Análisis y Plan de Mejoras - Sistema Prácticas 1 a 1

Versión: 2.0
Fecha: 17 de octubre de 2025
Proyecto: SpeaklyPlan - Prácticas 1 a 1

Índice

Estado Actual del Sistema

Arquitectura Implementada

El sistema de prácticas 1 a 1 cuenta con:

Backend Completo

- **Modelos de datos (Prisma):**
  - `PracticeInvitation` - Gestión de invitaciones
  - `PracticeConnection` - Conexiones entre usuarios
  - `PracticeMeeting` - Sesiones programadas/completadas
  - `PracticeNotification` - Notificaciones del sistema
- **API Endpoints:**
  - `/api/practice/invitations` - GET/POST invitaciones
  - `/api/practice/sessions` - GET/POST sesiones
  - `/api/practice/history` - Historial de prácticas
  - `/api/practice/connections` - Gestión de conexiones
  - `/api/practice/notifications` - Sistema de notificaciones
  - `/api/practice/search-user` - Búsqueda de usuarios

Servicios Backend

- `practice-service.ts` - Lógica de negocio principal
- `practice-notification-service.ts` - Notificaciones

Frontend Limitado

- **Dashboard:** Solo muestra un módulo "Prácticas 1 a 1" sin funcionalidad
- **No hay página dedicada:** No existe `/app/practice/page.tsx`
- **No hay componentes UI:** Los componentes en `/components/practice/` son mínimos

---

Análisis de Funcionalidades

Lo que FUNCIONA (Backend)

1. Sistema de Invitaciones

```
// Enviar invitación
POST /api/practice/invitations
{
  "receiverEmail": "user@example.com",
  "message": "¿Practicamos juntos?"
}

// Ver invitaciones recibidas/enviadas
GET /api/practice/invitations?type=received
GET /api/practice/invitations?type=sent
```

2. Gestión de Conexiones

- Crear conexiones al aceptar invitaciones
- Listar compañeros de práctica
- Estadísticas de sesiones por conexión

3. Programación de Sesiones

```
POST /api/practice/sessions
{
  "partnerId": "user123",
  "scheduledFor": "2025-10-20T15:00:00Z",
  "topic": "Business English",
  "externalLink": "https://meet.google.com/abc-defg-hij"
}
```

4. Historial y Estadísticas

```
GET /api/practice/history
// Devuelve:
// - Lista de sesiones pasadas
// - Total de minutos practicados
// - Total de sesiones
// - Compañeros más frecuentes
```

Lo que NO FUNCIONA (Frontend)

Sin Interfaz de Usuario

- No hay página para ver invitaciones
- No hay página para agendar sesiones
- No hay página de historial
- Solo existe un botón en el dashboard que no lleva a ningún lado

Sin Flujo Completo

- Usuarios no pueden:
  - Buscar otros usuarios
  - Enviar invitaciones
  - Aceptar/rechazar invitaciones
  - Programar sesiones

- Iniciar videollamadas
- Ver su historial

Problemas Identificados

1. Desconexión Backend-Frontend (Crítico)

- Backend completamente funcional pero sin UI
- Inversión técnica sin retorno para el usuario
- **Impacto:** Funcionalidad inutilizable

2. Agendamiento Manual (Alto)

- Usuarios deben crear links de Zoom/Meet manualmente
- No hay integración con calendarios
- No hay recordatorios automáticos
- **Impacto:** Fricción, sesiones olvidadas

3. Sin Sala de Práctica Integrada (Medio)

- Usuarios deben salir de la plataforma
- No hay registro automático de sesiones
- No hay timer integrado
- **Impacto:** Experiencia fragmentada

4. Búsqueda de Usuarios Limitada (Medio)

- Solo búsqueda por email
- No hay filtros por nivel, horario, intereses
- No hay matching automático
- **Impacto:** Difícil encontrar compañeros

5. Notificaciones Incompletas (Bajo)

- Solo notificaciones básicas
- No hay recordatorios 24h/1h antes
- No hay notificaciones push (solo email)
- **Impacto:** Sesiones olvidadas

Plan de Mejoras Propuesto

Fase 1: UI Básica (Prioridad CRÍTICA)

Objetivo: Hacer funcional el sistema existente

1.1 Página Principal de Prácticas

`/app/practice/page.tsx`

Contenido: - **Tabs:** - Invitaciones (recibidas/enviadas) - Mis Compañeros - Sesiones Programadas - Historial

1.2 Modal de Nueva Invitación

`/components/practice/invite-modal.tsx`

- Buscar usuario por email/nombre
- Mensaje personalizado
- Vista previa del perfil

1.3 Cards de Invitación

`/components/practice/invitation-card.tsx`

- Información del remitente
- Mensaje
- Botones: Aceptar / Rechazar
- Timestamp

1.4 Lista de Compañeros

`/components/practice/partners-list.tsx`

- Avatar, nombre, nivel
- Estadísticas (sesiones, minutos)
- Botón: "Programar Sesión"

1.5 Modal de Agendar Sesión

`/components/practice/schedule-modal.tsx`

- Selector de fecha/hora
- Tema de conversación
- Opción: Crear link automático (ver Fase 2)
- O pegar link manual

Estimación: 2-3 días
Impacto: Sistema completamente funcional (MVP)

Fase 2: Agendamiento Inteligente (Prioridad ALTA)

2.1 Integración con Google Calendar API (GRATUITA) **Ventajas:** - 100% gratuito - Usado por millones - Fácil integración - Recordatorios automáticos

Implementación:

```
// lib/services/calendar-service.ts

import { google } from 'googleapis'

export async function createCalendarEvent({
  userId,
  partnerId,
  scheduledFor,
  topic,
  meetLink
}: {
  userId: string
  partnerId: string
  scheduledFor: Date
```

```
  topic: string
  meetLink?: string
}) {
  // 1. Obtener tokens OAuth del usuario
  const userAuth = await getUserGoogleAuth(userId)

  // 2. Crear cliente de Calendar
  const calendar = google.calendar({ version: 'v3', auth: userAuth })

  // 3. Obtener emails de ambos usuarios
  const [user, partner] = await Promise.all([
    getUserEmail(userId),
    getUserEmail(partnerId)
  ])

  // 4. Crear evento
  const event = await calendar.events.insert({
    calendarId: 'primary',
    conferenceDataVersion: 1,
    requestBody: {
      summary: `Práctica de Inglés: ${topic}`,
      description: `Sesión 1 a 1 con ${partner.name} en SpeaklyPlan.\n\nTema: ${topic}`,
      start: {
        dateTime: scheduledFor.toISOString(),
        timeZone: 'America/Bogota'
      },
      end: {
        dateTime: new Date(scheduledFor.getTime() + 30 * 60000).toISOString(),
        timeZone: 'America/Bogota'
      },
      attendees: [
        { email: user.email },
        { email: partner.email }
      ],
      conferenceData: {
        createRequest: {
          requestId: `practice-${Date.now()}`,
          conferenceSolutionKey: { type: 'hangoutsMeet' }
        }
      },
      reminders: {
        useDefault: false,
        overrides: [
          { method: 'email', minutes: 24 * 60 }, // 24h antes
          { method: 'popup', minutes: 60 },      // 1h antes
          { method: 'popup', minutes: 10 }       // 10min antes
        ]
      }
    }
  })

  return {
    eventId: event.data.id,
    meetLink: event.data.hangoutLink,
```

```
    htmlLink: event.data.htmlLink
  }
}
```

Flujo Completo:

1. **Primera Vez (OAuth):**
   - Usuario hace clic en "Conectar Google Calendar"
   - Popup de autorización de Google
   - SpeaklyPlan guarda tokens (refresh token)
2. **Agendar Sesión:**

   ```
   Usuario A → Programa sesión con Usuario B
   ↓
   Sistema crea evento en Calendar de A
   Sistema envía invitación a B (email)
   B acepta → Evento se agrega a Calendar de B
   ↓
   Ambos reciben recordatorios automáticos
   Ambos obtienen link de Google Meet
   ```
3. **Recordatorios Automáticos:**
   - Google envía emails 24h antes
   - Notificaciones push 1h antes
   - Notificaciones push 10min antes

Ventajas Técnicas: - Links de Meet generados automáticamente - Recordatorios nativos de Google - Sincronización con calendario personal - Timezone handling automático - Reprogramación fácil

Código de Setup:

```
// app/api/auth/google-calendar/route.ts

import { google } from 'googleapis'

const oauth2Client = new google.auth.OAuth2(
  process.env.GOOGLE_CLIENT_ID,
  process.env.GOOGLE_CLIENT_SECRET,
  `${process.env.NEXTAUTH_URL}/api/auth/google-calendar/callback`
)

export async function GET(request: Request) {
  const { searchParams } = new URL(request.url)
  const action = searchParams.get('action')

  if (action === 'authorize') {
    // Generar URL de autorización
    const scopes = [
      'https://www.googleapis.com/auth/calendar',
      'https://www.googleapis.com/auth/calendar.events'
    ]

    const url = oauth2Client.generateAuthUrl({
      access_type: 'offline',
      scope: scopes,
      prompt: 'consent'
```

```
    })

    return NextResponse.redirect(url)
  }

  // Handle callback...
}
```

2.2 Alternativa: Calendly API (GRATUITA hasta cierto límite) **Ventajas:** - Interfaz dedicada para scheduling - Múltiples zonas horarias - Buffer times - Integra con Google/Outlook

Implementación:

```
// lib/services/calendly-service.ts

import axios from 'axios'

const calendlyAPI = axios.create({
  baseURL: 'https://api.calendly.com',
  headers: {
    Authorization: `Bearer ${process.env.CALENDLY_API_KEY}`
  }
})

export async function createSchedulingLink({
  userId,
  duration = 30,
  title = 'Práctica de Inglés 1 a 1'
}) {
  const response = await calendlyAPI.post('/scheduling_links', {
    max_event_count: 1,
    owner: `https://api.calendly.com/users/${userId}`,
    owner_type: 'EventType',
    event_type: {
      name: title,
      duration: duration,
      description_html: '<p>Sesión de práctica de inglés 1 a 1</p>'
    }
  })

  return response.data.resource.booking_url
}
```

Flujo: 1. Usuario A invita a Usuario B 2. Sistema genera link único de Calendly 3. Usuario B selecciona horario disponible 4. Ambos reciben confirmación automática

2.3 Solución Híbrida (RECOMENDADO) **Combo:** Google Calendar (backend) + UI Custom (frontend)

Por qué: - No dependemos de UI externa (Calendly) - Control total de la experiencia - Gratis 100% - Más integración con gamificación

Implementación:

1. **Selector de Horarios Inteligente**

```tsx
// components/practice/smart-scheduler.tsx

export function SmartScheduler({ partnerId }: { partnerId: string }) {
  // 1. Obtener disponibilidad del partner
  const { data: availability } = useSWR(
    `/api/practice/availability/${partnerId}`,
    fetcher
  )

  // 2. Mostrar slots disponibles
  return (
    <div className="grid grid-cols-3 gap-2">
      {availability.slots.map(slot => (
        <Button
          key={slot.start}
          onClick={() => scheduleSession(slot)}
          variant={slot.available ? 'default' : 'ghost'}
          disabled={!slot.available}
        >
          {formatTime(slot.start)}
        </Button>
      ))}
    </div>
  )
}
```

2. **Backend de Disponibilidad**

```tsx
// app/api/practice/availability/[userId]/route.ts

export async function GET(
  request: Request,
  { params }: { params: { userId: string } }
) {
  const { searchParams } = new URL(request.url)
  const date = searchParams.get('date') || new Date().toISOString()

  // 1. Obtener eventos del usuario desde Google Calendar
  const calendar = await getUserCalendar(params.userId)
  const events = await calendar.events.list({
    calendarId: 'primary',
    timeMin: startOfDay(date).toISOString(),
    timeMax: endOfDay(date).toISOString()
  })

  // 2. Generar slots cada 30 minutos
  const slots = generateTimeSlots(date, 30)

  // 3. Marcar slots ocupados
  const availableSlots = slots.map(slot => ({
    start: slot,
    end: addMinutes(slot, 30),
    available: !isOverlapping(slot, events.data.items)
  }))
```

```
  return NextResponse.json({ slots: availableSlots })
}
```

Estimación: 3-4 días
Impacto: Agendamiento sin fricción, recordatorios automáticos

---

Fase 3: Sala de Práctica Integrada (Prioridad MEDIA)

3.1 WebRTC con Daily.co (GRATUITA hasta 10K minutos/mes) **Ventajas:** - Video/audio integrado - No se sale de la plataforma - API súper simple - Gratis hasta 10K min/mes (~160 sesiones de 60min)

Implementación:

```
// components/practice/practice-room.tsx

import Daily from '@daily-co/daily-js'

export function PracticeRoom({ meetingId }: { meetingId: string }) {
  const [callFrame, setCallFrame] = useState<any>(null)

  useEffect(() => {
    // Crear sala
    const frame = Daily.createFrame({
      showLeaveButton: true,
      iframeStyle: {
        width: '100%',
        height: '600px',
        border: '0'
      }
    })

    // Unirse
    frame.join({
      url: `https://yourdomain.daily.co/${meetingId}`,
      userName: session.user.name
    })

    setCallFrame(frame)

    return () => frame.destroy()
  }, [meetingId])

  return (
    <div className="practice-room">
      {/* Daily.co renderiza aquí */}

      {/* Panel lateral con herramientas */}
      <aside>
        <SessionTimer />
        <TopicPrompts />
        <NoteTaker />
      </aside>
    </div>
```

```
  )
}
```

Flujo: 1. Sistema crea sala Daily.co al agendar 2. Ambos usuarios reciben link a `/practice/room/[id]` 3. Click → Video llamada en la misma página 4. Timer + notas integradas 5. Al finalizar → Feedback mutuo

Código de Backend:

```
// lib/services/daily-service.ts

import axios from 'axios'

const dailyAPI = axios.create({
  baseURL: 'https://api.daily.co/v1',
  headers: {
    Authorization: `Bearer ${process.env.DAILY_API_KEY}`
  }
})

export async function createDailyRoom(meetingId: string) {
  const response = await dailyAPI.post('/rooms', {
    name: meetingId,
    privacy: 'private',
    properties: {
      max_participants: 2,
      enable_screenshare: true,
      enable_chat: true,
      enable_knocking: false,
      exp: Math.floor(Date.now() / 1000) + 3600 // 1 hora
    }
  })

  return response.data.url
}
```

Estimación: 2-3 días
Impacto: Experiencia profesional, todo en un solo lugar

Fase 4: Mejoras de Descubrimiento (Prioridad BAJA)

4.1 Búsqueda Avanzada

- Filtros por nivel (A1-C2)
- Filtros por zona horaria
- Filtros por intereses (business, travel, casual)

4.2 Matching Automático (Algoritmo)

```
function findBestMatch(userId: string) {
  const user = await getUser(userId)

  // Scoring
  const candidates = await getAllUsers()
  const scored = candidates.map(candidate => ({
    user: candidate,
    score: calculateMatchScore(user, candidate)
```

```
  }))

  // Factores de score:
  // - Diferencia de nivel (±1 nivel ideal)
  // - Zona horaria compatible (±3 horas)
  // - Intereses compartidos
  // - Disponibilidad horaria
  // - Rating de sesiones anteriores

  return scored.sort((a, b) => b.score - a.score)[0]
}
```

4.3 Sugerencias Proactivas

- "3 usuarios disponibles ahora"
- "María busca practicar business English"
- "Juan tiene el mismo nivel que tú"

Estimación: 2-3 días
Impacto: Más sesiones, mejor engagement

Priorización y Roadmap

Sprint 1 (Semana 1) - MVP Funcional

Objetivo: Sistema utilizable de extremo a extremo

- Página principal `/practice`
- Modal de invitaciones
- Lista de compañeros
- Agendar sesión (manual con link externo)
- Historial básico

Entregables: - Los usuarios pueden usar todo el sistema existente - Pueden enviar/aceptar invitaciones - Pueden programar sesiones - Pueden ver su historial

Testing: - Crear 2 usuarios de prueba - Flujo completo: Invitar → Aceptar → Agendar → Completar - Verificar notificaciones

---

Sprint 2 (Semana 2) - Agendamiento Inteligente

Objetivo: Automatizar scheduling y recordatorios

- OAuth con Google Calendar
- Crear eventos automáticamente
- Generar links de Google Meet
- Selector de disponibilidad
- Recordatorios automáticos

Entregables: - Click → Evento en Calendar - Link de Meet generado - Emails de recordatorio

Testing: - Programar sesión → Verificar evento en Google Calendar - Verificar emails 24h/1h antes - Click en Meet link → Video funciona

---

Sprint 3 (Semana 3) - Sala Integrada

Objetivo: Todo en un solo lugar

- Integración Daily.co
- Componente `PracticeRoom`
- Timer de sesión
- Panel de notas
- Feedback post-sesión

Entregables: - Video/audio en la plataforma - No salir de SpeaklyPlan - Sesiones registradas automáticamente

Testing: - Iniciar sesión → Video funciona - Tomar notas durante - Timer cuenta correctamente - Feedback se guarda

---

Sprint 4 (Semana 4) - Pulido

Objetivo: Experiencia premium

- Búsqueda avanzada
- Matching automático (v1)
- Estadísticas detalladas
- Sugerencias proactivas
- Exportar notas

Entregables: - Sistema de recomendaciones - Dashboard analítico - Experiencia "wow"

Implementación Técnica

Dependencias Nuevas

```
{
  "dependencies": {
    "@daily-co/daily-js": "^0.53.0",
    "googleapis": "^126.0.1",
    "date-fns": "^2.30.0",
    "date-fns-tz": "^2.0.0"
  }
}
```

Variables de Entorno

```
# Google Calendar API
GOOGLE_CLIENT_ID=your_client_id
GOOGLE_CLIENT_SECRET=your_client_secret

# Daily.co (Video)
DAILY_API_KEY=your_daily_api_key
DAILY_DOMAIN=yourdomain.daily.co

# Calendly (opcional)
CALENDLY_API_KEY=your_calendly_key
```

Nuevos Modelos Prisma

```
model CalendarIntegration {
  id            String   @id @default(cuid())
  userId        String   @unique
  user          User     @relation(fields: [userId], references: [id])

  provider      String   // "google", "outlook"
  accessToken   String   @db.Text
  refreshToken  String   @db.Text
  expiresAt     DateTime

  createdAt     DateTime @default(now())
  updatedAt     DateTime @updatedAt

  @@index([userId])
}

model UserAvailability {
  id            String   @id @default(cuid())
  userId        String
  user          User     @relation(fields: [userId], references: [id])

  dayOfWeek     Int      // 0-6
  startTime     String   // "09:00"
  endTime       String   // "17:00"
  timezone      String   // "America/Bogota"

  isActive      Boolean  @default(true)

  @@index([userId])
  @@index([dayOfWeek])
}
```

Métricas de Éxito

Sprint 1 (MVP)

- **Adopción:** 30% de usuarios activos envían al menos 1 invitación
- **Conversión:** 50% de invitaciones son aceptadas
- **Sesiones:** Al menos 10 sesiones completadas en la primera semana

Sprint 2 (Scheduling)

- **Calendar:** 80% de usuarios conectan Google Calendar
- **Recordatorios:** 90% de sesiones tienen recordatorios activos
- **Show Rate:** 85%+ de usuarios asisten a sesiones programadas

Sprint 3 (Video)

- **Uso:** 70% de sesiones usan sala integrada vs links externos
- **Duración:** Sesiones duran 20+ minutos en promedio
- **Feedback:** 90%+ dejan feedback post-sesión

Sprint 4 (Pulido)

- **Engagement:** Usuarios hacen 2+ sesiones por semana
- **Retención:** 60%+ regresan después de 1 mes
- **NPS:** Score > 50

Wireframes de Referencia

Página Principal `/practice`

```
Dashboard > Prácticas 1 a 1


  (3)       (5)       (2)
Invitaciones Compañeros  Sesiones   Historial


Invitaciones Recibidas (3)

 María González  [Nivel B2]
"¿Practicamos business English?"
Hace 2 horas
                    [  Aceptar] [  Rechazar]


[+ Nueva Invitación]
```

Modal de Agendar Sesión

```
Programar Sesión con María González


Fecha y Hora

  20 de octubre, 2025
  15:00 - 15:30 COT


Tema (opcional)

Business presentations


Videollamada
Crear link de Google Meet automático
Usar link personalizado

Enviar invitación a Calendar
Recordatorios 24h y 1h antes
```

```
[Cancelar]  [Programar ]
```

Recomendación Final

Ruta Crítica (4 semanas)

Semana 1: MVP UI (funcionalidad básica) - Esto desbloquea TODO el sistema - Usuarios pueden empezar a usar inmediatamente - ROI instantáneo

Semana 2: Google Calendar + Meet - Elimina fricción #1 (scheduling manual) - Recordatorios automáticos - Links generados automáticamente - Experiencia profesional

Semana 3: Sala Daily.co - Elimina fricción #2 (salir de la plataforma) - Métricas automáticas - Todo en un lugar

Semana 4: Pulido + Analytics - Optimizar conversión - Recomendaciones inteligentes - Gamificación avanzada

Herramientas Gratuitas Recomendadas

1. **Google Calendar API**
   - Gratis
   - Confiable
   - Usado mundialmente
   - Mejor opción
2. **Daily.co**
   - 10K min/mes gratis
   - API simple
   - Calidad excelente
3. **Calendly**
   - Solo como alternativa
   - UI externa (no ideal)
   - Menos control

KPIs a Monitorear

- **Invitaciones enviadas** (objetivo: 50+/semana)
- **Tasa de aceptación** (objetivo: 60%+)
- **Sesiones completadas** (objetivo: 30+/semana)
- **Duración promedio** (objetivo: 25+ min)
- **Usuarios activos** (objetivo: 100+)
- **Retorno semanal** (objetivo: 40%+)

Notas Finales

El sistema backend está 90% completo. Solo necesita: 1. Interfaces de usuario (crítico) 2. Integración con Google Calendar (alto valor) 3. Video integrado (mejora experiencia)

Inversión de tiempo total: 4 semanas **Costo adicional:** $0 (todas herramientas gratuitas) **Impacto esperado:** Sistema completamente funcional y profesional

---

Próximo Paso Inmediato: Crear página `/app/practice/page.tsx` con las 4 tabs básicas.

¿Empezamos con el Sprint 1?