# 8.1 ARIKETA - AJAX - SARRERA

## 1 Ariketa

Emandako web orritik abiatuta, gehitu ezazu beharrezkoa den JavaScript kodea helburu hauetarako:

1. Orrialdea kargatzean, testu-koadroak, lehenetsita, orrialdearen beraren URLa erakutsi behar du.
2. "Erakutsi edukiak" botoia sakatzean erabiltzaileak sartutako URLari dagokion edukia , AJAX eskaeren bidez deskargatu behar da. Zerbitzaritik jasotako erantzunaren edukia "Fitxategiaren edukiak" gunean erakutsi behar da.
3. "Eskaeraren egoerak" eremuan, eskaera zein egoeratan dagoen erakutsi behar da uneoro (Hasi gabe, kargatzen, osatuta, etab.).
4. Erakutsi zerbitzariaren erantzunaren goiburu guztien edukia "Zerbitzariaren erantzunaren HTTP goiburuak" gunean.
5. Erakutsi zerbitzariaren erantzunaren egoeraren kodea eta testua "Egoera-kodea" eremuan.

## 2 Ariketa

Emandako HTML orriak ***ticker*** izeneko eremu bat du, zerbitzariak sortutako albisteak erakusteko. Gehitu JavaScript kodea non:

1. Aldiro (adibidez, segundu bakoitzean), eskaera bat egiten zaio zerbitzariari AJAX bidez, eta erantzunaren edukia albisteetarako gordetako eremuan erakusten da.
2. Zerbitzariak bidalitako edukiaz gain, erantzuna zein ordutan jaso den erakutsi behar da.
3. "Gelditu" botoia sakatzen denean, aplikazioak zerbitzariari egindako aldizkako eskaerak geldiarazten ditu. Botoi horren gainean berriz sakatuz gero, aldizkako eskaerei berrekingo zaie.
4. "Aurrekoa" eta "Hurrengoa" botoien logika gehitu iezaiozu. Botoi horiek zerbitzariari egindako eskaerak geldiarazten dituzte eta une horretan erakusten denaren aurreko edo ondorengo edukiak erakusteko aukera ematen dute.
5. Zerbitzariaren erantzuna jasotzen denean, ***ticker*** izeneko eremua nabarmenduko da bisualki.
6. Alda ezazu aplikazioa, eskaerak egiteko XMLHttpRequest objektu bera etengabe berrerabil dadin.

## 3 ARIKETA

Baliozkotze konplexuaren adibide bat aukeratutako erabiltzaile-izen bat libre dagoen edo beste erabiltzaile batek erabiltzen duen egiaztatzea da. Datu-base oso handia erabiltzea eskatzen duen balidazioa denez, ezin da bezeroaren nabigatzailean egin. Aurretik erakutsitako teknikak eta ematen den web orria erabiliz:

1. Sor ezazu script bat, AJAXekin eta zerbitzariaren laguntzarekin, non erabiltzaileak aukeratutako izena libre dagoen ala ez egiaztatzen den.
2. Zerbitzariaren script-ari compruebaDisponibilidad.php  esaten zaio, eta izena duen parametroari login deitzen zaio.
3. Zerbitzariaren erantzuna "bai" edo "ez" da, erabiltzaile-izena libre dagoela esateko, eta beste erabiltzaile batek erabil dezakeela esateko, edo dagoeneko hartuta dagoela esateko.
4. Zerbitzariaren erantzunetik abiatuta, erabiltzaileari mezu bat erakutsi, egiaztapenaren emaitza adieraziz.

## 4 ArIKETA

Normalean, erabiltzaile-izen baten erabilgarritasuna baliozkotzen denean, ordezko balioen zerrenda bat agertzen da, aukeratutako izena erabilgarri ez badago. Alda ezazu aurreko ariketa, zerbitzariak itzulitako balio alternatibo batzuk erakutsi ahal izateko.

Zerbitzariaren script-ak compruebaDisponibilidadXML.php  izena du eta izena duen parametroa login da. Zerbitzariaren erantzuna XML dokumentu bat da, eta egitura hau du:

* Erabiltzaile-izena libre badago: code.6beb5621364789ab1484a6918c1c98678f97bcd7
* Erabiltzaile-izena okupatuta badago: code.6c7f5737463989caea8a3e7bda25a5dc8ef1a231
* Erabiltzaile-izen alternatiboak elementuen zerrenda moduan erakutsi behar dira (<ul></ul>).

Aldatu aurreko zerrenda, izen alternatibo bakoitzerako estekak erakuts ditzan. Ordezko izen baten estekan klikatzean, erabiltzailearen loginaren testu-koadroan kopiatzen da.

## 5 ARIKETA

Berregin ezazu aurreko ariketa zerbitzariaren erantzunak JSON formatuan prozesatzeko. Aldaketa hauek gertatu dira:

1. Zerbitzariaren script-ak compruebaDisponibilidadJSON.php  izena du eta izena duen parametroari login deitzen zaio.
2. Zerbitzariaren erantzuna JSON objektu bat da, eta egitura hau du:

Erabiltzailearen izena libre dago: code.3d1e6360bcbd7143aa4cd480b2e43de5a59a4d2a js]{ disponible: "no", alternativas: ["...", "...", ..., "..."] }[/code

## 6 ARIKETA

Sor ezazu script bat non, herrialde bateko probintzien zerrenda eta hautatutako probintzia bakoitzeko udalerrien zerrenda modu dinamikoan kargatuko duen.

1. Hutsik dauden bi zerrenda zabalgarrien HTML kodea definitu ezazu.
2. Orria kargatzen denean, kargatu probintzien zerrenda goitibeherako lehen zerrendan. Zerbitzariaren scripta cargaProvinciasXML.php da. Erantzunaren formatua XML da, eta egitura hau du:

<provincias>

<provincia>

<codigo>01</codigo>

<nombre>Álava</nombre>

</provincia>

...

</provincias>

Aukerak goitibeherako zerrendan txertatzeko, bi teknika erabil daitezke:

1. < option> etiketa bakoitzaren zerrendaren eta HTML kodearen innerHTML propietatea.
2. Aukera motako elementuak sortzea (new Option (izena, balioa)) eta goitibeherako zerrendako array options [] delakoari gehitzea.
3. Gehi ezazu semantikoki, probintzien zerrendara egokitutako gertaera, probintzia bat hautatzen denean udalerri guztiak automatikoki kargatu daitezen beste zerrendan.
4. Probintzia jakin bat hautatzen denean, AJAX bidez kargatzen da udalerrien zerrenda osoa goitibeherako beste zerrendan. Zerbitzariaren script cargaMunicipiosXML.php da. Zerbitzarira bidali behar den parametroa probintziaren kodea da, eta parametroari probintzia deritzo. Zerbitzariak espero duen metodoa POST da. Erantzunaren formatua XML da, eta egitura hau du:

<municipios>

<municipio>

<codigo>0014</codigo>

<nombre>Alegría-Dulantzi</nombre>

</municipio>

...

</municipios>

## 7 ARIKETA

Alda ezazu aurreko ariketa zerbitzariaren erantzunak JSON formatuan jasateko. Aldaketa hauek egin dira:

1. Probintziak kargatzeko erabiltzen den zerbitzariaren script-a cargaProvinciasJSON.php  deitzen da eta zerbitzariaren erantzunak formatu hau du:

{ "01": "Álava/Araba", "02": "Albacete", "03": "Alicante/Alacant", ... }

1. Udalerriak kargatzeko erabilitako zerbitzariaren script-a cargaMunicipiosJSON.php da eta zerbitzariaren erantzunak formatu hau du:

{ "0014": "Alegría-Dulantzi", "0029": "Amurrio", ... }