

卒業論文
デジタル版しおりアプリの開発

関西学院大学工学部
情報工学課程 西谷研究室
37022463 山本果音

2026年3月

概 要

旅行アプリ「旅しお」では旅行計画を簡単に立てることが可能であるが、予定作成時の情報整理や位置情報の提示が不十分であり、細かい旅程を決めるまで一貫して利用するのには向いていない。本研究では、旅行前の計画段階において、ユーザーがスムーズに情報を整理・共有できることを目的とし、デジタル版しおりの開発を行った。具体的な特徴として、アカウント機能や共有機能、旅行名や説明文をキーとした検索機能、旅行日付に基づく時系列整理、目的地の地図表示機能、移動時間表示機能、訪問順序入れ替え機能、チェックリスト機能を実装し、新たな予定を立てやすくした。結果としてユーザーは旅行計画を効率的に立てることができ、一貫した利用が可能になった。

目次

第1章 序論	4
第2章 手法	5
2.1 開発手法	5
2.1.1 Django	5
第3章 結果と考察	7
3.1 アプリ機能	7
3.1.1 アカウント機能	8
3.1.2 共有機能	10
3.1.3 旅行作成画面	12
3.1.4 旅程の視覚化管理	13
3.1.5 UI 設計とスタイルの工夫	14
3.1.6 旅程の編集・調節機能	15
3.1.7 検索機能	17
3.1.8 チェックリスト機能	18
第4章 まとめ	21

目 次

3.1	アプリ機能一覧.	7
3.2	ログイン画面.	8
3.3	新規登録画面.	9
3.4	ログアウト画面.	9
3.5	リンク作成画面.	10
3.6	共有 URL 表示画面.	11
3.7	パスワード入力画面.	11
3.8	パスワード入力画面.	12
3.9	共有表示画面.	12
3.10	旅行作成画面.	13
3.11	色とアイコンで視覚的に整理された旅行一覧画面.	14
3.12	全体から 1 日目のタブを押したときの変化.	15
3.13	車による移動時間の自動算出結果を表示した画面.	17
3.14	検索結果画面.	18
3.15	施設名の入力画面.	18
3.16	チェックリスト設定画面.	19
3.17	チェックリスト追加画面.	19
3.18	チェックリストの管理画面.	20

目 次

第1章 序論

修学旅行などのイベントでは、参加者に対して事前にしおりが配布されることが多い。しおりは旅程の把握や所持品の確認、集合時間の共有などを目的とした重要な情報媒体であり、旅行の準備や当日の行動において欠かせない役割を果たしている。近年では、スマートフォンの普及により、紙媒体に代わって Web サービスを活用した旅行情報の閲覧・共有のニーズが高まっており、「旅しお」[1] のような Web アプリも登場している。しかし、これらのサービスにはいくつかの課題が存在する。

1つ目は、旅行名や説明文に対するキーワード検索機能が存在しないため、ユーザーはスクロール操作によって目的のしおりを探す必要がある点である。

2つ目は、地図表示機能がないため、目的地の位置関係や全体像を視覚的に理解することが難しく、利用者は外部ツールや手作業で補完しなければならない。

3つ目は、移動時間の自動算出機能が存在しないため、各目的地間の所要時間をユーザー自身が都度手動で調べる必要がある点である。これにより、旅行全体のスケジュールを立てる際に手間がかかり、計画の精度や効率性が低下する可能性がある。

これらの課題は、旅行計画の際にユーザーにとって大きな障壁となっており、情報の整理や視覚的な理解の低下が考えられる。そこで本研究では、視覚的な理解と効率的に旅行計画を行える「デジタル版しおり」として機能する旅行アプリの開発を目指す。

第2章 手法

2.1 開発手法

本研究では、限られた開発期間内で効率的に成果を上げるために、機能ごとに優先順位を設定し、それぞれについて計画・設計・実装・テストの工程を小刻みに繰り返す手法を採用した。各段階で得られた知見や、研究室内のメンバーからのフィードバックをもとに、仕様やUIの見直しを行いながら、柔軟に改善を重ねた。これにより、利用者の視点を反映した機能の追加や調整が可能となり、アプリケーションの完成度と実用性を段階的に高めることができた。

2.1.1 Django

開発環境として、Python で実装されたサーバーサイド Web アプリケーションフレームワークである Django を選定した [2]。Django を選定した理由は以下の3つである。

1つ目は、ユーザー認証や管理画面、データベース操作などの機能が初期状態で組み込まれており、追加設定を行わずとも迅速に開発を開始できる点である。これにより、開発初期の環境構築にかかる時間を大幅に削減することが可能となった。

2つ目は、データベース操作をすべて Python で記述できるため、学習コストが低い点である。Django では、モデルの定義からマイグレーションの実行までを一貫して Python で記述できるため、SQL の詳細な知識がなくても複雑なデータ構造の設計や変更が容易に行える。

3つ目は、セキュリティ対策がフレームワークに標準で備わっている点である。Django はクロスサイトスクリプティング (XSS) や SQL インジェクションなどの脆弱性に対して、初期状態で対策が施されている。そのため、開発者が個別にセキュリティ機能を実装する必要がなく、安全性の高い Web アプリケーションを効率的に構築することができた。

また,実装にはPython, HTML, JavaScript, CSS を使用し,UI ライブラリとしては Bootstrap を採用した. 開発にあたっては Copilot[3] を参照しながら,開発を行った.

第3章 結果と考察

3.1 アプリ機能

今回、開発したアプリでは大きく分けて7つの機能を実装した。それぞれの機能は図3.1の通りである。それぞれの機能について次節以降で記述する。

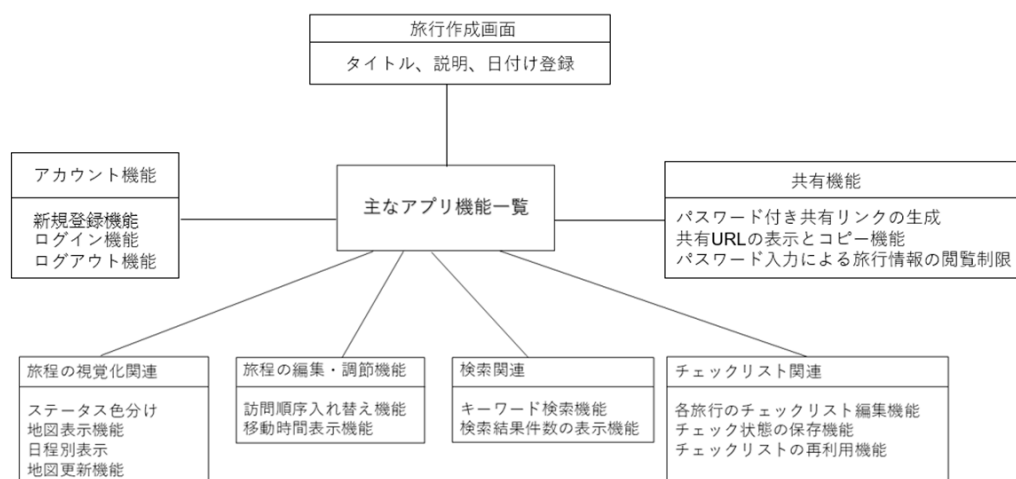


図 3.1: アプリ機能一覧.

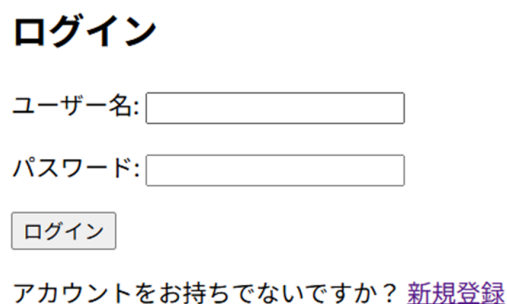
Django は MTV（Model-Template-View）アーキテクチャに基づいており、それぞれ「アプリ内で扱う情報の構造を定義し、データベースとの連携を担うモデル層（Model）」「ユーザーの操作に応じて処理を行い、適切なデータや画面を返すビュー層（View）」「画面表示を構成するテンプレート層（Template）」の3つの役割に分かれている。たとえば、旅行作成機能では、ユーザーが入力した旅行タイトル・説明・日付などの情報を、テンプレートを通じてフォームとして表示し、ビューがその入力を受け取って処理し、モデルを介してデータベースに保存する。保存された情報は、旅行一覧や詳細ページで再びビュー

によって取得され、テンプレートに渡されて HTML として出力される。この一連の流れにより、旅行情報の登録・閲覧・編集・削除といった基本的な CRUD 操作が実現されている。このように Django の MTV アーキテクチャは、機能ごとの責任範囲を明確に分離することで、コードの可読性や再利用性を高めている。

3.1.1 アカウント機能

本アプリケーションでは、ユーザー認証機能としてログイン機能と新規登録機能を実装した。

図 3.2 に示したとおり、ログイン画面では登録済みのユーザーがユーザー名とパスワードを入力することで、個別の旅行プランやチェックリストにアクセスできるようになっている。



ログイン

ユーザー名:

パスワード:

アカウントをお持ちでないですか? [新規登録](#)

図 3.2: ログイン画面。

一方、まだ登録を行っていないユーザーに対しては、図 3.4 のように新規登録画面を用意しており、ユーザー名・パスワード・確認用パスワードの 3 項目を入力することでアカウントを作成できる仕組みとした。パスワードの確認入力を設けることで、入力ミスによるログイン失敗を防ぎ、ユーザー体験の向上を図っている。

図 3.5 に示したとおりログアウト機能も実装しており、ユーザーがログアウト操作を行うと、確認ダイアログが表示されるようになっている。このダイアログでは「ログアウトしますか?」というメッセージとともに、OK またはキャンセルの選択肢が提示される。専用のログアウトページは設けていないが、ユーザーの意図しないログアウトを防ぐための確認ステップを導入することで、操作ミスの抑制とユーザー体験の向上を図っている。

新規登録

ユーザー名: この項目は必須です。半角アルファベット、半角数字、@/./+/_/ 150文字以下にしてください。

パスワード:

- あなたの他の個人情報と似ているパスワードにはできません。
- パスワードは最低 8 文字以上必要です。
- よく使われるパスワードにはできません。
- 数字だけのパスワードにはできません。

パスワード(確認用): 確認のため、再度パスワードを入力してください。

すでにアカウントをお持ちですか？ [ログイン](#)

図 3.3: 新規登録画面.

127.0.0.1:8000 の内容

ログアウトしますか？

図 3.4: ログアウト画面.

3.1.2 共有機能

本アプリでは、図 3.6 に示したとおりユーザーが作成した旅行計画を他者と共有するためのリンク生成機能を実装した。この機能により、ユーザーはパスワード付きの専用 URL を発行し、家族や友人と旅行情報を安全かつ手軽に共有できる。共有リンクの作成時には、ユーザーがあらかじめ用意された選択肢から有効期間を選択できるように設計しており、設定された期限を過ぎたリンクに対しては、システムが自動的に無効と判断し、アクセスを制限する仕組みを導入している。

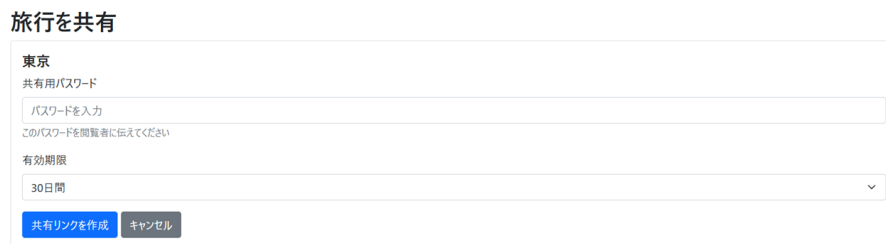


図 3.5: リンク作成画面。

図 3.7 に示したとおり、ユーザーがパスワードを入力して共有リンクを作成すると、専用 URL が画面上に表示される。また、ワンクリックで URL をコピーできるボタンを設置しており、手動入力によるミスを防ぎつつ、スムーズな共有を実現している。コピー完了後には、コピーしましたと表示されるため、操作結果が視覚的に分かりやすく、ユーザーの安心感にもつながっている。

図 3.8 のように、このリンクを受け取った相手は、パスワードを入力することで旅行計画の詳細を閲覧できる。パスワードが正確に入力された場合は共有された旅行画面をみることができ。

パスワードが正確に入力されなかった場合は図 3.9 のようにメッセージが表示される。

観光場所の追加や削除などにより情報を編集した場合でも、リンクが有効期限内であれば常に最新の旅行情報が反映されるようになっている。これにより、共有先のユーザーも常に最新の旅程を確認でき、変更内容の伝達漏れを防ぐことができる。

共有リンクが作成されました

以下のURLを閲覧者に共有してください

共有URL

http://127.0.0.1:8000/shared/Od1aUNCSD8k4R8siH96bXh2tbvPIEkIsiMhWk7f8szU/

コピー

旅行詳細に戻る

旅行一覧に戻る

図 3.6: 共有 URL 表示画面.

共有旅程の閲覧

東京

パスワード

共有パスワードを入力してください

閲覧する

図 3.7: パスワード入力画面.

共有旅程の閲覧

東京

パスワードが正しくありません

×

パスワード

共有パスワードを入力してください

閲覧する

図 3.8: パスワード入力画面.

また, 情報を更新するたびに新たなリンクを作成する必要がないため, 複数のリンクが発生して混乱することもなく, スムーズな共有が可能となっている.

共有された側のユーザーは, 図 3.10 に示したとおり旅行情報を閲覧することはできるが, 内容の編集はできないように制限している. これにより, 情報の改ざんや誤操作を防ぎつつ, 安全かつ一方向的な共有を実現している.

東京

2026年2月13日 から 2026年2月15日 まで

📅 全体
📅 1日目
📅 2日目
📅 3日目

全日程の場所

1 📍 **東京駅**

東京都千代田区丸の内1-9-1, 東京都千代田区丸の内1-9-1

🚗 車 約25分 (16.5km)

2 📍 **ディズニープラザホテル**

千葉県浦安市舞浜2-1-1, 千葉県浦安市舞浜2-1-1

🚗 車 約96分 (3.6km)

3 📍 **葛西臨海公園**

東京都江戸川区臨海町6-2, 東京都江戸川区臨海町6-2

🚗 車 約17分 (8.1km)

4 📍 **浦安駅**

千葉県浦安市北栄1-13-1, 千葉県浦安市北栄1-13-1

図 3.9: 共有表示画面.

3.1.3 旅行作成画面

図 3.11 のような旅行作成画面は, 本アプリケーションにおける起点となる機能であり, ユーザーが旅行の基本情報を登録するための欠かせないインターフェースである. この画

面を通じて入力された情報は、以降のスケジュール管理や地図表示、チェックリスト生成など、他の機能と密接に連携する基盤となる。

新しい旅行を作成

旅行タイトル

旅行のタイトルを入力してください

説明

メモ

開始日

yyyy/mm/dd

終了日

yyyy/mm/dd

保存 キャンセル

図 3.10: 旅行作成画面.

3.1.4 旅程の視覚化管理

ユーザーが旅行の全体像や日程ごとの計画を直感的に把握できるよう、複数の視覚化機能を実装した。

まず、図 3.12 に示したとおり登録された旅行期間に基づいて、各旅行を「予定」「旅行中」「終了」「未定」の 4 つのステータスに分類し、色分けによって一覧画面上に表示する機能を実装した。これにより、ユーザーは「どこに行ったか」「これからどこに行くか」といった旅行の進行状況を一目で把握できるようになった。具体的には、「旅行中」は緑、「予定」は青、「終了」はグレー、「未定」は黄色となっている。さらに、色だけでなくアイコンを加えることでより視覚的にわかりやすい工夫を行った。

また、旅行詳細画面では、Google Maps API [4] を用いて登録された観光場所を地図上に表示する機能を導入した。これにより、各観光地の位置関係や移動ルートを視覚的に確認でき、旅行計画全体の構造を把握しやすくなった。

加えて、旅行期間を事前に入力しておくことで、旅行の日数分だけタブを動的に生成する機能を実装した。旅行に含まれる観光場所を図 3.13 に示したとおり、「全体」「1 日目」「2 日目」など日程ごとに切り替えて表示する機能も併せて導入し、ユーザーはタブを操作

旅行一覧



図 3.11: 色とアイコンで視覚的に整理された旅行一覧画面。

することで、旅行全体の流れを把握しながら、各日の予定を個別に確認・編集することが可能となった。これにより、日程ごとの計画を視覚的に整理しやすくなり、過不足のあるスケジュールや移動の偏りにも気づきやすくなった。また、編集対象の日付を明確に切り替えられることで、誤操作の防止にもつながっている。

タブの切り替えに連動して、地図上に表示される観光地もその日に対応するもののみに自動で更新されるため、日程ごとの旅程がより視覚的に管理しやすい設計となっている。

これらの機能により、ユーザーは旅行の計画段階において、旅程を効率的に把握・調整できるようになり、直感的な旅行プランニングが可能となった。

3.1.5 UI 設計とスタイルの工夫

本システムでは、Bootstrap 5.1.3[5] を採用し、ユーザーインターフェースの視覚的理解を促進した。

観光地情報などの表示に Bootstrap のカードコンポーネントを活用し、情報の視認性と整理性を高めている。カードは、場所情報や旅行詳細をカード形式で表示することで、各情報のまとまりを視覚的に明確化した。これにより、ユーザーは一目で情報の区切りを理解できる。



図 3.12: 全体から 1 日目のタブを押したときの变化.

さらに Font Awesome[5] を併用し、ボタンにアイコンを付与することで、言語に依存しない直感的な操作を可能にした。

3.1.6 旅程の編集・調節機能

旅行の計画を柔軟に調整できるよう、旅程編集に関する 2 つの機能を実装した。

まず、観光場所の訪問順序をドラッグすることで自由に入れ替える機能を導入した。これにより、ユーザーは移動効率や興味関心に応じて旅程を柔軟に再構成できるようになった。従来のように順序を変更するたびに項目を削除し、再度追加し直す必要がなくなったことで、操作性が向上し、旅程編集の負担が大幅に軽減された。

また図 3.14 に示したとおり、各観光地間の移動時間を自動で算出・表示する機能を実装した。算出されるのは車での移動を想定した所要時間であり、ユーザーは移動にかかる時間を考慮しながら、現実的かつ無理のないスケジュールを立てることが可能となった。

これらは Mapbox の Directions API[6] を活用して開発を行っている。以下にその処理の流れを示す。

アクティブなタブに表示されている観光地リストから、各地点の座標（経度・緯度）を

配列として取得する。この配列の先頭を出発地、末尾を目的地、それ以外を経由地として分割し、Directions API に渡す形式に整形する。travelDuration(type, start, waypoints, dest) は、Mapbox Directions API を使ってルート情報を取得する関数である。

- type: 移動手段 (例: driving, walking, cycling)
- start: 出発地の座標 ({ lon, lat })
- waypoints: 経由地の配列
- dest: 目的地の座標

経由地は任意の個数を指定できるため、配列として管理している。指定形式に変換した座標データと移動手段をもとに、以下の形式の URL に対してリクエストを送信する。

```
https://api.mapbox.com/directions/v5/{profile}/{coordinates}
```

に対してリクエストを送信することで、複数地点間の移動時間・距離を一度に取得できる。

- ・profile: 移動手段を指定するパラメータであり、driving (車), walking (徒歩), cycling (自転車) の 3 種類が用意されている。本アプリでは driving を使用している。

- ・coordinates: 出発地・経由地・目的地の座標を「経度, 緯度」形式でセミコロン (;) 区切りで列挙する。

この形式により、複数地点間のルートを 1 回のリクエストでまとめて取得することが可能となる。

API からのレスポンスには routes 配列が含まれており、最適なルートが routes[0] に格納されている。この中の legs 配列には、各区分 (出発地→経由地→目的地) ごとの詳細情報が含まれており、それぞれの leg には以下の情報が含まれる。

- ・duration: 所要時間 (秒単位)
- ・distance: 移動距離 (メートル単位)

取得した各区分のデータをもとに、所要時間は分単位に、距離はキロメートル単位 (小数点 1 桁) に変換し、「車 約 15 分 (3.2km)」のような形式で、観光地カードの間に動的に挿入して表示している。移動時間が可視化されたことで、訪問地の組み合わせや順序の検討がより直感的かつ効率的に行えるようになった。

これらの機能は、旅行の編集・調節を支援する上で不可欠な要素であり、ユーザーが旅程全体の流れを把握しながら、日程ごとの詳細な計画を直感的に調整できる環境を提供している。



図 3.13: 車による移動時間の自動算出結果を表示した画面.

3.1.7 検索機能

旅行情報の登録・閲覧・編集を効率的かつ直感的に行えるよう、2つの検索機能を実装した。

旅行一覧画面に多くの旅行が表示されている場合、目的の旅行を探し出すのに時間がかかることがある。そこで、図 3.15 のように旅行名や説明欄に含まれる単語をもとに部分一致によるキーワード検索を行い、該当するデータを絞り込めるようにした。いずれの検索においても、大文字・小文字を区別しない部分一致検索を実現しているため、ユーザーは入力時に文字の大小を意識する必要がない。そのため、検索漏れを防ぐことができ、よりスムーズに目的の情報へアクセスできるようになった。また、検索結果の件数を取得し、「○件」という形式で表示することで、検索結果の量を一目で把握できるようにした。

観光地の登録機能においては、Mapbox の Search Box API[7] を活用した地名検索機能を導入した。施設名の入力画面は図 3.16 である。これにより、ユーザーは住所だけでなく、観光地名やカテゴリ、通り名などを入力することで、目的の場所を効率的に検索できるようになった。

以前は完全無料である OpenStreetMap の Nominatim を利用していたが、目的の観光地が検索にヒットしにくいという課題があった。そこで、Mapbox の Search Box API への切り替えにより、検索精度が向上し、より多くの地名を正確に取得できるようになったことで、ユーザー体験の向上につながった。

また、Mapbox の Search Box API は月 10 万回まで無料で利用できるため、コスト面でも

旅行一覧

「東京」の検索結果: 1件

→ 旅行中

東京

2026年1月7日 から 2026年1月9日 まで

登録済みの場所: 4

図 3.14: 検索結果画面.

導入しやすいという利点がある.

新しい場所を追加

追加する日
1日目

場所の名前
ディズニー

ディズニーアンバサダーホテル
千葉県浦安市舞浜 2 - 1 1

ディズニーストア
東京都渋谷区宇田川町 2 0 - 1 5

東京ディズニーセレブレーションホテル
千葉県浦安市明海 7 - 1 - 1

リゾートゲートウェイステーション駅
千葉県浦安市舞浜 1 - 4

ベイサイドステーション駅
〒100-0001 東京都千代田区千代田 1-1-1

説明

図 3.15: 施設名の入力画面.

3.1.8 チェックリスト機能

旅行準備を効率的に進められるよう, チェックリストに関する複数の機能を実装した. まず, 図 3.17 のように旅行ごとに標準のチェックリスト設定を適用できる機能を導入した. これにより, ユーザーは毎回ゼロからリストを作成する必要がなく, あらかじめ用意された基本項目をもとに効率よく準備を開始できるようになった. さらに, 基本項目をテンプレート化することで, 今回だけ書き忘れたという漏れを防ぐ効果も期待できる.

デフォルトチェックリスト設定

ここで設定した項目が、新しい旅行を作成したときに自動的に追加されます。

いつも持つもの	
パスポート	<input checked="" type="checkbox"/> <input type="checkbox"/>
航空券	<input checked="" type="checkbox"/> <input type="checkbox"/>
財布	<input checked="" type="checkbox"/> <input type="checkbox"/>
クレジットカード	<input checked="" type="checkbox"/> <input type="checkbox"/>
現金	<input checked="" type="checkbox"/> <input type="checkbox"/>
スマートフォン	<input checked="" type="checkbox"/> <input type="checkbox"/>
充電器	<input checked="" type="checkbox"/> <input type="checkbox"/>
モバイルバッテリー	<input checked="" type="checkbox"/> <input type="checkbox"/>
保険証	<input checked="" type="checkbox"/> <input type="checkbox"/>
+ 項目を追加	

図 3.16: チェックリスト設定画面.

また, 各旅行ごとに持ち物や準備項目を自由に編集できるチェックリスト編集機能を導入した. これにより, ユーザーは旅行の目的や季節, 同行者に応じて, 図 3.18 のように必要な項目を柔軟に追加することも可能となった. 状況に応じたカスタマイズが行えることで, より実際のニーズに即した準備ができるようになり, 忘れ物や準備漏れの防止にもつながる.

項目を追加

×

項目名

キャンセル

追加

図 3.17: チェックリスト追加画面.

さらに, 図 3.19 に示したとおり, 各項目のチェック状況を保持する機能を実装し, ユーザーが確認済みの持ち物を記録できるようにした. これにより, 準備の進捗状況を可視化しながら, 持ち物の確認状況を一目で把握できるようになり, 確認漏れや二重確認の負担を軽減することが可能となった. チェックの状態は旅行ごとに保存されるため, 複数の旅行を並行して準備する場合でも, 各旅程の進捗を個別に管理できる. これらの機能は, 旅行前の準

いつも持つもの	
<input type="checkbox"/> パスポート	
<input type="checkbox"/> 航空券	
<input checked="" type="checkbox"/> 現金	
<input checked="" type="checkbox"/> クレジットカード	
<input checked="" type="checkbox"/> 現金	
<input checked="" type="checkbox"/> スマートフォン	
<input checked="" type="checkbox"/> 充電器	
<input type="checkbox"/> モバイルバッテリー	
<input checked="" type="checkbox"/> 保険証	
+ 項目を追加	

図 3.18: チェックリストの管理画面.

備を支援する上で重要な役割を果たしており, ユーザーが安心して出発できるよう, 実用性と柔軟性を兼ね備えたチェックリスト管理環境を提供している.

第4章 まとめ

本研究では、python で書かれたサーバー側 Web アプリケーションフレームワークである Django を開発環境として、視覚的な理解と効率的に旅行計画を行えるデジタル版しおりを開発した。

開発手法としては、機能ごとに優先順位を設定し、重要度の高いものから順に実装を進めることで、ユーザーに対して価値を早期に提供することを重視した。このアプローチにより、開発初期から研究室内での試用とフィードバックが可能となり、得られた意見を反映しながら、より実用性と完成度の高いグループウェアアプリへと発展させることができた。

機能面では地図の表示や色分け機能によって旅行全体の構成を視覚的に把握しやすくし、キーワード検索機能や地名検索機能により、目的地の絞り込みや情報の整理を効率的に行えるようにした。また、Django の認証機能を活用してログイン機能を実装し、ユーザーごとの情報管理やアクセス制御を可能にした。さらに、チェックリストや日程ごとのタブ表示、移動時間の自動算出といった機能を組み合わせることで、旅行前の準備から当日の行動管理、旅行後の振り返りまでを一貫してサポートできる設計とした。

結果として、旅行計画を可視化し、ユーザーが直感的に全体像を把握できるアプリケーションを開発することができた。

謝辞

本研究を進めるにあたり、御指導いただきました西谷滋人教授には心から感謝いたします。また、実際にアプリを使用していただき、貴重なご意見や改善のヒントをくださった西谷研究室の皆さまにも、深く感謝いたします。皆さまからの率直なフィードバックが、本研究の質を高める大きな支えとなりました。お礼申し上げます。

参考文献

- [1] 旅しお - <https://tabisio.com/> (accessd on 05 Sep 2025).
- [2] Django - <https://www.djangoproject.com/> (accessd on 05 Sep 2025).
- [3] GitHub Inc - <https://github.com/> (accessd on 05 Sep 2025).
- [4] Google Maps API- <https://developers.google.com/maps/documentation/javascript/overview?hl=j> (accessd on 05 Sep 2025).
- [5] Bootstrap - <https://getbootstrap.com/docs/5.1/getting-started/introduction/> (accessd on 05 Sep 2025).
- [6] Font Awesome - <https://fontawesome.com/> (accessd on 05 Sep 2025).
- [7] Directions API - <https://docs.mapbox.com/api/navigation/directions/> (accessd on 20 Dec 2025).
- [8] Search Box API - <https://docs.mapbox.com/api/search/search-box/> (accessd on 20 Dec 2025).

目 次