Section 1.1 Exemplar Code : Iteration 2

With Iteration 2, we have continued to use appropriate coding practices in order to maximize readability of our code, along with keeping these methods consistent across the system. Here we emphasize our new features : a favourites system, ratings system, user view etc.

```
171 ############## Favourite bookmark ##############
172 get '/favourite' do
173   #Make sure user is logged in
174   if session[:user].nil?
175     redirect '/loginrequired'
176   else
177     @title = 'My favourite'
178     erb :favorite
179   end
180 end
181
182 post '/favorite' do
183   if session[:user].nil?
184     redirect '/loginrequired'
185   else
186     #Add favorite with reference to user and bookmark
187     result = User.toggleFavorite(session[:user][:id],params[:bid])
188     if result
189       redirect '/'
190     else
191       @title = 'Failed to favorite.'
192       @message = 'Please try again!'
193       erb :general_error
194     end
195   end
196 end
```

Listing A Favourites Handler (models/user.rb)

```
3  module Comment
4
5      DB = SQLite3::Database.new '../models/bookmark_project.sqlite'
6
7      # Create new entry:
8      def Comment.new(id,comment_substance,attributed_to,created_by,date_created)
9          query = "INSERT INTO comments VALUES (?,?,?,?,?);"
10         begin
11             DB.execute query,id,comment_substance,attributed_to,created_by,date_created
12             result = true
13         rescue SQLite3::ConstraintException
14         end
15     end
```

Listing B Adding new comment (models/comments.rb)

```
<main>
  <table>
    <tr>
      <th>User ID</th>
      <th>First Name</th>
      <th>Last Name</th>
      <th>Date Joined</th>
      <th>Email </th>
      <th>Suspended?</th>
    </tr>
    <% page = params[:page].nil? ? 1 : params[:page].to_i # get the current page
    users = User.selectAll(page) %>
  <% if not users.nil? %>
    <% users.each do |user| %>
      <tr>
          <!-- Selecting all of the user data to be displayed in each column. -->
        <td><%= user[:user_id]%></td>
        <td><%= user[:firstname]%></td>
        <td><%= user[:lastname]%></td>
        <td><%= user[:date_joined]%></td>
          <td><a href="/update_profile?uid=<%=user[:user_id]%>"><%= User.findEmail(user[:user_id])[0][:email]%></a></td>
        <td><% if user[:suspended] != 1 %>
              No
            <% else %>
              Yes
            <% end %>
            </td>
      </tr>
    <% end %>
  <% end %>
  </table>
```

Listing C User view (views/admin.erb)

```
### Update profile ###
get '/update_profile' do
  # Error page if the logged in user and update user not the same
  if session[:user].nil?
    redirect '/loginrequired'
  elsif session[:user][:id].to_i == params[:uid].to_i
    erb :update_profile
  elsif  session[:user][:admin] == true
    erb :update_profile_admin
  elsif session[:user][:id].to_i != params[:uid].to_i
    @title = 'You cannot perform such action!'
    @message = 'You cannot modify other\'s profile!'
    erb :general_error
  end
end

post '/update_profile' do
  #Check a user is signed in
  if session[:user].nil?
    redirect '/loginrequired'
  #Check if the user is editing their own profile
  elsif session[:user][:id].to_i == params[:uid].to_i

    @fname = params[:fname]
    @lname = params[:lname]
    @department = params[:department]
    @email = params[:em]
    @suspended = params[:suspended]==nil ? 0 : 1

    result = User.updateDetail(params[:uid],@fname,@lname,@department,@email,@suspended)
    if result
      @message = 'updated the profile!'
      # update info in session
      session[:user][:name] = User.findName(session[:user][:id])
      erb :success
    else
      @title = 'Something\'s wrong!'
      @message = 'Please try again.'
      erb :general_error
    end
```

Listing D Update profile handler (app/user.rb)

```ruby
module Tag

  DB = SQLite3::Database.new '../models/bookmark_project.sqlite'

  # Create a new tag with name, return id
  def Tag.new(tag_name)
    addresult = false
    query = "INSERT INTO tag (tag_name) VALUES (?);"
    begin
      DB.execute query,tag_name
      addresult = true
    rescue SQLite3::ConstraintException
    end
    if addresult
      query = 'SELECT last_insert_rowid();'
      result = DB.execute query
    end
    return result
  end
```

Listing E Tag creation (models/tag.rb)

```ruby
post '/update_bookmark' do
  # Make sure the user is an admin or the user that created the bookmark
  if session[:user].nil? then redirect '/loginreqired' end
  @bookmark = Bookmark.getBookmark(params[:id])
  if @bookmark.length == 0
    @title = 'No such bookmark'
    @message = 'Sorry, cannot find the bookmark.'
    erb :general_error
  else
    @bookmark = @bookmark[0]
    #Check that it is the user who created the bookmark who is updating it
    if session[:user][:id] !=  @bookmark[:created_by]
      @title = 'You cannot perform such action!'
      @message = 'You cannot modify other\'s profile!'
      erb :general_error
    else
      #Insert all information into database
      pub = params[:pub].nil? ? 1:0;
      tags = params[:tags].split(",");
      create_tags = params[:createTags].split(",")
      result = Bookmark.updateBookmark(params[:id],params[:title],urlConversion(params[:url]),params[:description],pub,tags,create_tags)
      if result
        @title = 'Success!'
        @message = 'Updated this bookmark'
        erb :success
      else
        @title = 'Failed to update bookmark'
        @message = 'Sorry, something went wrong'
        erb :update_bookmark
      end
    end
  end
end
```

Listing F Update bookmark (app/index.rb)

```ruby
    ##################  Favorite  ##################
    # List all the favorite bookmark id of specific user
    def User.listFavorite(user_id)
      query = 'SELECT bookmark_id FROM user_favorite WHERE user_id = ?'
      result = []
      rows = DB.execute query, user_id
      if rows.length != 0
        rows.each do |row|
          result.push(row[0])
        end
      end
      return result
    end

    # Add or delete favorite for a user
    def User.toggleFavorite(user_id,bookmark_id)
      result = false
      check = 'SELECT COUNT(*) FROM user_favorite WHERE user_id = ? AND bookmark_id = ?'
      #make sure user and bookmark ID are accessible
      if user_id && bookmark_id
        number = DB.execute(check,user_id,bookmark_id)[0][0]
        #If favorite is being added, adjust table
        if number.to_i == 0
          query = 'INSERT INTO user_favorite (user_id,bookmark_id) VALUES (?,?)'
        #If favorite is being deleted, adjust table
        elsif number.to_i == 1
          query = 'DELETE FROM user_favorite WHERE user_id = ? AND bookmark_id = ?'
        end
        begin
          DB.execute query,user_id,bookmark_id
          result = true
        end
      end
      return result
    end
```

Listing G User favorite functions (models/user.rb)