Framework de desarrollo de aplicaciones en la Universidad Miguel Hernández de Elche

SDKUMH - Software Development Kit UMH

Índice

1. Preámbulo **1**

2. Tecnologías empleadas **2**

2.1 Metodologías de gestión de proyectos y recursos humanos 2

2.2 Arquitectura básica del sistema 3

2.2.1 Capa de presentación o interfaz gráfico de usuario 4

2.2.1.1 Repositorio de artefactos 4

2.2.2 Capa de lógica de negocio 4

2.2.3 Capa de servicios o API de servicios 5

2.2.4 Capa de persistencia o de acceso a datos 5

3. Herramientas integradoras, de construcción de proyectos y resolución de dependencias **5**

3.1 Herramientas integradoras del resto de herramientas 5

3.2 Herramientas integradoras, de construcción de proyectos y resolución de dependencias 5

4. Control de versiones e integración continua **6**

5. Test unitarios **6**

1. Preámbulo

En el presente documento se establecen las bases en el desarrollo de aplicaciones web en la Universidad Miguel Hernández de Elche para permitir a los desarrolladores de la institución y a los de las empresas que acudan a una licitación de la misma, desarrollar bajo un modelo de trabajo más estandarizado con el fin de mostrar a los usuarios finales, un producto más duradero, con una interfaz mucho más amigable, moderno y orientado a satisfacer las necesidades que actualmente se requieren en los procesos administrativos de la Universidad.

El framework SDKUMH satisface las necesidades que en este momento tiene la Universidad Miguel Hernández de Elche, dándole una forma estandarizada de trabajo, buscando la manera de integrar diversas tecnologías, preferiblemente de software libre, que permitiera desarrollar una herramienta de entorno empresarial.

2. Tecnologías empleadas

Tras realizar investigación, desarrollos y pruebas sobre herramientas y tecnologías diferentes, el framework SDKUMH se implementa en base a cuatro ámbitos:

- herramientas que configuran el entorno de desarrollo
- tecnologías empleadas
- metodologías para gestión de proyectos y recursos humanos
- despliegues en producción

Dentro de estos ámbitos se han ido considerando para cada proyecto, según las necesidades deseadas a lo largo de sus diferentes procesos, diversas herramientas que se encuentran englobadas en las siguientes:

2.1 Metodologías de gestión de proyectos y recursos humanos

En contraposición con las metodologías clásicas, el uso de metodologías ágiles se basa en conseguir versiones operativas del producto que estamos desarrollando al final de cada ciclo de desarrollo propuesto y en conseguir que el usuario final se integre en el proyecto como parte del equipo de desarrollo.

A su vez, en la UMH buscamos siempre una mejora continua en sus productos software y un time-to-market optimizado, por lo que siempre se ha de buscar una gran calidad de las aplicaciones en su desarrollo y una elevada satisfacción de los usuarios, principios sobre los cuales se asientan este tipo de metodologías ágiles, proporcionando valor al producto de forma constante en cada iteración y sin necesidad de esperar a tener el producto completamente desarrollado para realizar pruebas a todos los niveles con el continuo e imprescindible feedback de los usuarios finales que usarán la aplicación en su día a día.

En la UMH hemos optado por la implantación de metodologías ágiles de trabajo Scrum y Kanban, que requieren una formación exhaustiva del equipo de desarrollo y el uso de ciertas herramientas de apoyo que ayuden a tener estos ciclos iterativos bajo control y que saquen el máximo rendimiento de nuestro equipo.

Las herramientas de apoyo escogidas son:

- JIRA y OpenProject: Gestión de proyectos e incidencias.
- Confluence: Gestión de la documentación y gestión del conocimiento.

A pesar de que resulta de importancia que la formación no sea algo puntual que se tenga que realizar únicamente al principio de la creación de un equipo ágil, es necesario que estos equipos estén motivados continuamente, por lo que el papel de un facilitador o un coach dentro de la organización, será de vital importancia.

A su vez, será necesaria una formación técnica adicional en conceptos como compartición del código, integración continua o pruebas unitarias.

2.2 Arquitectura básica del sistema

La programación por capas es un estilo de programación en el que el principal objetivo es la separación entre la lógica de negocio y la lógica de diseño. La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles, y en caso de que sobrevenga algún cambio, éste se centra sólo en el nivel requerido aislándolo del resto del código perteneciente a otros niveles.

A su vez, trabajar por capas permite la reutilización de código, otorga facilidad para el mantenimiento, separación de responsabilidades y escalabilidad de la aplicación.

Aunque dependerá de la complejidad del negocio, el SDKUMH normalmente usará una arquitectura de 4 capas.

- Capa de presentación o interfaz gráfico de usuario: Mediante la capa de presentación se separa la interacción del usuario respecto a la lógica de negocio. Debe cumplir con los requisitos impuestos de eficiencia, usabilidad y accesibilidad.
- Capa de lógica de Negocio o modelo: Encargada de mantener la coherencia de la aplicación mediante la implementación de las reglas de negocio necesarias. Esta capa será la encargada de interactuar con la de persistencias cuando sea necesario.
- Capa de servicios o API de servicios: Es la capa encargada de proporcionar acceso a la capa de la lógica de negocio exponiendo los recursos clave de la organización. Su comunicación se realizará a través del protocolo HTTP y con formatos estándar de intercambio de información como XML o JSON.
- Capa de persistencia o de acceso a datos: Si la aplicación está diseñada con orientación a objetos, la persistencia se logra por serialización del objeto o almacenamiento en una base de datos. Una capa de persistencia encapsula el comportamiento necesario para mantener los objetos, es decir, leer, escribir y borrar objetos en el almacenamiento persistente (base de datos).

Debido a la multitud de plataformas empleadas donde se mostrará la capa de presentación (entornos, web, dispositivos móviles, etc.), será necesario desacoplarla con el fin de que terceros puedan desarrollar interfaces alternativos de cara a su explotación en dichas plataformas.

De forma básica, dividiremos a los desarrolladores en dos especialidades: front-end y back-end.

- Los desarrolladores front-end se encargarán de “lo que el usuario puede ver” e interactuar. Se encargan del diseño, funcionalidad y experiencia de usuario.
- Los desarrolladores de back-end, por otra parte, hacen posible el front-end. Trabajan con tecnologías de servidor y se encargan de la comunicación entre la base de datos y el navegador.

2.2.1 Capa de presentación o interfaz gráfico de usuario

Hacemos uso de Angular para aplicaciones web. Angular se trata de un framework utilizado para el desarrollo de aplicaciones web de una sola página, y está desarrollado en TypeScript. El propósito de Angular es el de generar una mayor cantidad de aplicaciones basadas en un navegador, permitiendo el uso del modelo vista controlador (MVC), o mejor dicho, una variante más flexible llamada MVW (Model View Whatever).

El uso de angular permite tener desacoplada la capa de presentación o interfaces de usuario de gran parte de la lógica detrás de ellas.

Para formularios web hacemos uso de Orbeon Forms en caso de que requieran validaciones complejas o amplias colecciones de formularios. Orbeon es un sofware que permite este tipo de gestión de formularios que incluyan este tipo de características. Implementa el estándar W3C XForms y está disponible en una Edición Community de código abierto, así como en una Edición Profesional con soporte comercial.

Para las aplicaciones móviles hacemos uso del framework de desarrollo Ionic. Se trata de un kit de herramientas de código abierto para construir aplicaciones móviles y de escritorio haciendo uso de tecnologías web (HTML, CSS, Javascript y Angular).

2.2.1.1 Repositorio de artefactos

Para la gestión de los artefactos generados y sus versiones, en la UMH empleamos Nexus. Nexus se emplea para las librerías de frontend Angular de la plantilla UMH. La URL de acceso al repositorio de artefactos es: https://nexus.umhnet.es/

2.2.2 Capa de lógica de negocio

Una de las funcionalidades más importantes que presenta el framework SDKUMH, es la utilización de los flujos empresariales. Esta funcionalidad, se ha adaptado a las necesidades del framework, haciendo uso de Camunda BPM.

Camunda es una solución de gestión de procesos empresariales (BPM) orientada a desarrolladores y personas del entorno empresarial. Incorpora un motor de procesos empresariales de alto rendimiento con la flexibilidad y escalabilidad necesarias para manejar una amplia variedad de procesos críticos.

3. Herramientas integradoras, de construcción de proyectos y resolución de dependencias

3.1 Herramientas integradoras del resto de herramientas

En la UMH empleamos diferentes IDEs, tales como Visual Studio Code y Visual Studio, de Microsoft, y Pycharm u otros IDEs de Jetbrains.

3.2 Herramientas integradoras, de construcción de proyectos y resolución de dependencias

En sí, el uso de tecnologías de construcción de proyectos tienen varias funciones sobre este framework. En primera instancia se hace de esta tecnología para la gestión de dependencias, lo cual permite tener acceso a librerías ubicadas en un repositorio central, y además el poder compartir dependencias entre cada uno de los proyectos que tiene SDKUMH.

En la UMH se emplean dos herramientas:

- Maven: Herramienta de construcción de proyectos Java.
- MSBuild: herramienta para construir proyectos en .NET

4. Control de versiones e integración continua

Para el control de versiones en la UMH empleamos mayormente repositorios de tipo GIT. En concreto usamos GITLAB. La URL interna (solo accesible desde la UMH) del repositorio es: https://git.umhnet.es/

Respecto a la integración continua en la UMH se emplea Jenkins. Jenkins permite compilar y desplegar de manera automática en todos los entornos. La URL interna (solo accesible desde la UMH) para acceder a Jenkins es: https://intcontinua.umhnet.es/

5. Test unitarios

En la UMH realizamos tests unitarios en JAVA mediante jUnit. Para .NET hacemos uso de NUnit y xUnit (.NET core)
CI