Práctica 3: Aplicaciones web en el servidor

La siguiente práctica tiene como objetivo reforzar los conocimientos adquiridos relativos a generación de servicios utilizando aplicaciones de servidor. Se construirá una aplicación web utilizando un patrón de diseño. Consta de cuatro apartados:

a) En un primer apartado se procederá a la creación de una aplicación con la funcionalidad de un servidor HTTP básico.
b) En un segundo apartado, se definirán los servicios y añadirá la lógica de negocio.
c) En el tercer apartado se programarán las clases del modelo de datos, que accederán a un sistema gestor de bases de datos.
d) En un cuarto apartado, se crearán las vistas de la aplicación.

Objetivos:

- Practicar con los conocimientos estudiados relativos a la creación de servicios utilizando aplicaciones de servidor.
- Entender el patrón MVC y utilizarlo para la generación de aplicaciones de servidor.
- Entender la utilidad de los formatos para el intercambio de datos, como puede ser JSON.

1. Servidor HTTP básico.

En este primer apartado se va a crear una aplicación con funcionalidad de servidor HTTP básico. Se llamará ***server.js*** y el puerto del servidor va a ser el 23700.

A continuación se presenta un código de ayuda:

```
var express = require('express');
var app = express();

app.get("/", function(req, res) {
   res.send('Sever SSAATT running...');
});

app.listen(23700);
```

Al conectarnos al servidor, se recibe una respuesta como la de la figura 3.1.

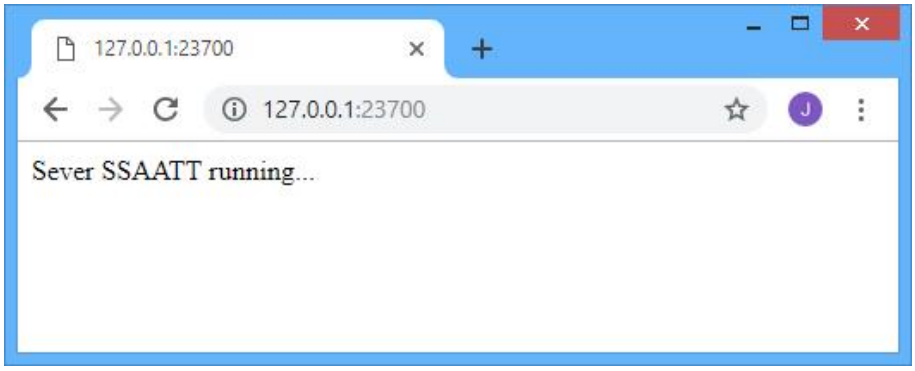

Figura 3.1. Respuesta del servidor.

Para poder utilizar HTTPS, usaremos la aplicación ***ngrok***. Debemos indicar el puerto de nuestro servidor básico (23700), tal y como se muestra en la figura 3.2. La respuesta del servidor aparece en la figura 3.3.

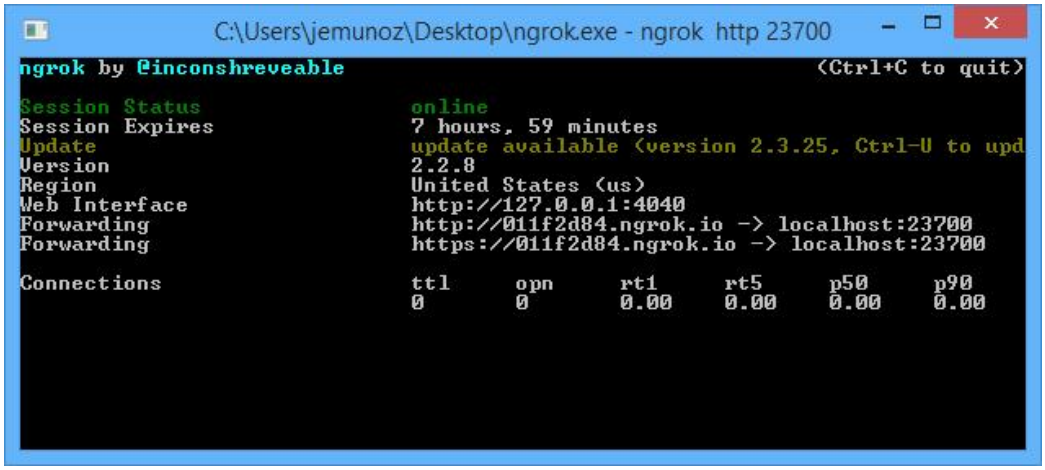

Figura 3.2. Activación de ngrok.

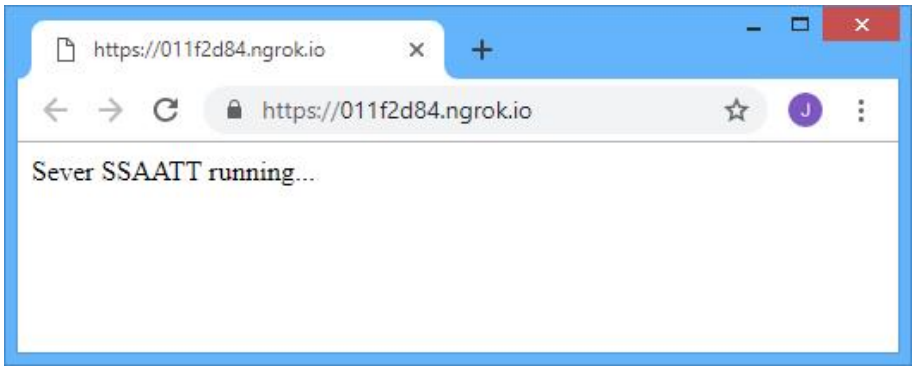

Figura 3.3. Respuesta del servidor utilizando una aplicación de túneles ***ngrok***.

Preguntas:

a. Sin ngrok, ¿qué puerto se usa para HTTP?
b. Con ngrok, ¿qué puerto se usa para HTTP? ¿Y para HTTPS?

2. Definición de los servicios

En este apartado se definirán los servicios y se planificará la lógica de negocio de nuestro controlador (estos servicios los creamos en ***servidor.js***). Como ya conocemos desde la práctica 1, contemplamos tres servicios:

1. Contactos. Este servicio lista los contactos de un usuario y se invoca utilizando un método GET. Hay que proporcionarle el nombre del usuario en la url. En el controlador debe existir un código parecido a:

```
app.get("/contactos/:id", function(req, res) {
      res.setHeader('Access-Control-Allow-Origin','*');
      res.setHeader('Content-Type', 'application/json');
 });
```

La línea res.setHeader(Access-Control-Allow-Origin’,’*’) se incluye para permitir al navegador el acceso a diferentes orígenes (Cross-Origin Resource Sharing, CORS). Gracias a ella, hemos podido crear una aplicación cliente en la práctica 1 que se ubica en un servidor diferente al de los servicios.

La línea res.setHeader(’Content-Type’, ’application/json’) se incluye para que el servidor envíe una respuesta codificada en JSON.

2. Envía. Es el servicio para enviar mensajes desde un remitente a un destinatario. Utiliza un método POST y hay que pasarle el nombre del remitente (***id_org***), destinatario (***id_dest***), tipo de mensaje (***tipo***) y contenido (***contenido***). En node.js se utiliza el paquete ‘body-parser’ (hay que instalarlo) para obtener datos del mensaje de petición. Simplemente hay que incluir las líneas:

```
var bodyParser=require("body-parser");
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({extended: true}));
```

Para este servicio, se incluirá un código similar a:

```
app.post("/envia", function(req, res) {
    var dtoMensajes=req.body;
    res.setHeader('Access-Control-Allow-Origin','*');
    res.setHeader('Content-Type', 'application/json');
});
```

3. Recibe. Utilizado para leer los mensajes que un remitente ha enviado al destinatario mediante un método GET. Hay que proporcionar el nombre del remitente y destinatario. El siguiente código debe aparecer en el controlador:

```
app.get("/recibe/:id_org/:id_dest", function(req, res) {
    res.setHeader('Access-Control-Allow-Origin','*');
    res.setHeader('Content-Type', 'application/json');
});
```