Sistema de Trámite Documentario

Documento de Microdiseño

Caso de Uso: Registrar Tramites

Índice

[1. Descripción General de funcionalidades 2](#_Toc374543310)

[1.1. Registrar tramite 2](#_Toc374543311)

[2. Detalle de Implementación a nivel de Modelos 3](#_Toc374543312)

[3. Detalle de Implementación a nivel de Controladores 4](#_Toc374543313)

[4. Detalle de Implementación a nivel de Vistas 5](#_Toc374543314)

[5. Detalle de Implementación a nivel de Tablas 6](#_Toc374543315)

## Descripción General de funcionalidades

El presente documento esta designado a la elaboración de las funcionalidades del sistema de trámite documentario.

## Registrar tramite

|  |  |  |  |
| --- | --- | --- | --- |
| **Elaborado Por:** |  | | |
| **Descripción General:** | Registrar tramite*:* | | |
| Esta funcionalidad tiene como objetivo registrar datos de los trámites. | | |
| **Referencias** | Requerimientos: | Especificación del caso de uso Registrar Tramite | |
| STD\_REG\_TRA.docx | |
| Diseño de Interfaz Usuario: | Especificación del caso de uso Registrar Tramite | |
| STD\_REG\_TRA.docx | |
| Diseño de Arquitectura: | Diagrama de Arquitectura del software | |
| STD\_DAS.docx | |
| Otros: | Matriz de trazabilidad de casos de uso vs. Requerimientos funcionales STD\_MATR\_TRAZ.xlsx | |
| Documento de análisis | |
| STD\_ANA\_DOC.docx | |
| **Clases Requeridas:** | **Nombre de la Clase** | **Tipo – Stereotype** | **Acción** |
| tramites.java | Model | Escritura |
| ctramites.java | Controller | Escritura |
| registratramites.java | View | Escritura |
| **Tablas:** | **Nombre de la tabla** | | **Tipo de Objeto** |
| tb\_std\_tramites | | Tabla |

## Detalle de Implementación a nivel de Modelos

|  |  |
| --- | --- |
| **Modelo** | **Descripción** |
| tramites.java | Clase persistente utiliza para hacer persistente a la entidad trámites. |
| requisitos.java | Clase persistente utiliza para hacer persistente a la entidad requisitos. |
| areas.java | Clase persistente utiliza para hacer persistente a la entidad áreas. |
| expedientes.java | Clase persistente utiliza para hacer persistente a la entidad expedientes. |
| alumnos.java | Clase persistente utiliza para hacer persistente a la entidad alumnos. |
| usuarios.java | Clase persistente utiliza para hacer persistente a la entidad usuarios. |
| perfil.java | Clase persistente utiliza para hacer persistente a la entidad perfil. |

**tramites.java**

* **Ruta:**

app/models/.java

* **Descripción:**

Clase de modelo utilizada para hacer persistencia a la entidad **trámites,** extendiendo de la claseModel,debe ser declarada como una entidad, y poseer relación con la tabla *“tb\_std\_tramites”*:

1. @Entity
2. @Table(name = "tramites")
3. public class Tramites extends Model

* **Atributos:**

Dado que se trata de una clase entidad de playFramework, los atributos deben poseer una referencia a una columna de la tabla “tb\_std\_tramites”:

Para los atributos que tienen primary\_key en la base de datos:

1. @Requiered
2. @Colum(name = "nombre\_atributo", nullable = false)
3. public tipo atributo;

Para los atributos que no posean primary\_key en la base de datos:

1. @Colum(name = "nombre\_atributo", nullable = true)
2. public tipo atributo;

* **Métodos:**

Debe poseer el método que sobrescriba el método toString() para mostrar el código del almacén y su nombre referente al mismo:

1. @Override
2. public String toString()

## Detalle de Implementación a nivel de Controladores

|  |  |
| --- | --- |
| **Controlador** | **Descripción** |
| ctramites.java | Clase controlador la cual permite hacer el proceso gestionar los datos de trámites. |
| crequisitos.java | Clase controlador la cual permite hacer el proceso gestionar los datos de requisitos. |
| careas.java | Clase controlador la cual permite hacer el proceso gestionar los datos de áreas. |
| cexpedientes.java | Clase controlador la cual permite hacer el proceso gestionar los datos de expedientes. |
| calumnos.java | Clase controlador la cual permite hacer el proceso gestionar los datos de alumnos. |
| cusuarios.java | Clase controlador la cual permite hacer el proceso gestionar los datos de trámites. |
| cperfil.java | Clase controlador la cual permite hacer el proceso gestionar los datos de los perfiles. |

**ctramites.java**

* **Ruta:**

app/controllers/ctramites.java

* **Descripción:**

Clase controlador la cual permite hacer el proceso del negocio, recuperar la información de un tramite.

* **Métodos:**

Debe de habilitar la vista del formulario de gestión y registro de trámites.

public static void tramites()

## Detalle de Implementación a nivel de Vistas

|  |  |
| --- | --- |
| **Vista** | **Descripción** |
| registrotramites.html | Objeto de vista que registra datos en la entidad tramite. |
| gestiontramites.html | Objeto de vista que gestiona, muestra, actualiza, elimina, datos en la entidad tramite. |
| registrorequisitos.html | Objeto de vista que registra datos en la entidad tramite. |
| gestionrequisitos.html | Objeto de vista que gestiona, muestra, actualiza, elimina, datos en la entidad tramite. |
| registroareas.html | Objeto de vista que registra datos en la entidad tramite. |
| gestionareas.html | Objeto de vista que gestiona, muestra, actualiza, elimina, datos en la entidad tramite. |
| registroexpedientes.html | Objeto de vista que registra datos en la entidad tramite. |
| gestionexpedientes.html | Objeto de vista que gestiona, muestra, actualiza, elimina, datos en la entidad tramite. |
| registroalumnos.html | Objeto de vista que registra datos en la entidad tramite. |
| gestionalumnos.html | Objeto de vista que gestiona, muestra, actualiza, elimina, datos en la entidad tramite. |
| usuarios.html | Objeto de vista que registra datos en la entidad tramite. |
| registrousuarios.html | Objeto de vista que gestiona, muestra, actualiza, elimina, datos en la entidad tramite. |

**Registrotramites.list.html**

* **Ruta:**

app/views/productos/list.html

* **Descripción:**

Objeto de vista, extiende del layout ‘CRUD/layout.html’, muestra la búsqueda de una entidad *Producto*, debe de contar de un campo de entrada numérica y un botón envié el valor recibido como parámetro y haga referencia al método del controlador *CProducto*:

1. href = "@{CProducto.producto()}"

**mostrar.html**

* **Ruta**:

app/views/cproducto/mostrar.html

* **Descripción:**

Objeto de vista, extiende del layout ‘main.html’, donde se muestra la información encontrada por el controlador CProducto, la cual deberá tener la siguiente especificación:

1. Código
2. Tipo
3. Marca
4. Modelo
5. Precio (S/.)
6. Punto de reposición
7. Código del almacén donde se encuentra

Para cada atributo del producto listado con anterioridad se declarará:

1. <input type =”text” name=“producto.atributo” id =”atributo” value=”${product.atributo}”>

Para el atributo, ‘código del almacén’ del almacén de la lista de almacenes que concuerde con uno el del almacén del producto se especificará:

<option value="${almacen.codAlmacen}" > ${almacen.toString()} </option>

## Detalle de Implementación a nivel de Tablas

-- ------------------------------------------------------------

-- Tabla que almacena los datos de los estados de los trámites.

-- ------------------------------------------------------------

CREATE TABLE tb\_std\_estado\_tramites (

idtb\_std\_estado\_tramite INTEGER NOT NULL ,

nombre\_estadotramite VARCHAR ,

PRIMARY KEY(idtb\_std\_estado\_tramite));

COMMENT ON TABLE tb\_std\_estado\_tramites IS 'Tabla que almacena los datos de los estados de los tramites.';