Task1
This was solved by adding the line:
```
app.use(express.static('client'))
```
This allows us to use the index.html inside the client folder as our frontpage

Task2
In the model folder I have created a schema for the users, it consist of a name that is unique and required and age that must be a number, the _id is auto generated from mongodb
```
const userSchema = new mongoose.Schema({
name: {
type: String,
required: true,
unique: true,
min: 1,
max: 255
},
age: {
type: Number,
required: true
}
})
```

Task3
I created a folder for all the routes called routes and inside that a file called users.js that holds all the api endpoints. Each endpoint is surrounded with try/catch to avoid server breakdown from bad input and I choose to respond with json object when this occurs because I find it easier to show it in the frontend.
```
router.get("/", async (req, res) => {
try {
const users = await User.find()
res.status(200).send(users)
} catch (error) {
signale.error(error.name)
res.status(400).send({ "code": 400, "method": "get", "message": error.name })
}
})
```

I import the router to the main JS file with the following line :
```
app.use('/api/user', userRoute);
```
And export it with
```
module.exports = router
```

Task4
The html file is quite empty and only the static elements and a empty div exist. I make heavy use of literals to populate the empty div with the data from api, I like this solution as it is very easy to plug in the data. I had some trouble configuring the buttons however. The solution was to put the user._id in as the buttons id so I can use that to call the onclick function and have a reference to the user to modify.

```
function template(user) {
return (
`
<div class="user-container">
<input type="text" id="name" name="name" placeholder="${user.name}">
<input type="text" id="age" name="age" placeholder="${user.age}">
<button id="u${user._id}" onclick="update(id);">Update</button>
<button id="${user._id}" onclick="deleteUser(id)">Delete</button>
</div>
`
)
}
```

I used fetch to interact with the api.
The table like structure was achieved with css grid.