Definición del Proyecto

Objetivos principales

- Hacer un programa de JAVA JSP mediante un proyecto de Eclipse siguiendo las versiones, nomenclatura y lineamientos de este espacio de aprendizaje, que permita mediante una única View, comunicarse con múltiples componentes Controller y Model, para la administración de un "sistema de registros de documentos".
- Promover la investigación de componentes usando el Paradigma Orientado a Objetos, mediante GNU/Linux, Eclipse y el lenguaje de programación Java y Javascript.
- Integrar la operación de objetos de transferencia de datos, objetos de acceso a datos, objetos de DOM, clases y objetos de Java y otros conceptos de objetos en la web (enum y abstract).
- Evaluar la capacidad individual de los estudiantes para el desarrollo de una solución completa de software usando los conocimientos, temáticas, técnicas y nomenclatura de este espacio de aprendizaje.

Restricciones generales

- El proyecto debe usar código exclusivo desarrollado por el estudiante para resolver la problemática planteada en esta definición de proyecto. Se admite y se recomienda el uso del código generado por el docente y del código del autor del libro de la clase para el desarrollo más rápido de la solución, siempre y cuando se entregue la documentación respectiva de @author evitando así el uso de plagio. Cualquier otro código público, de internet u otra fuente (e.g. de otros libros o personas), queda prohibido para su uso en el proyecto. Se permite el uso de las librerías por defecto incluídas en la versión de Eclipse de la clase como son la librería java.util. Entregar exclusivamente código fuente del profesor o de los autores del libro, sin modificaciones funcionales importantes/relevantes por parte del estudiante, no le genera calificación. El estudiante deberá generar resultados tangibles y funcionales para calificar puntaje del proyecto.
- Para todo caso, usando código propio del estudiante, código del autor del libro de la clase, o el código de su docente como autor, no se debe incluir en el sistema código sin utilidad, es decir, se debe incluir únicamente código que tenga una funcionalidad para el sistema. Atención: El código sin utilidad podrá ser considerado un código de relleno ficticio para hacer parecer el proyecto como completo, generando una reducción de puntos en su calificación.
- La calificación de su proyecto será entregada de acuerdo con la calidad funcional, la calidad visual, la adecuada abstracción, encapsulamiento, modularización y de A/D/POO (Análisis, Diseño y Programación Orientada a Objetos), y sobre el cumplimiento de los requerimientos que se listan en este documento.
- Recuerde que las imágenes y el apoyo visual de esta definición de proyecto son una guía y un marco de trabajo para cada proyecto, no son el 100% de la solución que debe crear cada estudiante. Estas imágenes son parte del requerimiento de la solución a desarrollar.
- El código debe ser entregado mediante un archivo de compresión 7z https://www.7-zip.org/ .
- Para el nombre de archivo, documentación, video de demostración y otros, vea la Lectura: Planificación Académica de la clase, sección "Sobre proyectos".
- El sistema debe ser completamente funcional mediante la ejecución del programa .war como un programa aislado y ejecutable en un servidor Web Apache Tomcat de GNU/Linux, así como se solicitó el elemento Tarea #1 de esta asignatura.
- Haga uso de Bootstrap según lo enseñado a través de los contenidos síncronos y asíncronos de la clase, para aplicar el estilo de su aplicación final.

Requerimientos

Una vez ejecutado su sistema mediante un servidor web Apache Tomcat usando la versión de la asignatura, el proyecto deberá demostrar la siguiente funcionalidad.

- **Deberá existir una única View en todo el proyecto, donde se centraliza toda la funcionalidad del sistema, la cual invoca distintas ventanas modales de Bootstrap (** https://getbootstrap.com/docs/5.1/components/modal/**), con distintos Controller, Objetos, Javascripts, DTO y DAO para las diferentes funcionalidades del sistema.**
  - Debe existir un menú superior que permita crear un nuevo registro de documento. Esta acción deberá habilitar una ventana modal para la recepción de los datos de registro: responsable, tipo, entrada/retiro y descripción.
  - Debe existir un menú superior que permita limpiar por completo el modelo de datos. Su programa deberá leer un archivo de modelo de datos en la ruta generada por Path() según el FileManager, sobre un archivo llamado DocumentsModel.csv.
  - Debe existir un menú superior que permita visualizar la información del autor del sistema, como una ventana modal que imprime la información de nombre, correo institucional, asignatura, fecha, docente y número de cuenta. Redacte en esta ventana modal una descripción de la funcionalidad del sistema (2~3 párrafos) y de su análisis principal como el que desarrolló mediante la Documentación del Entrega Avance 1.
  - Para el resto de funcionalidades, se lista a continuación una serie de pantallas con explicaciones asociadas.

Fig 1. Ejemplar de Vista principal que contiene todos los elementos del sistema. En la barra superior se debe definir "un nombre del sistema junto con el nombre del estudiante". El componente central "Listado de registro actuales en el modelo de datos" posee un "scroll vertical", donde se muestran todos los registros leídos desde el modelo de datos.

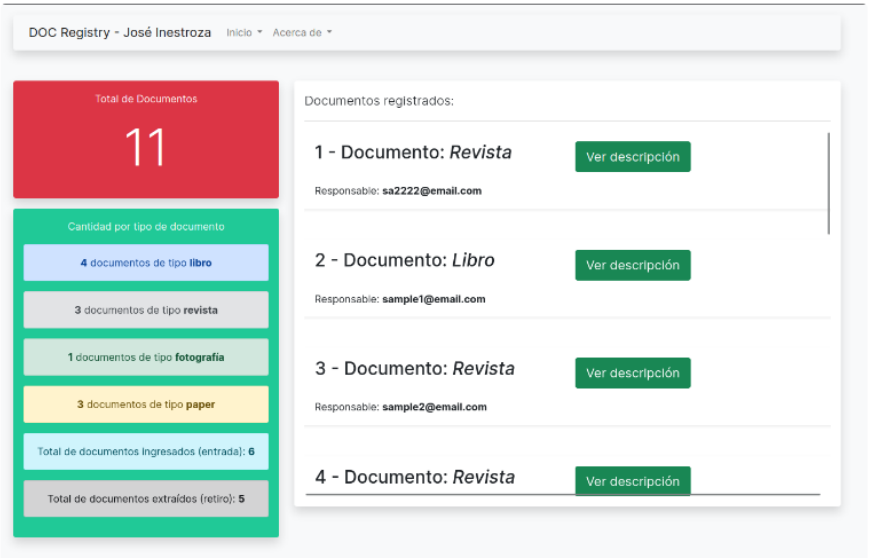


Fig 2. Ejemplar de Menú desplegable "Inicio". El botón "Limpiar el Modelo de Datos" debe limpiar el archivo de Modelo asociado en el sistema según lo descrito anteriormente. Cada vez que se limpia el modelo de datos, se debe crear un nuevo archivo de texto en blanco que posea la fecha de limpieza (e.g. RemovidoEl_2022-08-08-22-10-23.log). El archivo no debe tener ningún contenido pero debe contener la fecha en formato YYYY-MM-DD-HH-MM-SS.

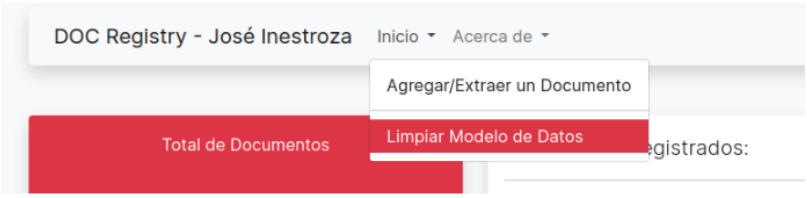


Fig 3. Ventana modal de crear un nuevo registro. Al completar esta información se registran los datos en el modelo de datos y se refleja la información inmediatamente en pantalla. La descripción debe tener un mínimo de 320 palabras, de lo contrario no se puede registrar la información en el modelo de datos.Si el usuario a escrito algún texto en esta ventana modal sin haber presionado el botón cancelar o el botón guardar, y el usuario refresca el navegador, entonces, al entrar en "crear un nuevo registro" el sistema debe recordar (mediante sesiones) lo último escrito por el usuario antes de el refrescado de la pantalla.

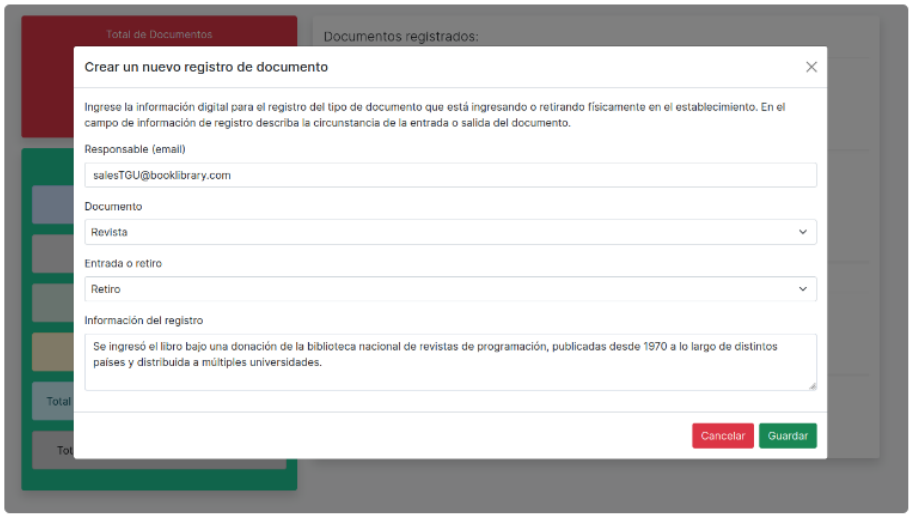


Fig 4. Ejemplar del menú desplegable “Acerca de”.

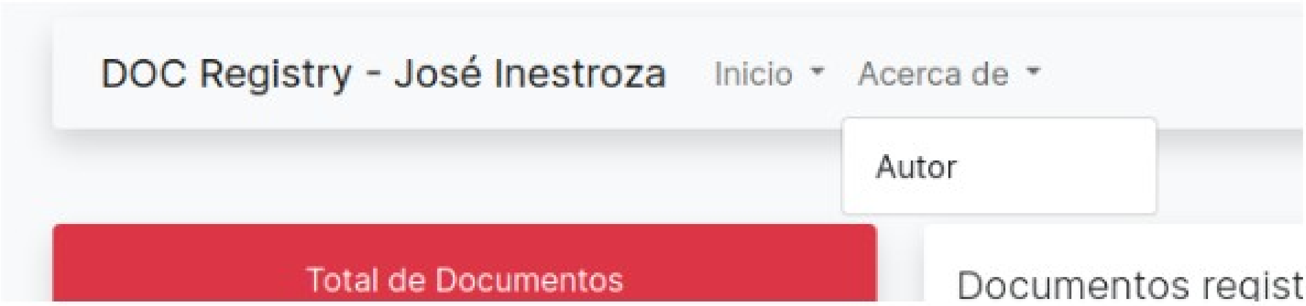


Fig 5. Ventana modal de “Acerca de”.

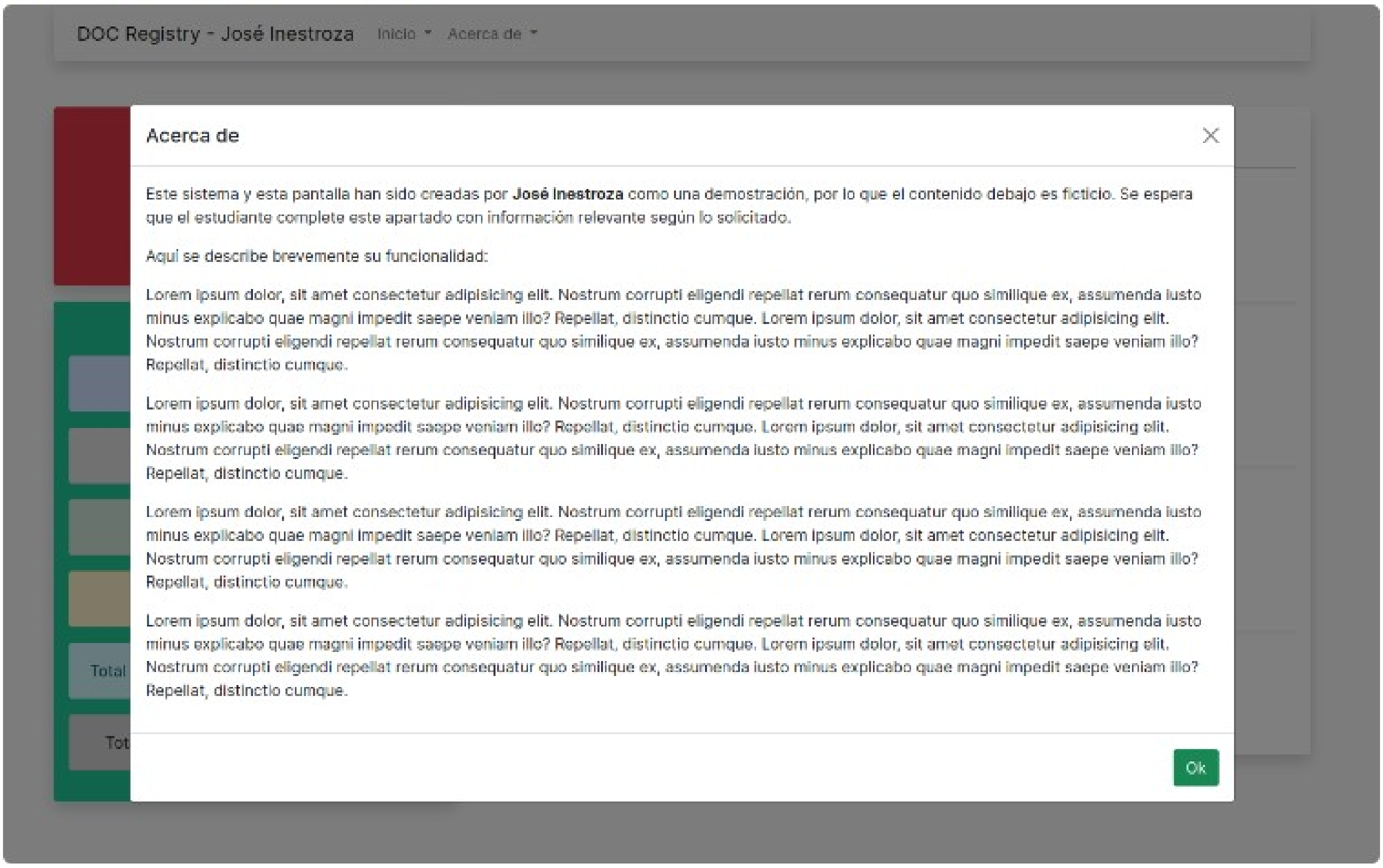


Fig 6. Resultado al presionar el botón "Ver descripción" de un registro.

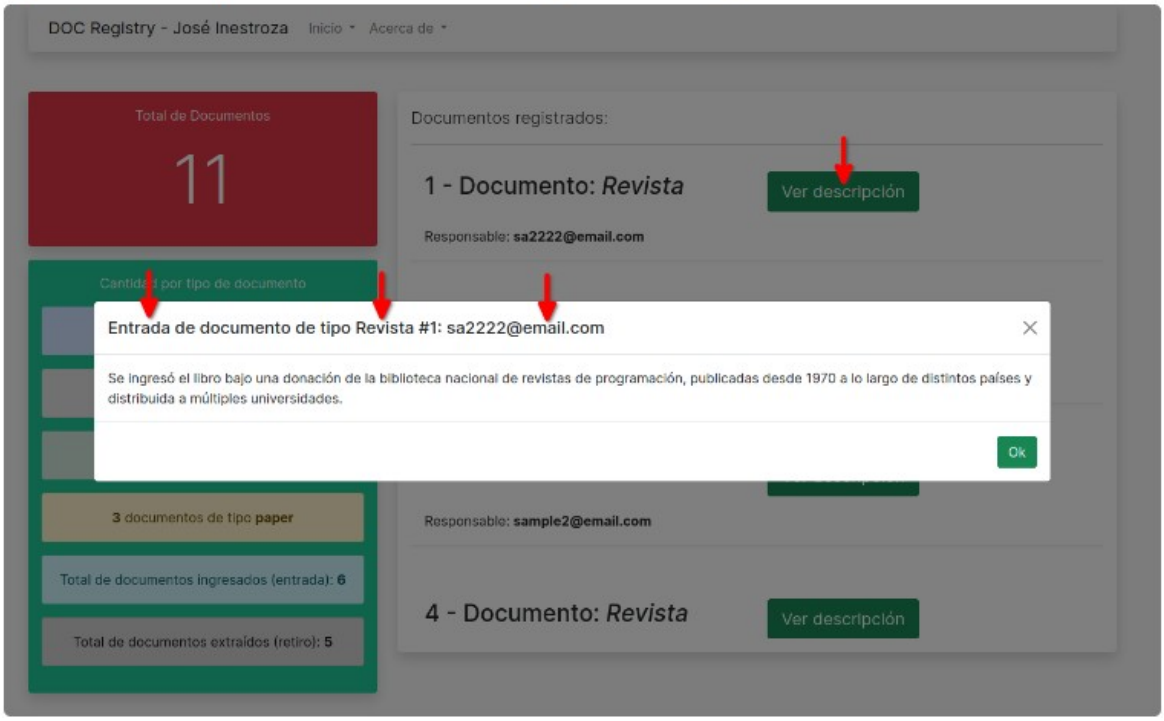
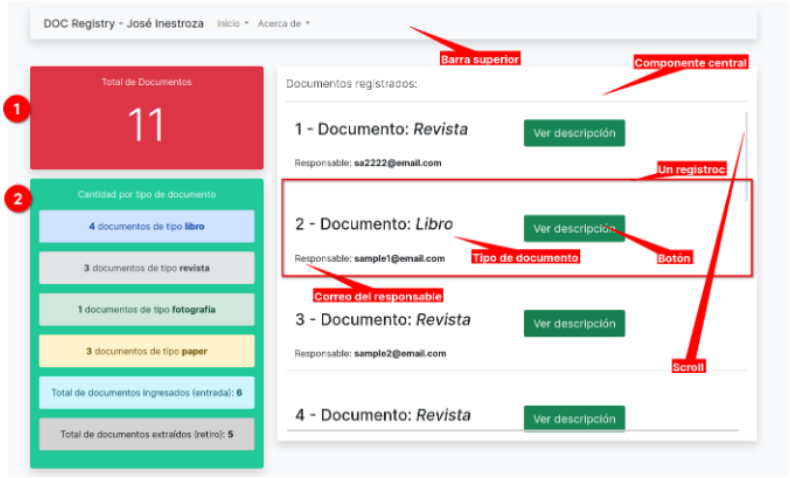


Fig 7. Componentes generales. En el componente central se listan todos los registros de datos leídos desde el modelo de datos. Cada registro posee un número secuencial según el orden de inserción, junto con el tipo de registro, un botón para visualizar la descripción del registro y el correo electrónico del responsable. Este componente central posee un "scroll vertical", donde se muestran todos los registros leídos desde el modelo de datos. Los componentes laterales derechos muestran (1) la cantidad total de registros del modelo, (2) la cantidad total de registros para el tipo de documento y por el estado de entrada/salidas de los documentos. El sistema debe estar leyendo frecuentemente de una forma eficiente, si hay cambios en el modelo de datos, y reflejar la información respectiva en el "componente central" en (1) y (2).



- **Finalmente, durante los días de publicación del proyecto se le pedirán diferentes entregables. Todas las entregas son obligatorias. Estos se listan a continuación:**
  - Entrega Avance 1: Documentación. De acuerdo con la Lectura: Planificación Académica, la documentación de análisis refiere a un archivo MD (MarkDown) que contiene los hallazgos encontrados por el estudiante, al momento de establecer una solución "descrita", junto con los diferentes tipos de diagramas que puede usar el estudiante para establecer su A/DOO. Refiérase a la Lectura: Planificación Académica para saber más.
  - Entrega Avance 2: Avance como un archivo 7-zip. Cualquiera que sea el código de avance (vistas, controladores, objetos, modelos, etc) debe ser entregado en este día. Una entrega de código, como la que representa este avance, no puede contener errores de ejecución (sólo debe tener código funcional), entregando únicamente un Proyecto de Eclipse sin un exportado .war por separado. Su avance debe ser considerable, al menos conteniendo todos los aspectos visuales ya desarrollados, incluyendo la lógica de todas las ventanas modales y estilo CSS.
  - Entrega Avance 3: Primer Release como un archivo 7-zip. Este código deberá ser la primera propuesta completa de funcionalidad del sistema como una propuesta casi completa de solución que incluye peticiones AJAX e inclusive la creación de algunos (aunque no necesariamente todos) los Modelos de datos. Esta entrega de código no puede tener errores de ejecución (solo debe contener código funcional), incluyendo un Proyecto de Eclipse y un exportado .war (que incluye el código fuente dentro del war, idéntico a lo solicitado en la Tarea #1).
  - Entrega Final: Debe entregar un video mp4 (máximo 20MB), junto con un 7z final (con toda la funcionalidad de este requerimiento) que incluye todo lo mencionado en la Lectura: Planificación Académica (máximo 20MB). Siguiendo las indicaciones de la Lectura: Planificación Académica, haga un video de evidencia explicando rápidamente la funcionalidad y la codificación. Siga las reglas de la planificación. En este último día de entrega el estudiante debe entregar su código final con cualquier corrección encontrada en los entregables anteriores, incluyendo un Proyecto de Eclipse y un exportado .war ejecutable (que incluye el código fuente dentro del war, idéntico a lo solicitado en la Tarea #1).