

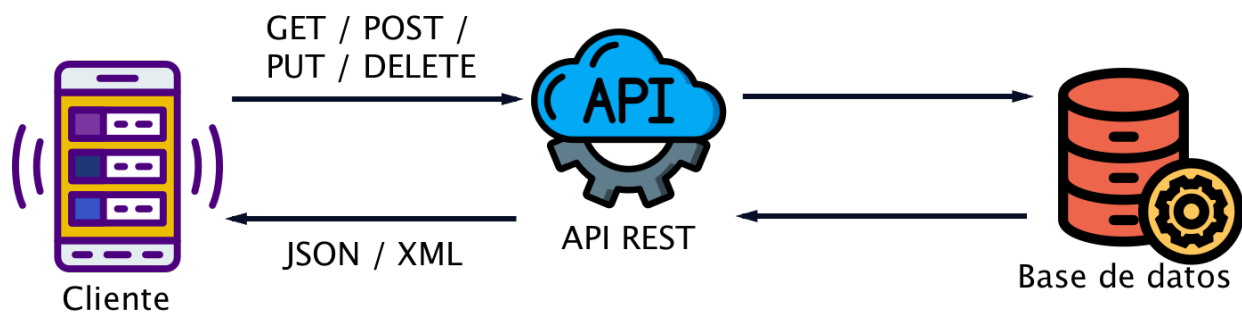


UNIVERSIDAD  
TECNOLÓGICA  
METROPOLITANA  
*del Estado de Chile*

# Proyecto Rest

**Benjamin Jesús Herrera Santibáñez**  
**Vania Michelle Palacios Gómez**

*Ingeniería Civil en Computación mención en Informática,*  
[bherrerass@utem.cl](mailto:bherrerass@utem.cl)  
[vpalacios@utem.cl](mailto:vpalacios@utem.cl)



Profesor: Sebastián Salazar Molina.

Asignatura: Computación Paralela y Distribuida.

Sección: 411.

## Índice

Índice	02
Introducción	03
Problemática	04
Tecnología	07
Solución	11
Conclusión	13
Bibliografía	14

## Introducción

A lo largo de la historia de la computación hemos avanzado muchos pasos en tanto los niveles comunicativos como también los de programación, por ello es vital para nosotros las nuevas generaciones comprender el correcto funcionamiento de la computación como esta a su vez está conformada los servidores y sus redes. Por ejemplo, en las aplicaciones, sitios webs e incluso juegos, tienen un componente denominado backend y frontend, ¿Por qué estos conceptos son tan importantes en prácticamente todos los apartados tanto de mercado como de formación profesional?

Por lo tanto, debemos recordar ¿qué es un backend y frontend?

- Backend: También llamado el lado del servidor, consta de servidores que sirven datos a pedido, aplicaciones que canalizan datos y bases de datos que organizan la información. La tarea principal de un desarrollador backend es crear aplicaciones que puedan extraer datos y entregarlos al frontend, por lo que es toda la parte de codificación y datos que el usuario no ve de forma directa.
- Frontend: Se construye utilizando una combinación de tecnologías como el lenguaje de marcado de hipertexto (HTML), JavaScript y las hojas de estilo en cascada (CSS). Por lo que un desarrollador front-end diseña y crea elementos de interacción con el usuario en una página web o aplicación, como botones, menús, páginas, enlaces y gráficos. De forma más simple es la parte gráfica y lo que ve el usuario.

Habiendo recordado lo que es un backend como frontend, se entiende la importancia del desarrollo de este trabajo, por lo que habiendo repasado también no hay que olvidar un backend este compuesto de un lenguaje de programación como una base de datos, listaremos alguno de los más importantes a continuación:

Bases de datos:

- Oracle
- Teradata,
- Microsoft SQL Server,
- IBM DB2.

Lenguajes de programación:

- Ruby
- Python
- JavaScript
- C#

Con la revisión de todos los conceptos antes mencionados vitales para entender el trabajo a realizar ahora daremos paso a la problemática.

## Problemática

Los objetivos principales a completar durante la solución de la problemática son:

- Comprender el funcionamiento del protocolo HTTP (sus verbos y estados).
- Comprender el funcionamiento de aplicaciones stateless, mecanismos asíncronos y funcionamiento REST.
- Diseñar e implementar un proyecto de software con requerimientos específicos.

Teniendo esto en cuenta damos paso a la problemática:

En el escenario supuesto de que la Universidad Tecnológica Metropolitana, está evaluando una forma de implementar un sistema de democracia digital (para consultas, paros, encuestas, etc). Para esto se le solicita construir un api rest que permita realizar esta tarea. El sistema debe contemplar lo siguiente.

### **Usuarios**

El sistema debe contener los usuarios de la UTEM, se necesita que el mecanismo de autenticación sea el provisto por Google (oauth 2.0), lo que garantiza la validez de la autenticación, es necesario además que exista un manejo de roles, para una personalización futura. Cada llamada a los servicios api, debe identificar al usuario y tener un access token, para validar la llamada contra los servicios de Google, esto se debe realizar usando una cabecera jwt.

### **Consultas.**

Se necesita registrar las consultas, los elementos mínimos necesarios son disponer de un token único que permita identificar esta consulta para la interacción de los sistemas y un nombre para identificar la consulta de forma humana y un flag que permita determinar si dicha consulta está activa o no. Por otro lado, cada opción asociada a la consulta debe tener un atributo numérico (que se usará como identificador de la opción) y un campo para el texto que describe la opción. En esta etapa se permite que la carga de las consultas y sus detalles, se realicen directamente a la capa de persistencia de datos (Por ejemplo, se permita la carga por un script sql).

### **Votos.**

Se debe poder registrar el voto, es necesario controlar que sólo se pueda realizar un voto por usuario y consulta, por otro lado, el diseño debe garantizar el anonimato de la elección.

### **API**

El api a construir debe tener algunas consideraciones generales para su implementación.

- El mecanismo de transferencia debe ser Json.
- Cualquier error (controlado o no) debe producir salidas en formato Json.
- Las operaciones deben estar autenticadas.

### **Autenticación.**

Se necesita una operación que permita la autenticación contra google y almacenar los datos necesarios para validar el token de acceso provisto por Google. Se recomienda que las operaciones de entrada, aquellas que interactúan con Google, están protegidas al menos por credencial/contraseña. La validación se realizará con un objeto jwt 1 que se incluirá en una cabecera "authorization".

### **Votación.**

Las operaciones mínimas necesarias, son:

#### */voter/polls*

Se requiere lo siguiente:

- La operación debe ser GET.
- Cabecera de autorización
  - Authorization: Bearer jwt.
- Se debe validar en base a este token (provisto por Google), el usuario y si su token está activo o no.
- La salida debe indicar aquellas encuestas con al menos la siguiente información:
  - token. Identificador de encuesta.
  - nombre. El nombre de la encuesta.
  - activo. Flag que indica si la encuesta está vigente o no.
  - opciones. Listado de opciones asociadas a la encuesta
- número de la opción.
- descripción de la opción.

#### */voter/vote*

Esta opción debe permitir registrar un voto, es importante que esto sea anónimo.

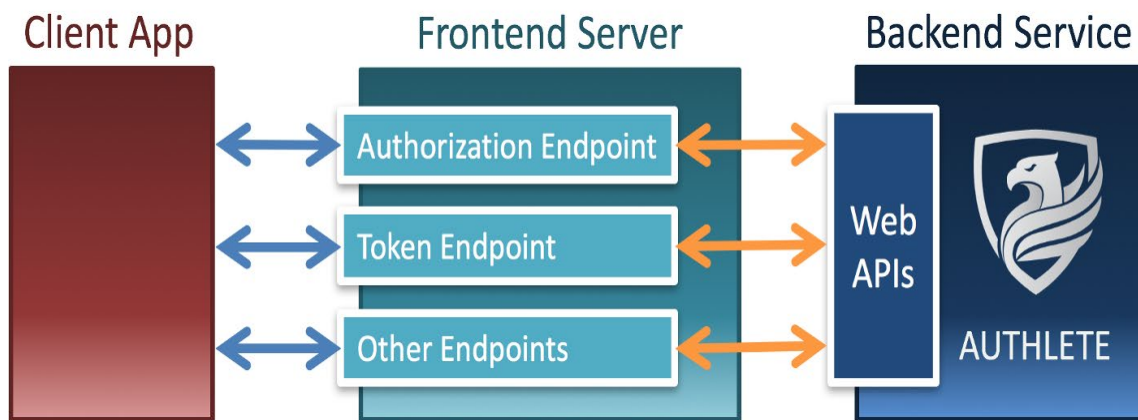
- La operación debe ser POST.
- Cabecera de autorización
  - Authorization: Bearer jwt.
- Se debe validar en base a este token (provisto por Google), el usuario y si su token está activo o no.
- El cuerpo de la petición debe contener:
  - Token de la encuesta.
  - El número de la selección.
- La salida debe responder con código 200 ó 201 y un objeto que indica que la operación fue exitosa.

/voter/{token}/results

Esta opción debe permitir desplegar los resultados del proceso.

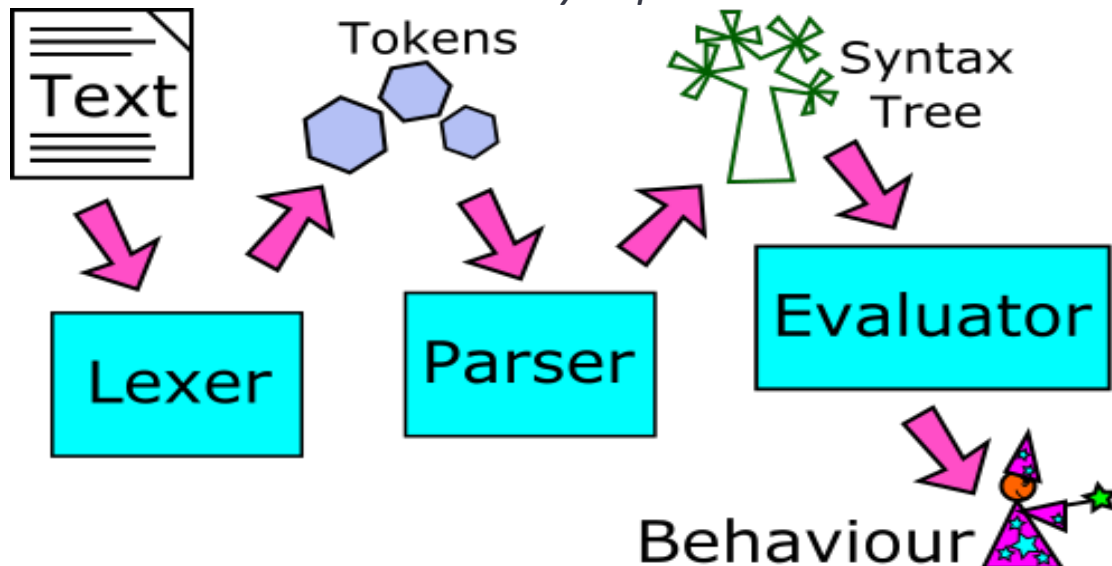
- La operación debe ser GET.
- Cabecera de autorización
  - Authorization: Bearer jwt.
  - Se debe validar en base a este token (provisto por Google), el usuario y si su token está activo o no.
- Cómo parte de la url se debe indicar el uso del token de la encuesta.
- La salida debe responder con un código 200 e indicar:
  - El nombre de la encuesta.
  - Un listado con el nombre de la opción y la cantidad de votos acumulados.

*Ilustración 1: Interacción entre backend, frontend y client*



*Fuente: <https://darutk.medium.com/>*

*Ilustración 2: Funcionamiento y comportamiento de un token*



*Fuente: <https://accu.org/>*

## Tecnología

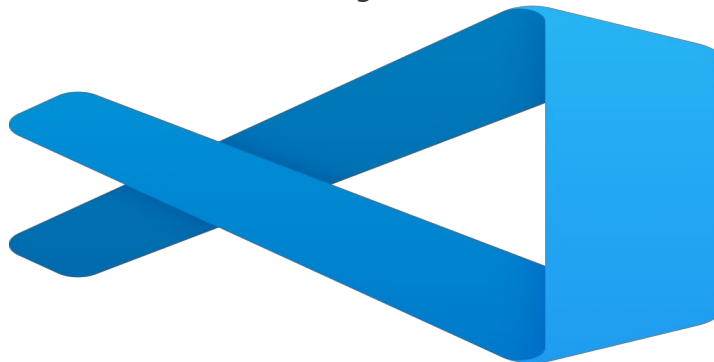
Para la solución de la problemática anteriormente mencionada y con el objetivo de satisfacer los puntos solicitados se listarán las tecnologías que se utilizarán en la realización de este proyecto y un breve resumen de su definición con sus funcionalidades:

- Visual Studio Code (VS Code): Es un editor de código fuente desarrollado por Microsoft. Es un software multiplataforma gratuito disponible para Windows, GNU/Linux y macOS. VS Code se integra bien con Git, admite la depuración de su código y viene con muchas extensiones para que pueda escribir y ejecutar código en cualquier lenguaje de programación.

Entre las principales características fue seleccionado para este trabajo por:

1. Detección de errores: Esto ayuda a encontrar errores en el código. Por lo tanto, no necesitamos encontrar errores línea por línea a simple vista. también el programa puede detectar de forma automática pequeños errores antes de ejecutar el código, evitando que tengamos que estar constantemente ejecutando buscando posibles errores.
2. Autocompletado: Esta función está relacionada con la edición de código, el autocompletado y el resaltado de sintaxis, lo que le permite ser más flexible al escribir código, lo que ayuda y permite mayor rapidez en la escritura del código, ya que el mismo programa arregla la sintaxis cuando no esta en su estructura correcta o el mismo lo resalta con alguna sugerencia, además de autocompletar en ciertas secciones cuando sabe exactamente lo que queremos realizar
3. Extensiones: Permiten personalizar y agregar funciones adicionales de forma modular y no relacionada. Por ejemplo, agregue nuevos temas al editor y conéctese a otros servicios para programar en diferentes idiomas. Las extensiones mejoran tu experiencia. Lo más importante es que la extensión se ejecuta en un proceso separado y no afecta el rendimiento del editor. Este es el punto mas fuerte de VS code, ya que a través de sus extensiones podemos descargar paquetes que nos permite trabajar con otros lenguajes, en este con un par de solicitudes se puede descargar las librerías y empezar a programar rápidamente como las de NodeJS.

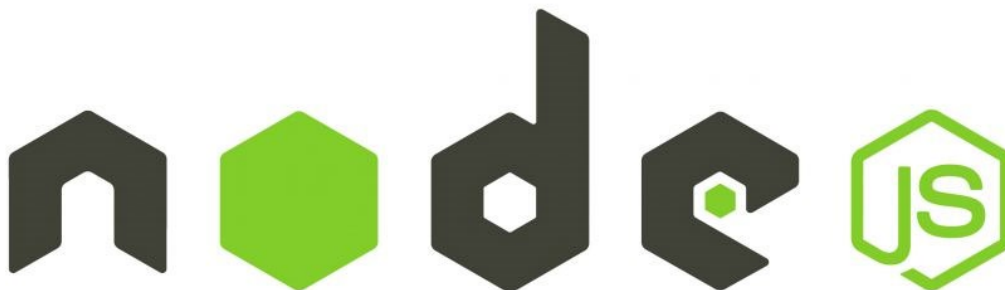
***Ilustración 03: Logo VS Code***



***Fuente: <https://snapcraft.io/code>***

- NodeJS: Permite ejecutar JavaScript fuera del navegador. Estos son importantísimos para JavaScript, ya que una de las desventajas de javascript es que todo está programado y testeado en navegadores. Node le permite usar JavaScript para realizar acciones como host local. Esto es exactamente lo que requieren lenguajes como Python, C++ o Java, ya que JavaScript ya no está ligado al navegador. Esto ha llevado al desarrollo de aplicaciones web isomórficas que ejecutan JavaScript en el navegador para interactuar con el cliente y JavaScript en el backend para procesar las solicitudes de los clientes. Debido a que es JavaScript en ambos dominios, existe la posibilidad de compartir el código entre ambos dominios, lo que reduce el costo total de mantenimiento y prueba de su aplicación. Además de que una de las características mas importantes y por las que fue seleccionado para este trabajo son:
  1. No es complejo de programar si ya se posee conocimientos en JS.
  2. La compilación es en tiempo de ejecución, esto logra que sea muy optimizado y con rápidos tiempos de respuesta
  3. Alto nivel de rendimiento producto a la ejecución en tiempo real
  4. Expansión de código mediante los módulos NPM.

***Ilustración 04: NodeJS***



***Fuente: <https://snapcraft.io/code>***



Los módulos NPM (Node Package Manager), como su nombre indica, gestiona las dependencias de sus proyectos de nodo. Por ejemplo, puede usar esto para instalar Bootstrap y usarlo en su proyecto. NPM usa archivos JSON para realizar un seguimiento de sus dependencias. Contiene información sobre bibliotecas instaladas, versiones, etc. Lo que nos facilita bastante la programación, ya que usaremos algunos módulos que nos ayudaran bastante, entre los principales estan:

1. express: Nos permite poder crear el servidor y manejar peticiones http.
2. bcryptjs: Para cifrar los datos del usuario para luego guardarla en la base de datos.
3. cors: Comunicar backend con otros servidores y facilitar las reglas de comunicación.
4. dotenv: Sirve para crear variables de entorno.
5. jsonwebtoken: sirve para autenticar usuarios en nuestros servidores, la autorización y también nos permite administrar los roles.
6. Morgan: Nos sirve para ver las peticiones del servidor.
7. Helmet: agrega mayor seguridad al servidor, evitando dar tanta información de que protocolos o versiones se estan usando y realizar validaciones sino las hemo hecho nosotros.
8. mongoose: nos permite hace la conexión con la base de datos, no es una base de datos como tal, sino que hace el puente con la base de datos que en este caso será mongodb.

#### ***Ilustración 05: Ejemplo de la instalación NPM de modulos***



```
Parikshit@DESKTOP-F3EF709 MINGW64 ~/Desktop/GeeksForGeeks
$ npm install express --save
npm WARN saveError ENOENT: no such file or directory, open 'C:\Users\Parikshit\Desktop\GeeksForGeeks\package.json'
npm WARN enoent ENOENT: no such file or directory, open 'C:\Users\Parikshit\Desktop\GeeksForGeeks\package.json'
npm WARN GeeksForGeeks No description
npm WARN GeeksForGeeks No repository field.
npm WARN GeeksForGeeks No README data
npm WARN GeeksForGeeks No license field.

+ express@4.16.3
added 49 packages from 47 contributors in 14.458s
```

***Fuente: <https://www.geeksforgeeks.org/>***

- **Mongodb:** Es una base de datos NoSQL orientada a documentos que se utiliza para el almacenamiento masivo de datos. En lugar de usar tablas y filas como las bases de datos relacionales tradicionales, MongoDB usa colecciones y documentos. Los documentos se componen de pares clave-valor, que son la unidad básica de datos.

Entre sus principales características destaca por:

1. Cada base de datos contiene colecciones que contienen documentos. Cada documento puede tener un número variable de campos. El tamaño y el contenido de cada documento pueden variar no necesariamente deben ser el mismo.
2. Las cadenas no necesitan tener un esquema predefinido. Además, los campos se pueden crear sobre la marcha.
3. Las relaciones jerárquicas, los arreglos de almacenamiento y otras estructuras más complejas se pueden representar más fácilmente utilizando los modelos de datos disponibles en MongoDB.
4. Mongo es más rápido para cargas de trabajo reales.

La rapidez con la que trabaja mongodb y por lo que fue seleccionado para este proyecto fue por su característica indexación ya que puede crear índices para mejorar el rendimiento de búsqueda en MongoDB. Se puede indexar cualquier campo en un documento MongoDB.

*Ilustración 06: Logo de mongoDB*



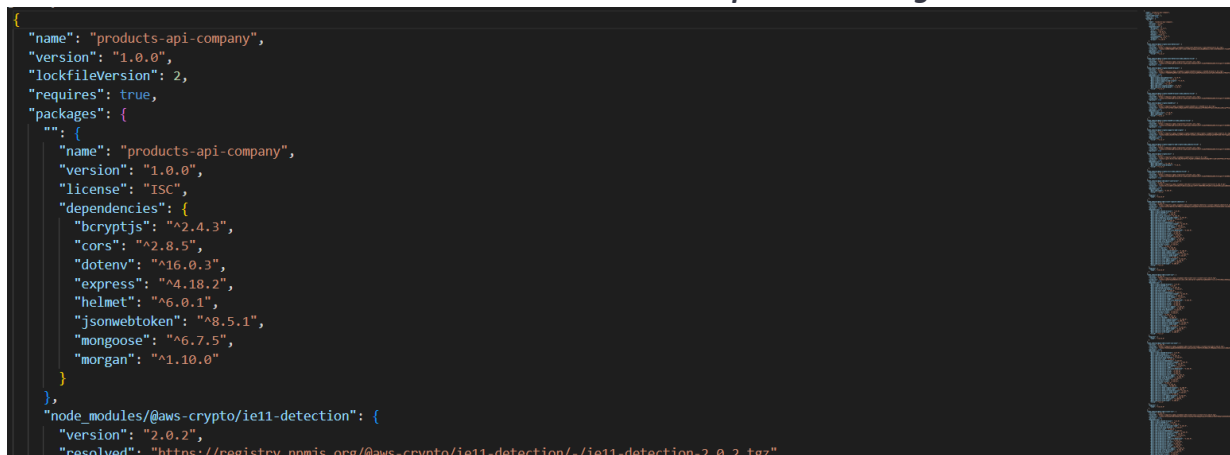
**Fuente:** <https://upload.wikimedia.org>

## Solución

Habiendo las explicado las principales tecnologías a utilizar ahora explicaremos el como se desarrollará esta solución, de forma resumida y un paso a paso.

1. Primero se instalará visual studio code.

**Ilustración 07: Foto de visual studio con parte del código**

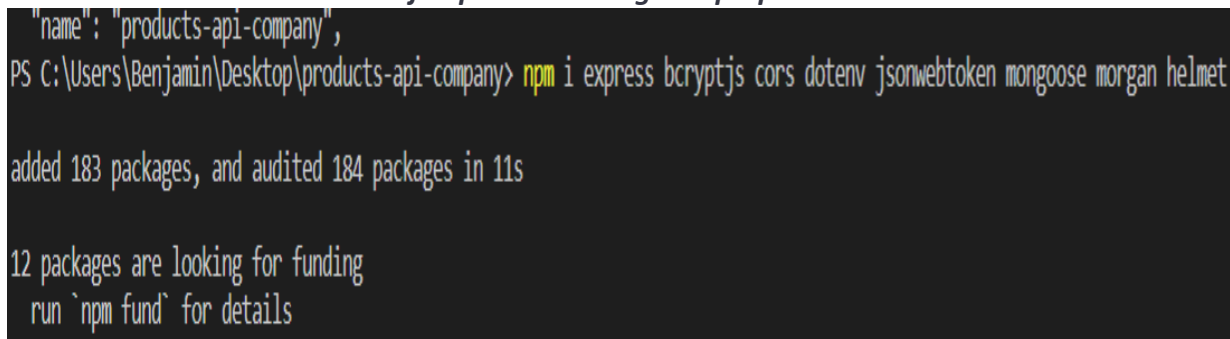
A screenshot of the Visual Studio Code editor with a dark theme. The main editor window displays a JSON file, which is a package.json. The code is as follows:

```
{
  "name": "products-api-company",
  "version": "1.0.0",
  "lockfileVersion": 2,
  "requires": true,
  "packages": {
    "": {
      "name": "products-api-company",
      "version": "1.0.0",
      "license": "ISC",
      "dependencies": {
        "bcryptjs": "^2.4.3",
        "cors": "^2.8.5",
        "dotenv": "^16.0.3",
        "express": "^4.18.2",
        "helmet": "^6.0.1",
        "jsonwebtoken": "^8.5.1",
        "mongoose": "^6.7.5",
        "morgan": "^1.10.0"
      }
    },
    "node_modules/@aws-crypto/ie11-detection": {
      "version": "2.0.2",
      "resolved": "https://registry.npmjs.org/@aws-crypto/ie11-detection/-/ie11-detection-2.0.2.tgz",
```

**Fuente: Elaboración propia**

2. Para utilizar el comando npm en visual studio code necesitamos las librerías de Python para que el comando sea reconocido por lo que procederemos también a su instalación.
3. Luego de ver que el comando funcione correctamente, se instalaran todos los módulos y otros adicionales en el futuro si se requieran.

**Ilustración 08: Ejemplo de descarga de paquetes de NPM**

A screenshot of a terminal window with a dark background. The prompt is 'PS C:\Users\Benjamin\Desktop\products-api-company>'. The command entered is 'npm i express bcryptjs cors dotenv jsonwebtoken mongoose morgan helmet'. The output shows 'added 183 packages, and audited 184 packages in 11s', followed by a message '12 packages are looking for funding' and a suggestion 'run `npm fund` for details'.

```
PS C:\Users\Benjamin\Desktop\products-api-company> npm i express bcryptjs cors dotenv jsonwebtoken mongoose morgan helmet

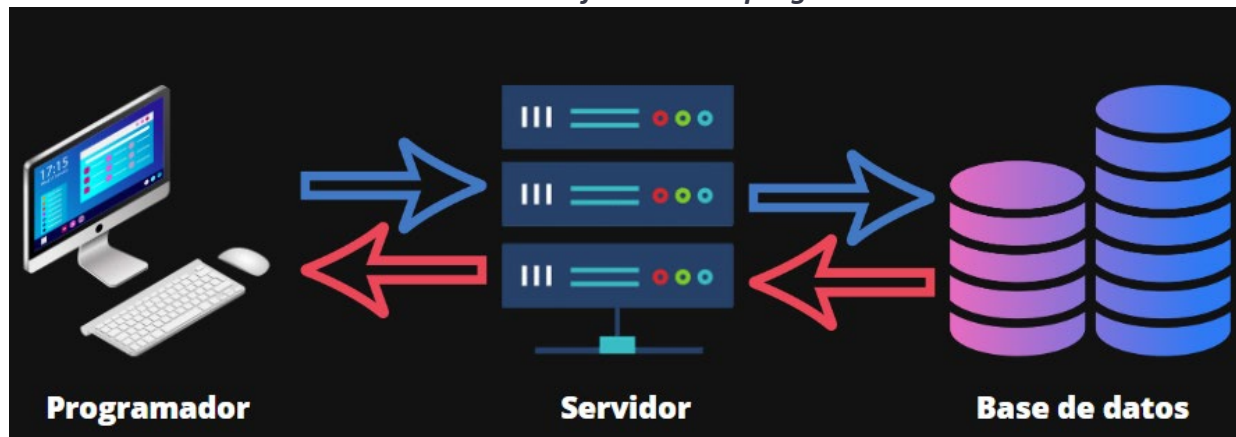
added 183 packages, and audited 184 packages in 11s

12 packages are looking for funding
  run `npm fund` for details
```

**Fuente: Elaboración propia**

4. Con lo visto anteriormente y la instalación de mongoDB, tenemos el núcleo del proyecto listo, ahora se estarán realizando las codificaciones correspondientes y montando el servidor.

***Ilustración 09: Modelo referencial de programación del servidor***



***Fuente: Elaboración propia***

Notas importantes: Todos los datos de los usuarios estarán encriptados como se dejó en claro en el apartado tecnológico.

Por otra parte, la problemática esta siendo solucionada con todos los materiales mencionados en el apartado tecnológico, ya que la programación del api rest con sus tokens se están realizando a través de nodejs y el almacenamiento de estos datos encriptados siendo guardados en la base de datos de mongodb, por lo que podemos decir que se están satisfaciendo todos los requerimientos dados por el problema.

## Conclusión

Con lo mostrado a lo largo de este informe que tuvo como objetivo demostrar los conocimientos presentados y como se puede llevar a cabo la solución mas adecuada a la problemática en cuestión, también cabe destacar que los conocimientos en backend son vitales para producir una solución satisfactoria, debido a que son la base de la construcción y diseño de la solución.

Por otro lado, con lo estudiado e investigación hecha también se ahondaron en conceptos como:

- Protocolos HTTPS.
- Funcionamientos Rest.
- Implementación de software.
- Sincronización con base de datos.
- Buenas costumbres de programación.

Para finalizar este trabajo es de un alto valor profesional, ya que nos nutre y prepara para un futuro laboral, ¿Por qué es esto? Ya que en el mundo real nosotros tendremos que buscar soluciones optimas a las problemáticas que nos vayan asignado, esto requerirá de forma muy resumida de:

- Satisfacer los objetivos de la problemática.
- Generar planes de acción.
- Contribuir mutuamente en el equipo.
- Generación de ideas.
- Buscar los métodos más eficientes.

Todo lo mencionado en los puntos anteriores son cubiertos por el trabajo dado, haciendo que este sea de gran importancia para nuestro futuro y el trabajo en equipo.

***Ilustración 10: Simbolismo del trabajo en equipo***



***Fuente: <https://i.ytimg.com>***

## Bibliografía

01-"Frontend y backend: Qué son, en qué se diferencian y ejemplos". Blog de HubSpot | Marketing, Ventas, Servicio al Cliente y Sitio Web. <https://blog.hubspot.es/website/frontend-y-backend#:~:text=Por%20sus%20aplicaciones%20y%20características,programación%20de%20sus%20funcionalidades%20principales> (accedido el 5 de diciembre de 2022).

02-<https://assemblerinstitute.com/blog/backend-vs-frontend#:~:text=En%20definitiva,%20un%20desarrollador%20Frontend,la%20web%20a%20diferentes%20dispositivos> (accedido el 5 de diciembre de 2022).

03-"Los 5 lenguajes de programación para aplicaciones web más usados". Universidad Anáhuac Mérida. <https://merida.anahuac.mx/posgrado/blog/lenguajes-programacion-aplicaciones-web-mas-usados> (accedido el 5 de diciembre de 2022).

04-"Programación paralela". Ferestrepoca.GitHub.io by ferestrepoca. [https://ferestrepoca.github.io/paradigmas-de-programacion/paralela/paralela\\_teoría/index.html](https://ferestrepoca.github.io/paradigmas-de-programacion/paralela/paralela_teoría/index.html) (accedido el 5 de diciembre de 2022).

05-Linda. "5 ways to fix the npm install not working issue". MiniTool. <https://www.partitionwizard.com/partitionmanager/npm-install-not-working.html> (accedido el 5 de diciembre de 2022).

06-"Node.js". Node.js. <https://nodejs.org/en/> (accedido el 5 de diciembre de 2022).

07-Microsoft. "Documentation for visual studio code". Visual Studio Code - Code Editing. Redefined. <https://code.visualstudio.com/docs/?dv=win> (accedido el 5 de diciembre de 2022).

08-<https://i.ytimg.com/vi/vUhIMsV0nAI/hqdefault.jpg> (accedido el 5 de diciembre de 2022).

09-T. Kawasaki. "New architecture of oauth 2.0 and openid connect implementation". Medium. <https://darutk.medium.com/new-architecture-of-oauth-2-0-and-openid-connect-implementation-18f408f9338d> (accedido el 5 de diciembre de 2022).

10-"How to write a programming language: Part 2, the parser". ACCU. [https://accu.org/journals/overload/26/146/balaam\\_2532/](https://accu.org/journals/overload/26/146/balaam_2532/) (accedido el 5 de diciembre de 2022).

11-" "Install code on windows | snap store". Snapcraft. <https://snapcraft.io/code> (accedido el 5 de diciembre de 2022).

12-"¿Qué es y cómo instalar visual studio code?" KeepCoding Tech School. <https://keepcoding.io/blog/que-es-y-como-instalar-visual-studio-code/> (accedido el 5 de diciembre de 2022).

13-"Qué es Node.js y para qué sirve". Blog de LucusHost.  
<https://www.lucushost.com/blog/que-es-node-js/> (accedido el 6 de diciembre de 2022).

14-"What is node.js? - devteam.space". DevTeam.Space.  
<https://www.devteam.space/blog/what-is-node-js/> (accedido el 6 de diciembre de 2022).

15-<https://media.geeksforgeeks.org/wp-content/uploads/npm-install-express-save.png>  
(accedido el 6 de diciembre de 2022).

16-"¿Qué Es MongoDB?" MongoDB. <https://www.mongodb.com/es/what-is-mongodb>  
(accedido el 6 de diciembre de 2022).

17-"Qué es MongoDB y características". OpenWebinars.net.  
<https://openwebinars.net/blog/que-es-mongodb/> (accedido el 6 de diciembre de 2022).

18- Wikimedia Commons.  
[https://upload.wikimedia.org/wikipedia/commons/thumb/9/93/MongoDB\\_Logo.svg/2560px-MongoDB\\_Logo.svg.png](https://upload.wikimedia.org/wikipedia/commons/thumb/9/93/MongoDB_Logo.svg/2560px-MongoDB_Logo.svg.png) (accedido el 6 de diciembre de 2022).

19-"La importancia de un buen backend". Codize. <https://www.codize.ar/blog/codize-blog-1/la-importancia-de-un-buen-backend-45> (accedido el 6 de diciembre de 2022).

20-"Importancia del desarrollo web back-end - Sustenta Web". Sustenta Web.  
<https://sustentaweb.cl/importancia-del-desarrollo-web-back-end/> (accedido el 6 de diciembre de 2022).