ED01

EXPOSICIÓN DEL CASO:

La empresa BK ha recibido un nuevo encargo de software.

Se trata de diseñar una aplicación para una tienda especializada en vender productos estéticos.

La tienda desea trabajar con software libre. Además, desea explícitamente que la aplicación sea capaz de cumplir las siguientes tareas:

- Proporcionar facturas de las ventas.
- Llevar la cuenta de lo que vende cada trabajador.
- Controlar el stock de productos en almacén.
- Operar con lector de código de barras y tarjetas de crédito.
- Controlar los precios de los productos y ofrecer la posibilidad de operar con ellos.
- El tiempo de respuesta de la aplicación ha de ser lo menor posible.
- No se podrán procesar dos peticiones a la vez, aunque haya varios equipos funcionando simultáneamente.
- La empresa también quiere almacenar información de sus trabajadores: DNI, nombre, apellidos, número de la Seguridad Social, fecha de nacimiento, teléfono y localidad. Asimismo, de los productos interesa almacenar: código, marca, nombre comercial, precio, cantidad.

Tendrás que diseñar una planificación del proyecto de desarrollo de ese software que cumpla con las premisas estudiadas en la presente unidad de trabajo.

Esencialmente, el proyecto se divide en los siguientes apartados:

- Sintetiza el análisis de requerimientos del sistema para nuestro cliente. Plantea el diseño y determina el modelo de ciclo de vida más idóneo para esta aplicación.
- Planifica la codificación, indicando el lenguaje de programación y las herramientas que usarías para la obtención del código fuente, objeto y ejecutable, explicando por qué eliges esas herramientas.
- Planifica las restantes fases del ciclo de vida, indicando en cada una el objetivo que persigues y cómo lo harías.

Kevin Zamora Amela

ED01

RESOLUCIÓN:

1. Sintetiza el análisis de requerimientos del sistema para nuestro cliente. Plantea el diseño y determina el modelo de ciclo de vida más idóneo para esta aplicación.

Análisis:

- Requisitos funcionales:
  - Capacidad para Generar y Proporcionar Facturas de las ventas.
  - Interfaz/Vista para Supervisar y Gestionar el número y el contenido de las Ventas realizadas por cada trabajador.
  - Interfaz/Vista para Monitorear y Controlar el Stock de los Productos en almacen.
  - Funcionalidad y Medios para poder Operar con lectores de Codigo de Barras y Tarjetas de Crédito.
  - Funciones CRUD para Gestionar, Controlar y Operar sobre las diferentes propiedades/campos pertenecientes a la entidad Producto; considerando mismamente al 'precio' como una propiedad/campo de esta entidad/clase/interfaz.
  - Interfaz/Vista para Introducir, Gestionar y Consultar los parámetros que componen la entidad/clase/interfaz Trabajador. En consecuencia: Se precisará de elementos: de entrada (inputs) para su 'cumplimentación' y aporte, de la conexión con un servicio de almacenamiento (data base / base de datos) y de los consiguiente 'componentes' para la presentación y organización de dicha información (outputs), ante la empresa.

    Por otra parte, también se precisará otra Interfaz/Vista para Introducir, Gestionar y Consultar la información referente a la entidad Producto, la cual estará contenida y almacenada en una tabla de nuestra base de datos.

    Cada variable/campo de ambas entidades contará con su propio tipo de dato, el cual será definido inicialmente a continuación: [Entidad Trabajador]: DNI(String/Char) / nombre y apellidos (String/Char) / número de la Seguridad Social (Number/Int) / fecha de nacimiento (Date/LocalDate (etc.)) / teléfono (Number/Int o String/Char) / localidad (String/Char) [Entidad Producto]: código (Number/Int o String/Char) / marca (String/Char) / nombre comercial (String/Char) / precio (float/double) / cantidad (number/int) PD: Para la implementación de todas las propiedades/campos se utilizará la nomenclatura en formato 'Camel Case', PE: 'fechaNacimiento' o 'fechaDeNacimiento'.

- Requisitos NO funcionales:
  - Utilizar 'Software' Libre'
  - Optimización de la Aplicación: Tiempo de Respuesta Bajo (todo lo posible)
  - Limitación y Gestión del proceso de Generación de Peticiones: Límite de una Petición Simultanea y Sincrona. Por ende, sólo un/a usuario/a podrá crear su petición, a la vez. (Limitación funcional similar al 'envío de datos half duplex').

ED01

Diseño:

Employe {
    id: int;
    dni: String; (Depende del lenguaje de programación usado)
    name: String;
    lastName: String;
    SSnumber: int;
    bornedDate: LocalDate; (Depende del lenguaje de programación usado) };

Product {
    id: int;
    code: int;
    brand: String; (Depende del lenguaje de programación usado)
    comercialName: String;
    price: double;
    discount: double;
    quantity: int; };

Invoice {
    id: int;
    invoiceNumber: int;
    expeditionDate: LocalDate; (Depende del lenguaje de programación usado)
    productName: String;
    stock: int;
    price: double;
    seller: String; → (PE: employe.dni) };

- Para calcular el total de una factura:
  - (1) Multiplicaremos el Precio Unitario (su PVP sin impuestos) por el Nº de Unidades de cada producto.
  - (2) Realizaremos la suma ‘vertical’ de cada uno de los ‘subtotales’ (los resultados de cada multiplicación entre el PVP (Precio de Venta al Público) y el número de unidades de cada producto).
  - (3) Aplicamos el/los impuestos correspondientes (PE: IVA) sobre el resultado obtenido de la operación anterior.
- Para calcular el total de ventas realizadas por un trabajador:
  - Realizamos una busqueda sobre los campos ‘seller’ y ‘expeditionDate’, los cuales está contenido en la ‘interfaz Invoice’ pero a su vez, importa el dato ‘dni’ (PE), perteneciente a la ‘clase Employe’.
  - Realizaremos la suma aritmética de los importes de la facturas resultantes de la búsqueda, para calcular el volumen de ventas. Y contablizaremos el Nº de facturas para contabilizar a su vez el número de ventas mensuales de cada empleado/a.

Ciclo de vida:

- Para este proyecto hemos elegido el Modelo Iterativo Incremental debido mayormente a la consideración de los siguientes factores y para concretar 'uno', aunque el abordaje más idóneo sería implementar un Modelo Ágil, como PE: la metodología Scrum:
  - La necesidad de adaptarse, reciclarse y diversificar, respondiendo ante la situación completa y constantemente cambiante (junto con sus oportunidades, demandas y otros factores a considerar), presentada por parte del mercado y del mundo actual.
  - La mayor Versatilidad y 'Simplicidad' (de Uso e Implementación) que nos brinda este, en contraposición con sus modelos precursores (M. Cascada y M. Cascada con Realimentación) y consiguientes (M. en Espiral [más complejo]).

2. Planifica la codificación, indicando el lenguaje de programación y las herramientas que usarías para la obtención del código fuente, objeto y ejecutable, explicando por qué eliges esas herramientas.

Planificación:

- Lenguajes de Programación:
  - **Para 'Backend'** (lado Servidor): utilizaremos **Java, .NET, PHP (**y uno' para nuestra Base de Datos como: **MySQL, Oracle DB** u otro similar) **o los servicios de Firebase** (suministrados por Google) indistintamente y según factores como: el tiempo del que dispongamos para su desarrollo hasta la entrega al/a la cliente, el importe del presupuesto aceptado del que partamos, el número de conexiones (cargas y descargas) previstas sobre la aplicación resultante, nuestro nivel de dominio en cada uno de esos lenguajes y el nivel de adecuación y funcionalidades que nos brinden los 'frameworks' y herramientas de cada uno'.

    Para tomar dicha decisión, habrá que realizar un estudio sobre todos los factores implicados y con la menor demora posible. Cade destacar que los tres primeros son muy similares en la mayoría de los aspectos, así que escogeremos el lenguaje en el que tengamos más y mejor nivel de dominio. A su vez, también tendremos en cuenta el tipo y la modalidad de servicio de 'hosting' que nuestro/a cliente esté dispuesto a contratar y mantener, ja que sin considerar este aspecto, nuestra futura aplicación puede acabar resultando totalmente 'inutil' y sin aplicación práctica y en la 'realidad'. PD: Todos los lenguajes y 'frameworks' comentados son libres y gratuitos.
  - **Para 'Frontend'** (lado Usuario/a): utilizaremos **Angular o React** (JS) indistintamente y según se nos adapten sus componentes y prestaciones, a nuestro proyecto. Si el cliente desea una mayor 'libertad' y versatilidad, procederemos a seleccionar y elegir el segundo (React); ya que este sólo utiliza **HTML, CSS y JavaScript** (Puro → No TS), para el desarrollo e implementación de nuestras aplicaciones web. Ambos 'frameworks' de JavaScript son Libres y Gratuitos.

    PD: Si el cliente nos comentara que desea aún más nivel de adaptabilidad (a los diferentes formatos de pantalla) y a su vez, la posibilidad de cambiar los iconos PE (mostrados según el 'tipo dispositivo móvil'), o poder copilar nuestra aplicación, para así convertirla (al finalizar o a posteriori) en una 'aplicación nativa' (PE una 'app' de: Android o IOS), implementaríamos 'React' (o Angular) mediante el 'framework Ionic' (o alguno similar), en su lugar.
- IDE (Entorno de Desarrollo Integrado):
  - Elegimos el IDE **VS Code** suministrado por Microsoft de forma libre y gratuita, en base a su Versatilidad y Variedad (Leng. Prog.) y a las Extensiones disponibles, para aumentar así sus funcionalidades.

- **SiteMap** (Mapa del Sitio) **y Mapa de Interrelación** entre las tablas de nuestra **Base de Datos**:

  La elaboración de estos esquemas nos ayudará y mucho a concretar y lograr planificar mejor, para así poder definir y acotar unos plazos de 'entrega' y desarrollo parciales, que puedan resultar factibles, coherentes y ajustados/ajustándose a la 'realidad' (de nuestro ámbito de aplicación y de la aplicación propiamente), así como también a los posibles imprevistos que nos puedan surgir.

  En la elaboración de los dos esquemas anteriores, se analiza en detalle la interrelación entre vistas (páginas/secciones) que debería comprender la 'Funcionalidad' de nuestra aplicación; esto comprende: *botones, enlaces, redireciones, acciones de respueusta ante 'interrupciones', etc.* A su vez, el segundo 'mapa', nos sirve para analizar en detalle cómo y de qué manera se relacionan/interrelacionan nuestras diferentes Entidades (*PE: Un producto sólo tiene (generalmente) un fabricante, pero un mismo fabricante puede tener asignado más de un producto*)

3. Planifica las restantes fases del ciclo de vida, indicando en cada una el objetivo que persigues y cómo lo harías.

- A partir de este punto, el proceso de desarrollo e implementación de nuestra futura aplicación, se fundamentará, como no podía ser de otra forma, en los pasos anteriores. Para organizarnos en base a nuestro/a análisis y planificación previa haremos uso de varias herramientas, para así lograr 'facilitar' y organizar mejor, en la medida de lo posible, nuestra tarea:
  - Para la gestión y adjudicación de tareas utilizaremos **Trello**, Gira o alguna otra herramienta similar, debido y en base a su 'gran' versatilidad y facilidad de uso; esta y algunas otras opciones son libres y gratuitas también.
  - Para trabajar en equipo y lograr coordinarnos adecuadamente para su consecución, utilizaremos **Github**, Gitlab o alguna otra plataforma similar. Siendo que nuestros clientes muestran predisposición por el 'software' libre, procedemos a seleccionar Github como nuestro medio de trabajo en equipo.

  Mediante el uso y la combinación de los dos recursos anteriores, estaremos en disposición de poder organizar todo nuestro trabajo y así poder finalizarlo para y entregarlo al/a la cliente/a en la fecha acordada previamente (durante la elaboración y la aceptación del presupuesto expedido), pero previamente a finalizarlo y resultando la última fase o una de las 'últimas' de nuestro proceso de desarrollo, procederemos a 'testear' y poner a prueba nuestra aplicación resultante, antes de entregarla al/a la cliente/a. Para ello, implementaremos un conjunto de **Pruebas Unitarias**, mediante la herramienta (disponible/adecuada para los lenguajes escogidos finalmente) para realizar tal fin (*PE: para Angular, podemos utilizar Jasmine; y para .NET i para la verificación de la conexión con la base de datos, podemos utilizar Swagger y/o Postman)*. PD: Se a procedido a escoger la opción de "Pruebas Unitarias" meramente en base a dos factores: el mayor nivel de experiencia manifestado en su realización, por mínimo que resulte, y la mayor modularidad y focalización a la hora de poner a prueba cierta/s función/es, en específico.

  Todas las fases de la planificación adjunta se encuentran focalizados en el mismo objetivo: conseguir desarrollar, comprobar y entregar una aplicación completamente funcional a nuestro/a cliente final.

ED01

Criterios de puntuación. Total 10 puntos.

Planificación correcta de análisis de requisitos y elección de modelo de ciclo de vida: 3 puntos.

Planificación correcta de las herramientas de programación en base a los requerimientos del cliente y secuencia de pasos a seguir: 2 puntos.

Planificación correcta de pruebas, documentación, explotación y mantenimiento: 3 puntos.

Uso de vocabulario específico, redacción clara, ideas fundamentales y orden en la secuencia de pasos: 2 puntos.

Recursos necesarios para realizar la Tarea.

Ordenador con procesador de textos.

Conexión a Internet.

Consejos y recomendaciones.

Se pretende poner en práctica los conceptos aprendidos, de la forma más clara posible.

Para ello, se recomienda no extenderse mucho en las respuestas y dar prioridad en el orden de secuencia de pasos indicando siempre el objetivo que se persigue con cada uno de ellos.

Indicaciones de entrega.

Una vez realizada la tarea elaborarás un único documento donde figuren las respuestas correspondientes. El envío se realizará a través de la plataforma de la forma establecida para ello, y el archivo se nombrará siguiendo las siguientes pautas:

apellido1_apellido2_nombre_SIGxx_Tarea

Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo la alumna Begoña Sánchez Mañas para la primera unidad del MP de ED, debería nombrar esta tarea como...

sanchez_manas_begona_ED01_Tarea

Kevin Zamora Amela

ED01

Kevin Zamora Amela

ED01

Kevin Zamora Amela