需求分析：这是项目的起始阶段，主要目标是收集和分析用户的需求，并生成需求规格说明书。

设计：在需求明确后，进入设计阶段。设计师会制定系统架构，确定系统各部分的交互方式，并生成设计文档。

编码：在设计阶段完成后，开发人员根据设计文档开始编写代码。

测试：编码完成后，测试团队会对系统进行测试，包括单元测试、集成测试和系统测试等，以确保软件的质量。

维护：在软件发布后，可能还需要进行维护，包括修复错误、更新功能和优化性能等。

单一职责原则（Single Responsibility Principle, SRP）：

一个类应该只有一个引起变化的原因。

意味着一个类应该只有一个职责，当这个职责发生变化时，只需要修改这个类。

开放封闭原则（Open-Closed Principle, OCP）：

软件实体（类、模块、函数等）应该是可扩展的，但是不可修改的。

当需求变化时，通过扩展现有代码而不是修改现有代码来满足新的需求。

里氏替换原则（Liskov Substitution Principle, LSP）：

子类型必须能够替换掉它们的父类型。

也就是说，程序中的对象应该能够使用其父类型来替换，而不会产生错误或异常。

接口隔离原则（Interface Segregation Principle, ISP）：

使用多个专门的接口，而不使用单一的总接口，客户端不应该依赖它不需要的接口。

这意味着应该细化接口，避免客户端依赖不需要的接口方法。

依赖倒置原则（Dependency Inversion Principle, DIP）：

要依赖于抽象，不要依赖于具体。

高层模块不应该依赖于低层模块，它们都应该依赖于抽象；抽象不应该依赖于细节，细节应该依赖于抽象。

迪米特法则（Law of Demeter, LoD）：

一个软件实体应当尽可能少地与其他实体发生相互作用。

也就是说，一个对象应该对其他对象保持最少的了解，降低类之间的耦合度。

组合优于继承原则（Favor Composition over Inheritance）：

尽量使用对象组合，而不是通过继承来达到复用的目的。

组合通常使得系统更加灵活，因为它允许在运行时改变对象的行为，而继承则是一种静态关系，在编译时就确定了。

**软件设计文档**

**文档信息**

* **文档名称**：
* **版本号**：
* **编写人**：
* **审核人**：
* **完成日期**：
* **修订记录**：

**1. 引言**

简短介绍文档的目的、范围以及相关背景信息。包括项目概述、目标用户、预期解决的问题等。

用于编写一个德州扑克的后台代码。效果是实现：（1）通过配置创建不同的扑克牌数，(2)通过配置创建不同数量的玩家，（3）保存每个玩家的游戏结果到数据库，（4）提供作弊方法

**2. 总体设计**

* **系统架构**：描述软件的整体架构，包括但不限于分层结构、微服务架构、客户端-服务器模型等。

不涉及

* **模块划分**：列出主要的软件模块及其功能，并说明它们之间的关系和交互方式。

配置管理：提供配置读写接口。

数据库管理：初始化数据库；提供数据库读写接口。

游戏主进程：提供德州扑克游戏主题程序，附带作弊功能。核心进程，读取配置，读写数据库，

* **技术选型**：列出所采用的关键技术和工具，如编程语言、数据库、框架等。

**使用C++面向对象编码，使用mysql数据库，不涉及架构设计。**

**3. 数据设计**

* **数据结构**：定义数据实体、属性及它们之间的关系。
* **数据库设计**：包括数据库模式（ER图）、表结构、主键、外键等设计。
* **数据流图**：展示数据如何在系统各部分间流动。

**数据结构：**

配置的数据结构：

struct BaseConf{

uint32\_t poker\_num;

unit32\_t player\_num;

bool god\_enable;

}

扑克牌数据结构：

enum class Suit { //花色

HEARTS, DIAMONDS, CLUBS, SPADES

};

enum class Rank { //牌面数字

TWO, THREE, FOUR, FIVE, SIX, SEVEN, EIGHT, NINE, TEN,

JACK, QUEEN, KING, ACE

};

**4. 接口设计**

* **外部接口**：定义与外部系统交互的接口规范，包括API协议、数据格式等。
* **内部接口**：描述模块间的接口设计，包括调用方式、参数列表、返回值等。

**5. 功能设计**

针对每个主要功能模块，详细说明其设计思路、算法选择、流程图等。

**6. 安全性设计**

* **数据安全**：数据加密、访问控制等措施。
* **系统安全**：防止攻击、备份恢复策略等。

**7. 性能设计**

* **性能指标**：响应时间、吞吐量、资源消耗等。
* **优化策略**：缓存机制、负载均衡、并发处理等方案。

**8. 用户界面设计**

（如果适用）描述用户界面的布局、交互设计原则、原型图或草图。

**9. 测试计划**

概述测试策略、测试用例设计、预期测试结果等。

**10. 附录**

包括参考文献、术语表、详细设计图等附加信息。