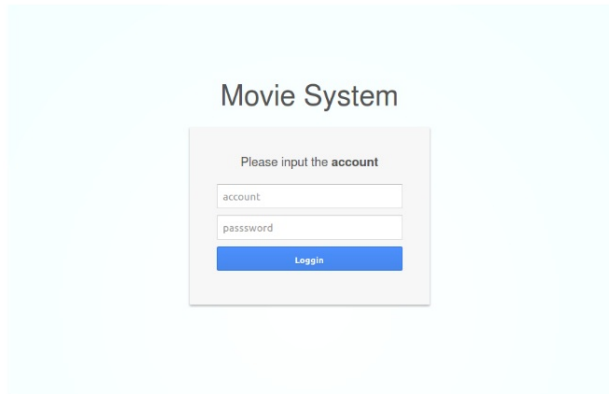


# Movie management system



Movie System

Please input the account

account

password

Login

李止戈 侯嘉伟 吕呈恒

## Introduction

### Purpose of the Project.

- People sometimes don't know which movie to see.
- Good and movies are mixed together in most of the movie platform.
- Most of the movie system lacks a good management.

Thus we need to build a strong and convenient movie management system. And because we are building a platform, so we don't have any references. All the things are done by ourselves.

### Scope of the Project.

1. Web
  - django back-end framework
  - javascript and bootstrap front-end framework
  - sqlite3 as the database
2. crawler
  - scrapy+python
  - mysql+sqlite database

Those files are separated. **It means you don't have to install the crawler if you just want to run the web applications.** The same for the crawler.

### Overview of the Document

We submit 7 files: report, source code, user guidance, final presentation ppt, experimental data, a single page document containing student information and his contribution, bonus document.

1. Report, user guidance, ppt, contribution document, bonus document
  - It's just a pdf contains the necessary information.

2. Source Code

Web document contains the files for running website. Crawler document contains the files for running crawler. (They are separated from each other)

1. Web

- api document is used to connect the back-end and the front-end.
- collected\_static document contains the front-end framework
- database document contains the model of the database.
- nervous document is used to run the django framework.
- wechat document is used to implement some functions of the website.

2. Crawler

- The main crawl code is contained in the doubanmovie/MoiveSpider.py file. Other files are just created for running scrapy framework.

3. Experimental data

- This file contains our **database and data**. (In sqlite3 form)

## Requirements Specification

### Web

## 1.Functional requirements

- Log in with three kind of identity:User, admin, superuser.
- Super User can delete or add the admin.
- Super User can post an announcement where every admin can see.
- Super User can see the whole system information overview.
- Normal user can add an movie application .
- Normal user can withdraw their application and choose to change it or delete it.
- Normal user can see the movie in database, and check the sort the movie by score, year, comment num.
- Admin can approve or reject the application created by user.
- Admin can see the pending, approved, rejected applications.
- Admin can check each movie's link, picture, user who applicated this movie and many movie information.
- Admin can communicate with the user through the mail.
- Admin can sort the movie by score, year, comment num.

## 2.Non-functional requirements

- Install Linux System
- Install neccessary softwares.
- python2.7.11(Linux system already has)
- pip, django web, django-nose, django-coverage,ipython

```
1. sudo apt-get install python-pip
2. sudo pip install django
3. sudo pip install django_nose
4. sudo pip install coverage
5. sudo pip install ipython
```

If there is any error states that lack of other dependency xxx, please just directly “sudo pip install xxx”

## 3.Domain requirements

If you just want to use it in 127.0.0.1, then normal computer can run.

## Crawler

### 1.Functional requirements

- Crawl the information from the douban website.
- Add the information to the database.

### 2.Non-functional requirements

- Install Linux System
- Install neccessary softwares.
- python2.7.11(Linux system already has)
- install scrapy

```
1. sudo apt-get install python-pip
2. sudo pip install lxml
3. sudo pip install OpenSSL
4. sudo apt-get install python-dev
5. sudo apt-get install libevent-dev
6. pip install scrapy
```

If there is any error states that lack of other dependency xxx, please just directly “sudo pip install xxx”

## 3.Domain requirements

The normal computer can run.

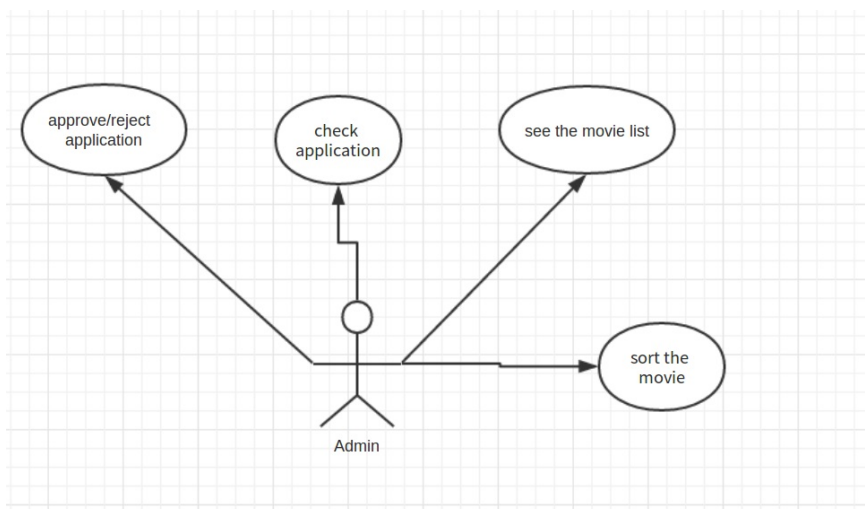
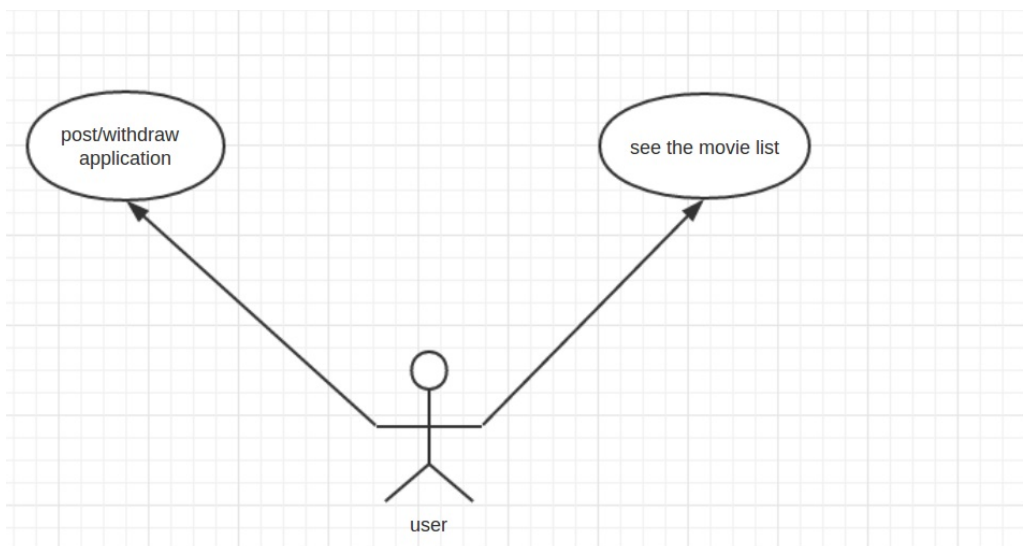
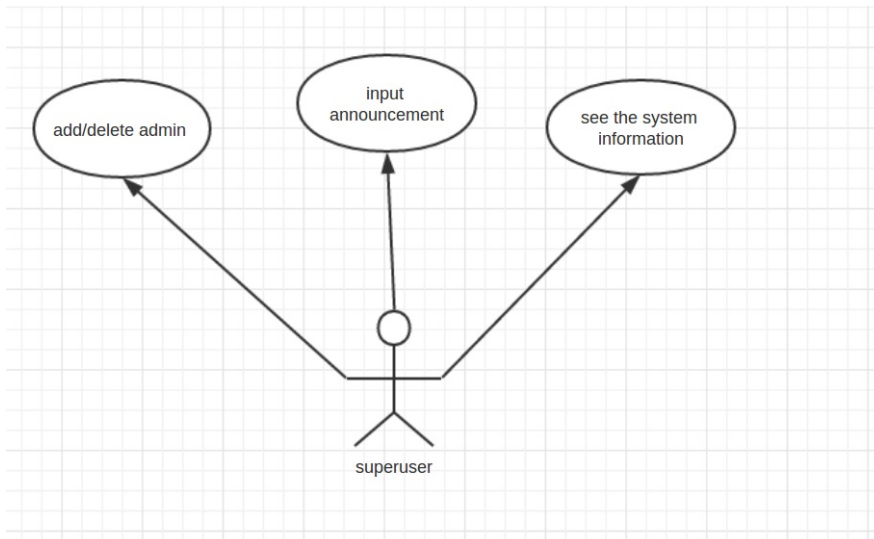
## Software Design

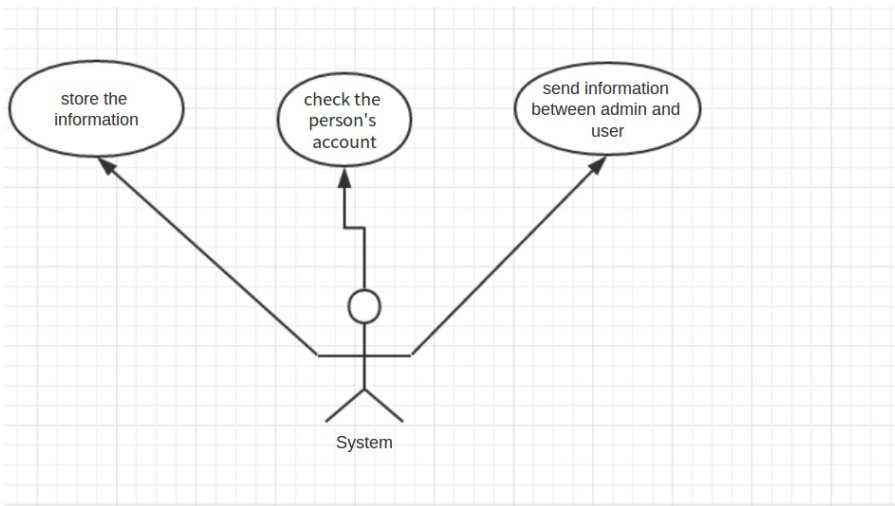
The architecture of the application is represented following the recommendations of the Rational Unified Process and the Rational

Architecture Practice guidelines. And this document presents the architectural as a series of views:

- Use Case View
- Logical View
- Process View
- Implement View
- Deploy View

## Use-Case View





The image of the use case is showed above. These are the functions of all the systems. And it provide a strong movie manage system and carefully arrange the authority according to different identity.

### Architecturally-Significant Use Cases

The architecturally-significant use cases are those, that “exercise” the most critical parts of the system architecture and demonstrate the core system functionality. In this system they are:

**Account Input:** This use case allows user to input their account and password . When a use starts to login the website, this use case starts.

Basic Scenarios:

- User log in.
- User input their account name and password.
- System receives the message and recognize its identity, then shows the corresponding site.

**Application Input:** This use case allows normal user to input the basic information of their favorite movies and wait for the admin to approve.

Basic Scenarios:

- User log in.
- User fill in the information of the movies.
- The system stores the information and shows it on the admin site.

**Movies Output:** This use case allows the system to show the movies in database in detail and can also sort according to different status.

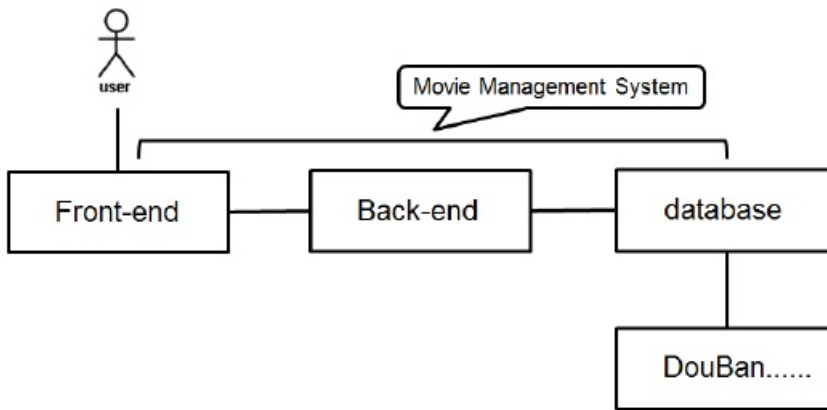
Basic Scenarios:

- User log in.
- User clicks to choose the sort method
- System shows the movies.

### Logical View

This section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages. It describes the logical structure of the system. It starts from the overview of the architecture and then presents its key structure, behavioral elements and mechanisms.

### Overview



There are three dominant structures in the application design model:

1. Front-end of the system that is used to designed of the interactions with the user.
2. Back-end of the system that is used to operate the front-end template and control the database.
3. The database of the system is used to store the infomation of the user, movie.

### Architecturally-Significant Model Elements:

Front-end:

- Bootstrap frmawork
- Javascript
- ajax transmission

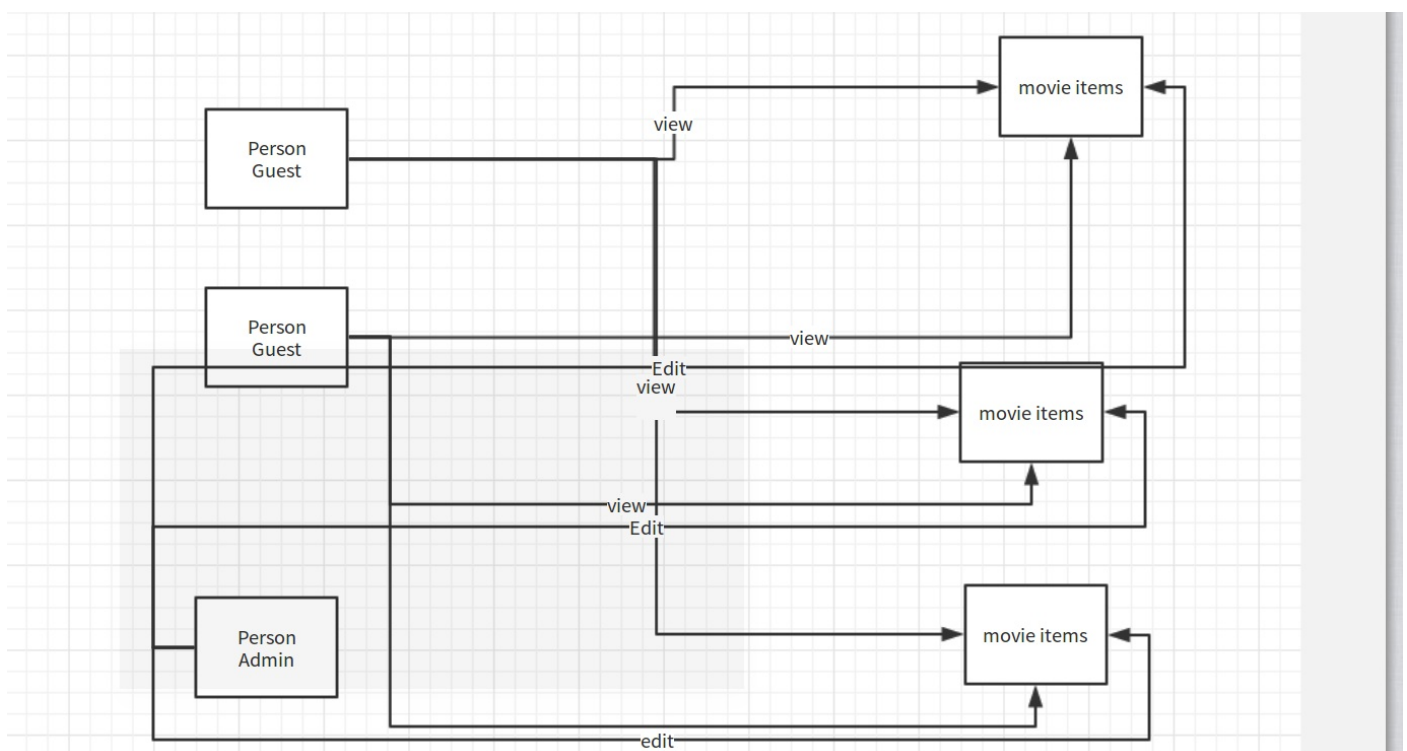
Back-end:

- django framework
- md5 encryption

DataBase:

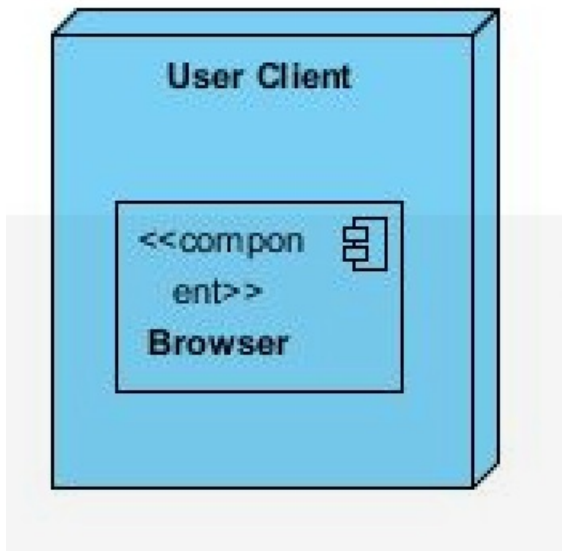
- sqlite3 storing method

### Process View



### Deployment View

The deployment view of a system shows the physical nodes on which the system executes and the assignment of the system processed to the nodes.



## Implementation View

Organized by IDE , in this system, we use python for back-end. We use javascript, html and css for front-end.

## Use Case Sprecification

### Error

#### 1.Definition

This is the requirement description for the Error use case. Error use case is for system to throw an error message.

#### 2.Preconditions

- The input parameters are inlegal.
- The website is not successfully executed.
- Try to modify the data illegally.

#### 3.Post Conditions

- Goes to the 404 page.

#### 4.Scenarios

- The system throus an 404 page.

#### 5.Exceptions or Branches

- The system will go to the debug page and the user should fix the problem.

#### 6.Note

- NULL

## Upload application

#### 1.Definition

This is the requirement description for the user to submit their application for movies.

#### 2.Preconditions

- First the user login with the identity of the normal user.
- They input the parameter of the movies.

#### 3.Post Conditions

- The applicatiосn is upload and wait for the appove.

#### 4.Scenarios

- The parameters are checked whether they have illegal char.
- The uploader's identity is checked to judge whether they are normal user.
- The application is stored in the database.

#### 5.Exceptions or Branches

- NULL.

#### 6.Note

- NULL

## Movie List Output

#### 1.Definition

This is the requirement description for the movie list output case. It is used to show the movies in database.

#### 2.Preconditions

- The identity of the user is required.

#### 3.Post Conditions

- Show the movie list with the information of the year, score, comment num, name.

#### 4.Scenarios

- NULL

#### 5.Exceptions or Branches

- NULL

#### 6.Note

- NULL

## Data Analysis Report

### Introduction

#### 1.Purpose

This document is about our data. I will introduce some details about our database, including the method how we download the data. And then, I will analysis these data and introduce what we can do on this database.

#### 2.Background

Because we will establish a movie system, we should have a database which stores many movies. Only in this way, can we show the movie management intuitively and conveniently. We can get this information from the traditional movie system. In this project, we download the information of movies from DouBan movie system. In the elementary work, we crawl only 245 movies from DouBan web which are enough to show a framework of our system.

### Down The Data

#### 1.introduction

Scrapy is an application framework for crawling web sites and extracting structured data which can be used for a wide range of useful applications, like data mining, information processing or historical archival. Even though Scrapy was originally designed for web scraping, it can also be used to extract data using APIs or as a general purpose web crawler.

#### 2.Details

We explain how to install scrapy in the guide book detailedly.

Before you start scraping, you will have to set up a new Scrapy project. Enter a directory where you'd like to store your code and run: 'scrapy startproject tutorial'.

Then, we should define the data which we need. In Scrapy, it is completed by Scrapy Items. In this project, we use XPath to crawl information. XPath expressions are very powerful, and are the foundation of Scrapy Selectors. In fact, CSS selectors are converted to XPath under-the-hood. You can see that if you read closely the text representation of the selector objects in the shell. We can use chrome to get XPath quickly. Copy XPath is enough. But for some clauses, XPath is not enough to distinguish them or XPath is



complex, so we should use regular expressions. For example, when we download information 'year', we can write '((d+))' which show that we only want integers.

The movies of DouBan web are in different pages. We need to write a spider with interlinkage. The most convenient method is writing a regular expression. We can find regular pattern among the web urls. They are end with integers and same as each other in the first several words. After this, the spider is entirely automatic.

There are some things we should consider when we use scrapy. DouBan web can shield the spiders automatically, so we should modify the parameters in settings.py to change the USER\_AGENT.

## **Store the Data**

### **1.Introduction**

MySQL is an open-source relational database management system. MySQL has so many advantages: high performance, low cost, good reliability. As a result, MySQL becomes the most popular open-source database nowadays. It is very convenient and easy for us to use. So we use MySQL when crawl data and translate it into sqlite3 when we set up the website.

### **2.Details**

We explain how to install MySQL and python for MySQLDB in the guide book detailedly.

If we want to download data into the database, we should modify pipelines.py. It isn't very difficult. There is a point we should be careful that we must first dispose the characters and then store them. When we use this to download the movies, we should set up a database firstly in shell. The information of database here is the same as information in pipelines.py. Different people have different databases in their computers.

o everyone needs to modify the pipelines.py and ensure that the information is the same. Sqlite3 is also a common database. It is much simpler and easier than Mysql. We have so many methods translating Mysql to Sqlite3. I use navicat here.

So far, we get a full database which has many movies. We can import it to the web easily.

## **Analysis and Future Work**

### **1.Analysis**

Because this is a movie system, we must add some functions to make it convenient to use. For example, users can search a movie quickly. Besides, they can sort movies according to scores, years and comment numbers. All of these make this system much friendlier.

### **2.Future Work**

So far, the database of this system is only from DouBan web. And it only has a small number of movies. These are enough to show the system. But they are not enough to set an integrated movie system. In the future, we need to make our database more powerful and bigger. The data can download from websites and can also from all users.

## **Conclusion**

At last, we achieve this system as a website. We have a succinct login interface.

In super-admin interface, you can all the information of the whole system and create(delete) admin. Besides, you can issue announcement conveniently. And all the users can see it quickly.

In admin interface, you can check and verify all the application. You can choose 'Approved' or 'Rejected'. If an admin choose 'Approved', he should complement the information of the movie. If an

admin choose 'Rejected', he should give the reason why rejected. In the users interface, you can add applications and look all the movies information. If a user want to add an application, he

should write some information about the movie he add.

A user also can look all the movies information. Beside this, he can use several functions, such as searching, sorting and so on.

At here, I will conclude movie management system briefly. First, it is a generally-completed system for movie management. The system has different levels of users. Second, as for management,

users can see the recommend list and do searching, selecting and sorting. Last, as for operation, users can submit and verify and modify the movie database.