**ÍNDICE DE LA MEMORIA QUE HA DE ENTREGAR EL ALUMNO**

**1. Portada.** Datos del proyecto: Título, Ciclo, Curso, Fecha, Centro, Logo del centro. Datos del alumno y datos de los profesores tutores

**2.** **Documento Descripción del proyecto**

2.1. Contexto del proyecto

2.1.1. Ámbito y entorno

2.1.2. Análisis de la realidad.

2.1.3. Solución y justificación de la solución propuesta

2.1.4. Destinatarios.

2.2. Objetivo del proyecto

**3. Documento de Acuerdo del proyecto.**

3.1. Requisitos funcionales y no funcionales.

3.2. Tareas

3.3. Metodología a seguir para la realización del proyecto.

3.4. Planificación temporal de tareas.

3.5. Presupuesto (gastos, ingresos, beneficio)

3.6. Contrato/Pliego de condiciones

**4. Documento de análisis y diseño**

4.1 Modelado de datos. Análisis y diseño de la base de datos

4.1.1 Diagrama E/R

4.1.2 Diagrama relacional

4.2 Análisis y diseño del sistema funcional (Diagramas que procedan como el de casos de uso, de clases, de secuencia, de actividad, de estados..)

4.3 Análisis y diseño de la interfaz de usuario. Mockups.

4.4 Diseño de la arquitectura de la aplicación

4.4.1 Tecnologías/Herramientas usadas y descripción de las mismas

4.4.2 Arquitectura de componentes de la aplicación

**5. Documento de implementación e implantación del sistema**

5.1 Pruebas

5.1.1 Pruebas unitarias

5.1.2 Pruebas funcionales

5.2. Instalación y configuración.

5.3. Manual de usuario

**6. Documento de cierre**

6.1. Resultados obtenidos y conclusiones.

6.2. Diario de bitácora

**7. Bibliografía.**

**8. Anexos.**

**DESARROLLO DEL PROYECTO**

**DOCUMENTO DE PROPUESTA DE PROYECTO**

**TíTULO DEL PROYECTO**

**TíTULO CORTO DEL PROYECTO**

**CONTEXTO BREVE**

**OBJETIVO BREVE**

**DOCUMENTO DE DESCRIPCIÓN DEL PROYECTO**

En este apartado realizaremos una descripción detallada del proyecto

**CONTEXTO**

En este apartado se puede incluir cuatro subapartados:

* Ámbito del proyecto

Describe la empresa/cliente para la cual realizas el proyecto, departamentos, situación a qué se dedican... Si es un proyecto que no va dirigido a un cliente en concreto y es por motivación propia explícalo y di qué te ha motivado para realizar el proyecto.

* Análisis de la realidad

Realiza también un **análisis de la situación actual** del sistema y **necesidades** encontradas, críticas al sistema actual si lo hubiese…Deberás especificar si el proyecto va a ser una actualización/ampliación/adaptación de un proyecto ya desarrollado o bien partes de cero. Si es una actualización deberás dejar claro qué es lo que hay hasta ese momento, si vas a emplear componentes, diseños o herramientas ya disponibles…

* Solución y justificación de la solución propuesta.

Soluciones que quieres aportar y justificación de la solución elegida.

* Destinatarios.

Usuarios destinatarios del proyecto. Puedes hablar del rol que tendrán cada grupo de usuarios.

**OBJETIVO**

Descripción general de lo que pretende ser tu aplicación. No hace falta detallar demasiado la funcionalidad ya que luego se hará en la lista de requisitos.

Puede empezar: “El objetivo del proyecto es desarrollar una aplicación web…”

**DOCUMENTO DE ACUERDO DEL PROYECTO**

**LISTA DE REQUISITOS FUNCIONALES Y NO FUNCIONALES**

Tendrá formato tabla, y cada requisito identificado por las siglas RF (requisito funcional) O RNF (requisito no funcional) seguido de un número identificativo y una descripción del requisito. Al finalizar el proyecto se realizarán las pruebas del sistema y se validará si cada requisito se ha cumplido.

REQUERIMIENTOS FUNCIONALES

Los requisitos funcionales establecen los comportamientos del sistema. Un [requisito](https://es.wikipedia.org/wiki/Ingeniería_de_requisitos) funcional define una función del sistema de [software](https://es.wikipedia.org/wiki/Software) o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas.

Ejemplo  
  
RF-1 El sistema debe permitir agregar clientes. Dado el nombre, la edad, la dirección y el teléfono del cliente se dará de alta el cliente en la base de datos con un identificador único.  
  
RF-2 El sistema debe permitir modificar clientes. Dado el id de cliente, se permitirá la modificación de cualquier campo.  
  
....

REQUERIMIENTOS NO FUNCIONALES

Un requisito no funcional es un [requisito](https://es.wikipedia.org/wiki/Requisito_(sistemas)) que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los [requisitos funcionales](https://es.wikipedia.org/wiki/Requisito_funcional). Por tanto, se refieren a todos los requisitos que no describen información a guardar, ni funciones a realizar, sino características de funcionamiento.

Algunos ejemplos de requisitos no funcionales típicos son los siguientes:

* rendimiento
* disponibilidad
* accesibilidad
* usabilidad en distintos navegadores
* estabilidad
* portabilidad
* costo
* operatividad
* interoperabilidad
* escalabilidad
* concurrencia
* mantenibilidad
* interfaz
* seguridad
* posicionamiento
* codificación
* formatos

Ejemplo:

RNF-1 El sistema debe ser de fácil navegabilidad.  
  
RNF-2 El sistema debe ser seguro y mantener la integridad de la información.  
  
RNF-3 El sistema debe acceder de forma correcta y rápida a la información.  
  
RNF-4 El sistema debe soportar tanto plataformas Windows como Unix.  
  
RNF-5 El sistema debe mantener copias de respaldo para evitar pérdidas de información.

**DEFINICIÓN DE TAREAS**

En su núcleo, un proyecto es un conjunto de tareas, cada una de las cuales representa una pieza de trabajo que debe realizarse para completar el proyecto de forma satisfactoria. En esta fase se definen las tareas a realizar en el proyecto

Ejemplo

     1) Elección proyecto a realizar. Investigar sobre contexto y objetivo principal del proyecto.

     2) Entrevistas con cliente para recogida de requisitos funcionales y no funcionales. Realización de diagramas de casos de uso.

3) Descripción de tareas y planificación temporal de las mismas.

     3) Realización del presupuesto y acuerdo del proyecto.

     4) Ánalisis del proyecto. Realización de diagramas.

     5) Diseño del proyecto. Realización de diagramas.

     6) Implementación del proyecto

     7) Pruebas de la aplicación

**METODOLOGÍA**

Metodología que vas a seguir en el proyecto. Una vez que ya tienes las tareas hay que decidir qué metodología vamos a aplicar para realizar las tareas.

Puedes adaptarte a los hitos establecidos por el tutor, que aconseja un modelo de ciclo de vida iterativo incremental aplicándolo con una metodología ágil en la que se establecen dos sprints, es decir, dos entregas al cliente en las fechas X e Y.

Dentro de cada sprint se desarrollan las diferentes fases del ciclo de vida de desarrollo del software desde el establecimiento de requisitos pasando por el análisis y el diseño hasta la implementación y pruebas de la aplicación.

**PLANIFICACIÓN TEMPORAL DE TAREAS**

La organización de tareas en hitos y fases proporciona cierta estructura al proyecto y lo hace más fácil de evaluar en cuanto a su progreso.

Realizaremos la planificación temporal del primer sprint y del segundo sprint de acuerdo al calendario establecido de las distintas tareas (ver fechas entrega para el curso actual). Se deberán marcar dichos hitos en vuestro diagrama. Además de los hitos establecidos por el cliente (o en su defecto tutor de prácticas) habrá que hacer una planificación temporal de las tareas a realizar en cada Sprint. A cada tarea habrá que asignar una temporalización.  Muy útil realizar un diagrama de Gannt

     Por ejemplo en el primer Sprint podríamos señalar las **siguientes tareas**:

     1) Elección proyecto a realizar. Investigar sobre contexto y objetivo principal del proyecto.

     2) Recogida de requisitos funcionales y no funcionales. División de requisitos en 2 sprints

     3) Presupuesto y acuerdo del proyecto

     4) Ánalisis del primer sprint del proyecto

     5) Diseño del primer sprint del proyecto

     6) Implementación del primer sprint del proyecto

     7) Pruebas

     8) Ánalisis del segundo sprint del proyecto

     5) Diseño del segundo sprint del proyecto

     6) Implementación del segundo sprint del proyecto

     7) Pruebas

Cada una de estas tareas tendrán subtareas. Por ejemplo la implementación podría dividirse en fases según el proyecto elegido por ejemplo  Implementacón alta usuarios, implementación modificación usuarios... La fase de diseño tendrá los mockups, diseño bbdd....El nivel de detalle depende del alumno.

**PRESUPUESTO**

Creamos el presupuesto (tal y como aprendisteis en EIE).

**CONTRATO**

Creación y firma del contrato del proyecto con el cliente (según modelo colgado en la plataforma u otro modelo). Se puede añadir un apartado de pliego de condiciones si se desea.

**DOCUMENTO DE ANÁLISIS Y DISEÑO DEL SISTEMA**

El objetivo fundamental de este apartado consiste en la representación explicita de todas las funciones y subfunciones de que consta el sistema que se vaya a desarrollar así como la estructura de las relaciones entre las mismas y la descripción de los conjuntos de datos del sistema, etc.

En este apartado se realizará la definición del sistema desde el punto de vista lógico o conceptual exclusivamente prescindiéndose de los aspectos tecnológicos.

Debido a la gran variedad de proyectos que se pueden realizar, la descripción dependerá del tipo de proyecto. Aun así cuando sea aplicable se deberá incluir información sobre:

* Diagrama modular de la aplicación. Para esto se utilizará la nomenclatura de alguna metodología de desarrollo estandarizada y actual como UML.
* Unidades de programación utilizadas (funciones, procedimientos, scripts, etc.)
* Diagrama entidad/relación o diagrama de clases a utilizar en el diseño de la base de datos. .
* Todas las aplicaciones de acceso a datos deben contemplar el control de seguridad.
* Distribución de la aplicación con proceso de instalación.

Para desarrollar los apartados anteriores se pueden tener en cuenta la realización de los siguientes puntos

* Análisis de los casos de uso. Diagrama de casos de uso para mostrar de forma gráfica los requisitos y los actores.
* Modelado de datos. Análisis y diseño de la base de datos: Diagrama E/R de base de datos (opcional pero muy recomendable) y diagrama relacional de bd, podéis usar Workbench mysql.
* Diseño de la interfaz de usuario: Mockups de la aplicación se puede utilizar Balsamic Mockup o cualquier otro con el que se haya trabajado.
* Diseño del sistema. Podéis meter todos los diagramas UML que queráis diagrama clases si oo, diagrama flujo datos, eventos, actividades, secuencia...) para el diseño del código de la aplicación.
* Diseño de la arquitectura de la aplicación
  + - Tecnologías/Herramientas usadas y descripción de las mismas. Se puede añadir una justificación de por qué has elegido esas tecnologías y/o herramientas.
    - Arquitectura de componentes de la aplicación. Se detallará mediante esquemas, dibujos o texto los componentes lógicos y físicos de tu aplicación. Es decir si usas un framework, o un modelo de datos en concreto tendrás una organización física del código fuente por carpetas y o paquetes impuesta por dicho framework/modelo. Por ejemplo en el paquete dao las clases que acceden a bd, en el paquete html los htmls, en el paquete js...Hablar del MVC...

**DOCUMENTO DE IMPLEMENTACIÓN E IMPLANTACIÓN DEL SISTEMA**

**IMPLEMENTACIÓN DEL SISTEMA**

(Opcional) En este apartado se incluirán para cada uno de los módulos del sistema desarrollado el código fuente. Se deberá incluir inicialmente un listado organizado de los ficheros con código fuente acompañados de una breve descripción de lo que contienen.

**PRUEBAS**

* **PRUEBAS UNITARIAS**

Se aconseja la realización y utilización de pruebas unitarias para ir verificando los diferentes módulos a medida que se van desarrollando. Aunque no es necesario su documentación

* **PRUEBAS FUNCIONALES**

El objetivo de este apartado es demostrar que el sistema globalmente cumple con las especificaciones definidas en los apartados anteriores.

Para ello será obligatorio documentar las pruebas funcionales de la aplicación. Se denominan pruebas funcionales o Functional Testing (pruebas de caja negra), a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con los requisitos para los cuales han sido creados. El enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida.

Para la documentación de esta parte se puede seguir el siguiente formato

|  |  |
| --- | --- |
| **Requisito RF1** | |
| **Precondición** |  |
| **Datos de entrada** |  |
| **Datos de salida** |  |

|  |  |
| --- | --- |
| **Requisito RF5 El sistema permite editar un usuario** | |
| **Precondición** | Desde el listado de usuarios se pulsa el botón edición Usuario de la tabla usuarios |
| **Datos de entrada** | Se modifica los valores en los campos nombre, email, password y repetición de password y se selecciona rol |
| **Datos de salida** | 1. La pantalla debe tener rellenados los campos con los datos del usuario seleccionado. OK 2. Si ya existe un usuario con ese email se le indica en un mensaje de error. OK 3. Si no rellena los campos obligatorios de nombre, email y contraseña, se le indica en mensaje de error. OK 4. Si las contraseñas no coinciden se le indica en un mensaje de error. OK 5. En caso de que la validación sea correcta, se modifica el registro en la tabla Users y se cierra la ventana de edición de usuario indicándole que el usuario se ha modificado correctamente. OK |

**INSTALACIÓN Y CONFIGURACIÓN**

Siempre que sea posible o aplicable se entregará el proyecto con utilidades de instalación que faciliten la implantación del sistema en un servidor.

Hacer referencia a vuestro repositorio git donde esté el código fuente.

**MANUAL DE USUARIO Y/O AYUDA INTEGRADA EN LA APLICACIÓN**

Se deberá incluir un manual de usuario de la aplicación y/o desarrollar una ayuda integrada en dicha aplicación. En éste último caso, indicar que se ha desarrollado y cómo acceder a ella.

En este apartado se incluirán los subapartados que el alumno considere necesarios para que un potencial usuario de su proyecto pueda adquirir con facilidad las nociones necesarias para su utilización. Se recomienda incluir entre otros los siguientes apartados:

* Descripción general del sistema
* Puesta en marcha
* Manejo de los menús o acceso a cada módulo
* Gestión periódica del sistema
* Realización de copias de seguridad
* Mensajes de error
* Glosario de términos

**DOCUMENTO DE CIERRE**

**RESULTADOS OBTENIDOS Y CONCLUSIONES**

El cierre de un proyecto es la culminación del proceso proyectual, y el momento

de hacer balance del mismo. Durante el cierre se advierte cómo de bien o de mal se ha terminado y, en especial, si se han alcanzado los objetivos previstos.

Objetivos principales

* Diagnosticar el funcionamiento, tratando de analizar las desviaciones entre las previsiones iniciales y el resultado
* Analizar si se han alcanzado la funcionalidad prevista.
* Analizar la desviación en cuanto a la planificación en horas del proyecto
* Conocimientos adquiridos.
* Posibilidades de mejora.

**DIARIO DE BITÁCORA**

Recogerá todo el seguimiento del proyecto. Cada semana las tareas realizadas. Se podrá considerar cuaderno de bitáctora los commits de github.

**PAUTAS PARA EL DESARROLLO DEL PROYECTO**

Uso obligatorio de un sistema de gestión de versiones tipo GIT O BITBUCKET

Uso obligatorio de un formato común para todos los apartados

**PAUTAS PARA LA ENTREGA DEL PROYECTO**

Es necesario puntualizar que una memoria sin un estilo definido, mal estructurada, mal redactada, con faltas de ortografía, y sin un índice de contenidos, bajará la nota, así que es importante cuidar la presentación de la misma. Podéis utilizar el documento de formato recomendado por el profesor o crear vosotros uno (Ver documentoFormato)

**Portada de la memoria**

- Título.

- Datos identificativos.

- Logo del Centro Público Integrado de F.P. “Los Enlaces”

**Tamaño del papel, márgenes y estilo** (recomendado)

- Tamaño A4.

- Interlineado simple, si es posible (dejando las correspondientes separaciones de párrafo, mínimo 1,5 espacios).

- Margen del lado del encuadernado no menos de 3cm. Lado contrario, no menos de 1 cm

**Tipo de letra,** etc., se deja libre a la creatividad. Se recomienda cualquiera de la familia Serif (la letra Times New Roman, vale), pues gastan menos tinta o tóner y se leen mejor en el caso de tener que imprimir.

- Tamaño recomendable (pero no obligatorio) de 12 para los párrafos.

- Justificado a izquierda y derecha.

**Índice**

- Paginado al principio.

- Atenerse al guión oficial.

IMPORTANTE: Aunque todos los documentos (Documento Descripción del proyecto, Documento Análisis...etc) se entregan conjuntamente en la memoria, cada documento irá separado del resto por una hoja de portada que contenga únicamente el título del documento al que hace referencia.

**Modo de entrega**

- Entrega de la documentación del proyecto en secretaría:

El alumno se quedará con una copia de la portada sellada como registro del depósito. La entrega se realizará normalmente una semana antes de la exposición y defensa de los proyectos. Se concretará la fecha más adelante.

Se deberá entregar una copia de la memoria tanto en papel encuadernada como en soporte digital. Además la copia en digital deberá también contener el código fuente de la aplicación.

Para los alumnos de ERASMUS y el que está fuera de la comunidad autónoma, lo enviarán al tutor en digital siempre y cuando estén en el país de erasmus el día de la entrega. Deberán también presentar la memoria encuadernada y el soporte digital del proyecto antes de presentar el proyecto al tutor.

**Exposición y defensa de los proyectos:**

- Tiempo máximo para la exposición: 20 minutos.

- Aconsejable 12 transparencias (legibles), para reproducir con los medios utilizados en las aulas y de los que el alumnado es conocedor.

- Después de la exposición por parte del alumno, el equipo docente dispondrá de 10 minutos para hacer las preguntas que considere oportunas.

- Cada 25-30 minutos entra una persona.

- La sesión es pública.