<个人碳排放积分系统>

软件架构文档

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 23.5.13 | <1.0> |  | 吴天宇、陈泽铭、陈家豪 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

2. 系统概述 4

3. 系统模块 4

4. 系统交互流程 5

5. 技术架构 6

6. 数据库设计 7

7. 核心算法设计 8

8. 质量属性的设计 9

软件架构文档

# 简介

## 目的

本设计架构文档旨在定义和说明公共出行个人碳排放积分系统的设计方案。该系统旨在鼓励用户使用公共交通工具，减少个人碳排放量，并通过碳积分的奖励和兑换机制促进用户的参与和行为改变。

## 范围

本文档将涵盖公共出行个人碳排放积分系统的整体设计概念和技术实现。它将描述系统的主要模块和功能，并提供相应的技术选择和解释。此外，还将介绍系统的交互流程、数据库设计以及系统部署和运维方案。

## 定义、缩略词和缩写词

公共出行个人碳排放积分系统：指本文档所描述的软件系统，用于管理用户的碳排放数据、发放积分并支持积分的兑换和交易。

碳积分：用于衡量个人碳排放减少量的计量单位，用户通过乘坐公共交通工具和采取减排措施来获得积分。

出行工具公司：提供公共交通工具的公司或组织，例如地铁、公交等。

认证机构：负责验证和监督用户碳排放数据的独立机构。

用户：使用公共出行个人碳排放积分系统的个人或实体。

## 参考资料

暂无

# 系统概述

## 目标和功能

目标：该系统的目标是促进绿色出行，减少碳排放，提高公共交通的使用率，并鼓励用户采取环保行为。

功能：系统将实现以下功能：

碳排放配额购买：出行工具公司向认证机构购买碳排放总量配额。

积分奖励机制：认证机构根据购买的碳排放配额，将其转化为积分，并通过活动和奖励发放给乘坐公共交通工具的用户。

积分兑换和实物奖励：用户可以通过积攒积分，向认证机构兑换实物奖励。

反馈机制：认证机构根据收集到的积分数据，向出行工具公司提供反馈，以鼓励他们采取更多的环保措施。

用户画像和大数据分析：结合用户的个人信息和出行数据，进行用户画像和大数据分析，以实现更精准的推广和奖励策略。

互联网运营体系：结合常见的互联网运营体系，如用户等级、积分和成就系统等，增加用户参与度和激励效果。

## 参与角色

出行用户：使用公共交通工具进行出行的个人用户。

出行工具公司：公交车、地铁等公共交通工具的运营公司。

认证机构：负责测量、认证和管理碳排放数据，并发放碳积分的机构。

# 系统模块

## 用户管理模块

功能：处理用户注册、登录、个人信息管理等任务。

包括用户身份验证、访问控制和权限管理。

提供用户画像和个性化设置。

## 碳排放配额管理模块

功能：管理碳排放配额的购买、分配和转换。

出行工具公司向认证机构购买碳排放总量配额，用于覆盖其运营过程中的碳排放。

管理和追踪配额的使用情况，并根据购买的配额将一部分转化为积分。

## 积分奖励模块

功能：根据购买的碳排放配额，将一部分转化为积分，并通过活动和奖励发放给乘坐公共交通工具的用户。

根据用户的出行行为和贡献，计算和发放相应的积分奖励。

处理用户积分的累积和使用，提供积分查询和兑换功能。

## 奖励兑换模块

功能：允许用户使用积分兑换实物奖励。

管理奖励目录和库存，确保奖励的可用性和有效性。

处理用户的兑换请求，扣除相应的积分，并安排奖励的发放和配送。

## 数据分析和用户画像模块

功能：基于用户的个人信息和出行数据，进行用户画像和大数据分析。

收集和存储用户的出行数据和行为信息。

进行数据清洗、预处理和分析，以生成用户画像和行为模式。

提供个性化推荐和奖励策略，以鼓励用户选择绿色出行方式。

## 反馈和监控模块

功能：提供反馈机制，监控系统运行情况。

认证机构根据收集到的积分数据，向出行工具公司提供反馈和价格反馈，以影响下次购买的碳排放总量。

收集用户的反馈和意见，以改进系统功能和用户体验。

监控系统性能和稳定性，及时发现和解决潜在的问题。

## 第三方接口模块

功能：与外部系统和服务进行集成。

包括支付接口，用于处理积分兑换时的支付操作。

可与公共交通工具的数据接口进行对接，获取实时的出行信息和碳排放数据。

# 系统交互流程

公共出行个人碳排放积分系统的设计中，系统交互流程部分描述了系统各个组件之间的交互流程和消息传递方式。

## 用户注册和登录的交互流程

1.用户通过前端界面填写注册信息，并发送注册请求到后端服务器。

2.后端服务器验证注册信息的有效性，创建用户账号，并返回注册成功的响应。

3.用户使用注册的账号和密码进行登录，发送登录请求到后端服务器。

4.后端服务器验证用户的身份信息，若验证通过，则返回登录成功的响应，用户获得访问权限。

## 记录出行和计算碳排放的交互流程

1.用户在前端界面记录出行行为，包括出行工具、行程距离等信息。

2.用户提交出行记录请求到后端服务器。

3.后端服务器接收到请求，根据用户提供的出行数据，计算出行的碳排放量。

4.后端服务器将计算得到的碳排放量与用户的出行记录关联，并存储到数据库中。

## 积分奖励和兑换的交互流程

1.后端服务器根据设定的积分奖励规则，计算用户应获得的积分奖励。

2.后端服务器将计算得到的积分奖励与用户的账号关联，并存储到数据库中。

3.用户可以在前端界面查看自己的积分余额，并选择兑换实物奖品。

4.用户提交积分兑换请求到后端服务器。

5.后端服务器验证用户的积分余额和兑换规则，若兑换条件满足，则进行兑换操作，并更新用户的积分余额和奖品库存。

## 碳排放配额购买和认证的交互流程

1.出行工具公司向认证机构发送碳排放配额购买请求，包括购买数量和支付信息。

2.认证机构接收到购买请求，验证支付信息和配额库存，若满足条件，则进行配额购买操作，并更新配额库存和支付记录。

3.出行工具公司获得购买的碳排放配额，并将购买记录与自己的账号关联。

# 技术架构

## 前端应用程序

技术栈：web端使用react框架，移动端使用flutter框架

功能：

用户注册和登录功能。

显示用户积分、兑换奖励的界面等。

## 后端服务

技术栈：采用可扩展的后端开发框架Java SpringBoot

功能：

处理用户请求，包括用户注册、登录、积分查询、奖励兑换等功能。

管理碳排放配额和积分，包括购买、分配和转换。

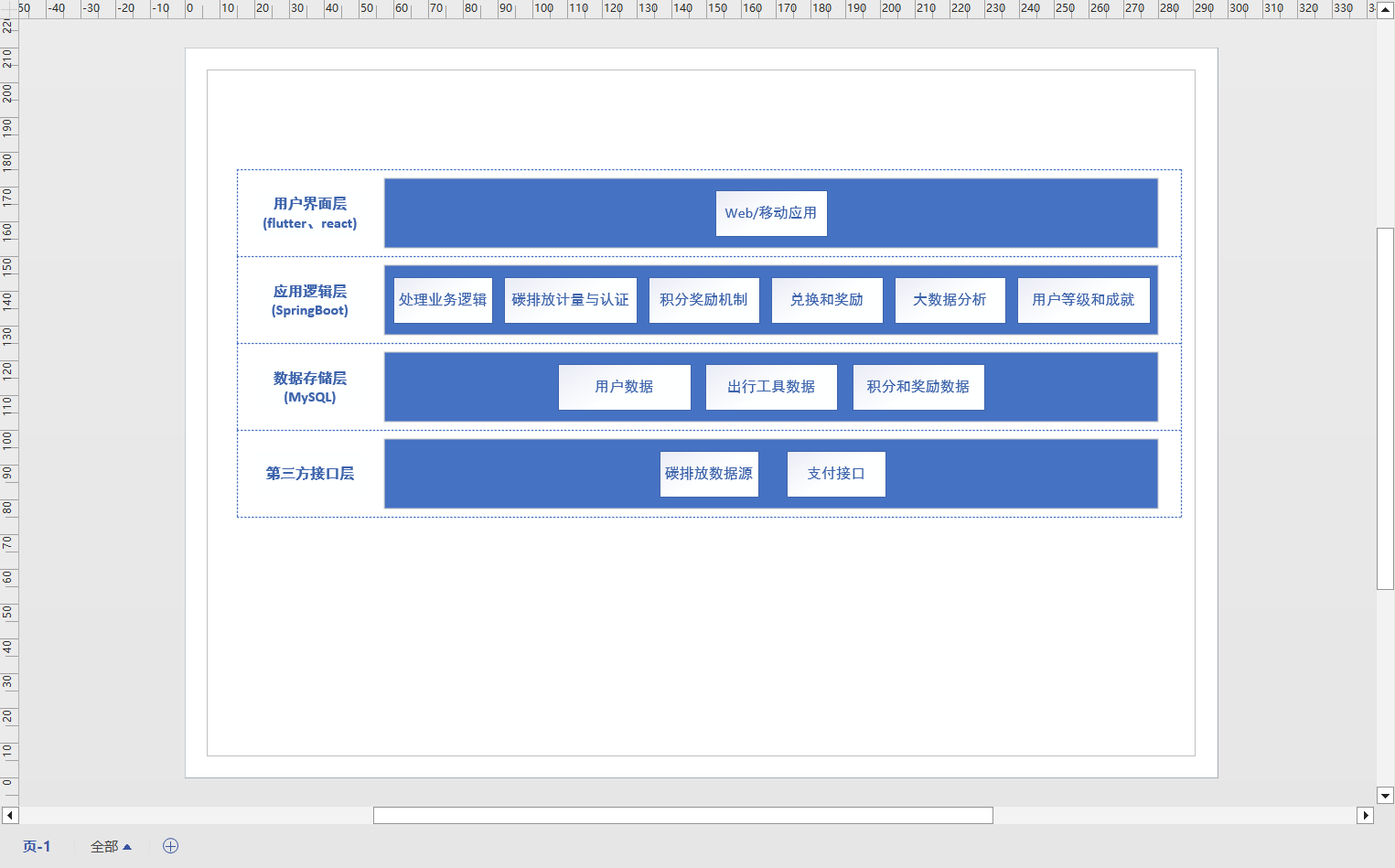
生成用户奖励。

提供API接口供前端应用程序调用。

## 数据库

存储系统所需的各种数据，包括用户信息、碳排放数据、积分和奖励信息等。

使用关系型数据库MySQL



# 数据库设计

公共出行个人碳排放积分系统的数据库设计是关键的一部分，它负责存储系统所需的各种数据，并提供数据的可靠性和一致性。

## 用户数据表

包括用户ID、用户名、密码、邮箱、联系方式等基本信息。

还可包含用户画像标签，如年龄段、性别、出行偏好等。

用于用户注册、登录和个人信息管理。

## 概述出行数据表

记录用户的出行行为数据，包括出行时间、出行工具、行程距离、碳排放量等。

与用户数据表通过外键关联，建立用户和出行数据的关联关系。

用于碳排放测量和认证，以及用户行为分析。

## 碳排放配额表

记录出行工具公司购买的碳排放配额信息，包括公司ID、购买数量、购买时间等。

用于管理碳排放配额的购买、分配和转换。

## 积分奖励表

记录用户的积分奖励信息，包括用户ID、积分数量等。

与用户数据表通过外键关联，建立用户和积分奖励的关联关系。

用于积分的累积、使用和兑换。

## 奖励目录表

记录系统提供的实物奖励信息，包括奖励ID、奖励名称、所需积分、库存量等。

用于奖励兑换时的验证和扣除积分操作。

## 概述反馈意见表

记录用户的反馈意见，包括用户ID、反馈内容、反馈时间等。

用于用户与系统管理员的沟通和问题解决。

# 核心算法设计

## 碳排放量计算算法

该算法的目的是计算用户在使用公共交通工具时所产生的碳排放量。为了实现计算的精准性，可以考虑使用以下变量进行计算：

用户乘坐的公共交通工具类型和座位数；

公共交通工具行驶的路线和距离；

公共交通工具的运行时间和时段；

环境因素，如气温、湿度、风速等。

基于以上变量，可以使用机器学习算法对碳排放量进行预测和计算，比如可以采用基于多元线性回归模型的方法。这种方法可以利用历史数据建立一个回归模型，通过输入新的变量值，预测出用户碳排放量。同时，还可以考虑将计算结果与实际数据进行对比，通过反馈机制不断优化算法。

## 积分激励算法

该算法的目的是激励用户更多地使用公共交通工具，以减少碳排放量。为了实现积分激励的个性化和精准性，可以考虑采用以下策略：

根据用户的出行规律、消费行为和地理位置等信息，进行个性化推送，提供出行建议和奖励；

设计一套积分奖励机制，将用户的碳排放量和奖励积分进行绑定，让用户获得足够的积分后可以换取具体实物或者服务奖励；

建立用户积分等级制度，让用户可以通过不断积攒积分，提升自己的等级和享受更多的权益和优惠。

以上两个算法可以通过人工智能技术和大数据分析等方法进行实现和优化，从而实现公共出行个人碳排放积分系统的智能化处理和优化。

# 质量属性的设计

## 可靠性

异常处理：设计健壮的异常处理机制，包括错误日志记录、错误信息返回给用户等，以保证系统的可靠性和容错性。

数据完整性：使用事务管理和数据校验机制，确保数据的完整性和一致性，避免数据丢失或损坏。

## 性能

响应时间：设计高效的算法和数据结构，以减少计算和查询时间，提高系统的响应速度。

缓存机制：使用缓存技术，将频繁访问的数据缓存起来，以提高系统的响应速度和吞吐量。

## 可维护性

代码可读性：编写清晰、易读、易理解的代码，以方便团队成员维护和修改代码。

文档和注释：提供详细的文档和注释，描述系统的设计、功能和模块之间的关系，以便于团队的沟通和维护。

## 安全性

身份验证和授权：设计安全的身份验证和授权机制，确保只有授权的用户可以访问系统的敏感功能和数据。

数据加密：对敏感数据进行加密，以保护用户的个人信息和系统的数据安全。