# IIMB Samarpan Platform – Enterprise Wireframe Specification

## Purpose

This document translates the **Product Requirements Document (PRD)** for the IIMB Samarpan platform into a detailed wireframe specification. It provides a comprehensive blueprint for an enterprise‑grade web application using modern web technologies (e.g., React or Next.js with Supabase for the backend). The goal is to ensure that the design accommodates all functional and non‑functional requirements outlined in the PRD while adhering to best practices for enterprise user experience, security, scalability and accessibility.

## Information Architecture and Routing

The application consists of public, student‑authenticated and admin‑authenticated areas. Routes reflect the main functional modules. All pages should be responsive (desktop/tablet/mobile) and accessible.

| Section | Route | Purpose | Notes |
| --- | --- | --- | --- |
| **Home / Feed** | / | List of opportunities with search & filters for **duration**, **type** and **skill**; includes hero section and leaderboard sidebar. | Public view; allows browsing without login; CTA to register or sign in. |
| **Opportunity Detail** | /opportunity/:id | Full description (time, location, skills) with **Apply** button. | Shows status (Open/Closed) and guidelines. |
| **One‑Click Apply** | Modal or /apply/:id | Streamlined form capturing user’s profile; confirmation and success screen. | Only available to authenticated students; triggers email confirmation and logs application. |
| **Profile** | /profile | Student’s dashboard: personal data, list of applications with statuses, total coins, badges and ranking. | Tabs for Activity, Badges and Settings. |
| **Leaderboard** | /leaderboard | Leaderboard ranking students by coins; filters by timeframe (semester, all‑time). | Anonymisation option for privacy. |
| **Auth** | /login, /register | Email/password authentication; planned SSO integration (Phase 2). | Minimal, secure forms; error handling. |
| **Admin Dashboard** | /admin | At‑a‑glance metrics and quick actions for administrators. | Access controlled via role. |
| **Admin Opportunities** | /admin/opportunities | Table of all posted opportunities with create, edit, close and delete actions. | Bulk status updates and search. |
| **Create / Edit Opportunity** | /admin/opportunities/new, /admin/opportunities/:id/edit | Forms for posting opportunities: title, description, duration, type, skills, status. | Live preview of listing. |
| **Applications List** | /admin/applications/:opportunityId | View student applicants for a given opportunity; update completion status and export data. | Awards coins when marked completed. |
| **Analytics** | /admin/analytics | Charts for number of opportunities, applications and apply rate. | Visualise data over time; breakdown by type/duration. |
| **About / Contact** | /about, /contact | Optional informational pages. | Enhances credibility. |

## Design System

To ensure consistency across Replit‑hosted components and facilitate handoff to Figma, the following design system should be defined in Figma and implemented in code:

1. **Grid & Layout:**
2. Desktop: 12‑column grid, 80 px margins, 24 px gutters.
3. Tablet: 8‑column grid with 24 px margins.
4. Mobile: 4‑column grid with 16 px margins.
5. **Typography:** Inter or Poppins font families; headings bold; body 16 px; captions 14 px; maintain a clear hierarchy.
6. **Color Palette:**
7. Primary: blue (#1E3A8A) for actions and links.
8. Secondary: cyan (#0EA5E9) for highlights.
9. Neutral: white for backgrounds, dark grey (#0F172A) for text.
10. Success / Warning / Error tokens for statuses and notifications.
11. **Components:** Buttons (primary/secondary/tertiary), inputs (text, select, multi‑select), filter chips, cards, tables, tabs, badges, progress bars, modal dialogs, toasts, skeleton loaders, tooltips.
12. **Accessibility:** High color contrast, keyboard navigation, alt text for all images, ARIA labels for interactive elements. Align with WCAG 2.1 AA guidelines.
13. **Responsive Behaviour:** Use auto‑layout and flexbox; components stack vertically on mobile; filters become collapsible drawers.

## Global Layout (Header & Footer)

All pages share a consistent header and footer.

### Header

* **Logo + Title** linking to /.
* **Navigation**: links to Opportunities, Leaderboard and About/Contact. For students: show a coin counter and badge icons; for admins: show link to Admin Dashboard.
* **Auth state:** Sign in / Sign up buttons when logged out; avatar menu with Profile, Admin (if applicable) and Logout when logged in.
* **Global Search**: a search box in the header that filters the opportunities list.

### Footer

* Links to About, Contact, Privacy, Terms.
* Copyright and Vikasana credit.

## Wireframes by Page

The following sections describe each key page’s layout, components and interactions. Figma designers should create low‑fidelity frames for desktop, tablet and mobile using these descriptions, then progressively enhance fidelity.

### 1. Home / Opportunities Feed (/)

**Purpose:** Discover and filter opportunities.

**Main Areas:** - **Hero Section**: Title such as “Find ways to contribute”, a short description referencing the mission and search bar. Provide quick access to the leaderboard. - **Filters Sidebar** (left on desktop; collapsible drawer on mobile): - Duration (Instant, 1–3 days, 1 week, 2–4 weeks) with radio or chip group. - Type (Teaching, Donation, Mentoring, etc.). - Skills (multi‑select chips). - Clear filters button. - **Opportunity Cards Grid**: Each card summarises title, 2‑line description, duration, type and required skills. Include a CTA (“View”) and status badge (Open/Closed). Cards should load progressively; skeleton loaders display while fetching. - **Sidebar Right (optional)**: Top contributors mini‑leaderboard; new badges unlocked (highlighting gamification). On mobile this can move below the cards.

**Interactions:** - Filters and search update results immediately (debounced). Combined filters use AND logic. - Sorting options (Newest, Ending soon, Most applied) via a dropdown. - Pagination or infinite scroll loads additional opportunities. - Empty state when no results match; show suggestions (e.g., “Try clearing filters”).

**Acceptance criteria:** At least 10 cards per page; filters update results; search by keyword.

### 2. Opportunity Detail (/opportunity/:id)

**Purpose:** Provide all information about an opportunity and allow application.

**Main Areas:** - **Header Section**: Title; status badge; share button; favourite/save for later (optional).  
- **Details Pane**: - Description (rich text) explaining the cause and tasks. - Duration and time commitment (e.g., 2 hours/week for 4 weeks).  
- Location (campus or remote) and schedule.  
- Required skills and type (displayed as chips).  
- Contact information (email/phone).  
- FAQs (optional). - **Apply Card (sticky on desktop)**: Contains an **Apply** button (enabled when status = Open), eligibility criteria and a note about confirmation email. On mobile, place this card below the details.

**Interactions:** - Clicking **Apply** triggers the one‑click apply modal. If already applied, display “Applied on [date]; view in Profile”. - If status is Closed, disable the button and label “Closed”.

### 3. One‑Click Apply (Modal)

**Purpose:** Reduce friction for students to sign up.

**Layout:** - Title: “Apply to [Opportunity]”. - Prefilled user info (Name, Email; read‑only).  
- Checkbox confirming availability (“I can commit to the time/duration”).  
- CTA: **Confirm & Apply**. Secondary CTA: **Cancel**.  
- On success: show success message with options to view in Profile or go back to feed.

**States:** Loading state (spinner on CTA); success; error (display error banner). The modal should be accessible and can also be implemented as an inline form on small screens.

### 4. Profile (/profile)

**Purpose:** Show student’s participation history and gamification progress.

**Layout:** - **Header**: Avatar, student name and program/branch.  
- **Stats Strip**: total coins, completed opportunities, current leaderboard rank (e.g., “Rank #7 this semester”). Visualised with progress bars for coin milestones (Bronze, Silver, Gold).  
- **Tabs**: - **Activity**: table or list of all applications. Columns: Opportunity title, Status (Pending, Accepted, Completed), Applied on date, Completed on date (if self‑reported). Actions: view details; mark complete (if allowed).  
- **Badges**: grid of unlocked and locked badges. Each badge shows icon, name and description; locked badges appear greyed with threshold.  
- **Settings**: toggles for email/SMS notifications, privacy options for appearing on the leaderboard, and profile details.  
- **Gamification**: dynamic coin bar and badges highlight.

### 5. Leaderboard (/leaderboard)

**Purpose:** Foster friendly competition by displaying rankings based on coins earned.

**Layout:** - Filters at top: timeframe (This month, This semester, All time); optional cohort filter (batch or program).  
- Table: Rank, Student name (or anonymised handle if privacy opted out), Coins, Badges count, Completed opportunities. - Highlight the current student’s position in the leaderboard with a sticky banner (“You are #7”). - Provide toggle for anonymisation (e.g., display “Student#1234”).

### 6. Auth (Login / Register)

**Purpose:** Secure entry point for both students and admins.

**Components:** - Email and password fields with validation and error messages. - CTA buttons: “Login” and link to “Create account”. Use reCAPTCHA if necessary. - Provide SSO (IIMB login) placeholder for Phase 2. - Link to “Forgot password”.

### 7. Admin Dashboard (/admin)

**Purpose:** Provide administrators with overview metrics and quick links.

**Sections:** - **KPI Cards**: display number of active opportunities, total applications (7‑day/30‑day), apply rate.  
- **Quick Actions**: “Create Opportunity”, “View Applications”, “Analytics”, “Manage Badges” (for future enhancements).  
- **Recent Activity**: list of latest applications (name, opportunity, date). Clicking leads to application detail page.

### 8. Admin Opportunities (/admin/opportunities)

**Purpose:** Manage all posted opportunities.

**Layout:** - Table with columns: Title, Status, Type, Applicants (#), Created/Updated dates, Actions (Edit, Close, Delete).  
- Filters/search bar by title, status, type and duration.  
- Bulk actions: select multiple opportunities to close or delete.  
- Button “New Opportunity” linking to the creation form.

### 9. Create / Edit Opportunity

**Purpose:** Allow admins to post or modify opportunities with all necessary details.

**Form Fields:** - **Title** (required) - **Short Description** (max 160 characters) - **Full Description** (rich text or Markdown) - **Duration**: either select preset durations (Instant, 1–3 days, 1 week, 2–4 weeks) or specify custom hours/weeks. - **Type**: radio or multi‑select (Teaching, Donation, Mentoring, etc.) - **Skills**: multi‑select chip list.  
- **Capacity** (optional maximum number of participants). - **Status**: Open/Closed.  
- **Location & Schedule** (optional).  
- **Visibility**: Public or private (link only).  
- **Preview**: show how the card and detail page will look.

**Validation:** Required fields; future dates if schedule specified; descriptive length.

**Actions:** Save draft, Publish. Cancel returns to opportunities list.

### 10. Applications List

**Purpose:** Show administrators who applied for a particular opportunity.

**Layout:** - Table: Applicant name, Email, Applied on date, Status (Pending/Completed), Notes field.  
- Filters: by status.  
- Actions: mark completion (awards coins), export to CSV, contact via mailto link.

### 11. Analytics

**Purpose:** Provide administrators with metrics on usage and engagement.

**Content:** - KPI tiles summarising counts: opportunities posted, total applications, average apply rate.  
- Line chart showing applications over time (daily/weekly).  
- Pie or bar charts breaking down applications by type and duration.

**Interactivity:** Date range selector; export option.

### 12. Email Templates

The application should generate email notifications as plain‑text and HTML templates:

* **Student Sign‑Up Confirmation:** includes opportunity summary, schedule and instructions.
* **Admin Daily Digest:** lists new applications by opportunity.

## Gamification Design

To align incentives and encourage sustained participation, the gamification layer awards digital coins and badges:

1. **Coin Earning:** Students earn coins when administrators mark an opportunity as “Completed” in the Applications list. Default rule: 1 coin per hour of commitment; configurable later. Ensure anti‑gaming measures (e.g., prevent multiple completions on the same opportunity).
2. **Badge Tiers:** Bronze (10 coins), Silver (25 coins), Gold (50 coins), plus special badges such as “First Apply”, “First Completion” and semester streaks. Badges display in profile; locked badges appear greyed with a tooltip describing the requirement.
3. **Leaderboard:** Ranking is determined by total coins; tie‑breakers include number of completions and earliest completion time. Provide filters by timeframe. Students may anonymise their display on the leaderboard for privacy.
4. **Gamification UI:** Show coin total in the header; display badges next to the coin tally; show milestone progress bar; highlight new badges via toast notifications or modal overlays. Provide a dedicated Leaderboard page with filters and personal ranking.

## Non‑Functional & Enterprise Considerations

The wireframe must support the non‑functional requirements specified in the PRD:

* **Performance:** Pages should load under 2 seconds for at least 200 concurrent users. Use code-splitting and caching; keep initial payload lightweight; prefetch API data.
* **Scalability:** Use a modular architecture with a decoupled frontend (React/Next.js) and backend (Node.js or Supabase). Design database schemas to handle growth in users, opportunities and applications. Implement pagination for all list endpoints.
* **Security:** Enforce HTTPS; implement role‑based access control; sanitise all inputs; store sensitive data (passwords) securely; integrate with Replit secret management.
* **Accessibility:** Meet WCAG 2.1 AA guidelines: proper semantics, focus management, keyboard navigation, alt text and ARIA labels.
* **Maintainability:** Use component‑based architecture, TypeScript and linting; document API contracts; set up continuous integration; include unit and integration tests.

## Implementation Notes for Replit

* **Tech Stack:** Use React or Next.js for the frontend; deploy to Replit. Use Supabase or PostgreSQL for the backend (Supabase provides authentication, database and storage). Node.js with Express can serve API endpoints.
* **Directory Structure:**

/src  
 /components # Reusable UI components (cards, tables, modals)  
 /pages # Route‑level components for Next.js  
 /hooks # Custom hooks (fetch, auth, gamification)  
 /styles # TailwindCSS or CSS‑in‑JS definitions  
 /utils # Helper functions (date formatting, validation)  
/server  
 /api # REST endpoints (opportunities, applications, analytics, auth)  
 index.js  
/public # Static assets

* **State Management:** Use React Context or Redux Toolkit for user state (auth, coins, applications). Use SWR or React Query for data fetching and caching.
* **Deployment:** Use Replit’s hosting with a custom domain (via GoDaddy). Set up HTTPS using Let’s Encrypt. Configure environment variables for API keys and database connections.
* **Future Enhancements:** Plan for Phase 2 and Phase 3 features (SSO integration, push notifications, transcripts, external NGO access) by designing extensible APIs and reserving UI placeholders.