

Plan MVP - Sistema de Prácticas 1 a 1

Versión: 1.0

Fecha: 16 de octubre de 2025

Proyecto: SpeaklyPlan - Tutor AI

Índice

1. [Visión General](#)
 2. [Alcance del MVP](#)
 3. [Arquitectura de Datos](#)
 4. [API Endpoints](#)
 5. [Estructura Frontend](#)
 6. [Flujos de Usuario](#)
 7. [Sistema de Notificaciones](#)
 8. [Gamificación e Incentivos](#)
 9. [Fases de Implementación](#)
 10. [Consideraciones Técnicas](#)
-

Visión General

Objetivo Principal

Crear un sistema MVP que permita a los usuarios de SpeaklyPlan practicar inglés en sesiones 1 a 1 con otros usuarios, fomentando el aprendizaje colaborativo y la práctica conversacional.

Propuesta de Valor

- **Para el usuario:** Practicar inglés con compañeros de estudio en tiempo real
- **Para la plataforma:** Aumentar engagement y retención mediante interacción social
- **Para el aprendizaje:** Aprendizaje peer-to-peer y práctica contextual

Características Clave del MVP

- ✓ Invitaciones directas entre usuarios
 - ✓ Notificaciones email y push
 - ✓ Gestión de invitaciones (estados)
 - ✓ Lista de compañeros de práctica
 - ✓ Sala de práctica moderna con video/audio
 - ✓ Historial y seguimiento de prácticas
-

Alcance del MVP

Incluido en MVP

- Sesiones 1 a 1 únicamente
- Invitaciones directas por email/username
- Notificaciones email y push en perfil
- Lista de invitaciones con estados (enviadas/recibidas)
- Lista de compañeros conectados
- Sala de práctica con chat de texto
- Timer de sesión
- Notas y feedback post-sesión
- Historial de prácticas completadas
- Puntos de gamificación por prácticas

Fuera del MVP (v2.0)

- Sesiones grupales (3+ personas)
 - Video/audio integrado (se usarán links externos inicialmente)
 - Calendario y programación avanzada
 - Matching automático por nivel
 - Análisis de pronunciación durante la sesión
 - Exportación de notas
 - Sistema de calificación entre usuarios
-

Arquitectura de Datos

Modelos Prisma

1. PracticeInvitation

```

model PracticeInvitation {
  id          String   @id @default(cuid())
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt

  // Relaciones
  senderId    String
  sender       User    @relation("SentInvitations", fields: [senderId], references: [id], onDelete: Cascade)

  receiverId  String
  receiver     User    @relation("ReceivedInvitations", fields: [receiverId], references: [id], onDelete: Cascade)

  // Datos de la invitación
  message     String?  // Mensaje personal opcional
  status      InvitationStatus @default(PENDING)

  // Metadatos
  respondedAt DateTime?

  @@index([senderId])
  @@index([receiverId])
  @@index([status])
}

enum InvitationStatus {
  PENDING
  ACCEPTED
  REJECTED
  CANCELLED
  EXPIRED
}

```

2. PracticeConnection

```
model PracticeConnection {
  id          String   @id @default(cuid())
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt

  // Relaciones (bidireccional)
  user1Id     String
  user1       User     @relation("ConnectionsAsUser1", fields: [user1Id], references:
[id], onDelete: Cascade)

  user2Id     String
  user2       User     @relation("ConnectionsAsUser2", fields: [user2Id], references:
[id], onDelete: Cascade)

  // Estadísticas
  totalSessions Int @default(0)
  lastSessionAt DateTime?

  // Estado
  isActive     Boolean @default(true)

  @@unique([user1Id, user2Id])
  @@index([user1Id])
  @@index([user2Id])
}
```

3. PracticeSession

```

model PracticeSession {
  id          String   @id @default(cuid())
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt

  // Relaciones
  connectionId String
  connection    PracticeConnection @relation(fields: [connectionId], references: [id],
onDelete: Cascade)

  initiatorId String
  initiator    User      @relation("InitiatedSessions", fields: [initiatorId], refer-
ences: [id])

  partnerId    String
  partner      User      @relation("PartnerSessions", fields: [partnerId], references:
[id])

  // Datos de la sesión
  scheduledFor DateTime? // Null = inmediata
  startedAt    DateTime?
  endedAt      DateTime?
  durationMinutes Int?    // Calculado al finalizar

  status      SessionStatus @default(SCHEDULED)

  // Contenido
  topic      String? // Tema de conversación
  notes      String? // Notas durante la sesión
  externalLink String? // Link a Zoom, Meet, etc.

  // Feedback (post-sesión)
  initiatorFeedback String?
  partnerFeedback   String?
  initiatorRating    Int?    // 1-5
  partnerRating      Int?    // 1-5

  @@index([connectionId])
  @@index([initiatorId])
  @@index([partnerId])
  @@index([status])
  @@index([scheduledFor])
}

enum SessionStatus {
  SCHEDULED
  IN_PROGRESS
  COMPLETED
  CANCELLED
  NO_SHOW
}

```

4. PracticeNotification

```

model PracticeNotification {
  id          String   @id @default(cuid())
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt

  // Relación
  userId      String
  user        User      @relation(fields: [userId], references: [id], onDelete:
Cascade)

  // Contenido
  type        NotificationType
  title       String
  message     String
  actionUrl   String? // Link a la página relevante

  // Estado
  isRead       Boolean @default(false)
  readAt       DateTime?

  // Referencias
  invitationId String?
  sessionId    String?

  @@index([userId])
  @@index([isRead])
  @@index([createdAt])
}

enum NotificationType {
  INVITATION_RECEIVED
  INVITATION_ACCEPTED
  INVITATION_REJECTED
  SESSION_SCHEDULED
  SESSION_STARTING_SOON
  SESSION_COMPLETED
  FEEDBACK_REQUESTED
}

```

5. Actualización del Modelo User

```
model User {
  // ... campos existentes ...

  // Relaciones de práctica
  sentInvitations      PracticeInvitation[] @relation("SentInvitations")
  receivedInvitations  PracticeInvitation[] @relation("ReceivedInvitations")

  connectionsAsUser1   PracticeConnection[] @relation("ConnectionsAsUser1")
  connectionsAsUser2   PracticeConnection[] @relation("ConnectionsAsUser2")

  initiatedSessions    PracticeSession[] @relation("InitiatedSessions")
  partnerSessions      PracticeSession[] @relation("PartnerSessions")

  practiceNotifications PracticeNotification[]

  // Preferencias de práctica
  practiceAvailable    Boolean @default(true)
  practiceTopics        String[] // Array de temas de interés
  practiceTimezone     String?
}
```

API Endpoints

1. Invitaciones

POST /api/practice/invitations

Enviar una nueva invitación

Request:

```
{
  "receiverEmail": "usuario@example.com",
  "message": "¡Hola! ¿Quieres practicar inglés conmigo?"
}
```

Response:

```
{
  "success": true,
  "invitation": {
    "id": "inv_123",
    "senderId": "user_1",
    "receiverId": "user_2",
    "status": "PENDING",
    "message": "¡Hola! ¿Quieres practicar inglés conmigo?",
    "createdAt": "2025-10-16T10:00:00Z"
  }
}
```

Acciones:

1. Validar que el receiver existe
2. Validar que no hay invitación pendiente entre estos usuarios

3. Crear la invitación
4. Crear notificación para el receiver
5. Enviar email al receiver
6. Retornar invitación creada

GET `/api/practice/invitations?type=received|sent`

Obtener invitaciones (enviadas o recibidas)

Response:

```
{
  "success": true,
  "invitations": [
    {
      "id": "inv_123",
      "sender": {
        "id": "user_1",
        "name": "Juan Pérez",
        "email": "juan@example.com",
        "image": "...",
        "level": "Intermediate"
      },
      "receiver": {
        "id": "user_2",
        "name": "María García",
        "email": "maria@example.com",
        "image": "...",
        "level": "Advanced"
      },
      "status": "PENDING",
      "message": "¡Hola! ¿Quieres practicar inglés conmigo?",
      "createdAt": "2025-10-16T10:00:00Z"
    }
  ]
}
```

PATCH `/api/practice/invitations/[id]`

Aceptar o rechazar una invitación

Request:

```
{
  "action": "accept" || "reject" || "cancel"
}
```

Response:


```
{
  "success": true,
  "invitation": {
    "id": "inv_123",
    "status": "ACCEPTED",
    "respondedAt": "2025-10-16T10:05:00Z"
  },
  "connection": {
    "id": "conn_456",
    "user1Id": "user_1",
    "user2Id": "user_2"
  }
}
```

Acciones (si accept):

1. Actualizar estado de invitación
2. Crear PracticeConnection
3. Crear notificación para el sender
4. Enviar email al sender
5. Sumar puntos de gamificación (ambos usuarios)
6. Retornar invitación y conexión

2. Compañeros

GET /api/practice/connections

Obtener lista de compañeros conectados

Response:

```
{
  "success": true,
  "connections": [
    {
      "id": "conn_456",
      "partner": {
        "id": "user_2",
        "name": "María García",
        "email": "maria@example.com",
        "image": "...",
        "level": "Advanced",
        "practiceAvailable": true
      },
      "totalSessions": 5,
      "lastSessionAt": "2025-10-15T14:00:00Z",
      "createdAt": "2025-10-10T10:00:00Z"
    }
  ]
}
```

DELETE /api/practice/connections/[id]

Eliminar un compañero de práctica

Response:

```
{
  "success": true,
  "message": "Compañero eliminado correctamente"
}
```

3. Sesiones

POST /api/practice/sessions

Crear una nueva sesión de práctica

Request:

```
{
  "partnerId": "user_2",
  "scheduledFor": "2025-10-16T15:00:00Z", // Opcional, null = inmediata
  "topic": "Business English",
  "externalLink": "https://meet.google.com/abc-defg-hij"
}
```

Response:

```
{
  "success": true,
  "session": {
    "id": "sess_789",
    "connectionId": "conn_456",
    "initiatorId": "user_1",
    "partnerId": "user_2",
    "status": "SCHEDULED",
    "scheduledFor": "2025-10-16T15:00:00Z",
    "topic": "Business English",
    "externalLink": "https://meet.google.com/abc-defg-hij"
  }
}
```

Acciones:

1. Validar que existe conexión entre usuarios
2. Crear sesión
3. Crear notificación para el partner
4. Si es inmediata, crear notificación de "sesión iniciando"
5. Retornar sesión creada

GET /api/practice/sessions?status=scheduled|in_progress|completed

Obtener sesiones

Response:

```
{
  "success": true,
  "sessions": [
    {
      "id": "sess_789",
      "status": "SCHEDULED",
      "partner": {
        "id": "user_2",
        "name": "María García",
        "image": "...",
      },
      "scheduledFor": "2025-10-16T15:00:00Z",
      "topic": "Business English",
      "externalLink": "https://meet.google.com/abc-defg-hij"
    }
  ]
}
```

PATCH /api/practice/sessions/[id]

Actualizar estado de sesión

Request (iniciar):

```
{
  "action": "start"
}
```

Request (finalizar):

```
{
  "action": "complete",
  "notes": "Great conversation about business presentations",
  "feedback": "María tiene excelente pronunciación y vocabulario",
  "rating": 5
}
```

Request (cancelar):

```
{
  "action": "cancel"
}
```

Response:

```
{
  "success": true,
  "session": {
    "id": "sess_789",
    "status": "COMPLETED",
    "startedAt": "2025-10-16T15:00:00Z",
    "endedAt": "2025-10-16T15:30:00Z",
    "durationMinutes": 30
  },
  "pointsEarned": 50
}
```

Acciones (si complete):

1. Actualizar sesión con notas y feedback
 2. Calcular duración
 3. Actualizar PracticeConnection (totalSessions, lastSessionAt)
 4. Sumar puntos de gamificación
 5. Crear notificación para el partner solicitando feedback
 6. Retornar sesión actualizada
-

4. Historial

GET `/api/practice/history?limit=20&offset=0`

Obtener historial de prácticas

Response:

```
{
  "success": true,
  "history": [
    {
      "id": "sess_789",
      "partner": {
        "id": "user_2",
        "name": "María García",
        "image": "...",
      },
      "topic": "Business English",
      "startedAt": "2025-10-16T15:00:00Z",
      "endedAt": "2025-10-16T15:30:00Z",
      "durationMinutes": 30,
      "notes": "Great conversation...",
      "yourFeedback": "María tiene excelente...",
      "partnerFeedback": "Juan mejoró mucho...",
      "yourRating": 5,
      "partnerRating": 4,
      "pointsEarned": 50
    }
  ],
  "total": 12,
  "stats": {
    "totalSessions": 12,
    "totalMinutes": 360,
    "averageRating": 4.5,
    "totalPartners": 3
  }
}
```

5. Notificaciones

GET /api/practice/notifications?unreadOnly=true

Obtener notificaciones

Response:

```
{
  "success": true,
  "notifications": [
    {
      "id": "notif_999",
      "type": "INVITATION_RECEIVED",
      "title": "Nueva invitación de práctica",
      "message": "Juan Pérez te ha invitado a practicar inglés",
      "actionUrl": "/practica/invitaciones",
      "isRead": false,
      "createdAt": "2025-10-16T10:00:00Z"
    }
  ],
  "unreadCount": 3
}
```

PATCH /api/practice/notifications/[id]/read

Marcar notificación como leída

Response:

```
{
  "success": true,
  "notification": {
    "id": "notif_999",
    "isRead": true,
    "readAt": "2025-10-16T10:30:00Z"
  }
}
```

PATCH /api/practice/notifications/read-all

Marcar todas como leídas

Response:

```
{
  "success": true,
  "updated": 5
}
```

Estructura Frontend

Rutas y Páginas

/practica	
 page.tsx	# Dashboard principal
 invitar	
  page.tsx	# Enviar invitación
 invitaciones	
  page.tsx	# Listar invitaciones
 companeros	
  page.tsx	# Listar compañeros
 sesion	
  [id]	
  page.tsx	# Sala de práctica
 historial	
 page.tsx	# Historial de prácticas

Componentes Principales

1. Dashboard (/practica/page.tsx)

```
// Layout con 4 secciones principales
<PracticeDashboard>
  <StatsOverview />           {/* Estadísticas rápidas */}
  <QuickActions />            {/* Botones: Invitar, Ver invitaciones */}
  <UpcomingSessions />        {/* Próximas sesiones programadas */}
  <RecentActivity />          {/* Actividad reciente */}
</PracticeDashboard>
```

Características:

- Contador de invitaciones pendientes (badge)
 - Contador de notificaciones no leídas
 - Vista de próximas sesiones
 - Acceso rápido a todas las secciones
-

2. Enviar Invitación (/practica/invitar/page.tsx)

```
<InviteForm>
  <EmailSearch />             {/* Buscar usuario por email */}
  <UserPreview />             {/* Preview del usuario encontrado */}
  <MessageInput />            {/* Mensaje personalizado */}
  <SendButton />
</InviteForm>
```

Características:

- Búsqueda de usuarios por email
 - Validación en tiempo real
 - Preview de perfil del usuario
 - Mensaje personalizado opcional
 - Confirmación visual de envío
-

3. Lista de Invitaciones (/practica/invitaciones/page.tsx)

```

<InvitationsManager>
  <TabsSelector>                                {/* Recibidas / Enviadas */}
    <ReceivedTab>
      <InvitationCard>
        <SenderInfo />
        <InvitationMessage />
        <ActionButtons />                      {/* Aceptar / Rechazar */}
      </InvitationCard>
    </ReceivedTab>

    <SentTab>
      <InvitationCard>
        <ReceiverInfo />
        <InvitationStatus />                  {/* Pending / Accepted / Rejected */}
        <CancelButton />
      </InvitationCard>
    </SentTab>
  </TabsSelector>
</InvitationsManager>

```

Características:

- Tabs para recibidas/enviadas
- Estados visuales claros (pending, accepted, rejected)
- Acciones contextuales según estado
- Filtros por estado
- Ordenamiento por fecha

4. Lista de Compañeros (/practica/companeros/page.tsx)

```

<ConnectionsList>
  <SearchBar />                                {/* Buscar compañeros */}
  <ConnectionCard>
    <PartnerInfo />                            {/* Nombre, nivel, foto */}
    <ConnectionStats />                        {/* # sesiones, última sesión */}
    <ActionButtons>
      <StartSessionButton />                  {/* Iniciar práctica ahora */}
      <ScheduleButton />                     {/* Programar sesión */}
      <RemoveButton />                       {/* Eliminar compañero */}
    </ActionButtons>
  </ConnectionCard>
</ConnectionsList>

```

Características:

- Grid/lista responsive
- Indicador de disponibilidad (online/offline)
- Estadísticas de práctica compartida
- Acciones rápidas (iniciar sesión)
- Confirmación para eliminar

5. Sala de Práctica (/practica/sesion/[id]/page.tsx)

```

<PracticeRoom>
  <SessionHeader>
    <PartnerInfo />
    <SessionTimer />           {/* Contador de tiempo */}
    <TopicDisplay />
  </SessionHeader>

  <MainArea>
    <ExternalLinkPrompt />     {/* Link a Meet/Zoom */}
    <NotesEditor />           {/* Editor de notas en tiempo real */}
    <QuickPhrases />          {/* Frases comunes para copiar */}
  </MainArea>

  <SessionControls>
    <StartButton />
    <PauseButton />
    <EndButton />             {/* Abre modal de feedback */}
  </SessionControls>

  <FeedbackModal>
    <RatingStars />           {/* Calificación 1-5 */}
    <FeedbackText />          {/* Feedback escrito */}
    <SubmitButton />
  </FeedbackModal>
</PracticeRoom>

```

Características:

- Timer visible de sesión
 - Editor de notas compartido
 - Botón para abrir link externo (Meet/Zoom)
 - Frases útiles para copiar
 - Modal de feedback al finalizar
 - Guardado automático de notas
-

6. Historial (/practica/historial/page.tsx)

```
<PracticeHistory>
  <StatsCards>
    <TotalSessionsCard />
    <TotalMinutesCard />
    <AverageRatingCard />
    <PartnersCountCard />
  </StatsCards>

  <FilterBar>
    <DateRangePicker />
    <PartnerFilter />
    <SortSelector />
  </FilterBar>

  <HistoryTimeline>
    <SessionHistoryCard>
      <SessionInfo />
      <PartnerInfo />
      <Duration />
      <Notes />
      <FeedbackDisplay />      {/* Feedback mutuo */}
      <RatingsDisplay />      {/* Ratings mutuos */}
    </SessionHistoryCard>
  </HistoryTimeline>
</PracticeHistory>
```

Características:

- Estadísticas agregadas
- Filtros avanzados (fecha, compañero)
- Timeline visual de sesiones
- Ver notas y feedback
- Exportar historial (v2.0)

Componentes Compartidos

NotificationBadge

```
<NotificationBadge>
  <BellIcon />
  <UnreadCount />      {/* Badge con número */}
  <DropdownMenu>
    <NotificationList />
    <MarkAllReadButton />
    <ViewAllLink />
  </DropdownMenu>
</NotificationBadge>
```

Ubicación: Header de todas las páginas



InvitationBadge

```
<InvitationBadge>
  <MailIcon />
  <PendingCount />                                { /* Badge con número */ }
  <Link href="/practica/invitaciones" />
</InvitationBadge>
```

Ubicación: Header de todas las páginas

Flujos de Usuario

Flujo 1: Enviar Invitación

1. Usuario A  /practica/invitar
2. Ingresa email de Usuario B
3. Sistema busca y muestra preview de Usuario B
4. Usuario A escribe mensaje personalizado (opcional)
5. Usuario A  Click "Enviar invitación"

Backend:

6. Crear PracticeInvitation (status: PENDING)
7. Crear PracticeNotification para Usuario B
8. Enviar email a Usuario B
9. Mostrar confirmación a Usuario A

Usuario B recibe:

10. Email con invitación
11. **Notificación** push en perfil (badge)
12. Puede ir a /practica/invitaciones

Flujo 2: Aceptar Invitación

Usuario B:

1. Ve badge de notificación en header
2. Click en notificación → /practica/invitaciones
3. Ve invitación de Usuario A
4. Click "Aceptar"

Backend:


5. Actualizar PracticeInvitation (status: ACCEPTED)
6. Crear PracticeConnection (user1: A, user2: B)
7. Crear PracticeNotification para Usuario A
8. Enviar email a Usuario A
9. Sumar puntos a ambos usuarios (+10 pts)
10. Mostrar confirmación a Usuario B

Usuario A recibe:


11. Email "¡Tu invitación fue aceptada!"
12. Notificación push en perfil
13. Usuario B aparece en su lista de compañeros

Flujo 3: Iniciar Sesión Inmediata


Usuario A:

1.  /practica/companeros
2. Ve a Usuario B en su lista
3. Click "Practicar ahora"
4. Modal: "¿Iniciar sesión inmediata con Usuario B?"
5. Opcional: agregar tema, link externo
6. Click "Iniciar"


Backend:

7. Crear PracticeSession (status: SCHEDULED, scheduledFor: now)
8. Crear PracticeNotification para Usuario B
9. Enviar email a Usuario B
10. Redirigir Usuario A  /practica/sesion/[id]

Usuario B recibe:

11. Email "¡Usuario A quiere practicar ahora!"
12. Notificación push con link directo
13. Click  /practica/sesion/[id]

Ambos en sala:

14. Ven link externo para video **call**
 15. Click "Iniciar sesión" (cambia status a IN_PROGRESS)
 16. Timer comienza
 17. Pueden tomar notas
 18. Click "Finalizar sesión"  Modal de feedback
-

Flujo 4: Completar Sesión y Feedback

Usuario A en sala:

1. Click "Finalizar sesión"
2. Modal de feedback aparece

Modal contiene:

3. "¿Cómo fue tu práctica con Usuario B?"
4. Rating: ★★★★★ (1-5 estrellas)
5. Feedback (textarea): "Escribe un comentario..."
6. Click "Enviar feedback"

Backend:

7. Actualizar PracticeSession:
 - status: COMPLETED
 - endedAt: now
 - durationMinutes: calculado
 - initiatorFeedback: "..."
 - initiatorRating: 5
8. Actualizar PracticeConnection (totalSessions++, lastSessionAt)
9. Sumar puntos a Usuario A (+50 pts por sesión)
10. Crear PracticeNotification para Usuario B ("Deja tu feedback")
11. Mostrar confirmación: "¡+50 puntos!"

Usuario B recibe:

12. Notificación "Deja tu feedback sobre la sesión"
13. → /practica/historial
14. Ve la sesión con badge "Pendiente feedback"
15. Click → Modal de feedback similar
16. Envía feedback y rating

Backend:

17. Actualizar PracticeSession (partnerFeedback, partnerRating)
 18. Sumar puntos a Usuario B (+50 pts)
 19. Ambos pueden ver feedback mutuo en historial
-

Sistema de Notificaciones

Tipos de Notificaciones

Tipo	Trigger	Destinatario	Canales	Acción
INVITA-TION_RECEIVED	Nueva invitación enviada	Receiver	Email + Push	Ver invitaciones
INVITA-TION_ACCEPTED	Invitación aceptada	Sender	Email + Push	Ver compañeros
INVITA-TION_REJECTED	Invitación rechazada	Sender	Push	-
SESSION_SCHEDULED	Sesión programada	Partner	Email + Push	Ver sesión
SESSION_STARTING_SOON	5 min antes de sesión	Ambos	Push	Unirse a sesión
SESSION_COMPLETED	Sesión finalizada por uno	Partner	Push	Dejar feedback
FEEDBACK_REQUESTED	Feedback pendiente	Usuario	Push	Completar feedback

Notificaciones Email

Template Base:

```

<!DOCTYPE html>
<html>
<head>
  <style>
    /* Estilos modernos, responsive */
  </style>
</head>
<body>
  <div class="email-container">
    <header>
      
    </header>

    <main>
      <h1>{{ title }}</h1>
      <p>{{ message }}</p>

      <div class="cta">
        <a href="{{ actionUrl }}" class="button">
          {{ actionText }}
        </a>
      </div>
    </main>

    <footer>
      <p>SpeaklyPlan - Tu Tutor de Inglés con IA</p>
      <a href="#">Configurar notificaciones</a>
    </footer>
  </div>
</body>
</html>

```

Ejemplos:

1. Nueva Invitación:

- Título: "¡Nueva invitación de práctica!"
- Mensaje: "Juan Pérez te ha invitado a practicar inglés juntos"
- CTA: "Ver invitación"

2. Invitación Aceptada:

- Título: "¡Tu invitación fue aceptada!"
- Mensaje: "María García aceptó tu invitación. ¡Ahora son compañeros de práctica!"
- CTA: "Iniciar sesión"

3. Sesión Programada:

- Título: "Sesión de práctica programada"
- Mensaje: "Tienes una sesión con Juan Pérez el 16/10 a las 15:00"
- CTA: "Ver detalles"

Notificaciones Push (En Perfil)

Implementación:

- Componente `<NotificationBadge>` en header
- Badge con contador de notificaciones no leídas
- Dropdown al hacer click

- Lista de últimas 5 notificaciones
- Link “Ver todas” → /practica/notificaciones (v2.0)

Polling:

- Request a `/api/practice/notifications?unreadOnly=true` cada 30 segundos
- Actualizar badge con contador
- Mostrar notificación toast cuando hay nuevas

Optimización (v2.0):

- WebSockets para notificaciones en tiempo real
- Service Workers para notificaciones del navegador

Gamificación e Incentivos

Sistema de Puntos

Acción	Puntos
Enviar primera invitación	+5 pts
Aceptar invitación	+10 pts (ambos)
Completar sesión (15-30 min)	+30 pts
Completar sesión (30-60 min)	+50 pts
Completar sesión (60+ min)	+75 pts
Dejar feedback detallado	+5 pts
Recibir rating 5 	+10 pts
5 sesiones con mismo compañero	+25 pts (bonus)
10 sesiones totales	+50 pts (achievement)
Práctica diaria (3 días seguidos)	+30 pts (streak)

Achievements

```
const practiceAchievements = [
  {
    id: 'first_connection',
    title: 'Primera Conexión',
    description: 'Has conectado con tu primer compañero de práctica',
    icon: '👋',
    points: 10
  },
  {
    id: 'conversation_starter',
    title: 'Conversador Iniciado',
    description: 'Has completado tu primera sesión de práctica',
    icon: '💬',
    points: 20
  },
  {
    id: 'social_butterfly',
    title: 'Mariposa Social',
    description: 'Has practicado con 5 compañeros diferentes',
    icon: '🦋',
    points: 50
  },
  {
    id: 'dedicated_partner',
    title: 'Compañero Dedicado',
    description: 'Has completado 10 sesiones con el mismo compañero',
    icon: '🏆',
    points: 75
  },
  {
    id: 'practice_streak_7',
    title: 'Racha de 7 días',
    description: 'Has practicado durante 7 días consecutivos',
    icon: '🔥',
    points: 100
  }
];
```

Integración con Gamificación Existente

El sistema de prácticas se integra con el sistema de gamificación existente:

```
// En gamification-service.ts
export async function awardPracticePoints(
  userId: string,
  actionType: PracticeActionType,
  metadata?: any
) {
  const points = PRACTICE_POINTS_MAP[actionType];

  await awardPoints(userId, points, {
    source: 'PRACTICE',
    actionType,
    metadata
  });

  // Check for achievements
  await checkPracticeAchievements(userId);

  // Update user statistics
  await updatePracticeStats(userId);
}
```

Fases de Implementación

Fase 1: Base de Datos y API (2-3 días)

Tareas:

- [x] Planificación y diseño
- [] Crear modelos Prisma
- [] Migrar base de datos
- [] Seed de datos de prueba
- [] Implementar API de invitaciones
- [] Implementar API de conexiones
- [] Implementar API de sesiones
- [] Implementar API de historial
- [] Implementar API de notificaciones
- [] Testing de endpoints

Entregables:

- Modelos Prisma creados
- Migración exitosa
- Todos los endpoints funcionando
- Documentación API actualizada

Fase 2: Frontend Core (3-4 días)

Tareas:

- [] Crear estructura de rutas `/practica`
- [] Implementar Dashboard principal
- [] Implementar página “Enviar invitación”
- [] Implementar página “Lista de invitaciones”
- [] Implementar página “Compañeros”

- [] Implementar componentes compartidos
- [] Integrar con APIs
- [] Responsive design
- [] Testing UI

Entregables:

- Todas las páginas funcionando
 - Navegación fluida
 - Design consistente
 - Mobile-friendly
-

Fase 3: Sala de Práctica (2-3 días)

Tareas:

- [] Implementar página de sesión
- [] Timer de sesión
- [] Editor de notas
- [] Link externo (Meet/Zoom)
- [] Controles de sesión
- [] Modal de feedback
- [] Guardado automático
- [] Testing de sala

Entregables:

- Sala de práctica funcional
 - Timer preciso
 - Feedback funcionando
 - UX intuitiva
-

Fase 4: Notificaciones (2 días)

Tareas:

- [] Implementar servicio de email
- [] Templates de email
- [] Componente NotificationBadge
- [] Sistema de polling
- [] Notificaciones toast
- [] Marcar como leído
- [] Testing de notificaciones

Entregables:

- Emails funcionando
 - Notificaciones push funcionando
 - Badge actualizado en tiempo real
 - UX de notificaciones pulida
-

Fase 5: Historial y Stats (2 días)

Tareas:

- [] Implementar página de historial
- [] Timeline de sesiones
- [] Cards de estadísticas
- [] Filtros y ordenamiento
- [] Ver feedback mutuo
- [] Gráficas de progreso
- [] Testing de historial

Entregables:

- Historial completo funcionando
 - Estadísticas precisas
 - Filtros funcionando
 - UX informativa
-

Fase 6: Gamificación (1-2 días)

Tareas:

- [] Integrar con sistema existente
- [] Implementar puntos por acciones
- [] Crear achievements de práctica
- [] Mostrar puntos ganados
- [] Actualizar perfil de usuario
- [] Testing de gamificación

Entregables:

- Puntos funcionando
 - Achievements funcionando
 - Integración completa
 - Feedback visual de puntos
-

Fase 7: Testing y Refinamiento (2-3 días)

Tareas:

- [] Testing end-to-end
- [] Corrección de bugs
- [] Optimización de performance
- [] Mejoras UX según feedback
- [] Documentación final
- [] Preparación para producción

Entregables:

- Sistema estable
 - Bugs críticos resueltos
 - Performance optimizado
 - Documentación completa
-

Consideraciones Técnicas

Seguridad

Validaciones:

- ☒ Solo usuarios autenticados pueden acceder
- ☒ Validar que el usuario es parte de la invitación/conexión/sesión
- ☒ Rate limiting en endpoints de invitaciones (máx 10/día)
- ☒ Sanitizar inputs de usuario (mensajes, notas, feedback)
- ☒ Validar que conexión existe antes de crear sesión

Privacidad:

- ☒ No exponer emails en APIs públicas
 - ☒ Solo mostrar información relevante del compañero
 - ☒ Permitir bloquear usuarios (v2.0)
 - ☒ Permitir eliminar conexiones
 - ☒ GDPR compliance (eliminar datos al borrar cuenta)
-

Performance

Optimizaciones:

- ☒ Índices en campos frecuentemente consultados
- ☒ Paginación en listas (invitaciones, historial)
- ☒ Lazy loading de componentes pesados
- ☒ Caché de datos estáticos (niveles, temas)
- ☒ Debounce en búsquedas
- ☒ Optimistic updates en UI

Carga Inicial:

- Dashboard: <2s
 - Lista de invitaciones: <1s
 - Sala de práctica: <1.5s
-

Escalabilidad

Base de Datos:

- Índices optimizados para queries comunes
- Cleanup de sesiones antiguas (>6 meses)
- Archivado de notificaciones leídas (>30 días)

APIs:

- Rate limiting por usuario
- Caché de respuestas frecuentes
- Compresión de respuestas

Frontend:

- Code splitting por ruta
- Lazy loading de imágenes
- Virtual scrolling en listas largas

Monitoreo

Métricas Clave:

- Número de invitaciones enviadas/día
- Tasa de aceptación de invitaciones
- Número de sesiones completadas/día
- Duración promedio de sesiones
- Tasa de abandono en sala
- Tiempo promedio para dejar feedback

Alertas:

- Caída en tasa de aceptación (<40%)
- Aumento en sesiones canceladas (>20%)
- Errores en envío de emails
- Lag en notificaciones push (>1 min)

Experiencia Móvil

Prioridades:

- ☒ Diseño mobile-first
 - ☒ Touch-friendly (botones grandes)
 - ☒ Scroll suave
 - ☒ Bottom sheet en lugar de modals
 - ☒ Gestos intuitivos (swipe para eliminar)
 - ☒ Notificaciones nativas (v2.0)
-



Métricas de Éxito

KPIs del MVP

Métrica	Objetivo	Medición
Tasa de adopción	30% de usuarios activos usan práctica	% usuarios con ≥ 1 sesión
Tasa de aceptación	60% invitaciones aceptadas	Aceptadas / Enviadas
Sesiones por usuario	2 sesiones/semana	Promedio sesiones / usuario / semana
Duración promedio	25 minutos/sesión	Promedio durationMinutes
Tasa de finalización	80% sesiones completadas	Completed / (Completed + Cancelled)
Tasa de feedback	70% usuarios dejan feedback	Sessions con feedback / Total
Rating promedio	4.0+ estrellas	Promedio de ratings
Retención	50% usuarios regresan en 7 días	% usuarios con ≥ 2 sesiones en 7 días



Roadmap Post-MVP (v2.0)

Features Futuras

- 1. Video/Audio Integrado**
 - WebRTC para video/audio directo
 - Sin necesidad de links externos
 - Grabación de sesiones (opcional)
- 2. Calendario Avanzado**
 - Vista de calendario mensual
 - Programación de sesiones recurrentes
 - Sincronización con Google Calendar
- 3. Matching Inteligente**
 - Sugerencias de compañeros por nivel
 - Matching por intereses/temas
 - Compatibilidad de horarios
- 4. Análisis de Pronunciación**
 - Speech-to-text durante sesión
 - Feedback automático de pronunciación
 - Tracking de progreso fonético

5. Grupos y Clases

- Sesiones 3-5 personas
- Roles (moderador, participante)
- Temas estructurados

6. Sistema de Reputación

- Badges según ratings
- Perfil público de práctica
- Recomendaciones entre usuarios



Notas Finales

Decisiones de Diseño

¿Por qué links externos en MVP?

- Implementar WebRTC es complejo (4-5 días adicionales)
- Meet/Zoom son herramientas familiares
- Permite lanzar MVP más rápido
- v2.0 tendrá video integrado

¿Por qué notificaciones por polling?

- WebSockets requiere infraestructura adicional
- Polling cada 30s es suficiente para MVP
- Más simple de implementar y mantener
- v2.0 usará WebSockets

¿Por qué solo 1 a 1?

- Reduce complejidad de coordinación
- Mejor para conversaciones íntimas
- Matching más simple
- Experiencia más personal
- Grupos en v2.0

Dependencias Externas

```
{
  "nuevas_dependencias": {
    "@radix-ui/react-dialog": "^1.0.5",
    "@radix-ui/react-tabs": "^1.0.4",
    "date-fns": "^2.30.0",           // Ya existe
    "react-hot-toast": "^2.4.1",    // Ya existe
    "react-hook-form": "^7.48.2",   // Ya existe
    "zod": "^3.22.4"                // Ya existe
  }
}
```


Tiempo Total Estimado

- **Fase 1 (Backend):** 2-3 días
- **Fase 2 (Frontend Core):** 3-4 días
- **Fase 3 (Sala):** 2-3 días
- **Fase 4 (Notificaciones):** 2 días
- **Fase 5 (Historial):** 2 días
- **Fase 6 (Gamificación):** 1-2 días
- **Fase 7 (Testing):** 2-3 días

Total: 14-19 días de desarrollo

Equipo Necesario

- **1 Backend Developer:** APIs y base de datos
- **1 Frontend Developer:** UI/UX y componentes
- **1 Full-Stack Developer:** Integración y testing

O alternativamente:

- **1 Senior Full-Stack:** Puede hacer todo en 14-19 días
-



Checklist de Inicio

Antes de comenzar la implementación:

- ☐ Plan aprobado por stakeholders
 - ☐ Base de datos de desarrollo lista
 - ☐ Servidor de email configurado (para notificaciones)
 - ☐ Ambiente de desarrollo configurado
 - ☐ Git branch creado (`feature/practice-mvp`)
 - ☐ Datos de prueba preparados
 - ☐ Diseño UI aprobado (wireframes/mockups)
 - ☐ Documentación API compartida con equipo
-



Conclusión

Este plan MVP establece una base sólida para el sistema de prácticas 1 a 1 de SpeaklyPlan. El enfoque está en:

- ✓ **Simplicidad:** Features esenciales, bien ejecutadas
- ✓ **UX:** Flujo intuitivo y gratificante
- ✓ **Escalabilidad:** Arquitectura preparada para crecer
- ✓ **Engagement:** Gamificación e incentivos sociales

Con este MVP, los usuarios podrán conectar, practicar y mejorar su inglés de forma colaborativa, aumentando significativamente el valor y la retención de la plataforma.

¡Manos a la obra! 🚀

Documento creado por: DeepAgent AI

Para: SpeaklyPlan - Tutor AI

Última actualización: 16 de octubre de 2025