Background：《三国杀》作为一款经典的策略卡牌游戏，凭借其独特的机制和丰富的角色技能，广受欢迎。然而，由于苟卡策划导致新出的卡牌在越来越阴间的同时也越来越贵，令热爱桌游学生群体不堪重负但又自甘堕落。同时，科大的一直有名为Killer的都市传说，这些Killer能在每年选课时都让学生谈之色变。



因此，我们团队结合这二者，发起了《科大杀》项目。

Problem:

1,Unity开发环境和工具的利用

2.用户需求的收集

3.对战逻辑的自主设计与调优（针对不同功能卡牌）

4.核心区代码、数据库和AI接口的集成

### ****3. Functional Requirements****

**（1）核心游戏机制**

**1V1 卡牌对战模式**：严格遵循《三国杀》规则，提供出牌、选牌、弃牌等操作。

图片

**AI对战系统**：通过强化学习训练AI，使其具备与真人对战相似的策略能力。

图片

**游戏记录与日志**：玩家可以回顾对局数据和分析策略。

**（2）用户界面与体验**

**简洁易用的UI**：降低上手难度，适合非游戏专业的师生。

图片

**视觉与音效设计**：符合校园主题，带有本校特色和幽默元素。

图片

**（3）数据存储与管理**

4. Non-functional Requirements

（1）性能要求

快速响应：玩家操作后，系统在1秒内响应，AI决策在4秒内完成。

低资源消耗：在校园机房设备上流畅运行，不要求高性能硬件。

（2）可靠性与可维护性

稳定性：游戏在长时间运行后不会崩溃或卡顿。

更新与维护：定期修复已知BUG，优化AI策略。

（3）可移植性

平台限制：初期仅支持PC端，不开发移动版。

操作系统兼容性：在Windows和MacOS上运行。

USER-CASE Diagram

Chapter 2: Software Design

software process model：敏捷开发

design methodology：

1. 组件分解法（Decomposition of Problem and Process）：

游戏核心机制： 使用Unity开发游戏对局逻辑。

AI对战模块： 使用PyTorch训练AI决策模型。

数据管理模块： 使用MySQL进行玩家数据和对局记录管理。

音画设计模块： 使用Photoshop和音频编辑工具进行UI和音频素材制作。

1. 面向对象编程C#

### ****——封装（Encapsulation）****

在《科大杀》中，封装用于将**卡牌、角色、游戏逻辑**进行模块化管理。

——继承（Inheritance）卡牌和角色具有**通用特性**和**特定行为**，通过继承来实现。

——多态（Polymorphism）不同卡牌具有**相同操作接口**，但表现不同效果。

4.抽象（Abstraction）：抽象类和接口用于定义**玩家、卡牌、技能**等核心抽象概念。

CLASS-DIAGRAM

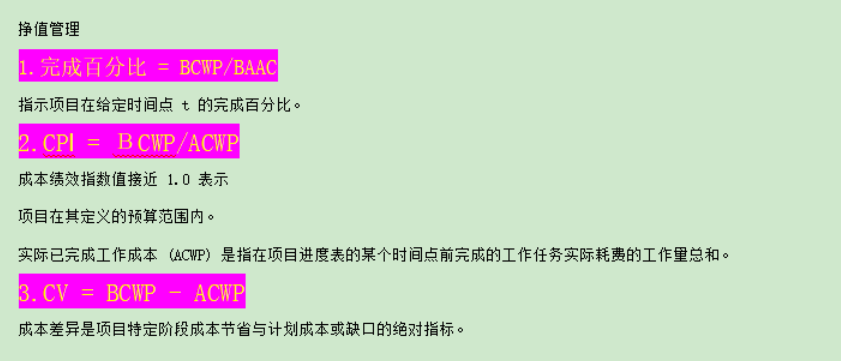
Chapter3:Agile development

领歌

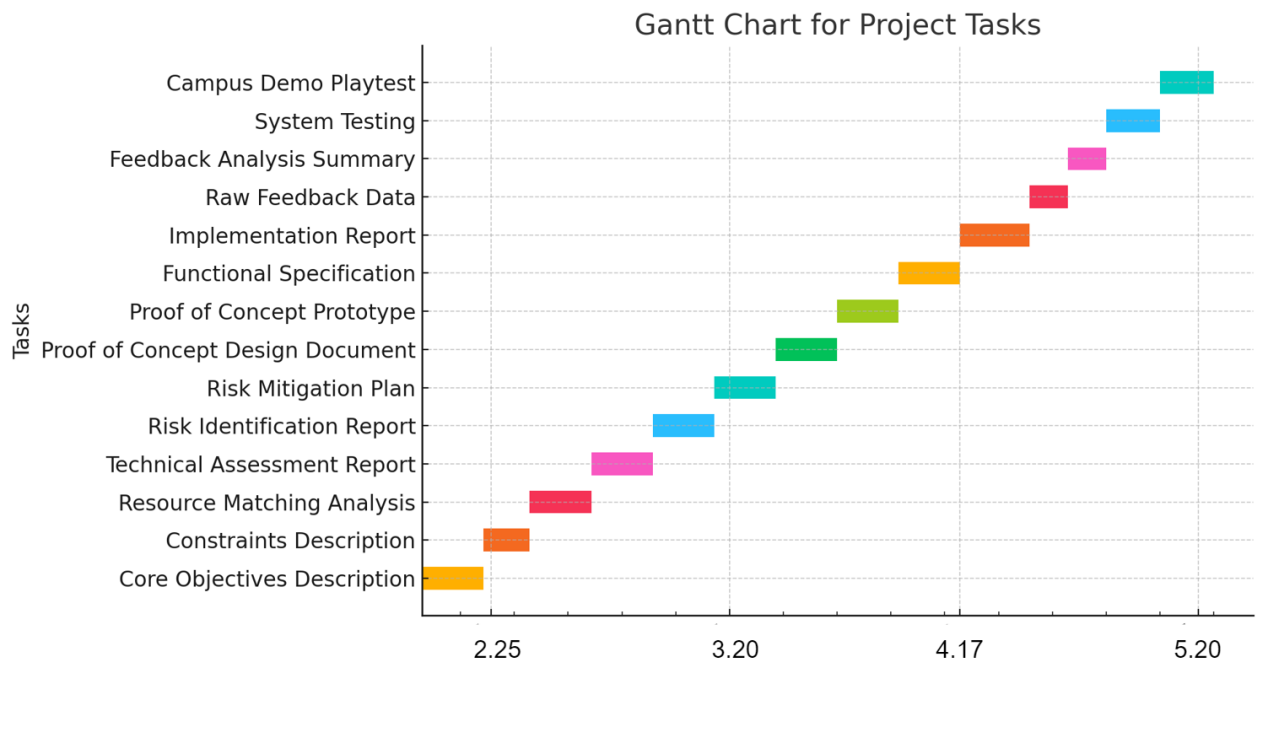
Chapter 4: Budget and Schedule

建议和领歌杂一起：

假设当前Sprint切片节点的budget，简单列出这几个指标



假设BAC（budget at completion） = 40,000 元

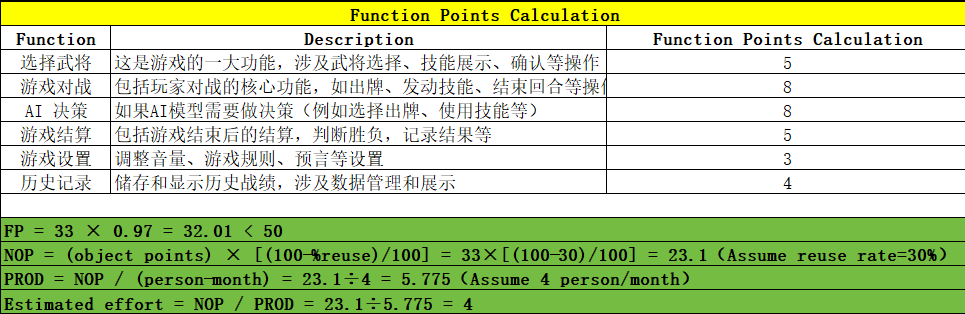
****



最后单独PPT展示

Software Metrics and Equations

Chapter5：FP=33×0.97=32.0



基于这个表得出esitimated effort！

### Use-case oriented Metrics

测试用例10个/per测试用例 共100个

KLOC=3.75

使用bailey-basili：教育/学术环境的小团队开发项目

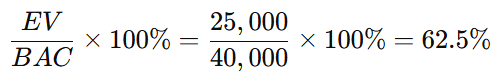


E=8.88 person months

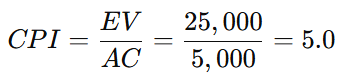
Chapter6：Tracking and Control

EV（挣得价值） = 25,000 元

AC（实际成本） = 5,000 元



项目已经完成了62.5%



成本效率非常高，说明花费较少。



节省了2万元

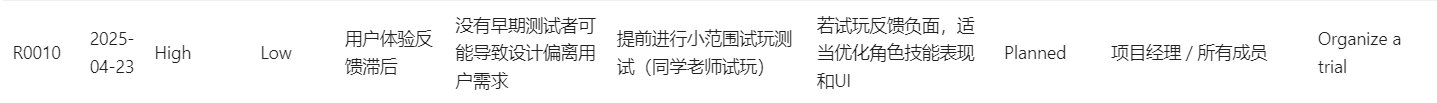
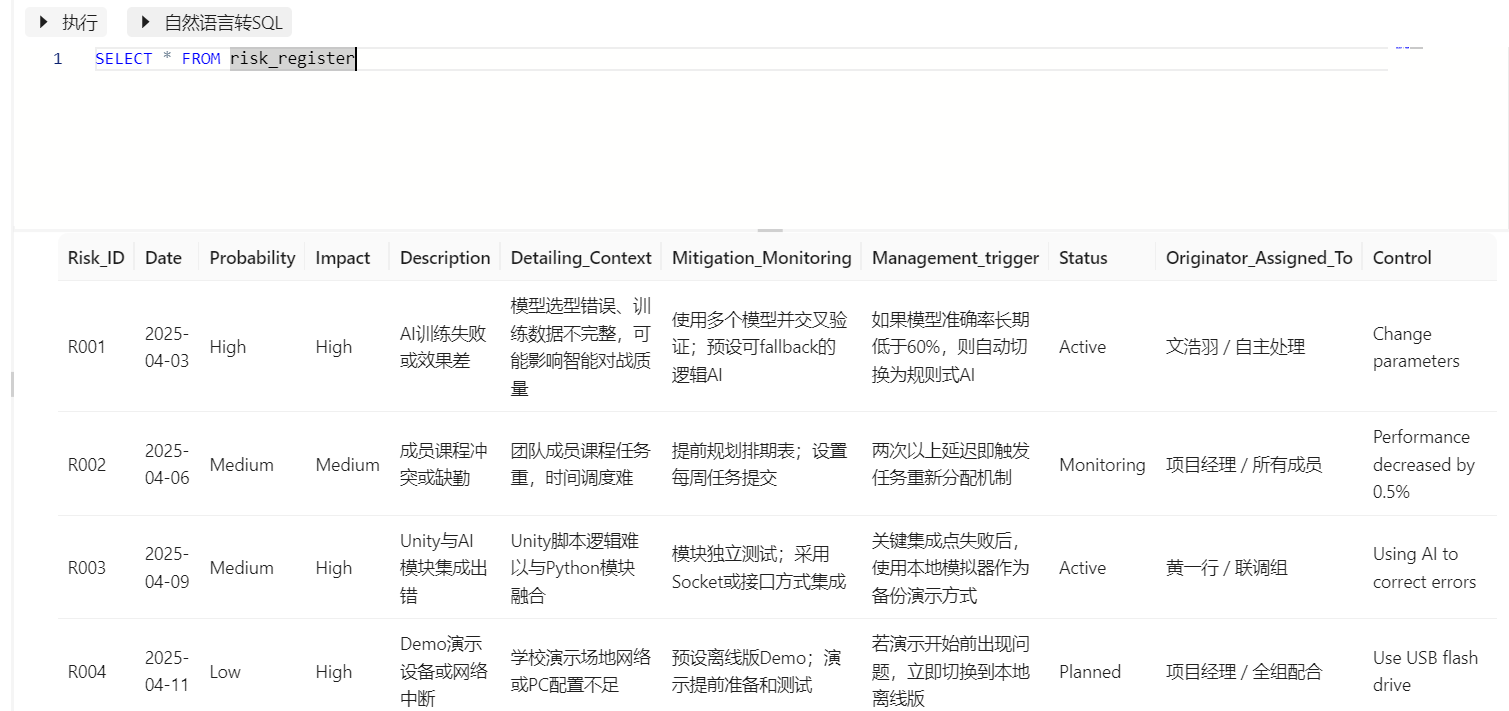
放在领歌各个sprint写一个

UPDATE risk\_register

SET Control = 'AI accuracy>80%'

WHERE Risk\_ID = R1;

Chapter6\_control\_rist management —— Chapter 7: Risk Management



UPDATE risk\_register

SET Control = 'AI accuracy>80%'

LIMIT 1;

UPDATE risk\_register

SET Control = 'Performance decreased by 0.5%'

LIMIT 2;

UPDATE risk\_register

SET Control = 'Using AI to correct errors'

LIMIT 3;

UPDATE risk\_register

SET Control = 'Change parameters'

LIMIT 1;

UPDATE risk\_register

SET Control = 'Use USB flash drive'

LIMIT 4;

UPDATE risk\_register

SET Control = 'Organize documents regularly'

LIMIT 5;

UPDATE risk\_register

SET Control = 'PM assigns responsibilities'

LIMIT 6;

UPDATE risk\_register

SET Control = 'Upload by a single person'

LIMIT 7;

UPDATE risk\_register

SET Control = 'Change tools'

LIMIT 8;

UPDATE risk\_register

SET Control = 'unified art style'

LIMIT 9;

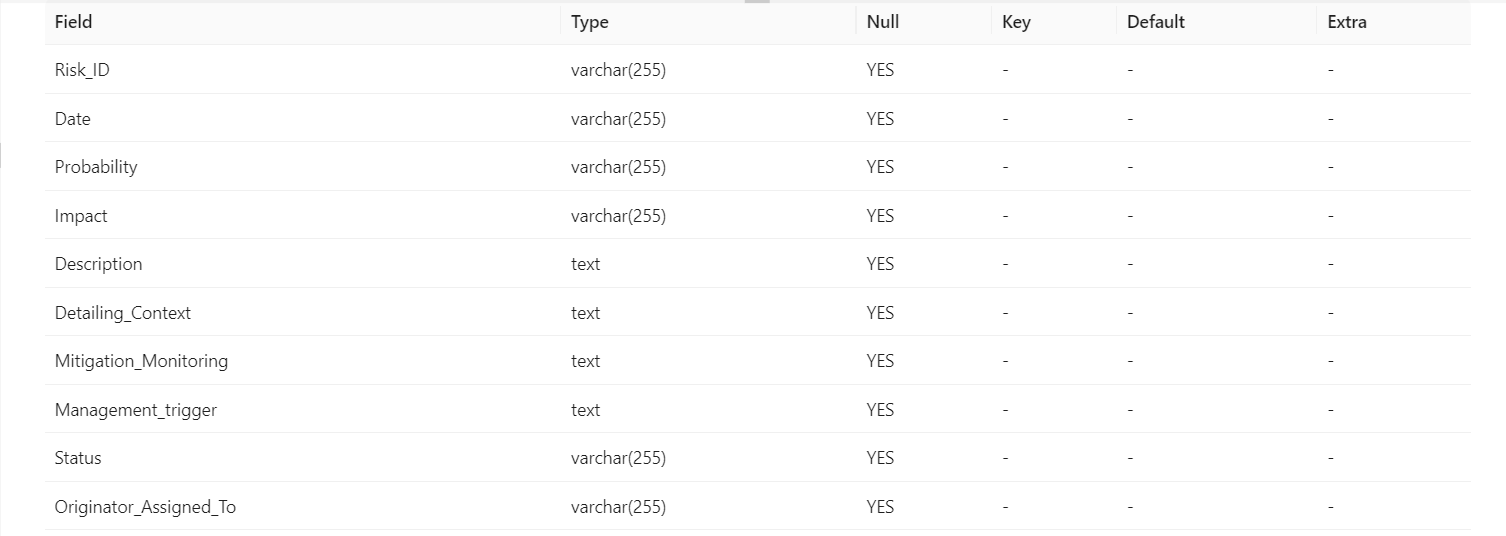
UPDATE risk\_register

SET Control = 'Organize a trial'

LIMIT 10;

INSERT INTO risk\_register (Control) VALUES ('Member performance decreased by 0.5%');





Chapter 8: Software Quality Assurance

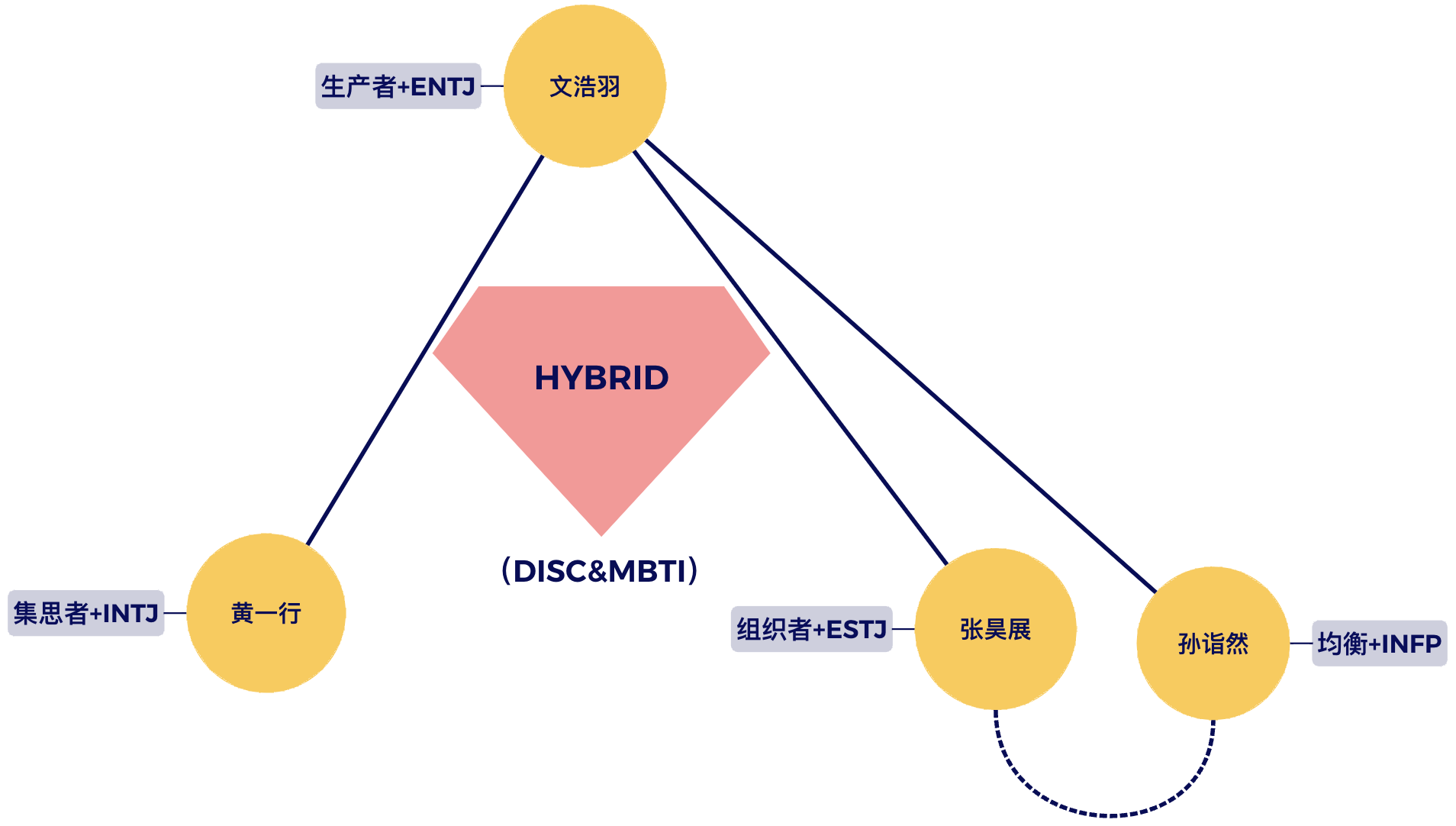
Chapter 9: Software Configuration Management

Baseline 图

Repository

版本控制

Chapter 10: Software Team Organization and Teams (including team structure, contribution-based evaluation, personality test, etc.)



contribution-based evaluation

Chapter 11: Course Reflection, Practices, and Comments

Course Reflection, Practices, and Comments

发表感想？成果和评论