SkyNav MVP — Product Specification Document

Document Version: 1.0
Status: Draft
Last Updated: February 7, 2026
Author: Shayde
Project Codename: SkyNav

Table of Contents

1. Executive Summary
2. Product Vision & Strategy
3. Target Users & Personas
4. MVP Scope & Boundaries
5. Functional Requirements
6. Non-Functional Requirements
7. System Architecture
8. Technology Stack
9. Data Models & Schema Design
10. API Specification
11. Authentication & Authorization
12. User Interface Specification
13. Content Strategy
14. Deployment Architecture
15. Development Milestones
16. Testing Strategy
17. Scalability & Future Roadmap

18. Risk Assessment
19. Appendix

---

1. Executive Summary

SkyNav is a mobile-first progressive web application (PWA) that provides skydivers with structured operational intelligence for unfamiliar dropzones. The platform replaces fragmented word-of-mouth knowledge with a centralized, community-enriched repository of dropzone information, landing area intelligence, hazard awareness data, and travel preparation tools.

The MVP targets traveling skydivers — jumpers who visit dropzones outside their home DZ — and delivers immediate utility through five core features: a searchable dropzone directory, interactive landing maps with hazard and pattern overlays, moderated community notes, personal favorites with travel mode, and an administrative content management console.

The system is built on a serverless Azure architecture (Azure Functions, Cosmos DB, Blob Storage, Azure Maps) with a React/Next.js PWA frontend, designed for cost-efficient solo operation at launch with a clear scaling path as the user base grows.

Estimated MVP Timeline: 10–13 weeks (solo builder)

---

2. Product Vision & Strategy

2.1 Problem Statement

Skydivers traveling to unfamiliar dropzones face a significant safety and experience gap. Critical operational knowledge — landing area layouts, local hazards, pattern conventions, and facility logistics — is typically exchanged informally through social media groups, word-of-mouth, or discovered on arrival. This fragmented information ecosystem creates unnecessary risk, particularly for intermediate-level jumpers who may lack the network or experience to know what questions to ask.

2.2 Solution

SkyNav provides a single, structured source of dropzone intelligence that a skydiver can consult before arriving at a new DZ. The platform combines:

- **Structured operational data** curated by administrators (DZ profiles, landing maps, hazard annotations)

- **Community-contributed insights** moderated for accuracy (community notes)
- **Personal utility tools** for trip planning (favorites, travel mode)

2.3 Product Principles

Principle	Description
Safety First	Every feature should reduce risk for visiting skydivers. Information accuracy is non-negotiable.
Utility Over Volume	Fewer features, deeply useful. No feature bloat.
Mobile-First	The primary use case is a skydiver checking their phone before a jump or during travel.
Community Trust	All community content passes through moderation. No unverified information reaches end users.
Progressive Enrichment	Start with admin-seeded content, layer community input over time.

2.4 Key Metrics (MVP)

Metric	Target	Measurement Method
Dropzones seeded at launch	5–10	Manual count
Profile completeness score	≥ 80% fields populated	Automated audit
Time to find DZ info	< 30 seconds	User testing observation
Community note approval rate	≥ 60% of submissions	Moderation pipeline metrics
Return user rate (30-day)	≥ 40%	Analytics

---

3. Target Users & Personas

3.1 Primary Persona — The Traveling Jumper

- **Experience Level:** Intermediate to advanced (100+ jumps)

- **Behavior:** Visits 2–5 unfamiliar DZs per year
- **Pain Point:** Arrives at a new DZ without knowledge of landing areas, hazards, or local procedures
- **Goal:** Jump safely and confidently without relying on finding the right person to ask
- **Device:** Primarily mobile (phone), occasionally tablet or laptop

3.2 Secondary Persona — The Instructor / Coach

- **Experience Level:** Expert (1000+ jumps, rated instructor or coach)
- **Behavior:** Travels to DZs for events, boogies, coaching gigs
- **Pain Point:** Needs quick landing area reference when coaching visiting students
- **Goal:** Verify local procedures quickly to provide accurate guidance
- **Device:** Mobile

3.3 Tertiary Persona — The DZ Regular

- **Experience Level:** Any
- **Behavior:** Primarily jumps at their home DZ
- **Pain Point:** Wants to contribute local knowledge to help visitors
- **Goal:** Share accurate intel about their home DZ via community notes
- **Device:** Mobile or desktop

3.4 Admin Persona — Content Administrator

- **Role:** Platform operator (initially the solo builder)
- **Responsibilities:** Seed DZ content, manage map annotations, moderate community notes
- **Goal:** Maintain content quality and accuracy across all dropzone profiles
- **Device:** Desktop (admin console optimized for larger screens)

---

4. MVP Scope & Boundaries

4.1 In Scope

Feature ID	Feature	Priority
F1	Dropzone Directory	P0 — Core
F2	Interactive Landing Map	P0 — Core
F3	Community Notes	P1 — Important
F4	Favorites / Travel Mode	P1 — Important
F5	Admin Content Management	P0 — Core

4.2 Explicit Non-Goals (MVP)

The following are intentionally excluded from the MVP to maintain scope discipline. They represent potential post-MVP roadmap items.

Excluded Feature	Rationale
AI-assisted jump planning	Requires significant training data and validation; safety-critical
Automated weather ingestion	Adds integration complexity; skydivers already use dedicated weather tools
Social networking features	Risks scope creep; community notes provide sufficient social layer
Real-time collaboration	Requires WebSocket infrastructure; not needed for MVP use case
Native mobile builds (iOS/Android)	PWA provides sufficient mobile experience; native can follow if validated
User-to-user messaging	Social feature; out of scope
DZ rating/review system	Risks subjectivity issues; community notes serve this purpose with moderation
Automated DZ data import	Manual curation ensures quality for initial seed content

4.3 MVP Definition of Done

The MVP is considered complete when:

1. A user can browse and search a directory of seeded dropzone profiles

2. Each profile displays structured operational info, a landing map with annotations, and approved community notes
3. Users can submit community notes that enter a moderation queue
4. Authenticated users can save favorite dropzones and access a travel mode view
5. An admin can create/edit DZ profiles, manage map annotations, and moderate community notes
6. The application is responsive and functional on mobile devices
7. The application is deployed to a production environment with CI/CD

---

5. Functional Requirements

5.1 F1 — Dropzone Directory

5.1.1 Directory Browsing

ID	Requirement	Priority
F1.1	The system shall display a paginated list of all dropzones	P0
F1.2	Each directory listing shall show the DZ name, location (city/state), and a thumbnail or status indicator	P0
F1.3	The directory shall support infinite scroll or pagination (≥ 20 items per page)	P1
F1.4	The directory shall display a loading skeleton while data is fetched	P1

5.1.2 Search & Filtering

ID	Requirement	Priority
F1.5	Users shall be able to search dropzones by name (text input, client-side or server-side)	P0
F1.6	Users shall be able to search dropzones by location (city, state, or region)	P0
F1.7	Search results shall update as the user types (debounced, ≤ 300ms delay)	P1

F1.8	The system shall display a “no results” state with a clear message	P1

5.1.3 Dropzone Profile

ID	Requirement	Priority
F1.9	Each dropzone shall have a dedicated profile page accessible via URL (deep-linkable)	P0
F1.10	The profile shall display: DZ name, location, operational info, landing summary, travel notes	P0
F1.11	The profile shall embed the interactive landing map (F2)	P0
F1.12	The profile shall display approved community notes (F3)	P0
F1.13	The profile shall include a “Save to Favorites” action for authenticated users	P1
F1.14	Operational info shall include at minimum: aircraft types, altitude, jump ticket pricing, manifest procedures, contact info, hours of operation	P0
F1.15	Travel notes shall include at minimum: nearest airport, driving directions summary, lodging options, camping availability, local amenities	P1

5.2 F2 — Interactive Landing Map

5.2.1 Map Rendering

ID	Requirement	Priority
F2.1	Each dropzone profile shall display an interactive map centered on the DZ location	P0
F2.2	The map shall support pan, zoom, and pinch-to-zoom on mobile devices	P0
F2.3	The map shall render satellite/aerial imagery as the base layer	P0
F2.4	The map shall support toggling between satellite and standard map views	P2

5.2.2 Annotations & Overlays

ID	Requirement	Priority

F2.5	The map shall display landing area boundaries as polygon overlays	P0
F2.6	The map shall display hazard markers (obstacles, power lines, water, restricted areas) with distinct iconography	P0
F2.7	The map shall display traffic pattern direction indicators (arrows/lines showing standard pattern flow)	P0
F2.8	Each annotation shall display a label or tooltip on tap/click with a description	P0
F2.9	Annotations shall be visually distinguished by type using color coding: landing areas (green), hazards (red/orange), patterns (blue)	P1
F2.10	All map annotations are admin-seeded for MVP (no user-submitted annotations)	P0

5.2.3 Map Interaction

ID	Requirement	Priority
F2.11	The map shall include a legend explaining annotation types and colors	P1
F2.12	Users shall be able to tap a “Full Screen” control to expand the map on mobile	P1
F2.13	The map shall load efficiently on mobile connections (lazy-load tiles, minimize initial payload)	P1

5.3 F3 — Community Notes

5.3.1 Note Submission

ID	Requirement	Priority
F3.1	Authenticated users shall be able to submit a text note associated with a specific dropzone	P0
F3.2	The submission form shall include a text area (max 1000 characters) and a submit button	P0
F3.3	Upon submission, the note shall enter a “pending” moderation state and not be visible to other users	P0
	The submitting user shall see a confirmation message indicating the note	

F3.4	is pending review	P1
F3.5	Users shall not be able to submit more than 3 pending notes per dropzone at a time (spam prevention)	P2

5.3.2 Note Display

ID	Requirement	Priority
F3.6	The DZ profile shall display all approved community notes in reverse chronological order	P0
F3.7	Each note shall display: content text, author display name, submission date	P0
F3.8	If no approved notes exist for a DZ, display a prompt encouraging the user to submit the first note	P1

5.3.3 Moderation Pipeline

ID	Requirement	Priority
F3.9	All submitted notes shall default to “pending” status	P0
F3.10	Admins shall be able to approve, reject, or delete any note	P0
F3.11	Rejected notes shall not be visible to any non-admin user	P0
F3.12	The system shall maintain a status field on each note: `pending`, `approved`, `rejected`	P0
F3.13	Admins shall be able to view all notes filtered by status	P1

5.4 F4 — Favorites & Travel Mode

5.4.1 Favorites

ID	Requirement	Priority
F4.1	Authenticated users shall be able to save a dropzone to their favorites list	P0
F4.2	Users shall be able to remove a dropzone from their favorites	P0

F4.3	The favorites list shall be accessible from the user's dashboard or a dedicated "My DZs" section	P0
F4.4	The favorite/unfavorite action shall be available on both the directory listing and the DZ profile page	P1
F4.5	Favoriting shall provide immediate visual feedback (icon toggle) with optimistic UI update	P1

5.4.2 Travel Mode

ID	Requirement	Priority
F4.6	When viewing a favorited DZ, users shall see a "Travel Mode" view that surfaces key travel info	P1
F4.7	Travel mode shall display a structured checklist of pre-jump preparation items (e.g., USPA membership, reserve repack date, local currency/payment)	P1
F4.8	Travel mode shall surface the DZ's contact info, address, and hours prominently	P1
F4.9	The travel checklist shall be static/generic for MVP (not user-customizable)	P1

5.5 F5 — Admin Content Management

5.5.1 Dropzone Management

ID	Requirement	Priority
F5.1	Admins shall be able to create new dropzone profiles via a structured form	P0
F5.2	Admins shall be able to edit all fields of an existing dropzone profile	P0
F5.3	Admins shall be able to delete a dropzone (soft delete preferred)	P1
F5.4	All admin DZ edits shall be recorded in an audit history with timestamp and admin ID	P1

5.5.2 Map Annotation Management

ID	Requirement	Priority
F5.5	Admins shall be able to create map annotations (landing areas, hazards, patterns) for any DZ	P0
F5.6	Admins shall be able to edit or delete existing annotations	P0
F5.7	The annotation editor shall support: placing markers on a map, drawing polygons, setting annotation type, entering description text	P0
F5.8	Annotations shall have a `status` field allowing admins to disable annotations without deleting them	P2

5.5.3 Moderation Dashboard

ID	Requirement	Priority
F5.9	Admins shall have a dashboard showing all pending community notes	P0
F5.10	The dashboard shall allow batch actions (approve/reject multiple notes)	P2
F5.11	The dashboard shall display note content, author, associated DZ, and submission date	P0
F5.12	The moderation queue shall show a count badge of pending items	P1

---

6. Non-Functional Requirements

6.1 Performance

ID	Requirement	Target
NFR-1	Initial page load (Time to Interactive)	≤ 3 seconds on 4G connection
NFR-2	API response time (p95)	≤ 500ms
NFR-3	Map tile load time	≤ 2 seconds for initial viewport
NFR-4	Search response time	≤ 300ms (client-side) / ≤ 500ms (server-side)

NFR-5	Lighthouse Performance score	≥ 80 (mobile)

6.2 Availability & Reliability

ID	Requirement	Target
NFR-6	Uptime target	99.5% (monthly)
NFR-7	Data durability	Cosmos DB default replication
NFR-8	Graceful degradation	App functional with map service outage (show static fallback)

6.3 Security

ID	Requirement	Target
NFR-9	Authentication	All write operations require authentication
NFR-10	Authorization	Role-based access (user, admin) enforced at API level
NFR-11	Data transmission	HTTPS enforced on all endpoints
NFR-12	Input validation	Server-side validation on all user inputs
NFR-13	XSS prevention	Sanitize all user-generated content before rendering
NFR-14	Rate limiting	API rate limiting on unauthenticated endpoints

6.4 Accessibility

ID	Requirement	Target
NFR-15	WCAG compliance	Level AA for all non-map content
NFR-16	Keyboard navigation	All interactive elements keyboard-accessible
NFR-17	Screen reader support	Semantic HTML, ARIA labels on interactive components

6.5 Browser & Device Support

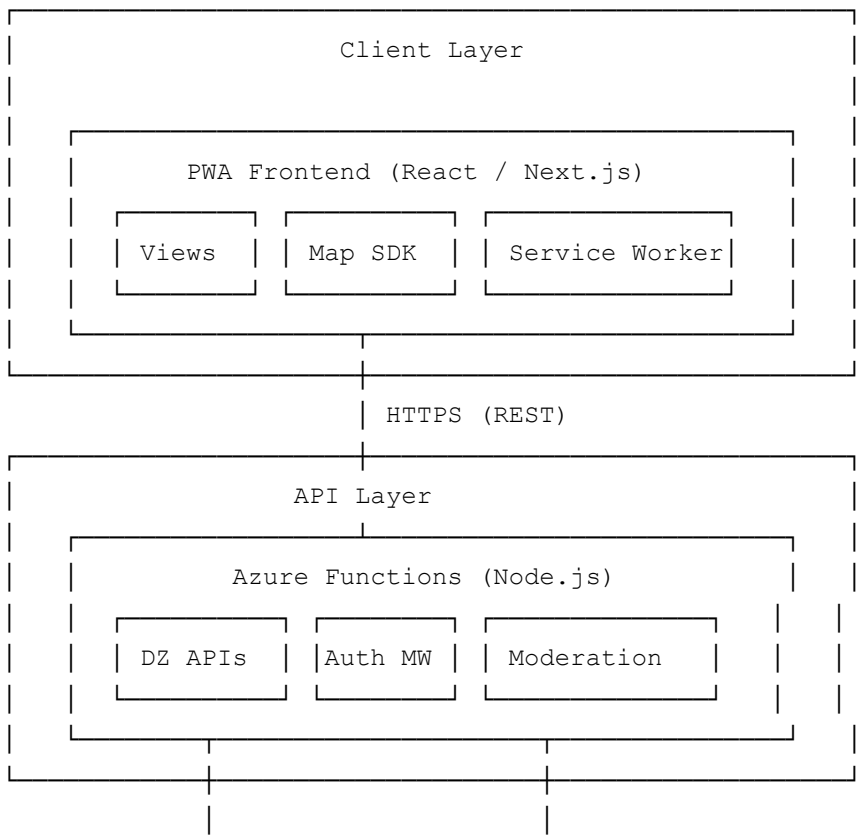
ID	Requirement	Target
NFR-18	Mobile browsers	Safari (iOS 15+), Chrome (Android 10+)
NFR-19	Desktop browsers	Chrome, Firefox, Safari, Edge (latest 2 versions)
NFR-20	Minimum viewport	320px width
NFR-21	PWA installability	Meets PWA install criteria (manifest, service worker, HTTPS)

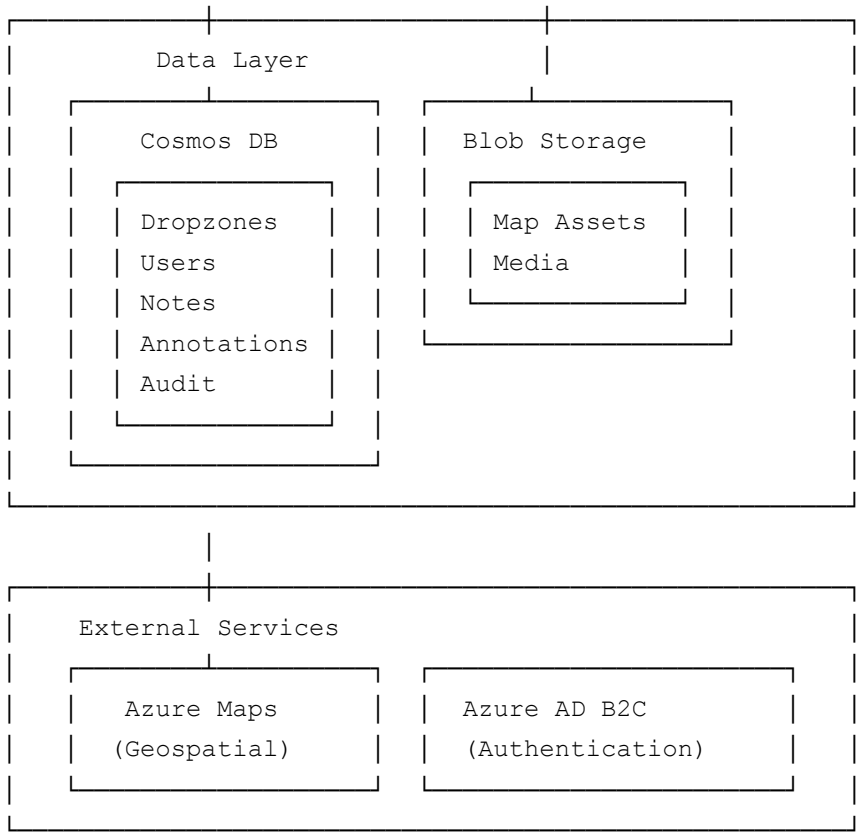
---

7. System Architecture

7.1 Architecture Overview

SkyNav follows a serverless, API-first architecture pattern. The frontend is a statically deployed PWA that communicates with a serverless API layer, backed by a managed NoSQL database and cloud storage.





7.2 Architecture Decisions

Decision	Choice	Rationale
Frontend framework	React + Next.js	SSR/SSG support, strong PWA ecosystem, future-proof
Hosting model	Serverless (Azure Functions)	Cost-efficient at low scale, auto-scaling at high scale
Database	Cosmos DB (NoSQL)	Flexible schema for evolving DZ data, geo-queries, serverless tier available
Map provider	Azure Maps	Consistent Azure ecosystem, satellite imagery, drawing tools SDK
Authentication	Azure AD B2C	Enterprise-grade auth with social login, free tier covers MVP
Storage	Azure Blob Storage	Low-cost media storage, CDN-ready
Architecture style	API-first, stateless	Clean separation enables future native mobile clients

7.3 Key Design Constraints

- **Stateless API:** No server-side sessions. All auth via JWT tokens.
- **Single-region deployment** for MVP. Multi-region is a post-MVP concern.
- **Admin-seeded map data:** No user-submitted geospatial annotations in MVP.
- **Moderation bottleneck accepted:** All community notes require manual approval. This is acceptable given expected MVP volume.

---

8. Technology Stack

8.1 Frontend

Component	Technology	Version	Notes
Framework	Next.js	14.x+	App Router, SSG for DZ pages
UI Library	React	18.x+	—
Styling	Tailwind CSS	3.x+	Utility-first, mobile-first
Map SDK	Azure Maps Web SDK	Latest	Satellite view, drawing tools
State Management	React Context + hooks	—	Zustand if complexity warrants
PWA	next-pwa or Workbox	—	Service worker, offline shell
HTTP Client	fetch / SWR	—	SWR for data caching
TypeScript	TypeScript	5.x+	Strict mode enabled

8.2 Backend

Component	Technology	Version	Notes
Runtime	Azure Functions	v4 (Node.js)	HTTP-triggered, stateless
Language	TypeScript	5.x+	Shared types with frontend
Validation	Zod	Latest	Schema validation on all inputs

Auth middleware	Custom + Azure AD B2C SDK	—	JWT verification

8.3 Data & Storage

Component	Technology	Notes
Primary database	Azure Cosmos DB	SQL API, serverless tier for MVP
Blob storage	Azure Blob Storage	Map assets, future media
CDN	Azure CDN (optional)	Post-MVP for static asset performance

8.4 Infrastructure & DevOps

Component	Technology	Notes
Source control	Git (GitHub)	Monorepo structure
CI/CD	GitHub Actions	Build, test, deploy pipeline
Frontend hosting	Azure Static Web Apps	Integrated with Functions backend
IaC (optional)	Azure CLI / Bicep	Resource provisioning scripts
Monitoring	Application Insights	Logging, error tracking, performance

---

9. Data Models & Schema Design

9.1 Cosmos DB Collections

The database uses five collections, each with a defined partition key for query efficiency.

9.1.1 Dropzones Collection

Partition Key: `/id`

```
{
  "id": "string (GUID)",
  "name": "string",
  "slug": "string (URL-friendly, unique)",
```

```
  "location": {
    "city": "string",
    "state": "string",
    "country": "string (ISO 3166-1 alpha-2)",
    "coordinates": {
      "latitude": "number",
      "longitude": "number"
    },
    "address": "string (optional)"
  },
  "operationalInfo": {
    "aircraft": ["string"],
    "maxAltitude": "number (feet AGL)",
    "jumpTicketPrice": "string",
    "manifestProcedure": "string",
    "contactPhone": "string (optional)",
    "contactEmail": "string (optional)",
    "website": "string (optional)",
    "hoursOfOperation": "string",
    "uspaGroupMember": "boolean",
    "additionalNotes": "string (optional)"
  },
  "landingSummary": {
    "description": "string",
    "surfaceType": "string (grass | gravel | tarmac | mixed)",
    "elevation": "number (feet MSL, optional)",
    "generalNotes": "string (optional)"
  },
  "travelInfo": {
    "nearestAirport": "string",
    "drivingNotes": "string (optional)",
    "lodgingOptions": "string (optional)",
    "campingAvailable": "boolean",
    "localAmenities": "string (optional)",
    "additionalNotes": "string (optional)"
  },
  "status": "string (active | inactive | draft)",
  "createdAt": "string (ISO 8601)",
  "updatedAt": "string (ISO 8601)",
  "createdBy": "string (admin user ID)"
}
```

9.1.2 MapAnnotations Collection

Partition Key: `/dropzoneId`

```
{
  "id": "string (GUID)",
  "dropzoneId": "string (FK → Dropzones)",
  "type": "string (landing | hazard | pattern)",
  "geometry": {
    "type": "string (Point | Polygon | LineString)",
    "coordinates": "array (GeoJSON format)"
  },
  "label": "string",
  "description": "string",
  "style": {
    "color": "string (hex, optional)",
    "icon": "string (optional)"
  },
  "status": "string (active | disabled)",
  "sortOrder": "number (optional)",
  "createdAt": "string (ISO 8601)",
  "updatedAt": "string (ISO 8601)",
  "createdBy": "string (admin user ID)"
}
```

9.1.3 CommunityNotes Collection

Partition Key: `/dropzoneId`

```
{
  "id": "string (GUID)",
  "dropzoneId": "string (FK → Dropzones)",
  "userId": "string (FK → Users)",
  "authorDisplayName": "string",
  "content": "string (max 1000 chars)",
  "status": "string (pending | approved | rejected)",
  "moderatedBy": "string (admin user ID, optional)",
  "moderatedAt": "string (ISO 8601, optional)",
  "rejectionReason": "string (optional)",
  "createdAt": "string (ISO 8601)"
}
```

9.1.4 Users Collection

Partition Key: `/id`

```
{
  "id": "string (GUID, matches auth provider ID)",
```

```
  "email": "string",
  "displayName": "string",
  "role": "string (user | admin)",
  "favorites": ["string (dropzone IDs)"],
  "createdAt": "string (ISO 8601)",
  "lastLoginAt": "string (ISO 8601)"
}
```

9.1.5 AuditHistory Collection

Partition Key: `/entityId`

```
{
  "id": "string (GUID)",
  "entityType": "string (dropzone | annotation | note | user)",
  "entityId": "string",
  "action": "string (create | update | delete | approve | reject)",
  "performedBy": "string (user ID)",
  "timestamp": "string (ISO 8601)",
  "changes": "object (optional, before/after diff)"
}
```

9.2 Indexing Strategy

Collection	Indexed Fields	Index Type
Dropzones	`name`, `slug`, `location.state`, `status`	Composite
Dropzones	`location.coordinates`	Geospatial
MapAnnotations	`dropzoneId`, `type`, `status`	Composite
CommunityNotes	`dropzoneId`, `status`, `createdAt`	Composite
Users	`email`	Unique
AuditHistory	`entityId`, `timestamp`	Composite

---

10. API Specification

10.1 API Conventions

- **Base URL:** `https://<app>.azurewebsites.net/api`
- **Format:** JSON request and response bodies
- **Authentication:** Bearer token (JWT) in `Authorization` header for protected endpoints
- **Versioning:** URL prefix (`/api/v1/`) — implemented when needed post-MVP
- **Error format:**

```
{
  "error": {
    "code": "string",
    "message": "string"
  }
}
```

10.2 Endpoint Summary

Public Endpoints (No Auth Required)

Method	Path	Description
`GET`	`/api/health`	Health check / API status
`GET`	`/api/dropzones`	List all active dropzones (paginated)
`GET`	`/api/dropzones/:slug`	Get single dropzone profile by slug
`GET`	`/api/dropzones/:id/annotations`	Get map annotations for a dropzone
`GET`	`/api/dropzones/:id/notes`	Get approved community notes for a dropzone

Authenticated Endpoints (User Role)

Method	Path	Description
`POST`	`/api/notes`	Submit a community note
`GET`	`/api/me/favorites`	Get current user's favorites
`PUT`	`/api/me/favorites/:dropzoneId`	Add dropzone to favorites
`DELETE`	`/api/me/favorites/:dropzoneId`	Remove dropzone from favorites

GET	/api/me/profile	Get current user profile

Admin Endpoints (Admin Role)

Method	Path	Description
POST	/api/admin/dropzones	Create a new dropzone
PUT	/api/admin/dropzones/:id	Update a dropzone
DELETE	/api/admin/dropzones/:id	Soft-delete a dropzone
POST	/api/admin/annotations	Create a map annotation
PUT	/api/admin/annotations/:id	Update a map annotation
DELETE	/api/admin/annotations/:id	Delete a map annotation
GET	/api/admin/notes	List all notes (filterable by status)
PATCH	/api/admin/notes/:id	Update note status (approve/reject)
DELETE	/api/admin/notes/:id	Delete a note

10.3 Detailed Endpoint Specifications

`GET /api/dropzones`

Query Parameters:

Parameter	Type	Required	Description
search	string	No	Text search on name and location
state	string	No	Filter by state
page	number	No	Page number (default: 1)
limit	number	No	Items per page (default: 20, max: 50)

Response: `200 OK`

```
{
```

```
  "data": [
    {
      "id": "...",
      "name": "Skydive DeLand",
      "slug": "skydive-deland",
      "location": { "city": "DeLand", "state": "FL", "country": "US" },
      "status": "active"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 20,
    "total": 8,
    "totalPages": 1
  }
}
```

`POST /api/notes`

Auth: Required (user role)

Request Body:

```
{
  "dropzoneId": "string (required)",
  "content": "string (required, 1–1000 chars)"
}
```

Response: **`201 Created`**

```
{
  "id": "...",
  "dropzoneId": "...",
  "status": "pending",
  "createdAt": "2026-02-07T12:00:00Z"
}
```

Error Cases:

Status	Code	Condition
400	`VALIDATION_ERROR`	Missing fields, content exceeds 1000 chars
401	`UNAUTHORIZED`	No valid auth token

404	`NOT_FOUND`	Dropzone ID does not exist
429	`RATE_LIMITED`	User has 3+ pending notes for this DZ

---

11. Authentication & Authorization

11.1 Authentication Provider

Azure AD B2C (or equivalent lightweight provider such as Clerk or Auth0 as a fallback) handles user identity.

Supported Sign-In Methods (MVP):

- Email + password
- Google OAuth
- Apple Sign-In (recommended for iOS PWA users)

11.2 Token Flow

1. User authenticates via the provider's hosted login page
2. Provider issues a JWT access token
3. Frontend stores the token in memory (not localStorage for security)
4. Frontend attaches `Authorization: Bearer <token>` to all authenticated API requests
5. Azure Functions middleware validates the JWT signature and extracts the user's `sub` claim as the user ID

11.3 Role Definitions

Role	Capabilities
Anonymous	Browse directory, view DZ profiles, view maps, view approved notes
User	All anonymous capabilities + submit notes, manage favorites, view own profile
Admin	All user capabilities + CRUD dropzones, manage annotations, moderate notes, view audit history

11.4 Role Assignment

- New users default to the `user` role
- Admin role is assigned manually (database update or admin panel) — no self-service admin registration in MVP

---

12. User Interface Specification

12.1 Design Principles

- **Mobile-first:** Design for 375px viewport, scale up to desktop
- **Touch-friendly:** Minimum 44px tap targets, generous spacing
- **Fast navigation:** Bottom tab bar on mobile for primary sections
- **Minimal chrome:** Content-forward, minimal UI decoration
- **Offline-aware:** Graceful degradation when offline (show cached data, disable submissions)

12.2 Information Architecture

```
├── Home / Directory (default view)
│   ├── Search Bar
│   └── DZ Listing Cards
│
├── DZ Profile (/:slug)
│   ├── Header (name, location, status)
│   ├── Operational Info Section
│   ├── Landing Map (embedded, expandable)
│   ├── Landing Summary
│   ├── Community Notes
│   │   └── Submit Note Form (authenticated)
│   └── Travel Info Section
│
├── My DZs / Favorites
│   ├── Saved DZ Cards
│   └── Travel Mode View (per DZ)
│
├── Profile / Settings
│   ├── Display Name
│   ├── Account Info
│   └── Sign Out
```

```
│
└── Admin Console (/admin, admin role only)
    ├── DZ Manager
    │   ├── Create / Edit DZ Form
    │   └── DZ List with Status
    ├── Annotation Manager
    │   ├── Map Editor
    │   └── Annotation List
    └── Moderation Queue
        └── Pending Notes List
```

12.3 Key Screen Specifications

12.3.1 Directory Screen (Home)

- **Layout:** Full-width search bar at top, card grid/list below
- **Card contents:** DZ name, city/state, thumbnail or status badge
- **Actions:** Tap card → navigate to DZ profile
- **Mobile:** Single-column card list
- **Desktop:** 2–3 column card grid

12.3.2 DZ Profile Screen

- **Layout:** Single-column scrollable page
- **Sections (top to bottom):**
  1. Hero: DZ name, location, favorite toggle
  2. Operational Info: structured key-value display
  3. Landing Map: interactive map embed (~300px height on mobile, expandable)
  4. Landing Summary: text block
  5. Community Notes: reverse-chronological list with submission form
  6. Travel Info: structured key-value display
- **Sticky element:** Favorite toggle accessible on scroll (optional)

12.3.3 Admin Console

- **Layout:** Desktop-optimized, sidebar navigation

- **Sections:** DZ Manager, Annotation Editor, Moderation Queue
- **DZ Form:** Multi-section form with validation (operational info, location, travel info)
- **Annotation Editor:** Map interface with drawing tools (polygon, marker, line) and metadata form
- **Moderation Queue:** Table view with bulk actions, status filters

---

13. Content Strategy

13.1 Seed Content Plan

The MVP launches with 5–10 pilot dropzones. Seed DZs should represent a mix of:

- High-traffic DZs with significant visitor volume
- Geographic diversity (at least 2–3 different states/regions)
- DZs where the builder has personal knowledge or contacts

Suggested Seed Criteria:

Criterion	Reasoning
USPA Group Member DZs	Verified operations, public info available
Year-round operations	Maximizes utility for traveling skydivers
Known canopy piloting programs	Relevant to core user base with advanced interests
Active boogie/event schedule	Attracts travelers

13.2 Content Quality Standards

Content Type	Quality Bar
Operational Info	Verified against DZ website or direct contact. Updated at least quarterly.
Landing Map Annotations	Based on aerial imagery review and, ideally, first-hand knowledge.
Community Notes	Moderated for accuracy, relevance, and tone. No promotional content, rumors, or hearsay.

Travel Info	Factually accurate, practical, sourced from maps/DZ websites.

13.3 Moderation Guidelines

Community notes are approved if they are:

1. **Factual and actionable** — provide information a visiting jumper can use
2. **Specific to the DZ** — not generic skydiving advice
3. **Respectful in tone** — no personal attacks, grievances, or inappropriate language
4. **Not duplicative** — add information not already covered in the DZ profile

Notes are rejected if they contain promotional content, unverifiable safety claims, personal disputes, or off-topic commentary.

---

14. Deployment Architecture

14.1 Environment Tiers

Environment	Purpose	URL Pattern	Data
Dev	Active development, feature branches	`dev-skynav.azurestaticapps.net`	Synthetic/test data
Staging	Pre-production validation	`staging-skynav.azurestaticapps.net`	Copy of production data
Production	Live user-facing environment	`skynav.app` (or similar)	Real data

14.2 CI/CD Pipeline (GitHub Actions)

```
Push to feature branch:
  → Lint + Type Check
  → Unit Tests
  → Build Frontend + Backend
  → Deploy to Dev (automatic)

Pull Request to main:
```

```
  → All above checks
  → Integration Tests
  → Deploy to Staging (automatic)
  → Manual approval gate

Merge to main:
  → Deploy to Production
  → Smoke tests
  → Monitor Application Insights
```

14.3 Repository Structure (Monorepo)

```
skynav/
├── apps/
│   ├── web/                    # Next.js PWA frontend
│   │   ├── src/
│   │   │   ├── app/            # Next.js app router pages
│   │   │   ├── components/     # Shared UI components
│   │   │   ├── lib/            # Utilities, API client, hooks
│   │   │   └── styles/         # Global styles, Tailwind config
│   │   ├── public/             # Static assets, PWA manifest
│   │   └── package.json
│   └── api/                    # Azure Functions backend
│       ├── src/
│       │   ├── functions/      # Function definitions
│       │   ├── middleware/     # Auth, validation, error handling
│       │   ├── services/      # Business logic
│       │   └── models/        # Data access layer
│       └── package.json
├── packages/
│   └── shared/                # Shared TypeScript types, constants
│       ├── src/
│       │   ├── types/         # Shared interfaces
│       │   └── constants/     # Shared enums, config
│       └── package.json
├── infra/                     # Infrastructure scripts (Bicep/CLI)
├── docs/                      # Documentation
├── .github/
│   └── workflows/             # CI/CD pipelines
├── package.json               # Workspace root
└── turbo.json                 # Monorepo build orchestration
```

---

15. Development Milestones

Milestone 1 — Foundation & Infrastructure (Weeks 1–2)

Objective: Establish project structure, CI/CD, and Azure resource provisioning.

Task	Deliverable	Acceptance Criteria
Initialize monorepo with Turborepo	Repository with `apps/web`, `apps/api`, `packages/shared`	Successful `turbo build` across all packages
Scaffold Next.js app with Tailwind, TypeScript	Running dev server at localhost	Mobile-responsive shell with placeholder nav
Scaffold Azure Functions project	Function app with health endpoint	`GET /api/health` returns `200 OK`
Provision Azure resources	Cosmos DB, Blob Storage, Static Web App	Resources accessible, connection strings configured
Configure CI/CD pipeline	GitHub Actions workflows	Push to branch triggers build and deploy to dev
Implement auth skeleton	Azure AD B2C tenant, login flow	User can sign in/out, JWT issued

Exit Criteria: Running frontend shell + API health endpoint deployed to dev environment.

Milestone 2 — Dropzone Directory (Weeks 3–4)

Objective: Full DZ browsing workflow with seeded content.

Task	Deliverable	Acceptance Criteria
Implement Dropzone data model	Cosmos DB collection, TypeScript types	Schema matches spec, CRUD operations work
Build DZ CRUD API endpoints	REST endpoints per spec	All endpoints return correct status codes and data
Build directory listing page	Responsive card grid with search	User can browse, search, and navigate to profiles
Build DZ profile page	Detail page with all sections	All profile fields rendered, deep-linkable URL

Seed 5 pilot DZs	Test data in Cosmos DB	Profiles complete with operational + travel info

Exit Criteria: Browsable, searchable DZ directory with seeded data.

Milestone 3 — Interactive Map (Weeks 5–7)

Objective: Map rendering with annotation overlays on DZ profiles.

Task	Deliverable	Acceptance Criteria
Integrate Azure Maps SDK	Map component rendering in profile	Satellite view loads, pan/zoom works on mobile
Implement MapAnnotation data model	Cosmos DB collection, API endpoints	CRUD works, GeoJSON coordinates stored
Build annotation overlay rendering	Landing areas, hazards, patterns visible	Annotations display with correct colors and labels
Build annotation tooltips	Tap/click shows description	All annotation types show label + description
Seed annotations for pilot DZs	Map data for 5 DZs	At least landing area + 2 hazards per DZ
Admin: annotation editor UI	Map-based editing interface	Admin can place/edit/delete annotations on a map

Exit Criteria: Landing maps with interactive hazard and pattern overlays for all seeded DZs.

Milestone 4 — Community Notes (Weeks 8–9)

Objective: Note submission and moderation pipeline.

Task	Deliverable	Acceptance Criteria
Implement CommunityNote data model	Cosmos DB collection, API endpoints	Submit creates pending note, status transitions work
Build note submission UI	Form on DZ profile page	Authenticated user can submit, sees confirmation

Build note display on profiles	Note list on DZ profile	Only approved notes visible, reverse chronological
Build moderation dashboard	Admin view with approve/reject actions	Admin sees pending queue, can approve/reject
Input validation + rate limiting	Server-side checks	1000 char limit enforced, 3-note pending limit works

Exit Criteria: Users can submit notes, admins can moderate, approved notes appear on profiles.

Milestone 5 — Favorites & Travel Mode (Week 10)

Objective: Personal utility features for authenticated users.

Task	Deliverable	Acceptance Criteria
Build favorites API	Add/remove/list endpoints	Favorites persisted to user document
Build favorites toggle UI	Heart/star icon on directory + profile	Immediate visual feedback, works on both views
Build “My DZs” dashboard page	Favorites list view	User sees saved DZs with quick access
Build travel mode view	Structured checklist + key info	Travel info surfaced prominently, checklist displayed

Exit Criteria: Users can save DZs and access travel preparation tools.

Milestone 6 — Admin Console (Weeks 11–12)

Objective: Complete content management interface.

Task	Deliverable	Acceptance Criteria
Build admin layout + navigation	Sidebar nav, role-gated routing	Only admins can access /admin routes
Build DZ creation/editing form	Multi-section form with validation	All DZ fields editable, saves to Cosmos DB
Build DZ listing	Table view with status, edit,	Admin can manage all DZs from one

management	delete	view
Finalize annotation manager	Integrated map editor + list	Full annotation CRUD through admin UI
Finalize moderation dashboard	Queue with filters, counts	Badge shows pending count, bulk actions (stretch)
Implement audit logging	AuditHistory writes on all admin actions	Audit trail visible for admin changes

Exit Criteria: Full admin control panel for all content management operations.

Milestone 7 — MVP Polish & Testing (Week 13)

Objective: Production readiness.

Task	Deliverable	Acceptance Criteria
Mobile UX audit	Fix list of mobile-specific issues	All primary flows work smoothly on iOS + Android
Performance tuning	Lighthouse ≥ 80, optimized bundle	TTI ≤ 3s on 4G
Error handling & edge cases	Graceful error states, empty states	No unhandled errors in primary flows
PWA validation	Manifest, service worker, installability	App installable on iOS + Android
Analytics integration	Application Insights events	Page views, key actions tracked
End-to-end testing	Manual test pass on all features	All acceptance criteria verified

Exit Criteria: MVP ready for pilot testers.

---

16. Testing Strategy

16.1 Testing Levels

Level	Scope	Tools	Coverage Target
Unit	Individual functions, utilities, components	Vitest, React Testing Library	≥ 70% for business logic
Integration	API endpoints, database operations	Vitest, supertest	All API endpoints
Component	UI component rendering and interaction	React Testing Library	Key interactive components
E2E	Critical user flows	Playwright (post-MVP)	Primary flows only
Manual	Full UX validation	Checklist-driven	All features, all devices

16.2 Critical Test Scenarios

Scenario	Type	Priority
DZ directory loads and displays all active DZs	Integration	P0
Search returns correct results for name and location	Integration	P0
DZ profile renders all sections correctly	Component	P0
Map renders with annotations on DZ profile	Component	P0
Unauthenticated user cannot submit a note	Integration	P0
Submitted note enters pending status	Integration	P0
Admin approve/reject changes note visibility	Integration	P0
Favorite toggle persists across sessions	Integration	P1
Admin DZ form validates required fields	Component	P1
API returns 401 for missing/invalid token	Integration	P0

---

17. Scalability & Future Roadmap

17.1 MVP Scaling Characteristics

Component	Scaling Model	MVP Capacity
Frontend	Static hosting (CDN)	Effectively unlimited
API	Azure Functions consumption plan	Auto-scales to demand
Database	Cosmos DB serverless	Auto-scales RU/s with demand
Maps	Azure Maps S0 tier	Rate-limited by subscription tier

17.2 Post-MVP Roadmap (Prioritized)

Phase	Feature	Description
v1.1	Enhanced search	Geospatial “near me” search, map-based DZ discovery
v1.1	User-submitted DZ updates	Structured submission form for DZ info corrections
v1.2	Weather integration	Current conditions and wind forecast for each DZ
v1.2	Offline DZ profiles	Service worker caching of favorited DZ data
v2.0	AI-assisted jump planning	Personalized DZ recommendations based on experience level and preferences
v2.0	Performance tracking	Integration with FlySight GPS data for canopy performance analysis
v2.0	Native mobile apps	React Native or Expo wrappers for iOS and Android
v3.0	Real-time collaboration	Live wind/condition reports from jumpers on-site
v3.0	Social features	Follow jumpers, trip planning groups

---

18. Risk Assessment

Risk	Likelihood	Impact	Mitigation
Map SDK complexity — Azure Maps			Allocate extra buffer in

integration takes longer than estimated	Medium	High	Milestone 3; prototype map component early
Content cold start — Insufficient seed content reduces perceived value	High	High	Pre-commit to seeding 5+ DZs with complete profiles before pilot launch
Auth complexity — Azure AD B2C configuration overhead	Medium	Medium	Consider Clerk or Auth0 as lower-friction alternatives if setup exceeds 2 days
Solo builder bottleneck — Moderation volume exceeds single admin capacity	Low (MVP)	Medium	Acceptable at MVP scale; plan automated pre-screening for post-MVP
Scope creep — Feature requests during development expand MVP	High	High	Strict adherence to non-goals list; park all new ideas in post-MVP backlog
Mobile UX gaps — PWA limitations on iOS (push notifications, etc.)	Medium	Low	Communicate limitations; PWA provides sufficient utility for MVP validation
Data accuracy decay — DZ info goes stale without update mechanism	Medium	High	Establish quarterly review cadence; post-MVP: enable community corrections

---

19. Appendix

A. Glossary

Term	Definition
DZ	Dropzone — a facility where skydiving operations are conducted
USPA	United States Parachute Association — the governing body for sport parachuting in the US

AGL	Above Ground Level — altitude measurement referenced to the ground
MSL	Mean Sea Level — altitude measurement referenced to sea level
Canopy	The parachute used for the controlled descent and landing
Pattern	The flight path a canopy pilot follows during the landing approach
Swoop / Swooping	High-performance canopy landing involving a high-speed approach
Boogie	A skydiving event or festival, often attracting traveling jumpers
Manifest	The DZ's scheduling system for assigning jumpers to aircraft loads
PWA	Progressive Web Application — a web app installable on mobile devices
JWT	JSON Web Token — a standard for secure token-based authentication

B. Reference Documents

- Azure Maps Web SDK Documentation: https://learn.microsoft.com/en-us/azure/azure-maps/
- Azure Functions Developer Guide: https://learn.microsoft.com/en-us/azure/azure-functions/
- Cosmos DB SQL API Reference: https://learn.microsoft.com/en-us/azure/cosmos-db/
- Azure AD B2C Documentation: https://learn.microsoft.com/en-us/azure/active-directory-b2c/
- Next.js Documentation: https://nextjs.org/docs
- USPA Dropzone Directory: https://uspa.org/find/dropzones

C. Estimated MVP Timeline Summary

```
Milestone 1: Foundation & Infrastructure ........ Weeks 1–2
Milestone 2: Dropzone Directory ................. Weeks 3–4
Milestone 3: Interactive Map .................... Weeks 5–7
Milestone 4: Community Notes .................... Weeks 8–9
Milestone 5: Favorites & Travel Mode ............ Week 10
Milestone 6: Admin Console ...................... Weeks 11–12
Milestone 7: MVP Polish & Testing ............... Week 13
                                                  ──────────
```

Total Estimated Duration:	10–13 weeks

End of Specification Document