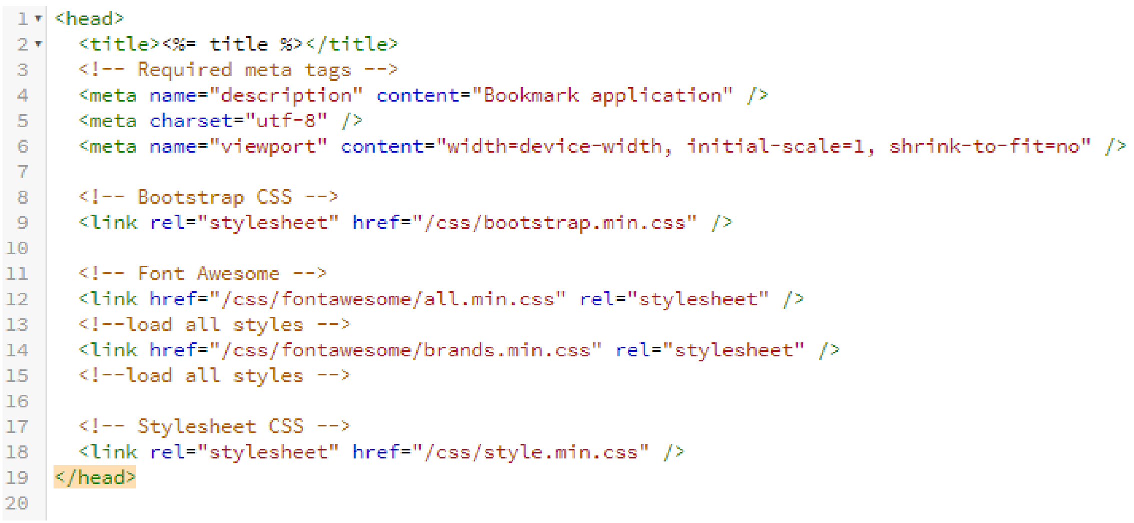
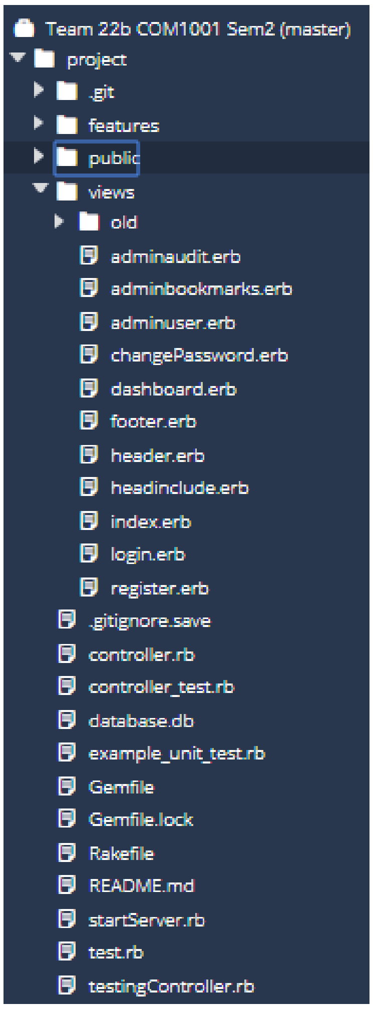
Team 22b – Anna Penny, Nicholas Ruffles, Jake Robison, Stanislaw Malinowski, Lujain Hawsawi, Abdulrahmna AlTerkait

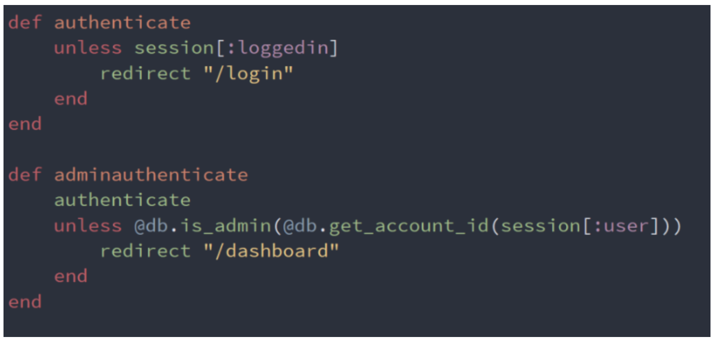


```
<head>
  <title><%= title %></title>
  <!-- Required meta tags -->
  <meta name="description" content="Bookmark application" />
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="/css/bootstrap.min.css" />

  <!-- Font Awesome -->
  <link href="/css/fontawesome/all.min.css" rel="stylesheet" />
  <!--load all styles -->
  <link href="/css/fontawesome/brands.min.css" rel="stylesheet" />
  <!--load all styles -->

  <!-- Stylesheet CSS -->
  <link rel="stylesheet" href="/css/style.min.css" />
</head>
```

Listing A- headinclude.erb

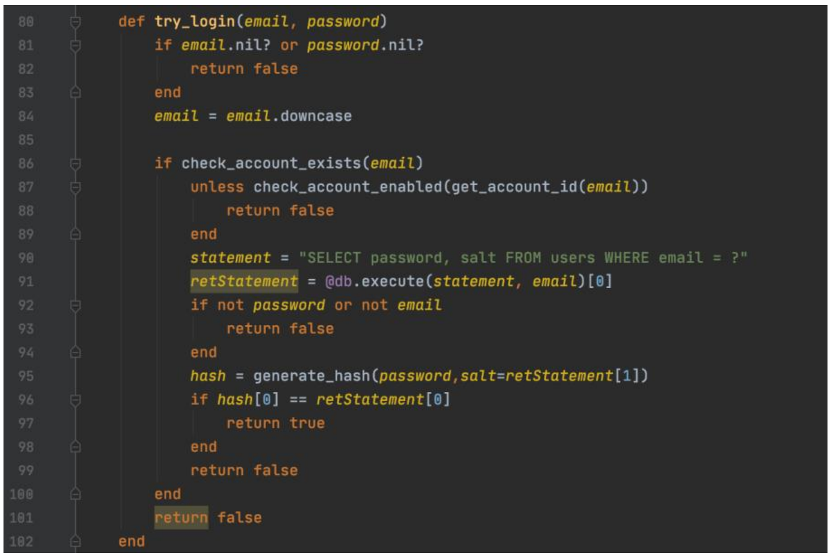


```
Team 22b COM1001 Sem2 (master)
  project
    .git
    features
    public
    views
      old
      adminaudit.erb
      adminbookmarks.erb
      adminuser.erb
      changePassword.erb
      dashboard.erb
      footer.erb
      header.erb
      headinclude.erb
      index.erb
      login.erb
      register.erb
    .gitignore.save
    controller.rb
    controller_test.rb
    database.db
    example_unit_test.rb
    Gemfile
    Gemfile.lock
    Rakefile
    README.md
    startServer.rb
    test.rb
    testingController.rb
```



```
def authenticate
    unless session[:loggedin]
        redirect "/login"
    end
end

def adminauthenticate
    authenticate
    unless @db.is_admin(@db.get_account_id(session[:user]))
        redirect "/dashboard"
    end
end
```

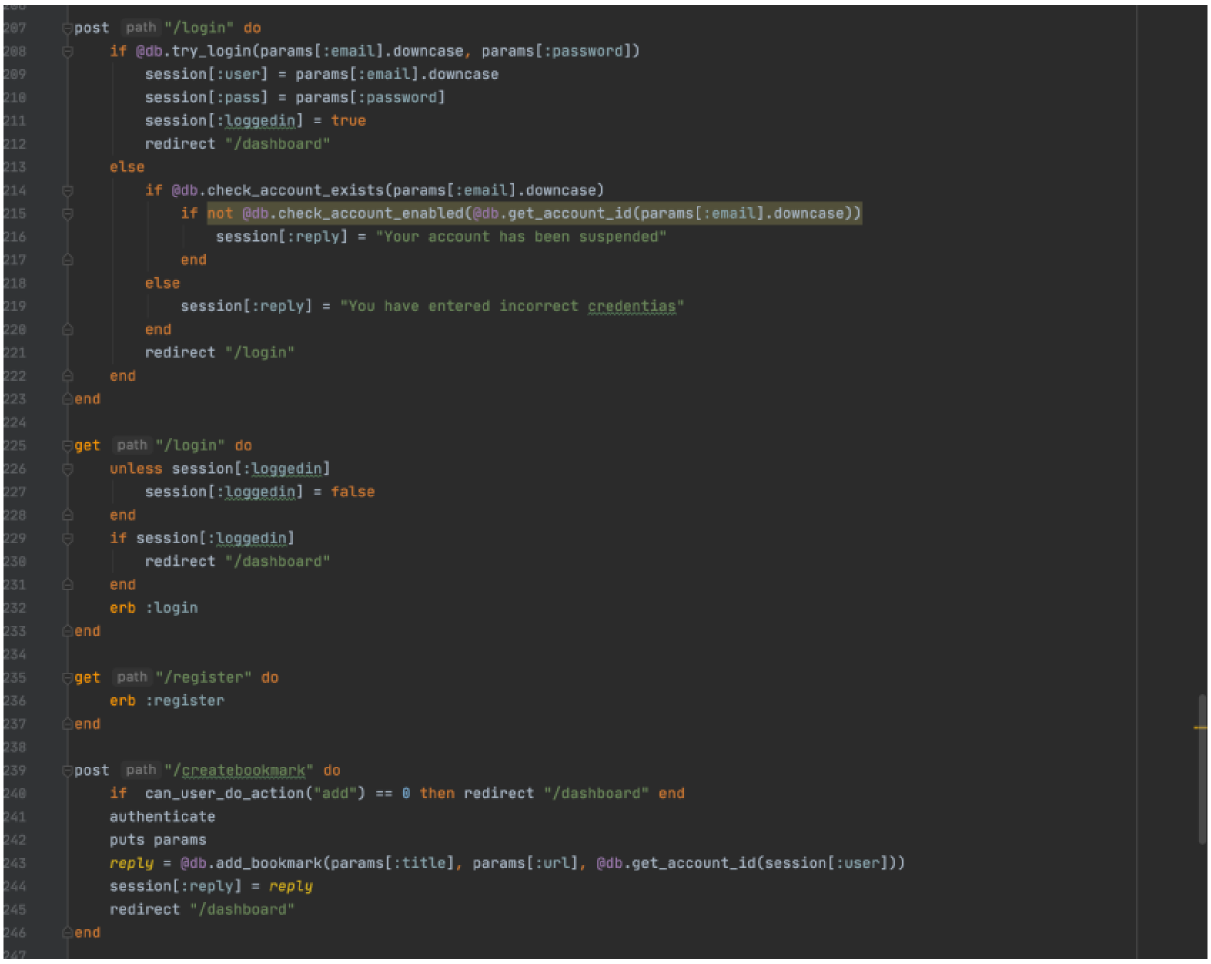
Listing C – startServer.rb



```
    def try_login(email, password)
        if email.nil? or password.nil?
            return false
        end
        email = email.downcase

        if check_account_exists(email)
            unless check_account_enabled(get_account_id(email))
                return false
            end
            statement = "SELECT password, salt FROM users WHERE email = ?"
            retStatement = @db.execute(statement, email)[0]
            if not password or not email
                return false
            end
            hash = generate_hash(password,salt=retStatement[1])
            if hash[0] == retStatement[0]
                return true
            end
            return false
        end
        return false
    end
```

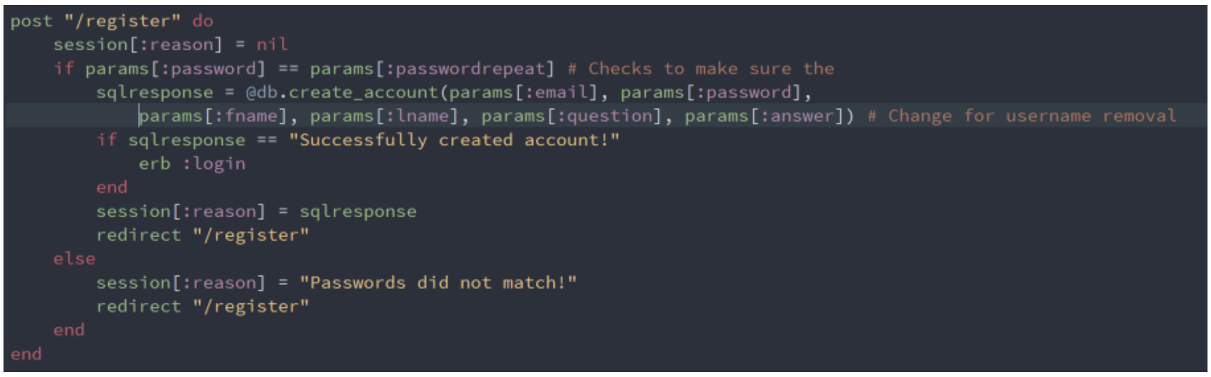
Listing D – controller.rb

Listing B – file structure



```ruby
post  path "/login" do
    if @db.try_login(params[:email].downcase, params[:password])
        session[:user] = params[:email].downcase
        session[:pass] = params[:password]
        session[:loggedin] = true
        redirect "/dashboard"
    else
        if @db.check_account_exists(params[:email].downcase)
            if not @db.check_account_enabled(@db.get_account_id(params[:email].downcase))
                session[:reply] = "Your account has been suspended"
            end
        else
            session[:reply] = "You have entered incorrect credentias"
        end
        redirect "/login"
    end
end

get  path "/login" do
    unless session[:loggedin]
        session[:loggedin] = false
    end
    if session[:loggedin]
        redirect "/dashboard"
    end
    erb :login
end

get  path "/register" do
    erb :register
end

post  path "/createbookmark" do
    if  can_user_do_action("add") == 0 then redirect "/dashboard" end
    authenticate
    puts params
    reply = @db.add_bookmark(params[:title], params[:url], @db.get_account_id(session[:user]))
    session[:reply] = reply
    redirect "/dashboard"
end
```

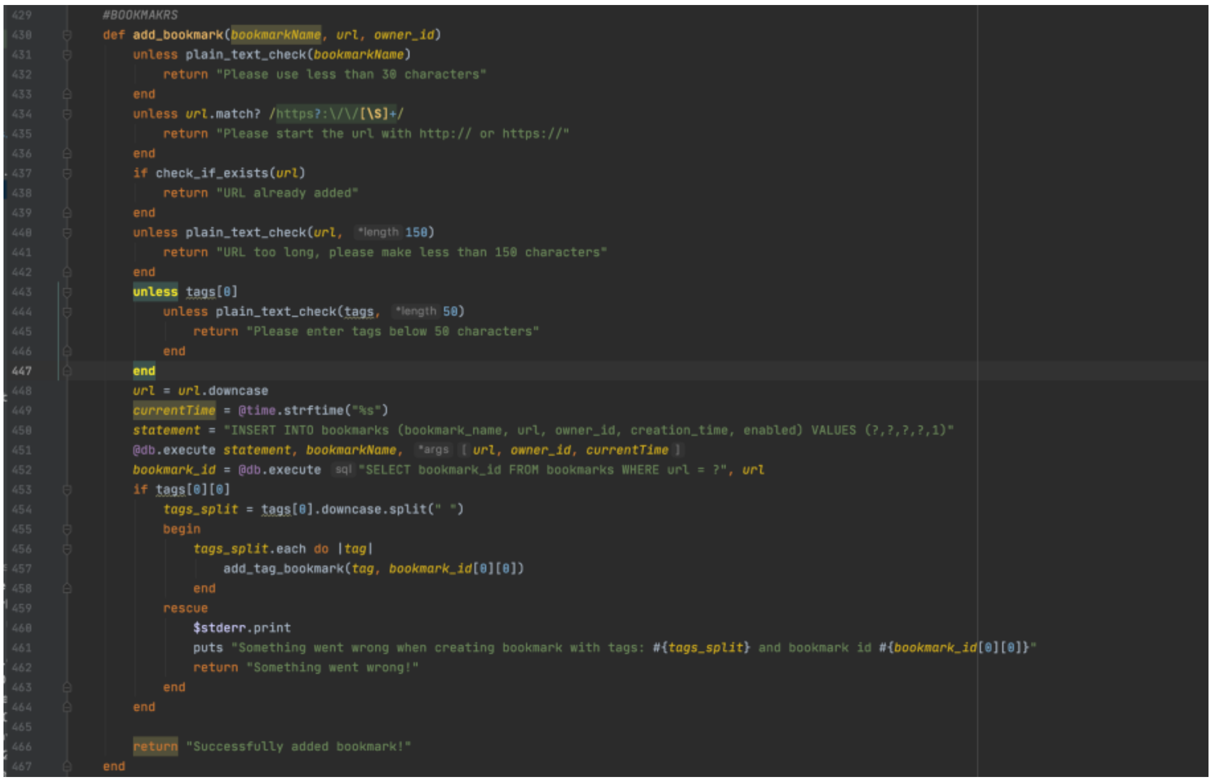
Listing E – startServer.rb



```ruby
post "/register" do
    session[:reason] = nil
    if params[:password] == params[:passwordrepeat] # Checks to make sure the
        sqlresponse = @db.create_account(params[:email], params[:password],
            params[:fname], params[:lname], params[:question], params[:answer]) # Change for username removal
        if sqlresponse == "Successfully created account!"
            erb :login
        end
        session[:reason] = sqlresponse
        redirect "/register"
    else
        session[:reason] = "Passwords did not match!"
        redirect "/register"
    end
end
```

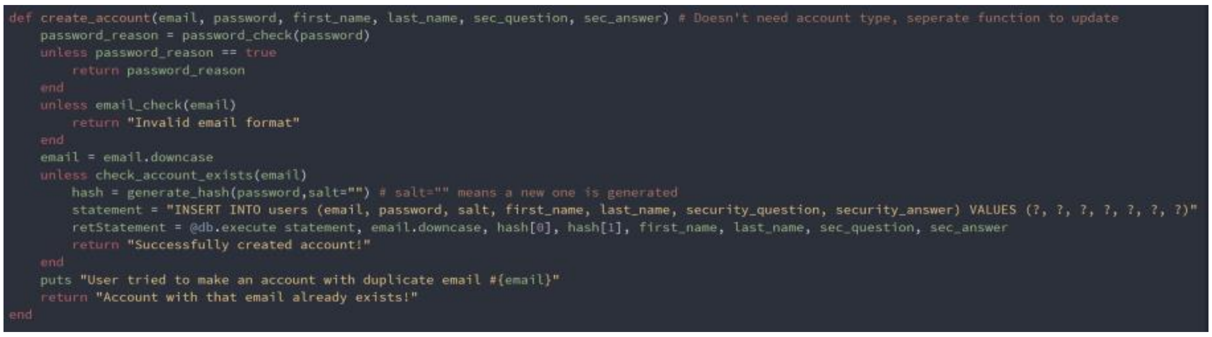
Listing F – startServer.rb

Team 22b – Anna Penny, Nicholas Ruffles, Jake Robison, Stanislaw Malinowski, Lujain Hawsawi, Abdulrahmna AlTerkait



```ruby
    #BOOKMAKRS
    def add_bookmark(bookmarkName, url, owner_id)
        unless plain_text_check(bookmarkName)
            return "Please use less than 30 characters"
        end
        unless url.match? /https?:\/\/[\S]+/
            return "Please start the url with http:// or https://"
        end
        if check_if_exists(url)
            return "URL already added"
        end
        unless plain_text_check(url, *length 150)
            return "URL too long, please make less than 150 characters"
        end
        unless tags[0]
            unless plain_text_check(tags, *length 50)
                return "Please enter tags below 50 characters"
            end
        end
        url = url.downcase
        currentTime = @time.strftime("%s")
        statement = "INSERT INTO bookmarks (bookmark_name, url, owner_id, creation_time, enabled) VALUES (?,?,?,?,1)"
        @db.execute statement, bookmarkName, *args [ url, owner_id, currentTime ]
        bookmark_id = @db.execute sql "SELECT bookmark_id FROM bookmarks WHERE url = ?", url
        if tags[0][0]
            tags_split = tags[0].downcase.split(" ")
            begin
                tags_split.each do |tag|
                    add_tag_bookmark(tag, bookmark_id[0][0])
                end
            rescue
                $stderr.print
                puts "Something went wrong when creating bookmark with tags: #{tags_split} and bookmark id #{bookmark_id[0][0]}"
                return "Something went wrong!"
            end
        end

        return "Successfully added bookmark!"
    end
```

Listing G – controller.rb



```ruby
def create_account(email, password, first_name, last_name, sec_question, sec_answer) # Doesn't need account type, seperate function to update
    password_reason = password_check(password)
    unless password_reason == true
        return password_reason
    end
    unless email_check(email)
        return "Invalid email format"
    end
    email = email.downcase
    unless check_account_exists(email)
        hash = generate_hash(password,salt="") # salt="" means a new one is generated
        statement = "INSERT INTO users (email, password, salt, first_name, last_name, security_question, security_answer) VALUES (?, ?, ?, ?, ?, ?, ?)"
        retStatement = @db.execute statement, email.downcase, hash[0], hash[1], first_name, last_name, sec_question, sec_answer
        return "Successfully created account!"
    end
    puts "User tried to make an account with duplicate email #{email}"
    return "Account with that email already exists!"
end
```

Listing H – controller.rb