**需求分析**

**参与开发者：张溪桐 汪李昀红 万沅昕**

1. **系统概述**
2. 产品目标

开发一款高效、安全、易用的文本防伪软件，通过先进的文本识别与验证技术，快速判断文本是否由人完成。

1. 目标用户

学生，教师，作者，审稿人。

1. 产品范围

功能范围

核心功能：文本防伪验证：通过输入或扫描文本信息，快速验证产品真伪。

用户管理：支持用户注册、登录、个人信息管理及商品绑定。

非功能范围

不支持除文本防伪外的其他防伪技术（如二维码、RFID等）。

1. **总体设计**

1.系统架构设计

文本防伪软件采用分层架构设计，主要分为用户界面层、业务逻辑层、数据存储层和接口层。这种分层架构有助于提高系统的可维护性、可扩展性和可测试性。

### 1. 用户界面层

- \*\*功能\*\*：为用户提供简洁直观的操作界面，支持多种设备和操作系统（如iOS、Android、Web）。

- \*\*实现方式\*\*：

- \*\*移动端\*\*：使用微信小程序或原生应用开发，支持扫描、输入文本等功能。

- \*\*Web端\*\*：使用HTML5、CSS3和JavaScript框架（如Vue.js或React.js）开发，提供企业用户管理后台。

- \*\*特点\*\*：响应式设计，适配不同屏幕尺寸；简洁的用户交互流程，降低用户学习成本。

### 2. 业务逻辑层

- \*\*功能\*\*：处理系统的业务逻辑，包括文本防伪验证、商品信息管理、用户管理等。

- \*\*模块划分\*\*：

- \*\*文本防伪验证模块\*\*：接收用户输入或扫描的文本信息，解密并与数据库中的防伪信息比对，返回验证结果。

- \*\*商品信息管理模块\*\*：存储和管理商品的详细信息，支持查询和更新。

- \*\*用户管理模块\*\*：处理用户注册、登录、权限管理等功能，支持用户绑定商品。

- \*\*数据统计与分析模块\*\*：收集用户行为数据，生成统计报告，支持企业决策。

- \*\*技术选型\*\*：使用Node.js或Python作为后端开发语言，结合Express.js或Flask框架。

### 3. 数据存储层

- \*\*功能\*\*：存储系统的数据，包括防伪信息、用户信息、商品信息等。

- \*\*数据库选择\*\*：

- \*\*关系型数据库\*\*：MySQL用于存储结构化数据，如用户信息、商品信息等。

- \*\*非关系型数据库\*\*：Redis用于缓存高频访问数据，提升系统性能。

- \*\*特点\*\*：数据加密存储，确保信息安全；分布式存储设计，支持高并发访问。

### 4. 接口层

- \*\*功能\*\*：提供系统内部模块间通信的接口，以及与外部系统的接口。

- \*\*实现方式\*\*：

- \*\*内部接口\*\*：使用RESTful API设计，模块间通过HTTP请求通信。

- \*\*外部接口\*\*：提供API文档，支持与其他系统（如ERP、电商平台）集成。

- \*\*特点\*\*：接口标准化，易于扩展和维护。

## 二、模块划分

### 1. 用户管理模块

- \*\*功能\*\*：用户注册、登录、个人信息管理、权限设置。

- \*\*接口\*\*：

- 用户注册：`POST /api/user/register`

- 用户登录：`POST /api/user/login`

- 用户信息更新：`PUT /api/user/{userId}`

### 2. 文本防伪验证模块

- \*\*功能\*\*：接收文本信息，进行解密和比对，返回验证结果。

- \*\*接口\*\*：

- 验证文本：`POST /api/verify/text`

- 查询商品信息：`GET /api/product/{productId}`

### 3. 商品信息管理模块

- \*\*功能\*\*：存储和管理商品信息，支持查询和更新。

- \*\*接口\*\*：

- 添加商品信息：`POST /api/product`

- 更新商品信息：`PUT /api/product/{productId}`

- 查询商品信息：`GET /api/product/{productId}`

### 4. 数据统计与分析模块

- \*\*功能\*\*：收集用户行为数据，生成统计报告。

- \*\*接口\*\*：

- 提交用户行为数据：`POST /api/analytics/behavior`

- 查询统计报告：`GET /api/analytics/report`

## 三、技术选型

### 1. 前端

- \*\*移动端\*\*：微信小程序开发框架（WXML、WXSS、JavaScript）。

- \*\*Web端\*\*：Vue.js或React.js框架，结合HTML5、CSS3。

### 2. 后端

- \*\*开发语言\*\*：Node.js或Python。

- \*\*框架\*\*：Express.js（Node.js）或Flask（Python）。

- \*\*数据库\*\*：MySQL（关系型数据库），Redis（缓存数据库）。

### 3. 接口

- \*\*API设计\*\*：RESTful API，支持JSON格式数据交互。

### 4. 安全

- \*\*加密技术\*\*：SHA-256用于数据传输加密，AES用于数据存储加密。

- \*\*身份验证\*\*：JWT（JSON Web Tokens）用于用户身份验证。

## 四、数据流程

1. \*\*用户登录\*\*：

- 用户通过客户端输入账号和密码。

- 系统后端验证用户信息，生成JWT并返回给客户端。

- 客户端存储JWT，用于后续请求的身份验证。

2. \*\*文本防伪验证\*\*：

- 用户通过客户端输入或扫描文本信息。

- 客户端将文本信息发送到后端的`/api/verify/text`接口。

- 后端解密文本信息，与数据库中的防伪信息比对，返回验证结果到客户端。

- 客户端展示验证结果给用户。

3. \*\*商品信息查询\*\*：

- 用户通过客户端请求商品信息。

- 客户端调用后端的`/api/product/{productId}`接口。

- 后端查询数据库，返回商品详细信息到客户端。

- 客户端展示商品信息给用户。

4. \*\*数据统计与分析\*\*：

- 客户端将用户行为数据（如验证次数、查询时间等）发送到后端的`/api/analytics/behavior`接口。

- 后端收集并存储用户行为数据。

- 企业用户可以通过Web端后台查询统计报告，调用`/api/analytics/report`接口获取数据分析结果。

## 五、系统特点

1. \*\*高效性\*\*：采用先进的OCR技术和加密算法，确保文本识别和验证的高效性。

2. \*\*安全性\*\*：数据加密存储和传输，采用JWT进行身份验证，确保用户数据和防伪信息的安全。

3. 易用性：简洁的用户界面设计，支持多种设备，降低用户操作难度。

4. 可扩展性：分层架构设计，支持未来功能模块的扩展和技术升级。

5. 兼容性：支持多种操作系统和设备，确保用户在不同平台上都能使用。

---

以上是文本防伪软件的总体设计概述，涵盖了系统架构、模块划分、技术选型和数据流程等方面。如果你有更具体的需求或需要进一步细化某些部分，请随时告诉我！

1. **功能需求**
2. **性能指标**
3. **接口需求**
4. **其他要求**

一、性能需求

1. 识别效率

速度

1. 识别准确率

对中文的准确率

1. 用户体验需求
2. 界面设计

简洁明了，操作流程简单易懂，符合用户的使用习惯

1. 交互设计

提供清晰的用户反馈，如扫描成功、验证通过等信息需及时、准确地展示给用户