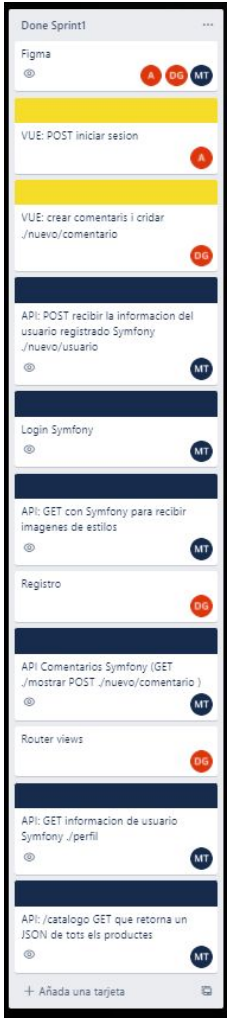
Presentacion Tecnica

Marc Castro , Camilo Pérez y Dylan Vargas


Done Sprint1
Figma
A DG MT
VUE: POST iniciar sesion
A
VUE: crear comentaris i cridar ./nuevo/comentario
DG
API: POST recibir la informacion del usuario registrado Symfony ./nuevo/usuario
MT
Login Symfony
MT
API: GET con Symfony para recibir imagenes de estilos
MT
Registro
DG
API Comentarios Symfony (GET ./mostrar POST ./nuevo/comentario)
MT
Router views
DG
API: GET informacion de usuario Symfony ./perfil
MT
API: /catalogo GET que retorna un JSON de tots els productes
MT
Añada una tarjeta


Done Sprint2
API: POST FORM y JSON con Symfony para enviar los productos seleccionados para comprar
MT
API: POST FORM y JSON con Symfony para enviar los servicios seleccionados para reservar
MT
Perfil de usuario
A
VUE: mostrar comentaris consultant api ./comentarios
A
VUE: POST FORM Login
A
Añada una tarjeta


Done Sprint3
API: GET Symfony servicios
MT
VUE: añadir comentarios
DG
VUE: Mostrar productes i que l'usuari els pugui seleccionar i al final cridar a l'API amb un post FORM(fetch /catalogo, post ./comprar)
MT
VUE: Barra navegación de la página web
A
VUE: POST JSON y FORM seleccionar los servicios con fecha.
MT
VUE: Pinia
MT
Interfaz de Symfony
MT
VUE: Diseño de login
MT
VUE: Diseño de la página de tienda
MT
VUE: Diseño de la página de reserva
MT
VUE: Diseño de la página principal, explicación de nuestros servicios
A
VUE: Galeria de imagenes
DG
VUE: Diseño de la página de perfil de usuario
A
VUE: Diseño de la página de registro
DG
Añada una tarjeta

Problemas y soluciones

Durante el proceso del proyecto no hemos tenido dificultades en programar ya que casi la mayoría es enviar y recibir información, pero a mitades hemos llegado en momentos de pensar mucho en cómo conseguir cierta información de una tabla a otra o guardar la información de una manera, pero uno de los problemas que hemos tenido es con el pinia ya que supuestamente se utiliza para mantener la información en la sesión y de esta manera el usuario se podrá mantener la sesión, pero al final la solución es que si ponía tiene una relación con un componente este componente no puede estar en App.vue ya que no cargaba la información.

Aspectos técnicos, requisitos técnicos y funcionalidades

Fronted:

- **Registro de cuenta de usuario:** Hemos hecho un componente registro en vue que envía los datos (nombre, apellido, teléfono, email, contraseña) por POST a la base de datos y con sweetalert hacemos que salte un alert si el usuario ya está registrado o no existe. Requisitos técnicos para funcionar
- **Compra de productos:** Hemos hecho un componente tienda que mostrará todos los productos que pilla de la base de datos con un GET, después en guardar, el usuario tiene que estar logueado o sino saldrá un sweetalert diciéndole que “tienes que iniciar sesión para poder comprar”, ya logueado puede seleccionar las unidades que quiera de un producto o varios, pero tiene un tope que es la cantidad de unidades disponibles y no podrá restar menos 0 y se enviará un POST.
- **Cita reservada:** En pedir una reserva ante todo comprueba si el usuario está logueado y sino saldrá un sweetalert, en el momento que el usuario escoge un día, en el apartado de las horas pueden salir pintados en rojo significando que ja esta reservado y que solamente puedes escoger una hora y se enviará un POST.
- **Vista de tus servicios:** Hemos realizado un envío de datos por GET de la base de datos del catálogo de productos y luego pintamos los campos nombre, cantidad y precio, luego por POST enviamos los datos de la compra a una tabla tickets donde te guarda la cantidad y el precio del total de productos.
- **Valoración de producto/servicio:** A través de POST rellenamos y enviamos los datos de “valoración” a una tabla “comentarios” donde guarda la valoración numérica y un comentario de texto.
- **Vista de valoración:** exportamos la información de los comentarios por props y hacemos un bucle v-for de la valoración de los comentarios para poner las estrellas que correspondan a cada comentario.
- **Perfil:** El usuario podrá ver su información, puede ver su historial de compras y reservas, puede cancelar servicios y por último puede cambiar la contraseña.

Backend:

La mayoría del código ha sido más de recolectar o guardar información y en la interfaz de Symfony poner un cuadro de búsqueda que puedas consultar un usuario, productos, servicios…

El requisitos técnicos sería PHP 8.0 y twig para las plantillas

Tener activadas las extensiones de JSON y ctype