アプリ名称（仮）：どこ行く？（飲食店意思決定サポートツール）
バージョン：v1.0（仕様初版）
作成日：YYYY-MM-DD

【1. システム概要】
本システムは、複数人で外食先を決定する際に発生する心理的負荷（意見調整、否定反応、気まずさ）を低減し、参加者全員が納得しやすい飲食店を自動で提案する Web アプリケーションである。
ホストが発行した共有URLにアクセスするだけで参加可能（ログイン不要）。
参加者は候補店に対して簡易評価を行い、その評価データを基に不満が出にくい店を算出し提示する。

【2. 用語定義】
ホスト: ルーム作成者。条件設定を行う権限を持つ。
参加者: 招待URLからアクセスしたユーザー。ニックネームのみで参加。
評価:「超アリ / 良い / 悪い」の 3 段階評価。
理由タグ:「高い」「遠い」「混みそう」等の任意選択要素。
A（特異ユーザー）: 評価傾向から内部的に推定される、不満を出しやすいユーザー。UIには表示しない。

【3. システム機能一覧（機能要件）】
3.1 ルーム管理機能
- ホストがルームを作成できる
- ルームURLを生成し共有可能
- ルーム名は任意入力

3.2 参加機能
- 招待URLからアクセス
- ニックネーム入力で参加（ログイン不要）
- ブラウザ単位でユーザー識別（ローカルストレージ）

3.3 条件設定機能（ホストのみ）
- エリア設定（現在地 / 手動入力）
- 気分設定（複数選択）
- 予算設定
- 距離設定
-「店候補を取得」処理
- 参加者側は閲覧のみ

3.4 店候補取得機能
- 外部APIから店情報を取得
- 表示項目：写真、店名、ジャンル、距離、予算
- 店詳細表示（写真、メニュー、口コミ、営業時間）

3.5 評価機能
- 参加者は候補店ごとに評価を行う
- 評価形式：「超アリ / 良い / 悪い」

- 任意理由タグを選択可能
- 評価進捗を表示

3.6 集計・提案機能
- 全員の評価データを集計
- Aユーザー推定ロジックで不満リスクの低い店を抽出
- ランキング表示（1〜3位）
-「この店に決定」機能
- 地図アプリ・予約サイトへの誘導

【4. 非機能要件】
- ログイン不要
- 同時アクセスに対応
- スマホ最適化（幅375px）
- 個人情報を扱わない
- A推定結果は内部処理のみ使用

【5. 内部アルゴリズム仕様（A推定）】
A推定基準例:
- ネガティブ評価が多い
- 多数肯定に対して1人だけ否定
-「超アリ」が極端に少ない
- 評価の一貫性が低い

A反映ロジック:
- Aが否定した店のスコアを下げる
- Aが否定していない店のスコアを上げる
- 重みはA推定確信度によって調整
- 最終結果は「不満リスクの低い順」で提示

【6. 画面仕様（UI要件）】
- 1画面1操作のシンプル構造
- 初見ユーザーが迷わない導線
- 心理負担の少ない文言
- 押しやすい評価ボタン

【7. 想定ユーザー・ユースケース】
ユーザー:
- 友人グループ
- 同僚
- 家族

ユースケース:
- その場で外食先を決める
- 気分がバラバラなグループで調整
- 人間関係を悪くせずに合意形成したい

【8. 制約事項】
- 店情報はAPI依存
- 位置情報はユーザー許可が必要
- A推定は誤判定の可能性あり（UIには表示しない）

【9. リスク・対策】
- A推定誤判定 → 重みを弱める
- 店情報誤り → ソースを明記、再取得可能
- 評価偏り → 最低評価数を設定
- 途中離脱 → 進捗通知など検討

【10. 将来拡張（案）】
- 予約連携
- チャット機能
- AIによる嗜好学習
- 過去履歴管理

A特定のロジック（暫定）

1. 評価スコアの数値化
ユーザーが各店舗に対して行う評価を、以下の数値に変換する：
- 「超アリ」：**5**
- 「良い」　：**3**
- 「悪い」　：**-1**

このスコアは店舗スコア算出などにも用いるが、A特定ロジックでは主に「件数ベースの割合」に利用する。

2. A判定対象ユーザーの条件（最低評価件数）
各ユーザー u について、評価した店舗の件数を $\mathbf{N_{(u)}}$ とする。
- $N_{(u)} < 15$ のユーザーは、サンプル数が不足しているとみなし、
  A候補の計算対象から除外する。
- $N_{(u)} \geq 15$ のユーザーのみ、A候補として評価する。

3. 各ユーザーの指標計算
ユーザー u について、以下を定義する。
- bad_count_u
  「悪い」と評価した店舗数
- super_count_u
  「超アリ」と評価した店舗数

これを用いて、次の指標を計算する：
3.1 ネガティブ率

neg_rate_u = bad_count_u / N_u
- 値域：0〜1

- 高いほど「悪い」が多いユーザー。

3.2 超アリ率

super_rate_u = super_count_u / N_u

- 値域：0〜1
- 低いほど「強い肯定が少ない」ユーザー。

4. A スコアの算出

ユーザー u に対して、以下の式で **A スコア A_score_u** を定義する。

A_score_u = 0.6 * neg_rate_u
+ 0.4 * (1 - super_rate_u)

- neg_rate_u が大きいほど A_score_u は増加
- super_rate_u が小さいほど A_score_u は増加
- 理論上の最大値は 1.0
  - neg_rate_u = 1.0（全件「悪い」）
  - super_rate_u = 0.0（「超アリ」ゼロ）のとき

重みは以下の方針に基づく：

- ネガティブ率（悪いの多さ）を最重視：**0.6**
- 超アリ率の低さ（強い肯定の少なさ）を次に重視：**0.4**

5. A 候補の選定方法

1. まず、$N_{(u)} \geq 15$ を満たすユーザーのみを候補とし、各ユーザーの A_score_u を計算する。
2. その中で最大の A スコアを持つユーザー u* を求める：

A_score_max = max_u A_score_u
u× = argmax_u A_score_u

6. 閾値による A の有無の判定

A が存在するかどうかを判定するための閾値 T を、**暫定値として 0.7** に設定する。

判定ルール：

もし A_score_max >= 0.7 なら
　ユーザー u* を「A」として扱う
そうでなければ
　このルームには「A はいない」とみなす

- T=0.7 はテストで調整する前提の**仮値**であり、実データを見ながら後から変更可能とする。

7. 運用上のイメージ（直感チェック）

- 例 1：
  - $N_{(u)} \geq 15$
  - neg_rate_u = 0.5（半分が「悪い」）
  - super_rate_u = 0.0（「超アリ」ゼロ）
    →

A_score_u = 0.6*0.5 + 0.4*(1-0) = 0.3 + 0.4 = 0.7
→ ギリギリ A 判定になるレベル

- 例 2：
  - neg_rate_u = 0.2
  - super_rate_u = 0.3
    →
    A_score_u = 0.6*0.2 + 0.4*(1-0.3) = 0.12 + 0.28 = 0.40
    → A とはみなさない、通常の好みのあるユーザー

他に決める必要がある方式

店の総合スコア算出方式（単純平均か、別の重み付けか）
→ とりあえず単純平均

「評価が十分集まった」とみなす条件（最低評価数・タイムアウト）
→ メンバーが共通の１５軒を評価した場合。（数値は変更してもよさそう、選択制？）

最終候補の絞り込みロジック（A のペナルティも含めて）
→ 上位何件を出すかも選択制かな
　A ペナルティは総合点からの大幅減点

同点のときのタイブレークルール
→ 距離、予算などの評価軸を最初に選択させる

未評価の扱い
→ 作らない予定（スワイプ形式がいいけど、３段階評価が難しいか、、

条件変更時の再計算ルール
→ 未定

自動決定か、人が最後に決めるのか
→ 最後は自動決定　（順位は一応見れるようにする、、？）