The Tasty Recipes, PHP och MySQL

Applikationer för internet, ID1354

Daniel Fäldt Hooshidar dhoo@kth.se
2018-11-18

https://github.com/Hooshidar/TastyRecipes

Introduktion

Uppgiften var att lägga till ytterligare funktionalitet på hemsidan *”The Tasty Recipes”*. Detta gjordes med hjälp av PHP och MySQL. Kravspecifikationerna för hemsidan är följande:

- Tillåt användare att logga in på hemsidan
- Användare ska kunna skriva kommentarer på recepten
- Användaren ska kunna ta bort sina egna kommentarer

För fler poäng ska sidan även innehålla:

- Ett ”bli medlem” formulär
- XML till recepten

Litteraturstudie

För att få kunskaper om PHP och MySQL användes w3shools, PHP-manualen samt youtube-videor. Föresläsnings-slides har även var till hjälp vid inlärning.

Youtube-videor har varit det nyttigaste för mig då det finns mycket material att titta och lära sig av. Man får se exempel där kod implementeras och används på olika sätt.

Metod

Jag har valt att arbeta i Dreamweaver då det blir enkelt att testa koden som skrivs mot apache-servern samt testa olika webbläsare.

För att utveckla hemsidan användes phpMyAdmin för databashantering. phpMyAdmin följde med XAMPP som används för apache-servern. Jag behövde alltså bara starta phpMyAdmin via XAMPP och sedan skapa en tabell för användare och kommentarer. När databasen var klar skrev jag en databashanterare som gör att min kod kan nå databasen. Användare som finns i databasen har ett användarnamn, e-mail och ett lösenord, databasen sparar sedan en id för varje användare som använder AUTO_INCREMENT som en säkerhet om något skulle hända med användarnamn eller e-mail. Kommentarerna som sparas har ett meddelande, användaren, datum och ett kommentars-id. Denna id använder också AUTO_INCREMENT för att alltid vara unikt.

Sedan skrevs kod för att kunna komma åt och fylla dessa olika tabeller. Först skapades ett formulär där användare kan registrera sig på hemsidan. När en användare är registrerad kan användaren lägga till kommentarer på recepten. Alltså skapades kommentarsfältet för recepten. När dessa databaser fungerade lades det till en radera-knapp där användaren som skrev kommentaren kan radera den.

Alla olika ”actions” som triggas på hemsidan startar en egen php-fil. Jag la alla dessa filer i en egen mapp ”includes” för att få en bättre struktur. Alla filer som räknas som includes har en inc.php på slutet av filnamnet.

Under tiden som allt skapades hade jag hemsidan igång på min apache-server för att testa och se om hemsidan agerar som jag vill.

Jag uppdaterade även CSS:en kontinuerligt när nya sektioner lades till.

Resultat

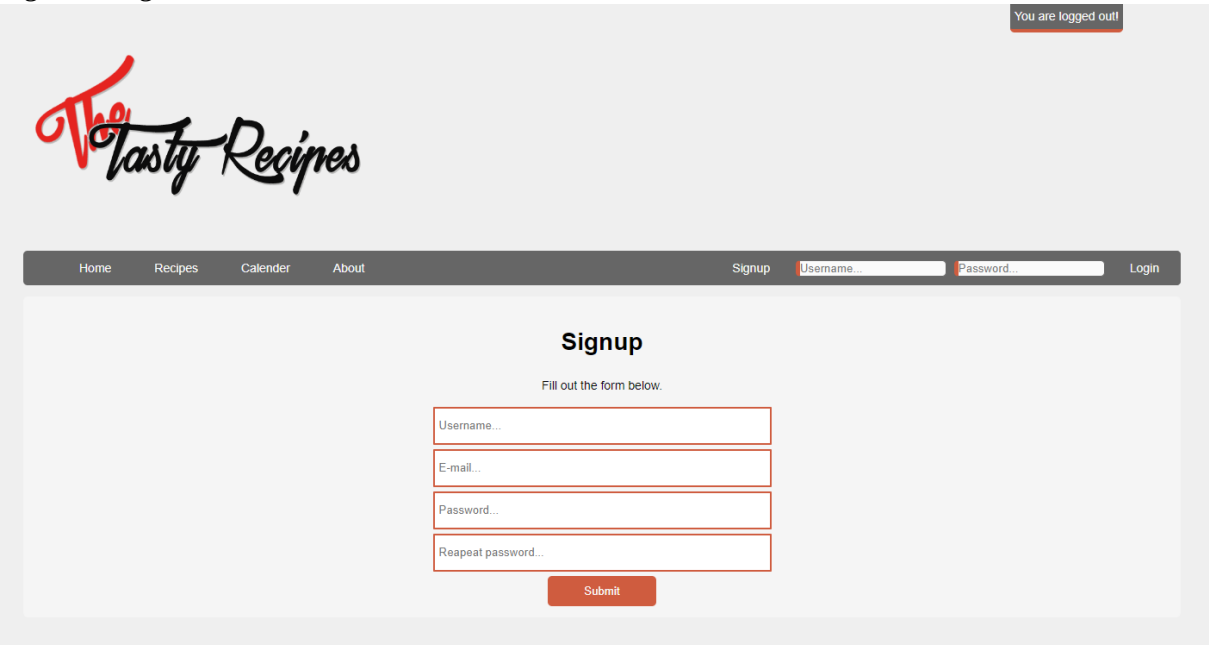
Här kommer jag presentera hur hemsidan utvecklats samt de nya funktionaliteterna.

Utseende

Den största ändringen utseendemässigt är att jag lagt till möjlighet att logga in och registrera sig i navigerings-baren samt en liten indikator i högra hörnet som säger om man är inloggad eller inte.

Figur 1: Navigeringsfältet

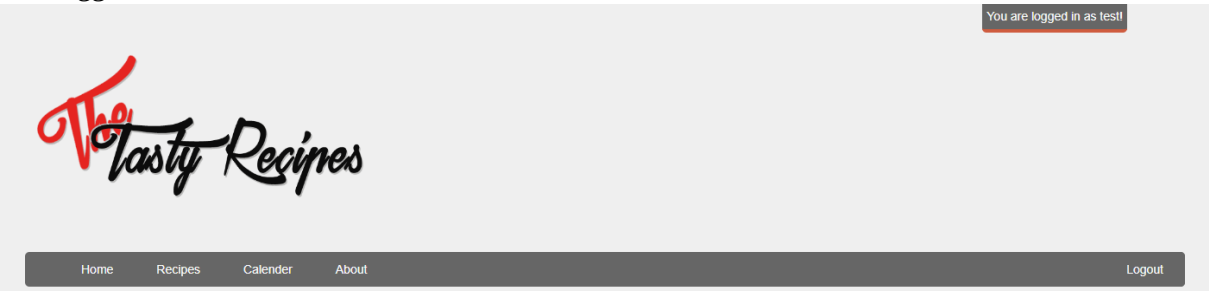
Om man sedan klickar in sig på ”Signup” tas man till ett registreringsformulär som låter användaren registrera sig.



Figur 2: Registreringsformuläret

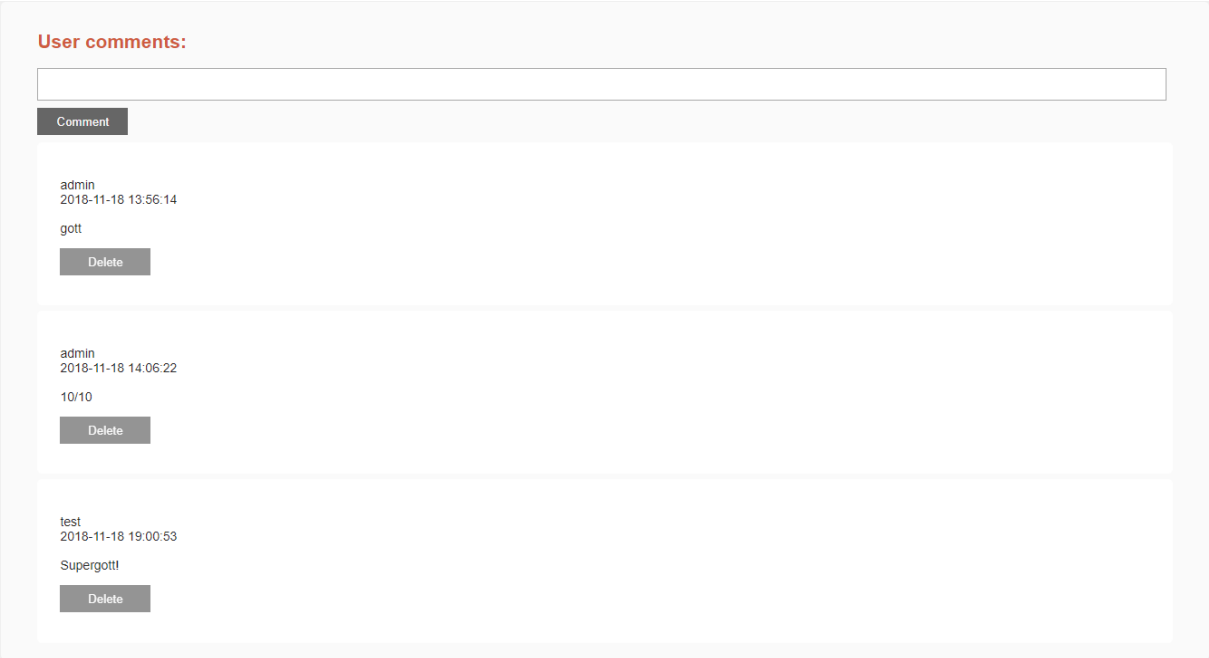
När en registrerad användare sedan loggat in ändras utseendet av navigerings-baren för att gömma information som inte längre är relevant. Inloggningsindikatorn ändrar också text samt visar vem man

är inloggad som.



Figur 3: navigeringsfältet inloggad

Användaren kan även skriva och ta bort egna kommentarer. Kommentarerna visar datum när kommentaren skrevs samt vilken användare som skrev det.



Figur 4: Kommentarsfältet

Om en användare inte är inloggad går det inte att skriva kommentarer men man kan fortfarande se alla skrivna kommentarer.

Figur 5: Kommentarsfältet utloggad

Funktionalitet

För att navigeringsbaren inte ska visa irrelevant information används en IF-sats som kollar om det finns en användare inloggad.

```
if (isset($_SESSION['userId'])){
    echo '<form action="includes/logout.inc.php" method="POST">
   <button type="submit" name="logout-submit">Logout</button>
   </form>';
}
else {
    echo'<form action="includes/login.inc.php" method="POST">
   <input type="text" name="mailuid" placeholder="Username...">
   <input type="password" name="pwd" placeholder="Password...">
   <button type="submit" name="login-submit">Login</button>
   </form>
   <form action="Signuppage.php" method="POST">
   <button type="submit">Signup</button>
   </form>';
}
```

Figur 6: Navigeringsfältet kod

För inloggning kollas det om användarmanet eller e-mailen som skrivs in finna i databasen och sedan om det matchar det lösenord som skrevs. Det finns även en kod-bit som kollar om användaren klickar

på submit utan att fyllt i något fält.

```php
if (empty($mailuid) || empty($password)){
    header("Location: ../index.php?error=emptyfields");
    exit();
}
else {
    $sql = "SELECT * FROM users WHERE uidUsers=? OR emailUsers=?;";
    $stmt = mysqli_stmt_init($conn);
    if (!mysqli_stmt_prepare($stmt, $sql)){
        header("Location: ../index.php?error=sqlerror");
        exit();
    }
    else {

        mysqli_stmt_bind_param($stmt, "ss", $mailuid, $mailuid);
        mysqli_stmt_execute($stmt);
        $result = mysqli_stmt_get_result($stmt);

        if ($row = mysqli_fetch_assoc($result)){
            $pwdCheck = password_verify($password, $row['pwdUsers']);
            if($pwdCheck == false){
                header("Location: ../index.php?error=wrongpwd");
                exit();
            }
            else if ($pwdCheck == true){
                session_start();
                $_SESSION['userId'] = $row['idUsers'];
                $_SESSION['userUid'] = $row['uidUsers'];

                header("Location: ../index.php?login=success");
                exit();
            }
            else {
                header("Location: ../index.php?error=wrongpwd");
                exit();
            }
        }
        else {
            header("Location: ../index.php?error=nouser");
            exit();
        }
```

Figur 7: Inloggningskoden

När en användare ska registrera sig ses det till att fälten skrivs in rätt samt att användaren tar ett unikt användarnamn. För varje fel som görs ges ett litet meddelande om vad som är fel samt skrivs det i URLen (se kodens kommentarer i bilden nedan var tydligare förklaring).

```php
<?php
/*Ser till så att personen klickade på signup-knappen*/
if (isset($_POST['signup-submit'])){
    require 'dbhLogin.inc.php';

    $username = $_POST['uid'];
    $email = $_POST['mail'];
    $password = $_POST['pwd'];
    $passwordRepeat = $_POST['pwd-repeat'];

    /* Kollar om något av fälten är tomma, isåfall skickas man tillbaka till signuppage.php och den info som skrevs visas i adressfältet (förutom password) */
    if (empty($username) || empty($email) || empty($password) || empty($passwordRepeat)){
        header("Location: ../Signuppage.php?error=emptyfields&uid=".$username."&mail=".$email);
        exit();
    }
    /* Ser till så att formatet på mailet är korrekt samt om de används de tllåtna tecken nedan.*/
    else if(!filter_var($email, FILTER_VALIDATE_EMAIL) && !preg_match("/^[a-zA-Z0-9]*$/", $username)){
        header("Location: ../Signuppage.php?error=invalidmailuid");
        exit();
    }
    /* Ser till så att formatet på mailet annars skickas man tillbaka till signuppage.php med ifyllt usename*/
    else if(!filter_var($email, FILTER_VALIDATE_EMAIL)){
        header("Location: ../Signuppage.php?error=invalidmail&uid=".$username);
        exit();
    }
    /* Ser till så att det används tillåtan tecken ananrs skickas man tillbaka till signuppage.php med ifylld email*/
    else if(!preg_match("/^[a-zA-Z0-9]*$/", $username)){
        header("Location: ../Signuppage.php?error=invaliduid&mail=".$email);
        exit();
    }
    /*Ser till så att använder skriver korrekt password annars skickas man tillbaka till singuppage.php med ifyllt username och email*/
    else if($password !== $passwordRepeat){
        header("Location: ../Signuppage.php?error=passwordcheck&uid=".$username."&mail=".$email);
        exit();
    }
    /* Kollar om username redan finns*/
    else {
        /*hitta usernames i databasen. använder sig av en palceholder "?" för att folk inte ska kunna förstöra databasen*/
        $sql = "SELECT uidUsers FROM users WHERE uidUsers=?";
        $stmt = mysqli_stmt_init($conn);
        /* Kollar om statement (stmt) är redo för exekvering i $sql*/
        if (!mysqli_stmt_prepare($stmt, $sql)){
            header("Location: ../Signuppage.php?error=sqlerror");
            exit();
        }
        else {
            /* Kör det inskrivna användarnamnet mot databasen för att se om det redan finns. mysql_stmt_num_rows returnerar True eller False */
            mysqli_stmt_bind_param($stmt, "s", $username);
            mysqli_stmt_execute($stmt);
            mysqli_stmt_store_result($stmt);
            $resultCheck = mysqli_stmt_num_rows($stmt);
            if($resultCheck > 0){
                header("Location: ../Signuppage.php?error=usertaken&mail=".$email);
                exit();
            }
            /* Sista checken som ser till att vi faktsikt kan köra vårat statement mot databasen */
            else {
                $sql = "INSERT INTO users (uidUsers, emailUsers, pwdUsers) VALUES (?, ?, ?)";
                $stmt = mysqli_stmt_init($conn);
                if (!mysqli_stmt_prepare($stmt, $sql)){
                    header("Location: ../Signuppage.php?error=sqlerror");
                    exit();
                }
                /* om du tagit dig hit har du klarat alla prövningar. Här läggs informationen in i databasen*/
                else {
                    $hashedPwd = password_hash($password, PASSWORD_DEFAULT);

                    mysqli_stmt_bind_param($stmt, "sss", $username, $email, $hashedPwd);
                    mysqli_stmt_execute($stmt);
                    header("Location: ../Signuppage.php?signup=success");
                    exit();
                }
            }
        }
    }
    /* Stänger av vårat statement och stänga av connection för att spara på resurser*/
    mysqli_stmt_close($stmt);
    mysqli_close($conn);
}
/* Om man kom åt formuläret på annan väg skickas man till signuppage för signup */
else{
    header("Location: ../Signuppage.php");
    exit();
}
```

Figur 8: Registreringsformulärets kod

För kommentarerna finns det setComments, getComments och deleteComments. setComments lägger in kommentarer i databasen genom att ta användarnamnet, datumet och meddelandet från ”formen” som fylls i och placera det i databasen.

```php
<?php
require 'dbhLogin.inc.php';
if(isset($_POST['commentSubmit'])){
    $uid = $_POST['uid'];
    $date = $_POST['date'];
    $message = $_POST['message'];

    $sql = "INSERT INTO comments (uid, date, message)
                VALUES ('$uid', '$date', '$message')";
    $result = mysqli_query($conn, $sql);
    header("Location: ../Meatballs.php");
}
```

Figur 9: koden för setComments

getComments skriver ut kommentarerna från databasen till kommentarsfältet genom en query i databasen där vi för varje hittad kommentar skriver ut den.

```php
<?php
require 'dbhLogin.inc.php';
$sql = "SELECT * FROM comments";
$result = mysqli_query($conn, $sql);
    while($row = mysqli_fetch_assoc($result)){
        $id = $row['uid'];
        echo "<div class='commentBox'><p>";
        echo $row['uid'].'<br>';
        echo $row['date'].'<br><br>';
        echo nl2br($row['message']);
        echo "</p>
            <form class='delete-form' method='POST' action='includes/deleteComments.inc.php'>
                <input type='hidden' name='cid' value='".$row['cid']."'>
                <button type='submit' name='commentDelete'>Delete</button>
            </form>
            </div>";
    }
```

Figur 10: koden för getComments

deleteComments tar bort en kommentar om det är rätt användare som skapat kommentaren genom att göra en query mot databasen och om kommentarens användar-id och användarnamnet på den inloggade stämmer tas meddelandet bort.

```php
<?php
session_start();
require 'dbhLogin.inc.php';
if (isset($_POST['commentDelete'])){
    $cid = $_POST['cid'];
    $uid = $_SESSION['userUid'];

    $sql = "DELETE FROM comments WHERE cid='$cid' AND uid='$uid'";
    $result = mysqli_query($conn, $sql);
    header("Location: ../Meatballs.php");
}
```

Figur 11: Koden för deleteComments

Heuristik

- **Visibility of system status**
  Användaren ser alltid statusen om hen är inloggad eller inte. I URLen får man alltid information om var man befinner sig samt en rubrik ovanför innehållet på varje sida

- **Match between system and the real world**
  Innehållet på alla sidor har alltid samma struktur och layout. Det är enkelt att följa informationen och navigera sig mellan sidor.

- **Consistency and standards**
  Språket på sidan är koncist och instruktioner betyder samma sak på alla sidor. Hemsidans navigeringsfält är också något som gör att användaren känner sig ”hemma” då nästan alla stora hemsidor använder en sådan.

- **Recognition rather than recall**
  Navigeringsfältet finns på alla sidor och gör det enkelt för användaren att navigera sig fram från olika sidor. Det gör att användaren inte behöver leta sig fram för att hitta något specifikt utan hen vet ungefär hela tiden hur man tar sig vidare.

- **Aesthetic and minimalist design**
  Hemsidans layout överlag använder inte för många olika färger eller för många fönster som gör det krångligt. Det används heller inte för många skuggor eller annorlunda fonter som gör det svårt att läsa.

Diskussion

Att lära sig php och lite MySQL har varit roligt men utmanande. Jag tror att jag har haft mycket användning av det tidigare programmeringskurserna i utbildningen då det gör det lättare att sätta sig in i ett nytt språk. Med det sagt har det varit svårare att veta hur man gör databaser på bästa sätt. Till en början gjorde jag olika databaser för varje delmoment, tex en databas för användare och en annan för kommentarer. Detta kan man såklart göra i samma databas med flera olika tabeller och det underlättar väldigt mycket då man enbart behöver en databashanterare. Sedan har det varit klurigt att komma fram till bästa sättet att ha olika kommentarsfält, om det är bästa att använd sig av flera olika tabeller eller om man lägger in en ytterligare kolumn som har någon form av id som specificerar till vilket kommentarsfält den ska hamna.

Det som hjälpt mig mest har varit youtube-videor då det är enkelt att följa någon som hela tiden förklarar vad som händer och varför det görs. Problemet blir att man inte lär sig lika mycket som hade gjort om man läste en bok eller manualer då man hela tiden får precis det man söker. Som tur är finns det inget som är exakt det man säker så man hela tiden måste gå in och anpassa koden för eget bruk.