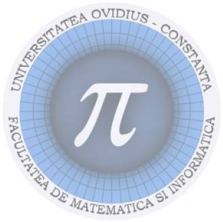
Universitatea Ovidius



Facultatea de Matematică și Informatică

Lucrare de licență

Dezvoltare aplicație web SPA bazată pe Python/Django și AngularJS

Student:
Ștefan Daniel Mihăilă

Profesor îndrumător:
Lect. dr. Andrei Rusu

iulie 2015

Cuprins

Internetul este un amalgam de tehnologii, legate împreună cu bandă adezivă, sfoară și gumă de mestecat. Nu este ceva proiectat într-un mod elegant, pentru că este un organism în creştere, nu o maşinărie construită cu intenţie.

Mattias Petter Johansson (Programator la Spotify)

0 Introducere

Internetul a evoluat continuu și a ajuns în punctul în care poate face o mulțime de lucruri pentru care nici măcar nu a fost creat. Aproape toți programatorii din ziua de azi sunt programatori web, iar aplicațiile web seamănă tot mai mult cu aplicațiile desktop. În aceste condiții, a devenit foarte important pentru dezvoltatori să poată crea astfel de aplicații într-un mod rapid și eficient, iar uneltele pe care le au la dispoziție au fost reînnoite permanet cu altele mai bune.

Arhitectura web clasică este una client-server, în care clientul (browserul) cere o pagină folosind protocolul HTTP, serverul o crează dinamic folosind un limbaj de programare server-side (C#, Java, Python, PHP, Scala etc.) și o trimite browserului pentru afișare. Prin HTTP, conexiunile sunt întotdeauna inițiate de către client, care cere pagina web.

Această arhitectură este limitată. Să ne imaginăm de exemplu că avem o pagină web care afișează în timp real scorurile unor partide de fotbal. După încărcarea paginii, server-ul nu-i poate comunica browserului că un scor s-a schimbat. Browserul va afișa scorurile neactualizate până când utilizatorul reîmprospătează pagina.

Această problemă a fost rezolvată prin intermediul AJAX[1], o tehnică ce permite browserului să facă cereri asincrone către server după ce pagina a fost încărcată, prin intermediul JavaScript.

Următoarea etapă în acest proces incremental a fost crearea de *Single-Page Application*[2], denumite în continuare SPA. Într-un SPA, tot codul HTML, JavaScript și CSS este fie descărcat în momentul în care pagina este încărcată prima dată, fie în mod asincron, de obicei ca răspuns la acțiunile utilizatorului.

SPA oferă utilizatorului senzația unei aplicații fluide și poate uneori să ofere iluzia că aceasta răspunde la acțiuni imediat, fără să mai aștepte răspunsul serverului. Vom vedea în aplicația construită pentru această lucrare, de exemplu, că atunci când utilizatorul dorește ștergerea unei resurse, această resursă este

[1] Asynchronous JavaScript and XML; în aplicațiile moderne se utilizează cu preferință JSON (JavasScript Object Notation) în loc de XML, dar denumirea a rămas.

[2] http://en.wikipedia.org/wiki/Single-page_application

întâi înlăturată din UI, apoi o cerere asincronă îi spune serverului să șteargă resursa din baza de date. Desigur, pentru că se comunică cu serverul prin TCP/IP, această comunicare poate eșua, caz în care un mesaj de eroare este afișat și resursa reapare în UI, dar în mai mult de 90% din cazuri, când totul merge bine, utilizatorul are senzația că resursa este ștearsă instant.

Două companii foarte mari, Google și Facebook, au creat fiecare câte un framework pentru crearea de SPA: AngularJS și React, confirmând importanța acestui tip de aplicații. Experiență fluidă pentru utilizator, împreună cu alte avantaje pe care le vom discuta în capitolul următor au făcut ca SPA să crească foarte mult în popularitate în ultimii ani.

Cândva oamenii credeau că internetul este o altă lume, dar acum realizează că este o unealtă pe care o folosim în lumea noastră.

Tim Berners-Lee, inventatorul *www*-ului

1

Aplicații SPA: moduri de dezvoltare

În acest capitol vom enumera diferite moduri în care se pot dezvolta SPA și vom discuta avantajele și dezavantajele acestor moduri.

1.1 Framework-uri MVC JavaScript

Anumite framework-uri JavaScript pentru creare de aplicații web, cum ar fi Backbone.js[1], AngularJS[2], Ember.js[3], React[4] și Meteor[5] și-au propus să ușureze dezvoltarea de aplicații web SPA.

Aceste framework-uri oferă de obicei și posibilitatea organizării codului folosind șablonul arhitectural *Model–view–controller*[6] (MVC). MVC a fost folosit inițial în dezvoltarea aplicațiilor desktop, dar s-a dovedit mai târziu util și pentru dezvoltarea părții de back-end (server-side) a aplicațiilor web. Abia de curând el a fost adoptat și pe front-end (client-side).

MVC decuplează datele și logica aplicației de prezentare (interfața cu utilizatorul). Într-un framework JavaScript MVC, view-ul este reprezentat de șabloane HTML, controller-ul este un obiect JS care se ocupă de comunicarea dintre view și model, iar modelul este un obiect JS care, de obicei, mapează obiectele din baza de date de pe server la obiecte afișate de interfața cu utilizatorul. Desigur, o aplicație ce rulează în browser nu are acces în mod direct la baza de date, de aceea această mapare se face apelând un API REST[7].

În continuare enumerăm câteva avantaje ale folosirii unui framework JS pentru dezvoltarea de aplicații SPA:

[1] http://backbonejs.org/
[2] https://angularjs.org/
[3] http://emberjs.com/
[4] http://facebook.github.io/react/
[5] https://www.meteor.com/
[6] http://en.wikipedia.org/wiki/Model-view-controller
[7] http://en.wikipedia.org/wiki/Representational_state_transfer

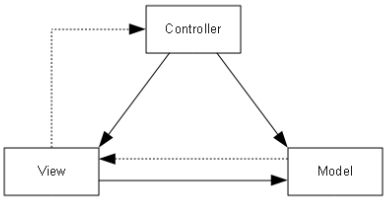


Figura 1.1: Interacțiunea dintre componentele MVC

- Folosirea MVC: un proiect MVC este mai ușor de navigat, este mai ușor de modificat și mai ușor de înțeles. De asemenea, colaborarea dintre designer și programator este ușurată de MVC.
- Viteza de dezvoltare (după depășirea curbei de învățare).
- Ușurința de dezvoltare.
- Diminuarea codului necesar a fi scris.

Bineînțeles, există și dezavantaje:

- Necesitatea învățării unei tehnologii noi. Ce este și mai trist, este că foarte posibil această tehnologie va fi depășită în doar câțiva ani.
- Unele framework-uri (AngularJS) au o curbă de învățare abruptă.
- Frameworkurile au tendința de a nu suporta browserele mai vechi.

1.2 AJAX

AJAX este modalitatea "clasică" prin care pot fi create SPA. Avantaje:

- Fiind o modalitate mai veche, este cunoscută de mai mulți programatori.
- Suport mai bun pentru browserele vechi.

Dezavantaje:

- Este necesar să se scrie mult cod.
- Folosirea MVC este mai dificilă.
- Viteză mică de dezvoltare.

1.3 WebSocket

WebSocket[8] este un protocol ce permite comunicare bidirecțională cu serverul. Atunci când clientul inițiază comunicarea cu serverul prin HTTP, serverul îi poate cere clientului să treacă la WebSocket. Dacă trecerea are loc, atunci

[8]http://en.wikipedia.org/wiki/WebSocket

serverul îi poate trimite notificări clientului, lucru care nu este posibil pe HTTP. Performanța WebSocket este mai bună decât AJAX. În plus, soluția este mai elegantă. Cu AJAX, clientul face *polling*, adică întreabă serverul la un anumit interval de timp dacă informații noi sunt disponibile. Cu WebSocket, nevoia pentru polling este eliminată.

WebSocket este un protocol relativ nou și este implementat doar pe browserele moderne. Pentru browserele mai vechi există librării JS care simulează WebSocket folosind AJAX.

WebSocket și frameworkurile JS pentru SPA nu sunt mutual exclusive. De exemplu, Meteor folosește WebSocket atunci când clientul suportă acest protocol și *SockJS*[9] atunci când protocolul nu este suportat.

1.4 Plugin-uri pentru browser

Java Applet-urile, o tehnologie creată de *Sun Microsystems* (cumpărat de Oracle) au promis prin anii '90 că vor revoluționa modul de dezvoltarea al aplicațiilor web, dar nu au reușit să se țină de promisiune. Rata lor de adopție a rămas foarte mică.

Macromedia, ulterior achiziționat de *Adobe* a reușit să facă o treabă mult mai bună cu *Flash*, care a atins o rată de adopție mult mai mare, dar această rată este în prezent în continuă scădere. Unul din motivele începutului sfârșitului pentru Flash a fost decizia companiilor *Google* și *Apple* de a nu suporta pluginul pe platformele lor mobile (*Android* și *iOS*). În prezent cel mai cunoscut site care folosește Flash este YouTube, dar acesta oferă și o alternativă bazată pe HTML5.

Microsoft a creat un rival pentru Flash, și anume *Silverlight*, dar tehnlologia a rămas aproape necunoscută, singurul site mare care a adoptat această tehnologie fiind *NetFlix*.

Aceste plugin-uri ușurau crearea de aplicații web complexe pentru dezvoltatori, dar desigur că era și un preț de plătit:

- Cerințe mari de resurse pentru calculatorul utilizatorului.
- Controlate de o companie.
- Closed-source.
- Interoperabilitate scăzută (suport scăzut pentru Linux, platforme mobile, browsere mai puțin cunoscute).

Din fericire, industria web "a decis" într-un mod organic să sprijine *Open Web*[10], iar în prezent chiar și companiile din spatele acestor tehnologii suportă, în ceea ce privește web-ul cel puțin, tehnologiile open-source și interoperabilitatea.

[9] http://sockjs.org
[10] http://en.wikipedia.org/wiki/Open_Web