World of Peace Platform

give me a promp i want to build a platform like quora reddit naming world of peace with all necessary things it has include all give me a proffessional high level prompt covering all

Nice 🔥 — this is exactly the kind of thing you should **not describe vaguely** to an AI builder/agent. You want a *founder-level product brief*, not “make a reddit clone”.

Below is a **professional, high-level master prompt** you can directly give to an AI developer (Lovable / Cursor / Devin / ChatGPT Agent / any code-gen AI) to design and build your platform.

📄 MASTER BUILD PROMPT — *World of Peace*

Prompt Title:
Build a Full-Scale Social Knowledge & Community Platform (Quora + Reddit Hybrid) called “World of Peace”

---

1. Product Vision

I want you to design and build a production-ready web platform named **World of Peace** — a large-scale social community and knowledge sharing platform inspired by Reddit and Quora, but focused on **positive discussions, meaningful conversations, learning, and healthy communities instead of toxicity**.

The platform should allow users to:

- Ask questions
- Share experiences
- Create communities
- Discuss topics
- Help others
- Build reputation
- Follow interests
- Discover knowledge

The goal is to create a **knowledge-first + community-first ecosystem**, not a meme-only social network.

---

2. Tech Stack (MANDATORY)

Build it as a modern scalable SaaS product using:

Frontend

- Next.js (App Router)

- React
- TypeScript
- TailwindCSS
- Zustand or Redux Toolkit
- TanStack Query (React Query)
- Responsive mobile-first UI

Backend

- Node.js
- Express.js or NestJS
- REST APIs (plus optional GraphQL layer)
- JWT Authentication
- Role Based Access Control (RBAC)

Database

- MongoDB (primary DB)
- Redis (caching + sessions + rate limiting)

Storage

- AWS S3 (images, avatars, post media)

Realtime

- WebSockets / Socket.io (notifications, live comments, chats)

Deployment

- Dockerized
- Nginx reverse proxy
- Deployable on AWS EC2
- CI/CD ready

---

3. Core Platform Modules

A. Authentication & Users

Implement a full auth system:

- Email/password signup
- Google OAuth login
- Email verification
- Forgot/reset password

- Session management
- Device tracking
- Profile creation onboarding

User profile should include:

- Avatar
- Bio
- Skills/interests
- Location (optional)
- Links (GitHub, LinkedIn, website)
- Followers/Following
- Reputation score
- Badges

B. Communities (like Subreddits)

Users can create and manage communities.

Community features:

- Name, description, banner, rules
- Public / Private / Restricted communities
- Moderators & admins
- Join / Leave
- Community categories (tech, health, relationships, education, finance, etc.)
- Community settings panel
- Community feed

Moderation:

- Approve/reject posts
- Ban users
- Mute users
- Remove comments
- Pin posts

C. Posts System

Users can create:

1. Question posts (Quora style)

2. Discussion posts
3. Story posts
4. Image posts
5. Polls

Post features:

- Rich text editor
- Markdown support
- Tags
- Media upload
- Post drafts
- Edit history
- Post analytics (views)

Interactions:

- Upvote / Downvote
- Save
- Share
- Report
- Bookmark

Ranking algorithm:

- Hot
- New
- Top
- Trending

D. Answers & Comments

- Nested comments (infinite depth)
- Replies
- Mention users (@username)
- Like/Dislike
- Sort comments (top/new/controversial)
- Best answer selection (for questions)
- Comment editing & deletion
- Anti-spam protection

E. Reputation & Gamification

Create a reputation system:

Users earn points when:

- Their answer is upvoted
- Their post trends
- Their comment is liked
- They help others

Badges:

- Helpful contributor
- Top writer
- Mentor
- Community leader
- Verified expert

Leaderboards:

- Global
- Community-specific

F. Follow System

Users can:

- Follow users
- Follow topics
- Follow communities

Feed personalization based on:

- interests
- activity
- follows

G. Notifications (Realtime)

Implement a full notification system:

Notify when:

- someone replies
- someone follows

- comment on post
- mention
- badge earned
- moderator action

Realtime via websockets + stored notifications.

H. Messaging (Direct Chat)

Private messaging system:

- 1-to-1 chat
- typing indicator
- read receipts
- media sharing

I. Search System

Powerful global search:

Search:

- Users
- Posts
- Questions
- Communities
- Tags

Include filters:

- newest
- most upvoted
- unanswered

(Use Elasticsearch or Mongo text indexing)

J. Moderation & Safety (VERY IMPORTANT)

Implement strong anti-toxicity controls:

- Report post/user/comment
- Shadow ban
- Content moderation queue

- Keyword filter
- Rate limiting
- Spam detection
- Account suspension
- Community rule enforcement

Admin panel should manage:

- reports
- bans
- flagged content

K. Admin Dashboard

Super Admin panel:

- View all users
- Ban/verify users
- Remove posts
- Manage communities
- View analytics
- Platform metrics (DAU, MAU, posts/day)

L. AI Features (Unique Feature of World of Peace)

Add AI assistance:

- AI summarize long discussions
- AI suggest answers
- AI toxicity detection
- AI recommend communities
- AI auto-tag posts

---

4. Feed & Recommendation Engine

Implement a smart home feed:

Personalized ranking based on:

- user interests
- previous interactions
- community membership

- trending posts

---

5. UI/UX Requirements

Design must feel:

- Calm
- Minimal
- Clean
- Positive (not chaotic like reddit)

Include:

- Light & Dark mode
- Accessibility
- Mobile responsiveness
- SEO optimized pages

---

6. Security

Must include:

- Rate limiting
- CSRF protection
- XSS protection
- Input sanitization
- Secure cookies
- Password hashing (bcrypt)
- Helmet middleware

---

7. Performance & Scalability

- Redis caching
- Lazy loading comments
- Pagination
- Image optimization
- CDN ready
- Horizontal scaling ready

---

8. Deliverables

The output should include:

- Database schema design
- API architecture
- Folder structure
- Frontend pages
- Backend endpoints
- Deployment instructions
- Environment variables template
- Docker setup

---

9. Important Constraints

This is not a simple clone —
It must be **production-grade, scalable, and modular** so mobile apps (React Native) can be built later using the same APIs.

You can copy-paste this directly into an AI coding tool — this is essentially a **startup-level PRD (Product Requirement Document)**.
If you want, I'll also design the **MongoDB schema (users, posts, comments, votes, communities)** next — that will save you *days* of mistakes later.

give me frontend only

Perfect 👍 — so now we'll make a **Frontend-Only Product Requirement Prompt** (this is what you give to an AI like Cursor / Lovable / v0 / ChatGPT Agent to generate the UI of your platform).

This version assumes **NO backend exists yet**.
Everything should run using **mock data + API service layer** so later you can connect your Node.js backend without rewriting UI.

📄 FRONTEND-ONLY MASTER PROMPT

Build "World of Peace" (Quora + Reddit Hybrid) — UI/UX Only

---

1. Objective

Create a complete frontend web application named **World of Peace** — a community knowledge-sharing and discussion platform inspired by Quora and Reddit.

This task is **frontend only**.
Do NOT implement backend logic.
Instead, design the UI and architecture as if a real backend exists, using mock APIs and service abstractions.

The UI must be production-quality and ready to connect to real APIs later.

---

2. Tech Stack (STRICTLY FOLLOW)

Use:

- Next.js 14+ (App Router)
- React
- TypeScript
- TailwindCSS
- ShadCN UI components
- Zustand (global state)
- TanStack Query (React Query) for API handling
- Axios (API layer)
- React Hook Form + Zod validation
- Next Themes (dark/light mode)

The app must be responsive (mobile + tablet + desktop).

---

3. Application Architecture (VERY IMPORTANT)

Create a clean scalable architecture:

```
src/
 ├── app/
 ├── components/
 ├── features/
 ├── hooks/
 ├── services/
 ├── store/
 ├── types/
 ├── utils/
 ├── mock/
 └── providers/
```

Requirements:

- All API calls must go through `services/api.ts`
- Create fake endpoints using mock JSON files
- No component should directly use hardcoded data
- UI must work with loading, error, and empty states

---

4. Pages to Implement

Public Pages

- Landing Page
- Explore Communities
- Explore Questions
- Single Post View (public)
- User Profile (public)
- Search Page

Authentication

- Login
- Signup
- Forgot Password
- Email Verification UI
- Onboarding (select interests & topics)

---

5. Main Authenticated App (After Login)

Create a layout similar to a modern community platform.

Main Layout

Include:

- Left Sidebar
- Main Feed
- Right Sidebar (suggestions & trending)
- Top Navbar

Navbar

Must include:

- Logo (World of Peace)
- Search bar (global search)
- Create Post button
- Notifications bell
- Messages icon
- User avatar dropdown
    - Profile

  - Saved
  - Settings
  - Logout

Left Sidebar

Include navigation:

- Home Feed
- Popular
- Communities
- Questions
- Following
- Saved Posts
- Messages
- Notifications
- Settings

Also show joined communities list.

Home Feed

Create an intelligent feed UI that displays:

Each post card shows:

- Author avatar + name
- Community name
- Time posted
- Post title
- Post content preview
- Images (if available)
- Tags
- Upvotes / Downvotes
- Comment count
- Share
- Save

Include filters:

- Hot

- New
- Trending
- Following

Create Post Modal/Page

User can create:

- Question
- Discussion
- Story
- Poll
- Image post

Features:

- Rich text editor
- Markdown support
- Image upload preview
- Tag selector
- Community selector
- Save Draft button
- Post button

Post Details Page

When user opens a post:

Display:

- Full post content
- Media
- Vote buttons
- Save
- Share
- Report

Below it

give me the prompt to match exactly like quoara reddit

Got it 👌
You don’t want a “similar concept” anymore — you want the **UI behavior, structure, and UX patterns to feel extremely close to Quora + Reddit** so users instantly understand the platform.

Below is a **strict UI replication-style frontend prompt** you can paste directly into an AI builder (Cursor, v0, Lovable, Replit AI, ChatGPT Agent, etc.).

📄 FRONTEND PROMPT — “World of Peace”

(Exact UX Behavior Inspired by Quora + Reddit)

Project Goal

Build a **frontend-only web application** named **World of Peace** that combines:

- Quora’s clean knowledge-sharing UI
- Reddit’s community discussion and voting system

The platform should feel instantly familiar to users of Quora and Reddit in navigation, layout, interactions, and behavior — while keeping a clean, modern, and positive aesthetic.

Do NOT build backend logic.
Use mock APIs and service abstraction.
The UI must be fully ready to connect to real REST APIs later.

Tech Stack (Mandatory)

Use exactly:

- Next.js 14+ (App Router)
- React + TypeScript
- TailwindCSS
- ShadCN UI
- Zustand (state management)
- TanStack Query (data fetching)
- Axios (API layer)
- React Hook Form + Zod
- Next Themes (dark/light)
- React Markdown (for posts & answers)

Mobile responsive is required.

Core UX Principle

The UI must replicate:

Quora style

- Question & Answer focus
- Writer reputation
- Clean reading layout
- Rich answer editor
- Profile-centric contributions

Reddit style

- Communities (subreddits equivalent)
- Upvote / Downvote voting
- Comment threads
- Feed scrolling
- Post cards
- Moderation feel

Application Layout

Global Layout

Create a 3-column layout (Reddit style):

Left Sidebar — Navigation & Communities
Center — Feed / Questions / Posts
Right Sidebar — Suggestions / Trending / Who to follow

Also include a fixed top navigation bar.

Top Navigation Bar

Must include:

- Platform logo “World of Peace”
- Global search bar (like Reddit search)
- Create Post button
- Notifications bell (dropdown panel)
- Messages icon
- User avatar dropdown:
    - Profile
    - Answers

    - Saved
    - Settings
    - Logout

Sticky navbar on scroll.

Left Sidebar (Reddit-like)

Include:

Main Navigation

- Home
- Popular
- Questions
- Communities
- Following
- Saved
- Notifications
- Messages

My Communities Section
List joined communities with icons.

Create Community Button

Collapsible on mobile.

Home Feed (Reddit Behavior)

Create an infinite scroll feed.

Each post card must show:

- Upvote & downvote vertical buttons (left side)
- Author avatar
- Username
- Community name
- Posted time
- Post title
- Text preview (collapsed)
- Images (if any)
- Tags
- Comment count

- Share button
- Save button

Sorting filters:

- Hot
- New
- Top
- Trending

Question Feed (Quora Behavior)

A dedicated Questions page that shows:

Each question card:

- Question title
- Number of answers
- Last activity time
- Ask Answer button
- Follow Question button

Clicking opens a **reading-focused page** like Quora.

Post Detail Page (Hybrid View)

Top:

- Full post or question
- Voting system
- Follow button
- Share
- Save

Below:

- Answers (Quora style)
- Comments (Reddit style nested threads)

Answer System (Very Important)

Implement Quora-like answer experience:

Each answer shows:

- Writer profile card
- Bio/headline
- Credentials (mocked)
- Answer content (rich formatted)
- Upvote
- Downvote
- Comment
- Share
- “Mark as helpful”

Best answer pinned at top.

Rich Text Editor

Implement an editor similar to Quora:

Features:

- Bold / Italic / Headings
- Bullet & numbered list
- Code block
- Quote block
- Links
- Image upload preview
- Draft saving (local storage)

Comments (Reddit-style Threads)

- Infinite nested replies
- Collapse/expand threads
- Vote on comments
- Reply to specific comment
- Sort comments: top/new/controversial

Communities (Subreddit Equivalent)

Community page must contain:

Header:

- Banner
- Icon

- Community name
- Description
- Join/Leave button
- Member count

Tabs:

- Posts
- Questions
- Media
- About

Moderation UI (mocked):

- Remove post
- Lock comments
- Pin post

User Profile (Quora-like)

Tabs:

- Answers
- Questions
- Posts
- Followers
- Following

Profile includes:

- Avatar
- Bio
- Credentials
- Topics
- Stats:
  - Answers written
  - Upvotes received
  - Reputation score

Notifications Panel

Dropdown notifications (like Reddit):

Types:

- Someone answered your question
- Someone replied
- Someone followed you
- Your post upvoted

Unread indicator required.

Messaging UI

Reddit-style chat interface:

- Conversation list (left)
- Chat window (right)
- Typing indicator (mock)
- Seen status

Search System

Global search page with filters:

Tabs:

- Posts
- Questions
- Communities
- Users

Show realistic results using mock data.

Create Post / Ask Question

Modal with tabs:

- Ask Question
- Create Post
- Share Story
- Poll

Fields:

- Title
- Body (rich editor)

- Tags
- Community selection

State & Data Layer Requirements

You MUST:

- Create `services/api.ts`
- Use Axios instance
- Simulate API delay
- Show loading skeletons
- Handle error states
- Use mock JSON data

NO HARDCODED DATA inside components.

UI/UX Quality Requirements

Design must feel:

- Clean like Quora
- Interactive like Reddit
- Calm colors
- Not meme-style

Include:

- Dark mode
- Skeleton loaders
- Empty states
- Toast notifications

Final Deliverables

Provide:

1. Full page routing
2. Reusable component library
3. Mock API service layer
4. Responsive UI
5. Production-ready folder structure

You can paste this directly — this is basically telling the AI:

> “Recreate Quora + Reddit UX patterns, but with a new brand.”

If you want, next I can also give you **the exact color palette + typography system** so it doesn’t look like a cheap clone (this is where most clones fail).

If you want, next I can also give you the exact color palette + typography system so it doesn’t look like a cheap clone (this is where most clones fail).
yes

Perfect 😄
This part is **actually more important than code**.

Most “reddit/quora clones” fail because they copy the layout but **not the design language**.
Good products are recognizable by **color psychology + typography hierarchy**, not components.

Below is a ready-to-use **design system** for **World of Peace**.
You can directly give this to an AI builder or your frontend so the site looks like a real startup — not a college project.

🎨 World of Peace — Design System (Color + Typography + UI Rules)

---

1. Brand Identity (Very Important)

Platform Theme: Calm, intelligent, thoughtful discussion
(Not aggressive like Reddit, not corporate like LinkedIn)

The feeling should be:

- Safe
- Educational
- Mature
- Positive
- Trustworthy

So avoid:
❌ neon colors
❌ harsh reds
❌ meme-style UI
❌ gaming style gradients

We want **“healthy internet” vibes**.

---

2. Color Palette

Primary Brand Color (Peace Blue)

Use this everywhere (logo, buttons, highlights)

```
Primary: #2563EB
Primary Hover: #1D4ED8
Primary Light: #DBEAFE
Primary Soft BG: #EFF6FF
```

Why?
Blue = trust + intelligence (this is why knowledge platforms use blue).

Secondary Accent (Growth Green)

Used for helpful actions, accepted answers, positive actions.

```
Secondary: #22C55E
Secondary Hover: #16A34A
Soft Green BG: #DCFCE7
```

Used for:

- Helpful answer
- Best answer
- Reputation gained
- Success toast

Warning / Attention (Minimal)

```
Amber: #F59E0B
Soft Amber BG: #FEF3C7
```

Used for:

- Reports
- Pending moderation
- Draft not saved

Error (Keep soft — no harsh red)

```
Error: #EF4444
Soft Error BG: #FEE2E2
```

Neutrals (MOST IMPORTANT — 70% of UI)

Light Mode

```
Background: #F8FAFC
Card: #FFFFFF
Border: #E2E8F0
Text Primary: #0F172A
Text Secondary: #475569
Muted Text: #94A3B8
Hover: #F1F5F9
```

Dark Mode

```
Background: #0B1120
Card: #111827
Border: #1F2937
Text Primary: #E5E7EB
Text Secondary: #9CA3AF
Hover: #1F2937
```

---

3. Typography System (CRITICAL)

Your typography will decide whether users feel:
“real platform” or **“cheap clone”**

Font Choice (Use exactly)

Primary Font: `Inter`
(Do NOT use Poppins everywhere — that’s why most clones look fake)

Why Inter?

- Used by modern SaaS platforms
- Excellent readability for long answers
- Perfect for knowledge content

Add in Next.js:

```
next/font/google → Inter
weights: 400, 500, 600, 700
```

Text Hierarchy

Usage	Size	Weight
Page Title	28px	700
Post Title	22px	600
Question Title	24px	600
Section Title	18px	600
Body Text	15px	400
Comment Text	14px	400
Meta Info	13px	500
Small labels	12px	500

IMPORTANT:

- Line height: `1.6` for reading content
- Paragraph spacing: `16px`
- Max reading width: `680px` (this is why Quora feels readable)

---

4. Layout Rules (Most Developers Miss This)

Reading Layout

The **post/answer page MUST be centered and narrow**.

```
Max content width: 680px
```

If you make it full width → users won't read long answers.

Card Design

All posts should be inside cards:

```
border-radius: 12px
border: 1px solid #E2E8F0
padding: 16px
background: white
```

Hover:

```
box-shadow: 0 2px 8px rgba(0,0,0,0.05)
```

No heavy shadows.

Spacing System (Follow Strictly)

Use 8px spacing scale:

```
4px
8px
12px
16px
20px
24px
32px
40px
48px
64px
```

Never random spacing.

---

5. Buttons

Primary Button (Create Post)

Blue filled.

Secondary Button

Outline.

Ghost Button

Used for vote, save, share.

Vote buttons should look like Reddit but softer:

- Upvote → turns blue
- Downvote → muted gray (not red)

---

6. Voting Design

Important:
Do NOT use bright red/green arrows.

Use:

- Active Upvote → Blue (#2563EB)
- Neutral → Gray
- Downvote → Dark gray

This keeps platform non-toxic psychologically.

---

7. Avatars & Icons

- Rounded (not square)
- Use `lucide-react` icons
- Avatar size: 32px (comments), 40px (posts), 64px (profile)

---

8. Feed Behavior (Key UX Detail)

Post cards must:

- Highlight slightly on hover
- Be clickable
- Show preview (3 lines collapsed text)
- Lazy load images

---

9. Micro-Interactions (What makes it feel real)

Add:

- Skeleton loaders

- Button press animation (scale 0.97)
- Upvote animation
- Save bookmark animation
- Toast notifications
- Infinite scroll loading spinner

---

10. Dark Mode Rules

Dark mode should NOT invert colors blindly.

Keep:

- Blue stays blue
- Cards darker, not black
- Text slightly gray (#E5E7EB)

Final Instruction (Give this to the AI)

> Follow this design system strictly. Do not invent random colors, fonts, or spacing. Every component must respect the color palette, typography scale, and layout rules to maintain a consistent brand identity for the World of Peace platform.

If you want next, I'll give you something extremely useful:
the exact folder structure + component list (Navbar, PostCard, VotePanel, CommentTree, AnswerEditor, etc.) — this alone will save you 2–3 weeks of trial-and-error.

now create a 2nd prompt so it matches with my existing means i will give the 2nd instruction to add a token model credit system per post a credit will be n3eeded and when user post ai will summarize the content with proper animation loader ai summarization for the post and create a trending poupu;lar section in the home page also show also credit can be buyed by the user also like this

Perfect — this is exactly how real products are built 👌
You already gave the AI the **base platform prompt**.
Now this is your **SECOND INSTRUCTION PROMPT (Feature Expansion Patch)** — you give this after the first one so the AI *extends the existing app*, not rebuilds it.

This prompt tells the AI:

> "Modify the existing World of Peace frontend and add an AI + Credits economy system."

You can paste this directly.

📄 SECONDARY FEATURE PROMPT

Add AI Credit Economy + AI Summarization to “World of Peace”

Instruction Type

This is NOT a new project.

You must **extend the already created World of Peace frontend application** and integrate a **token/credit based AI assistance system** into the existing UI and architecture.

Do not change the current layout, routing, or design system.
All new features must follow the previously defined color palette, typography, and component structure.

---

1. Credit (Token) Economy System

Implement a **credit-based usage model**.

Each user owns a wallet balance called:

> **Peace Credits**

Rules

- Every time a user creates a post → 1 credit is consumed
- If the user has 0 credits → they cannot post
- Show an error state + purchase prompt

Credits are required for:

- Creating a post
- AI summarization

UI Requirements

Credit Display

Add a credit indicator in the Navbar (top right near avatar)

Example:
🟦 `12 Credits`

Behavior:

- Hover shows tooltip: “Credits are used to create posts and AI features”
- Clicking opens wallet page

Wallet / Credits Page

Create a page:

`/credits`

Show:

- Current credit balance
- Usage history (mock data)
- Purchased history (mock)
- How credits work explanation

Buy Credits UI

Create credit purchase cards:

Package	Credits
Starter	10
Plus	50
Pro	150

Each card shows:

- Price (mock)
- Buy button
- Highlight recommended plan

(Do NOT implement real payment — simulate success purchase and update state.)

After purchase:

- Show success modal
- Animate credit counter increasing

---

2. Posting Restriction Logic

When user clicks **Create Post**:

If credits = 0:

- Block post creation
- Show modal:

“You need credits to post”

Buttons:

- Buy Credits
- Cancel

If credits > 0:

- Allow posting
- Deduct 1 credit
- Animate deduction

---

3. AI Summarization Feature

After a user submits a post:

Simulate an AI processing flow.

Flow

1. User clicks Post
2. Post enters “Processing by AI…” state
3. Show animated AI loader
4. After delay → generate summary (mocked text)

AI Loader UI (Important)

Create a beautiful animated loader:

Display message:
“AI is understanding your post…”

Animation ideas:

- Pulsing orb
- Typing dots
- Skeleton shimmering
- Progress bar

Duration: 3–6 seconds (simulated)

AI Summary Block

After processing, attach a summary to the post:

Add a section at top of post:

> 🤖 AI Summary

Inside a soft blue box:

- 3–5 line concise explanation
- Collapsible

- Expand “Read Full Post”

Include:

- Copy summary button
- Regenerate summary button (costs 1 credit)

If regenerate clicked → deduct 1 credit and rerun loader.

---

4. Trending & Popular Section

Modify Home Page.

Add a **Right Sidebar Widget**:

Trending Now

Show:

- 5 trending posts
- Fire icon 🔥
- View count
- Upvotes

Popular Communities

Show:

- Top 5 communities
- Member count
- Join button

Active Topics

Display tags:
`#MentalHealth`
`#CareerAdvice`
`#Programming`
`#Relationships`
`#SelfImprovement`

Use mock algorithm:
Trending = highest upvotes + recent.

---

5. Credit Usage Feedback

Add clear feedback:

When credits used:

- Floating toast
  “1 credit used”

When insufficient:

- Warning toast

When purchase success:

- Celebration animation (confetti optional)

---

6. State Management

Create a global credit store:

`store/creditStore.ts`

Must contain:

- balance
- spendCredits()
- addCredits()
- transactionHistory

Persist using localStorage.

---

7. UI Rules

Follow the design system:

- Use primary blue for AI
- Use soft blue background for AI summary
- Do NOT add neon or purple AI themes
- Keep calm and educational tone

---

8. Edge Cases

Handle:

- Refresh page (credits persist)
- Trying to post without credits
- Regenerate summary without credits
- Multiple rapid clicks

- Loading states

---

9. Final Deliverables

Update the existing project to include:

- Credits wallet UI
- Purchase UI
- AI summarization flow
- AI animated loader
- Credit deduction logic
- Trending sidebar
- Toast feedback system

The app must feel like a real SaaS product with an AI-powered knowledge platform and a usage-based economy.

You now have something very close to a **Monetizable Reddit + Quora + AI platform**.
If you want, next I can also design the **pricing strategy (how many credits per action so users don't feel cheated but you still profit)** — that part actually decides whether this succeeds or dies.

now what more things i can add so to make it more livable real life real world application frontend part

Good question — right now you have a **feature-complete clone**.
But a *real product* (something people actually keep using daily) is not built by more big features...
It's built by **small real-life behaviors**.

Quora and Reddit didn't grow because of posts — they grew because the app feels *alive even when you are not posting*.

Below are the **next frontend features you should add** (these are exactly the things that separate a portfolio project from a startup-level product).

What to Add Next (Frontend Only) to Make "World of Peace" Feel Real

---

1️⃣ Daily Engagement System (VERY IMPORTANT)

Right now users will:

login → browse → leave → never come back

You need a **reason to return daily**.

Add: Daily Streaks

Create a streak tracker:

User earns streak when they:

- answer a question
- write a helpful comment
- log in & read posts (3+ minutes)

UI additions:

- streak flame icon near avatar
- profile shows “7 Day Helping Streak”
- breaking streak shows warning modal

Why this matters:
This alone can **double retention**.

2️⃣ Drafts & Continue Writing

Real users never write a post in one sitting.

Add:

- Auto save draft (every 5 seconds)
- “Continue writing?” popup when user returns
- Drafts page `/drafts`

Show:

- last edited time
- preview
- resume editing

(Quora has this — and it massively increases posting.)

3️⃣ Read Later & Knowledge Library

Right now “Saved posts” is weak.

Turn it into a **personal knowledge library**.

Features:

- Save post into collections
- Create folders:
    - Career
    - Mental Health
    - Programming
    - Finance

UI:
`Save → Choose Collection`

Now your app becomes a **learning tool**, not just a social network.

4️⃣ Follow Topics (Not Only Users)

This is one of Quora's strongest mechanics.

Add topic system:

Users follow:

- Anxiety
- JavaScript
- Relationships
- Career Advice

Then home feed changes.

Add:

- topic chips under posts
- "Follow Topic" button
- topic page `/topic/[slug]`

This makes feed **personalized without backend AI yet**.

5️⃣ Reading Mode (This is HUGE)

Long posts are hard to read on screens.

Add a:

"Focus Reading Mode"

When user clicks:

> Enter Reading Mode

UI changes:

- hide sidebars
- center post
- increase font size
- warm background
- progress bar while reading
- 8 min read indicator

Now your platform feels like **Medium + Quora hybrid**.

This is a *massive realism upgrade*.

6️⃣ Contribution Quality Feedback

Right now users don’t know if they are doing well.

Add a contribution meter:

After posting show:

“Your post quality score: 82%”

Based on:

- length
- formatting
- readability
- tags

UI:
small analytics panel under post.

This increases writing quality dramatically.

7️⃣ Soft Moderation UI (Critical for Realism)

Real platforms always guide behavior.

Before posting, show:

Pre-Post Reminder

> “World of Peace encourages respectful, helpful answers. Avoid insults and personal attacks.”

Also add:

- edit window timer (10 min to edit)
- post history (edited 2 times)

Now your app feels like a real community.

8️⃣ Who Viewed Your Profile

Very addictive but simple frontend feature.

Add:

`Profile → Visitors`

Show mocked users:

- viewed 2 hours ago
- viewed yesterday

People *will come back just to check this*.

9️⃣ Onboarding Interest Selection

Right after signup:

“Choose 5 topics you care about”

(very important)

Show:

- chips
- selectable tags
- visual icons

This immediately fixes empty feed problem.

🔟 Weekly Email Digest (Frontend Simulation)

Even without backend, simulate:

Notification panel shows:

Weekly Digest

> “5 new answers in JavaScript this week”

This gives a sense of a living ecosystem.

1️⃣1️⃣ Reputation Levels (Gamification)

Instead of just points:

Create levels:

Points	Rank
0–50	Newcomer
50–200	Contributor

Points	Rank
200–700	Helper
700–2000	Mentor
2000+	Sage

Show badge near username.

This is extremely powerful psychologically.

1️⃣2️⃣ Post Reaction System (But NOT Emoji Spam)

Do not copy Facebook.

Add only:

- Helpful
- Insightful
- Thankful

This fits your platform philosophy.

1️⃣3️⃣ Gentle Notifications (Tiny but Powerful)

Add small UI details:

- “Someone appreciated your answer”
- “3 people saved your post”
- “You helped 12 people this week”

This creates emotional reward.

1️⃣4️⃣ Community Guidelines Page

Make a proper page:

`/guidelines`

This surprisingly increases trust a LOT.

Users subconsciously treat the platform as real.

1️⃣5️⃣ Empty State Design (Most important overlooked thing)

Right now when no posts exist → blank.

Instead show:

“Be the first to help someone today.”

- illustration
- Ask Question button

This is what real apps do.

What You Have Now vs What Makes It Real

Right now you have:

> a working social platform UI

After adding these:

> a believable living community product

This is the difference between:
portfolio project vs **startup MVP**

If you want, next I can tell you **the exact order you should implement these** (very important — wrong order will waste weeks).

> what more i can add also i want to add images user can select images beside to post optional and show see in feed also

Nice — now you’re thinking like a product builder 👍
At this stage you don’t need “bigger” features... you need **social realism**.
People stay on a platform when they can *express*, not only *write*. Text-only communities die fast.
So yes — **image posts** are actually a very important upgrade.

I’ll give you two things:

1. What extra real-world features you can still add (high value ones)
2. A ready **3rd prompt** you can paste to your AI so it adds image posting + feed preview properly.

🧠 More Real-World Frontend Features You Can Add

---

1. Post Collections (Underrated but Powerful)

Let users organize saved content.

Instead of just “Saved”, allow:

- Create collection

- Private/Public collection
- Share collection link

Examples:

- “React Learning”
- “Mental Health Advice”
- “Startup Tips”

This makes your platform a **knowledge organizer**, not just a social feed.

---

2. Anonymous Posting Mode

Very important for honesty (Quora grew because of this).

Add a toggle in post editor:

`Post as Anonymous`

Behavior:

- Profile hidden
- Shows “Anonymous Helper”
- Moderators can still see real user (mock UI)

Users will post sensitive topics:

- relationships
- mental health
- career stress

This massively increases activity.

---

3. Poll Posts

People don’t always want to write.

Add post type:
Poll

Options:

- up to 6 choices
- vote once
- show percentage bar animation

This boosts engagement a lot.

---

4. Reading Time Indicator

On every post:

```
5 min read
```

(estimate from word count)

This small detail instantly makes the platform feel professional.

---

5. Edit History Viewer

Add:

```
Edited 2 times → View history
```

Show previous versions in modal.

This builds **trust** — real knowledge platforms do this.

---

6. Soft Presence System

Show:

- “3 people are reading this post”
- “Someone is writing an answer…”

You don’t even need real realtime yet — simulate it.

Your platform will suddenly feel *alive*.

---

7. Bookmark Highlights

When a user selects text in a post:

Show small popup:

```
Highlight | Save quote
```

Then store highlighted quotes in saved section.

Now it becomes a study tool.

---

8. Appreciation Messages

After someone helps you, allow:

```
Send Thanks
```

This sends a small appreciation notification.

This is psychologically powerful — StackOverflow and Quora rely on this behavior.

---

9. Progressive Onboarding Tips

Don’t show everything at once.

Add floating helper tips:

- “Try answering a question”
- “Follow topics to improve your feed”
- “Use credits to summarize long posts”

Real apps teach users slowly.

---

10. Post Insights (For Writers)

Under their post show:

- Views
- Reads
- Saves
- Shares
- Avg read time

Writers LOVE analytics → they return daily.

🖼️ Now — Image Posting Feature (Important)

You want:

> user can optionally add images and they appear inside feed cards

This is correct — but **do NOT treat it like Instagram**.
Your platform is *knowledge-first*.
Images should support content, not replace it.

So images must be:

- inside post
- previewed in feed
- expandable
- not full photo social network.

📄 THIRD PROMPT — Add Image Post & Media Support

Paste this after your previous prompts.

Instruction

Extend the existing **World of Peace frontend project** by adding **image attachment support** to posts while preserving the knowledge-focused UI.
This is not a social photo app — images support discussions and answers.

Do not change existing layouts or design system.

---

1. Post Editor — Image Upload

Inside Create Post / Ask Question editor:

Add section:
Attach Images (Optional)

Requirements:

- Allow multiple images (max 4)
- Drag & drop upload
- Click to upload
- Show preview thumbnails
- Remove image button (X icon)
- Reorder images

Supported:

- JPG
- PNG
- WEBP

Use local preview (URL.createObjectURL mock).

---

2. Image Validation

Implement UI validation:

- Max 5MB each
- Show error message if exceeded
- Disable post if uploading

Show uploading progress bar (simulated).

---

3. Feed Display (Very Important)

In feed post cards:

If post contains images:

Show a **media preview grid**

Rules:

- 1 image → full width rounded image
- 2 images → 2-column grid
- 3–4 images → compact grid

Images must:

- lazy load
- maintain aspect ratio
- have rounded corners
- clickable

---

4. Image Viewer Modal

Clicking an image opens a modal viewer:

Features:

- dark backdrop
- zoom
- next/previous arrows
- ESC to close
- swipe on mobile

Include caption if post has text.

---

5. Post Detail Page

On full post view:

Images appear:

- between title and content
- responsive
- centered
- max width 720px

---

6. Answer Images

Allow users to attach images inside answers too.

In the rich editor:
Add image insert button.

---

7. Performance UX

Add:

- skeleton placeholder while loading
- blurred image preview
- lazy loading

---

8. Accessibility

Include:

- alt text input per image
- keyboard navigation in viewer

---

9. Empty Image State

If user tries to post only images:

Show warning:

> “Please include some text so others understand your post.”

(knowledge platform rule)

---

10. Styling Rules

Follow existing design system:

- border-radius: 12px
- no harsh drop shadows
- calm UI
- no Instagram-like heart overlays

Final Result

The platform should now support:

- text posts
- image-supported discussions
- answer illustrations
- expandable media in feed

The feature must feel like **Quora articles with supportive visuals**, not a photo-sharing network.

After this, your app becomes a **realistic knowledge community** — not just a UI clone anymore.

If you want next, I can also design the **database structure you should prepare later in backend** (trust me, this is where 80% of social apps fail).