👥 CASOS DE USO - Sistema de Asistencias

🧮 ¿QUÉ PUEDE HACER CADA USUARIO?

Basándome en el diagrama y la API implementada, aquí están los casos de uso principales para cada tipo de usuario.

🎓 ESTUDIANTE / ASISTENTE

1. 📋 CONSULTAR SERVICIOS DISPONIBLES

Objetivo: Ver qué servicios/talleres/cursos están disponibles

Endpoint: `GET /servicios/`

Respuesta esperada:

```
{
  "items": [
    {
      "id": 1,
      "nombre_servicio": "Taller de Emprendimiento",
      "descripcion": "Aprende a crear tu propia empresa",
      "id_departamento": 1
    },
    {
      "id": 2,
      "nombre_servicio": "Clase de Programación Python",
      "descripcion": "Fundamentos de Python",
      "id_departamento": 2
    }
  ]
}
```

Flujo en la app móvil:

```
1. Estudiante abre la app
2. Ve lista de servicios disponibles
3. Puede filtrar por departamento
```

---

2. 🗓️ CONSULTAR SESIONES DE UN SERVICIO

Objetivo: Ver las próximas sesiones de un servicio que me interesa

Endpoint: `GET /sesion/servicio/{id_servicio}`

Ejemplo: `GET /sesion/servicio/1`

Respuesta esperada:


```
{
  "items": [
    {
      "id": 101,
      "nombre_sesion": "Sesión 1: Idea de Negocio",
      "fecha": "2025-10-05",
      "hora_inicio": "14:00:00",
      "hora_fin": "16:00:00",
      "lugar_sesion": "Aula 301",
      "n_maximo_asistentes": 30,
      "inscritos_actuales": 15,
      "id_facilitador": "F001"
    },
    {
      "id": 102,
      "nombre_sesion": "Sesión 2: Plan de Negocios",
      "fecha": "2025-10-12",
      "hora_inicio": "14:00:00",
      "hora_fin": "16:00:00",
      "lugar_sesion": "Aula 301",
      "n_maximo_asistentes": 30,
      "inscritos_actuales": 0
    }
  ]
}
```


Flujo en la app móvil:

```
1. Estudiante selecciona un servicio
2. Ve calendario de sesiones
3. Puede ver:
   - Fecha y hora
   - Lugar
   - Cupos disponibles (30 max - 15 inscritos = 15 disponibles)
   - Facilitador
```

---

3. ☑ REGISTRAR ASISTENCIA A UNA SESIÓN

Objetivo: Marcar mi asistencia cuando llego a la sesión

Endpoint: `POST /asistencias/`

Cuerpo de la petición:

```
{
  "id_sesiones": 101,
  "id_persona": 12345,
  "documento_identidad": "1234567890",
  "usuario_creacion": "estudiante@utb.edu.co",
  "geo_latitud": 10.3932,
  "geo_longitud": -75.4832,
  "observaciones": "Asistencia mediante QR"
}
```

Respuesta:

```
{
  "message": "Asistencia registrada correctamente"
}
```

Flujo en la app móvil:

```
1. Estudiante llega a la sesión
2. Escanea código QR mostrado por el facilitador
   O
   Selecciona manualmente la sesión y presiona "Registrar Asistencia"
3. La app captura:
   - ID de la sesión (del QR o selección)
   - ID del estudiante (de su perfil)
   - Ubicación GPS (para validar que está en el lugar)
   - Hora actual (timestamp automático)
4. Se envía el registro
5. Recibe confirmación
```

Validaciones que podría hacer la app:

- ☑ Verificar que la sesión está en curso (horario correcto)
- ☑ Verificar que no haya registrado asistencia antes (evitar duplicados)
- ☑ Verificar que hay cupos disponibles
- ☑ Verificar que está cerca del lugar (geolocalización)

---

4. 📊 CONSULTAR MI HISTORIAL DE ASISTENCIAS

Objetivo: Ver a qué sesiones he asistido

Endpoint: `GET /asistencias/persona/{id_persona}`

Ejemplo: `GET /asistencias/persona/12345`

Respuesta esperada:

```
{
  "items": [
    {
      "id": 501,
      "id_sesiones": 101,
      "fecha_creacion": "2025-10-05T14:05:00",
      "observaciones": "Asistencia mediante QR",
      "geo_latitud": 10.3932,
      "geo_longitud": -75.4832
    },
    {
      "id": 502,
      "id_sesiones": 95,
      "fecha_creacion": "2025-09-28T10:15:00",
      "observaciones": "Asistencia manual"
    }
  ]
}
```


Flujo en la app móvil:

```
1. Estudiante va a "Mi Historial"
2. Ve lista de sesiones a las que ha asistido
3. Puede ver:
   - Fecha y hora de asistencia
   - Nombre de la sesión/servicio
   - Ubicación donde registró
```

---

5. 📍 VER DETALLES DE UNA SESIÓN

Objetivo: Ver información completa de una sesión

Endpoint: `GET /sesion/{id}`

Ejemplo: `GET /sesion/101`

Respuesta:


```
{
  "id": 101,
  "nombre_sesion": "Sesión 1: Idea de Negocio",
  "descripcion": "Aprenderás a identificar oportunidades de negocio",
  "fecha": "2025-10-05",
  "hora_inicio": "14:00:00",
  "hora_fin": "16:00:00",
  "lugar_sesion": "Aula 301 - Edificio A",
  "id_modalidad": 1,
```

```
  "n_maximo_asistentes": 30,
  "inscritos_actuales": 15,
  "id_facilitador": "F001",
  "id_servicio": 1
}
```

👩‍🏫 FACILITADOR / PROFESOR

1. 📅 CONSULTAR MIS SESIONES PROGRAMADAS

Objetivo: Ver qué sesiones tengo asignadas

Endpoint: `GET /sesion/` (con filtro en frontend por id_facilitador)

Nota: Idealmente se filtrarían del lado del servidor, pero con la respuesta completa se puede filtrar en la app.

Flujo en la app móvil:

```
1. Facilitador inicia sesión
2. La app filtra las sesiones donde id_facilitador == su ID
3. Ve su calendario de sesiones
```

---

2. 👥 VER LISTA DE ASISTENTES A UNA SESIÓN

Objetivo: Ver quién asistió a mi clase/sesión

Endpoint: `GET /asistencias/sesion/{id_sesion}`

Ejemplo: `GET /asistencias/sesion/101`

Respuesta esperada:

```
{
  "items": [
    {
      "id": 501,
      "id_persona": 12345,
      "documento_identidad": "1234567890",
      "fecha_creacion": "2025-10-05T14:05:00",
      "observaciones": "Asistencia mediante QR"
    },
    {
      "id": 502,
      "id_persona": 67890,
      "documento_identidad": "0987654321",
      "fecha_creacion": "2025-10-05T14:07:00",
      "observaciones": "Asistencia manual"
    }
```

```
  ]
}
```

Luego, con cada `id_persona`, puede consultar:

Endpoint: `GET /personas/{id}`

Respuesta:

```
{
  "id": 12345,
  "nombre_asistente": "Juan Pérez",
  "identificacion": "1234567890",
  "codigo_banner": "T00012345",
  "tipo_de_identificacion": "CC"
}
```

Flujo en la app móvil:

```
1. Facilitador selecciona una sesión
2. Presiona "Ver Asistentes"
3. La app consulta asistencias de esa sesión
4. Para cada asistencia, consulta datos de la persona
5. Muestra lista completa:
   ☑ Juan Pérez - CC 1234567890 - 14:05
   ☑ María García - CC 0987654321 - 14:07
6. Total: 2 de 30 cupos
```

---

3. 📱 GENERAR CÓDIGO QR PARA LA SESIÓN

Objetivo: Facilitar el registro rápido de asistencia

Flujo en la app móvil:

```
1. Facilitador abre la sesión activa
2. Presiona "Generar QR"
3. La app genera un QR con:
   {
     "id_sesion": 101,
     "codigo_sesion": "SESS-101-20251005",
     "timestamp": "2025-10-05T14:00:00"
   }
4. Muestra el QR en pantalla grande
5. Los estudiantes escanean el QR
6. Cada escaneo registra una asistencia automáticamente
```

Backend necesario (opcional):

```
# Endpoint para validar QR
POST /asistencias/registrar-qr/
Body: {
  "qr_data": "SESS-101-20251005",
  "id_persona": 12345,
  "geo_latitud": 10.3932,
  "geo_longitud": -75.4832
}
```

---

4. 📝 REGISTRAR ASISTENCIA MANUAL

Objetivo: Registrar asistencia de un estudiante que no pudo escanear QR

Endpoint: `POST /asistencias/`

Flujo en la app móvil:

```
1. Facilitador presiona "Registro Manual"
2. Busca estudiante por:
   - Nombre
   - Documento
   - Código Banner
3. Selecciona al estudiante
4. Confirma registro
5. Se crea la asistencia
```

---

5. 📊 VER ESTADÍSTICAS DE LA SESIÓN

Objetivo: Ver métricas de asistencia

Endpoints combinados:

- `GET /sesion/{id}` - Info de la sesión
- `GET /asistencias/sesion/{id}` - Lista de asistentes

Cálculos en la app:

```
const sesion = await getSesion(101);
const asistencias = await getAsistenciasSesion(101);

const estadisticas = {
  total_inscritos: sesion.n_maximo_asistentes,
  total_asistentes: asistencias.items.length,
  porcentaje_asistencia: (asistencias.items.length / sesion.inscritos_actuales) *
```

```
100,
  cupos_disponibles: sesion.n_maximo_asistentes - sesion.inscritos_actuales
};
```

Vista en la app:

```
📊 Estadísticas - Sesión 1

👥 Asistentes: 15 / 30
📈 Tasa de asistencia: 50%
🎯 Cupos disponibles: 15
⏰ Última asistencia: 14:25
```

---

6. ❌ ELIMINAR ASISTENCIA (corrección)

Objetivo: Corregir errores en el registro

Endpoint: `DELETE /asistencias/{id}`

Ejemplo: `DELETE /asistencias/501`

Flujo en la app móvil:

```
1. Facilitador ve lista de asistentes
2. Identifica registro erróneo
3. Presiona "Eliminar"
4. Confirma acción
5. Se elimina el registro
```

🔐 **ADMINISTRADOR** (Casos adicionales)

1. 🏢 GESTIONAR DEPARTAMENTOS

Crear departamento:

```
POST /departamentos/
Body: {
  "descripcion": "Departamento de Ingeniería",
  "centro": "Facultad de Ingeniería"
}
```

Listar departamentos:

```
GET /departamentos/
```

---

2. 🛍️ GESTIONAR SERVICIOS

Crear servicio:

```
POST /servicios/
Body: {
  "id_departamento": 1,
  "nombre_servicio": "Taller de Robótica",
  "descripcion": "Introducción a la robótica educativa"
}
```

Ver servicios de un departamento:

```
GET /servicios/departamento/1
```

---

3. 👨‍🏫 GESTIONAR FACILITADORES

Registrar facilitador:

```
POST /facilitadores/
Body: {
  "nombre": "Dr. Carlos Méndez",
  "email": "cmendez@utb.edu.co",
  "telefono": "3001234567",
  "activo": true
}
```

Listar facilitadores:

```
GET /facilitadores/
```

---

4. 👥 GESTIONAR PERSONAS

Registrar estudiante:

```
POST /personas/
Body: {
```

```
  "codigo_banner": "T00012345",
  "identificacion": "1234567890",
  "tipo_de_identificacion": "CC",
  "nombre_asistente": "Ana López"
}
```

Buscar por documento:

```
GET /personas/documento/1234567890
```

---

5. 🧮 CREAR SESIONES

Crear nueva sesión:

```
POST /sesion/
Body: {
  "id_servicio": 1,
  "id_periodo": 1,
  "id_tipo": 1,
  "nombre_sesion": "Sesión 3: Prototipado",
  "descripcion": "Creación de prototipos",
  "fecha": "2025-10-19",
  "hora_inicio": "14:00:00",
  "hora_fin": "16:00:00",
  "lugar_sesion": "Lab de Robótica",
  "id_modalidad": 1,
  "id_facilitador": "F001",
  "n_maximo_asistentes": 25
}
```

🧮 FLUJOS COMPLETOS EN LA APP MÓVIL

FLUJO 1: Estudiante registra asistencia mediante QR

```
┌─────────────────────────────────────────┐
│  ESTUDIANTE                             │
├─────────────────────────────────────────┤
│ 1. Abre app móvil                       │
│ 2. Inicia sesión                        │
│ 3. Va a "Registrar Asistencia"          │
│ 4. Presiona "Escanear QR"               │
│ 5. Apunta cámara al QR del facilitador  │
│ 6. App captura:                         │
│    - id_sesion del QR                   │
│    - id_persona del perfil              │
```

```
│    - GPS actual                          │
│ 7. POST /asistencias/                    │
│ 8. ☑ "Asistencia registrada"             │
└──────────────────────────────────────────┘
```

FLUJO 2: Facilitador toma asistencia

```
┌──────────────────────────────────────────┐
│  FACILITADOR                             │
├──────────────────────────────────────────┤
│ 1. Abre app móvil                        │
│ 2. Va a "Mis Sesiones"                   │
│ 3. Selecciona sesión de hoy              │
│ 4. Presiona "Generar QR"                 │
│ 5. Muestra QR en pantalla                │
│ 6. Estudiantes escanean                  │
│ 7. Va a "Ver Asistentes"                 │
│ 8. GET /asistencias/sesion/101           │
│ 9. Ve lista en tiempo real:              │
│    ☑ Juan Pérez - 14:05                  │
│    ☑ María García - 14:07                │
│    ☑ Pedro Ruiz - 14:09                  │
│ 10. Exporta lista (opcional)             │
└──────────────────────────────────────────┘
```

FLUJO 3: Estudiante consulta servicios y sesiones

```
┌──────────────────────────────────────────┐
│  ESTUDIANTE                              │
├──────────────────────────────────────────┤
│ 1. Abre app móvil                        │
│ 2. Va a "Servicios Disponibles"          │
│ 3. GET /servicios/                       │
│ 4. Ve lista de servicios                 │
│ 5. Selecciona "Taller Emprendimiento"    │
│ 6. GET /sesion/servicio/1                │
│ 7. Ve calendario de sesiones:            │
│    📅 Oct 5 - 2:00 PM - Aula 301         │
│    📅 Oct 12 - 2:00 PM - Aula 301        │
│ 8. Selecciona sesión                     │
│ 9. GET /sesion/101                       │
│ 10. Ve detalles completos                │
│ 11. Agrega a su calendario (opcional)    │
└──────────────────────────────────────────┘
```

🎯 RESUMEN DE CAPACIDADES

Acción	Estudiante	Facilitador	Admin
Ver servicios disponibles	☑	☑	☑
Ver sesiones de un servicio	☑	☑	☑
Registrar propia asistencia	☑	❌	❌
Ver mi historial	☑	❌	❌
Ver asistentes de sesión	❌	☑	☑
Generar QR para sesión	❌	☑	☑
Registro manual de asistencia	❌	☑	☑
Crear servicios	❌	❌	☑
Crear sesiones	❌	⚠ (limitado)	☑
Gestionar facilitadores	❌	❌	☑
Gestionar departamentos	❌	❌	☑

🔄 INTEGRACIÓN CON LA APP FLUTTER

Ejemplo de código Flutter para registrar asistencia:

```dart
// servicio_asistencia.dart
import 'package:http/http.dart' as http;
import 'dart:convert';

class ServicioAsistencia {
  final String baseUrl = 'http://127.0.0.1:8000';

  Future<bool> registrarAsistencia({
    required int idSesion,
    required int idPersona,
    required String documento,
    required String usuario,
    double? latitud,
    double? longitud,
  }) async {
    final url = Uri.parse('$baseUrl/asistencias/');

    final body = {
      'id_sesiones': idSesion,
      'id_persona': idPersona,
      'documento_identidad': documento,
      'usuario_creacion': usuario,
      'geo_latitud': latitud,
      'geo_longitud': longitud,
      'observaciones': 'Asistencia mediante QR'
    };
```

```
    final response = await http.post(
      url,
      headers: {'Content-Type': 'application/json'},
      body: json.encode(body),
    );

    return response.statusCode == 200;
  }

  Future<List<dynamic>> obtenerSesionesPorServicio(int idServicio) async {
    final url = Uri.parse('$baseUrl/sesion/servicio/$idServicio');
    final response = await http.get(url);

    if (response.statusCode == 200) {
      final data = json.decode(response.body);
      return data['items'] ?? [];
    }
    return [];
  }

  Future<List<dynamic>> obtenerMiHistorial(int idPersona) async {
    final url = Uri.parse('$baseUrl/asistencias/persona/$idPersona');
    final response = await http.get(url);

    if (response.statusCode == 200) {
      final data = json.decode(response.body);
      return data['items'] ?? [];
    }
    return [];
  }
}
```

☑ CONCLUSIÓN

La API implementada permite **3 flujos principales**:

1. 📱 **Estudiantes:** Consultar servicios/sesiones y registrar su asistencia
2. 👨‍🏫 **Facilitadores:** Gestionar sesiones y verificar asistencias
3. 👨‍💼 **Administradores:** Configurar todo el sistema (servicios, sesiones, usuarios)

Todos los endpoints necesarios están implementados y listos para integrarse con la app Flutter. 🚀