



El futuro digital
es de todos

MinTIC



The logo features the text "Mision TIC 2022" in a bold, sans-serif font. The word "Mision" is in blue, and "TIC 2022" is in red. A red curved line starts from the top of the letter "i" in "Mision" and ends at the top of the letter "i" in "2022". The background of the logo is a white circle with a gray halftone dot pattern.

Formato de Informe de Seguimiento



Universidad de Caldas



El futuro digital
es de todos
Conformación del grupo

MinTIC

Misión
TIC 2022
Nivel de

Integrantes (nombre completo)	Cédula	Rol	participación (alto, medio, bajo, retirado)
Sierra Reyes Rodrigo Armando	71332267	Diseñador UI - Diseñador de Software	Alto
Velez Felipe Andres		Tester – Diseñador de Software	Alto
Vasquez Vargas Jorge Ivan	71373279	Líder de equipo – Administrador de la configuración - Diseñador de Software	Alto
Toro Muñoz Juan Manuel	N/A	N/A	Retirado
Tabaco Duran Didier Julian	N/A	N/A	Retirado



El futuro digital
es de todos

MinTIC



Reuniones de Sprint 2

Sesión 01 - 18 de noviembre

Participantes:

Sierra Reyes Rodrigo Armando

Velez Felipe Andres

Vasquez Vargas Jorge Ivan

Conclusiones:

Se diseñaron las tareas para ejecutar en el Sprint 2 basados en la experiencia de Felipe Velez en su rol de Tester con el reto 3 de Sendgrid

Sesión 02 - 21 de noviembre

Participantes:

Sierra Reyes Rodrigo Armando

Velez Felipe Andres

Vasquez Vargas Jorge Ivan

Conclusiones:

Se registran las tareas en el tablero Kanban, se hace la revisión de la documentación de Twilio y Sendgrid para el uso del código en Node.js. Se genera el código y se realizan las pruebas de funcionamiento. Se realiza la grabación del video explicativo de la implementación.



El futuro digital
es de todos

MinTIC

Mision
TIC 2022

Tablero Kanban

<https://github.com/JorgeVasquez423/DigitalTeamBackend/projects/1>

Tareas inicio Sprint 2

A screenshot of a GitHub project Kanban board titled "Tienda Tecnologica by Digital Team". The board has five columns: "Items", "En progreso", "Pendientes", "Pruebas", and "Sprint 2 Finalizado". The "En progreso" column contains six cards, each representing a task from "019 - Registro y activación en Twilio" to "024 - Video funcionamiento". The "Pendientes" column contains one card, "018 - Reto Semana 3". The "Pruebas" column contains one card, "018 - Reto Semana 3". The "Sprint 2 Finalizado" column is empty.

Tareas final Sprint 2

A screenshot of the same GitHub project Kanban board after Sprint 2. The "En progreso" column now contains one card, "023 - Backend envío de SMS". The "Pendientes" column is empty. The "Pruebas" column contains one card, "018 - Reto Semana 3". The "Sprint 2 Finalizado" column now contains all the cards from "019" to "024", indicating they have been completed.



El futuro digital es de todos

MinTIC



Repositorio Github

<https://github.com/JorgeVasquez423/DigitalTeamBackend>

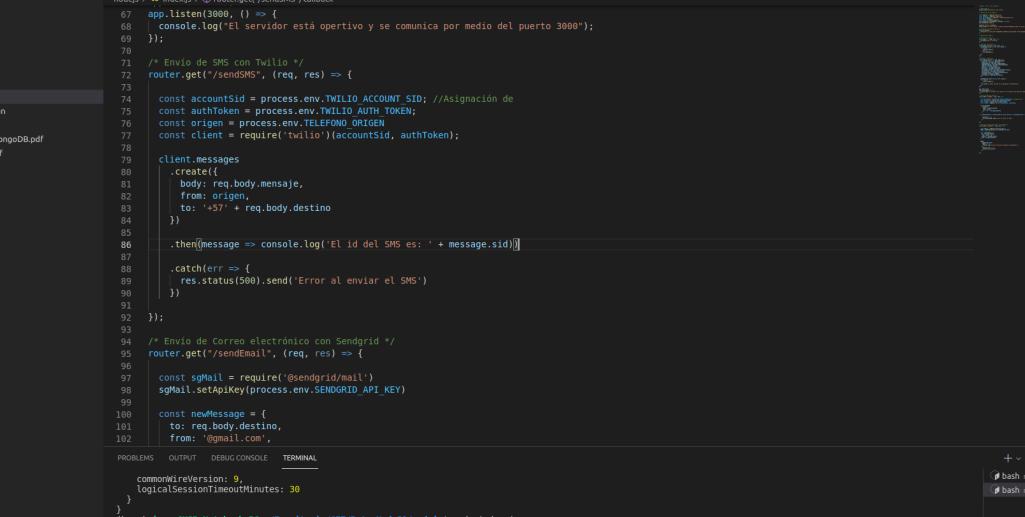
<https://github.com/JorgeVasquez423/DigitalTeamBackend.git>

Video implementación

<https://youtu.be/21IOZBAQ8no>

Código

<https://drive.google.com/drive/u/1/folders/1ofNomgUvEkIlyWEbVfGuZTJ173XOoOYC>



The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a folder structure for "RETO NODEJS" containing node_modules, vendor, models, persona.js, index.js, package-lock.json, package.json, Reto NodeJS y MongoDB.pdf, and Reto Sendgrid.pdf.
- Code Editor:** The active file is index.js, which contains code for a Node.js application. It includes logic for listening on port 3000, sending SMS messages using Twilio, and sending emails using Sendgrid.
- Terminal:** The terminal at the bottom shows the command "node index.js" being run and the application responding with its welcome message and environment variables.

```
File Edit Selection View Go Run Terminal Help
RETO NODEJS
  node_modules
  vendor
  models
    persona.js
  index.js
  package-lock.json
  package.json
  Reto NodeJS y MongoDB.pdf
  Reto Sendgrid.pdf

index.js x
File Edit Selection View Go Run Terminal Help
RETO NODEJS
  node_modules
  vendor
  models
    persona.js
  index.js
  package-lock.json
  package.json
  Reto NodeJS y MongoDB.pdf
  Reto Sendgrid.pdf

index.js x
1  // index.js
2  /*
3  * Este archivo es el punto de entrada para la aplicación Node.js.
4  * Se encarga de configurar el servidor y las rutas para manejar las solicitudes.
5  */
6  const express = require('express');
7  const router = express.Router();
8  const bodyParser = require('body-parser');
9  const mongoose = require('mongoose');
10
11  // Importación de los modelos
12  const Persona = require('./models/persona');
13
14  // Configuración del servidor
15  const app = express();
16  const PORT = process.env.PORT || 3000;
17
18  // Interceptor para manejar errores
19  app.use((err, req, res, next) => {
20     if (err instanceof mongoose.Error.ValidationError) {
21         res.status(400).json({ error: err.message });
22     } else {
23         res.status(500).json({ error: 'Internal Server Error' });
24     }
25  });
26
27  // Interceptor para manejar las respuestas
28  app.use((req, res, next) => {
29     res.header('Content-Type', 'application/json');
30     next();
31  });
32
33  // Interceptor para manejar las solicitudes
34  app.use(bodyParser.json());
35
36  // Rutas
37  router.get('/listar', async (req, res) => {
38     try {
39         const personas = await Persona.find();
40         res.json(personas);
41     } catch (error) {
42         res.status(500).json({ error: 'Hubo un error al listar las personas' });
43     }
44  });
45
46  router.post('/crear', async (req, res) => {
47     try {
48         const persona = new Persona(req.body);
49         await persona.save();
50         res.json(persona);
51     } catch (error) {
52         res.status(500).json({ error: 'Hubo un error al crear la persona' });
53     }
54  });
55
56  router.put('/actualizar/:id', async (req, res) => {
57     try {
58         const persona = await Persona.findById(req.params.id);
59         if (!persona) {
60             res.status(404).json({ error: 'La persona no existe' });
61         } else {
62             const updatedPerson = {
63                 nombre: req.body.nombre,
64                 apellido: req.body.apellido,
65                 edad: req.body.edad,
66                 correo: req.body.correo
67             };
68             await persona.updateOne(updatedPerson);
69             res.json(persona);
70         }
71     } catch (error) {
72         res.status(500).json({ error: 'Hubo un error al actualizar la persona' });
73     }
74  });
75
76  router.delete('/eliminar/:id', async (req, res) => {
77     try {
78         const persona = await Persona.findById(req.params.id);
79         if (!persona) {
80             res.status(404).json({ error: 'La persona no existe' });
81         } else {
82             await persona.remove();
83             res.json({ message: 'La persona ha sido eliminada' });
84         }
85     } catch (error) {
86         res.status(500).json({ error: 'Hubo un error al eliminar la persona' });
87     }
88  });
89
90  // Ruta para enviar SMS
91  router.get('/enviarSMS', (req, res) => {
92     const accountSid = process.env.TWILIO_ACCOUNT_SID; // Asignación de
93     const authToken = process.env.TWILIO_AUTH_TOKEN;
94     const origen = process.env.TELEFONO_ORIGEN
95     const client = require('twilio')(accountSid, authToken);
96
97     client.messages
98       .create({
99         body: req.body.mensaje,
100        from: origen,
101        to: '+57' + req.body.destino
102       })
103
104       .then(message => console.log('El id del SMS es: ' + message.sid))
105
106       .catch(err => {
107         res.status(500).send('Error al enviar el SMS')
108       })
109     });
110
111  // Envío de Correo electrónico con Sendgrid
112  router.get('/enviarEmail', (req, res) => {
113
114    const sgMail = require('@sendgrid/mail')
115    sgMail.setApiKey(process.env.SENDGRID_API_KEY)
116
117    const newMessage = {
118      to: req.body.destino,
119      from: 'correo@gmail.com',
120
121      commonWireVersion: 9,
122      logicalSessionTimeoutMinutes: 30
123    }
124
125    (base) jorge@USB-Notebook-PC:~/Escritorio/GIT/Reto NodeJS/nodejs$ node index.js
Bienvenido a este reto
Your environment variable TWILIO ACCOUNT SID has the value: AC473fc27856174f93f3a01c9db6593e9c
El servidor está operativo y se comunica por medio del puerto 3000
SM07306125f544e039998c0d70b4360d1
C:
(base) jorge@USB-Notebook-PC:~/Escritorio/GIT/Reto NodeJS/nodejs$
```



**El futuro digital
es de todos**

MinTIC

The logo consists of the word "Mission" in blue and red block letters, with "TIC 2022" in red block letters below it. A red curved line starts from the top of the letter "i" in "Mission" and ends at the bottom of the letter "c" in "TIC 2022".

The screenshot shows a Visual Studio Code interface with the following details:

- File Explorer:** Shows the project structure with files like `node_modules`, `nodejs`, `modelos`, `personajes`, `env`, `index.js`, `package-lock.json`, `package.json`, `Reto NodeJS y MongoDB.pdf`, and `Reto Sendgrid.pdf`.
- Code Editor:** The `index.js` file is open, containing code for sending an email using Sendgrid. The code includes imports for `express`, `body-parser`, `mongoose`, and `twilio`. It defines a `router` and sets up routes for sending SMS and emails.
- Terminal:** The terminal shows the command `node index.js` being run, and the output indicates that the server is running on port 3008 and the Twilio account SID is set to `A473fc27856174f93f3a01c9db6593e9c`.

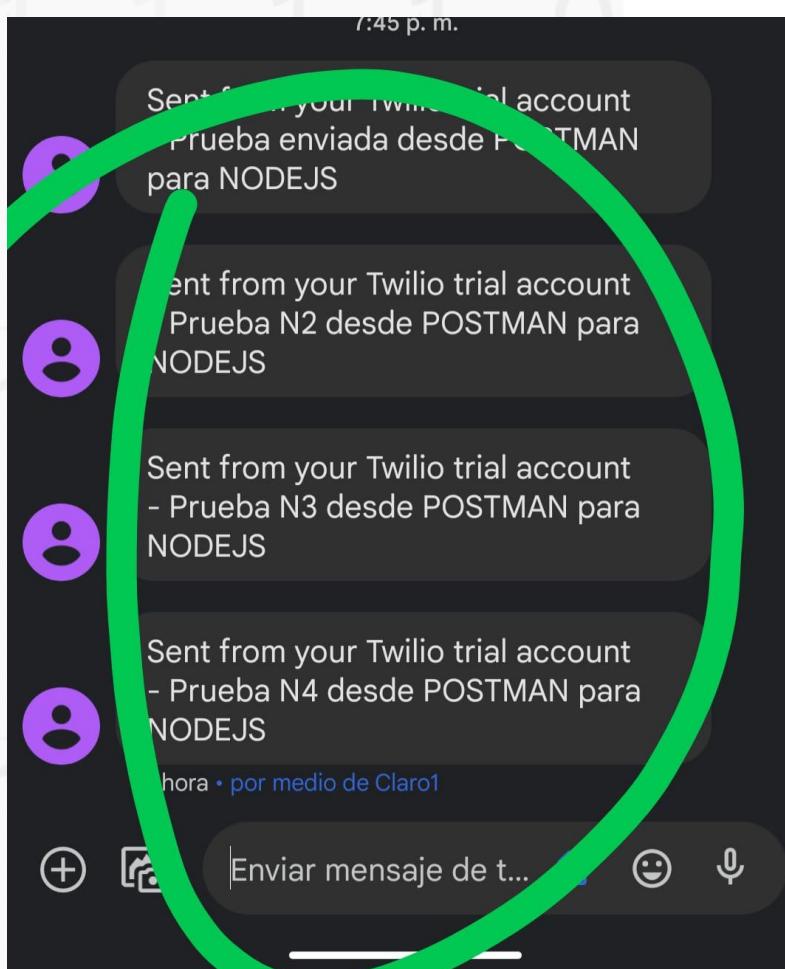
The screenshot shows the Postman application window. In the top navigation bar, there are links for File, Edit, View, Help, Home, Workspaces, Reports, and Explore. A search bar is located at the top right. The main workspace is titled "My Workspace" and contains a collection named "sendSMS". The collection details show a GET request to "http://localhost:3000/sendSMS". The "Body" tab is selected, showing the content type as "x-www-form-urlencoded" with fields "mensaje" and "destino" set to their respective values. The "Response" section displays an error message: "Could not get response" and "Error: socket hang up". Below the error message, there is a link to "View in Console". The left sidebar includes sections for Collections, APIs, Environments, Mock Servers, and Monitors, along with a "Create Collection" button. The bottom navigation bar has links for Find and Replace, Console, Bootstrap, Runner, and Trash.



El futuro digital
es de todos

MinTIC

Mision
TIC 2022



El futuro digital
es de todos

MinTIC

Misión
TIC 2022

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'My Workspaces' and various collection icons. The main area displays a POST request to 'http://localhost:3000/sendEmail'. The 'Body' tab is selected, showing three parameters: 'destino' with value 'progruebas.misiontic@gmail.com', 'asunto' with value 'Prueba de envío desde Sendgrid', and 'mensaje' with value 'Prueba de envío desde Sendgrid'. Below the request, there's a response section with a small illustration of a person holding a rocket.

The screenshot shows a Gmail inbox with a dark wood-grain background. An incoming email is displayed with the subject 'Prueba envío Correo desde NODEJS'. The email was sent by 'felipeandresvlez@gmail.com' to 'travesia.sendgrid.net'. The message body contains the text 'Este es un mensaje de prueba de Node.js'. The Gmail interface includes a sidebar with options like Recibidos, Destacados, and Envíados.