30/03-25

Leon Edlund
19990510-8115
1ME236

Gemensam to-do lista

Beskrivning av applikation

Applikationen som har skapats är en gemensam to-do lista tänkt att underlätta vid samarbete på olika projekt. Användare som är medlemmar i en lista kan skapa uppgifter som behöver utföras och med hjälp av ett poängsystem hålla koll på vem som bidragit mest i projektet.

Kravspecifikation

- Användare måste logga in för att kunna använda appen.
- En användare ska kunna skapa en gemensam todo-lista.
- En användare ska kunna bjuda in andra användare till sin todo-lista genom att dela en länk.
- En viss användare kan endast läggas till i samma todo-lista en gång.
- En användare ska inte kunna uppdatera data som tillhör någon annan användare.
- Det ska finnas ett grafiskt gränssnitt där användare kan:
  - Se vilka uppgifter som är klara/inte klara.
  - Se vem som gjort vad.
- Alla deltagare i en todo-lista ska kunna:
  - Lägga till nya uppgifter.
  - Markera uppgifter som avklarade.
- Uppgifter i listan ska innehålla:
  - En titel (obligatorisk).
  - Antal poäng (obligatoriskt).
  - En beskrivning (valfri).
- Avklarade uppgifter ska inte försvinna utan bara markeras som avklarade.
- Vem som klarade en uppgift ska sparas.
- Den användare som markerar en uppgift som klar får det antal poäng som uppgiften är värd.
- I anslutning till varje todo-lista ska det tydligt visas hur många poäng varje deltagare har.

Egna Krav

- En unik bild som identifierar och är kopplad till varje lista.
- Användare ska visuellt kunna överskåda hur mycket av ett projekt som blivit avklarat.

Leon Edlund
19990510-8115
1ME236

Single page application

Applikationen är en single page application (SPA), vilket enligt MDN (u.å.a) innebär att endast ett HTML-dokument returneras från servern, och innehållet uppdateras dynamiskt med JavaScript och fetch-anrop.

Denna struktur gör att appen kan erbjuda en användarupplevelse som liknar en traditionell mobilapp, där navigering sker utan fullständig omladdning. Med mer tid hade animationer och övergångar kunnat utvecklas vidare för att förstärka denna känsla ytterligare.

Custom Elements och shadow-dom

Hela gränssnittet är uppbyggt med custom elements, vilket innebär att egna HTML-element med specifik funktionalitet definieras genom klasser som ärver från HTMLElement och registreras via *customElements.define()* (MDN, u.å.b).

Genom att använda custom elements är det möjligt att skapa återanvändbara element och kapsla in funktionalitet kopplat till specifika delar av applikation. S*hadow DOM* används för att skapa en inre DOM struktur kopplad till elementen vilket skyddar dem från att bli påverkade av extern CSS eller Javascript och leder till mer robust kod (MDN u.å.c).

Routing

För frontend-routing används det externa biblioteket *vanilla-router*, som erbjuder ett enkelt sätt att definiera sökvägar med tillhörande route-handlers. Detta möjliggör navigering mellan olika vyer utan att sidan behöver laddas om.

Innan en vy visas görs en validering om användaren är inloggad genom ett API-anrop till en endpoint som försöker hämta användarens information. Om användaren inte är autentiserad returnerar servern en HTTP 401-kod, och användaren omdirigeras till inloggningssidan.

Backend och säkerhet

Backenden är byggd med PHP och strukturerad som ett REST-API med hjälp av ramverket *Slim*. All data lagras i en MySQL-databas, och kommunikationen sker med JSON.

Databasen består av fyra tabeller: *users, projects, project_members och tasks.* Varje tabell har en unik primärnyckel som sätts med AUTO_INCREMENT för att identifiera olika rader. Dessa ID:n används även som främmande nycklar för att koppla samman information från olika tabeller och förhindrar duplicerad data, vilket gör databasen mer effektiv och minskar risken för fel vid uppdatering av information (Nixon, 2018, ss. 210–211).

Exempelvis används projektets ID som en främmande nyckel i *tasks*-tabellen för att koppla uppgifter till rätt lista och tabellen *project_members* knyter användare till projekt via *user_id* och *project_id*.

Prepared statements används för alla SQL-frågor för att förhindra SQL-injektion, och autentisering sker via Google OAuth, vilket ökar säkerheten då lösenord och användarnamn inte behöver sparas direkt i databasen.

Utvärdering av kravspecifikationen

Samtliga krav i kravspecifikationen uppfylls. Användaren kan logga in via Google för att skapa och se data kopplat till sitt konto. Funktionaliteten för att skapa en lista är implementerad, och andra användare kan bjudas in via en delbar länk för att delta i en gemensam lista. Uppgifter med poäng kan skapas och när en användare markerar en uppgift som avklarad, sparas poängen och totalpoäng uppdateras i en highscore-lista som är synlig för alla deltagare.

En slumpad SVG-bild kopplas till varje projekt för att göra applikationen mer visuellt tilltalande och skapa visuell igenkänning. En progress-bar har även implementerats för att visa hur mycket av listan som har avklarats. Detta ger en tydlig översikt över projektets status och är tänkt att motivera deltagarna att slutföra listan.

Externa kodbibliotek

Slim
Används för att skapa API endpoints med tillhörande route-handlers på backenden.
Webbplats: https://www.slimframework.com/

Vanilla-router
Hanterar frontend routing och ansvarar för att ladda in rätt vy baserat på route. Använder history mode vilket sparar föregående sida i webbläsarens historik och möjliggör navigering med “fram” och “bak”- knapparna.
Webbplats: https://www.npmjs.com/package/vanilla-router

Gsap
Används för animationer, till exempel vid öppning/stängning av menyn.
Webbplats: https://www.npmjs.com/package/gsap

Useless-blobs
Används för att generera en slumpad SVG bild som sedan sparas i databasen och kopplas till en specifik lista.
Webbplats: https://github.com/jbukuts/useless-blobs

Date-fns
Används för att konvertera datumet då ett projekt skapats till dd/mm-yy.
Webbplats: https://www.npmjs.com/package/date-fns

Länk

https://melab.lnu.se/~le223nd/webbteknik-6/to-do-app/

Leon Edlund
19990510-8115
1ME236

Källor

MDN (u.å.a). *SPA (Single-page application)*. https://developer.mozilla.org/en-US/docs/Glossary/SPA [30-03-25]

MDN (u.å.b). Using custom elements. https://developer.mozilla.org/en-US/docs/Web/API/Web_components/Using_custom_elements [30-03-25]

MDN (u.å.c). *Using shadow DOM*. https://developer.mozilla.org/en-US/docs/Web/API/Web_components/Using_shadow_DOM [30-03-25]

Nixon, R (2018). *Learning PHP MySQL JavaScript - With jQuery CSS HTML5*. uppl. 5. O'Reilly Media, Inc: Sebastopol.