Clinfy - Pruebas unitarias

Users

Componente probado	Users Service
Archivo del test	users.service.spec.ts
Estado	Passed 9 of 9
Endpoints/Funciones a probar	
canDo	- Debe retornar true cuando el usuario tiene el permiso consultado - Debe retornar UnauthorizedException cuando el permiso consultado no está asignado al usuario
refreshToken	- Debe llamar a JWT service para refrescar el token de acceso
login	- Debe retornar accesToken y refreshToken cuando las credenciales ingresadas son válidas - Debe retornar NotFoundException cuando el email es incorrecto - Debe retornar UnauthorizedException cuando la contraseña es incorrecta
findByEmail	- Debe retornar un usuario cuando el email es válido y está registrado en la BD - Debe retornar NotFoundException cuando el email no pertenece a ningún usuario
register	- Debe registrar un nuevo usuario con su respectivo email y contraseña

---

Consultorio

Componente probado	Consultorio Controller
Archivo del test	consultorio.controller.spec.ts
Estado	Passed 22 of 22
Endpoints/Funciones a probar	
consultorios/new	- Debe crear un consultorio - Debe lanzar un error cuando el número de consultorio ya existe - Debe lanzar error cuando el número es invalido - Debe lanzar error cuando las observaciones están vacías - Debe lanzar un error cuando el body recibe campos extra
consultorios/edit/:id	- Debe actualizar un consultorio exitosamente - Debe actualizar solo el número del consultorio - Debe actualizar solo las observaciones del consultorio - Debe lanzar error cuando el consultorio no existe - Debe lanzar error cuando el id es inválido - Debe lanzar error si el número de consultorio ya existe - Debe lanzar error si el body está vacío
consultorios/delete/:id	- Debe eliminar un consultorio - Debe lanzar error cuando el consultorio no existe - Debe lanzar error si el consultorio tiene empleados asociados
consultorios/all	- Debe retornar todos los consultorios con su estructura correcta - Debe retornar un array vacío si no hay consultorios creados
consultorios/:id	- Debe retornar un consultorio con su estructura correcta por ID

	- Debe lanzar error si el consultorio no existe

Componente probado	Consultorio Service
Archivo del test	consultorio.service.spec.ts
Estado	Passed 22 of 22
Endpoints/Funciones a probar	
create	- Debe crear un consultorio - Debe lanzar un error cuando el número de consultorio ya existe - Debe lanzar error cuando el número es invalido - Debe lanzar error cuando las observaciones están vacías - Debe lanzar error cuando el número es inválido
edit	- Debe actualizar un consultorio exitosamente - Debe actualizar solo el número del consultorio - Debe lanzar error cuando el consultorio no existe - Debe lanzar error si el nro de consultorio ya existe
delete	- Debe eliminar un consultorio - Debe lanzar error cuando el consultorio no existe - Debe lanzar error si el consultorio tiene empleados asociados
findAll	- Debe retornar todos los consultorios con su estructura correcta - Debe retornar un array vacío si no hay consultorios creados
findOne	- Debe retornar un consultorio con su estructura correcta por ID - Debe lanzar NotFoundException si el consultorio no existe

Turno

Componente probado	Turno Controller
Archivo del test	turno.controller.spec.ts
Estado	Passed 14 of 14
Endpoints/Funciones a probar	
turnos/agendar	- Debe agendar un turno correctamente - Debe fallar cuando la fecha es inválida - Debe fallar cuando el motivo está vacío - Debe fallar cuando fechaHoraTurno o motivo no es una cadena - Debe fallar cuando procedimiento no es un número - Debe fallar cuando doctor no es un número - Debe fallar cuando faltan campos requeridos - Debe fallar cuando el DTO es nulo

---

Componente probado	Turno Service
Archivo del test	turno.service.spec.ts
Estado	Passed 16 of 16
Endpoints/Funciones a probar	
agendarTurno	- Debe agendar correctamente un turno - Debe lanzar ConflictException cuando la fecha ingresada es inválida o cuando el doctor ya tiene un turno agendado - Debe detectar solapamiento cuando se registra el turno sobre otro que aún no terminó - Debe lanzar error cuando procedimiento, doctor, paciente o especialidad no existe - Debe manejar errores de transacción de la base de datos - Debe manejar errores al guardar el turno