推薦アプリ実装ノート（Django＋Next.js）

📦 Djangoからデータ送信

views.py

- データ保管用の `/data` フォルダから `result/` のデータをコピーをする。
- 公開用のデータが入った `/media` フォルダを作成し、そこに `result_data.csv`, `result_map.html` を保存。
    - → `/data` の中はあくまで「内部処理用のデータ置き場」で、直接URLからアクセスされるべきではない。
    - → `/media` フォルダは「Web経由で公開してもOKな場所」として設定する。
- 結果データを CSV ファイルとして経由させてからデータを MySQL で保存。
    - Application のデータ処理ロジックで CSV ファイルを生成。
    - Django の view で CSV ファイルを読み取り、MySQL へ保存。
    - → お互いの依存性を弱めるため。
- 推薦ロジックは自作したもので、パラメータの調整や新しい指標（混雑度合など）の追加に対応しやすくするため、CSVという形でロジックとWeb部分をゆるくつなぐ。
    - → ロジック部分だけを簡単に改良・再利用可能にしている。
- 現在はスクレイピングでデータ収集しているが、時間がかかりロジックが複雑なため、将来的には他サービスのAPIに切り替える可能性を見込んでいる。
    - → ロジック部分とWebをCSVで分離し、どんなデータ取得手段にも対応できる柔軟な設計にしている。
- 出力形式の異なる複数のAPIやデータソースに対応するために、まず中間データ（CSV）に統一してから処理を行う構成。
    - → 将来的に他のデータ取得方法（例：別のAPIやスクレイピングツールなど）へ切り替える際にも、ロジック全体を大きく書き換える必要がなくなり、切り替えコストを低く抑えることができる。

🔗 DjangoのURLルーティング

- 特にメモすべきものはなし。これまでの勉強内容を参照するだけでよい。

⚙ Next.jsのデータ受け取り

`lib/api.ts`

- `API_URL` を環境（開発か本番、Dockerかブラウザか）によって自動で切り替える。
- `runTabelog()`
    - ユーザーが入力した値を Backend のアプリケーションが認識できるようにデータを加工する。
- `formatToApplicationPayload`
    - API でデータを送信・Djangoからレスポンスを受け取る形式に整形。

🖥️ Next.jsの画面表示

ページ遷移のやり方

```
import { useRouter } from "next/navigation";

export default function ResultPage() {
  const router = useRouter();

  return (
    <button onClick={() => router.push("/form")}>⬅ 検索画面に戻る</button>
  );
}
```

✏ データ入力画面

- `/components/` で分離

✅ 確認画面（`/confirm/page.tsx`）

- ページをまたいでデータ（オブジェクト）を受け取り・送出する必要あり → `sessionStorage` を使用。

sessionStorage の使い方

```
// オブジェクトの保存
const data = { name: "チャット", age: 5 };
sessionStorage.setItem("user", JSON.stringify(data));

// オブジェクトの取り出し
const raw = sessionStorage.getItem("user");
const user = raw ? JSON.parse(raw) : null;
```

- `raw` が null なら null、文字列ならオブジェクトに変換。
- 表示部分は `components/ConfirmSummary` で管理。
    - sessionStorage で取得したデータを `data` に格納し、表示コンポーネントに渡す。
- `/components/` に分離。

📊 結果画面

- `/components/` に分離。