



CHALMERS
UNIVERSITY OF TECHNOLOGY

Restaurang hemsida

DAT076- Webb-applikationer

VT 2017

Anders Huynh (ahuynh@student.chalmers.se)
Philip Lu Chao (philipc@student.chalmers.se)
Danny Lam (ldanny@student.chalmers.se)
Johan Aronsson (johaaro@student.chalmers.se)

Beskrivning av applikationen

Restaurang hemsidan är anpassad för en specifik restaurang där man kan ta del av tjänster såsom att boka bord, titta på menyn eller att se kontaktinfo och karta. Denna hemsida agerar olika beroende på inloggning, finns ingen befintlig inloggning så agerar den som en helt vanlig restaurang hemsida där man kan boka, kolla kontaktinfo mm. Loggar man dock in sig som admin över hemsidan kan man övervaka alla bokningar som har gjorts, där kan man också sortera bokningarna efter datum och tid. Man kan även få alternativet att skapa ett arbetskonto för restaurangen. Har ett sådant konto gjorts kommer man vidarebefordras till en nyhetssida där alla arbetare kan skriva och läsa de senaste nytt inom restaurangen.

Fungerande use cases

- Lämna kommentar samt en rating (**Figur 1**)
- Titta på menyn (**Figur 2**)
- Leta upp adress på karta (**Figur 3**)
- Boka sittplatser (**Figur 4**)
- Anställda kan logga in (**Figur 5**)
- Ägaren kan se bokningar (**Figur 6**)
- Anställda kan göra ett nytt inlägg på nyhetssida (**Figur 7**)
- Anställda kan gilla en nyhet (**Figur 7**)
- Ägaren kan skapa ett nytt konto (**Figur 8**)

Flödet av applikationen

Flödet av applikationen beskrivs på ett övergripande sätt och samtliga delar av applikationen illustreras med bilder. Första sidan innehåller en välkomsttext, en bild samt en kommentarruta där besökare kan lämna en kommentar tillsammans med ett betyg. Utöver detta består sidan även av en navigeringsbar som innehåller Home, Menu, Booking och Contact.



Figur 1: Första sidan, navigeringsbar och kommentar box

Besökaren kan även navigera till Menu där man kan se menyn för dagens lunch och användaren kan även välja att titta på a la carté menyn.

Sushi Munkeback

HomeMenu ▼BookingContact

Menu

Lunch: vardagar kl. 11:00 - 15:00

Sushi och Maki-rullar	pris	lunchpris
Lite - 8 bitar 2 lax, 1 räka, 1 omelett & 4 rullar	65:-	60:-
Mellan - 10 bitar 2 lax, 1 tonfisk, 1 vitfisk, 1 räka 1 omelett & 4 rullar	80:-	70:-
Stor - 12 bitar 2 lax, 1 tonfisk, 1 vitfisk, 1 räka, 1 krabbfisk, 1 omelett & 5 rullar	90:-	80:-
Extra stor - 15 bitar 3 lax, 1 tonfisk, 1 vitfisk, 1 räka, 1 krabbfisk, 1 avokado, 1 omelett & 6 rullar	105:-	95:-
Mamma sushi - 10 bitar 2 räkor, 1 tofu, 1 omelett, 1 krabbfisk, 1 avokado & 4 rullar	80:-	70:-
Vegetariska - 8 bitar 2 avokado, 1 tofu, 1 omelett & 4 rullar	60:-	55:-

Styckpriser

Maki-rullar	10:-	Krabbfisk	10:-
Omelett	10:-	Lax	15:-
Tofu	10:-	Räka	15:-
Avokado	10:-	Vitfisk (tilapia)	15:-
Tonfisk	15:-	Bläckfisk	15:-

Figur 2: Meny sidan

Besökaren kan även hitta kontaktuppgifter om restaurangen genom att navigera vidare till Contact. I denna sida kan man hitta kontaktuppgifter samt vart restaurangen ligger på kartan.

Home	Menu	Booking	Contact
------	------	---------	---------

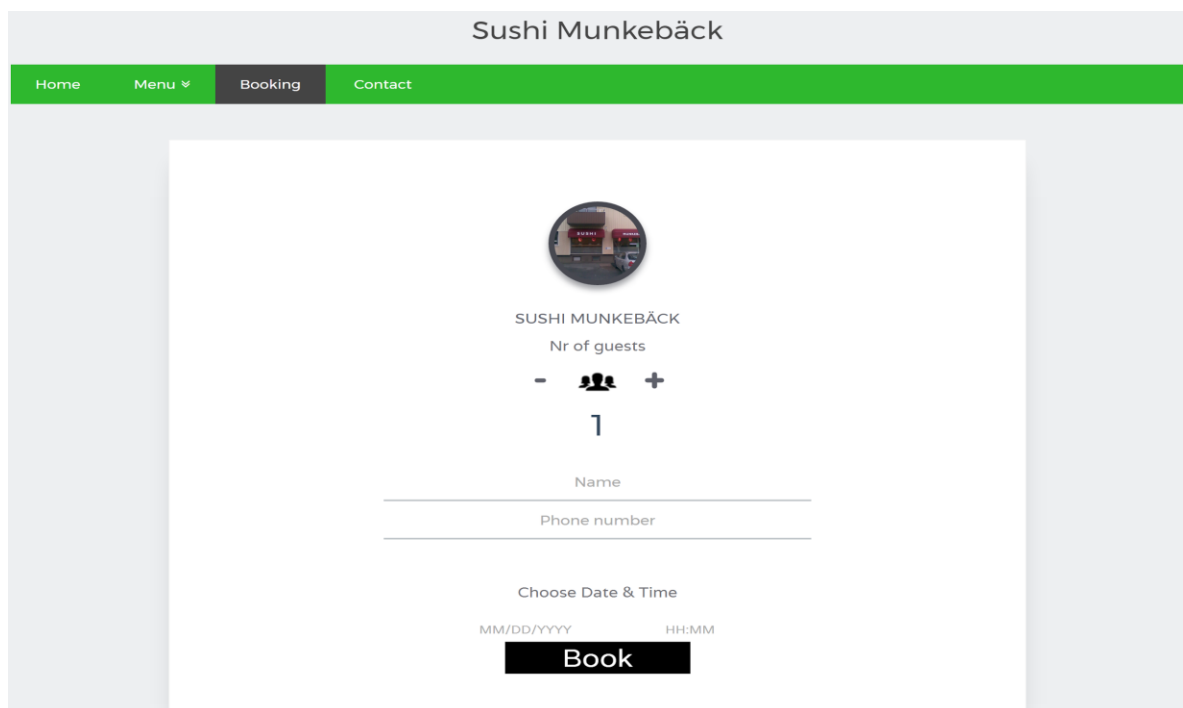
About us

- Phone nr: 031-212805
- Address: Ahrenbergsgatan 9
- 416 73 Göteborg
- info@sushimunkeback.se

Google

Figur 3: Kontaktuppgifter

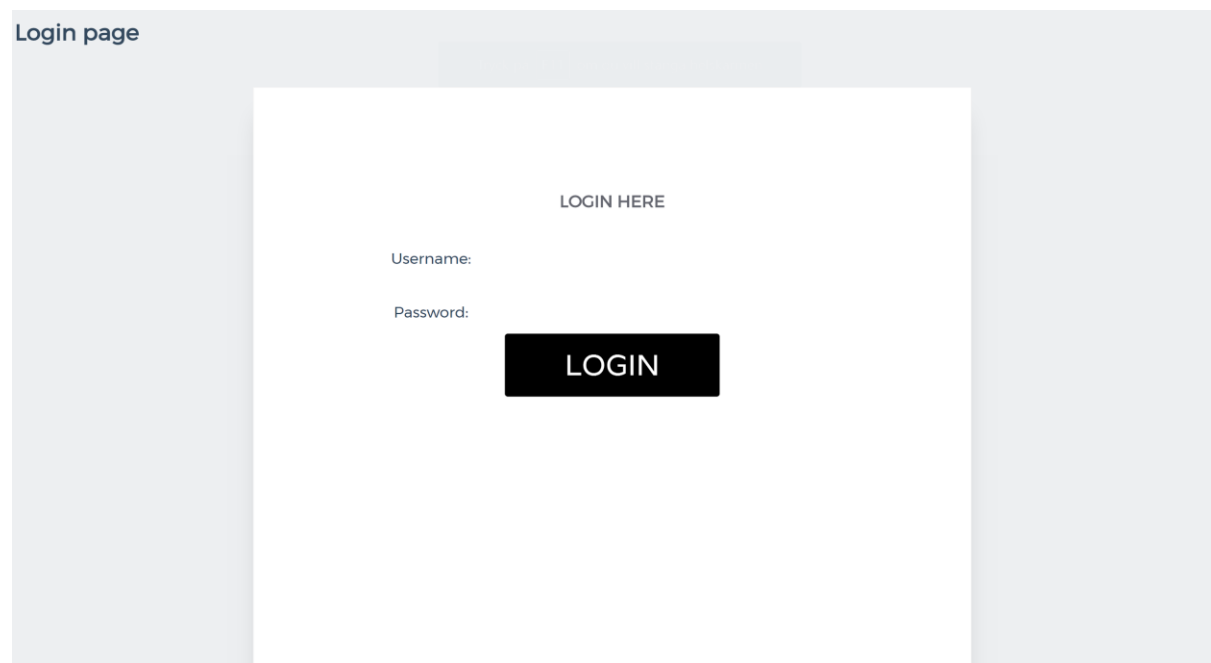
Besökaren kan även boka sittplatser i bokningssidan där man behöver fylla i namn, telefonnummer, antal sittplatser, datum och tid. Efter lyckad bokning kommer det ett meddelande.



The screenshot shows the 'Sushi Munkebäck' booking interface. At the top is a navigation bar with 'Home', 'Menu', 'Booking' (active), and 'Contact'. The main content area features a circular logo of the restaurant, the name 'SUSHI MUNKEBÄCK', and a section for 'Nr of guests' with a minus button, an icon of two people, and a plus button, with the number '1' displayed below. Below this are input fields for 'Name' and 'Phone number'. Further down is a 'Choose Date & Time' section with labels 'MM/DD/YYYY' and 'HH:MM'. At the bottom is a large black 'Book' button.

Figur 4: Bokningssidan

Som en anställd kan man logga in och verifiera sig själv för att komma åt sidorna såsom bokningslistan, nyhetsflödet.



The screenshot shows the 'Login page' for employees. It has a header 'Login page' on the left. The main content area is titled 'LOGIN HERE' and contains labels for 'Username:' and 'Password:' followed by input fields. A large black 'LOGIN' button is positioned below the password field.

Figur 5: Inloggningssidan för de anställda

En inloggad anställd kan man komma åt en sektion för att se en lista alla bokningar. På den sidan kan de även filtrera ut bokningar efter datum och tid.

Figur 6: Sidan innehållande alla bokningar

List of all Bookings

Go Back Delete

Showing 1 to 5 of 5 entries

Search:

Customer name	Customer phone number	Reservation date	Reservation time	Amount of people
Jonas	0737274918	Thursday, 16 March, 2017	15:00	1
Lasse	0731817162	Thursday, 30 March, 2017	13:00	1
Lisa	0808080811	Wednesday, 29 March, 2017	09:30	6
Meja	0789898911	Friday, 21 April, 2017	18:30	2
Simon	0737274716	Friday, 31 March, 2017	18:30	1

Search Customer name

Search Phone number

Search Reservation date

Search Reservation time

Search Amount of people

Showing 1 to 5 of 5 entries

Previous1Next

Utöver bokningar kan en anställd lämna kommentarer internt för alla anställda, i ett så kallade nyhetsflöde.

Comments

Name

Enter your comment here

Comment by [Html Comment Box](#)

(23 hours ago) **Tintanäe Mätine** said:


Ja absolut! :)

1

(23 hours ago) **Mätine Tintanäe** said:

Hej, någon som kan byta skift med mig imorgon?

1



Figur 7: Nyhetsflöde för de anställda

Restaurangägaren har även ett speciellt konto där han kan skapa nya konton för de anställda på restaurangen

Create new user

CREATE NEW EMPLOYEE ACCOUNT

User created

Username: Håkan

Password:

CREATE

[Go Back](#)

Figur 8: Skapa nya användare

Klient

Vi har valt att använda oss av html kombinerat med css för att rendera sidorna på klientsidan. Vi valde dessa då vi hade mest erfarenhet inom ramverken och de täckte alla kriterier vi hade. Som komplement till dessa använde vi oss också utav mustache för att skapa templates. På så sätt kunde vi rendera listor utav bokningar m.m. Dessa html sidor renderas allt eftersom, detta då vissa html sidor ej skall vara renderade till en början. Det vi skapat är en så kallad bassida som agerar som en bakgrund till allt annat som skall renderas. Detta görs genom att vid start av hemsidan renderas bassidan där det sedan renderas ytterligare html sidor vid behov i en div som finns på bassidan.

Server

Node.js har använts för att bygga vår applikation, som ramverk använder vi oss av express.js. Med hjälp av detta ramverk kan vi på ett enkelt sätt svara på en begäran från klienten. Då en "request" skickas till express så kommer den att gå vidare till en av de "end points" som skapats. Här skickas sedan begäran vidare beroende på vad som skall göras. Någon typ av "middleware" tar upp begäran och renderar en vy, autentiserar en användare etc. Det tillåter helt enkelt oss att navigera i applikationen på ett väldigt smidigt sätt.

Middleware

Vi har använt oss av ett antal "middleware" moduler kopplade till node.js för att skapa denna applikation. Nedan följer en kort resumé över dessa och hur vi har använt oss av dem.

Passport

Denna modul används för att autentisera en användare, både "admins" och "employees". Detta gör att vissa sidor inte kan kommas åt om man inte är inloggad som en viss användare. Applikationen skapar två strategier, en för "login" och en för "signup", se Figur 7 för login. När en begäran görs via express från login sidan så kommer login strategin användas för att autentisera användaren och se till att denne finns med i vår databas. Vi valde att arbeta med passport för att det enkelt gick att integrera med express. Det har också möjlighet till OAuth autentikation vilket gör det utbyggbart det skulle bli aktuellt att använda sig av sådana möjligheter.

Vi använder två ytterligare moduler tillsammans med denna för att öka säkerheten och för att interagera med användaren. Dessa heter Bcrypt och connect-flash, mer information följer nedan.

```

passport.use('local-signin', new LocalStrategy({
  usernameField : 'username',
  passwordField : 'password',
  passReqToCallback : true
},
function(req, username, password, done) {
  admin.findOne({
    if (user == null) {

    if (bcrypt.compareSync(password, user.password)) {
      return done(null, user)
    }

    return done(null, false, req.flash('message', 'Wrong password'));
  }).catch(function(err) {}
  })
}))

```

Figur 7: Passport strategi och bcrypt

Bcrypt

Används för att "hasa" de lösenord som skapas till varje användare, vilket gör så att lösenordet som sparas i databasen inte går att tyda i skrift. I samband med login och "signup" strategierna så görs "queries" till databasen. Här används funktioner från "bcrypt" som "hashar" lösenordet. Se implementation i Figur 7.

Connect-flash

Denna modul används för att skicka flash meddelanden i applikationen. I samband med att vi "router" så skickar vi med flash meddelanden som vi m.h.a mustache renderar på vyn. Detta för att användaren får respons från applikationen kring vad som händer. Ett exempel som händer i applikationen är då en ny användare skapas. Då vi trycker på knappen "Creat user" kommer express att ta emot denna begäran, och ruta vidare tillsammans med ett flash meddelande "User created". Det här renderas sedan upp på den sida vi router till. Detta kan också ses i figur 8.

Cookieparser

Cookieparser används för att spara användning av hemsidan, alltså den sparar allt användaren gör på hemsidan som "secrets". Dessa "secrets" kan sedan användas vid nästa session av användning.

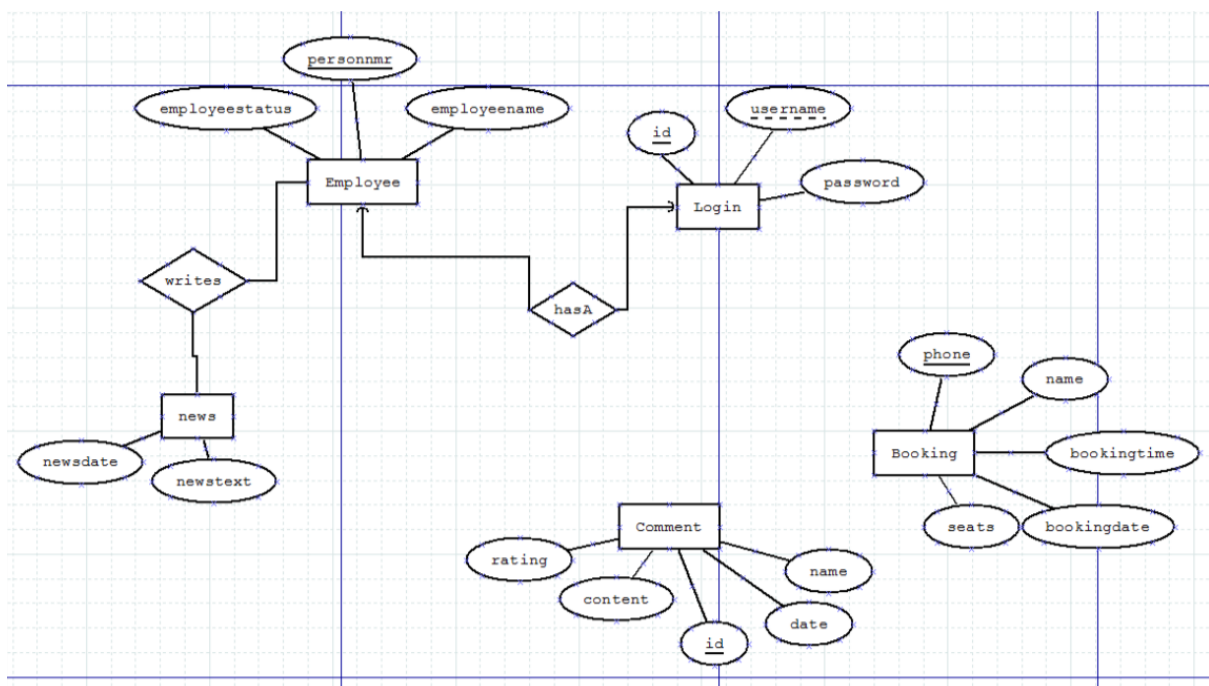
Express-session

Express-sessions används för att anropa cookieparser som sparar cookies vid avslutning av en session på hemsidan. Den används även vid start av användning genom att hämta de cookies som cookieparser sparar och laddar de vid uppstart av hemsida.

Databas

Databasen är byggd med hjälp av MySQL och för att interagera med databasen använder vi oss av sequelize. Då sequelize är byggd för att interagera med databaser och designad för NodeJS var valet givet för oss att använda det. Sequelize underlättar för att radera, hämtning, insättning samt redigering i databasen.

Tanken var skapa en databas som innehåller följande entiteter: Employee, Login, Comment, Booking och News. Där följande relationer existerar: En anställd har en login och kan skriva till nyhetsflödet. Bokningar och kommentarerna ligger som fristående entiteter utan några relationer till andra entiteter då vi anser att det är icke relevant för webbapplikationen. Se figur 7 för en vy av vår ursprungliga och ideala databasstruktur.

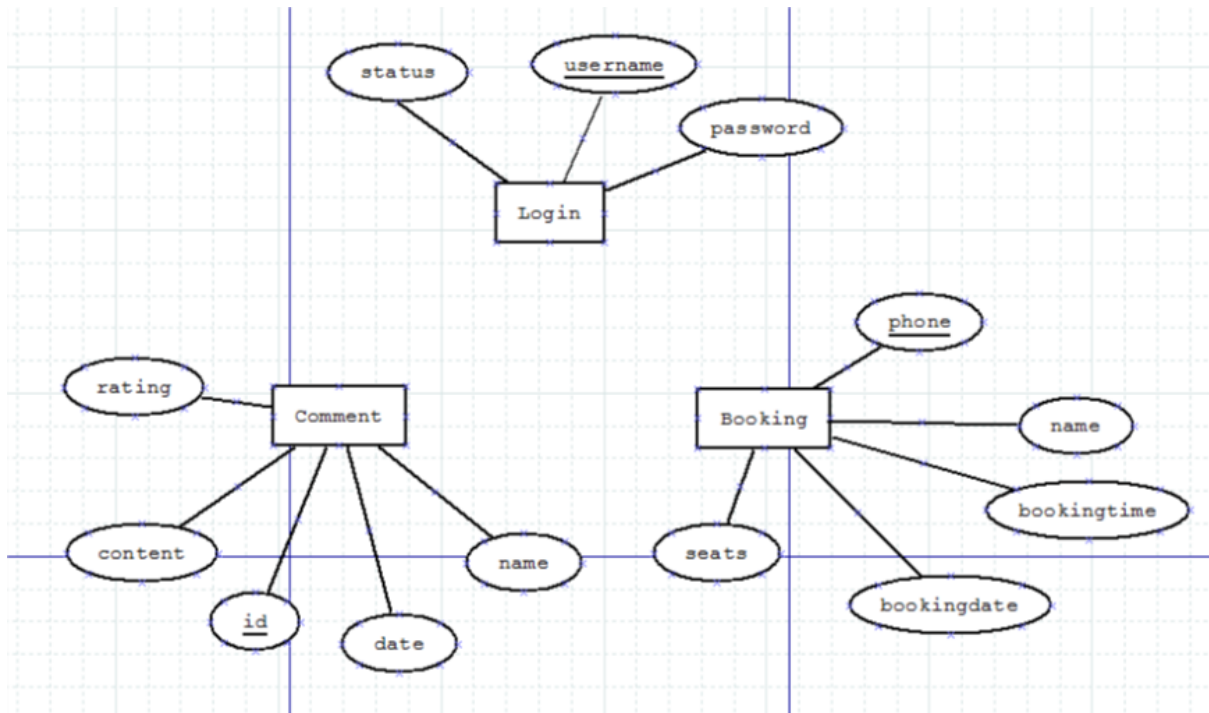


Figur 8: Ursprungliga databasstruktur

På grund av tidsbrist, komplikationer under sammankopplingen valde vi att ändra vår databasstruktur sådan att den uppfyller de grundliga kraven och att den nya strukturen uppfyller applikationens syfte. För att uppfylla databasens funktionalitet för vår applikation krävs det dessa entiteter, Login, Booking, Comment. Vi valde att låta entiteterna vara fristående, utan några relationer till varandra då det inte är ett måste för att uppfylla våra krav, samt att det underlättar sammankopplingen av det hela.

- Entitet Login: Innehåller uppgifter för ett login.
- Entitet Booking: Innehåller information om bokningarna som görs i applikationen
- Entitet Comments: Innehåller kommentarerna och betygsättningen som görs av en besökare.

Se figur 8 för den nuvarande databasstrukturen applikationen använder sig utav.



Figur 9: Nuvarande databasstruktur

Test

Testen är skapade med test ramverket mocha med expect biblioteket som finns tillgängligt med chai. Samt chai-http för att testa routes.

Struktur

Vår struktur är uppbyggd på följande sätt:

auth - Innehåller passport som används för att logga in och skapa användare.

Database - Fil för att ansluta till databasen med sequelize.

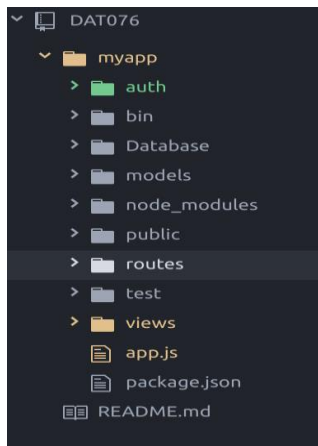
models - Databas modeller som kan ses i figur 8 finns tillgängliga här, skapas med sequelize.

public - Innehåller css och javascript filer som används av html filer, samt bilder.

routes - Sköter flödet i applikationen, dirigerar om till och renderar "views".

views - Innehåller alla html sidor som renderar det som användaren ser.

Strukturen kan också ses på figur 10.



Figur 10: Struktur

Vad skulle en annan programmerar ha nytta av att veta

För en extern programmerare är det bra att veta de överliggande verktygen såsom NodeJS och express ramverket. Då dessa ligger till grund för applikationens funktion. Kortfattad information om detta finns i rapporten men mer information om detta finns tillgängligt på nätet. Det är också bra att läsa ovanstående del om passport för att förstå hur autentiseringsprocessen fungerar.