作业收集平台PaperStack  
系统设计文档

版本 1

# 版本变更记录

《系统设计》版本1为初始版本。

# 目录

[作业收集平台PaperStack 系统设计文档 1](#_Toc24295520)

[版本变更记录 2](#_Toc24295521)

[目录 3](#_Toc24295522)

[文档目标 4](#_Toc24295523)

[概要设计 4](#_Toc24295524)

[软件架构 4](#_Toc24295525)

[硬件条件 5](#_Toc24295526)

[开发环境及工具 5](#_Toc24295527)

[详细设计 5](#_Toc24295528)

[通用设计 6](#_Toc24295529)

[登录子系统设计 6](#_Toc24295530)

[用户界面 6](#_Toc24295531)

[静态设计 6](#_Toc24295532)

[动态设计 6](#_Toc24295533)

[数据结构 6](#_Toc24295534)

[注册子系统设计 6](#_Toc24295535)

[用户界面 6](#_Toc24295536)

[静态设计 6](#_Toc24295537)

[动态设计 6](#_Toc24295538)

[数据结构 6](#_Toc24295539)

[个人信息子系统设计 7](#_Toc24295540)

[用户界面 7](#_Toc24295541)

[静态设计 7](#_Toc24295542)

[动态设计 7](#_Toc24295543)

[数据结构 7](#_Toc24295544)

[作业收集子系统设计 7](#_Toc24295545)

[用户界面 7](#_Toc24295546)

[静态设计 7](#_Toc24295547)

[动态设计 7](#_Toc24295548)

[数据结构 7](#_Toc24295549)

[平台管理子系统设计 7](#_Toc24295550)

[用户界面 7](#_Toc24295551)

[静态设计 7](#_Toc24295552)

[动态设计 7](#_Toc24295553)

[数据结构 7](#_Toc24295554)

[总结 7](#_Toc24295555)

# 文档目标

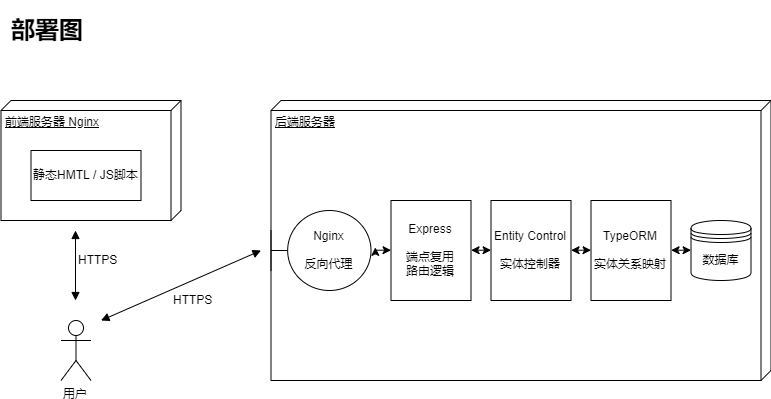
本文档为系统设计文档，目标是在上一阶段所提出的需求分析文档的基础上，进一步对系统进行设计，将系统需求转换为软件表示，描述软件的总体概貌。

本文档分概要设计和详细设计两部分。概要设计主要介绍系统的软件架构和硬件条件等，并将系统划分为子系统和模块，并指出模块所负责的用例；详细设计主要借助UML的类图和顺序图描述模块的静态设计和动态设计，并介绍用户界面和数据表的设计。

# 概要设计

本小节主要介绍系统的软件架构以及系统部署所需的硬件条件，并介绍开发系统所需的环境和工具。

## 软件架构



系统采用B/S架构，分为前端和后端两部分。前端负责呈现图形化用户界面并与用户进行视觉元素交互，以及将用户的数据发送至后端并呈现后端处理后的结果。后端则负责保存用户数据、处理前端请求和返回请求结果。

前端用户界面采用单页面应用程序（SPA）设计，采用TypeScript和React框架开发。代码编译为静态HTML文件和JavaScript文件，由前端HTTP服务器托管。用户通过浏览器访问前端HTTP服务器，获得静态HTML页面和在浏览器上运行的JavaScript脚本。当用户需要执行操作时，前端页面显示视觉元素，浏览器通过HTTP协议访问后端服务器，发送数据并获取操作结果，不需要经过前端HTTP服务器。

后端通过HTTP协议提供服务。用户需要执行操作时，浏览器根据操作的类型请求不同的后端HTTP端点。后端服务器根据HTTP的端点和参数来区别用户请求的服务，从数据库中获取数据，处理用户的请求并返回结果。后端服务器采用Express框架进行开发， 由TypeORM负责对象关系映射。数据库软件使用PostgreSQL。

实际部署时应当使用加密的HTTP协议。前端服务器直接由Nginx托管静态文件。后端服务器运行于Node.js环境中，并由Nginx反向代理到HTTP端口。

## 硬件条件

在系统试运行阶段，为满足需求分析报告提出的“25人同时浏览和下载和容纳 500人的信息存储”要求，若将前端服务器和后端服务器部署于同一台物理机上，则按每人浏览所需10Mbp带宽，产生数据500MiB进行粗略估算，物理机须有20Mbps进出口带宽和至少500GiB的HDD存储空间，处理器不少于2核心，基准速度不低于2.0GHz。

## 开发环境及工具

系统使用TypeScript语言开发，开发环境运行Node.js。受益于TS和Node.js的跨平台特性，系统可在任意主流操作系统中开发。编译器为标准TSC，代码格式化工具为ESLint，代码协作工具为Git。

团队使用Draw.io进行UML建模。

团队中推荐的代码编辑器为Visual Studio Code。

团队使用Adobe XD和Zepplin.io进行用户界面设计。

# 详细设计

本小节主要借助UML的类图和顺序图描述模块的静态设计和动态设计，并介绍用户界面和数据表的设计。

## 通用设计

## 登录子系统设计

### 用户界面

### 静态设计

### 动态设计

### 数据结构

## 注册子系统设计

### 用户界面

### 静态设计

### 动态设计

### 数据结构

## 个人信息子系统设计

### 用户界面

### 静态设计

### 动态设计

### 数据结构

## 作业收集子系统设计

### 用户界面

### 静态设计

### 动态设计

### 数据结构

## 平台管理子系统设计

### 用户界面

### 静态设计

### 动态设计

### 数据结构

# 总结