1. 性能测试

1. 首先，确保您的项目服务已经启动：

```bash

# 在项目根目录下

make

```

2. 启动性能测试环境：

```bash

cd tests/performance

docker-compose up -d

```

3. 等待所有容器启动完成，可以通过以下命令查看容器状态：

```bash

docker-compose ps

```

4. 运行性能测试：

```bash

# 进入JMeter容器

docker exec -it goby\_jmeter\_1 bash

# 在容器内执行测试

jmeter -n -t /jmeter/performance\_test.jmx -l /results/results.jtl -e -o /results/report

如果需要修改测试参数，可以编辑 `jmeter/performance\_test.jmx` 文件

登录请求成功：

{

  "email": "22009201315@stu.xidian.edu.cn",

  "password": "12345678",

  "is\_seller": **true**

}

图形用户界面, 应用程序, 表格, Word

描述已自动生成

**性能测试结果概览**

* **APDEX（应用性能指数）**：
  + **说明**：APDEX 值为 1.000，表示所有请求的响应时间都在容忍阈值（500ms）内，没有用户感到沮丧（1秒500ms）。这表明系统性能非常好，用户体验流畅。
  + **意义**：APDEX 是衡量用户体验的指标，值越接近 1 表示用户体验越好。
* **请求摘要**：
  + **说明**：饼图显示所有请求均通过（100%），没有失败请求。
  + **意义**：请求通过率是衡量系统稳定性和可靠性的重要指标。
* **统计数据**：
  + **说明**：总请求数为 5000，没有失败请求，错误率为 0%。平均响应时间为 6.93ms，最小响应时间为 3ms，最大响应时间为 70ms。中位数响应时间为 6ms，90%、95% 和 99% 的响应时间分别为 10ms、11ms 和 16ms。
  + **意义**：这些统计数据提供了请求性能的详细概览，表明系统在不同百分位数下的响应时间表现。
* **网络吞吐量**：
  + **说明**：每秒事务数为 491.21，接收的数据量为 257.12 KB，发送的数据量为 127.60 KB。
  + **意义**：这些数据表明系统的网络吞吐量和数据传输效率。

图表, 散点图

描述已自动生成

**Latency Vs Request（延迟与请求数）**

* **说明**：该图表展示了不同请求数下的中位延迟时间。横轴表示每秒的全局请求数，纵轴表示中位延迟时间（毫秒）。从图中可以看出，随着请求数的增加，中位延迟时间也逐渐增加。在请求数较少时，延迟时间较低，随着请求数的增加，延迟时间显著增加，表明系统在高负载情况下的性能下降。
* **意义**：了解请求数与延迟时间的关系有助于评估系统在不同负载下的延迟表现，识别出系统的处理能力。

图表

描述已自动生成

**Response Time Percentiles Over Time（随时间变化的响应时间百分位数）**

* **说明**：该图表展示了随时间变化的响应时间百分位数。横轴表示时间，纵轴表示响应时间（毫秒）。从图中可以看出，不同百分位数的响应时间在测试过程中有显著变化。在某个时间点（10:21:11），响应时间达到峰值，表明系统在该时间段内的性能较差。
* **意义**：了解响应时间随时间的变化有助于识别系统中的瞬时性能瓶颈和异常情况，帮助进行性能优化和故障排查。

图表, 散点图

描述已自动生成

**Response Time Vs Request（响应时间与请求数）**

* **说明**：该图表展示了不同请求数下的中位响应时间。横轴表示每秒的全局请求数，纵轴表示中位响应时间（毫秒）。从图中可以看出，随着请求数的增加，中位响应时间也逐渐增加。在请求数较少时，响应时间较低，随着请求数的增加，响应时间显著增加，表明系统在高负载情况下的性能下降。
* **意义**：了解请求数与响应时间的关系有助于评估系统在不同负载下的性能表现，识别出系统的处理能力。

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

**Response Time Overview（响应时间概览）**

* **说明**：该图表展示了不同响应时间范围内的请求数量。横轴表示响应时间范围，纵轴表示请求数量。从图中可以看出，大部分请求（约5000个）的响应时间在500毫秒以内，而响应时间超过500毫秒的请求数量非常少，几乎没有请求的响应时间超过1500毫秒。

图表, 折线图

描述已自动生成

**Response Time Percentiles（响应时间百分位数）**

* **说明**：该图表展示了不同百分位数下的响应时间分布。横轴表示百分位数，纵轴表示响应时间（毫秒）。从图中可以看出，大部分请求的响应时间集中在较低的百分位数范围内，随着百分位数的增加，响应时间逐渐增加。在90%以上的百分位数时，响应时间显著增加，达到约70毫秒。这表明虽然大部分请求的响应时间较短，但仍有少量请求的响应时间较长。

图表, 折线图

描述已自动生成

**Time Vs Threads（时间与线程数）**

* **说明**：该图表展示了不同线程数下的平均响应时间。横轴表示活跃线程数，纵轴表示平均响应时间（毫秒）。从图中可以看出，随着线程数的增加，平均响应时间也逐渐增加。在线程数较少时，响应时间较低，随着线程数的增加，响应时间显著增加，表明系统在高并发情况下的性能下降。
* **意义**：了解线程数与响应时间的关系有助于评估系统在不同负载下的性能表现，识别出系统的并发处理能力。

登录请求失败：

{

  "email": "22009201315@stu.xidian.edu.cn",

  "password": "1234567",

  "is\_seller": **true**

}

图形用户界面, 应用程序, Word

描述已自动生成

**性能测试结果概览（错误情况）**

* **错误信息**：
  + **说明**：所有请求均返回 401 Unauthorized 错误，错误数量为 5000，占所有请求的 100%。
  + **意义**：401 Unauthorized 错误通常表示身份验证失败，可能是由于认证信息不正确或缺失导致的。需要检查认证机制和用户凭证。

文本, 应用程序

描述已自动生成

**响应时间概览（错误情况）**

* **说明**：柱状图显示在测试过程中出现了大量错误请求（约5000个），没有请求的响应时间在500ms以内，也没有请求的响应时间在500ms到1500ms之间，响应时间超过1500ms的请求数量为0。
* **意义**：该图表表明系统在测试过程中出现了严重问题，导致几乎所有请求都失败，需要进一步调查和解决。

1. 安全性测试

1. 启动安全测试环境：

```bash

cd tests/security

docker-compose up -d

```

2. 运行不同类型的扫描：

a. 基础扫描：

```bash

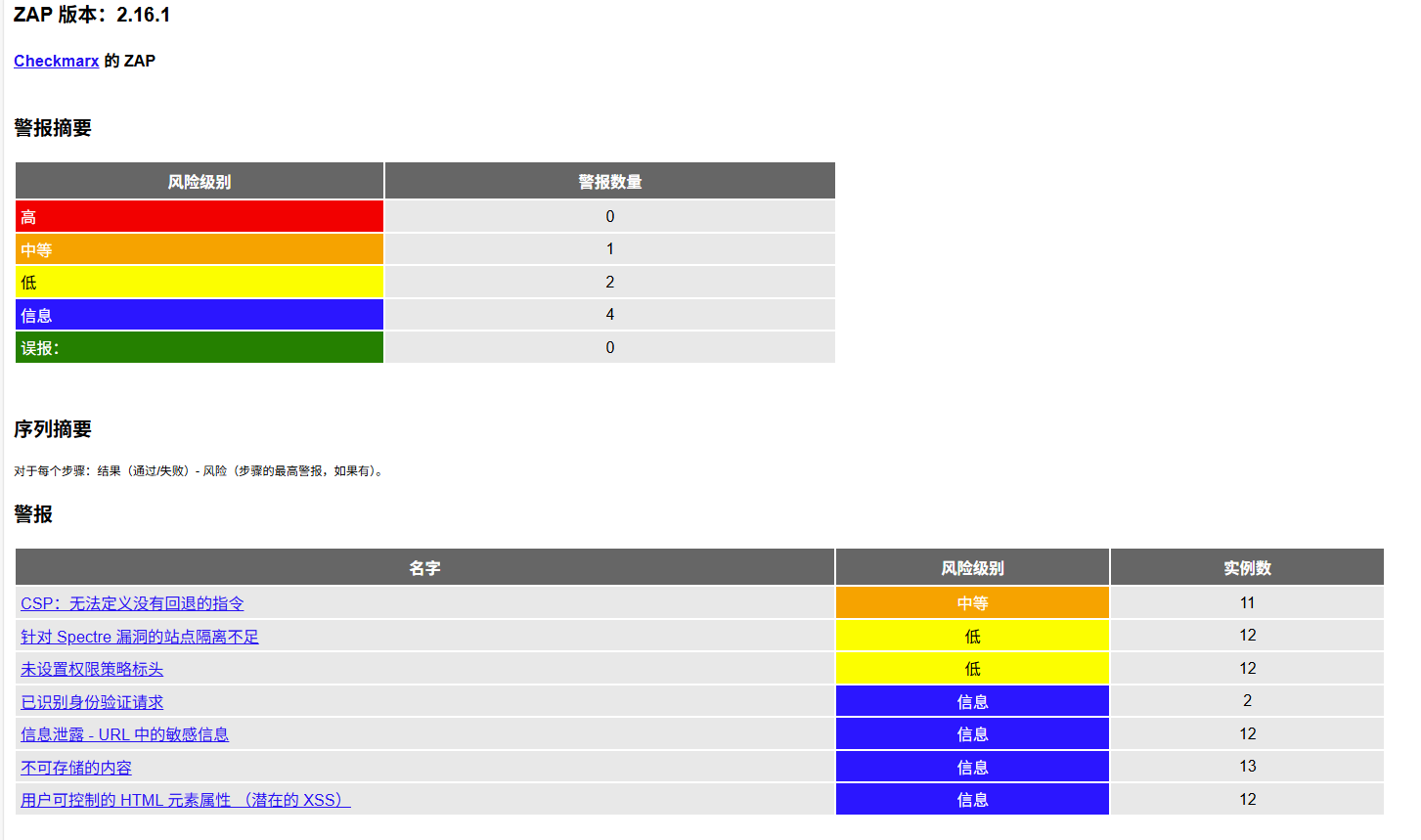
docker-compose run zap-baseline

```

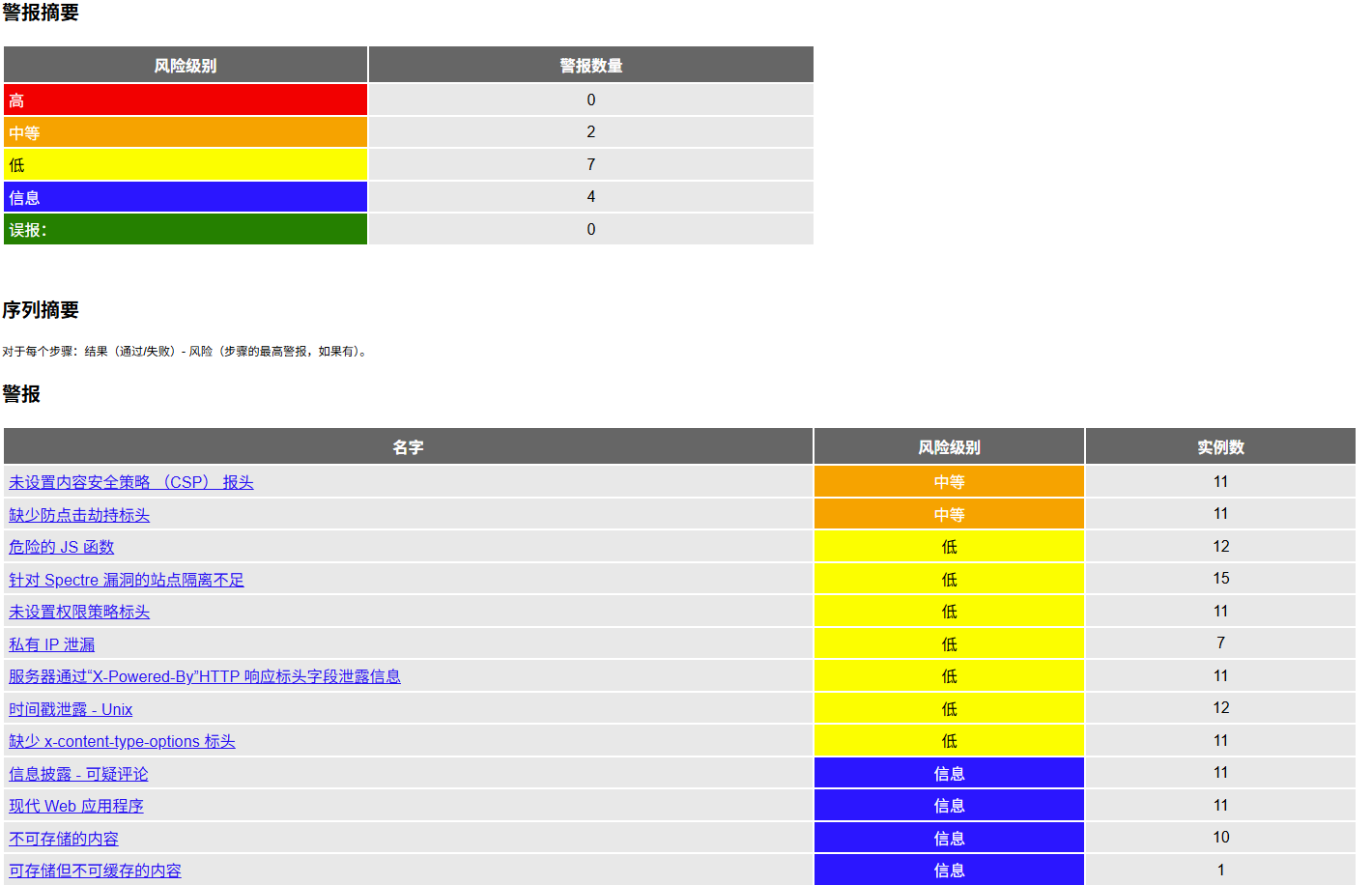
b. API扫描：

```bash

docker-compose run zap-api-scan



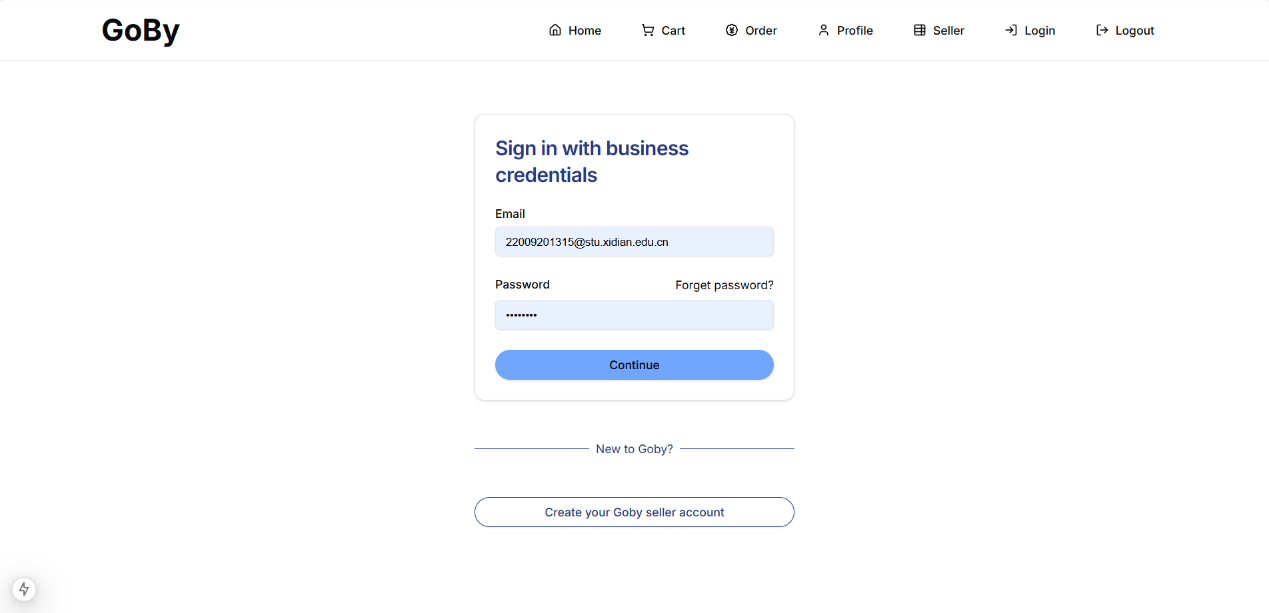
**Zap基线扫描（zap-baseline）报告概览：**快速扫描，旨在识别最常见的安全漏洞。这种扫描方式通常用于初步评估应用程序的安全性，生成的报告（testreport）包含一些基本的发现和建议。



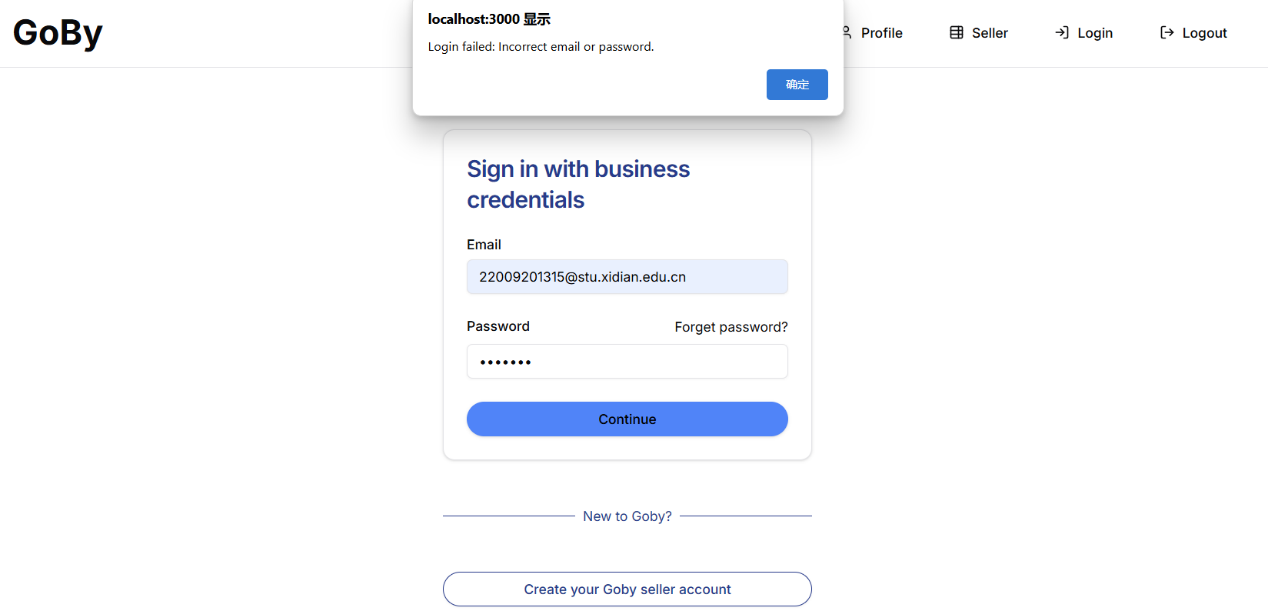
**Zap全面扫描（zap-base）报告概览：**深入扫描，识别更广泛的安全问题，包括不太常见的漏洞。通常会包含更详细的信息和更多的漏洞发现。



1. Web UI测试



**登录UI**

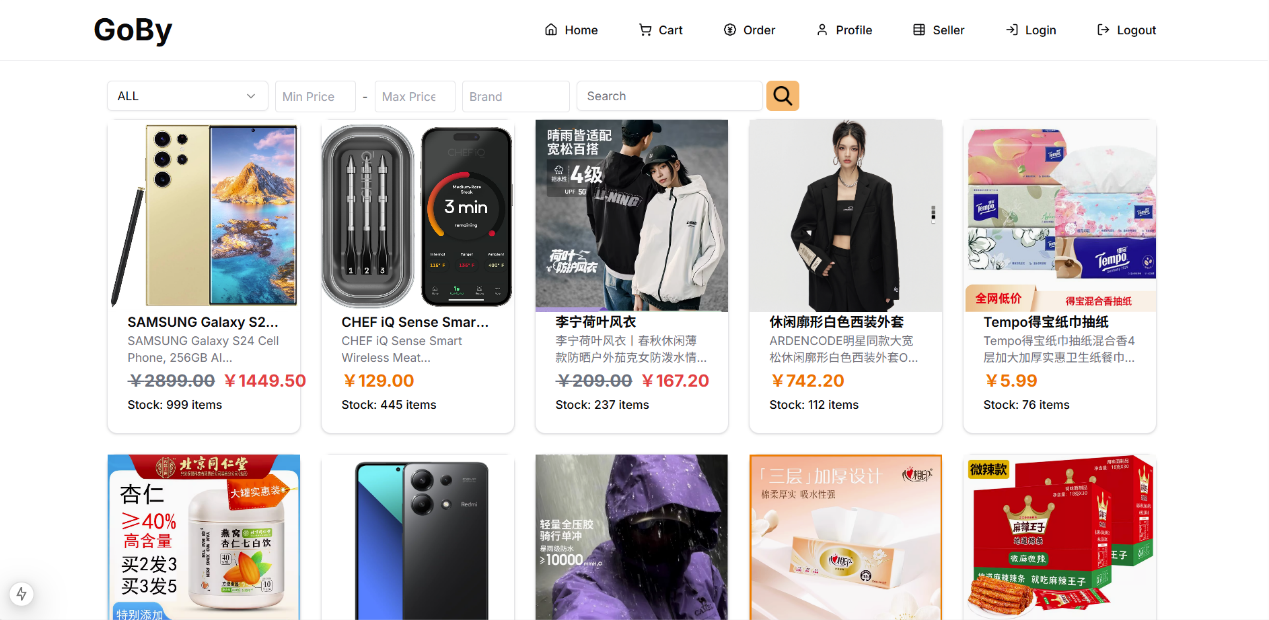
****

**密码错误弹出提示**

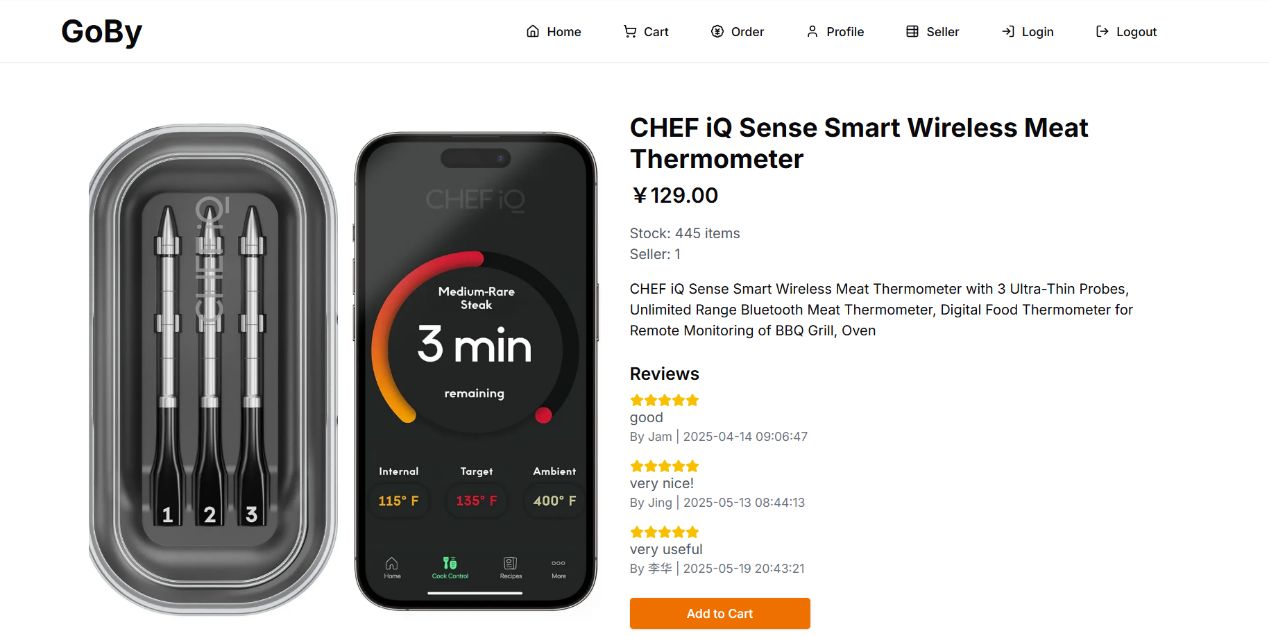
**图形用户界面, 应用程序

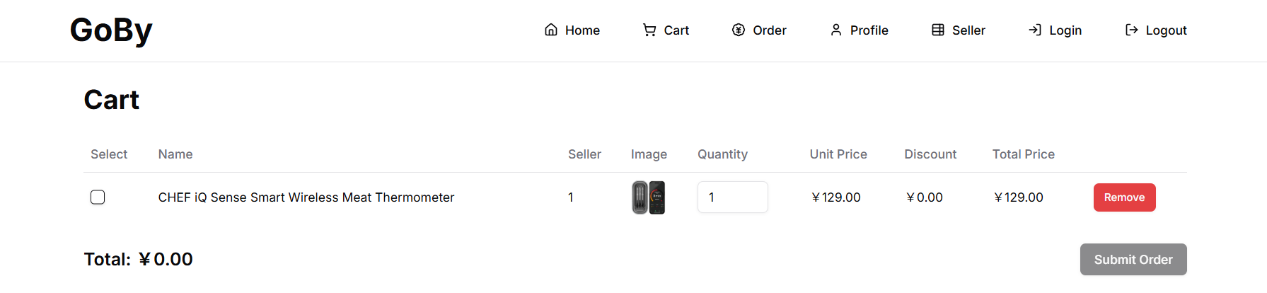
描述已自动生成**

**创建新用户**

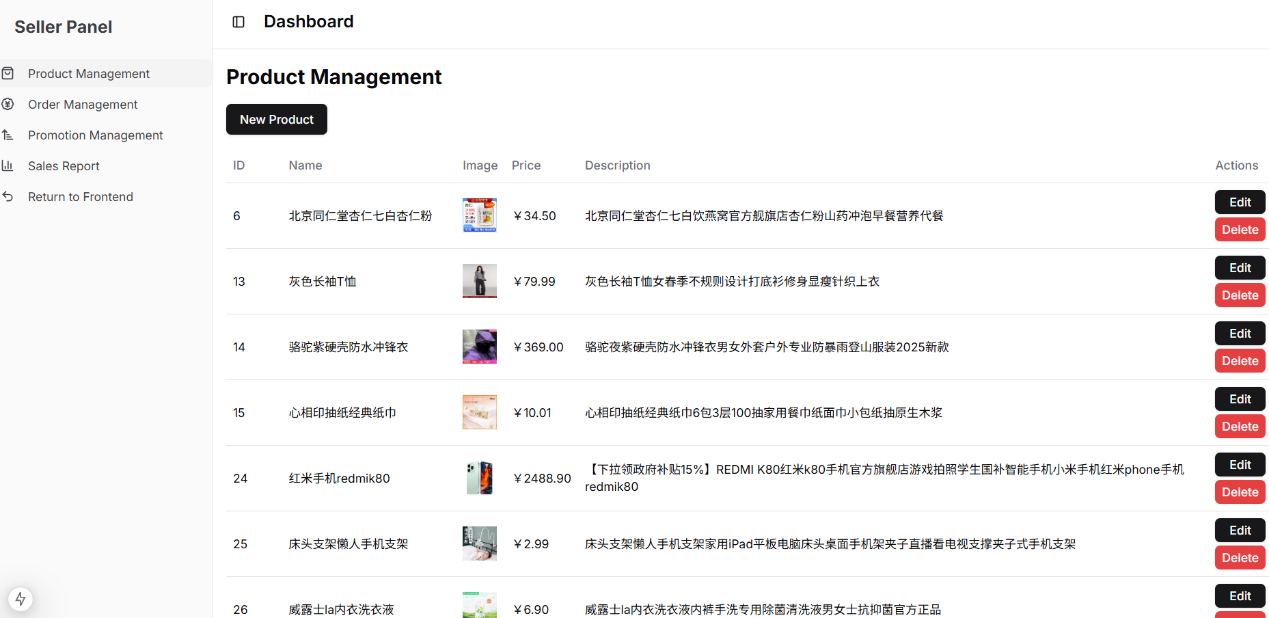


**购物大厅UI**



**商品UI**

**购物车UI**



**商品上架UI**