

Exemplar code Iteration 1

A. login.rb

```
1 |VALID_EMAIL_REGEX = /\A([\w+\-].?)+@[a-z\d\-]+(\.[a-z]+)*\.[a-z]+\z/i
2
3 #get logout request and redirect to index page
4 ▾ get '/logout' do
5   session.clear
6   flash[:info] = "You have been successfully logged out"
7   redirect '/index'
8 end
9
10
11 #get login request and redirect to login page
12 ▾ get '/login' do
13   redirect '/index' if session[:logged_in]
14   erb :login
15 end
16
17 ▾ post '/login' do
18   session.clear
19   @login = true
20
21
22   @email = params[:email]
23   @pass = params[:password]
24   @userFound = User.find_user(@email, @pass)
25
26   @email_ok = !@email.nil? && @email != "" && @email =~ VALID_EMAIL_REGEX
27   @pass_ok = !@pass.nil? && @pass != ""
28   @all_ok = @email_ok && @pass_ok
29
30   if @userFound
31     #Gather all user data
32     query = "SELECT * FROM users WHERE email = ?;"
33     @userInfo = $db.execute query, @email
34     @active = @userInfo[0][5]
35     puts @active
36   end
37
38   if @active == 1
39     session[:logged_in] = true #User now logged in
40     session[:admin] = false #User is not an admin until confirmed
41
42     # store user info in session data
43     session[:email] = @userInfo[0][4]
44     session[:user_id] = @userInfo[0][0]
45     session[:first_name] = @userInfo[0][1]
46
47     if @userInfo[0][6] == 1
48       session[:admin] = true #User is an admin
49       redirect '/admin'
50     end
51
52     flash[:notice] = "You have successfully signed in"
53     redirect '/index'
54   else
55     session[:logged_in] = false
56     erb :login
57   end
58 end
59 end
60
```

B. user.rb

```
1 ▾ module User
2
3   def User.find_user(email, pass)
4     result = false
5     query = "SELECT password FROM users WHERE email = ?;"
6     rows = $db.execute query, email
7     if (rows.count == 1)
8       hash_check = BCrypt::Password.new(rows[0][0])
9       result = hash_check.is_password?(pass)
10    end
11    return result
12  end
13
14
15  def User.logged_in?(session)    #not sure if we need separate function for this
16    session.nil? ? false : true  #we can verify if user is logged using function find_user
17  end
18
19  def User.new(user_id, first_name, last_name, email, password)
20    result = false
21    query = "INSERT INTO users (user_id, first_name, last_name, password, email, active_status,admin) VALUES(?, ?, ?, ?, ?,?,?);"
22    begin
23      $db.execute query, user_id, first_name, last_name, BCrypt::Password.create(password),email,0,0
24      result = true
25    rescue SQLite3::ConstraintException
26      puts "Insertion Failed"
27    end
28    return result
29  end
30 end
31 end
```

C. bookmarks.rb

```
1 module Bookmark
2
3   def Bookmark.new(bookmark_name, link, description, creator, last_updated)
4     result = false
5     insert = "INSERT INTO bookmarks (bookmark_name, link, description, creator, last_updated, report_status) VALUES (?, ?, ?, ?, ?, ?)"
6     begin
7       $db.execute insert, bookmark_name, link, description, creator, last_updated, 0
8       result = true
9       puts "Insertion success"
10    rescue SQLite3::ConstraintException
11      puts "Insertion Failed"
12    end
13    return result
14  end
15
16  def Bookmark.find_by(search)
17    result[]
18    db = SQLite::Database.new './database/acme_db.sqlite'
19    query = "SELECT"
20  end
21
22 end
```

D. new_bookmark.rb

```
1 get '/new_bookmark' do
2   redirect '/index' if !session[:logged_in]
3   erb :new_bookmark
4 end
5
6 post '/new_Bookmark' do
7   redirect '/index' unless session[:logged_in]
8
9   #date-time format (YYYY/MM/DD HH:MM)
10  @last_updated = Time.now.strftime("%Y/%m/%d %H:%M").to_s
11
12  @bookmark_name = params[:bookmark_name]
13  @link = params[:link]
14  @description = params[:description]
15  @report_status=0
16
17  Bookmark.new(@bookmark_name, @link, @description, session[:user_id], @last_updated)
18
19  #   add_bookmark = "INSERT INTO bookmarks (bookmark_name, link, description, creator, last_updated) VALUES (?, ?, ?, ?, ?)"
20
21  #   $db.execute add_bookmark, params[:bookmark_name], params[:link], params[:description], session[:user_id], @last_updated
22
23  redirect '/'
24 end
```

E. bookmark_table.erb

```
1 <table class="table">
2   <thead>
3     <tr>
4       <th>No.</th>
5       <th>Bookmark Name</th>
6       <th>Link</th>
7       <th>Created by</th>
8       <th>Last Updated</th>
9       <th>Status</th>
10      <th>More Info</th>
11    </tr>
12  </thead>
13
14  <tbody>
15    <% @bookmark_list.each do |bookmark| %>
16      <tr>
17
18
19        <td><%= bookmark[0] %></td>
20        <td><%= bookmark[1] %></td>
21        <td><a class="btn btn-info btn-xs" href="<%= bookmark[2] %>">Link</a></td>
22        <td><%= ($db.execute "SELECT email FROM users WHERE user_id = ?",bookmark[4])[0][0] %></td>
23        <td><%= bookmark[5] %></td>
24        <td><%= bookmark[6] %></td>
25        <td><a class="btn btn-info btn-xs" href="<%= bookmark[3] %>"> More Info</a></td>
26      </tr>
27    <% end %>
28  </tbody>
29 </table>
```

F. createDatabase.rb

```
1 |require 'sqlite3'
2 |require 'csv'
3
4 |csv_text_feedback = File.read("public/csv/feedback.csv")
5 |csvFeedback = CSV.parse(csv_text_feedback, :headers => true)
6
7 |csv_text_bookmark_history = File.read("public/csv/bookmark_history.csv")
8 |csvBookmarkHistory = CSV.parse(csv_text_bookmark_history, :headers => true)
9
10 |csv_text_bookmarks = File.read("public/csv/bookmarks.csv")
11 |csvBookmarks = CSV.parse(csv_text_bookmarks, :headers => true)
12
13 |csv_text_users = File.read("public/csv/users.csv")
14 |csvUsers = CSV.parse(csv_text_users, :headers => true)
15
16 |csv_text_favourites = File.read("public/csv/favourites.csv")
17 |csvFavourites = CSV.parse(csv_text_favourites, :headers => true)
18
19 |csv_text_login_history = File.read("public/csv/login_history.csv")
20 |csvLoginHistory = CSV.parse(csv_text_login_history, :headers => true)
21
22 |csv_text_tags = File.read("public/csv/tags.csv")
23 |csvTags = CSV.parse(csv_text_tags, :headers => true)
24
25 |csv_text_tagged_bookmarks = File.read("public/csv/tagged_bookmarks.csv")
26 |csvTaggedBookmarks = CSV.parse(csv_text_tagged_bookmarks, :headers => true)
27
28 |inputs = ARGV
29 |if inputs[0] == "testdb"
30 |  DB = SQLite3::Database.new 'database/acme_test_db.sqlite'
31 |else
32 |  DB = SQLite3::Database.new 'database/acme_db.sqlite'
33 |end
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83 |DB.execute <<--SQL
84 |CREATE TABLE IF NOT EXISTS "bookmarks" (
85 |  "bookmark_id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
86 |  "bookmark_name" TEXT,
87 |  "link" TEXT UNIQUE,
88 |  "description" TEXT,
89 |  "creator" INTEGER,
90 |  "last_updated" TEXT,
91 |  "report_status" INTEGER DEFAULT 0,
92 |  "rating" INTEGER DEFAULT 0,
93 |  "num_ratings" INTEGER DEFAULT 0
94 |);
95 |SQL
96
97 |DB.execute <<--SQL
98 |CREATE TABLE IF NOT EXISTS "users" (
99 |  "user_id" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
100 |  "first_name" TEXT,
101 |  "last_name" TEXT,
102 |  "password" TEXT NOT NULL,
103 |  "email" TEXT NOT NULL,
104 |  "active_status" INTEGER NOT NULL DEFAULT 1,
105 |  "admin" INTEGER NOT NULL DEFAULT 0
106 |);
107 |SQL
108
109
110
111
112
113
114
115
116
117
118
119
120 |csvFeedback.each do |row|
121 |  DB.execute "insert into feedback values ( ?, ?, ?, ?, ?)", row.fields
122 |end
123
124 |csvTaggedBookmarks.each do |row|
125 |  DB.execute "insert into tagged_bookmarks values ( ?, ?)", row.fields
126 |end
127
128 |csvBookmarks.each do |row|
129 |  DB.execute "insert into bookmarks values ( ?, ?, ?, ?, ?, ?, ?, ?, ?)", row.fields
130 |end
131
132
133 |csvBookmarkHistory.each do |row|
134 |  DB.execute "insert into bookmark_history values ( ?, ?, ?)", row.fields
135 |end
136
137 |csvUsers.each do |row|
138 |  DB.execute "insert into users values ( ?, ?, ?, ?, ?, ?, ?)", row.fields
139 |end
140
141 |csvTags.each do |row|
142 |  DB.execute "insert into tags values ( ?, ?)", row.fields
143 |end
144
145
146
147 |csvLoginHistory.each do |row|
148 |  DB.execute "insert into login_history values ( ?, ?)", row.fields
149 |end
150
151 |csvFavourites.each do |row|
152 |  DB.execute "insert into favourites values ( ?, ?)", row.fields
153 |end
```