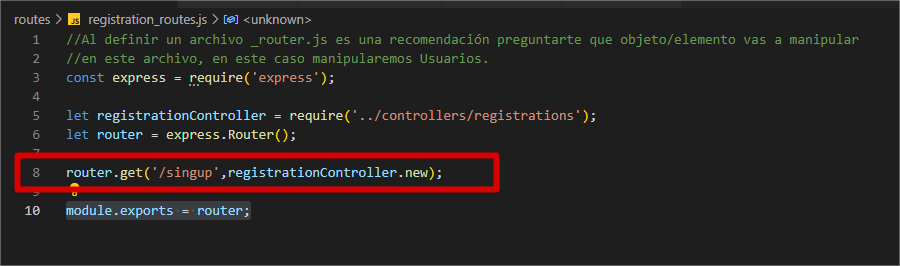
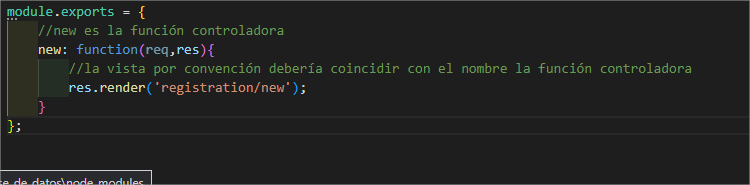
Express

1. Creación del archivo “routes”. Luego definición de la ruta.

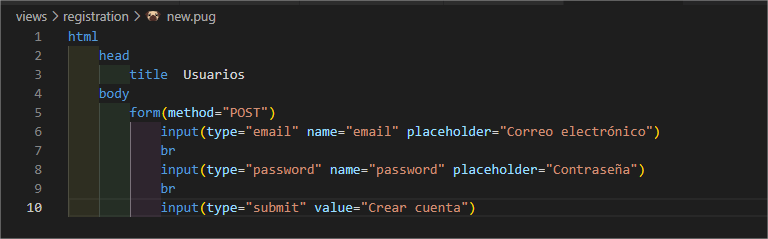


Notar que es necesario definir la URL. Esto se hacer con el objeto “router” llamando al verbo http necesario. En este caso “GET”. Lleva 2 parametros la ruta obviamente y luego el método controlador que en este caso es registrationController.new.

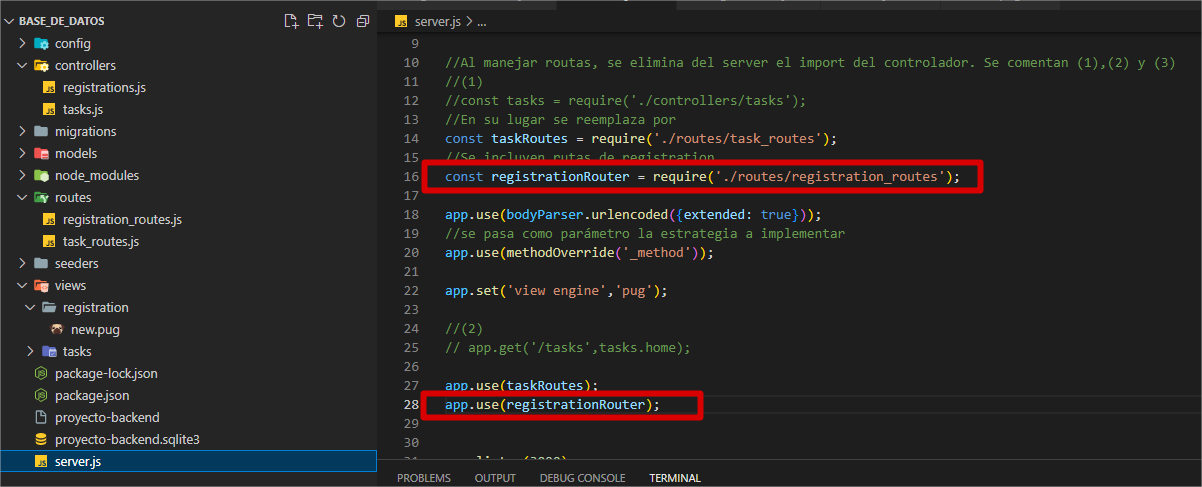
1. Definir el método controlador :



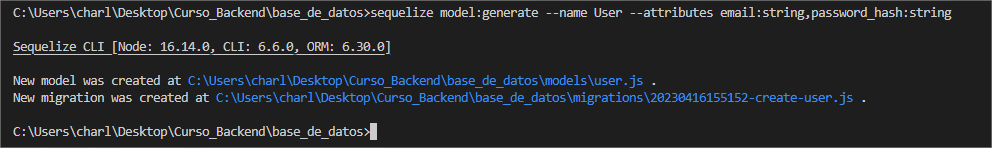
1. Definir la vista



1. Exportar en server.js

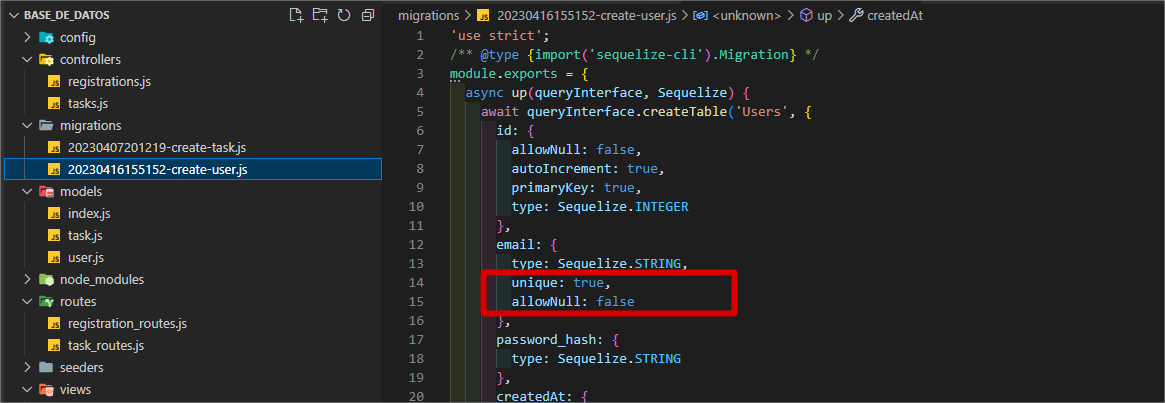


1. Generación del modelo “Usuarios” mediante el cli de sequelize. Advertir que genera un nuevo modelo y una nueva migración.

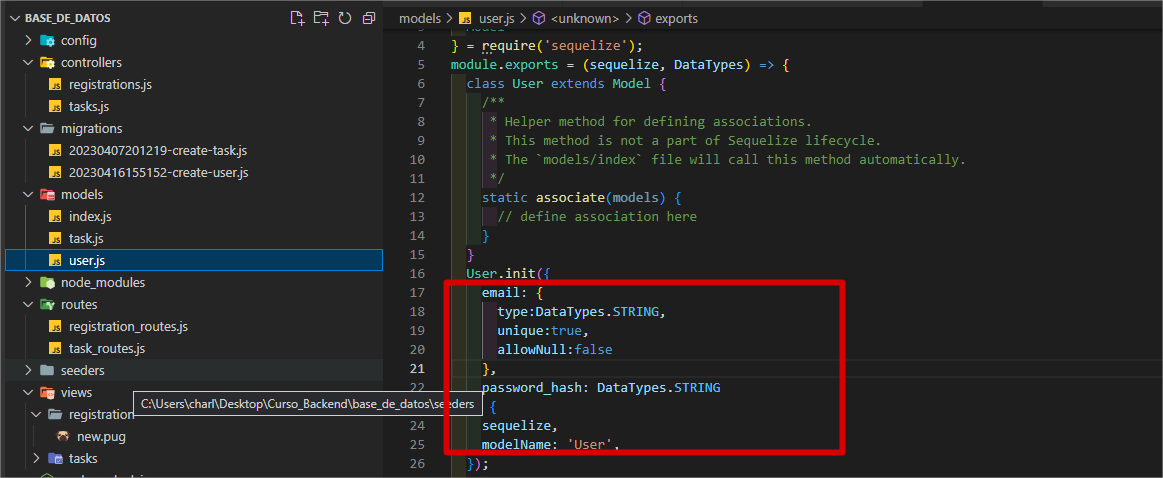


1. \*\*\*IMPORTANTE\*\*\* Se deben incluir validaciones tanto a nivel del modelo como a nivel de la base de datos. ¿Por qué? ¿Por qué no es suficiente solo con incluirlas a nivel modelo? Porque por ejemplo si 2 usuarios intentan en simultaneo dar de alta el mismo usuario es posible que nuestra aplicación con las validaciones únicamente a nivel modelo no llegue a validar unicidad. Por ello, la base de datos se encuentra totalmente preparada para controlar concurrencia y podría rechazar esta situación perfectamente.

Validaciones a nivel base de datos en archivo de migraciones



Validaciones a nivel modelo en archivo de la carpeta modelos.



1. Encriptación de la contraseña. Se importa la biblioteca “bcrypt”. Para esta operación es necesario utilizar “hooks”. ¿Qué mierda es esto? Los hooks son eventos que se disparan antes o después de realizar una determinada operación.
2. Creación del usuario. Lo primero es definir una nueva ruta a donde se enviarán los datos del formulario. ¿Donde? Carpeta routes>registration\_routes.js.
3. Lo siguiente definir la función controladora, tener en cuenta que en el formulario enviarás 2 parámetros, el usuario y la contraseña. Entonces, en la función controladora debes crear un estructura de datos JSON que soporte los mismos.

|  |  |
| --- | --- |
|  |  |