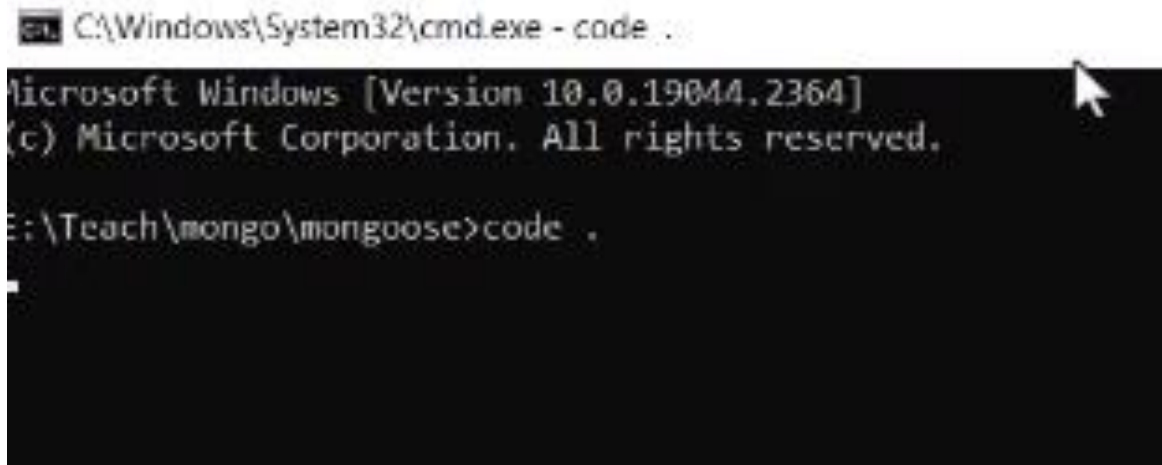


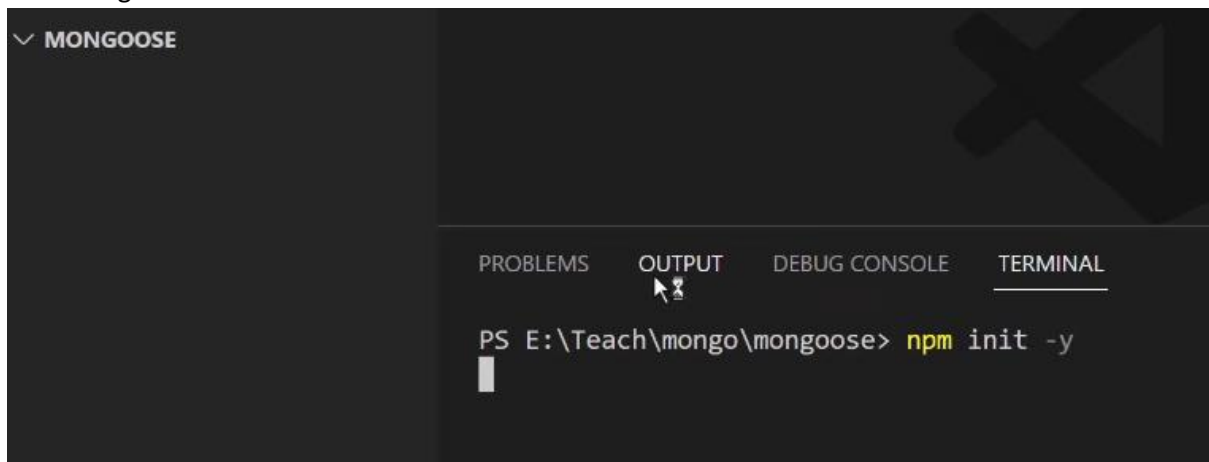
Create folder, type cmd in address bar, code .



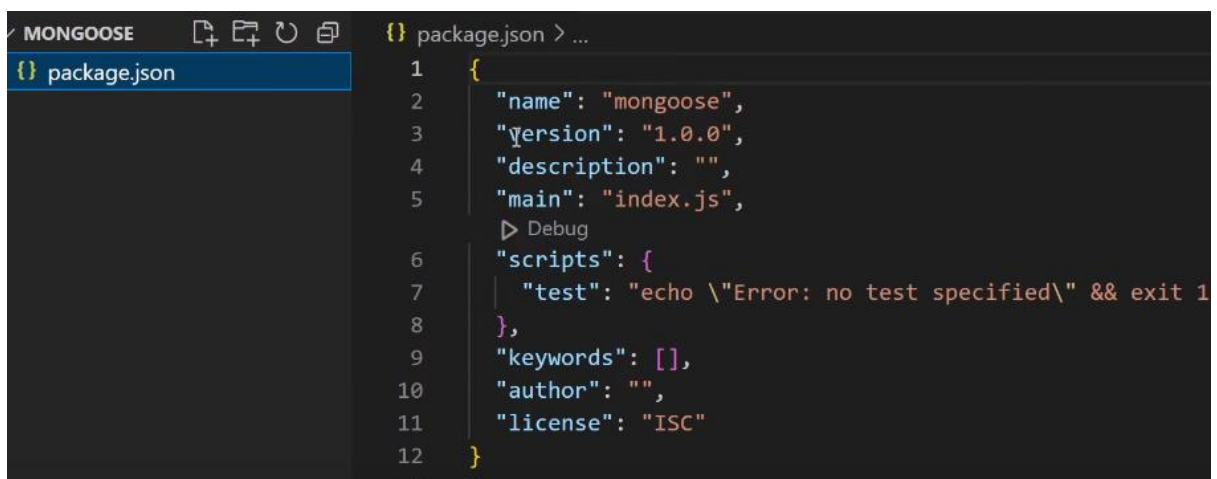
```
C:\Windows\System32\cmd.exe - code .  
Microsoft Windows [Version 10.0.19044.2364]  
(c) Microsoft Corporation. All rights reserved.  
E:\Teach\mongo\mongoose>code .
```

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.

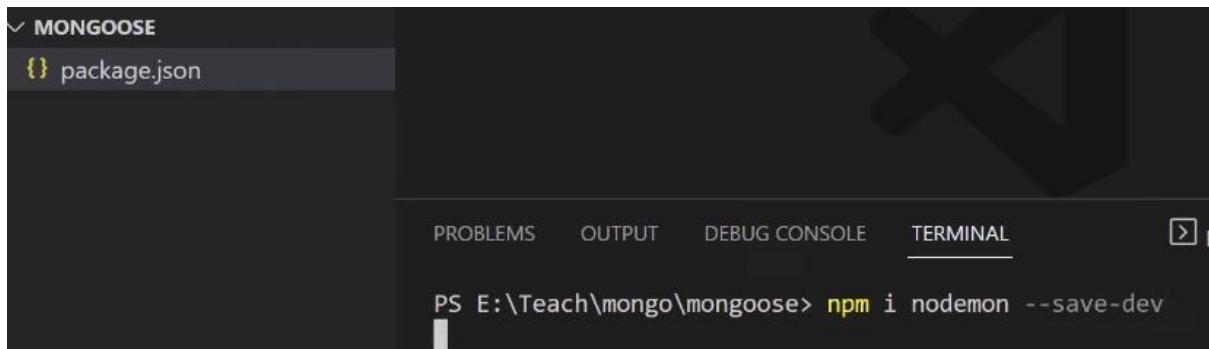
Y meaning : Yes



```
PS E:\Teach\mongo\mongoose> npm init -y
```



```
{  
  "name": "mongoose",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \"Error: no test specified\" && exit 1"  
  },  
  "keywords": [],  
  "author": "",  
  "license": "ISC"  
}
```



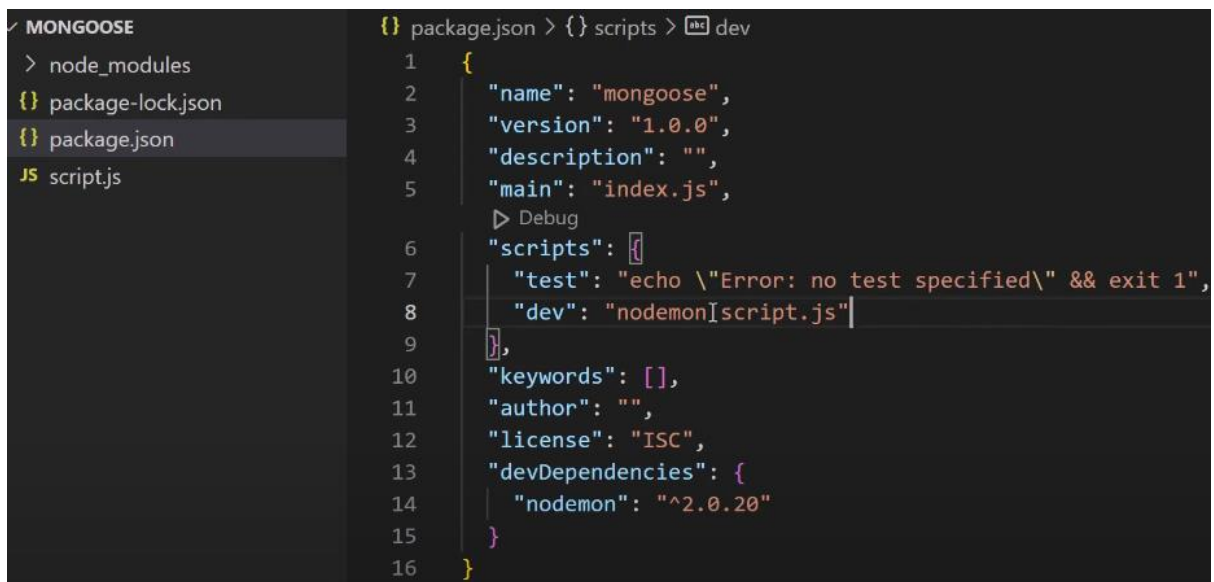
The screenshot shows the VS Code interface for a project named 'MONGOOSE'. The Explorer sidebar on the left shows a file tree with 'package.json' selected. The bottom panel shows the 'TERMINAL' tab with a PowerShell prompt at 'E:\Teach\mongo\mongoose' where the command 'npm i nodemon --save-dev' has been entered.

--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



The screenshot shows the VS Code editor with the 'package.json' file open. The Explorer sidebar on the left shows the file tree with 'package.json' selected. The editor displays the following JSON content:

```
1 {
2   "name": "mongoose",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1",
8     "dev": "nodemon script.js"
9   },
10  "keywords": [],
11  "author": "",
12  "license": "ISC",
13  "devDependencies": {
14    "nodemon": "^2.0.20"
15  }
16 }
```

Don't run every time: node script.js

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

```

{} package.json > {} scripts > dev
1 {
2   "name": "mongoose",
3   "version": "1.0.0",
4   "description": "",
  }

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL**

```

PS E:\Teach\mongo\mongoose> npm run dev

> mongoose@1.0.0 dev
> nodemon script.js

[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node script.js`
[nodemon] clean exit - waiting for changes before restart

```

```

JS script.js
1

```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL**

```

PS E:\Teach\mongo\mongoose> npm i mongoose

added 101 packages, and audited 134 packages in 18s

```

Terminal hide/show shortcut :ctrl +~

```

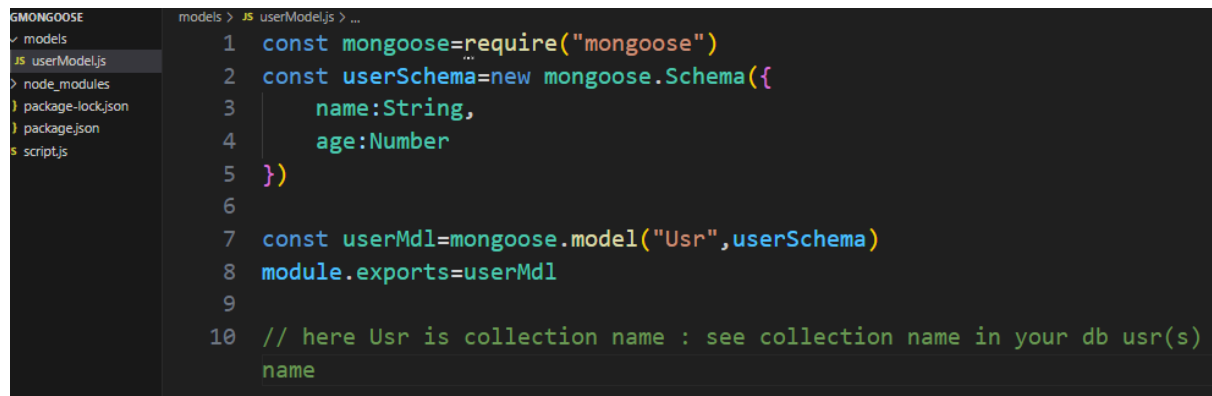
1 const mongoose=require("mongoose")
2
3 mongoose.connect("mongodb://127.0.0.1:27017/gtest").then
4   (()=>{
5     console.log("connected")
6   }).catch(()=>{
7     console.log("connection error")
8   })

```

Schema : inbuilt class of mongoose

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

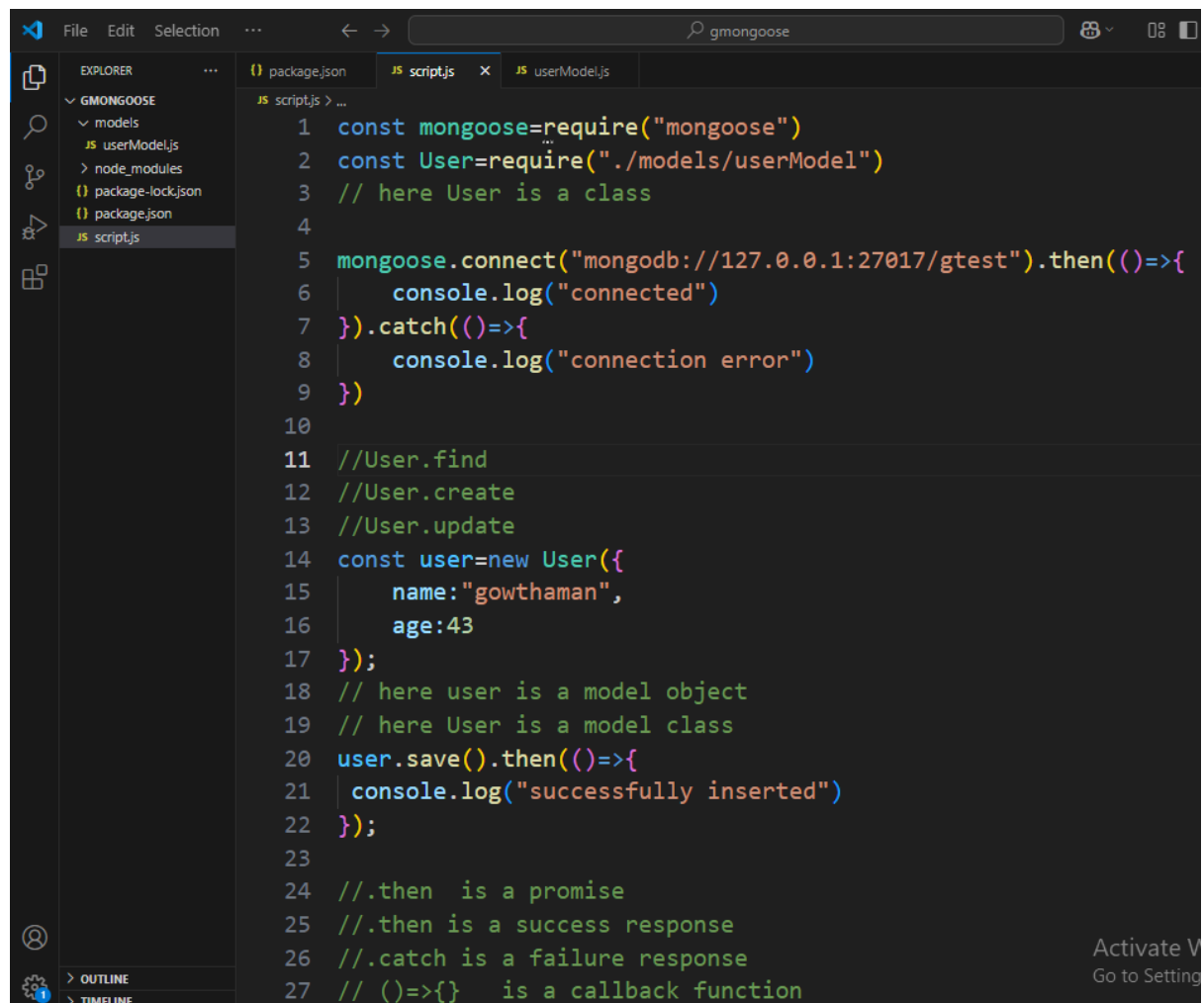
userSchema: userDefined object.



```

1  const mongoose=require("mongoose")
2  const userSchema=new mongoose.Schema({
3      name:String,
4      age:Number
5  })
6
7  const userMdl=mongoose.model("User",userSchema)
8  module.exports=userMdl
9
10 // here User is collection name : see collection name in your db user(s)
    name
    
```

Insert method 1: (with out using async await)



```


1  const mongoose=require("mongoose")
2  const User=require("./models/userModel")
3  // here User is a class
4
5  mongoose.connect("mongodb://127.0.0.1:27017/gtest").then(()=>{
6      console.log("connected")
7  }).catch(()=>{
8      console.log("connection error")
9  })
10
11 //User.find
12 //User.create
13 //User.update
14 const user=new User({
15     name:"gowthaman",
16     age:43
17 });
18 // here user is a model object
19 // here User is a model class
20 user.save().then(()=>{
21     console.log("successfully inserted")
22 });
23
24 // .then is a promise
25 // .then is a success response
26 // .catch is a failure response
27 // ()=>{} is a callback function
    
```

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

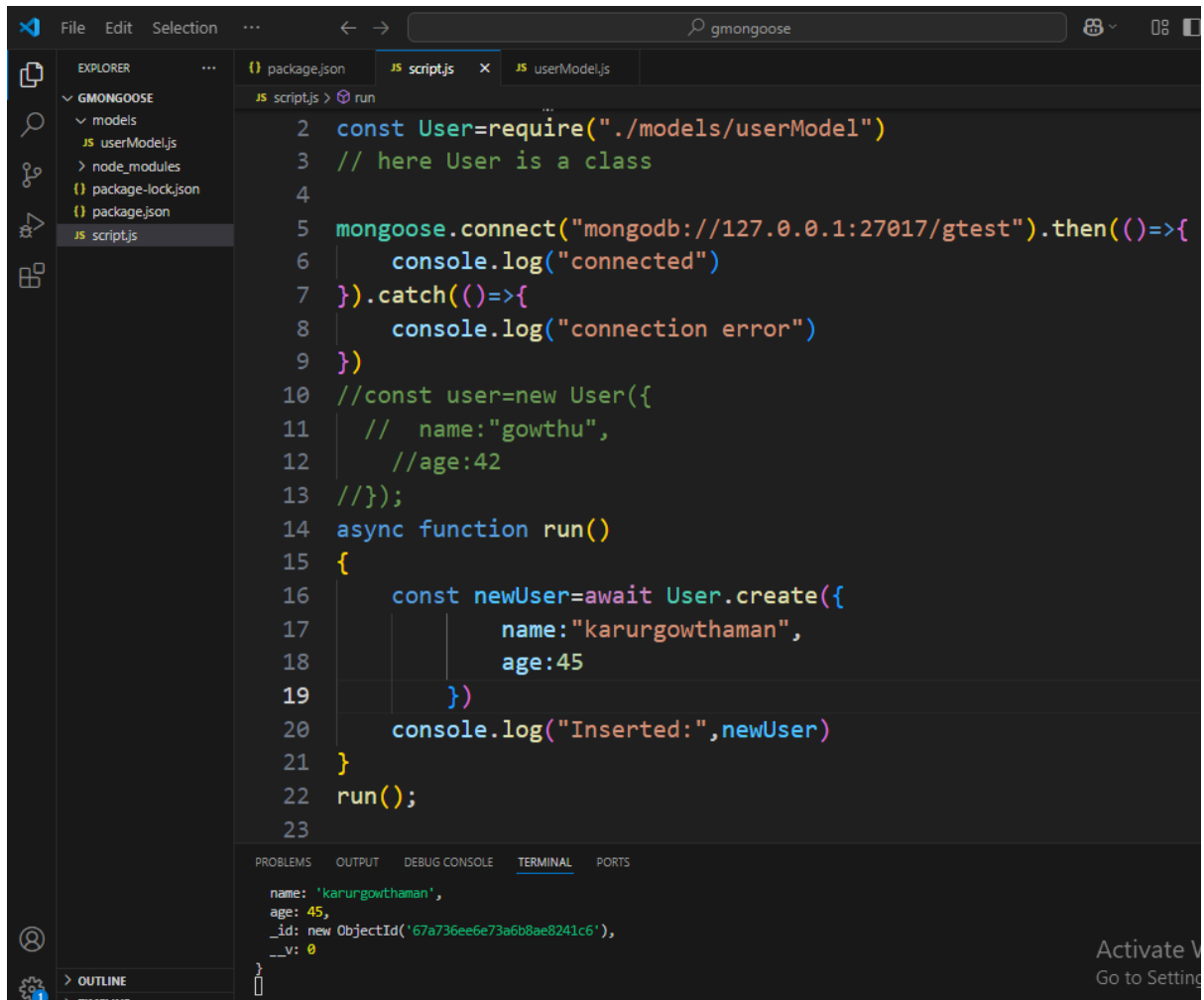


```
1  const mongoose=require("mongoose")
2  const User=require("../models/userModel")
3  // here User is a class
4
5  mongoose.connect("mongodb://127.0.0.1:27017/gtest").then(()=>{
6    console.log("connected")
7  }).catch(()=>{
8    console.log("connection error")
9  })
10 //User.find
11 //User.create
12 //User.update
13 const user=new User({
14   name:"gowthu",
15   age:42
16 });
17 // here user is a model object
18 // here User is a model class
19
20 async function run()
21 {
22   const newUser=await user.save()
23   //meaning: wait till to store data in db and assign result to newUser.
24   console.log("Inserted:",newUser)
25 }
26 run(); // calling function
27
```

Activate Windows
Go to Settings to activate

Method:3 insertion (without using save method)

Only using create



The screenshot shows a VS Code editor with a project named 'gmongoose'. The Explorer sidebar on the left shows the file structure: 'models' (containing 'userModel.js'), 'node_modules', 'package-lock.json', and 'package.json'. The main editor window displays the 'script.js' file with the following code:

```
1 const mongoose = require('mongoose');
2 const User = require('./models/userModel');
3 // here User is a class
4
5 mongoose.connect("mongodb://127.0.0.1:27017/gtest").then(() => {
6   console.log("connected")
7 }).catch(() => {
8   console.log("connection error")
9 })
10 //const user = new User({
11 //  name: "gowthu",
12 //  age: 42
13 //});
14 async function run()
15 {
16   const newUser = await User.create({
17     name: "karurgowthaman",
18     age: 45
19   });
20   console.log("Inserted:", newUser);
21 }
22 run();
23
```

At the bottom of the editor, the TERMINAL tab is active, showing the output of the script:

```
name: 'karurgowthaman',
age: 45,
_id: new ObjectId('67a736ee6e73a6b8ae8241c6'),
__v: 0
```

The bottom right corner of the editor has a message: "Activate VS Code Go to Settings".

```

14  async function run()
15  {
16      const newUser=await User.create({
17          name:"karurgowthaman",
18          age:45
19      })
20      //-----change-----
21      newUser.name="gow";
22      newUser.age=18;
23      await newUser.save()
24      //-----
25      console.log("Inserted:",newUser)
26  }
27  run();
28

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

Inserted: {
  name: 'gow',
  age: 18,
  _id: new ObjectId('67a73853c94d7c8bcd5f356e'),
  __v: 0
}

```

Schema types:

```
3   const userSchema = new mongoose.Schema({
4     name: String,
5     age: Number,
6     email: String,
7     createdAt: Date,
8     updatedAt: Date,
9     bestFriend: mongoose.SchemaTypes.ObjectId,
10    hobbies: [String],
11    address: {
12      city: String,
13      street: String
14    }
15  });
16
```

Nested schema:

```
const addressSchema = new mongoose.Schema( {
  city: String,
  street: String
})

const userSchema = new mongoose.Schema({
  name: String,
  age: Number,
  email: String,
  createdAt: Date,
  updatedAt: Date,
  bestFriend: mongoose.SchemaTypes.ObjectId,
  hobbies: [String],
  address: addressSchema
});
```


Before error handle :

Age only accept integer types

```

14  async function run()
15  {
16      const newUser=await User.create({
17          name:"karurgowthaman",
18          age:"45 asdfasd"
19      })
20      //-----change-----
21      //newUser.name="gow";
22      //newUser.age=18;
23      await newUser.save()
24      //-----
25      console.log("Inserted:",newUser)
26  }
27  run();
28
29  //or
30  //user.save().then(()=>{
31  //console.log("successfully inserted")

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

},
  _message: 'Usr validation failed'
}

Node.js v20.16.0
[nodemon] app crashed - waiting for file changes before starting...

```

Error handle using try catch block.

```

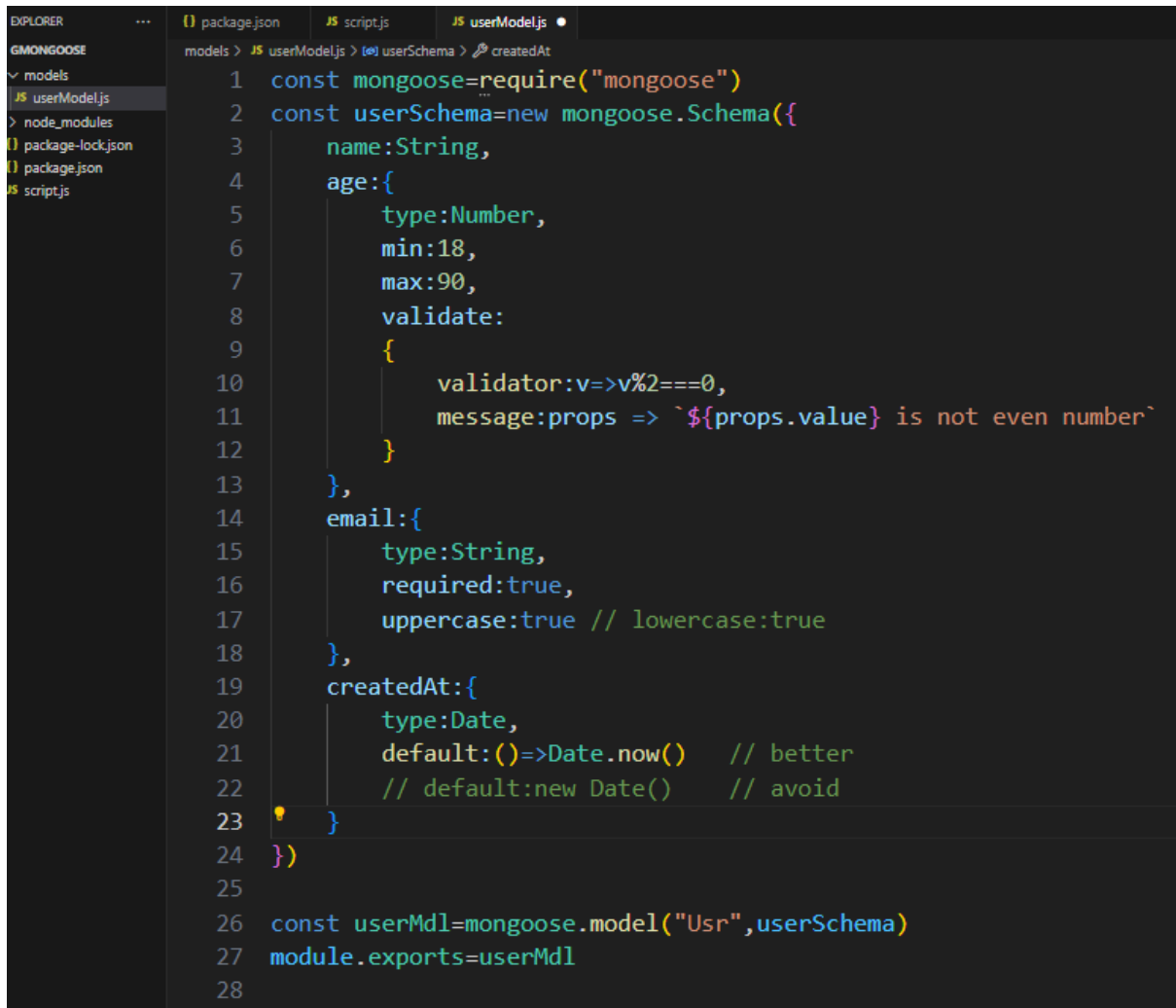
9  })
10 //const user=new User({
11   // name:"gowthu",
12   //age:42
13 //});
14 async function run()
15 {
16   try
17   {
18     const newUser=await User.create({
19       name:"karurgowthaman",
20       age:"dfasd"
21     })
22     await newUser.save()
23     console.log("Inserted:",newUser)
24   }
25   catch(e)
26   {
27     console.log(e.message)
28   }
29 }
30 run();

```

connected
[nodemon] restarting due to changes...
[nodemon] starting `node script.js`
Usr validation failed: age: Cast to Number failed for value "dfasd" (type string) at path "age"
connected
[]

Use: console.log(e.errors)

SCHEMA VALIDATAION:

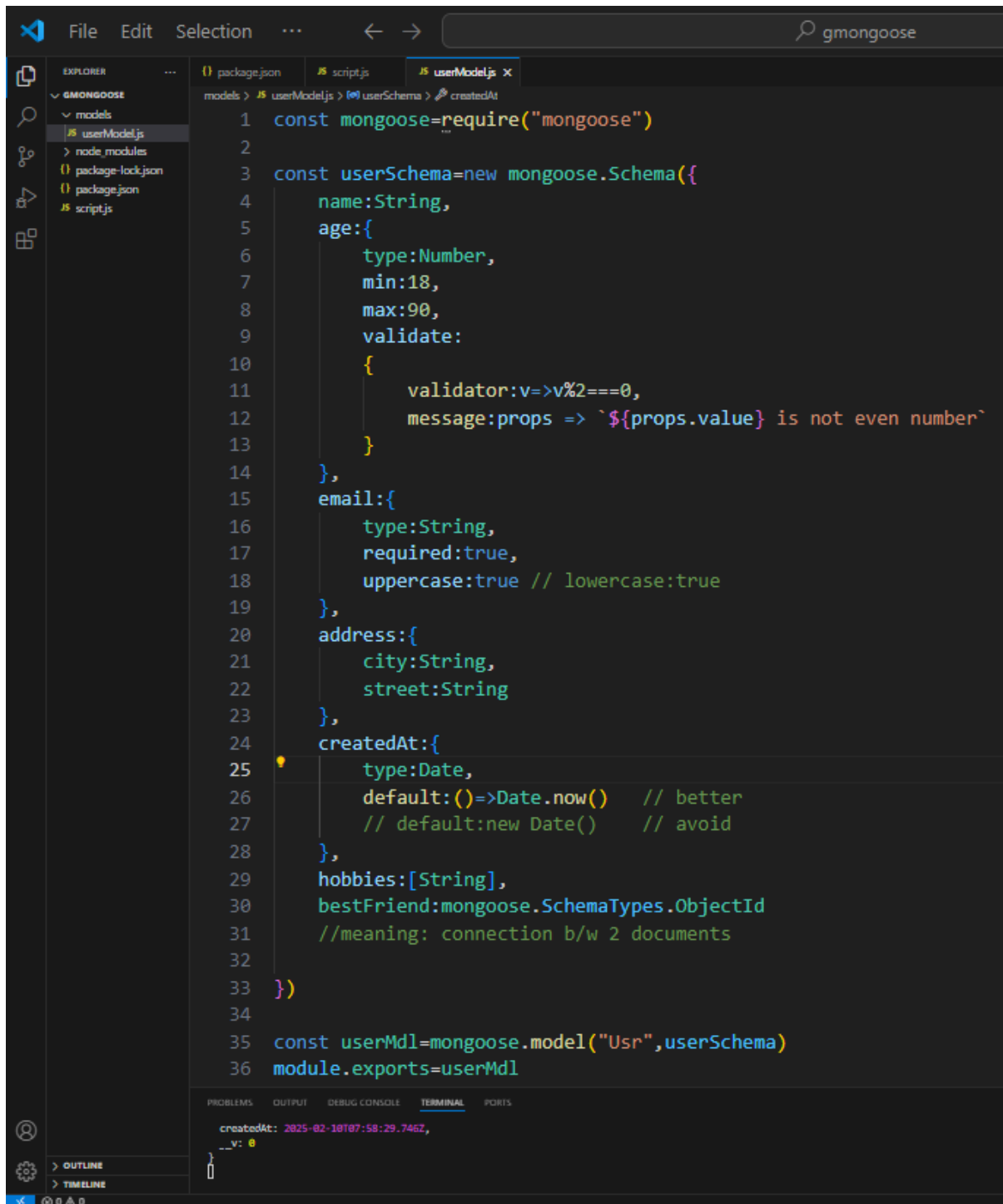


The image shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project structure with a 'models' directory containing 'userModel.js'. The code editor shows the content of 'userModel.js' with the following code:

```
1 const mongoose=require("mongoose")
2 const userSchema=new mongoose.Schema({
3   name:String,
4   age:{
5     type:Number,
6     min:18,
7     max:90,
8     validate:
9     {
10      validator:v=>v%2===0,
11      message:props => `${props.value} is not even number`
12     }
13  },
14  email:{
15    type:String,
16    required:true,
17    uppercase:true // lowercase:true
18  },
19  createdAt:{
20    type>Date,
21    default:()=>Date.now() // better
22    // default:new Date() // avoid
23  }
24 })
25
26 const userMdl=mongoose.model("Ussr",userSchema)
27 module.exports=userMdl
28
```

```
13 //});
14 async function run()
15 {
16   try
17   {
18     const newUser=await User.create({
19       name:"karurgowthaman",
20       age:43,
21       email:"gowthamAN@Gmail.com",
22       // createdAt:5
23     })
24     await newUser.save()
25     console.log("Inserted:",newUser)
26   }
27   catch(e)
28   {
29     console.log(e.message)
30   }
31 }
32 run();
33
34 //or
35 //user.save().then(()=>{
36 //console.log("successfully inserted")
37 //});
38
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
}
[nodemon] restarting due to changes...
[nodemon] starting `node script.js`
Usr validation failed: age: 43 is not even number
connected
[]
```

Note: age only accept even numbers (validation)



The image shows a Visual Studio Code editor window with a project named 'gmongoose'. The Explorer sidebar on the left shows the file structure: 'models' > 'userModel.js'. The main editor area displays the content of 'userModel.js', which defines a Mongoose schema for a user. The schema includes fields for 'name' (String), 'age' (Number with min: 18, max: 90, and a custom validator for even numbers), 'email' (String, required, uppercase), 'address' (Object with 'city' and 'street' as Strings), 'createdAt' (Date with a default of Date.now()), 'hobbies' (Array of Strings), and 'bestFriend' (ObjectId representing a reference to another user document). The model is named 'User' and is exported as 'userMdl'.

```
1 const mongoose=require("mongoose")
2
3 const userSchema=new mongoose.Schema({
4   name:String,
5   age:{
6     type:Number,
7     min:18,
8     max:90,
9     validate:
10    {
11      validator:v=>v%2===0,
12      message:props => `${props.value} is not even number`
13    }
14  },
15  email:{
16    type:String,
17    required:true,
18    uppercase:true // lowercase:true
19  },
20  address:{
21    city:String,
22    street:String
23  },
24  createdAt:{
25    type>Date,
26    default:()=>Date.now() // better
27    // default:new Date() // avoid
28  },
29  hobbies:[String],
30  bestFriend:mongoose.SchemaTypes.ObjectId
31  //meaning: connection b/w 2 documents
32 },
33 })
34
35 const userMdl=mongoose.model("User",userSchema)
36 module.exports=userMdl
```

The bottom panel of the editor shows the 'TERMINAL' tab with the following output:

```
createdAt: 2025-02-10T07:58:29.745Z,
__v: 0
```

```
14 async function run()
15 {
16   try
17   {
18     const newUser=await User.create({
19       name:"karurgowthaman",
20       age:42,
21       email:"gowthamAN@Gmail.com",
22       hobbies:["studies","sports","learning"],
23       address:{
24         city:"pothanur",
25         street:"56 east street"
26       },
27     })
28     await newUser.save()
29     console.log("Inserted:",newUser)
30   }
31   catch(e)
32   {
33     console.log(e.message)
34   }
35 }
36 run();
37
38 //or
39 //user.save().then(())=>{
40 //console.log("successfully inserted")
41 //}
```

LEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

reatedAt: 2025-02-10T07:58:29.746Z,
y: 0

Query methods:

findById

```
34 //
35 //-----query methods-----
36 async function run()
37 {
38   try
39   {
40     const findUser=await User.findById("67a9aee7b78f3521b0c8ae71")
41
42     console.log(findUser)
43   }
44   catch(e)
45   {
46     console.log(e.message)
47   }
48 }
49 run();
50 //-----
51
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
createdAt: 2025-02-10T07:58:29.746Z,
__v: 0
}
[nodemon] restarting due to changes...
[nodemon] starting 'node script.js'
connected
{
  address: { city: 'karur', street: 'second street' },
  _id: new ObjectId('67a9aee7b78f3521b0c8ae71'),
  name: 'karungowthaman',
  age: 42,
  email: 'GOWTHAMAN@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:46:47.467Z,
  __v: 0
}
[]
```

find query methods:

```
36 async function run()
37 {
38   try
39   {
40     // const findUser=await User.findById("67a9aee7b78f3521b0c8ae71")
41     const findUser=await User.find({name:"karungowthaman"})
42
43     console.log(findUser)
44   }
45   catch(e)
46   {
47     console.log(e.message)
48   }
49 }
50 run();
51 //-----
```

BLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

demon] restarting due to changes...
demon] starting 'node script.js'
nected

```
_id: new ObjectId('67a9ace2f898b404157fa838'),
name: 'karungowthaman',
age: 42,
email: 'GOWTHAMAN@GMAIL.COM',
hobbies: [ 'studies', 'sports', 'learning' ],
createdAt: 2025-02-10T07:38:10.839Z,
__v: 0
,
_id: new ObjectId('67a9ae70b39e0eff86bf7171'),
name: 'karungowthaman',
age: 42,
email: 'GOWTHAMAN@GMAIL.COM',
hobbies: [ 'studies', 'sports', 'learning' ],
```

task: findOne query methods (first one only appear)

use: exists query methods only show object id

Mongoose gowthaman kalvi sahas karur 98940-83890

ex:

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

where function query methods:

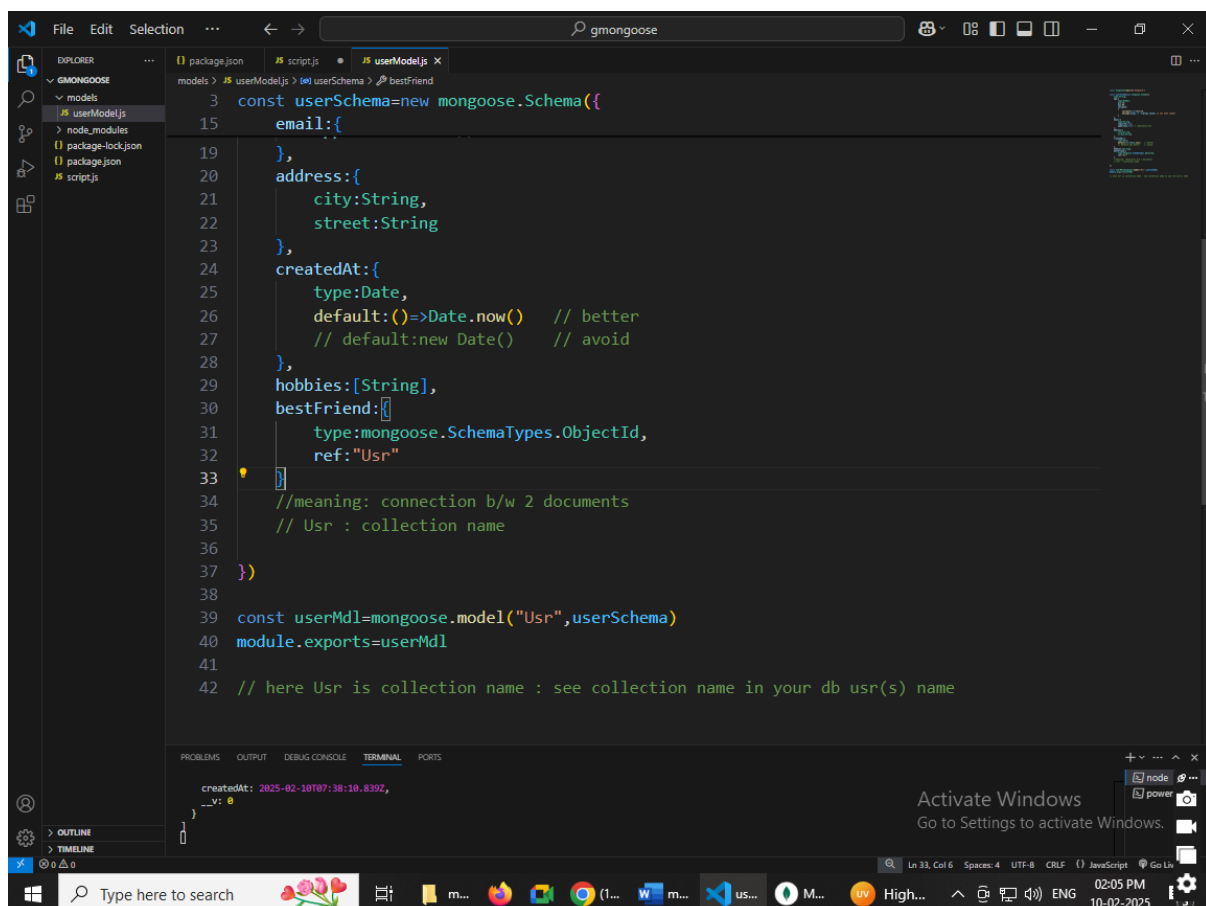
use: avoid deleteOne,deleteMany,updateOne,updateMany

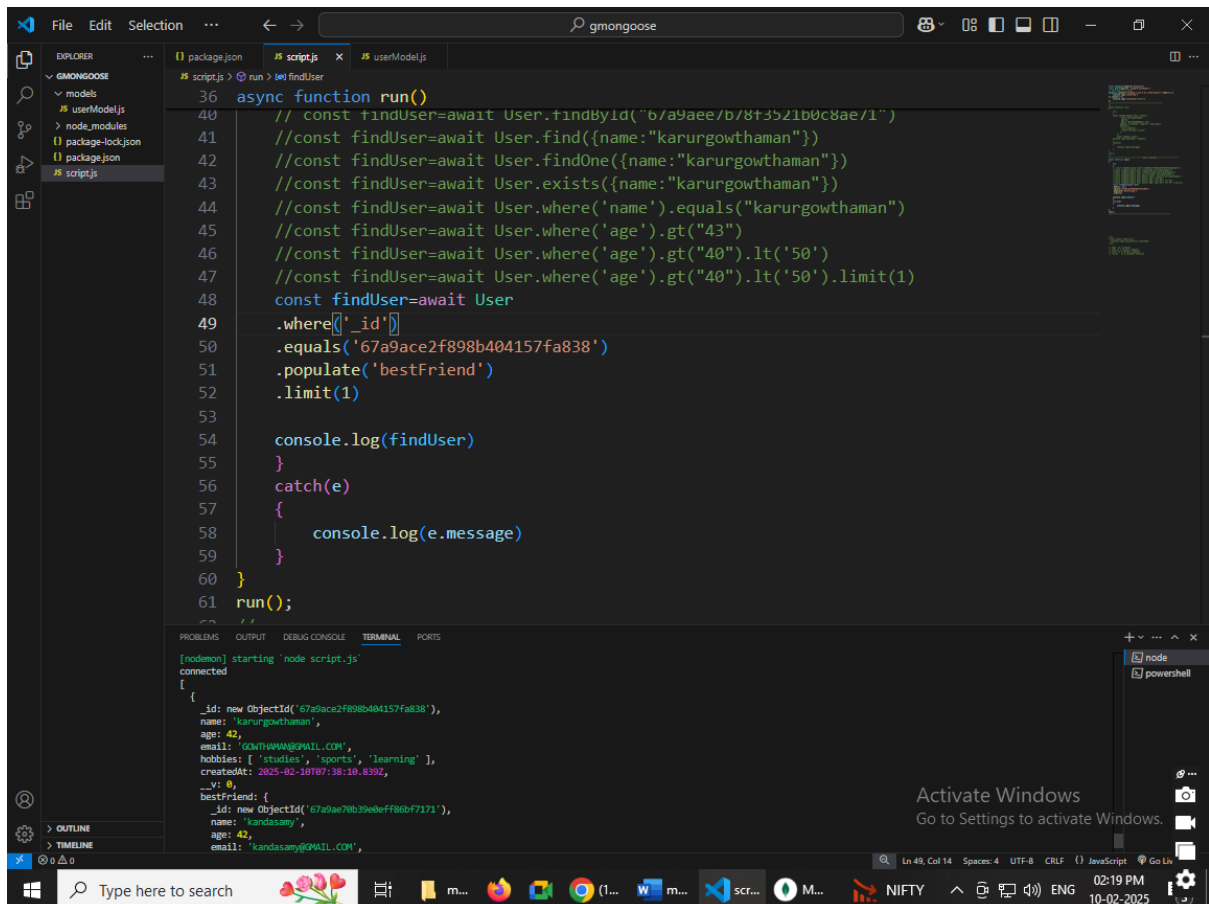
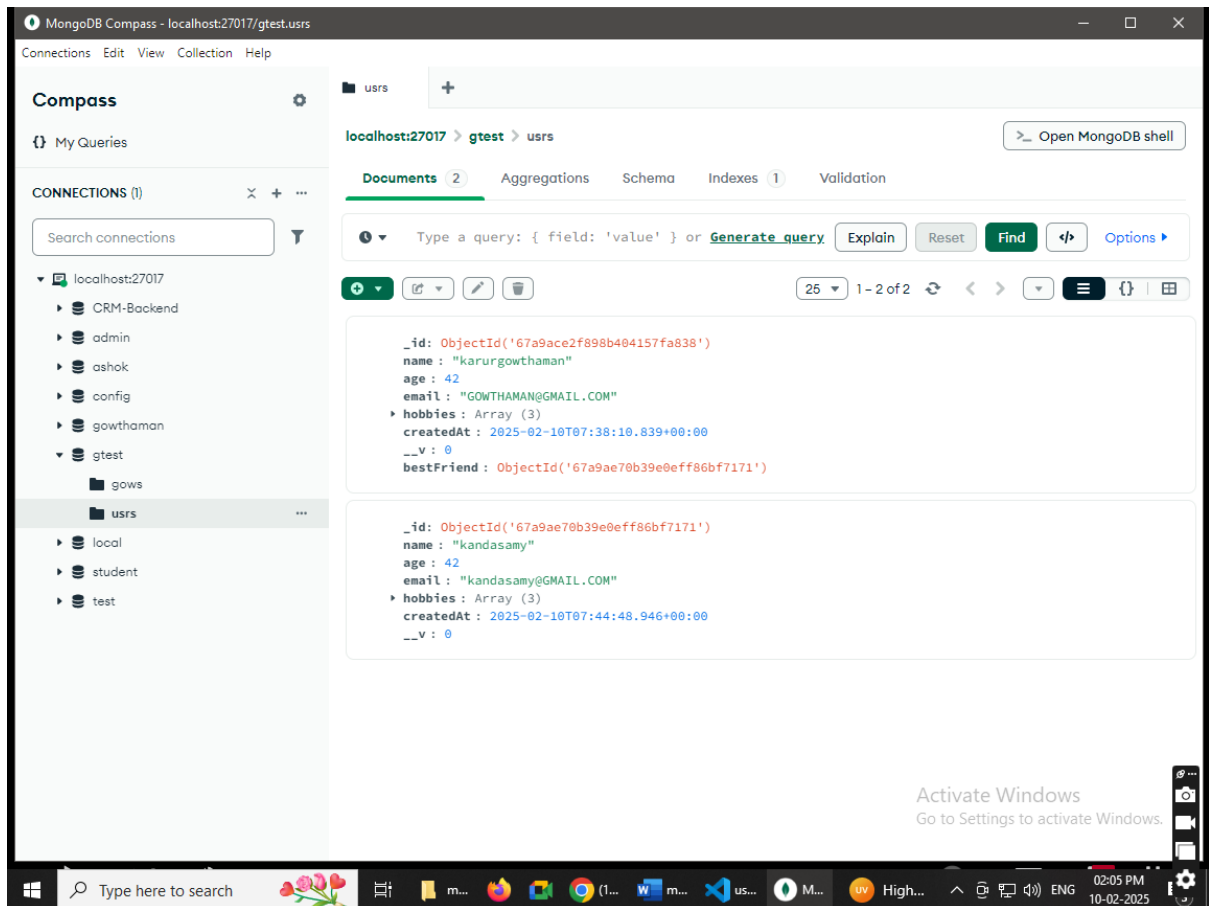
```
const findAllUsers=await User.where('name').equals("karurgowthaman")
```

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

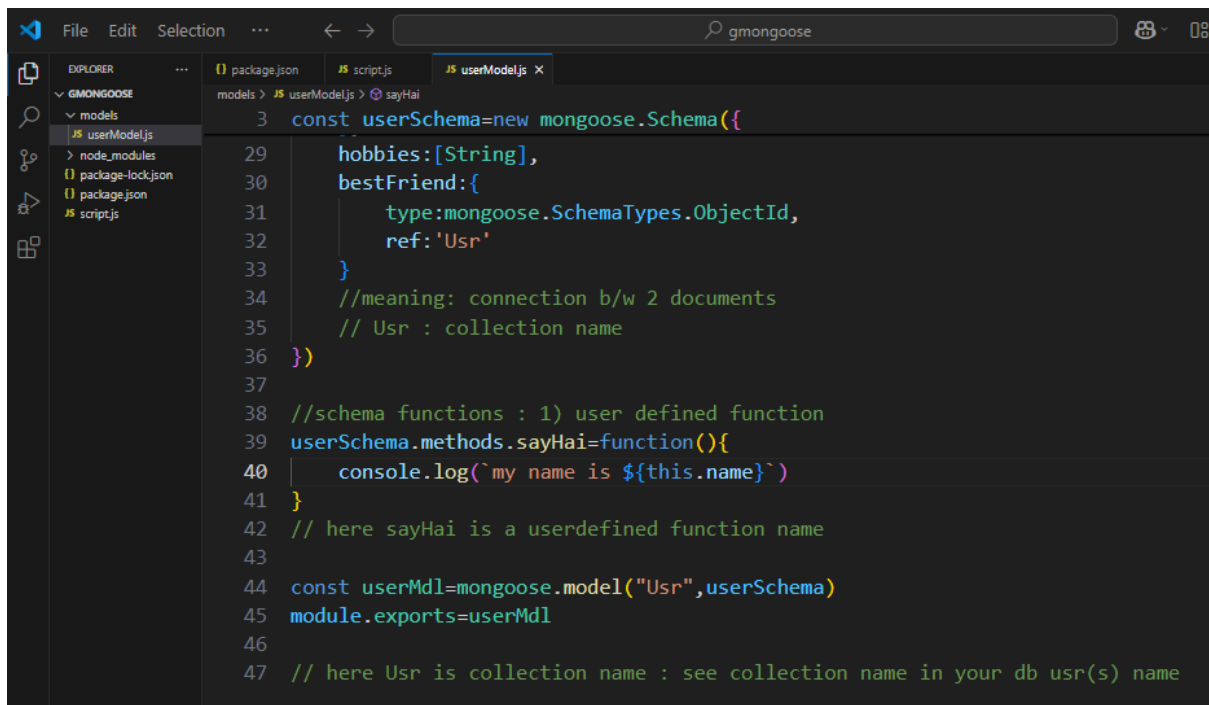
```
const findUser=await User.where('age').gt("40").lt('50')
```

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





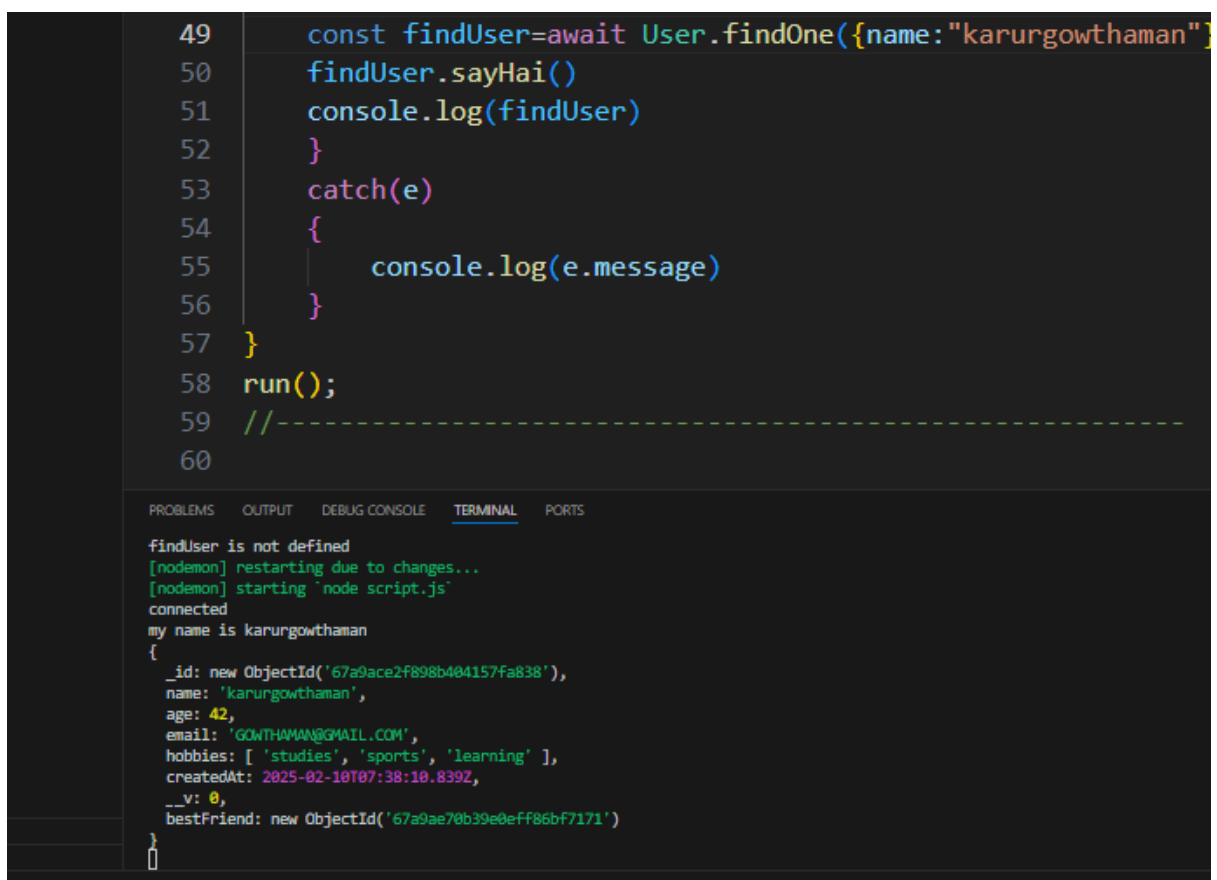
Schema Methods:



```

3  const userSchema=new mongoose.Schema({
29    hobbies:[String],
30    bestFriend:{
31      type:mongoose.SchemaTypes.ObjectId,
32      ref:'usr'
33    }
34    //meaning: connection b/w 2 documents
35    // usr : collection name
36  })
37
38  //schema functions : 1) user defined function
39  userSchema.methods.sayHai=function(){
40    console.log(`my name is ${this.name}`)
41  }
42  // here sayHai is a userdefined function name
43
44  const userMdl=mongoose.model("usr",userSchema)
45  module.exports=userMdl
46
47  // here usr is collection name : see collection name in your db usr(s) name

```



```

49  const findUser=await User.findOne({name:"karurgowthaman"})
50  findUser.sayHai()
51  console.log(findUser)
52  }
53  catch(e)
54  {
55    console.log(e.message)
56  }
57  }
58  run();
59  //-----
60

```

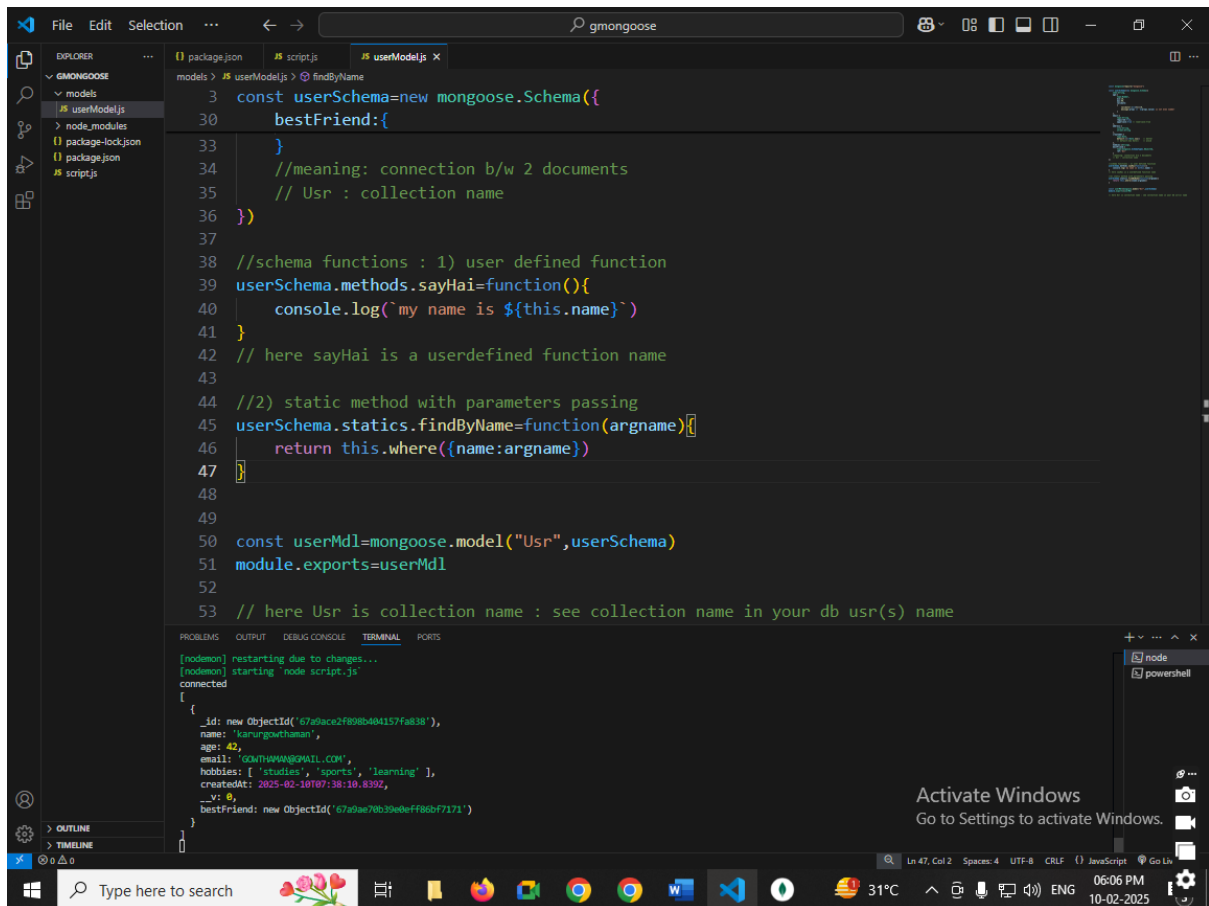
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

findUser is not defined
[nodemon] restarting due to changes...
[nodemon] starting 'node script.js'
connected
my name is karurgowthaman
{
  _id: new ObjectId('67a9ace2f898b404157fa838'),
  name: 'karurgowthaman',
  age: 42,
  email: 'GOWTHAMAN@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:38:10.839Z,
  __v: 0,
  bestFriend: new ObjectId('67a9ae70b39e0eff86bf7171')
}

```

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



The image shows a Visual Studio Code editor window with a file explorer on the left. The main editor displays a JavaScript file named `userModel.js` with the following code:

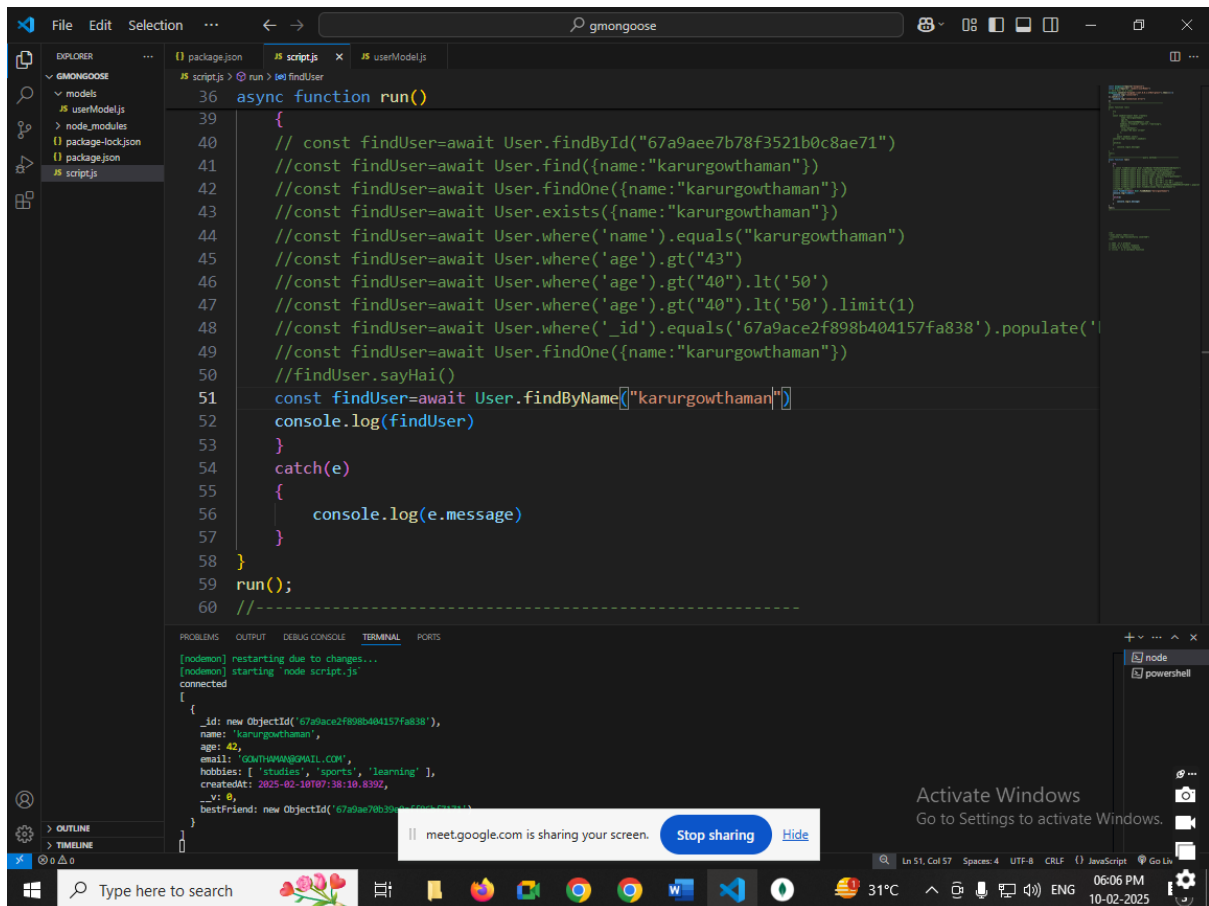
```
3 const userSchema=new mongoose.Schema({
30   bestFriend:{
33   }
34   //meaning: connection b/w 2 documents
35   // Usr : collection name
36 })
37
38 //schema functions : 1) user defined function
39 userSchema.methods.sayHai=function(){
40   console.log(`my name is ${this.name}`)
41 }
42 // here sayHai is a userdefined function name
43
44 //2) static method with parameters passing
45 userSchema.statics.findByName=function(argname){
46   return this.where({name:argname})
47 }
48
49
50 const userMdl=mongoose.model("Usr",userSchema)
51 module.exports=userMdl
52
53 // here Usr is collection name : see collection name in your db usr(s) name
```

The terminal window at the bottom shows the output of the code execution:

```
[nodemon] restarting due to changes...
[nodemon] starting 'node script.js'
connected
{
  _id: new ObjectId('67a8ace2f898b404157fa838'),
  name: 'karungowthaman',
  age: 42,
  email: 'GOWTHAMAN@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:38:10.830Z,
  __v: 0,
  bestfriend: new ObjectId('67a8ae70b39e6ff86b7171')
}
```

The status bar at the bottom indicates the current line is 47, column 2, in UTF-8 encoding, with CRLF line endings. The system tray shows the date as 10-02-2025 and the time as 06:06 PM.

Lineno:46 use find keyword



```

36 async function run()
37 {
38   // const findUser=await User.findById("67a9aee7b78f3521b0c8ae71")
39   //const findUser=await User.find({name:"karurgowthaman"})
40   //const findUser=await User.findOne({name:"karurgowthaman"})
41   //const findUser=await User.exists({name:"karurgowthaman"})
42   //const findUser=await User.where('name').equals("karurgowthaman")
43   //const findUser=await User.where('age').gt("43")
44   //const findUser=await User.where('age').gt("40").lt('50')
45   //const findUser=await User.where('age').gt("40").lt('50').limit(1)
46   //const findUser=await User.where('_id').equals('67a9ace2f898b404157fa838').populate('bestFriend')
47   //const findUser=await User.findOne({name:"karurgowthaman"})
48   //findUser.sayHai()
49   const findUser=await User.findByName("karurgowthaman")
50   console.log(findUser)
51 }
52 catch(e)
53 {
54   console.log(e.message)
55 }
56 }
57 run();
58 //-----

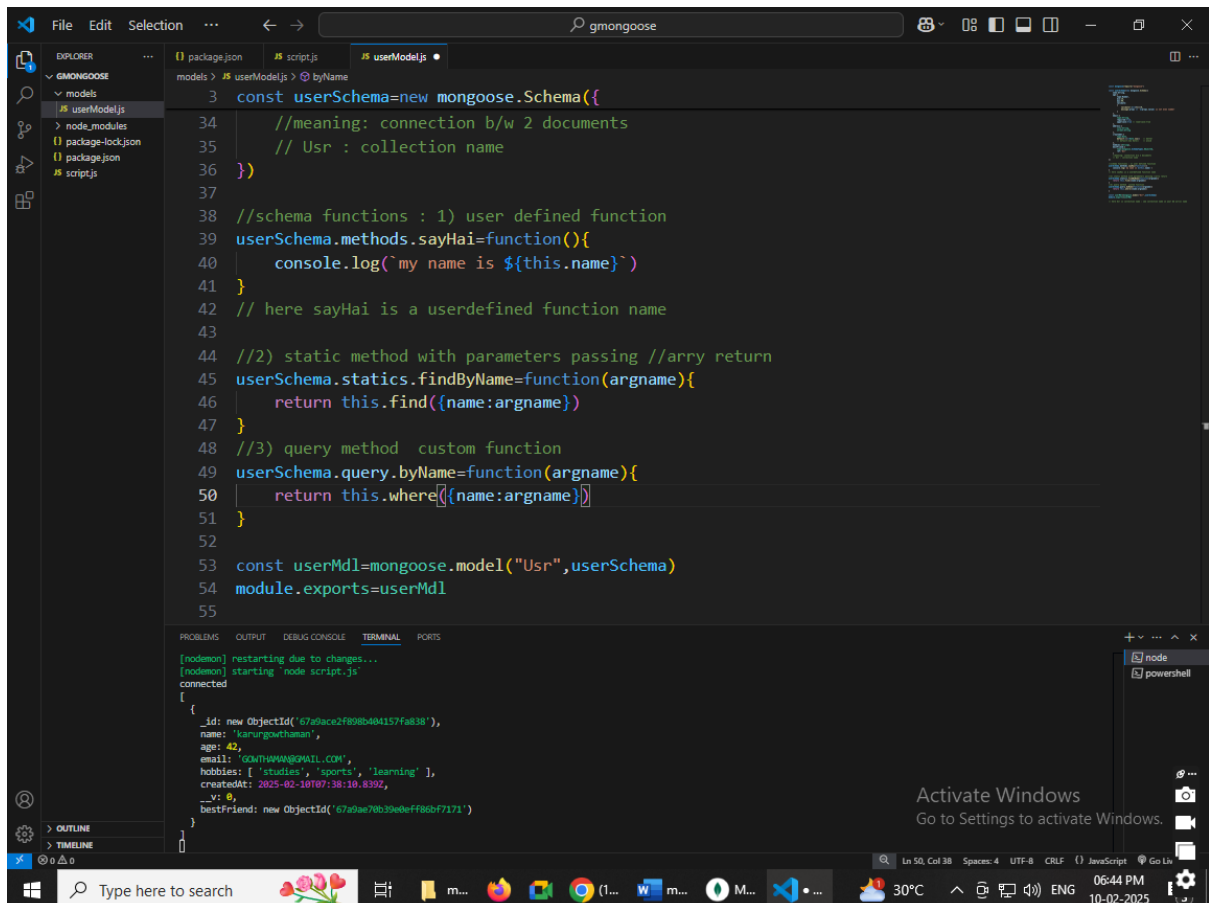
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

[nodemon] restarting due to changes...
[nodemon] starting 'node script.js'
connected
{
  _id: new ObjectId('67a9ace2f898b404157fa838'),
  name: 'karurgowthaman',
  age: 42,
  email: 'GOWTHAMAN@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:38:10.839Z,
  _v: 0,
  bestFriend: new ObjectId('67a9aee7b78f3521b0c8ae71')
}

```



```

34 //meaning: connection b/w 2 documents
35 // Usr : collection name
36 })
37
38 //schema functions : 1) user defined function
39 userSchema.methods.sayHai=function(){
40   console.log(`my name is ${this.name}`)
41 }
42 // here sayHai is a userdefined function name
43
44 //2) static method with parameters passing //arry return
45 userSchema.statics.findByName=function(argname){
46   return this.find({name:argname})
47 }
48 //3) query method custom function
49 userSchema.query.byName=function(argname){
50   return this.where({name:argname})
51 }
52
53 const userMdl=mongoose.model("Usr",userSchema)
54 module.exports=userMdl
55

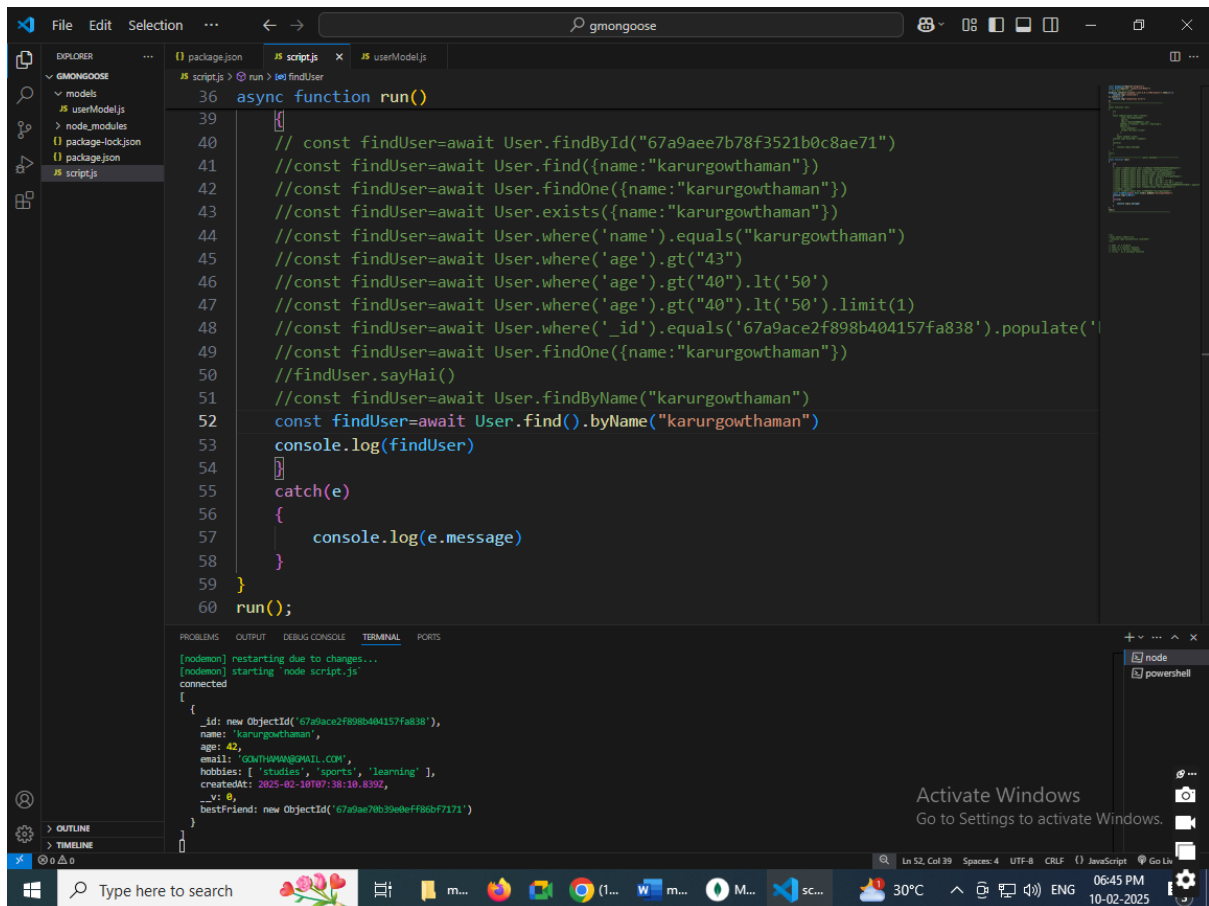
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

[nodemon] restarting due to changes...
[nodemon] starting 'node script.js'
connected
{
  _id: new ObjectId('67a9ace2f898b404157fa838'),
  name: 'karurgowthaman',
  age: 42,
  email: 'GOWTHAMAN@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:38:10.839Z,
  _v: 0,
  bestFriend: new ObjectId('67a9aee7b78f3521b0c8ae71')
}

```



```
36 async function run()
37 {
38   // const findUser=await User.findById("67a9aee7b78f3521b0c8ae71")
39   //const findUser=await User.find({name:"karurgowthaman"})
40   //const findUser=await User.findOne({name:"karurgowthaman"})
41   //const findUser=await User.exists({name:"karurgowthaman"})
42   //const findUser=await User.where('name').equals("karurgowthaman")
43   //const findUser=await User.where('age').gt("43")
44   //const findUser=await User.where('age').gt("40").lt('50')
45   //const findUser=await User.where('age').gt("40").lt('50').limit(1)
46   //const findUser=await User.where('_id').equals('67a9ace2f898b404157fa838').populate('bestfriend')
47   //const findUser=await User.findOne({name:"karurgowthaman"})
48   //findUser.sayHai()
49   //const findUser=await User.findByName("karurgowthaman")
50   const findUser=await User.find().byName("karurgowthaman")
51   console.log(findUser)
52 }
53 catch(e)
54 {
55   console.log(e.message)
56 }
57 }
58 run();
```

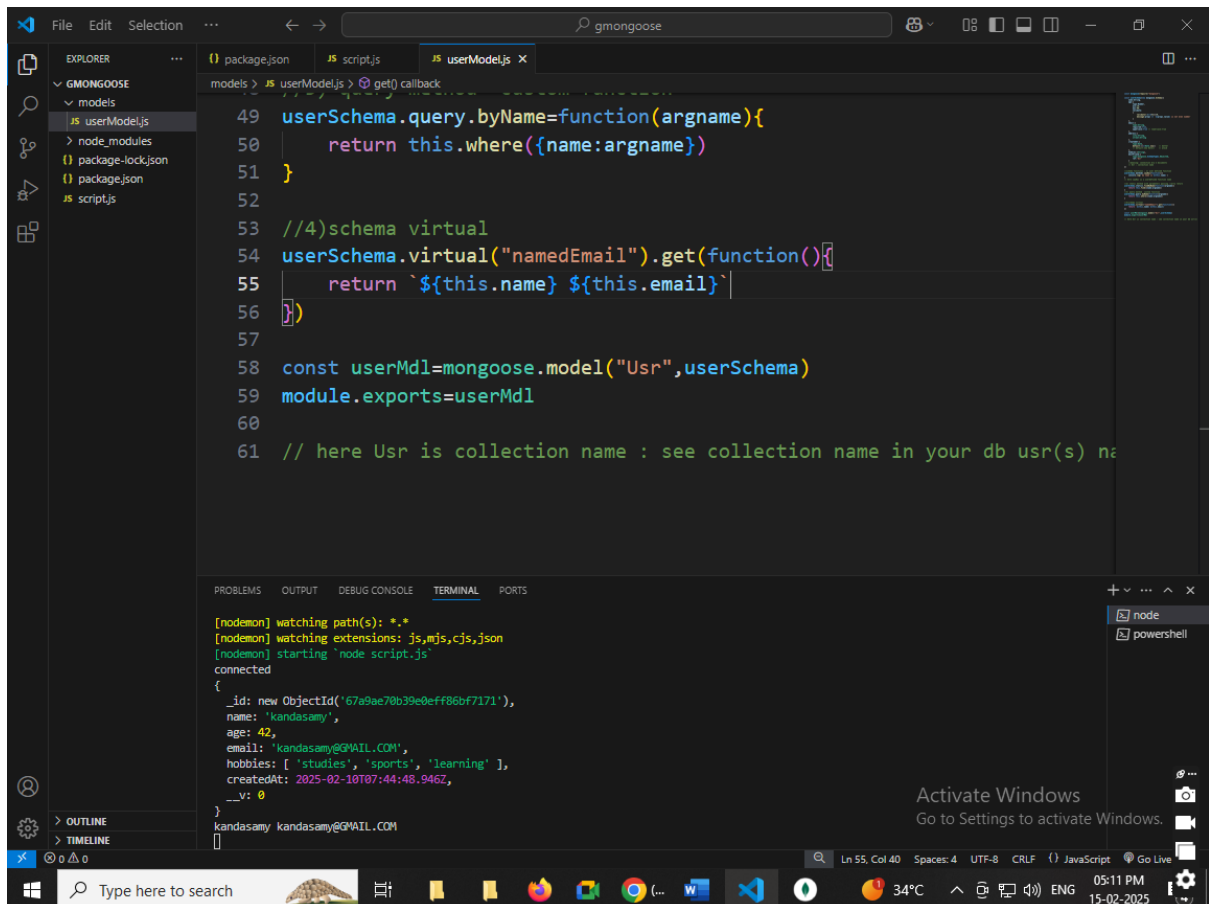
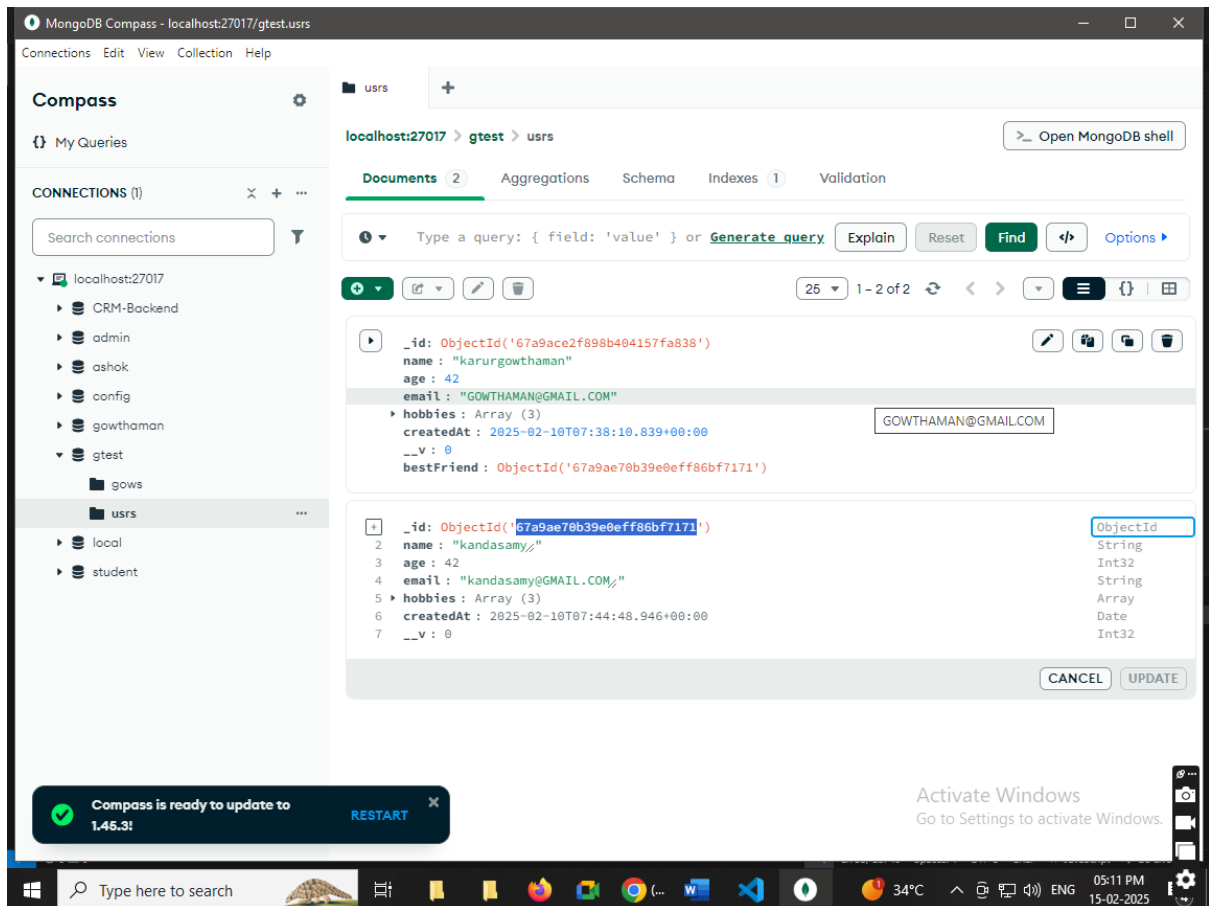
```
[nodemon] restarting due to changes...
[nodemon] starting 'node script.js'
connected
{
  _id: new ObjectId('67a9ace2f898b404157fa838'),
  name: 'karurgowthaman',
  age: 42,
  email: 'GOWTHAMAN@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:38:10.830Z,
  __v: 0,
  bestfriend: new ObjectId('67a9a7b039debeff86b7171')
}
```

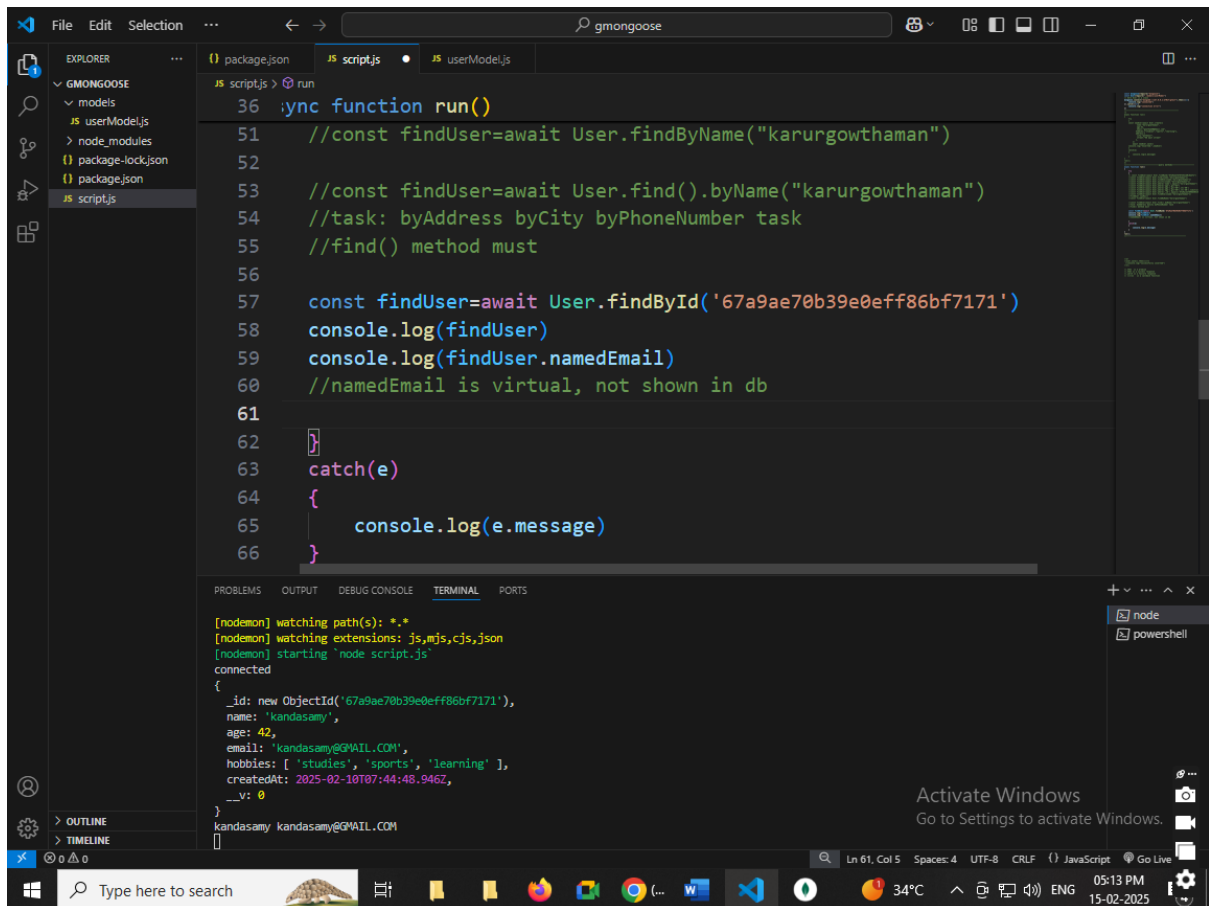
Schema Virtuals:

Meaning : Adding new field

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

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





The screenshot shows a Visual Studio Code editor with a project named 'gmongoose'. The Explorer sidebar on the left shows the file structure: 'models' folder containing 'userModel.js', and 'scripts' folder containing 'script.js'. The 'script.js' file is open in the editor, showing a JavaScript function 'run()' that uses Mongoose to find a user by ID. The code is as follows:

```
36 .ync function run()  
51 //const findUser=await User.findByName("karurgowthaman")  
52  
53 //const findUser=await User.find().byName("karurgowthaman")  
54 //task: byAddress byCity byPhoneNumber task  
55 //find() method must  
56  
57 const findUser=await User.findById('67a9ae70b39e0eff86bf7171')  
58 console.log(findUser)  
59 console.log(findUser.namedEmail)  
60 //namedEmail is virtual, not shown in db  
61  
62 }  
63 catch(e)  
64 {  
65     console.log(e.message)  
66 }
```

The TERMINAL panel at the bottom shows the output of the script execution:

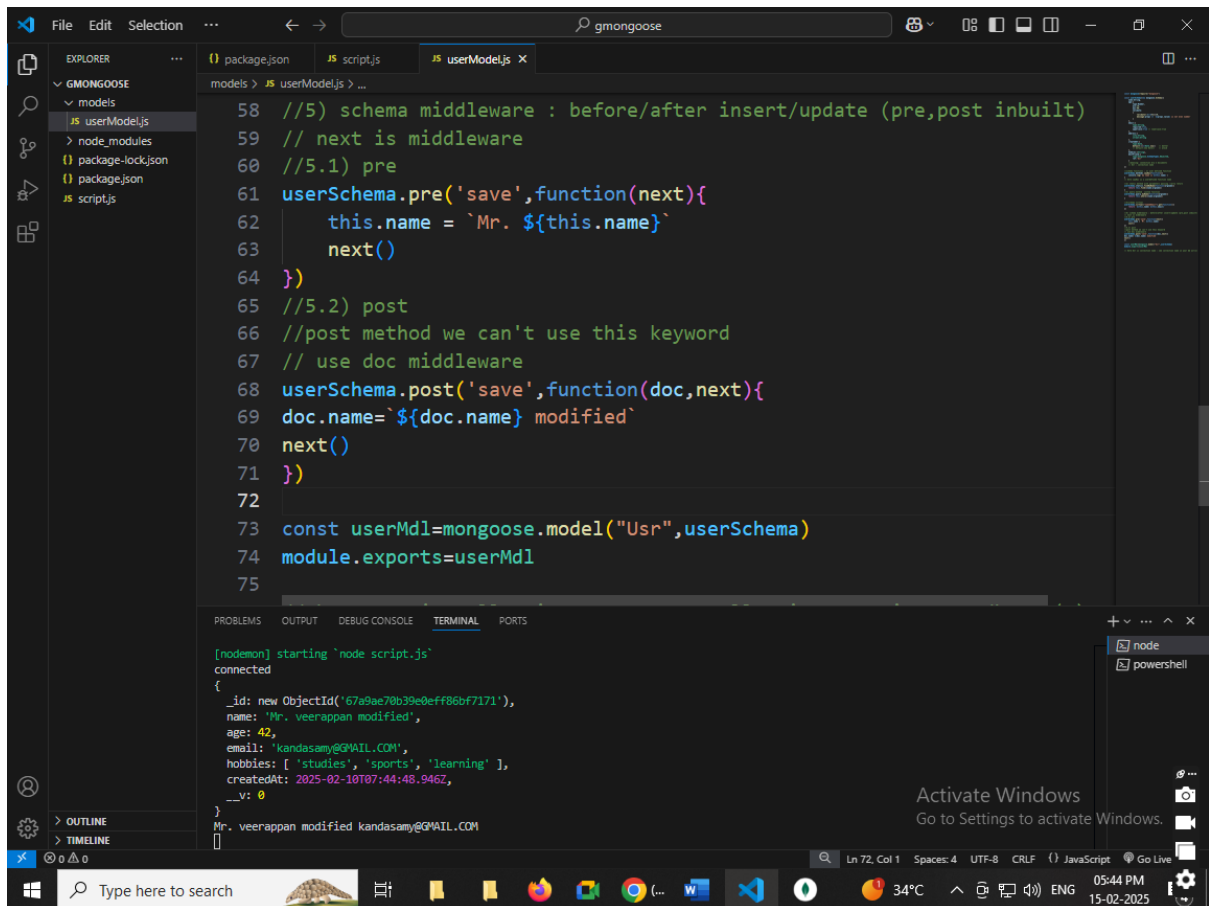
```
[nodemon] watching path(s): *.*  
[nodemon] watching extensions: js,mjs,cjs,json  
[nodemon] starting 'node script.js'  
connected  
{  
  _id: new ObjectId('67a9ae70b39e0eff86bf7171'),  
  name: 'kandasamy',  
  age: 42,  
  email: 'kandasamy@GMAIL.COM',  
  hobbies: [ 'studies', 'sports', 'learning' ],  
  createdAt: 2025-02-10T07:44:48.946Z,  
  __v: 0  
}  
kandasamy kandasamy@GMAIL.COM
```

The status bar at the bottom indicates the cursor is at line 61, column 5, with 4 spaces, UTF-8 encoding, and CRLF line endings. The language is set to JavaScript.

Schema Middlewares:

Meaning: before insert / after insert some changes

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



```

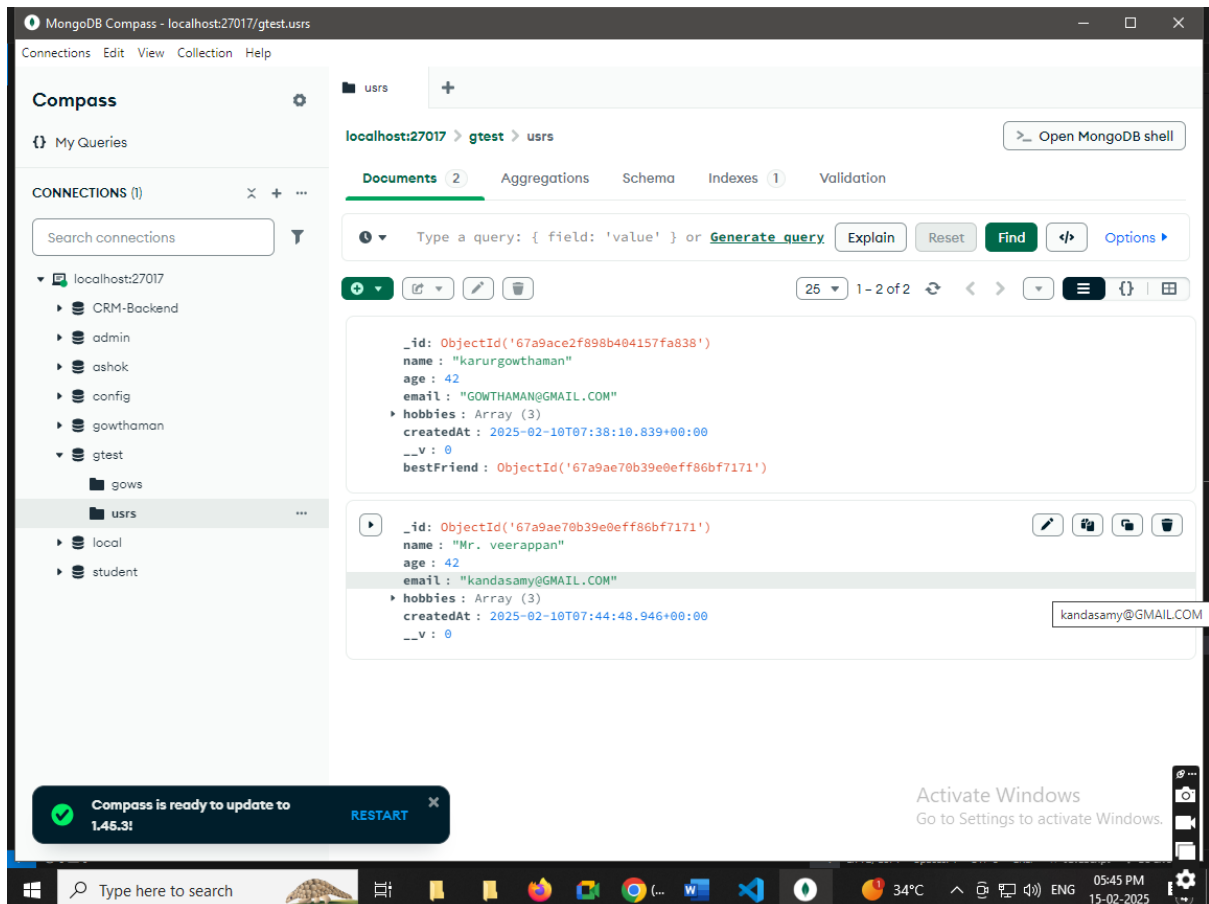
58 //5) schema middleware : before/after insert/update (pre,post inbuilt)
59 // next is middleware
60 //5.1) pre
61 userSchema.pre('save',function(next){
62   this.name = `Mr. ${this.name}`
63   next()
64 })
65 //5.2) post
66 //post method we can't use this keyword
67 // use doc middleware
68 userSchema.post('save',function(doc,next){
69   doc.name=`${doc.name} modified`
70   next()
71 })
72
73 const userMdl=mongoose.model("Usr",userSchema)
74 module.exports=userMdl
75

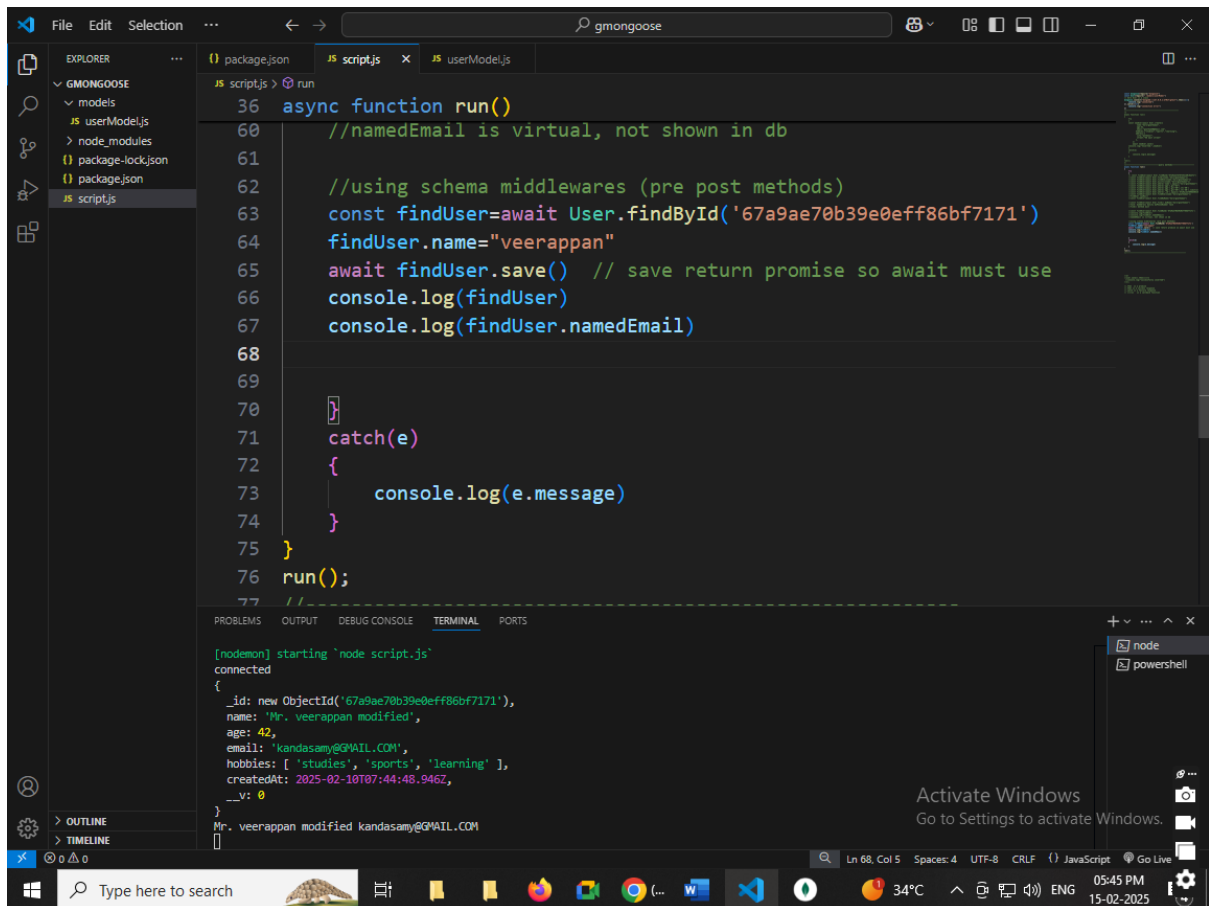
```

```

[nodemon] starting 'node script.js'
connected
{
  _id: new ObjectId('67a9ae70b39e0eff86bf7171'),
  name: 'Mr. veerappan modified',
  age: 42,
  email: 'kandasamy@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:44:48.946Z,
  __v: 0
}
Mr. veerappan modified kandasamy@GMAIL.COM

```





```
36 async function run()
60   //namedEmail is virtual, not shown in db
61
62   //using schema middlewares (pre post methods)
63   const findUser=await User.findById('67a9ae70b39e0eff86bf7171')
64   findUser.name="veerappan"
65   await findUser.save() // save return promise so await must use
66   console.log(findUser)
67   console.log(findUser.namedEmail)
68
69
70 }
71 catch(e)
72 {
73   console.log(e.message)
74 }
75 }
76 run();
77 //
```

```
[nodemon] starting 'node script.js'
connected
{
  _id: new ObjectId('67a9ae70b39e0eff86bf7171'),
  name: 'Mr. veerappan modified',
  age: 42,
  email: 'kandasamy@GMAIL.COM',
  hobbies: [ 'studies', 'sports', 'learning' ],
  createdAt: 2025-02-10T07:44:48.946Z,
  __v: 0
}
Mr. veerappan modified kandasamy@GMAIL.COM
```

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