Universidad San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Análisis y diseño de sistemas 2
Inga. Claudia Liceth Rojas Morales
Tutor de curso: Monica Raquel Calderon Muñoz

PROYECTO

Fase 2

INTRODUCCIÓN

El curso de Análisis y diseño de sistemas 2 busca la ampliación del conocimiento de las buenas prácticas así como herramientas adecuadas para que dichas prácticas sean correctamente ampliadas.

Los **patrones de diseño,** son una solución general, reutilizable y aplicable a diferentes problemas de diseño de software. Se trata de plantillas que identifican problemas en el sistema y proporcionan soluciones apropiadas a problemas generales, a los que se han enfrentado los desarrolladores durante un largo período de tiempo a través de prueba y error.

Se espera que el estudiante logre la implementación de los patrones de diseño en su codificación.

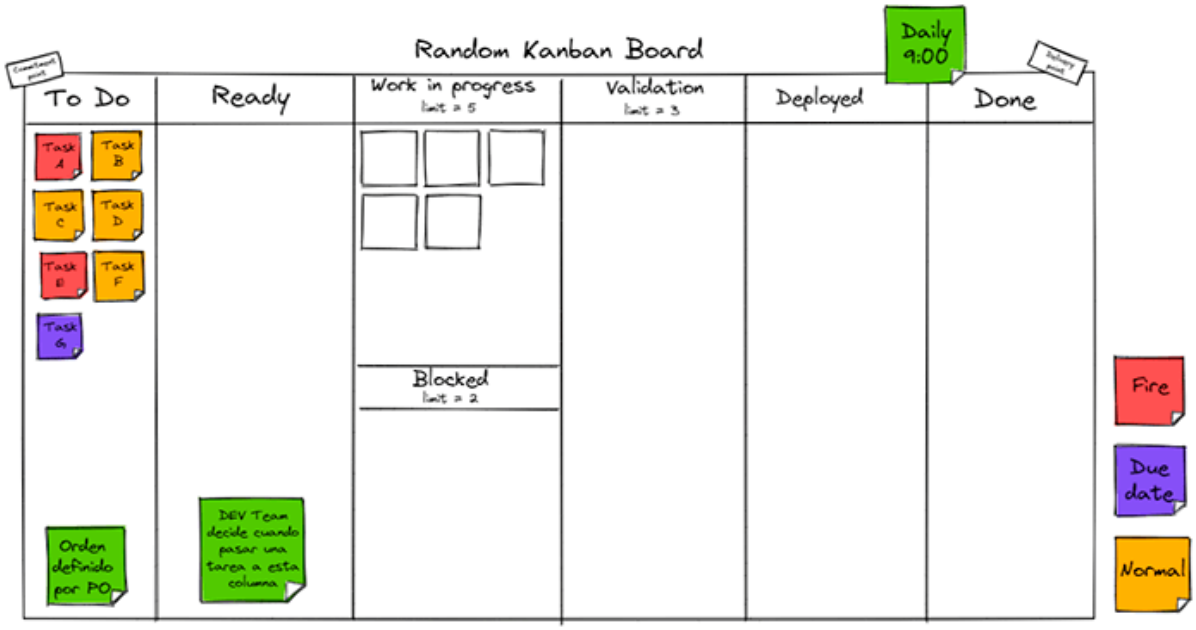
Se debe utilizar herramientas para manejo de contenedores tales como **Docker, DockerFile** y **Docker-Compose,** así como se debe de utilizar **Gitlab** como repositorio del código y debe de implementar la estrategía de branching **Git-Flow.** Tome en cuenta que se revisará que el trabajo haya sido equitativo y que los commits brinden valor al proyecto. El nombre para cada rama feature del proyecto deberá contar con la siguiente estructura:

[Número de ticket Jira/Trello]-[Nombre del ticket]
Ejemplo: ABC-1234-Ticket-Ejemplo

Se debe hacer uso de herramientas de gestión de proyectos, **Trello o JIRA**, para llevar el control de las tareas por medio de la creación de un tablero Kanban en donde mínimo se debe de visualizar las etapas:

- To Do
- Ready
- Work in progress
- Blocked
- Validation

- Deployed
- Done



Ejemplo tablero Kanban

Adicional, debe de utilizar herramientas que permitan la implementación de la metodología SCRUM tales como:

- Product Backlog
- Sprint Backlog
- Sprint Planning

Entre otras que considere necesarias para la mejora del avance del equipo, debe de presentar una documentación donde se detalle el Sprint Retrospective al finalizar cada Sprint.

Tome en cuenta que, por estar utilizando la metodología **ScrumBan** debe de utilizar **WIP** como límite en lugar de **Story Points.**

Hospital y centro médico veterinario Gatifu

Su equipo ha sido contratado para la continuación y mejora de la plataforma web de un hospital y centro médico veterinario, debe de tomar en cuenta que entre los nuevos requerimientos y mejoras se encuentran:

1. **Servicio de Grooming**
   Adicional a los servicios médicos que el hospital y centro médico brindan, se desea implementar un servicio de Grooming el cual cuenta con:

Baño	100
Corte de pelo	100
Masaje	50
Corte de uñas	50
Cepillado	100
Limpieza de oídos	50
Cepillado de dientes	50
Shampoo anti pulgas	75
Pañuelo o moño	Cortesía

   Se puede escoger un único servicio por contratar o el paquete completo, si el cliente desea el paquete completo se le hace un descuento del 45% sobre el valor total.

2. **Farmacia**
   Para brindar un mejor servicio, se ha puesto a disposición del cliente los mejores productos para la salud de la mascota. Entre los productos disponibles se encuentran
   - Concentrados medicados
   - Antipulgas y garrapatas
   - Entre otros.

   Los productos pueden ser dinámicamente agregados por el administrador del sistema así como los precios del mismo. Se debe llevar un control de inventario y si un producto deja de tener stock, ya no se debe de mostrar en pantalla del cliente.

3. **Ofertas y pagos**
   Debido a que se han incrementado los servicios por ofrecer, se tendrá la nueva opción en donde el cliente podrá optar a paquetes de servicio para poder ser beneficiado con descuentos.

   Estos paquetes de servicios serán totalmente personalizados por parte del administrador del sistema.

Para la implementación del diseño arquitectónico debe de tomar en cuenta los siguientes requerimientos:

1. El lenguaje y el framework para implementar el FrontEnd queda a discreción del equipo, sin embargo no debe de olvidarse que debe de ser intuitiva, atractiva y de fácil mantenimiento.
2. Los lenguajes para implementar el Backend queda a discreción del equipo, así como la base de datos y el tipo de la misma. Puede utilizar bases de datos en la nube (Amazon RDS, DynamoDB, Mongo, entre otros)
3. Se debe de implementar un diseño arquitectónico basado en servicios y microservicios donde se maneje la lógica de backend de cada una de las partes del sistema de forma separada.
4. Para la implementación de microservicios se debe elegir una parte del sistema backend que pueda implementarse como servicio y dividirlo en al menos **4 microservicios** por medio del aumento de granularidad, en donde cada microservicio realice una funcionalidad.
5. Se debe de implementar un servicio **middleware** que sirva de intermediario entre las solicitudes de FrontEnd y el Backend, este servicio únicamente debe dejar pasar los JSON provenientes del FrontEnd hacia el Backend y viceversa.
   Toda la comunicación desde FrontEnd hacia Backend y viceversa debe pasar por este servicio, caso contrario no se calificará.

6. Todo lo solicitado (FrontEnd y servicios) debe de ser implementado haciendo uso de Docker y Docker-Compose.

PATRONES DE DISEÑO

Para una facilidad de manejo y mantenimiento del proyecto, debe de implementar cuatro patrones de diseño en los microservicios que considere que se amolden mejor, dichos patrones de diseño quedan a discreción del estudiante cuáles escoger.

Para los patrones de diseño debe de presentar su respectivo diagrama de clases del patrón implementado correspondiente así como se debe de visualizar a nivel de código en donde fue implementado.

LOG Y ERRORES

Se deberá guardar en base de datos los JSON que van desde el FrontEnd hacia el Backend y viceversa y que pasan por el middleware; se debe de crear registros con los siguientes campos por cada comunicación que exista.

Método Middleware	Entrada	Salida	EsError	Fecha y hora

/pago	{
	“campo1”:”valor”
	“campo2”:”valor”
	}
		{
		“salida1”:”valor”
		“salida2”:”valor”
		}
			No	11/08/2022 15:50:12

/cita	{ “campo1”:”valor” “campo2”:”valor”	{ “salida1”:”valor” “salida2”:”valor”	Si	11/08/2022 22:50:12

	}	}		

La funcionalidad para guardar los logs se puede colocar dentro del Middleware o como un servicio aparte conectado al Middleware (queda a discreción del equipo)

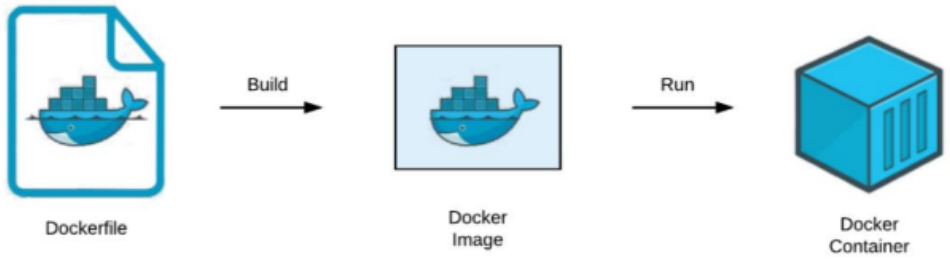
SEGURIDAD

- Se solicita que toda la información sensible de los usuarios **como correo y contraseñas** se almacene de forma **encriptada en base de datos**. Además, al momento de enviar datos sensibles de usuario desde el FrontEnd a los servicios y viceversa deben de viajar de manera encriptada dentro del **JSON.** Por lo que quiere decir que debe de existir **dos** tipos de encriptación:
    - La encriptación para el envío a través de JSON
    - La encriptación para el almacenamiento en la base de datos

IMPORTANTE: Por ser datos sensibles, la encriptación NO debe de ser la misma en el frontend a la que se realiza en el backend

- El método de encriptación queda a discreción de los estudiantes.
- Se debe de validar que el usuario cuenta con una sesión activa previo a acceder a cualquier página que lo requiera, de lo contrario debe de ser redireccionado a la página de inicio.
- Se debe validar que únicamente puedan subir archivos de office, imágenes o PDF.

FLUJO DE TRABAJO DOCKER



Proceso de contenerización

La contenerización es el empaquetado que se hace de una aplicación a través de un contenedor para su uso en pruebas o en producción.

Tras contenerizar todos los elementos de la aplicación se procederá a utilizar docker compose para ejecutar la aplicación, para ello, se debe de definir un archivo docker-compose.yml

DOCUMENTACIÓN SPRINTS

Se debe de implementar durante el desarrollo de la fase como **mínimo 4 sprints.** Tras la finalización de cada Sprint deben de redactar un documento que conste de los siguientes elementos:

- Tabla comparativa en donde se observen los elementos del Sprint Backlog que se logró terminar y los que no con justificación para los que no se lograron terminar.

- Imágen en donde se visualiza el tablero **previo al inicio del sprint.**
- Imágen en donde se visualiza el tablero **en la finalización del sprint.**
- Sprint retrospective: **CADA UNO DE LOS INTEGRANTES** debe de responder las preguntas:
    - ¿Qué se hizo bien durante el Sprint?
    - ¿Qué se hizo mal durante el Sprint?
    - ¿Qué mejoras se deben implementar para el próximo sprint?

IMPORTANTE: Debe de aparecer la opinión de cada estudiante, no una opinión general.

- Cálculo realizado para el WIP.
- Tabla en donde se indique:
    - Nombre de la historia
    - Nombre de la persona
    - Nombre de la rama (Con la respectiva notación)
    - Link hacia la rama

DOCUMENTACIÓN

- Documento con los **PROTOTIPOS** del frontend (**MOCKUPS)**
    - Se recomienda el uso del programa Balsamiq
    - Tome en cuenta que son **PROTOTIPOS,** por lo que **NO** deben de ser como las pantallas finales, solo debe de ser una idea general.
- Diagrama del diseño arquitectónico implementado.
- Diagramas de casos de uso
- Diagrama de componentes
- Diagrama de despliegue.
- Diagrama de clases de los patrones de diseño

CONSIDERACIONES

- El lenguaje para utilizar, framework y demás herramientas para crear el proyecto queda a discreción del estudiante, sin embargo, las herramientas **Docker y Docker Compose son obligatorias**
- Se debe de utilizar control de versiones GitLab.
- **COPIAS TOTALES O PARCIALES TENDRÁN UNA NOTA DE 0 Y SERÁN REPORTADAS A LA ESCUELA DE SISTEMAS**
- Deben de realizar **al menos** cuatro sprints.
- La estrategía de branching a utilizar debe ser **Git-Flow.**
- La documentación debe de ser almacenada en el repositorio de control de versiones utilizando markdown.
- A lo largo de la realización de todo el proyecto, dado el uso de metodologías ágiles, pueden existir ciertas modificaciones del aplicativo que deben de ser ajustadas y agregadas. Se irá indicando el tipo de prioridad que estás poseerán.

ENTREGABLES

- Link a repositorio con el código fuente que incluya los archivos DockerFile y Docker Compose.
- El auxiliar todo el tiempo debe de tener acceso al repositorio así como a la herramienta del control de trabajo. **Si no se tiene acceso al repositorio por lo menos QUINCE días antes de la entrega NO se calificará.**

FECHA DE ENTREGA 16 de octubre del 2022 antes de las 23:59