PLAN DE PRUEBAS

**GRUPO 1: Daniel Santano, José Luis Díez,
Miguel Lázaro y José Antonio Llamas**

MÉTODOS A TESTEAR

Servicios:

- Espacio:

  public Page<Espacio> findAllPageable(Pageable pageable)

  Asignado a Daniel Santano (realizado)

  Mockeamos el repositorio y diciéndole que devuelva una lista con espacios comprobamos que devuelve el Page<Espacio> y que este no es nulo dado que no debería serlo nunca.

- Reserva:

  public List<LocalTime> getHorarios()

  Asignado a Miguel Lázaro(realizado)

  Paso como resultado la lista que hay en el servicio de Reservas y comparo con la lista que me devuelve el servicio

  public List<Reserva> buscarPorUsuarioId(**long** id)

  Asignado a Miguel Lázaro(realizado)

  Mockeo el repositorio haciendo que me devuelva una lista que previamente creo y compruebo que el servicio me devuelva esa lista.

  public List<Reserva> buscarPorFechaYEspacio(LocalDate fecha, Espacio espacio)

  Asignado a Miguel Lázaro(realizado)

  Mockeo el repositorio haciendo que me devuelva una lista que previamente creo y compruebo que el servicio me devuelva esa lista.

  public List<LocalTime> buscarHorasDisponibles(List<LocalTime> horarios, LocalDate fecha, Espacio espacio)

  Asignado a Miguel Lázaro(realizado)

  Mockeo el repositorio haciendo que me devuelva una lista que previamente creo y compruebo que el servicio me devuelva esa lista.

- Usuario:

  public Usuario buscarPorEmail(String email)

  Asignado a José Luis Díez (realizado)

  Comprobar que si buscamos un email que existe nos devuelva el usuario de ese email

  Comprobar que si buscamos un email que no existe nos devuelva null

  public List<Usuario> findByEmail(String email)

  Asignado a José Luis Díez (realizado)

  Comprobar que si buscamos un email que existe nos devuelva la lista de usuarios con ese email

  Comprobar que si buscamos un email que no existe nos devuelva la lista vacía

  public String cifrarPassword(String password)

  Asignado a José Antonio Llamas (realizado)

  Comprobar que si se le pasan al encoder dos cadenas de caracteres iguales, las cadenas encriptadas también son iguales.

  Comprobar que si se le pasan al encoder dos cadenas de caracteres diferentes, las cadenas encriptadas no son iguales.

  public void validarRegistro(Usuario usuario)

  Asignado a José Antonio Llamas (realizado)

  Una vez creado un usuario con el atributo de registroConfirmado con valor False, comprobar que después de usar el método, el usuario tiene el atributo con valor True.

  public List<Usuario> listarUsuariosPendientes()

  Asignado a José Luis Díez (realizado)

  Comprobar que al buscar los usuarios pendientes nos devuelva la misma lista de usuarios pendientes que en el repositorio

  public List<Usuario> listarUsuariosActivos()

  Asignado a José Luis Díez (realizado)

  Comprobar que al buscar los usuarios activos nos devuelva la misma lista de usuarios pendientes que en el repositorio

public void desactivarUsuario(Long id)

Asignado a José Antonio Llamas (realizado por Miguel Lázaro)

Una vez creado un usuario con el atributo de Activo con valor True, mockeamos el repositorio pasando un usuario anteriormente creado, comprobar que después de usar el método, el usuario tiene el atributo con valor False.

public List<Usuario> listarUsuariosInactivos()

Asignado a José Luis Díez (realizado)

Comprobar que al buscar los usuarios inactivos nos devuelva la misma lista de usuarios pendientes que en el repositorio

public void activarUsuario(Usuario usuario)

Asignado a José Luis Díez (realizado)

Una vez creado un usuario con el atributo de Activo con valor False, comprobar que después de usar el método, el usuario tiene el atributo con valor True.

Repositorios:

- Reserva:

public List<Reserva> findByUsuarioId(**long** id)

Asignado a Miguel Lázaro (realizado)

Compruebo que la lista que me devuelve es vacía ya que en el data.sql no existe ninguna reserva con ese id

public List<Reserva> findByFechaAndEspacio(LocalDate fecha, Espacio espacio)

Asignado a Miguel Lázaro (realizado)

Compruebo que la lista que me devuelve es vacía ya que en el data.sql no existe ninguna reserva con esa fecha y espacio

- Usuario:

Usuario findFirstByEmail(String email)

Asignado a Daniel Santano (realizado)

Una vez añadidos datos con el data.sql y sabiendo que hay, al menos,un usuario en la base de datos y buscando ese usuario sabiendo su email, comprobar que el email indicado y el email del usuario obtenido son iguales.

List<Usuario> findByEmail(String email)

Asignado a Daniel Santano (realizado)

Una vez añadidos datos con el data.sql y sabiendo que hay, varios usuarios buscamos un usuario por su email y comprobamos que la lista que contiene ese usuario tiene el email del usuario igual al usuario indicado.

List<Usuario> findByRegistroConfirmadoFalse()

Asignado a José Antonio Llamas (realizado)

Una vez añadidos datos con el data.sql y sabiendo que hay, al menos, uno con el registro sin confirmar, comprobar que la lista que devuelve la consulta no está vacía.

List<Usuario> findByRegistroConfirmadoTrueAndActivoTrue()

Asignado a José Antonio Llamas (realizado)

Una vez añadidos datos con el data.sql y sabiendo que hay, al menos, uno con el registro confirmado y el estado de activo es True, comprobar que la lista que devuelve la consulta no está vacía.

List<Usuario> findByActivoFalse()

Asignado a Daniel Santano (realizado)

Una vez añadidos datos con el data.sql y sabiendo que hay, varios usuarios buscamos todos los usuarios con el atributo activo siendo False, también sabemos que hay usuarios que cumplen esa condición, luego comprobamos que la lista que devuelve la consulta no está vacía.