Universidad Zaragoza

Trabajo Fin de Grado

Una wiki para regiones delimitadas sobre una malla global discreta

Autor

Diego Raúl Roldán Urueña

Director

Rubén Béjar Hernández

Grado en Ingeniería Informática

Escuela de Ingeniería y Arquitectura

Julio 2025

I. Resumen

Obligatorio según la normativa (revisar). Puede ser algo como Introducción + Conclusiones condensadas en, pongamos, 300 palabras.

II. Tabla de contenidos o Índice

III. Introducción

Uno o dos párrafos de introducción al trabajo: brevemente qué se quería hacer, y cuáles han sido los principales resultados alcanzados.

El resto del documento se estructura como sigue: explicar la estructura del documento. Puede ser buena idea señalar que se ha cumplido explícitamente con la recomendación de 20 páginas/10.000 palabras, o explicar explícitamente por qué no se ha hecho.
Este documento recoge tanto documentación como

1. Contexto

Poner en contexto este TFG: ¿cómo encaja en el Mundo? ¿Es algo innovador o hay cosas similares en el mercado? Si hay cosas similares, ¿cómo se compara esta TFG con ellas? ¿Se hace en una empresa, en un grupo de investigación o es un proyecto independiente? ¿Se plantea como un primer paso hacia algo mayor o como un trabajo por sí mismo? ¿O es una mejora de algo existente? ¿Es una parte de algo que se ha hecho trabajando con más gente o es el resultado de un trabajo individual? Si es parte de un producto/sistema más grande, o una mejora o nueva versión, habrá que esforzarse en la memoria, y luego en la presentación, por trazar límites claros entre lo que había y lo que se ha hecho.

2. Fuentes de datos integrados / mostrados / procesados [elegir lo que sea]

Si es un proyecto que integra, muestra, procesa… datos, hablar de las fuentes de los mismos. De dónde salen, qué licencias tienen, cuánto ha habido que procesarlos… Si el proceso de los datos ha sido una parte importante del trabajo, habrá que contarlo en detalle en la sección 3. Si el proyecto usa datos pero el procesado no se ha hecho como parte del trabajo, entonces basta con contarlo aquí.

IV. Análisis del problema

Explicar brevemente, de manera general, lo que se va a hacer.
Se va a desarrollar una aplicación web con estilo wiki en la que cada articulo sea identificado geográficamente.

1. Requisitos / Historias de Usuario / Casos de Uso [elegir lo que sea]

Se pueden combinar requisitos y casos de uso. No se suelen hacer otras combinaciones (p.ej. no tiene sentido tener requisitos e historias de usuario, dado que ambas cosas sirven para lo mismo). Los requisitos pueden dividirse en funcionales y no funcionales, pero esta distinción es a veces un poco arbitraria. Si hay requisitos estrictos de prestaciones o de seguridad, generalmente serán no funcionales, describirlos lo más objetiva y cuantitativamente posible.

Titulo	Descripción	Prioridad
Crear página	Crear una nueva página de la wiki requiere seleccionar una o más celdas de un DGG sobre un mapa para establecer el lugar al que corresponde esa página. Una vez determinado el lugar, se le asigna automáticamente un id único (Area Unique Identifier, AUID) y un hash con el que se construye su PURL (Permanent URL). Una vez creada una página, el conjunto de celdas que la definen no se puede cambiar.	M
Contenido de una página	Título. Descripción. Palabras clave (pueden ser categorías o tags). Enlaces (a otras páginas de la locatopedia, o a otras páginas web en general). Fecha de creación. Periodo de tiempo en el que la entrada es “válida” (se pueden definir cosas que solo tienen interés temporalmente). Media (imágenes, vídeos, ficheros adjuntos…). Etc.	M
Borrar páginas	En principio borrar una página no debería ser una acción común. Si alguien la enlazaba, ese enlace se rompe, lo que nunca es bueno en la Web. Podemos marcar páginas como “obsoletas”, quizás apuntando a una “versión mejorada” de la página o algo así, si nos hemos equivocado al crearla. El único que debería poder borrar páginas sería un usuario administrador.	S
Editar página	Las páginas deberían guardar el historial de cambios de las mismas, no solo la última versión. También se debería poder tener en cada página algún tipo de chat que permita tener discusiones sobre la misma (y que estas discusiones queden también guardadas). Esto es como funciona p.ej. la Wikipedia.	M
Buscador de texto	Permite buscar palabras o frases en función de los títulos, subtítulos, descripciones de páginas (palabras clave etc.)	S
Buscador espacial	Permite seleccionar en el mapa una región (¿?rectangular (AKA “bounding box”) y muestra los artículos que	S

	contengan celdas dentro de esa región. El buscador puede usarse como filtro adicional sobre la consulta basada en texto (p.ej., “páginas que contengan `Zaragoza` pero que estén localizadas en el bounding box que corresponde a Sudamérica”).	
Gestión de usuarios	Un usuario debe haber iniciado sesión para editar una página. ¿Iniciar sesión con Google? Debemos distinguir usuarios administradores y otros.	M
Página Home	Muestra las páginas más destacadas, páginas que hablen sobre tu ubicación actual, últimas actualizaciones, recurso del día...	M
Crear página en función de una existente	Permite crear una página nueva a partir de una página ya existente. La idea es facilitar la creación de localizaciones “alternativas” o parecidas modificando el conjunto de celdas original. Se debe facilitar que las páginas derivadas de otras automáticamente estén enlazadas con estas, posiblemente asignando una semántica a esta relación (p.ej. “modificación”, “ampliación”, ...).	S
Links a otras páginas	Permite añadir links para hacer referencia a otras páginas de Wikiplace.	M

Tabla 1. Primeras ideas

2. Interfaces de usuario

Para productos con GUI, orientados principalmente a humanos. Desde el punto de vista de la captura de requisitos: borradores, bocetos, navegación entre pantallas etc. No la GUI tal y como queda finalmente, que se contaría en la sección 3.

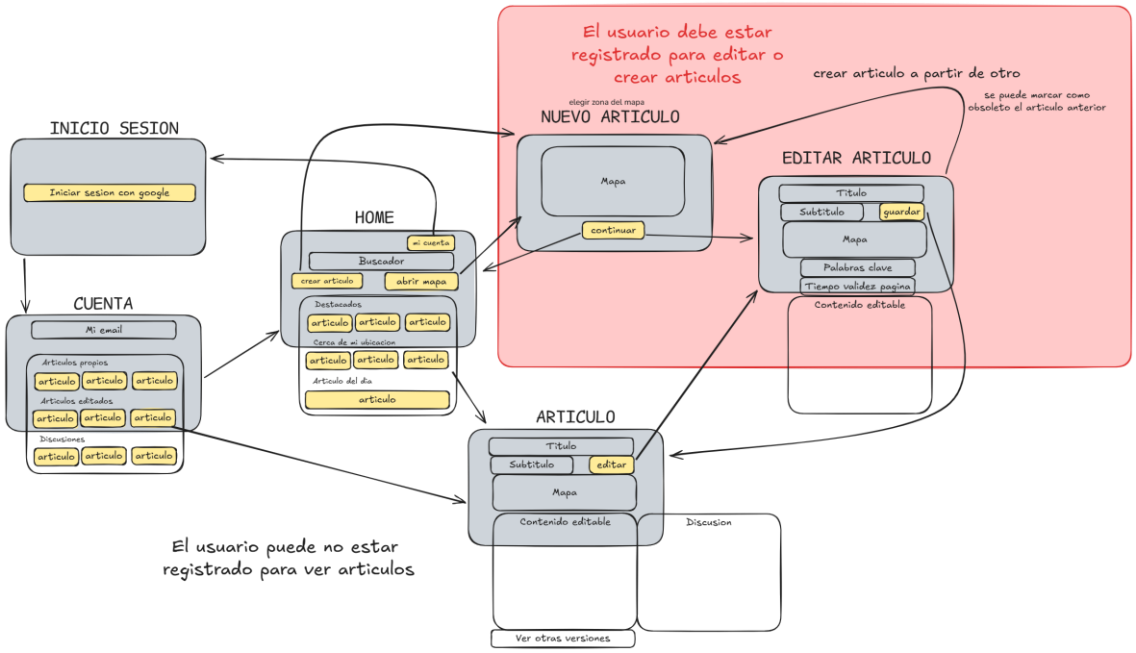

Figura 1. Primera versión del mapa de navegación de la aplicación.

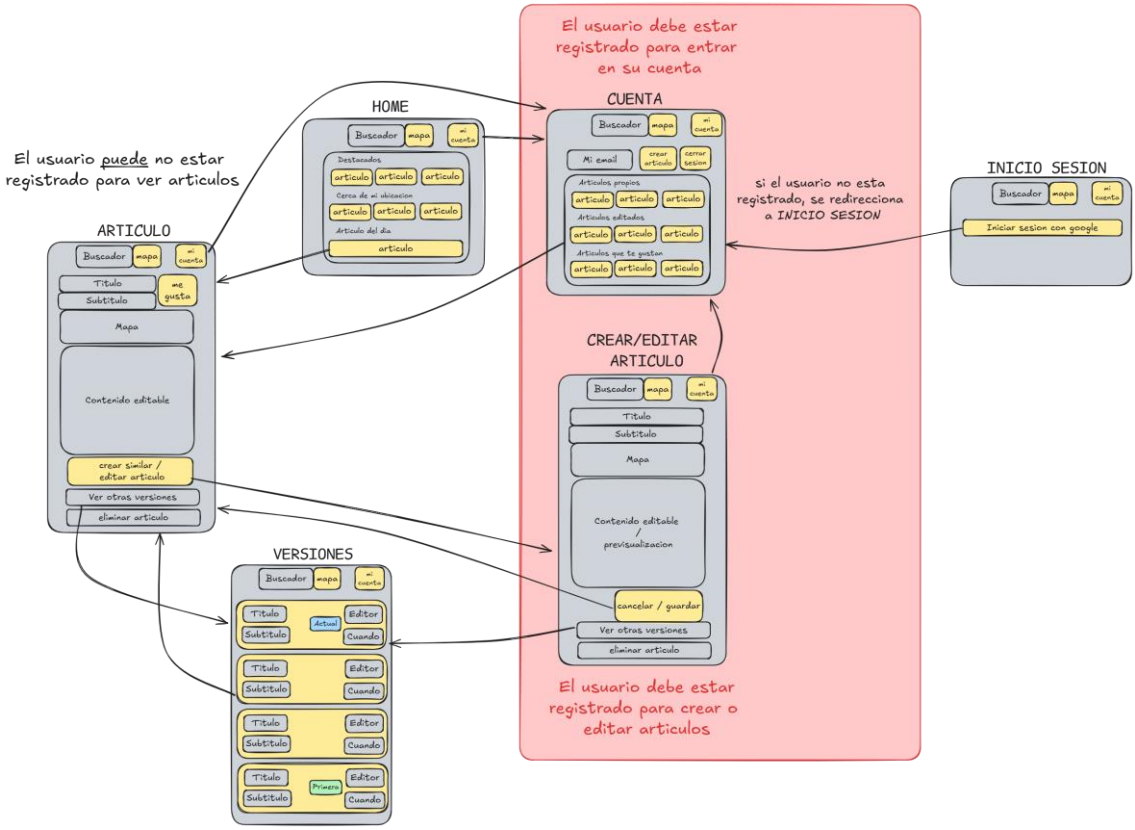

Figura 2. Versión final del mapa de navegación de la aplicación.

3. Interfaces para la integración en aplicaciones [este título de sección seguramente haya que adaptarlo a lo que se haya hecho]

Para productos sin GUI, o con una GUI de administración o similar pero cuyo principal valor está en proporcionar servicios a otros sistemas software (p.ej. ofreciendo una API). Una API no significa "una API REST". Una API pueden ser las clases y operaciones públicas principales de una biblioteca de software, un conjunto de scripts de procesamiento de datos… En cualquier caso aquí se describe esa API desde el punto de vista de la captura de requisitos: qué se quiere ofrecer a otros sistemas. El resultado final se describiría en la sección 3.

V. Diseño de la solución

1. Arquitectura

Describir la idea general: estilo arquitectural principal del sistema, frameworks utilizados y qué implicaciones arquitecturales tienen. Si hay consideraciones importantes de prestaciones o de seguridad que ha habido que tener en cuenta.

1. Contexto

Esta sección seguramente solo tendrá sentido para sistemas que son parte de otros mayores, para dejar claro qué parte es lo del TFG, y qué parte es lo que ya había.

2. Modelo de datos

Un modelo de entidades y relaciones (diagrama de clases UML o, si no hay otro remedio, notación prehistórica E-R). Más las explicaciones fundamentales: qué es cada entidad, qué es cada atributo, qué implican las relaciones.

Se puede complementar con un modelo de implementación (modelo relacional, documentos JSON en MongoDB o lo que sea), pero lo más crítico es el modelo de entidades y relaciones.

3. Paquetes y clases

- Un diagrama UML con los paquetes principales. Tanto del front-end como del back-end, si hay de ambos. Sin mucho detalle, es una idea general.
- Un diagrama UML de clases con las clases principales. De nuevo tanto de front-end como de back-end si tiene sentido. Centrarse en las clases principales, en sus relaciones y en sus operaciones principales.

Y, por supuesto, todas las explicaciones pertinentes que permitan entender esos diagramas: descripción de los paquetes, papel de las clases principales en el sistema, patrones de diseño y estilos arquitecturales empleados, principales decisiones tomadas (decisiones implica que había varias opciones sensatas y se ha elegido una de ellas basándose en criterios racionales),

4. Componentes y conectores

- Un diagrama UML con los componentes principales y sus conectores. Dependiendo del tipo de sistema esos componentes serán objetos, o serán procesos que agruparán a muchos objetos, o serán servicios web…
- Algún diagrama UML (secuencia, interacción, máquinas de estados…) ilustrando el comportamiento dinámico del sistema suele ser interesante, especialmente si este comportamiento, o alguna parte del mismo, tiene cierta complejidad.

Y, por supuesto, todas las explicaciones pertinentes que permitan entender esos diagramas: qué es cada componente, qué permite cada conector, documentación de las interfaces (puertos/API) más relevantes del sistema, aspectos de concurrencia/asincronicidad/paralelismo/operaciones periódicas, estilos y

patrones arquitecturales y de diseño que tiene más sentido explicar aquí que en la parte de módulos, decisiones de diseño relevantes en esta parte etc.

Relacionar componentes con módulos es interesante (cada línea de código que hemos puesto en un módulo debería ejecutarse en algún componente), sobre todo si ayuda a clarificar algún aspecto complejo.

5. Distribución

- Un diagrama UML del despliegue (artefactos en nodos interconectados) y/o de la instalación (artefactos sobre el sistema de ficheros).

Y, por supuesto, todas las explicaciones pertinentes que permitan entender esos diagramas: qué es cada nodo, qué tipo de rutas los interconectan, cómo se relacionan los artefactos de estos diagramas con los componentes de la sección anterior… Todo esto es especialmente interesante si hay un despliegue automatizado, cosas en Cloud, en contenedores (Docker directamente, o se usa Kubernetes o similar…) y habría que explicar todo eso en esta sección.

2. Implementación

Si esto es lo bastante grande, puede tenerse un capítulo entero (sería el 4) para la Implementación.

Aspectos interesantes de la implementación que sean de bajo nivel de abstracción, o estén localizados en módulos/componentes particulares y que no se hayan considerado lo bastante interesantes para el punto de arquitectura se pueden poner aquí.

Aquí se puede describir la GUI que ha quedado al final del sistema, e incluso poner un diagrama de navegación definitivo si tiene sentido.

Si la construcción se ha automatizado (TravisCI, Github Actions o similares), describir aquí el pipeline de construcción. Describir el uso de sistemas de control de versiones (cuántos repositorios, qué workflows de Git etc.).

3. Procesamiento de datos

Si el proyecto ha tenido una parte importante de procesamiento de datos, explicarlo aquí. Si no, es probable que esta sección sobre.

4. Pruebas

Se incluyen tests automáticos y tests manuales para validación y verificación, pero también tests de prestaciones o tests de estrés (sobrecarga) si el tipo de proyecto los requiere. Herramientas usadas, planificación de las pruebas, guiones de las pruebas manuales, integración de las pruebas automáticas en el pipeline de construcción. Si se ha usado Sonar o algún otro analizador estático de código explicarlo aquí.

Cualquier tipo de proyecto va a requerir pruebas: cómo se validan los resultados es crítico se haga lo que se haga.

VI. Gestión del proyecto

Señalar la metodología de gestión que se ha seguido. Dependiendo de la misma incluir planes, división del trabajo en iteraciones/sprints, horas de esfuerzo realizadas, análisis de riesgos realizado, etc.

VII. Plan de producto / plan de negocio / análisis de mercado

Opcional. En la mayoría de los TFG no vamos a tener esto, pero si el TFG tiene orientación a mercado (por ejemplo, es algo que se ha hecho con la intención de comercializarlo, o de llegar a comercializar una versión futura), cabe perfectamente tener un capítulo sobre esto en la memoria.

VIII. Conclusiones

Resumir los resultados/aportaciones principales del proyecto. Dar ideas de por dónde podría seguir el trabajo. Se puede incluir una valoración personal del proyecto y una descripción de las incidencias y problemas encontrados y resueltos durante el mismo.

IX. Bibliografía

Las citas bibliográficas que se hayan hecho en el texto. Numeradas consecutivamente según se indica en las recomendaciones de la EINA.

X. Anexos

Manual de administración si tiene sentido.
Manual de despliegue/instalación si tiene sentido.
Manual de usuario si tiene sentido.
Diagramas detallados que en el cuerpo principal de la memoria eran demasiado.
Resultados detallados de pruebas que en el cuerpo principal de la memoria eran demasiado.
Documentación completa de la/s API más relevantes.
Etc.