Here is the **Zero-Code User Blueprint for SPARC Program Generation:**

---

**Project Title:** (Give your program idea a simple name)

**Prepared By:** (Your Name)

**Date:** (Today's Date)

---

**Instructions for You (The Visionary!):**

- **No Tech Jargon Needed!** Just describe your idea in plain English. Think about *what* you want the program to do and *why*, not *how* it does it technically.
- **Be Detailed:** The more information and specific examples you give, the better the AI (our team of virtual coding assistants, called SPARC) can understand and build exactly what you want. Imagine you're describing it to someone who needs to build it perfectly without asking you follow-up questions.
- **Focus on the Goal:** What problem does this solve? What process does it make easier?
- **Don't Worry About Code:** SPARC will figure out the best programming languages, databases, and technical stuff based on your description and its own research.

---

**Section 1: The Big Picture - What is this program all about?**

1. **Elevator Pitch:** If you had 30 seconds to describe your program to a friend, what would you say? What's the main goal?
   - *Your Answer:*
2. **Problem Solver:** What specific problem does this program solve, or what task does it make much easier or better?
   - *Your Answer:*
3. **Why Does This Need to Exist?** What's the key benefit it offers? (e.g., saves time, saves money, organizes information, connects people, provides entertainment, etc.)
   - *Your Answer:*

**Section 2: The Users - Who is this program for?**

1. **Primary Users:** Describe the main type of person (or people) who will use this program. (e.g., small business owners, students, hobbyists, families, everyone, etc.)
    - *Your Answer:*
2. **User Goals:** When someone uses your program, what are the top 1-3 things *they* want to accomplish with it?
    - *Example: For a recipe app, users might want to: 1. Find recipes quickly. 2. Save favorite recipes. 3. Create a shopping list.*
    - *Your Answer:*
        - 1.
        - 2.
        -
            1. (Add more if needed)

**Section 3: The Features - What can the program *do*?**

1. **Core Actions:** List the essential actions or tasks users can perform within the program. Be specific. Use action words.
    - *Example: Create an account, Log in, Search for items, Add item to cart, View cart, Check out, View order history, Write a review, Upload a photo, Send a message.*
    - *Your Answer (List as many as needed):*
        -
            -
        -
            -
        -
            -
2. **Key Feature Deep Dive:** Pick the MOST important feature from your list above. Describe step-by-step how you imagine someone using *that specific feature* from start to finish. What do they see? What do they click? What happens next?

   - *Your Answer:*

**Section 4: The Information - What does it need to handle?**

1. **Information Needed:** What kinds of information does the program need to work with, store, or display?
   - *Examples: Usernames, passwords, email addresses, product names, prices, descriptions, photos, dates, customer addresses, order details, blog post text, comments.*
   - *Your Answer (List all types):*
     - 
       - 
     - 
       - 
     - 
       - 
2. **Data Relationships (Optional but helpful):** Do any pieces of information naturally belong together?
   - *Example: An "Order" includes a list of "Products", a "Customer Address", and a "Date". A "Blog Post" has "Comments" associated with it.*
   - *Your Answer:*

**Section 5: The Look & Feel - How should it generally seem?**

1. **Overall Style:** Choose words that describe the general vibe. (e.g., Simple & Clean, Professional & Formal, Fun & Colorful, Modern & Minimalist, Artistic & Creative, Rugged & Outdoorsy)
   - *Your Answer:*
2. **Similar Programs (Appearance):** Are there any existing websites or apps whose *look* (not necessarily function) you like? Mentioning them helps the AI understand your visual preference.
   - *Your Answer:*

**Section 6: The Platform - Where will it be used?**

1. **Primary Environment:** Where do you imagine most people using this program? (Choose one primary, others secondary if applicable)
    - [ ] On a Website (accessed through Chrome, Safari, etc.)
    - [ ] As a Mobile App (on iPhone/iPad)
    - [ ] As a Mobile App (on Android phones/tablets)
    - [ ] As a Computer Program (installed on Windows)
    - [ ] As a Computer Program (installed on Mac)
    - [ ] Other (Please describe):
    - *Your Primary Choice & any Secondary Choices:*
2. **(If Mobile App):** Does it need to work *without* an internet connection sometimes? (Yes/No/Not Sure - AI will research implications)
    - *Your Answer:*

**Section 7: The Rules & Boundaries - What are the non-negotiables?**

1. **Must-Have Rules:** Are there any critical rules the program *must* follow?
    - *Examples: Users must be over 18, Prices must always show tax included, Specific information must be kept private, A specific calculation must be performed exactly this way.*
    - *Your Answer:*
2. **Things to Avoid:** Is there anything the program should absolutely *not* do?
    - *Examples: Never share user emails, Don't allow users under 13 to sign up, Don't automatically charge a credit card.*
    - *Your Answer:*

**Section 8: Success Criteria - How do we know it's perfect?**

1. **Definition of Done:** Describe 2-3 simple scenarios. If the program handles these scenarios exactly as described, you'd consider it a success for that part.
    - *Example 1: "When I sign up with my email and password, I should get a confirmation email, and then be able to log in immediately."*

- *Example 2: "When I search for 'blue widgets', it should show me only blue widgets, displaying their picture, name, and price."*
- *Your Scenarios:*
    - 1.
    - 2.
    - 3.

**Section 9: Inspirations & Comparisons - Learning from others**

1. **Similar Programs (Functionality):** Are there any existing programs, websites, or apps that do something similar to what you envision (even if only partly)?
    - *Your Answer (List names if possible):*
2. **Likes & Dislikes:** For the programs listed above (or similar ones you know), what features or ways of doing things do you REALLY like? What do you REALLY dislike or find frustrating? This helps SPARC build something better.
    - *Likes:*
    - *Dislikes:*

**Section 10: Future Dreams (Optional) - Where could this go?**

1. **Nice-to-Haves:** Are there any features that aren't essential right now but would be great to add later?
    - *Your Answer:*
2. **Long-Term Vision:** Any thoughts on how this program might evolve in the distant future?
    - *Your Answer:*

**Section 11: Technical Preferences (Strictly Optional!)**

- **Note:** Our AI assistants are experts at choosing the best technical tools. Only fill this out if you have a *very strong*, specific reason for a particular choice (e.g., compatibility with an existing system you *must* use).

1. **Specific Programming Language?** (e.g., Python, JavaScript, Java) **Why?**
    - *Language:*

   - *Reason (Mandatory if language specified):*

2. **Specific Database?** (e.g., Supabase, PostgreSQL, MySQL) **Why?**
   - *Database:*
   - *Reason (Mandatory if database specified):*
3. **Specific Cloud Provider?** (e.g., Google Cloud, AWS, Azure) **Why?**
   - *Provider:*
   - *Reason (Mandatory if provider specified):*

---

**Final Check:**

- Have you answered all the questions in Sections 1-9 as clearly and detailed as possible?
- Have you used simple, everyday language?
- Have you focused on the *what* and *why*?

**Ready to Build!**

Once you submit this completed blueprint, the SPARC orchestration will take over. It will:

1. Use **Deep Research** to analyze your vision, explore similar programs, investigate technical options, and fill in any knowledge gaps.
2. Use the **Specification Writer** to turn your answers and the research into formal requirements.
3. Use the **Architect** to design the system structure.
4. Use **Code**, **TDD**, **Supabase Admin**, **MCP Integration**, and **Security Reviewer** modes iteratively to build, test, and secure the program based on the specifications and architecture.
5. Use the **System Integrator** to connect all the pieces.
6. Use the **Documentation Writer** to create guides.
7. Use **DevOps** to set up infrastructure and deploy the application.
8. Finally, it will present the completed, working program to you based on the Success Criteria you provided!