**Diseño de pruebas de los servicios**

|  |  |  |
| --- | --- | --- |
| **Pruebas sobre la implementación de com.taller1.service.implementation.ProductServiceImpl**  *PRUEBAS UNITARIAS* | | |
| **Método a probar** | **Paramétro(s)** | **Resultado esperado (Prueba unitaria)** |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | null, null, null, null | El método lanza NullPointerException con mensaje “Producto no puede ser nulo”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product, null, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza NullPointerException con mensaje “No escribió un id de la categoria existente para producto”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product, ID (Integer) de Instancia de Productcategory, null, Instancia de Unitmeasure | El método lanza NullPointerException con mensaje “No escribió un id de la subcategoria existente para producto”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, null | El método lanza NullPointerException con mensaje “No escribió el id de unidad de medida del producto”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Productnumber=”0” y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza Exception con mensaje “Product number must be greater than 0”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Productnumber=”A” y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza NumberFormatException con mensaje “For input string: “A””. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Daystomanufacture=0 y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza Exception con mensaje “Days to manufacture must be greater than 0”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Sellstartdate después de Sellenddate y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza Exception con mensaje “Sell end date is before sell start date”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Sellstartdate y Sellenddate nulos y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza NullPointerException con mensaje “Cannot invoke "java.sql.Timestamp.before(java.sql.Timestamp)" because "Sellenddate" is null”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Sellstartdate=null y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza NullPointerException con mensaje “Cannot invoke "java.sql.Timestamp.getTime()" because "ts" is null”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Sellenddate=null y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza NullPointerException con mensaje “Cannot invoke "java.sql.Timestamp.before(java.sql.Timestamp)" because "Sellenddate" is null”. No hay llamado a ningún método de prodRepository (mock), pero hay llamados a los métodos findById(Integer id) de prodCatRepository (mock) y prodSubRepository (mock). |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método llama correctamente al método save de prodRepository (mock) para guardar el producto asociado a una categoría, subcategoría y unidad de medida y no genera ninguna excepción. |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), null | El método lanza NullPointerException con el mensaje “Cannot invoke "com.taller1.model.prod.Product.getProductnumber()" because "p" is null”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Productnumber menor a 1 y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza Exception con el mensaje “Product number must be greater than 0”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Productnumber=“0” y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza Exception con el mensaje “Product number must be greater than 0”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Productnumber=“2” y el resto de los atributos respetando las condiciones dadas en el enunciado | El método llama correctamente al método save de prodRepository (mock) para guardar el producto actualizado y no genera ninguna excepción. |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product Daystomanufacture=0 y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza Exception con el mensaje “Days to manufacture must be greater than 0”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Daystomanufacture =2 y el resto de los atributos respetando las condiciones dadas en el enunciado | El método llama correctamente al método save de prodRepository (mock) para guardar el producto actualizado y no genera ninguna excepción. |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Sellenddate=null y Sellstartdate=null y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza NullPointerException con mensaje “Cannot invoke "java.sql.Timestamp.before(java.sql.Timestamp)" because "Sellenddate" is null”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Sellenddate antes de Sellstartdate y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza Exception con mensaje “Sell end date is before sell start date”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Sellenddate=null y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza NullPointerException con mensaje “Cannot invoke "java.sql.Timestamp.before(java.sql.Timestamp)" because "Sellenddate" is null”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Sellstartdate=null y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza NullPointerException con mensaje “Cannot invoke "java.sql.Timestamp.getTime()" because "ts" is null”. Hay una interacción con prodRepository (mock), dado que se hace el llamado al método findById(Integer id). |
| updateProduct(Integer pId, Product p) | 1 (ID de un producto existente), Instancia de Product con Sellstartdate antes que Sellenddate y el resto de los atributos respetando las condiciones dadas en el enunciado | El método llama correctamente al método save de prodRepository (mock) para guardar el producto actualizado y no genera ninguna excepción. |
| *PRUEBAS DE INTEGRACIÓN* | | |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | null, null, null, null | El método lanza NullPointerException con mensaje “Producto no puede ser nulo”. |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product, null, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza NullPointerException con mensaje “No escribió un id de la categoria existente para producto”. |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Productnumber=“0” y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza Exception con mensaje “Product number must be greater than 0”. |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con Sellenddate antes de Sellstartdate y el resto de los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método lanza Exception con mensaje “Sell end date is before sell start date”. |
| saveProduct (Product p, Integer pCatId, Integer pSubId, Unitmeasure um) | Instancia de Product con los atributos respetando las condiciones dadas en el enunciado, ID (Integer) de Instancia de Productcategory, ID (Integer) de Instancia de Productsubcategory, Instancia de Unitmeasure | El método se ejecuta correctamente, llama al método save de ProductRepository y queda registrado dentro de la base de datos del CrudRepository (Se verifica con el método findById del repositorio y comparando el Productid y Sellstartdate del encontrado con el insertado) |
| updateProduct(Integer pId, Product p) | Instancia inicial de Product con Productnumber=“1”, Daystomanufacture=1, Sellstartdate=new Timestamp (System.currentTimeMillis()-100), Sellenddate=new Timestamp (System.currentTimeMillis()-100), Productid=1, name=“Name Product”, color=“Blue”, size=“Small”  Parámetros:  1, null | El método lanza NullPointerException con mensaje “Cannot invoke "com.taller1.model.prod.Product.getProductnumber()" because "p" is null”. |
| updateProduct(Integer pId, Product p) | Instancia inicial de Product con Productnumber=“1”, Daystomanufacture=1, Sellstartdate=new Timestamp (System.currentTimeMillis()-100), Sellenddate=new Timestamp (System.currentTimeMillis()-100), Productid=1, name=“Name Product”, color=“Blue”, size=“Small”  Parámetros:  1, Instancia de Product con los atributos de la instancia inicial pero con name=“New name” | El método se ejecuta correctamente, llama al método findById y save de ProductRepository y el producto queda actualizado dentro de la base de datos del CrudRepository (Se verifica comparando el name del producto insertado, con el name del producto encontrado) |
| updateProduct(Integer pId, Product p) | Instancia inicial de Product con Productnumber=“1”, Daystomanufacture=1, Sellstartdate=new Timestamp (System.currentTimeMillis()-100), Sellenddate=new Timestamp (System.currentTimeMillis()-100), Productid=1, name=“Name Product”, color=“Blue”, size=“Small”  Parámetros:  1, Instancia de Product con los atributos de la instancia inicial pero con color=“Red” | El método se ejecuta correctamente, llama al método findById y save de ProductRepository y el producto queda actualizado dentro de la base de datos del CrudRepository (Se verifica comparando el color del producto insertado, con el color del producto encontrado) |
| updateProduct(Integer pId, Product p) | Instancia inicial de Product con Productnumber=“1”, Daystomanufacture=1, Sellstartdate=new Timestamp (System.currentTimeMillis()-100), Sellenddate=new Timestamp (System.currentTimeMillis()-100), Productid=1, name=“Name Product”, color=“Blue”, size=“Small”  Parámetros:  1, Instancia de Product con los atributos de la instancia inicial pero con size=“Medium” | El método se ejecuta correctamente, llama al método findById y save de ProductRepository y el producto queda actualizado dentro de la base de datos del CrudRepository (Se verifica comparando el size del producto insertado, con el size del producto encontrado) |
| updateProduct(Integer pId, Product p) | Instancia inicial de Product con Productnumber=“1”, Daystomanufacture=1, Sellstartdate=new Timestamp (System.currentTimeMillis()-100), Sellenddate=new Timestamp (System.currentTimeMillis()-100), Productid=1, name=“Name Product”, color=“Blue”, size=“Small”  Parámetros:  1, Instancia de Product con los atributos de la instancia inicial pero con Productnumber=“2” | El método se ejecuta correctamente, llama al método findById y save de ProductRepository y el producto queda actualizado dentro de la base de datos del CrudRepository (Se verifica comparando el Productnumber del producto insertado, con el Productnumber del producto encontrado) |
| updateProduct(Integer pId, Product p) | Instancia inicial de Product con Productnumber=“1”, Daystomanufacture=1, Sellstartdate=new Timestamp (System.currentTimeMillis()-100), Sellenddate=new Timestamp (System.currentTimeMillis()-100), Productid=1, name=“Name Product”, color=“Blue”, size=“Small”  Parámetros:  1, Instancia de Product con los atributos de la instancia inicial pero con Sellenddate=new Timestamp(System.currentTimeMillis()+1000) y Sellstartdate(new Timestamp(System.currentTimeMillis()-1000) | El método se ejecuta correctamente, llama al método findById y save de ProductRepository y el producto queda actualizado dentro de la base de datos del CrudRepository (Se verifica comparando el Sellstartdate del producto insertado, con el Sellstartdate del producto encontrado) |

|  |  |  |
| --- | --- | --- |
| **Pruebas sobre la implementación de com.taller1.service.implementation.SpecialOfferServiceImpl**  *PRUEBAS UNITARIAS* | | |
| **Método a probar** | **Paramétro(s)** | **Resultado esperado (Prueba unitaria)** |
| saveSpecialOffer(Specialoffer s) | null | El método lanza NullPointerException con mensaje “SpecialOffer cannot be null”. No hay llamado a ningún método de soRepository (mock). |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con category=null y el resto de los atributos respetando las condiciones dadas en el enunciado. | El método lanza Exception con mensaje “SpecialOffer’s category cannot be null”. No hay llamado a ningún método de soRepository (mock). |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con discountpct=null y el resto de los atributos respetando las condiciones dadas en el enunciado. | El método lanza Exception con mensaje “SpecialOffer’s discount must be greater than 0”. No hay llamado a ningún método de soRepository (mock). |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con discountpct=BigDecimal.valueOf(-1) y el resto de los atributos respetando las condiciones dadas en el enunciado. | El método lanza Exception con mensaje “SpecialOffer’s discount must be greater than 0”. No hay llamado a ningún método de soRepository (mock). |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con modifieddate=null y el resto de los atributos respetando las condiciones dadas en el enunciado. | El método lanza Exception con mensaje “SpecialOffer’s modified date must be now and cannot be null”. No hay llamado a ningún método de soRepository (mock). |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con modifieddate=new Timestamp(System.currentTimeMillis()+100000) y el resto de los atributos respetando las condiciones dadas en el enunciado. | El método lanza Exception con mensaje “SpecialOffer’s modified date must be now and cannot be null”. No hay llamado a ningún método de soRepo (mock). |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con los atributos respetando las condiciones dadas en el enunciado. | El método llama correctamente al método save de soRepo (mock) para guardar la Specialoffer y no genera ninguna excepción. |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  0, Instancia inicial de Specialoffer | El método lanza NoSuchElementException con el mensaje “no value present”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, null | El método lanza NullPointerException con el mensaje “Cannot invoke "com.taller1.model.sales.Specialoffer.getCategory()" because "s" is null”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial pero con category=null | El método lanza Exception con el mensaje “Category cannot be null or blank”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial pero con discountpct=null | El método lanza Exception con el mensaje “Discount cannot be null nor less than 0”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial pero con discountpct=BigDecimal.valueOf(-1) | El método lanza Exception con el mensaje “Discount cannot be null nor less than 0”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial pero con discountpct=BigDecimal.valueOf(0) | El método lanza Exception con el mensaje “Discount cannot be null nor less than 0”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial pero con modifieddate=null | El método lanza Exception con el mensaje “Modified date must be now and not null”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial pero con modifieddate=new Timestamp(System.currentTimeMillis()-100000) | El método lanza Exception con el mensaje “Modified date must be now and not null”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial pero con modifieddate=new Timestamp(System.currentTimeMillis()+100000) | El método lanza Exception con el mensaje “Modified date must be now and not null”. Hay una interacción con soRepo (mock), dado que se hace el llamado al método findById(Integer id). |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial y maxqty=15 y minqty=5 | El método llama correctamente al método save de soRepo (mock) para guardar el producto actualizado y no genera ninguna excepción. |
| *PRUEBAS DE INTEGRACIÓN* | | |
| saveSpecialOffer(Specialoffer s) | null | El método lanza NullPointerException con mensaje “SpecialOffer cannot be null” |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con category=null y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza NullPointerException con mensaje “SpecialOffer’s category cannot be null” |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con discountpct=BigDecimal.valueOf(-1) y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza NullPointerException con mensaje “SpecialOffer’s discount must be greater than 0” |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con modifieddate=new Timestamp(System.currentTimeMillis()-100000) y el resto de los atributos respetando las condiciones dadas en el enunciado | El método lanza NullPointerException con mensaje “SpecialOffer’s modified date must be now and cannot be null” |
| saveSpecialOffer(Specialoffer s) | Instancia de Specialoffer con los atributos respetando las condiciones dadas en el enunciado | El método se ejecuta correctamente, llama al método save de SpecialofferRepository y queda registrado dentro de la base de datos del CrudRepository (Se verifica con el método findById del repositorio y comparando el Specialofferid y description del Specialoffer encontrado con el insertado) |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parámetros:  0, Instancia de Specialoffer inicial | El método lanza NoSuchElementException con el mensaje “no value present”. |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parámetros:  1, null | El método lanza NullPointerException con el mensaje “Cannot invoke "com.taller1.model.sales.Specialoffer.getCategory()" because "s" is null”. |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parámetros:  1, Instancia inicial de Specialoffer con category=null | El método lanza Exception con el mensaje “Category cannot be null or blank”. |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parámetros:  1, Instancia inicial de Specialoffer con discountpct=BigDecimal.valueOf(0) | El método lanza Exception con el mensaje “Discount cannot be null nor less than 0”. |
| updateSpecialOffer(Integer sId, Specialoffer s) | Instancia inicial de Specialoffer con category=“Special Category”, discountpct=BigDecimal.valueOf(5), modifieddate=new Timestamp (System.currentTimeMillis()), specialofferid=1, description=“Description”, startdate= new Timestamp(System.currentTimeMillis()-100) enddate= new Timestamp(System.currentTimeMillis()+1000)  Parametros:  1, Instancia de Specialoffer con los atributos de la instancia inicial y maxqty=15 y minqty=5 | El método se ejecuta correctamente, llama al método findById y save de SpecialofferRepository y queda registrado dentro de la base de datos del CrudRepository (Se verifica con el método findById del repositorio y comparando la description del Specialoffer encontrado con el insertado) |

|  |  |  |
| --- | --- | --- |
| **Pruebas sobre la implementación de com.taller1.service.implementation.SpecialOfferProductServiceImpl**  *PRUEBAS UNITARIAS* | | |
| **Método a probar** | **Paramétro(s)** | **Resultado esperado (Prueba unitaria)** |
| saveSpecialOfferProduct(Specialofferproduct sp, Integer pId, Integer soId) | null, 1, 1 | El método lanza NullPointerException con mensaje “SpecialOfferProduct must not be null”. No hay llamado a ningún método de sopRep (mock). |
| saveSpecialOfferProduct(Specialofferproduct sp, Integer pId, Integer soId) | Instancia de Specialofferproduct con los atributos respetando las condiciones dadas en el enunciado, 0, 1 | El método lanza NoSuchElementException con mensaje “no value present”. No hay llamado a ningún método de sopRep (mock), pero si hay un llamado al método findById(Integer id) de pRep (mock). |
| saveSpecialOfferProduct(Specialofferproduct sp, Integer pId, Integer soId) | Instancia de Specialofferproduct con los atributos respetando las condiciones dadas en el enunciado, 1, 0 | El método lanza NoSuchElementException con mensaje “no value present”. No hay llamado a ningún método de sopRep (mock), pero si hay un llamado al método findById(Integer id) de pRep (mock) y soRep (mock). |
| saveSpecialOfferProduct(Specialofferproduct sp, Integer pId, Integer soId) | Instancia de Specialofferproduct con modifieddate=null y el resto de los atributos respetando las condiciones dadas en el enunciado, 1, 1 | El método lanza Exception con mensaje “SpecialOfferProduct modified date must be now and cannot be null”. No hay llamado a ningún método de sopRep (mock). |
| saveSpecialOfferProduct(Specialofferproduct sp, Integer pId, Integer soId) | Instancia de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()-100000)y el resto de los atributos respetando las condiciones dadas en el enunciado, 1, 1 | El método lanza Exception con mensaje “SpecialOfferProduct modified date must be now and cannot be null”. No hay llamado a ningún método de sopRep (mock). |
| saveSpecialOfferProduct(Specialofferproduct sp, Integer pId, Integer soId) | Instancia de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()+100000)y el resto de los atributos respetando las condiciones dadas en el enunciado, 1, 1 | El método lanza Exception con mensaje “SpecialOfferProduct modified date must be now and cannot be null”. No hay llamado a ningún método de sopRep (mock). |
| saveSpecialOfferProduct(Specialofferproduct sp, Integer pId, Integer soId) | Instancia de Specialofferproduct con los atributos respetando las condiciones dadas en el enunciado, 1, 1 | El método llama correctamente al método save de sopRepo (mock) para guardar la Specialofferproduct y no genera ninguna excepción. |
| editSpecialOfferProduct(SpecialofferproductPK spId, Specialofferproduct sp) | Instancia inicial de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()), id=Instancia de SpecialofferproductPK  Parámetros:  Intancia de SpecialofferproductPK, null | El método lanza NullPointerException con mensaje “Cannot invoke "com.taller1.model.sales.Specialofferproduct.getModifieddate()" because "sp" is null”. |
| editSpecialOfferProduct(SpecialofferproductPK spId, Specialofferproduct sp) | Instancia inicial de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()), id=Instancia de SpecialofferproductPK  Parámetros:  Intancia de SpecialofferproductPK, Instancia inicial de Specialofferproduct con modifieddate=null | El método lanza Exception con mensaje “Modified date must be now and not null”. |
| editSpecialOfferProduct(SpecialofferproductPK spId, Specialofferproduct sp) | Instancia inicial de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()), id=Instancia de SpecialofferproductPK  Parámetros:  Intancia de SpecialofferproductPK, Instancia inicial de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()-100000) | El método lanza Exception con mensaje “Modified date must be now and not null”. |
| editSpecialOfferProduct(SpecialofferproductPK spId, Specialofferproduct sp) | Instancia inicial de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()), id=Instancia de SpecialofferproductPK  Parámetros:  Intancia de SpecialofferproductPK, Instancia inicial de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()+100000) | El método lanza Exception con mensaje “Modified date must be now and not null”. |
| editSpecialOfferProduct(SpecialofferproductPK spId, Specialofferproduct sp) | Instancia inicial de Specialofferproduct con modifieddate=new Timestamp(System.currentTimeMillis()), id=Instancia de SpecialofferproductPK  Parámetros:  Intancia de SpecialofferproductPK, Instancia inicial de Specialofferproduct con rowguid=2 | El método llama correctamente al método save de sopRep (mock) para guardar el specialofferproduct actualizado y no genera ninguna excepción. |
| *PRUEBAS DE INTEGRACIÓN* | | |
|  |  |  |