REUNION DE DEMOSTRACION Y REVISION DEL SPRINT X

NOMBRE DEL PROYECTO

27/06/2024

AGENDA

- Historias de Usuario completadas en este Sprint
- Demostración del trabajo completado
- Pendientes

- Historias de Usuario completadas en este Sprint

HISTORIA DE USUARIO			
ID Historia	Como (Rol)...	Deseo....	Para....
HU01	Alumno	Deseo un recordatorio para visualizar las fechas de entrega de	Para tener un mejor
HU02	Alumno	Deseo que me de respuestas acerca de los horarios de mis	Para estar atento a la hora
HU03	Analista Financiero	Deseo agendar mis examenes y tareas pendientes	Para estar mejor
HU01	Docente	Deseo poder generar reporte de notas en una determinada fecha	Para tener un informe de la
HU02	Docente	Deseo poder tener un calendario donde contenga las fechas de	Para tener orden en cada
HU03	Docente	Deseo tener un registro completo de las asistencias de los	Para poder considerarlo en

		OTROS DATOS DE LA EPICA O HISTORIA DE USUARIO				
ID Historia	Como (Rol)...	Criterios de Aceptación	Prioridad	Estimación	Dependencias	Sprint
HU01	Alumno	Debe mostrar las fechas de las tareas y exámenes	1	15		1
HU02	Alumno	Debe mostrar los hararios de cada asignatura y su salon de	1	10		1
HU03	Analista Financiero	Debe poder agendar un evento como recordatorio	1	20		1
HU01	Docente	Debe mostrar las califiaciones de los alumnos		20		2
HU02	Docente	Debe mostrar un calendario de eventos de cada clase		20		2
HU03	Docente	Debe mostrar un registro de las asistencias		15		2

- Demostración del trabajo completado

```python
# Función principal del chatbot
def main():
    # Cargar datos desde los archivos CSV
    asignaturas_data = read_csv_data('descripcion_asignaturas.csv')
    horarios_data = read_csv_data('horarios_clases.csv')

    print("💬 ChatGPT API en Python")

    table = Table("Comando", "Descripción")
    table.add_row("exit", "Salir de la aplicación")
    table.add_row("new", "Crear una nueva conversación")
    print(table)

    # Contexto del asistente
    context = {"role": "system", "content": "Eres un asistente muy útil."}
    messages = [context]

    while True:
        content = __prompt()

        if content == "new":
            print("🆕 Nueva conversación creada")
            messages = [context]
            content = __prompt()

        messages.append({"role": "user", "content": content})

        # Manejar consultas sobre asignaturas
        if "asignaturas" in content.lower():
            asignaturas = ", ".join([asignatura['Asignatura'] for asignatura in asignaturas_data])
            respuesta = f"Las asignaturas disponibles son: {asignaturas}. ¿Sobre cuál específicamente quieres información?"
            messages.append({"role": "assistant", "content": respuesta})
            print(f"[bold green]> [/bold green] [green]{respuesta}[/green]")
```

- Pendientes