第一章 系统需求分析

# 1.1 导言

## 1.1.1 目的

本文档描述了短视频共享网站的功能需求和系统结构设计。

## 1.1.2 范围

该短视频共享网站面向互联网用户。在这个网站中，用户可以进行视频的分享，对其他用户的分享视频进行下载、评论，与其他用户进行交流，同时，网站的管理人员也可以对网站内的秩序进行管理。

## 1.1.3 参考文献

数据库系统概念（原书第6版）[美] Abraham Silberschatz，Henry F.Korth，S.Sudarshan 著，杨冬青，李红燕，唐世渭 译, 机械工业出版社，2012

软件工程与计算（卷二）软件开发的技术基础 丁二玉，刘钦 著，机械工业出版社

# 1.2 总体描述

网站满足用户自行上传和下载不超过15秒的短视频的需求，同时满足用户对视频进行评论和对其他用户评论进行评论的需求。

# 1.3 详细需求描述

## 1.3.1对外接口需求

### 1.3.1.1 用户界面

### 1.3.1.2 通信接口

### 1.3.1.3 软件接口

使用MySQL作为底层数据库存储相关数据。

## 1.3.2 功能需求

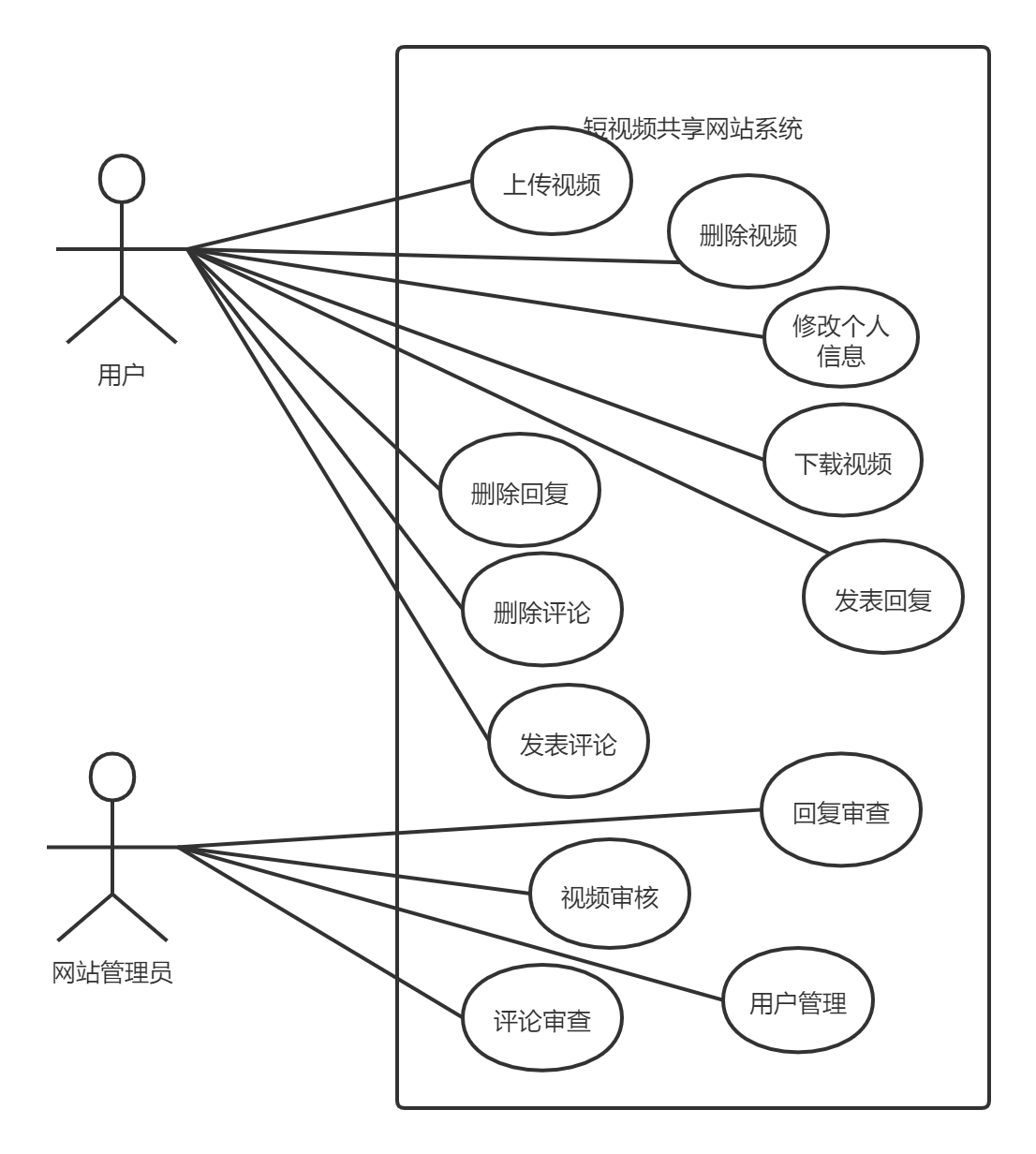
### 1.3.2.1 用户功能需求

1. 用户可以在网站上建立自己的账户，修改自己的信息。
2. 用户可以在网站上下载视频观看。
3. 用户可以在视频下方评论或回复，与其他用户进行交流。
4. 用户可以在网站上分享上传自己的视频。
5. 用户可以删除自己分享发布的视频。
6. 用户可以删除自己发表的评论和回复。

### 1.3.2.2 网站管理员功能需求

1. 管理员可以删除用户上传的违规的视频。
2. 管理员可以在任何视频下删除违规的评论或回复。
3. 管理员可以对多次违规的用户进行封禁。

## 1.3.3 用例



### 1.3.3.1 上传视频

* 用例编号：1
* 用例名称：上传视频
* 参与者：用户
* 触发条件：用户想要上传视频
* 前置条件：用户已登录；用户状态正常
* 主要事件流
  + 用户点击上传视频选项
  + 用户选择要上传的视频
  + 检测视频长度是否不超过15s
  + 系统提示用户成功信息
* 异常
  + 用户被封禁：系统提示用户封禁状态
  + 视频长度超出限制：提示用户视频长度过长

### 1.3.3.2 修改个人信息

* 用例编号：2
* 用例名称：修改个人信息
* 参与者：用户
* 触发条件：用户想要修改昵称
* 前置条件：用户已登录；用户状态正常
* 主要事件流
  + 用户点击自己的名称
  + 用户输入新的名称
  + 系统提示用户成功信息
* 异常
  + 用户被封禁：系统提示用户封禁状态

### 1.3.3.3 下载视频

* 用例编号：3
* 用例名称：下载视频
* 参与者：用户
* 触发条件：用户想要下载网站上的视频
* 前置条件：用户已登录；视频存在在网页上；用户状态正常
* 主要事件流
  + 用户点击想要下载的视频
  + 用户选择下载视频
  + 系统向用户传输视频
* 异常
  + 视频不存在网站上：系统提示视频不存在
  + 用户被封禁：系统提示用户封禁状态

### 1.3.3.4 发表评论

* 用例编号：4
* 用例名称：发表评论
* 参与者：用户
* 触发条件：用户想要对视频发表评论
* 前置条件：用户已登录；视频存在在网页上；用户状态正常
* 主要事件流
  + 用户点击想要评论的视频
  + 用户输入想发表的评论内容
  + 系统提示用户成功信息
* 异常
  + 视频不存在网站上：系统提示视频不存在
  + 用户被封禁：系统提示用户封禁状态

### 1.3.3.5 视频审核

* 用例编号：5
* 用例名称：视频审核
* 参与者：网站管理员
* 触发条件：管理员想要删除视频
* 前置条件：视频存在网页上
* 主要事件流
  + 网站管理员选择要删除的视频
  + 网站管理员选择删除
* 异常

### 1.3.3.6 用户管理

* 用例编号：6
* 用例名称：用户管理
* 参与者：网站管理员
* 触发条件：管理员想要封禁用户
* 前置条件：无
* 主要事件流
  + 网站管理员选择要封禁的用户
  + 管理员选择封禁
* 异常

### 1.3.3.7 删除视频

* 用例编号：7
* 用例名称：删除视频
* 参与者：用户
* 触发条件：用户想要删除自己发布的视频
* 前置条件：视频存在网站上；视频是用户发布的
* 主要事件流
  + 用户选择要删除的视频
  + 用户选择删除
* 异常
  + 视频不存在网站上：系统提示视频不存在

### 1.3.3.8 评论审查

* 用例编号：8
* 用例名称：评论审查
* 参与者：网站管理员
* 触发条件：管理员想要删除评论内容
* 前置条件：视频及评论存在网站上
* 主要事件流
  + 网站管理员选择要审查评论的视频
  + 网站管理员选择要删除的评论
  + 网站管理员选择删除
* 异常

### 1.3.3.9 发表回复

* 用例编号：9
* 用例名称：发表回复
* 参与者：用户
* 触发条件：用户想对评论进行评论
* 前置条件：评论存在网站上；用户状态正常
* 主要事件流
  + 用户选择要回复的评论
  + 用户输入要回复的内容
  + 网页提示用户成功信息
* 异常
  + 视频不存在网站上：系统提示视频不存在
  + 用户被封禁：系统提示用户封禁状态

### 1.3.3.10 删除评论

* 用例编号：10
* 用例名称：删除评论
* 参与者：用户
* 触发条件：用户想删除自己发表的评论
* 前置条件：评论存在网站上；评论是用户发布的
* 主要事件流
  + 用户选择要删除的评论
  + 选择删除
* 异常
  + 评论不存在网站上：系统提示视频不存在

### 1.3.3.11 删除回复

* 用例编号：11
* 用例名称：发表回复
* 参与者：用户
* 触发条件：用户想删除自己发表的回复
* 前置条件：回复的评论存在网站上；回复存在网站上；回复是用户发布的
* 主要事件流
  + 用户选择要删除的回复
* 选择删除异常
  + 评论不存在网站上：系统提示评论不存在
  + 回复不存在网站上：系统提示回复不存在
  + 用户被封禁：系统提示用户封禁状态

### 1.3.3.12 回复审查

* 用例编号：12
* 用例名称：回复审查
* 参与者：网站管理员
* 触发条件：管理员想删除回复
* 前置条件：回复的评论存在网站上；回复存在网站上
* 主要事件流
  + 管理员选择要删除的回复
  + 管理员选择删除
* 异常
  + 评论不存在网站上：系统提示评论不存在
  + 用户被封禁：系统提示用户封禁状态

## 1.3.4 数据词典

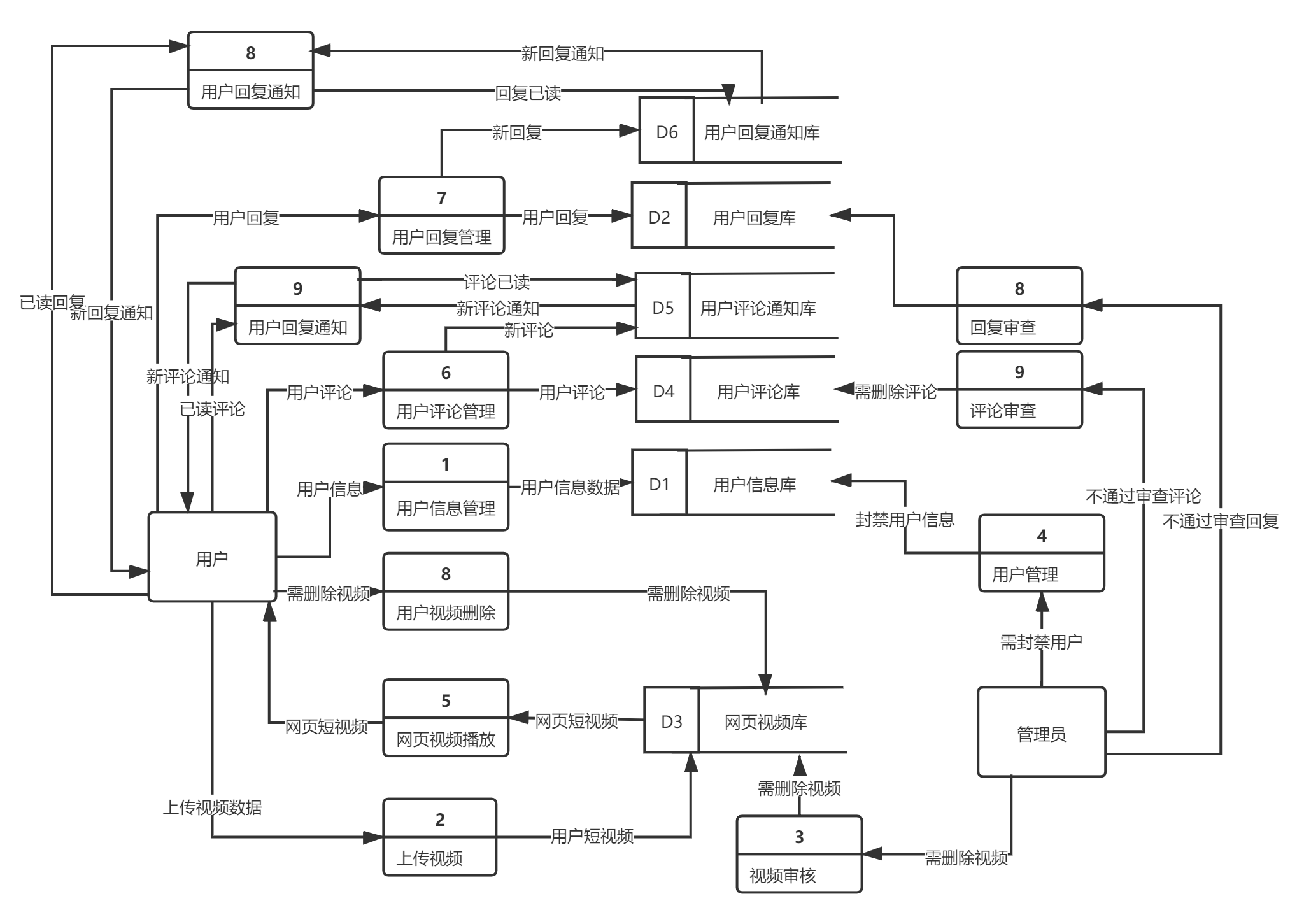
### 1.3.4.1 数据项

* 视频名称：
  + 数据项名：V\_name
  + 数据项含义说明：用户上传的短视频标题
  + 数据类型：varchar
  + 别名：视频名
  + 长度：1~50
  + 取值含义：视频的标题
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 视频路径：
  + 数据项名：V\_addr
  + 数据项含义说明：用户上传的短视频存储路径
  + 数据类型：varchar
  + 别名：视频地址
  + 长度：1~30
  + 取值含义：视频数据的存储路径
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 视频编号：
  + 数据项名：V\_id
  + 数据项含义说明：用户视频的身份id
  + 数据类型：int
  + 别名：视频id
  + 长度：>1
  + 取值含义：视频数据的身份编号
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 视频介绍：
  + 数据项名：V\_intro
  + 数据项含义说明：用户对其上传视频的介绍
  + 数据类型：varchar
  + 别名：视频简介
  + 长度：0~255
  + 取值含义：用户输入的介绍内容
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 视频日期：
  + 数据项名：V\_time
  + 数据项含义说明：用户上传视频时间
  + 数据类型：datetime
  + 别名：上传时间
  + 长度：>0
  + 取值含义：上传视频的时间
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 评论编号：
  + 数据项名：C\_id
  + 数据项含义说明：用户发表的评论的标识id
  + 数据类型：int
  + 别名：评论id
  + 长度：>1
  + 取值含义：用户评论的独特标识id
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 评论内容：
  + 数据项名：C\_content
  + 数据项含义说明：用户发表的评论内容
  + 数据类型：varchar
  + 别名：用户评论
  + 长度：1~255
  + 取值含义：用户的评论内容
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 评论时间：
  + 数据项名：C\_time
  + 数据项含义说明：用户发表评论的时间
  + 数据类型：datetime
  + 别名：评论时间
  + 长度：>0
  + 取值含义：用户发表评论的时间
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 回复编号：
  + 数据项名：R\_id
  + 数据项含义说明：用户发表的回复的标识id
  + 数据类型：int
  + 别名：回复id
  + 长度：>1
  + 取值含义：用户回复的独特标识id
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 回复内容：
  + 数据项名：R\_content
  + 数据项含义说明：用户发表的回复内容
  + 数据类型：varchar
  + 别名：用户评论
  + 长度：1~255
  + 取值含义：用户的回复内容
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 回复时间：
  + 数据项名：R\_time
  + 数据项含义说明：用户发表回复的时间
  + 数据类型：datetime
  + 别名：回复时间
  + 长度：>0
  + 取值含义：用户发表回复的时间
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 回复通知编号：
  + 数据项名：NR\_id
  + 数据项含义说明：新回复通知的id
  + 数据类型：int
  + 别名：回复通知id
  + 长度：>0
  + 取值含义：新回复通知的独特标识id
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 回复通知时间：
  + 数据项名：NR\_time
  + 数据项含义说明：新回复通知的时间
  + 数据类型：datetime
  + 别名：回复通知时间
  + 长度：>0
  + 取值含义：新回复通知的时间
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 回复通知状态：
  + 数据项名：NR\_read
  + 数据项含义说明：回复通知是否已读
  + 数据类型：bool
  + 别名：新回复
  + 长度：1
  + 取值含义：1表示通知已读；0表示通知未读
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 评论通知编号：
  + 数据项名：NC\_id
  + 数据项含义说明：新评论通知的id
  + 数据类型：int
  + 别名：评论通知id
  + 长度：>0
  + 取值含义：新评论通知的独特标识id
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 评论通知时间：
  + 数据项名：NC\_time
  + 数据项含义说明：新评论通知的时间
  + 数据类型：datetime
  + 别名：评论通知时间
  + 长度：>0
  + 取值含义：新评论通知的时间
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 评论通知状态：
  + 数据项名：NC\_read
  + 数据项含义说明：评论通知是否已读
  + 数据类型：bool
  + 别名：新评论
  + 长度：1
  + 取值含义：1表示通知已读；0表示通知未读
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 用户编号：
  + 数据项名：U\_id
  + 数据项含义说明：用户的身份id
  + 数据类型：int
  + 别名：用户id
  + 长度：>0
  + 取值含义：用户的独特身份id
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 用户名称：
  + 数据项名：U\_name
  + 数据项含义说明：用户自定义的名称
  + 数据类型：varchar
  + 别名：昵称
  + 长度：1~20
  + 取值含义：用户的名称
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 用户密码：
  + 数据项名：U\_passwd
  + 数据项含义说明：用户的登录密码
  + 数据类型：varchar
  + 别名：密码
  + 长度：0~20
  + 取值含义：用户设定的密码
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 用户注册时间：
  + 数据项名：U\_time
  + 数据项含义说明：用户注册的时间
  + 数据类型：datetime
  + 别名：注册时间
  + 长度：>0
  + 取值含义：用户注册的时间
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 用户状态：
  + 数据项名：U\_banned
  + 数据项含义说明：用户是否被封禁
  + 数据类型：bool
  + 别名：封禁状态
  + 长度：1
  + 取值含义：1表示用户被封禁；0表示用户状态正常
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：
* 用户身份：
  + 数据项名：U\_is\_staff
  + 数据项含义说明：用户的身份
  + 数据类型：varchar
  + 别名：用户身份
  + 长度：1
  + 取值含义：1表示用户是管理员；0表示是普通用户
  + 与其他数据项的逻辑关系：
  + 与其他数据项的联系：

### 1.3.4.2 数据结构

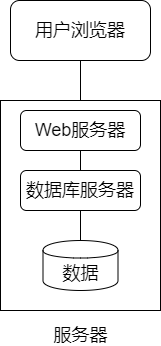
* 用户：
  + 数据结构名：user
  + 含义说明：存放了用户的相关信息
  + 组成：
    - U\_id
    - U\_name
    - U\_passwd
    - U\_time
    - U\_banned
    - U\_is\_staff
* 视频：
  + 数据结构名：video
  + 含义说明：存放了视频的相关信息
  + 组成：
    - V\_id
    - V\_addr
    - V\_intro
    - V\_time
    - U\_id
* 评论：
  + 数据结构名：comment
  + 含义说明：存放了评论的相关信息
  + 组成：
    - C\_id
    - C\_content
    - C\_time
    - V\_id
    - U\_id
* 回复：
  + 数据结构名：reply
  + 含义说明：存放了回复的相关信息
  + 组成：
    - R\_id
    - R\_content
    - R\_time
    - U\_subject\_id
    - C\_id
    - U\_object\_id
* 评论通知：
  + 数据结构名：noticeC
  + 含义说明：存放了评论通知的相关信息
  + 组成：
    - NC\_id
    - NC\_read
    - NC\_time
    - U\_id
    - C\_id
* 回复通知：
  + 数据结构名：noticeR
  + 含义说明：存放了回复通知的相关信息
  + 组成：
    - NR\_id
    - NR\_read
    - NR\_time
    - U\_id
    - R\_id

## 1.3.4 数据流图



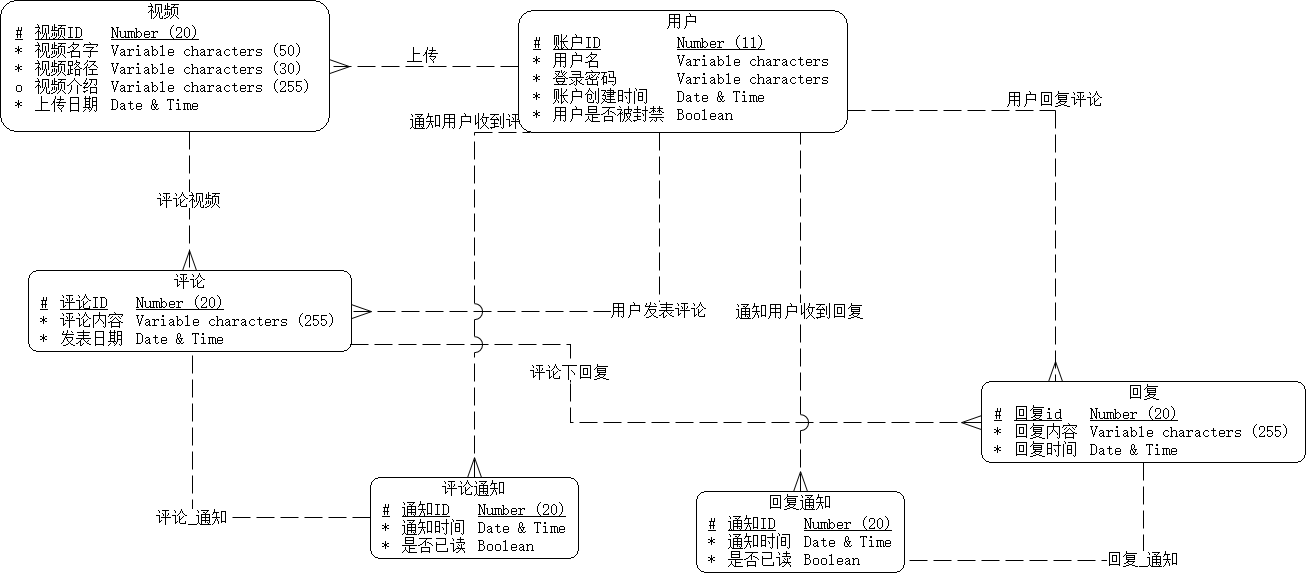
第二章 系统设计

# 2.1 系统体系结构设计



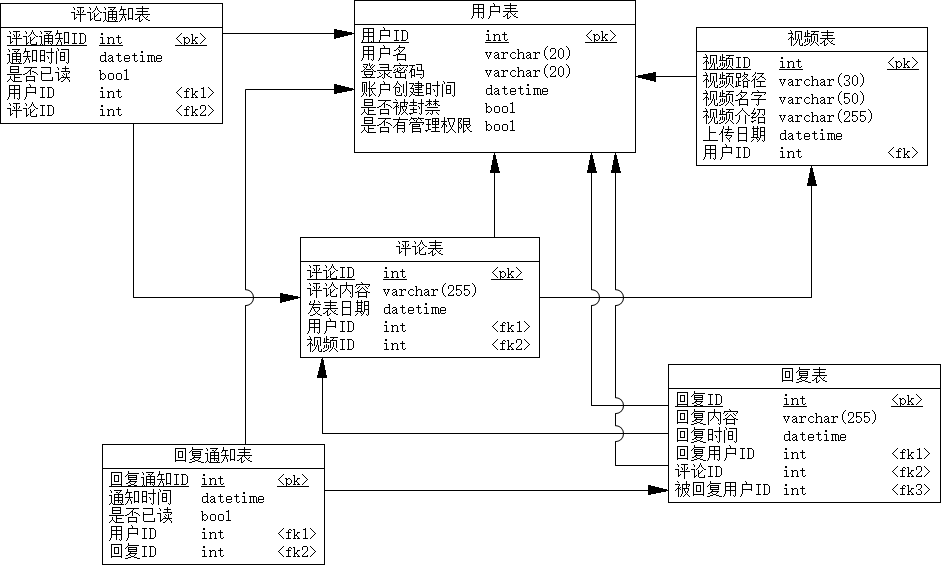
短视频分享网站的结构采用两层Web应用体系结构，应用程序运行在Web服务器中。用户浏览器（以下简称用户）通过HTTP协议向服务器发送请求，服务器从数据库中获取必要信息，并动态生成HTML页面返回给用户的请求。

第三章 概念结构设计



第四章 逻辑结构设计

# 4.1 基表设计



# 4.2 每个表的结构(用mysql建表代码来说明)

/\*==============================================================\*/

/\* Table: user 用户表 \*/

/\*==============================================================\*/

create table user

(

U\_id int not null auto\_increment,

U\_name varchar(20) not null,

U\_passwd varchar(20) not null,

U\_time datetime not null,

U\_banned bool not null,

U\_is\_staff bool not null,

primary key (U\_id)

);

/\*==============================================================\*/

/\* Table: video 视频表 \*/

/\*==============================================================\*/

create table video

(

V\_id int not null auto\_increment,

V\_addr varchar(30) not null,

V\_name varchar(50) not null,

V\_intro varchar(255),

V\_time datetime not null,

U\_id int not null,

primary key (V\_id)

);

alter table video add constraint FK\_user\_video foreign key (U\_id)

references user (U\_id) on delete cascade on update cascade;

/\*==============================================================\*/

/\* Table: comment 评论表 \*/

/\*==============================================================\*/

create table comment

(

C\_id int not null auto\_increment,

C\_content varchar(255) not null,

C\_time datetime not null,

U\_id int not null,

V\_id int not null,

primary key (C\_id)

);

alter table comment add constraint FK\_user\_comment foreign key (U\_id)

references user (U\_id) on delete cascade on update cascade;

alter table comment add constraint FK\_video\_comment foreign key (V\_id)

references video (V\_id) on delete cascade on update cascade;

/\*==============================================================\*/

/\* Table: reply 回复表 \*/

/\*==============================================================\*/

create table reply

(

R\_id int not null auto\_increment,

R\_content varchar(255) not null,

R\_time datetime not null,

U\_subject\_id int not null,

C\_id int not null,

U\_object\_id int,

primary key (R\_id)

);

alter table reply add constraint FK\_comment\_reply foreign key (C\_id)

references comment (C\_id) on delete cascade on update cascade;

alter table reply add constraint FK\_userObject\_reply foreign key (U\_object\_id)

references user (U\_id) on delete cascade on update cascade;

alter table reply add constraint FK\_userSubject\_reply foreign key (U\_subject\_id)

references user (U\_id) on delete cascade on update cascade;

/\*==============================================================\*/

/\* Table: noticeC 评论通知表 \*/

/\*==============================================================\*/

create table noticeC

(

NC\_id int not null auto\_increment,

NC\_time datetime not null,

NC\_read bool not null,

U\_id int not null,

C\_id int not null,

primary key (NC\_id)

);

alter table noticeC add constraint FK\_comment\_noticeC foreign key (C\_id)

references comment (C\_id) on delete cascade on update cascade;

alter table noticeC add constraint FK\_user\_noticeC foreign key (U\_id)

references user (U\_id) on delete cascade on update cascade;

/\*==============================================================\*/

/\* Table: noticeR 回复通知 \*/

/\*==============================================================\*/

create table noticeR

(

NR\_id int not null auto\_increment,

NR\_time datetime not null,

NR\_read bool not null,

U\_id int not null,

R\_id int not null,

primary key (NR\_id)

);

alter table noticeR add constraint FK\_reply\_noticeR foreign key (R\_id)

references reply (R\_id) on delete cascade on update cascade;

alter table noticeR add constraint FK\_user\_noticeR foreign key (U\_id)

references user (U\_id) on delete cascade on update cascade;

第五章 短视频网页设计

后端选用python的django框架编写，连接python自带的sqlite3数据库，前端网页采用bootstrap4来设计。

另外django的设计模式需要用到数据库建表，查询，更新等的思想，例如在项目的models.py中就需要设计数据库中各表的内容，外键，完整性定义等等，就可以直接用实验设计要求的第3步中所建的表的形式，因此非常适合作为本课程实验设计网页的web框架。bootstrap4也是一个比较成熟的前端开源工具库，与django有很好的适配性，python包中就有django-bootstrap4的包，下载该库后可以直接在django HTML中使用bootstrap4。因此 经过组内讨论，决定使用django+ bootstrap4来实现短视频网页。

# 5.1 django简介

Django是一个开放源代码的Web应用框架，由Python写成。采用了MTV的框架模式，即模型M，视图V和模版T。模型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。视图存取模型及调取恰当模板的相关逻辑。模型与模板的桥梁。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。

# 5.2 bootstrap4简介

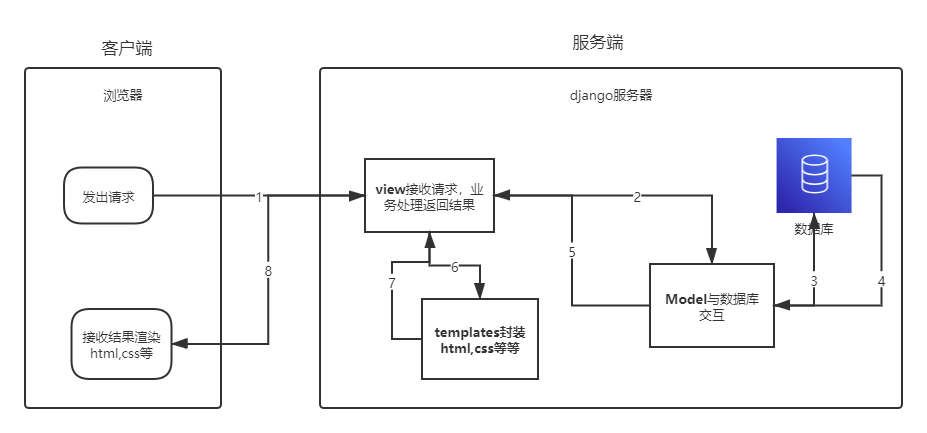
Bootstrap 是全球最受欢迎的前端开源工具库，它支持 Sass 变量和 mixin、响应式栅格系统、自带大量组件和众多强大的 JavaScript 插件。并且能很好的与django框架结合。

# 5.3 实现的主要功能

* 用户系统：包括用户的注册，登入和登出。且用户必须登录后才能使用网站的相应功能
* 管理员系统：网站管理员可以登录管理员站点对视频，评论，回复等进行审核，可以对一些不当的视频或者言论进行删除
* 用户可以浏览文件管理器上传本地视频，或者下载视频，或者删除自己上传的视频，或者编辑自己上传的视频的信息
* 用户搜索系统：可以根据短视频的名字或者用户的名字来检索视频
* 用户评论系统：参考了微博、b站平台的评论形式，采用二级评论模式，用户可以对上传的视频进行评论，对评论进行回复，对回复进行回复，也可以删除自己的评论或回复
* 用户通知系统：当有用户评论了你的视频或回复了你的视频，可以在通知界面看到评论通知和回复通知，并选择其中的一些进行回复

# 5.4 设计方法

软件架构图为



大致设计方法是在models.py中设计数据库的表，在urls.py中添加url模式，并在views.py中添加视图函数，在urls.py中将url与相应的处理的视图函数绑定，在templates/[app name]/下编写视图函数返回的html文档并在其中使用views.py内的视图函数传入的模板变量。具体的实现代码可以看项目打包的代码，使用方法可以看打包的根目录下的README.md文档，且项目已经上传到<https://github.com/weizhehuang0827/ShortVideo>