<UniGPT>

软件架构文档

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <12/5/2024> | <1.0> | <建立各架构内容> | <陈启炜，韦东良，黄峻涛，叶懿芯> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 参考资料 4

2. 用例视图 4

3. 逻辑视图 4

3.1 概述 4

3.2 在构架方面具有重要意义的设计包 4

4. 进程视图 4

5. 部署视图 4

6. 实现视图 5

7. 技术视图 5

8. 数据视图（可选） 5

9. 核心算法设计（可选） 5

10. 质量属性的设计 5

软件架构文档

# 简介

## 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面作出的重要决策。

1. 指导开发和设计过程。为开发团队提供一个清晰的指导，帮助团队成员理解系统的整体结构和设计原则，从而更好地规划和执行开发工作。

2. 交流沟通：通过本文档，开发人员、设计人员、测试人员等可以更好地了解系统设计和功能需求，避免信息传递和理解上的偏差。

3. 支持维护和扩展：软件架构文档记录了系统的整体设计和结构，有助于后续系统的维护和扩展工作。通过了解系统架构和设计原则，便于开发人员更有效地修改、扩展或优化。

4. 评审和审查：作为评审和审查的重要依据，帮助团队成员和管理层对系统的设计和实现进行评估和监控，确保系统的质量和符合性。

## 参考资料

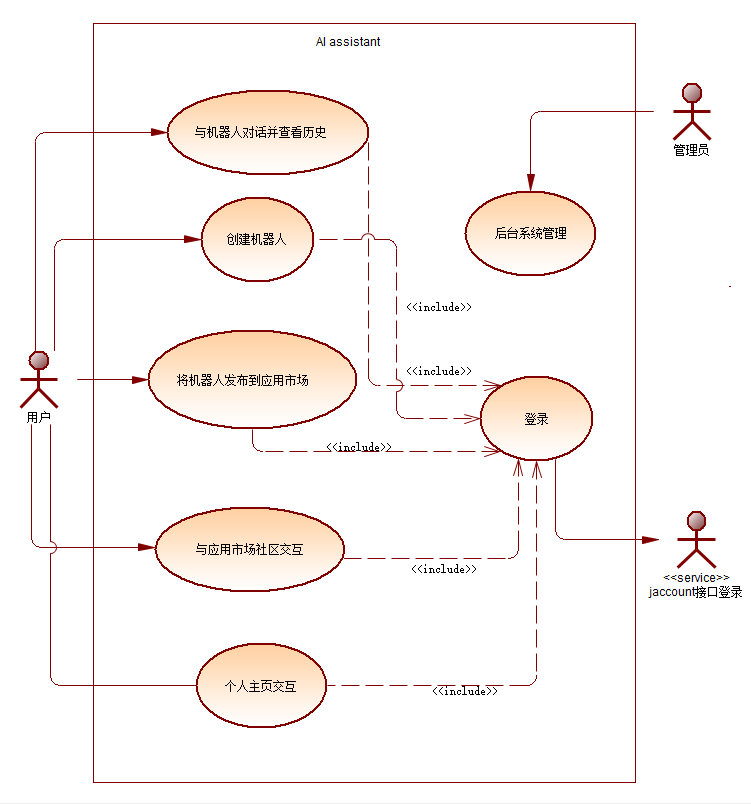
【REACT教程】<https://react.docschina.org/learn/describing-the-ui>

【SpringBoot教程】https://www.cnblogs.com/toutou/p/9650939.html

【SpringBoot教程】<https://blog.csdn.net/weixin_64972949/article/details/131107879>

【Mysql教程】https://aistudio.baidu.com/projectdetail/5064334?ad-from=6625

# 用例视图



**1. 与机器人交互并查看对话历史**

**1.1 简介**

这个用例中，用户可以与机器人发起新的对话，查看对话历史，进入特定的历史继续对话。

**1.2 事件流**

1.2.1 基本流

用例开始于用户进入机器人的对话页面。

1. 系统要求用户指定后续的操作。（发起新的对话，查看对话历史，进入特定的历史继续对话）

2. 用户指定操作后，下面的子流之一将被执行。

用户选择发起新的对话，那么发起新的对话子流被执行。

用户选择查看对话历史，那么查看对话历史子流被执行。

用户选择进入特定的历史进行对话，那么进入特定的历史子流被执行。

1.2.1.1 发起新的对话

1. 用户在聊天页侧边栏选择发起新的对话。

2. 与机器人对话子流被执行。

1.2.1.2 查看对话历史

1. 用户通过主页的“最近使用”标签或者聊天页的侧边栏可以查看对话的历史。

1.2.1.3 进入特定的历史

1. 用户点击主页的“最近使用”标签下的机器人图标，或者点击聊天页的侧边栏可以进入特定的 历史。

2. 与机器人对话子流被执行。

1.2.1.4 与机器人对话

1. 用户在聊天页面，通过底部的消息输入框，可以与机器人进行对话。

2. 用户可以点击点击消息输入框左侧的表单按钮，通过表单生成格式化的消息。

1.2.2 备选流

1.2.2.1 保存用户的对话上下文

用户在与机器人对话的过程中退出系统，系统将保留用户与机器人的对话记录，并计入历史，当用 户下一次使用系统时可以查询。

1.2.2.2 保存用户与机器人对话的表单

用户在编辑与机器人对话的表单时退出系统，系统将保留最新的表单，当用户下一次使用该机器人 时，可以恢复和查看表单。

**2. 创建机器人**

**2.1 简介**

这个用例中，用户可以创建定制化的机器人。

**2.2 事件流**

2.2.1 基本流

用例开始于用户点击主页的创建机器人按钮。

1. 子流填写机器人的基本信息被执行。

2.2.1.1 填写机器人的基本信息

1. 系统弹出表单，提示用户输入机器人的名称和描述，并选择机器人的头像。

2. 用户选择是否定义机器人的提示列表，如果用户选择是，子流填写用户的提示列表被执行；如 果否，子流提交创建机器人表单被执行。

2.2.1.2 填写用户的提示列表

1. 用户可以添加一个或者多个的提示词和内容。

2. 子流提交创建机器人表单被执行。

2.2.1.3 提交创建机器人表单

1. 用户点击表单的提交按钮，若提交成功，系统将把提交信息反馈给用户。

2.2.2 备选流

2.2.2.1 保存用户填写的创建机器人表单

在用户填写创建机器人表单的任何时候，用户可能选择保存表单而不是提交表单，此时“提交”步 骤被替换为：上方弹出已保存为草稿的提示，用户保存的表单存储在系统中，在用户下一次创建机器人时恢复。

**3. 将机器人发布到应用市场**

**3.1 简介**

这个用例中，用户可以将自己创建的机器人发布到应用市场，供更多人使用。

**3.2 事件流**

3.2.1 基本流

用例开始于用户填写创建机器人时勾选发布到应用市场，或者在“我创建的”机器人中选择发布到 应用市场。

1. 子流填写发布机器人到应用市场表单被执行。

3.2.1.1 填写发布机器人到应用市场表单

1. 系统弹出表单，提示用户输入发布到应用市场的详细介绍和图片。

2. 子流提交发布机器人到应用市场表单被执行。

3.2.1.2 提交发布机器人到应用市场表单

1. 用户点击表单的提交按钮，若提交成功，系统将把提交信息反馈给用户，用户的机器人将进入 应用市场，供其他用户使用和评价。

3.2.2 备选流

3.2.2.1 用户取消发布到应用市场

在用户填写发布机器人到应用市场的表单时，点击取消按钮，系统将终止用户机器人的发布。

3.2.2.2 用户重复发布相同的机器人到应用市场

如果用户发布已经发布的机器人到应用市场，系统将终止此次发布并返回给用户提示信息。

**4. 与应用市场社区交互**

**4.1 简介**

在这个用例中，用户可以对应用市场上的机器人进行查询、点赞、评论、收藏。

**4.2 事件流**

4.2.1 基本流

用例开始于用户进入系统的应用市场界面。

1. 系统要求用户指定后续的操作。（查询机器人，点赞机器人，评论机器人，收藏机器人）

2. 用户指定操作后，下面的子流之一将被执行。

用户选择查询机器人，那么查询机器人子流被执行。

用户选择点赞机器人，那么点赞机器人子流被执行。

用户选择评论机器人，那么评论机器人子流被执行。

用户选择收藏机器人，那么收藏机器人子流被执行。

4.2.1.1 查询机器人

1. 在应用市场界面，用户通过搜索框查询机器人。

2. 收到用户的请求后，系统对数据库进行检索，返回用户搜索结果。

4.2.1.2 点赞机器人

1. 在机器人的详情页面，用户点击心形按钮，可以对当前机器人点赞，表示对机器人创建者的赞 同。

4.2.1.3 评论机器人

1. 在机器人的详情页面，用户在评论的输入框中输入内容，点击发送按钮，其他人将看到评论的 内容。

2. 在其他用户的评论下，用户可以对该条评论点赞、回复，参与社区讨论。

4.2.1.4 收藏机器人

1. 在机器人的详情页面，用户点击星形按钮，可以将当前机器人收藏。

2. 用户收藏之后，可以从主页面中“收藏”标签下找到收藏的机器人。

4.2.2 备选流

4.2.2.1 用户查询不存在的机器人

1. 用户尝试查询不存在的机器人，在检索数据库确认不存在之后，系统将返回用户机器人不存在 的提示。

4.2.2.2 用户查询不存在的机器人

1. 用户尝试查询不存在的机器人，在检索数据库确认不存在之后，系统将返回用户机器人不存在 的提示。

**5. 个人主页交互**

**5.1 简介**

这个用例中，用户可以修改个人信息，查看创建的机器人，查看收藏的机器人。

**5.2 事件流**

5.2.1 基本流

用例开始于用户点击右上角的头像，进入个人主页。

1. 系统要求用户指定后续操作。

用户选择修改个人信息，那么修改个人信息子流被执行。

用户选择查看创建的机器人，那么查看创建的机器人子流被执行。

用户选择查看收藏的机器人，那么查看收藏的机器人子流被执行。

5.2.1.1 修改个人信息

1. 用户可以修改用户名和头像。

5.2.1.2 查看创建的机器人

1. 通过个人主页的“已创建”标签，可以查看和使用已经创建的机器人。

5.2.1.3 查看收藏的机器人

1. 通过个人主页的“收藏”标签，可以查看和使用已经收藏的机器人。

**6. 登录**

**6.1 简介**

在这个用例中，用户通过jaccount账号登录系统。

**6.2 事件流**

6.2.1 基本流

用例开始于用户首次进入系统。

1. 跳转jaccount登录子流被执行。

6.2.1.1 跳转jaccount登录

1. 用户首次进入系统，系统请求jaccount认证服务器，跳转到jaccount登录界面，用户输入账号和 密码。

2. 提交登录子流被执行。

6.2.1.2 提交登录

1. 用户点击提交按钮。

2. jaccount服务器进行认证，如果用户登录成功，将允许用户进入系统。

6.2.2 备选流

6.2.2.1 账号或密码错误

1. 用户在进行jaccount账号登录时，输入了不匹配的账号或密码。

2. jaccount认证服务器返回失败结果，页面跳转回开始页面，拒绝用户进入系统。

**7. 系统管理**

**7.1 简介**

在这个用例中，管理员通过系统登录进行系统管理操作，包括维护后台服务器、维护数据库等。

**7.2 事件流**

7.2.1 基本流

用例开始于管理员成功登录系统。

1. 管理员选择后续的操作（如维护后台服务器、维护数据库、风险控制等）。

2. 管理员选择指定的操作后，后续的子流将被执行。

管理员选择维护后台服务器时，维护后台服务器子流被执行。

管理员选择维护数据库时，维护数据库子流被执行。

7.2.1.1 维护后台服务器

1. 监控服务器状态，利用工具进行分析。

2. 性能优化，根据监控数据对服务器进行性能优化，包括调整系统参数，优化数据库配置等。

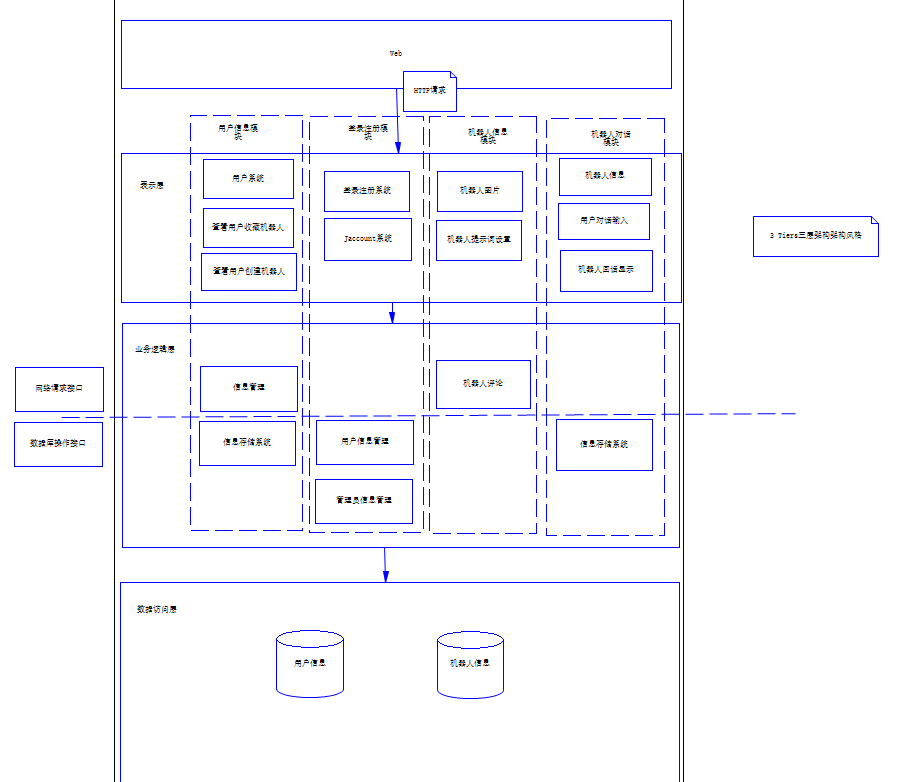
3. 安全管理，定期审查安全日志，保证服务器的安全性。

7.2.1.2 维护数据库

1. 管理员定期备份数据库，确保数据的安全性和可恢复性。

2. 优化数据库性能，如索引优化、查询优化、表空间管理等。

# 逻辑视图



## 概述

采用3 Tiers三层架构的架构风格。通过表示层、业务逻辑层、数据访问层的划分将前后端分离。将用户信息、登录注册、机器人信息、机器人对话等不同业务划分，执行不同的业务逻辑和对数据库操作。

表示层，负责向用户呈现界面，并接收用户请求发送给业务逻辑层。界面为用户呈现个人主页、应用市场、机器人信息、机器人对话界面等页面。

业务逻辑层，负责执行业务逻辑以处理用户请求，并调用数据访问层提供的持久性操作。包括用户登陆与注册、创建与修改个人资料信息、创建与修改机器人、向应用市场添加机器人、处理用户发来的对话并调用相应api进行回复、创建用户对话历史等业务逻辑。

数据访问层，负责执行数据库持久性操作，主要使用mysql数据库存储用户信息（包括对话历史）和机器人信息。

## 在构架方面具有重要意义的设计包

本项目采用Spring Boot 框架，以简化 Spring 应用程序的搭建和配置过程。重要的设计包如下：

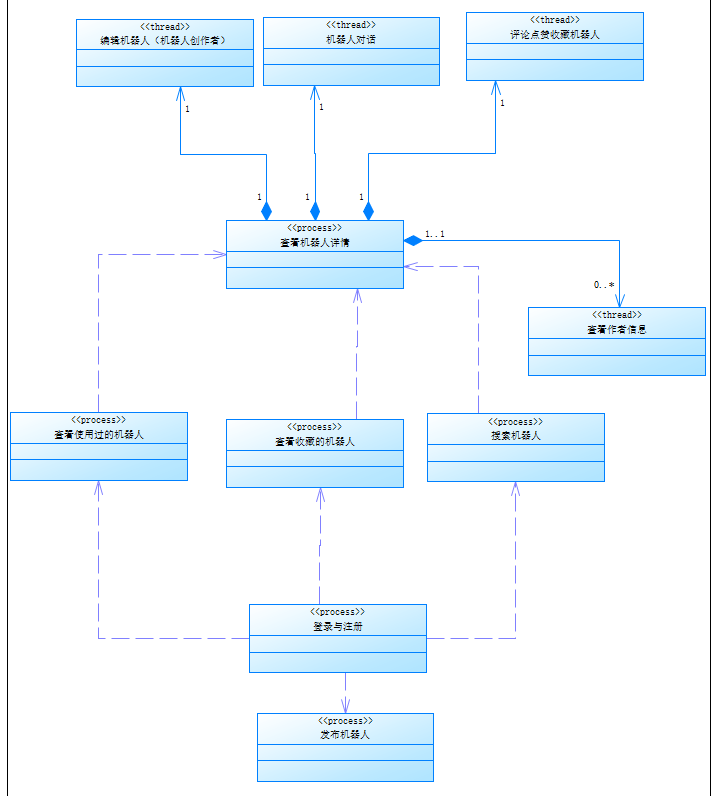
1. Config：配置类，提供基本的登录检测、跨域支持和web socket技术支持

2. Controller: 控制器类，负责和前端交互，处理 HTTP 请求

3. Service: 服务类，负责具体的业务逻辑

4. repository：每个repository 都对应数据库内的一张表

# 进程视图

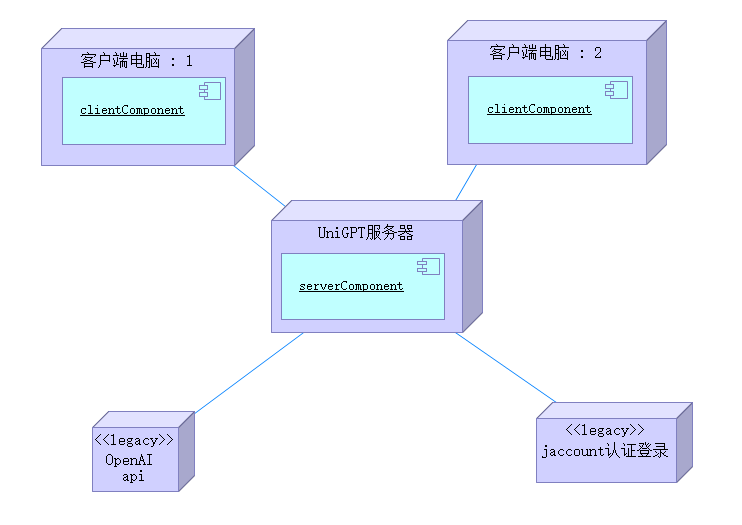


轻量级进程：编辑机器人，机器人对话，评论点赞收藏机器人，发布机器人，查看作者信息

重量级进程：查看机器人详情及其并行线程，登录与注册及其并行线程

通过HTTP/WEBSOCKET进行前后端通信。

# 部署视图



客户端电脑

用户使用电脑通过Web浏览器访问UniGPT系统，通过http和Socket方式与UniGPT建立联系。客户端电脑与UniGPT服务器之间的通信通过HTTP协议进行，用于向服务器发送请求并接收响应。通过Socket协议，客户端电脑可以与UniGPT服务器进行实时的双向通信。

UniGPT服务器

UniGPT服务器作为系统的核心组件，通过HTTP与各个客户端连接，并为其提供服务。UniGPT服务器与Jaccount系统服务器连接，用于验证用户登录信息的有效性。通过Socket与OpenAI服务器连接，以调用GPT服务。

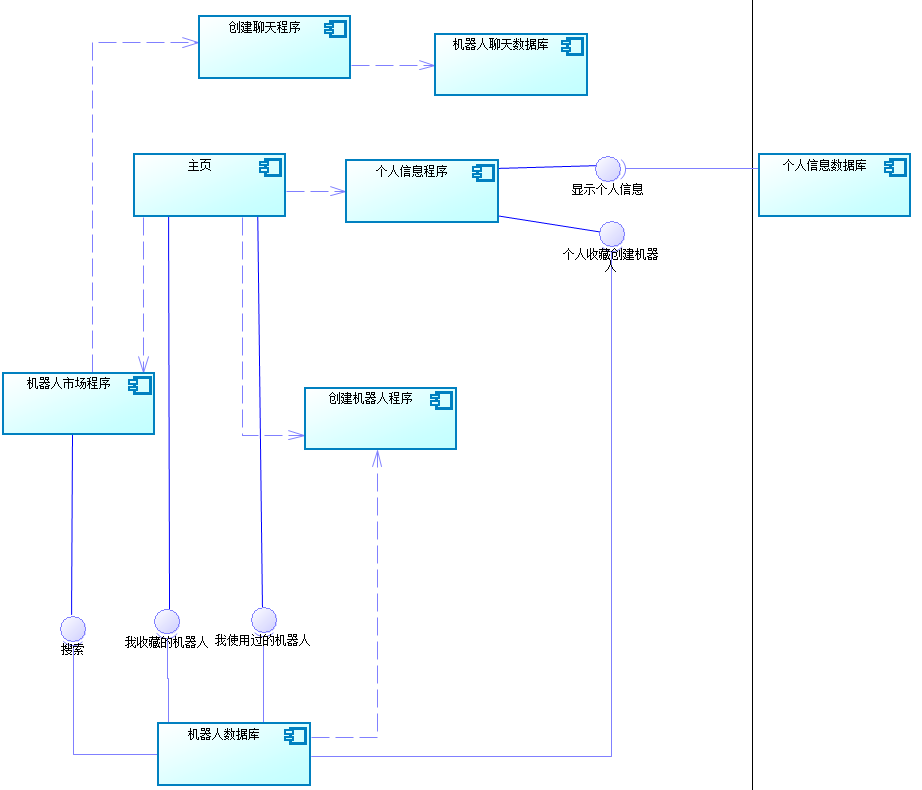
Jaccount legacy

Jaccount legacy是上海交通大学Jaccount系统的服务器，用于处理用户登录信息。UniGPT服务器通过与Jaccount legacy的连接，可以验证用户登录信息的有效性，以确保只有经过授权的用户才能访问UniGPT系统。

OpenAI legacy

OpenAI legacy是OpenAI的服务器，提供GPT服务。UniGPT服务器通过API调用OpenAI服务器上的GPT服务，以获取生成文本的能力。通过与OpenAI legacy的连接，UniGPT服务器可以为用户提供高质量的自然语言生成服务。

# 实现视图



子系统：机器人数据库，个人信息数据库（外接Jaccount系统数据库），机器人聊天数据库（外接OpenAI等大模型端）等

整体结构：用户登录主页后可跳转至不同子页面，通过后端连接到不同数据库进行信息处理等操作

# 技术视图

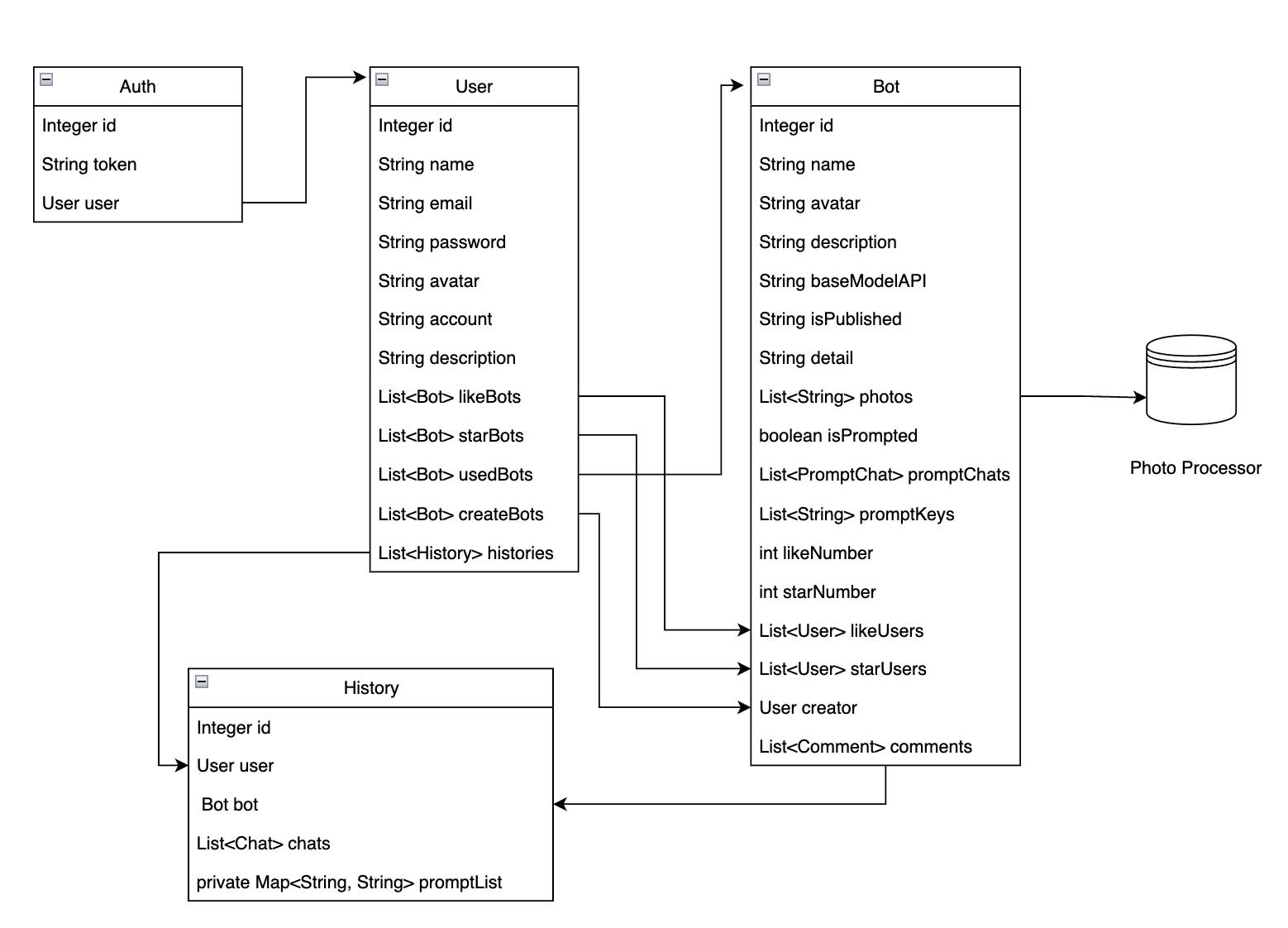
前端：html, css, JavaScript, React框架, Material UI组件库, web socket, VS Code

后端：Spring, Spring Boot, Maven, web socket, IDEA

数据库: MySQL

测试: APIFox

# 数据视图（可选）

**数据流和处理

在这个系统中，数据流涉及用户与机器人的互动，例如：

用户可以创建、喜欢、收藏或评论机器人，这些操作都会影响相应的数据存储和关系映射。

机器人可以存储和管理用户生成的内容，如评论和喜欢的历史记录，这些数据可以用于进一步的分析和用户体验优化。

此外，图中的“Photo Processor”表示外部服务，用于处理机器人的照片数据。

应用实例

在实际应用中，这样的数据视图可以帮助开发团队确保数据一致性和系统的可扩展性。例如，当增加新的功能如机器人推荐系统时，开发者可以参照这个数据模型来设计数据交互逻辑和更新数据库模式。

总之，这个数据视图提供了一个全面的视角来观察和管理系统中的数据流动和存储。通过这种方式，开发者可以更有效地管理复杂系统的需求，提高系统的性能和用户满意度。

# 质量属性的设计

1. 性能

Web Socket技术应用：采用 Web Socket 技术优化实时通讯，减少服务器响应时间并提高聊天页面的实时更新能力，实现更流畅的用户交互体验。

React 框架应用：使用 React 框架以提升网页的重绘速度和整体性能。React 的虚拟DOM机制可以最小化实际DOM操作的次数，大大加快了视图更新速度。

懒加载和代码分割：实现懒加载和代码分割技术，按需加载资源，减少首次加载时间，提高应用的启动速度和性能。

2. 可扩展性

前后端模块化架构：明确前后端分离，前端组件化实践（采用 React 现代框架），后端遵循清晰的分层架构（ Spring Boot 框架），易于维护和扩展。

模块的原子性：确保各模块的高内聚低耦合，模块之间通过定义良好的API进行通信，增强模块间的独立性，方便模块的复用和重组。

导航结构优化：利用前端路由（React Router）优化应用导航，减少代码冗余，使路由管理更为简洁和灵活。

3. 可靠性

错误检测与异常处理：在关键的前后端交互点实施严格的错误检测和异常处理机制，确保系统稳定运行，并对可能的运行时错误给出及时反馈。

机部署策略：通过多机部署应用服务器和数据库，实施负载均衡和故障转移策略，确保在主服务器出现故障时，备用服务器可以接管工作，保证服务的高可用性和数据的持续性。

4. 易用性

多语言支持与本地化：设计用户界面时，提供多语言支持和本地化选项，以适应全球用户的需求。

系统功能提示信息：在用户界面的关键操作点提供清晰的帮助提示和文档链接，帮助用户理解和使用系统功能，减少操作误区。

5. 可移植性

跨平台兼容性：开发时使用跨平台技术，确保软件能够在不同操作系统（Windows、MacOS、Linux）和设备（PC、移动设备）上运行。

适应不同分辨率和设备：设计响应式用户界面，自动适应各种屏幕尺寸和分辨率，提高应用在各种设备上的表现和用户体验。