

```

9 # return the login page
10 ▾ get '/login' do
11     erb : "Login/index"
12 end
13
14 # post login form data, evaluate and set session values as required
15 # params;
16 # username
17 # password
18 ▾ post '/login' do
19
20     @username = params[:username]
21     @password = params[:password]
22
23     success = User.authenticate(@username, @password)
24
25     if success then
26         user = User.getByUsername(@username)
27
28         session[:loggedIn] = true
29         session[:username] = user.username
30         session[:userId] = user.userId
31         session[:admin] = user.admin
32         redirect '/bookmarks/all'
33     end
34
35     @loginError = "Username or password isn't correct."
36
37     erb : "Login/index"
38 end
39
40 # logout the currently logged in user
41 ▾ get '/login/logout' do
42     session.clear
43     redirect '/login'
44 end

```

Controllers Login.rb

```

87 ▾ <div class="col quart side-nav">
88     <h5>
89         Information
90     </h5>
91     <p>Author <span class="right-align"><%= @user.username %></span></p>
92     <p>Date <span class="right-align"><%= @bookmark.createdAt %></span></p>
93
94     <br>
95     <ul>
96         <li><a class="btn favourite">Favourite</a></li>
97         <li><a href="/bookmarks/report/<%= @bookmark.bookmarkId %>" class="btn report">Report</a></li>
98         <li><a class="btn edit">Edit</a></li>
99     </ul>
100 </div>
101 </div>
102
103 <%= erb : "Shared/footer" %>
104
105 <script>
106     var elem = document.getElementById('toggle-dropdown');
107
108     elem.onclick = function () {
109
110         var dropElem = document.querySelector('.new-tags-container');
111
112         dropElem.classList.toggle('active');
113     };
114 </script>

```

Views/Bookmarks/view.erb

```

127 ▾ form input[type="text"], input[type="password"], input[type="url"] {
128     font-size: 14px;
129     padding: 5px 0;
130     background: transparent;
131     border: none;
132     border-bottom: solid black 1px;
133 }
134
135 ▾ form input[type="submit"] {
136     font-family: 'Sen';
137     font-size: 16px;
138     padding: 7.5px 15px;
139     color: white;
140     background-color: black;
141     border: black 1px solid;
142     cursor: pointer;
143 }
144 ▾ form input[type="submit"]:hover {
145     transition: 0.2s;
146     background-color: white;
147     color: black;
148 }

```

Public/css/style.css

```

8 ▾ class Bookmark
9
10     DB = SQLite3::Database.new 'database.db'
11
12 ▸ def initialize(bookmarkId, createdAt, title, description, resource, archived, userId)↵
29     end
30
31 ▾ def self.newBookmark(title, description, resource, archived, userId)
32
33     query = "INSERT INTO bookmarks('title', 'description', 'resource', 'archived', 'userId', 'createdAt') VALUES(?, ?, ?, ?, ?, ?);"
34
35     begin
36         DB.execute query, title, description, resource, archived, userId, Time.now.inspect
37     rescue SQLite3::Exception
38         return false
39     end
40
41     return true
42 end
43
44 # Return all the known bookmarks in the database as Bookmark objects
45 # Returns: an array of Bookmark objects
46 ▾ def self.getAll
47
48     toReturn = []
49
50     result = DB.execute "SELECT * FROM bookmarks;"
51
52     for bookmark in result do
53
54         bookmarkObj = Bookmark.new(bookmark[0], bookmark[1], bookmark[2], bookmark[3], bookmark[4], bookmark[5], bookmark[6])
55         toReturn.push(bookmarkObj)
56
57     end
58
59     return toReturn
60 end
61
62 # Return all the known bookmark in the database as Bookmark objects
63 # Returns: an array of Bookmark objects
64 ▾ def self.getByTitle
65     toReturn = []
66
67     result = DB.execute "SELECT title, createdAt FROM bookmarks ORDER BY createdAt ASC;"
68
69     for bookmark in result do
70
71         bookmarkObj = Bookmark.new(nil, bookmark[1], bookmark[2], nil, nil, nil, nil)
72         toReturn.push(bookmarkObj)
73
74     end
75     return toReturn
76 end
77
78 end
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105 ▾
106
107
108
109
110
111
112
113
114
115
116
117 ▾
118
119
120
121
122 ▾
123
124
125
126
127
128
129

```

## Models Bookmark.rb

```
7 class Tag
8
9   DB = SQLite3::Database.new 'database.db'
10
11 def initialize(tagId, tag)↔
12
13 def self.getByBookmarkId(bookmarkId)
14
15   toReturn = []
16
17   query = "SELECT
18           bookmarks_to_tags.bookmarkId,
19           bookmarks_to_tags.tagId,
20           tags.tag
21   FROM
22     bookmarks_to_tags
23   INNER JOIN tags ON bookmarks_to_tags.tagId = tags.tagId
24   WHERE
25     bookmarks_to_tags.bookmarkId = ?;"
26
27   result = DB.execute query, bookmarkId
28
29   for tag in result do
30
31     newTag = Tag.new(tag[1], tag[2])
32     toReturn.push(newTag)
33
34   end
35
36   return toReturn
37 end
38
39 def self.newTag(tag, bookmarkId)
40
41   tagId = Tag.doesTagExist(tag)
42
43   if tagId != nil then
44
45     if Tag.doesLinkExist(bookmarkId, tagId) == false then
46
47       Tag.createLink(bookmarkId, tagId)
48
49     else
50
51       return false
52
53     end
54
55   else
56
57     Tag.createTag(tag)
58     tagId = doesTagExist(tag)
59     Tag.createLink(bookmarkId, tagId)
60
61   end
62
63   return true
64 end
65
66 end
```

## Models Tag.rb