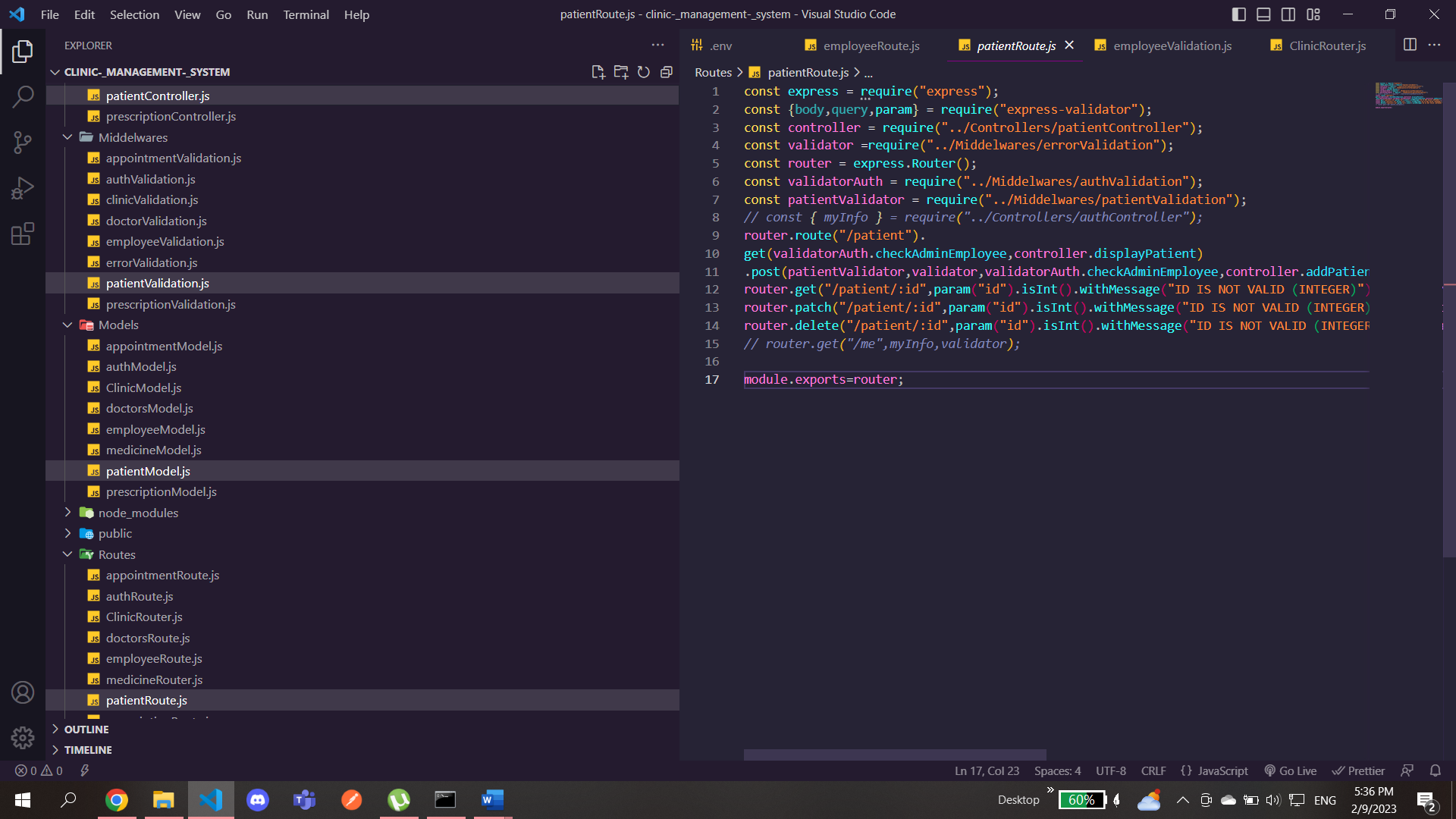
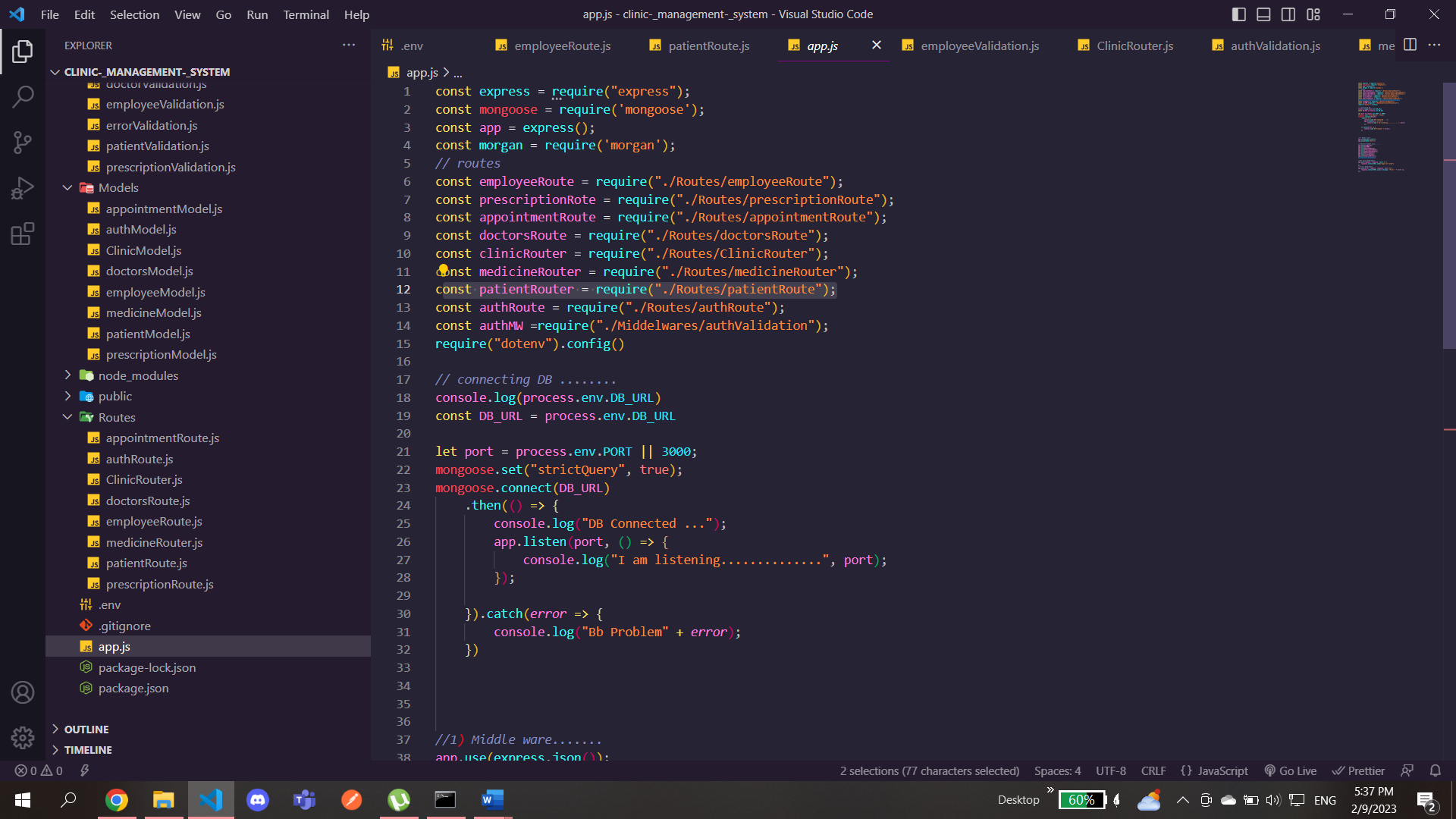
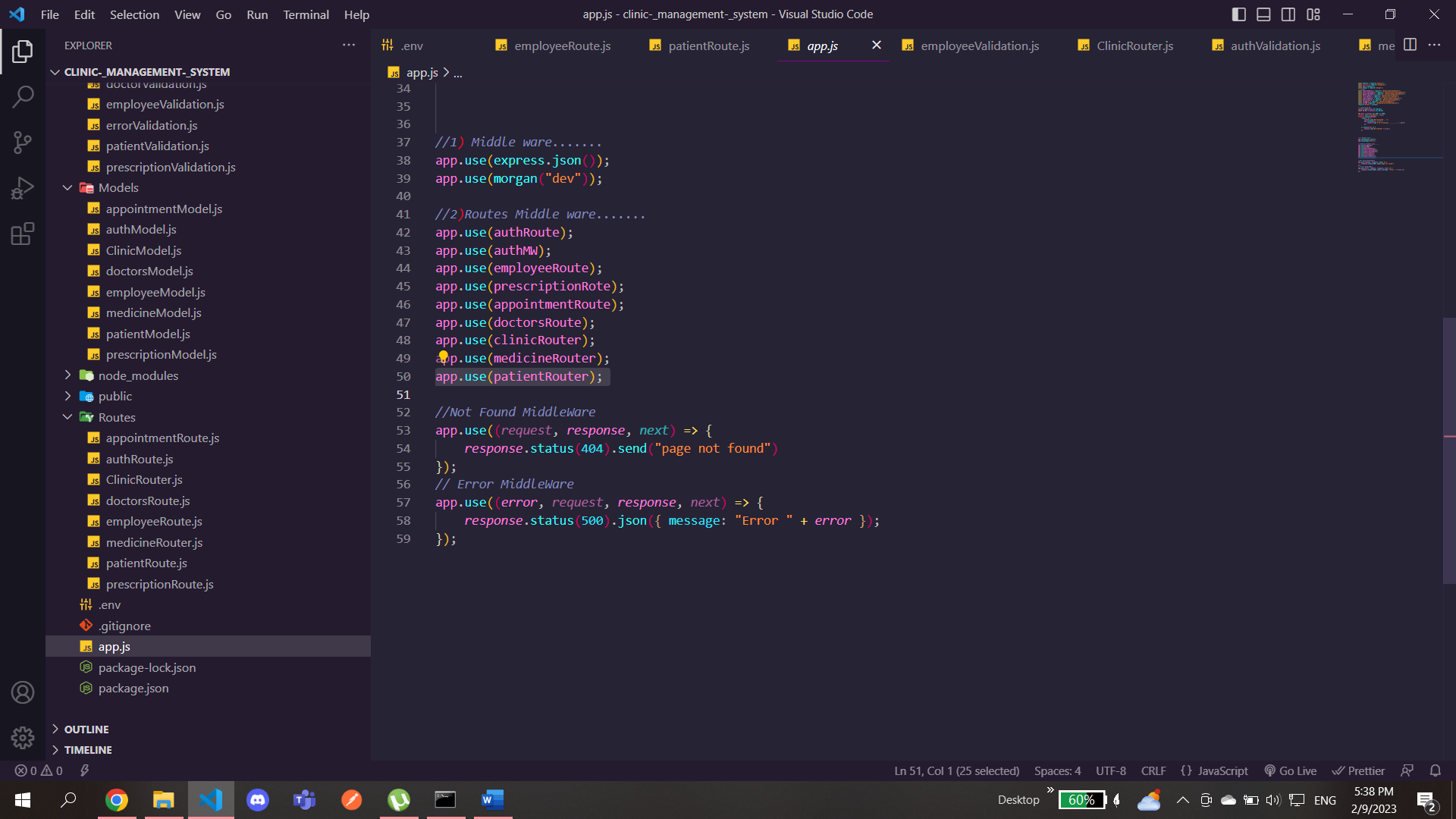
Patient

First step is the creation of the whole process files including the schema, controller, middleware and route

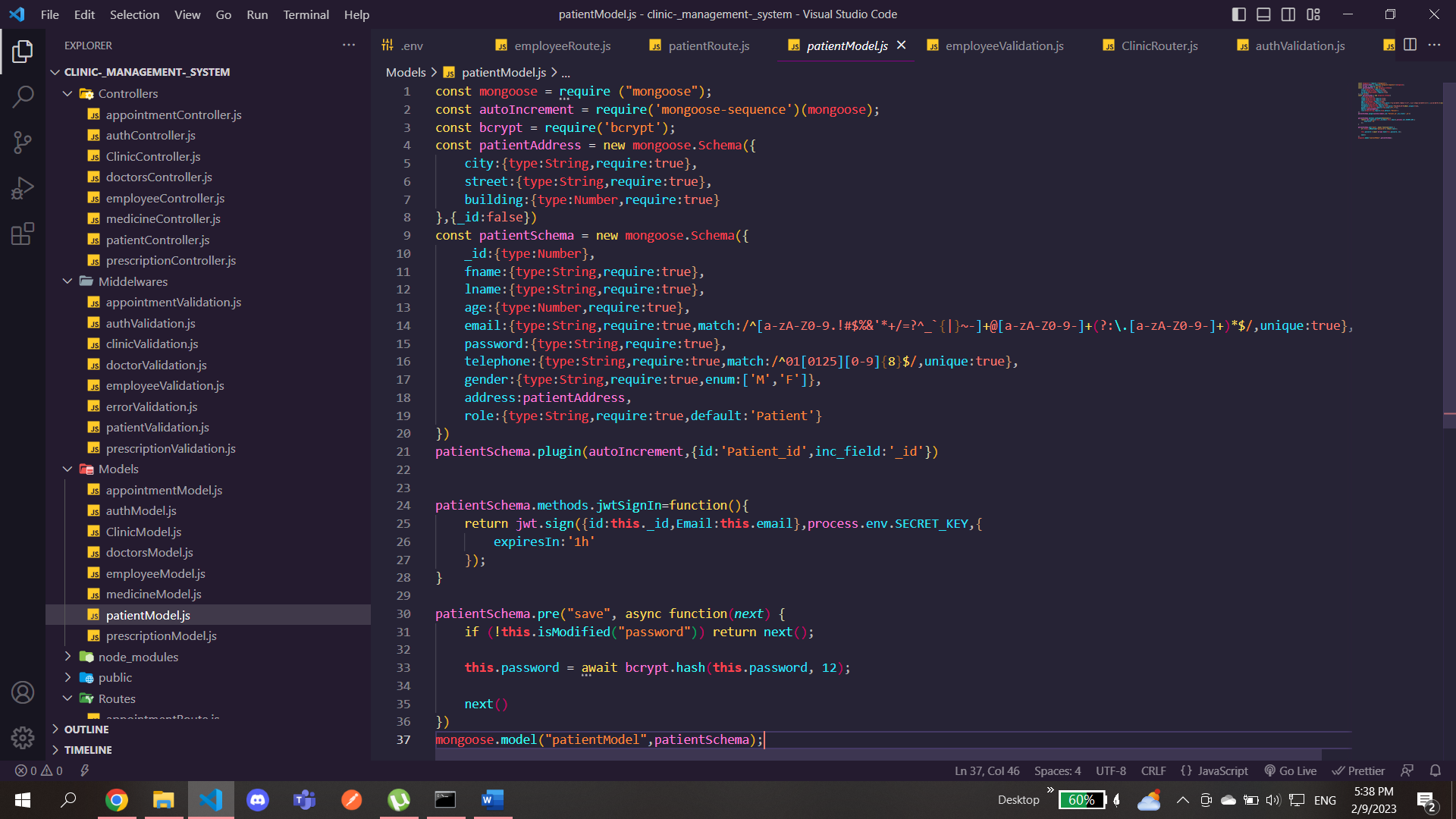


Don’t forget to import it in the app folder and make the server use it

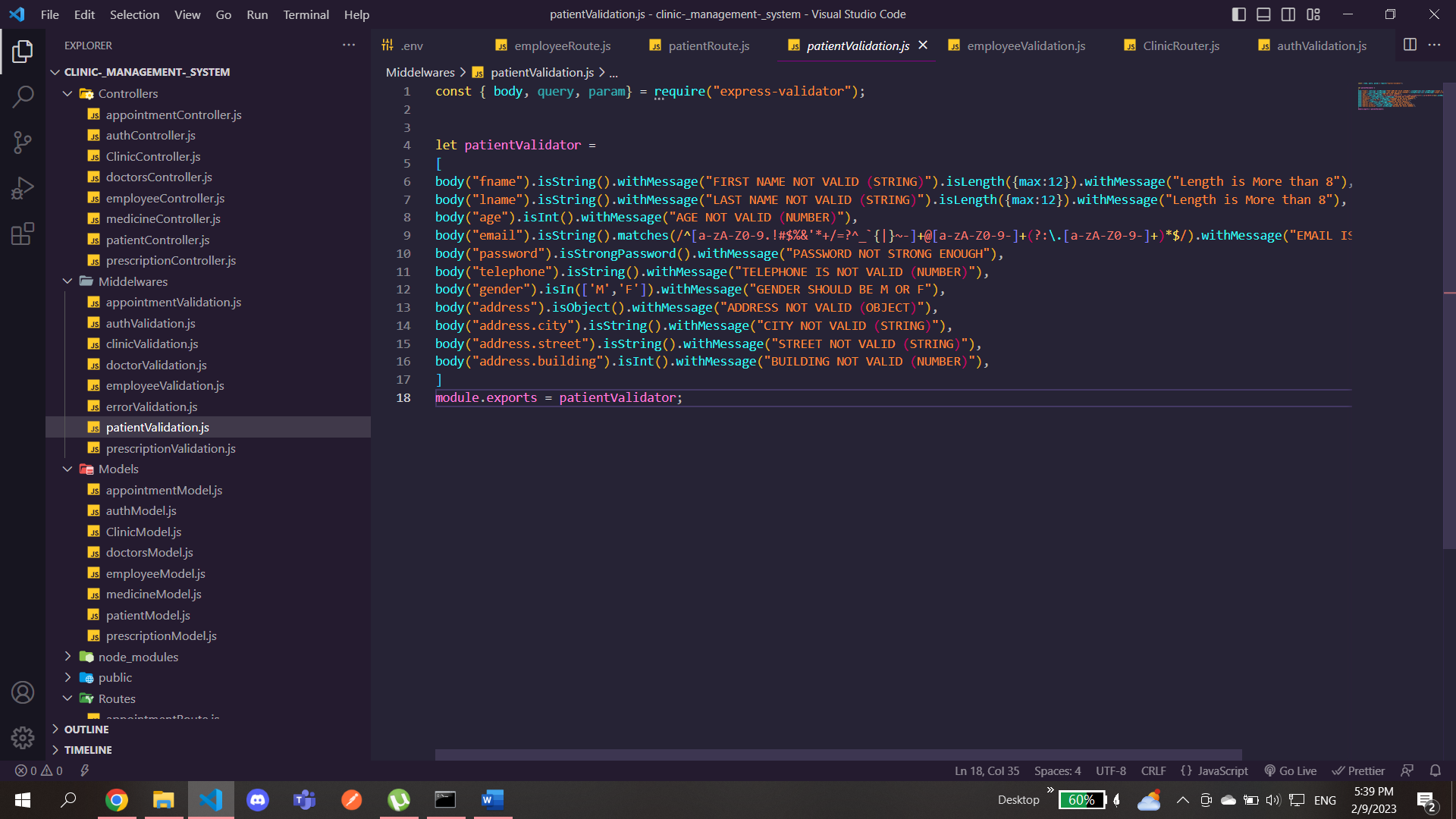




We create the schema for the patient with its fields and also included bcrypt package so that every patient is created his/her password is hashed

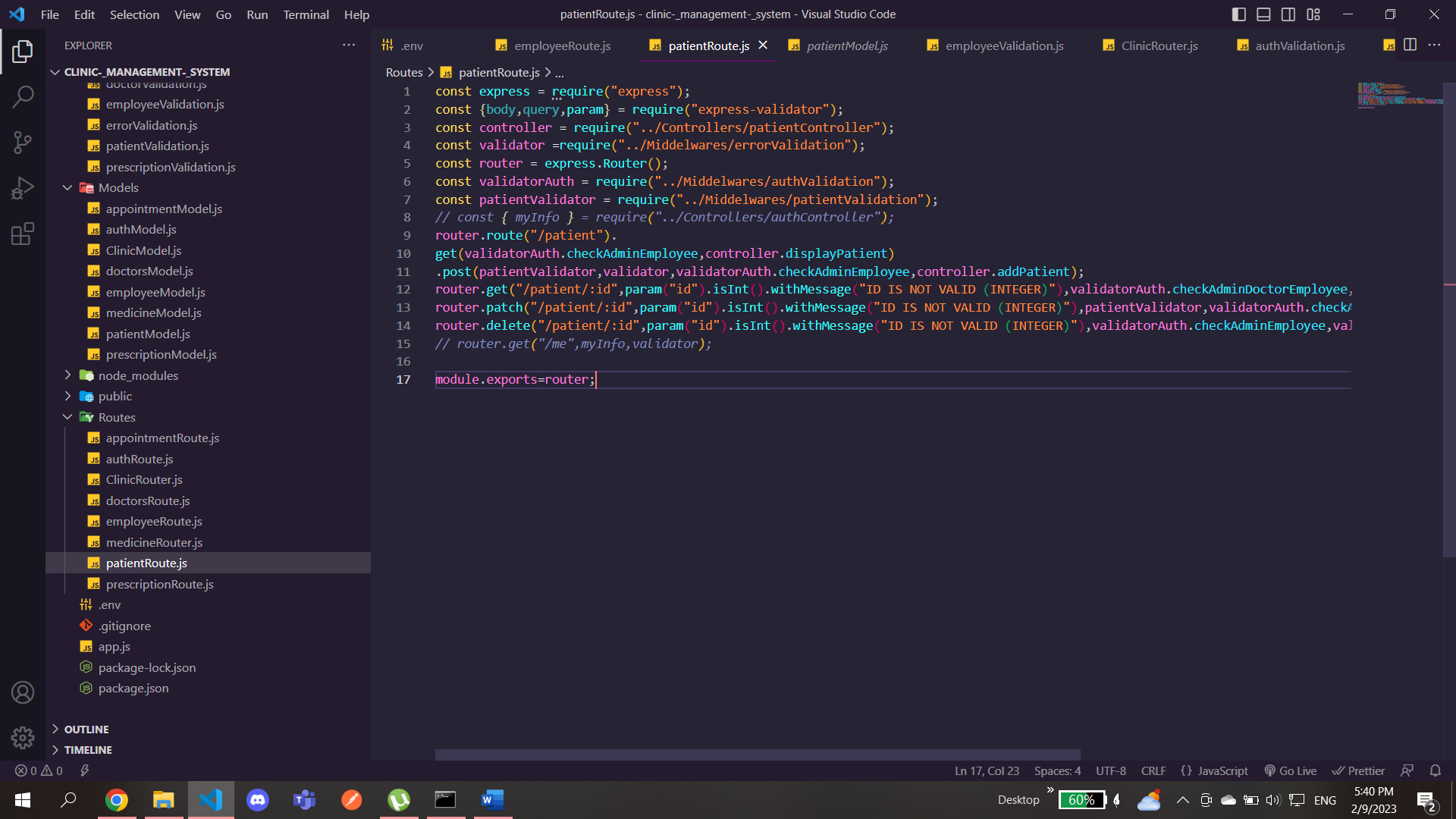


Here you can see the validation of the patient which is then exported to the router file

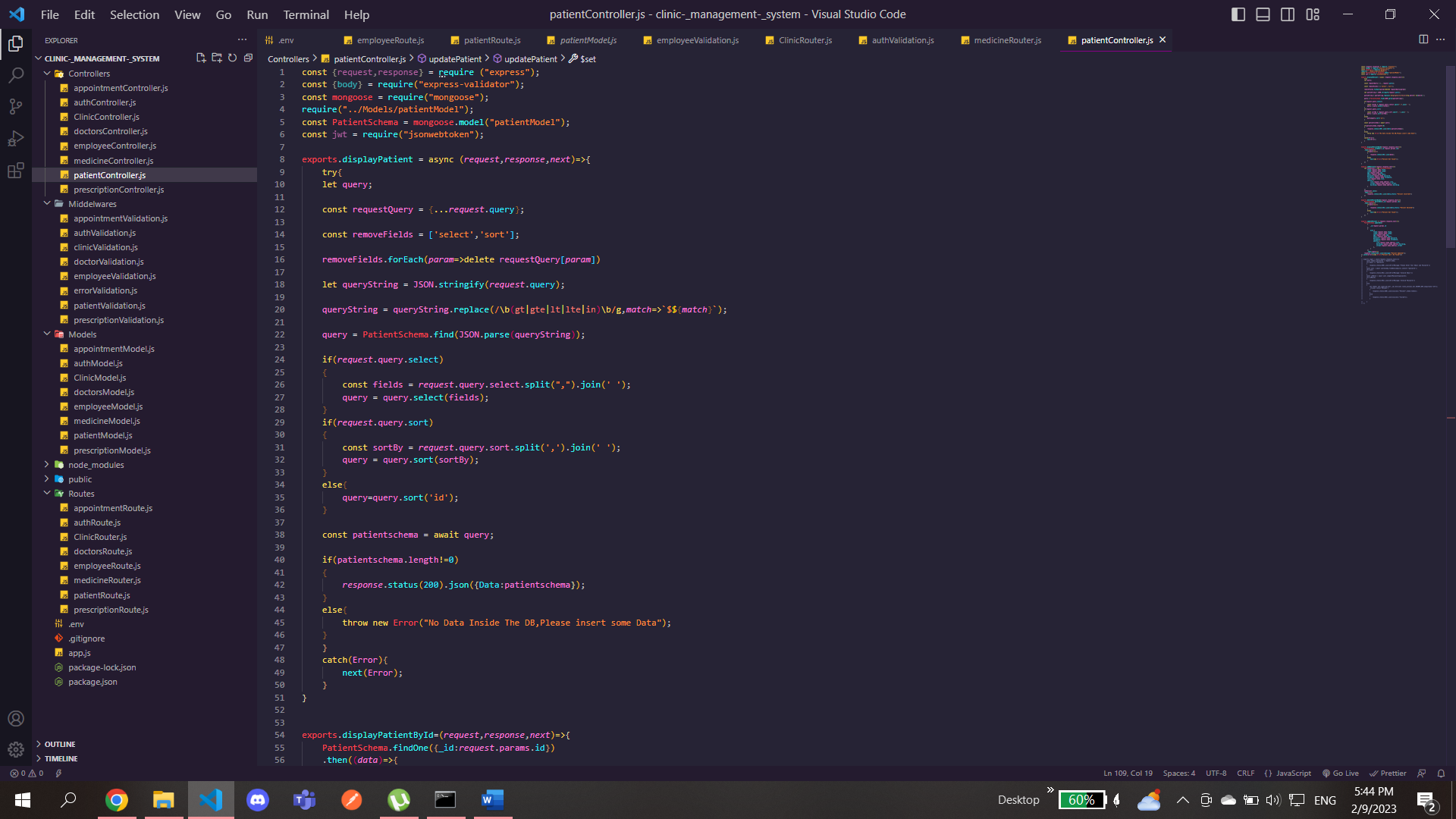


Here is the route file for the patient we imported the middleware file(validation)

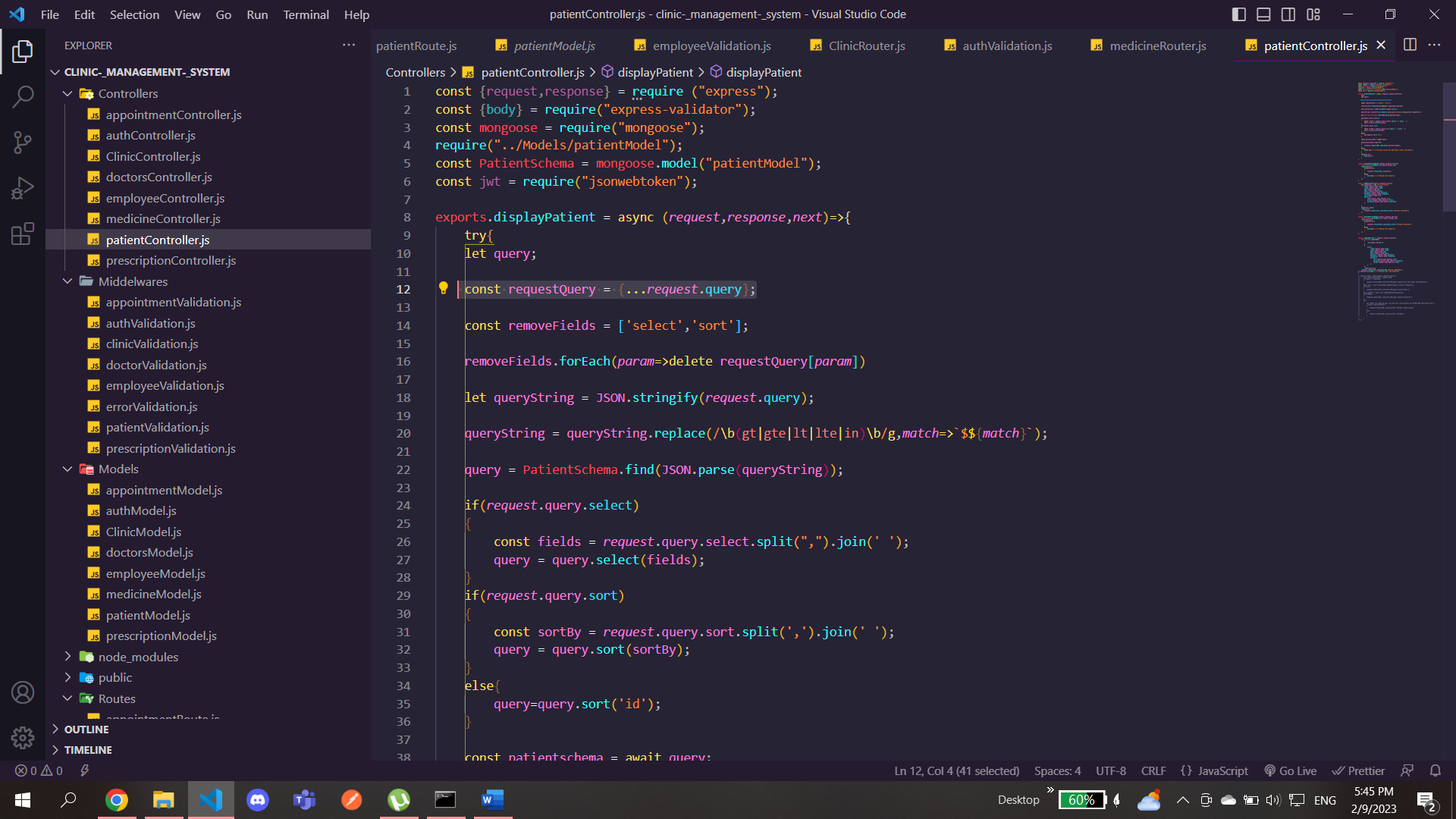
So we can use it before every get, post, or patch so that the data is not sent unvalidated



For the display method in the patient controller we also included a sort and filter methods



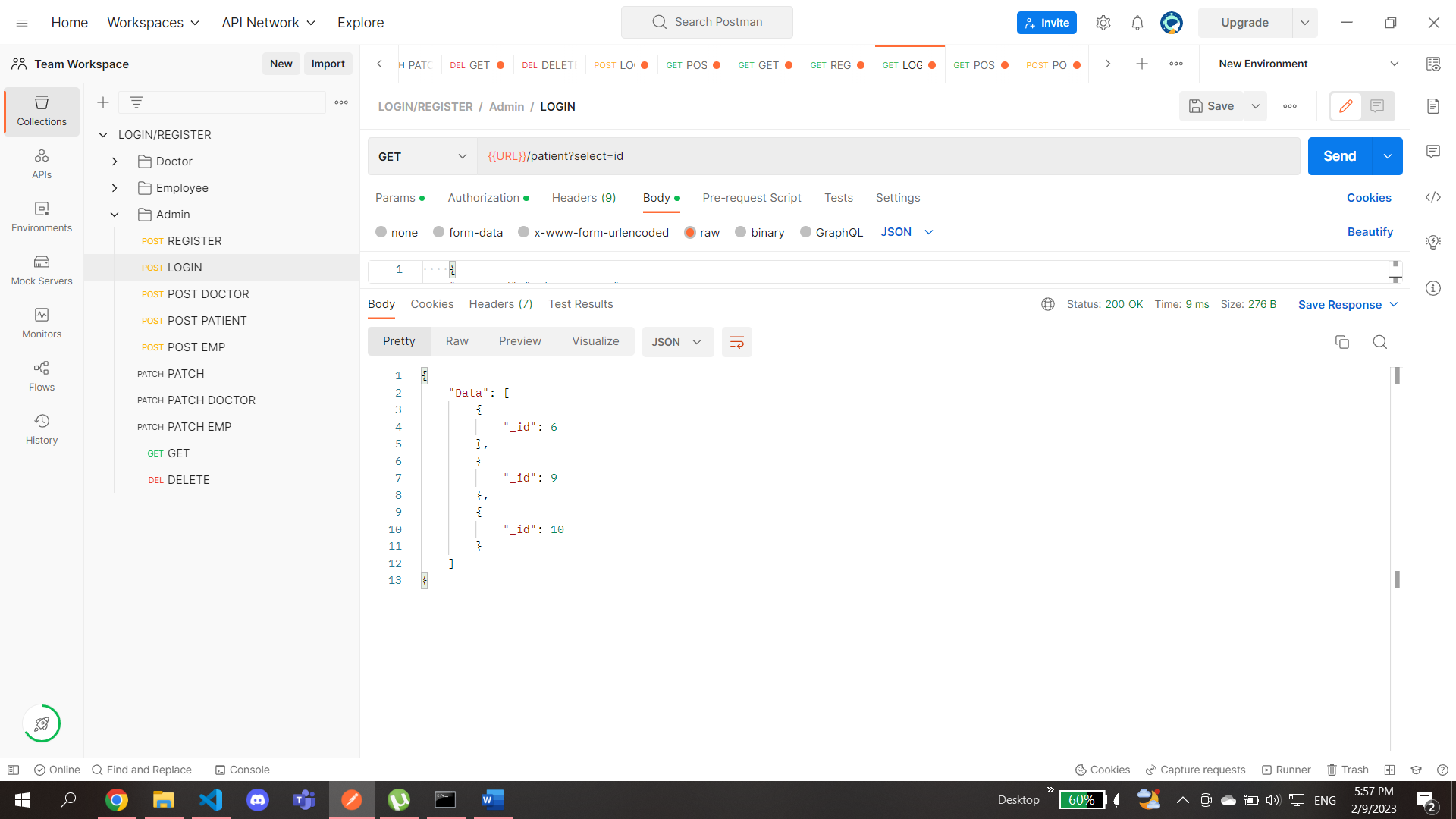
Here in the requestQuery a spread operator is used so we avoid the limitations of the length of the query whether we like to include like greater than a certain number and also between two numbers so the options are opened for typing



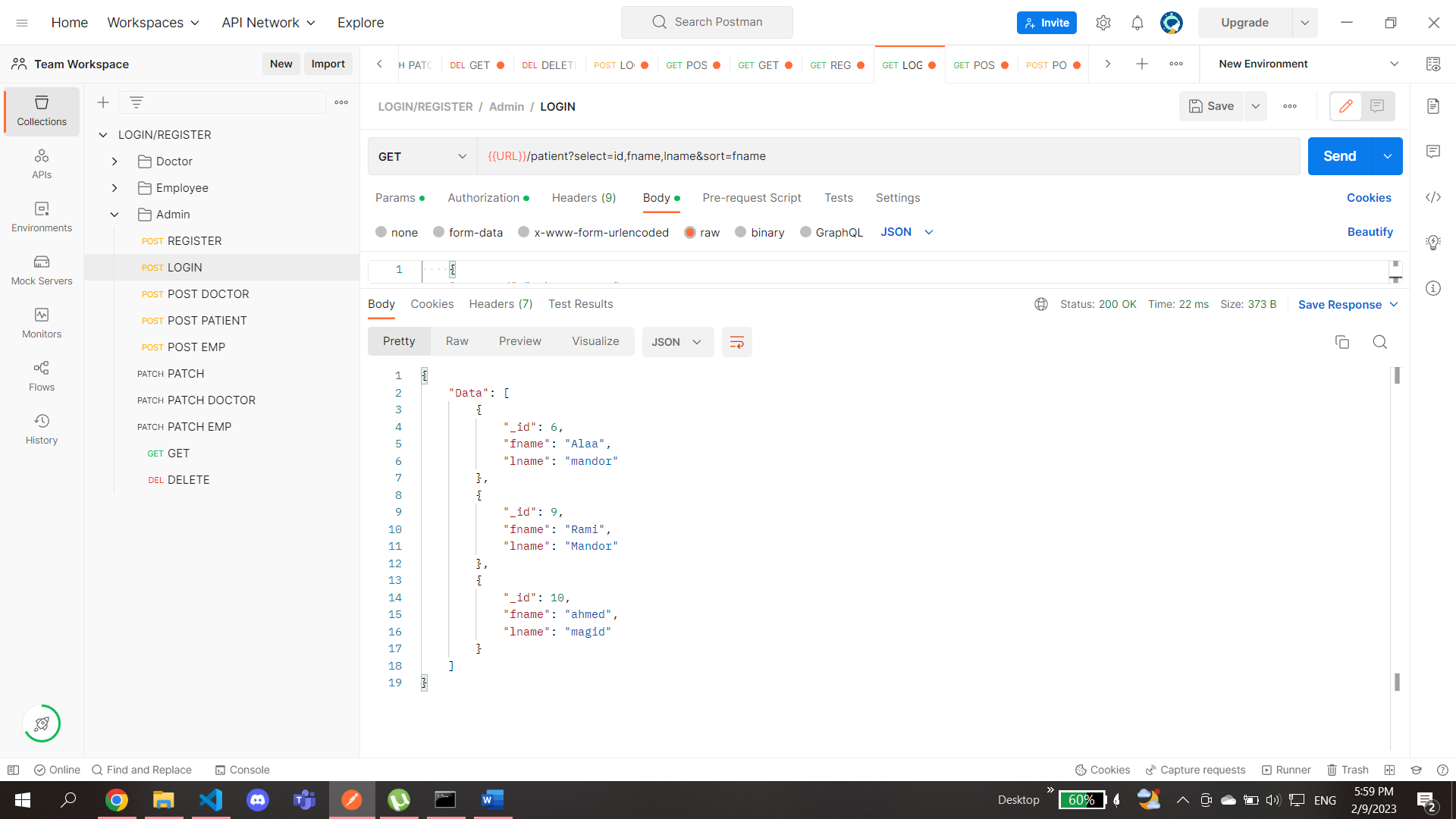
We create two options one is sort and the other is select by select it means filtering you select only to display the data by first name or last name only or by both and so on

In the queryString variable we used replace method to replace any gt ,gte , lt in the query with the same previous expressions plust a $ before it so it fits the aggregation method you use to display a certain data based on it

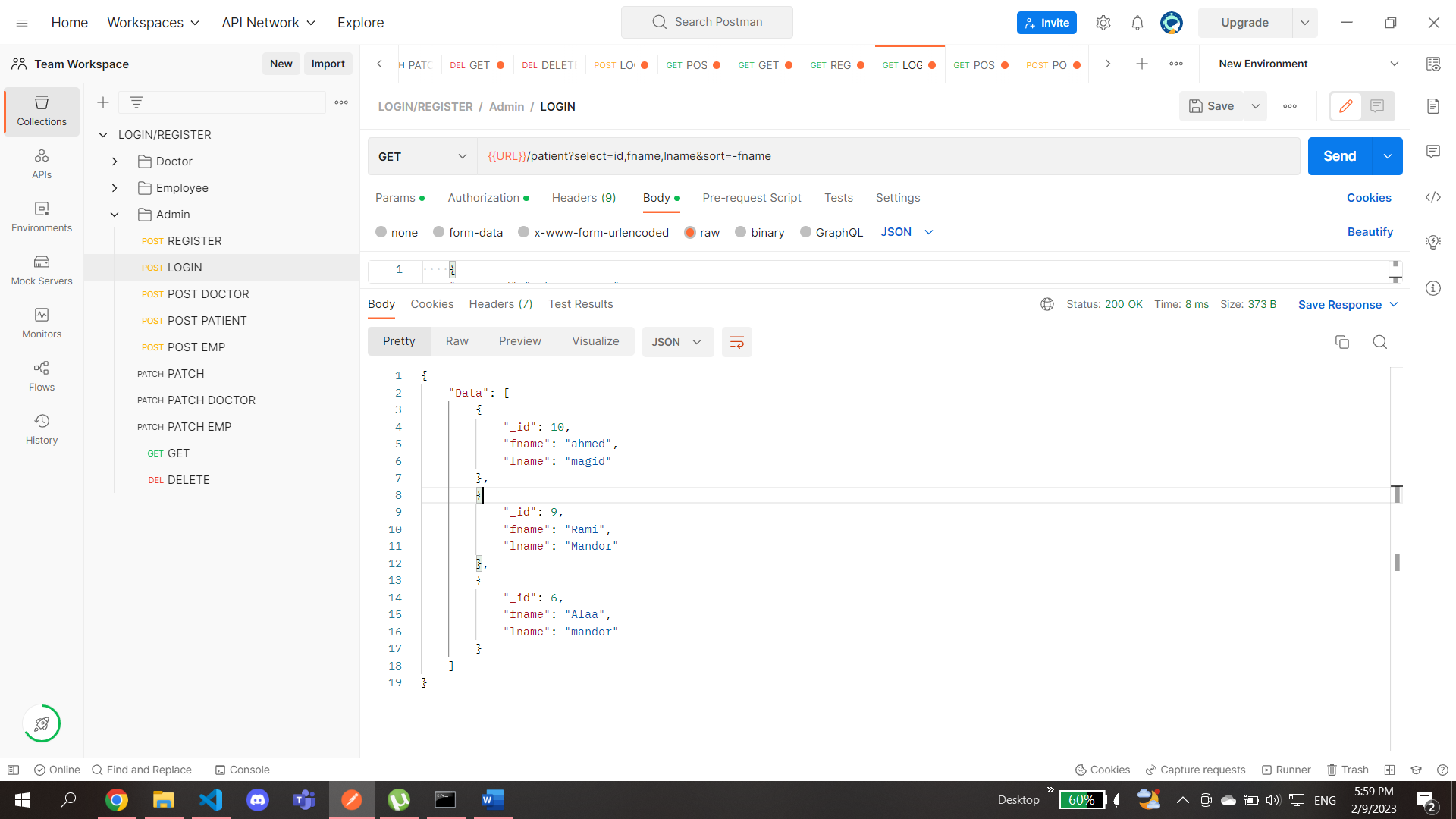
Of course we stringfiy it first so we can handle it and use the replace method then parse it so we can deal with it in the javascript or vscode



Here we selected only the id to be displayed



Here we only selected the fname and lname and id and sorted it by fname



We can also sort it in descending order by just typing – ve before the name of the sort