RyugakuTalk 機能仕様書

目次

概要

アプリケーションの目的

RyugakuTalkは、留学中・留学希望者・関係者が質問・共有・交流できる安全なオンラインコミュニティプラットフォームです。

設計思想

1. **安全性の重視**: 留学中の学生の安全を最優先に考慮
2. **多様性の尊重**: 個人アカウントから組織アカウントまで、様々なユーザータイプに対応
3. **視覚的な情報発見**: 地図上での投稿可視化により、地理的な情報を直感的に把握
4. **コミュニティ形成**: 地域・大学・目的別のコミュニティで、より密接な交流を促進
5. **AI支援**: 過去のログを学習したAIによる、日記作成や質問サポート

技術スタック

- **フロントエンド**: Next.js 14 (App Router), React, TypeScript
- **スタイリング**: Tailwind CSS, @tailwindcss/typography
- **バックエンド**: Next.js API Routes, Supabase
- **データベース**: PostgreSQL (Supabase)
- **認証**: Supabase Auth
- **リアルタイム通信**: Supabase Realtime

- **地図**: Leaflet, React-Leaflet
- **Markdown**: react-markdown

認証・ユーザー管理機能

目的

安全で信頼性の高いユーザー認証システムを提供し、個人・組織アカウントの区別と適切な権限管理を実現する。

設計思想

- **柔軟な認証方式**: Google OAuthとメール認証の両方に対応
- **アカウントタイプの多様性**: 個人、教育機関、企業、政府機関の4種類をサポート
- **認証ステータス管理**: 組織アカウントには認証プロセスを実装
- **ゲストユーザー対応**: 閲覧のみ可能なゲストユーザーをサポート

要件定義

ユーザー登録・ログイン

- **実装ファイル**: `src/app/auth/signin/page.tsx`, `src/app/auth/signup/page.tsx`
- **機能**:
  - メールアドレスとパスワードによる登録・ログイン
  - Google OAuthによるログイン
  - パスワードリセット機能
  - メール確認（開発環境では無効化可能）

アカウントタイプ

1. **個人アカウント** (`individual`)
   - デフォルトのアカウントタイプ
   - 全ての基本機能にアクセス可能

    - 学生ステータス（現役留学生、留学経験者、留学希望者）を設定可能

2. **教育機関アカウント** (`educational`)
    - 大学・学校などの教育機関向け
    - 認証が必要（`verification_status: 'verified'`）
    - 安否確認機能へのアクセス
    - コミュニティ管理機能

3. **企業アカウント** (`company`)
    - 企業向け
    - 認証が必要
    - 安否確認機能へのアクセス

4. **政府機関アカウント** (`government`)
    - 政府機関向け
    - 認証が必要
    - 安否確認機能へのアクセス

認証プロセス

- **実装ファイル**: `src/app/verification/request/page.tsx`
- **機能**:
    - 組織アカウントが認証申請を提出
    - 管理者による認証審査
    - 認証ステータス: `unverified`, `pending`, `verified`, `rejected`

データモデル

```
interface User {
  id: string
  email: string
  name: string
  account_type: 'individual' | 'educational' | 'company' | 'government'
  verification_status: 'unverified' | 'pending' | 'verified' | 'rejected'
  organization_name?: string
  student_status?: 'current' | 'experienced' | 'applicant'
  study_abroad_destination?: string
  languages?: string[]
```

```
  icon_url?: string
}
```

タイムライン・投稿機能

目的

留学に関する情報共有、質問、日記投稿を可能にし、ユーザー間の知識共有と交流を促進する。

設計思想

- **カテゴリ別の整理**: 質問、日記、つぶやき、情報共有、公式投稿の5カテゴリ
- **視覚的な区別**: カテゴリごとに異なるアイコンと色で表示
- **Markdown対応**: 日記と公式投稿ではMarkdown形式での記述を可能に
- **画像の柔軟な挿入**: 最大4枚の画像を投稿に添付可能
- **解決済み機能**: 質問投稿に解決済みステータスを追加

要件定義

投稿作成

- **実装ファイル**: `src/app/posts/new/page.tsx`
- **機能**:
  - タイトル、本文、カテゴリの設定
  - タグの追加（正規留学、語学留学、交換留学など）
  - 留学先の選択（複数選択可能）
  - 緊急度設定（質問のみ）
  - Markdownエディタ（日記・公式投稿）
  - 画像アップロード（最大4枚）
  - カバー画像の選択
  - コミュニティへの投稿（オプション）
  - クエストとの紐付け（オプション）

投稿一覧（タイムライン）

- **実装ファイル**: `src/app/timeline/page.tsx`
- **機能**:
  - 最新投稿の表示
  - コミュニティ投稿の表示
  - カテゴリフィルタ（質問、日記、つぶやき、全て）
  - メインカテゴリフィルタ（学習、就労、生活）
  - 詳細カテゴリフィルタ
  - 学生ステータスフィルタ
  - 投稿者情報の表示（アカウントタイプ、学生ステータス）
  - 解決済みバッジの表示
  - カバー画像の表示（日記投稿）

投稿詳細

- **実装ファイル**: `src/app/posts/[id]/page.tsx`
- **機能**:
  - 投稿内容の表示
  - Markdownレンダリング（日記・公式投稿）
  - 画像のインライン表示
  - コメント機能
  - いいね機能
  - 解決ボタン（質問投稿の作成者のみ）
  - 編集・削除機能（作成者のみ）
  - 外部共有機能（X、LINE、URLコピー）
  - 通報機能

投稿カテゴリ

1. **質問** (`question`)
   - 緊急度設定可能（低、通常、高、緊急）
   - 解決済みステータス
   - 解決ボタン
2. **日記** (`diary`)
   - Markdown対応
   - 最大4枚の画像

- カバー画像設定

3. **つぶやき** (`chat`)
   - タイトル不要
   - シンプルなテキスト投稿
4. **情報共有** (`information`)
   - 一般的な情報共有
5. **公式投稿** (`official`)
   - 組織アカウント向け
   - Markdown対応
   - 最大4枚の画像

Markdownエディタ

- **実装ファイル**: `src/components/MarkdownEditor.tsx`
- **機能**:
  - 目次挿入
  - 見出し（大・小）
  - 箇条書きリスト
  - 番号付きリスト
  - 引用
  - 画像挿入（ファイル選択）
  - URL挿入
  - 太字・斜体
  - AIコンシェルジュ連携（日記作成サポート）

データモデル

```
interface Post {
  id: string
  title: string
  content: string
  category: 'question' | 'diary' | 'chat' | 'information' | 'official'
  author_id: string
  community_id?: string
  tags?: string[]
  study_abroad_destinations?: string[]
```

```
  urgency_level?: 'low' | 'normal' | 'high' | 'urgent'
  is_resolved?: boolean
  images?: string[]
  cover_image_url?: string
  likes_count: number
  comments_count: number
  created_at: string
}
```

眺める（マップ）機能

目的

世界中のユーザーの活動（日記、つぶやき）や「困りごと（質問）」を地図上で視覚的に発見できるようにする。

設計思想

- **地理的な可視化**: 実際の地球の地図上に投稿を表示
- **1ユーザー = 1ピン**: ユーザーごとにグループ化し、優先順位に基づいて表示する投稿を決定
- **視覚的な区別**: 投稿タイプと緊急度を色・形状・アニメーションで表現
- **リアルタイム性の強調**: 24時間以内の投稿に特別な表示

要件定義

表示対象

- **原則**: 現役留学生（`student_status: 'current'`）のみ表示
- **位置情報**: `study_abroad_destination`が設定されているユーザー
- **閲覧権限**: 全てのユーザーが閲覧可能

ピン表示の優先順位

1. **未解決の質問**（最優先）
2. **24時間以内の日記**
3. **24時間以内のつぶやき**

4. **その他の投稿**

投稿タイプ別ピン表現

1. **日記** (`diary`)
    - キラキラアニメーション（24時間以内のみ）
    - バッジ: 'D'（白色、太字）
2. **つぶやき** (`chat`)
    - ピンの横に吹き出し表示（先頭10文字）
    - バッジ: 'C'（白色、太字）
3. **質問** (`question`)
    - パルスアニメーション（未解決のみ）
    - 緊急度で色変更
    - バッジ: '?'（白色、太字）

緊急度による色分け

- **低**: 緑系
- **通常**: 青系
- **高**: オレンジ系
- **緊急**: 赤系

フィルタ機能

- **カテゴリ**: 質問、日記、つぶやき
- **緊急度**: 質問のみ（低、通常、高、緊急）
- **国・地域**: 留学先によるフィルタ
- **所属コミュニティ**: コミュニティによるフィルタ

ピンクリック時の挙動

- 投稿詳細ページへの直接遷移
- ホバー時: ポップアップで投稿内容のプレビュー表示

実装ファイル

- **メインページ**: `src/app/map/page.tsx`
- **地図コンポーネント**: `src/components/MapView.tsx`
- **座標データ**: `src/lib/countryCoordinates.ts`

データモデル

```
interface UserPostData {
  user: User
  posts: Post[]
  displayPost: Post // 優先順位に基づく表示投稿
  displayType: 'question' | 'diary' | 'chat' | 'normal'
}
```

コミュニティ機能

目的

地域・大学・目的別のコミュニティを形成し、より密接な交流と情報共有を促進する。

設計思想

- **多様なコミュニティ**: 地域、大学、目的別のコミュニティをサポート
- **メンバーシップ管理**: 承認制・オープン制の選択
- **イベント・クエスト**: コミュニティ内での活動を促進
- **管理者権限**: コミュニティ作成者による管理機能

要件定義

コミュニティ一覧

- **実装ファイル**: `src/app/communities/page.tsx`
- **機能**:
  - 全コミュニティの表示

    - 検索機能
    - カテゴリフィルタ
    - 新規コミュニティ作成

コミュニティ詳細

- **実装ファイル**: `src/app/communities/[id]/page.tsx`
- **タブ構成**:
    1. **タイムライン**: コミュニティ内の投稿
    2. **メンバー**: メンバー一覧、承認待ちメンバー管理
    3. **イベント**: イベント一覧、作成、参加登録
    4. **クエスト**: クエスト一覧、作成、完了申請

コミュニティ作成

- **実装ファイル**: `src/app/communities/new/page.tsx`
- **機能**:
    - コミュニティ名、説明、カテゴリの設定
    - アイコン画像のアップロード
    - メンバーシップタイプの選択（承認制・オープン制）
    - プライバシー設定（公開・非公開）

イベント機能

- **機能**:
    - イベント作成（タイトル、説明、日時、場所）
    - 参加登録・キャンセル
    - 参加者一覧表示
    - イベント編集・削除（作成者のみ）

クエスト機能

- **機能**:
    - クエスト作成（タイトル、説明、報酬額）
    - 完了申請
    - 承認・却下（コミュニティ管理者）
    - クエスト編集・削除（作成者のみ）

データモデル

```typescript
interface Community {
  id: string
  name: string
  description: string
  category: string
  icon_url?: string
  membership_type: 'open' | 'approval'
  privacy: 'public' | 'private'
  created_by: string
  created_at: string
}

interface Event {
  id: string
  community_id: string
  title: string
  description: string
  event_date: string
  location?: string
  created_by: string
  created_at: string
}

interface Quest {
  id: string
  community_id: string
  title: string
  description: string
  reward_amount: number
  created_by: string
  created_at: string
}
```

AIコンシェルジュ機能

目的

過去のログを学習したAIによる回答生成、日記作成サポート、質問サポートを提供する。

設計思想

- **多モード対応**: チャット、日記作成、質問サポートの3モード
- **Markdownエディタ連携**: 日記作成時は直接エディタに挿入

- **過去ログの活用**: 過去の投稿・コメントを学習データとして活用

要件定義

AIコンシェルジュページ

- **実装ファイル**: `src/app/ai/page.tsx`
- **機能**:
  - モード選択（チャット、日記作成、質問サポート）
  - クエリ入力
  - AI回答の表示
  - エラーハンドリング

Markdownエディタ連携

- **実装ファイル**: `src/components/MarkdownEditor.tsx`
- **機能**:
  - ツールバーにAIコンシェルジュボタン
  - クエリ入力ダイアログ
  - 生成された内容をカーソル位置に挿入

APIエンドポイント

- **実装ファイル**: `src/app/api/route.ts`
- **機能**:
  - モードに応じたAI処理
  - 過去ログの検索・分析
  - 回答生成

データモデル

```
interface AIRequest {
  query: string
  mode: 'chat' | 'diary' | 'question'
  limit?: number
  topK?: number
}
```

通知機能

目的

ユーザーに重要な情報を適切なタイミングで通知し、コミュニティへの参加を促進する。

設計思想

- **多様な通知タイプ**: 運営、コミュニティ、緊急、DMなど
- **リアルタイム更新**: Supabase Realtimeによる即座の通知
- **未読管理**: 未読通知数の表示と既読管理

要件定義

通知一覧

- **実装ファイル**: `src/app/notifications/page.tsx`
- **機能**:
  - 通知一覧の表示
  - 通知タイプ別のアイコン表示
  - 既読・未読の区別
  - 通知クリック時の遷移
  - 既読マーク機能
  - リアルタイム更新

通知タイプ

1. **運営からのお知らせ** (`announcement`)
2. **コミュニティイベント** (`community_event`)
3. **コミュニティクエスト** (`community_quest`)
4. **緊急の質問** (`urgent_question`)
5. **安否確認** (`safety_check`)
6. **DM** (`dm`)
7. **コメント** (`comment`)
8. **いいね** (`like`)

通知作成

- **実装ファイル**: `src/lib/notifications.ts`
- **機能**:
  - 通知の作成
  - 通知タイプの検証
  - リンクURLの設定

データモデル

```
interface Notification {
  id: string
  user_id: string
  type: 'announcement' | 'community_event' | 'community_quest' | 'urgent_question'
| 'safety_check' | 'dm' | 'comment' | 'like'
  title: string
  content?: string
  link_url?: string
  is_read: boolean
  created_at: string
}
```

安否確認機能

目的

災害・テロ発生時などに、組織アカウントが現地留学生の安否を確認し、迅速な対応を可能にする。

設計思想

- **組織アカウント専用**: 認証済みの組織アカウントのみが利用可能
- **位置情報連動**: コミュニティ、地理的エリア、特定ユーザーを対象に設定
- **定期送信**: 定期的な安否確認を自動化
- **リアルタイム監視**: 回答状況をリアルタイムで監視

要件定義

安否確認一覧

- **実装ファイル**: `src/app/safety-check/page.tsx`
- **機能**:
  - 安否確認一覧の表示
  - 新規安否確認の作成
  - ステータス表示（進行中、完了、キャンセル）
  - 定期送信の表示

安否確認作成

- **機能**:
  - タイトル、メッセージの設定
  - 対象の選択:
    - コミュニティ
    - 地理的エリア（円形、国・地域）
    - 特定ユーザー
  - 定期送信の設定:
    - 有効/無効
    - 送信間隔（毎日、毎週、毎月）
    - 送信時刻

安否確認詳細

- **実装ファイル**: `src/app/safety-check/[id]/page.tsx`
- **機能**:
  - 安否確認詳細の表示
  - 回答フォーム（安全、危険）
  - 回答一覧（作成者のみ）
  - 未回答者の表示（作成者のみ）
  - 統計情報（送信数、回答数、回答率）（作成者のみ）
  - エクスポート機能（JSON形式）
  - 編集機能（作成者のみ、進行中のみ）

回答機能

- **機能:**
  - 対象ユーザーが回答可能
  - 回答ステータス: 安全、危険
  - メッセージの追加
  - 位置情報の記録（オプション）

データモデル

```
interface SafetyCheck {
  id: string
  created_by: string
  community_id?: string
  title: string
  message: string
  target_area?: {
    type: 'circle' | 'country'
    lat?: number
    lng?: number
    radius?: number
    country?: string
    region?: string
  }
  target_user_ids?: string[]
  status: 'active' | 'completed' | 'cancelled'
  response_count: number
  total_sent: number
  is_recurring?: boolean
  recurrence_type?: 'daily' | 'weekly' | 'monthly'
  recurrence_time?: string
  next_send_at?: string
  created_at: string
  completed_at?: string
}

interface SafetyCheckResponse {
  id: string
  safety_check_id: string
  user_id: string
  status: 'safe' | 'unsafe'
  message?: string
  location?: any
  responded_at: string
}
```

チャット機能

目的

ユーザー間の直接的なメッセージ交換を可能にし、より密接な交流を促進する。

設計思想

- **リアルタイム通信**: Supabase Realtimeによる即座のメッセージ送受信
- **未読管理**: 未読メッセージ数の表示
- **通知連携**: 新規DM受信時に通知を送信

要件定義

チャット一覧

- **実装ファイル**: `src/app/chat/page.tsx`
- **機能**:
  - 会話一覧の表示
  - 最新メッセージのプレビュー
  - 未読メッセージ数の表示
  - ユーザー検索（ID・タグ）

チャット詳細

- **実装ファイル**: `src/app/chat/[userId]/page.tsx`
- **機能**:
  - メッセージ一覧の表示
  - メッセージ送信
  - リアルタイム更新
  - 送信時刻の表示
  - 既読管理

データモデル

```
interface Message {
  id: string
  sender_id: string
  receiver_id: string
```

```
  content: string
  is_read: boolean
  created_at: string
}
```

プロフィール機能

目的

ユーザーの情報を表示・編集し、コミュニティ内での信頼関係を構築する。

設計思想

- **詳細な情報表示**: 留学先、専攻、言語、学生ステータスなど
- **投稿履歴**: ユーザーの投稿一覧を表示
- **統計情報**: 投稿数、コメント数、いいね数など

要件定義

プロフィール表示

- **実装ファイル**: `src/app/profile/[id]/page.tsx`
- **機能**:
  - ユーザー情報の表示
  - アカウントタイプバッジ
  - 学生ステータスバッジ
  - 投稿一覧
  - 統計情報

プロフィール編集

- **実装ファイル**: `src/app/profile/[id]/edit/page.tsx`
- **機能**:
  - 名前、自己紹介の編集
  - 留学先の設定

- 専攻の設定
- 言語の設定
- アイコン画像のアップロード
- 学生ステータスの設定

データモデル

```
interface Profile {
  id: string
  email: string
  name: string
  bio?: string
  university?: string
  study_abroad_destination?: string
  major?: string
  languages?: string[]
  student_status?: 'current' | 'experienced' | 'applicant'
  icon_url?: string
  account_type: 'individual' | 'educational' | 'company' | 'government'
  verification_status: 'unverified' | 'pending' | 'verified' | 'rejected'
  organization_name?: string
  contribution_score: number
}
```

UI/UX設計

目的

直感的で使いやすく、レスポンシブなユーザーインターフェースを提供する。

設計思想

- **モダンなデザイン**: グラデーション、シャドウ、アニメーションを活用
- **レスポンシブ対応**: モバイル、タブレット、デスクトップに対応
- **アクセシビリティ**: 適切なコントラスト、フォーカス表示、ARIA属性
- **一貫性**: 統一されたデザインシステム

要件定義

ナビゲーション

- **実装ファイル**: `src/components/Header.tsx`, `src/components/TopTabNavigation.tsx`, `src/components/BottomTabNavigation.tsx`
- **機能**:
  - デスクトップナビゲーション（xl以上）
  - ハンバーガーメニュー（xl未満）
  - タブナビゲーション（タイムライン、眺める、コミュニティ、安否確認）
  - モバイル用ボトムナビゲーション

レスポンシブ対応

- **ブレークポイント**:
  - `sm`: 640px
  - `md`: 768px
  - `lg`: 1024px
  - `xl`: 1280px
- **画面サイズに応じた表示**:
  - ボタンのテキストを非表示（lg未満）
  - ハンバーガーメニューへの切り替え（xl未満）

コンポーネント

1. **AccountBadge**: アカウントタイプの表示
   - 個人: グレー
   - 組織: ゴールドグラデーション（#B39855, #FFF9E6）
2. **StudentStatusBadge**: 学生ステータスの表示
   - 現役留学生: 青
   - 留学経験者: 青
   - 留学希望者: 青
3. **UserAvatar**: ユーザーアイコンの表示
   - 画像がある場合: 画像表示
   - 画像がない場合: イニシャル表示
4. **FloatingActionButton**: 投稿作成のクイックアクセス

    - 質問、日記、つぶやき、チャットへのリンク

5. **AIConciergeButton**: AIコンシェルジュへのクイックアクセス

データベーススキーマ

主要テーブル

profiles

ユーザープロフィール情報

```
CREATE TABLE profiles (
  id UUID PRIMARY KEY REFERENCES auth.users(id),
  email TEXT,
  name TEXT NOT NULL,
  bio TEXT,
  university TEXT,
  study_abroad_destination TEXT,
  major TEXT,
  languages TEXT[],
  student_status TEXT CHECK (student_status IN ('current', 'experienced',
'applicant')),
  icon_url TEXT,
  account_type TEXT DEFAULT 'individual' CHECK (account_type IN ('individual',
'educational', 'company', 'government')),
  verification_status TEXT DEFAULT 'unverified' CHECK (verification_status IN
('unverified', 'pending', 'verified', 'rejected')),
  organization_name TEXT,
  contribution_score INTEGER DEFAULT 0,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);
```

posts

投稿情報

```
CREATE TABLE posts (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  title TEXT NOT NULL,
  content TEXT NOT NULL,
  category TEXT NOT NULL CHECK (category IN ('question', 'diary', 'chat',
'information', 'official')),
```

```
  author_id UUID REFERENCES profiles(id) ON DELETE CASCADE,
  community_id UUID REFERENCES communities(id) ON DELETE SET NULL,
  tags TEXT[],
  study_abroad_destinations TEXT[],
  urgency_level TEXT CHECK (urgency_level IN ('low', 'normal', 'high', 'urgent')),
  is_resolved BOOLEAN DEFAULT FALSE,
  images TEXT[],
  cover_image_url TEXT,
  likes_count INTEGER DEFAULT 0,
  comments_count INTEGER DEFAULT 0,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);
```

communities

コミュニティ情報

```
CREATE TABLE communities (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  name TEXT NOT NULL,
  description TEXT,
  category TEXT,
  icon_url TEXT,
  membership_type TEXT DEFAULT 'open' CHECK (membership_type IN ('open',
'approval')),
  privacy TEXT DEFAULT 'public' CHECK (privacy IN ('public', 'private')),
  created_by UUID REFERENCES profiles(id) ON DELETE CASCADE,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  updated_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);
```

notifications

通知情報

```
CREATE TABLE notifications (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  user_id UUID REFERENCES profiles(id) ON DELETE CASCADE NOT NULL,
  type TEXT NOT NULL CHECK (type IN ('announcement', 'community_event',
'community_quest', 'urgent_question', 'safety_check', 'dm', 'comment', 'like')),
  title TEXT NOT NULL,
  content TEXT,
  link_url TEXT,
  is_read BOOLEAN DEFAULT FALSE,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);
```

safety_checks

安否確認情報

```
CREATE TABLE safety_checks (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  created_by UUID REFERENCES profiles(id) ON DELETE CASCADE NOT NULL,
  community_id UUID REFERENCES communities(id) ON DELETE SET NULL,
  title TEXT NOT NULL,
  message TEXT NOT NULL,
  target_area JSONB,
  target_user_ids UUID[],
  status TEXT NOT NULL DEFAULT 'active' CHECK (status IN ('active', 'completed',
'cancelled')),
  response_count INTEGER DEFAULT 0,
  total_sent INTEGER DEFAULT 0,
  is_recurring BOOLEAN DEFAULT FALSE,
  recurrence_type TEXT CHECK (recurrence_type IN ('daily', 'weekly', 'monthly')),
  recurrence_time TIME,
  next_send_at TIMESTAMP WITH TIME ZONE,
  created_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  completed_at TIMESTAMP WITH TIME ZONE
);
```

safety_check_responses

安否確認回答情報

```
CREATE TABLE safety_check_responses (
  id UUID DEFAULT gen_random_uuid() PRIMARY KEY,
  safety_check_id UUID REFERENCES safety_checks(id) ON DELETE CASCADE NOT NULL,
  user_id UUID REFERENCES profiles(id) ON DELETE CASCADE NOT NULL,
  status TEXT NOT NULL CHECK (status IN ('safe', 'unsafe')),
  message TEXT,
  location JSONB,
  responded_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
  UNIQUE(safety_check_id, user_id)
);
```

まとめ

RyugakuTalkは、留学コミュニティの形成と安全な情報共有を目的とした包括的なプラットフォームです。個人アカウントから組織アカウントまで、様々なユーザータイ

プに対応し、投稿、コミュニティ、AI支援、安否確認など、多様な機能を提供しています。

主要な特徴

1. **多様なアカウントタイプ**: 個人、教育機関、企業、政府機関
2. **視覚的な情報発見**: 地図上での投稿可視化
3. **コミュニティ形成**: 地域・大学・目的別のコミュニティ
4. **AI支援**: 日記作成や質問サポート
5. **安全機能**: 安否確認システム
6. **リアルタイム通信**: チャット、通知の即座の更新

今後の拡張可能性

- 多言語対応の拡充
- 動画投稿機能
- ライブストリーミング
- モバイルアプリの開発
- 分析・レポート機能