**哈尔滨工业大学计算学部**

**2025年春季学期《软件工程》**

**Lab 2：项目计划与原型设计**

|  |  |  |
| --- | --- | --- |
| **姓名** | **学号** | **联系方式** |
| 宫名扬 | 2022113557 | 13950003961 |
| 常添 | 2022111699 | 13936408592 |
| 方恒杰 | 2022113568 | 19859626220 |

**目 录**

[1 实验要求 1](#_Toc199087135)

[2 项目概述 1](#_Toc199087136)

[3 用户故事 2](#_Toc199087137)

[3.1 用户故事清单及优先级 2](#_Toc199087138)

[3.2 用户故事1: 库存查询 3](#_Toc199087139)

[3.3 用户故事2: 入库管理 3](#_Toc199087140)

[3.4 用户故事3: 出库管理 4](#_Toc199087141)

[3.5 用户故事4: 销售开单 4](#_Toc199087142)

[3.6 用户故事5: 采购开单 4](#_Toc199087143)

[3.7 用户故事6: 员工账号管理 5](#_Toc199087144)

[3.8 用户故事7: 销售报表查看 5](#_Toc199087145)

[3.9 用户故事8: 采购报表查看 5](#_Toc199087146)

[3.10 用户故事9: 客户管理 6](#_Toc199087147)

[3.11 用户故事10: 供应商管理 6](#_Toc199087148)

[3.12 用户故事11: 数据看板 6](#_Toc199087149)

[4 用户故事的工作量估算 7](#_Toc199087150)

[5 迭代计划 7](#_Toc199087151)

[6 使用CodeArts或其他工具管理用户故事和迭代计划 8](#_Toc199087152)

[6.1 建立团队、项目、成员 8](#_Toc199087153)

[6.2 Product Backlog（全部用户故事） 8](#_Toc199087154)

[6.3 Sprint Planning（规划迭代） 9](#_Toc199087155)

[6.4 形成每个Sprint的Story Board 10](#_Toc199087156)

[6.5 形成每个Sprint的Burndown Chart 11](#_Toc199087157)

[7 使用大模型辅助生成用户故事和迭代计划 11](#_Toc199087158)

[8 原型设计 12](#_Toc199087159)

[8.1 客户管理（大模型生成的原型） 12](#_Toc199087160)

[8.2 用户故事1: 库存查询 13](#_Toc199087161)

[8.3 用户故事2: 入库管理 14](#_Toc199087162)

[8.4 用户故事3: 出库管理 14](#_Toc199087163)

[8.5 用户故事4: 销售开单 14](#_Toc199087164)

[8.6 用户故事5: 采购开单 15](#_Toc199087165)

[9 计划与实际进度 15](#_Toc199087166)

[10 小结 15](#_Toc199087167)

[文档全部完成之后，请更新上述区域]

# 实验要求

根据实验指导书中的要求完成的实验目标与过程。

实验目标：

- 根据项目需求建立用户故事清单，使用敏捷开发方法为用户故事建模卡片，设计优先级，估计工作量，构造迭代计划。

- 练习使用 CodeArts 或其他适当的 Scrum 项目管理工具为项目建立代计划。

- 练习使用大模型设计软件界面。

- 练习使用 MockPlus 或其他合适的原型设计工具为每个用户故事设计软件原型。

实验步骤：

科学小组数完成的 Project，完成以下任务：

- 整理讨论问题，提取所有用户故事清单；

- 为每个用户故事制作卡片；

- 分析用户故事的优先级；

- 组内成员共同表决和按重要程度，估算每用户故事的工作量；

- 整理以上结果，设计项目的迭代开发计划；

- 使用 CodeArts 或其他自选的 Scrum 项目管理工具建立和管理代计划；

- 选取一个用户故事，使用大模型进行界面设计；

- 使用 MockPlus 或其他合适的原型设计工具为选择的用户故事进行原型设计（GUI）。

# 项目概述

我们小组开发的是一个面向中小型企业的ERP仓储管理系统，旨在提供完整的企业资源管理解决方案，帮助企业实现业务流程自动化和数据可视化。该系统采用B/S架构，前端使用Vue框架开发，后端基于Python的Django框架实现，数据库采用MySQL。系统主要功能模块包括库存管理（支持库存查询、入库管理、出库管理、库存预警）、采购管理（采购计划、采购开单、采购退货、供应商管理）、销售管理（销售开单、销售退货、客户管理）、基础数据管理（产品信息、产品分类、计量单位）以及系统管理等模块，通过这些功能模块的有机结合，为企业提供从采购、入库、销售到出库的完整业务流程管理，帮助企业提高运营效率、降低库存成本并优化资源配置。

系统名称：ERP仓储管理系统

团队成员：宫名扬、常添、方恒杰

# 用户故事

## 用户故事清单及优先级

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 用户故事编号 | 用户故事简称 | 用户故事描述 | 优先级估算（采用5、4、3、2、1的方式，数字越大表示优先级越高） | 上游  故事编号 |
|  | 库存查询 | 作为仓库管理员，我希望能够查询当前所有产品的库存状态，以便及时了解库存情况 | 5 | - |
|  | 入库管理 | 作为仓库管理员，我希望能够记录产品入库信息，以便准确跟踪库存增加情况 | 5 | 5 |
|  | 出库管理 | 作为仓库管理员，我希望能够记录产品出库信息，以便准确跟踪库存减少情况 | 5 | 4 |
|  | 销售开单 | 作为销售人员，我希望能够创建销售订单，以便记录客户购买的产品和数量 | 4 | - |
|  | 采购开单 | 作为采购人员，我希望能够创建采购订单，以便记录从供应商购买的产品和数量 | 4 | - |
|  | 员工账号管理 | 作为系统管理员，我希望能够管理员工账号，以便控制系统访问权限 | 3 | - |
|  | 销售报表查看 | 作为数据分析师，我希望能够查看销售报表，以便分析销售趋势和制定销售策略 | 2 | 4 |
|  | 采购报表查看 | 作为数据分析师，我希望能够查看采购报表，以便分析采购趋势和优化采购策略 | 2 | 5 |
|  | 客户管理 | 作为客户关系经理，我希望能够管理客户信息，以便维护客户关系 | 2 | - |
|  | 供应商管理 | 作为采购经理，我希望能够管理供应商信息，以便维护供应商关系 | 2 | - |
|  | 数据看板 | 作为系统用户，我希望能够在登录后看到重要数据的可视化展示，以便快速了解业务状况 | 1 | 1, 7, 8 |

## 用户故事1: 库存查询

使用卡片图形的形式，描述每个用户故事。

* 正面：文字描述（作为一个<角色>, 我想要<活动>, 以便于<商业价值>）、简要的示意图；
* 文本

  AI 生成的内容可能不正确。反面：确认信息（该用户故事执行成功后的结果、各种执行失败情况后的结果）

## 日程表 AI 生成的内容可能不正确。图形用户界面, 应用程序 AI 生成的内容可能不正确。用户故事2: 入库管理

## 文本 AI 生成的内容可能不正确。图形用户界面, 文本, 应用程序 AI 生成的内容可能不正确。用户故事3: 出库管理

## 图片包含 文本 AI 生成的内容可能不正确。用户故事4: 销售开单

## 文本 AI 生成的内容可能不正确。用户故事5: 采购开单

## 文本 AI 生成的内容可能不正确。用户故事6: 员工账号管理

## 文本 AI 生成的内容可能不正确。用户故事7: 销售报表查看

## 文本 AI 生成的内容可能不正确。用户故事8: 采购报表查看

## 图片包含 文本 AI 生成的内容可能不正确。 用户故事9: 客户管理

## 用户故事10: 供应商管理

## 用户故事11: 数据看板

# 用户故事的工作量估算

针对识别出的每一个故事，使用Story Point估算其工作量，工作量的单位是天。

使用预定的值：1/2、1、2、3、5、8、13、20，单位为“小时”；

团队成员分别估计，差异较大时面对面讨论，发现分歧，形成共识。

填写下列表格（表格里给出了三轮，若第一轮就达成共识或者估算差异不大，就不需要进入第二轮，依此类推；最后一列是大家最终达成的共识）。

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 故事编号 | 故事简称 | 小组成员对其工作量估算 | | | | | | | | | 最终估算 |
| 第一轮 | | | 第二轮 | | | 第三轮 | | |
|  | 库存查询 | 5 | 7 | 5 | 5 | 5 | 5 | - | - | - | 5 |
|  | 入库管理 | 8 | 8 | 10 | 8 | 8 | 8 | - | - | - | 8 |
|  | 出库管理 | 8 | 6 | 8 | 8 | 8 | 8 | - | - | - | 8 |
|  | 销售开单 | 8 | 8 | 6 | 8 | 8 | 8 | - | - | - | 8 |
|  | 采购开单 | 8 | 8 | 8 | 8 | 8 | 8 | - | - | - | 8 |
|  | 员工账号管理 | 5 | 4 | 5 | 5 | 5 | 5 | - | - | - | 5 |
|  | 销售报表查看 | 8 | 8 | 7 | 8 | 8 | 8 | - | - | - | 8 |
|  | 采购报表查看 | 8 | 10 | 8 | 8 | 8 | 8 | - | - | - | 8 |
|  | 客户管理 | 5 | 6 | 5 | 5 | 5 | 5 | - | - | - | 5 |
|  | 供应商管理 | 5 | 5 | 6 | 5 | 5 | 5 | - | - | - | 5 |
|  | 数据看板 | 13 | 11 | 13 | 13 | 13 | 13 | - | - | - | 13 |

# 迭代计划

若本项目采用两次迭代，根据各用户故事的优先级和工作量估算，将用户故事分配到各次迭代当中，计算各次迭代的总工作量。确保这样的安排符合第3节给出的依赖关系和优先级安排，以及各次迭代的总工作量的平衡。

请根据需要增加下表中的行数，但不能增加迭代次数。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 迭代次数 | 包含的用户故事 | 故事的优先级 | 故事的工作量估计 | 计划起止时间 | 本次迭代的  总工作量 |
| 1 | 库存查询 | 5 | 5 | 5.7-5.10 | 42小时 |
| 入库管理 | 5 | 8 | 5.11-5.14 |
| 出库管理 | 5 | 8 | 5.15-5.19 |
| 销售开单 | 4 | 8 | 5.20-5.22 |
| 采购开单 | 4 | 8 | 5.23-5.26 |
| 员工账号管理 | 3 | 5 | 5.27-5.28 |
| 2 | 销售报表查看 | 2 | 8 | 5.29-6.2 | 39小时 |
| 采购报表查看 | 2 | 8 | 6.3-6.7 |
| 客户管理 | 2 | 5 | 6.8-6.10 |
| 供应商管理 | 2 | 5 | 6.11-6.13 |
| 数据看板 | 1 | 13 | 6.14-6.20 |

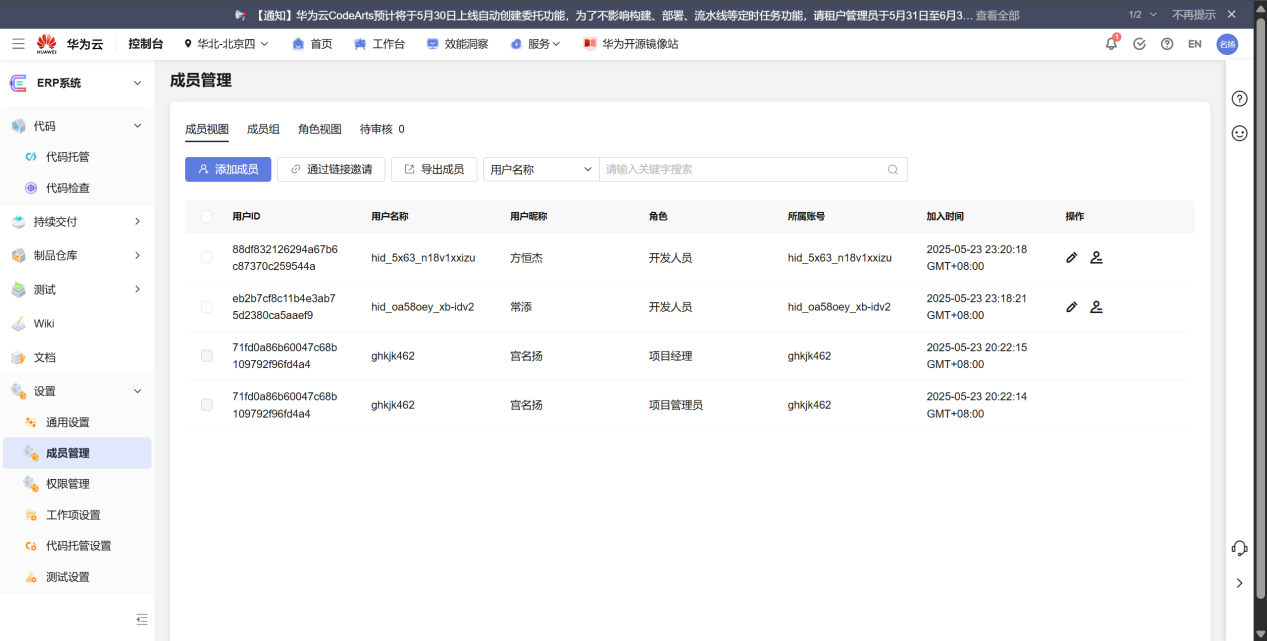
# 使用CodeArts或其他工具管理用户故事和迭代计划

根据第3、4、5各部分的内容，使用CodeArts或其他Scrum项目管理工具建立你们的项目管理计划，将结果以截图的形式放在此处。

我们选择使用CodeArts，过程如下：

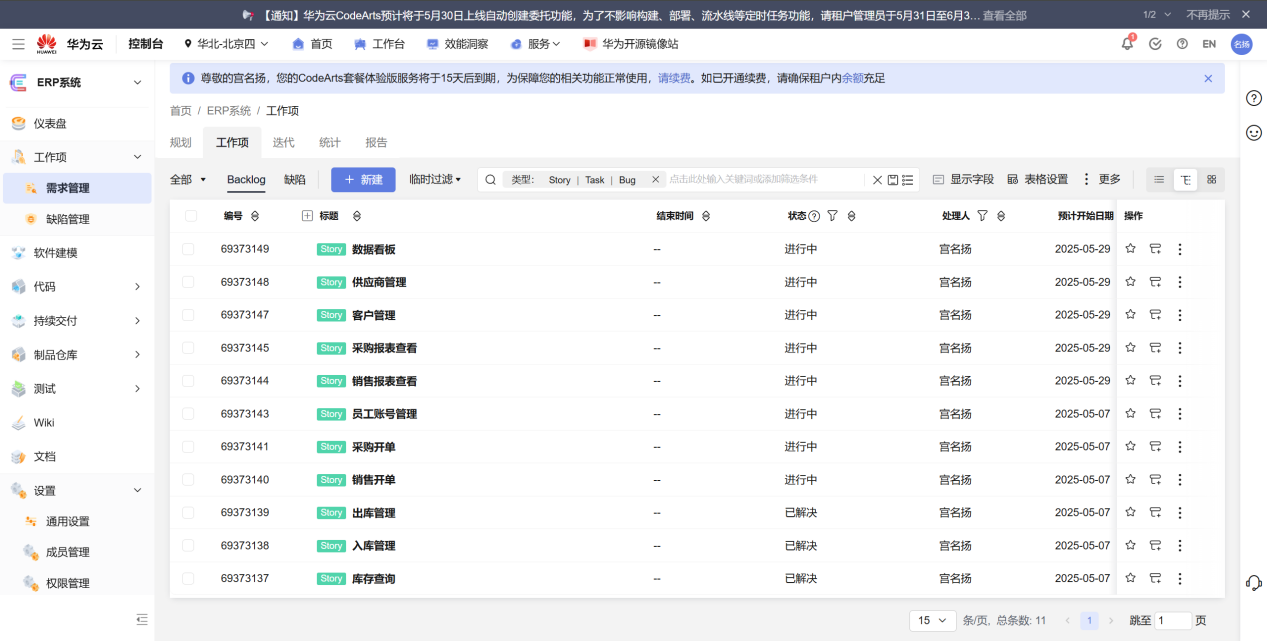
## 建立团队、项目、成员

在成员管理中添加成员、设置成员角色：



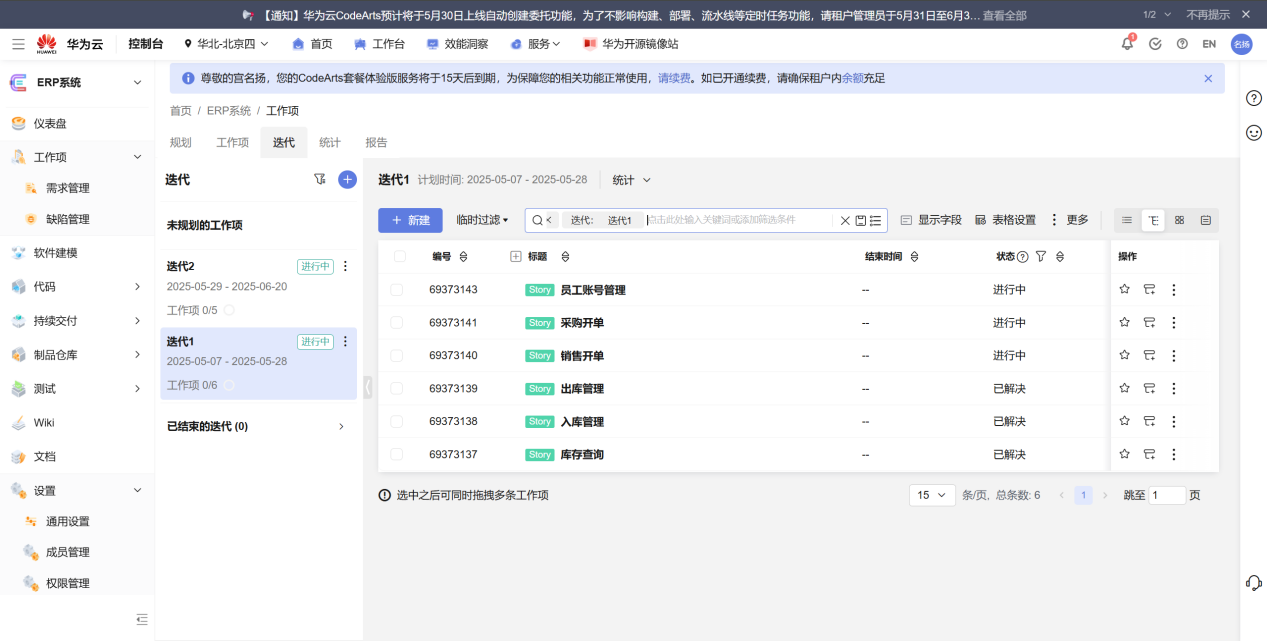
## Product Backlog（全部用户故事）

建立上述用户故事：

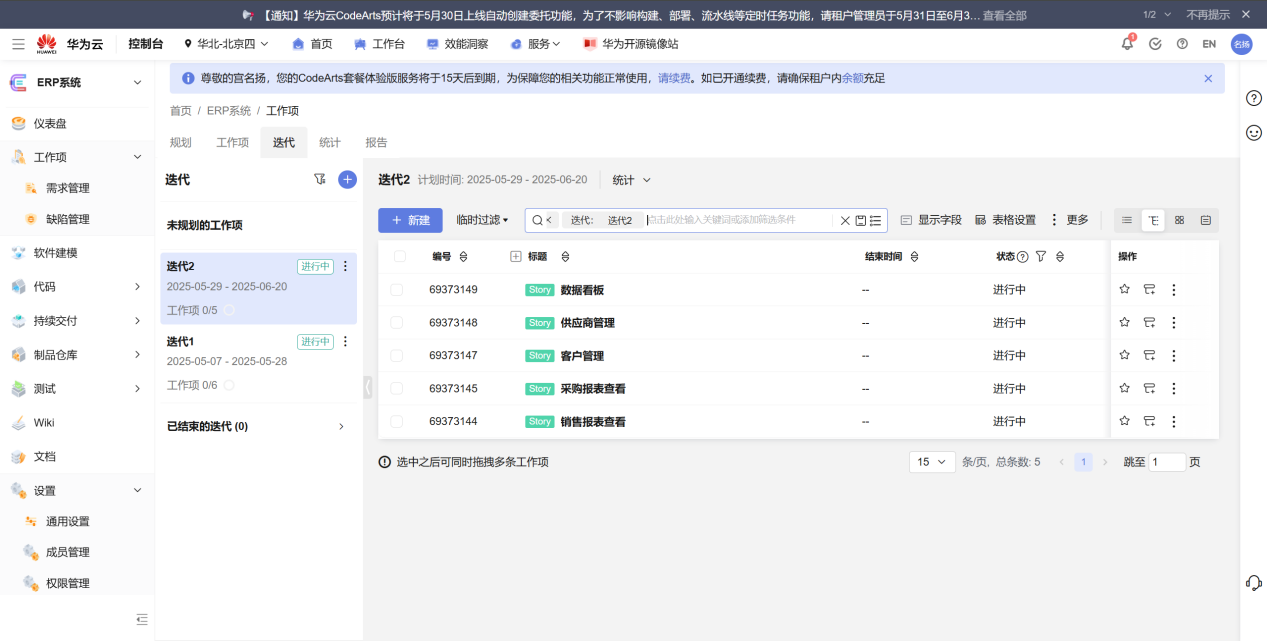


## Sprint Planning（规划迭代）

第一轮迭代：

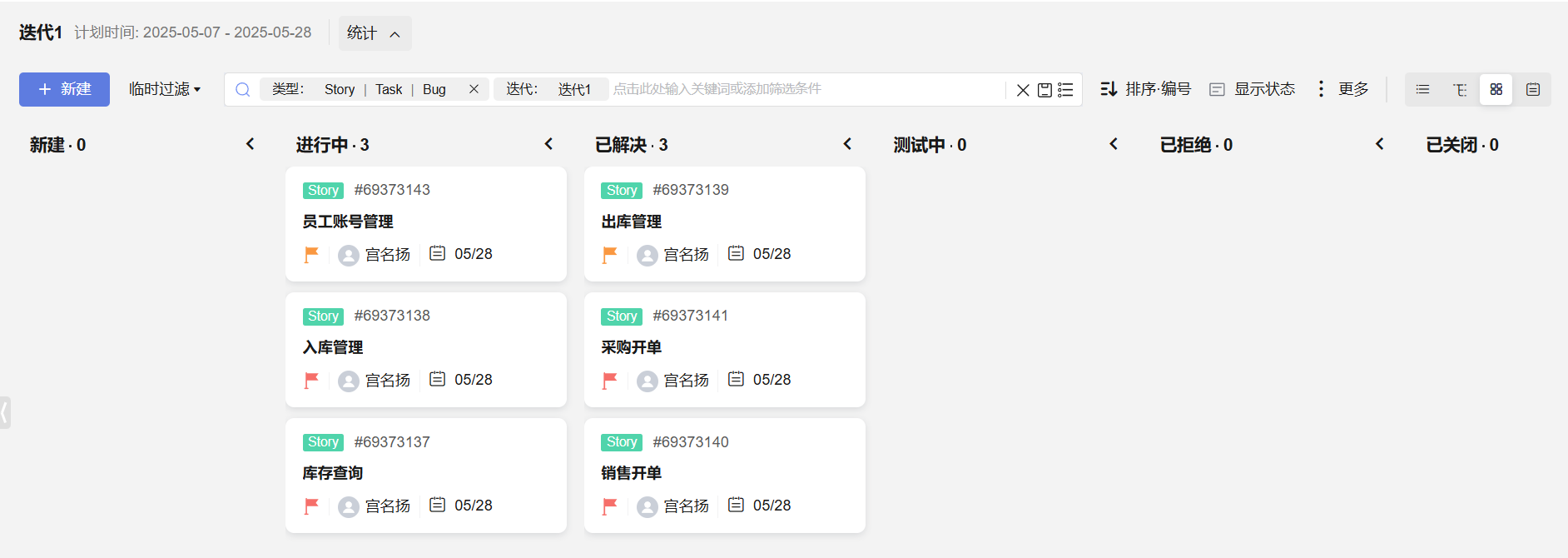


第二轮迭代：

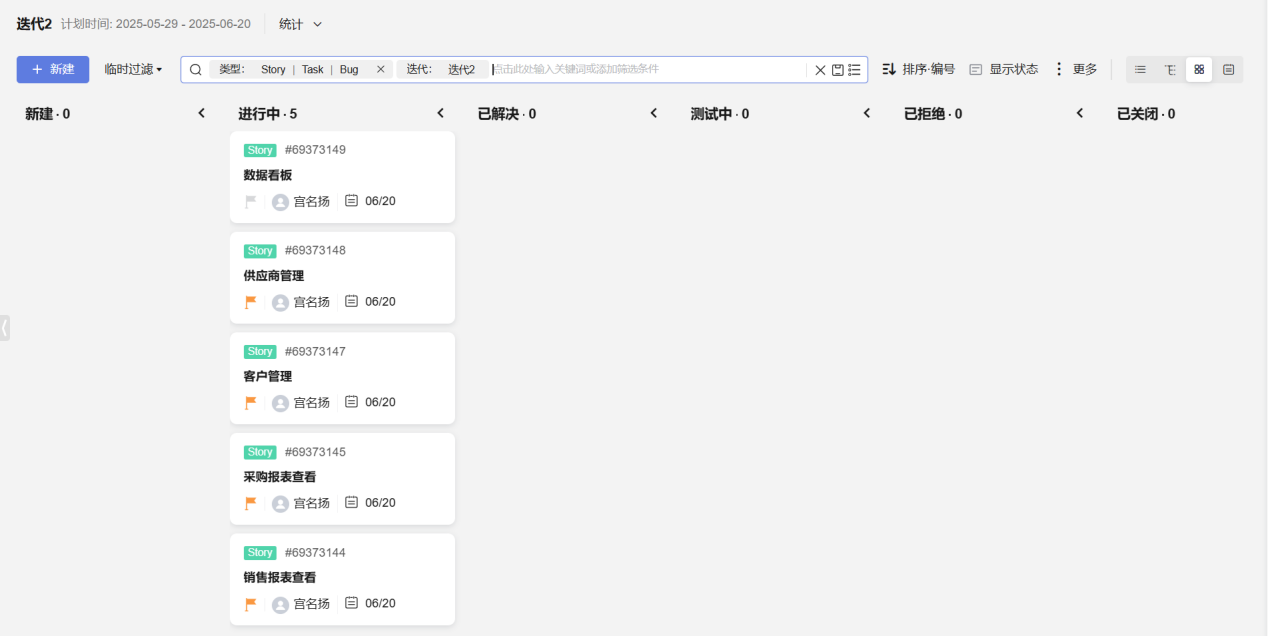


## 形成每个Sprint的Story Board

第一轮迭代卡片模式：



第二轮迭代卡片模式：



## 形成每个Sprint的Burndown Chart

第一轮迭代燃尽图：



第二轮迭代燃尽图：



# 使用大模型辅助生成用户故事和迭代计划

如果采用了大模型辅助生成用户故事和迭代计划，此处给出对应的提示词。注：此部分不是必选项，可以自行选择是否采用大模型辅助，评判以结果为依据，不以是否采用大模型为依据。

**7.1 批量生成用户故事的提示词**

请扮演一名敏捷需求分析师。根据下列“业务模块关键词”产出 6-10 条符合 INVEST 原则的用户故事，并执行自检确保每条故事可独立、可估算、具备业务价值。  
输出格式必须是 **JSON 数组**；数组中每个对象包含字段：  
• no (从 1 开始的流水号)  
• title  
• roles  
• activity  
• value  
• success (成功验收条件，1-3 句)  
• failure (失败场景列举，2-4 句)  
业务模块关键词：  
【库存管理，采购报表，客户管理，供应商管理，销售报表，员工账号管理】

– 要求模型输出的 JSON 可直接导入需求系统  
– 在提示里强调“INVEST 自检”以提高故事质量  
– 关键词列表可按实际需要增删，模型需逐条覆盖

**7.2 两轮迭代计划生成的提示词**

请依据下面提供的用户故事 JSON（已评审通过）制定 **2 轮 Sprint** 的迭代计划：  
• 每个 Sprint 周期为 40 小时左右  
• 先为每条故事进行时间估算 Story Point（SP）  
• 为 Sprint-1 与 Sprint-2 分别列出：  
– Sprint 目标（一句话聚焦价值）  
– 纳入的故事清单（含 SP 值与合计）  
– 每条故事需分解 4-6 个主要任务  
– 为 Sprint 给出“完成度度量”说明（Definition of Done、关键 KPI 或演示标准）

# 原型设计

针对第3节识别出的用户故事，选取1个用户故事使用大模型生成原型；

针对第3节识别出的每个用户故事，利用MockPlus或其他原型设计工具建立其原型，将原型截图放在以下各小节里。

此处应至少包含5个用户故事的原型设计，且均为优先级最高的用户故事，不能包含登录、用户管理等故事。

## 客户管理（大模型生成的原型）

| **要素** | **内容** |
| --- | --- |
| **用户故事** | **US-09 客户管理** |
| **使用的大模型** | ChatGPT (o3 reasoning model) |
| **生成方式** | 以**结构化提示词**描述需求 → 模型一次性输出 JSON 需求 + Python 原型脚本；脚本运行后自动绘制低保真 UI 并导出 PDF / PNG |

**1. 提示词（Prompt）**

你是前端产品原型助手。

请根据以下用户故事生成

① 一份 JSON 需求描述（字段：no、title、role、activity、value、success、failure），

② 一段可直接运行的 Python 脚本，使用 matplotlib 生成双页 PDF：

左页为“用户故事卡片（正面）”，右页为“用户故事卡片（反面）”。

正面需在卡片底部绘制低保真 UI：顶栏写“客户信息管理”，

包含“客户编码 / 客户名称 / 联系电话”输入框、保存按钮，以及表头“编码 名称 电话 状态”+ 一行示例数据。

脚本要求：

- 全程中文注释；

- 自动嵌入中文字体或提示放置 NotoSansSC；

- 保存为 US09\_prototype.pdf；

- 生成成功后打印 ✅ 提示。

【用户故事】

作为一个客户关系经理，我想要管理客户信息，以便于维护客户关系，提供个性化服务。

成功：系统成功创建、修改或删除客户信息，更新客户联系方式和交易记录。\

失败：1) 客户编码已存在… 2) 联系电话格式无效… 3) 删除有交易记录的客户… 4) 系统故障…

**2. 生成结果截图：**



## 用户故事1: 库存查询



## 用户故事2: 入库管理

图形用户界面, 应用程序

AI 生成的内容可能不正确。

## 用户故事3: 出库管理

图形用户界面, 应用程序

AI 生成的内容可能不正确。

## 用户故事4: 销售开单

电脑软件截图

AI 生成的内容可能不正确。

## 用户故事5: 采购开单

图形用户界面, 应用程序

AI 生成的内容可能不正确。

# 计划与实际进度

|  |  |  |  |
| --- | --- | --- | --- |
| 任务名称 | 计划时间长度（分钟） | 实际耗费时间（分钟） | 提前或延期的原因分析 |
| 库存查询 | 300 | 240 | 团队成员对数据库查询有较丰富经验，提前完成 |
| 入库管理 | 480 | 600 | 表单验证逻辑复杂，处理边缘情况耗时 |
| 出库管理 | 480 | 540 | 库存校验规则调整，增加了开发工作量 |
| 销售开单 | 480 | 480 | 符合预期，按时完成 |
| 采购开单 | 480 | 480 | 符合预期，按时完成 |
| 员工账号管理 | 300 | 240 | 账号管理功能简单且相对熟悉 |

# 小结

利用大模型生成用户故事和项目计划，同小组讨论规划在效果和效率方面的对比；

利用大模型生成原型，同利用工具完成，在效果和效率方面的对比；

其他方面的建议和体会。

1. 用户故事与迭代计划  
    ● **效率**：借助大模型（一次性生成＋两次微调）产出 10 条 INVEST 合格的用户故事仅耗时约 15 分钟；传统做法中，小组头脑风暴-记录-润色-互评完整闭环至少需要 2-3 小时。  
    ● **效果**：  
     – 模型能够快速给出结构化字段、统一措辞，并自动补足成功/失败场景，减少了格式不一致和遗漏验收准则的问题。  
     – 然而，模型对企业特定规则、数据约束的洞察不足，需业务成员补充落地细节（例如特殊税率、采购审批链）。
2. 低保真原型  
    ● **效率**：调用大模型一次生成「JSON + Matplotlib 脚本」并运行导出 PDF，整体约 10 分钟；使用 Axure/Figma 重新画同等信息量的页面，设计熟手也要 30-40 分钟。  
    ● **效果**：  
     – 脚本输出的矢量 UI 可快速迭代、批量复用，文本修改即可刷新全部页面；参数化优势明显。  
     – 视觉层次与交互细节不及专业原型工具丰富，例如组件库、动态效果、拖拽交互等需后期补绘。
3. 其他建议与体会  
    ① **Prompt 明确化**：效果高度依赖提示词粒度。将角色、输出字段、格式样例、验收衡量标准写得越具体，返工越少。  
    ② **“人+机”混合流程**：让模型负责“起草＋排版”，团队侧重“业务校验＋风险评估”，双方长处互补。  
    ③ **自动化脚本资产化**：把生成 UI 的 Python 模板沉淀为仓库工具，后续新增故事只改 JSON 即可批量导出，持续放大一次投入的收益。  
    ④ **质量门槛控制**：为避免模型幻觉或规则偏差，建立轻量评审清单（字段完整性、业务合法性、安全合规）作为交付前必过环节。

总体来看，大模型在需求-设计早期阶段显著提升速度和格式一致性，是“小步快跑、快速对齐”的利器；而在深度业务洞察、视觉完备度方面仍需专业成员与工具共同完善，形成高效闭环。