

Requirements Specification Document

Project: Travel Planning Social Platform

Team Name: MIN-roamer

Team Slogan: Roam Min, find your mates.

Date: 2026-10-10

1. Introduction

1.1 Purpose

This document defines the functional requirements, non-functional requirements, data requirements and interface requirements of the "Travel Planning Social Platform", serving as the unified basis for development, testing and acceptance. It is submitted as an attachment to Assignment 2 (Team Formation and Topic Selection).

1.2 Background

Planning a group trip usually means switching between maps, notes and chat records, and gathering food, transport and weather information from many places. Existing products are generally "booking-heavy, planning-light", and lack a companion-social feature deeply integrated with trip planning.

Referencing the open-source project TREK, this project aims to provide a **Fuzhou-first, visual map planning + social companionship** travel platform that helps users intuitively arrange trips with sticky notes on a map, while getting destination info, managing budgets, and finding travel companions.

1.3 Terminology

Term	Meaning
Sticky Note	A place / item entry for a certain day of a trip; can be dragged to reorder within the day
Owner	The trip creator, with full permission to edit, delete, invite and remove members
Member	A user invited into a trip; can edit schedules and add expenses, but cannot delete the trip
Review Note	A text + image review posted by a user under a place (Xiaohongshu-like UGC content)
Find Mates	The social feature for publishing / browsing travel-companion posts
POI	Point of Interest — a location marker on the map
P0/P1/P2	Feature priority: P0 must-do (core loop), P1 should-do (experience), P2 nice-to-have

1.4 References

- *The Method of Construction — Modern Software Engineering* (Zou Xin)
- TREK open-source project (github.com/mauriceboe/TREK)

2. Overall Description

2.1 Product Positioning

A Web application (responsive, mobile-first) that combines **trip planning, destination information, budget management and social sharing**, doing Fuzhou deep and well as the demo city, with support for switching to other cities.

2.2 Target Users

User Type	Core Need
Trip Planner	Arrange daily schedules intuitively on a map, without switching between apps
Group Traveler	Co-edit one trip with multiple people, find travel companions
Local / Visiting Tourist	Quickly get Fuzhou attractions, food, transport and weather
Content Sharer	Publish trips to the community, record and share travel experiences

2.3 User Roles & Permission Matrix

Action	Guest	Member	Owner
Browse public trips / community square	☒	☒	☒
Register / Login	☒	—	—
Create a trip	☒	☒	☒
Edit own trip	☒	☒ (as Owner)	☒
Edit joined trip	☒	☒	☒
Delete trip	☒	☒ (Owner only)	☒
Invite / remove members	☒	☒ (Owner only)	☒
Record expenses	☒	☒	☒
Like / favorite	☒	☒	☒
Publish mate post	☒	☒	☒
Publish review note	☒	☒	☒
Use AI assistant	☒	☒	☒

2.4 Operating Environment

- **Frontend:** modern browsers (Chrome / Edge / Safari), responsive layout for mobile
- **Backend:** Java 17+, Spring Boot 3.x, Spring Data JPA
- **Database:** H2 (file mode), single file, zero config, native Spring Boot + JPA support
- **AI service:** Tongyi Qianwen (Qwen), called through a backend proxy

2.5 Constraints & Assumptions

- Course project, ~12 weeks, 14 members
- Excluded: plugin system, third-party OIDC login, PWA offline, flight/hotel import, real-time WebSocket co-editing
- Fuzhou data is manually curated; other cities get POI via map-platform APIs
- No scraping of Xiaohongshu / Dianping; review notes are in-platform UGC
- AI depends on the Qwen API and a valid API key; without a key the AI module degrades gracefully and does not affect other features

3. Functional Requirements

3.1 Module 1: User & Profile (Group A, 2 members)

This is the infrastructure of the whole platform. Without an account system, we don't know who posted trips, who finds mates, or who favorited what. Community, find-mates and multi-user sharing all build on the user system.

ID	Feature	Description	Priority
U-01	Register	Register with username, email, password	P0
U-02	Login	Email + password login, returns JWT token	P0
U-03	Logout	Clear token, redirect to login page	P0
U-04	Profile	Edit avatar, nickname, bio, language preference	P1
U-05	My page	Show my published trips, favorited trips, mate posts	P1
U-06	User search	Search users by username, add as trip member	P1
U-07	Follow / fans	Follow users, view their public trips	P2

Acceptance criteria:

- Guests can only browse public trips; cannot edit, like, or post mates
 - All APIs have JWT auth middleware
 - Profile page can edit avatar, nickname, bio
 - Passwords are stored encrypted, never in plaintext
-

3.2 Module 2: Trip Planning (Group B, 3 members)

ID	Feature	Description	Priority
T-01	Trip list	Home page shows all trip cards I created + joined	P0
T-02	Create trip	Name, destination, start date, end date	P0
T-03	Edit / delete trip	Modify info; Owner can delete	P0
T-04	Map display	Leaflet interactive map, shows all place markers	P0
T-05	Place search	Search places by keyword, add to trip	P0
T-06	Sticky-note schedule	Drag a place/item to a day, reorder within the day	P0
T-07	Schedule detail	Click a sticky note to view title, time, note, place	P0
T-08	Route connection	Connect same-day places on the map in order	P1
T-09	Daily note	Write a note for each day	P2

Sticky-note interaction:

- Left side is a day panel ordered by date, right side is the map
- After searching a place, click "add to trip" → choose which day
- Sticky notes can be dragged to reorder within a day, or moved to another day
- Different days use different colors; map marker colors match

Acceptance criteria:

- Map can zoom, drag and click
- Search debounced at 300ms, results shown in a dropdown
- Sticky-note order is saved to the backend in real time and survives refresh
- Route connection follows schedule order, different days in different colors

3.3 Module 3: Destination Info (Group C, 4 members)

Core strategy: Fuzhou is the demo city with a batch of curated data. Each place's detail page aggregates user "review notes", like Xiaohongshu / Dianping reviews.

ID	Feature	Description	Priority
I-01	Fuzhou preset data	Preset ~50-100 popular attractions, food and districts with name, coordinates, intro, recommendation	P0
I-02	Switch city	Switch city at the top (Fuzhou default, Xiamen/Quanzhou optional); map center and recommendations follow	P0
I-03	City POI search	Search via external API (Amap/Baidu free tier) for that city's places	P0
I-04	Food recommendations	Popular restaurant/food list by city with rating, average price, tags	P0
I-05	Weather	Show the next 7 days' forecast (Open-Meteo, free, no key)	P0
I-06	Place review notes	Users post text + image "review notes" under a place; others read and like	P1
I-07	Transport guide	How to reach the city (plane/train/bus) + local transport (metro/bus/taxi)	P1
I-08	Info aggregation	Trip detail sidebar shows destination weather + food + top attractions	P1
I-09	Attraction detail	Clicking a place card shows image, intro, rating, review-note list	P1

On "online reviews and Xiaohongshu posts":

Scraping Xiaohongshu/Dianping has anti-crawl and legal risks; not recommended for a course project. Instead:

1. Fuzhou local data: manually curate 50-100 popular attractions/food into the database
2. Other cities: call Amap/Baidu open-platform free APIs for POI name, address, rating
3. User review notes (Xiaohongshu-like): build a UGC feature where users post notes under a place
4. No real-time scraping, avoiding legal and anti-crawl issues

Acceptance criteria:

- App opens defaulted to Fuzhou; map shows the Fuzhou area
- After switching city, map center, recommendation list and weather all follow
- Each place detail page has: basic info + rating + user review-note list
- Users can post their own review notes on a place detail page
- Weather data comes from Open-Meteo, no key required
- Fuzhou preset data covers at least “三坊七巷, 鼓山, 上下杭, 烟台山, 东街口” and 20+ popular restaurants

3.4 Module 4: Budget & Expense (Group D, 2 members)

ID	Feature	Description	Priority
B-01	Expense record	Record each expense: amount, category, date, note	P0
B-02	Expense categories	Transport, lodging, food, tickets, shopping, other	P0
B-03	Budget setting	Set the trip budget, show spent/remaining in real time	P0
B-04	Expense list	List expenses newest-first, filterable by category	P0
B-05	Statistics charts	Pie chart by category, bar chart by date	P1
B-06	Budget alert	Show red warning when over budget	P1

Acceptance criteria:

- All amounts stored as integer cents, avoiding floating-point errors
- Budget progress bar updates in real time, turns red when over
- Pie/bar charts use ECharts, refresh after add/delete
- Expense list supports filtering by category and date

3.5 Module 5: Community & Social (Group E, 2 members)

The differentiator vs. ordinary trip planners, including community sharing, multi-user sharing and find-mates.

ID	Feature	Description	Priority
S-01	Publish trip	Set a trip public and publish it to the community square	P0
S-02	Community square	Browse all public trips, card view of cover/title/destination	P0
S-03	Trip detail (public)	Open others' public trips, read-only	P0
S-04	Like / favorite	Like and favorite public trips	P1
S-05	Multi-user collaboration	Invite users to join my trip and co-edit	P1
S-06	Publish mate post	Post "find mates": destination, time, headcount, needs	P0
S-07	Mate square	Browse mate posts, filter by destination/time	P0
S-08	Contact author	View author profile, DM/comment	P2

Multi-user collaboration:

- The trip creator is **Owner**: edit, delete, invite, remove members
- Invited users are **Members**: edit schedules, add expenses, but cannot delete the trip
- Trips are private by default, visible only to Owner and Members
- Owner can one-click publish a trip as "public" to the community

Acceptance criteria:

- Community square is browsable without login (like/favorite requires login)
- Public trip's view page is separate from the edit page; public view has no edit button
- Mate post contains: title, destination, departure date, headcount, intro
- Mate square supports filtering by destination
- Members are invited by username search

3.6 Module 6: AI Assistant (AI Group, 1 member)

AI involves dialog management, prompt engineering and backend API proxying, so it is a separate group implemented later in development.

ID	Feature	Description	Priority
AI-01	AI generate itinerary	Input destination + days, AI generates a draft itinerary the user can adjust	P1
AI-02	AI travel Q&A	Sidebar AI assistant for questions: attraction history, food, transport, tips	P1

Acceptance criteria:

- AI itinerary generation calls a backend proxy, not the AI API directly from the frontend
- The AI assistant sidebar can be summoned anytime and supports multi-turn conversation
- AI answers stay travel-related (attractions, food, transport, lodging)

4. Non-Functional Requirements

Category	Requirement	Metric
Performance	Debounce search; fast API; smooth page load	Search debounce 300ms; API < 1s; first screen < 3s
Security	Encrypted passwords; JWT auth; AI key protected; input validation	BCrypt; all writes check login; parameterized SQL; XSS escaping
Usability	Responsive for mobile & desktop; EN/ZH switch; clear feedback	375px and 1280px both fine; language switch without reload
Compatibility	Mainstream browsers	Chrome 90+, Edge 90+, Safari 14+, Firefox 88+
Maintainability	Frontend/backend separated; modular; unified style; Git PR flow	ESLint + Prettier; Checkstyle; PR merged after ≥1 review
Extensibility	Cities extensible via external API; low coupling	Add a city with just city ID + coordinates
Reliability	Data persistence; friendly error handling	Regular DB backup; friendly frontend errors, no blank screen

5. Data Requirements (Database Design Overview)

5.1 Core Tables

Table	Description
<code>users</code>	Users (id, username, email, password_hash, avatar_url, bio, language, created_at, updated_at)
<code>trips</code>	Trips (id, owner_id, title, destination, start_date, end_date, cover_image, description, is_public, created_at, updated_at)
<code>trip_members</code>	Trip members (id, trip_id, user_id, role, joined_at)
<code>trip_days</code>	Trip days (id, trip_id, date, note)
<code>day_items</code>	Sticky notes (id, trip_day_id, place_id, title, start_time, end_time, note, sort_order, created_at)
<code>places</code>	Places (id, city, name, address, lat, lng, category, description, rating, cover_image, is_preset, created_at)
<code>place_reviews</code>	Review notes (id, place_id, author_id, title, content, rating, images, likes_count, created_at, updated_at)
<code>review_likes</code>	Review likes (id, review_id, user_id)
<code>expenses</code>	Expenses (id, trip_id, title, amount_cents, category, date, note, created_at)
<code>trip_likes</code>	Trip likes (id, trip_id, user_id, created_at)
<code>trip_favorites</code>	Trip favorites (id, trip_id, user_id, created_at)
<code>buddy_posts</code>	Mate posts (id, author_id, title, destination, start_date, end_date, people_needed, description, status, created_at, updated_at)

5.2 Data Constraints

- Amounts stored as integer **cents**, converted to yuan on display, avoiding floating-point errors
- All timestamps stored in UTC, converted to the user's local timezone on display
- Foreign keys use sensible cascade policies (e.g., deleting a trip cascades schedules, expenses)
- Passwords stored as BCrypt hashes, never plaintext

Full field design and ER diagram are in Section 5 of the Project Development Document.

6. Interface Requirements (API Design Overview)

6.1 Conventions

- **Base URL:** `/api/v1`
- **Auth:** `Authorization: Bearer <token>`
- **Unified response format:**

```
{ "code": 0, "message": "success", "data": {} }
```

6.2 Main Endpoint Groups

Module	Method	Path	Description
Auth	POST	/auth/register	Register
	POST	/auth/login	Login
	GET	/auth/me	Get current user
Users	GET	/users/:id	View user profile
	PUT	/users/me	Edit profile
	GET	/users/search?q=	Search users (for inviting)
Trips	GET	/trips/mine	My trips
	POST	/trips	Create trip
	GET	/trips/:id	Trip detail
	PUT	/trips/:id	Update trip
	DELETE	/trips/:id	Delete trip
	POST	/trips/:id/publish	Publish to community
Trip Days/Items	GET	/trips/:id/days	Get all days & sticky notes
	POST	/days/:id/items	Add sticky note
	PUT	/items/:id	Update sticky note
	DELETE	/items/:id	Delete sticky note
	PUT	/items/reorder	Batch reorder
Places	GET	/places/search?q=&city=	Search places
	GET	/places/hot?city=	Hot places of a city
	GET	/places/:id	Place detail
	GET	/places/:id/reviews	Review-note list
	POST	/places/:id/reviews	Post review note
City	GET	/cities/hot	Hot cities
	GET	/city/info?name=	City weather + transport overview
AI	POST	/ai/generate-itinerary	AI generate itinerary
	POST	/ai/chat	AI travel Q&A
Weather	GET	/weather?lat=&lng=&days=	Weather forecast
Expenses	GET	/trips/:id/expenses	Expense list

Module	Method	Path	Description
	POST	/trips/:id/expenses	Add expense
	PUT	/expenses/:id	Update expense
	DELETE	/expenses/:id	Delete expense
	GET	/trips/:id/expenses/summary	Expense statistics
Community	GET	/community/trips	Community square
	GET	/community/trips/:id	Public trip detail
	POST	/community/trips/:id/like	Like
	POST	/community/trips/:id/favorite	Favorite
Members	GET	/trips/:id/members	Member list
	POST	/trips/:id/members	Add member
	DELETE	/members/:id	Remove member
Buddy	GET	/buddy/posts	Mate square list
	POST	/buddy/posts	Post mate post
	GET	/buddy/posts/:id	Mate post detail
	DELETE	/buddy/posts/:id	Delete mate post

Full endpoint list is in Section 6 of the Project Development Document.

7. Acceptance Criteria Summary

7.1 P0 Features (all must pass)

1. Register, login, logout work; passwords stored encrypted
2. Guests can browse community & public trips; logged-in users can plan, like, post mates
3. Trip create/edit/delete works; trip list displays correctly
4. Map is interactive (zoom/drag/marker); place search works
5. Sticky notes can be reordered and survive refresh
6. Fuzhou preset data (attractions + food) works out of the box, covering core landmarks
7. Switching city updates map, recommendations and weather
8. Weather shows the next 7 days' forecast
9. Expense record, budget setting, expense list work
10. Trips can be published to the community; the square is browsable
11. Mate posts can be published and browsed
12. All P0 APIs have JWT auth, no privilege escalation

7.2 P1 Features (try to complete)

1. Profile editing, my page
2. Route connection (same-day places connected on the map)
3. AI itinerary generation
4. AI travel Q&A (multi-turn)
5. Place review notes (post + like)
6. Transport guide
7. Info aggregation (trip sidebar with weather/food)
8. Attraction detail page
9. Expense statistics charts (pie + bar)
10. Budget alert
11. Trip like/favorite
12. Multi-user collaboration (invite + co-edit)

7.3 P2 Features (time permitting)

1. Daily notes
2. Follow / fans
3. Contact mate author (DM/comment)

8. Risks & Mitigations

Risk	Impact	Mitigation
AI API key unavailable or quota exhausted	AI module unusable	AI is degradable: hide the entry without a key, no impact on other modules
Map/POI API rate limit exceeded	Place search unusable	Add local caching; Fuzhou preset data doesn't depend on external APIs; prepare a backup API
14-member collaboration code conflicts	Lower efficiency	Strict Git branch strategy + PR review; modular development, each group owns its module
Heavy mobile adaptation workload	Poor mobile experience	Tailwind CSS responsive utilities; prioritize core flows on mobile
Low quality of Fuzhou preset data	Poor UX	Manually review preset data; compile from public information
Slow map tiles in mainland China	Slow map loading	OpenStreetMap tiles can be slow; switch to a domestically accessible tile source (e.g., Amap tiles)

This document is used together with the Project Development Document, whose tech-stack selection, Git collaboration rules and milestones are the implementation basis. This document is updated continuously as the project iterates.

