Scoring

Each candidate will be evaluated on the decisions made when designing the programmatic elements of the application, which include:

* Appropriate use of native functionality
* Effective use of object-oriented design patterns
* Efficiency and elegance of programmatic design

Exam results will be distributed to each candidate by email, and will include feedback designed to highlight your strengths and areas for improvement.

General Requirements

Each candidate is expected to comply with the following general requirements for application development:

* The application must be the original work of the candidate. Candidates may explore best practices and patterns found in online resources.
* All application development must be done in free dev salesforce environment.
* Each candidate will need to create his or her own sample data set to support application development and manual testing.
* The application must scale for large data and user volumes. However, it is not necessary to load the data volumes stated; a sample data set is sufficient.
* The application must handle programmatic exceptions gracefully. Any end-user errors or exceptions should be presented in a user-friendly format.
* The user interface must be functional and meet the stated requirements, but time should not be spent creating an elaborate user interface as aesthetics will not be evaluated. The user interface should match the flow and structure of the wireframes provided.
* Comments within code are highly recommended as they may be useful to help better understand the function of the code developed during the evaluation process.
* Do not include any personally identifying data in the organization or code comments, with the exception of your email address on user records.

Company Overview

In a throwback to the 80s classic mix tapes, Camilla's Custom Vinyl (CCV) has cornered the market for custom vinyl records with your very own music mix! A custom vinyl record is perfect for audiophiles and other lovers of anachronistic technology. Customers call in to the CCV’s mix hotline to create their own record with the assistance of CCV’s team of music-savvy Mix Masters.

Project Overview

Camilla’s Custom Vinyl has experienced a period of rapid growth in their sales pipeline. Their IT infrastructure has been stretched to the limit and CCV is rebuilding their systems on the Force.com platform. They have the data and security model built out and now need the expertise of a Force.com Developer to build out some automated update logic and custom user interfaces for them.

This document outlines the current state of the application and the remaining requirements the candidate will need to fulfill. It should be assumed by the candidate that no significant declarative configuration changes are necessary to fulfill the requirements; only code development and very minimal configuration of a mechanism to invoke the code will be required.

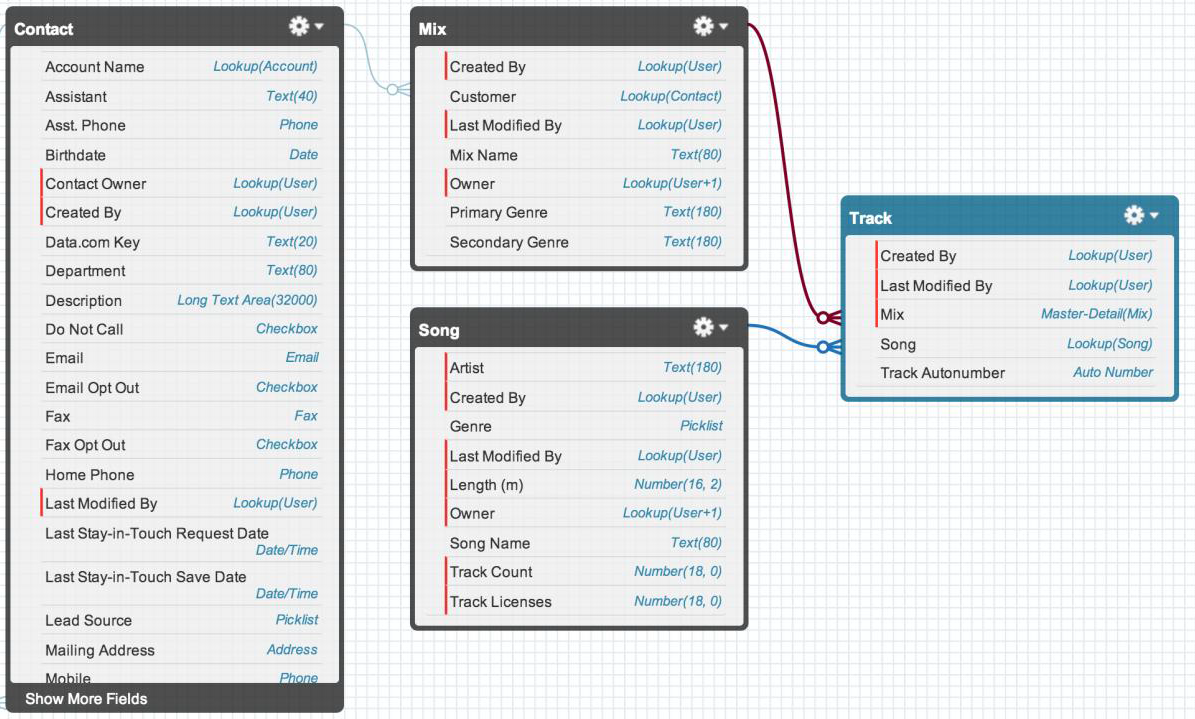
Data Model Overview

This section of the document details the data model of the application, for the candidate’s reference.

CCV uses four object types in its organization: Contact (standard), Mix (custom), Song (custom), and Track (custom). Each CCV customer has a corresponding Contact record in the system. The CCV song library is stored in the Song table. Mix records are created for a particular customer to be pressed into vinyl. Track records are created to indicate which Songs are included in the Mix.

Entity Relationship Diagram

Below is the Entity Relationship Diagram (ERD) of the application being developed. The ERD depicts all of the objects that play a significant role and are needed to understand the application requirements.



Data Requirements

Contact (Contact)

Contacts are CCV customers. Sharing is public read/write.

Song (Song\_\_c)

Songs from new and classic artists are available for inclusion in customer mixes. Songs are licensed from a recording company and may only be sold in specific areas and included in a set number of mixes. Sharing is public read/write.

Key Fields:

Song Name (Name) – Name of the song. Name field.

Artist (Artist\_\_c) – Name of the artist. Text field.

Length (m) (Length\_m\_\_c) – Length of the song in minutes. Number field.

Genre (Genre\_\_c) – Genre of the song. Picklist field.

Track Count (Track\_Count\_\_c) – Number of times a song is used as a track. Candidate is responsible for updating this field. Number field.

Track Licenses (Track\_Licenses\_\_c) – Number of times CCV is licensed to include the song in a mix. A validation rule (should be created) ensures the track count never exceeds the number of available track licenses. Number field.



Mix (Mix\_\_c)

A custom collection of tracks for a customer that will be pressed onto vinyl. Sharing is private.

Key Fields:

Mix Name (Name) – Name of the customer’s mix. Name field.

Customer (Customer\_\_c) – Contact purchasing the mix. Lookup to contact.

Primary Genre (Primary\_Genre\_\_c) – Primary genre of the mix. Candidate is responsible for updating this field. Text field.

Secondary Genre (Secondary\_Genre\_\_c) – Secondary genre of the mix. Candidate is responsible for updating this field. Text field.



Track (Track\_\_c)



An individual song that is part of a mix. Sharing is controlled by parent.

Key Fields:

Track Auto Number (Name) – Auto number to globally identify a track in the database. Name field.

Mix (Mix\_\_c) – Mix this track is part of. Re-parentable master-detail to mix.

Song (Song\_\_c) – Song for this track. Lookup to song.

Data Volumes



The application must scale to handle the following data volumes:



Objects

Contacts – 1,000s

Songs – 1,000s

Mixes – 1,000s

Tracks – 100,000s

Relationships

Tracks per Mix – Max of 20

Tracks per Song – Average of 100

Application Users

There are two primary user groups for the application:

**Library Managers**

Responsible for securing contracts with music publishers to license their tracks for use in CCV mixes Mix

**Masters**

Work with customers on the phone to build their custom mix using the Mix Builder

Are music experts able to help customers find and choose the right songs for the mix

Profiles

Library Managers

Can create, read, update, delete all Songs Cannot see Contacts, Mixes, or Tracks

Mix Masters

Can read all Songs

Can create, edit, update and delete Contacts, Mixes, and Tracks

Logic Requirements

This section of the document details the automation that the candidate is required to implement programmatically.

Mix

The following fields on the Mix object should be automatically updated:

Primary Genre – Most frequent genre of songs included in the album. Leave blanks for Mixes with no Tracks. Ties are broken alphabetically (A-Z case-insensitive).

Secondary Genre – Second most frequent genre of songs included in the album. Leave blank for Mixes with no Tracks or with all Tracks of the same genre. Ties are broken alphabetically (A-Z case- insensitive).

Song

The following field on the Song object should be automatically updated:

Track Count – Number of tracks related to the song

When to Update

All changes to the Track object that can change the three automatically updated fields should be taken into account. Specifically, the candidate should handle required updates when Tracks are created, deleted, restored from the recycle bin, or have their song and mix relationships changed or cleared (only for song relationship).

The candidate can assume for the purposes of their submissions that a Song's genre will not change, and hence changes to the Song object do not need to be tracked.

For example, changing a Song's genre does not require recalculating the primary or secondary genre for any Mixes with Tracks related to that Song.

In the event of any errors that prevent updating the automatically updated fields candidate should ensure that the result of the transaction does not lead to the value of the fields becoming invalid.

Updates to the automatically updated fields should occur in real-time and support bulk data uploads.

**Note**: Historical data is often loaded from businesses acquired by CCV and track info may be missing the related song. Assume the relationship is a design constraint.

User Interface Requirements

This section of the document details user interface-driven functionality that candidates must implement programmatically.

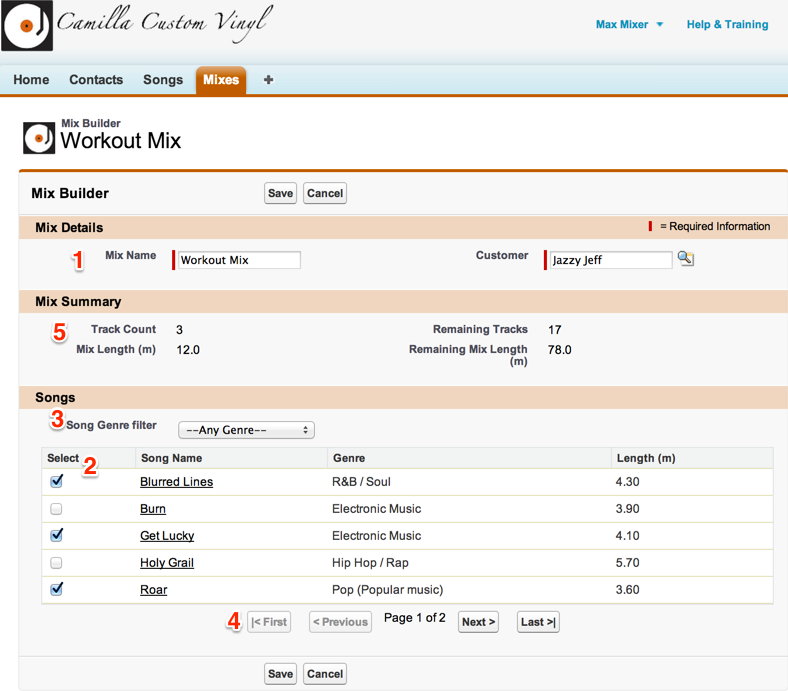
Mix Builder

Camilla Custom Vinyl would like to reduce the number of clicks needed by Mix Masters to build a Mix by developing a new user interface (UI). The new UI should replace the standard Mix create and edit pages.

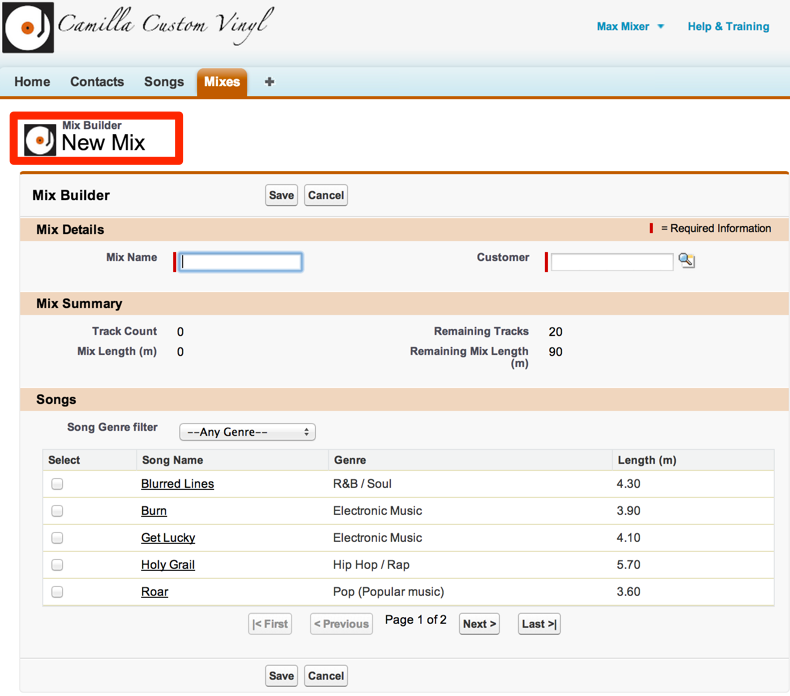
Mix Masters will create Mixes by clicking the New button from the standard Mix tab, the New button from the Mix related list on Contact records, or by using the Quick Create menu in the sidebar.

The following mock-ups attempt to portray the desired outcome of the user interface.

Note: The user interface has been simplified for the purpose of the programming assignment, and does not necessarily include all the elements that would be present in a fully functional interface. Pagination for the new user interface (UI) is required although the look and feel is up to the developer.

Figure 1 – Edit Mix View

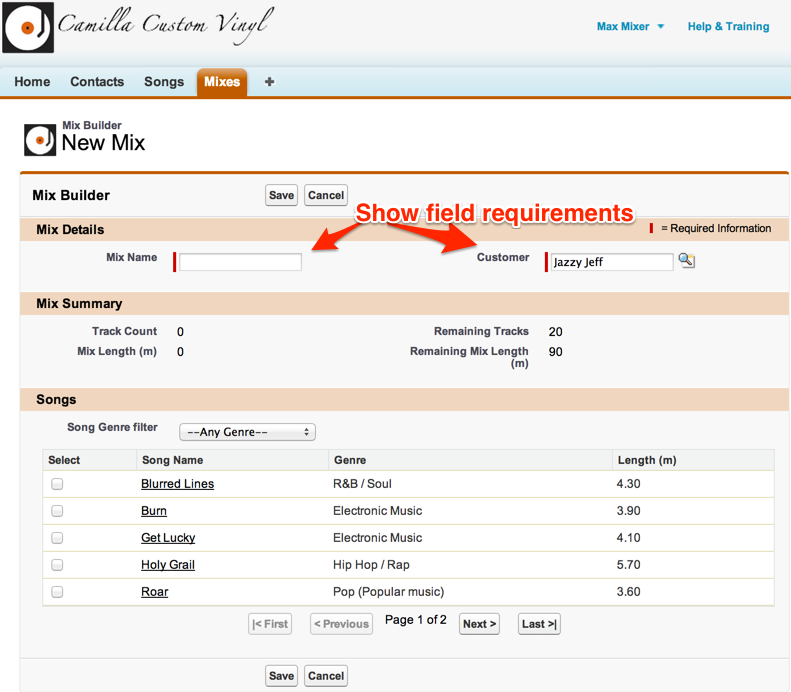
As shown in Figure 1, the Mix Builder will include the ability to (1) edit the Mix details, (2) add and remove Songs from the Mix, (3) filter the Songs displayed by genre, (4) page through the Song results, and (5) see calculated summary stats about the Mix.

Figure 2 – New Mix View

As shown in Figure 2, the UI will display the name of the Mix in the header when editing existing mixes.

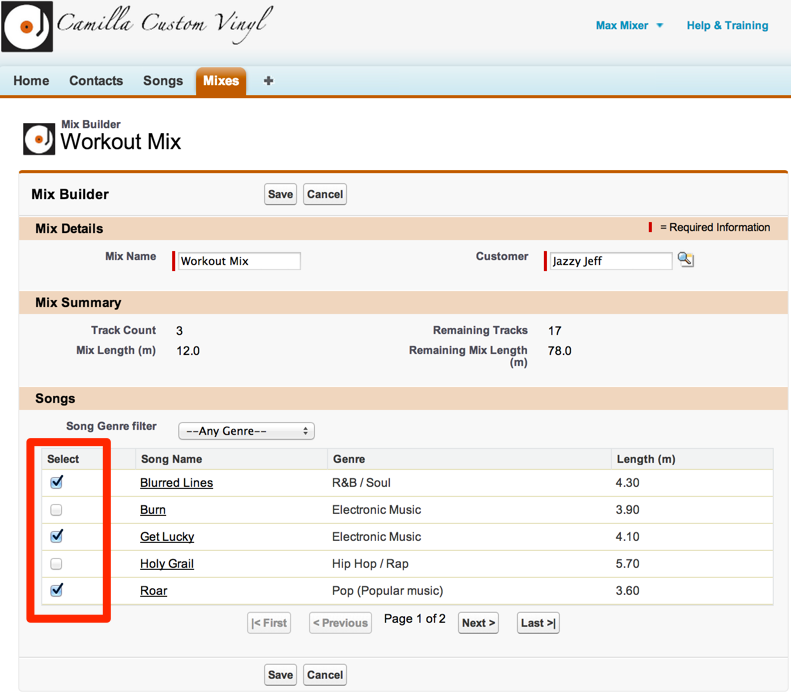
For new mixes the header should display “New Mix”.

Mixes, which have a private sharing model, are not editable except by the Mix owner or users the Mix has been shared with. The UI should not allow users to edit Mixes they don’t have edit permissions for.

Figure 3 – Field Requirements

As shown in Figure 3, users will be able to edit the Mix Name and Customer fields and the user interface should indicate both fields are required.

For Mixes created from the related lists of a Contact, the customer lookup should be pre-populated with that contact. When editing Mixes the current values for Mix Name and Customer should be displayed.

Figure 4 – Song Results

As shown in Figure 4, users will be able to add and remove the Songs included as Tracks on the mix.

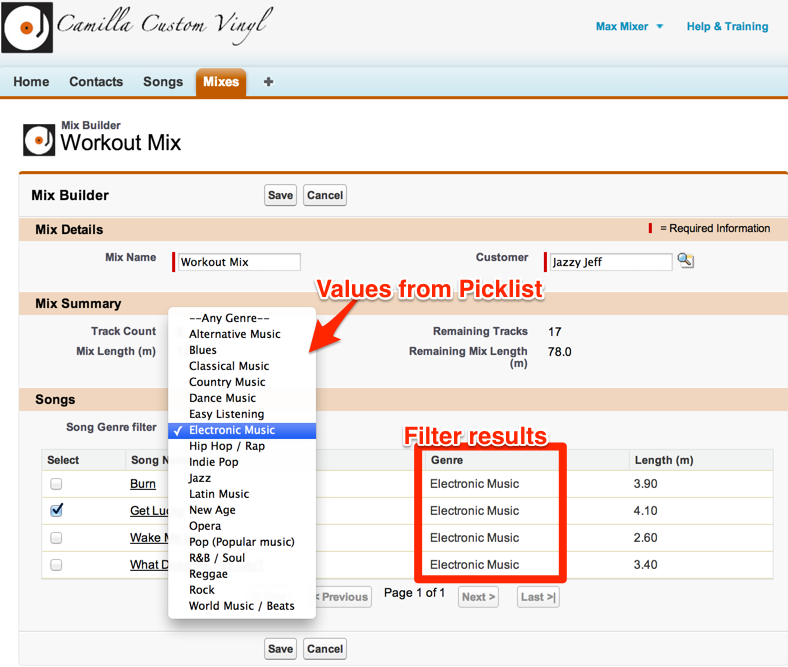
When a Mix is edited Songs that are included as Tracks should show up as selected in the Song results.

Songs should be presented in alphabetical order from A-Z (case-insensitive). Song results should include the following columns:

Select – Checkboxes for selecting/de-selecting songs

Song Name - Song Name hyperlinked to the Song detail page opening in a new tab Genre – Genre of the song

Length (m) – Length of song in minutes

Figure 5 – Genre Filter

As shown in Figure 5, users will be able to filter the Songs shown by genre. The options for the genre filter should be driven dynamically based on available picklist values and include one additional option for “Any Genre” that appears first in the list.

When the Mix Builder first loads the filter should be set to “Any Genre” and display all Songs. Song selections should be preserved when using the genre filter.

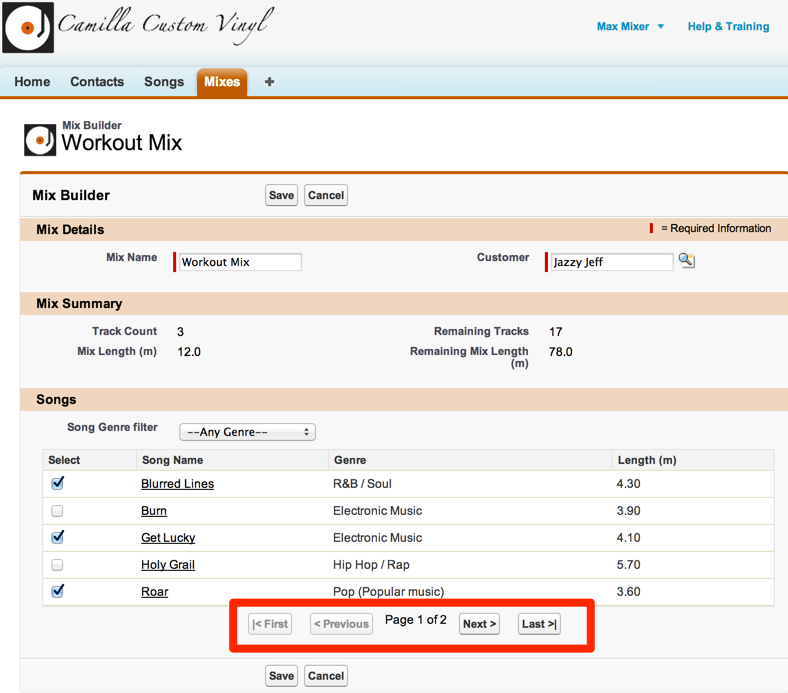


Figure 6 - Pagination

As shown in Figure 6, users should be able to paginate through the songs five at a time. Pagination controls should be available for going to the first page, the previous page, the next page, and the last page of results. If on the first page, the first and previous buttons should be disabled. If on the last page, the next and last buttons should be disabled. If there is only one page of results, the first, previous, next, and last buttons should be disabled. The current page number should be displayed as well as the total number of pages of results.

Song selections should be preserved while changing pages.



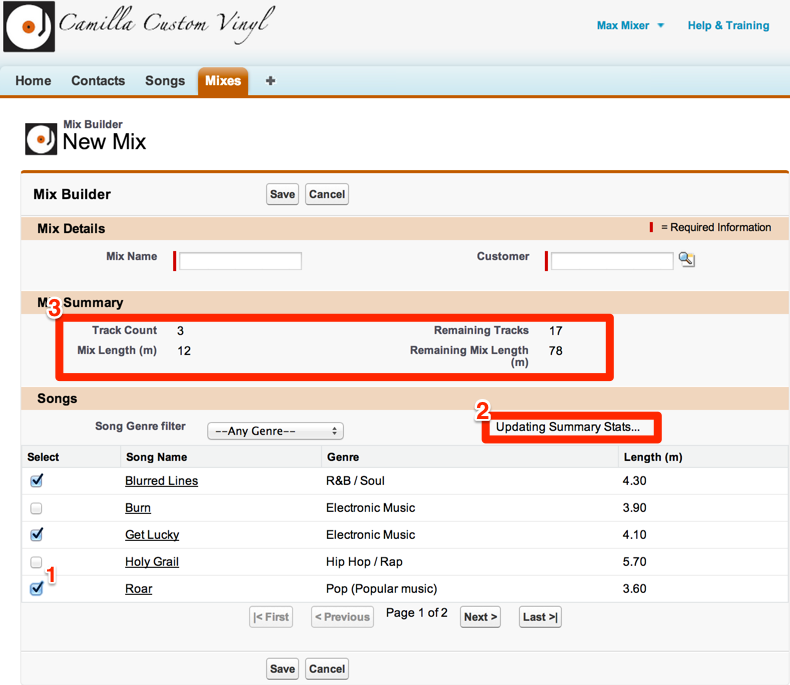


Figure 7 – Mix Summary

As shown in Figure 7, a summary section will display the following stats about the current Mix:

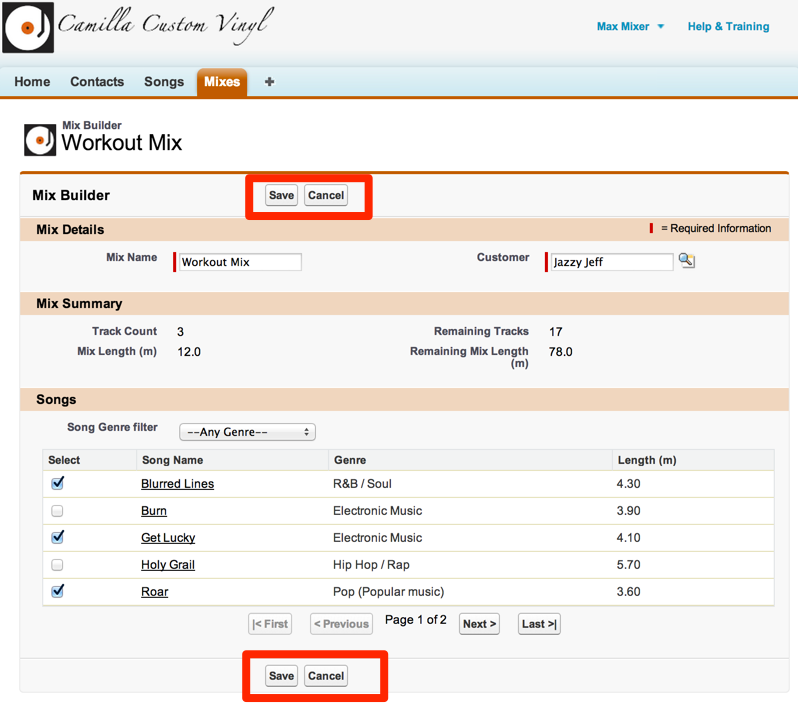
Track Count – The number of Songs included in the Mix

Remaining Tracks – The number of remaining Songs that can be added to the Mix out of a max of 20

Mix Length (m) – The total length in minutes of the Songs included in the Mix

Remaining Mix Length (s) – The number of remaining minutes for additional Songs out of a max of 90.

The summary stats should be defaulted when the Mix Builder loads and updated whenever a Song is selected or de-selected. A status message should be shown when summary stats are being updated.

Figure 8 – Save and Cancel Buttons

As shown in Figure 8, the Mix Builder will have buttons to Save or Cancel the mix.

When the cancel button is clicked no changes to the Mix should be saved and the user returned to the Mix detail record for existing Mixes and to the Mix tab for new Mixes.

When the save button is clicked the following events should occur:

* Validate that required fields have been populated.
* Validate that the max number of tracks or max mix length have not been exceeded. Note: the max number of tracks and max mix length may be exceeded while building a Mix.
* Save changes to the Mix record
* Create Track records for Songs added to the Mix
* Delete Track records for Songs that have been removed from the Mix

If either validations fail or any database operations fail all committed changes should be rolled back. For example, if there was an exception while deleting a Track record and changes to the Mix had already been saved those changes should be rolled back.

For successful saves the user should be returned to the Mix detail page.



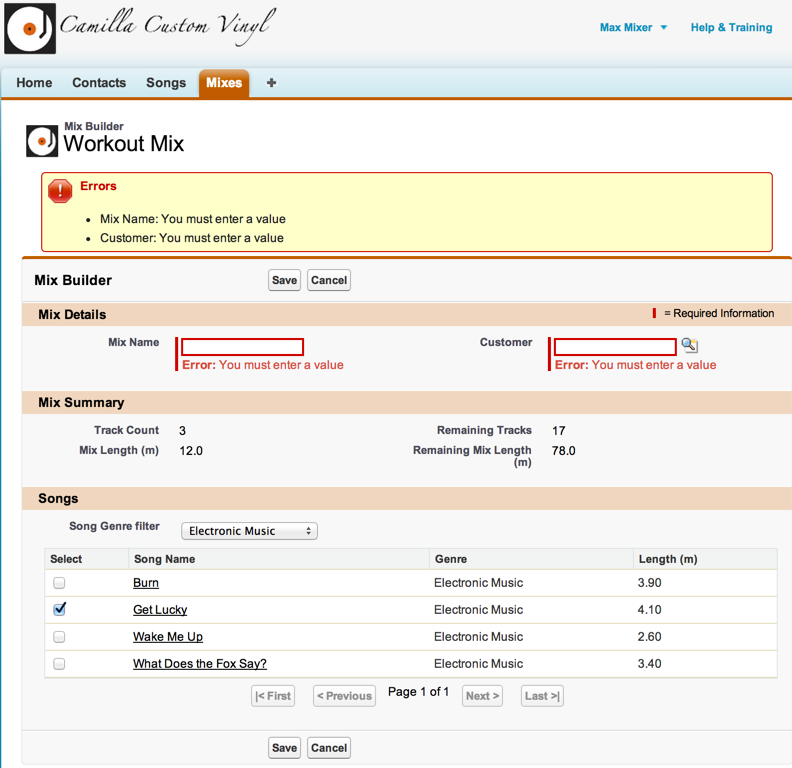


Figure 9 – Error Messages

As shown in Figure 9, meaningful, user-friendly error messages should be displayed to the user if an error occurs at any time. For the required fields, mix name and customer, an additional error message should be displayed below the field if not filled out on save.

Testing Requirements

Apex Tests

The candidate is responsible for writing Apex tests to validate the application behavior against business requirements. The tests should assert that the automated logic and user interface function correctly, validate application security, and ensure that the application scales for large data volumes. The application must meet the code coverage requirements for deployment into a full production organization. Apex tests should be written to be data- and-organization-independent.