**Test Fall av Meddelandetjänst – Node med Rest API**

**Viktig information:** **Ännu finns inga krav på validering av skriven meddelandetext.**

**TF 1.1 – Huvudscenario; Användaren skriver ett meddelande.**

1. Användaren öppnar applikationens meddelandeservice så att meddelandefönstret är öppnat och skrivfältet ligger i fokus.
2. Användaren skriver in ett meddelande och klickar på knappen ”Submit Message”.
3. Meddelandemodulen presenterar meddelandet ovanför skrivfältet.

**Alternativa scenarios**

**2a.** Användaren struntar i att skriva ett meddelande och klickar på ”Submit Message”.

1. Meddelandemodulen filtrerar bort tomma meddelanden och inget nytt meddelande visas.

**2b.** Användare trycker på enter-tangenten efter meddelandet är skrivet.

**1.** Meddelandemodulen går vidare med steg 3.

**2c.** Användaren struntar i att skriva ett meddelande och trycker därefter på enter-tangenten.

1. Meddelandemodulen filtrerar bort tomma meddelanden och inget nytt meddelande visas.

**TF 1.2 – Huvudscenario; Användaren raderar ett meddelande.**

**Efterkrav:** Verifiera att meddelandet är borttaget.

1. Användaren öppnar applikationens meddelandeservice så att meddelandefönstret är öppnat.
2. Användaren väljer att radera något av meddelandena.
3. Meddelandemodulen tar bort meddelandet.

**TF 1.3 – Huvudscenario; Användaren redigerar ett meddelande.**

**Efterkrav:** Verifiera att den gamla kommentaren ersatts med den nya.

1. Användaren öppnar applikationens meddelandeservice så att meddelandefönstret är öppnat.
2. Meddelandemodulen visar en eller flera skrivna meddelanden (om inte, skapa meddelanden att redigera).
3. Användaren genomför valet att redigera något av meddelandena.
4. Meddelandemodulen presenterar meddelandet med en markör som indikerar att meddelandet är satt i redigeringsläge.
5. Användaren förändrar meddelandet och sparar förändringarna genom att klicka på knappen ”Submit Message”.
6. Meddelandemodulen sparar det nya meddelandet.

**Alternativa scenarios**

**5a.** Användaren förändrar meddelandet och sparar förändringarna genom att trycka på enter-tangenten.

1. Redigeringsmodulen genomför steg 6.
2. Användaren trycker på knappen ”Submit Message”.
3. Meddelandemodulen gör ingenting (ingen förändring sker).

**5b.** Användarenförändrar meddelandet och sparar förändringarna genom att avlägsna fokus från skrivfältet.

1. Redigeringsmodulen genomför steg 6.
2. Användaren trycker på knappen ”Submit Message”.
3. Meddelandemodulen gör ingenting (ingen förändring sker).

**5c.** Användaren raderar meddelandet och väljer därefter att spara ett ”tomt meddelande”.

1. Meddelandemodulen tar bort meddelandet.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Teststeg | Testfall | Genomförande | Förväntat resultat | Resultat |
| 1 | 1.1 | Huvudscenario | Meddelande presenteras | Meddelande presenteras |
| 2 | 1.1 | Alternativt scenario 2a | Tomt meddelande bortfiltrerat | Tomt meddelande bortfiltrerat |
| 3 | 1.1 | Alternativt scenario 2b | Meddelande presenteras | Meddelande presenteras |
| 4 | 1.1 | Alternativt scenario 2c | Tomt meddelande bortfiltrerat | Tomt meddelande bortfiltrerat |
| 5 | 1.2 | Huvudscenario | Meddelande borttaget | Meddelande borttaget |
| 6 | 1.3 | Huvudscenario | Uppdaterat meddelande sparat | Uppdaterat meddelande sparat |
| 7 | 1.3 | Alternativt scenario 5a | Uppdaterat meddelande presenteras efter 5a:1 men ingen förändring efter 5a:3 | Uppdaterat meddelande samt ytterligare ett meddelande som är tomt hat skapats. |
| 8 | 1.3 | Alternativt scenario 5b | Uppdaterat meddelande presenteras efter 5b:1 men ingen förändring efter 5b:3 | Uppdaterat meddelande presenteras efter 5b: 1 men ingen förändring eter 5b:3 |
| 9 | 1.3 | Alternativt scenario 5c | Meddelande borttaget | Meddelande borttaget. |

**Test Fall av Chat – Node med Socket.io**

**Viktig information: Meddelandetjänsten har nu utvecklats till en chat. Ännu finns inga krav på validering av skriven meddelandetext.**

**TF 2.1 – Huvudscenario; Användaren skriver ett meddelande.**

**Förkrav:** Användaren har öppnat upp 3 webbläsare (Firefox, Chrome och Safari) sida vid sida. I samtliga webbläsare har applikationens chat öppnats.

**Efterkrav:** Skrivet meddelande skall visas i samtliga öppna webbläsare/klienter direkt efter att meddelandet skickats till chatmodulen. Varje meddelande som är skrivet på den egna klienten skall visas med en röd kryss-knapp längst upp till höger inom meddelandets gränser. Alla andra meddelanden skall endast visa ett meddelande. Det skall inte gå att skriva in tomma meddelanden.

1. Användaren skriver meddelandet ”ff” i Firefox och klickar på knappen ”Submit Message”.
2. Chatmodulen presenterar meddelandet ovanför skrivfältet i alla tre webbläsare. Meddelandet ska ha en röd kryss-knapp (för radering av meddelandet) längst upp till höger inom meddelandets område i den webbläsare där meddelandet är skrivet.
3. Användaren trycker på en av webbläsarnas omladdningskrapp för att ladda om hela webbapplikationen, alternativt kopierar webbapplikationens url för att klistra in den i en ny webbläsarflik.
4. Chatmodulen laddas om på nytt och alla hittills skrivna meddelanden presenteras. Inget av meddelandena har en röd kryssruta, varje meddelande innehåller endast en skriven text.
5. Användaren skriver ett meddelande och trycker därefter på enter-tangenten.
6. Chatmodulen presenterar meddelandet enligt steg 2.

*Användaren upprepar steg 1 – 2 med undantaget att användaren skriver meddelandet ”saf” i Safari och ”cr” i Chrome.*

**Alternativa scenarios**

**1a.** Användaren struntar i att skriva ett meddelande och klickar på ”Submit Message”.

1. Chatmodulen filtrerar bort tomma meddelanden och inget nytt meddelande visas i någon av webbläsarna.

**1b.** Användare trycker på enter-tangenten efter meddelandet är skrivet.

**1.** Chatmodulen går vidare med steg 2.

**1c.** Användaren struntar i att skriva ett meddelande och trycker därefter på enter-tangenten.

1. Meddelandemodulen filtrerar bort tomma meddelanden och inget nytt meddelande visas i någon av webbläsarna.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Teststeg | Testfall | Genomförande | Förväntat resultat | Resultat |
| 1 | 2.1 | Huvudscenario1 - 3 | Meddelandet presenteras i samtliga webbläsare och med röd kryss-knapp i den webbläsare/klient där meddelandet är skrivet. | Meddelande presenteras i samtliga webbläsare och med röd kryss-knapp i den webbläsare/klient där meddelandet är skrivet. |
| 2 | 2.1 | Alternativt scenario 1a | Tomt meddelande bortfiltrerat | Tomt meddelande bortfiltrerat |
| 3 | 2.1 | Alternativt scenario 1b | Meddelandet presenteras i samtliga webbläsare och med röd kryss-knapp i den webbläsare/klient där meddelandet är skrivet. | Meddelande presenteras i samtliga webbläsare och med röd kryss-knapp i den webbläsare/klient där meddelandet är skrivet. |
| 4 | 2.1 | Alternativt scenario 1c | Tomt meddelande bortfiltrerat | Chatmodulen presenterar ett tomt meddelande. |
| 5 | 2.1 | Huvudscenario 4 | Applikationen laddas om Chatmodulen visas med alla skrivna meddelanden. Inget av meddelandena har en röd kryss-ruta. | Applikationen laddas om Chatmodulen visas med alla skrivna meddelanden. Inget av meddelandena har en röd kryss-ruta. |
| 6 | 2.1 | Huvudscenario 6 | Meddelandet presenteras i samtliga webbläsare och med röd kryss-knapp i den webbläsare/klient där meddelandet är skrivet. | Meddelandet presenteras i samtliga webbläsare och med röd kryss-knapp i den webbläsare/klient där meddelandet är skrivet. |
| 7 | 2.1 | Huvudscenario | Meddelande borttaget | Meddelande borttaget |

**TF 2.2 – Huvudscenario; Flera klientanvändare raderar meddelanden i Chatmodulen.**

**Viktig Information:** I testfallet kommer följande förkortningar att användas.

WL1: Webbläsare 1

WL2: Webbläsare 2

WL3: Webbläsare 3

**Ytterligare exempel på förkortningar:**

**M1:1** Första meddelandet skrivet i webbläsare 1.

**M2:3** Tredje meddelandet skrivet i webbläsare 2.

**M3:2** Andra meddelandet skrivet i webbläsare 3.

**Förkrav:** Användaren har öppnat upp 3 webbläsare sida vid sida. I samtliga webbläsare har applikationens chat öppnats.

**Efterkrav:**

* Skrivna meddelanden syns i alla webbläsare.
* Radering av meddelande i en webbläsare sker ”samtidigt” i alla webbläsare.
* Om webbläsare laddas om skall möjligheten att radera egna meddelanden skrivna **före omladdningen** upphöra.
* Har man laddat om webbläsaren skall man kunna radera alla egna meddelanden som är skrivna **efter omladdningen**.

1. Användaren skriver M1:1 i WL1.
2. Chatmodulen presenterar M1:1 i samtliga WL.
3. Användaren skriver M2:1 i WL 2.
4. Chatmodulen presenterar M2:1 i samtliga WL.
5. Användaren skriver M3:1 i WL 3.
6. Chatmodulen presenterar M3:1 i samtliga WL.
7. Användaren skriver M2:2 i WL2.
8. Chatmodulen presenterar M2:2 i samtliga WL.
9. Användaren skriver M1:2 i WL1.
10. Chatmodulen presenterar M1:2 i samtliga WL.
11. Användaren skriver M1:3 i WL1.
12. Chatmodulen presenterar M1:3 i samtliga WL
13. Användaren raderar M1:3 i WL1.
14. Chatmodulen raderar M1:3 från samtliga WL.
15. Användaren raderar M1:1 i WL1.
16. Chatmodulen raderar M1:1 från samtliga WL.
17. Användaren trycker på en av webbläsarnas omladdningskrapp för att ladda om hela webbapplikationen, alternativt kopierar webbapplikationens url (rooturl/chat) för att klistra in den i en ny webbläsarflik.
18. Chatmodulen laddas om på nytt och alla hittills skrivna meddelanden presenteras. Inget av meddelandena har en röd kryssruta, varje meddelande innehåller endast en skriven text.
19. Användaren raderar M2:2 från WL2.
20. Chatmodulen raderar M2:2 från samtliga WL.
21. Användaren skriver M3:3 i WL3.
22. Chatmodulen presenterar M3:3 i samtliga WL.
23. Användaren skriver M1:4 i WL1.
24. Chatmodulen presenterar M1:4 i samtliga WL.
25. Användaren raderar M3:3 från WL3.
26. Chatmodulen raderar M3:3 från samtliga WL.
27. Användaren raderar M2:1 från WL2.
28. Chatmodulen raderar M2:1 från samtliga WL.

**Alternativa Scenarios**

**19 a.** Direkt efter omladdning av WL3 skriver användaren M3:3 i WL3.

1. Chatmodulen presenterar M3:3 i samtliga WL.
2. Användaren skriver M2:3 i WL2.
3. Chatmodulen presenterar M2:3 i samtliga WL.
4. Användaren raderar M2:3 från WL2.
5. Chatmodulen raderar M2:3 från samtliga WL.
6. Användaren raderar M3:3 från WL3.
7. Chatmodulen raderar M3:3 från samtliga WL.
8. Användaren skriver M1:4 i WL1.
9. Chatmodulen presenterar M1:4 i samtliga WL.
10. Användaren raderar M1:4 från WL1.
11. Chatmodulen raderar M1:4 från samtliga WL.

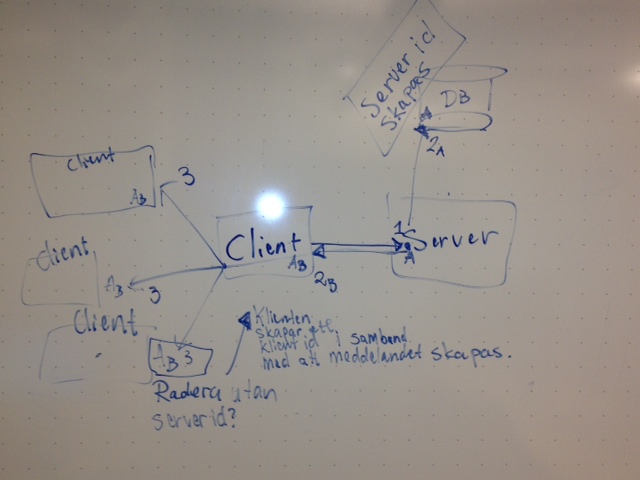
**Testutslag av TF 2.2**

I tabellen nedan redovisas för enkelhetens skull endast de testförsök som inte gav ett förväntat utslag. Alla förväntade resultat framgår tydligt av testfall 2:2 ovan.

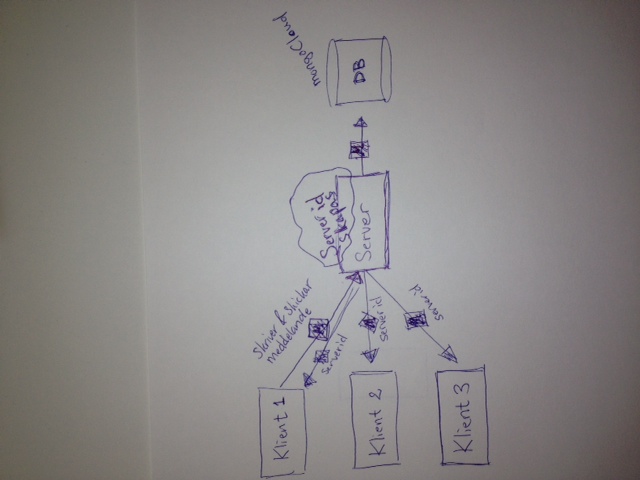
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Teststeg | Testfall | Genomförande | Förväntat resultat | Resultat |
| 15 | 2.2 | Huvudscenario | Chatmodulen raderar M1:1 från samtliga WL. | Inget händer. Console rapporterar this.collection is undefined. Message.js line 34. Sker ca var sjätte gång testfall 2.2 genomförs. |
| 20 | 2.2 | Huvudscenario | Chatmodulen raderar M2:2 från samtliga WL. | M2:2 raderas från WL1 och WL2 men ej från WL3. |
| 26 | 2.2 | Huvudscenario | Chatmodulen raderar M3:3 från samtliga WL. | M3:3 raderas från WL3 men ej från WL1 och WL2. |
| 28 | 2.2 | Huvudscenario | Chatmodulen raderar M2:1 från samtliga WL | M2:1 raderas från WL1 och WL2 men ej från WL3 |
| 5 | 2.2 | Alternativt scenario 19 a | Chatmodulen raderar M2:3 från samtliga WL | M2:3 raderas från WL1 och WL2 men ej från WL3 |
| 7 | 2.2 | Alternativt scenario 19 a | Chatmodulen raderar M3:3 från samtliga WL | M3:3 raderas från samtliga WL |
| 11 | 2.2 | Alternativt scenario 19 a | Chatmodulen raderar M1:4 från samtliga WL | M1:4 raderas från WL1 och WL2 men ej WL3 |

**Analys - testresultat TF 2.2**

Jag tror kärnan till problemet ligger i att jag använt Backbone-modellens cid (client id) för att identifiera vilket meddelande som skall tas bort. Ett meddelande har ingen server id förrän meddelandet sparats i databasen och sedan hämtats igen. Problemet är att jag uppfattat det som att man via socket ska skicka meddelande-objektet i json format till server.js som visserligen sparar till databasen men som samtidigt sänder tillbaka meddelandet till klienten som en broadcast så att alla meddelanden når fram till samtliga klienter. När ett broadcastat meddelande sedan skall raderas krävs det ett id för att hitta just det meddelandet i databasen. Det id som jag då använder mig av för att hämta ut ett meddelande från databasen är ett client-id som Backbone modellen skapat på klientsidan. Problemet är att varje klient skapar egna client-id till sina meddelande-objekt ovetande om varandra, vilket kan medföra dubbletter av id. När jag sedan talar om för databasen att radera ett meddelande med ett client-id istället för ett server id (som jag inte har tillgång till) så är risken stor att det sker någon form av konflikt. Därav buggen som upptäcktes i försök 15.

Övriga buggar bör bero på ett logiskt fel i koden. Troligtvis i min MessagesView som är en Backbone CollectionView. Felet sker där ett borttaget meddelande broadcastas för att även ta bort meddelandena på övriga klienter.

**Uppföljning av analys - testresultat TF 2.2**

Denna uppföljning innebär att jag efter buggfix för buggar som upptäcktes under detta TF, åter igen genomfört TF 2.2. Genomförandet testar även implementation för en ny funktionalitet som meddelar alla i chatten när en ny person anslutit till chatten. TF 2.2 är nu felfritt genomfört i min lokala version. I korta drag åtgärdades buggen genom att jag går händelserna i förväg och manuellt sätter ett mongoose ObjecktId på meddelandeobjektet när det nått fram till servern men innan meddelandeobjektet skickas till databasen. Meddelandet får då ett unikt server-id som följer med till databasen men också tillbaka till klienten/klienterna. Ännu ett fel hittades på klientsidan som innebar att flera Backbone – meddelandemodeller delade på samma id, vilket givetvis orsakade stora problem. Även det felet är åtgärdat.

**Test Fall av kontaktformulär – Klientvalidering och respons.**

**Viktig information:**

**Stäng ej av javascript i webbläsaren, detta test avser endast validering på klientsidan. Krav för korrekt data:**

**Förnamn (First Name):** Endast bokstäver och får ej vara tomt.

**Efternamn (Surname):** Endast bokstäver och får ej vara tomt.

**Epost (Email):** Godkänner exempelvis formatet [namn@epost.se](mailto:namn@epost.se) och får ej vara tomt.

**Webbsida (Web Page):** Fältet får ej vara tomt.

**Alternativt scenario 1a** har testats och ska testas med utgångspukt ifrån samtliga fält i formuläret, för enkelhetens skull har jag uttryckligen skrivit detta scenario för första fältet.

**TF 3.1 – Huvudscenario; Användaren fyller i ett formulär, klickar på ”Submit Form”. Användaren skickas till applikationens start-sida och visar ett rättmeddelande som bekräftar att formuläret är skickat.**

**Förkrav: Användaren har tagit sig till applikationens kontaktformulär.**

**Efterkrav:** Kontaktformuläret har skickats och användaren får ett rättmeddelande som bekräftar att ifyllt formulär är skickat. Rättmeddelandet går att kryssa bort.

1. Användaren skriver in ett förnamn och ställer markören i nästa fält.
2. Formuläret ger ingen respons då enskilt fält är korrekt ifyllt.
3. Användaren skriver in ett efternamn och ställer markören i nästa fält.
4. Formuläret ger ingen respons då enskilt fält är korrekt ifyllt.
5. Användaren skriver in en epost-adress och ställer markören i nästa fält.
6. Formuläret ger ingen respons då enskilt fält är korrekt ifyllt.
7. Användaren skriver en webb-adress och lämnar fältet.
8. Formuläret ger ingen respons då enskilt fält är korrekt ifyllt.
9. Användaren väljer att skicka formuläret.
10. Applikationen skickar användaren till start-sidan och visar ett rättmeddelande som bekräftar att ifyllt formulär är skickat.
11. Användaren läser rättmeddelandet och kryssar bort det.
12. Applikationen avlägsnar rättmeddelandet.

**Alternativa scenarios**

1. Under en godtycklig tidpunkt då formuläret fylls i, väljer användaren att skicka formuläret.
2. Formuläret upptäcker att alla fält ej är ifyllda eller har korrekt angiven data.

1a. Formuläret visar ett felmeddelande under varje fält som ej klarat validering.

1a. Användaren rättar alla felmeddelanden.

1. Formuläret avlägsnar ett felmeddelande för varje gång användaren rättar ett fält.
2. Användaren går vidare med steg 7 i huvudscenariot.

1b. Användaren väljer att rätta två felmeddelanden.

1. Formuläret avlägsnar ett felmeddelande för varje gång användaren rättar ett fält.
2. Användaren går vidare med steg 7 i huvudscenariot.
3. Formuläret visar ett felmeddelande under varje fält som fortfarande ej klarar validering.
4. Användaren går vidare med alternativt scenario **a** – 1 – 1a – 1a.
5. Formuläret upptäcker att alla fält är korrekt ifyllda och går vidare med steg 7 i huvudscenariot.

**1a**. Användaren lämnar fältet tomt eller skriver in felaktig data.

1. Formuläret visar ett felmeddelande under fältet.

1a. Användaren rättar felmeddelandet.

1. Formuläret avlägsnar felmeddelandet.
2. Användaren går vidare med steg 2 i huvudscenariot.

1b. Användaren går vidare med steg 2 – 9 i huvudscenariot förutom att aktuellt fält fortsättningsvis lämnas tomt eller med felaktig inskriven data.

1. Formuläret visar fortfarande felmeddelandet under fältet.
2. Användaren väljer att skicka formuläret.
3. Formuläret visar fortfarande felmeddelandet under fältet.
4. Användaren går vidare med alternativt scenario 1a – 1 – 1a.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Teststeg | Testfall | Genomförande | Förväntat resultat | Resultat |
| 2, 4, 6, 8 | 3.1 | Huvudscenario | Ingen respons | Ingen respons |
| 10 | 3.1 | Huvudscenario | Användaren befinner sig på startsidan och rättmeddelande visas. | Användaren befinner sig på startsidan och rättmeddelande visas. |
| 12 | 3.1 | Huvudscenario | Applikationen avlägsnar rättmeddelandet. | Applikationen avlägsnar rättmeddelandet i samtliga webbläsare i datorn för både Windows och mac OS X men **inte för safari i iphone och ipad**. |
| a – 1 – 1a | 3.1 | Alternativt scenario | Formuläret visar ett felmeddelande under varje fält som ej klarat validering. | Formuläret visar ett felmeddelande under varje fält som ej klarat validering. |
| a – 1 – 1a – 1a - 1 | 3.1 | Alternativt scenario | Formuläret avlägsnar ett felmeddelande för varje gång användaren rättar ett fält. | Formuläret avlägsnar ett felmeddelande för varje gång användaren rättar ett fält. |
| a – 1 – 1a – 1b - 1 | **3.1** | Alternativt scenario | Formuläret avlägsnar ett felmeddelande för varje gång användaren rättar ett fält. | Formuläret avlägsnar ett felmeddelande för varje gång användaren rättar ett fält. |
| a – 1 – 1a – 1b - 3 | **3.1** | Alternativt scenario | Formuläret visar ett felmeddelande under varje fält som fortfarande ej klarar validering. | Formuläret visar ett felmeddelande under varje fält som fortfarande ej klarar validering. |
| 1a - 1 | **3.1** | Alternativt scenario | Formuläret visar ett felmeddelande under fältet. | Formuläret visar ett felmeddelande under fältet. |
| 1a – 1 – 1a - 1 | **3.1** | Alternativt scenario | Formuläret avlägsnar felmeddelandet. | Formuläret avlägsnar felmeddelandet. |
| 1a – 1 – 1b - 1 | **3.1** | Alternativt scenario | Formuläret visar fortfarande felmeddelandet under fältet. | Formuläret visar fortfarande felmeddelandet under fältet. |
| 1a – 1 – 1b - 3 | **3.1** | Alternativt scenario | Formuläret visar fortfarande felmeddelandet under fältet. | Förväntat resultat förutom när testet genomförs med utgångspunkt från det sista fältet i formuläret (Web Page). |
|  |  |  |  |  |