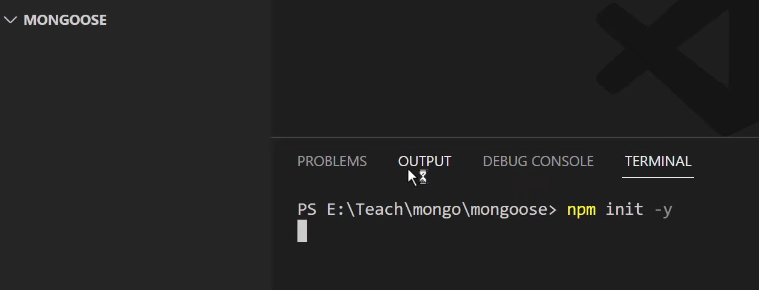
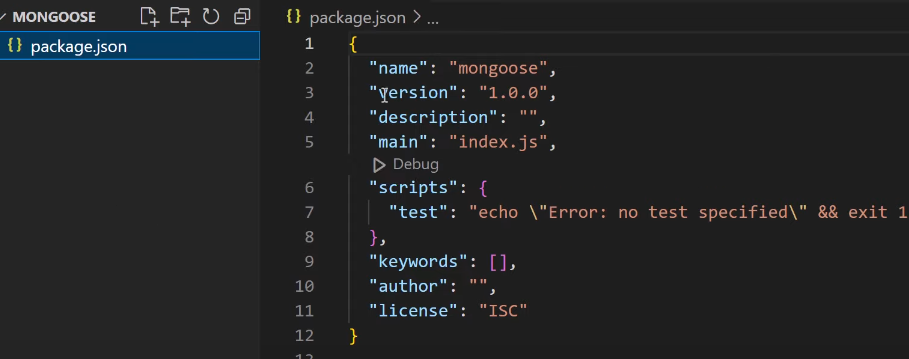
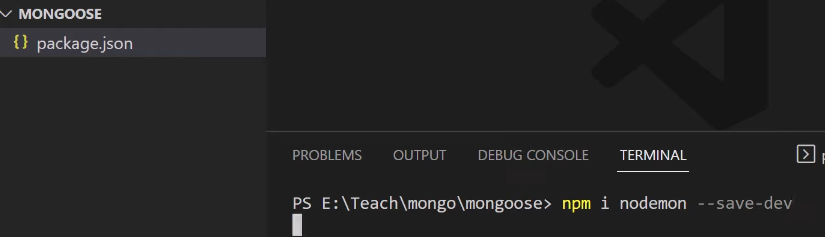
Create folder, type cmd in address bar, code .



npm init : Initialize a new project and create a package. json file. npm install <package-name> : Install a package and add it to the dependencies object in your package. json file.

Y meaning : Yes  




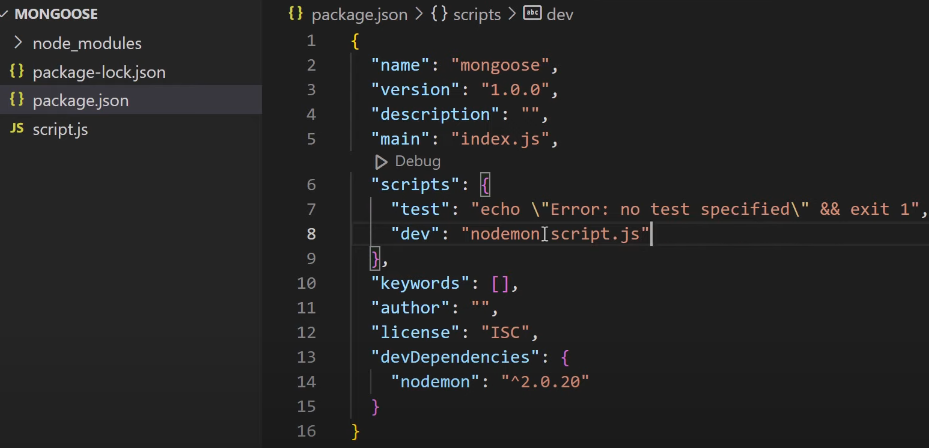


--save is default use (npm install, it will work after clone in github)

--save-dev is for development

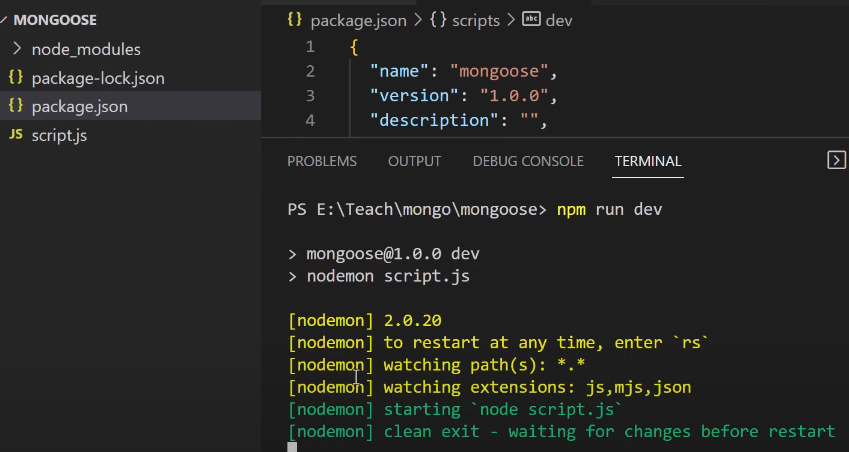
When you use the --save-dev flag, the package is added to your devDependencies object.

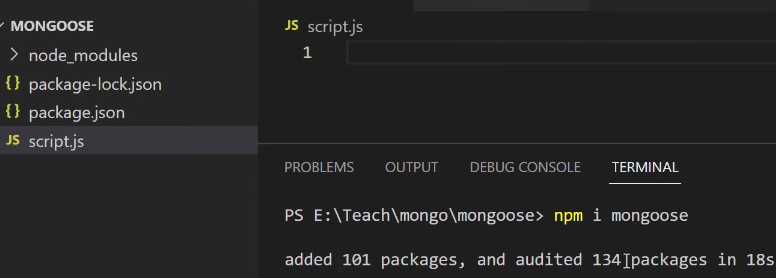
if you download a repo from github and type npm install, the devDependencies are ignored



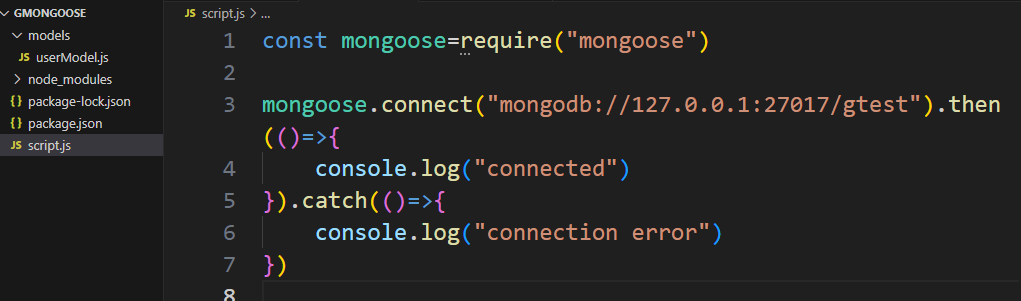
Don’t run every time: node script.js

Run only one time : npm run dev (it will take automatically refresh when saved)





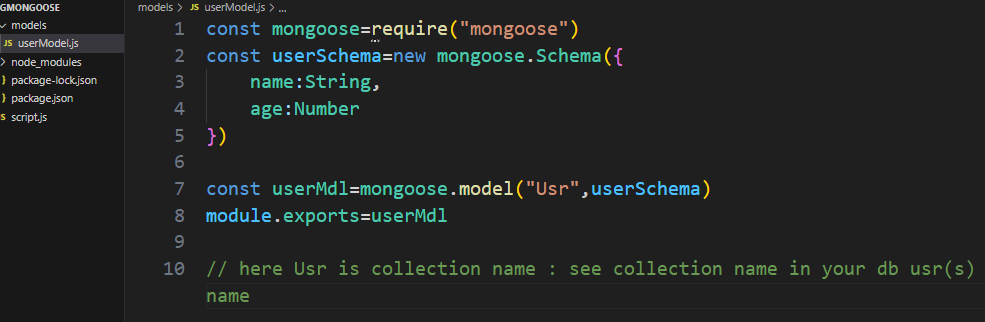
Terminal hide/show shortcut :ctrl +~



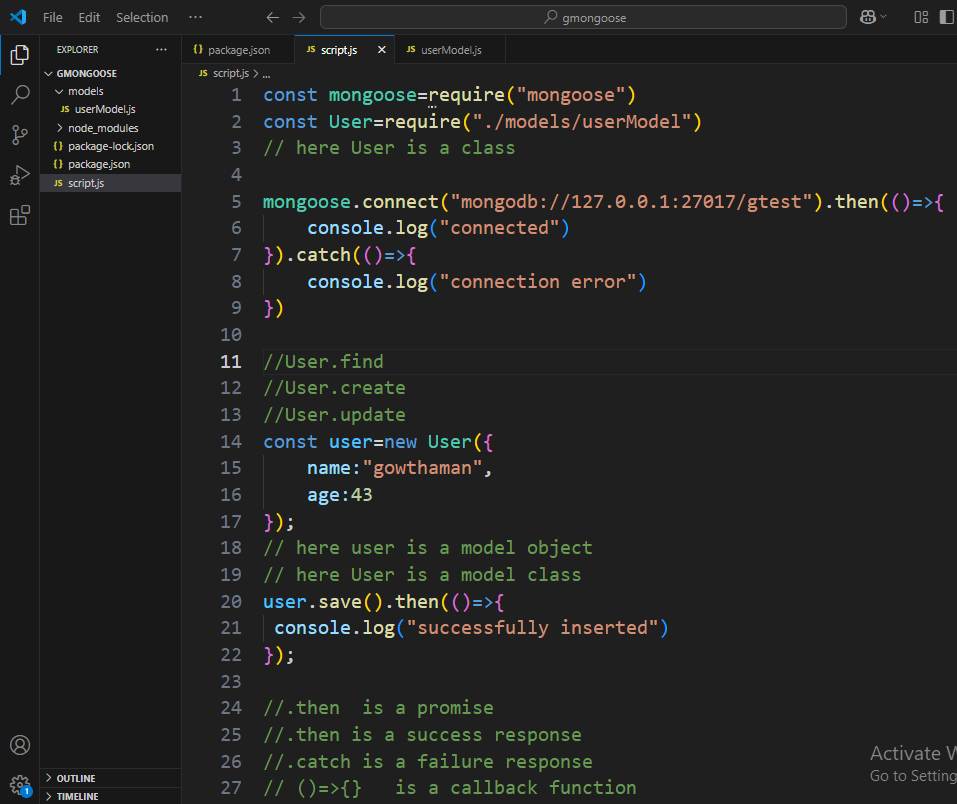
Schema : inbuilt class of mongoose

Use: only allow to store schema defined keys and datatype.

userSchemea: userDefined object.



Insert method 1: (with out using async await)



Do not change name & age key. because schema defined.

Do not add extra field. Only name and age accept

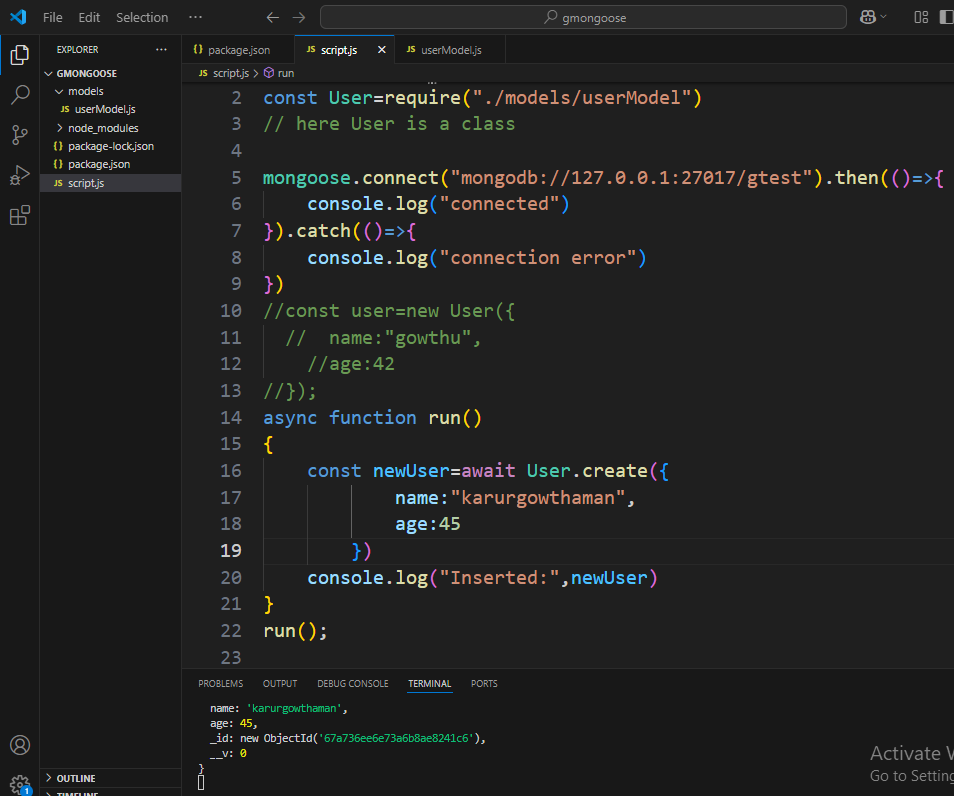
Same insert method using async function because promise return (.then)

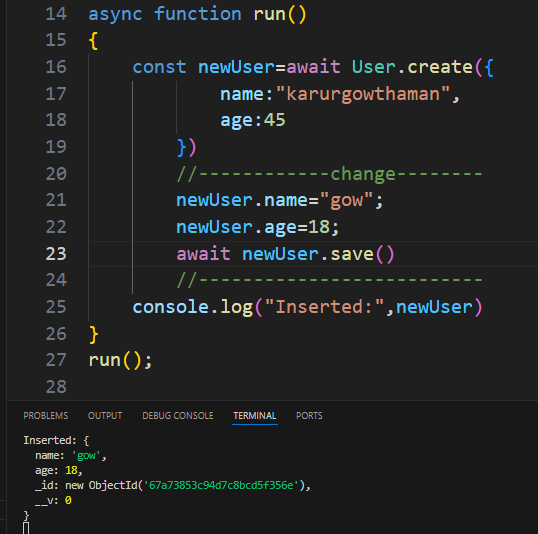
Method 2: insertion using save with async function



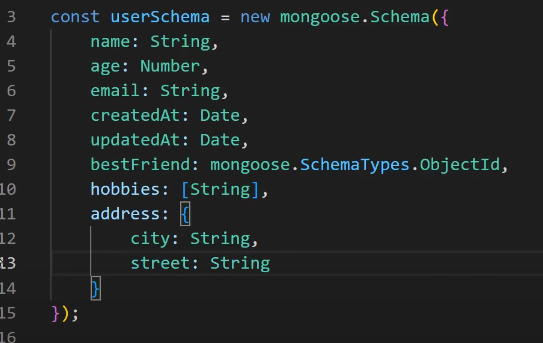
Method:3 insertion (without using save method)

Only using create

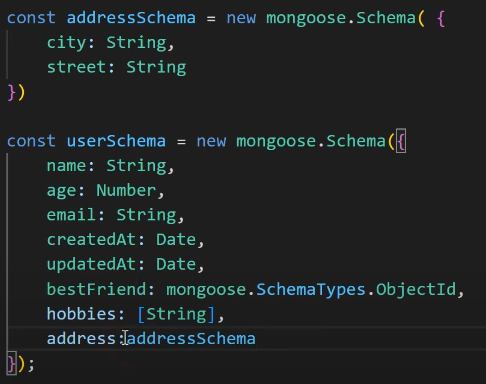




**Schema types:**

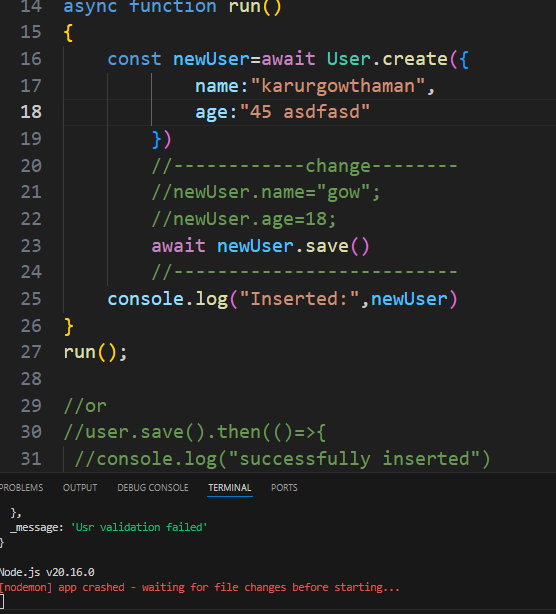


Nested schema:

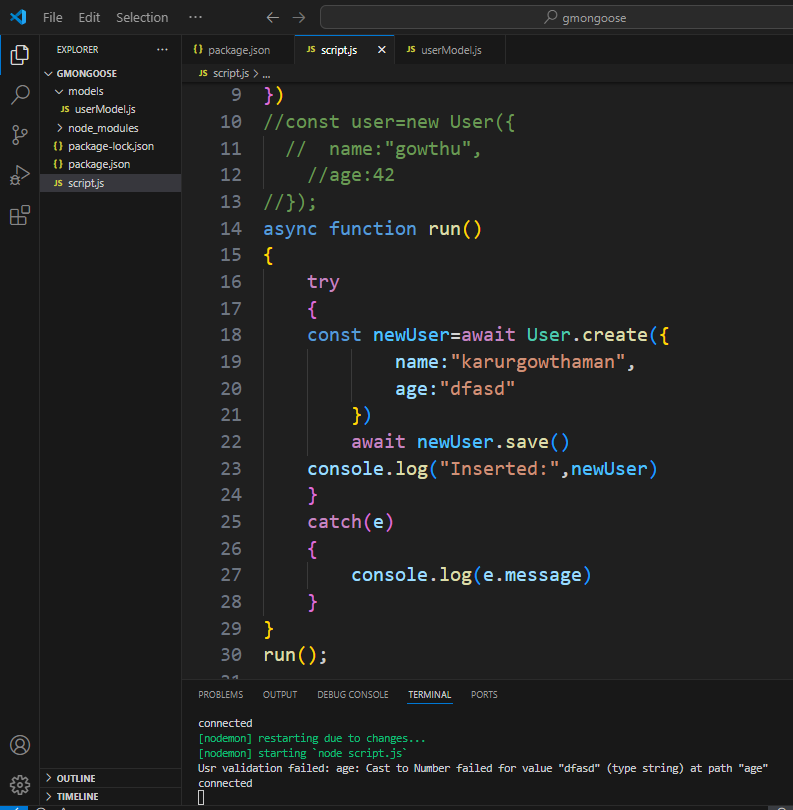


Before error handle :

Age only accept integer types

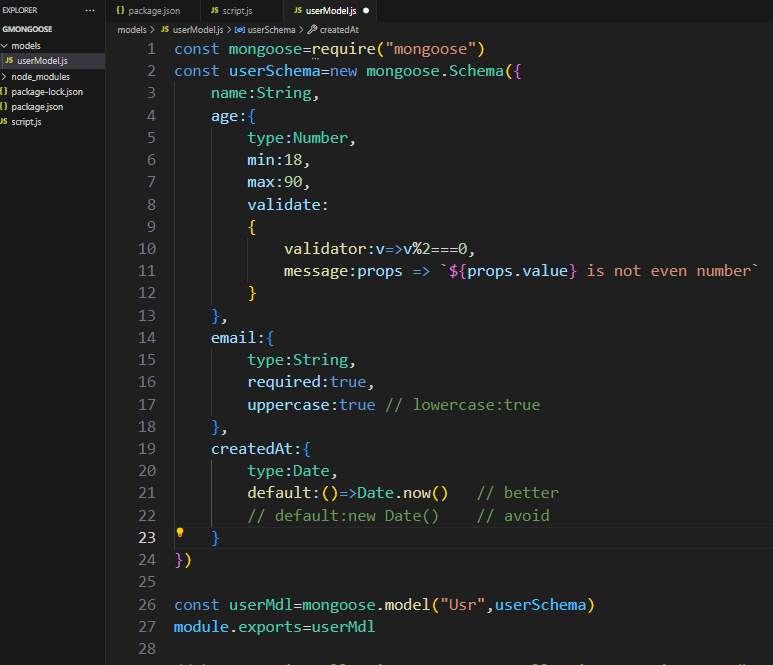


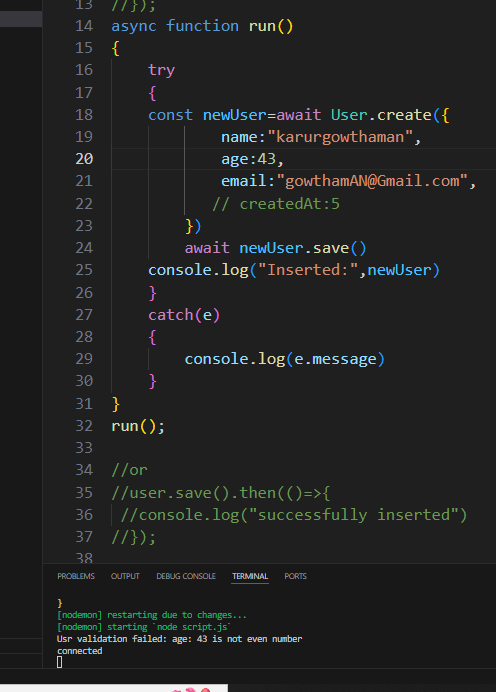
Error handle using try catch block.



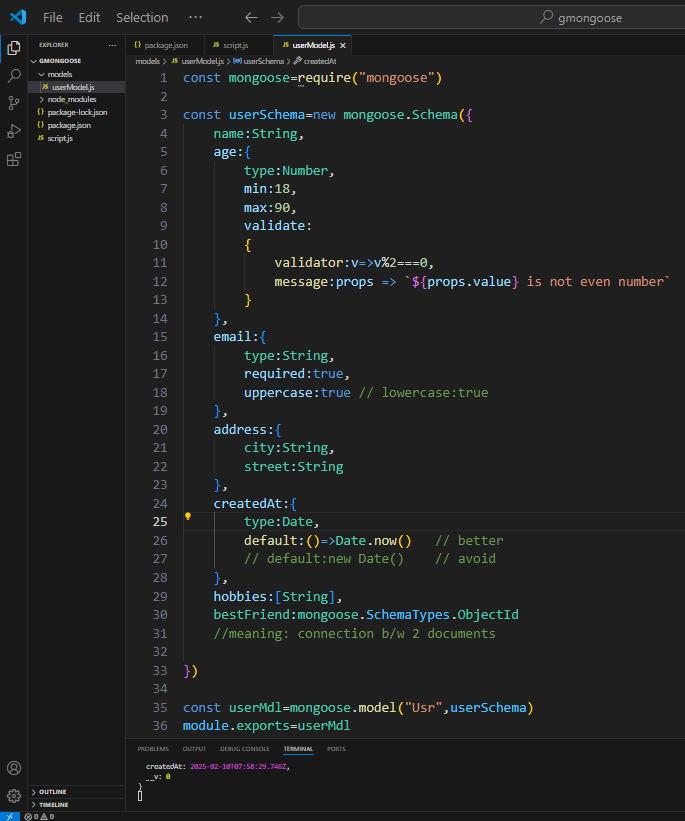
Use: console.log(e.errors)

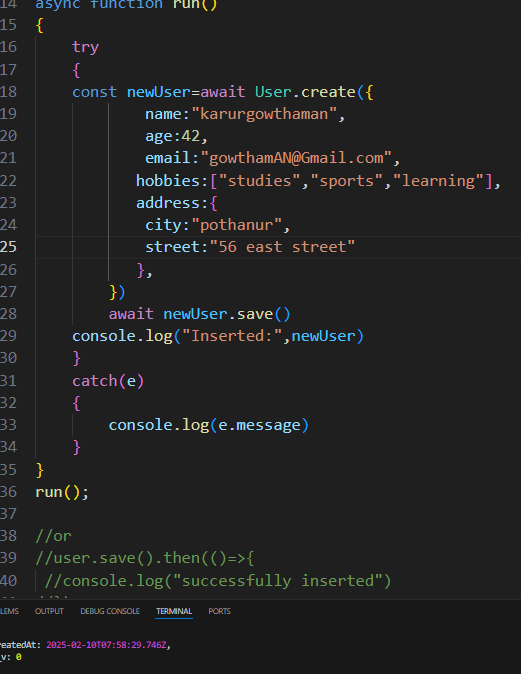
**SCHEMA VALIDATAION:**





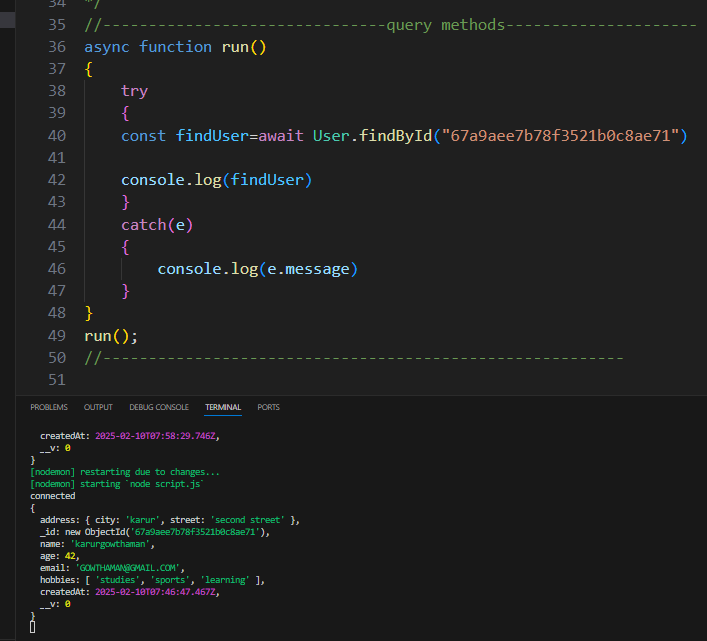
Note: age only accept even numbers (validation)



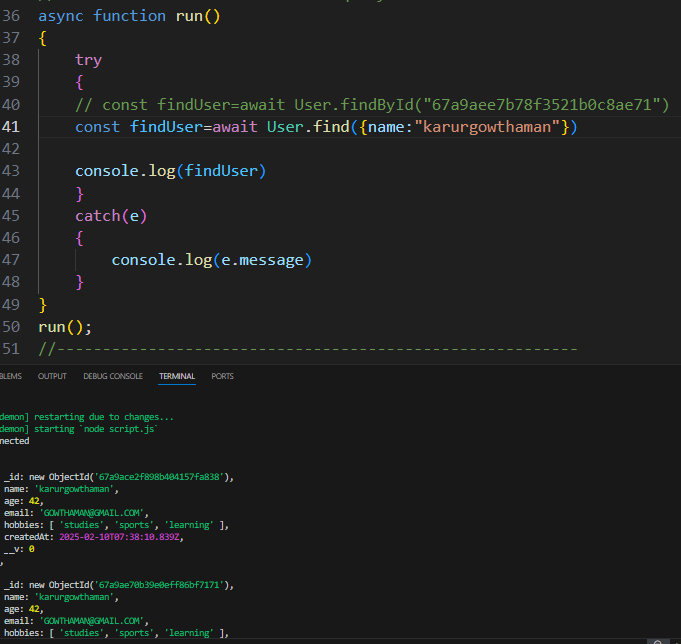


**Query methods:**

**findById**

****

**find query methods:**

****

**task: findOne query methods (first one only appear)**

**use: exists query methods only show object id**

**ex: //const findUser=await User.findOne({name:"karurgowthaman"})**

**const findUser=await User.exists({name:"karurgowthaman"})**

**where function query methods:**

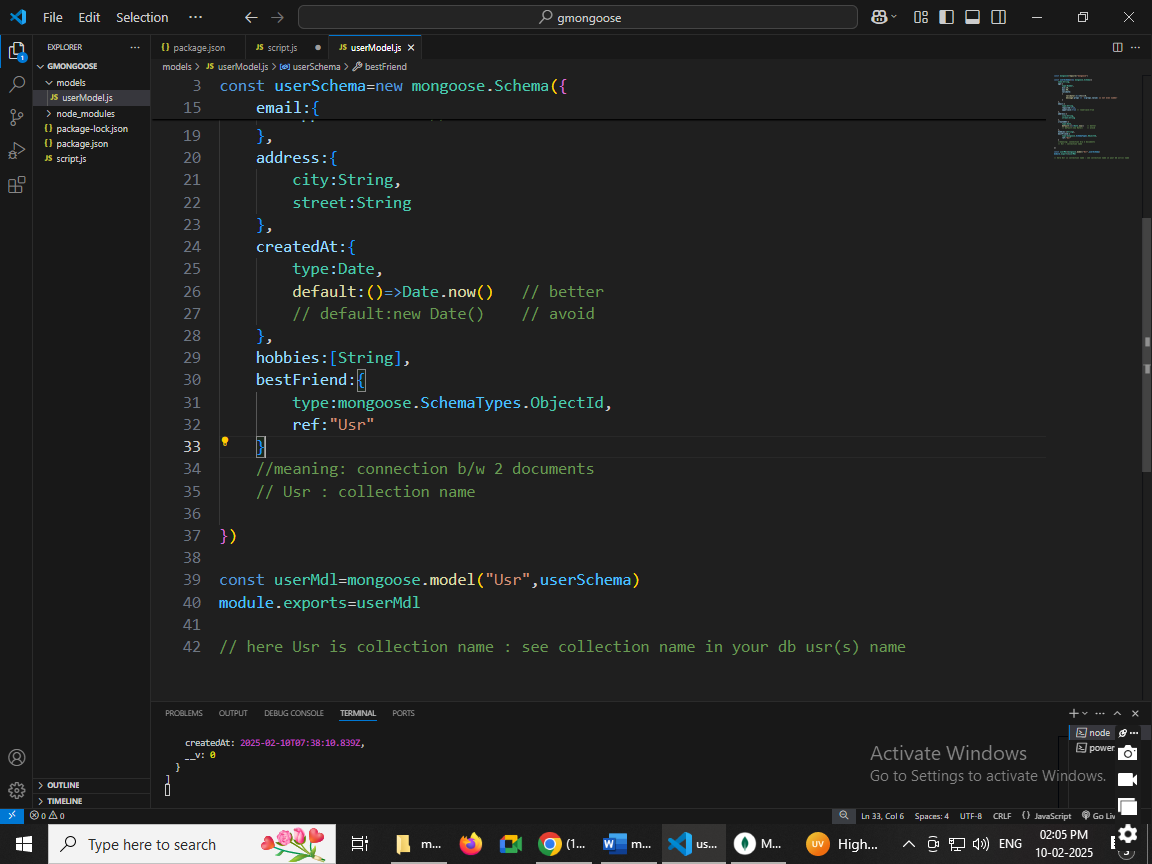
**use: avoid deleteOne,deleteMany,updateOne,updateMany**

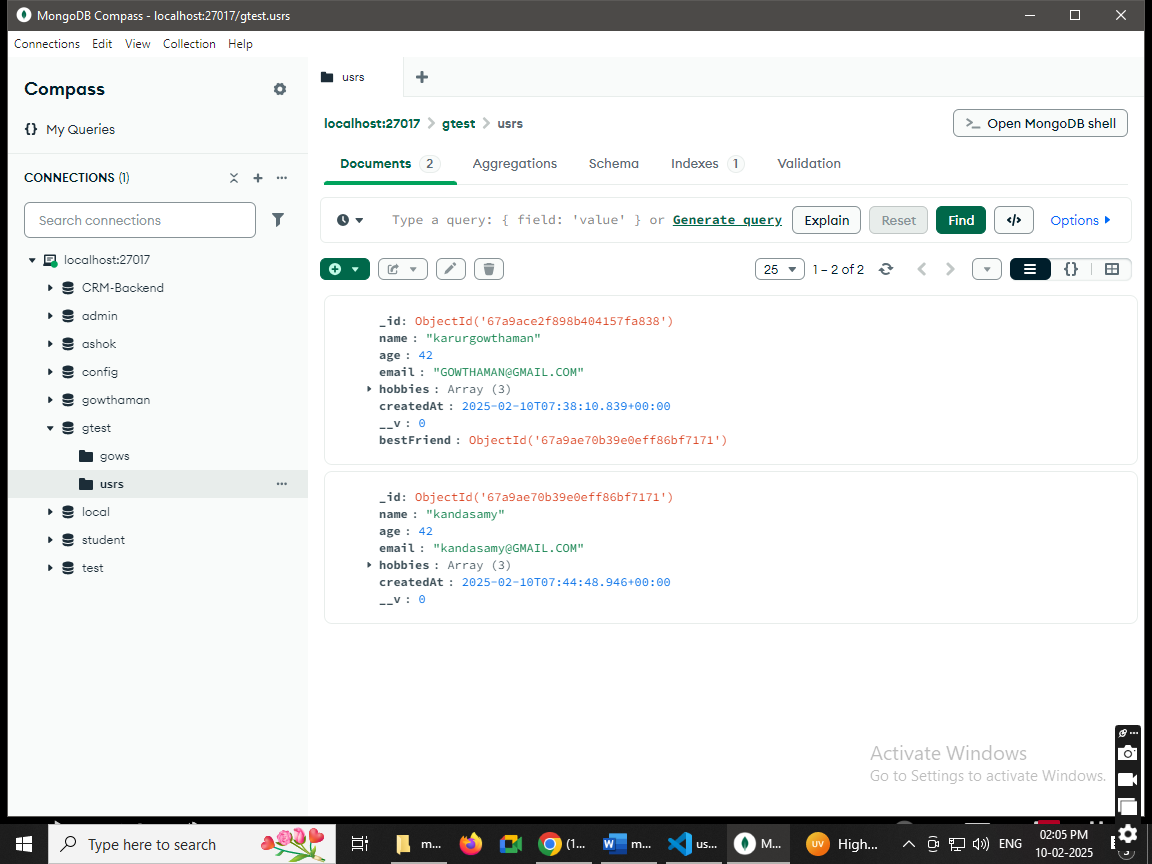
const findAllUsers=await User.where('name').equals("karurgowthaman")

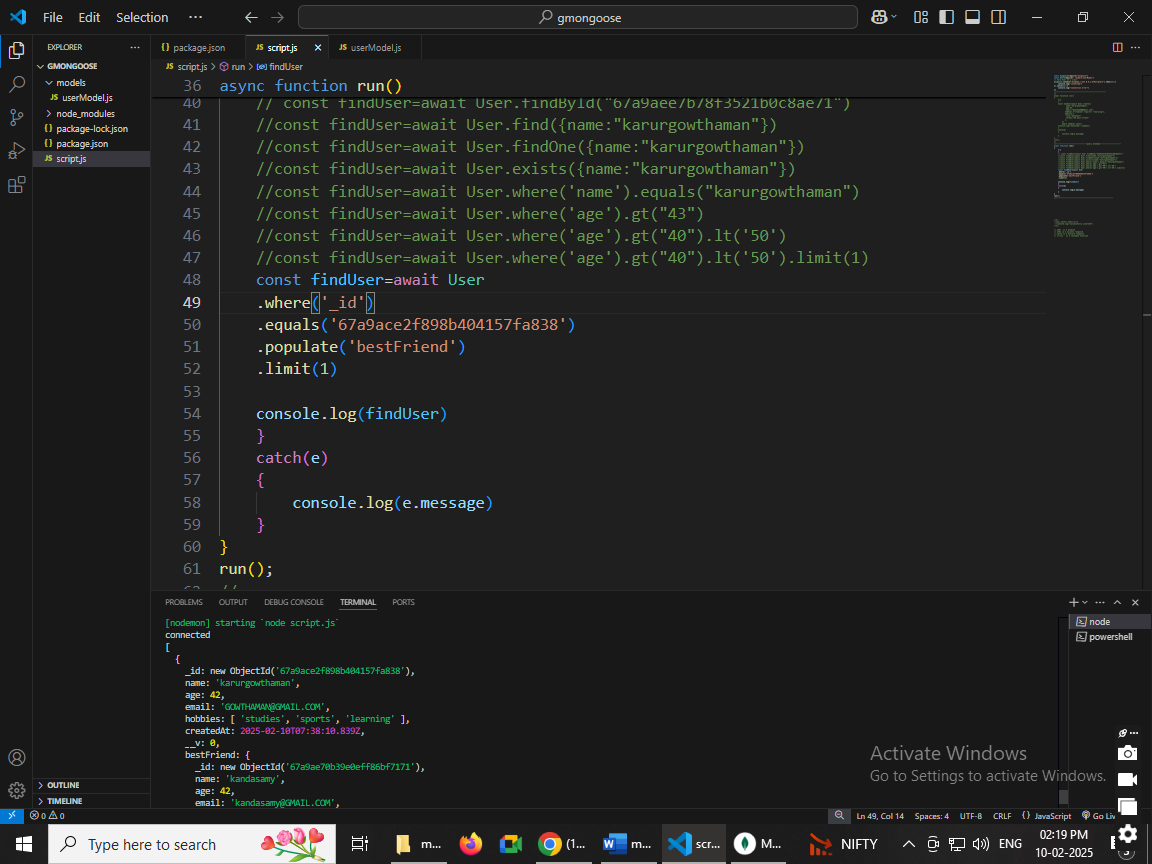
const findUser=await User.where('age').gt("43")

const findUser=await User.where('age').gt("40").lt('50')

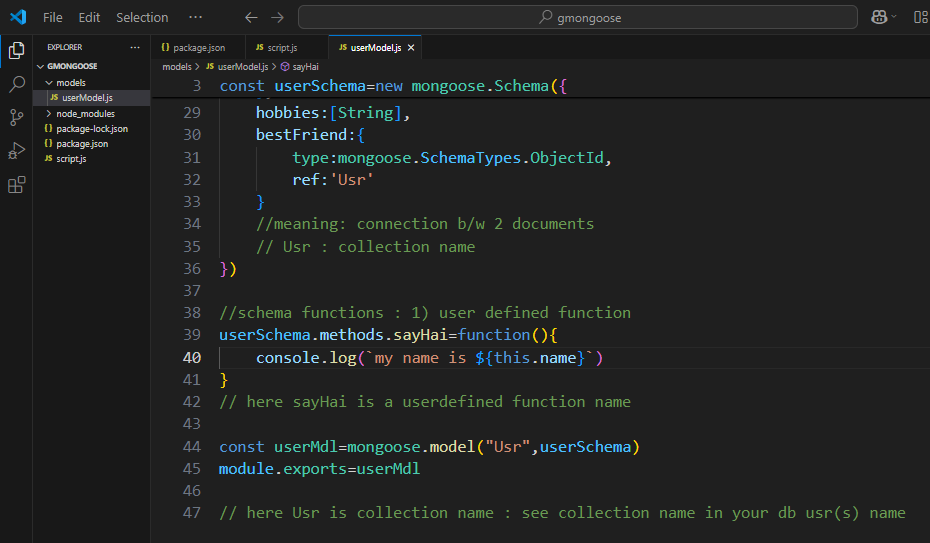
const findUser=await User.where('age').gt("40").lt('50').limit(1)

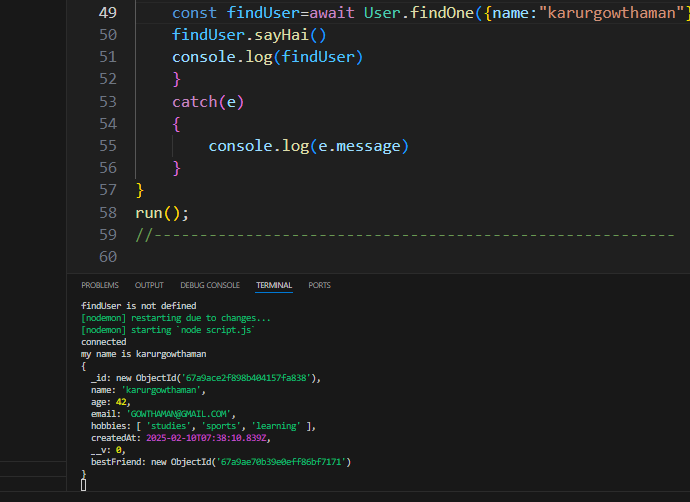




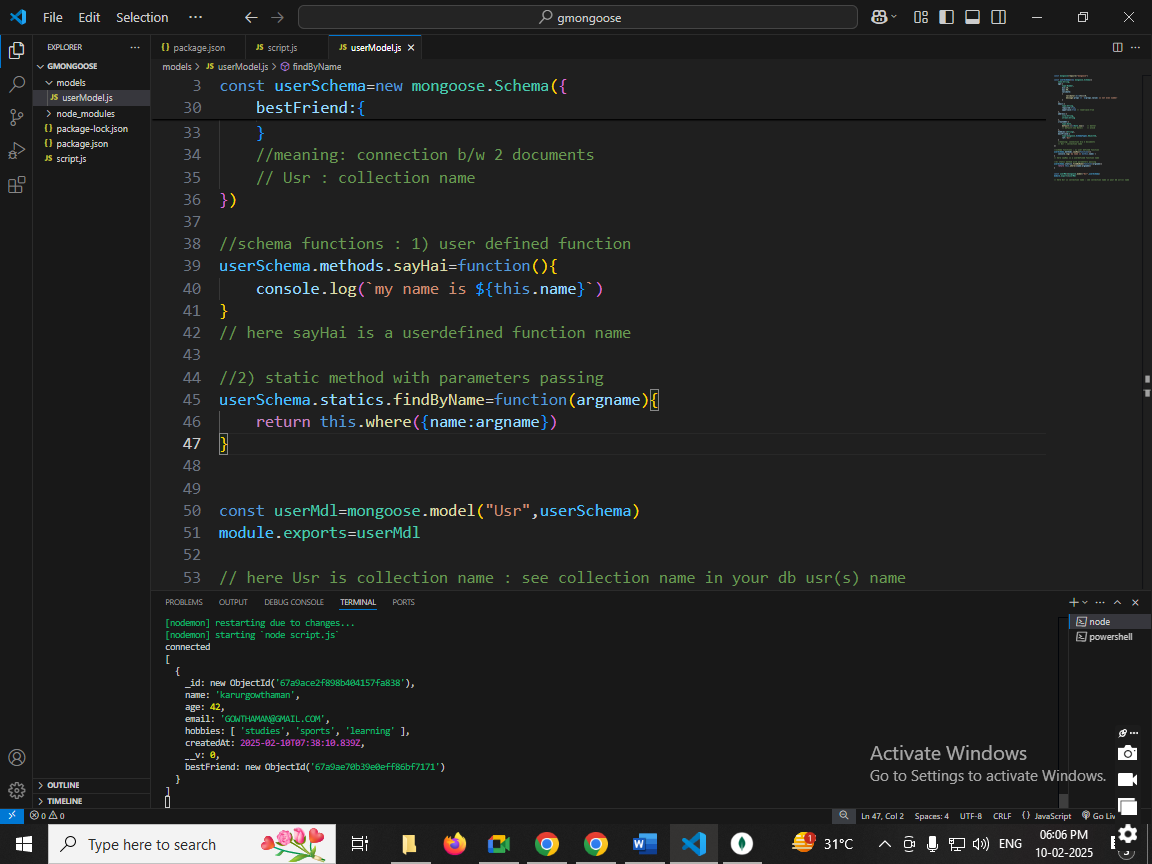


**Schema Methods:**

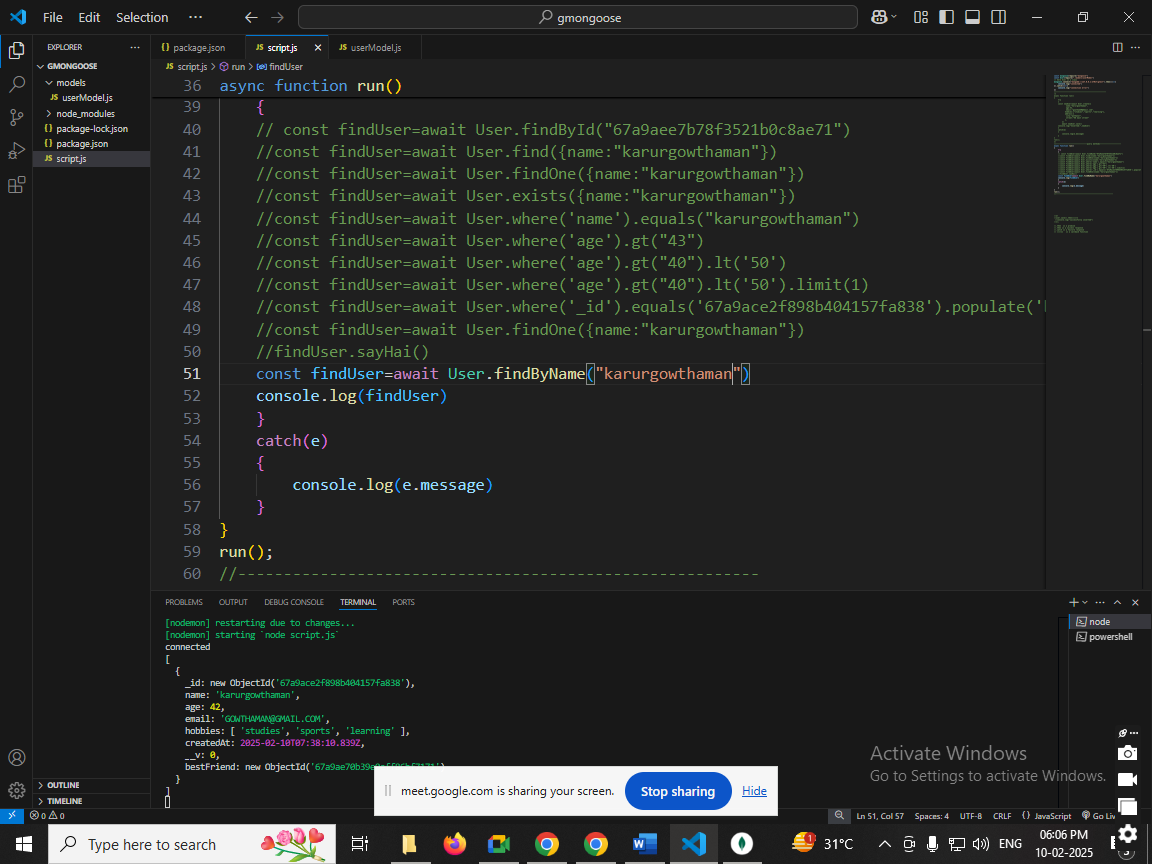


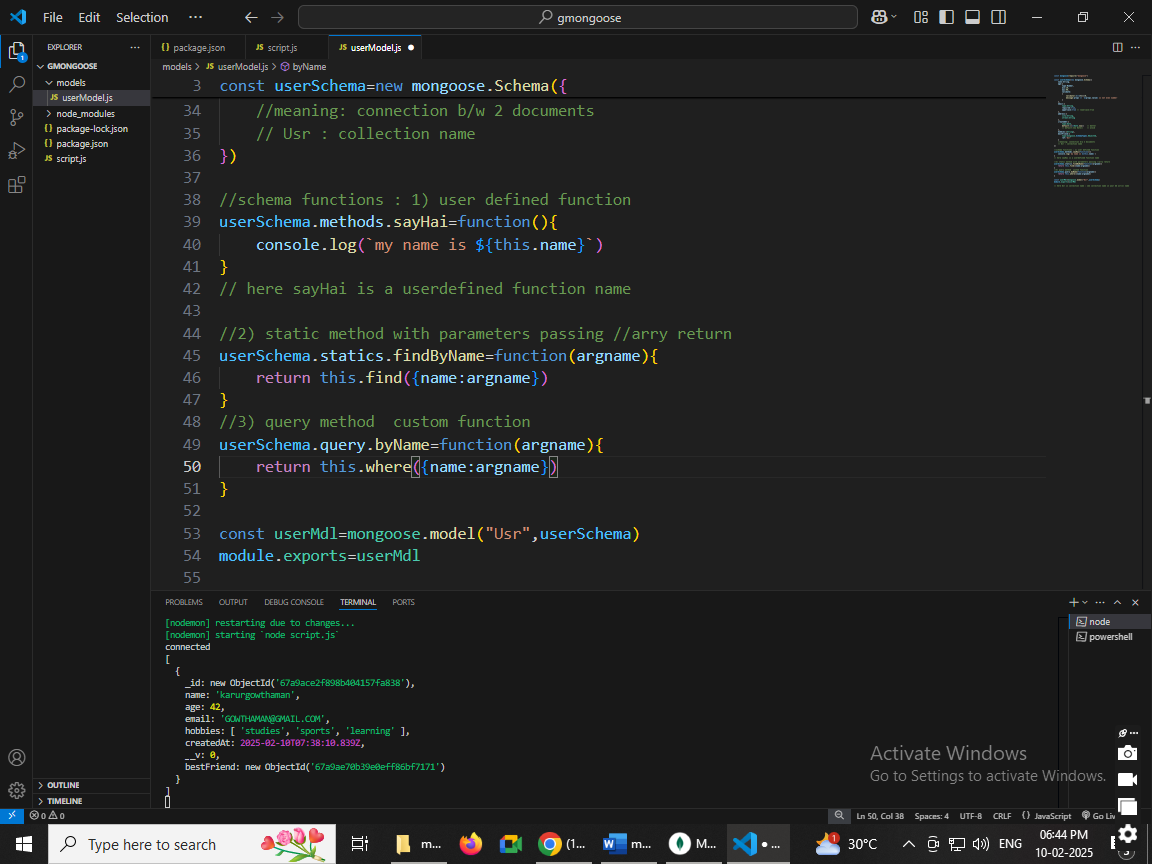


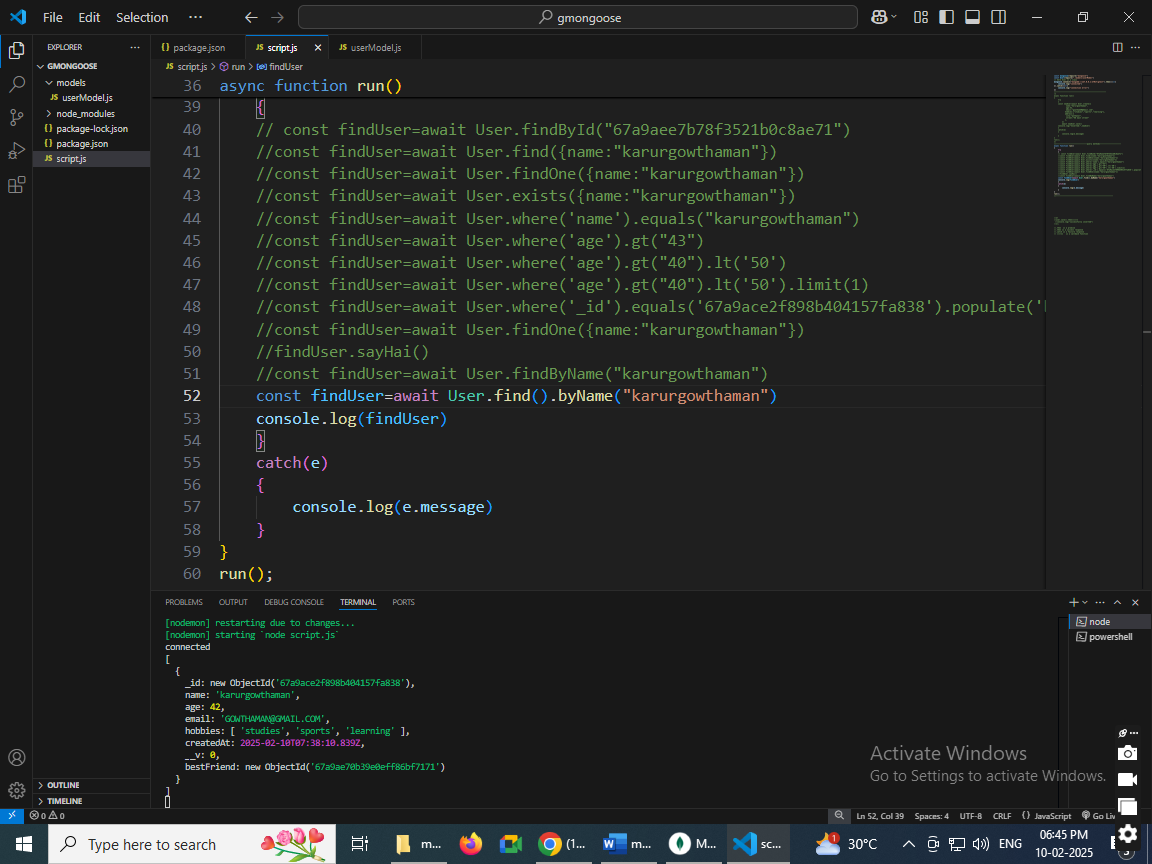
Line no. 50 sayHai() userdefined function call to line no 39



Lineno:46 use find keyword





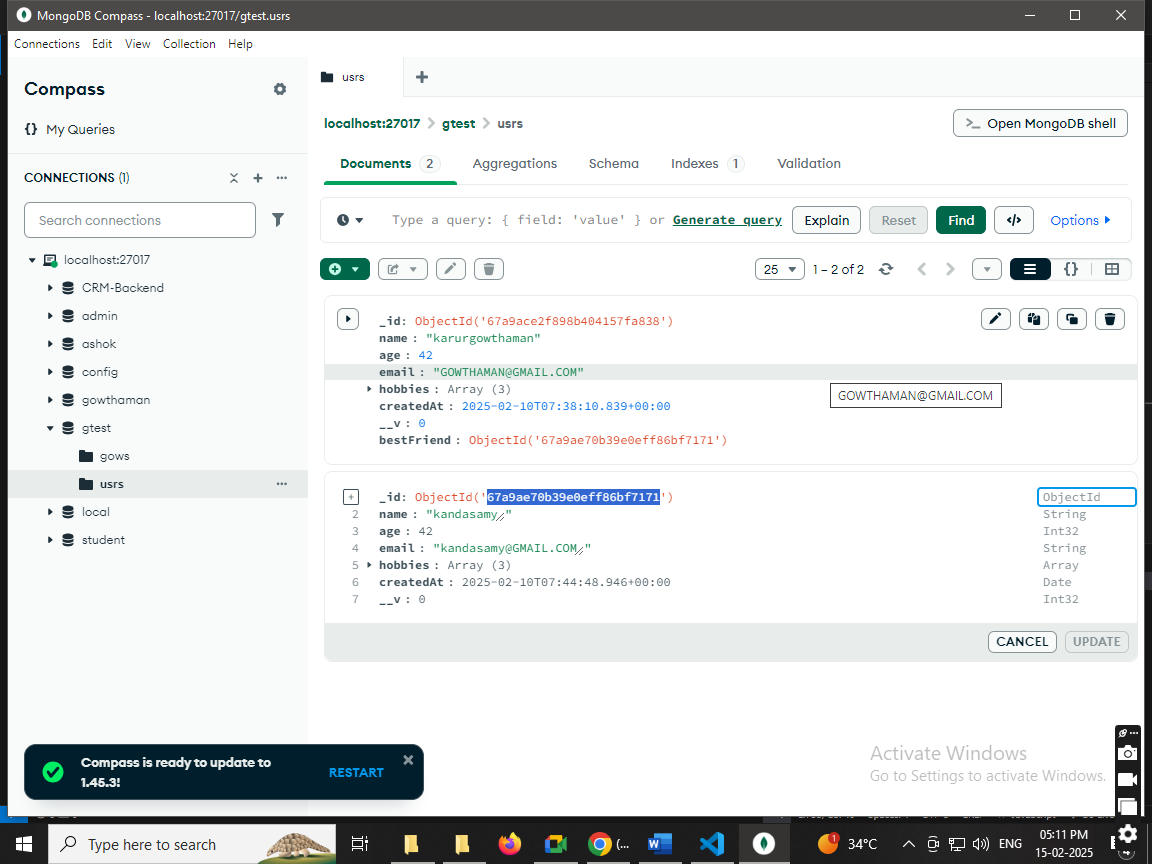


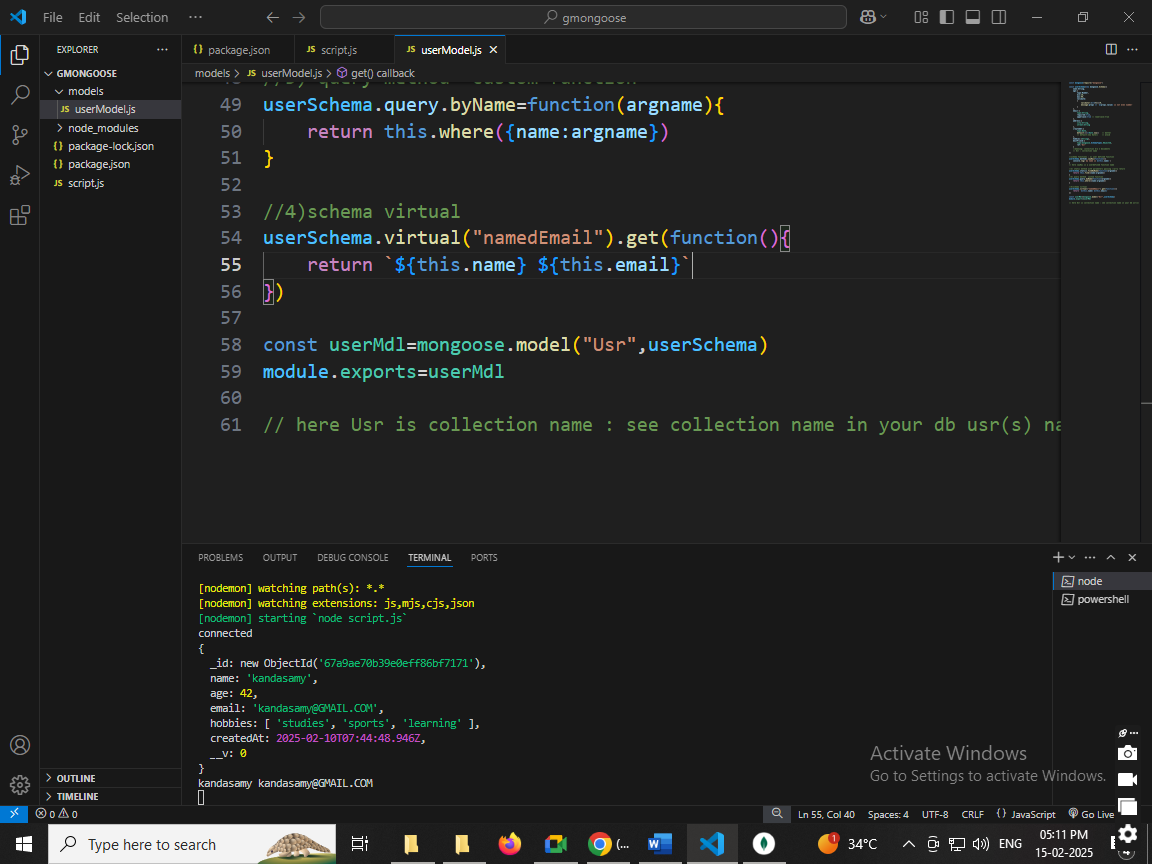
**Schema Virtuals:**

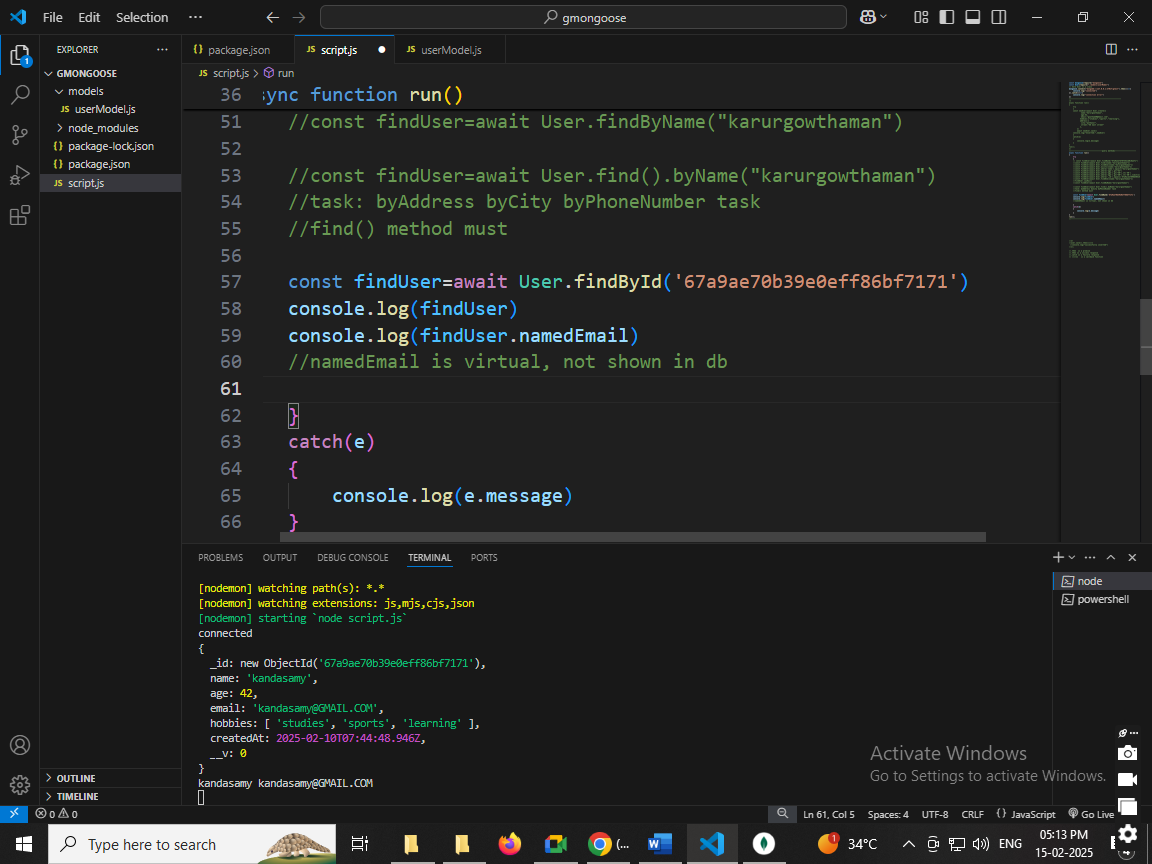
Meaning : Adding new field

Ex: {namedEmail:”gowthaman@gmail.com”}

But this email not visible. But we can see virtually using code.



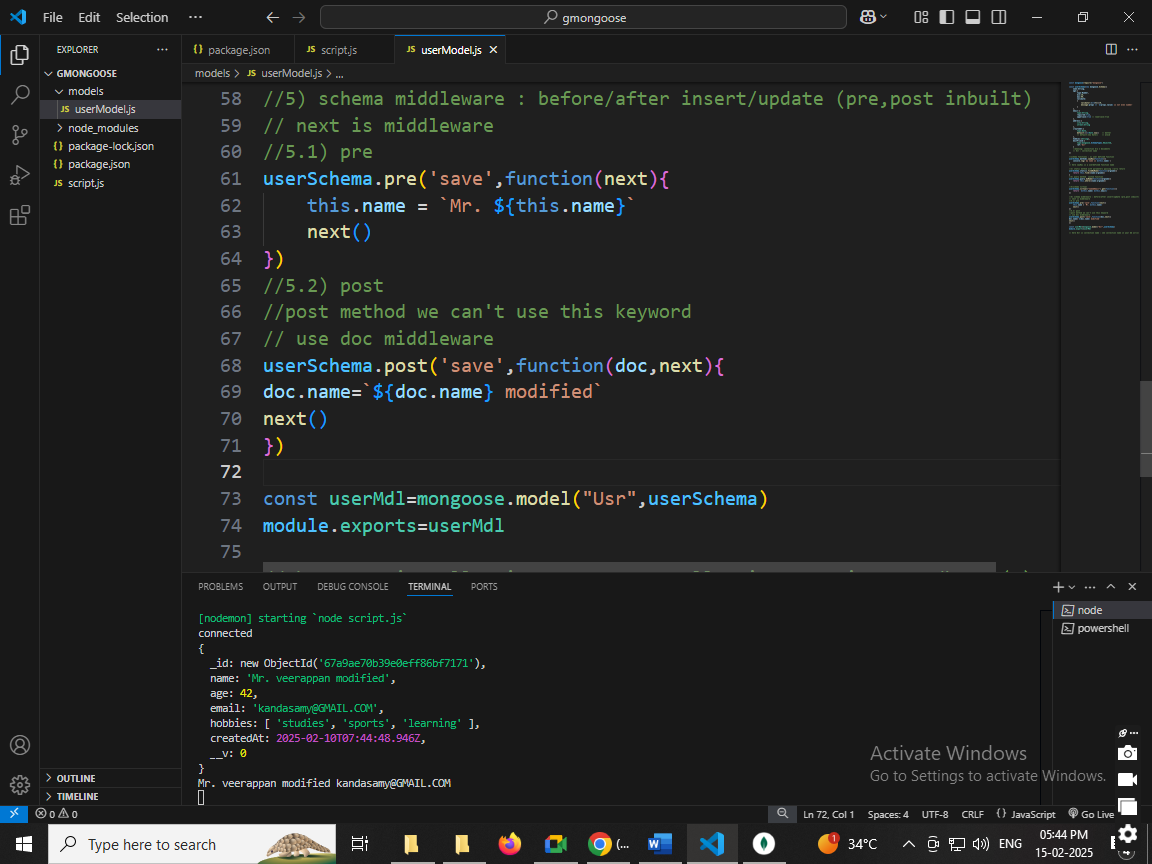


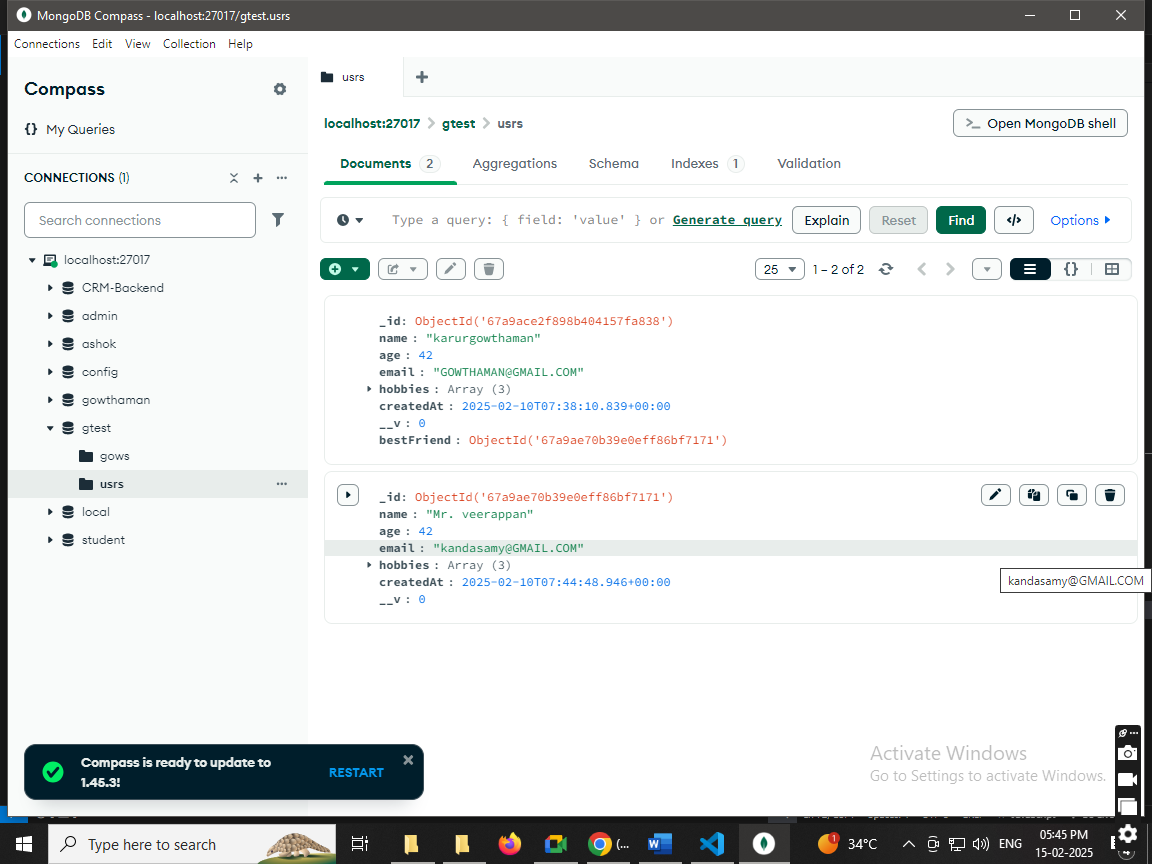


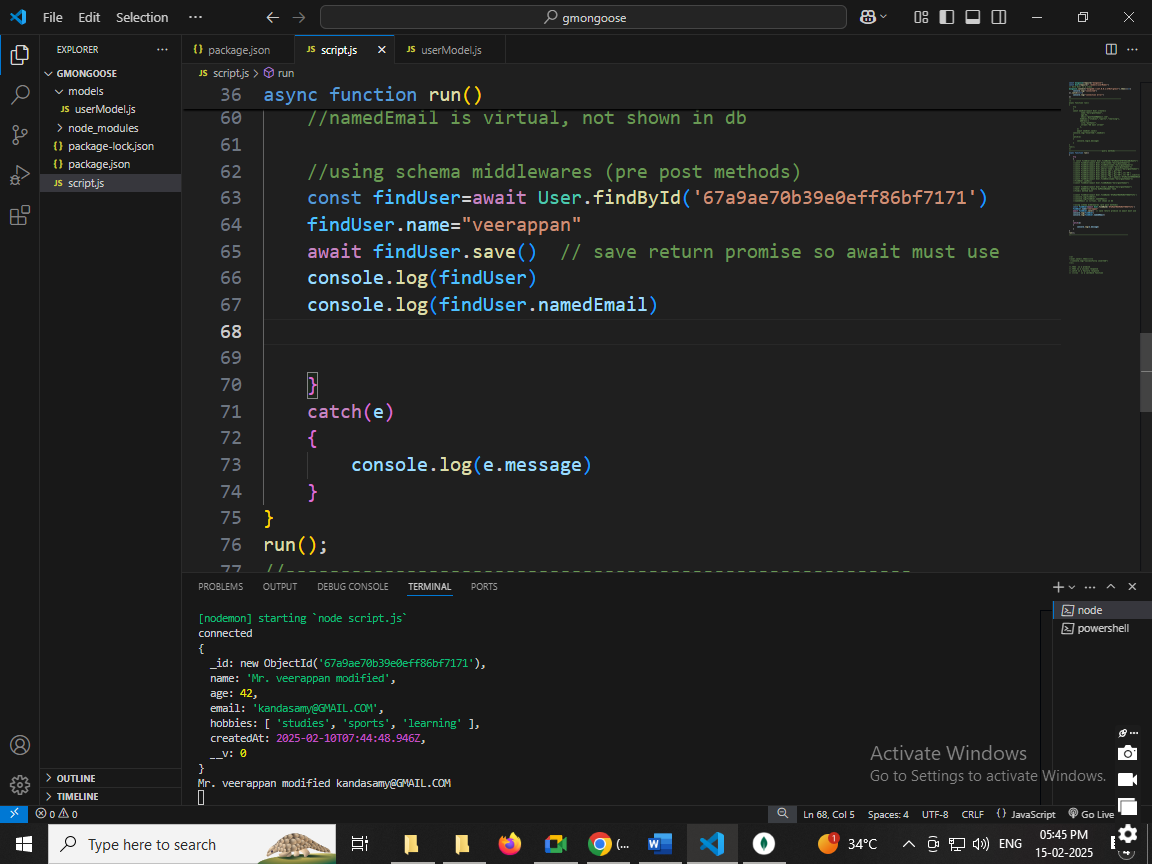
Schema Middlewares:

Meaning: before insert / after insert some changes

Ex: Mr. yourname(add Mr. before insert)







=================end thanks jvl code==============