Proyecto en Grupo: app_project

Descripción del proyecto

En grupos de 2–3 personas, vais a crear una pequeña aplicación que:

1. Tenga un sistema básico de **registro y login** usando **expresiones regulares (regex)** para validar los datos del usuario.
2. Gestione correctamente los usuarios (mínimo: registro, almacenamiento y login).
3. Consuma **una API pública** de forma limpia.
4. Muestre los datos obtenidos de la API de forma clara y organizada.
5. Esté publicado y versionado en un repositorio de **GitHub** llamado **app_project**.

El objetivo es que practiquéis:

- trabajo en equipo,
- uso básico de Git y GitHub,
- regex,
- consumo de APIs,
- manipulación de datos en formato JSON.

Parte 1: Creación del repositorio en GitHub

1. Un miembro del grupo crea un repositorio llamado **app_project**.
2. Añadís un archivo **README.md** describiendo:
    a. objetivo del proyecto,
    b. integrantes del grupo,
    c. instrucciones de ejecución.
3. Todos los miembros deben colaborar mediante **commits**, **branches** o **pull requests**.

Parte 2: Registro y Login con Regex

Debéis implementar un pequeño sistema de autenticación con:

Registro

Solicitar:

- nombre de usuario
- email
- contraseña

Con la validación que corresponda con regex.

Almacenamiento

Los usuarios deben guardarse en un fichero (por ejemplo, JSON).
Opcional: no guardar contraseñas en texto plano (usar hash).

Login

El usuario podrá iniciar sesión comprobando:

- email existente
- contraseña correcta (comparación de hashes, si corresponde)

Parte 3: Consumo de una API pública

Una vez que el usuario ha iniciado sesión correctamente, el programa deberá:

1. Hacer una petición a una API elegida por el grupo.
2. Recibir y procesar la respuesta en formato JSON.
3. Mostrar los datos más importantes de manera legible con la lógica y fines que decida el grupo.

APIs propuestas (elegid 1)

PokeAPI

URL: https://pokeapi.co/
Qué hace: Datos de Pokémon (tipos, estadísticas, habilidades, evoluciones).
Uso típico: Proyectos educativos, juegos, análisis de datos.

OpenWeatherMap

URL: https://openweathermap.org/api
Qué hace: Clima en tiempo real, pronósticos e históricos.
Uso típico: Apps de clima, modelos predictivos.

The Dog API / The Cat API

URL: https://thedogapi.com/, https://thecatapi.com/
Qué hace: Información y fotos de razas de perros y gatos.
Uso típico: Apps de entretenimiento, proyectos con imágenes.

CoinGecko API

URL: https://www.coingecko.com/en/api
Qué hace: Datos de criptomonedas (precios, rankings, exchanges).
Uso típico: Apps financieras, análisis de tendencias.

REST Countries

URL: https://restcountries.com/
Qué hace: Datos de países (población, idioma, moneda, fronteras…).
Uso típico: Visualizaciones, dashboards internacionales.

NASA APIs

URL: https://api.nasa.gov/
Qué hace: Imágenes y datos espaciales, meteoritos, satélites.
Uso típico: Proyectos educativos, análisis científico.

Dog CEO (Dog API)

URL: https://dog.ceo/dog-api/
Qué hace: Fotos aleatorias de perros por raza.
Uso típico: Ejemplos de JSON, apps de entretenimiento.

Open Movie Database (OMDb API)

URL: http://www.omdbapi.com/
Qué hace: Datos sobre películas y series.
Uso típico: Apps de cine, análisis de datos culturales.

Chuck Norris Jokes API

URL: https://api.chucknorris.io/
Qué hace: Chistes de Chuck Norris en JSON.
Uso típico: Ejemplos simples y divertidos.

Jikan API (Anime / MyAnimeList)

URL: https://jikan.moe/
Qué hace: Datos de animes, mangas, personajes y rankings.
Uso típico: Proyectos educativos, análisis de popularidad.

ENTREGA

El repositorio debe contener:

- Código funcional (login + consumo de API)
- README completo
- Archivo con usuarios (JSON o similar)
- Commits de todos los miembros
- Ejemplo de ejecución/capturas de pantalla
- Código organizado en módulos (mínimo 2–3 archivos)