User Interface Software Project 4 Writeup

Divit Singh (divit52)

Sanchit Chadha (schadha)

This web application is for anyone who is looking to view and share their music with a community of music artists. Users can create an account to interact with this web application. As a teaser, when a user is not logged in, they can see the data in the community through the ‘Community’ page, however, they can’t access any of the data. Only when a user is logged in, can they view and play albums and tracks from different users. As it stands, our application supports all of the functional requirements specified in the documentation for this project. We have an index/home view that contains a basic overview of our service and login/create account functionalities.

We have three types of accounts for our application. The first account (admin) can only be set in the backend. The admin account has the ability to edit/delete any non-admin user, or add/edit/delete any album, track and comment from CollabTunes. In addition to this, an admin also can promote and demote any regular user to our second type of account: moderator. The moderator can delete any comments, tracks, and albums that is associated with an account. They can’t delete users and they can’t appoint regular users to be moderators as well. The only way for a user to become a moderator is if an admin specifically says so or if the user who is creating their account knows the moderator code (currently ‘test’). The third type of account is the regular user. They can upload, collaborate, comment, and browse other people’s works. They can only delete content that they own. In addition, if there are comments from other users that are on their page, they can delete them as well. As for the unregistered user, they can only view the albums that users from the community have uploaded. They can’t access the music or comments until they create an account with us.

We also have a profile page (My Music) where a user can edit all their information (except username). This page shows the albums that a user has under their name as well as giving them an option to add more content. This page also has a personal newsfeed. This newsfeed consists of only the events that are associated to that specific user. If the user clicks the community tab, they are directed to content from the community. In the community tab, there is a newsfeed that consists of the actions of their collaborators e.g. uploaded a new track, added a new comment etc. Our feed logs 5 types of events (technically 6): add album, add comment, add track, add collaborator (both ways, ie. Event of A collaborating with B and vice versa) and changing genre.

We decided to implement both posting comments as well as adding/removing friends. Friends in this case can be thought of as collaborators. Comments are ordered chronologically that include the author’s name, time created, as well as a link back to the author’s profile. A person can request to collaborate with anyone who they want to. The person who receives the request has the choice of declining or accepting their collaboration request. The incentive for collaboration is that it allows a user to add a track to their collaborator’s album. This connection is a bi-directional, permission based model such as Facebook. Of course, the owner of the album has the ability to delete unwanted tracks. As mentioned before, we implemented adding comments as well to the albums because we believe that it would promote more social interaction in our application.

When a user creates a new account, they have the ability to pick their favorite genre. This is presented in a drop-down list to satisfy one of the requirements for this project. We have also implemented data-validation in our forms that prevent incorrect or blank data from being entered. We have tested our code on both Mozilla and Chrome browsers and found that our code is compatible with both.

In regards to Amazon’s Mechanical Turk, we chose crowdsourcing for user testing because workers would be able to provide us with feedback for our website. Having workers look at images or try and populate the database wasn’t really a beneficial or rewarding task, therefore we chose user testing over data manipulation. In the HIT layout, we incorporated a question for the user to answer so that the user would actually have to use the web application and not provide generic feedback.

We obtained the feedback of 5 unique workers who generally all left positive remarks about the website design and layout. One particular worker said to incorporate ID3 tags from uploaded MP3 files to populate track information. We decided to not pursue that suggestion because our aim for the application is for users to be able to upload tracks that are created by the uploader and thus wouldn’t necessarily have ID3 information and the uploader would want to custom name their track anyways. Another worker stated that there was no text shown in the Collaborator section when the user doesn’t have any collaborators. We now show a message stating that there are no collaborators and to add collaborators through the community page.

Overall, the crowdsourcing testing really provided us with a confidence boost because most of the workers had positive comments to make about our website design and layout. This gave us motivation to continue on with the project, adding features and clean the layout even further. It was a good way of generating unbiased feedback.

URL: <http://ec2-54-173-61-131.compute-1.amazonaws.com/CS5774/CollabTunes/>

User Info

Username: schadha Password: test Admin

Username: divit52 Password: test Admin

Username: test Password: test Moderator

Username: kluther Password: test Regular User

SET UP (If running locally)

Assuming Xampp is installed correctly on your machine:

1. Unzip CollabTunes.zip to Xampp/htdocs folder
2. Inside, it contains a collabtunes.sql file
   1. Import this file into a new database called collabtunes
3. Edit the config.php file to set the correct SERVER\_PATH, DB\_PASS, and DB\_DATBASE
4. If file uploads do not work, then:
   1. For windows:
      1. Edit xampp/php/php.ini file
      2. Set upload\_max\_filesize=50M
      3. Set post\_max\_size=50M
      4. Restart your apache server
   2. For Mac OS X
      1. Edit /Applications/XAMPP/xamppfiles/etc/php.ini
      2. Set upload\_max\_filesize=50M
      3. Set post\_max\_size=50M
      4. Restart your apache server
5. Navigate to the localhost directory for this application in Chrome/Firefox and you should be good to go