PLATAFORMA WEB DE E-LEARNING PARA COLEGIOS

Eduardo Rodríguez Carro – TFG Desarrollo Web

Índice

- Motivación
- Introducción
- Tecnologías
- Diseño
- Desarrollo
- Resultado
- Pruebas y conclusiones

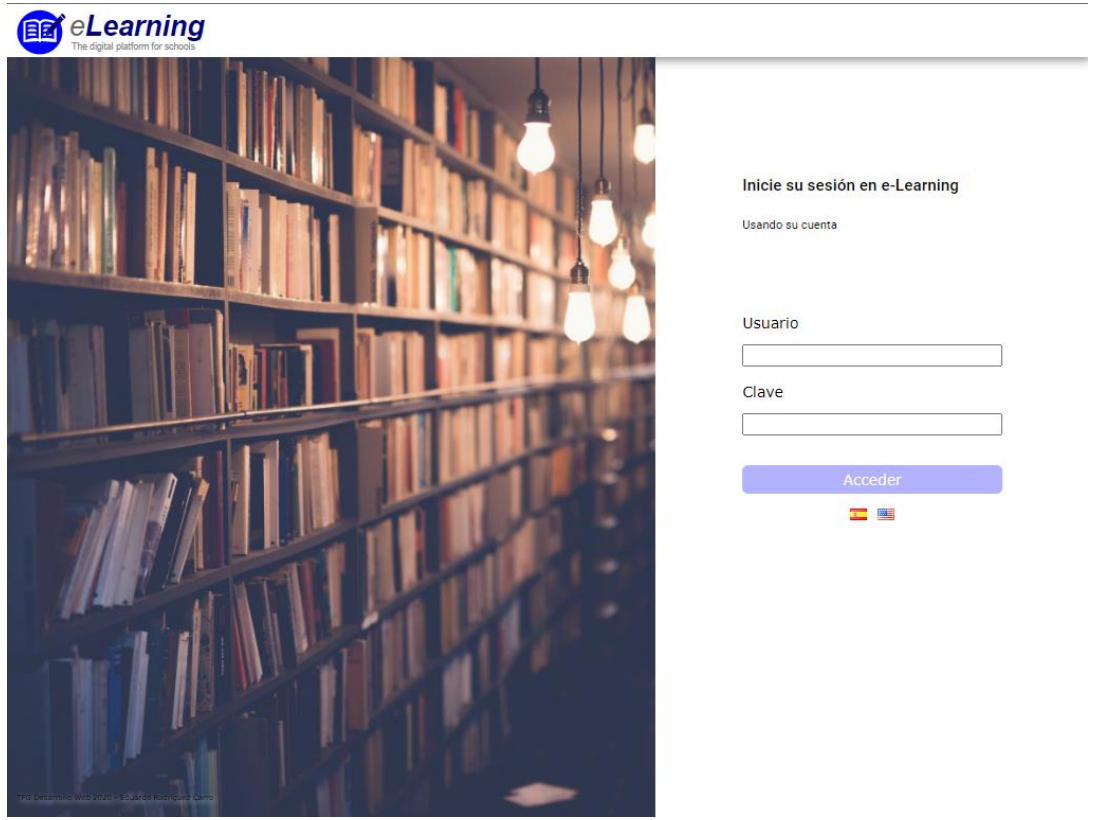
Motivación

- Este proyecto esta motivado principal por la situación actual del COVID y el reto que supone para las instituciones de enseñanza.
- Por la complejidad que supone para los profesores e instituciones el uso de diferentes aplicaciones y herramientas por no existir una solución integrada para todos los casos
- Así mismo existe un afán de aplicar los conocimientos de organización, análisis y desarrollo aprendidos durante la realización del grado
- Por ultimo la experimentación con nuevas tecnologías heterogéneas que forman parte de los desarrollos mas punteros actualmente

Introducción

- La solución planteada comprende una aplicación web de fácil uso para la enseñanza online, orientada al uso por centros de enseñanza donde se ha de implicar a los tutores legales de estos como parte del proceso.

- El Proyecto tiene un alcance limitado debido a los recursos y tiempo disponibles pero comprende una base solida sobre la que ver los conceptos para este tipo de aplicaciones.



Planificación

Planificación del trabajo

- Creación de un plan estructurado e hitos
- Evaluación de riesgos
- Elección de tecnologías y arquitectura

Análisis y diseño

- Toma de requisitos
- Análisis técnico y funcional
- Diseño de modelos de datos
- Diseño UI/UX, Maqueta (Wireframe)

Implementación

- Creación del entorno de Integración continua
- Desarrollo de los servicios
- Desarrollo de la interfaz
- Desarrollo y ejecución de las pruebas en el entorno de integración continua
- Informes de pruebas

Finalización

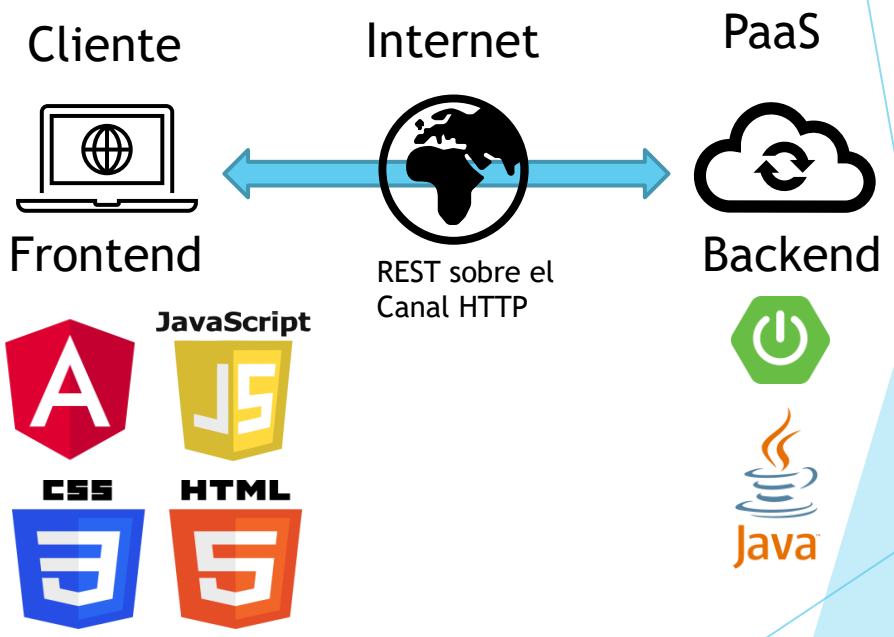
- Memoria
- Presentación
- Video

Tecnologías

La solución se divide en dos aplicaciones independientes en un mismo paquete.

La primera parte(Backend) esta pensada para ser ejecutada proveedor de servicios PaaS y contiene la capa de negocio y la capa de acceso junto con una capa de seguridad. Esta parte se ejecuta usando la VM de Java y el framework de Spring Boot que provee de una completa plataforma de desarrollo.

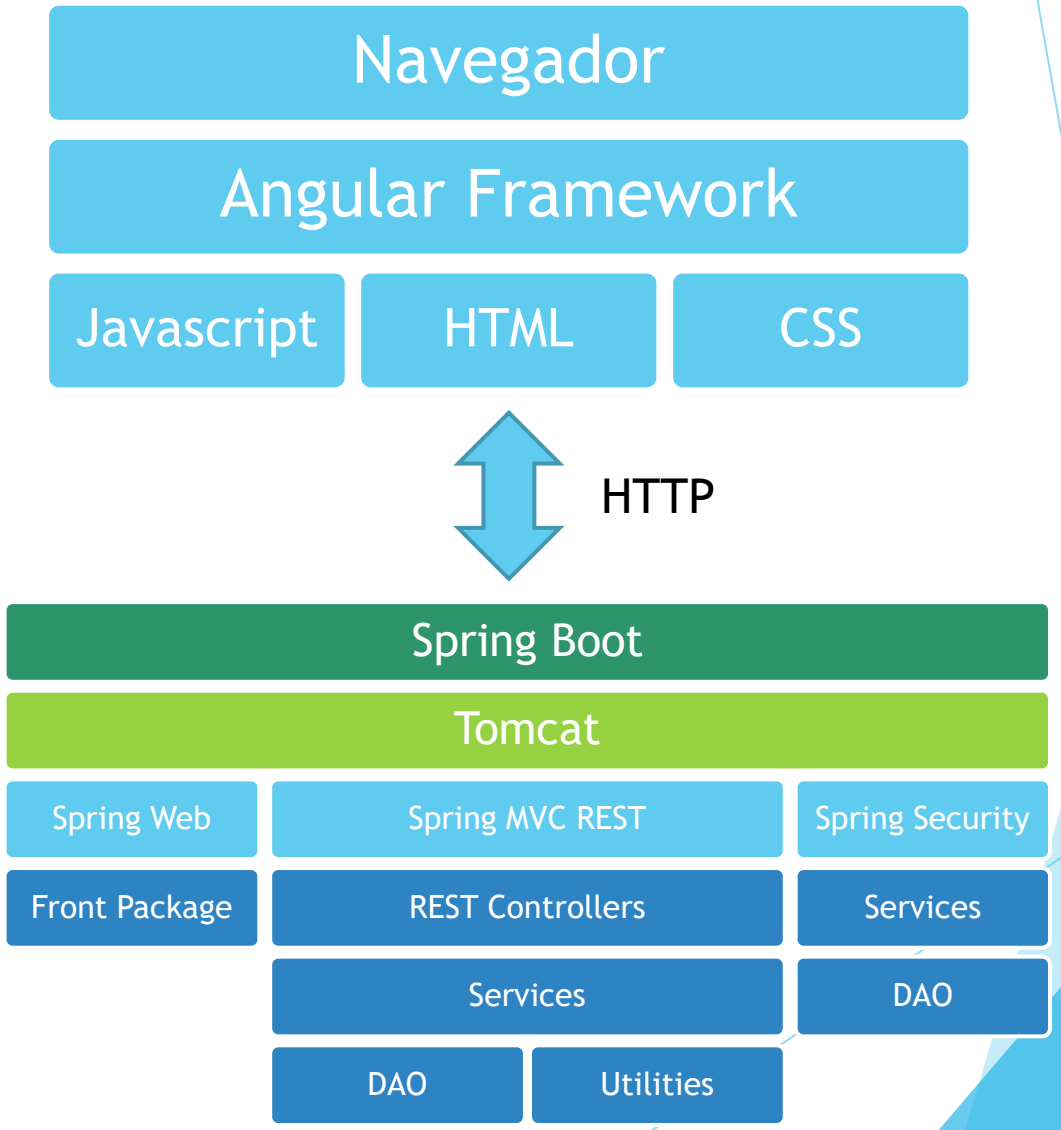
La segunda parte (Frontend) se descarga al cliente desde el paquete del Backend y provee de una aplicación Angular que se ejecuta íntegramente en cliente consumiendo los servicios de backend. Esta aplicación genera la vista de HTML y CSS necesaria para el navegador



Diseño I - Arquitectura

La solución se diseña como dos aplicaciones separadas que se comunican mediante un protocolo sin estado.

Esto hace de la aplicación altamente escalable y resiliente al poder ejecutar tantas instancias del backend como se requieran para procesar los datos, mientras que el frontend es ejecutado por cada uno de los clientes individualmente.

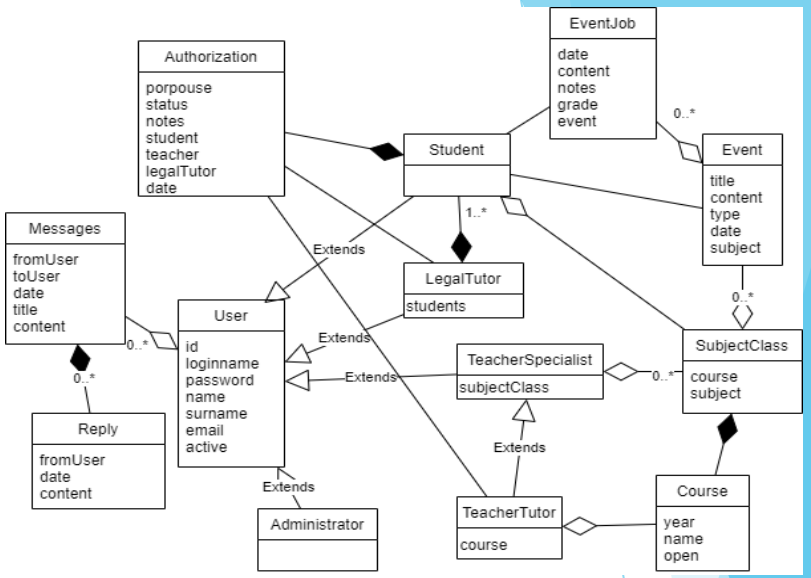


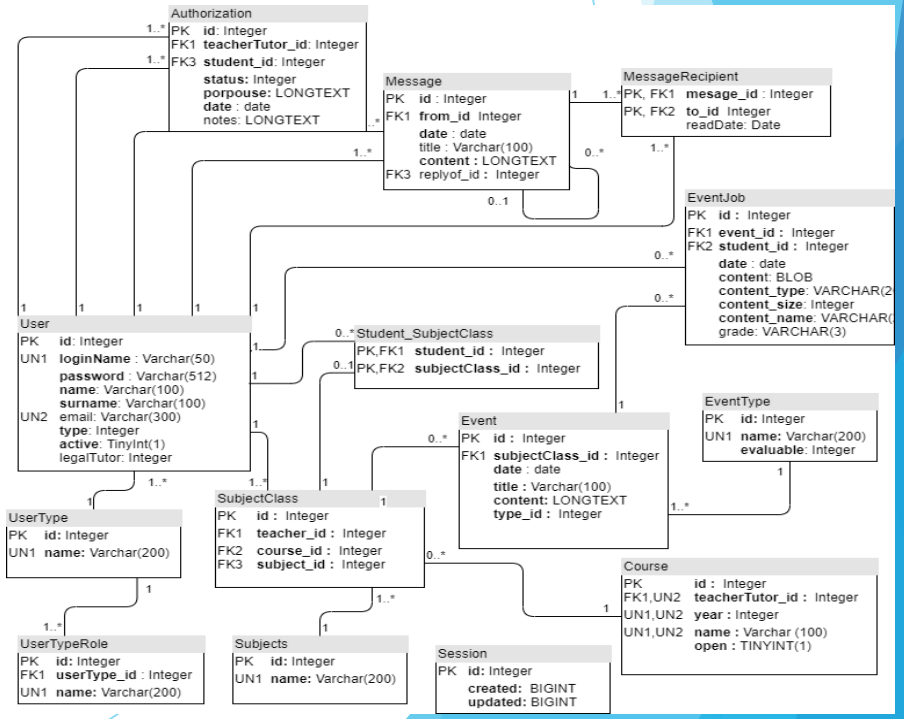
Diseño II – Base de datos

Para el almacenamiento de datos se ha planteado el uso de un SGBD relacional dado que su coste actualmente es menor que una base de datos NoSQL en los proveedores actuales y no se ha detectado realmente una necesidad de su uso dado que a nivel institucional no se esperan unos volúmenes de datos enormes, pero si se requiera una integridad referencial fuerte.

Para el diseño se ha iniciado con un modelo lógico y una vez refinado se ha creado el modelo físico para la base de datos seleccionada, en este caso MariaDB.

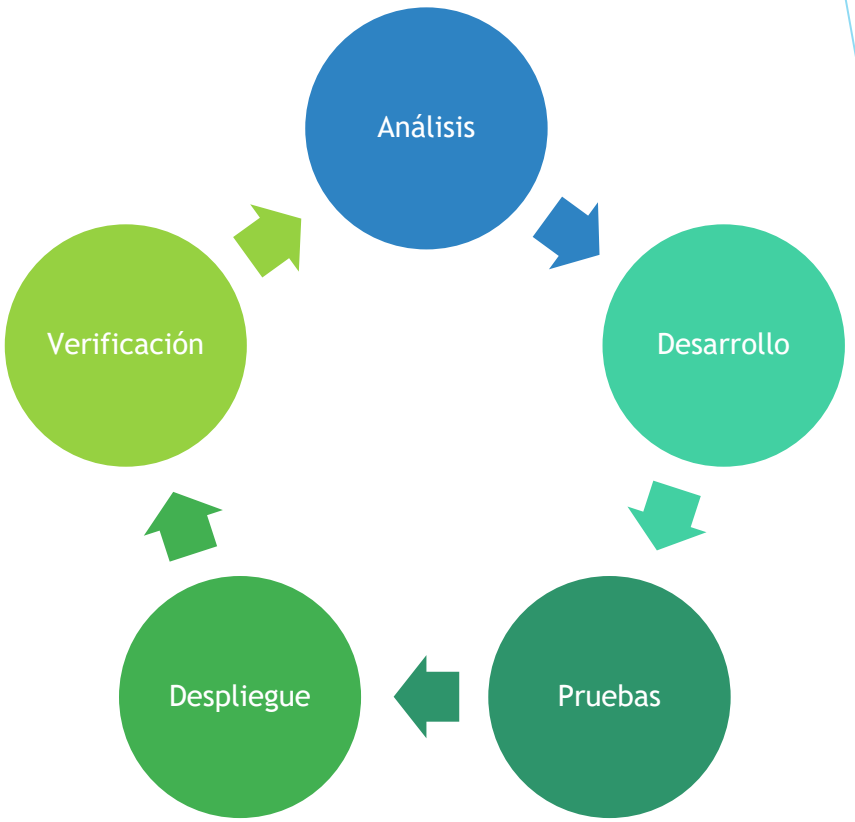
Así mismo para el acceso a la base de datos se ha optado por Spring Data JPA. Lo que permitirá el usar otros gestores de datos o cambiar fácilmente esta capa para un uso de una base de datos NoSQL en un futuro.





Desarrollo I

- El proyecto se ha desarrollado paralelamente en dos lenguajes diferentes orientados a cada una de las partes y que comprenden formas de organizar las lógica y proyecto muy diferentes.
- Para poder avanzar en esta fase se ha implementado sobre Gitlab un el soporte de Integración continua junto con la realización de pruebas unitarias según se ha avanzado el proyecto a fin de tener una detección temprana de carencias



Desarrollo II - Frontend

- El Frontend **Typescript** con el framework Angular que convierte el resultado en **javascript**, **HTML** y **CSS** empaquetados como una **SPA** (Single page application)
- Para ello se ha indicado a **Typescript** el uso de **ES6** y **ECMA2020** como estándar **JavaScript**
- Este lenguaje usa activamente bajo la ayuda de **RxJS** la ejecución en paralelo bajo promesas y observadores nativos del navegador

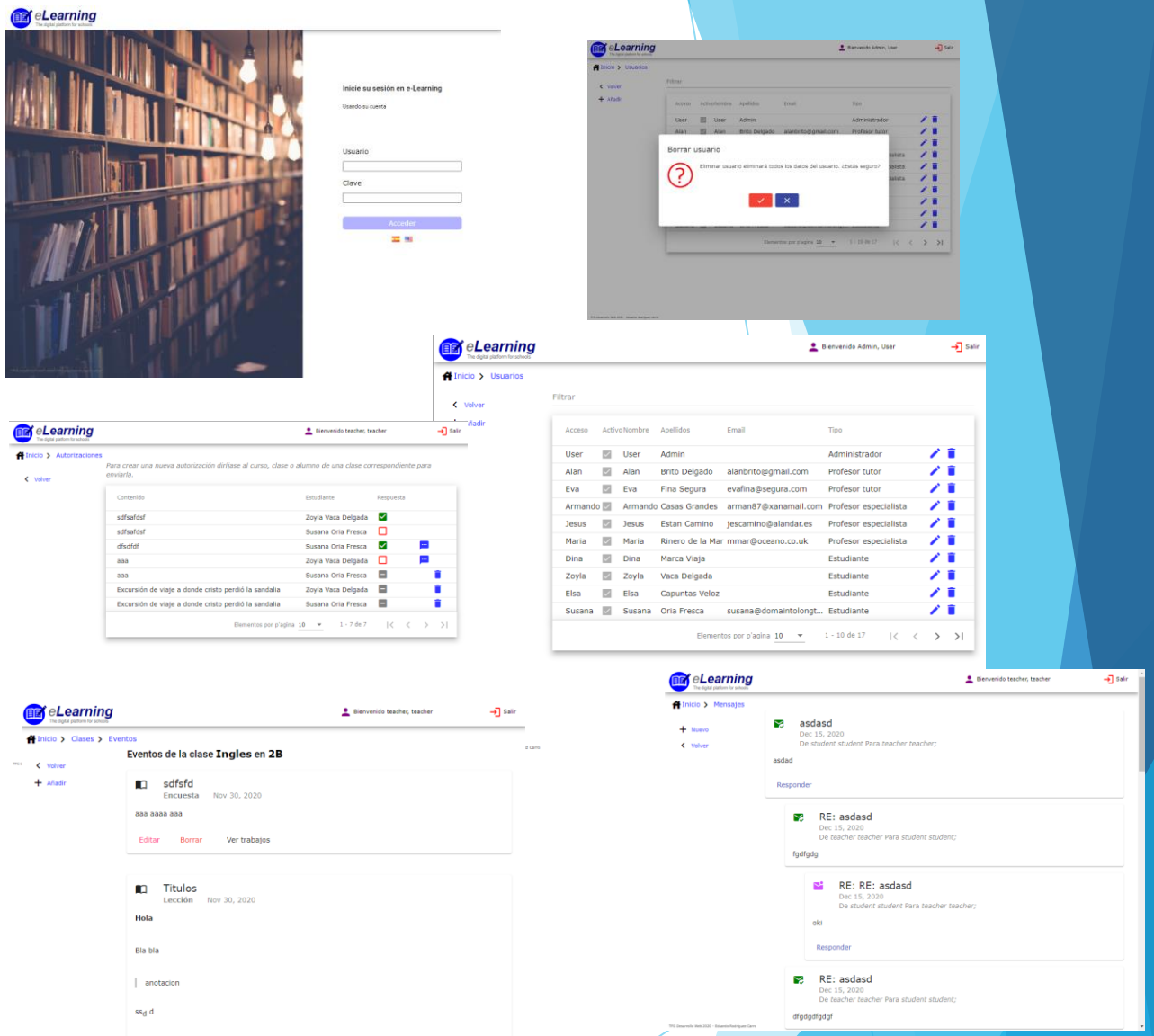
Desarrollo III - Backend

- El Backend del proyecto se ha desarrollado usando **Spring Boot**. Una tecnología basada en **Java** establecida como una de las mejoras alternativas para el desarrollo de servicios Cloud y con un amplio repertorio de componentes de libre uso para el desarrollo.
- Spring Boot permite la ampliación de forma sencilla del proyecto así como la integración en diferentes entornos y servicios

```java
/**
 * The Class CourseController.
 * @author Eduardo Rodriguez Carro
 */
@RestController
@CrossOrigin
@RequestMapping("/courses")
public class CourseController {

    /** The course service. */
    @Autowired
    private CourseService courseService;

    /**
     * Listado de usuarios por tipo.
     *
     * @return the all
     * @throws Exception the exception
     */
    @PreAuthorize("hasRole('ROLE_API_COURSES')")
    @RequestMapping(value = "/all", method = RequestMethod.GET)
    public ResponseEntity<List<Course>> getAll() throws Exception {
        return ResponseEntity.ok(courseService.getAll());
    }

    /**
     * Listado de usuarios por tipo.
     *
     * @param authentication the authentication
     * @return the my courses
     * @throws Exception the exception
     */
    @PreAuthorize("hasRole('ROLE_API_MY_COURSES')")
    @RequestMapping(value = "/my", method = RequestMethod.GET)
```

Resultado

- Como resultado obtenemos un paquete JAR con todos los elementos que puede ser ejecutado en cualquier maquina con una JVM
- Spring Boot esta preparado para su integración con Docker y existen multitud de configuraciones incluyendo el despliegue en Cloud
- La aplicación web resultante se puede ejecutar en cualquier navegador moderno y mediante los procesos de transpiración de Typescript se puede ofrecer una alternativa para los navegadores menos actuales.

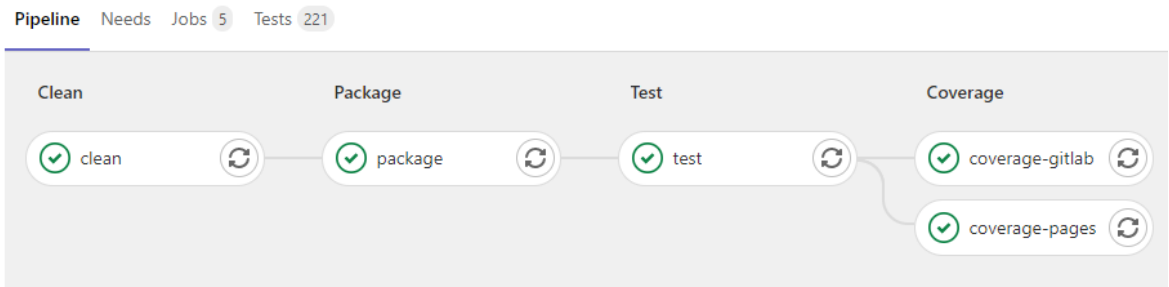


Pruebas

- Se implementa un conjunto de pruebas de backend con Junit y jacoco y de FrontEnd Con Jasmine para verificar durante el Desarrollo que no se generan problemas en las funcionalidades al crear otras nuevas.
- Todas estas pruebas se ejecutan de forma automática en cada subida de código mediante el soporte de pipelines de Gitlab y envían por email un resultado de la misma para una detección temprana de fallos

schoolelearning

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.uoc.tfg.sel.web		35 %		25 %	61	102	147	231	33	72	0	10
org.uoc.tfg.sel.security		0 %		0 %	23	23	53	53	11	11	4	4
org.uoc.tfg.sel.service		79 %		66 %	36	111	58	255	15	74	0	9
org.uoc.tfg.sel.web.model		43 %		100 %	41	48	60	84	41	47	6	8
org.uoc.tfg.sel.security.model		20 %		0 %	29	39	42	57	26	36	2	4
org.uoc.tfg.sel.config		18 %		n/a	8	11	21	29	8	11	2	3
org.uoc.tfg.sel.validation		19 %		20 %	18	23	26	36	13	18	1	4
org.uoc.tfg.sel.repository.model		91 %		16 %	15	164	20	260	0	149	0	13
org.uoc.tfg.sel		37 %		n/a	1	2	2	3	1	2	0	1
Total	1.654 of 3.815	56 %	133 of 206	35 %	232	523	429	1.008	148	420	15	56



Conclusiones

- Objetivos cumplidos
  - Crear un producto mínimo viable
  - Uso de nuevas tecnologías
- Objetivos futuros
  - Ampliar la funcionalidad del sitio
    - Perfil de usuarios
    - Notificaciones por correo electrónico
    - Cuestionarios en línea
    - Tablón del usuario en la pagina principal
  - Mejorar el diseño de la interfaz junto con mas imágenes e iconografía de ayuda al usuario