**Medical Sensors**

The main database is created by using **MongoDB,** which is a general purpose, document-based, distributed database platform built for modern application developers and for the cloud.

We use MongoDB to store the data of users in order to manage their privileges and help with authentication.

To interact with the database we use **REST API (RESTful API)**.

An API for a website is code that allows two software programs to communicate with each other. The API spells out the proper way for a developer to write a program requesting services from an operating system or other application.

A RESTful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data. That data can be used to GET, PUT, POST and DELETE data types, which refers to the reading, updating, creating and deleting of operations concerning resources.

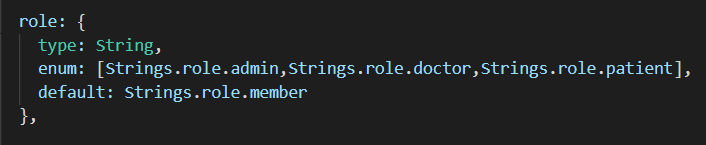
We use **node.js** server to make the app available to serve HTTP requests. It provides the interaction between users and your application.

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Using Visual Studio Code which is a streamlined code editor with support for development operations like debugging, task running, and version control, we create node.js models in the database containing the main information of the users and devices and then we use this models to create services node.js file which controls the operations that these data is used for , such as : creating users , logging in , creating devices…etc.

Model examples from the used MongoDB Database:

* We define an object for the possible roles which are Admin, Doctor or Patient while assigning the default role as member.



- We create an array containing the names of all departments in both the user and the device models.

department: [{

    type: String,

    enum: ["Surgeon", "Veterinarian","Neurologist","Pathologist","Psychiatrist","Therapist","Nurse", "Otorhinolaryngologist",

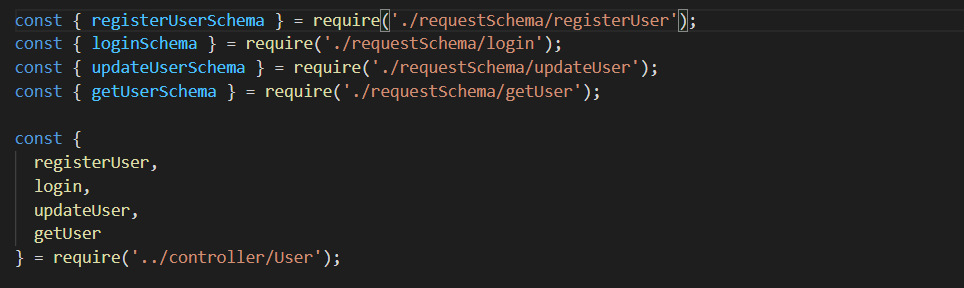
    "Cardiologist","Pulmonologist","Pediatrician","Orthopedist ","Dentist","Gynaecologist","Ophthalmologist"],

  }],

* We also create some objects to store the email, password and ID of the users.



To manage API requests through the router we create a router.js file to handle all the request of registering, logging in, updating and getting users:



Each type of these requests has its distinctive file to define its parameters for example the register request file defines the required information for regeistreatiun such as name. email and password:

