Actividad preliminar. Retomando mi primer sprint

José Ma. Gutiérrez, Adrián González, Luis Lira

Licenciatura en Desarrollo de Sistemas Web

Código: 211762638

Proyecto II

Product backlog:

Identificador (ID) de la Historia	Enunciado de la Historia	Alias	Estado	Dimensión/ Esfuerzo	Interacción (Sprint)	Prioridad	Comentarios
D01	Definición de tecnologías a usar para el proyecto (Backend, Frontend, BDD)	Definir tecnologías	Por iniciar	2	1	1	Se buscará información sobre las tecnologías que permitan la creación ágil de la API. El framework elegido deberá contar con un estándar de seguridad y las paqueterías del mismo deberán estar libres de vulnerabilidades.
D02	Diseño de estructura de datos para mockups y BDD	Diseño de datos	Por iniciar	2	1	1	Se definirá de qué manera se organizará el formato JSON para la información de cada elemento de la BDD. En esta estructura se definirán los nombres de los campos que contendrá el archivo y el tipo de valor que almacenará.
D03	Crear el servidor con el framework de backend seleccionado	Creación de servidor	Por iniciar	2	1	1	Se levantará en local el servidor y se probará que funcione correctamente. Las pruebas se harán en local con un cliente de POSTMAN, navegador y luego se creará un enlace público con nGrok para probar desde un cliente remoto.

D04	Creación de archivo JSON (Mockup) para simular consultas HTTP	Creación Mockup base	Por iniciar	1	1	1	Se creará un archivo JSON con el formato definido de datos para trabajar durante el desarrollo. Este archivo se usará sólo como una base de la información
D06	Recopilar información de los campeones en el sitio oficial	Recopilación de datos	Por iniciar	5	1	1	Se creará un archivo con un Arreglo de objetos en JSON para recopilar la información. La información a recopilar será de los 140 campeones existentes en el juego. La recopilación será hecha por Luis Lira (Director del proyecto), en un archivo externo llamado, campeones.json, el cuál será un Array con todos los objetos JSON de cada uno de los campeones.
HU01	Obtener los datos de todos los campeones por medio de HTTP GET	Hacer Get All	Por iniciar	1	1	1	Los usuarios podrán obtener toda la información de la Tabla/Colección campeones. Se podrán usar filtros por las etiquetas de región y rol en el juego. Se podrá poner un límite de resultados devueltos.
HU02	Obtener los datos de un solo campeón	Hacer Get One	Por iniciar	1	1	1	Los usuarios podrán obtener sólo los datos de un solo campeón de la Tabla/Colección campeones

D05	Creación de Base de datos en el servidor local	Creación de instancia de BDD	Por iniciar	1	1	2	Se creará la instancia de la base de datos, así como determinar la cadena de conexión a la misma. La cadena de conexión deberá ser confidencial y se accederá a ella a través de variables de entorno.
HU03	Simular petición HTTP Put hacia la API	Simular actualización	Por iniciar	1	2	2	Se simulará el tiempo de respuesta de una petición HTTP Put con un delay de 1 segundos.
HU04	Simular petición HTTP Post hacia la API	Simular creación	Por iniciar	1	2	2	Se simulará el tiempo de respuesta de una petición HTTP Post con un delay de 1 segundos.
HU05	Simular petición HTTP Delete hacia la API	Simular eliminar	Por iniciar	1	2	2	Se simulará el tiempo de respuesta de una petición HTTP Delete con un delay de 1 segundos.
D07	Creación de documentación de la API	Creación de ejemplos	Por iniciar	2	3	3	Se crearán los ejemplos de uso de la API para los usuarios. Los ejemplos de uso deberán especificar cómo se realizará la petición a la API y con ejemplos de lo que devolverá en caso de éxito o de error.
HU06	Creación de sitio estático como web de la API	Sitio principal	Por iniciar	1	3	3	Se creará la página principal al entrar por la ruta “/” a la API.
D08	Llenado de BDD con toda la información recopilada de los campeones.	Inserción de datos en BDD	Por iniciar	1	4	4	Se cargará el archivo JSON con toda la información recopilada en la BDD. Sólo se importará el archivo campeones.json a la base de datos y se confirmará que se haya creado correctamente.

HU07	Incorporación de la documentación de la web para ejemplos de uso	Vista de ejemplos	Por iniciar	3	4	3	Se creará una vista “/docs” para poner la documentación creada
HU08	Recuadro para poder probar la API sin salir del sitio	Formulario para probar API	Por iniciar	3	4	3	Se agregará un formulario donde el usuario pueda probar la API para obtener todos o un solo campeón de la BDD.
D09	Despliegue de la aplicación	Despliegue	Por iniciar	3	4	4	Se hará el despliegue de la aplicación y pruebas de funcionamiento en la plataforma que se seleccionó junto a la definición de tecnologías. Las pruebas funcionamiento consistirán a hacer llamadas a la API desde postman y desde un navegador.

Cambios al product backlog

- HU01: Se agregaron condiciones para el filtrado de los resultados al hacer una petición sobre toda la información de los campeones.
- HU02, HU03, HU04, HU05: Se aceptarán espacios y caracteres especiales en las búsquedas, creación, actualización y eliminación de un solo campeón
- HU03, HU04, HU05: Se bajó el tiempo de respuesta del servidor de 1.5 segundos a 1 segundo.
- Arquitectura de la aplicación: La API será dividida en una arquitectura similar a MVC para abstraer la lógica en diferentes capas (Rutas, Controladores y Conexión a BDD) y beneficiar el mantenimiento de la misma y la escalabilidad.

Fichas del Sprint Backlog:

HU01

Como: Usuario de la API

Quiero: Recibir los datos de todos los campeones

Para: Lograr manipular toda la información como me sea más conveniente

Condiciones:

- La información de cada campeón debe estar en formato JSON
- Todos los objetos deben venir en un arreglo
- En caso de error, debe regresar un objeto con el código y mensaje de error
- La API debe permitir la petición desde un dominio y desde localhost
- El endpoint debe comenzar desde “api/”
- El endpoint debe tener el nombre de tabla/colección de la base de datos, por ejemplo “api/champions”
- Por medio de los parámetros de búsqueda se debe poder hacer filtro por región y/o rol
- Por medio de los parámetros de búsqueda se debe poder hacer limitación de resultados devueltos

HU02

Como: Usuario de la API

Quiero: Recibir los datos de un sólo campeón

Para: Lograr manipular su información como me sea más conveniente

Condiciones:

- La información debe estar en formato JSON
- En caso de error, debe regresar un objeto con el código y mensaje de error
- La API debe permitir la petición desde un dominio y desde localhost
- El endpoint debe comenzar desde “api/champions/”
- El endpoint para cada campeón debe ser con su nombre, por ejemplo, “api/champions/[nombre del campeón]”

HU03

Como: Usuario de la API

Quiero: Usar el verbo HTTP PUT

Para: Agregar un método de actualización a la aplicación que crearé consumiendo la API

Condiciones:

- La petición no actuará directamente sobre lo base de datos, es decir, no actualizará realmente el recurso, sino que, al usar el endpoint con PUT, se procesará solamente en el servidor y tras un retraso de 1 a 2 segundos regresará un objeto con el código de estado 200 y el nombre del campeón al que le “actualizó” la información.
- El retraso con el que se enviará la respuesta de la petición debe hacerse con un método asíncrono para que no detenga el hilo de ejecución del servidor.
- En caso de error, debe regresar un objeto con el código y mensaje de error

- La API debe permitir la petición desde un dominio y desde localhost
- El endpoint debe comenzar desde “api/champions/[nombre del campeón]”

HU04

Como: Usuario de la API

Quiero: Usar el verbo HTTP POST

Para: Agregar un método de creación a la aplicación que crearé consumiendo la API

Condiciones:

- La petición no actuará directamente sobre lo base de datos, es decir, no creará realmente el recurso, sino que, al usar el endpoint con POST, se procesará solamente en el servidor y tras un retraso de 1 a 2 segundos regresará un objeto con el código de estado 201 y el nombre del recurso que fue “creado”.
- El retraso con el que se enviará la respuesta de la petición debe hacerse con un método asíncrono para que no detenga el hilo de ejecución del servidor.
- En caso de error, debe regresar un objeto con el código y mensaje de error
- La API debe permitir la petición desde un dominio y desde localhost
- El endpoint debe comenzar desde “api/champions/[nombre del campeón]”

HU05

Como: Usuario de la API

Quiero: Usar el verbo HTTP DELETE

Para: Agregar un método de eliminar a la aplicación que creare consumiendo la API

Condiciones:

- La petición no actuará directamente sobre lo base de datos, es decir, no eliminará realmente el recurso, sino que, al usar el endpoint con DELETE, se procesará solamente en el servidor y tras un retraso de 1 a 2 segundos regresará un objeto con el código de estado 200 y el nombre del campeón que fue “eliminado”.
- El retraso con el que se enviará la respuesta de la petición debe hacerse con un método asíncrono para que no detenga el hilo de ejecución del servidor.
- En caso de error, debe regresar un objeto con el código y mensaje de error
- La API debe permitir la petición desde un dominio y desde localhost
- El endpoint debe comenzar desde “api/champions/[nombre del campeón]”

HU06

Como: Usuario de la API

Quiero: Tener un sitio web al que pueda ingresar

Para: Leer más al respecto de la API y tener más información de la misma.

Condiciones:

- El sitio debe tener un estilo minimalista
- El sitio debe tener una barra de navegación con la opción de ir a hacia la documentación
- En la página principal del sitio debe haber un ejemplo de uso con el endpoint de la API

HU07

Como: Usuario de la API

Quiero: Tener una pestaña de documentación

Para: Saber qué respuesta me enviará la API, con qué tipos de datos y ejemplos de uso

Condiciones:

- La documentación debe de tener un ejemplo de cómo hacer las peticiones con al menos 3 lenguajes de programación.
- Cada campo del objeto JSON que regresará deberá tener una pequeña descripción de qué es y de qué tipo de datos es el valor del campo.
- La documentación debe estar organizada en el siguiente orden: GET All, GET One, POST, PUT, DELETE
- Debe haber un menú lateral permita navegar entre la documentación

HU08

Como: Usuario de la API

Quiero: Un recuadro donde pueda probar la API

Para: Que pueda usarla sin necesidad de hacer un desarrollo por mi cuenta o usar herramientas como Postman desde el inicio

Condiciones:

- El recuadro para probar la API debe ser fácil de reconocer

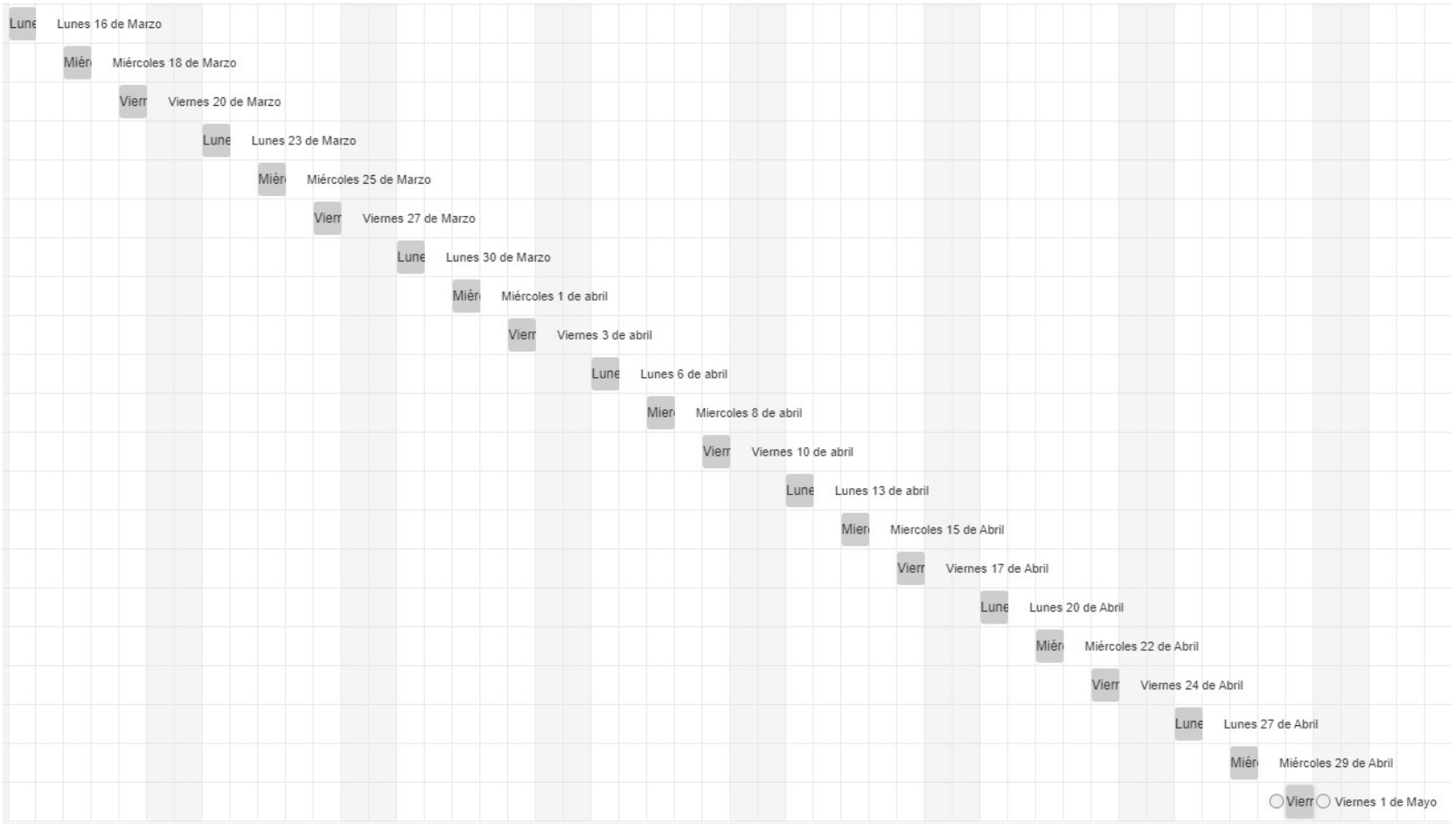
- El área del recuadro debe tener un input donde el usuario pueda poner de manera opcional el nombre de uno de los campeones
- Debe haber un textarea donde se mostrará la respuesta que regresa la API
- La respuesta de la API debe estar formateada en la estructura de un JSON para que sea más legible

Sprint Schedule

Lunes	Martes	Miércoles	Jueves	Viernes
16 Marzo Sprint Planning (Sprint 1)	17 Marzo Daily Standup (10:00 am)	18 Marzo Daily Standup (10:00 am)	19 Marzo Daily Standup (10:00 am)	20 Marzo Daily Standup (10:00 am)
23 Marzo Daily Standup (10:00 am) Grooming	24 Marzo Daily Standup (10:00 am)	25 Marzo Daily Standup (10:00 am) Testing	26 Marzo Daily Standup (10:00 am)	27 Marzo Revisión del sprint Retrospectiva del sprint
30 Marzo Sprint Planning (Sprint 2)	31 Marzo Daily Standup (10:00 am)	1 Abril Daily Standup (10:00 am) Grooming	2 Abril Daily Standup (10:00 am) Testing	3 Abril Daily Standup (10:00 am)
6 Abril Revisión del sprint Retrospectiva del sprint	7 Abril Sprint Planning (Sprint 3)	8 Abril Daily Standup (10:00 am)	9 Abril Daily Standup (10:00 am)	10 Abril Daily Standup (10:00 am)
13 Abril Daily Standup	14 Abril Daily Standup	15 Abril Daily Standup	16 Abril Daily Standup	17 Abril Revisión del sprint

(10:00 am) Grooming	(10:00 am)	(10:00 am) Testing	(10:00 am)	Retrospectiva del sprint
20 Abril Sprint Planning (Sprint 4)	21 Abril Daily Standup (10:00 am)	22 Abril Daily Standup (10:00 am)	23 Abril Daily Standup (10:00 am)	24 Abril Daily Standup (10:00 am) Grooming
27 Abril Daily Standup (10:00 am)	28 Abril Daily Standup (10:00 am) Testing	29 Abril Daily Standup (10:00 am) Implementación	30 Abril Daily Standup (10:00 am) Implementación	1 Mayo Revisión del sprint Retrospectiva del sprint Cierre del proyecto

Calendario de reuniones



Resultado de reunión #1

- Definición de roles
    - Jose Gutiérrez (Product Owner)

    - Adrián González (Scrum Master)
    - Luis Lira (Desarrollador)
- Cambios en el product backlog
    - Especificados al después del product backlog