CSC261 Project Milestone1

Project Details

Name of the project: JsTube
Team number: 17
Team name: Jays Club
Group members: Junfei Liu(jliu137), Jinghan Lu (jlu54), Jinghao Jiang (jjiang27)

Problem Statement

As compared to excel, we preferred to implement a database since it provides higher efficiency when handling a large scale of data and presumably provides more security benefits. Both the number of users and videos can grow incredibly fast. Too much data makes it hard to import and retrieve using excel, but a database can easily store a huge amount of data and enables efficient querying. Taking advantage of the database, users will not need to wait for a few seconds or even minutes for, for example, a simple query to retrieve the last video they watched.

Also, databases provided presumably better security benefits. Our database contains confidential information that should require certain permission to be accessed like administrator passwords.

Target User

Anyone who would like to use our platform for studying and uploading videos related to studying would be our target users. Administrators would be us or people who own the administrator accounts. Administrators will work with video uploads, case solving, and user issues.

List of Relations

videos(video_name:varchar(50), id:int, user_id:int, upload_time:datetime, membership_requirment: boolean, link:varchar(2083))

- **id** should be unique for each video and will serve as the primary key
- **user_id** should be unique and serve as a foreign key for **id** in **user**
- **video_name, upload_time, membership_requirment** are the basic information of the video
- **link** refers to the link of the video

user(id:int, user_name:varchar(20), email:varchar(20), password:varchar(20), membership: boolean)

- **id** should be unique for each user and will serve as the primary key
- **user_name, email, password** are the basic information for the user
- **membership** refers to whether the user has valid a membership (true for joined and false for not)

membership(user_id: int, level: int, valid_through: date)

- **user_id** should be unique for each user and serves as a foreign key for **id** in **videos**
- **level, valid_through** marks the user's membership status

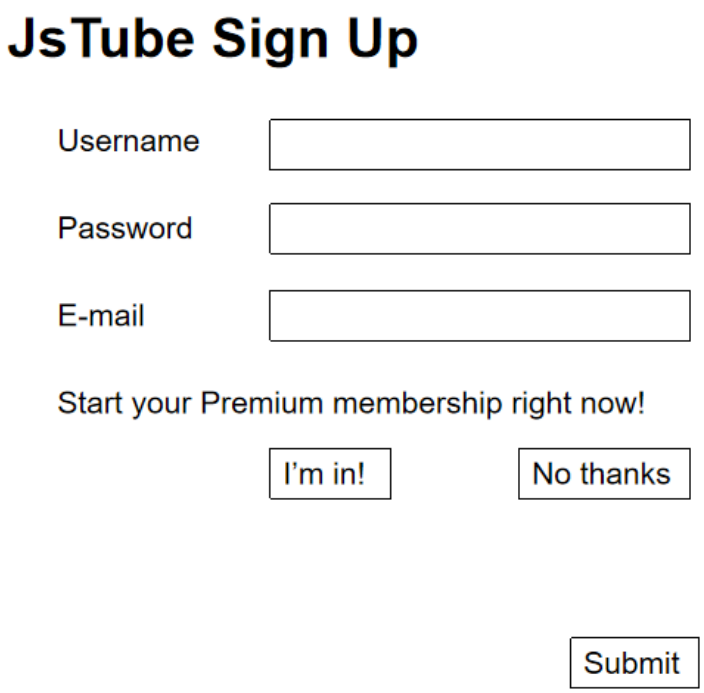
administrator(id:int, password: varchar (20), email:varchar(20))

- **id** should be unique for each administrator and will serve as the primary key
- **password, email** are the basic administrator account information

case(case_id: int, administrator_id: int, type:int,video_id:int)

- **case_id** should be unique for each case and will serve as the primary key
- **administrator_id** should be unique and serve as a foreign key for **id** in **administrator**
- **video_id** should be unique and serve as a foreign key for **id** in **videos**
- **type** marks the category to which the case belongs (a mapping will be created from the int stored in this table to the actual category name)

Web-Interface



JsTube Administrator Login

ID

Password

Submit

JsTube Videos

video_name1

Premium Only ✔

JsTuber name1

Upload time1

video_name2

Premium Only ✘

JsTuber name2

Upload time2

Data

Our data will be created both by referencing the real-world data and created by our own. We will reference our own video history on video sharing platforms like youtube or bilibili to create the video data. And we will come up with user and administrator information on our own.

The database will be populated in two ways: one from loading the given data we created as described above and the other is from front-end user interactions. Our database will also add information about newly uploaded videos or user sign up. We will come up with administrators' accounts and related information on our own.