# Guía de Proyecto Web Avanzadas

Cada grupo debe crear un proyecto para crear con cualquier temática para una PYME (Pequeña y mediana empresa).

## Para el primer bimestre:

Cada grupo **obligatoriamente** debe crear una **aplicación web monolítica(monolita)** usando cualquier stack tecnológico, con la condición de que la aplicación web debe **obligatoriamente** tener conexión con una **base de datos** relacional o NoSQL, y funcionalidad de un chat con **websocket**.

**¿Qué implica una aplicación monolítica con páginas web?**

* Renderizado de vistas HTML desde el servidor.
* Routing del lado del servidor.
* Lógica de negocio, persistencia y presentación todo en un solo proyecto.
* Soporte de WebSocket dentro del mismo stack

**Stacks tecnológicos sugeridos para crear la aplicación web monolítica:**

* Node.js + HTLM5+CSS3+JS+Express + Socket.IO/WS
* Next.js (Fullstack React) + Prisma + socket.io, ws, o next-websocket
* Django(Python) + Django Channels (ASGI) para WebSocket
* Flask(Python) + Jinja2 + SQLAlchemy + Flask-SocketIO
* Spring Boot (Java) que incluye soporte nativo a WebSocket
* ASP.NET Core MVC (con Razor Pages o Views) + Entity Framework + WebSocket mediante SignalR.
* Laravel (PHP) + Eloquent ORM+ beyondcode/laravel-websockets o Pusher para WebSocket
* Ruby on Rails (Ruby) + ORM ActiveRecord + Action Cable para WebSocket

## Para el segundo bimestre:

Transformar su aplicación presentada para que tenga una parte frontend, otra parte backend. Tanto el frontend como el backend pueden ser construidos con cualquier stack tecnológico.

Para el backend, obligatoriamente se deben crear microservicios

Se debe realizar despliegue tanto del frontend como backend en contenedores con Docker

**Stack tecnológicos recomendados:**

**Frontend**

* React + socket.io, WS)
* Next.js
* Vue.js
* Angular
* Svelte
* SolidJS
* Qwik
* Blazor (WebAssembly)
* Razor Pages

**Backend**

* NodeJS + Express.js
* Fastify (JS/TS)
* NestJS
* FastAPI
* Flask
* Spring Boot
* ASP.NET Core
* Laravel

Como combinaciones frontend y backend sugeridos de forma resumida tenemos:

|  |  |  |
| --- | --- | --- |
| **Nombre perfil** | **Frontend sugerido** | **Backend sugerido** |
| JavaScript Fullstack | React, **Next**.js, Vue, Svelte | **NestJS**, Express, AdonisJS |
| Pythonic | **React**, Vue | **FastAPI**, Flask, Django+DRF |
| JVM / Enterprise | **Angular**, React | **Spring Boot**, Micronaut |
| .NET | **Blazor**, Razor | **ASP.NET Core** + SignalR |
| PHP | **Vue**, Inertia.js | **Laravel** + Echo |
| Funcional / Escalable | Svelte, Vue | Phoenix (Elixir) |
| High Performance | React, SvelteKit | Go (Fiber, Gin), Rust (Actix) |

**Para el primer bimestre:**

Cada grupo deberá generar una presentación Power Point, Canva o cualquier otra herramienta, para realizar la presentación de su trabajo. Esta presentación debe cubrir las siguientes especificaciones:

1. Introducción con el planteamiento del problema.
2. Objetivo general, específicos
3. Cronograma de actividades.
4. Requerimientos funcionales o Historias de usuario.
5. Diagrama de arquitectura de la aplicación.
6. Link del repositorio público, con participación (commits) de todos los integrantes.
7. Pantallas realizadas:
   1. Una pantalla de de SignIn (llamada también Login) que lea y guarde información encriptada con una base de datos (dato contraseña).
   2. Pantalla de SignUp (llamada también Registro, Enrolamiento de nuevo usuario).
   3. Pantalla de Reseteo (llamada también Olvide clave).
   4. Pantalla Principal (llamada también Main o Desktop) que se presentará una vez que se autentique.
   5. Pantalla Chat (usar Websockets)
   6. Desde la pantalla principal deben realizarse operaciones CRUD:
      1. Crear (Create) un nuevo producto
      2. Consultar (Read/Query/Search) productos, consultar por producto
      3. Actualizar (Update) producto
      4. Borrar (Delete) un productor
8. Debe existir una conexión con una base de datos Relacional o una base de datos NoSQL, la elección de la base de datos es a libre elección de cada grupo.

**Para el segundo bimestre:**

Cada grupo deberá generar una presentación Power Point, Canva o cualquier otra herramienta, para realizar la presentación de su trabajo. Esta presentación debe cubrir las siguientes especificaciones:

1. Presentar una introducción (planteamiento) de que se trata su aplicativo
2. Diagrama de la arquitectura de la aplicación web anterior
3. Diagrama de arquitectura del frontend y backend con microservicios
4. Análisis y explicación de refactoring de base de datos (database per microservice)
5. Uso de dos patrones de diseño usados para crear microservicios
6. Medidas de seguridad implementadas (JWT, OAuth 2.0 / OpenID Connect)
7. Despliegue en contenedores tanto el frontend como backend
8. Uso de monitoreo y logs
9. Uso de DevOps para CI y CD.

**Consideraciones generales:**

1. Debe validar datos requeridos desde pantalla (datos requeridos, solo números, longitud de datos).
2. Debe existir navegabilidad entre las pantallas (interfaces).
3. Todos los integrantes del grupo expondrán su trabajo.
4. Cada grupo debe colocar en Teams, en la carpeta correspondiente a su grupo:
   1. El código fuente de la aplicación o link de github público (o el repositorio que use).
   2. La presentación PPT usada o link público.