Gruppuppgift JS3 – Robin Z, Arvid, Ylva

Interaktion

Vår applikation är en fototjänst likt Instagram där man ska kunna göra följande:

- Kommentera en bild som en användare med ett namn man själv anger
- Redigera/radera kommentarer (Ej möjligt ändra användarnamn i efterhand)
- Se senaste kommentar
- …? Mer funktionalitet om tid finnes.

Handlingsplan

Alla i gruppen ansvarar själva för att inhämta den kunskap som behövs för att kunna arbeta med ramverket och skriva enhetstester.

Gruppen skapar en Trelloboard där alla uppgifter finns listade. Varje gruppmedlem ansvarar sedan för att själv ta sig an ett antal uppgifter allt eftersom.

Varje gruppmedlem ansvarar för att skriva tester för sin egen kod.

Varje feature skapas i en ny branch. Resterande gruppmedlemmar ansvarar för att granska den kod som sedan committas och mergas branchen in i master.

Följande är vår övergripande handlingsplan:

- Skapa git-repo
- Sätta upp en Trelloboard
- Sätta upp arbetsmiljö
- Sätta oss in i ramverket
- Välja testverktyg
- Skapa applikation
    - Se Trelloboard https://trello.com/b/WTXMfpyK/javascript-3
- Skriva enhetstester till en coverage om minst 90%

Val av ramverk

Vue.js – motivering: samtliga gruppmedlemmar är intresserade av att prova på ramverket som är omtalat för tillfället och har fått bra kritik.

Alla gruppmedlemmar har tidigare jobbat med andra ramverk under LIA-perioden (Angular 2 samt React) och vill gärna prova på något nytt.

Vue.js har den funktionalitet som behövs för vår applikation och har bra dokumentation vilket känns som bra förutsättningar för att hinna genomföra projektet på ett bra vis.

Val av testramverk

Via Vue-cli fick vi med ett antal testverktyg ”på köpet”, vilka vi valde att använda oss av och som dessutom har fungerat bra. Dessa var:

- Karma: Test runner
- Mocha: Test framework
- Chai: Assertion library

Reflektion

Under projektet har arbetet flutit på bra och alla i gruppen har aktivt tagit sig an olika delar av uppgiften vilket har gjort att alla både har skrivit kod och tester.

Vi har följt vår ursprungliga handlingsplan till stor del, men valde att en bit in i arbetet också använda oss av VueX för hantering av state samt valde bort att använda oss av ikoner i applikationen på grund av (kända) svårigheter att få Vue och Font Awesome att fungera tillsammans (se https://github.com/Justineo/vue-awesome/issues/7).

Projektet har gjort att vi verkligen inser nyttan med enhetstester; det ger en trygghet som utvecklare att veta att applikationens olika delar fungerar som förväntat. Vi kan också se fördelen med en gediget enhetstestad applikation den dag någonting går sönder – enhetstesterna blir då en stor hjälp i buggletande, något vi märkte när vi införde VueX för statehantering halvvägs in i projektet.

Vi har verkligen haft stor nytta av coverage-rapporteringen, som gjorde att vi enkelt kunde se om vi missat att testa viss kod, och som också med sitt färgkodade UI sporrade oss att nå så hög testtäckning (det vill säga att allt skulle vara grönt!) som möjligt.

Utmaningen med att skriva enhetstester känner vi dels är att ”komma in i tänket” att skriva kod utifrån BDD- eller TDD-principen - något vi inte lyckades med, då vi inte konsekvent skrev våra tester innan vi skrev kod – och dels att skriva testbar kod!

Design

Fig 1. Startvy

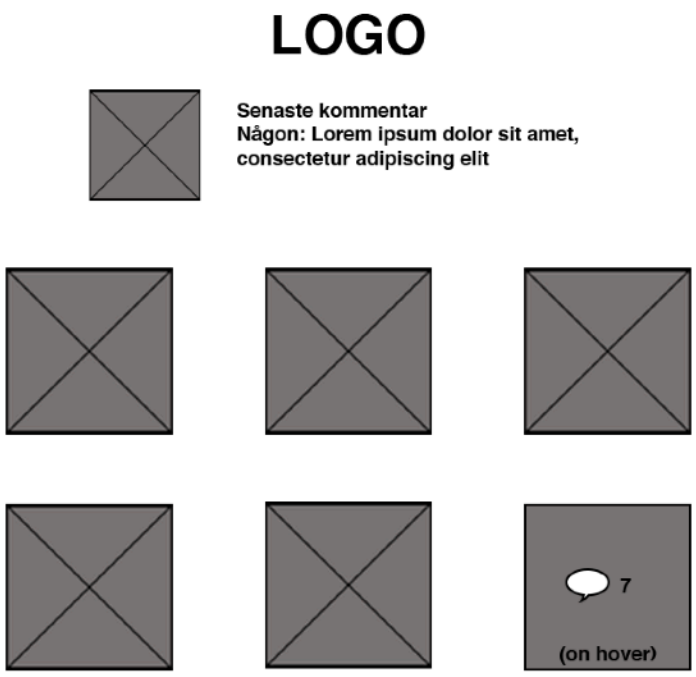

Fig 2. Modal