Nombre: Jesús Aparicio Ortiz Grupo: G2-06

Informe individual de actividades del proyecto

Datos generales

URL del Repositorio de GitHub: https://github.com/gii-is-DP1/dp1-2020-g2-06
Nombre de usuario en GitHub: JesusAparicioOrtiz
Rama del usuario en Github: jesapaort

Participación en el proyecto

Historias de usuario en las que he participado

He implementado completas las historias de usuario HU-2,HU-11,HU-17 y HU-18, además, he desarrollado junto a mi compañero Alejandro Barranco las historias de usuario HU-7,HU-8,HU-17 y H-18 y junto a mi compañero Víctor Javier Granero las historias de usuario H-14 y HU-15. He desarrollado con Alejandro Barranco y Juan Ramón Ostos la HU-3.

Funcionalidad implementada

-He implementado los controladores: NormaWebController, ProblemaController, NoticiaController, AdministradorController, CustomErrorController, PerfilController, PreguntaTutorController.De los mencionados anteriormente he implementado yo solo el NormaWebController, AdministradorController, CustomErrorController y el PerfilController. Además, he añadido en todos los controladores la validación de sesión y he añadido la gestión de archivos tanto zips como imágenes en todos los controladores que lo han requerido.
-He añadido varios métodos a los servicios: administradorService, alumnoService, creadorService, fileService, normaWebService, noticiaService, preguntaTutorService, problemaService y tutorService.De los mencionados anteriormente he implementado yo solo el administradorService, fileService y normaWebService. Además, he añadido la encriptación de las contraseñas al guardar los alumnos, creadores y tutores.
-He creado las entidades: Auth, NormaWeb y Problema. De los mencionados anteriormente he implementado yo solo todos ellos. Además de añadir los mensajes de errores al validar en la gran mayoría de entidades.
-He creado las vistas: administrador/panelAdmin.jsp , alumno/alumnoDetails.jsp , normasWeb/createOrUpdateNormaWebForm.jsp , normasWeb/normasWebList.jsp , problemas/createOrUpdateProblemaForm.jsp , problemas/problemaDetails.jsp , problemas/problemasList.jsp y exception.jsp.De las mencionadas anteriormente he implementado yo solo todas ellas a excepción de alumno/alumnoDetails.jsp y problemas/problemaDetails.jsp. También, he implementado la paginación usando AJAX en el cliente para las siguientes vistas: alumnos/AlumnosDetails , noticias/noticiasList.Además, he implementado parte de todos los formularios que implican subir un archivo (multipart form).
-He añadido a la clase Utils las funciones diferenciador() yo solo y la función getActualSeason() en un principio la realicé yo y Alejandro Barranco le realizó ciertos cambios.

-He realizado la configuración del SecurityConfiguration para otorgar permisos a la hora de realizar las distintas peticiones. Además, he implementado todo lo relacionado con la encriptación de las contraseñas con la librería Bcrypt.
-He realizado como he comentado anteriormente la gestión de la subida de archivos a la página, de modo que todas las imágenes se guardan en subcarpetas dentro de la carpeta "static", y las imágenes quedan referenciadas con una ruta en cada entidad que tenga una imagen (p. ej.: alumno,tutor,creador,etc).

Esto hace un total de 19 clases implementadas por mí y 9 interfaces definidos.

Pruebas implementadas

Pruebas unitarias

-He creado tests unitarios para 7 servicios (AlumnoService, AuthService, ArticuloService,FileService,NormaWebService,PreguntaTutorService y ProblemaService) y 6 controladores (AlumnoController,ArticuloController,FileController,NormaWebController,PreguntaTutorController,ProblemaController). Eso hace un total de 13 clases de test unitarios con un total de 42 métodos anotados con @Test.

Pruebas de Controlador

-He creado 1 caso de prueba positivo y 2 negativos de controlador para la HU-16, 1 caso de prueba positivo y 1 negativo para la HU-26, 1 caso de prueba positivo y 1 negativo para la HU-27, 1 negativo para la HU-26, 3 casos de prueba positivo y 1 negativo para la HU-40, 1 caso de prueba negativo para la HU-15.

Ejemplos de funcionalidades implementadas

Entidades (máximo de dos ejemplos)

```
package org.springframework.samples.petclinic.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.Table;
import javax.validation.constraints.NotEmpty;

import lombok.EqualsAndHashCode;
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
@EqualsAndHashCode(callSuper=true)
@Entity
@Table(name = "normas_web")
public class NormaWeb extends NamedEntity {

	@ManyToOne
```

```
	@JoinColumn(name="autor_email")
	private Tutor autor;

	@NotEmpty(message= "El campo 'Descripción' no puede estar vacío")
	@Column(length = 5600)
	private String descripcion;
}
```

-Nombre: normaWeb
-ruta: src/main/java/org/springframework/samples/petclinic/model/NormaWeb.java

-La entidad normaWeb hereda de NamedEntity, ya que esta contiene el atributo “name” y a su vez hereda de BasedEntity, ya que contiene el identificador que actúa como clave primaria de la entidad normaWeb. Además, normaWeb tiene los atributos autor y descripción. Autor es el tutor que crea la normaWeb y este atributo tiene una relación ManyToOne con Tutor, ya que un tutor puede realizar muchas normas web pero una normaWeb solo puede ser realizada por un Tutor. Por otro lado, el atributo descripción es una cadena de texto que tiene una restricción simple (sintáctica) para que este atributo no pueda estar vacío, en caso de estarlo se mostrará el mensaje de error “El campo 'Descripción' no puede estar vacío”, además la longitud de la cadena no puede superar los 5600 caracteres. La clase tiene la etiqueta @Entity para indicar que es una entidad y las etiquetas @Getter,@Setter y @ EqualsAndHashCode(callSuper=true) de Loombok para generar los getters, setters y el equals y hashcode de la clase de la que hereda.

Servicio

* A pesar de que he realizado otros servicios yo solo, voy a explicar FileService, ya que es un servicio un tanto diferente a todos los demás y creo que es interesante explicarlo. Este servicio lo he realizado yo entero a excepción de una función, la cual la comento a continuación.

```
package org.springframework.samples.petclinic.service;

import java.awt.Color;
import java.awt.Graphics2D;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.net.MalformedURLException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;

import javax.imageio.ImageIO;
import javax.validation.ConstraintViolationException;

import org.springframework.core.io.Resource;
import org.springframework.core.io.UrlResource;
import org.springframework.samples.petclinic.repository.FileRepository;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import org.springframework.util.FileSystemUtils;
import org.springframework.web.multipart.MultipartFile;

@Service
public class FileService implements FileRepository{
```

```
        @Override
        public void saveFile(MultipartFile file,Path path,String diferenciador) {
          try {
            Files.copy(file.getInputStream(), path.resolve(diferenciador));
          } catch (Exception e) {
            throw new RuntimeException("Could not store the file. Error: " +
e.getMessage());
          }
        }


        @Override
        public Resource load(String filename,Path path) {
          try {
            Path file = path.resolve(filename);
            Resource resource = new UrlResource(file.toUri());

            if (resource.exists() || resource.isReadable()) {
              return resource;
            } else {
              throw new RuntimeException("Could not read the file!");
            }
          } catch (MalformedURLException e) {
            throw new RuntimeException("Error: " + e.getMessage());
          }
        }

        @Override
        public void delete(Path path) throws IOException {
          FileSystemUtils.deleteRecursively(path);
        }

        public void imageCrop(String path,FileService fileService) throws IOEx-
ception {

                    File imageFile = new File("src/main/resources/static/" +
path);
                    BufferedImage init = ImageIO.read(imageFile);
                    BufferedImage bi = new BufferedImage(init.getWidth(),
init.getHeight(),BufferedImage.TYPE_INT_RGB);
                    bi.getGraphics().drawImage(init, 0, 0, null);


                    int h = bi.getHeight();
                    int w = bi.getWidth();

                      if(h>w) {
                            int dif = h-w;
                            bi = bi.getSubimage(0, dif/2, w, w);
                      }

                      else if(w>h) {
                            int dif = w-h;
                            bi = bi.getSubimage(dif/2, 0, h, h);
                      }
```

```
                    fileService.delete(Paths.get("src/main/resources/static/" +
path));
                    File pathFile = new File("src/main/resources/static/" +
path);
                    ImageIO.write(bi,"jpg", pathFile);

            }


}
```

-Nombre: FileService
-Ruta: src/main/java/org/springframework/samples/petclinic/service/FileService.java

-El servicio FileService sirve para gestionar los archivos que se suben a la página, ya sean los .zip que se tienen que subir para enviar al juez que valida los problemas o para subir imágenes tanto para los perfiles de los usuarios como para los artículos y las noticias. De este modo hay tres funciones, uno para guardar los archivos(saveFile), otro para cargarlos(load) y otro para eliminarlos(delete). Además hay un método más para que a la hora de subir una foto esta se recorte(imageCrop), esta última función la ha realizado mi compañero Alejandro Barranco.
-La función saveFile obtiene un archivo de tipo MultipartFile, una ruta y un diferenciador. A partir de estos datos se copia el archivo pasado como parámetro(file) en la ruta pasada como parámetro(path) con el nombre de archivo pasado como parámetro(diferenciador).
-La función load no se utiliza en la aplicación actual, sin embargo, se creó para que en caso de que en un futuro se quiera obtener un archivo de una ruta se pudiera obtener sin problemas. Dicho esto, la función dado el nombre del archivo y la ruta donde se encuentra dicho archivo, obtiene la ruta del fichero haciendo un path.resolve y a continuación obtiene un objeto Resource que es el archivo en sí.
-La función delete dado una ruta de un fichero, lo elimina.
-La función imageCrop dada una imagen la recorta a lo alto o a lo ancho en función de sus dimensiones.

Controlador

```
package org.springframework.samples.petclinic.web;

import java.util.List;
import java.util.Optional;
import java.util.stream.Collectors;

import javax.servlet.http.HttpServletRequest;
import javax.validation.Valid;

import org.springframework.beans.BeanUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.samples.petclinic.model.NormaWeb;
import org.springframework.samples.petclinic.service.NormaWebService;
import org.springframework.samples.petclinic.service.TutorService;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
```

```
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;

import lombok.extern.slf4j.Slf4j;

@Slf4j
@Controller
@RequestMapping("/normasWeb")
public class NormaWebController {

	private static final String VIEWS_NORMAWEB_CREATE_OR_UPDATE_FORM = "nor-
masWeb/createOrUpdateNormaWebForm";

	@Autowired
	private NormaWebService normaWebService;

	@Autowired
	private TutorService tutorService;

	@GetMapping()
	public String listNormasWeb(ModelMap modelMap) {
		String vista = "normasWeb/normasWebList";
		modelMap.addAttribute("normas_web",normaWebService.findAll());
		return vista;
	}

	@GetMapping(value = "/new")
	public String initCreationForm(ModelMap model) {
		NormaWeb normas_web = new NormaWeb();
		model.addAttribute("normaWeb", normas_web);
		return VIEWS_NORMAWEB_CREATE_OR_UPDATE_FORM;
	}

	@PostMapping(value = "/new")
	public String processCreationForm(@Valid NormaWeb normaWeb,BindingResult
result,ModelMap model) {
		if (result.hasErrors()) {
			List<String> errores = result.getAllErrors().stream().map(x-
>x.getDefaultMessage()).collect(Collectors.toList());
			model.addAttribute("message", errores);
			return VIEWS_NORMAWEB_CREATE_OR_UPDATE_FORM;
		}
		else {
			normaWeb.setAutor(tutorService.findByEmail(SecurityContextHol-
der.getContext().getAuthentication().getName()).get());
			normaWebService.saveNormaWeb(normaWeb);
			return "redirect:/normasWeb/";
		}
	}

	@GetMapping("/{id}/edit")
	public String editNormaWeb(@PathVariable("id") int id, ModelMap model,
HttpServletRequest request) {
		Optional<NormaWeb> normaWeb = normaWebService.findById(id);
```

```
		if(!normaWebService.findById(id).get().getAutor().ge-
tEmail().equals(SecurityContextHolder.getContext().getAuthentication().getName()))
{
			model.addAttribute("message","Solo puedes editar tus normas");
			Log.warn("Un usuario esta intentando editar una norma sin te-
ner los permisos necesarios, con sesion "+request.getSession());
			return listNormasWeb(model);
		}
		if(normaWeb.isPresent()) {
			model.addAttribute("normaWeb", normaWeb.get());
			return VIEWS_NORMAWEB_CREATE_OR_UPDATE_FORM;
		}
		else {
			model.addAttribute("message", "No podemos encontrar la norma
Web que intenta borrar");
			return listNormasWeb(model);

		}
	}

	@PostMapping("/{id}/edit")
	public String editNormasWeb(@PathVariable("id") int id, @Valid NormaWeb mo-
difiedNormaWeb, BindingResult binding, ModelMap model, HttpServletRequest request)
{
		Optional<NormaWeb> normaWeb = normaWebService.findById(id);
		if(!normaWebService.findById(id).get().getAutor().ge-
tEmail().equals(SecurityContextHolder.getContext().getAuthentication().getName()))
{
			model.addAttribute("message","Solo puedes editar tus normas");
			Log.warn("Un usuario esta intentando editar una norma sin te-
ner los permisos necesarios, con sesion "+request.getSession());
			return listNormasWeb(model);
		}
		if(binding.hasErrors()) {
			List<String> errores = binding.getAllErrors().stream().map(x-
>x.getDefaultMessage()).collect(Collectors.toList());
			model.addAttribute("message", errores);
			return VIEWS_NORMAWEB_CREATE_OR_UPDATE_FORM;
		}
		else {
			BeanUtils.copyProperties(modifiedNormaWeb, normaWeb.get(),
"id");
			normaWebService.saveNormaWeb(normaWeb.get());
			model.addAttribute("message","Norma Web actualizada con
éxito");
			return listNormasWeb(model);
		}

	}

	@GetMapping("/{id}/delete")
	public String deleteNormasWeb(@PathVariable("id") int id, ModelMap model,
HttpServletRequest request) {
		Optional<NormaWeb> normaWeb = normaWebService.findById(id);
		if(!normaWebService.findById(id).get().getAutor().ge-
tEmail().equals(SecurityContextHolder.getContext().getAuthentication().getName()))
{
```

```
                    model.addAttribute("message","Solo puedes eliminar tus nor-
mas");
                    Log.warn("Un usuario esta intentando editar una norma sin te-
ner los permisos necesarios, con sesion "+request.getSession());
                    return listNormasWeb(model);
             }
             if(normaWeb.isPresent()) {
                    normaWebService.delete(normaWeb.get());
                    model.addAttribute("message", "La Norma Web se ha borrado con
exito");
             }
             else {
                    model.addAttribute("message", "No podemos encontrar la norma
Web que intenta borrar");
             }
             return listNormasWeb(model);
      }

}
```

-Nombre: NormaWebController
-Ruta:
src/main/java/org/springframework/samples/petclinic/web/NormaWebController.java

-Este controlador gestiona todo lo relacionado con la visualización, creación, edición y eliminación de normas web. De este modo la función lisNormasWeb obtiene todas las normas web del servicio y redirige a la vista donde se listan todas las normas web. La función initCreationForm y processCreationFrom gestionan el crear una Norma Web, de modo que primero redirige a un formulario para crearla y en el post se envían dichos datos al servicio para que este se encargue de guardar la Norma Web, en caso de haber errores en el formulario el post devuelve a la vista del formulario y se muestran los errores. Las funciones editNormaWeb y editNormasWeb gestionan la edición de las normas web, de modo que el get redirige al formulario donde se presentan los datos actuales de la Norma Web y tras modificarlos y hacer el post se guardan los cambios a través del servicio, en caso de haber errores en el formulario, el post devuelve a al formulario y se muestran los errores. Por último, la función deleteNormasWeb gestiona la eliminación de normas web, de modo que, dado una Norma Web, la elimina mediante el servicio. La edición y eliminación de las normas web se realizan únicamente si la Norma Web implicada tiene como autor a la persona que está intentado realizar la acción con la Norma Web.

Repositorio

```
package org.springframework.samples.petclinic.repository;

import java.util.Collection;
import java.util.Optional;

import org.springframework.dao.DataAccessException;
import org.springframework.data.repository.Repository;
import org.springframework.samples.petclinic.model.NormaWeb;

public interface NormaWebRepository extends Repository<NormaWeb,Integer>{

      Collection<NormaWeb> findAll() throws DataAccessException;
```

```
	Optional<NormaWeb> findById(int id) throws DataAccessException;

	void deleteById(int id) throws DataAccessException;

	void save(NormaWeb normaWeb) throws DataAccessException;

	int count() throws DataAccessException;
}
```

-Nombre: NormaWebRepository
-Ruta:
src/main/java/org/springframework/samples/petclinic/repository/NormaWebRepository.java

-El repositorio de Norma Web se encarga de comunicarse con la base de datos y es muy sencillo, ya que extiende a repository, el cual nos brinda los métodos findAll() para obtener todas las normas web, findById() para obtener una Norma Web a partir de una id de Norma Web, deleteById() para borrar una Norma Web a partir de una id de Norma Web, save() para guardar una Norma Web y count() para contar la cantidad de normas web existentes.

Conversor o Formatter (si aplica)

No aplica.

Validador y anotación asociada (si aplica, máximo de dos ejemplos)

No aplica.

Etiquetas personalizadas (si aplica, máximo de dos ejemplos)

No aplica.

Ejemplos de pruebas implementadas

Pruebas unitarias (máximo de dos ejemplos)

```
@Test
		public void shouldSaveFile() throws IOException {
			Path path = Paths.get("TestTxt");
			String name = "test.txt";
			byte[] content = "Hello".getBytes();

			MultipartFile file = new MockMultipartFile(name,content);

			this.fileService.saveFile(file, path, "test.txt");

			byte[] content2 = null;
			Path path2 = Paths.get("TestTxt/test.txt");
			try {
			    content2 = Files.readAllBytes(path2);
			} catch (final IOException e) {
			}
```

```
                assertThat(content).isEqualTo(content2);
                this.fileService.delete(Paths.get("TestTxt/test.txt"));
        }
```

-Nombre Fichero: FileServiceTest
-Ruta: src/test/java/org/springframework/samples/petclinic/service/FileServiceTests.java

-Esta prueba realiza la comprobación de guardar un archivo en el sistema. El Arrange abarcaría la creación del fichero que se va a guardar, a continuación, vendría el Act con la llamada a la función saveFile del fileService con la que se guardaría el archivo y finalmente se realizaría el Assert con el assertThat comprobando que el contenido del archivo que se ha guardado es igual a un array de bytes vacío. Finalmente se borra el fichero del sistema para que no haya conflictos más adelante a la hora de volver a ejecutar la prueba, ya que no se pueden subir dos archivos con el mismo nombre.

```
@Test
	@Transactional
	void shouldUpdateNormaWeb() {
		NormaWeb normaWeb = this.normaWebService.findById(0).get();
		String oldDescripcion = normaWeb.getDescripcion();
		String newDescripcion = oldDescripcion + "X";

		normaWeb.setDescripcion(newDescripcion);
		this.normaWebService.saveNormaWeb(normaWeb);

		// retrieving new name from database
		normaWeb = this.normaWebService.findById(0).get();
		assertThat(normaWeb.getDescripcion()).isEqualTo(newDescripcion);
	}
```

-Nombre Fichero: NormaWebServiceTest
-Ruta:
src/test/java/org/springframework/samples/petclinic/service/NormaWebServiceTest.java

-Esta prueba realiza la comprobación de editar una Norma Web. El Arrange es la obtención de una NormaWeb de la base de datos, ya que las pruebas de servicio las estamos haciendo de manera sociable. A continuación, se obtiene la descripción de la Norma Web obtenida y se modifica dicha descripción, tras esto se realiza el Act, donde se llama al servicio para guardar la Norma Web y finalmente se realiza el Assert con assertThat para comprobar que la descripción de la Norma Web en la base de datos es la misma que la que se había indicado al modificarla. Por otro lado, la prueba tiene la etiqueta @Transactional, ya que involucra el hecho de editar una Norma Web de la base de datos.

Pruebas unitarias parametrizadas (si aplica)

No aplica.

Pruebas de controlador

```
@BeforeEach
		void setup() {

			mockMvc = MockMvcBuilders
			    .webAppContextSetup(context)
			    .apply(SecurityMockMvcConfigurers.springSecurity())
			    .build();
```

```
                    tutor = new Tutor();
                    tutor.setId(TEST_TUTOR_ID);

                    problema = new Problema();
                    problema.setId(TEST_PROBLEMA_ID);
                    problema.setDescripcion("hola");
                    problema.setCasos_prueba("0 0 0");
                    problema.setSalida_esperada("1 1 1");
                    problema.setSeasonYear(1);
                    Temporada aux = new Temporada();
                    aux.setNombre("Invierno");
                    problema.setSeason(aux);

                    given(problemaService.findById(any(Integer.class))).willRe-
turn(Optional.of(problema));
                    given(tutorService.findByEmail(SecurityContextHolder.getCon-
text().getAuthentication().getName())).willReturn(Optional.of(tutor));


        }

        //Caso positivo de HU-16+E1 Aclaraciones en Problemas
        @WithMockUser(username="jesus@us.es",authorities="tutor")
    @Test
    void testProcessCreationFormSuccessAsTutor() throws Exception {
            mockMvc.perform(post("/aclaraciones/new")
                                        .with(csrf())
                                        .param("texto", "El problema se puede re-
solver en 2 líneas")
                                        .param("idProblema", "0"))
                    .andExpect(status().is3xxRedirection())
                    .andExpect(view().name("redirect:/problemas/0"));
        }
```

-Nombre: AclaracionControllerTest
-Ruta:
src/test/java/org/springframework/samples/petclinic/web/AclaracionControllerTests.java

-Esta prueba realiza la comprobación de crear una Aclaración. El Arrange está dentro del @BeforeEach, donde se crea el problema con sus atributos. Además, se mockean las llamadas a los servicios para obtener un problema en función de una id y obtener un tutor en función de un email. A continuación, se realiza el Act con una petición post con el parámetro de texto y el id del problema al que se va a añadir la aclaración y finalmente se realiza el Assert con los andExcept, comprobando que se realiza una redirección al problema al que se añade la aclaración. Todo el test se realiza autenticado como tutor, de lo contrario no se podría realizar dicho comentario.

Principales problemas encontrados

El principal problema que hemos tenido no solo yo si no a nivel de grupo ha podido ser la organización a la hora de repartir las tareas, ya que al comienzo del proyecto si es verdad que empezamos repartiendo más o menos de manera correcta las tareas, pero finalmente hemos acabado cada uno haciendo lo que faltara por realizar. Sin embargo, no considero que haya sido un caos total, pues el trabajo se ha realizado de manera bastante equitativa. Por otro

lado, en cuanto al desarrollo del proyecto, la única parte donde he experimentado un poco más de problemas ha sido a la hora de realizar la gestión de archivos, ya que en un comienzo intenté almacenar directamente los archivos en el modelo como byte [], sin embargo, tras indagar más en el tema e informarme con diferentes fuentes acabé dando con la mejor forma de implementarlo. Finalmente, la forma de guardar dichas imágenes fue almacenarlas en la carpeta estática del proyecto para que posteriormente se pudieran mostrar en el navegador y únicamente referenciar dichas imágenes con la ruta en el modelo de las entidades. Hablando acerca de los archivos, también surgió el problema acerca de cómo nombrar los archivos para que tuvieran todos un nombre distinto, en un principio pensé en realizar un contador para que todos los archivos tuvieran un nombre distinto, pero eso finalmente iba a generar algunos problemas de concurrencia, de modo que finalmente decidimos nombrar a los archivos con una cadena que seguía un patrón tal que "AñoActual+MesActual+DíaActual+HoraActual+MinutoActual+SegundoActual+NanoSegund oActual", de esta manera es prácticamente imposible que dos personas creen un archive al mismo tiempo .Dejando de lado el tema de la implementación, un problema en mi opinión ha sido la introducción a GitHub, ya que ha sido hasta hace no mucho que realmente tanto yo como mis compañeros hemos acabado finalmente de enterarnos de cómo utilizar la herramienta correctamente.

Otros comentarios

Aunque lo he mencionado en los apartados anteriores voy a aclarar ciertas partes del proyecto que han sido más específicas y no las he detallado a la hora de comentar los controladores y servicios implementados.

-La primera es la creación de un servicio para el almacenamiento de todas las imágenes de la página, además he implementado o modificado todos los formularios que incluyen la subida de un archivo a la página para que estos soporten la subida de un archivo (multipart/form-data). En adición, he implementado la gestión de los archivos en todos los controladores que tienen archivos. También he añadido una restricción de tamaño de archivo en el ApplicationProperties.

-La segunda es la gestión de errores en la aplicación, implementando un CustomErrorController, de modo que todos los errores que salten en la aplicación sean gestionados por dicho controlador y para los códigos de errores más comunes se redirige a la vista de excepcion.jsp con un mensaje de error personalizado para el código de error. Si el código de error no es uno de los contemplados el mensaje de error es un mensaje de error general.

-La tercera es en cuanto a la realización de ciertas paginaciones en la página, ya que a pesar de no haber realizado yo la api para obtener los datos de la paginación he realizado ciertas paginaciones en las vistas, estas están indicadas en apartados anteriores.

Detailed report

09/01/2020 - 02/09/2021

Total: 130:03:14 Billable: 00:00:00 Amount: 0.00 USD

Date	Description	Duration	User
02/09/2021	Creación AuthServiceTests y Actualizacion prueba de articuloService CodeUs G2-L6	01:39:00 11:30:00AM - 01:09:00PM	Jesapaort
02/08/2021	Realización Documento Individual CodeUs G2-L6	04:00:00 05:00:00PM - 09:00:00PM	Jesapaort
02/08/2021	Actualización Historias de usuario documentación CodeUs G2-L6	02:00:00 01:00:00PM - 03:00:00PM	Jesapaort
02/07/2021	Creación Entregable Final (patrones de diseño y estilos arquitectónicos) y toma de decisiones CodeUs G2-L6	04:00:00 05:00:00PM - 09:00:00PM	Jesapaort
02/07/2021	Creación Entregable Final (patrones de diseño y estilos arquitectónicos) y toma de decisiones CodeUs G2-L6	03:00:00 11:30:00AM - 02:30:00PM	Jesapaort
02/06/2021	Revisión controladores y servicios CodeUs G2-L6	01:00:00 07:30:00PM - 08:30:00PM	Jesapaort
02/06/2021	Actualización SecutiryConfiguration. Actualización visualización de código en envíos. Pruebas de Historias de usuario 15-19 CodeUs G2-L6	03:30:00 11:30:00AM - 03:00:00PM	Jesapaort
02/06/2021	Documentación Entregable Final CodeUs G2-L6	01:00:00 12:30:00AM - 01:30:00AM	Jesapaort
02/05/2021	Actualizacion de seguridad del proyecto y Documentación del proyecto CodeUs G2-L6	03:30:00 09:30:00AM - 01:00:00PM	Jesapaort
02/04/2021	Actualizacion visualización de errores en formularios en vistas. CodeUs G2-L6	02:45:00 05:00:00PM - 07:45:00PM	Jesapaort
02/04/2021	Actualizacion ArticuloServiceTest y ProblemaServiceTest. Actualización paginación vistas. CodeUs G2-L6	02:10:00 01:00:00AM - 03:10:00AM	Jesapaort

02/03/2021	Actualización vistas paginadas. Implementación codificación de contraseñas. Actualización pruebas NormasWebService y ProblemaService CodeUs G2-L6	04:30:00 04:30:00PM - 09:00:00PM	Jesapaort
02/03/2021	Actualización Pruebas aclaracionController y Creación de AdministradorControllerTest. Implementación de PerfilControllerTest CodeUs G2-L6	03:45:00 11:15:00AM - 03:00:00PM	Jesapaort
02/03/2021	AclaracionControllerTest y AdministradorControllerTest CodeUs G2-L6	02:45:00 12:30:00AM - 03:15:00AM	Jesapaort
02/02/2021	Paginación noticias en noticiasList y envíos en alumnosDetails. CodeUs G2-L6	03:30:00 05:15:00PM - 08:45:00PM	Jesapaort
02/02/2021	Implementación de CustomErrorController y vista correspondiente de errores. CodeUs G2-L6	03:30:00 11:15:00AM - 02:45:00PM	Jesapaort
02/01/2021	Actualización de controladores y servicios para gestión de sesiones. Incluir administrador a login. Implementación metodo get para new publicacion en foro CodeUs G2-L6	04:00:00 05:15:00PM - 09:15:00PM	Jesapaort
01/18/2021	Documentación Entregable 3 CodeUs G2-L6	01:00:00 11:30:00AM - 12:30:00PM	Jesapaort
01/17/2021	Pruebas controlador Aclaracion CodeUs G2-L6	03:00:00 05:30:00AM - 08:30:00AM	Jesapaort
01/11/2021	Validaciones de sesión en controladores CodeUs G2-L6	00:50:00 02:30:00PM - 03:20:00PM	Jesapaort
01/10/2021	Creación de perfil de Alumno,Creador y Tutor. Creación de la entidad PreguntaTutor con su servicio,controlador y vista CodeUs G2-L6	04:20:00 05:00:00PM - 09:20:00PM	Jesapaort
01/10/2021	Pruebas unitarias de servicios CodeUs G2-L6	02:00:00 01:00:00PM - 03:00:00PM	Jesapaort
01/10/2021	Documentación Entregable 3 CodeUs G2-L6	03:20:00 01:00:00AM - 04:20:00AM	Jesapaort
01/09/2021	Documentación Entregable 3 CodeUs G2-L6	05:00:00 05:00:00PM - 10:00:00PM	Jesapaort
01/09/2021	Documentación Entregable 3 CodeUs G2-L6	03:00:00 12:00:00PM - 03:00:00PM	Jesapaort

01/08/2021	Instalación Docker y edición de planificación CodeUs G2-L6	03:30:00 05:30:00PM - 09:00:00PM	Jesapaort
01/08/2021	Credenciales y seguridad CodeUs G2-L6	02:00:00 01:00:00PM - 03:00:00PM	Jesapaort
01/08/2021	Pruebas FileService CodeUs G2-L6	00:30:00 12:30:00PM - 01:00:00PM	Jesapaort
01/07/2021	Pruebas de Servicios (Alumno,Articulo,File) CodeUs G2-L6	04:00:00 05:00:00PM - 09:00:00PM	Jesapaort
01/07/2021	Pruebas de Servicios CodeUs G2-L6	02:20:00 12:40:00PM - 03:00:00PM	Jesapaort
01/04/2021	Vista de competiciones CodeUs G2-L6	01:30:00 06:00:00PM - 07:30:00PM	Jesapaort
12/28/2020	Implementación de compartir soluciones de envíos CodeUs G2-L6	02:00:00 06:45:00PM - 08:45:00PM	Jesapaort
12/23/2020	Crear Controlador , Servicio y Vista de Administrador CodeUs G2-L6	00:30:00 08:15:00PM - 08:45:00PM	Jesapaort
12/23/2020	Organización de imagenes de entidades por carpetas, Actualización de controladores (volver a formulario cuando haya errores) CodeUs G2-L6	03:00:00 05:15:00PM - 08:15:00PM	Jesapaort
12/22/2020	Vista y Controlador de añadir noticia. Subida de imágenes a disco en entidades: noticia,alumno,articulo,competicion,creador,tutor CodeUs G2-L6	03:30:00 06:30:00PM - 10:00:00PM	Jesapaort
12/21/2020	Controlador entidad Problema CodeUs G2-L6	03:00:00 11:30:00AM - 02:30:00PM	Jesapaort
12/20/2020	Subida de archivos a disco CodeUs G2-L6	00:30:00 11:45:00PM - 12:15:00AM	Jesapaort
12/20/2020	Entidad Problema CodeUs G2-L6	02:30:00 07:00:00PM - 09:30:00PM	Jesapaort
12/19/2020	Subida de archivos a disco CodeUs G2-L6	00:45:00 05:15:00PM - 06:00:00PM	Jesapaort
12/19/2020	Subida de archivos a disco CodeUs G2-L6	02:00:00 12:00:00PM - 02:00:00PM	Jesapaort

12/17/2020	Subida de archivos a disco CodeUs G2-L6	03:00:00 06:00:00PM - 09:00:00PM	Jesapaort
12/16/2020	Subida de archivos a disco CodeUs G2-L6	02:00:00 06:00:00PM - 08:00:00PM	Jesapaort
12/16/2020	Subida de archivos a disco CodeUs G2-L6	02:30:00 12:30:00AM - 03:00:00AM	Jesapaort
12/05/2020	Ranking de alumnos (Historia de usuario 4) CodeUs G2-L6	01:00:00 08:00:00PM - 09:00:00PM	Jesapaort
11/28/2020	Actualización de Problemas y Pruebas Unitarias de Normas Web y Problemas CodeUs G2-L6	02:00:00 06:00:00PM - 08:00:00PM	Jesapaort
11/23/2020	Pruebas Unitarias de Normas Web y Problemas CodeUs G2-L6	01:00:00 11:30:00AM - 12:30:00PM	Jesapaort
11/22/2020	Entidad Problema CodeUs G2-L6	02:00:00 06:00:00PM - 08:00:00PM	Jesapaort
11/21/2020	Entidad NormaWeb CodeUs G2-L6	01:00:00 06:30:00PM - 07:30:00PM	Jesapaort
11/20/2020	Entidad NormaWeb CodeUs G2-L6	01:00:00 12:30:00PM - 01:30:00PM	Jesapaort
11/19/2020	Entidad NormaWeb CodeUs G2-L6	00:35:00 10:00:00PM - 10:35:00PM	Jesapaort
11/18/2020	Entidad NormaWeb CodeUs G2-L6	00:56:00 08:30:00PM - 09:26:00PM	Jesapaort
11/14/2020	Creación de mockups CodeUs G2-L6	03:10:00 12:00:00PM - 03:10:00PM	Jesapaort
11/13/2020	Construcción del modelo de negocio CodeUs G2-L6	00:53:14 01:37:00PM - 02:30:14PM	Jesapaort
11/04/2020	Puesta en común y descripción general del proyecto. CodeUs G2-L6	01:30:00 01:00:00PM - 02:30:00PM	Jesapaort
11/03/2020	Reglas de negocio. CodeUs G2-L6	01:00:00 01:30:00PM - 02:30:00PM	Jesapaort
10/19/2020	Enumeración de entidades, roles e historias de usuario. CodeUs G2-L6	01:50:00 10:40:00PM - 12:30:00AM	Jesapaort