Problem set 4: Odds, Ends, and User Management
Members: Mahdi, Ice, Violet
Group 7

Question 1.
Potential content for this site:
1. Movie Diary
Reviews, Critical articles about movies and User's experience of watching films and television Programs. This includes actors, production crew personnel, and fictional characters featured in these visual entertainment media
2. List of films
Compile and share lists of films, favorite genre films, library of discs, or a watchlist of films that users want to see
3. Follow Friend's activities
Comment, Like or share user's Twitter and Facebook friends to see what films they're watching, enjoying and discussing.
4. An unregistered user can see New reviews, Popular reviews, Popular Lists and read them but cannot Like, Share, or comment on them.
5. In order to do have activities, Users need to sign up. After that they can create a movie Diary, List
of films, see New, Popular and Popular this week Reviews, List of activities, Follow and be followed by others, invite friends, rate films from 1 to 10, comment, share reviews, view List of Likes to Films, Reviews and Lists. But before doing anything they need to complete their Profile. This profile include their Contact information, Profile Picture, a short Biography, favorite Films and Social website accounts like Facebook and twitter.

User to Content map
In order to map a user to content we use personalized User Profile including their favorite movies, genre, List of watched movies and watch list, Email address and Social accounts like Facebook and twitter. So they will be informed by email whenever a new review is posted. In order to recommend the users, we use Filtering techniques based on user Profile and their list of favorite movies and genre. This techniques gather and filter information and reviews that are most related to user's interests.

User to user map
This system is going to be like a social website that users can have activities including like, comment, share, invite friends, follow others, be followed and so on. User to user map contains information about user's relationship followers and followings. So whenever a user's review, List or activity is liked, commented, followed or rated he or she should be informed by email.

Question2.
Popular goal of SQL injection attacks is to bypass authorization. Another goal is to carry out data manipulation or reading arbitrary data. (Unauthorized Reading)

Prevent Bypassing Authorization

Ruby on Rails has a built-in filter for special SQL characters, which will escape ' , '', NULL character and line breaks. Using Model.find(id) or Model.find_by_some thing(something)

automatically applies this countermeasure. *But in SQL fragments, especially* in conditions fragments (*where("..")*), the *connection.execute()* or *Model.find_by_sql()* methods, it does not do anything with conditions, limits, raw SQL and parameters it has to be applied manually.
In our web site we use query parameters like this for User Authentication:
>> User . find(:all, :conditions => ["login = ? AND password =?", params[:user_name], params[:password])
As you can see, the first part of the array is an SQL fragment with question marks. The sanitized versions of the variables in the second part of the array replace the question marks.

Prevent Unauthorized Reading
After getting unauthorized access attackers may read arbitrary data from the database. For example:
an attacker tries to read user name and password of other users.
Review.find(:all, :conditions => "author = '#{params[:author]}'")

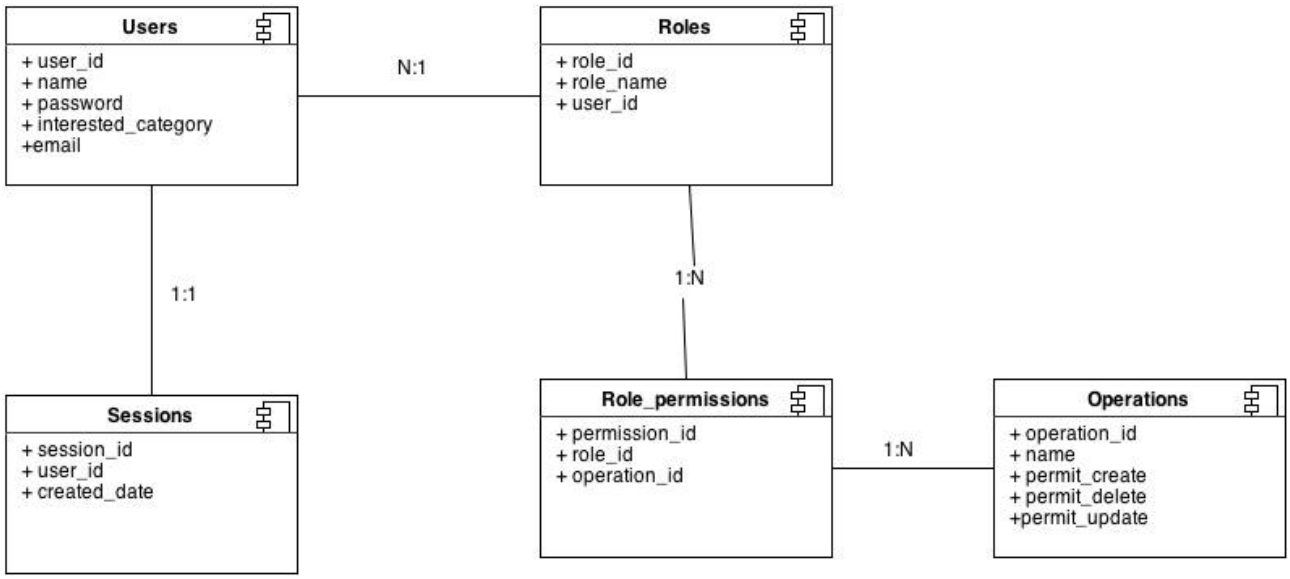
Now if an attacker input ') UNION SELECT id,login AS author,password AS title,1 FROM users

So the generated sql-query will be
SELECT * FROM reviews WHERE (author = '') UNION SELECT id,login AS author,password AS title,1 FROM users -- ')

Ruby on Rails filters special SQL characters, which will escape ' , " , NULL character and line breaks. Using Model.find(id) or Model.find_by_some thing(something) automatically applies this countermeasure.
For conditions we use 'sanitize_sql_for_conditions()' in our models

Question 3.
In our system, sensitive data is included: Database passwords,
We use Act_As_Secure Plugin to automatically encrypt sensitive data like Database passwords.

Users Data Model



Work Flow