createTs)) from
mysql.phpdemo";
$sql="select * from mysql.phpdemo order by createTs desc limit 1";
echo "Executing: " . $sql;
printf("<br/>");

$result=$conn->query($sql);
if($result) {
    echo "Det sidst persisterede: ";
    while($row = mysqli_fetch_array($result)) {
        for($i=0; $i<3; $i++) {
            printf("%s ", $row[$i]);
        }
        printf("<br/>");
    }
}
echo $conn->close()?"Conn closed":"Error closing connection";

?>
</body></html>

```

Nedenstående PHP-script er noget rod at se på og det eksisterer da også kun for at kunne se, hvordan PHP og processen med at få en LAMP/WAMP-applikation til at køre i praksis fungerer. Scriptet er skrevet på få minutter og deployment består også i en meget simpel operation: Kopier PHP-filen til den folder, Apache skal læse den fra.

Scriptet opretter en række i en tabel, læser den tilbage og display'er den på en web-side.

```

<html><body>
<?php
echo "php is running<br/><br/><br/><br/>";
$servername = "localhost";
$username = "claus";
$password = "";

echo "username $username<br/>";
echo 'username $username<br/>';

$conn = new mysqli($servername, "root", "");
mysqli_select_db($conn, 'mydb');

if (!$conn) {
    die("Connection failed: " . $conn->connect_error);
} else
    echo $username . " connected successfully: " . $conn->connect_error . "<br/><br/>";

$sql="INSERT INTO mysql.user (host, user, password, select_priv, insert_priv,
update_priv) VALUES ('localhost', 'Gudrun',PASSWORD('guest123'), 'Y', 'Y', 'Y');";

```

```

echo "Executing: " . $sql;
printf("<br/>");

$result=$conn->query($sql);
printf("Result %s<br/>", $result);
if($result) { printf("%s er gennemført<br/>", $sql); }
else { printf("%s fejlede<br/>", $sql); }

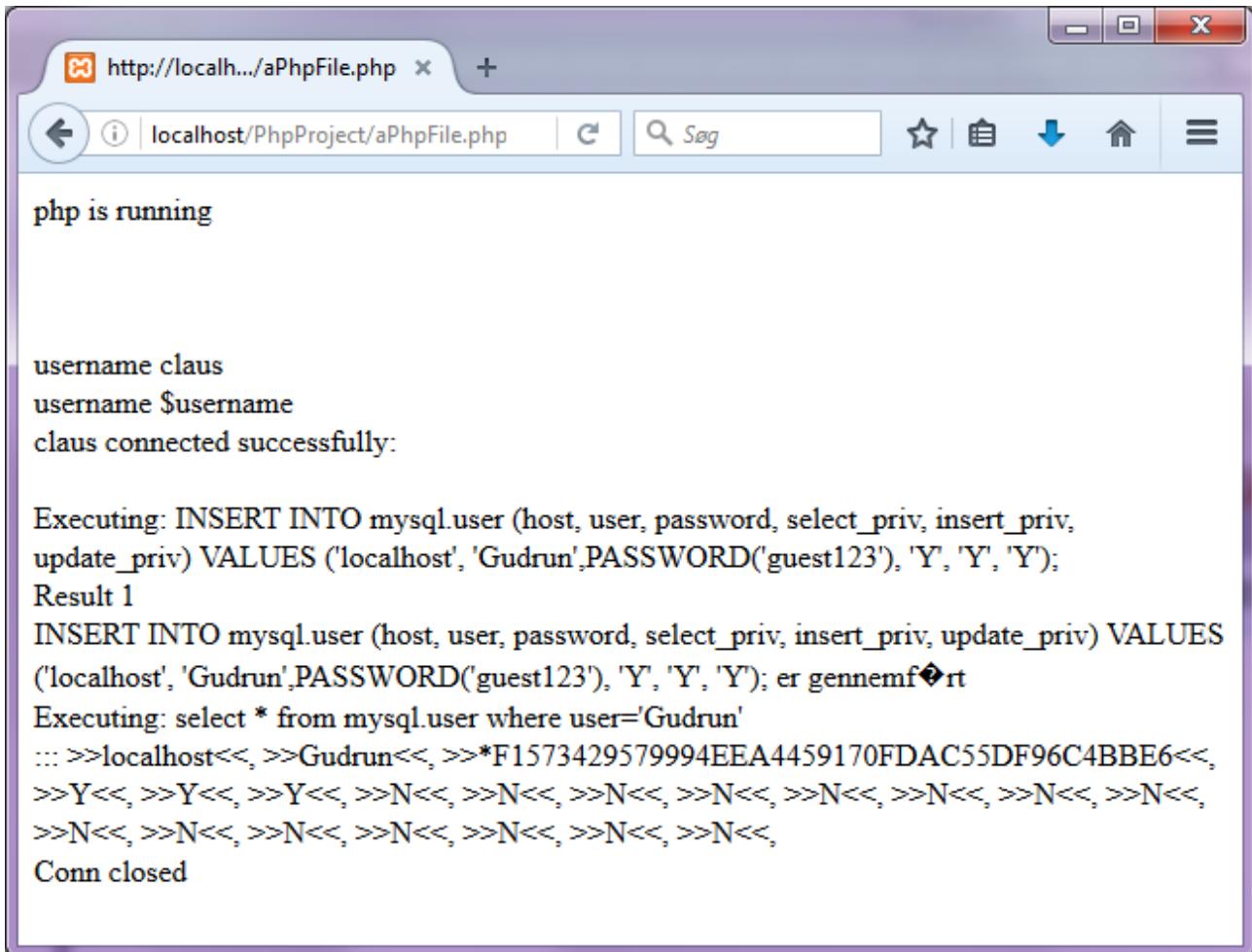
$sql="select * from mysql.user where user='Gudrun'";
echo "Executing: " . $sql;
printf("<br/>");

$result=$conn->query($sql);
if($result) {
    while($row = mysqli_fetch_array($result)) {
        printf("::::");
        for($i=0; $i<21; $i++) {
            printf("\t>%s<<, ", $row[$i]);
        }
        printf("<br/>");
    }
} else {
}

echo $conn->close()?"Conn closed":"Error closing connection";
?>
</body></html>

```

Kønt er barnet ikke, men der kommer data i en tabel, data bliver læst tilbage og vist på glaspladen. Browseren skal vist også have at vide, at det nok ikke er UTF-8, men muligvis ISO8859-1 der anvendes.



Noget hurtig ASP.NET

ASP.NET i traditionel udgave består primært af to filer på side. Den ene holder html-tags, og den anden, som bliver refereret til via CodeBehind, holder den C#-kode der skal afvikle forretningslogikken.

Kommunikationen mellem web-siden og koden bagved foregår via tags i namespace asp som f.eks. nedenfor, hvor <asp:Contents... fortæller, at her er der noget C#-koden skal forholde sig til. Alle html-elementer placeres i asp-namespace, så compileren kan knytte en begivenhed på web-siden sammen med handleren.

<%: Title %> bliver erstattet med "Contact" fra øverste linje. En konstruktion som f.eks. <%= this.navn %> og en public, læsbar property med navnet navn, vil indsætte C#-klassens property i html-koden.

Contact.aspx

```
<%@ Page Title="Contact" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true"
CodeBehind="Contact.aspx.cs" Inherits="VoresJC.Contact" %>

<asp:Content ID="BodyContent" ContentPlaceHolderID="MainContent" runat="server">
    <h2><%: Title %>.</h2>
    <h3>Your contact page.</h3>
    <address>
        One Microsoft Way<br />
        Redmond, WA 98052-6399<br />
        <abbr title="Phone">P:</abbr>
        425.555.0100
    </address>
</asp:Content>
```

```

<address>
    <strong>Support:</strong> <a href="mailto:Support@example.com">Support@example.com</a><br />
    <strong>Marketing:</strong> <a href="mailto:Marketing@example.com">Marketing@example.com</a>
</address>
</asp:Content>

```

```

Contact.aspx.cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace VoresJC {
    public partial class Contact : Page {
        protected void Page_Load(object sender, EventArgs e) {

        }
    }
}

```

Script til migrering af adgangskontrol

```

create database MySQLIdentity;
use MySQLIdentity;

```

```

CREATE TABLE `roles` (
    `Id` varchar(128) NOT NULL,
    `Name` varchar(256) NOT NULL,
    PRIMARY KEY (`Id`)
);

```

```

CREATE TABLE `users` (
    `Id` varchar(128) NOT NULL,
    `Email` varchar(256) DEFAULT NULL,
    `EmailConfirmed` tinyint(1) NOT NULL,
    `PasswordHash` longtext,
    `SecurityStamp` longtext,
    `PhoneNumber` longtext,
    `PhoneNumberConfirmed` tinyint(1) NOT NULL,
    `TwoFactorEnabled` tinyint(1) NOT NULL,
    `LockoutEndDateUtc` datetime DEFAULT NULL,
    `LockoutEnabled` tinyint(1) NOT NULL,
    `AccessFailedCount` int(11) NOT NULL,
    `UserName` varchar(256) NOT NULL,
    PRIMARY KEY (`Id`)
);

```

```

CREATE TABLE `userclaims` (
    `Id` int(11) NOT NULL AUTO_INCREMENT,
    `UserId` varchar(128) NOT NULL,
    `ClaimType` longtext,
    `ClaimValue` longtext,
    PRIMARY KEY (`Id`),
    UNIQUE KEY `Id` (`Id`),
    KEY `UserId` (`UserId`),
    CONSTRAINT `ApplicationUser_Claims` FOREIGN KEY (`UserId`) REFERENCES `users` (`Id`) ON DELETE CASCADE ON UPDATE NO ACTION
);

```

```

CREATE TABLE `userlogins` (
  `LoginProvider` varchar(128) NOT NULL,
  `ProviderKey` varchar(128) NOT NULL,
  `UserId` varchar(128) NOT NULL,
  PRIMARY KEY (`LoginProvider`, `ProviderKey`, `UserId`),
  KEY `ApplicationUser_Logins` (`UserId`),
  CONSTRAINT ` ApplicationUser_Logins` FOREIGN KEY (`UserId`) REFERENCES `users` (`Id`) ON DELETE CASCADE ON UPDATE NO ACTION
);

CREATE TABLE `userroles` (
  `UserId` varchar(128) NOT NULL,
  `RoleId` varchar(128) NOT NULL,
  PRIMARY KEY (`UserId`, `RoleId`),
  KEY `IdentityRole_Users` (`RoleId`),
  CONSTRAINT ` ApplicationUser_Roles` FOREIGN KEY (`UserId`) REFERENCES `users` (`Id`) ON DELETE CASCADE ON UPDATE NO ACTION,
  CONSTRAINT `IdentityRole_Users` FOREIGN KEY (`RoleId`) REFERENCES `roles` (`Id`) ON DELETE CASCADE ON UPDATE NO ACTION
);

```

Lidt om Reflection

Reflection handler om at et program er stand til at holde øje med sig selv og foretage modifikationer. Det kan bl.a. anvendes til dependency injection og ORM-produkter (Object Relation Mapping).

Herunder er der fire klasser for at illustrere reflection. Den første klasse er en Attribut til annotering af properties. Den næste anvendes blot som datatype til en property. Herefter er der en model, sådan som det anvendes i Entity-frameworket.

Til sidst er der en metode, der instantierer Model-klassen og lægge nogle værdier i properties. Koden løber så gennem de informationer, der er registreret om objektet, og finder ud af, hvad objektets klasse er, hvilke properties, der er annoteret [Key], hvilke properties der i det hele taget er tilgængelige samt deres datatyper, og hvad der er i dem.

Med den information kan man både oprette tabeller og indsætte data i dem. Tabellerne kunne f.eks. få navn efter den klasse, der holder data. De enkelte kolonner i tabellen kan så navngives ud fra navnene på klassens properties., og kolonnernes datatyper skal så matche det, der findes i reflection'en.

Der skal laves en god del mere for at komme i mål med et ORM-produkt, f.eks. hvad skal nok laves en autoNum kolonne, hvis ikke der er en [Key]. Lister skal over i deres egne tabeller, for at kunne overholde 1NF. Navngivning af properties for klasser inden i klassen, skal formentlig udvides med klassenavnet.

```

namespace Reflections {
    public class KeyAttribute : Attribute {
    }

    class Klasse {
        public bool afkrydsning { get; set; }
    }

    class Model {
        public string navn { get; set; }
        public Int32 nummer { get; set; }
        public Klasse klasse { get; set; }
        [Key]
        public Int64 theKey { get; set; }
    }

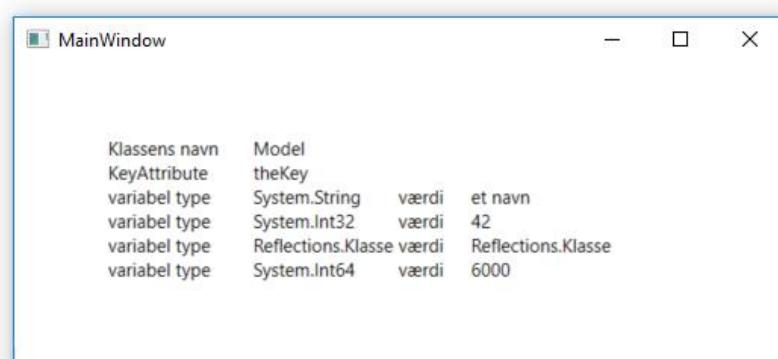
    public class OrmWannaBe {
        public string run() {
            Model model = new Model();
            model.navn = "et navn";
            model.nummer = 42;
            model.klasse = new Klasse();
            model.theKey = 6000;
            Type type = typeof(Model);
            StringBuilder properties=new StringBuilder();

            properties.Append("Klassens navn\t"+type.Name+"\r\n");

            MemberInfo[] memberInfo = typeof(Model).GetMembers();
            for (int i = 0; i < memberInfo.Length; i++) {
                foreach (CustomAttributeData customAttribute in memberInfo[i].CustomAttributes) {
                    if (customAttribute.AttributeType.Name == "KeyAttribute") {
                        properties.Append(customAttribute.AttributeType.Name + "\t" + memberInfo[i].Name + "\r\n");
                    }
                }
            }

            foreach (FieldInfo fieldInfo in type.GetRuntimeFields()) {
                properties.Append("variabel type\t"+fieldInfo.FieldType.ToString() + ("\tværdi\t"+fieldInfo.GetValue(maj 2017)
            }
            return properties.ToString();
        }
    }
}

```



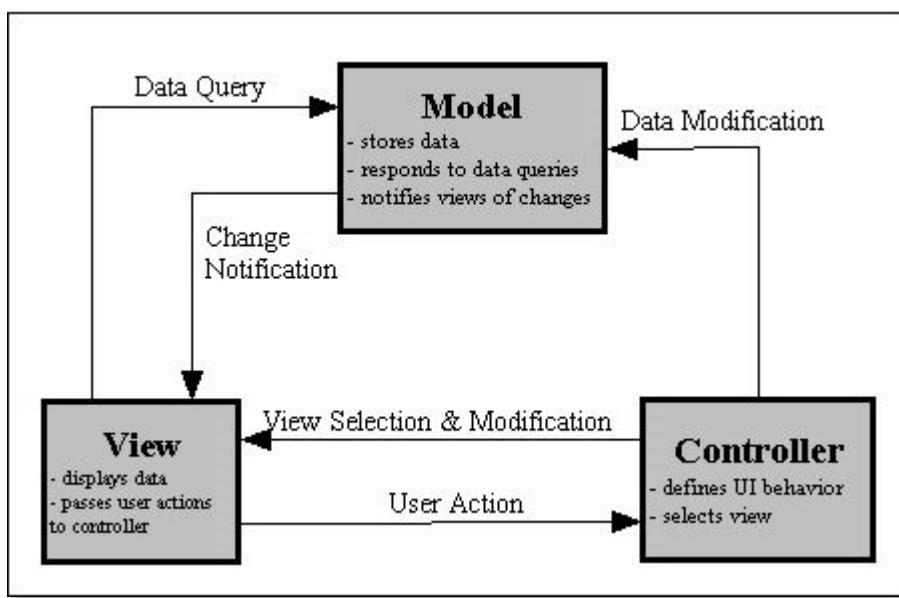
Klassens navn	Model		
KeyAttribute	theKey		
variabel type	System.String	værdi	et navn
variabel type	System.Int32	værdi	42
variabel type	Reflections.Klasse	værdi	Reflections.Klasse
variabel type	System.Int64	værdi	6000

Projektet som Pixiebog

Her er en, der ikke har fulgt med tiden. Don't be him!



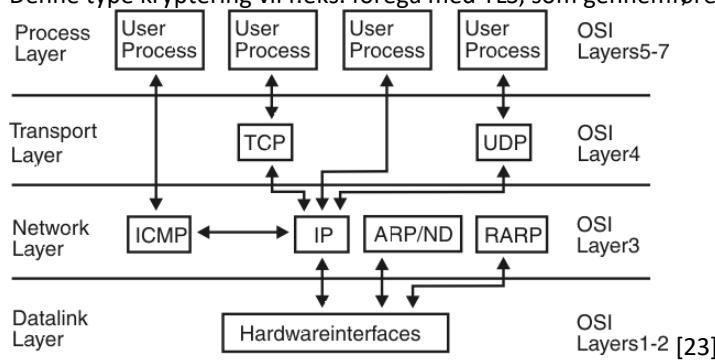
Herunder er det skitseret, hvordan sammenhængen mellem de tre dele af MVC kan opfattes.



[18]

"A controller is the means by which the user interacts with the application." (ootips.org [34])

Denne type kryptering vil f.eks. foregå med TLS, som gennemføres i transportlaget



VPN og krypterer hele pakken, der skal sendes, dvs. header og payload. Den arbejder på linklaget og gør, at

Et opslag på dr.dk uden VPN viste Wireshark sådan med forskellige protokoller og IP-adresser:

No.	Time	Source	Destination	Protocol	Length	Info
781	4.617037	193.162.153.164	192.168.0.11	DNS	388	Standard query response 0x4a55 A e4578.g.akamaiedge.net A 2.16.19.169 NS n0g.akamaiedge.net
782	4.621526	192.168.0.11	195.137.195.8	HTTP	771	GET /download/Forside/2016/Koncerthuset/Blaamarts/940/index.aframe HTTP/1.1
783	4.636167	195.137.195.8	192.168.0.11	HTTP	1348	HTTP/1.1 200 OK (text/html)
784	4.653749	192.168.0.11	159.20.6.22	TCP	54	53546 → 80 [ACK] Seq=15628 Ack=58896 Win=42240 Len=0
785	4.666666	192.168.0.11	193.162.153.164	DNS	82	Standard query 0xa2dc AAAA e4578.g.akamaiedge.net
786	4.666963	192.168.0.11	194.239.134.83	DNS	73	Standard query 0x49f0 A pcastol.dr.dk
787	4.677451	193.162.153.164	192.168.0.11	DNS	143	Standard query response 0xa2dc AAAA e4578.g.akamaiedge.net SOA n0g.akamaiedge.net
788	4.683688	194.239.134.83	192.168.0.11	DNS	219	Standard query response 0x49f0 A pcastol.dr.dk A 195.137.195.8 NS dns101.telia.com
789	4.686763	192.168.0.11	195.137.195.8	TCP	54	53568 → 80 [ACK] Seq=718 Ack=1295 Win=15104 Len=0
790	4.695128	192.168.0.11	104.77.230.232	TCP	66	53574 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
791	4.695389	192.168.0.11	195.137.195.8	TCP	66	53575 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
792	4.706036	104.77.230.232	192.168.0.11	TCP	66	80 → 53574 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=32
793	4.706185	192.168.0.11	104.77.230.232	TCP	54	53574 → 80 [ACK] Seq=1 Ack=1 Win=16384 Len=0
794	4.715333	195.137.195.8	192.168.0.11	TCP	66	80 → 53575 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1

```
> Frame 783: 1348 bytes on wire (10784 bits), 1348 bytes captured (10784 bits) on interface 0
> Ethernet II, Src: Netgear_6a:fe:22 (30:46:9a:6a:fe:22), Dst: WistronN_28:f1:00 (90:a4:de:28:f1:00)
└ Internet Protocol Version 4, Src: 195.137.195.8, Dst: 192.168.0.11
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1334
  Identification: 0x699c (27036)
  Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
  Time to live: 122
  Protocol: TCP (6)
  Header checksum: 0x4ae0 [validation disabled]
  [Header checksum status: Unverified]
  Source: 195.137.195.8
  Destination: 192.168.0.11
  0000  90 a4 de 28 f1 00 30 46 9a 6a fe 22 08 00 45 00  ...(..OF .j."..E.
  0010  05 36 69 9c 40 00 7a 06 4a e0 c3 89 c3 08 c0 a8  ..61.@.z. J.....
  0020  00 0b 00 50 d1 40 d4 ef bc ad dc 22 bf 23 50 18  ...P@... ...".#P.
  0030  01 00 00 77 00 00 48 54 54 50 2f 31 2e 31 20 32  ...w..HT TP/1.1 2
  0040  30 30 20 4f 4b 0d a0 43 6f 6e 74 65 6e 74 2d 54  00 OK..C ontent-T
  0050  79 70 65 3a 20 74 65 78 74 2f 68 74 6d 6c 0d 0a  type: tex t/html..
  0060  4c 61 73 74 2d 4d 6f 64 69 66 69 65 64 3a 20 54  Last-Mod ified: T
  0070  75 65 2c 20 30 37 20 4d 61 72 20 32 30 31 37 20 ue, 07 M ar 2017
  0080  30 39 3a 35 34 3a 30 32 20 47 4d 54 0d 0a 41 63  09:54:02 GMT..Ac
  0090  63 65 70 74 2d 52 61 6e 67 65 73 3a 20 62 79 74  cept-Ran ges: byt
  00a0  65 73 0d 0a 45 54 61 67 3a 20 22 30 31 39 64 34  es..ETag : "019d4
  00b0  62 66 32 38 39 37 64 3a 31 3a 30 22 0d 0a 53 65  bf2897d2 1:0"..Se
```

I denne session blev VPN anvendt. Der var et par ARP requests, men ellers var al trafikken mellem pc'en (192.168.0.11) og VPN-severen (83.170.84.216), og protokollerne ikke afslørede meget.

Apply a display filter ... <Ctrl-/>

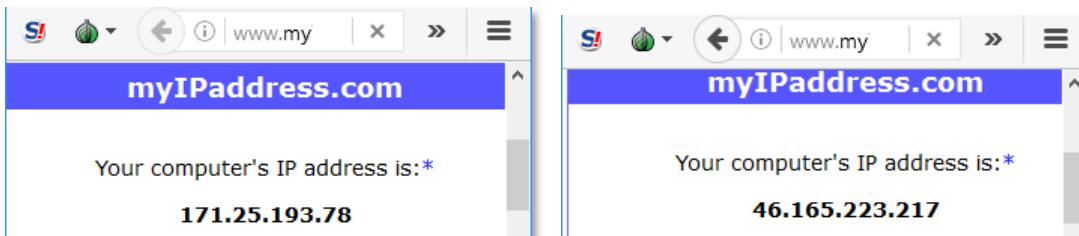
No.	Time	Source	Destination	Protocol	Length	Info
3793	150.787842	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3794	150.842789	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP
3795	150.940386	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP
3796	151.388933	83.170.84.216	192.168.0.11	PPP Co...	316	Compressed data
3797	151.393680	83.170.84.216	192.168.0.11	PPP Co...	316	Compressed data
3798	151.393907	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3799	151.471140	83.170.84.216	192.168.0.11	PPP Co...	318	Compressed data
3800	151.488390	192.168.0.11	83.170.84.216	GRE	46	Encapsulated PPP
3801	151.517513	83.170.84.216	192.168.0.11	PPP Co...	322	Compressed data
3802	151.517824	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3803	151.673372	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP
3804	151.917886	83.170.84.216	192.168.0.11	PPP Co...	316	Compressed data
3805	151.968034	192.168.0.11	83.170.84.216	PPP Co...	95	Compressed data
3806	152.121352	83.170.84.216	192.168.0.11	GRE	56	Encapsulated PPP

```
> Frame 3803: 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface 0
> Ethernet II, Src: Netgear_6a:fe:22 (30:46:9a:6a:fe:22), Dst: WistronN_28:f1:00 (90:a4:de:28:f1:00)
< Internet Protocol Version 4, Src: 83.170.84.216, Dst: 192.168.0.11
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x08 (DSCP: Unknown, ECN: Not-ECT)
        Total Identification: 32
        Identification: 0xe440 (58432)
    > Flags: 0x02 (Don't Fragment)
        Fragment offset: 0
        Time to live: 50
        Protocol: Generic Routing Encapsulation (47)
        Header checksum: 0xfb30 [validation disabled]
            [Header checksum status: Unverified]
        Source: 83.170.84.216
        Destination: 192.168.0.11
0000  90 a4 de 28 f1 00 30 46  9a 6a fe 22 08 00 45 08  ...(..0F .j."..E.
0010  00 20 e4 40 40 00 32 2f  fb 30 53 aa 54 d8 c0 a8  ..@.2/ .05.T...
0020  00 0b 20 81 88 0b 00 00  c9 e3 00 00 08 a8 00 00  .. ..... .
0030  00 00 00 00 00 00 00 00  ..... .

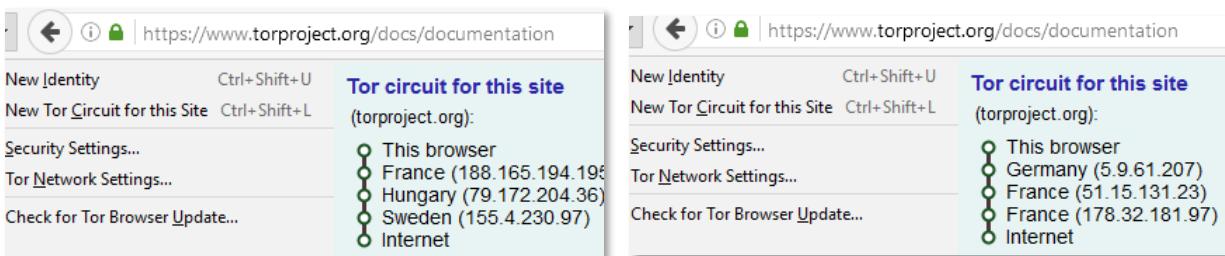
```

Et circuit opbygges ved at sende en Create-kommando til første node. Herefter bygges én nøgle, der skal anvendes til kryptering på første delstrækning. Efterfølgende sendes en Extend-kommado til sidste node i det partielt byggede circuit. Denne node vil så sende en Create til næste node.

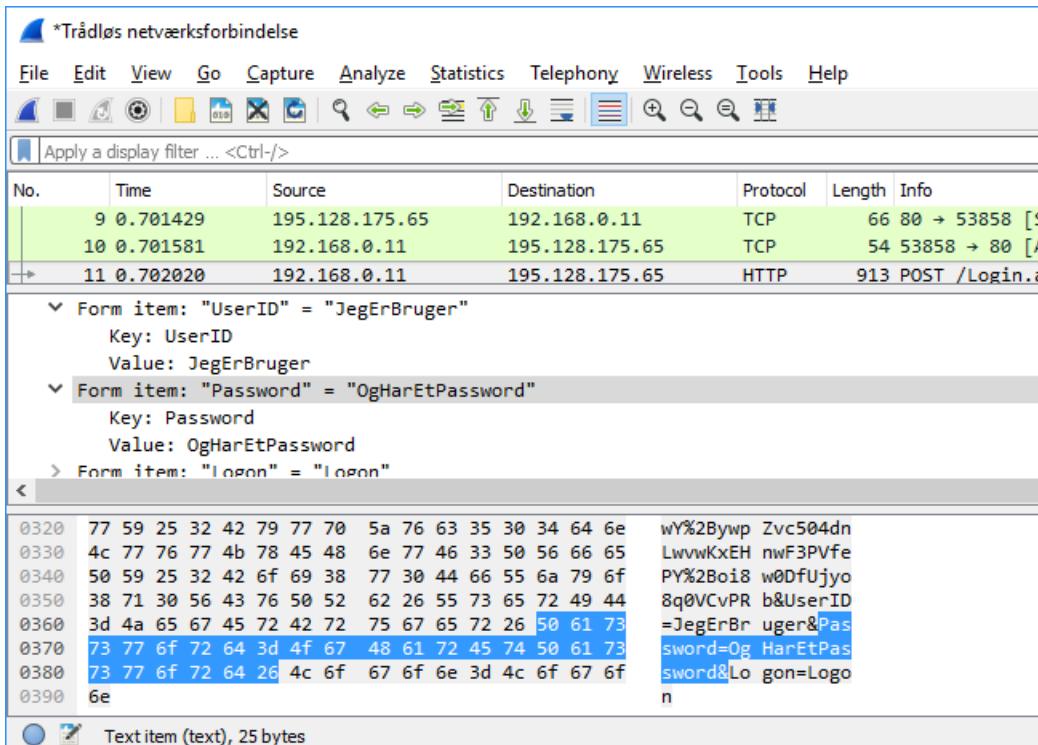
For hvert HTTP-request bygges et nyt circuit, hvilket hæmmer mulighederne for at følge trafikken. Det eneste der er sket mellem disse billede, er en refresh (Ctrl-R)



På samme måde er her to circuits for at ramme Torprojects dokumentation to gange med refresh.



I password input text feltet i på websiden vises det indtastede password som stjerner eller cirkler, men de opræder eller i klartekst. I nedenstående billede har Wireshark opsnappet en logon-dialog over en ikke-krypteret forbindelse.



Logon		
brugerId	lastLogon	logonCnt

User							
brugerId	createTs	password	salt	email	prim	erAdmin	

Arrangement					
arrangementId	arrangementBeskrivelse	createTs	arrangementWeb	arrangementDt	arrangementTid

spiseSted	spiseTid
-----------	----------

Tilmeldinger				
arrangementId	brugerId	createTs	erTilmeldt	spiser

Billeder				
billedAdresse	arrangementId	brugerId	createTs	kommentar

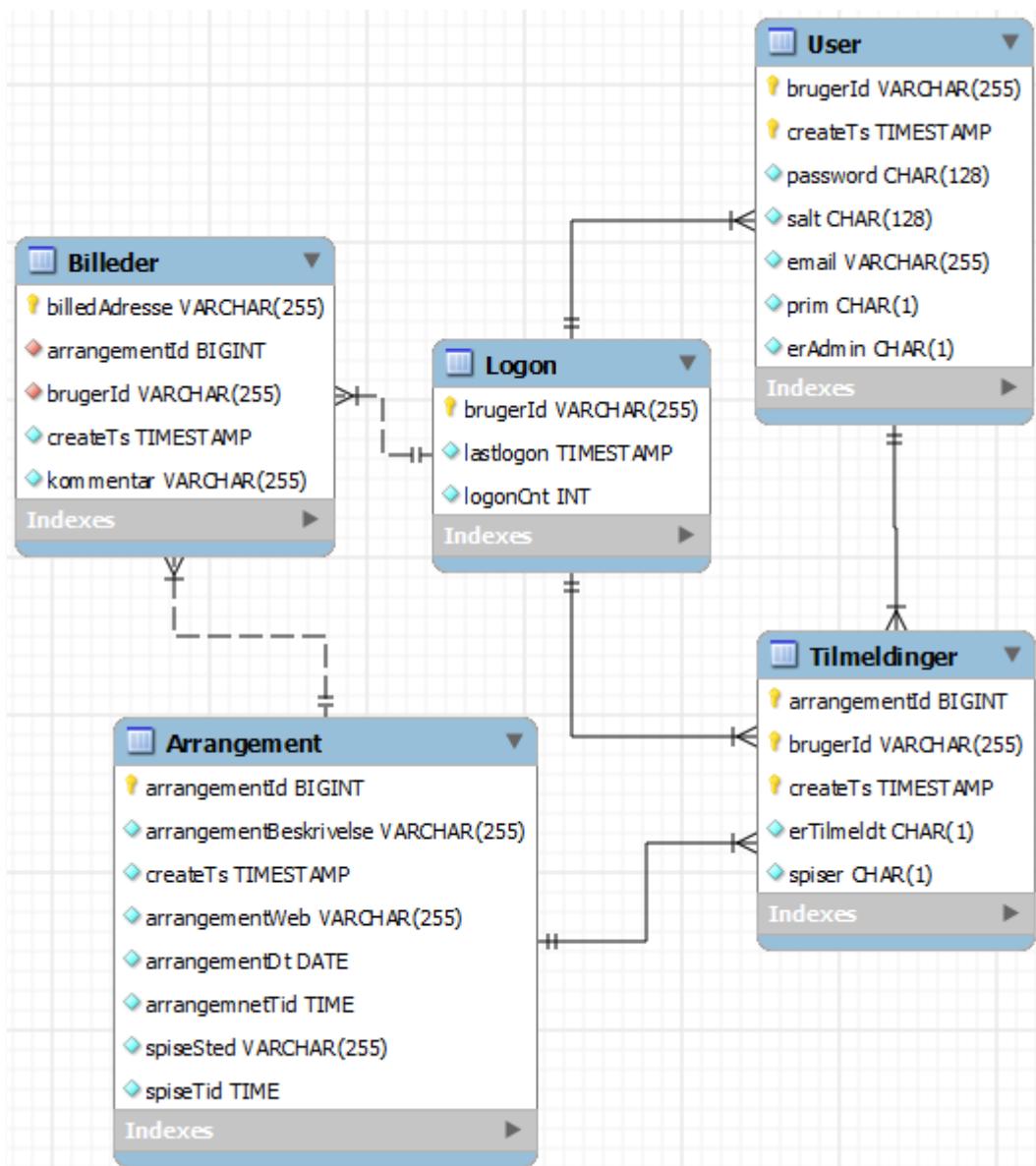
UserPassword			
brugerId	createTs	password	salt

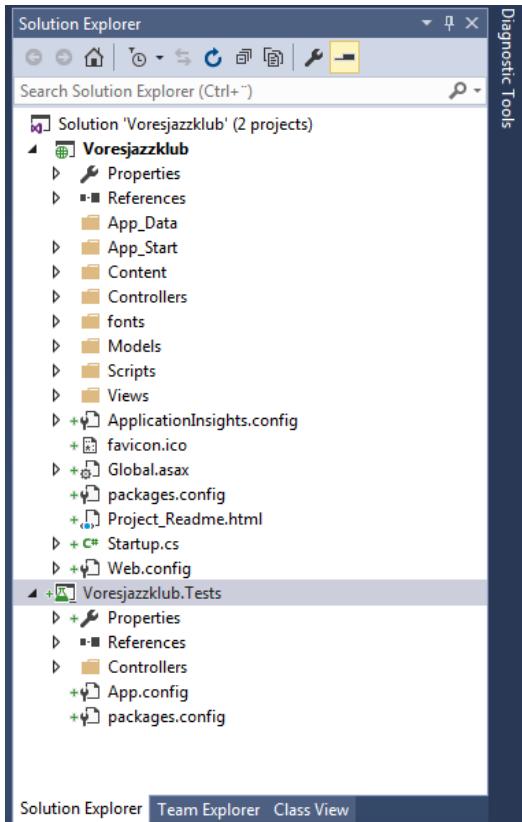
Email		
brugerId	createTs	email

Admin		
brugerId	createTs	erAdmin

Af praktiske grunde accepteres redundans, og normalisering ud over 3NF (og evt. Boyce Codd) vil ikke blive gennemført.

Det resulterende EER-diagram importeret i MySQL-Workbench.





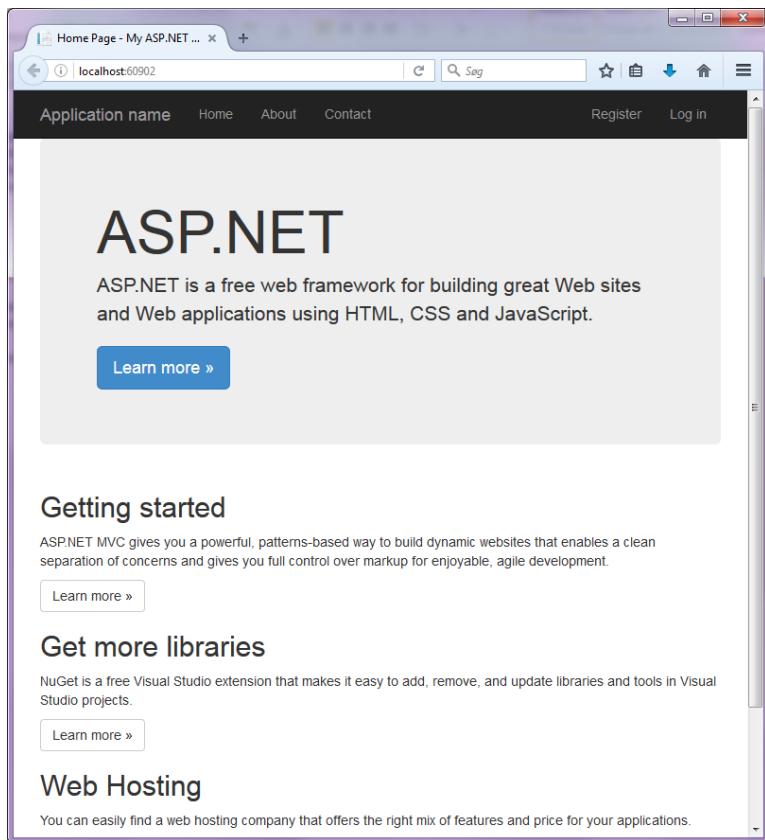
Der er små 600 filer og over 800 mapper i det automatisk genererede projekt.

Selenium skulle selvfølgeligt prøves, men der kom flere udfordringer ud af det. Google kunne ikke umiddelbart komme med nogen brugbar forklaring, så tilbage er kun at grave sig ned i problemet, hvilket formentlig er ganske tidskrævende.

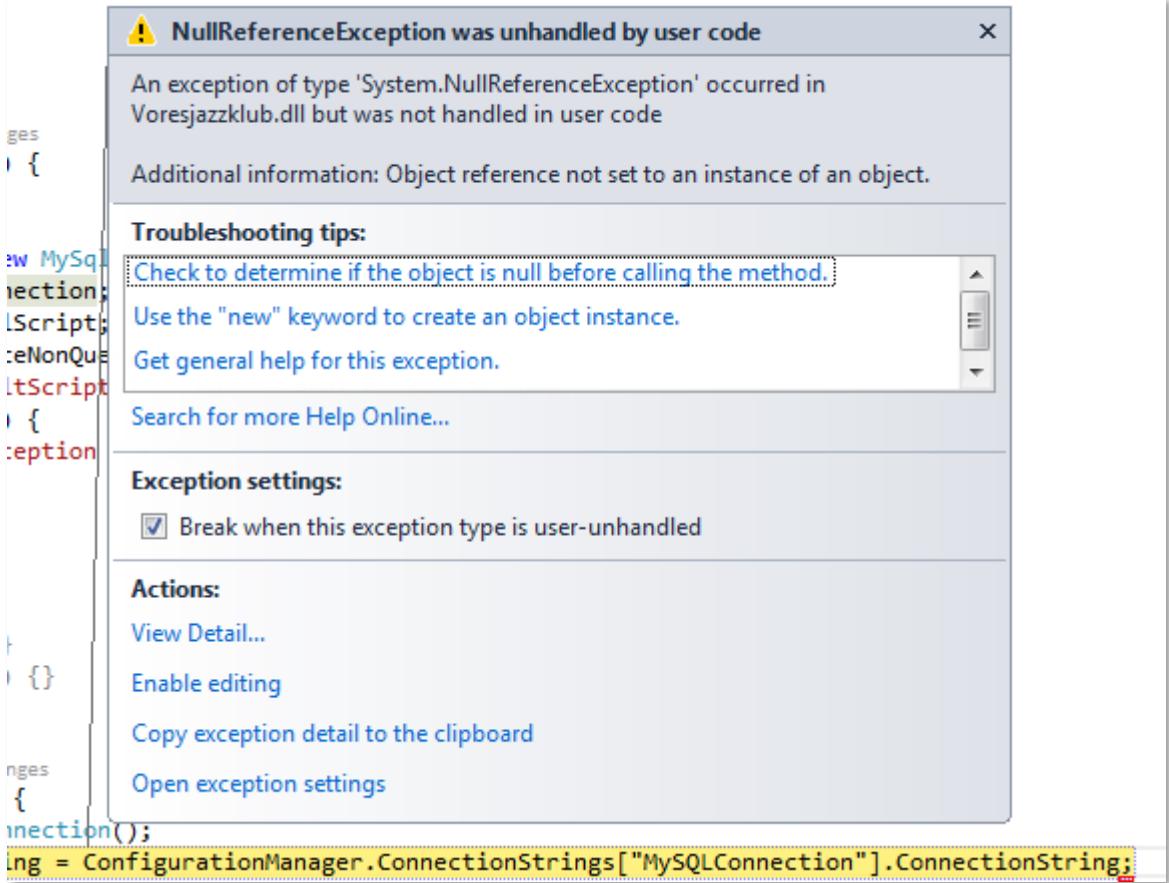
```
[TestMethod]
[TestCategory("Selenium")]
public void TestMethod1()
{
    DesiredCapabilities capability = DesiredCapabilities.Firefox();
    Uri server = new Uri("http://localhost:58578/Home/Index");
    this.driver = new RemoteWebDriver(server, capability);
    driver.Manage().Window.Maximize();/*
    driver.Navigate().GoToUrl(this.baseURL);
    driver.FindElementById("search - box").Clear();
    driver.FindElementById("search - box").SendKeys("tire");
    */
}
```

⚠ InvalidCastException was unhandled by user code
An exception of type 'System.InvalidCastException' occurred in WebDriver.dll but was not handled in user code
Additional information: Et objekt af typen 'System.Collections.Generic.Dictionary`2[System.String, System.Object]' var ikke konverteret til et objekt af typen 'System.String'.
System.InvalidCastException:
An exception of type 'System.InvalidCastException' occurred in WebDriver.dll but was not handled in user code
Additional information: Et objekt af typen 'System.String' k...
Troubleshooting tips:

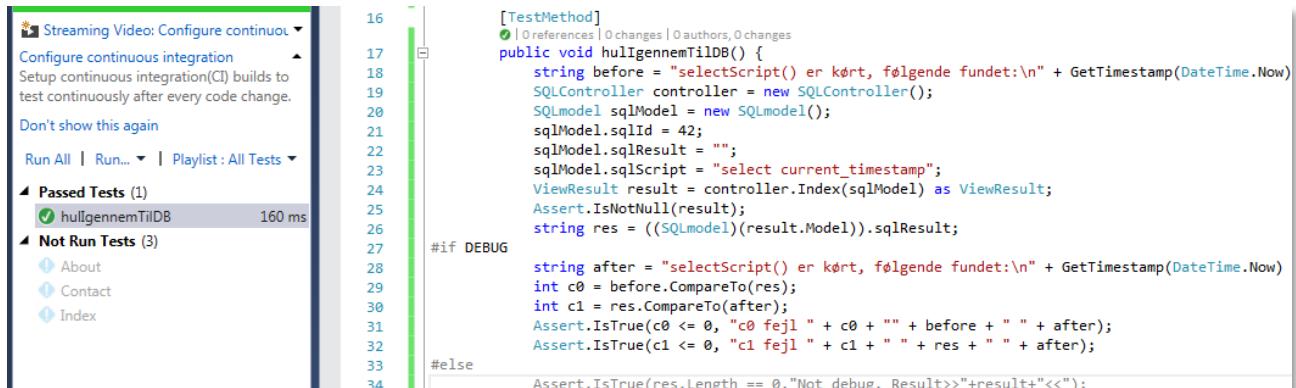
Den kørende web-applikation right out of the box.



Den første test gik helt galt da testprojektet ikke kunne finde MySQLConnection.



I Testprojektets App.config indsættes connection strings fra hovedprojektets Web.config, og så kører den første test på et igennem



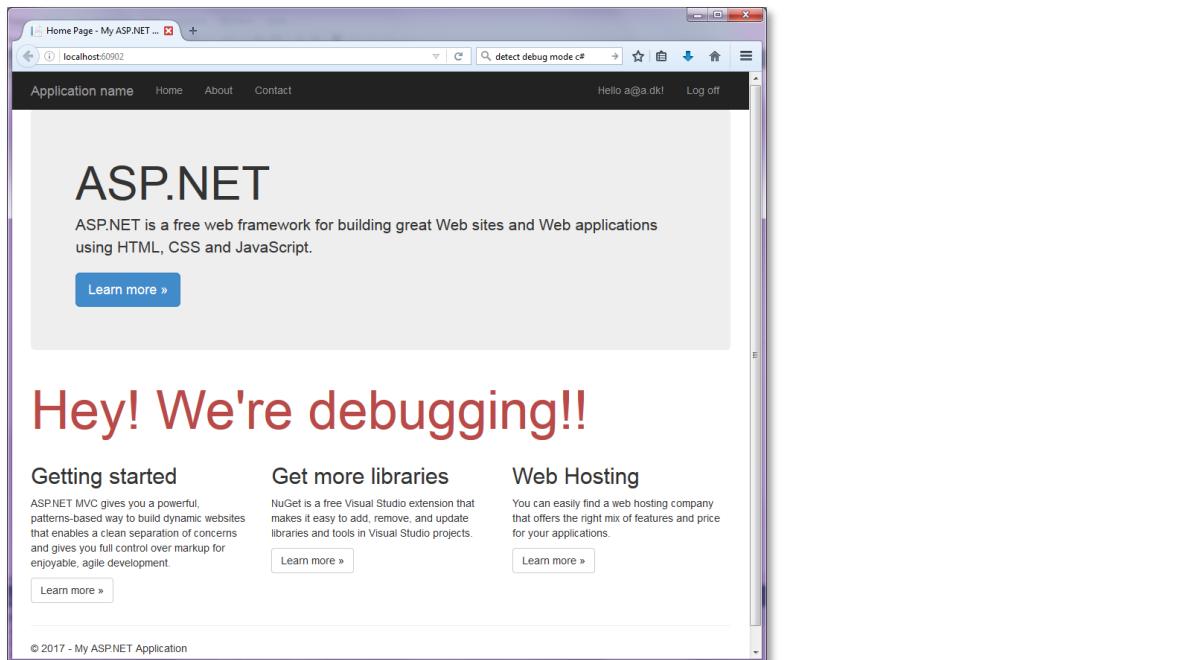
```
public class HomeController : Controller {
    public ActionResult Index() {
        #if DEBUG
            ViewBag.isDebugEnabled = "Hey! We're debugging!!";
        #endif
        string title = ViewBag.Title;
        return View();
    }
}
```

DEBUG er en variable, som VS sætter ud fra om den arbejder i Debug eller Release modus. Den kan så anvendes af C#'s precompiler til at inkludere eller ekskludere kode.

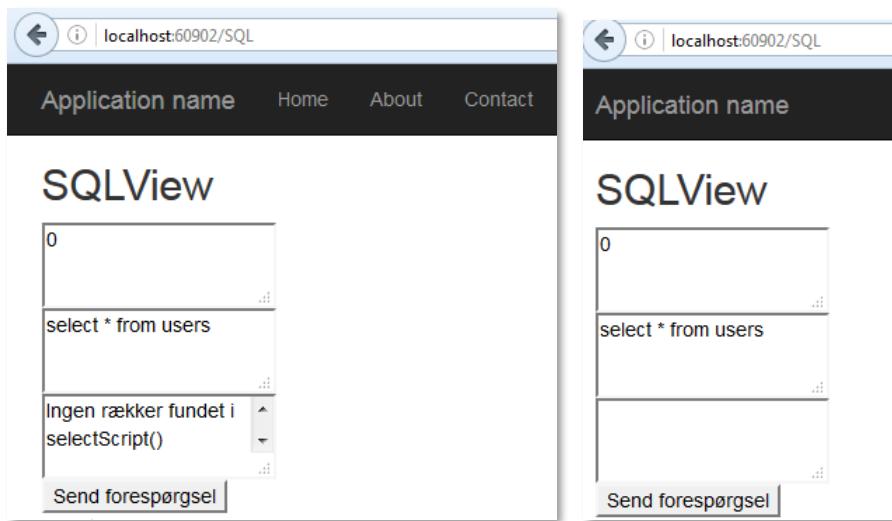
Her sættes en tekststreng i ViewBag, som er en ustruktureret samling objekter, der kan sendes til View'et. Teksten vises i rød og med en stor skrifftype

```
@{ if(ViewBag.isDebugging != null) {  
    <a href = "~/Views/Home/Index.cshtml" > ~/ Views / Home / Index.cshtml </ a >  
    <w class="text-danger large-text">@ViewBag.isDebugging</w>  
}  
}
```

.large-text {
 font-size: 72px;



Ved at være i Debug-modus, kan man køre SQL-scripts. Det kan man ikke i release-modus



```
41 }  
42     0 references | 0 changes | 0 authors, 0 changes  
43     public void nonResultScript() {  
44         try {  
45             getConnection();  
46             MySqlCommand cmd = new MySqlCommand();  
47             cmd.Connection = connection;  
48             cmd.CommandText = sqlScript;  
49             int rows = cmd.ExecuteNonQuery();  
50             sqlResult = "nonResultScript() er kørt. Antal rows: " + rows;  
51         } catch(MySqlException e) {  
52             sqlResult = "MySqlException " + e;  
53         } finally {  
54             connection.Close();  
55         }  
56     }  
57     public void selectScript() {}  
58     public void nonResultScript() {}  
59 #endif  
60 }
```

I øverste venstre hjørne står "Debug" og i kode testes på om DEBUG er sat. Derfor kan SQL-scriptet køres

Tilsvarende er VS i Release-modus og derfor anvendes to absolut tomme metoder, hvorfor der intet sker.

```
41 }  
42     -references | 0 changes | 0 authors, 0 changes  
43     public void nonResultScript() {  
44         try {  
45             getConnection();  
46             MySqlCommand cmd = new MySqlCommand();  
47             cmd.Connection = connection;  
48             cmd.CommandText = sqlScript;  
49             int rows = cmd.ExecuteNonQuery();  
50             sqlResult = "nonResultScript() er kørt. Antal rows: " + rows;  
51         } catch(MySqlException e) {  
52             sqlResult = "MySqlException " + e;  
53         } finally {  
54             connection.Close();  
55         }  
56     }  
57     public void selectScript() {}  
58     public void nonResultScript() {}  
59 #endif
```

Disse klasser kommer der ud af at hente AspNet.Identity.MySQL:

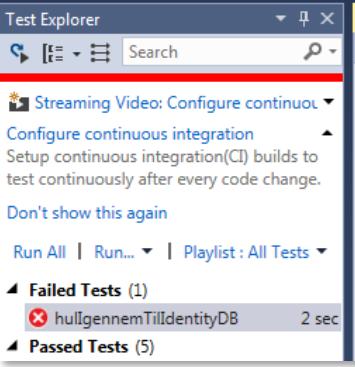
Biblioteket Dokumenter

AspNet.Identity.MySQL

Sortér efter: Mappe ▾

Navn	Ændringsdato	Type	Størrelse
.vs	15-04-2017 07:25	Filmappe	
bin	15-04-2017 07:25	Filmappe	
obj	15-04-2017 07:25	Filmappe	
packages	15-04-2017 07:26	Filmappe	
Properties	15-04-2017 07:24	Filmappe	
AspNet.Identity.MySQL.sln	15-04-2017 07:24	Microsoft Visual Studio Solution	1 KB
MySQLIdentity.sql	15-04-2017 07:24	SQL Text File	2 KB
Readme.txt	15-04-2017 07:24	Tekstdokument	2 KB
AspNet.Identity.MySQL.csproj	15-04-2017 07:24	Visual C# Project file	5 KB
IdentityRole.cs	15-04-2017 07:24	Visual C# Source file	2 KB
IdentityUser.cs	15-04-2017 07:24	Visual C# Source file	3 KB
MySQLDatabase.cs	15-04-2017 07:24	Visual C# Source file	8 KB
RoleStore.cs	15-04-2017 07:24	Visual C# Source file	3 KB
RoleTable.cs	15-04-2017 07:24	Visual C# Source file	5 KB
UserClaimsTable.cs	15-04-2017 07:24	Visual C# Source file	4 KB
UserLoginsTable.cs	15-04-2017 07:24	Visual C# Source file	5 KB
UserRoleTable.cs	15-04-2017 07:24	Visual C# Source file	3 KB
UserStore.cs	15-04-2017 07:24	Visual C# Source file	22 KB
UserTable.cs	15-04-2017 07:24	Visual C# Source file	12 KB
App.config	15-04-2017 07:24	XML Configuration File	1 KB
packages.config	15-04-2017 07:24	XML Configuration File	1 KB

Når der bliver lavet noget nyt, skal der også være kørt en test. Første gang den køres, forventes det, at den fejler, og når der er kodet tilpas meget, og testen køres igen, skal den blive grøn. Det er naturligvis også tilfælde her, hvor det tjekkes om der er forbindelse til den nye database.

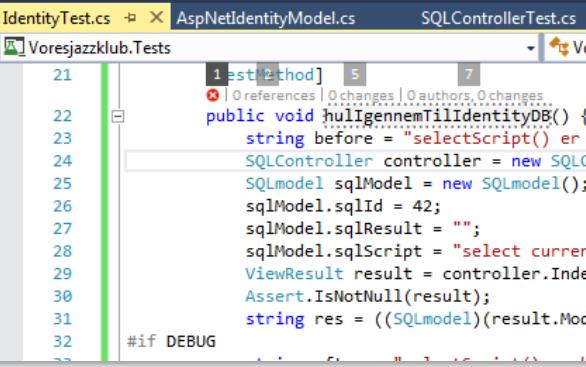


The screenshot shows the Visual Studio Test Explorer window. Under 'Failed Tests (1)', there is one test named 'hulgennemTilIdentityDB' which failed with a duration of 2 seconds. The code for this test is visible in the editor:

```

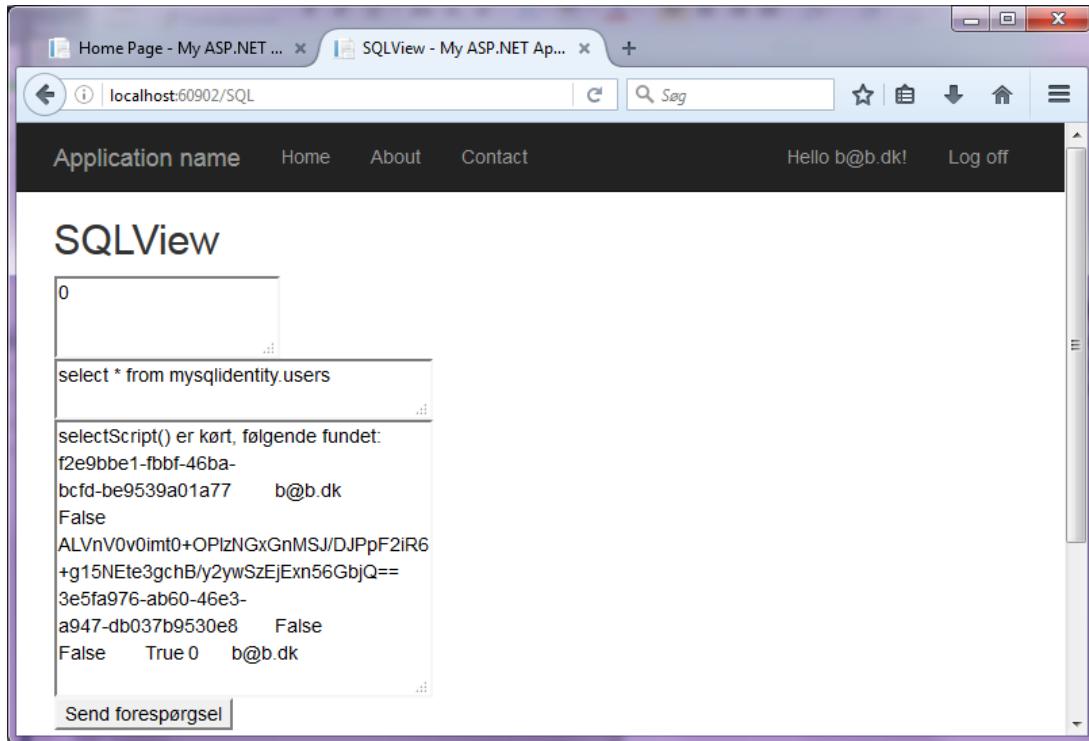
[TestMethod]
public void hulgennemTilIdentityDB() {
    string before = "selectScript() er ";
    SQLController controller = new SQLController();
    SQLModel sqlModel = new SQLmodel();
    sqlModel.sqlId = 42;
    sqlModel.sqlResult = "";
    sqlModel.sqlScript = "select currentViewResult result = controller.IdentityDB();
    Assert.IsNotNull(result);
    string res = ((SQLmodel)(result.Model));
    Assert.AreEqual(before, res);
}
#endif DEBUG

```

The screenshot shows the Visual Studio Test Explorer window again. This time, under 'Passed Tests (6)', there are six tests, including 'hulgennemTilIdentityDB' which passed with a duration of 174 ms. The code for this test is identical to the failed version above.

Ved at gå ind på siden SQL, hvor et vilkårligt SQL-script kan køres fra, hvilket muligvis kan opfattes som en sikkerhedstrussel, vælges alle registrerede brugere. Her genfindes så den nyligt registrerede bruger.



Downside ved at anvende Microsofts identitetsframework er, at det kompromitterer den valgte datamodel. Det betyder så også, at applikationens funktionalitet bliver lidt anderledes en oprindeligt tænkt.

Der er fremstillet to metoder til brug, hvor brugerens password skal håndteres. Det er ved oprettelse, ændring og tjek af validitet. Den ene får et password i klartekst ind og afleverer hash og salt. Som det ses, er hashen genereret ud fra en konstant værdi, en tilfældig værdi og password. Disse tre værdier er tekststrenge, der konateneres.

```
public void getHashedPasswordAndSalt(string password, out string salt, out string hash) {
    salt = getSalt();
    hash = getHashSha512(getConstantSalt() + salt + password);
```

Den anden metode tager password i klartekst og det salt, der er persistenter i databasen, og returnere samme værdi som i metoden ovenfor.

```
public string getHashedPassword(string password, string salt) {
    return getHashSha512(getConstantSalt() + salt + password);
```

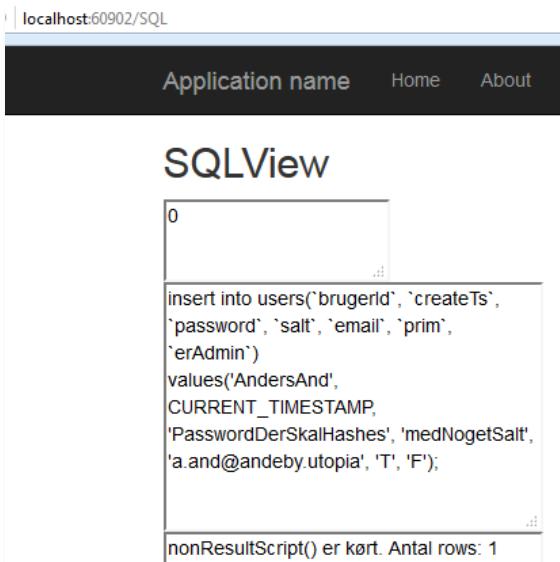
Selve hashingen står .Net for i klassen SHA512Managed i metoden ComputeHash. Teksten konverteres til bytes, og outputtet konverteres til en hexadecimal streng.

```
string getHashSha512(string stringToBeHashed) {
    StringBuilder hashedString = new StringBuilder();
    foreach(byte hashedByte in new SHA512Managed().ComputeHash(Encoding.Unicode.GetBytes(stringToBeHashed)))
        hashedString.Append(String.Format("{0:x2}", hashedByte));
}
return hashedString.ToString();
```

```
string getSalt() {
    StringBuilder salt = new StringBuilder();
    byte[] bytes = new byte[64];
    new RNGCryptoServiceProvider("").GetBytes(bytes);
    foreach(byte b in bytes) {
        salt.Append(String.Format("{0:x2}", b));
    }
    return salt.ToString();
```

Den sidste metode er til det konstante salt. Det bliver gemt i maskinens filsystem. Fordelen er, at salt kan beskyttes, så det kun er en administrator, der har adgang til det. Der er ingen garantier mod tyveri, men det er i hvert fald gjort besværligt.

```
public string getConstantSalt() { // anvendes i test, derfor public
    string constantSaltFilename = "...\\App_Data\\ConstantSalt";
    try {
        return System.IO.File.ReadAllText(constantSaltFilename);
    } // så må vi have det ikke er fordi nogen har slettet den og
    // ellers må du hellere finde din backup frem
    catch (FileNotFoundException) {
        string salt = getSalt();
        System.IO.File.WriteAllText(constantSaltFilename, salt);
    }
    return salt;
}
```



```
insert into users(`brugerId`, `createTs`, `password`, `salt`, `email`, `prim`, `erAdmin`)  
values('AndersAnd', CURRENT_TIMESTAMP, 'PasswordDerSkalHashes', 'medNogetSalt',  
'a.and@andeby.utopia', 'T', 'F');
```

Man starter med testmetoden. IDE'et vil ikke kunne kompilere testmetoden, da der hverken er en klasse eller metode IDE'et kan finde referencer til. Så laves det minimum, der skal til at kunne kompilere, hvorefter testen køres.

```

Process: [8408] iisexpress.exe
Lifecycle Events Thread: Stack Frame
Test Explorer Search
Streaming Video: Configure c
Configure continuous integration
Setup continuous integration(CI) builds to test continuously after every code change.
Don't show this again
Run All | Run... | Playlist: All
Failed Tests (1)
TestHentBruger 130 ms
Passed Tests (4)
About 12 ms
Contact < 1 ms
hullgennemTilDB 401 ms
Index 1 ms

```

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using System;
7  using Microsoft.VisualStudio.TestTools.UnitTesting;
8  using Voresjazzklub.Models;
9
10 namespace Voresjazzklub.Tests.Models {
11     [TestClass]
12     public class ModelsTest {
13         [TestMethod]
14         public void TestHentBruger() {
15             string brugerId = "AndersAnd";
16             UsersTableModel usersTableModel = new UsersTableModel();
17             usersTableModel.hentBruger(brugerId);
18             Assert.AreEqual(brugerId, usersTableModel.brugerId);
19         }
20     }

```

```

Passed Tests (5)
TestHentBruger 44 ms
About 12 ms
Contact < 1 ms
hullgennemTilDB 401 ms
Index 1 ms

```

```

10  namespace Voresjazzklub.Tests.Models {
11      [TestClass]
12      public class ModelsTest {
13          [TestMethod]
14          public void TestHentBruger() {
15              string brugerId = "AndersAnd";
16

```

De næste trin er at tilføre og ændre koden, i den testede metode, indtil testen bliver grøn. Alle test, til de berørte komponenter, skal også køres igen, for at se, om man har fået ødelagt noget.

Her er en af de underrutiner, der anvendes som et af skridtene i test af alle CRUD-operationerne. Først tømmes tabellen og det tjekkes om den er tom. Herefter oprettes fire rækker i tabellen, og det tjekkes om der så er fire elementer i listen, der hentes i databasen.

```

private void createFourEntries() {
    emptyTable();
    isUserListEmpty();

    rip.createUser();
    rap.createUser();
    rup.createUser();
    andersAnd.createUser();

    UsersTableModel usersTableModel = new UsersTableModel();
    List<UsersTableModel> utms = usersTableModel.getUserList();
    Assert.IsTrue(utms.Count == 4, "Der skulle være 4 personer i Users. Der er " + utms.Count);
}

```

Her en af de testmetoder, der gennemløber alle de mulige CRUD-operationer. Først oprettes fire rækker i tabellen med ovenstående metode.

```

[TestMethod]
public void TestUsersCRUD() {
    createFourEntries();
    readEntry();
    readList();
    updateEmail();
    deleteEntry();
}

```

Statisk test bliver udført af IDE'et. Her er en metode, der ikke altid returnerer noget. Det tæller som fejl.

```

private int manglerReturnValue() {
    int a=12;
    if(a==42) {
        return a
    }
}

private int manglerReturnValue() {
}

private int manglerReturnValue() {
    int a;
    if(a==42) {
        [!] (local variable) int a
    }
    reti
}

```

'LogonModelsController.manglerReturnValue()': not all code paths return a value
'LogonModelsController.manglerReturnValue()': not all code paths return a value
[!] Use of unassigned local variable 'a'

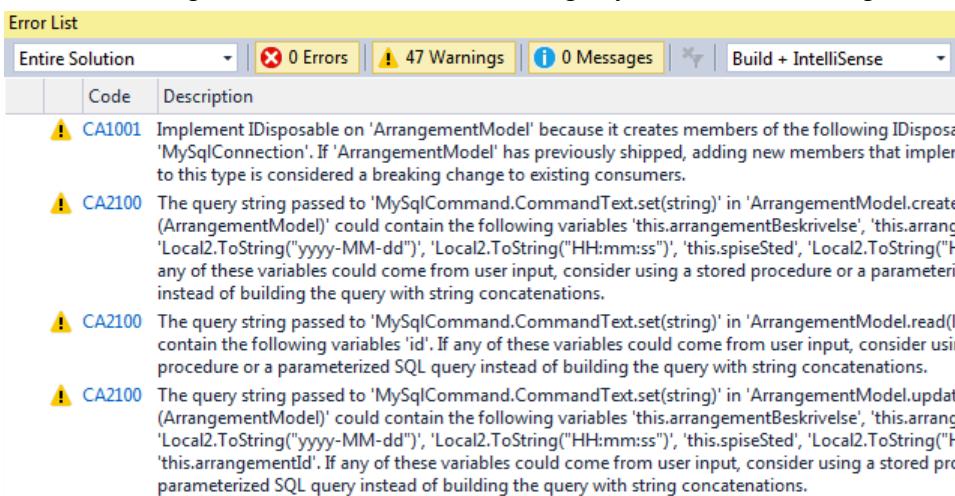
Nu returneres der rent faktisk noget, men variablen, der opereres på, har ikke fået tildelt en værdi

```

0 references | 0 changes | 0 authors, 0 changes
private int ingenReferencer() {
    return 42;
}

```

Dette er så endelig en metode, der ikke får røde bølgelinjer, men så ikke har nogle referencer.



Der er en mere dybdegående statisk analyse, som giver en længere række warnings. I en produktionsversion af produktet er hverken fejl eller advarsler acceptable. Der er to muligheder: Enten at rette i koden ud fra beskrivelsen, eller at konfigurerer analyseværktøjet til ikke give bestemte advarsler. Det sidste skal naturligvis kun ske, hvis der kan argumenteres forsvarligt for det.

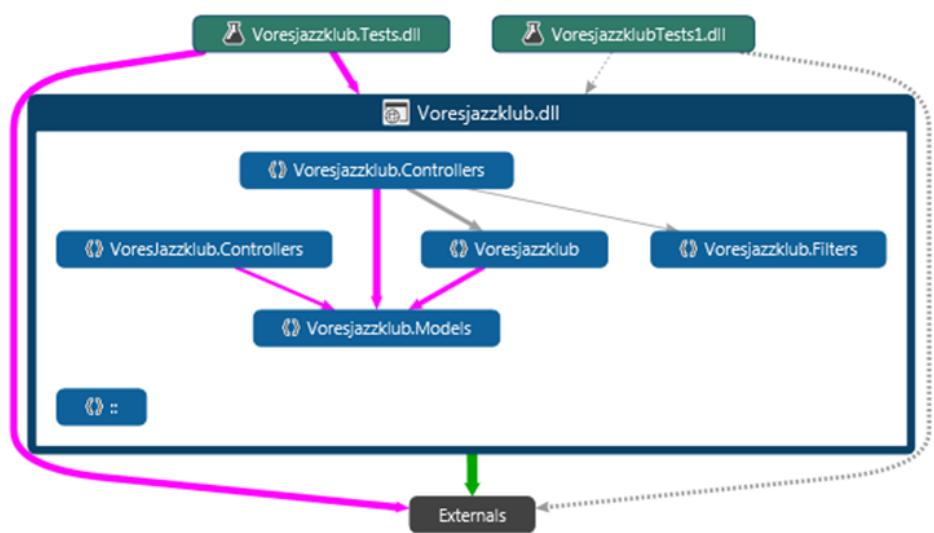
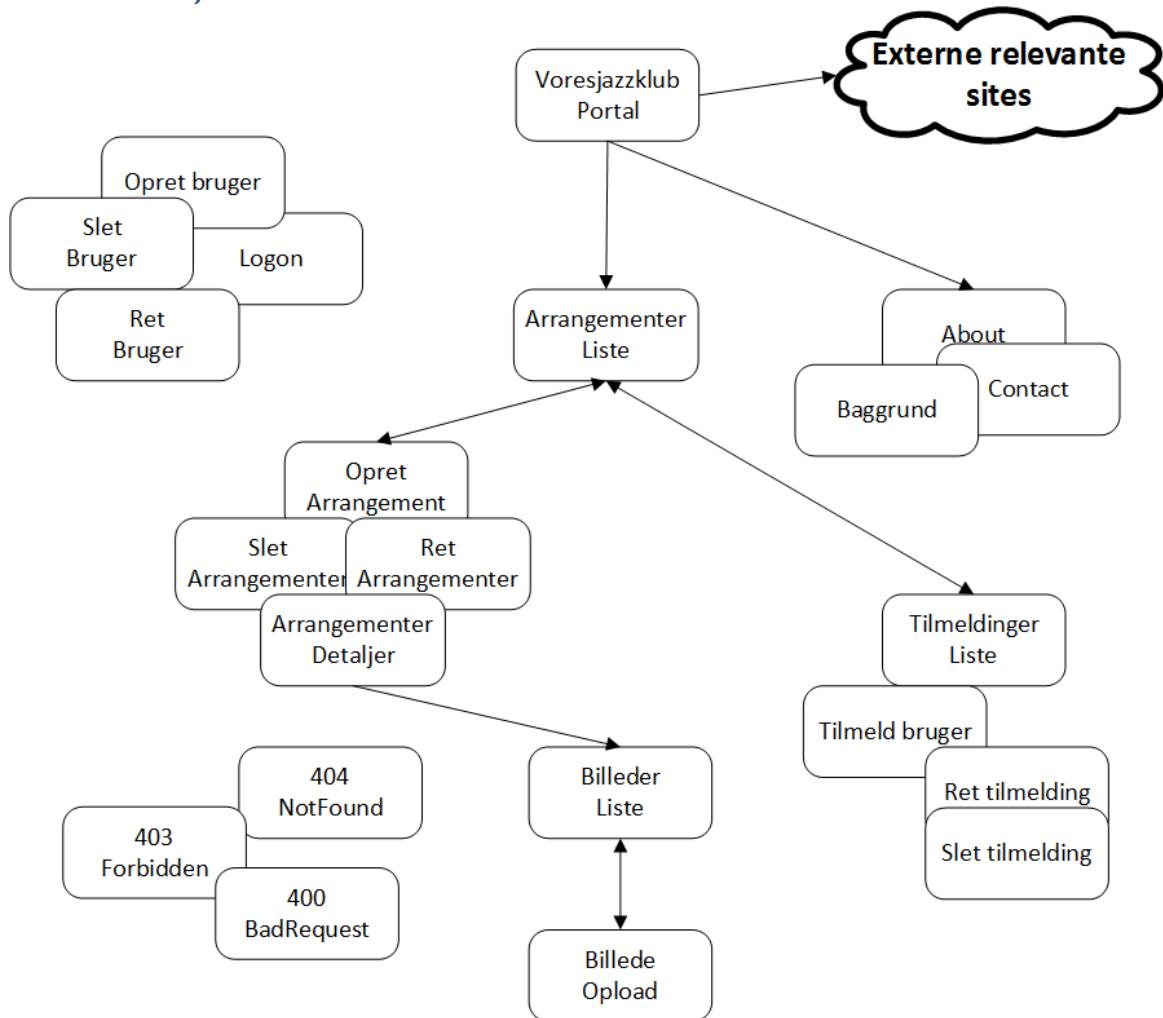
Deployment gik tilsyneladende godt, men applikationen kunne ikke køre på test.voresjazzklub.dk, kun lokalt på udviklingsmaskinen. Det skal undersøges, hvorfor der er forskellige opfattelser.

Output

Show output from: Build

```
2>Publishing folder Views/Manage...
2>Publishing folder Views/Shared...
2>Web App was published successfully ftp://test.voresjazzklub.dk/
2>
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
|
```

Siderne i Voresjazzklub



Walk through - a day at Voresjazzklub

Når man vælger Vores Jazzklub i menuen, kan logge ind eller oprette en ny bruger

Login

Kom indenfor

Bruger

password

[Back to List](#) | [Ny bruger](#)

© 2017 - Voresjazzklub

Efter vellykket login, kan vælges mellem Koncerter, brugs statistik, brugere og logout, ved at klikke på et af billederne. Der er også nogle muligheder i menuen. Forneden på siden kan ses, at det er Andersine, der er logget ind.

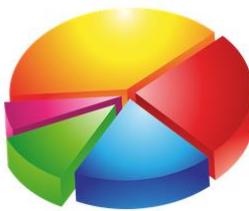
View - Voresjazzklub

localhost:60902/Jazzhome/Index

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

View

Vores Jazzklub



Velkommen Andersine - Sidst set 08.05.2017 09:45

[Brugs statistik](#) | [Koncerter](#) | [brugere](#) | [Farvel og tak for denne gang](#)

© 2017 - Voresjazzklub

Musen flyttes hen over lagkagediagram, for at vælge "Brugs statistik". Så skifter den til gråtone.

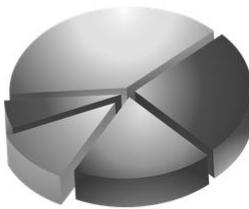
View - Voresjazzklub

localhost:60902/Jazzhome/Index

Jazz portalen Vores Jazzklub Baggrund CPI jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

View

Vores Jazzklub



Velkommen Andersine - Sidst set 08.05.2017 09:45

[Brugs statistik](#) | [Koncerter](#) | [brugere](#) | [Farvel og tak for denne gang](#)

© 2017 - Voresjazzklub
localhost:60902/logonmodels

Her kan man se, at Andersine har logget på 63 gange.

Index - Voresjazzklub

localhost:60902/logonmodels

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index

en anden måde at oprette nye brugere på, hvis de ikke selv kan

Logins på Voresjazzklub

brugerId	lastlogon	logonCnt	
AnderAnd	01-05-2017 09:14:48	0	Edit Details Delete
Andersine	11-05-2017 07:10:46	63	Edit Details Delete
Claus	10-05-2017 07:40:15	62	Edit Details Delete
Julie	06-05-2017 18:07:05	0	Edit Details Delete
Ole	03-05-2017 10:54:33	1	Edit Details Delete
QQ	06-05-2017 15:30:38	0	Edit Details Delete
Rap	08-05-2017 13:10:08	3	Edit Details Delete
Rip	08-05-2017 12:10:03	6	Edit Details Delete
RipAnd	23-04-2017 12:18:31	0	Edit Details Delete
Rup	01-05-2017 09:14:48	0	Edit Details Delete

135 som Model.Sum 135 som imperativ sammentælling

© 2017 - Voresjazzklub

Edit og Details for Andersine

Edit - Voresjazzklub

localhost:60902/logonmodels/Edit/Andersine

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Edit

Andersine

lastlogon	11-05-2017 07:10:46
logonCnt	63

[Save](#)

[Back to List](#)

© 2017 - Voresjazzklub

Details - Voresjazzklub

localhost:60902/logonmodels/Details/Andersine

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Details

Andersine

lastlogon 11-05-2017 07:10:46
logonCnt 63

[Edit](#) | [Back to List](#)

© 2017 - Voresjazzklub

Andersine skal ikke slettes fra listen, men en anden bruger, der aldrig har været logget på, QQ udvælges til sletning og efter bekræftelse, er QQ væk fra listen.

Delete - Voresjazzklub

localhost:60902/logonmodels/Delete/QQ

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Delete

Vil du virkelig fjerne

QQ?

lastlogon 06-05-2017 15:30:38
logonCnt 0

[Delete](#) | [Back to List](#)

© 2017 - Voresjazzklub

Index - Voresjazzklub

localhost:60902/logonmodels/Index

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index

en anden måde at oprette nye brugere på, hvis de ikke selv kan

Logins på Voresjazzklub

brugerId	lastlogon	logonCnt	
AnderAnd	01-05-2017 09:14:48	0	Edit Details Delete
Andersine	11-05-2017 07:10:46	63	Edit Details Delete
Claus	10-05-2017 07:40:15	62	Edit Details Delete
Julie	06-05-2017 18:07:05	0	Edit Details Delete
Ole	03-05-2017 10:54:33	1	Edit Details Delete
Rap	08-05-2017 13:10:08	3	Edit Details Delete
Rip	08-05-2017 12:10:03	6	Edit Details Delete
RipAnd	23-04-2017 12:18:31	0	Edit Details Delete
Rup	01-05-2017 09:14:48	0	Edit Details Delete

135 som Model.Sum 135 som imperativ sammentælling

© 2017 - Voresjazzklub

Tilbage med Vores Jazzklub menupunktet. Denne gang er musen flyttet hen over "Brugere".

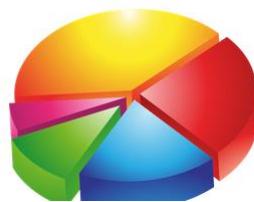
View - Voresjazzklub

localhost:60902/Jazzhome/Index

Jazz portalen Vores Jazzklub Baggrund CPH Jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

View

Vores Jazzklub


Index - Voresjazzklub

localhost:60902/userstablemodels

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index

Alle brugere

Opret ny bruger

Bruger	Oprettet dato og tid	password	Salt (sikring af password)	email	Primær email	Er administrator	
AndersAnd	08-05-2017 07:09:33	9bc641...	4b1db7...	andersand@andeby.utopia	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Edit Details Delete
Andersine	08-05-2017 07:06:12	897932...	3c6710...	andersine@gmail.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete
Claus	08-05-2017 07:10:15	d693ce...	5e6917...	claus@tetens.biz	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete
Rap	08-05-2017 07:10:58	aa5de8...	441d02...	Rap@rap.dk	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete
Rip	08-05-2017 07:10:32	ad9ff9...	4a9a3f...	rip@rip.dk	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Edit Details Delete
Rup	08-05-2017 07:11:16	4338bc...	dd1e01...	rup@rup.dk	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Edit Details Delete

© 2017 - Voresjazzklub

Details - Voresjazzklub

localhost:60902/Bruger/Andersine

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Details

Andersine

Oprettet dato og tid	08-05-2017 07:06:12
password	89793291e07e7e0224301f971dca4e43be0a1d6b498fd0f1579136a96749dfdf0bf28e1c7fd1ea6cef0898af7b6342c848002f6
Salt (sikring af password)	3c67102f2460ea82b9d8bd07c336e6e8030268ff141a02a20be6dd5d6c0446b7fead0751fca59ca48482c569df8208801cb407c
email	andersine@gmail.com
Primær email	<input checked="" type="checkbox"/>
Er administrator	<input checked="" type="checkbox"/>

[Edit](#) | [Back to List](#)

© 2017 - Voresjazzklub

Edit - Voresjazzklub

localhost:60902/UsersTableModels/Edit/Andersine

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Edit

Andersine

Oprettet dato og tid: 08-05-2017 07:06:12

password:

Salt (sikring af password): 3c67102f2460ea82b9d8bd07c336e6e8030268ff141a02a20be6dd5d6c0446b7fead0751fca59ca48482c569df8208801cb407c8d

email: andersine@gmail.com

Primær email:

Er administrator:

[Back to List](#)

© 2017 - Voresjazzklub

Delete - Voresjazzklub

localhost:60902/UsersTableModels/Delete/Andersine

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Delete

Er du sikker på, at du vil fjerne

Andersine?

Oprettet dato og tid: 08-05-2017 07:06:12

password: 89793291e07e7e0224301f971dca4e43be0a1d6b498fd0f1579136a96749dfdf0bf28e1c7fd1ea6ceef0898af7b6342c848002ff6

Salt (sikring af password): 3c67102f2460ea82b9d8bd07c336e6e8030268ff141a02a20be6dd5d6c0446b7fead0751fca59ca48482c569df8208801cb407c8d

email: andersine@gmail.com

Primær email:

Er administrator:

| [Back to List](#)

© 2017 - Voresjazzklub

Denne gang vælges menupunktet "Vores Jazzklub" frem for Delete-knappen, og musen placeres henover Koncerter

A screenshot of a web browser window titled "Index - Voresjazzklub". The URL in the address bar is "localhost:60902/arran/index". The page header is identical to the homepage. The main content is titled "Index" in bold black font, followed by "Koncerter & andet godt" in large green font. Below this is a link "Opret koncert | Forside". A table lists eight events with columns for Beskrivelse, Oprettet, Yderligere info, Dato, Tid, Hvor skal vi spise?, and Hvad tid. Each row includes edit and delete links. The events listed are:

Beskrivelse	Oprettet	Yderligere info	Dato	Tid	Hvor skal vi spise?	Hvad tid
Hej	29-04-2017 06:41:32	cph-jazz	2014-02-02	00:00	Leonoras	00:00
Hej	24-04-2017 09:58:23	Hej hej	2017-02-02	17:00	Pølsevognen	00:00
Nu med tid	24-04-2017 10:45:15	ask Google	2017-04-24	18:42	Her	18:30
Beskrivelse	24-04-2017 15:32:57	http://noget	2017-12-31	14:32	Kong Hans	17:37
Julekoncert	29-04-2017 06:30:51	dr.dk	2020-12-24	18:00	Forhallen	12:30
Hornmusik i Fælledparken	01-05-2017 06:50:32	Første may	2017-05-01	06:30	Samme sted	12:00
En koncert	02-05-2017 06:42:38	se på nettet	2017-10-01	17:56	Pølsevognen	17:00

© 2017 - Voresjazzklub

Edit

Hornmusik i Fælledparken

Beskrivelse	Hornmusik i Fælledparken
Oprettet	01-05-2017 06:50:32
Yderligere info	Første maj
Dato	2017-05-01
Tid	06:30
Hvor skal vi spise?	Samme sted
Hvad tid	12:00

[Save](#)

[Back to List](#)

© 2017 - Voresjazzklub

"Første may" er blevet ændret til "Første maj"

Details

Hornmusik i Fælledparken

Beskrivelse	Hornmusik i Fælledparken
Oprettet	11-05-2017 07:52:49
Yderligere Info	Første maj
Dato	2017-05-01
Tid	06:30
Hvor skal vi spise?	Samme sted
Hvad tid	12:00

[Tilmelding](#) | [Upload billeder](#) | [Vis billeder](#) | [Edit](#) | [Back to List](#)

© 2017 - Voresjazzklub

Der er én tilmeldt

Index - Voresjazzklub

localhost:60902/Tilmeldingers/Index/6

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index

Hornmusik i Fælledparken

Meld mig til Hornmusik i Fælledparken

arrangementid	brugerId	Oprettet	Tilmeldt	Spiser	
6	Claus	08-05-2017 08:43:56	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete

© 2017 - Voresjazzklub

og bliver Andersine også tilmeldt

Create - Voresjazzklub

localhost:60902/Tilmeldingers/Create/6

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Meld mig til

Hornmusik i Fælledparken

Tilmeldt

Spiser

[Create](#)

[Back to List](#)

© 2017 - Voresjazzklub

Index - Voresjazzklub

localhost:60902/Tilmeldingers/Index/6

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index

Hornmusik i Fælledparken

Meld mig til Hornmusik i Fælledparken

arrangementid	brugerId	Oprettet	Tilmeldt	Spiser	
6	Andersine	11-05-2017 07:57:37	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete
6	Claus	08-05-2017 08:43:56	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete

© 2017 - Voresjazzklub

localhost:60902/Tilmeldingers/Delete/6/Andersine

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Delete

Vil du melde

Andersine fra Hornmusik i Fælledparken?

arrangementid: 6
brugerkode: Andersine
Oprettet: 11-05-2017 07:57:37
Tilmeldt:
Spiser:

[Delete](#) | [Back to List](#)

© 2017 - Voresjazzklub

og nej ikke framelding

Upload billeder

Hornmusik i Fælledparken

Hornmusik i Fælledparken

Kommentar: Dette smukke billede h

Billede: Gennemse... Ingen fil valgt.

Upload billede og tekst

Overfør fil

Biblioteket Billeder

Sortér efter: Mappe

Favoritter

Biblioteker

Computer

Efnavn: jazzbilledede.png



Index - Voresjazzklub

localhost:60902/Upload/Index/6

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Upload billeder

Hornmusik i Fælledparken



/img/Upload/jazzbilleder.png
Hornmusik i Fælledparken
Andersine

Kommentar

Billede Ingen fil valgt

© 2017 - Voresjazzklub

Index - Voresjazzklub

localhost:60902/BillederArrangs/Index/6

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index for Billede & Arrangement

Billeder fra Hornmusik i Fælledparken

Create New

billedAdresse	arrangementId	brugerId	Oprettet	Kommentar	Arrangement	
/img/Upload/Julekoncert2000_NBENNERINSTRUMENTS_1.gif	6	Andersine	01-05-2017 06:54:58	Her er en kommentar til Julekonerten	Hornmusik i Fælledparken	Edit Details Delete
	6	Rap	08-05-2017 11:59:50	en jazzmusiker	Hornmusik i Fælledparken	Edit Details Delete
	6	Rap	08-05-2017 13:31:00	En jazzmusiker	Hornmusik i Fælledparken	Edit Details Delete
	6	Andersine	11-05-2017 08:01:02	Dette smukke billede har jeg taget fra koncerthen	Hornmusik i Fælledparken	Edit Details Delete

© 2017 - Voresjazzklub

Musen hen over et af billederne for at få det i en større version

Index - Voresjazzklub

localhost:60902/BillederArrangs/Index/6

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index for Billeder & Arrangement

Billeder fra Hornmusik i Fælledparken

Create New

billedAdresse	arrangementId	brugerId	Oprettet	kommentar	Arrang
/img/Upload/Julekoncert2000_NBENNERINSTRUMENTS_1.gif	6	Andersine	01-05-2017 06:54:58	Her er en kommentar til Julekonerten	Hornmu Fælledp
	6	Rap	08-05-2017 11:59:50	en jazzmusiker	Hornmu Fælledp

Edit

Hornmusik i Fælledparken



billedAdresse /img/Upload/jazzbillede.png

arrangementId 6

brugerId Rap

Oprettet 08-05-2017 11:59:50

Kommentar Iu med ny kommentar til "en jazzmusiker!"

Arrangement Hornmusik i Fælledparken

[Back to List](#)

© 2017 - Voresjazzklub

Details - Voresjazzklub

localhost:50902/BillederArrangs/Details/6/18

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Details

Hornmusik i Fælledparken



arrangementid 6
brugerid Rap
Oprettet 11-05-2017 08:21:22
Kommentar Nu med ny kommentar til "en jazzmusiker"
Arrangement Hornmusik i Fælledparken

[Edit](#) | [Back to List](#)

© 2017 - Voresjazzklub

Delete - Voresjazzklub

localhost:60902/BillederArrangs/Delete/6/18

Søg

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Delete

Vil du slette

Hornmusik i Fælledparken?



arrangementid: 6
brugerId: Rap
Oprettet: 11-05-2017 08:21:22
kommentar: Nu med ny kommentar til "en jazzmusiker"
Arrangement: Hornmusik i Fælledparken

[Delete](#) | [Back to List](#)

© 2017 - Voresjazzklub

Index - Voresjazzklub

localhost:60902/BillederArrangs/Index/6

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index for Billede & Arrangement

Billeder fra Hornmusik i Fælledparken

Create New

billedAdresse	arrangementId	brugerId	Oprettet	kommentar	Arrangement	
/img/Upload/Julekoncert2000_NBENNERINSTRUMENTS_1.gif	6	Andersine	01-05-2017 06:54:58	Her er en kommentar til Julekonerten	Hornmusik i Fælledparken	Edit Details Delete
	6	Rap	08-05-2017 13:31:00	En jazzmusiker	Hornmusik i Fælledparken	Edit Details Delete
	6	Andersine	11-05-2017 08:01:02	Dette smukke billede har jeg taget fra koncerthen	Hornmusik i Fælledparken	Edit Details Delete

© 2017 - Voresjazzklub

Der skal også oprettes et nyt arrangement via Create New linket

Create - Voresjazzklub

localhost:60902/arran/Create

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Create

Forventninger om en god koncert

Beskrivelse Beginning og Ibrahim Electric til Studie 2

Yderligere info <http://jazz.dk/copenhagen-jazz-festival-2>

Dato 2017-07-08

Tid 20:00

Hvor skal vi spise? Sporvejene

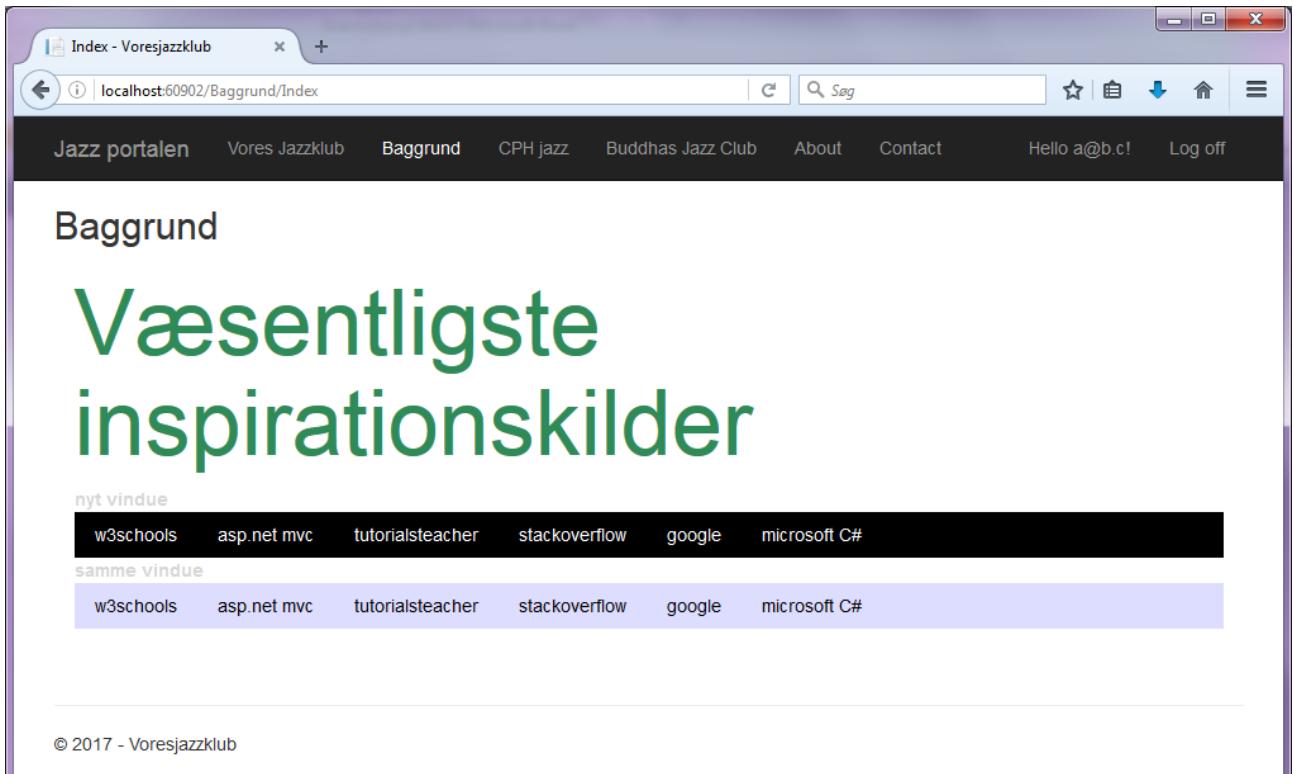
Hvad tid 17:00

[Back to List](#)

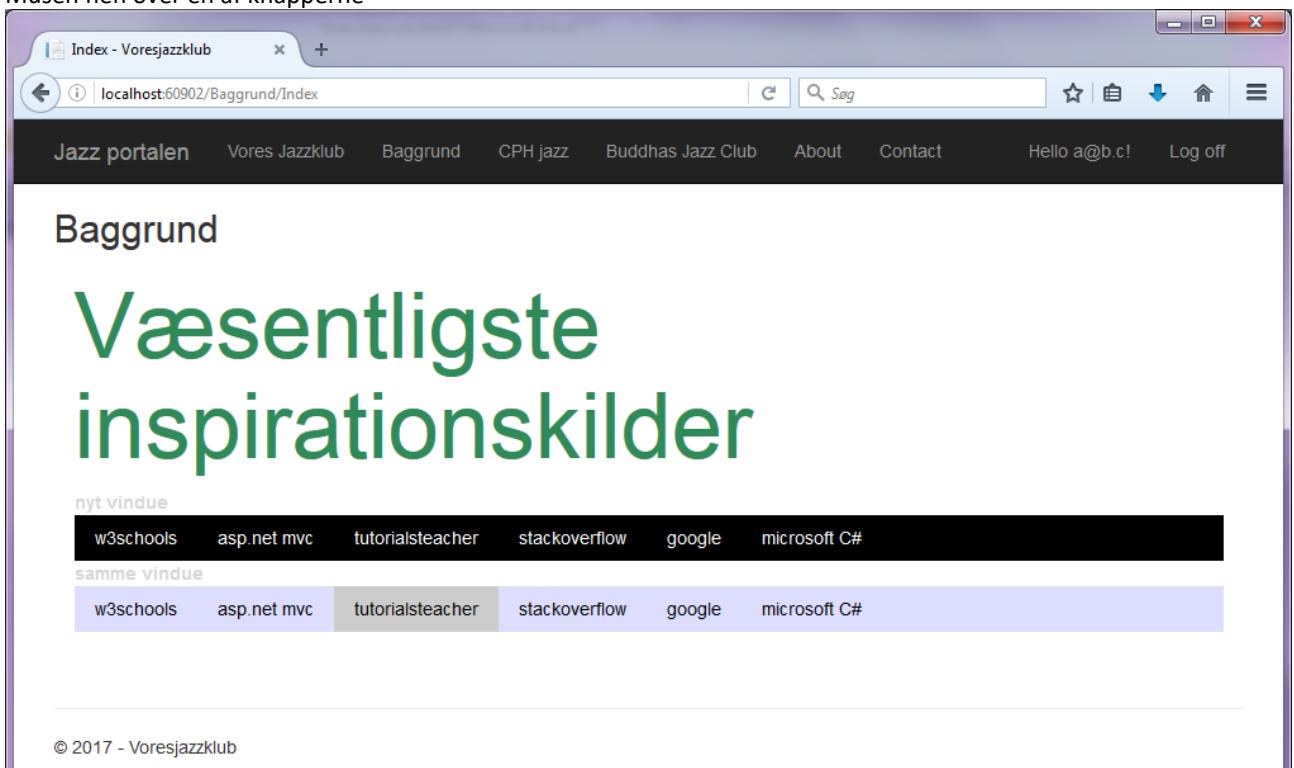
© 2017 - Voresjazzklub

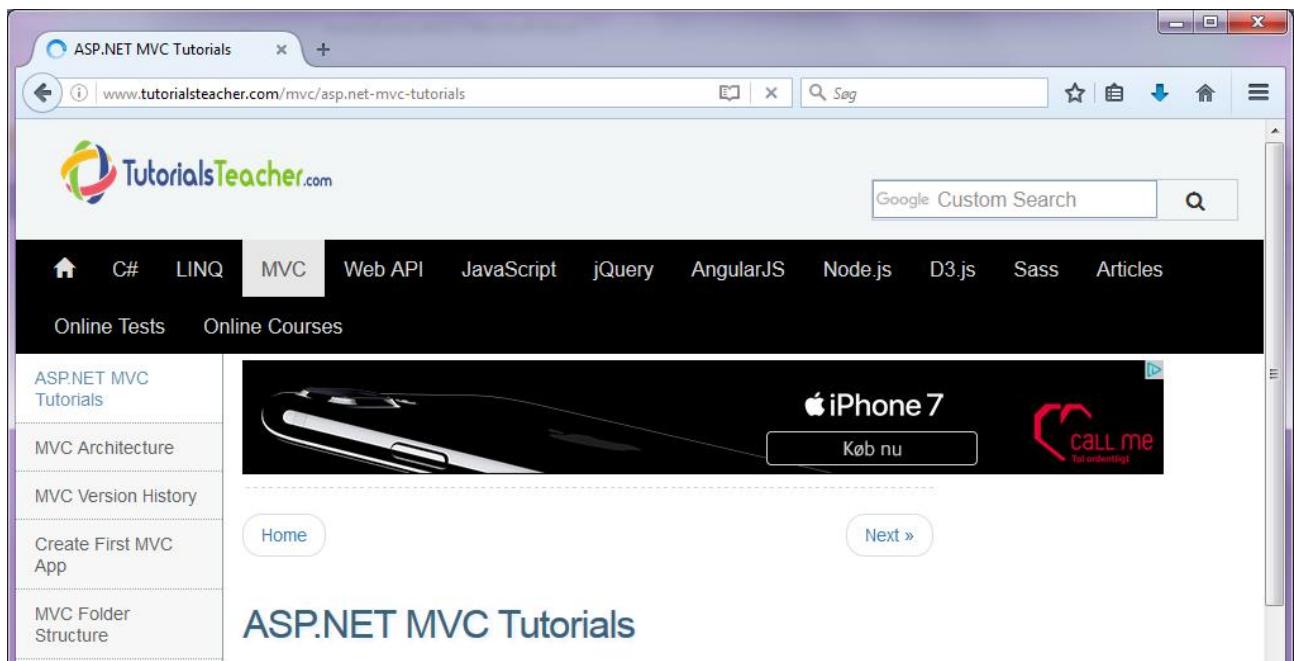
Beskrivelse	Oprettet	Yderligere info	Dato	Tid	Hvor skal vi spise?	Hvad tid
Hej	29-04-2017 06:41:32	cph-jazz	2014-02-02	00:00	Leonoras	00:00
Hej	24-04-2017 09:58:23	Hej hej	2017-02-02	17:00	Pølsevognen	00:00
Nu med tid	24-04-2017 10:45:15	ask Google	2017-04-24	18:42	Her	18:30
Beskrivelse	24-04-2017 15:32:57	http://noget	2017-12-31	14:32	Kong Hans	17:37
Julekoncert	29-04-2017 06:30:51	dr.dk	2020-12-24	18:00	Forhallen	12:30
Hornmusik i Fælledparken	11-05-2017 07:52:49	Første maj	2017-05-01	06:30	Samme sted	12:00
En koncert	02-05-2017 06:42:38	se på nettet	2017-10-01	17:56	Pølsevognen	17:00
Kira Skov & Maria Faust In The Beginning og Ibrahim Electric til Studie 2	11-05-2017 08:26:33	http://jazz.dk/openhagen-jazz-festival- 2017/nyheder/kira-skov-maria-faust- in-the-beginning-og-ibrahim-electric- til-studie-2/	2017-07-08	20:00	Sporvejene	17:00

Lidt info om, hvorfra man kan få viden om microsofts frameworks, css3 og html5

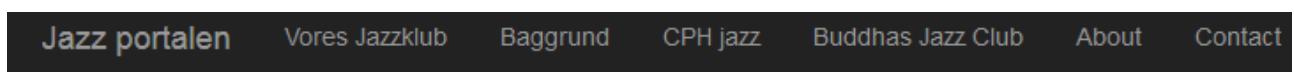


Musen hen over en af knapperne





Den øverste række knapper er forsynet med `<abbr title="...><input type="button" ...></abbr>`, hvor det, der står i title vises som hint, når musen holdes hen over knappen. Ydermere åbner knappen en ny fane i stedet for at udskifte nuværende fanes indhold.



Baggrund

Væsentligste inspirationskilder

nyt vindue

w3schools asp.net mvc tutorialsteacher stackoverflow google microsoft C#

samme vindue

www.tutorialsteacher.com/mvc/asp.net-mvc-tutorials

w3schools asp.net mvc tutorialsteacher stackoverflow google microsoft C#

To færre

The screenshot shows a web browser window with the following details:

- Title Bar:** Index - Voresjazzklub (Active tab) and ASP.NET MVC Tutorials.
- Address Bar:** www.tutorialsteacher.com/mvc/asp.net-mvc-tutorials
- Header:** TutorialsTeacher.com logo, Google Custom Search bar, and a navigation menu with links: Home, C#, LINQ, **MVC**, Web API, JavaScript, jQuery, AngularJS, Node.js, D3.js, Sass, Articles, Online Tests, and Online Courses.
- Left Sidebar:** A vertical sidebar with links: ASP.NET MVC Tutorials (selected), MVC Architecture, MVC Version History, Create First MVC App, and MVC Folder Structure.
- Main Content Area:** A large banner for Microsoft Azure: "Sikker og stabil e-handelsløsning med Azure". Below it is a "Home" button and a "Next »" button.
- Right Sidebar:** An advertisement for MuleSoft: "Integrating .NET ap Learn best practices" with a "Free Whitepaper" button.

CPH jazz og Buddhas Jazz Club er eksterne links. About-knappen giver lidt info om produkterne, der er anvendt i tilblivelsen af web-app'en

About - Voresjazzklub

localhost:60902/Home/About

Jazz portalen Vores Jazzklub Baggrund CPI jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

About.

IT's all about jazz

Mette Juul (voc/g) Ambrose Akinmusire (tp) Nikolaj Hess (p) Morten Ramsbøl (b) Morten Lund (dr) i Huset lørdag 17. sep. 2011



[Baggrund](#)

Anvendte produkter

- Første maskine
 - Microsoft Windows 10 Home version 10.0.14393 build 14393
 - Microsoft Visual Studio Community 2015 Version 14.0.25431.01 Update 3
 - Microsoft .NET Framework Version 4.6.01586
 - Mozilla Firefox 52.0
- Anden maskine
 - Microsoft Windows 7 Enterprise Version 6.1.7601 Service Pack 1 Build 7601
 - Microsoft Visual Studio Enterprise 2015, Version 14.0.25431.01 Update 3
 - Microsoft .NET Framework Version 4.6.01590
 - Mozilla Firefox 45.9.0
- Fælles
 - XAMPP v5.6.30
 - MySQL-Workbench 6.3
 - Wireshark 2.2.4

© 2017 - Voresjazzklub

Sidste knap på forsiden er logoff

The screenshot shows a Windows-style browser window titled "View - Voresjazzklub". The address bar shows "localhost:60902/Jazzhome/Index". The page header includes links for "Jazz portalen", "Vores Jazzklub", "Baggrund", "CPH jazz", "Buddhas Jazz Club", "About", "Contact", "Hello a@b.c!", and "Log off". The main content area features the title "View" and "Vores Jazzklub" in large green font. Below the title is a photograph of four musicians playing brass instruments. To the left is a 3D pie chart, and to the right are three user icons. A "Colourbox" watermark is visible over a central image of an open door.

Velkommen Andersine - Sidst set 11.05.2017 07:47

Brugs statistik | Koncerter | brugere | Farvel og tak for denne gang

© 2017 - Voresjazzklub
localhost:60902/arran/afslut

der sender brugeren til login billedet

The screenshot shows a Windows-style browser window titled "Login - Voresjazzklub". The address bar shows "localhost:60902/UsersTableModels/login". The page header includes links for "Jazz portalen", "Vores Jazzklub", "Baggrund", "CPH jazz", "Buddhas Jazz Club", "About", "Contact", "Hello a@b.c!", and "Log off". The main content area features the heading "Login" and "Kom indenfor" in large green font. Below this are two input fields labeled "Bruger" and "password", and a "Let me in, please" button. At the bottom are links for "Back to List | Ny bruger" and copyright information.

Bruger

password

Let me in, please

Back to List | Ny bruger

© 2017 - Voresjazzklub

Her vælges så at oprette en ny bruger. Der kan ikke være to brugere med samme navn, så Andersine kan ikke oprettes igen

Create - Voresjazzklub

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Create

En ny bruger

• Brugeren eksisterer allerede - prøv at hedde noget andet

Her kan man se fejlene

• Brugeren eksisterer allerede - prøv at hedde noget andet

Bruger

Oprettet dato og tid

password

Salt (sikring af password)

email

Primær email

Er administrator

[Back to List](#)

© 2017 - Voresjazzklub

hvorimod det går fint med PeterPlys

Create - Voresjazzklub

localhost:60902/UsersTableModels/Create

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Create

En ny bruger

Her kan man se fejlene

Bruger	PeterPlys
Oprettet dato og tid	
password	****
Salt (sikring af password)	
email	peter.plys@hundredemeterskoven.uk
Primær email	<input checked="" type="checkbox"/>
Er administrator	<input type="checkbox"/>
Create	

[Back to List](#)

© 2017 - Voresjazzklub

Index - Voresjazzklub

localhost:60902/userstablemodels

Jazz portalen Vores Jazzklub Baggrund CPH jazz Buddhas Jazz Club About Contact Hello a@b.c! Log off

Index

Alle brugere

Opret ny bruger

Bruger	Oprettet dato og tid	password	Salt (sikring af password)	email	Primær email	Er administrator	
AndersAnd	08-05-2017 07:09:33	9bc641...	4b1db7...	andersand@andeby.utopia	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Edit Details Delete
Andersine	08-05-2017 07:06:12	897932...	3c6710...	andersine@gmail.com	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete
Claus	08-05-2017 07:10:15	d693ce...	5e6917...	claus@tetens.biz	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Edit Details Delete
PeterPlys	11-05-2017 08:49:26	24822c...	ac9f24...	peter.plys@hundredemeterskoven.uk	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Edit Details Delete

Husk at logge af og ellers er session timeout sat til ti minutter, så man bliver smidt af efter ti minutter uden aktivitet.