**需求文档**

## 项目概述

本项目是一个基于 Web 的游戏平台，其中包括前端、使用 Flask 框架的后端、数据库和 AI 技术。游戏的主要玩法是用户选择图片，通过人工选择纠正机器审美，图片迭代后将放到 NFT 平台上售卖，为自闭症家庭赚取治疗费用，玩家通过参与游戏，可以为自闭症家庭赚取治疗费用，同时也可以参与到一次艺术创作中。

建造平台的目的是让自闭症在平台展现自我、实现自我，帮助自闭症家庭赚取治疗费用，吸引普通人参与，增加自闭症的社会关注度。

## 功能需求

1. 首页：展示网站的基本信息，包括网站名称、网站简介、网站主题等。还可以放置一些关于自闭症的相关信息和链接，以便用户了解更多信息。

2. 游戏介绍页面：介绍游戏的规则、奖励和背景，让用户了解游戏的基本情况。

3. 画廊页面：展示已生成的NFT作品，并提供购买链接。可以按照不同的分类方式展示作品，如画风、作者、价格等。

4. 自闭症科普页面：介绍自闭症的基本情况、症状和治疗方法，帮助用户了解自闭症相关知识。还可以提供一些相关资源和链接，如支持组织、医疗机构等。

5. 用户注册和登录页面：用户可以注册账号并登录，以便在游戏中保存分数和排名。用户可以选择使用邮箱或者手机号码注册，也可以通过社交账号登录。

6. 游戏页面：包括游戏主界面、游戏规则、倒计时、投票按钮、用户分数和排名等。游戏页面需要能够根据不同的轮次动态更新，展示不同的图片和投票结果。

## 游戏规则

1. 玩家可以登录后点击参与游戏，初始得分为0。

2. 游戏共有四局，每局包括8轮，每轮游戏持续10秒。

3. 每轮游戏从数据库中取出两张图片，展示给玩家进行选择。

4. 玩家选择其中一张图片进行投票，投票结果在游戏结束后进行计算。

5. 如果玩家选择的图片的投票数大于另一张图片，则玩家得分加一。否则得分不变。

6. 每轮游戏结束后，系统会向玩家反馈结果，显示玩家是否选对了得票数较多的图片。

7. 游戏结束后，系统会根据玩家得分排名，确定得分最高的玩家，并为其免费提供一幅 NFT 作品作为奖励。

## 游戏界面

游戏界面需要实现以下功能：

1. 动态效果：展示两张图片供玩家选择，玩家选择一张后，会根据选择的图片重新展示两张新的图片，形成类似于二叉树的动态效果。

2. 图片展示：图片存储在图床上，URL 地址存储在数据库中，需要通过后端从数据库中获取图片 URL，并在前端展示。

3. 投票功能：玩家可以点击图片进行投票，投票结果需要提交给后端进行统计。

4. 反馈功能：每轮游戏结束后，需要向玩家反馈游戏结果，显示玩家是否选择了得票数较多的图片。

5. 倒计时功能：每轮游戏需要持续10秒，需要在前端展示倒计时，倒计时结束后对本轮结果进行展示，5秒后自动进入下一轮游戏。

## 技术实现

本项目使用 Flask 框架实现后端逻辑，使用 MySQL 数据库存储图片 URL 和游戏数据。游戏界面的动态效果可以通过 JavaScript 或者 Vue/React 实现，实现方法是通过替换 img 标签的 href 属性实现动态刷新。为了避免频繁从数据库中获取图片 URL，可以在游戏开始前一次性获取所有图片 URL，然后根据逻辑判断进行选择。

## 技术需求

**前端：**

1. Web框架：使用HTML、CSS、JavaScript编写前端代码，可选用Vue.js、React.js等主流Web框架。

2. 图片处理：使用JavaScript实现图片处理和展示，要求图片可以动态加载，且加载速度快。

3. 网络通信：使用Ajax技术实现前后端数据交互，保证数据的准确性和及时性。

**后端：**

1. Web框架：使用Python编写后端代码，选用Flask框架实现Web服务。

2. 数据库：选用MySQL或者PostgreSQL等关系型数据库，存储游戏中的图片数据、用户信息和得分等相关数据。

3. 数据存储：使用Redis等内存数据库存储一些游戏中需要频繁访问的数据，如游戏配置信息、玩家登录信息等。

4. 机器学习：使用自主开发的VACGAN模型实现图片的迭代生成，提升图片的美观度和可玩性。

## 数据需求

1. 用户数据：包括用户ID、用户名、密码、邮箱、头像等个人信息
2. 游戏数据：包括游戏ID、游戏轮次、图片URL、图片选择结果、得分等数据
3. 画作数据：包括画作ID、作者、画作名称、画作风格、画作介绍、NFT售价等数据
4. 科普数据：包括科普ID、科普标题、科普内容等数据

## 非功能需求

1. 性能需求：游戏和画廊页面的加载速度应该快于3秒，确保用户不会因为等待页面加载而失去耐心。
2. 可靠性需求：游戏数据和用户数据的存储应该具备高可靠性，确保数据不会因为异常中断或其他问题而丢失。
3. 安全需求：用户的密码应该经过加密处理存储在数据库中，防止用户隐私被泄露。同时，应该采用CSRF防御等措施防止跨站脚本攻击。
4. 易用性需求：页面的UI设计应该美观大方，易于使用和导航。同时，游戏规则和画作介绍应该易于理解和阅读。
5. 可维护性需求：代码应该按照良好的编程习惯编写，注释清晰，易于维护。同时，应该定期进行代码审查和测试，确保代码质量。

• 可扩展性需求：系统应该具备较好的可扩展性，能够方便地添加新的游戏规则、画作和科普内容，满足用户不断变化的需求。窗体底端

## 测试需求

1. 单元测试：对于代码中的每一个函数或方法，都需要编写相应的单元测试，保证代码的正确性和健壮性。

2. 集成测试：在完成单元测试之后，需要对整个系统进行集成测试，保证各个模块之间的协作正常。

3. 性能测试：在完成集成测试之后，需要对系统的性能进行测试，包括并发性能、负载测试等，保证系统能够在高并发场景下正常运行。

4. 用户体验测试：在完成性能测试之后，需要对用户的游戏体验进行测试，包括游戏的流畅度、交互性、页面设计等方面，保证用户体验良好。

## 风险管理

1. 数据安全：在开发过程中，需要对数据库的安全性进行保护，采取一定的安全措施，如加密存储、访问控制等。

2. 网络安全：在Web服务中，需要对用户的网络请求进行安全控制，防止出现SQL注入、XSS攻击等恶意行为。

3. 隐私保护：在收集用户信息时，需要遵守相关法律法规，保护用户隐私，不泄露用户个人信息。

4. 稳定性：在高并发情况下，需要保证系统的稳定性，避免系统崩溃