**智能简历解析系统**

**概要设计文档**

目录

[**智能简历解析系统** 1](#_Toc139661735)

[**概要设计文档** 1](#_Toc139661736)

[**第1章** 引言 3](#_Toc139661737)

[**1.1** 编写目的 3](#_Toc139661738)

[**1.2** 系统目标 3](#_Toc139661739)

[**1.3** 主要软件需求 3](#_Toc139661740)

[**1.4** 软件设计约束、限制 3](#_Toc139661741)

[**第2章** 数据设计 4](#_Toc139661742)

[**2.1** 数据对象和形成的数据结构 4](#_Toc139661743)

[**2.2** 文件和数据交叉索引 5](#_Toc139661744)

[**第3章** 体系结构设计 6](#_Toc139661745)

[**3.1** 数据和控制流复审 6](#_Toc139661746)

[**第4章** 主要模块设计 7](#_Toc139661747)

[**4.1** 基于 REST Framework 的业务实现 7](#_Toc139661748)

[**4.2** 基于Django+Vue的简历上传与信息提取模块 8](#_Toc139661749)

[**4.3** 基于Django+FastAPI+Vue的简历分析模块 9](#_Toc139661750)

[**4.4** 基于Django的简历信息提取 9](#_Toc139661751)

[（一） Django框架介绍 9](#_Toc139661752)

[（二） Django框架的架构设计 10](#_Toc139661753)

[**4.5** 基于RoBERTa模型的岗位匹配 11](#_Toc139661754)

[**第5章** 界面设计 13](#_Toc139661755)

[**5.1** 登录及注册界面 13](#_Toc139661756)

[**5.2** 简历和岗位上传界面 13](#_Toc139661757)

[**5.3** 上传信息查看界面 16](#_Toc139661758)

[**5.4** 岗位详情界面 17](#_Toc139661759)

[**5.5** 简历详情界面 17](#_Toc139661760)

[**5.6** 单个简历分析界面 18](#_Toc139661761)

[**5.7** 整体岗位匹配界面 18](#_Toc139661762)

# 引言

## 编写目的

该文档的目的是描述“众里寻他”网站的概要设计，主要内容包括功能简介，系统结构设计，模块设计和界面设计

## 系统目标

企业可通过简历信息、岗位信息录入功能，以文本、PDF格式输入上传面试者简历，系统会进行检测，以可视化的方式展现求职者的能力水平，工作变动是否频繁以及简历的可信程度等各项数据，并且使用合理的算法推测出求职者的岗位匹配度。同时能够让用户通过筛选功能，选定条件，从面试者中为企业推荐最合适的人选，招聘期间，该应用程序可以帮助企业高效地了解简历中求职者的实际素质，快速匹配合适的岗位，减少人力成本，提升招聘效率。

## 主要软件需求

该系统的主要功能包括：

1. 用户上传求职者简历，系统自动检测并呈现求职者能力水平，简历可信程度，工作变动频率，与岗位匹配程度。
2. 选定条件对求职者进行筛选。
3. 求职者信息查询。

## 软件设计约束、限制

本项目要求在Pycharm上进行开发，开发经费较少，开发时间为五个月左右。

# 数据设计

本部分主要描述软件所涉及的外部数据的结构描述。如果数据以数据库文件呈现，则描述表的名称和表字段结构；如果数据以外部文件形式呈现，则要描述文件的内部结构。

## 数据对象和形成的数据结构

“众里寻他”网页的主要数据结构如下：

1. User表：存储用户信息，如表2.1.1所示

| 列名 | 数据类型 | 允许为空 | 自增 | 主键 | 唯一键 |
| --- | --- | --- | --- | --- | --- |
| uid | int | 否 | 是 | 是 | 否 |
| hash\_code | varchar(255) | 是 | 否 | 否 | 是 |

表2.1.1 user表

2. Worker 表：存储简历信息，如表2.1.2所示

| 列名 | 数据类型 | 允许为空 | 自增 | 主键 | 唯一键 |
| --- | --- | --- | --- | --- | --- |
| wid | int | 否 | 是 | 是 | 否 |
| worker\_name | varchar(255) | 否 | 否 | 否 | 否 |
| sex | varchar(255) | 是 | 否 | 否 | 否 |
| age | varchar(255) | 是 | 否 | 否 | 否 |
| phone\_number | varchar(255) | 是 | 否 | 否 | 否 |
| e\_mail | varchar(255) | 是 | 否 | 否 | 否 |
| location | varchar(255) | 是 | 否 | 否 | 否 |
| edu\_school | varchar(255) | 是 | 否 | 否 | 否 |
| edu\_level | varchar(255) | 是 | 否 | 否 | 否 |
| work\_year | int | 是 | 否 | 否 | 否 |
| statue | varchar(255) | 是 | 否 | 否 | 否 |
| urls | varchar(255) | 是 | 否 | 否 | 否 |
| url\_format | varchar(255) | 是 | 否 | 否 | 否 |
| hash\_code | varchar(255) | 是 | 否 | 否 | 是 |
| fileid | varchar(255) | 否 | 否 | 否 | 否 |

表2.1.2 worker表

3.job表：存储所有的工作信息，如表2.1.3所示

| 列名 | 数据类型 | 允许为空 | 自增 | 主键 | 唯一键 |
| --- | --- | --- | --- | --- | --- |
| jid | int | 否 | 是 | 是 | 否 |
| jname | varchar(255) | 是 | 否 | 否 | 否 |
| jneed\_age | int | 是 | 否 | 否 | 否 |
| jneed\_edu | varchar(255) | 是 | 否 | 否 | 否 |
| jneed\_other | longtext | 是 | 否 | 否 | 否 |
| jneed\_year | varchar(255) | 是 | 否 | 否 | 否 |
| hash\_code | varchar(255) | 是 | 否 | 否 | 否 |

表2.1.3 job表

4.have表：存储用户拥有的建立关系，如表2.1.4所示

| 列名 | 数据类型 | 允许为空 | 自增 | 主键 | 唯一键 |
| --- | --- | --- | --- | --- | --- |
| hid | int | 否 | 是 | 是 | 否 |
| uid | int | 否 | 否 | 否 | 否 |
| wid | int | 否 | 否 | 否 | 否 |

表2.1.4 have表

## 文件和数据交叉索引

对于表格have：

* PRIMARY KEY：hid列被定义为主键，这意味着它将唯一标识每一行数据，并且自动创建一个聚集索引。主键索引提供了快速的唯一性检查和数据访问。
* KEY：have\_hid\_92aeaf\_idx和have\_uid\_41684d\_idx列都被定义为键索引。键索引可以加快特定列的搜索速度，使查询更高效。

对于表格matchwork：

* PRIMARY KEY：minfo列被定义为主键，类似于上述表格，它将唯一标识每一行数据，并创建一个聚集索引。
* KEY：matchwork\_minfo\_f384d1\_idx和matchwork\_wid\_1120cb\_idx列都被定义为键索引。

建立上述索引用于加快数据库查询速度的数据结构。允许数据库系统更快地定位和访问表格中的数据。

# 体系结构设计

本章主要描述软件的逻辑组成接口，即软件结构图的绘制。

## 数据和控制流复审

对需求规格说明或产品规格说明中要实现的功能进行归纳分析，对涉及的数据和控制流进行汇总和归并，为概要设计做准备。

总的逻辑架构：该架构图分为用户层，表示层，业务层，数据访问层，到下面的数据库，以及数据的存储处理，和所在的操作系统。

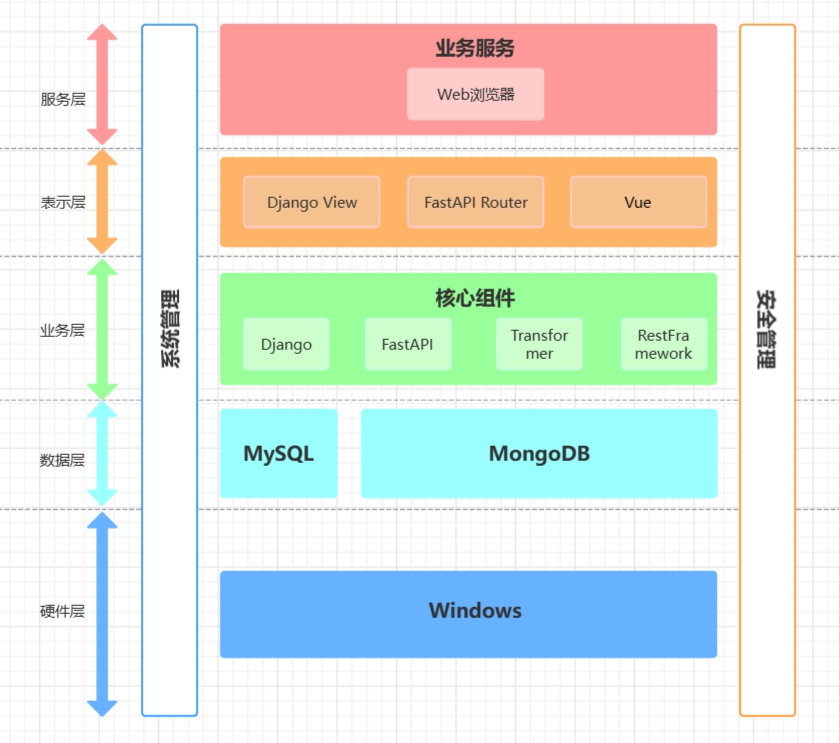


图3.1.1 系统架构图

# 主要模块设计

前端技术采用Vue，后端技术采用Django+FastAPI，数据来源有原始简历数据，后端匹配分析得出想要的结果导入MySQL数据库，后端调用MySQL传递给前端。举两个特征性例子来介绍我们的框架

前后端分离的设计模式使可以使前后端的耦合降低、组内分工明确、项目模块更明显以及业务逻辑更清晰，所以我们采用前后端分离来实现“众里寻他”的系统设计。

## 基于 REST Framework 的业务实现

Django REST Framework (DRF) 是一个用于构建 Web API 的强大框架，它在 Django 的基础上提供了许多用于快速开发 API 的工具和功能。DRF 的主要目标是简化构建高质量 Web API 的过程，提供了一套强大而灵活的工具，使开发人员能够轻松地创建、验证、序列化和渲染 API 数据。

以下是本项目中 Django REST Framework 的一些主要功能和用途：

1. 序列化和反序列化：DRF 提供了强大的序列化器，用于将复杂的数据结构转换为 JSON 或其他格式，以便在 API 的请求和响应中使用。序列化器还支持数据验证、字段级别的控制和嵌套关系的处理。
2. 视图类和混合类：DRF 提供了一组基于类的视图，如 APIView 和 GenericAPIView，这些视图类使得编写 API 视图变得简单和高效。混合类提供了一些常见功能的复用，如身份验证、权限检查和分页。
3. 路由和 URL 配置：DRF 提供了简单而灵活的路由系统，用于将 URL 映射到相应的视图。它支持标准的 URL 配置方式，也可以使用路由器自动生成 URL 配置。
4. 认证和权限：DRF 支持多种身份验证方式，如基于令牌的身份验证、基于会话的身份验证和 OAuth。它还提供了灵活的权限系统，用于限制用户对 API 的访问权限。
5. 请求和响应处理：DRF 提供了灵活的请求和响应处理机制，可以根据客户端的需求进行内容协商、内容压缩和内容协议选择等操作。
6. 分页和过滤：DRF 提供了简单的分页和过滤功能，使得处理大量数据和查询变得更加容易和高效。

以登录功能和查询功能为例，流程图如下图4.2.1、4.2.2所示

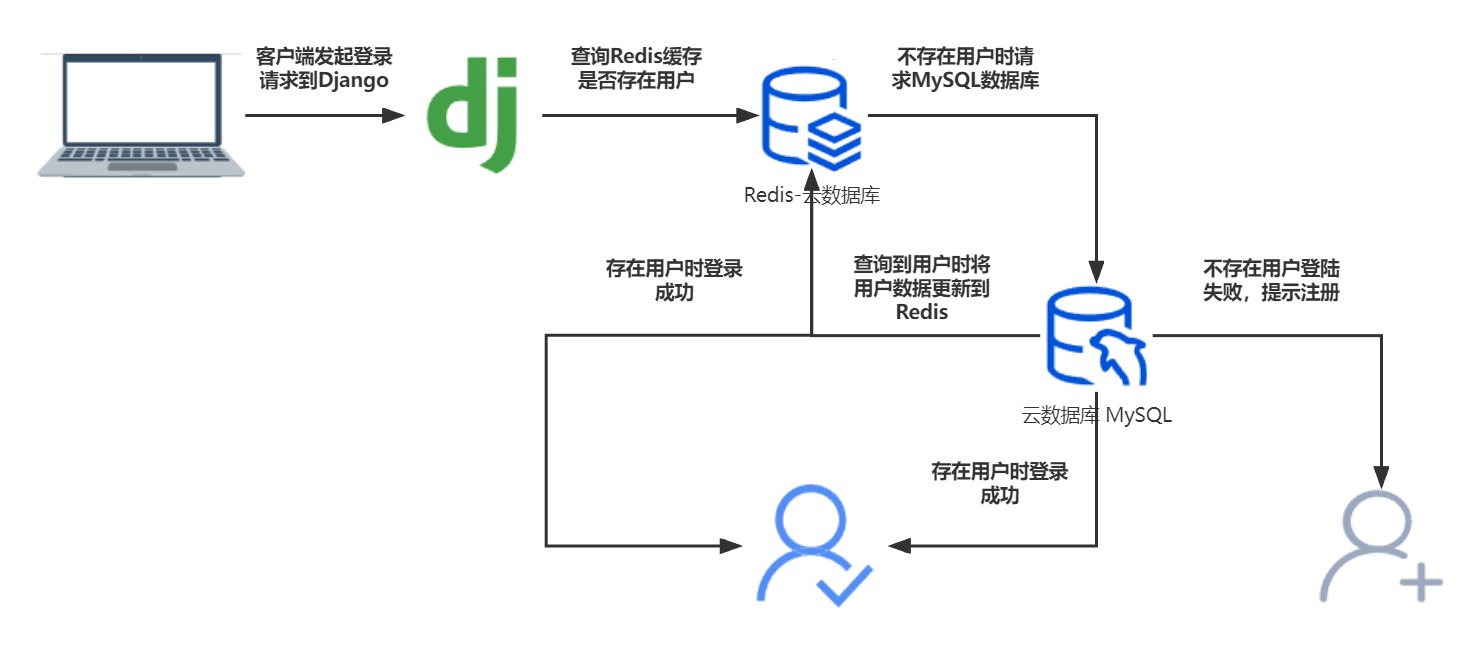


图4.2.1 登录流程图

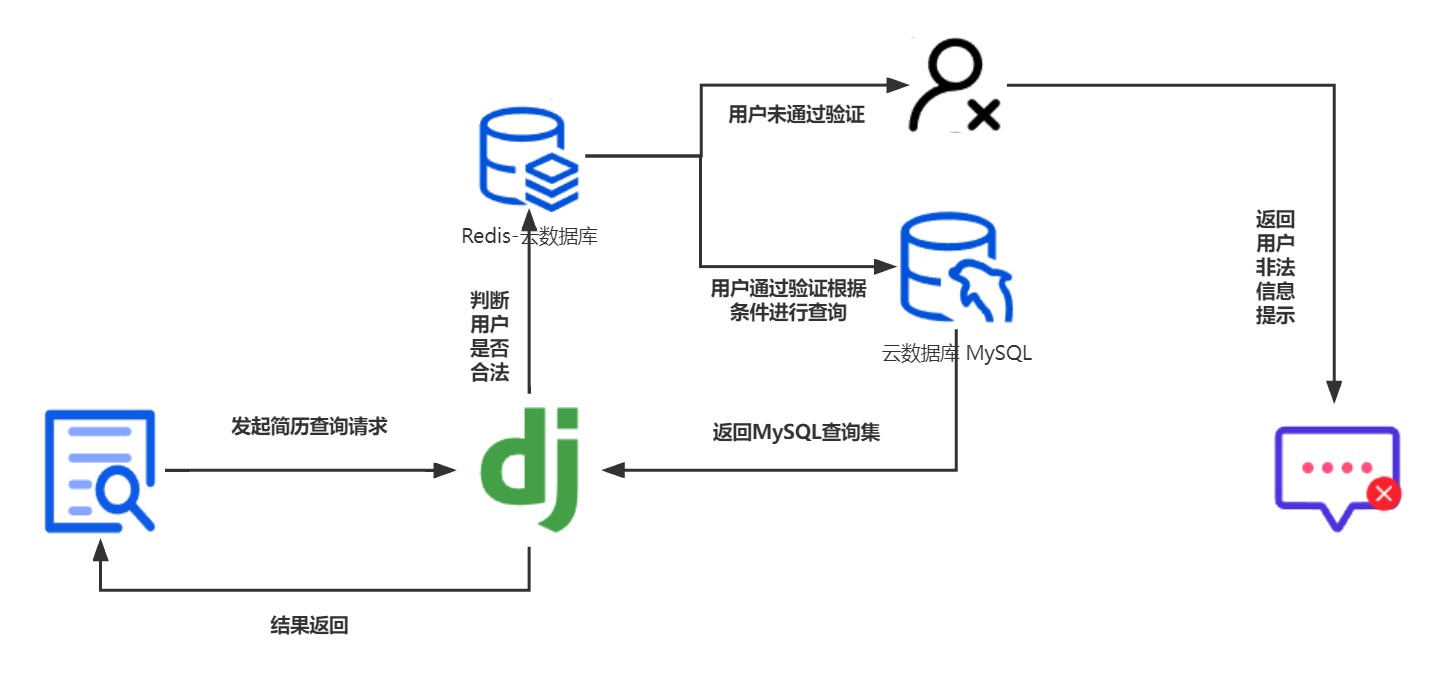


图4.2.2 查询流程图

## 基于Django+Vue的简历上传与信息提取模块

Django将业务逻辑分为三层来返回前端数据。简历上传与信息提取模块的业务流程如下：创建数据库的表，连接数据库，生成对应表操作的Model层文件，在View中编写业务逻辑，对要返回前端的数据进行封装，在urls中给出返回数据的接口地址，前端拿到数据并进行展示。以简历上传界面为例，流程图如下图4.2.2所示：

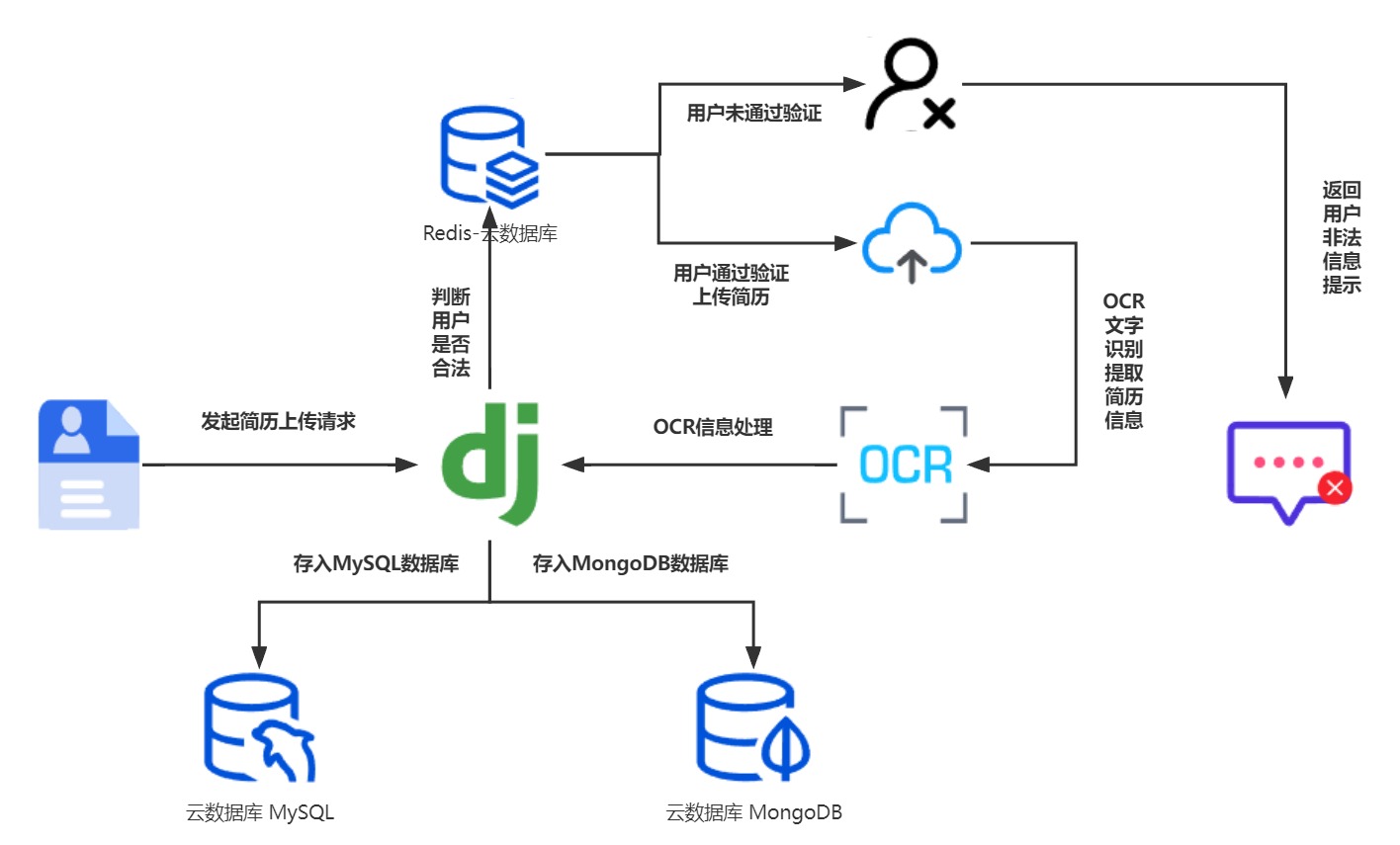


图4.2.2 简历上传与信息提取流程例图

## 基于Django+FastAPI+Vue的简历分析模块

这个模块同简历上传与信息提取模块的业务逻辑及流程相似，但不同的是这一个页面集成了两个后端的功能，当我们拿到参数后在业务逻辑层两个后端从数据库提取自己需要的数据，对简历的信息进行分析后将结果封装好返回给前端，同时会将分析的结果存入MongoDB，从而在下一次调用时明显加快返回数据的速度，前端拿到数据后在页面展示数据，我们以简历分析数据展示界面为例，流程图如下图4.3.1所示：

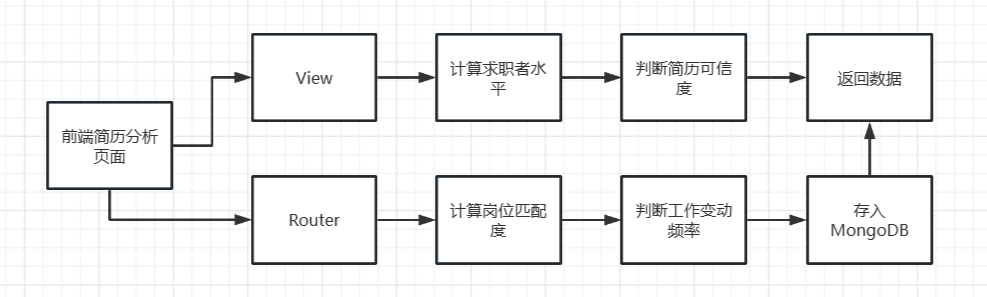


图4.3.1简历分析数据展示流程例图

## 基于Django的简历信息提取

1. Django框架介绍

Django是高水准的Python编程语言驱动的一个开源模型．视图，控制器风格的Web应用程序框架，它起源于开源社区。使用这种架构，程序员可以方便、快捷地创建高品质、易维护、数据库驱动的应用程序。这也正是OpenStack的Horizon组件采用这种架构进行设计的主要原因。另外，在Django框架中，还包含许多功能强大的第三方插件，使得Django具有较强的可扩展性。Django项目源自一个在线新闻Web站点，于2005年以开源的形式被释放出来。Django框架的核心组件有：

①用于创建模型的对象关系映射；

②为最终用户设计较好的管理界面；

③URL设计；

④设计者友好的模板语言；

⑤缓存系统。

1. Django框架的架构设计

Django已经成为web开发者的首选框架，是一个遵循 MVC 设计模式的框架。MVC是Model、View、Controller三个单词的简写，分别代表模型、视图、控制器。Django其实也是一个MTV 的设计模式。MTV是Model、Template、View三个单词的简写，分别代表模型、模版、视图。但是在Django中，控制器接受用户输入的部分由框架自行处理，所以Django里更关注的是模型（Model）、模板(Template)和视图（Views），称为MT模式。它们各自的职责如下表4.4.1所示：

表4.1.1 MTV各部分职责

|  |  |
| --- | --- |
| 层次 | 职责 |
| 模型（Model），即数据存取层 | 处理与数据相关的所有事务：如何存取、如何验证有效性、包含哪些行为以及数据之间的关系等。 |
| 模板(Template)，即表现层 | 处理与表现相关的决定：如何在页面或其他类型文档中进行显示。 |
| 视图（View），即业务逻辑层 | 存取模型及调取恰当模板的相关逻辑。模型与模板的桥梁。 |

从以上表述可以看出Django视图不处理用户输入，而仅仅决定要展现哪些数据给用户，而Django 模板 仅仅决定如何展现Django视图指定的数据。或者说, Django将MVC中的视图进一步分解为 Django视图 和 Django模板两个部分，分别决定“展现哪些数据”和“如何展现”，使得Django的模板可以根据需要随时替换，而不仅仅限制于内置的模板。

至于MVC控制器部分，由Django框架的URLconf来实现。URLconf机制是使用正则表达式匹配URL，然后调用合适的Python函数。URLconf对于URL的规则没有任何限制，你完全可以设计成任意的URL风格，不管是传统的，RESTful的，或者是另类的。框架把控制层给封装了，无非与数据交互这层都是数据库表的读、写、删除、更新的操作。在写程序的时候，只要调用相应的方法就行了，感觉很方便。程序员把控制层东西交给Django自动完成了。 只需要编写非常少的代码完成很多的事情。所以，它比MVC框架考虑的问题要深一步，因为我们程序员大都在写控制层的程序。这个工作交给了框架，仅需写很少的调用代码，大大提高了工作效率。

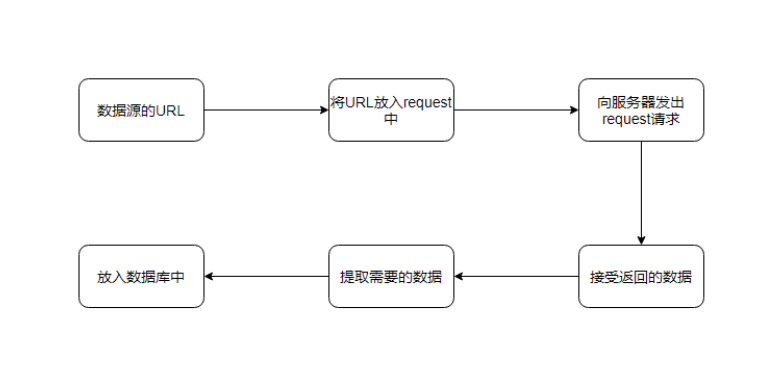


图4.1.1 简历信息提取流程图

经过事无巨细的设计，可以以非常高的精度提取出求职者的信息，从而为后续的分析打下基础。

## 基于RoBERTa模型的岗位匹配

我们通过在多个文本分类相关的模型间进行比较，最终选择了RoBERTa模型。

RoBERTa（Transformer-based Repre- sentations from BERT）是一种基于自注意力机制的预训练语言模型，它在2019年由Facebook AI提出，并被证明在多项自然语言处理（NLP）任务中取得了优秀的性能。

RoBERTa在设计上与BERT（Bidirectional Encoder Representations from Transformers）非常相似，但通过对训练过程进行了改进和扩展，从而提高了模型的性能。RoBERTa主要采用了以下策略：

1. 更长的训练时间：RoBERTa相比BERT使用了更大的训练数据集，并且在训练过程中迭代更多次。这使得模型能够从大规模数据中学习更丰富的语言表示。
2. 动态掩码：在训练阶段，RoBERTa采用了动态掩码的方法，即每次输入样本时都会随机生成遮盖词汇的掩码。这样做可以增加模型对于上下文的理解能力，使其不再依赖于特定位置的信息。
3. 删除NSP（Next Sentence Prediction）任务：与BERT不同，RoBERTa不再使用NSP任务作为辅助目标。这个任务要求模型判断两个句子是否连续，但实验证明，去除这个任务对于模型性能的提升更有益处。

通过这些改进，RoBERTa在多项NLP任务上都取得了令人瞩目的成果。它在GLUE基准（General Language Understanding Evaluation）中实现了当时最好的性能，并且在各种下游任务，如文本分类、命名实体识别和问答等方面都表现出色。

我们的模型通过在100种语言集上进行了预训练。然后在MNLI训练集和XNLI验证和测试集上对NLI的任务进行微调。最后，只对XNLI数据进行了额外的历时训练，在这些数据中，前提和假设的翻译被洗牌，使得每个例子的前提和假设都来自同一个原始英文例子，但前提和假设是不同的语言。这样就拥有了支持多语言的RoBERTa模型，并且能完成各种文本分类任务，从而对求职者适合的岗位精准判断。

同时，我们在1B句子对数据集上进行微调。我们使用了一个对比性学习目标：给定一个句子对，模型应该预测在一组随机抽样的其他句子中，哪一个在我们的数据集中实际上是与它配对的。这样训练出的RoBERTa模型能够进行用户简历信息与岗位要求的相似性计算。

通过以上两种微调方式，将两个不同微调方向的模型进行模型融合后，其结果既包含了企业对求职者的能力要求，也包含了模型对求职者的能力的精准判断，能后基于企业更具参考性的结果。

# 界面设计

智能简历解析系统是一个以网站为展示效果的项目，用户在登陆后可以上传简历和岗位并对之进行分析，界面设计如下：

## 登录及注册界面

登录及注册界面是整个网站的入口，已有账号的用户可以直接登录，还没有账号的用户可以注册后再选择登录，登录后即可上传简历和岗位并对之进行分析。 如图5.1.1所示：

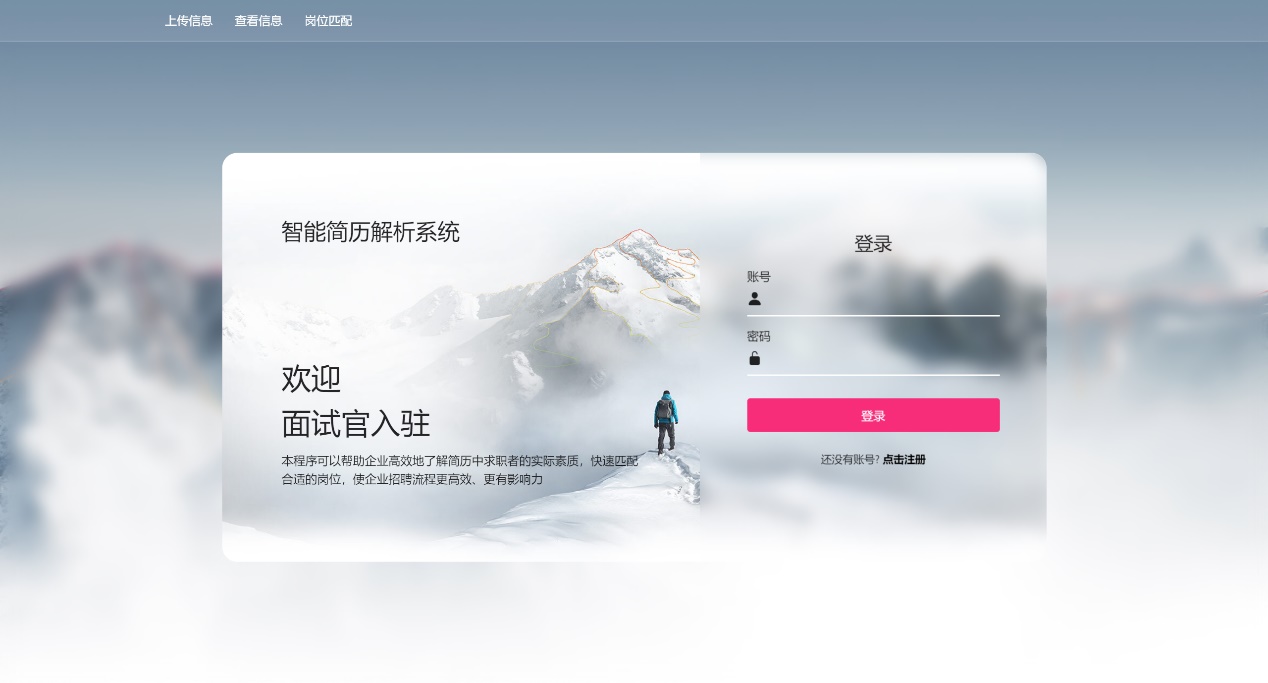


图5.1.1登录及注册页面

## 简历和岗位上传界面

在简历和岗位上传界面用户可以先选择自己要上传数据的类型，系统会根据用户选择上传的类型来展示相关上传界面。如图5.2.1~5.2.5所示：

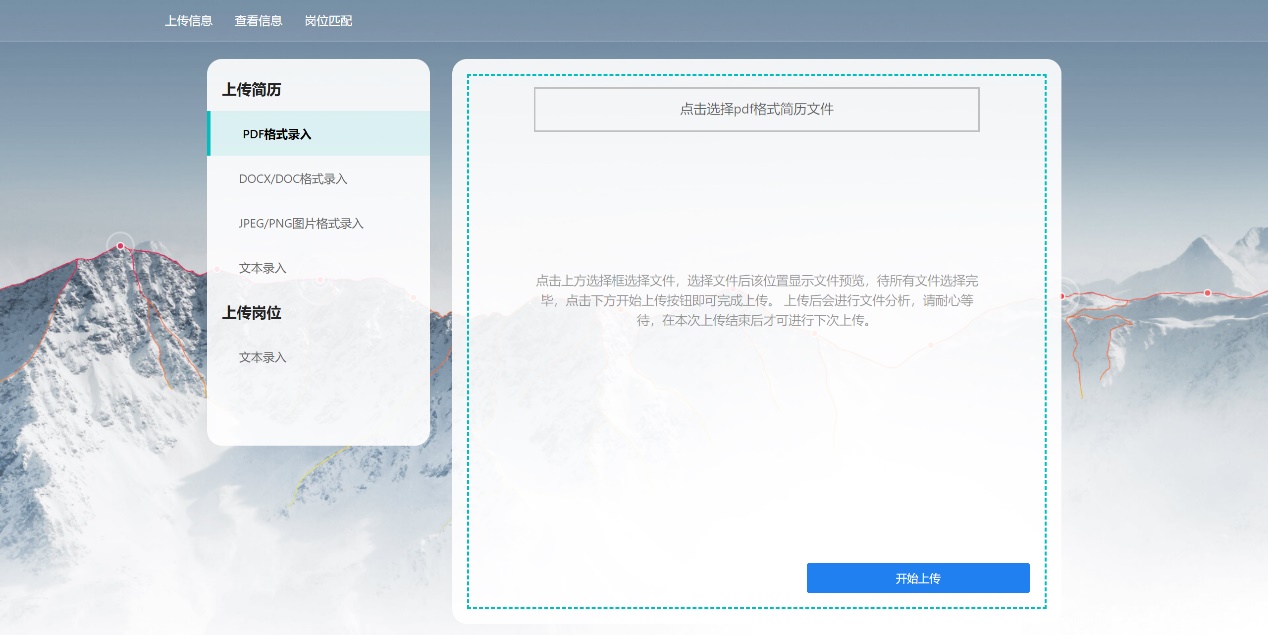


图5.2.1 上传PDF格式简历



图5.2.2 上传DOCX格式简历



图5.2.3 上传JPG/PNG格式简历



图5.2.4 上传文本格式简历



图5.2.5 上传文本格式岗位

## 上传信息查看界面

我们在此页面为用户展示用户上传的简历以及岗位的整体浏览信息，用户可点击进入岗位或简历详情页，具体效果如下图5.3.1，图5.3.2所示。



图5.3.1用户上传简历浏览界面



图5.3.2用户上传岗位浏览界面

## 岗位详情界面

用户可通过该页面查看岗位详情信息，具体效果如下图5.4.1所示。



图5.4.1岗位详情界面

## 简历详情界面

用户可通过该页面查看岗位详情信息，通过上方选择框可选择岗位对简历进行单个分析，具体如下图5.5.1所示：

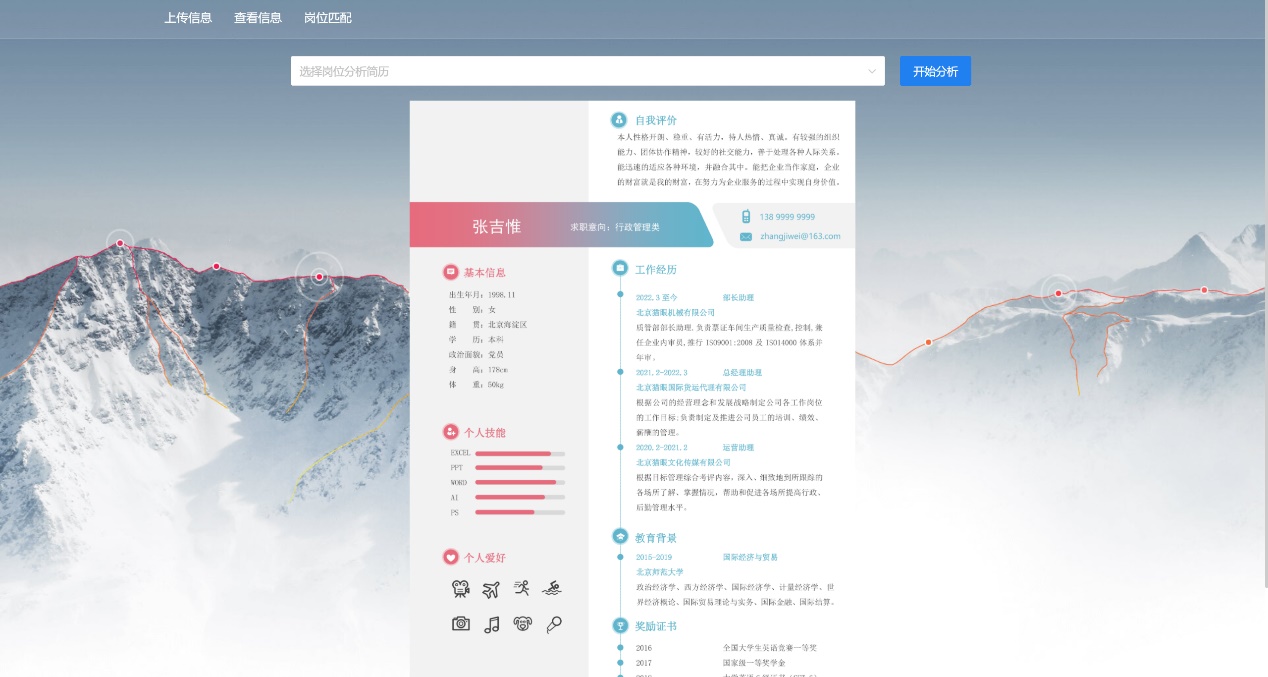


图5.5.1简历详情界面

## 单个简历分析界面

用户可通过该页面查看单个简历对选定岗位的分析结果，具体如下图5.6.1所示：



图5.6.1单个简历分析界面

## 整体岗位匹配界面

用户可通过在该页面选择条件对所有上传简历进行分析比较，系统会匹配出最契合条件的十个简历展示，具体如下图5.7.1所示：



图5.7.1整体简历匹配界面