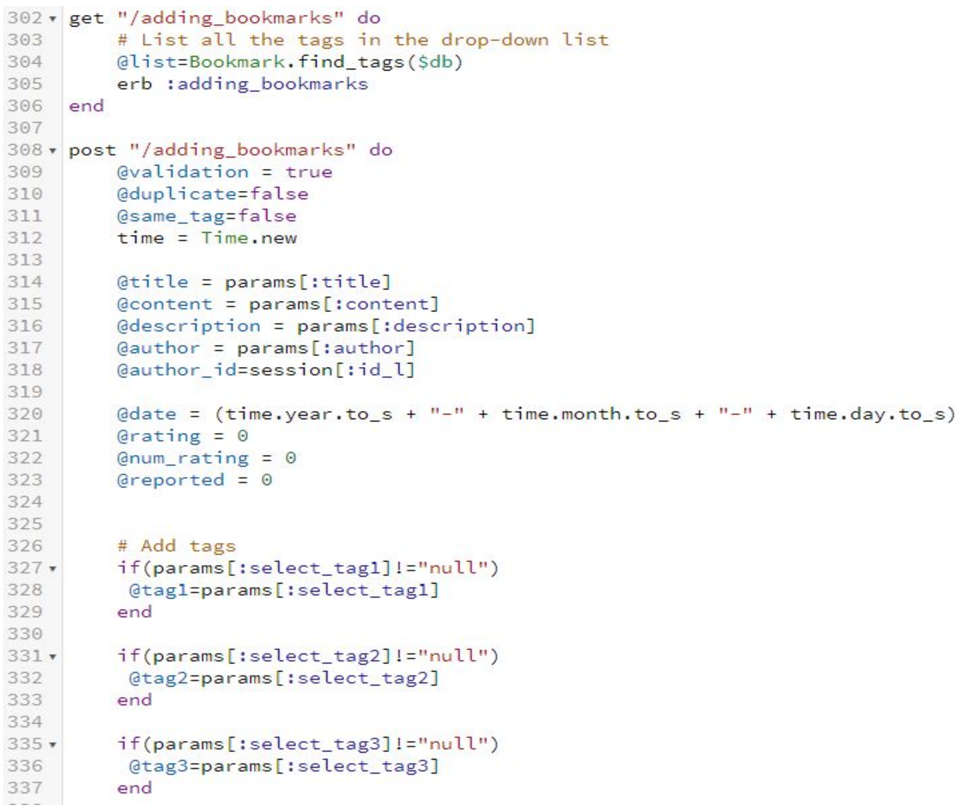
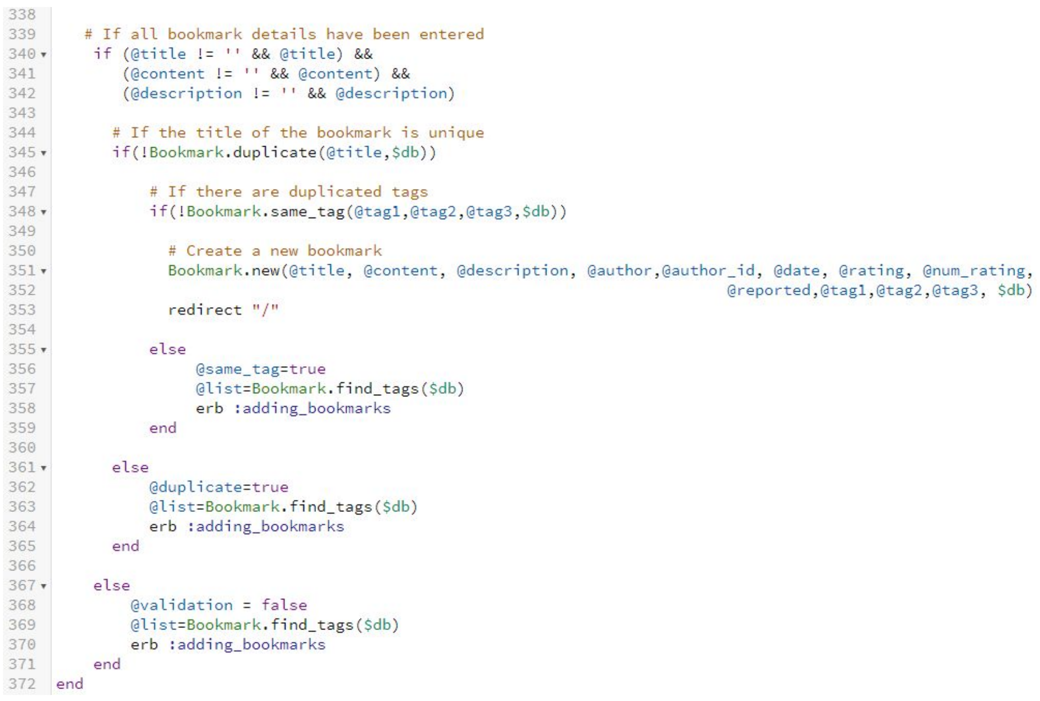
Exemplar Code:



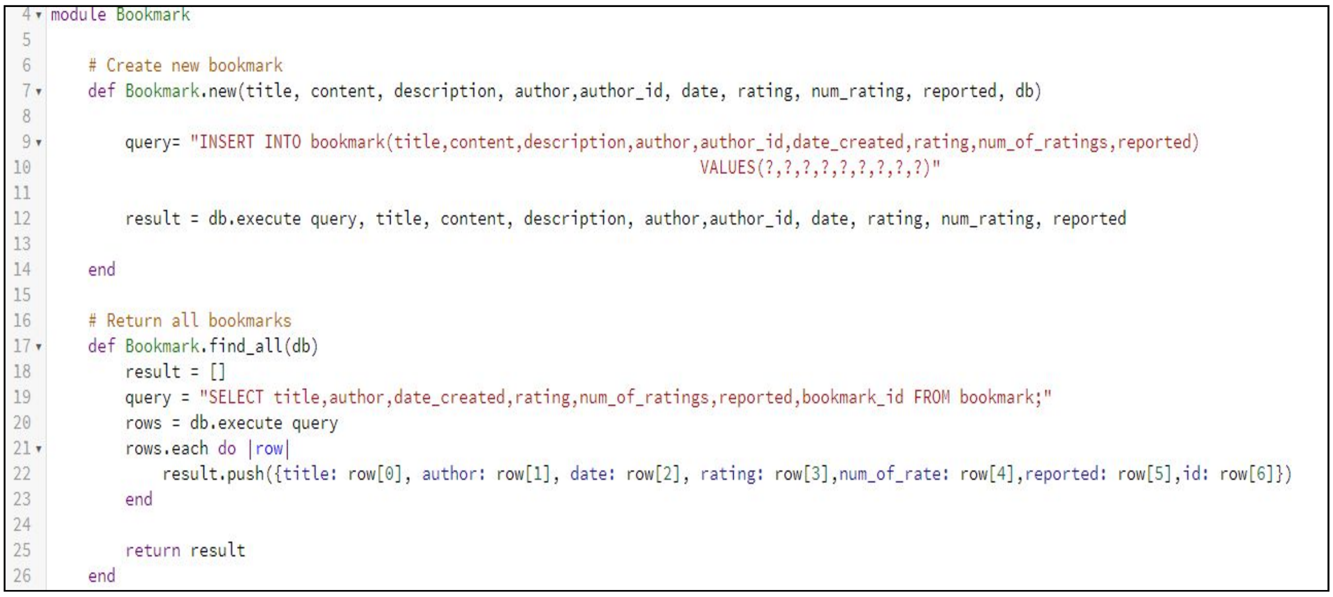
```
302 get "/adding_bookmarks" do
303     # List all the tags in the drop-down list
304     @list=Bookmark.find_tags($db)
305     erb :adding_bookmarks
306 end
307
308 post "/adding_bookmarks" do
309     @validation = true
310     @duplicate=false
311     @same_tag=false
312     time = Time.new
313
314     @title = params[:title]
315     @content = params[:content]
316     @description = params[:description]
317     @author = params[:author]
318     @author_id=session[:id_l]
319
320     @date = (time.year.to_s + "-" + time.month.to_s + "-" + time.day.to_s)
321     @rating = 0
322     @num_rating = 0
323     @reported = 0
324
325
326     # Add tags
327     if(params[:select_tag1]!="null")
328      @tag1=params[:select_tag1]
329     end
330
331     if(params[:select_tag2]!="null")
332      @tag2=params[:select_tag2]
333     end
334
335     if(params[:select_tag3]!="null")
336      @tag3=params[:select_tag3]
337     end
```



```
338
339    # If all bookmark details have been entered
340     if (@title != '' && @title) &&
341        (@content != '' && @content) &&
342        (@description != '' && @description)
343
344       # If the title of the bookmark is unique
345       if(!Bookmark.duplicate(@title,$db))
346
347           # If there are duplicated tags
348           if(!Bookmark.same_tag(@tag1,@tag2,@tag3,$db))
349
350             # Create a new bookmark
351             Bookmark.new(@title, @content, @description, @author,@author_id, @date, @rating, @num_rating,
352                                                                        @reported,@tag1,@tag2,@tag3, $db)
353             redirect "/"
354
355           else
356                @same_tag=true
357                @list=Bookmark.find_tags($db)
358                erb :adding_bookmarks
359           end
360
361       else
362           @duplicate=true
363           @list=Bookmark.find_tags($db)
364           erb :adding_bookmarks
365       end
366
367     else
368         @validation = false
369         @list=Bookmark.find_tags($db)
370         erb :adding_bookmarks
371     end
372 end
```

Listing A: app.rb

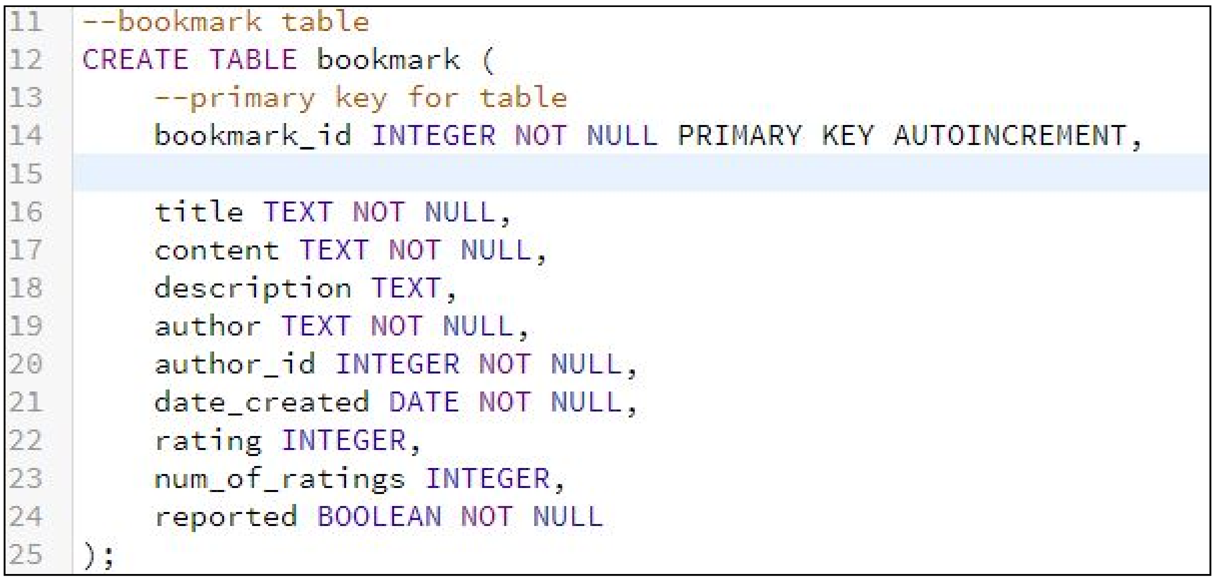
Code showing addition of bookmarks, including validations checks from line 339, if all the information has been obtained correctly.



```ruby
4  module Bookmark
5
6      # Create new bookmark
7      def Bookmark.new(title, content, description, author,author_id, date, rating, num_rating, reported, db)
8
9          query= "INSERT INTO bookmark(title,content,description,author,author_id,date_created,rating,num_of_ratings,reported)
10                                                                 VALUES(?,?,?,?,?,?,?,?,?)"
11
12         result = db.execute query, title, content, description, author,author_id, date, rating, num_rating, reported
13
14     end
15
16     # Return all bookmarks
17     def Bookmark.find_all(db)
18         result = []
19         query = "SELECT title,author,date_created,rating,num_of_ratings,reported,bookmark_id FROM bookmark;"
20         rows = db.execute query
21         rows.each do |row|
22             result.push({title: row[0], author: row[1], date: row[2], rating: row[3],num_of_rate: row[4],reported: row[5],id: row[6]})
23         end
24
25         return result
26     end
```

Listing B: models/bookmark.rb

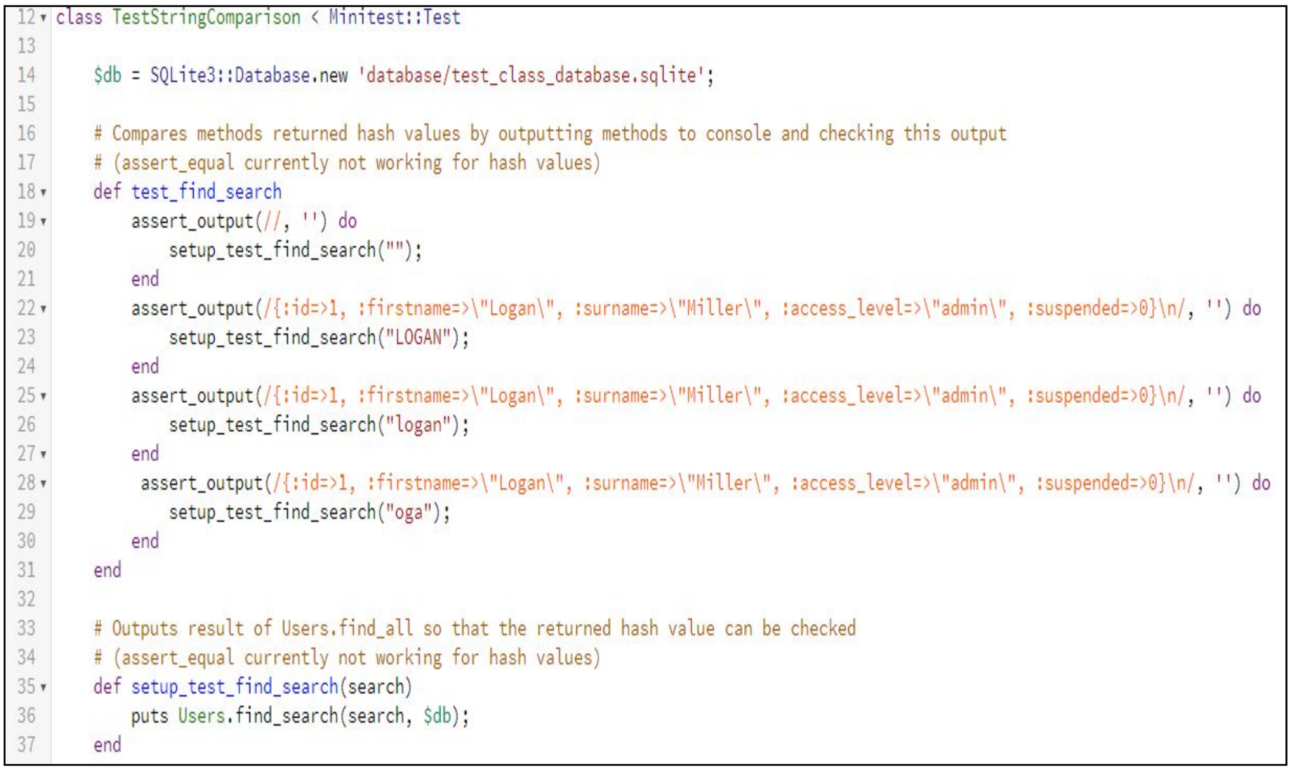
Lines 7-14 show the process of creating a bookmark, with values being inserted to the database. Lines 17-25 shows a function that loops through the database and returns all the bookmarks available



```sql
11 --bookmark table
12 CREATE TABLE bookmark (
13     --primary key for table
14     bookmark_id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
15
16     title TEXT NOT NULL,
17     content TEXT NOT NULL,
18     description TEXT,
19     author TEXT NOT NULL,
20     author_id INTEGER NOT NULL,
21     date_created DATE NOT NULL,
22     rating INTEGER,
23     num_of_ratings INTEGER,
24     reported BOOLEAN NOT NULL
25 );
```

Listing C: create_databases.sql

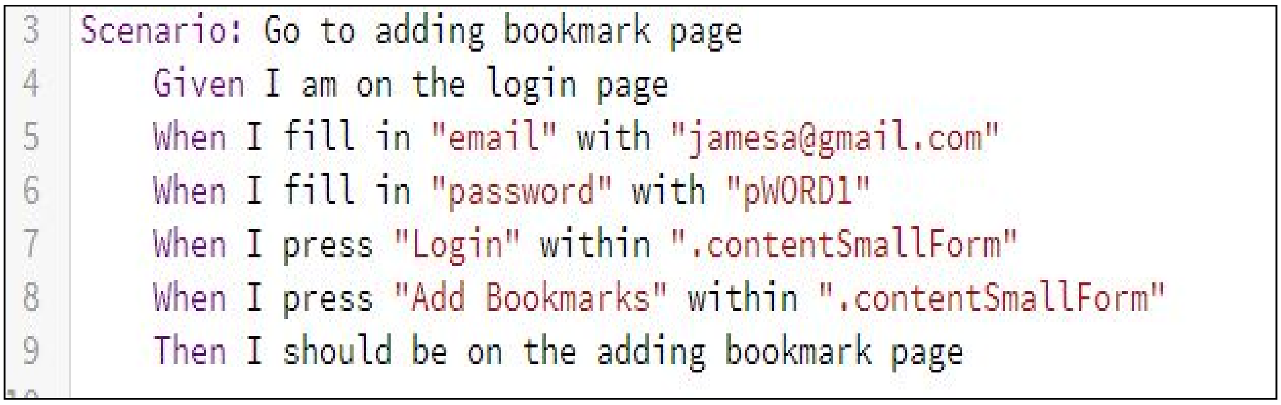
Code from creating a database, with this example being the bookmark table, including some key features such as the title and the primary key of an id on line 14.



```ruby
class TestStringComparison < Minitest::Test

    $db = SQLite3::Database.new 'database/test_class_database.sqlite';

    # Compares methods returned hash values by outputting methods to console and checking this output
    # (assert_equal currently not working for hash values)
    def test_find_search
        assert_output(//, '') do
            setup_test_find_search("");
        end
        assert_output(/{:id=>1, :firstname=>\"Logan\", :surname=>\"Miller\", :access_level=>\"admin\", :suspended=>0}\n/, '') do
            setup_test_find_search("LOGAN");
        end
        assert_output(/{:id=>1, :firstname=>\"Logan\", :surname=>\"Miller\", :access_level=>\"admin\", :suspended=>0}\n/, '') do
            setup_test_find_search("logan");
        end
         assert_output(/{:id=>1, :firstname=>\"Logan\", :surname=>\"Miller\", :access_level=>\"admin\", :suspended=>0}\n/, '') do
            setup_test_find_search("oga");
        end
    end

    # Outputs result of Users.find_all so that the returned hash value can be checked
    # (assert_equal currently not working for hash values)
    def setup_test_find_search(search)
        puts Users.find_search(search, $db);
    end
```

Listing D: user_test_class.rb

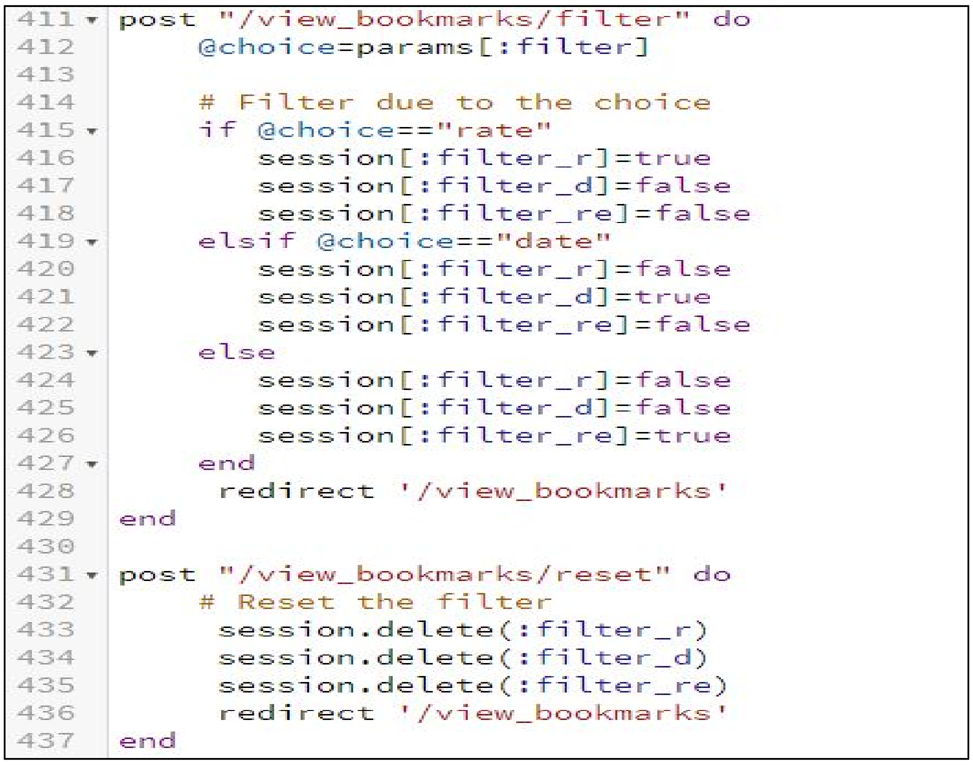
Above shows the minitests we did to make sure features such as searching for users, was done correctly.



```gherkin
Scenario: Go to adding bookmark page
    Given I am on the login page
    When I fill in "email" with "jamesa@gmail.com"
    When I fill in "password" with "pWORD1"
    When I press "Login" within ".contentSmallForm"
    When I press "Add Bookmarks" within ".contentSmallForm"
    Then I should be on the adding bookmark page
```

Listing E: features/add_bookmark.feature

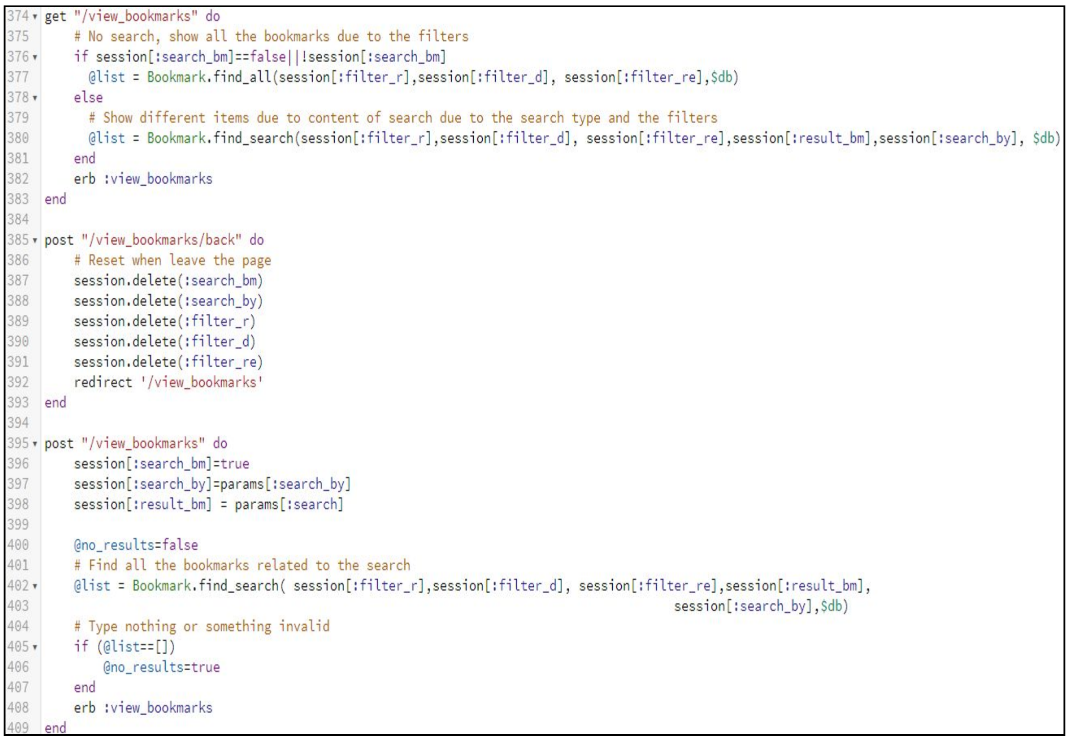
Above shows systematic tests that we implemented to make sure our program works as intended, given certain inputs.



```ruby
post "/view_bookmarks/filter" do
    @choice=params[:filter]

    # Filter due to the choice
    if @choice=="rate"
       session[:filter_r]=true
       session[:filter_d]=false
       session[:filter_re]=false
    elsif @choice=="date"
       session[:filter_r]=false
       session[:filter_d]=true
       session[:filter_re]=false
    else
       session[:filter_r]=false
       session[:filter_d]=false
       session[:filter_re]=true
    end
     redirect '/view_bookmarks'
end

post "/view_bookmarks/reset" do
    # Reset the filter
     session.delete(:filter_r)
     session.delete(:filter_d)
     session.delete(:filter_re)
     redirect '/view_bookmarks'
end
```

Listing F: app.rb

Code above, shows the filtering of bookmarks, with the use of sessions.



```ruby
get "/view_bookmarks" do
    # No search, show all the bookmarks due to the filters
    if session[:search_bm]==false||!session[:search_bm]
      @list = Bookmark.find_all(session[:filter_r],session[:filter_d], session[:filter_re],$db)
    else
      # Show different items due to content of search due to the search type and the filters
      @list = Bookmark.find_search(session[:filter_r],session[:filter_d], session[:filter_re],session[:result_bm],session[:search_by], $db)
    end
    erb :view_bookmarks
end

post "/view_bookmarks/back" do
    # Reset when leave the page
    session.delete(:search_bm)
    session.delete(:search_by)
    session.delete(:filter_r)
    session.delete(:filter_d)
    session.delete(:filter_re)
    redirect '/view_bookmarks'
end

post "/view_bookmarks" do
    session[:search_bm]=true
    session[:search_by]=params[:search_by]
    session[:result_bm] = params[:search]

    @no_results=false
    # Find all the bookmarks related to the search
    @list = Bookmark.find_search( session[:filter_r],session[:filter_d], session[:filter_re],session[:result_bm],
                                                                                       session[:search_by],$db)
    # Type nothing or something invalid
    if (@list==[])
        @no_results=true
    end
    erb :view_bookmarks
end
```

Listing G : app.rb

Above, shows code used to view the bookmarks, accessing them from the database based on the search modules used such as Bookmark.find_search.