Agenda

• Quick refresher on Project 2

• Recommendation system in Project 2

• CMUFLIX architecture & JSPs
Agenda

• Quick refresher on Project 2

• Recommendation system in Project 2

• CMUFLIX architecture & JSPs
Project 2

• CMUQFlix!

• A Movie Recommendation System
Project 2 Objectives

• Set up a front-end website with PostgreSQL as the back-end

• Allow users to login, “like” movies, and get personalized movies recommendations
Agenda

• Quick refresher on Project 2

• Recommendation system in Project 2

• CMUFLIX architecture & JSPs
Agenda

• Quick refresher on Project 2

• **Recommendation system** in Project 2

• CMUFLIX architecture & JSPs
Recommendation System: Case 1

• If a user $u$ has not yet “liked” any movies:
  • Display the top 5 “liked” movies in the database
Recommendation System: Case 2

• If a user $u$ has “liked” at least one movie:
  1. Find out what is the user $u$’s “movie clan” is
     • The user’s movie clan is the group of all users of have liked at least one movie $u$ is liked
     • In the figure, Anca’s movie clan would be:
       • Bob
       • Charlie
Recommendation System: Case 2

- If a user $u$ has “liked” at least one movie:
  2. Retrieve all the movies that have been liked by user $u$’s movie clan:
    - In the figure, for Anca, these movies are:
      - Argo
      - Troy
      - Bolt
Recommendation System: Case 2

- If a user $u$ has “liked” at least one movie:
  2. Retrieve all the movies that have been liked by user $u$’s movie clan:
    - In the figure, for Anca, these movies are:
      - *Argo* \( (\text{clan in-degree} = 3) \)
      - *Troy* \( (\text{clan in-degree} = 2) \)
      - *Bolt* \( (\text{clan in-degree} = 1) \)
Recommendation System: Case 2

- If a user $u$ has “liked” at least one movie:
  2. Retrieve all the movies that have been liked by user $u$’s movie clan:

- In the figure, for Anca, these movies are:
  - Argo (clan in-degree = 3)
  - Troy (clan in-degree = 2)
  - Bolt (clan in-degree = 1)

Anca has already liked this movie!
Recommendation System: Case 2

• If a user $u$ has “liked” at least one movie:

  2. Retrieve all the movies that have been liked by user $u$’s movie clan:

• The final list of recommendations for Anca:

  • *Troy* (clan in-degree = 2)
  • *Bolt* (clan in-degree = 1)

  • Top 5 with the largest clan in-degree
Agenda

• Quick refresher on Project 2

• Recommendation system in Project 2

• CMUFLIX architecture & JSPs
Agenda

• Quick refresher on Project 2

• Recommendation system in Project 2

• CMUFLIX architecture & JSPs
Architecture Reminder of CMUQFlix

- **Client Browser**
  - Front-end Tier (presentation)

- **Web Server**
  - Middle Tier (application logic)
    - Tomcat
    - Servlets, JSPs, Java

- **PostgreSQL Database**
  - Back-end Tier (data storage)
    - Movies, Users, ...

Connections:
- **HTTP** from Client Browser to Web Server
- **JDBC** from Web Server to PostgreSQL Database
Front-End Tier: HTML Forms

- HTML forms are a common way for
  - Getting user input at the front-end tier
  - Communicating the given input to the middle tier

- HTML forms are defined using the `FORM` tag

- `FORM` tag has three important attributes:
  - `NAME`: form name
  - `ACTION`: URI of the page to which the form is submitted
  - `METHOD`: method used to submit the form (Get*/Post**)

* A **Get** request gets (or retrieves) information (e.g. image, HTML page) from a server.
** A **Post** request posts (or sends) data from the client (e.g. usernames, passwords) to a server.
Front-End Tier: HTML Forms

• Three important tags used to get user input: **INPUT, SELECT, and TEXTAREA**

• **INPUT** is used to render text fields and buttons:

  ```html
  <INPUT TYPE="text" NAME="username" VALUE="John Brown">
  <INPUT TYPE="password" NAME="password" VALUE="123456">
  <INPUT TYPE="reset" VALUE="Reset Fields">
  <INPUT TYPE="submit" VALUE="Submit Form">
  ```

  | John Brown |
  |............|
  | Reset Fields | Submit Form |
Back-End Tier

- PostgreSQL:
  - We already know this very well!
Middle Tier: Apache Tomcat

- **Apache Tomcat** is an open-source web server developed by the Apache Software Foundation (ASF)

- Apache Tomcat is a servlet container i.e. it can invoke and execute servlets and JSPs (we’ll see what these are in a bit)

- Unlike web-servers that run on a TCP port number 80, **Tomcat runs on port number 8080**
Installing Tomcat

• Login to your VM (\(<\text{andrew\_id}\>@\(<\text{andrew\_id}\>\text{-db.qatar.cmu.local}\))
• Install JDK 7 (\(\text{apt-get install openjdk-7-jdk}\))
• Install Tomcat 7 (\(\text{apt-get install tomcat7 tomcat7-admin}\))
• Check that Tomcat is running
  • Type \(\text{service tomcat7 status}\), or
  • Browse to \(\text{http://}\(<\text{andrew\_id}\>\text{-db.qatar.cmu.local:8080}\)
• Manually start/stop/restart Tomcat
  (\(\text{service tomcat7 start/stop/restart}\))
• Deploy your application “cmuqflix” in Tomcat’s webapp
  (\(\text{cp -R cmuqflix /var/lib/tomcat7/webapps}\))
• View Tomcat’s log files (for debug info.)
  (\(\text{cd /var/log/tomcat7; cat *;}\))
JavaServer Pages (JSPs)

- A JavaServer Page (JSP) is primarily HTML that embeds java code to perform some application logic.
- A JavaServer Page can:
  - Interact with databases (using JDBC)
  - Generate custom HTML pages (as output)
  - Maintain session information for each client
JavaServer Pages (JSPs)

- A JavaServer Page (JSP) will consist of html, and “scriplets:”

<% your code here %>
JavaServer Pages (JSPs)

• A JavaServer Page (JSP) will consist of html, and "scriplets:"

  <% out.println("Hello world!"); %&gt;
JavaServer Pages (JSPs)

- **Session management**:
  - JSPs automatically have session tracking enabled!
  - Use the “session” variable:

```java
public Object getAttribute(String name)
```
This method returns the object bound with the specified name in this session, or null if no object is bound under the name.

```java
public void invalidate()
```
This method invalidates this session and unbinds any objects bound to it.

```java
public void setAttribute(String name, Object value)
```
This method binds an object to this session, using the name specified.