Functional Requirements

User Authentication and Profiles:

- Users can create an account (sign up) and log in/log out.
- Users can authenticate via email/password and optionally OAuth (Google, GitHub, etc.).
- Users can reset or recover forgotten passwords.
- Users can manage their profile settings (username, bio, profile picture).

Prompt Crafting & Optimization:

- Users can create new prompts from scratch.
- Users can select the AI model type (e.g., ChatGPT, MidJourney, Stable Diffusion) when crafting a prompt.
- The system suggests improvements to user-written prompts using an AI-powered optimizer.
- The system can provide real-time writing assistance (e.g., grammar check, tone adjustment, clarity suggestions).

Prompt Templates and Frameworks:

- Users can view and use predefined prompt templates for different AI models.
- Users can customize templates to fit their specific needs.

Prompt Saving and Organization:

- Users can save prompts to their personal library.
- Users can tag prompts with custom tags (e.g., "marketing", "creative writing", "art generation").
- Users can organize prompts into folders or categories.

Search and Filtering:

- Users can search their saved prompts by keywords, tags, or AI model type.
- Users can filter prompts by date created, model type, popularity, or tags.

Prompt Sharing and Collaboration:

- Users can share prompts via a unique link.
- Users can control sharing settings: public, private, or team-only.
- Users can view and duplicate prompts shared by others (with attribution).

Version Control:

- Users can see the history of edits for a prompt.
- Users can revert to an earlier version of a prompt.

Prompt Execution Support:

- (Optional) Integration links to external tools (e.g., open ChatGPT with a prompt ready to paste).
- (Optional) Preview how a prompt might perform or be interpreted.

Notifications:

- Users receive notifications for system updates, collaboration invites, or prompt comments (if collaboration is enabled).
- Users can customize notification preferences.

Admin and Moderation Features:

- Admins can manage users (ban/suspend, promote, etc.).
- Admins can moderate public prompts and user reports.

Nonfunctional Requirements

Performance:

- The app must load within 3 seconds on a standard broadband connection.
- Search results should display within 1 second after query submission.

Scalability:

- The system should handle scaling from 100 to 100,000 users with minimal impact on performance.
- Backend should support horizontal scaling (e.g., microservices architecture or cloud scaling).

Availability:

- System should aim for 99.9% uptime.
- Recovery time objective (RTO) should be < 30 minutes after a major failure.

Security:

- All passwords must be stored using salted, hashed encryption.
- The system must use HTTPS for all data transmission.
- Input validation must be enforced on all fields (to prevent SQL Injection, XSS, etc.).
- Role-based access control (RBAC) for admin vs. standard users.
- Rate limiting and CAPTCHA must be used to prevent abuse and bot attacks.

Maintainability:

- Codebase must follow standard coding conventions (e.g., ESLint, Prettier).

- The app should be modular, allowing features to be updated independently.
- Full API documentation must be maintained.

Portability:

- Web application must be responsive across mobile, tablet, and desktop devices.
- Browser compatibility: Chrome, Firefox, Safari, and Edge (latest two versions).

Usability:

- Clean, modern, minimalist interface.
- Onboarding tutorial for new users explaining key features.
- Contextual tooltips and help buttons where needed.
- WCAG 2.1 AA accessibility compliance (e.g., keyboard navigation, screen reader compatibility).

Backup and Recovery:

- Daily backups of user prompt data.
- Ability to restore system from backups within 24 hours.

Analytics and Logging:

- Track user interactions (anonymized) to improve UX.
- Log all critical system errors and user actions (e.g., login attempts, prompt saves).

Localization (Optional for later phases):

- Support multiple languages (starting with English, then expanding).
-
-
-
-

Persona 1: "Creative Explorer" – Sofia Martinez

- **Background**:
  - Age: 27
  - Occupation: Freelance Digital Artist and Illustrator
  - Location: Austin, TX
  - Tech Comfort Level: High
  - Tools Used: MidJourney, Stable Diffusion, Photoshop, Procreate
- **Goals**:
  - Quickly create compelling art prompts that generate specific styles or moods.
  - Organize hundreds of prompts by project and theme.
  - Experiment with prompt tweaking to improve art quality.
- **Frustrations**:
  - Losing good prompts because she forgets to save or organize them.
  - Difficulty refining prompts without wasting time trial-and-error.
  - No centralized place to manage visual AI prompts across projects.
- **Motivations**:
  - Produce better AI-generated art for client commissions.
  - Increase productivity by refining prompts faster.
  - Build a personal "prompt library" for ongoing inspiration.
- **How Sofia would use PromptPath**:
  - Use optimization suggestions to polish visual prompts for Stable Diffusion.

    - Save, tag, and organize prompts into folders by art project (e.g., "Fantasy Portraits").
    - Share favorite prompt collections with other artists via public links.
    - Save different versions of the same prompt to track how small tweaks impact results.

Persona 2: "Knowledge Seeker" – Daniel Kim

- **Background**:
    - Age: 34
    - Occupation: Content Strategist at a Marketing Agency
    - Location: Seattle, WA
    - Tech Comfort Level: Medium-High
    - Tools Used: ChatGPT, Notion, Google Docs, Jasper.ai
- **Goals**:
    - Craft clear, detailed prompts that generate marketing copy and blog posts.
    - Collaborate with coworkers to develop effective AI strategies.
    - Maintain a library of tested, reliable prompts for different content types.
- **Frustrations**:
    - Generic or vague prompts leading to low-quality AI outputs.
    - Rewriting the same types of prompts for different clients without a system.
    - Lack of version control when iterating prompts with teammates.
- **Motivations**:

  - Deliver better content faster to clients by mastering prompt engineering.
  - Stand out professionally by using AI tools more effectively.
  - Reduce the mental load by relying on proven templates.
- **How Daniel would use PromptPath**:
  - Use templates and writing assistance to polish marketing prompts.
  - Save prompts tagged by client or content type (e.g., "SEO Blog", "Ad Copy").
  - Invite coworkers to review and collaborate on prompt optimization.
  - Use search/filter tools to find and reuse prompts efficiently.

Persona 3: "Tech Innovator" – Priya Desai

- **Background**:
  - Age: 42
  - Occupation: AI Startup Founder and Developer
  - Location: San Francisco, CA
  - Tech Comfort Level: Very High (Expert)
  - Tools Used: Custom LLMs, OpenAI API, GitHub, Postman, VS Code
- **Goals**:
  - Design highly optimized system prompts for proprietary AI models.
  - Maintain strict versioning and backup of prompt iterations.
  - Analyze and improve prompts based on model responses.
- **Frustrations**:

    - Existing prompt tools are too simplistic for complex AI engineering.
    - Managing evolving system prompts in messy spreadsheets and docs.
    - Lack of serious collaboration/versioning support for prompt engineering.
- **Motivations**:
    - Create better user experiences in her AI products through better prompts.
    - Reduce team errors by systematizing prompt development workflows.
    - Gain competitive advantage through high-quality prompt engineering.
- **How Priya would use PromptPath**:
    - Build complex prompt libraries organized by product and use case.
    - Track detailed version histories and revert when needed.
    - Collaborate securely with engineers to refine prompts.
    - Analyze prompt changes and optimize based on model feedback.

Use Case Diagram (PlantUML)

```
@startuml
actor User
actor Admin

rectangle PromptPath {
  User -- (Sign Up / Log In)
  User -- (Create New Prompt)
  User -- (Optimize Prompt)
  User -- (Save Prompt)
  User -- (Browse Community Library)
  User -- (Rate Prompts)
  User -- (View Prompt Details)
  User -- (Edit Saved Prompts)

  Admin -- (Manage Users)
  Admin -- (Moderate Community Prompts)
}
```

```
(Sign Up / Log In) .> (Create New Prompt) : includes
(Create New Prompt) .> (Optimize Prompt) : optional
(Save Prompt) .> (Create New Prompt) : extends
(View Prompt Details) .> (Rate Prompts) : includes
(Browse Community Library) .> (View Prompt Details) : includes
@enduml
```

Detailed Textual Use Case Descriptions

---

1. Sign Up / Log In

- **Actor**: User
- **Description**: Users register with an email and password or log in with credentials or third-party services like Google/GitHub.
- **Preconditions**: User is not logged in.
- **Postconditions**: User gains access to their dashboard.
- **Extensions**: Password recovery, OAuth login.

---

2. Create New Prompt

- **Actor**: User
- **Description**: Users input a new prompt for an AI model (e.g., ChatGPT, MidJourney).
- **Preconditions**: User must be logged in.
- **Postconditions**: Prompt is ready for optimization or saving.
- **Extensions**: Includes access to optimization tools after writing.

---

3. Optimize Prompt

- **Actor**: User
- **Description**: The system analyzes the draft prompt and suggests improvements (e.g., clarity, specificity, creativity).
- **Preconditions**: Prompt must be created or loaded.
- **Postconditions**: Improved prompt is presented to the user for review.

---

4. Save Prompt

- **Actor**: User
- **Description**: Users save the optimized or original prompt to their personal library with tags and model type.
- **Preconditions**: User has written or optimized a prompt.
- **Postconditions**: Prompt is stored in the database, accessible for future use.

---

5. Browse Community Library

- **Actor**: User
- **Description**: Users explore public prompts created by others, searchable by tags, popularity, or model type.
- **Preconditions**: User is logged in (optional for public browsing).
- **Postconditions**: User can view and interact with public prompts.
- **Extensions**: Ability to filter, search, or sort prompts.

---

6. View Prompt Details

- **Actor**: User
- **Description**: User views the full content, creator, tags, and related prompts for a selected prompt.
- **Preconditions**: User is browsing the library or their saved prompts.
- **Postconditions**: Prompt details are displayed; user can rate, copy, or save.

---

7. Rate Prompts

- **Actor**: User
- **Description**: Users can rate prompts with a simple star system or thumbs up/down.
- **Preconditions**: User is viewing a prompt detail page.
- **Postconditions**: Rating is stored and affects prompt popularity metrics.

---

8. Edit Saved Prompts

- **Actor**: User
- **Description**: Users edit prompts they have previously saved.
- **Preconditions**: User must have a saved prompt.
- **Postconditions**: Updates overwrite or create a new version depending on user choice.

---

9. Manage Users

- **Actor**: Admin
- **Description**: Admins can view, suspend, or delete user accounts.

- **Preconditions**: Admin must be authenticated.
- **Postconditions**: Changes to user accounts are saved.

---

10. Moderate Community Prompts

- **Actor**: Admin
- **Description**: Admins can remove inappropriate prompts, highlight top-rated prompts, and manage public libraries.
- **Preconditions**: Admin must be authenticated.
- **Postconditions**: Community library content is curated and moderated.

🏗️ PromptPath System Architecture Overview

🖼️ 1. Frontend (Client-Side)

Tech Stack:

- **React** – Component-based UI development
- **Next.js** – Server-side rendering, routing, SEO-friendly pages
- **TypeScript** – Type-safe components and state management
- **Tailwind CSS (optional)** – For consistent and fast styling

Responsibilities:

- UI for logging in, prompt creation, optimization interface, prompt library browsing
- Communicates with backend via REST or GraphQL
- Manages sessions via cookies or JWT

- Displays responses from AI models (e.g., OpenAI)

🔧 2. Backend (Server-Side API)

Tech Stack:

- **Node.js** + **Express.js** (REST API framework)
- **TypeScript** – Strong typing and maintainable code
- **Redis** – For caching AI responses and improving performance

Responsibilities:

- Authentication (JWT or session-based)
- Business logic for prompts: create, edit, optimize, save, rate
- AI integration: send prompt to OpenAI API and return response
- Fetch and serve user data, prompt history, community prompts
- Admin moderation and user management routes

🧠 3. AI API Integration

Tech Stack:

- **OpenAI API** (via REST)

Responsibilities:

- Send user prompts to ChatGPT or other models (e.g., `gpt-4`)
- Receive and return optimized or evaluated prompts

- (Optional): Use prompt templates or system instructions for tone/format

Caching:

- Use **Redis** to store recent AI responses by prompt hash to reduce redundant requests

---

🗃️ 4. Database Layer

Tech Stack:

- **PostgreSQL** – Relational database
- **Prisma ORM** or **Knex.js** (optional) – For data modeling and querying with TypeScript

Key Tables:

- `Users` – User accounts, roles, login credentials
- `Prompts` – Raw/optimized prompt content, model type, metadata
- `Ratings` – User ratings and comments on community prompts
- `Tags` – Tagging system for prompt organization
- `PromptVersions` – Version control for user prompts

⚡ 5. Redis Caching Layer

Tech Stack:

- **Redis** – In-memory key-value store

Use Cases:

- Cache AI responses based on input hash

- Store session data (if using server-side sessions)
- Reduce database load for frequent queries (e.g., top-rated prompts, tag lists)

🔒 6. Authentication & Security

Possible Tools:

- **NextAuth.js** (for OAuth + session-based auth in Next.js)
- **bcrypt** for password hashing
- **Helmet.js** for securing HTTP headers
- **Rate limiting middleware** to protect against abuse

📡 7. Deployment / Infrastructure

Suggested Tools:

- **Frontend**: Vercel (Next.js-native) or Netlify
- **Backend**: Render, Heroku, Railway, or DigitalOcean
- **Database**: Supabase, Neon, or self-hosted PostgreSQL
- **Redis**: Upstash (serverless Redis), or managed Redis from Redis Enterprise
- **CI/CD**: GitHub Actions, Vercel auto-deploy

🔄 System Interaction Flow

1. User logs in via frontend (Next.js) → backend (Express) verifies with DB
2. User types a prompt → frontend sends it to backend via API

3. Backend checks Redis cache → if hit, return cached AI response
4. If no cache, backend sends prompt to OpenAI → receives response
5. Backend stores prompt + response in PostgreSQL → caches in Redis
6. User saves or rates the prompt → stored in `Prompts` and `Ratings` tables
7. Community Library displays most recent/highest-rated prompts → cached in Redis
8. Admin dashboard accesses moderation tools via secure backend route

🏠 1. Home / Landing Page

```
-----------------------------------------------------
| PromptPath [Logo]           [Login] [Register]    |
-----------------------------------------------------
|   Headline: "Master Prompt Engineering with AI"   |
|---------------------------------------------------|
|   [ Get Started Button ]                          |
|   [ Watch Demo ]                                  |
|---------------------------------------------------|
|   ✔ Craft better prompts                          |
|   ✔ Optimize for ChatGPT, MidJourney, etc.        |
|   ✔ Save, share, and explore                      |
-----------------------------------------------------
|        [ Features ]  [Testimonials]  [FAQ]        |
-----------------------------------------------------
|                 Footer: About | Contact           |
-----------------------------------------------------
```

🔐 2. Login / Register Page

```
-----------------------------------------------------
| PromptPath [Logo]                                 |
-----------------------------------------------------
|             [ Login to Your Account ]             |
|---------------------------------------------------|
|   Email:      [________________________]          |
|   Password:   [________________________]          |
|                                                  |
|   [ Log In ]                                      |
|   [ Login with Google ]                           |
|   Don't have an account? [ Register Here ]        |
-----------------------------------------------------
|     OR                                             |
-----------------------------------------------------
|             [ Create New Account ]                |
|   Name:        [_______________________]          |
|   Email:       [_______________________]          |
|   Password:    [_______________________]          |
|   [ Register ]                                    |
-----------------------------------------------------
```

✍️ 3. Prompt Editor Page

```
-----------------------------------------------------
| [PromptPath Logo]   [Home] [Library] [Profile]    |
-----------------------------------------------------
|    Create New Prompt                              |
|---------------------------------------------------|
| Model: [ ChatGPT ▼ ]                               |
| Tags:  [ creativity, SEO ]                        |
|---------------------------------------------------|
| Prompt:                                           |
| [                                               ] |
| [  "Write a story about a space-traveling cat..." ]|
| [                                               ] |
|---------------------------------------------------|
| [ Optimize Prompt ]   [ Save Prompt ]  [ Cancel ] |
|---------------------------------------------------|
|   Suggested Improvements (AI Feedback):          |
|   - Try specifying tone                          |
|   - Consider shortening for clarity              |
-----------------------------------------------------
```

🌐 4. Community Library Page

```
-----------------------------------------------------
| [PromptPath Logo]   [Editor] [Library] [Profile]  |
-----------------------------------------------------
|               🔍 [ Search prompts... ]             |
|---------------------------------------------------|
|   🔖 Filters: [ ChatGPT ] [Popular] [Creative]     |
-----------------------------------------------------
|  🔹 Prompt: "Write a job description for..."       |
|     Tags: hiring, business    ❤️ 24 Ratings        |
|     [ View ]  [ Save ]                            |
-----------------------------------------------------
|  🔹 Prompt: "Generate a logo idea for..."          |
|     Tags: design, branding     ❤️ 10 Ratings       |
|     [ View ]  [ Save ]                            |
-----------------------------------------------------
|                   [ Load More ↓ ]                |
-----------------------------------------------------
```

👤 5. Profile Page

```
-----------------------------------------------------
| [PromptPath Logo]   [Editor] [Library] [Logout]   |
-----------------------------------------------------
|   👤 Username: alex_rivera                         |
|   📧 Email: alex@example.com                       |
|---------------------------------------------------|
|   My Saved Prompts (12)                           |
|   ----------------------------------------------  |
|   • "Cold email for product launch"    [ Edit ]   |
|   • "Landscape scene for MidJourney"  [ Edit ]   |
|   • "Newsletter intro text"           [ Edit ]   |
|   ----------------------------------------------  |
|   [ Export My Prompts ]  [ Delete Account ]       |
-----------------------------------------------------
```