
TripWeave — Software Requirements Specification (SRS)

Document Version: 1.0
Product: TripWeave
Audience: Solo developer (you)
Purpose: Prevent confusion, scope creep, and bad decisions later

---

1. Introduction

1.1 Purpose of the System

TripWeave is a **web-based travel planning application** that helps users plan **end-to-end journeys** using multiple transport modes while ensuring:

- Time compatibility
- Adequate buffer times
- Reasonable cost aggregation

TripWeave **does not perform ticket booking**.
It only redirects users to external booking platforms.

1.2 Product Philosophy (VERY IMPORTANT)

- Planning ≠ Booking
- Core value = **intelligent route planning**
- Authentication and persistence are **optional enhancements**, not core

---

2. Version Roadmap (Hard Boundaries)

2.1 Version v1 — MVP (What you are building now)

Supported transport modes

- ✈️ Flight
- 🚆 Train

Explicit exclusions

- ❌ Bus
- ❌ Authentication
- ❌ Saved trips
- ❌ Payments
- ❌ Maps
- ❌ Dashboards
- ❌ Real-time tracking

Target outcome

- Portfolio-ready
- Fully usable without login
- Demonstrates system thinking and UX clarity

2.2 Version v2 — Persistence & Identity

New additions

- User authentication
- Saved trips
- User-specific trip history

What stays unchanged

- Core trip planning logic
- Screen flow
- Route generation algorithm

2.3 Version v3 — Monetization / Expansion (Optional)

- Bus support
- Pricing experiments
- API integrations
- Mobile app
- Accounts, preferences, analytics

---

3. User Types

3.1 Anonymous User (v1 primary user)

- Can plan trips
- Can view detailed routes
- Can access external booking links

- Cannot save trips

3.2 Authenticated User (v2+)

- All anonymous capabilities
- Can save and revisit trips
- Has persistent history

---

4. Functional Requirements (v1)

4.1 Trip Planning

- User shall be able to input:
    - Source city
    - Destination city
    - Date
    - Optional budget
    - Transport preferences (Flight / Train)
    - Buffer time
- System shall generate:
    - Valid single-mode or mixed-mode routes
    - Total duration
    - Estimated cost
    - Transfer count
    - Risk indicator (Low / Medium)

4.2 Route Evaluation

- System shall validate buffer times between segments
- System shall warn about tight connections
- System shall not rely on real-time availability

4.3 External Booking

- System shall provide external booking links per journey leg
- System shall clearly state that booking is handled externally

---

5. Non-Functional Requirements

5.1 Usability

- No login required in v1
- Minimal cognitive load
- Linear, predictable navigation

5.2 Performance

- Route generation may take 10–15 seconds
- Loading feedback must be visible

5.3 Maintainability

- Core logic must be authentication-agnostic

- Data models must allow optional ownership later

---

6. Data Model (Conceptual, Auth-Agnostic)

6.1 TripPlan (Core Object)

```
TripPlan
- id (unique)
- created_at
- input_data
- computed_routes
- owner_id (nullable, future use)
```

Rules

- `owner_id = null` in v1
- Ownership added only in v2
- Core planning logic never depends on owner

---

7. UX Screens (v1)

Screen List

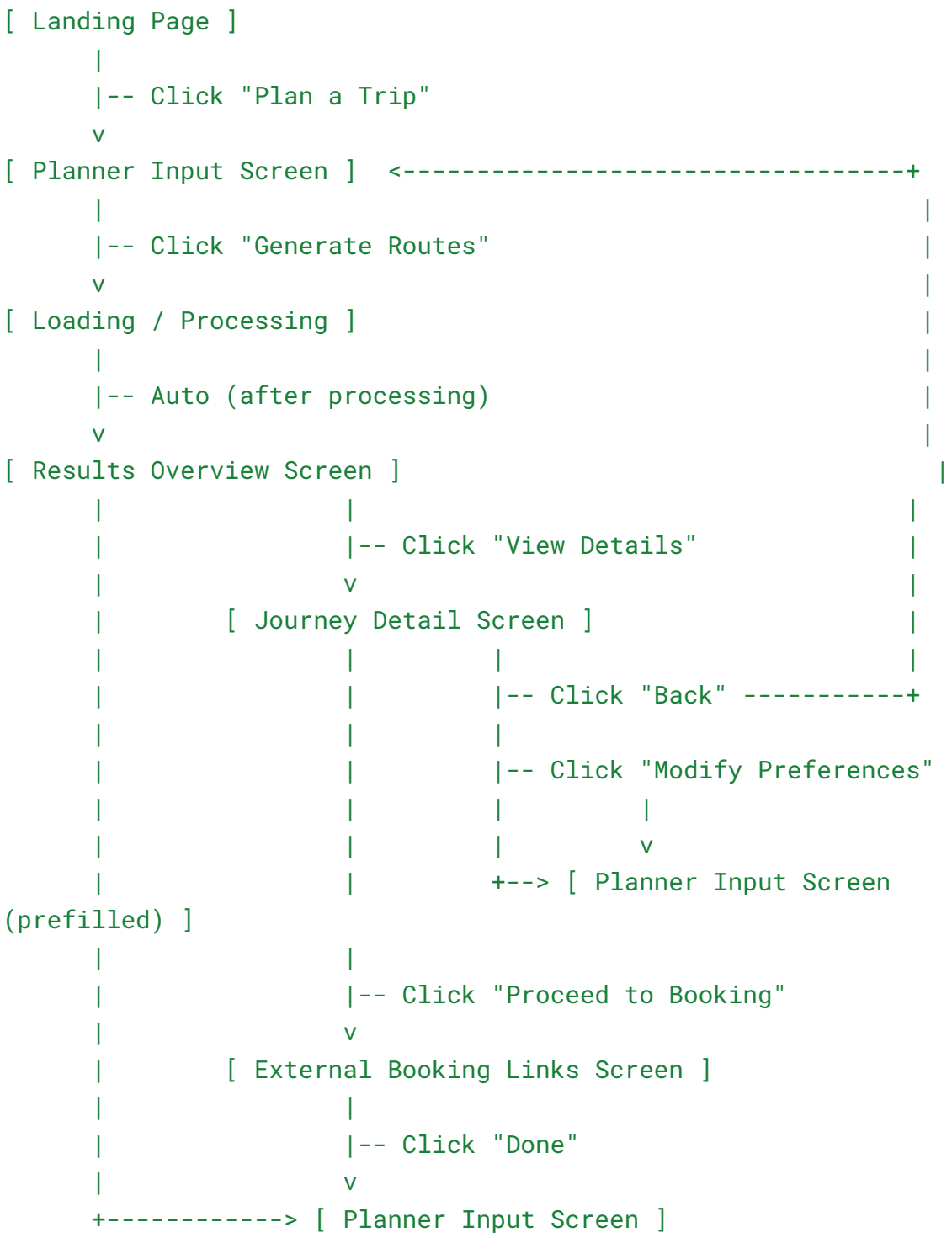
1. Landing Page
2. Planner Input Screen
3. Loading / Processing Screen
4. Results Overview Screen
5. Journey Detail Screen
6. External Booking Links Screen

No other screens are allowed in v1.

---

8. Screen → Button → Screen Flow

(TripWeave v1: Air + Rail)



---

9. Navigation Rules (Non-Negotiable)

- Planner Input Screen is the **hub**
- No screen may be a dead end
- Back navigation must always exist
- Login must never block planning (even in v2)

---

10. Authentication Design Rule (Future-Proofing)

Even though auth is not implemented in v1:

- Core logic must not assume a user exists
- Saving is a **separate concern**
- Trip generation must work independently

Golden Rule

> “Anyone can plan a trip.
> Only logged-in users can remember it.”

---

11. What NOT to Do (Reminder Section)

If you ever feel tempted to:

- Add auth early
- Add saved trips early
- Add buses early
- Add payments
- Add maps

STOP.
Re-read sections **2.1** and **10**.

---

12. Exit Criteria for v1 (When to stop)

v1 is DONE when:

- A user can plan Flight + Train journeys end-to-end
- All screens in Section 7 work
- The flow diagram in Section 8 is fully implemented
- No auth exists
- No saved trips exist

Anything beyond this is **v2+ work**.

Final note (for future-you)

If you're confused later, it means:

- You're trying to solve a **future version problem**
- While standing in **v1**