IP2 - Redogörelse

Beskrivning

Målet med projektet var att skapa en portföljsida med hjälp av React. React är ett Javascript bibliotek skapat av Facebook. React är en Single Page Application, SPA, vilket innebär att det bara finns ett HTML dokument. Istället skapas komponenter som sedan kan renderas när användaren interagerar med hemsidan. Huvudkomponenten för sidan är APP.js som är kopplat till HTML dokumentet. Projektet består av flertal komponenter med olika information. För att rendera dessa visuellt på sidan när användaren interagerar med applikationen används React Router. För stylingen av sidan har SASS används. Fördelen med SASS va att Variabler samt Mixins kunde återanvändas i flertalet av komponenterna till sidan. På så sätt minska antal rader kod. Nackdelen med detta är dock att det kan vara svårare att återanvända vissa komponenter i andra projekt. Därför har navbaren placerats i en egen separat mapp med tillhörande CSS kod för att kunna återanvändas lättare.

För de två komponenter med mest information samt information som kommer behöva uppdateras genom åren har objekt med informationen skapats. Med hjälp av metoden Map() skapas det automatiskt en ny div med information som läggs till i objektet. Tanken är även att skapa en del av sidan där jag kan skriva in på sidan de nya erfarenheter mm som sedan läggs i objektet (nästa version av sidan).

Sidan har responsiv design och ändrar utseende och form för mindre skärmar. Stylingen av sidan ska likna ett “business card”.

API:et som används är väderdata för stockholm. URL:en är tagen från hackernews som presenterades på en tidigare föreläsning. Koden är omskriven och datan som var relevant för sidan användes. AJAX användes för detta (se mer om AJAX under rubriken Målen).

Nästa version
Till nästa version är tanken att förnya komponenterna och använda hooks för att fördela informationen lättare. Iden är att ha en separat fil med information till de olika komponenterna så all info kan uppdateras på ett ställe. Även navbaren ska förbättras där en hamburgare meny kommer visas vid mindre skärmar.

React
Ett Javascript bibliotek skapat av Facebook. React använder sig av komponenter som kan länkas ihop för att lättare kunna dela upp projekt i mindre byggstenar samt återanvända dessa byggstenar. Detta är inte unikt för React. Den stora fördelen med React är dess virtuella DOM. Alla webbläsare arbetar med DOM - dokumentobjektmodell. DOM kan beskrivas som en hierarkisk datamodell för en sida som visas i webbläsaren. När en användare interagerar med en hemsida görs det förändringar till DOM. När förändringar görs till DOM måste webbläsaren rita om delar av sidan för att ladda in ny data. Beroende på hur mycket data som måste ändra påverkas laddningstid olika mycket. Detta kan ha en negativ effekt på användarupplevelsen. React har därför något de kallar för en virtuell DOM. Virtuella

DOM är en kopia av den riktiga DOM och med hjälp av den kan flera ändringar göras innan förändringar måste göras till riktiga DOM. På så sätt minskar laddningstiderna på sidan.

Create React App
En React application kan skapas manuellt men för att underlätta processen har facebook skapat en skräddarsydd uppsättning där de nödvändiga biblioteken samt en grundstruktur med kod. Genom ett kommando i terminalen skapas en grundstruktur som sen kan byggas på.

Node.js
Node js är en plattform som tillåter oss att köra Javascript på en dator / server. Med Node js behövs inget annat programmeringsspråk för att bygga servern för hemsidan. Det är en väldigt snabb plattform som körs på V8 motorn och använder sig av non blocking code (koden körs i sekvens). Med ett stort system av open source material som kan installeras med npm.

SASS - Syntactically Awesome Style Sheet / LESS
SASS och LESS är två varianter av tillägg till vanlig CSS. Det skapar möjlighet att strukturera koden bättre. Några fördelar med SASS och LESS är att det ger möjligheten till nästlade taggar. Andra fördelar är att det går att skapa variabler och återanvändbara funktioner (mixins). SASS och LESS är väldigt lika där SASS är föregångaren till LESS där skillnaden är att LESS är bakåtkompabilitet mot vanlig CSS. Ser man på statistiken så är SASS mer populärt för tillfället (2020-01). Därav har jag valt att använda det i detta projekt.

React Router

Vanligtvis är en hemsida uppbyggd av flera olika HTML-filer. React är en single page application, SPA. Det innebär att det bara består utav en HTML-fil.

Vad React Router gör är att den renderar olika komponenter utan att behöva kommunicera med servern. Vad användaren egentligen ser är en komponent som t.ex. /Home. Om användaren sen klickar vidare till t.ex. /About kommer React Router stoppa förfrågan till servern och istället ladda komponenten /About. På så sätt kan man undvika onödiga förfrågningar till servern som kan orsaka laddningstider på sidan.

Målen

HTML&CSS, 1.5 Olika CSS-ramverk:

HC1.5.1. Det är lätt att tro att CSS-ramverk börjar och slutar med Bootstrap, men så är ej fallet. Ge minst 3 exempel på css-ramverk, där minst 1 ska använda sig av Material Design.
Referenser:
https://hackernoon.com/top-5-most-popular-css-frameworks-that-you-should-pay-attention-to-in-2017-344a8b67fba1
https://www.sitepoint.com/free-material-design-css-frameworks-compared/

Materialize

Bygger på Googles Material Design. Responsiv design med inriktning på UX. Fördelar med Materialize är att det har många forks på gitHub och är under kontinuerlig utveckling. Dokumentationen är väldigt bra. Bra för att komma igång med Material Design.

Surface

Ett mindre CSS ramverk som bygger på Googles Material Design. Passar bra till att få ut en snabb prototyp.

Bulma

Bulma är ett CSS ramverket som baseras på FlexBox. Det är lätt att använda och ger dig möjligheten att enbart importera de styling komponenter du vill använda.

HC1.5.2. Välj ut 2 av dessa ramverk och berätta lite vad deras kärnprinciper/styrkor är. Vad är poängen med att använda just dessa ramverk och varför bör man använda dem?

Bulma

Bulma har SASS och är därför lätt att importera de funktioner du behöver. De använder sig av enkla och förståeliga class namn och modifiers. Du kan alltså lägga på modifiers till din class för att addera styling. Då Bulma använder sig av FlexBox är det lätt att positionera objekt på sidan och ger sidan en responsiv design.

Materialize

Material Design skapades av google för att förbättra UX. Enkelt förklarat bygger det på att använda sig av djup och animationer för att leda användaren i rätt riktning. Materialize är byggt runt Googles Material Design. Det är ett bra ramverk för den som vill lära sig använda Material Design på nätet på ett enkelt sätt. De har bra dokumentation på hemsidan som gör det lätt att lära sig. Det är väldigt likt Bootstrap men är ett nyare bibliotek vilket kommer med bättre animeringar. Materialize använder jQuery vilket kan vara en nackdel.

HTML&CSS, 1.6 SASS eller LESS:

HC1.6.1. SASS och LESS har vi använt en del. Vad är poängen med att använda dem? Ge gärna några konkreta exempel.
Referens: https://www.keycdn.com/blog/sass-vs-less/

SASS - Syntactically Awesome Style Sheet / LESS
SASS och LESS är två varianter av tillägg till vanlig CSS. Det skapar möjlighet att strukturera koden bättre. Några fördelar med SASS och LESS är att det ger möjligheten till nästlade taggar. Andra fördelar är att det går att skapa variabler och återanvändbara funktioner (mixins). SASS och LESS är väldigt lika där SASS är föregångaren till LESS där skillnaden är att LESS är bakåtkompabilitet mot vanlig CSS. Ser man på statistiken så är SASS mer populärt för tillfället (2020-01). Därav har jag valt att använda det i detta projekt.

Javascript, 1.1 Javascript, grundläggande syntax:

JS1.1.1. Grundläggande syntax bör ni behärska. Ta nån kodsnutt och beskriv vad den gör.

```
var text = "";
var i;
for (i = 0; i < 5; i++) {
  text += "The number is " + i + "<br>";
}
document.getElementById("demo").innerHTML = text;
```

Exemplet ovan visar en for loop. Två variabler definieras i början (text = tom sträng och i som definieras men ges inge värde). I for-loopen definieras först utgångs uttryck (i detta fall variabeln tilldelas värdet 0); villkor (i detta fall när i är mindre än 5); utökning uttryck (i detta fall i++ vilket innebär att i ökar med ett fram till villkoret mötts)). Under parentesen presenteras koden som loopen ska köra. I detta exempel sätts tidigare variabeln text till strängen “The number is “ plus värdet av i samt lägger till en radbrytning. Så loopen kommer köras 5 gånger och ge värdena “The number is 0”, “The number is 1” osv fram till “The number is 5” sen kommer loopen avslutas och nästa kodrad kommer köras. Där definieras i detta fall att resultatet av loopen ska skrivas ut i DOM.

Javascript, 1.2 Javascriptsversioner och dess kompatibilitet:

JS1.2.1. Nämn de tre senaste Javascriptsversionerna och dess kompatibilitet.
Referens: https://www.w3schools.com/js/js_versions.asp

Javascript presenterade sin första version 1997 ECMAScript 1 och 1998 lades de första ändringarna till. 1999 lanserades ändringar i ECMAScript 3 där regular expression och try/catch introducerades. 2009 började man jobba efter att kunna köra Javascript utanför webbläsaren genom att använda moduler för att paketera användbar kod och funktionalitet. Vad som kom ut av det var Node.js. Nu kunde man använda Javascript för att sköta servern.

Samma år lanserades JS ES5 som kom packad med nya funktionaliteter bland annat JSON, String.trim(), Array.isArray(), Array iteration Methods.
Slutligen 2015 lanserades JS ES6. Med ES6 lanserades const och let, nya array metoder. fram till 2018 (ES9) har en rad nya metoder och funktioner adderats till Javascript som arrow functions, async functions mm. Senaste uppdateringen 2019 (ES10) lanserade Array.flat, array.flatMap mm.

Det va lite onödig men intressant historia. Nu till kompabiliteten.

ES5 2009 var från 2013 kompatibel med:
Chrome
Firefox
IE
IE / Edge
Safari
Opera

ES6 2015 var kompatibel med:
Chrome
Firefox
Edge
Safari
Opera

ES7
Chrome
Opera

Javascript, 1.3 Hur Javascript körs i webbläsaren:

JS1.3.1. Hur körs Javascript i browsern?
Referens:
https://blog.sessionstack.com/how-does-javascript-actually-work-part-1-b0bacc073cf

Webbläsaren har tre stycken huvudprogram som är intressanta för att kunna läsa HTML, CSS och Javascript första programmet är DOM översättaren det tar ditt HTML-dokument och konverterar det och visar det i webbläsaren. Det andra programmet är en CSS översättare som tar CSS koden och stylar sidan. Sist men inte minst har vi Javascript Engine. Javascript motorn har olika namn i olika webbläsare men har samma funktion. Vad motorn gör är att ta koden och gör om den till binär kod. Dessa tre program kallas för JIT (Just In Time) Compilers. Låt oss fokusera på Javascript motorn. För att förstå den bättre måste vi först förstå vad ett program är. Enkelt beskrivet måste ett program bestå av ledigt minne och en del som analyserar och verkställer. Motorn består egentligen av två delar Memory heap och call stack. Varje gång du deklarerar en variabel eller en funktion använder du dig utav minnet i Memory heap. Som med allt minne så kommer det i begränsad form. Därför är det viktigt att tänka på att inte deklarera för många onödiga globala variabler som tar upp plats i minnet. När vi kör våran kod analyseras den av call stacken. Vad den gör är

att den går från toppen av koden och börjar analysera den. Så t.ex. ser vi att den en console.log(“Hello”) kommer den lägga den i call stacken och verkställa koden så fort den har verkställs tas den bort ur call stacken och nästa block körs. Detta är varför Javascript kallas för “a single threaded language”. Alltså Javascript har bara en call stack som bara kan göra en sak åt gången i den ordningen det presenteras.
Webbläsaren innehåller även något som kallas Javascript runtime environment som består av bland annat Web API’s (DOM, AJAX, TimeOut), Callback queue och en event loop. Om vi vill ändra ordningen i call stacken kan vi använda oss av web API:erna och flytta kod från call stacken till callback queue och placera tillbaka i call stacken när den är redo för att köras. Event loopen körs kontinuerligt och kollar call stacken efter kod att köra. Finns det ingen kod i call stacken kollar den callback queue om det finns något att köra. Finns det något i callback queue placerar det i call stacken.

Javascript, 1.4 Javascript-bibliotek:

JS1.4.1. Ge några exempel på JS-bibliotek. Det här fixar ni, ta något från era egna projekt.

Javascript, 1.5 Javascript-ramverk:

JS1.5.1. Ge några exempel på Javascript-ramverk. För-, nackdelar.
Referens: https://hackernoon.com/5-best-javascript-frameworks-in-2017-7a63b3870282 , (https://skillcrush.com/2019/05/22/javascript-frameworks-vs-libraries/)

Javascript bibliotek och ramverk är något som ofta blandas ihop. Så för att göra det lite tydligare så kan vi definiera skillnaden först innan vi listar några av vardera.
Ett JS bibliotek skulle kunna ses som olika möbler som adderar stil och funktionalitet till ett redan färdigt hus. Ramverket är själva ritningen/mallen du använder för att bygga huset. Ramverken ger dig en grundstruktur att bygga dina projekt i från. Fördelen med detta är att du får välstrukturerad kod som hjälper dig undvika vanliga kod problem. Nackdelen är att du får begränsad frihet i hur du skriver din kod. Medans biblioteken ger dig möjligheten att implementera funktionalitet till din redan existerande kod.

JS bibliotek
JQuarey
React
Quill

Ramverk

Agular
Vue
Ember JS

Javascript, 1.6 DOM:

JS1.6.1. DOM:en har ni stenkoll på! :) Förklara, vad är DOMen?

Alla webbläsare arbetar med DOM - Document Object Model. DOM kan beskrivas som en hierarkisk datamodell för en sida som visas i webbläsaren. När en användare interagerar med en hemsida görs det förändringar till DOM. När förändringar görs till DOM måste webbläsaren rita om delar av sidan för att ladda in ny data. Beroende på hur mycket data som måste ändra påverkas laddningstid olika mycket.

Referens: https://www.digitalocean.com/community/tutorials/introduction-to-the-dom
JS1.6.2. Vad är en Virtual DOM? Ge exempel. För-, nackdelar?
Referens:
https://www.quora.com/What-are-the-differences-pros-and-cons-between-Incremental-DOM-and-Virtual-DOM

Virtuella DOM är en kopia av den riktiga DOM och med hjälp av den kan flera ändringar göras innan förändringar måste göras till riktiga DOM. En av fördelarna med en virtuell DOM är att kommunikationen till servern minskar men vi kan fortfarande ändra innehållet på sidan. Istället för att behöva uppdatera hela sidan varje gång en användare interagerar med sidan kan vi nu istället enbart uppdatera berörd komponent och på så sätt minskar laddningstid.

Nackdelarna med virtuell DOM är bland annat att det kräver ett annat typ av tänkande. Något som skiljer sig från att jobba med vanliga DOM. Det är också relativt nyare vilket göra att det finns mindre information i förhållande till vanliga DOM. Ytterligare en nackdel skulle kunna vara att virtuella DOM håller koll på ändringar som görs som sparas i minnet vilket kan leda till prestandaproblem.

Referens: http://reactkungfu.com/2015/10/the-difference-between-virtual-dom-and-dom/

Javascript, 1.7 AJAX:

JS1.7.1. Vad är AJAX?
JS1.7.2. Vad används det till? För-, och nackdelar?
Referens: https://www.seguetech.com/ajax-technology/
w3schools.com/xml/ajax_intro.asp
https://dzone.com/articles/pros-and-cons-of-ajax

AJAX är en webbutveckling teknik som används för att skicka och ta emot data i bakgrunden på en hemsida utan att behöva refresha sidan. AJAX är alltså inte ett programmeringsspråk utan använder sig utav en XMLHttpRequest objekt som är inbyggt i webbläsaren. Vilket skickar förfrågningar av data till andra webbservrar. Och Sedan använda sig utav Javascript och DOM för att presentera eller använda data på en webbsida.

Med AJAX kan utbyte av data med andra webbservrar ske i bakgrunden asynkront och på så vis uppdatera delar av en webbsida utan att behöva ladda om hela sidan. Vilket gör sidorna snabbare och ger en bättre användarupplevelse. Nackdelar med AJAX är det är beroende av Javascript som implementeras olika i olika webbläsare. Vilket gör att AJAX är inte det bästa alternativet att använda till t.ex. mobila applikationer. Sidan kan bli svår att felsöka och kan också göra webbsidan mer öppen för säkerhetshot.

Javascript, 1.8 HTTP1.1 / 2.0:

JS1.8.1. Vad är skillnaderna mellan HTTP 1.1 och 2.0 och varför är 2.0 bättre?
Referens: https://kinsta.com/learn/what-is-http2/

HTTP kan förklaras lättast genom att titta på vad det är baserat på vilket är Client/Server modell. Det är i grunden två datorer, klientens dator som tar emot data och en server som delar med sig av data. De gör detta genom att skicka förfrågningar och får svar.

Den största skillnaden mellan HTTP 1.1 och 2.0 är att i nya versionen kan flera förfrågningar skickas parallellt över en enda TCP uppkoppling. Till skillnad från tidigare version när varje förfrågan var tvungen att skickas separat. Vad den nya versionen tillåter är alltså att ladda ner webbfiler asynkront från en server. Vilket i slutändad gör att webbsidan kan ladda snabbare. Nya versionen kan även ändra om text protokoll till binära protokoll. Genom att ändra om text kommandon till binär kod innan det skickas över nätverket undviker man kommandon som innehåller text och valfria mellanslag mm. En annan ny funktion är server push. Det kan beskrivas i enkla drag att servern kan skicka ytterligare cache information till klienten som förväntas i framtida förfrågningar. Man kan beskriva det som att om du skulle beställa mat på en restaurang och du ber servitören om en varmrätt. Servitören misstänker att du kanske vill ha vatten till maten och tar med det tillbaka. Istället för som med gamla versionen behöva be om det när hen kommer tillbaka med maten och ett ny förfrågan måste göras. Du kan alltså minska förfrågningarna till servern och på så sätt minska väntetiden.