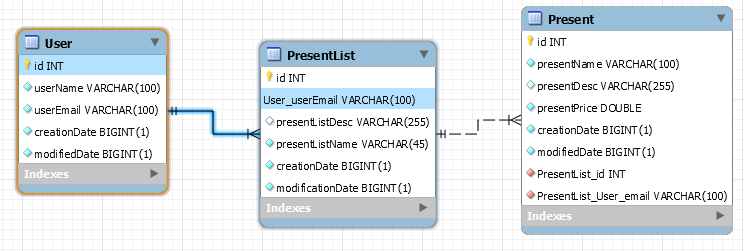
Ryan Nitz  
Danny Gallant  
Feb 7, 2021  
CS3103

Project deliverable Part B: Database Design  
ERD and Procedures for Present Directory.

ERD:

User has 0 or many PresentLists.  
PresentList has 0 or many Presents.  
Having this PresentList table makes the most sense for extensibility in every aspect.



Procedures

All enpoints except for gets are authorized requests.

User Table:

* **createUser(User user); return user.** 
  + **endpoint: post:/user/<email>**
  + This will take in all params needed to create a new user, then return the newly created record.
* **updateUser(User user); return user** 
  + **endpoint: put:/user/<email|id>**
  + This will take in all params to update a user, then return the updated record.
* **deleteUser(int id)** 
  + **endpoint: delete:/user/<email|id>**
  + This is will delete the user with the specified id.
* **getUserById(int id); return user =**
  + **endpoint: get:/user/<email|id>**
  + This will get the user record based on the id.
* **getUserByEmail(varchar email); return user** 
  + **endpoint: get:/user/<email|id>**
  + This will get the user record based on the email.

PresentList Table:

* **createPresentList(PresentList presentList); return presentList**
  + **endpoint: post:/presentlist**
  + This will take in all params needed to create and new presentList, then return the newly create record.
* **updatePresentList(PresentList presentList); return presentList**
  + **endpoint: put:/presentlist**
  + This will take in all params needed to update a presentList record, then return the updated record.
* **deletePresentList(int presentListId);**
  + **endpoint: delete:/presentlist/<id>**
  + This will delete the presentList with the specified id
* **getPresentListById(int presentListId) return presentList;**
  + **endpoint: get:/presentlist/<id>**
  + This will return the presentList record based on the id.
* **getPresentListByUserEmail(varchar userEmail); return n presentLists**
  + **endpoint: get:/presentlists/<email>**
  + This will return all presentLists that belong to a certain user based on email

Present Table:

* **createPresent(Present present) return present;**
  + **endpoint: post:/present**
  + This will take in all params needed to create a new present, then return the newly created record.
* **updatePresent(Present present) return present;**
  + **endpoint: put: /present**
  + This will take in all params needed to update a present, then return the udated record.
* **deletePresent(int presentId)** 
  + **endpoint: delete:/present/<id>**
  + This will delete the present with the specified id
* **getPresentById(int presentId) return present**
  + **endpoint: get:/present/<id>**
  + This will retrun the present record based on the id
* **getPresentsByUserEmail(varchar userEmail) return n presents**
  + **endpoint:**
  + This will return all presents that belong to a certain user, based on email