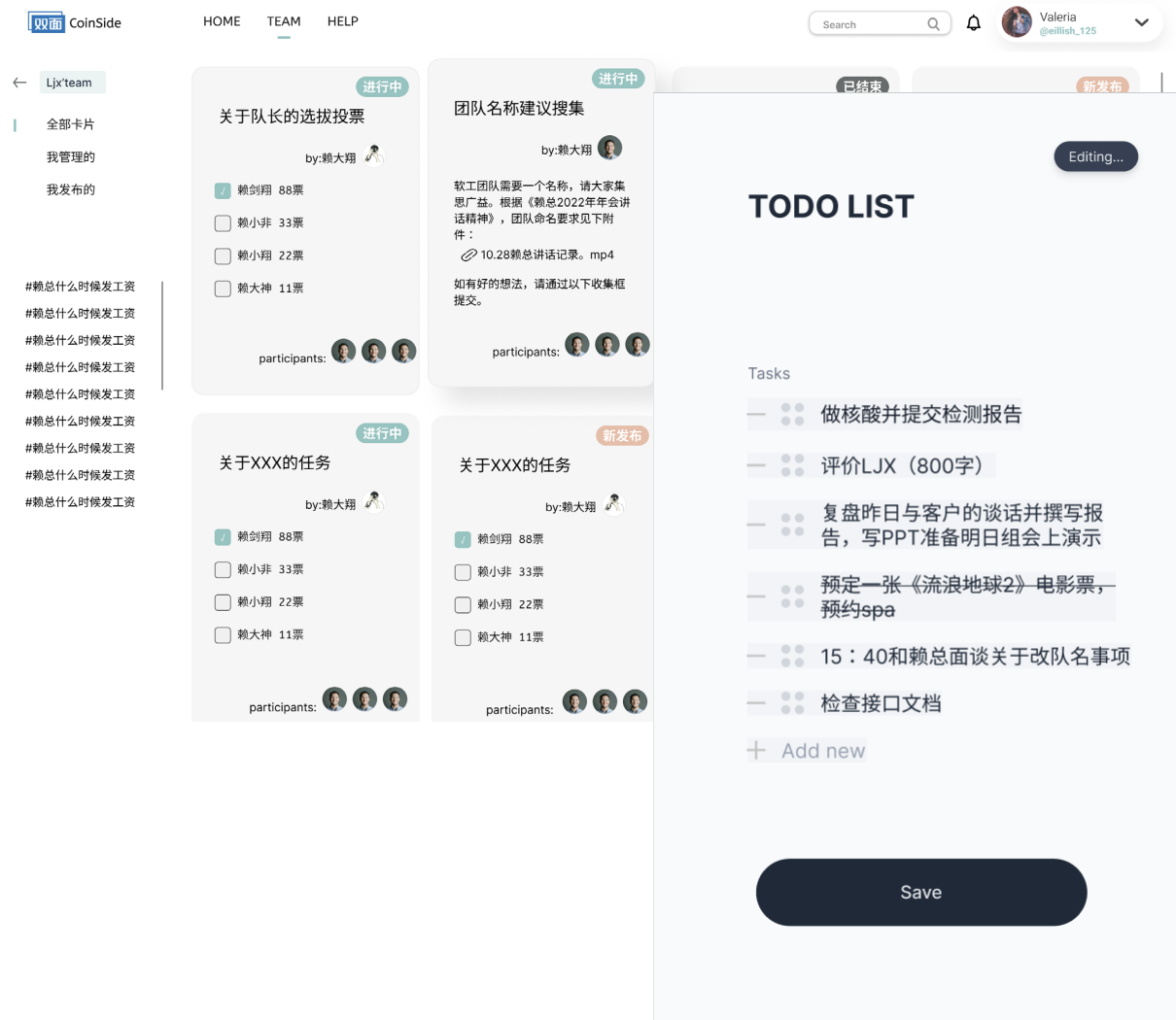




双面 CoinSide

面向团队协作的革新解决方案



目录

CONTENTS

项目进度

PART01



项目架构

PART02



项目测试

PART03



成员分工

PART04



项目进度

项目实际进度:

后端

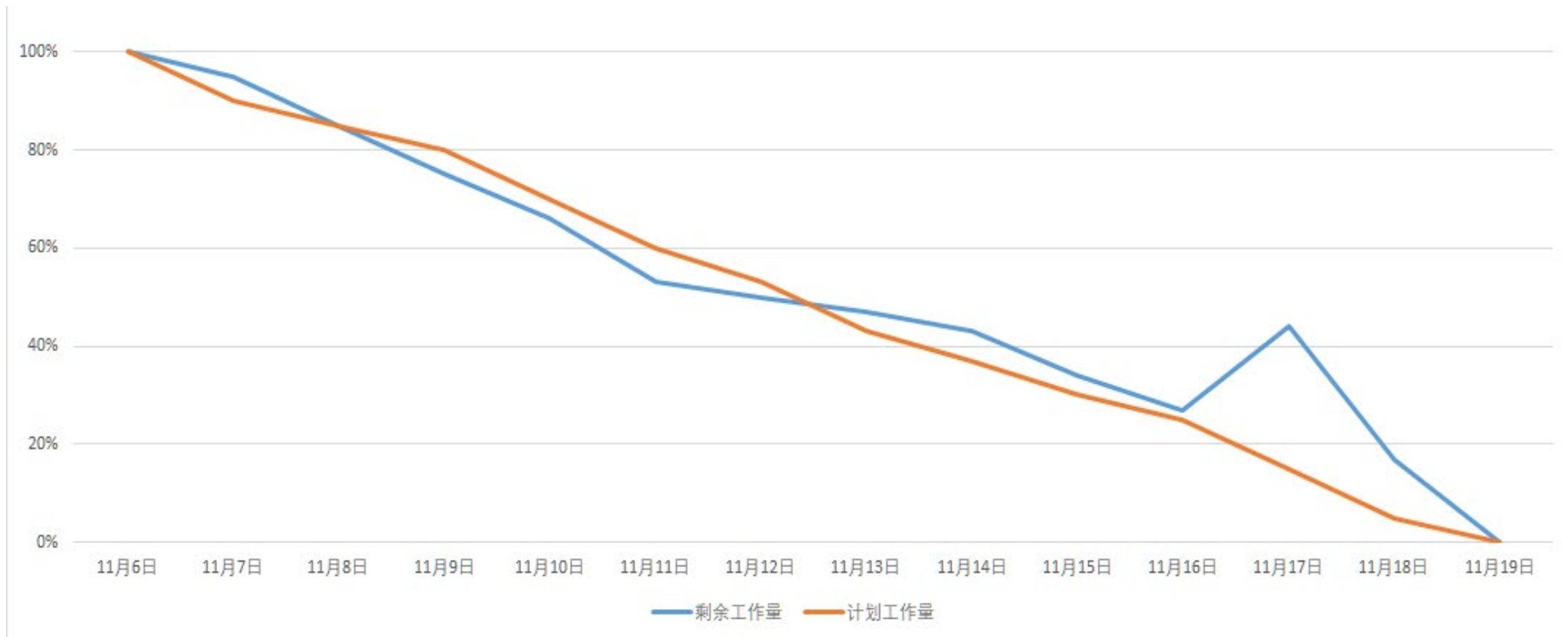
完成用户、团队、卡片及
markdown组件的内容，约占总任
务量的百分之六十

前端

完成用户主页、团队卡片展示页面、
卡片内部和部分组件页面、帮助页
面

项目进度

燃尽图：



项目进度

遇到的问题：

1.对Dubbo的使用和了解不足，且Dubbo文档内容较少，没有找到在Dubbo框架下spring的依赖注入方法，导致Java部分的服务层与bff层无法联通。

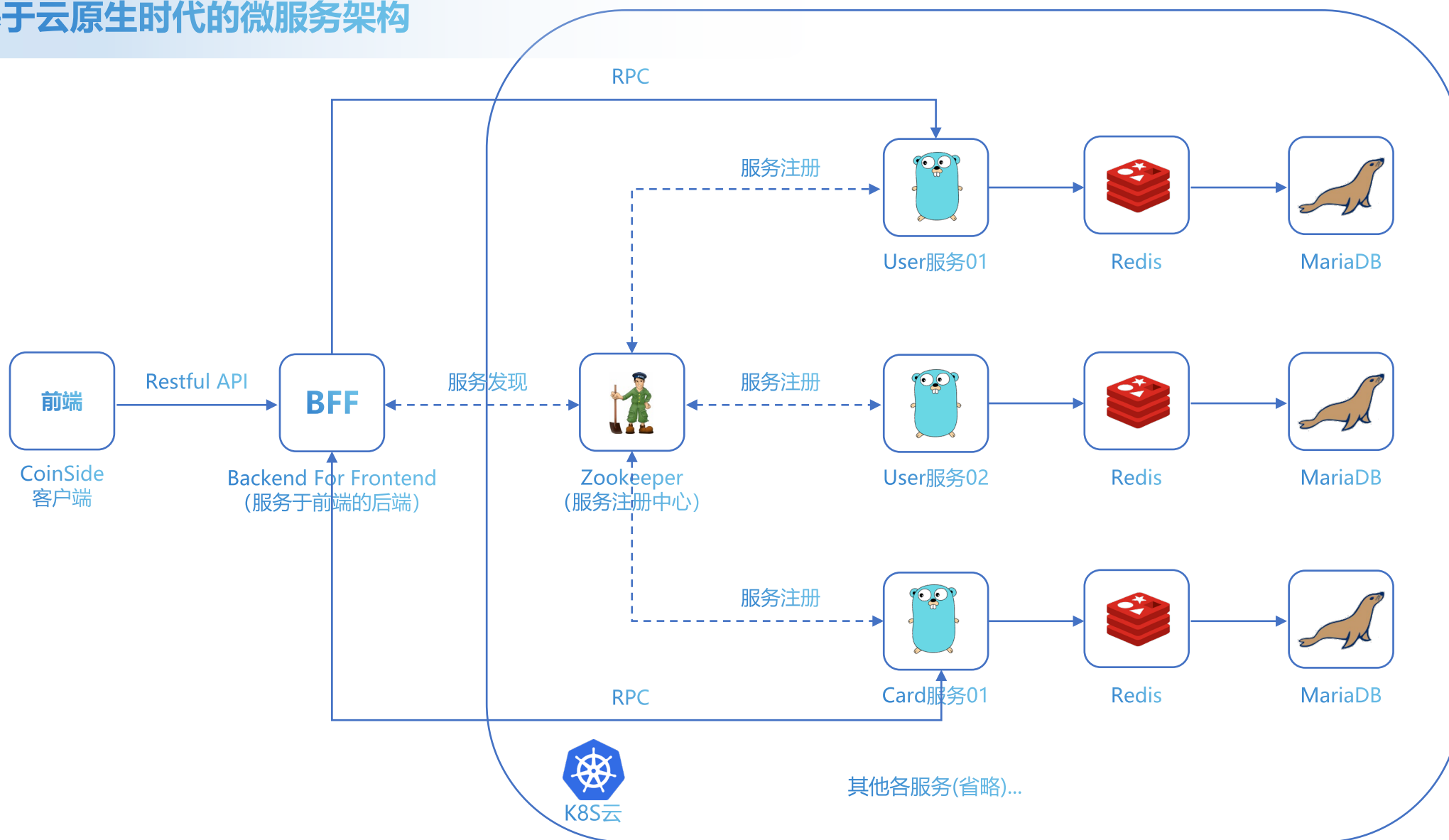
解决方案：使用Go的kratos微服务框架开发。

2.在进行团队卡片的操作时，需要确保在操作前后，团队成员仍在团队中不被移除等因高并发所带来的问题。

解决方案：使用分布式事务解决（将在下一阶段完成）。

项目架构

基于云原生时代的微服务架构



项目架构

基于云原生时代的微服务架构

后端项目结构

```
~/workplace/coinside-backend-kratos 10:32:07  
tree -L 2
```

```
.  
├── README.md  
├── api                                // grpc proto定义  
├── app  
│   ├── attachment                    // 附件组件服务  
│   ├── bff                          // bff层  
│   ├── card                         // 卡片服务  
│   ├── chat                         // 聊天组件服务  
│   ├── form                         // 收集表组件服务  
│   ├── markdown                     // markdown组件服务  
│   ├── team                         // 团队服务  
│   ├── todo                         // 代办组件服务  
│   ├── user                         // 用户服务  
│   └── vote                         // 投票组件服务  
├── docker-compose.yml  
├── go.mod  
├── go.sum  
├── pkg  
│   ├── README.md  
│   └── util                          // 公共util
```

23 directories, 6 files

项目架构

后端框架



开发语言: Golang



BFF层: Gin

GORM

data层: Gorm

微服务



微服务: Kratos
(bilibili开源微服务框架)

服务注册中心:
Zookeeper



数据库



MariaDB

缓存



Redis 分布式高性能缓存

Kratos->



项目架构

前端框架



组件



由Vue Draggable实现
数据驱动的组件拖拉拽

v-md-editor

由v-md-editor引入
markdown组件

项目架构

项目部署:

前端和后端部署

docker compose up (后期上k8s云)



支持环境

可以部署在包括任何x86_64和arch64架构的环境
(包括Linux、Unix、Windows、darwin)
需要docker以及docker compose

项目测试

测试计划及测试工具的选择

● 测试计划

● 单元测试

使用Go语言原生支持的单元测试

● 接口测试

包括用户接口、团队接口、卡片接口、token接口、markdown组件接口、附件组件接口、收集表组件接口、投票组件接口、代办组件接口进行RPC级别、HTTP级别测试

● 系统黑盒测试

从用户观点出发，尽可能发现软件的外部行为错误

● 测试工具



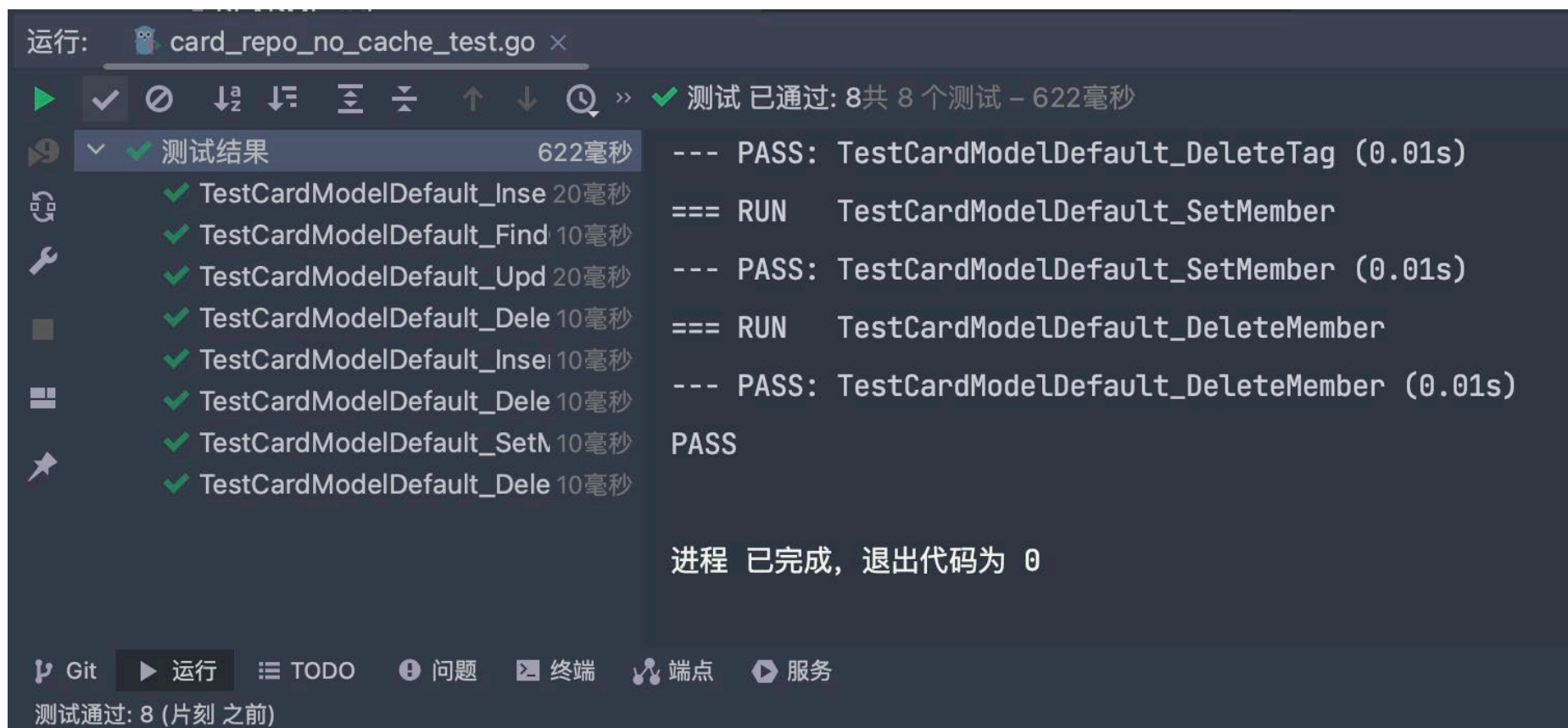
Apipost

- 采用Apipost接口测试工具
- 使用Mock+ 自动化测试

项目测试

已完成的测试

- 单元测试



运行: card_repo_no_cache_test.go x

测试 已通过: 8 共 8 个测试 - 622毫秒

测试结果	622毫秒
✓ TestCardModelDefault_Inse	20毫秒
✓ TestCardModelDefault_Find	10毫秒
✓ TestCardModelDefault_Upd	20毫秒
✓ TestCardModelDefault_Dele	10毫秒
✓ TestCardModelDefault_Inse	10毫秒
✓ TestCardModelDefault_Dele	10毫秒
✓ TestCardModelDefault_SetM	10毫秒
✓ TestCardModelDefault_Dele	10毫秒

```
--- PASS: TestCardModelDefault_DeleteTag (0.01s)
=== RUN   TestCardModelDefault_SetMember
--- PASS: TestCardModelDefault_SetMember (0.01s)
=== RUN   TestCardModelDefault_DeleteMember
--- PASS: TestCardModelDefault_DeleteMember (0.01s)
PASS
```

进程 已完成, 退出代码为 0

Git 运行 TODO 问题 终端 端点 服务

测试通过: 8 (片刻 之前)

项目测试

已完成的测试

RPC测试

Rock的私有团队

+新建...

Cookie管理器

全局参数

参数描述库

最近

API设计

API调试

自动化测试

文档分享

项目/团队

笔记

进群有礼

回收站

coinside-backend

用户接口

POST 创建用户

GET 获得用户

GET 查询用户id

PUT 修改用户头像

PUT 修改用户全名

PUT 修改用户配置文件

PUT 修改用户绑定邮箱

PUT 修改用户绑定手机

DEL 删除用户

token接口

团队接口

卡片接口

markdown组件接口

附件组件接口

收集表组件接口

投票组件接口

代办组件接口

新建grpc

新建grpc

PATCH 修改todo项

DEL 删除todo项

GRPC 新建grpc

POST 创建团队

GET 查询用户id

PUT 修改团队 (替换)

GRPC 新建grpc

开发中

新建grpc

导入proto

demo.proto

user.proto

card.proto

api.CardService

CreateCard

CreateCardStream

GetCardInfo

GetCardInfoStream

GetCardList

GetCardListStream

UpdateCard

UpdateCardStream

DeleteCard

DeleteCardStream

AddCardTaq

AddCardTaqStream

DeleteCardTaq

DeleteCardTaqStream

AddCardMember

AddCardMemberStream

服务器地址

localhost:9003

错误重连: 1 次

证书

查看proto

调用

请求内容

请求MetaData

提取字段和描述

美化

简化

生成模拟数据

1 { "team_id": 2, "title": "2022年11月18日", "content": "这是今天的主要内容, 大家及时查收" }

字段描述

响应内容

响应MetaData

成功示例

失败示例

1 { "code": "OK", "id": "4" }

响应码: 0 153.00ms

帮助

内置Mock变量

精简模式

控制台

上下分屏

新窗口打开

深色模式

缩放

设置

更新

成员分工

成员分工

赖剑翔

队长
主要负责后端服务开发

刘克鸿

后端部分负责人
后端框架搭建及部署

黄润隆

前端开发
负责前端页面开发

王帆

前端部分负责人
前端框架搭建及业务编写

郑植

产品经理
负责团队有关文档的编写与PPT、视频制作

程舸

美术/UI设计
部分页面编写

王铭靖

后端开发
负责业务代码编写

王小鹏

测试
负责后端接口的rpc测试、集成测试、撰写测试报告

高银涛

测试
负责前端的测试，包括功能、稳定性、易用性测试

成员分工

过程体会



后端开发：选择大于努力，我们在周四果断选择更换技术栈，选择bilibili开源的kratos框架，而且赶在beta测试前将后端主体框架切换完成，真可谓极限编程。



产品经理：真的很感谢组里大佬们这些天的辛勤努力，从框架构建到具体实现，他们都付出了许多。

成员分工

过程体会



前端开发：第一次尝试了vue3+typescript；ts的类型声明在前期阶段确实比较麻烦，但是随着项目的扩展，相比较js更容易找出错误；在本次项目中，使用了数据驱动式的组件拖拉拽以及组件渲染，在这一过程中困难重重，尤其是组件的信息结构设计，以及组件的布局信息保存，所幸最终还算完成了需求



前端开发：没有睡过一天好觉，前端的难点在于页面渲染和交互，vue3+ts的组合开发人员都是第一次用，难免有些生疏，所幸最后还是完成了大部分需求。



双面 CoinSide

面向团队协作的革新解决方案

