古人言

「古人言」团队出品

跨越千年時光

來一場真正的古人對話

前言

总有些惊喜的奇遇，比方说当我遇见你。从某种意义上说，世间一切，都是遇见。冷遇到暖，就有了雨；冬遇见春，有了岁月；天遇见地，有了永恒；人遇见人，有了生命。就像你在「古人言」小程序中遇见了……

时光漏斗翻转，转回千年永恒，听听古人在说些什么？

你是否希望耳闻李白“长风破浪会有时，直挂云帆济沧海”的洒脱，你是否钦佩刘禹锡“千淘万漉虽辛苦，吹尽狂沙始到金”的毅力；你是否感慨杜甫“安得广厦千万间，大庇天下寒士俱欢颜！”的家国情怀，你是否执迷于白居易笔下《长恨歌》的无奈。

因此，「古人言」小程序诞生了。这里有古人趣事的合集，品诗人文豪的生活余欢；这里有诗人的画像，穿梭千年，看他容颜；这里有标签对话，云当朝时政，所见所闻；这里有诗人生平阶段，顾他生历程。你将以诗人朋友的身份，体验沉浸式对话与情景式交互方式，全方位的从各种角度感受诗人的立体形象，带你来一场真正的跨越千年与古人进行“一场浪漫邂逅，一次有趣对话”。

如果评委**没有时间**全部看
可以**点击**推荐阅读**五星**的
章节看呀～

目录

1.产品介绍

1.1 说明和信息

「古人言」是一款以**沉浸式对话和情景式交互方式来打破时空限制、传播古诗词文化**的诗词类小程序。产品通过对传统的古诗词库进行创新性的再分类，以诗人作为起点，**逐步向下发散**，将诗人一生分为具有代表性的不同阶段，然后再往下对诗词库诗词进行标签分类。「古人言」小程序采用**对话模式**，旨在以趣味性方式呈现诗词本身，吸引用户注意力、激发用户对传统文化古诗词的兴趣，**帮助用户带入诗人朋友的角色中**，从新的视角感受诗词魅力和情感。借助闯关这类融合轻量游戏意味的形式，增强用户的参与感和成就感，帮助用户了解学习更多古诗词。同时小程序辅以**各式各样的诗人趣事**，给用户**多角度地展示诗人本身**，帮助用户从诗人出发更好的理解感悟古诗词。从而以小程序为载体，弘扬传承中国传统文化的瑰宝。

1.2 应用场景

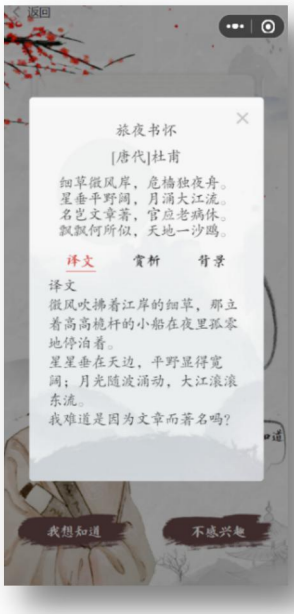
「古人言」非常适合在现今泛娱乐化信息爆炸时代下**内心浮躁**，或者是受外来文化裹挟**缺乏传统文化熏陶**，或者是平日忙碌、时间碎片化但是**对古诗词有着浓厚兴趣**，或者是对古代诗人的有着好奇心，或者是感觉课本诗词枯燥乏味，或者是喜欢探索新奇事物，或者是不断**追求科技与文化结合**，或者是有志于弘扬古诗词文化等，这些需要深入了解诗人和陶冶诗词情操的用户。「古人言」通过**具象化诗人形象**，以及对诗词库结构进行重新排布等方式，让用户走进诗人的一生，拉进与诗人的距离，鼓励用户闯关解锁诗人人生，作为诗人朋友和诗人**来一场跨越千年的对话**，以更轻松有趣有新意的方式帮助用户了解学习更多古诗词。

1.3 主要功能和使用方法

(一) 古人对话

选择感兴趣诗人 → 选择诗人生平阶段 → 选择感兴趣诗词标签 → 诗人生成问题询问用户意愿 → 用户选择对话选项 → 诗人用诗词作为答案回答用户 → 用户点击诗人对话框查看诗词详情 → 弹出诗词详情弹窗 → 左右切换诗歌原文、译文、赏析、背景 → 一次对话完成

小程序设计了与诗人对话功能，为现代人和古人提供沟通桥梁。用户可以根据自己的兴趣选择话题，开启和诗人的对话之旅，在对话中不断深入诗词内容，以新的角度理解诗人情感。「古人言」激发用户对诗人、诗词的好奇心，以轻松趣味的方式传播古诗词，在乐趣中学习、传承中国传统文化。小程序将古人对话页面分为从上到下的三个部分，分别为人生阶段和诗词标签、古人形象和对话框以及用户对话选项按钮。用户选择感兴趣的诗人后，将进入对话界面，整个对话模块建立在诗人和用户两个角色之间。该模块根据诗人外在特征、内在品格，定制了个性化的诗人形象，以诗人为主导角色，对用户进行引导。



(二) 游历一生

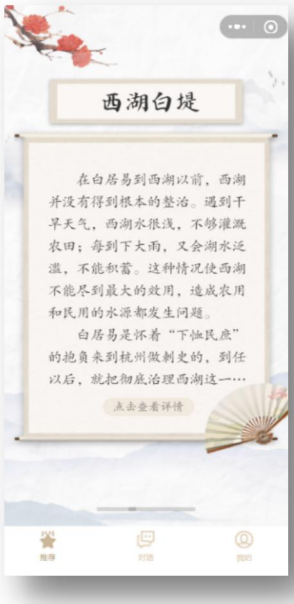
选择喜爱的诗人 → 选择感到好奇的人生阶段 → 弹出诗词选择题弹窗 → 根据诗词题目内容选择认为正确的选项 → 判断正确与否 → 回答正确后查看生平详情或回答错误后查看诗词详情 → 回答正确可以选择再来一题、下一阶段或回答错误后选择再来一题 → 完成一次闯关任务

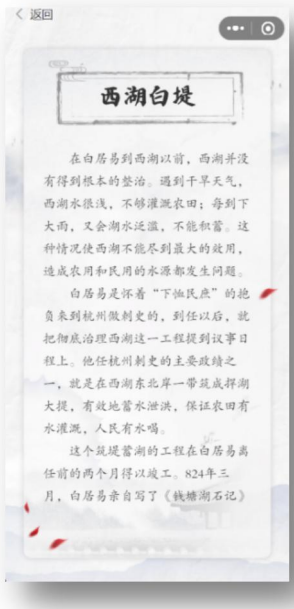
根据诗人的人生经历，不同时期的重要特征，通过将诗人的一生细化成多个具有代表性的人生阶段，其中每一阶段下包含多首诗人所作诗词，使用闯关解锁模式，让用户跟随诗人的脚步，品读赏析诗人所作诗词，感受诗人的悲欢离合，丰满、立体化诗人形象的同时，让用户走进诗人的内心、走进诗人的一生，与诗人交友，陪伴诗人度过一生。

(三) 奇闻逸事

首页推荐趣事 → 点击查看趣事详情 → 向左滑动刷新查看更多

此页面采用「卷轴式」的展示方式，用户可以清晰的看到根据用户画像推荐的趣事名称和部分内容，如果用户对此趣事比较感兴趣，可以进一步查看趣事详情，否则可以向左滑动刷新查看新的趣事推荐。





(四) 游历记录

个人中心模块查看游历记录 → 点击后跳转至相对应的诗人游历界面

该部分记录了用户的所有游历记录，以[卡片式]的展示方式，让用户可以清晰的看到自己已经游历过的诗人的生平，点击其中的游历记录，将会直接跳转至相对应的诗人的游历界面。

(五) 新手指南

对于初次使用小程序的新用户，提供了具有指引功能的新手指南，帮助新用户了解并熟悉小程序每个功能模块的操作流程和方式，确保用户能够快速入门，提高用户体验。

2.需求分析及设计

2.1 市场分析

2.1.1 产品背景

传统文化是国家软实力的重要组成部分之一。习近平总书记强调：“没有中华文化繁荣兴盛，就没有中华民族伟大复兴。”并且，党的十九届五中全会《建议》明确提出：“**传承弘扬中华优秀传统文化**”、“到 2035 年**建成文化强国**的目标”。传承弘扬中华优秀传统文化，不仅可以提高国家文化软实力，更能推进社会主义文化强国建设。以**创造性的方式转换中国传统文化**，才能使中国传统文化焕发新活力，获得创新性发展。

而**诗词文化**更是传统文化中的中坚力量，不管是牙牙学语的孩童，抑或是年过七旬的古稀老人，都或多或少的能吟诵出几首诗来。随着中国传统文化的复兴，这些美丽的文化得到了人们的进一步认可。

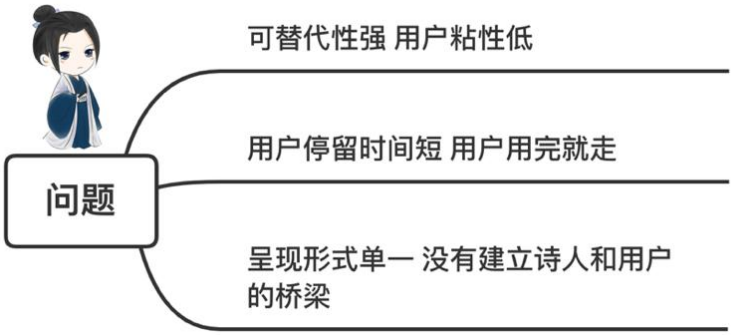
2.1.2 市场问题

古诗词类小程序目前存在以下问题：

1. **可替代性强**，对于信息爆炸的时代，信息查询的接口存在于方方面面，相较于传统的 web 端信息分页式的呈现，小程序单一的查询功能便显得相形见绌。

2. **用户停留时间短**，用户用完就走，单一的查询功能使得用户只是对诗词进行了简单的唯一检索，并没有真正的能感受到中华诗词的魅力所在。

3. **呈现形式单一**，市场上的诗词小程序几乎都为简单的呈现诗词内容，并没有建立用户和诗人的桥梁，单纯的诗词内容缺乏诗人情感和写诗生平经历的融入。



2.2 需求分析

2.2.1 业务需求

业务需求为：传播诗词文化，了解诗词魅力，体验与古人“对话”。

2.2.2 用户需求

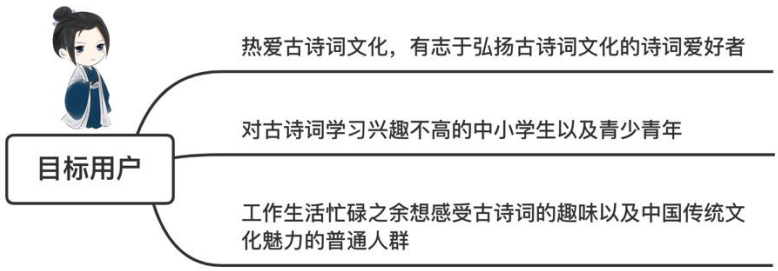
对于用户需求团队首先进行了用户调研。

团队设计“关于古诗词学习方式”的调研，共发放电子问卷 1500 份，收回有效问卷 1378 份，回收率 91.87%。根据团队设计的问卷问题，并结合数据统计结果，团队得出用户在使用诗词类程序时的有以下需求：

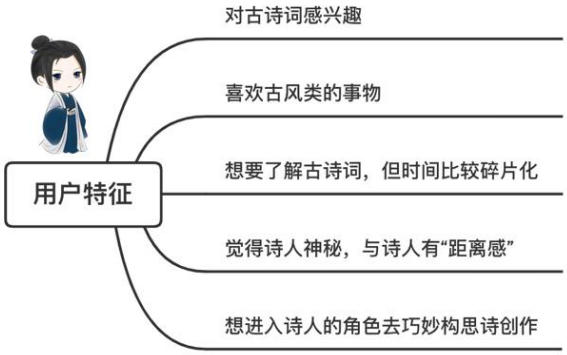
1. 79%的用户希望该程序使用便捷，即开即用
2. 85%的用户想体验与古人来一场沉浸式“对话”
3. 73%的用户认为单纯的信息查询古诗词形式过于单一，枯燥乏味
4. 65%的用户希望诗词能以创新性的方式进行呈现
5. 88%的用户对诗人的生平及趣事很感兴趣，希望得到更多了解
6. 67%的用户缺乏一个以诗人朋友的视角感受古诗文魅力的平台
7. 75%的用户希望能够理解性地学习古诗词
8. 77%的用户希望利用碎片化时间学习古诗词

2.3 用户分析

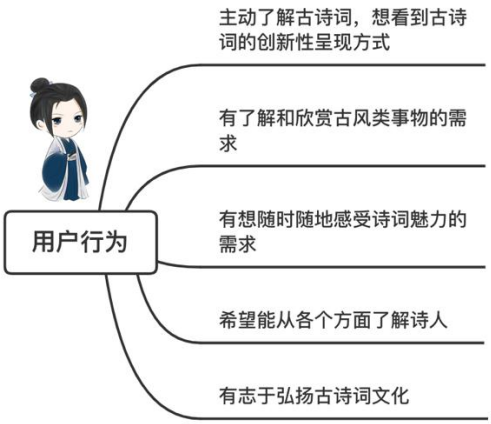
2.3.1 目标用户



2.3.2 用户特征



2.3.3 用户行为



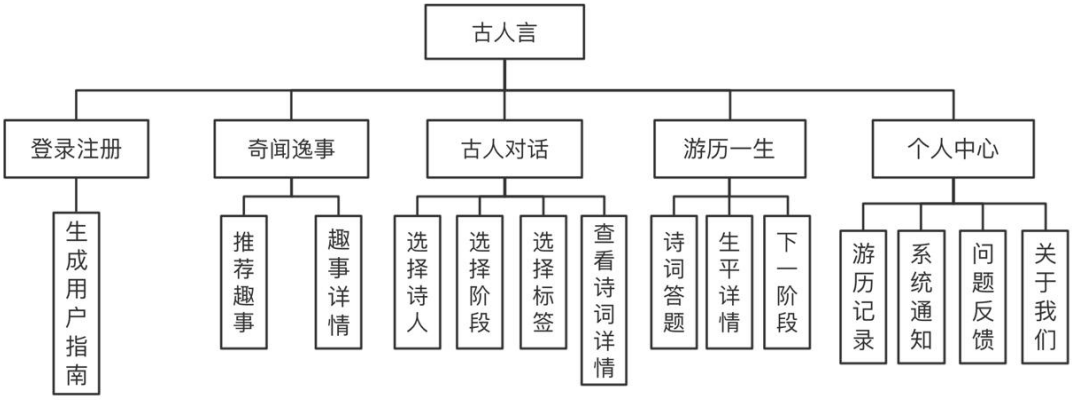
2.4 功能及 UI 设计

本文档将产品具体功能设置在产品介绍。此部分主要介绍功能的交互逻辑和 UI 设计。

2.4.1 功能设计

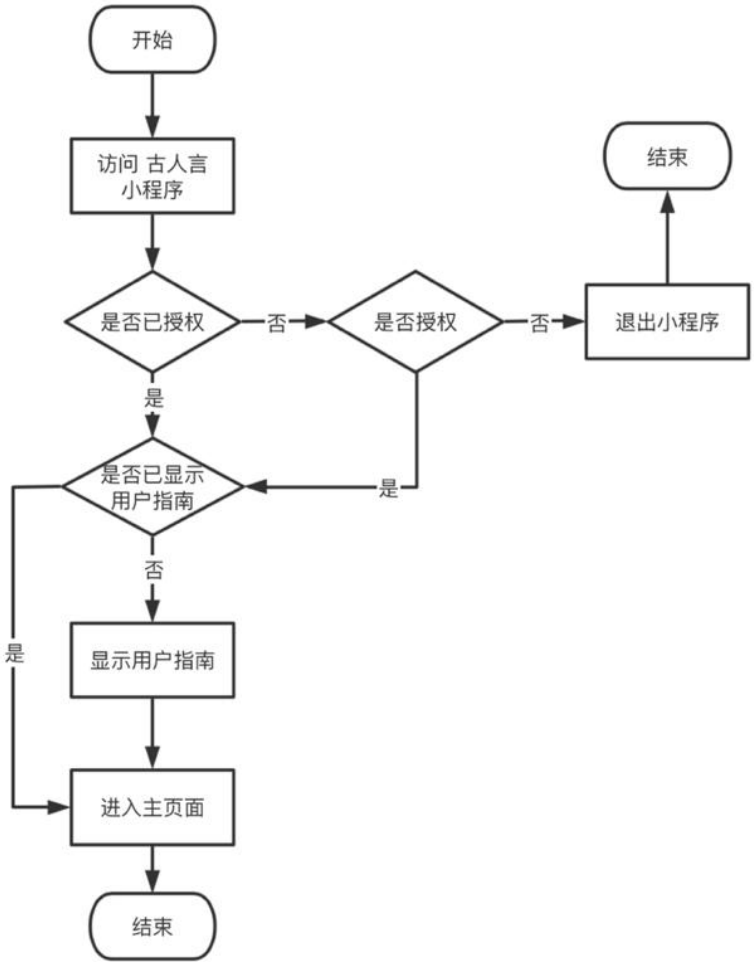
总体功能概述

主要功能模块包括登录注册模块、奇闻逸事模块、古人对话模块、游历一生模块、个人中心模块。



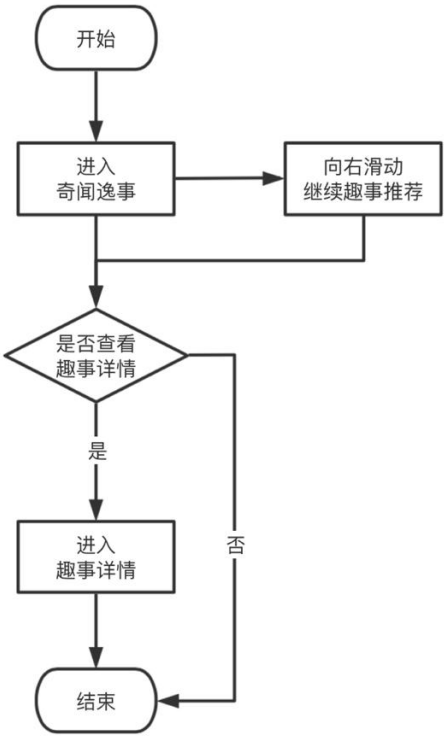
(1)登录注册模块

用户登录小程序后，系统进行账号检测，若账号已经处于已注册状态，则直接进入主页面，若尚未注册，则为自动生成用户操作指南。登录注册的处理流程如图所示。目前计划是只开放通过微信入口登录。



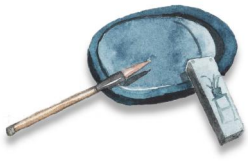
(2)奇闻逸事模块

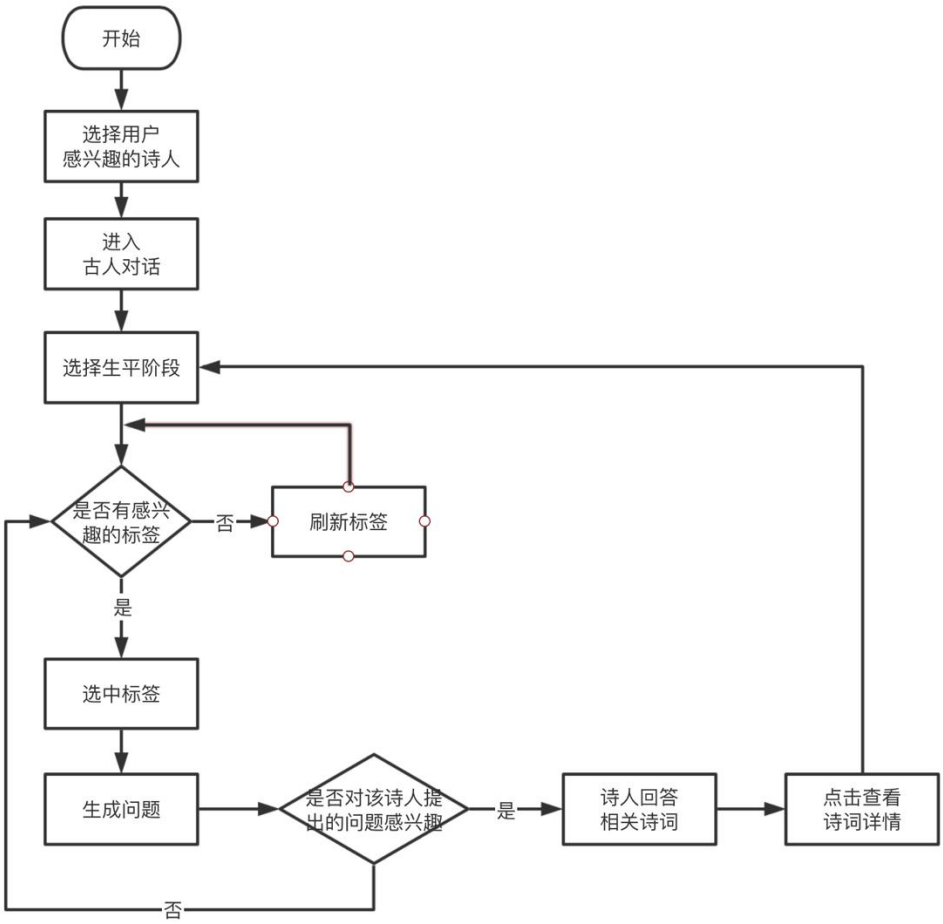
奇闻逸事模块包括推荐诗人趣事、查看趣事详情等功能。每次用户重新进入小程序时，首先显示奇闻逸事模块，用户可以左右滑动切换趣事，向左刷新获取新的趣事。推荐的诗人趣事默认显示趣事标题和部分趣事内容，用户可以点击“查看趣事详情”进入详情页面，通过上下滑动查看诗人趣事所有内容，若用户想返回上一页面，可以点击“返回”，即回到推荐趣事页面。



(3)古人对话模块

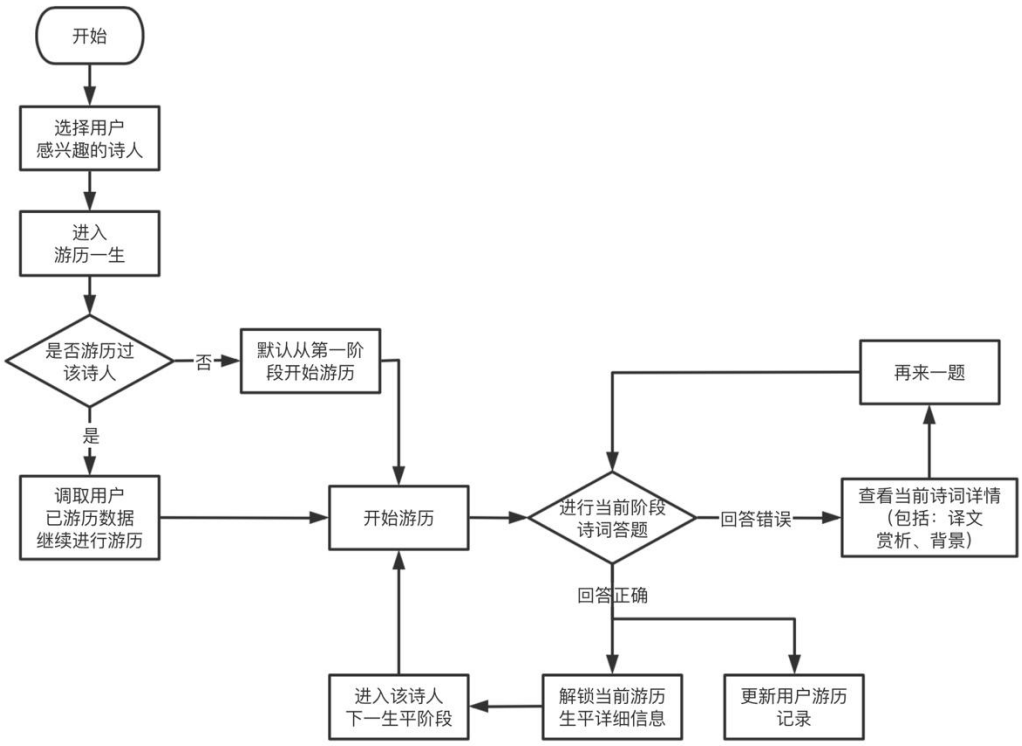
古人对话模块主要包含选择诗词标签、进行对话和查看诗歌详情等功能。用户选择诗人后，进入对话界面，整个对话模块建立在诗人和用户两个角色之间。该模块根据诗人外在特征、内在品格，定制了个性化的诗人形象，以诗人为主导角色，对用户进行引导。用户需要跟随诗人指引，首先选择感兴趣的生平阶段，选定阶段后，用户根据自身偏好，选择阶段中的一个诗词标签，其中诗词标签提供刷新功能。之后诗人将通过标签生成相关问题，询问用户意愿，若用户给予积极回应，诗人将以所作诗词为对话内容，回答用户，同时用户能够查看诗词的详情内容，进一步了解诗人和诗词；若用户对诗人提出的话题不感兴趣，诗人会提示用户切换标签，开启新的对话内容。





(4)游历一生模块

游历一生模块包括选择诗人、选择人生阶段、诗词答题等功能。该模块根据诗人的人生经历，不同时期的重要特征，将诗人的一生分成了多个具有代表性的人生阶段，其中每一阶段下包含多首诗人所作诗词。在该模块，用户可以先选择感兴趣的诗人，开启游历之旅，用户首次游历时，默认从诗人第一人生阶段开始，进行诗词答题，题目形式为选择题，表现为挖空诗词，共有四个选项，用户需要选择其中一个选项，使之和题目诗词匹配。如果回答正确，用户将解锁诗人当前生平详情，同时可以选择进入下一阶段；如果回答错误，用户可以查看题目对应诗词的详情内容，包括诗词译文、赏析和背景，由于只有在用户回答正确的情况下，才能解锁下一阶段，所以答错后用户可以选择“再来一题”，直至用户回答正确后，方可进入下一阶段。



(5)个人中心模块

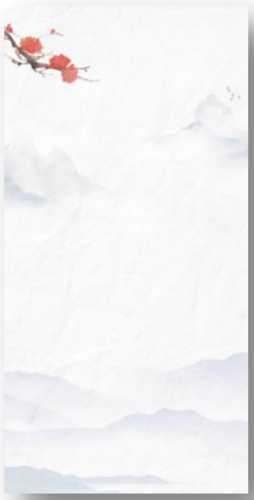
个人中心模块包括，游历记录、系统通知、问题反馈等功能。该模块负责记录用户信息和行为数据，提供用户与开发者的交流窗口。游历记录，帮助用户查看自己已经游历过哪些诗人的哪些生平；系统通知，是开发者对「古人言」小程序，更新迭代的相关通知；问题反馈，为用户反馈小程序问题和建议提供了便捷的接口；关于我们，则是小程序整体介绍以及关于「古人言」团队的设计研发的初心。

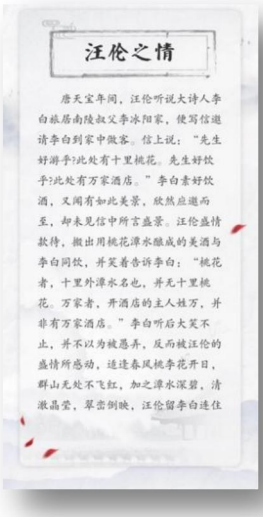
2.4.2 UI 设计

界面风格设计

青衫烟雨客，似是故人来。古，寻时代印记；风，品前人风尚。界面整体以古风元素搭配，结合毛笔、印章、卷轴元素。符合大众审美，切合小程序主题。如今古风已经成为一种文化，用户通过小程序能很容易地感知其中的浪漫情怀，不需要精深钻研，也

能代入其中情境，只因这是中华民族所共通的审美倾向、文化传承。小程序古风风格是当代话语对过去的重塑，同时，能激发用户的民族自豪感。





配色的选择与设计

背景：颜色选择#FFFFFF 至#DADEE5 之间过渡的颜色，整体配色给人整洁雅观的视觉效果，其中花枝元素选择#E2756A 周围过渡色，给人舒适却不乏新意、清新却不乏别致的效果。与页面的其他元素有较好的契合，让用户身临其境。通过调整色彩的饱和度和色温，让界面层次丰富。

元素：整体色调选用#FBF7F4，该色调白中微黄，属于暖色，中调饱和度，让用户体会到古风质感，有拿起古代文豪书卷之意，大大增加了对话的代入感，以#969492 色调为主的边框，富有古代建筑之美，与背景色搭配，给人平静、深邃的感觉。在主体上，用数据与感情符号交流，增加了情感化体验和视觉效果。

诗人形象设计

我们通过搜集研读历史记载中对该诗人的描述并加以想象后，进行了诗人的手绘。绘制过程中，我们将诗人的性格融合到诗人的画像中，让用户有一种别具一格的视觉体验。诗人服饰的搭配展现了朝代的变迁和人物心境的变化，例如李白带的乌纱帽，正如他在《答友人赠乌纱帽》中写到：领得乌纱帽，全胜白接篱。山人不照镜，稚子道相宜。毛笔、油纸伞、发簪、书卷等元素丰富了诗人形象，服饰颜色的多元化体现了诗人的精神，同时也给了用户无限遐想，你心中的大文豪到底是什么样子的呢?

情景式交互设计

有这样一句话：“若不深入其中，何来感同身受”，我们正是希望将用户代入其中有一段灵魂的邂逅，我们通过对话气泡框的设计，增强了对话感，又通过标签的选择，赋予了诗人新的生命力，随机问题的生成，避免了乏味单调，让用户有一种多元化的体

验感。体会诗人对朝代当局的眙恫，对与友人相见的欣忭，漫步竹林的肆意疏朗，情景对话的交互有一种无形胜有形的氛围感，让用户体会到“生而诗意，美而无言”。心灵的碰撞之余，我们让用户能联想到自己，将自己的情感射入诗人的处境中，感受诗人的希冀。

2.5 非功能性需求

2.5.1 性能需求

「古人言」作为信息查询类小程序，后台数据库拥有大量数据，对数据库访问查询速度要求较高，目前在阿里云学生服务器的前提下制定出如下标准：

功能	响应时间	并发数
登录授权	<2s	50
静态资源加载	<2s	40
标签列表生成	<2s	40
显示诗人详细生平	<2s	35
查询诗词详情	<2s	30

在优化数据库查询方面，团队成员从以下几个角度进行分析：

1. SQL 查询语句优化

适当建立索引以使得查询速度得到提升，借助 Explain 选择更好的索引和优化查询语句，SQL 的 Explain 通过图形化或基于文本的方式详细说明了 SQL 语句的每个部分是如何执行以及何时执行的，以及执行效果。通过对选择更好的索引列，或者对耗时久

的 SQL 语句进行优化达到对查询速度的优化。使用查询缓存，并在云服务器上将尽量多的内存分配给 MySQL 做缓存。

2. 进行数据库的主从复制、读写分离和负载均衡

利用数据库主从复制的功能，实现数据库的读写分离，从而改善数据库的负载压力。利用数据库的读写分离，当服务器在写数据的时候，访问主数据库（master），主数据库通过主从复制将数据更新同步到从数据库（slave），这样当服务器读数据的时候，就可以通过从数据库获得数据。

3. 进行数据库的分表、分区

通过分表可以提高访问效率，通过分区将一张表的数据分成多个区块，多块硬盘同时处理不同的请求，从而提高磁盘 I/O 读写性能。

4. 修改数据库引擎为 MyISAM

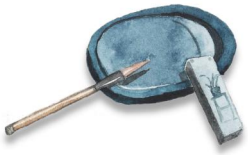
利用 MyISAM 管理非事务表。可以提供高速存储和检索，以及全文搜索能力。由于「古人言」中需要执行大量的 SELECT 查询，因此团队认为 MyISAM 是更好的选择。

2.5.2 安全性需求

高频访问限制

用户点击按钮后，短时间内无法再次点击，利用锁机制，限制用户高频访问接口。在古人对话，游历一生部分，点击后系统出现遮罩层，处理成功后重新开放，阻止用户高频访问接口。

会话失效问题



由于断网、长时间无操作、后台服务器重启、授权失效、凭证过期等问题导致的用户会话失效、掉线、无法正常使用软件，执行越权操作的情况，我们在后台设置了过滤器，检测到用户未处在在线状态下会重新对用户进行授权。

2.5.3 可靠性需求

数据完整性

后台引入了事务回滚机制，用户进行修改，插入操作时，若系统出现异常会将 错误数据进行回滚，避免数据的不完整性，同时也会将错误信息记录到日志中，以便程序员第一时间处理异常。

易恢复性

考虑团队开发为敏捷开发模式，使用版本控制工具 git，在版本迭代过程中，发布新版本若出现异常，确保有回滚方案，回退到性能、功能正常的版本供用户正常使用。另外，主从服务器分别会在不同时间段对数据库进行备份，确保数据库信息不丢失。

容错性

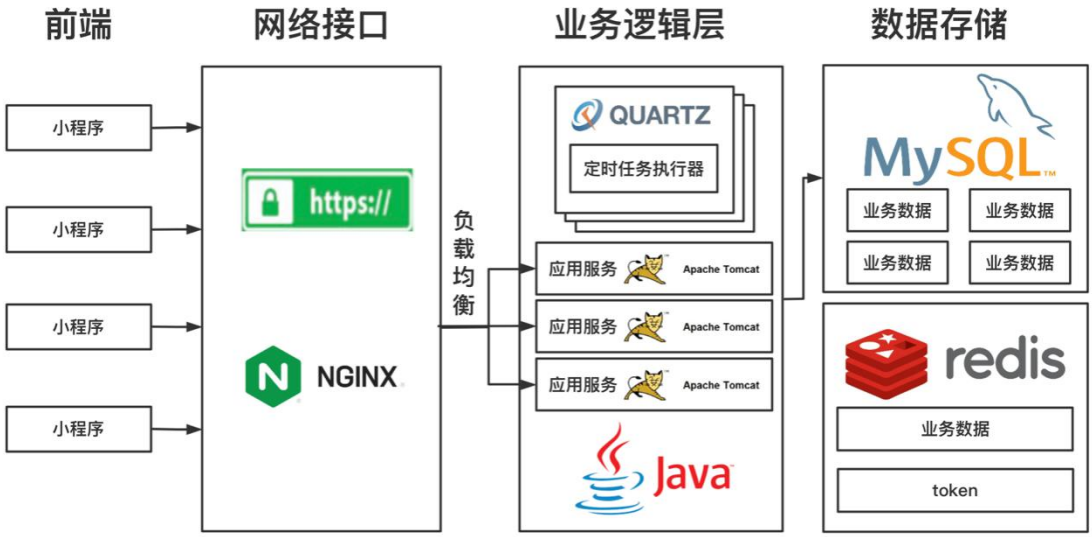
考虑用户突然经历断网，断电，或程序意外退出等情况，系统会定时保存用户的操作，待用户重新上线后恢复操作，避免因意外情况造成数据丢失。

3.技术方案

3.1 系统架构设计

「古人言」小程序 服务端采用 Spring Boot 框架，在有限的服务器资源下，采用 tomcat 集群+nginx 实现负载均衡提高服务器性能和可用性，可用 Redis 分布式锁实

现 Quart 调度集群进行定时任务处理。为了突破数据库的性能瓶颈，采用 MySQL 读写分离技术，并使用 redis 集群作为数据缓存系统。



3.2 部分技术介绍

小程序端

3.2.1 利用原生小程序语言

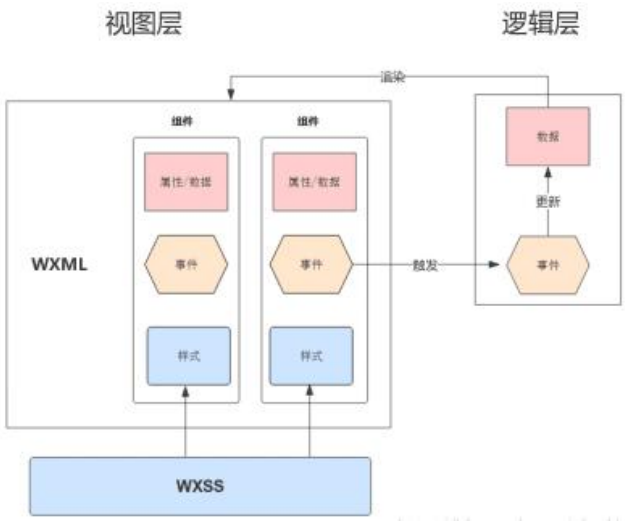
利用原生小程序语言，避免其他语言不兼容的问题，小程序语言为开发者提供了许多 API，方便开发者书写代码，实现交互逻辑功能。

微信小程序语言相比较其他语言具有以下优势：

1. 更系统：有自己独立并保持快速迭代的开发工具，写代码，调试，发布，全套搞定。

2. 更原生：提供诸多原生 API，通过微信客户端的渲染，使得小程序的交互实现更接近 app 原生。

3. 更安全：代码打包后需审核发布，接口不直接暴露。



3.2.2 color UI 库

color UI 是一个多元化的 UI 库，色彩鲜明，符合小程序的主题，使用户身临其境，复古却不乏活力。

3.2.3 云开发与传统后台服务器相结合

小程序端利用**云开发**和**传统后台服务器**相结合。云开发提供了一块存储空间，提供了上传文件到云端、带权限管理的云端下载能力，开发者可以在小程序端和云函数端通过 API 使用云存储功能。将字体移至云服务器，大大提高了字体的加载速度，有良好的用户体验。

后台服务端

3.2.4 nginx + tomcat 集群

为了解决「古人言」的高并发访问问题，后台系统采用 nginx 进行反向代理，tomcat 横向拓展集群；nginx 处理静态内容，tomcat 处理动态接口请求。nginx 负责动态请求的负载均衡分配，利用 nginx 的容错机制实现高可用。

3.2.5 后台缓存静态数据

系统中存在着大量对数据库的查询，消耗数据库资源，影响数据的插入，于是采用 redis 进行一些变动较小的数据进行缓存，如登录模块用户信息等。

3.2.6 MySQL 实现读写分离

考虑到系统的大多数模块都需要大量数据展示，因此存在大量查询数据库数据接口。若使用同一个数据库进行查询和插入操作，那么查询会占用大量的数据库资源，而插入的数据往往又需要更多的资源，因此决定使用 MySQL 集群来分离读操作和写操作，主数据库负责插入数据，多个从数据库负责读取数据，最后利用主数据库的二进制日志实现主从库数据同步。

3.2.7 采用 Docker 虚拟化

Docker 是一个开源的应用容器引擎，可以让开发者将应用以及依赖包打包到一个轻量级、可移植的容器中。Docker 通过创建进程的容器，不必重新启动操作系统，几秒内能关闭，可以在数据中心创建或销毁资源，不用担心额外消耗，通过更积极的资源分配，以低成本方式对一个新的实例实现一个更聚合的资源分配，大大提高数据中心的利用效率。

4.核心竞争力

4.1. 诗词融合对话模式以及诗词标签化

对话：“你绕着我转干什么？”“我在环游世界”，这是两个人之间简单而温馨的对话。与其他古诗类程序相比，古人言程序更着眼于对话，通过情景式沟通，打造人与情境合一。千余年斗转，仿佛古人言小程序中的李白，在林深出吟诵：“犬吠水声中，桃花带露浓。树深时见鹿，溪午不闻钟。”，而我在听。情景式对话，代入感更强，有利于吸引用户的兴趣，让传统文化更好的输出。

标签：每首诗歌总有中心的思想，或是欣赏良辰美景时的华蜜，亦或是怀才不遇的愤懑，这都藏在字字句句的碰撞间。标签正是这种思想的轮廓，或是对这种行为概括。标签能更精确的引导用户了解整首诗，看诗人的悲欢离散。

4.2 诗人形象立体化

画像：魏颢印象中的李白，“眸子炯然，哆如饿虎”，“或时束带，风流酝籍”。通过查阅资料，我们绘制了我们心中的诗人李白，他头戴乌纱帽，手捧书卷，诗人的画像总给人无限遐想，有一种无形的神秘感，也拉近用户与诗人心灵的距离。

趣事：通过爬虫技术我们获取诗人的趣事，我了解到王昌龄和李白一见如故，边泛舟边饮酒，范仲淹改革军事制度，他们是文豪大家，同时也是普通人，他们生活的点点滴滴也充满了乐趣，有时间他们也会给自己的生活加点糖。古人趣事让用户更亲近古人，体会他们的喜怒哀乐。

人生阶段：条分缕析的逻辑结构总给人清晰明了之意。我们将诗人的人生阶段进行划分，那是李白早年天才、辞亲远游、蹉跎岁月、西游献赋、李杜相识、安史入幕、溘然病逝；亦或是柳宗元早年生活、少年成才、革新时期、左迁永州，清晰的人生阶段划分，能让用户更好的了解诗人的一生，寻觅古人的印记亦或是心路历程。

4.3 产品概念

在泛娱乐化信息时代，人们大块的时间因电子产品的出现被分割成了无数零碎时间，大众在休闲之余的项目选择通常是玩游戏，而不是静下心来去阅读或者欣赏古诗词。「古人言」团队为满足用户对传统文化古诗的需要，将传统文化与科学技术结合在一起，创新性的制作了的“跨越千年与古人进行一场浪漫邂逅，一次有趣对话”的「小程序」小程序。古，历史回眸；人，世俗共鸣；言，亦欢亦悲哉！传统文化精髓，永不磨灭！

5.系统测试

5.1 测试环境

测试环境（Testing environment）是指测试人员利用一些工具及数据所模拟出的、接近真实用户使用环境的环境，测试环境的目的是为了使测试结果更加真实有效。测试环境应该与开发环境分隔开，使用独立的客户机、服务器和配置管理工具。

在进行测试工作前，团队确认并记录运行测试的环境，根据系统所需环境和测试工作计划相配合确保能够顺利进行测试的运行。本项目选取的测试环境中配置了多台阿里云学生云服务器、一台华为 Nove4（测试 Android）和一台 iPhone X（测试 iOS），这在一定程度上可以模拟系统真实的运行情况。测试环境的具体参数如下所示。

操作系统	iOS 14.0.1	Android 10
内存	256GB	128GB
处理器	A11 芯片	HiSilicon Kirin 970
主要测试工具	WeChat v8.0.5	WeChat v8.0.3
辅助测试工具	微信开发者工具	微信开发者工具

表 5.1.1 客户端测试环境

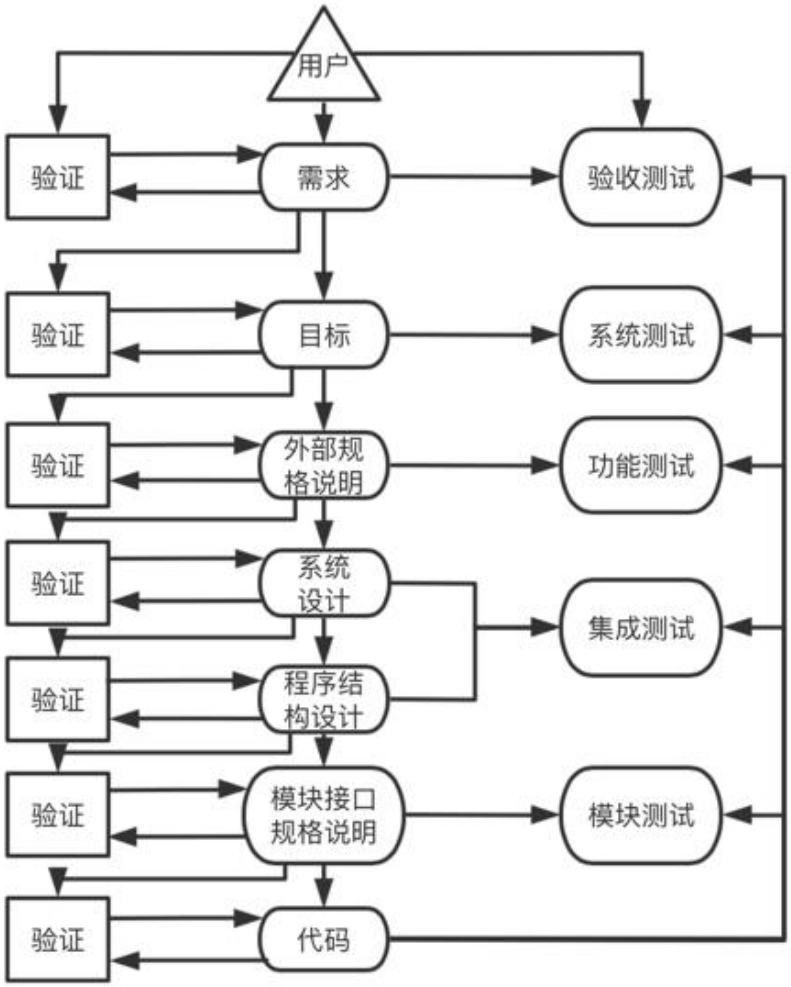
操作系统	CentOS 7.3
内存	4GB * 3 + 2GB *1
处理器	双核 * 3 + 单核 * 1
数据库	MySQL、redis
主要测试工具	Postman
辅助测试工具	微信开发者工具、Chrome

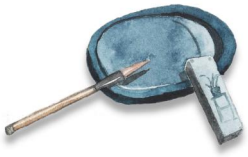
表 5.1.2 服务端测试环境

5.2 测试流程与方法

本系统测试的主要流程为：根据测试计划——自下而上逐步测试。先从代码部分，手动利用 Junit5、Postman 以及微信开发者工具进行模块测试(单元测试)；在确保接口及交互部分全部可行后，根据测试样例进行相对应的集成测试；接着团队根据测试计划，搭建相对应的测试环境，进行功能测试和系统测试；最后小程序发布上线，测试人员与软件开发人员一起进行测试。根据测试结果对系统进行完善与优化的同时将结果记录在案并整理成文档记录，方便后续查看。

测试工作流程图如下所示。





模块测试

模块测试又称单元测试，是针对软件设计的最小单位程序模块进行正确性检查的测试工作，模块测试需要从程序内部结构出发设计测试用例，多个模块可以平行地独立进行模块测试。本系统在开发过程中主要利用 Spring Boot 基于 Junit5 的单元测试，对后台接口及方法进行逐一测试。

集成测试

集成测试又称为组装测试或联合测试，在单元测试的基础上，需要将所有模块按照概要设计说明书和详细设计说明书的要求进行组装。本系统在开发过程中主要利用 Postman 测试工具，微信开发者工具等进行集成测试。

系统测试

通常意义上的系统测试包括压力测试（也称为强度测试）、容量测试、负载测试、性能测试、安全测试、容错测试等。团队根据测试工作计划和系统所需环境，搭建运行测试的环境。为了尽量模拟用户的真实使用环境，团队选用合适的操作系统和软件平台，了解符合测试软件运行的最低要求及用户使用的硬件配置，考虑到与其他软件共存时，是否能正常运行等。

5.3 测试结果

5.3.1 功能性测试

下面进行以游历一生模块的测试为例子阐述实际的系统功能性测试，其测试用例如下表所示。

设计目的	检测游历一生模块各项功能是否达到预期
测试指导	游历一生模块的需求分析、测试工作计划

测试数据		操作信息		
功能名称	步骤描述	前置条件	预期结果	测试状态
诗词答题正确	1. 测试人员进入游历页面； 2. 进行诗词答题，并选择正确答案	微信用户授权登录，且该用户登录信息未过期	1. 返回当前诗人该生平详细信息，并更新游历记录 2. 测试人员可以进入下一生平阶段 3. 测试人员可以选择再来一题	测试通过
诗词答题错误	1. 测试人员进入游历页面； 2. 进行诗词答题，并选择错误答案	微信用户授权登录，且该用户登录信息未过期	1. 返回当前诗词的赏析、解释、背景 2. 测试人员未通过该生平阶段，不更新游历记录 3. 测试人员可以选择再来一题	测试通过

对于系统中其他模块也编写了测试用例并进行相应的测试。通过一系列测试后，可以认为实现的系统已达到预期目标。

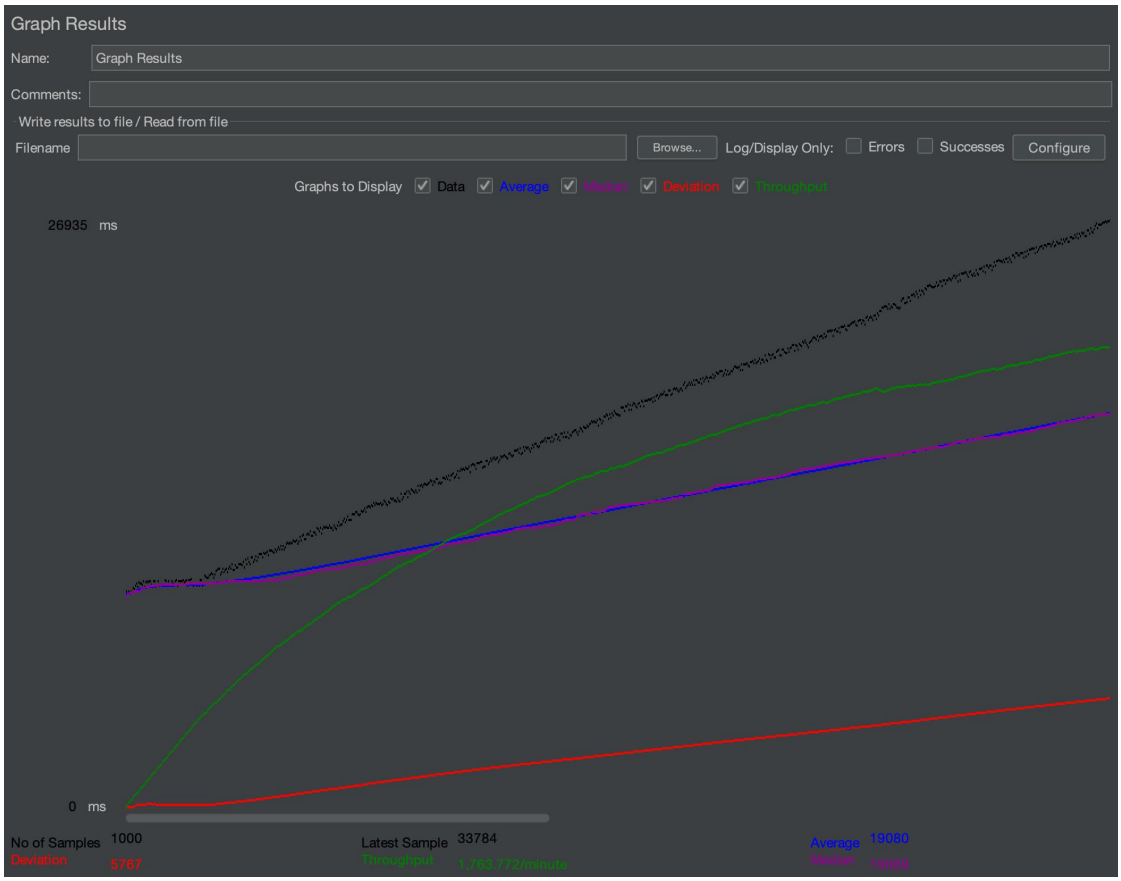
5.3.2 服务器端性能测试

团队采用 JMeter 作为服务器端性能测试工具。JMeter 作为一款广为流传的开源压测产品，最初被设计用于 Web 应用测试，如今 JMeter 可以用于测试静态和动态资源，例如静态文件、Java 小服务程序、CGI 脚本、Java 对象、数据库、FTP 服务器等等，

还能对服务器、网络或对象模拟巨大的负载，通过不同压力类别测试它们的强度和分析整体性能。

JMeter 的特点包括对 HTTP、FTP 服务器、数据库进行压力测试和性能测试；完全的可移植性；完全 Swing 和轻量组件支持包；完全多线程；缓存和离线分析/回放测试结果；可链接的取样器；具有提供动态输入到测试的功能等。在设计阶段，JMeter 能够充当 HTTP PROXY（代理）来记录浏览器的 HTTP 请求，也可以记录 Apache 等 WebServer 的 log 文件来重现 HTTP 流量，并在测试运行时以此为依据设置重复次数和并发度（线程数）来进行压测。

通过 JMeter 上可视化的统计数据对服务端性能进行评估，具体分析如下：



Graph Result

View Results in Table

Name: View Results in Table

Comments:

Write results to file / Read from file

Filename Browse... Log/Display Only: Errors Successes Configure

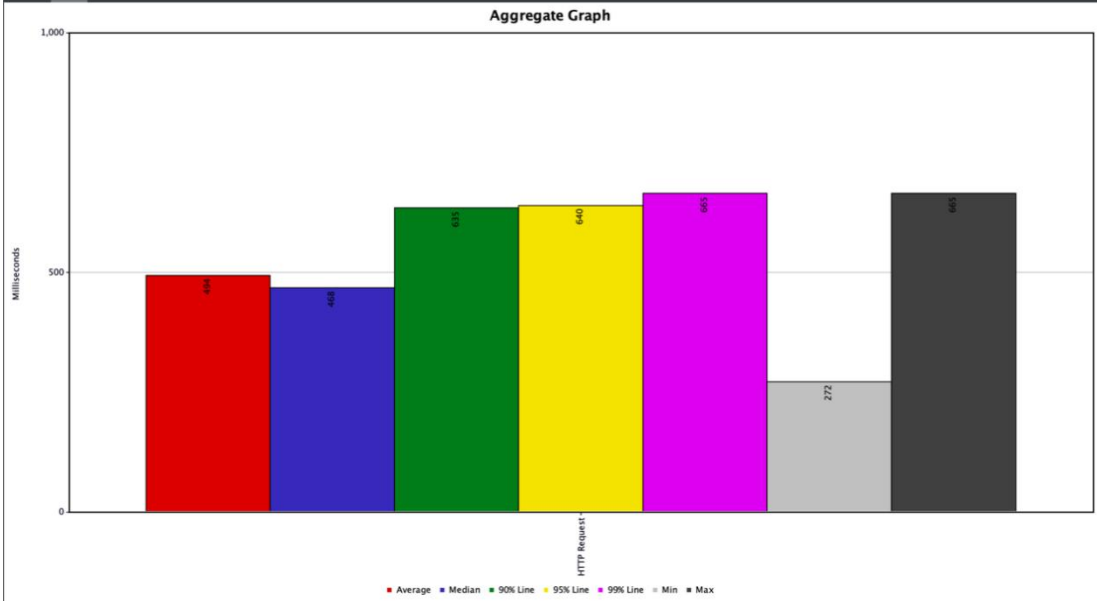
Sample # ↑	Start Time	Thread Name	Label	Sample Time(...	Status	Bytes	Sent Bytes	Latency	Connect Time(...
1	10:39:35.319	TestOne 1-905	HTTP Request	9513		5494	139	9508	9372
10	10:39:35.098	TestOne 1-169	HTTP Request	10021		8465	139	10021	9611
100	10:39:35.217	TestOne 1-644	HTTP Request	10835		5581	139	10835	9505
1000	10:39:35.296	TestOne 1-841	HTTP Request	33784		5054	139	33784	21215
101	10:39:35.081	TestOne 1-98	HTTP Request	10973		6080	139	10973	9642
102	10:39:35.115	TestOne 1-205	HTTP Request	10975		6625	139	10975	9607
103	10:39:35.153	TestOne 1-499	HTTP Request	10972		4644	139	10972	9566
104	10:39:35.228	TestOne 1-670	HTTP Request	10920		6273	139	10920	9495
105	10:39:35.074	TestOne 1-393	HTTP Request	11093		3885	139	11093	9647
106	10:39:35.348	TestOne 1-968	HTTP Request	10829		5519	139	10829	9364
107	10:39:35.098	TestOne 1-175	HTTP Request	11186		5196	139	11186	9623
108	10:39:35.327	TestOne 1-920	HTTP Request	10967		6848	139	10967	9394
109	10:39:35.088	TestOne 1-62	HTTP Request	11208		5684	139	11208	9635
11	10:39:35.170	TestOne 1-539	HTTP Request	10094		4784	139	10094	9524
110	10:39:35.249	TestOne 1-724	HTTP Request	11057		7617	139	11057	9471
111	10:39:35.364	TestOne 1-993	HTTP Request	10951		6908	139	10951	9359
112	10:39:35.065	TestOne 1-122	HTTP Request	11253		7356	139	11253	9658
113	10:39:35.081	TestOne 1-328	HTTP Request	11285		7336	139	11285	9638
114	10:39:35.284	TestOne 1-813	HTTP Request	11097		7544	139	11097	9406
115	10:39:35.222	TestOne 1-656	HTTP Request	11184		6792	139	11183	9497
116	10:39:35.095	TestOne 1-207	HTTP Request	11329		6965	139	11329	9611
117	10:39:35.343	TestOne 1-958	HTTP Request	11129		3378	139	11108	9378
118	10:39:35.216	TestOne 1-642	HTTP Request	11302		6402	139	11302	9504
119	10:39:35.115	TestOne 1-244	HTTP Request	11407		5454	139	11406	9607
12	10:39:35.315	TestOne 1-891	HTTP Request	9953		7217	139	9953	9394
120	10:39:35.097	TestOne 1-417	HTTP Request	11431		4594	139	11431	9630
121	10:39:35.108	TestOne 1-432	HTTP Request	11430		6692	139	11430	9615

Scroll automatically? Child samples? No of Samples 1000 Latest Sample 33784 Average 19080 Deviation 5767

请求测试展示

Label	# Samples	Average ↓	Median	90% Line	95% Line	99% Line	Min	Maximum	Error %	Throughput	Received KB/s...	Sent KB/sec
HTTP Request	1000	14594	14598	22586	23641	24505	760	26574	0.00%	36.8/sec	227.83	4.99
TOTAL	1000	14594	14598	22586	23641	24505	760	26574	0.00%	36.8/sec	227.83	4.99

Settings Graph



Aggregate Graph

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/...	Sent KB/sec	Avg. Bytes
HTTP Request	1000	19080	9513	33784	5767.01	0.00%	29.4/sec	181.51	3.99	6322.7
TOTAL	1000	19080	9513	33784	5767.01	0.00%	29.4/sec	181.51	3.99	6322.7

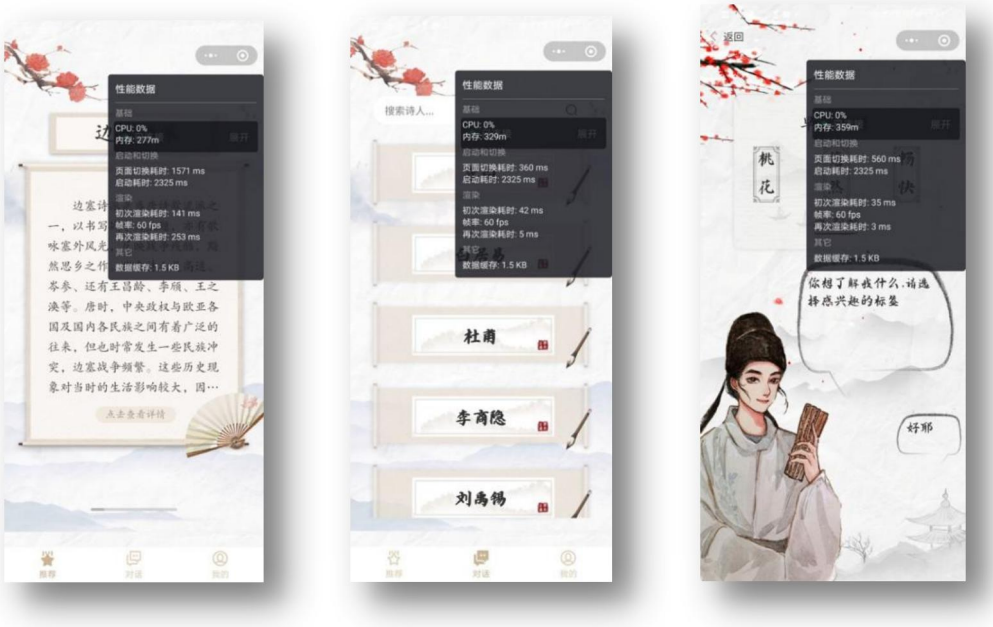
Summary Report

根据上述图片，服务器在单位时间内最大响应时间为 0.6s，平均响应时间不超过 0.5s，考虑到网络传输延迟时间，服务器对用户操作的响应时间能在 1 ~ 2s 内完成。

此外，服务器也测试了 5000 个线程同时进行，50000 个线程在 5s 内依次发出等模拟大数据量用户同时访问系统的场景，得出结果：tomcat 集群 + nginx 进行负载均衡可以较好的处理多线程及高并发问题。团队目前预计系统可以承受十万名用户在 5s 内依次请求，以及一万名用户同时发出请求。服务端性能测试情况良好。

5.3.3 小程序端性能测试

团队利用微信小程序官方文档中的性能测试工具 Trace 对小程序端进行性能测试。



通过上图所示的性能面板可以看出，「古人言」小程序初启动耗时在 2.3s 以内，网络请求的平均耗时为 0.5s，渲染和响应时间能保持在 1s 以内，页面切换时间能保持在 1.5s 以内，且不占用过多系统内存，基本符合当下用户的体验要求。

6.产品运营和发展

6.1 产品后期优化

1. 目前后台数据库收录诗词 10 万余首，由于中国古诗词的数量远不止于此，团队致力于继续不断收集古诗词并整理与之有关的诗人经历，真正带领用户跨越千年与古人进行对话。

2. 由于比赛时间有限，目前画师暂时完成部分诗人的**原创水墨画形象**，团队计划将该小程序推广后，征集号召喜爱绘画的用户，创作自己心中所想诗人形象。

6.2 未来展望

1. 弘扬诗词传统文化是每位中国人应尽的义务，团队计划在「古人言」小程序进一步推广后，进一步开发类似于诗词分享社区的模块，使用户在与古人对话后，能与身边热爱古诗词文化的其他人共同分享对话感受与心情，能记录心中感悟抑或是挥洒文笔写出属于自己的诗词。

2. 根据用户反馈，不断优化沉浸式对话和情景式交互方式，后期团队也将会考虑结合 VR、AR 等技术，致力于为用户打造与古人交流对话的平台，不断提高用户体验感。

3. 后续将会根据用户反馈进行 UI 的优化。

4. 做更深层次的市场调研与数据分析，使得用户需求得到最好的解决，用户数据得到最大化利用。

7. 团队分工

成员	团队分工
成员 A	利用爬虫整理后台数据、服务器环境部署、后端开发
成员 B	后端开发、数据库设计、测试、维护、系统设计
成员 C	小程序前端开发、测试、维护、系统设计、视频设计
成员 D	部分图像和元素设计、文案设计、用户体验与收集反馈