# Timeline Module - Full Technical + RAG Documentation (v2)

## 0. Purpose of this Document

This document is the **authoritative source of truth** for the ConnecWrk Timeline module. It is designed for: - Frontend engineers building / maintaining the Timeline experience at `/timeline` - Backend/API engineers powering feed, reactions, social graph, premium prompts, etc. - AI / RAG engineers who need to ingest this product surface into an internal knowledge base (for question-answering, support automation, analytics, etc.)

This version adds: 1. Deep data model + API mapping 2. UI/UX behavior and state rules 3. Privacy / security constraints 4. **RAG ingestion requirements**, including namespace layout, chunking strategy, metadata schema, retrieval patterns, and query routing guidance.

---

## 1. High-Level Module Overview

The **Timeline** is ConnecWrk's main social feed. It combines: - Professional content posts (news, opinions, announcements) - MSME promotions / business updates - Freelancer showcases - Talent / Artist discovery snippets - Platform announcements and articles - Engagement signals (reactions, shares) - Growth surfaces ("Create your Freelancer Page", "Post Assignment", etc.) - Right Sidebar discovery (Featured MSMEs, Featured Freelancers, Invite Friends, Premium upsell) - Left Sidebar profile module and quick navigation

This page is both **content consumption** and **onboarding orchestration**, nudging a new member to: - Complete their profile - Register as a Freelancer, Talent, or MSME - Post Assignments / Jobs - Invite contacts - Upgrade to Premium

---

## 2. Major Page Regions / Components

These regions exist simultaneously on `/timeline`:

### 2.1 Left Sidebar Component

**Purpose:** personal identity, credibility, and navigation.

**Features:** - Banner / cover photo (uploadable) - Profile avatar + premium ring (gold border if premium) - Name, professional title, profile completion % - CTA card (rotates between "Create Business Page", "Create Talent Page", "Create Freelancer Page", "Post Assignment") - Stats: Connections count, Profile Views - Quick nav links: - MSME Connect - Freelancer Connect - Assignments - Talent Connect - MSME Jobs - Articles and Trivia

**Key APIs:** - `GET /api/auth/getuserprofile`
Returns: name, avatar, cover image, premium status, profile completion, profile view count. - `POST /`

`api/setting/userbannerImage`
Upload new banner image (multipart/form-data with `bannerImage`). - `GET /api/contact/mycontactlist?page=1`
Used for connection count. - `GET /api/msme/msmelisting?page=1&type=self`
Used to determine if user already has MSME pages.

**Behavioral Rules:** - If profile completion <100%, show progress bar and link to "Click here to complete your profile". - CTA card is chosen from a pool (Freelancer Page / Talent Page / Business Page / Post Assignment) and can promote different flows depending on missing profile types. - Sidebar becomes sticky/fixed as user scrolls, but only after layout mount.

---

### 2.2 Registration Prompt Banner (Top of Feed)

**Purpose:** onboarding accelerator.

Cards shown: - **Are you a Freelancer? → Create Freelancer Page** - **Talent, Artist or Speaker? → Create Talent Page** - **Are you an MSME? → Create Business Page** - **Need a Freelancer? → Post Assignment** - **Looking for Talent/Artist for your event? → Search Talent/Artists**

**Additional Checks:** - We call: - `GET /api/msme/msmelisting?page=1&type=self` (does user already have business pages?) - `GET /api/talent/mytalentlist?page=1` - `GET /api/project/myfreelancerlist?page=1` - If the user already has a given profile type, that CTA may still show (marketing), but the copy/targeting logic may change in future (planned: hide redundant CTAs). - Banner has a close `X` button so it can be dismissed for session UX.

---

### 2.3 Timeline Feed / Timeline Scroll Component

**Purpose:** infinite scroll list of posts.

**Core API:** - `GET /api/share/shareupdate?page={page}` - Returns `data[]`: - `id` (postId) - `postTitle` - `postText` (HTML or rich text) - `image` or `imageUrl` - `postType` - 1 = Article / News / Text Post - 2 = MSME / Business promo - 3 = Freelancer highlight / service promo - 4 = Generic platform update / other content - 5 = YouTube / external media share - `createdAt` - `urlSharedUserId` (author / source userId) - `logUserTotalLike`, `totalLikes` - Also pagination: `currentPage`, `totalPage`.

**Scrolling Logic:** - Uses Intersection Observer on a sentinel div at the bottom of the list. - When sentinel becomes visible AND `currentPage < totalPage`, we fetch the next page. - Loading spinner appears while fetching more.

**Reaction Prefetch:** - After posts load, we also may prefetch reaction data for visible posts, so we can render active reaction state.

**Error Handling:** - If page fetch fails: - Show non-blocking toast. - Keep already-loaded posts. - Attempt retry when user scrolls again.

---

### 2.4 Post Card / Post Content Component

Each feed item renders with consistent structure:

**Header (Post Header Component):** - Brand identity / source. Often shows ConnecWrk logo + "ConnecWrk" + relative time ("2 hours ago"). - Ellipsis menu ( `...` ) which opens share menu: - "Copy Link" → builds sharable URL `https://connecwrk.com/single-post?postId={postId}`, uses `navigator.clipboard.writeText`. - No direct API call in header, but uses current `postId` for link generation.

**Body:** - Title (truncated if long) - Content snippet: - For long text: show first ~150 chars / ~40 words, followed by "Read more" link. - Supports HTML segments via safe parsing. - Media: - If `postType===5` and YouTube URL present → render responsive iframe. - If `image` or `imageUrl` present → render fixed-height preview with rounded corners.

**Connection CTA (smart social graph chip):** - For posts representing a user/org (MSME, Freelancer, etc.), we may render a connection button next to the title area. - Logic uses `GET /api/contact/frienduserinfo?page=1&frienduserId={authorId}` to detect: - `isLogUserFriend`: "Yes" / "No" - `isRequestSentToFriend`: "Yes" / "No" - We DO NOT call this until the card is in viewport (lazy optimize). - Button states: - **Connect** (green, plus icon) if not connected and no request pending. - **Pending** (orange pulse) if request sent. - Hidden if already connected OR if it's the logged-in user's own post. - Clicking "Connect" calls: - `GET /api/contact/addcontact?frienduserId={authorId}` which triggers a pending request.

**Footer (Post Footer Component):** - Reactions bar + Share button. - Reactions: - Hover/click reveals a 6-option emoji palette: 1. 👍Like (type=1, color #ff7800) 2. 👏Clap (type=2, color #FF9800) 3. Support (type=3, color #4CAF50) 4. 👊Appreciate/Love (type=4, color #E0245E) 5. ❤️Insightful (type=5, color #FFC107) 6. 😂Funny (type=6, color #03A9F4) - Current user's reaction is highlighted and shown inline. - Backed by: - `POST /api/share/postlike` - Payload:

```
{
  "postId": "post123",
  "userId": "user456",
  "isLikeYN": "Y",      // "N" to remove
  "hoverType": 1,        // reaction type 1-6
  "urlSharedUserId": "user789" // author id
}
```

- Likes / reaction count is clickable → opens Likes Popup.

---

### 2.5 Likes Popup Component

**Purpose:** show *who* reacted and *how*.

**Parent fetches before open:** `GET /api/share/getalluserlike?pages=1&postid={postId}` → returns array of: - `userId` - `userModel.firstName`, `userModel.lastName`, `userModel.profilePhoto` - `hoverType` (maps to 👍👏 👊❤️😂) - `createdAt`

**UI:** - Modal with scroll (~300px height, 400px wide) - Avatar + name + badge reaction emoji overlayed on avatar - Clicking avatar/name opens that user's profile in new tab

**Error handling:** - If fetch fails or returns empty → show graceful empty state.

---

### 2.6 Share Popup Component (Timeline Post Share Popup)

**Purpose:** internal share to contacts, or external share via WhatsApp.

**Data load:** - `GET /api/share/individualsharepostdetail?postId={postId}` returns: - `postTitle` - `postText` - `imageUrl` or `image` - metadata (author, timestamp)

**User actions:** 1. **Share with My Contacts**
`POST /api/share/sharewithmycontact` with body:

```
{
  "postId": "post123",
  "sharedata": "Optional personal message"
}
```

→ Sends to user's existing contacts in ConnecWrk.

1. **Send on WhatsApp**
   Opens WhatsApp (via deep link) with prefilled text that includes title, snippet, and canonical URL.

**Validation:** - User must be authenticated. - Must have a valid `postId`. - Shows toast on success/failure.

---

### 2.7 Right Sidebar Component

**Purpose:** discovery + monetization + referrals.

Subsections: 1. **Premium Upsell Widget** - Shown ONLY if the user is not premium. - CTA links to `/premium-membership`. 2. **Invite People to Join ConnecWrk** - Input box to invite via email. - `POST /api/contact/sendinvitation` with `{ email: "a@b.com, c@d.com" }`. 3. **Featured MSMEs / Featured Freelancers / Featured Talent** - APIs: - `GET /api/msme/featuredmsmelisting` - `GET /api/project/featuredfreelancerlist` - `GET /api/talent/alltalentlist?page=1` (then sample 3 profiles) - Shows avatar/logo, name, short role/industry, link to detail page. 4. **Ad / Promo block** - Reserved marketing / ad slot.

**Sticky behavior:** - Sidebar locks into fixed position after ~515px scroll so it stays in view.

**Image fallback:** - If profile photo for talent/MSME/freelancer 404s, we show a default placeholder.

---

### 2.8 Premium Popup Component (Promotional Overlay)

**Purpose:** upsell Premium membership to non-premium users.

**Behavioral logic:** - Appears ~10s after load. - If dismissed, may reappear once more ~60s later. - Max 2 shows per session (tracked in `sessionStorage`). - Will NOT render at all if: - User is not logged in, OR - User already has premium `premiumStatus.planActive === true` (fetched from `GET /api/auth/getuserprofile`).

**CTA:** clicking banner navigates to `/premium-membership` for plan comparison + payment initiation.

**Close interaction:** fade-out 1s, then unmount.

---

### 2.9 Global Notifications / Toasts

Used throughout Timeline for: - Copy link success / failure - Share success / failure - Reaction add/remove success / failure - Contact request sent - API failure fallback

Toast content MUST: - Be non-sensitive. - Avoid leaking internal IDs other users shouldn't see. - Never expose email / phone of another user unless that's already visible in UI.

---

## 3. Data Model / Entities

Below are logical (not literal DB) models as they matter to Timeline.

### 3.1 Post Entity (Simplified)

```
{
  "id": "post123",
  "postTitle": "India-EU free trade pact to play key role amid US ...",
  "postText": "<p>New Delhi, Oct 27 (IANS)...</p>",
  "postType": 1,
  "image": "file123.jpg",    // or null
  "imageUrl": "https://...",  // external if present
  "urlSharedUserId": "user456",
  "createdAt": "2025-10-29T18:00:00Z",
  "logUserTotalLike": 5,
  "totalLikes": 25
}
```

### 3.2 Reaction Entity (User ↔ Post)

```
{
  "postId": "post123",
  "userId": "user789",
  "hoverType": 3,          // 1..6
```

```
  "isLikeYN": "Y",        // "N" if removed
  "createdAt": "2025-10-29T18:05:00Z"
}
```

### 3.3 Social Graph / Contact Relationship

```
{
  "userId": "user123",          // viewer
  "friendUserId": "user456",    // profile shown in card
  "isLogUserFriend": "Yes",     // or "No"
  "isRequestSentToFriend": "No",// "Yes" if pending
  "status": "PENDING" | "CONFIRMED" | ...,
  "mutualConnections": 5
}
```

### 3.4 User Profile (Minimal for Timeline)

```
{
  "userId": "user456",
  "firstName": "Lavanya",
  "lastName": "Singh",
  "premiumStatus": [
    { "planId": 2, "planActive": true }
  ],
  "profilePhoto": "lavanya.jpg",
  "coverImage": "banner123.jpg",
  "profileCompletion": 20,
  "profileViews": 0,
  "connectionsCount": 1
}
```

---

## 4. Security / Privacy Rules for Timeline

1. **Auth Required**
   `/timeline` is an authenticated experience. Most API calls expect a Bearer token.

2. **Profile Exposure**
   We show: first name, last name, profile photo, company / title.
   We DO NOT expose: private email, phone, or internal IDs in UI.

3. **Connection Requests**
   A user cannot connect to themselves. Backend must validate:

4. Sender ≠ Recipient
5. Not already friends

6. Not blocked

7. **Rate limiting:**

8. Connection requests should be throttled (e.g. max 50/hr) to prevent spam.

9. **Reactions:**

10. Reactions are public engagement signals. Anyone who sees a post can open Likes Popup and see who reacted.

11. We must still respect profile visibility settings (future: if a profile is restricted to "connections only", we may need to mask details).

12. **Clipboard / Share:**

13. Copy Link uses a predictable public URL of that post. If post is not public in future, we may need gated share links.

14. **Premium Popup + Session Storage:**

15. Session data about popup views is local to the browser session and is not PII.

---

## 5. Error Handling / Resilience Patterns

- **Feed load fail** → partial UI still works, show toast, allow retry.
- **Reaction fail** → rollback optimistic UI change, toast error.
- **Contact add fail** → revert button to "Connect".
- **Likes popup fail** → show empty state instead of crashing parent.
- **Share popup fail** → notify user, keep popup open so they can retry or copy manually.
- **Image load fail** → fallback placeholder avatar/logo.

---

## 6. Performance & UX Optimizations

- **Intersection Observer** for infinite scroll → no full-page reloads.
- **Lazy fetch of connection status** to avoid N+1 spam for every card that's off-screen.
- **Hover-triggered reaction palette** rather than always-rendered heavy UI.
- **Sticky sidebars** for persistent access to growth CTAs + discovery.
- **SessionStorage gating** for Premium Popup → prevents spammy upsell.
- **Debounced/controlled API calls** for reactions and likes list.

---

## 7. RAG Ingestion Requirements (Critical)

This section defines how to ingest the Timeline module into a Retrieval-Augmented Generation (RAG) system so an assistant can: - Answer internal questions from product / engineering / support teams -

Power help-center style Q&A for "How does the Timeline work?" - Troubleshoot UX issues ("Why didn't I see the Premium popup?")

### 7.1 RAG Namespace Strategy

**Namespaces** let us separate knowledge by product surface so retrieval stays relevant. We MUST index Timeline docs into the following logical namespaces:

1. `timeline.product`
2. High-level UX behavior
3. What users can do on /timeline
4. Feature summaries for onboarding, PMs, support, marketing
5. `timeline.api`
6. Endpoint definitions, required params, auth rules, response schemas
7. Error cases and side effects
8. Rate limiting / throttling notes
9. `timeline.ui-components`
10. Component responsibilities (Left Sidebar, Post Content, Post Footer, Likes Popup, Share Popup, Right Sidebar, Premium Popup, Registration Prompt Banner)
11. State/interaction rules
12. Conditional rendering logic
13. Sticky/fixed behavior rules
14. `timeline.security-privacy`
15. What PII is displayed
16. Rules for connection requests, throttling, visibility
17. Clipboard / sharing implications
18. `timeline.rag-routing`
19. FAQ-style Q&A seeds
20. Common debugging heuristics ("I can't like a post", "Why don't I see Connect button?")
21. Known edge behaviors (premium popup frequency)

    Rationale: Splitting into these namespaces improves retrieval precision. For example, support queries like "How do reactions work?" should primarily search `timeline.ui-components` and `timeline.api`, while compliance queries hit `timeline.security-privacy`.

### 7.2 Chunking Strategy

We MUST chunk the document into **small, semantically complete sections ~200–400 words each** so retrieval is specific and not overly broad. Guidelines: - Each chunk = 1 topic: e.g. "Post Footer reactions API" or "Premium Popup display logic". - Do NOT mix unrelated areas ("Left Sidebar" + "Likes Popup") in the same chunk. - Keep endpoint descriptions (method, path, payload, response, error cases) together in one chunk. - Keep UI rules + behavioral conditions together. - Keep security rules separate.

This document SHOULD be auto-chunked into the following canonical chunk types: 1. `component_overview` (one per major component) 2. `api_reference` (one per endpoint group) 3. `workflow` (scroll loading, reaction UX flow, sharing flow) 4. `security_policy` 5. `faq_debug`

### 7.3 Metadata Schema for Each Chunk

Every chunk ingested into the vector store MUST attach structured metadata for filtering and routing. Minimum required metadata keys:

```
{
  "namespace": "timeline.api" | "timeline.product" | ...,
  "component": "PostFooter" | "TimelineScroll" | "PremiumPopup" |
"LeftSidebar" | "GlobalSecurity" | null,
  "feature_area": "reactions" | "sharing" | "premium" | "infinite_scroll" |
"connections" | "onboarding" | "profile_sidebar" | "likes_popup" |
"invite_contacts" | "featured_listings" | "privacy" | "error_handling" |
"performance" | "growth",
  "endpoint": "/api/share/postlike" | "/api/share/shareupdate" | null,
  "requires_auth": true | false,
  "pii_sensitivity": "high" | "medium" | "low",
  "version": "2025-10-29",
  "page_url": "/timeline",
  "component_type": "ui" | "api" | "workflow" | "policy" | "growth_surface"
}
```

**Notes:** - `requires_auth` = whether the described feature only works for logged-in users. - `pii_sensitivity`: - `high` = talks about PII (names, profile photos, connection data, etc.) - `medium` = behavioral logic that can indirectly expose behavior of private data (e.g. rate limiting) - `low` = purely public-facing UX description. - `feature_area` is REQUIRED. It's our main filter knob for retrieval (e.g. "show me everything about reactions").

### 7.4 Retrieval Policy / Query Routing Guidance

When the assistant receives a question:

**Examples and routing choices** 1. "Why am I seeing this upgrade popup while scrolling?"
- Route → filter namespace=`timeline.ui-components` AND feature_area=`premium` - If not enough context, also search `timeline.product` with feature_area=`premium`.

1. "How do I react with a clap instead of a like?"
2. Route → `timeline.ui-components` (component=`PostFooter`, feature_area=`reactions`)

3. Also fetch `timeline.api` chunks for `/api/share/postlike` to explain backend.
4. "What API returns the feed items?"
5. Route → `timeline.api` with endpoint= `/api/share/shareupdate`.
6. "User says they clicked Connect but nothing happened" (support issue):
7. Route → `timeline.api` feature_area= `connections`, endpoint like `/api/contact/addcontact`.
8. Also grab `timeline.security-privacy` for throttling / block logic.
9. "Do other users see my phone number in the timeline?"
10. Route → `timeline.security-privacy` feature_area= `privacy`.

**Assistant behavior rules:** - The assistant MUST NOT invent endpoints not in the RAG index. - If RAG returns chunks from multiple namespaces with conflicting information, `timeline.security-privacy` has priority for anything security/privacy-related. - If a query touches payments or premium entitlements, prefer `timeline.ui-components` (PremiumPopup) + `timeline.product` over generic guesses.

### 7.5 Redaction / PII Handling in RAG

When producing answers from Timeline knowledge: - DO include generic profile fields like "first name" / "last name" conceptually. - DO mention that avatars and names can be visible in Likes Popup. - DO NOT output any real user's personal data (emails, IPs, etc.). - DO NOT output internal rate limits numerically unless they are surfaced here as product policy (we mention 50/hr as conceptual; keep that but never invent stricter/looser rules). - DO NOT promise deletions or data handling outside described flows.

### 7.6 Versioning / Freshness

- `version` metadata MUST be attached to every chunk. Current: `2025-10-29`.
- If Timeline behavior changes (new reaction types, different popup cadence, etc.), we create a new version and DO NOT delete old chunks. Retrieval can prefer the most recent `version`.

---

## 8. FAQ / Debug Notes (for Support + RAG FAQ chunks)

These should each become their own `faq_debug` chunk with `feature_area` set appropriately.

### 8.1 "I don’t see the Premium popup"

- You will NOT see it if:
- You’re not logged in.
- You’re already premium (profile `premiumStatus.planActive === true`).
- You already saw it twice this browser session (tracked in `sessionStorage`).
- Also: It only triggers after ~10s and then ~60s.

**Metadata:**


```
{
  "namespace": "timeline.ui-components",
  "component": "PremiumPopup",
  "feature_area": "premium",
  "requires_auth": true,
  "pii_sensitivity": "low",
  "version": "2025-10-29",
  "component_type": "growth_surface"
}
```


---

### 8.2 "I clicked Like but the number didn't change"

Possible reasons: 1. Network/API failure from `POST /api/share/postlike`. 2. User lost auth token / token expired → backend rejected. 3. Optimistic UI rolled back after error. 4. Post was deleted / became unavailable.

Resolution path: - Confirm you're logged in. - Retry reaction. - If repeatable, open console/network logs for `/api/share/postlike`.

**Metadata:**


```
{
  "namespace": "timeline.api",
  "component": "PostFooter",
  "feature_area": "reactions",
  "endpoint": "/api/share/postlike",
  "requires_auth": true,
  "pii_sensitivity": "medium",
  "version": "2025-10-29",
  "component_type": "workflow"
}
```


---

### 8.3 "Why can't I send a connection request to this user?"

Possible reasons: - You already sent a request (`isRequestSentToFriend === "Yes"` → button shows "Pending"). - You're already connected (`isLogUserFriend === "Yes"` → button hidden). - Backend blocked it (you can't add yourself, you might be rate limited, or blocked by that user). - Network error calling `GET /api/contact/addcontact?frienduserId={id}`.

**Metadata:**

```
{
  "namespace": "timeline.api",
  "component": "PostContentConnectionCTA",
  "feature_area": "connections",
  "endpoint": "/api/contact/addcontact",
  "requires_auth": true,
  "pii_sensitivity": "medium",
  "version": "2025-10-29",
  "component_type": "workflow"
}
```


---

### 8.4 "Why is my brand/company not showing in Featured MSMEs on the right sidebar?"

- Sidebar highlights data from `GET /api/msme/featuredmsmelisting`.
- Only some MSMEs are selected (business logic could be editorial/promo).
- The sidebar only shows a few at a time (and may randomly sample talent/freelancers).
- On `/messages` page, we may hide some sidebar widgets entirely to avoid clutter.

**Metadata:**


```
{
  "namespace": "timeline.product",
  "component": "RightSidebar",
  "feature_area": "featured_listings",
  "requires_auth": true,
  "pii_sensitivity": "low",
  "version": "2025-10-29",
  "component_type": "ui"
}
```


---

### 8.5 "What does the profile completion % in the left sidebar mean?"

- It's a progress indicator toward a 'complete' professional presence.
- It encourages you to add:
- Banner image
- Profile photo
- Professional title / headline
- Skills / experience / education
- Talent / Freelancer / MSME pages if relevant
- It's used to drive onboarding CTAs in the Registration Prompt Banner.

**Metadata:**

```
{
  "namespace": "timeline.product",
  "component": "LeftSidebar",
  "feature_area": "onboarding",
  "requires_auth": true,
  "pii_sensitivity": "low",
  "version": "2025-10-29",
  "component_type": "ui"
}
```


---

## 9. TL;DR for RAG Consumers

- The Timeline page is an authenticated social/professional feed with infinite scroll, reactions, sharing, onboarding prompts, and discovery sidebars.
- Everything here should be chunked into ~200–400 word slices with rich metadata so retrieval can target:
- `reactions`, `connections`, `premium`, `onboarding`, `featured_listings`, etc.
- Each chunk MUST include:
- `namespace`, `component`, `feature_area`, `requires_auth`, `pii_sensitivity`, `version`, and `component_type`. Add `endpoint` when relevant.
- Use namespace routing rules to answer questions accurately without hallucinating missing endpoints or behavior.
- Always respect privacy notes: don't invent exposure of private emails/phones; stick to what Timeline actually reveals (name, avatar, title, etc.).

**END v2**