1. const express = require('express');
2. const bcrypt = require('bcrypt');
3. const jwt = require('jsonwebtoken');
4. const { OAuth2Client } = require('google-auth-library');
5. const client = new OAuth2Client('TU ID DE GOOGLE');
6. const Usuario = require('../models/usuario');
7. const app = express();

10. app.post('/google', async(req, res) => {
11. let token = req.body.idtoken;
12. let googleUser = await verify(token)
13. .catch(e => {
14. return res.status(403).json({
15. ok: false,
16. err: e
17. });
18. });
20. Usuario.findOne({ email: googleUser.email }, (err, usuarioDB) => {
21. if (err) {
22. return res.status(500).json({
23. ok: false,
24. err
25. });
26. };
27. if (usuarioDB) {
28. if (usuarioDB.google === false) {
29. return res.status(400).json({
30. ok: false,
31. err: {
32. message: 'Debe de usar su autenticación normal'
33. }
34. });
35. } else {
36. let token = jwt.sign({
37. usuario: usuarioDB
38. }, process.env.SEED, { expiresIn: process.env.CADUCIDAD\_TOKEN });
39. return res.json({
40. ok: true,
41. usuario: usuarioDB,
42. token,
43. });
44. }
45. } else {
46. // Si el usuario no existe en nuestra base de datos
47. let usuario = new Usuario();
48. usuario.nombre = googleUser.nombre;
49. usuario.email = googleUser.email;
50. usuario.img = googleUser.img;
51. usuario.google = true;
52. usuario.password = ':)';
53. usuario.save((err, usuarioDB) => {
54. if (err) {
55. return res.status(500).json({
56. ok: false,
57. err
58. });
59. };
60. let token = jwt.sign({
61. usuario: usuarioDB
62. }, process.env.SEED, { expiresIn: process.env.CADUCIDAD\_TOKEN });
63. return res.json({
64. ok: true,
65. usuario: usuarioDB,
66. token,
67. });
68. });
69. }
70. });
71. });
72. module.exports = app;

Programa de alumno que hace la pregunta

1. app.post('/google', (req, res) => {
2. var token = req.body.token || '';
3. const client = new OAuth2Client(GOOGLE\_CLIENT\_ID, GOOGLE\_CLIENT\_SECRET);
4. const ticket = client.verifyIdToken({
5. idToken: token,
6. audience: GOOGLE\_CLIENT\_ID
7. });
8. ticket.then(data => {
9. res.status(200).json({
10. ok: true,
11. ticket: data.payload,
12. userid: data.payload.sub
13. });
14. Usuario.findOne({ email: ticket.email }, (err, usuario) => {
15. if (err) {
16. return res.status(500).json({
17. ok: false,
18. mensaje: 'Error al buscar el usuario - login',
19. errors: err
20. });
21. }
22. if (usuario) {
23. if (usuario.google === false) {
24. return res.status(400).json({
25. ok: true,
26. mensaje: 'Debe Utilizar el método de Autenticación normal para poder acceder...'
27. });
28. } else {
29. // Crear un Token
30. usuario.password = ':)';
31. var token = jwt.sign({ usuario: usuario }, SEED, { expiresIn: 14400 });
32. res.status(200).json({
33. ok: true,
34. usuario: usuario,
35. token: token,
36. id: usuario.\_id,
37. // body: body
38. });
39. }
40. // Si el Usuario no existe en la base de datos
41. } else {
42. console.log(ticket);
43. var payload = ticket.payload;
44. var nuevoUsuario = new Usuario();
45. nuevoUsuario.nombre = ticket.name;
46. nuevoUsuario.email = ticket.email;
47. nuevoUsuario.password = ':)';
48. nuevoUsuario.img = data.picture;
49. nuevoUsuario.google = true;
50. console.log(nuevoUsuario);
51. /\*nuevoUsuario.save((err, usuarioBD) => {
52. if (err) {
53. return res.status(500).json({
54. ok: false,
55. mensaje: 'Error al crear el usuario - google',
56. errors: err
57. });
58. }
59. // Crear un Token
60. var token = jwt.sign({ usuario: usuarioBD }, SEED, { expiresIn: 14400 });
62. res.status(200).json({
63. ok: true,
64. usuario: usuarioBD,
65. token: token,
66. id: usuarioBD.\_id,
67. // body: body
68. });
69. });\*/
70. }
71. });
72. }).catch(err => {
73. if (err) {
74. return res.status(400).json({
75. ok: false,
76. mensaje: 'Token no válido',
77. errors: err
78. });
79. }
80. })
81. });