Webbaserede IT-systemer

Eksamen Efterår 2021 - Program Rapport

Navne	Studienummer
Simon Hindsgaul	69227
Max Andreas de Visser	69359
Sebastian Rohr	68956
William Dyrnesli Kristensen	68842

URL til systemet:
https://wits.ruc.dk/~madv/WITS_MINI_PROJEKT/menu.php

Github:
https://github.com/maxADeVisser/WITS_blog_site/tree/master

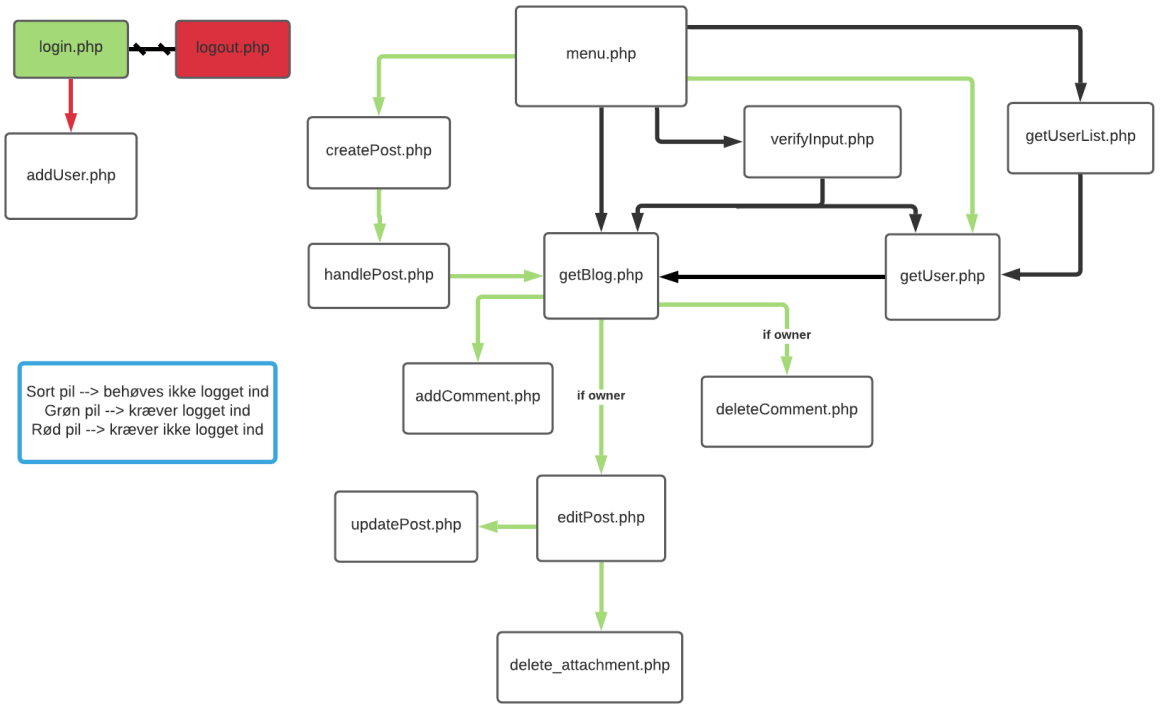
‘

System Struktur

Systemet består af 15 PHP-filer. For at skabe overblik, har vi valgt at opdele filerne i filer kun med PHP, dvs. filer der udelukkende eksekverer “server side” kode, og filer der blandt andet indeholder HTML og CSS (men også indeholder PHP), dvs. filer der blandt andet renderer i browseren (klienten). Nedenstående tabel viser opdelingen:

Filer med kun PHP (backend, server side)	Filer med HTML og CSS
- addComment.php - deleteComment.php - delete_attachment.php - logout.php - handlePost.php - updatePost.php - verifyInput.php	- login.php - menu.php - addUser.php - createPost.php - editPost.php - getBlog.php - getUser.php - getUserList.php

Vi har udarbejdet et diagram som kan hjælpe med at forstå strukturen i systemet. Ved en sort pil antydes at det er muligt at tilgå, uanset om man er logget ind eller ej. Ved grøn pil antydes at brugeren *skal* være logget ind. Ved rød pil antydes at brugeren ikke må være logget ind. Derudover kan kun login.php eller logout.php tilgås, én ad gangen aldrig begge på samme tid.



Kode forklaring

Vi bruger Bootstrap til alt CSS-styling af vores HTML. Vi tager højde for HTML-injections ved at sanitære alle de variabler der opererer som inputs i PHP echo-sætninger.

I alle filer er der implementeret funktionalitet der gør, at en bruger enten kan logge ind eller ud, samt gå tilbage til hovedmenuen (medmindre man selvfølgelig allerede er i hovedmenuen). Følgende er dokumentation for filers individuelle funktionalitet i systemet. For mere kodenær forklaring af de forskellige funktionaliteter, se kommentarer i selve koden.

- **menu.php** fungerer som hovedmenuen i systemet. Herfra kan en bruger søge efter et indlæg (via et pid) eller en bruger ved at vælge en bruger ud fra en liste af uids. Det er også muligt at oprette en nyt indlæg via createPost.php, se sine egne indlæg via getUser.php og tilgå getUserList.php.
- **login.php** er en side til at logge ind. Hvis man ikke har en bruger, kan en bruger registrere sig via en “Create New User”-knappen der eksekverer **addUser.php** (backend-fil der håndtere oprettelse af bruger).
- **logout.php** er en backend-fil, der når den eksekveres, logger brugeren ud af systemet, og nulstiller sessions-variabler. Herefter videresendes brugeren til menu.php
- **getUserList.php** er en liste af alle registrerede brugere, hvor hver bruger på listen er et hyperlink der videresender brugeren til getUser.php med den valgte brugers uid.
- **getUser.php** fungerer som en profilside for en bruger, hvor hver af brugerens post står på en liste, med et hyperlink, der videresender brugeren til getBlog.php med det valgte indlægs pid.
- **verifyInput.php** fungerer som mellemled, ved at tjekke om brugeren søger på et post eller en profilside, og anvender derefter location(header) til at sende brugeren hen til enten getUser.php eller getBlog.php.
- **createPost.php** er en menu til at oprette indlæg hvis man er logget ind, ved at udfylde titel, indhold og evt. tilføje vedhæfte billede. Det bliver sendt videre til handlePost.php.
- **handlePost.php** håndterer oprettelsen af indlægget i selve databasen ved at eksekvere de nødvendige API funktioner med inputs fra createPost.php. Herefter sendes brugeren direkte videre til getBlog.php med pid på det indlæg der er blevet oprettet.
- **getBlog.php** viser indholdet af et indlæg. Siden har øverst en titel, samt hvem forfatteren på indlægget er, stående som et hyperlink, hvor selve indlæggets indhold

og evt. billede vises under. Hvis brugeren der er logget ind er forfatter på indlægget, vises også en “Edit Post”-knap der videresender brugeren til editPost.php, hvor en bruger kan redigere indlægget. Uanset om man er forfatter til indlægget eller ej, har en bruger mulighed for at tilføje en kommentar til indlægget, hvis man er logget ind (via **addComment.php**). Yderligere, hvis en bruger er ejer af indlægget eller er forfatter på en kommentar, kan brugeren vælge at slette kommentaren (via **deleteComment.php**)

- **editPost.php** er en side til at redigere et eksisterende indlæg. Her pre-indlæses informationerne på et indlæg i redigeringsfelterne. Herfra kan man slette et billede (via **delete_attachment.php**, hvis der er et), en bruger kan tilgå getBlog.php for at få vist indlægget, og en bruger kan opdatere indholdet når det er redigeret. Indholdet bliver redigeret ved at bruge **updatePost.php** (som bruger api-metoden modify_post()).

User Interface

I udarbejdelsen af vores frontend, gjorde vi nøje overvejelser og iagttagelser af hvad større blogsider benyttede sig af, specielt heste-nettet.dk var en inspirationskilde. Vores frontend består af Bootstrap, som er et CSS rammeværktøj. Yderligere benytter Bootstrap sig af CSS- og Javascripts-designskabeloner i et grid-system. Vi benyttede os af Jakob Nielsens 10 heuristics, som et udgangspunkt for designet af vores hjemmeside.

Den heuristic vi havde mest fokus på at opfylde var “consistency and standards” (Nielsen, 2020). Det gjorde vi ved at fokusere på hvad standarder var for blog hjemmesider mht. placeringen af knapper og navngivningen af dem. En måde det kommer til udtryk på er, at log ind/log ud altid er synlig oppe i højre hjørne lige meget hvilken side brugeren befinder sig på. Ligeledes er menu altid synlig oppe i venstre hjørne (medmindre man selvfølgelig er i hovedmenuen).

Den næste heuristic vi har forsøgt at efterleve, er “match between system and real world” (Nielsen, 2020). Det er primært gjort ved at bruge kendt og relevant terminologi. Det kommer til udtryk i navngivningen af vores knapper og placeholder navne.

Et heuristic vi har haft mindre fokus på er “error prevention”. Vi har til en vis grad opfyldt den, ved at have et meget minimalistisk design, så man stort set ikke kan komme til at klikke på noget forkert. Fordelen ved det er, at det begrænser hvor mange fejl man kan lave. Vi har aktivt forsøgt at efterleve denne heuristic, ved at begrænse hvilke værdier man kan indtaste

når man søger efter post, som skal være numerisk og giver en fejlmeddelelse hvis det indeholder bogstaver.

“Recognition rather than recall” (Nielsen, 2020), har vi fulgt, ved at have en liste af alle eksisterende brugere, frem for at man skal kunne huske de forskellige brugeres uids i hovedet. Dette kunne havde været bedre opnået, hvis der kom matchende ord under søgningen af brugere i menu.

Den sidste heuristic vi har efterlevet er “aesthetic and minimalist design” (Nielsen, 2020). Det er primært den minimalistiske del vi har opfyldt, eftersom der ikke er nogen knapper, der ikke har en essentiel funktion. Dette ville være noget vi skulle holde fokus på, hvis vi arbejdede videre på hjemmesiden, fordi man hurtigt kan miste minimalismen, i takt med at man tilføjer mere funktionalitet.

Litteraturliste

Nielsen, J. (2020, November 15). *10 Usability Heuristics for User Interface Design*. Nielsen Norman Group. Retrieved November 16, 2021, from https://www.nngroup.com/articles/ten-usability-heuristics/