Pantalla: Itinerario con Reorganización por IA — App de Viajes en Grupo

0) Contexto y Objetivo

Tras la selección de actividades principales por parte del usuario/grupo, esta pantalla muestra un **itinerario automático** basado en precedencias. Además, ofrece una acción de **"Reorganizar con IA"** para generar **3 propuestas alternativas** optimizadas según parámetros de cada actividad: `location`, `precio_promedio`, `tiempo_promedio`, `tipo` (recreacional, comida, turística). El usuario puede **comparar y elegir** una de las tres opciones.

---

1) Requerimientos Funcionales (RF)

1. **Carga de Itinerario Base**

2. RF1.1: Mostrar el itinerario inicial generado por precedencia (orden definido al seleccionar actividades / marcado por el usuario).

3. RF1.2: Visualizar por item: nombre de actividad, tipo, duración estimada, costo promedio, ubicación (map pin o badge de barrio/ciudad), hora sugerida.

4. RF1.3: Resumen superior: total de tiempo, costo estimado total, número de actividades.

5. **Reorganización con IA**

6. RF2.1: Botón "Reorganizar con IA" abre modal/lateral con parámetros (presupuesto total del día, ventana horaria, preferencia de variedad de tipos, tolerancia a desplazamiento).

7. RF2.2: Al confirmar, el sistema genera **3 propuestas** etiquetadas (p. ej., *Equilibrada*, *Menor Desplazamiento*, *Más Económica*).
8. RF2.3: Cada propuesta muestra cronograma, métricas clave (tiempo de transporte, costo total, diversidad de tipos, tiempo ocioso) y **racional** corto (por qué este orden fue sugerido).

9. RF2.4: El usuario puede **previsualizar** cada opción y **Seleccionar** una como itinerario activo.

10. **Edición Manual (Drag & Drop)**

11. RF3.1: Soportar drag&drop para reordenar actividades.

12. RF3.2: Recalcular horas sugeridas y métricas al soltar.

13. RF3.3: Validar superposiciones/choques y advertir con badges de conflicto.

14. **Mapa y Distancias**

15. RF4.1: Mini mapa con markers y polilínea según el orden actual.

16. RF4.2: Mostrar tiempo de traslado estimado entre actividades (en línea o badge).

17. **Estados y Persistencia**

18. RF5.1: Estados: vacío (sin actividades), cargando, error, listo.

19. RF5.2: Guardar la opción elegida y el orden manual final como **versión del itinerario**.

20. RF5.3: Permitir **deshacer/rehacer** últimos cambios locales.

21. **Colaboración Básica**

22. RF6.1: Mostrar avatar/estado de miembros conectados.

23. RF6.2: Permitir comentarios/notas por actividad.

---

2) Requerimientos No Funcionales (RNF)

- RNF1 **Rendimiento:** generación de 3 propuestas < 3 s con caché de matriz de distancias; render < 200 ms tras respuesta.
- RNF2 **Confiabilidad:** si el motor IA falla, fallback a heurística local (greedy por proximidad + presupuesto).
- RNF3 **Usabilidad:** drag&drop accesible (teclado), tooltips claros, explicación breve de cada propuesta.
- RNF4 **Accesibilidad (WCAG 2.1 AA):** contraste suficiente, navegación por teclado, labels aria en botones y tarjetas.
- RNF5 **I18N:** soportar es/en; moneda local configurable (GTQ por defecto). Zona horaria **America/Guatemala**.
- RNF6 **Analítica:** eventos para click en reorganizar, vistas de propuestas, selección final, abandonos.
- RNF7 **Seguridad/Privacidad:** no exponer PII en logs; parámetros de IA anonimizados.
- RNF8 **Offline-first (parcial):** vista base y drag&drop funcionan offline; reorganización IA requiere conexión.

---

3) Modelos de Datos

3.1 Activity

```
{
  "id": "act_123",
  "nombre": "Museo Nacional",
  "tipo": "turistica", // "recreacional" | "comida" | "turistica"
  "location": { "lat": 14.61, "lng": -90.52, "direccion": "Zona 1" },
  "precio_promedio": 60.0, // en moneda activa
  "tiempo_promedio_min": 90,
  "ventana": { "abre": "09:00", "cierra": "17:00" },
  "precedencia": ["act_045"], // debe ir después de estas
```

```
  "notas": "Entrada incluye exposición temporal"
}
```

3.2 Itinerary

```
{
  "id": "it_2025_09_25",
  "grupoId": "grp_77",
  "fecha": "2025-10-12",
  "zonaHoraria": "America/Guatemala",
  "items": [
    { "activityId": "act_045", "start": "09:30", "end": "10:30",
"viaje_min": 12 },
    { "activityId": "act_123", "start": "10:45", "end": "12:15",
"viaje_min": 15 }
  ],
  "metricas": { "costo_total": 180.0, "tiempo_total_min": 360,
"traslados_min": 45, "diversidad": 0.67 },
  "version": 3,
  "estado": "activo" // draft | activo | archivado
}
```

3.3 CandidateItinerary (propuesta IA)

```
{
  "candidateId": "cand_B",
  "label": "Menor Desplazamiento",
  "itinerary": { /* mismo shape que Itinerary */ },
  "racional": "Minimiza suma de tiempos de traslado usando clustering por
proximidad.",
  "puntuacion": { "costo": 0.82, "tiempo": 0.91, "variedad": 0.65, "global":
0.86 }
}
```

3.4 Constraints & Preferencias de Grupo

```
{
  "presupuesto_max": 400.0,
  "ventana_dia": { "inicio": "08:00", "fin": "21:00" },
  "peso_variedad": 0.4,
  "peso_traslado": 0.35,
  "peso_costo": 0.25,
  "tolerancia_huecos_min": 20
}
```

---

4) API Contracts (OpenAPI-like)

Base URL: `/api/v1`

4.1 POST `/itineraries/auto-generate`

Body

```
{ "grupoId": "grp_77", "fecha": "2025-10-12", "activities": [Activity],
"zonaHoraria": "America/Guatemala" }
```

200 → `Itinerary`

4.2 POST `/itineraries/optimize`

Genera **3** propuestas. Acepta parámetros y devuelve candidatos con racionales.

Body

```
{
  "itineraryBase": Itinerary,
  "activities": [Activity],
  "constraints": Constraints,
  "n": 3,
  "estrategias": ["equilibrada", "menor_traslado", "mas_economica"]
}
```

200

```
{ "candidatos": [CandidateItinerary, CandidateItinerary,
CandidateItinerary] }
```

Errores: `422` (datos inválidos), `503` (motor IA no disponible; header `X-Fallback: heuristic`).

4.3 PATCH `/itineraries/{id}/reorder`

Body

```
{ "nuevoOrdenActivityIds": ["act_045", "act_123", "act_999"] }
```

200 → `Itinerary`

4.4 POST `/itineraries/{id}/commit-candidate`

Body

```
{ "candidateId": "cand_B" }
```

200 → `Itinerary` (version++ y estado `activo`).

4.5 GET `/maps/distance-matrix`

Query: `origins`, `destinations`, `mode=driving|walking` → matriz de tiempos/minutos y distancias/met.

4.6 POST `/feedback/itinerary`

Body

```
{ "itineraryId": "it_2025_09_25", "satisfaccion": 4, "comentarios":
"Muy bien balanceado." }
```

200 `{ "ok": true }`

---

5) Componentes de UI

- **Header de Itinerario** (resumen: costo total, tiempo total, #actividades, fecha, TZ badge).
- **Timeline / Day Planner** (lista ordenada con horas, tarjetas por actividad, indicadores de traslado).
- **Tarjeta de Actividad**
- Título + tipo (chip de color), duración, costo, ventana horaria.
- Badges: *fuera de horario*, *muy lejos*, *excede presupuesto*.
- Acciones: **Drag handle**, notas, eliminar del día.
- **Mini Mapa** con markers y ruta según orden.
- **Botón principal "Reorganizar con IA"** (primary).
- **Sheet/Modal de Configuración de IA**: sliders/pickers (presupuesto, ventana del día, pesos: variedad/traslado/costo, tolerancia de huecos).
- **Carrusel de Propuestas (3)**: tarjetas comparables con: etiqueta, métrica global, sub-métricas, preview timeline, botón **Seleccionar**.
- **Toasts/Alertas**: conflictos, fallbacks, guardado exitoso.
- **Collab Bar**: avatares de miembros conectados, contador de votos si se habilita votación.

5.1 Estados Visuales

- **Vacío:** CTA para "Agregar actividades".
- **Cargando:** skeletons en timeline y tarjetas de propuestas.
- **Error:** panel con reintentar + explicación.
- **Con Fallback Heurístico:** banner informativo.

---

6) Lógica de Optimización (alto nivel)

1. **Datos base**: tiempos de traslado (Distance Matrix), duración por actividad, ventanas horarias, precedencias duras.
2. **Objetivos** (ponderados):
3. Minimizar **traslados** (suma de minutos entre actividades).
4. Ajustar a **presupuesto** y minimizar costo total.
5. Maximizar **variedad** (entropía de tipos en secuencia, penalizando repeticiones consecutivas).
6. Minimizar **tiempos muertos** fuera de ventanas.
7. **Restricciones**:
8. Precedencias obligatorias.
9. Respeto de ventanas horarias; si no cabe, marcar conflicto o desplazar dentro del rango.
10. **Generación de 3 propuestas** (cambiar pesos/estrategia):
11. **Equilibrada**: pesos {variedad:0.4, traslado:0.35, costo:0.25}.
12. **Menor Desplazamiento**: {traslado:0.6, variedad:0.25, costo:0.15} con clustering geográfico (k-medoids) y orden tipo TSP greedy.
13. **Más Económica**: {costo:0.6, traslado:0.25, variedad:0.15} con filtro de outliers de precio y emparejamiento de comidas fuera de horas punta.

6.1 Pseudocódigo (heurístico con precedencias)

```
input: activities, constraints, distance_matrix

valid = enforce_precedences_and_windows(activities)

function score(order):
  cost = sum(precio_promedio(activity) for activity in order)
  travel = sum(distance_matrix[i,j].min_time for consecutive i->j)
  variety = entropy(types(order)) - repeat_penalty(order)
  idle = gaps_outside_windows(order)
  return w_cost*(-normalize(cost)) + w_travel*(-normalize(travel)) +
w_variety*(normalize(variety)) + w_idle*(-normalize(idle))

candidates = []
for strategy in [EQUILIBRADA, MENOR_TRASLADO, MAS_ECONOMICA]:
  set_weights(strategy)
  order = greedy_seed_by_precedence(valid)
  order = local_search_2opt(order, score)
  order = repair_windows(order)
  timeline = schedule_with_start(order, constraints.ventana_dia)
  candidates.append( (order, timeline, score(order)) )

return top 3 candidates with rationales
```

---

7) Flujos de Usuario

1. **Ver Itinerario Base → Reorganizar con IA → Comparar 3 propuestas → Seleccionar → Guardar**

2. **Ver Itinerario Base → Ajustes manuales (drag&drop) → Resolver conflictos → Guardar**
3. **Fallo IA → Mostrar banner y ofrecer Heurística → Generar propuestas heurísticas → Seleccionar**

---

8) Edge Cases

- Actividades con ventanas que no se solapan ⇒ marcar y proponer mover al día siguiente (fuera de alcance en esta pantalla: sólo avisar).
- Distancia o tiempo de traslado excesivo ⇒ badge “muy lejos” + sugerir intercambiar con actividad cercana.
- Presupuesto superado ⇒ mostrar sobrecosto y sugerir versión “Más Económica”.
- Precedencias cíclicas mal definidas ⇒ error 422 con detalle.

---

9) Telemetría (eventos sugeridos)

- `itinerary.view` (props: items, costo_total, traslados_min)
- `itinerary.ai_optimize.click` (props: constraints)
- `itinerary.ai_optimize.result` (props: n=3, lat_ms)
- `itinerary.ai_optimize.select` (props: candidateId)
- `itinerary.reorder.drag` / `drop`
- `itine