UNIVERSIDAD CATÓLICA DE HONDURAS

"NUESTRA SEÑORA REINA DE LA PAZ"

Alumno:

Oscar Andree Varela Godoy

Catedrático:

Carlos Antonio Flores

Asignatura:

Programación Móvil II

Tema:

Investigación #2

Indice

Contenido

Introducción:

Por medio de esta investigación (Glosario) buscamos encontrar mayor información de los paquetes para poder comprender el funcionamiento de cada uno de ellos, empezando desde conocer que son e investigando sobre su correcto funcionamiento para el momento de tener que utilizarlo ya estar familiarizado un poco con cada paquete que se investigo

Objetivos:

Objetivos Generales:

- Conocer que son los paquetes
- Saber cómo funcionan

Objetivos específicos:

- Reconocer que hace cada uno de los paquetes que se investigaron para dar un mejor uso a cada uno de estos.

Contenido:

1. NPM dontev:

¿Qué es?

Dotenv es un módulo de dependencia cero que carga variables de entorno desde un .envarchivo a process.env. El almacenamiento de la configuración en el entorno separado del código se basa en la metodología de la aplicación The Twelve-Factor

Instalación:

```
# instalar localmente (recomendado)
npm install dotenv –save
```

Uso:
Cree un .envarchivo en la raíz de su proyecto:
S3_BUCKET = " YOURS3BUCKET "
SECRET_KEY = " YOURSECRETKEYVASHERE "

Tan pronto como sea posible en su aplicación, importe y configure dotenv:
requerir ('dotenv') . consola de configuración ( )
. log (proceso . env) // elimina esto después de que hayas confirmado que funciona

.. o usando ES6?
import 'dotenv/config' // consulte https://github.com/motdotla/dotenv#how-do-i-use-dotenv-with-import
import express from 'express'

Eso es todo. process.envahora tiene las claves y los valores que definió en su .envarchivo:
requerir ('dotenv') . configuración ( )
. . .
s3 . getBucketCors ({ Cubo : proceso . env . S3_BUCKET } , función (err , datos) { })

2. Jsonwebtoken

¿Qué es jsonwebtoken?

JSON Web Token (JWT) es un estándar abierto (RFC-7519) basado en JSON para crear un token que sirva para enviar datos entre aplicaciones o servicios y garantizar que sean válidos y seguros.

El caso más común de uso de los JWT es para manejar la autenticación en aplicaciones móviles o web.

Instalacion:

$ npm instalar jsonwebtoken

Uso:

(Asíncrono) Si se proporciona una devolución de llamada, la devolución de llamada se llama con el `err`o el JWT.

(Sincrónico) Devuelve el JsonWebToken como cadena

`Payload` podría ser un objeto literal, un búfer o una cadena que represente un JSON válido.

3. Passport :

¿Qué es Passport?

Passport es un módulo de autorización popular para el nodo. En palabras simples, maneja todas las solicitudes de autorización de los usuarios en su aplicación. Passport admite más de 300 estrategias para que pueda integrar fácilmente el inicio de sesión con Facebook / Google o cualquier otra red social que lo use. La estrategia que discutiremos aquí es el Local donde usted autentica a un usuario utilizando su propia base de datos de usuarios registrados (con nombre de usuario y contraseña).

Instalación:

$ npm install passport

Uso:

Estrategias

Passport utiliza el concepto de estrategias para autenticar solicitudes. Las estrategias pueden ir desde la verificación de las credenciales del nombre de usuario y la contraseña, la autenticación delegada mediante OAuth (por ejemplo, a través de Facebook o Twitter) o la autenticación federada mediante OpenID .

Antes de autenticar las solicitudes, se debe configurar la estrategia (o estrategias) que utiliza una aplicación.

```
pasaporte _ use (new LocalStrategy (
 función (nombre de usuario, contraseña, hecho) {
   Usuario. findOne ({nombre de usuario: nombre de usuario}, función (err, usuario) {
    if(err)  {  return  done ( err ) ;  }
    if(! user)  {  return  hecho( nulo ,  falso ) ;  }
    if(!usuario . verificar Contraseña (contraseña)) {return hecho (nulo, falso);}
    return  hecho ( nulo ,  usuario ) ;
   } ) ;
 }
) ) ;
```

4. Passport-jwt:

¿Qué es Passport-jwt?

Una estrategia de Passport para la autenticación con un token web JSON.

Este módulo le permite autenticar puntos finales mediante un token web JSON. Está destinado a ser utilizado para proteger puntos finales RESTful sin sesiones.

Instalación:

npm install passport-jwt

Uso:

La estrategia de autenticación JWT se construye de la siguiente manera:
new JwtStrategy(options, verify)

```
var  JwtStrategy  = require ( ' pasaporte-jwt ' ) . estrategia ,
   ExtractJwt  = require ( ' pasaporte-jwt ' ) . ExtraerJwt ;
var  opciones  = { }
opta _ jwtFromRequest = ExtraerJwt . fromAuthHeaderAsBearerToken ( ) ;
opta _ secretoOrClave = ' secreto ' ;
opta _ emisor = ' cuentas.examplesoft.com ' ;
opta _ audiencia = ' susitio.net ' ;
pasaporte _ use ( nueva JwtStrategy  ( opciones , función ( jwt_payload , hecho ) {
   usuario _ findOne ( { id : jwt_payload . sub } , función ( err , usuario ) {
      si ( err ) {
         volver hecho ( err , falso ) ;
      }
      si ( usuario ) {
         volver hecho ( nulo ,  usuario ) ;
      } más {
         volver hecho ( nulo , falso ) ;
         //  o puedes crear una nueva cuenta
      }
   } ) ;
} ) ) ;
```

5. Nodemailer:

¿Qué es nodemailer?

Nodemailer es un módulo que nos permitirá añadir estas funcionalidades en nuestro servidor Node.js. En esta ocasión, utilizaremos una cuena de gmail, pero en la documentación de Nodemailer podemos encontrar instrucciones para hacerlo con diversos servicios.

Instalacion:

```
npm install nodemailer
```

Uso:

Después de instalar, definimos un controlador mailCtrl.js que exporte un método sendEmail:

```
var nodemailer = require('nodemailer'); // email sender function exports.sendEmail =
function(req, res){
   // nodemailer stuff will go here
};
```

Para enviar un email, primero debemos definir un transporter. Lo haremos de la siguiente manera, sustituyendo los datos por los de nuestra cuenta gmail:

```
var transporter = nodemailer.createTransport({
     service: 'Gmail',
     auth: {
        user: 'example@gmail.com',
        pass: 'password'
     }
});
```

Ahora que ya tenemos el transporter, definiremos el propio email:

```
var mailOptions = {
     from: 'Remitente',
     to: 'destinatario@gmail.com',
     subject: 'Asunto',
     text: 'Contenido del email'
};
```

y ya sólo nos queda enviar el email!

```
transporter.sendMail(mailOptions, function(error, info){
   if (error){
      console.log(error);
      res.send(500, err.message);
   } else {
      console.log("Email sent");
      res.status(200).jsonp(req.body);
   }
});
```

Bibliografía:

- npm: dotenv. (2022, 10 mayo). Npm. Recuperado 1 de junio de 2022, de https://www.npmjs.com/package/dotenv

- npm: jsonwebtoken. (2019, 18 marzo). Npm. Recuperado 1 de junio de 2022, de https://www.npmjs.com/package/jsonwebtoken

- npm: passport. (2022, 20 mayo). Npm. Recuperado 1 de junio de 2022, de https://www.npmjs.com/package/passport

- npm: passport-jwt. (2018, 13 marzo). Npm. Recuperado 1 de junio de 2022, de https://www.npmjs.com/package/passport-jwt

- npm: nodemailer. (2022, 4 mayo). Npm. Recuperado 1 de junio de 2022, de https://www.npmjs.com/package/nodemailer

- U. (2020b, mayo 26). Envía emails desde Node.js con Nodemailer - uesteibar. Medium. Recuperado 1 de junio de 2022, de https://medium.com/@uesteibar/env%C3%ADa-emails-desde-node-js-con-nodemailer-178cacf5cf6b#:%7E:text=Nodemailer%20es%20un%20m%C3%B3dulo%20que,para%20hacerlo%20con%20diversos%20servicios.