1. Introduction
   * Purpose: Briefly explain the purpose of the system being designed.
   * Scope: Describe the scope of the system, including the intended audience, features, and functionalities.
2. System Overview
   * High-level Description: Provide a high-level overview of the system's architecture, components, and their interactions.
   * Key Technologies: List the main technologies, frameworks, and tools used in the system.
3. Architectural Design
   * Architectural Patterns: Describe the architectural patterns used, such as client-server, layered, microservices, etc.
   * System Components: Define the main components or modules of the system, their responsibilities, and interactions.
4. Interface Design
   * External Interfaces: Describe the external interfaces exposed by the system, such as APIs, web services, etc.
   * Internal Interfaces: Define the internal interfaces between system components or modules.
5. Data Model and Storage
   * Data Model: Describe the data model, including entities, relationships, and attributes.
   * Storage Solutions: Explain the storage solutions used, such as relational databases, NoSQL databases, file systems, etc.
6. Security and Privacy
   * Authentication and Authorization: Detail the authentication and authorization mechanisms implemented in the system.
   * Data Security: Explain the measures taken to ensure data security, such as encryption, secure communication, etc.
   * Privacy Considerations: Address privacy concerns and compliance with relevant regulations.
7. Performance and Scalability
   * Performance Optimization: Describe performance optimization techniques and strategies used in the system.
   * Scalability Considerations: Explain how the system can handle increased load and growth.
8. Testing and Quality Assurance
   * Testing Strategy: Outline the testing strategy, including unit tests, integration tests, system tests, and performance tests.
   * Quality Assurance: Explain the quality assurance processes and tools, such as code reviews, static analysis, etc.
9. Deployment and Monitoring
   * Deployment Strategy: Describe the deployment strategy, including environments, CI/CD pipelines, and configuration management.
   * Monitoring and Alerting: Explain the monitoring and alerting mechanisms implemented to ensure system stability and reliability.
10. Maintenance and Support
    * Maintenance Plan: Outline the plan for system maintenance, including regular updates, bug fixes, and performance improvements.
    * Support Strategy: Describe the support provided to users, such as documentation, training, and helpdesk services.

Purpose\users\features

Non-functional requirements

Scale (QPS)\performance

\accuacy\freshness\consistency\security

API link front end and back end (RESTFUL API)

Diagram

Description automatically generated

WebSocket (stateful)

ICCRPA Platform Development Timeline

Starting from 04/24:

Week 1: Requirement Analysis and Research

1. Determine the target user group, requirements, and core features of the platform (currently basically determined, just need to sort out the existing icppra and question collation)
2. Collect competitor information and analyze market conditions (thoroughly understand Focus Health and sort out its success core)
3. Write project requirement documents (mainly simplified, sort out points 1 and 2)

Week 2-3: Interface Design and Overall Design

1. Design the platform's interaction framework and establish user operation process (System Design file, including two aspects: user usage topology, platform building topology)
2. Design the visual style of the platform, including colors, fonts, icons, etc. (find the most suitable award-winning UI design on <https://www.awwwards.com/>, select our theme color scheme on <https://colorhunt.co/>, and find the template for each Nav on <https://ui-patterns.com/explore>)
3. Create specific interface prototypes (design a visual UI interface using Balsamiq or Canva)
4. Write design documentation for future development reference (keep it as simple as possible, just explain clearly)

Week 4-6: Platform Prototype Development (Prototype)

1. Select technology stack and development tools (first develop the front end, React/HTML/CSS can be completed in about a week; CSS specific adjustments are mainly based on the Bootstrap library)
2. Build the basic framework and page structure of the platform (use Spring frame for the backend logic part, and Prisma for the database to build a simple registration and sign-up function)
3. Implement core functions and interaction logic (debug the relationship between front, back, and database)
4. Complete prototype development and conduct internal testing

Week 7-10: Platform Interaction and Function Optimization

1. Adjust and optimize the prototype based on internal test results (strictly follow the 28 law, first solve 80% of the problems with 20% of the time)
2. Develop other functions of the platform, such as payment, search, and comments (payment is the most important, comments are secondary, and search is the least important)
3. Enhance the platform's user experience, such as animation effects and response speed (React is fast, Spring completes encapsulation, and there is not much to do for SEO optimization)
4. Integrate third-party services, such as maps and social media (ChatGPT API can be integrated later, which is currently free, not difficult to integrate, but the practicality is not high because the data volume is too small)

Week 11-12: System Testing and Repair

1. Perform system function tests to ensure functional correctness (test various corner cases)
2. Perform performance tests and optimize system response speed and resource usage (directly deploy on AWS, saving many operations)
3. Conduct security tests to guard against potential security risks (customer password security encryption)
4. Fix problems found during testing

Week 13-14: Official Platform Launch and Promotion

1. Deploy the platform to the production environment (officially replace the WordPress-developed iccpra)
2. Conduct final testing before launch
3. Launch the platform and promote it (can be replicated at various points across the US)
4. Collect user feedback and continuously optimize