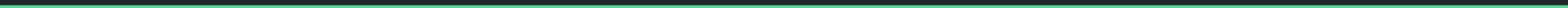


GRUPO 1

CIERRE SPRINT 3



User Stories Planificadas

Historias de usuario

US 8.1 - Comentario de rechazo de publicación
de parte del Admin

3 SP	Cumplido
------	----------

US 4.4 - Gestión de suscripción Premium

8 SP	Cumplido
------	----------

US 5.4 - Ordenar sugerencias por puntaje

5 SP	Cumplido
------	----------

US 4.3 - Armar Itinerario (IA)

13 SP	Cumplido
-------	----------

US 5.7 - Ver detalles publicación en la ficha

8 SP	Cumplido
------	----------

US 4.7 - Marcar publicaciones favoritas como
pendientes o realizados

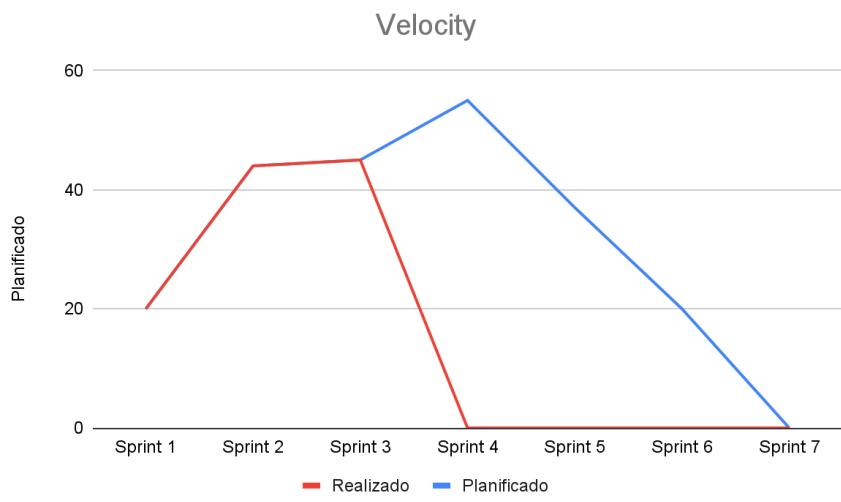
8 SP	Cumplido
------	----------

Velocity

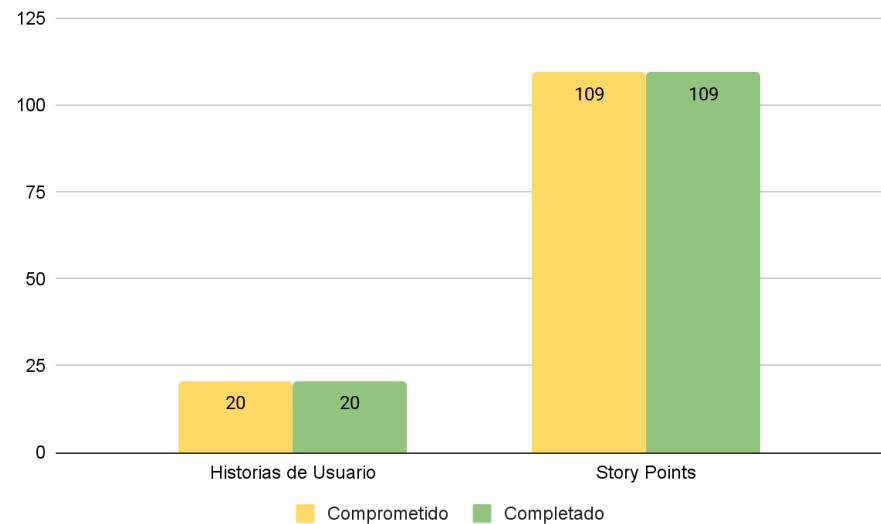
45 SP

Total:

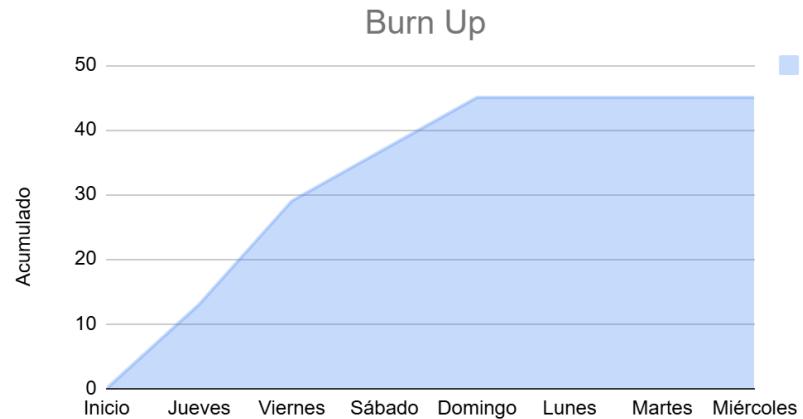
Métricas



Velocity actual: 45 SP



Métricas



Uso de IA

Uso de IA

- Herramienta utilizada: ChatGPT, Gemini y Copilot
- Versiones e iteraciones: Se le envió contexto con código ya hecho con el fin de generar código nuevo. Se utilizó también para resolución de errores.

Uso de IA - Ejemplos



hola chat ayuda

el front quedo bien pero no esta reorganizando . asi estan ordenadas de mayor coincidencia a menor coincidencia. si pongo menor coincidencia, deberia reordenarlas siendo primero Giza, despues Roma y ultimo el Louvre

lee todo detenidamente

mi suggestions.py

```
from fastapi import PIRouter, Depends, Query  
from sqlalchemy.orm import Session
```

sigue sin funcinar.

necesito q se genere el top10, y desp dependiendo del boton, ese top se muestre ascendente o descendente. por descendente (puntajes mas alto primero -> mayor coincidencia)

Perfecto, Agustina 🌟 leí todo nuevamente y ahora entiendo *exactamente* lo que necesitas: no querés que cambie el conjunto de sugerencias (el **top 10**), sino que el **orden del top 10** ya calculado se invierta según el botón de orden.

Ahora mismo tu backend *vuelve a ordenar antes de sacar el top 10*, por eso cambia el conjunto y no simplemente la dirección del orden.

Vamos a corregir eso.

✓ Solución correcta

Primero generás el top 10 en orden descendente (puntajes más altos primero), y **después**, dependiendo de `sort`, invertís el orden del resultado antes de devolverlo.

Así:

```
python Copy code  
  
@router.get("", response_model=List[schemas.PublicationOut])  
def getSuggestions(  
    db: Session = Depends(get_db),  
    user=Depends(get_current_user),  
    sort: str = Query("desc", enum=["asc", "desc"])  
):  
    pref = db.query(models.UserPreference).filter_by(user_id=user.id).first()  
    if not pref:  
        return []
```

Uso de IA - Ejemplos



perfecto. Ya lo solucione. A continuación, necesito crear un script que me genere, automáticamente, un conjunto de publicaciones, como para estandarizar algunas publicaciones para todos los desarrolladores del equipo que van a trabajar en el proyecto. Crear en una carpeta de assets también las imágenes asociadas a esas publicaciones. Necesito que me arme 8 publicaciones estandares. 4 que sean de hoteles y 4 de actividades.

Ver razonamiento ▾

¡Excelente idea! Estandarizar datos de prueba (lo que se conoce como "seeding") es una práctica genial para que todo el equipo trabaje sobre la misma base.

He preparado un script para ti llamado `seed_db.py`. Este script:

1. Se conecta a tu base de datos existente.
2. Busca un usuario "admin" (o el primer usuario que encuentra) para usarlo como autor.
3. Crea 4 hoteles y 4 actividades, pero solo si no existen ya (por el `place_name`).
4. Crea las categorías "hotel" y "actividad" si no existen.
5. **Descarga imágenes de placeholder** (desde `placeholder.co`) y las guarda en tu carpeta `backend/app/static/uploads/publications/`, tal como lo haría tu API.

Aquí están los archivos. El más importante es `seed_db.py`.

<code>seed_db.py</code>	30 oct, 17:04	Abrir
<code>gitignore_update.txt</code>	30 oct, 14:10	Abrir

me llenas este script con mas publicaciones para que coincidan mas sugerencias

```
import os
import shutil
# import requests <--- ELIMINADO
from sqlalchemy.orm import Session
from datetime import datetime

# Ajusta las importaciones para que funcione como un script
# Asume que se ejecuta desde la raíz del proyecto (ej:
```

¡Claro que sí! Aquí tenés tu script `seed_db.py` actualizado con 12 publicaciones más.

Me concentré en agregar variedad para que coincidan con las preferencias que tenés en el frontend (playa, montaña, noche, historia, etc.) y en cubrir los continentes y climas que faltaban (como África y climas tropicales).

Simplemente **reemplaza el contenido de tu archivo** con este código:

Python

```
import os
import shutil
```

Uso de IA - Justificación de Resultado Final

- Acelera el diseño y la codificación y el debugging.
- Permitió optimizar la lógica de recomendación y filtrado, evitando errores manuales.
- Facilitó la depuración y refactorización del proyecto.
- Ayudó a mantener consistencia entre backend y frontend, generando estructuras de datos y endpoints más claros.

Retro

Sprint 3 - Evolución de Riesgos

No se presentó ninguno de los riesgos

- Baja de un integrante
- Dependencia de terceros (APIs, cuotas y costos)
- Caída de sistema

Además, logramos capacitarnos sobre herramientas de integración de la IA, sobrellevando este riesgo de Gestión.

Sprint 3 - Retro

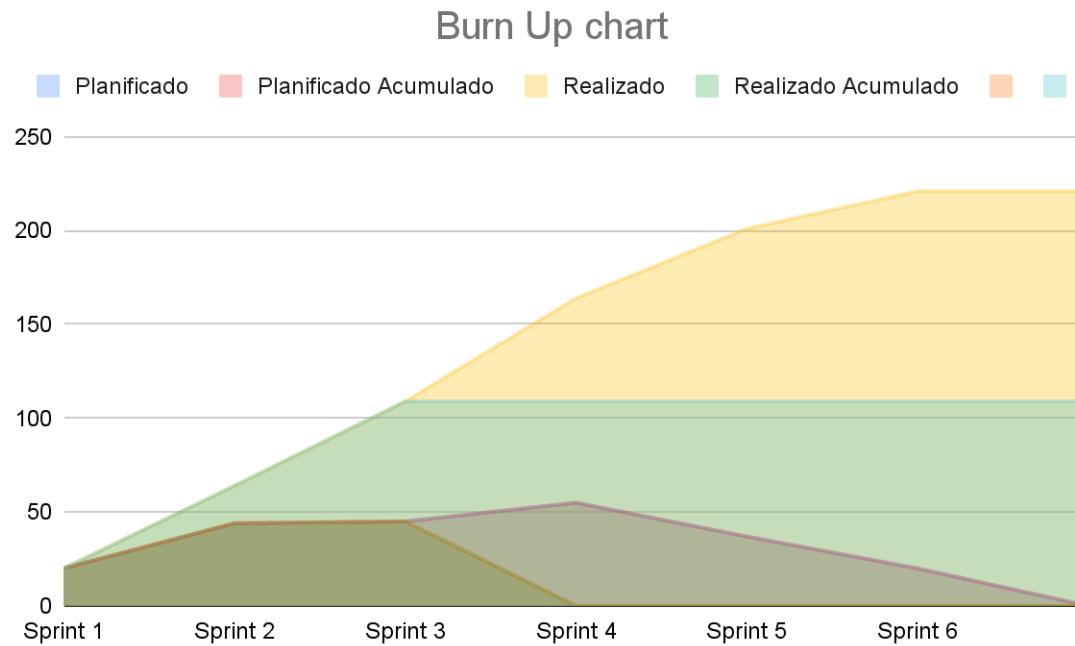
Aspectos Positivos

- En este sprint el desarrollo fue llevado a cabo de una manera más coordinada entre los integrantes. Se corrigió el error de commits y desarrollo en una única branch.

Aspectos a Mejorar

- + Debemos continuar trabajando la comunicación para dar aviso de actualizaciones en el desarrollo de forma bien clara.

Avance



DEMO

Compromisos para Sprint 4

Historias de usuario

- 4.9 Compartir itinerario (Premium)
- 5.8 Me gusta en reseñas (Premium)
- 5.9 Comentar una reseña
- 5.10 Reportar una reseña
- 6.1 Cargar gastos del viaje
- 6.2 Editar / Eliminar un gasto
- 6.3 Vista de gastos por categoría
- 6.4 Compartir planilla de gastos (Premium)
- 6.5 Invitar por mail a la planilla (Premium)
- 6.6 Cálculo de saldos por integrante
- 6.7 Exportar presupuesto

8 SP

5 SP

5 SP

5 SP

8 SP

3 SP

5 SP

5 SP

5 SP

3 SP

3 SP