

Web Development

Produced
by

Eamonn de Leastar (edeleastar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

<http://www.wit.ie>

<http://elearning.wit.ie>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE



Form Design

Web Development with Play

Objectives

- Implement three new views, each associated with the same controller.
- These view provide an 'account management' feature, allowing users to sign up and log in to the spacebook service
- Explore how information is transmitted to an app via a form. Have this information logged to the console

Signup, Login & Logout

- Our application currently does not currently support certain critical features:
 - Signing up to the service
 - Logging in with username/password
 - Logging out
- We might call this feature 'Account Management'. This would support a number of screens:
 - A screen to ask the user if they want to sign up or log in
 - A sign up screen
 - A log in screen
- We will provide all of the above via a new class we will call 'Accounts'.

Accounts

- New Controller
- Supports three views
 - signup.html
 - login.html
 - index.html
- Not “logout” view - if it is requested, we display the ‘index.html’ view

```
public class Accounts extends Controller
{
    public static void signup()
    {
        render();
    }

    public static void login()
    {
        render();
    }

    public static void logout()
    {
        index();
    }

    public static void index()
    {
        render();
    }
}
```

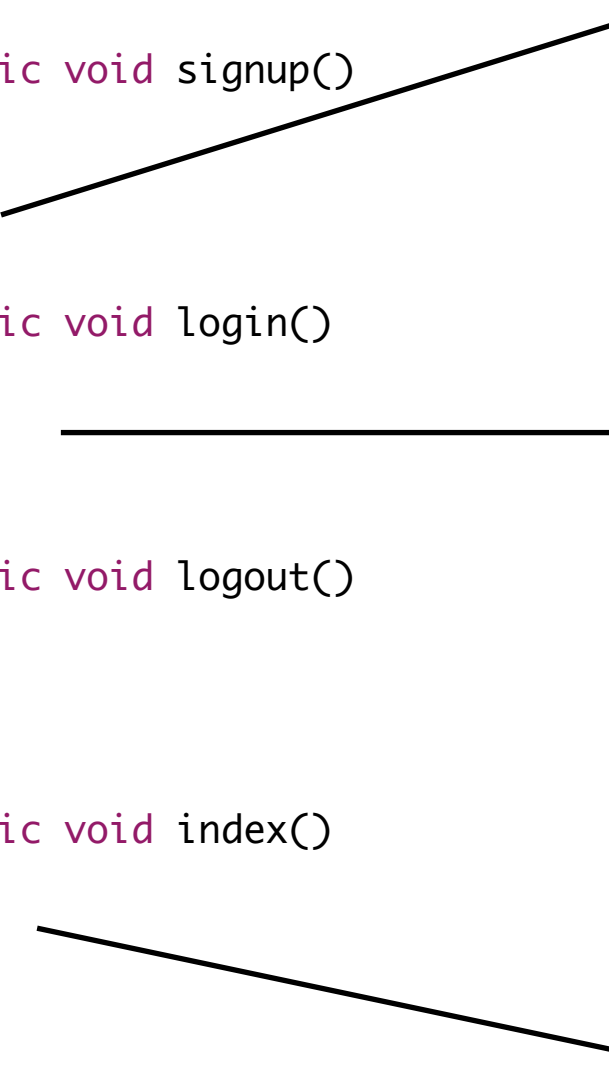
Account Views

```
public class Accounts extends Controller
{
    public static void signup()
    {
        render();
    }

    public static void login()
    {
        render();
    }

    public static void logout()
    {
        index();
    }

    public static void index()
    {
        render();
    }
}
```



```
{% extends 'main.html' %}
{% set title: 'Signup for Spacebook' %}

<nav class="ui menu">
  <a class="active item" href="/signup"> Signup </a>
  <a class="item" href="/login"> Login </a>
</nav>

<p> Signup to the Spacebook Service
```

```
{% extends 'main.html' %}
{% set title: 'Login to Spacebook' %}

<nav class="ui menu">
  <a class="item" href="/signup"> Signup </a>
  <a class="active item" href="/login"> Login </a>
</nav>

<p> Log in to the Spacebook Service
```

```
{% extends 'main.html' %}
{% set title: 'Welcome to Spacebook' %}

<nav class="ui menu">
  <a class="item" href="/signup"> Signup </a>
  <a class="item" href="/login"> Login </a>
</nav>

<p> Signup or Log in to the Spacebook Service
```

Routes File

Landing page

GET	/	Accounts.index
-----	---	----------------

Accounts

GET	/signup	Accounts.signup
-----	---------	-----------------

GET	/login	Accounts.login
-----	--------	----------------

GET	/logout	Accounts.logout
-----	---------	-----------------

Home page

GET	/home	Home.index
-----	-------	------------

Routing of Requests

```
public class Accounts extends Controller
{
    public static void signup()
    {
        render();
    }

    public static void login()
    {
        render();
    }

    public static void logout()
    {
        index();
    }

    public static void index()
    {
        render();
    }
}
```

```
#{extends 'main.html' /}
#{set title:'Signup for Spacebook' /}

<nav class="ui menu">
  <a class="active item" href="/signup">
  <a class="item" href="/login"> Login
</nav>

<p> Signup to the Spacebook Service
```

```
#{extends 'main.html' /}
#{set title:'Login to Spacebook' /}

<nav class="ui menu">
  <a class="item" href="/signup"> Signup
  <a class="active item" href="/login"> L
</nav>

<p> Log in to the Spacebook Service
```

```
#{extends 'main.html' /}
#{set title:'Welcome to Spacebook' /}

<nav class="ui menu">
  <a class="item" href="/signup"> Signup
  <a class="item" href="/login"> Login
</nav>

<p> Signup or Log in to the Spacebook Ser
```

http://localhost:9000/signup

http://localhost:9000/login

http://localhost:9000/logout

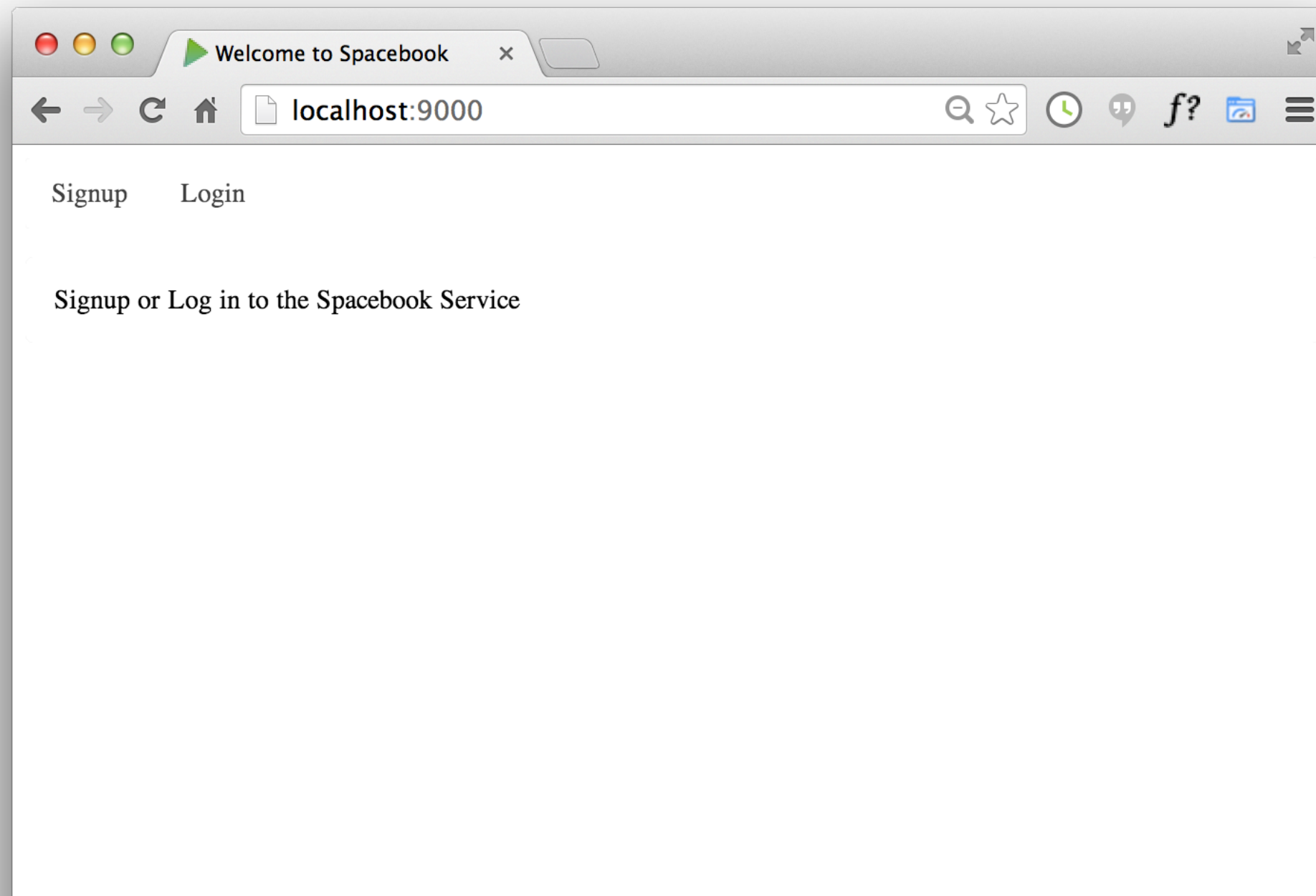
http://localhost:9000/

index.html

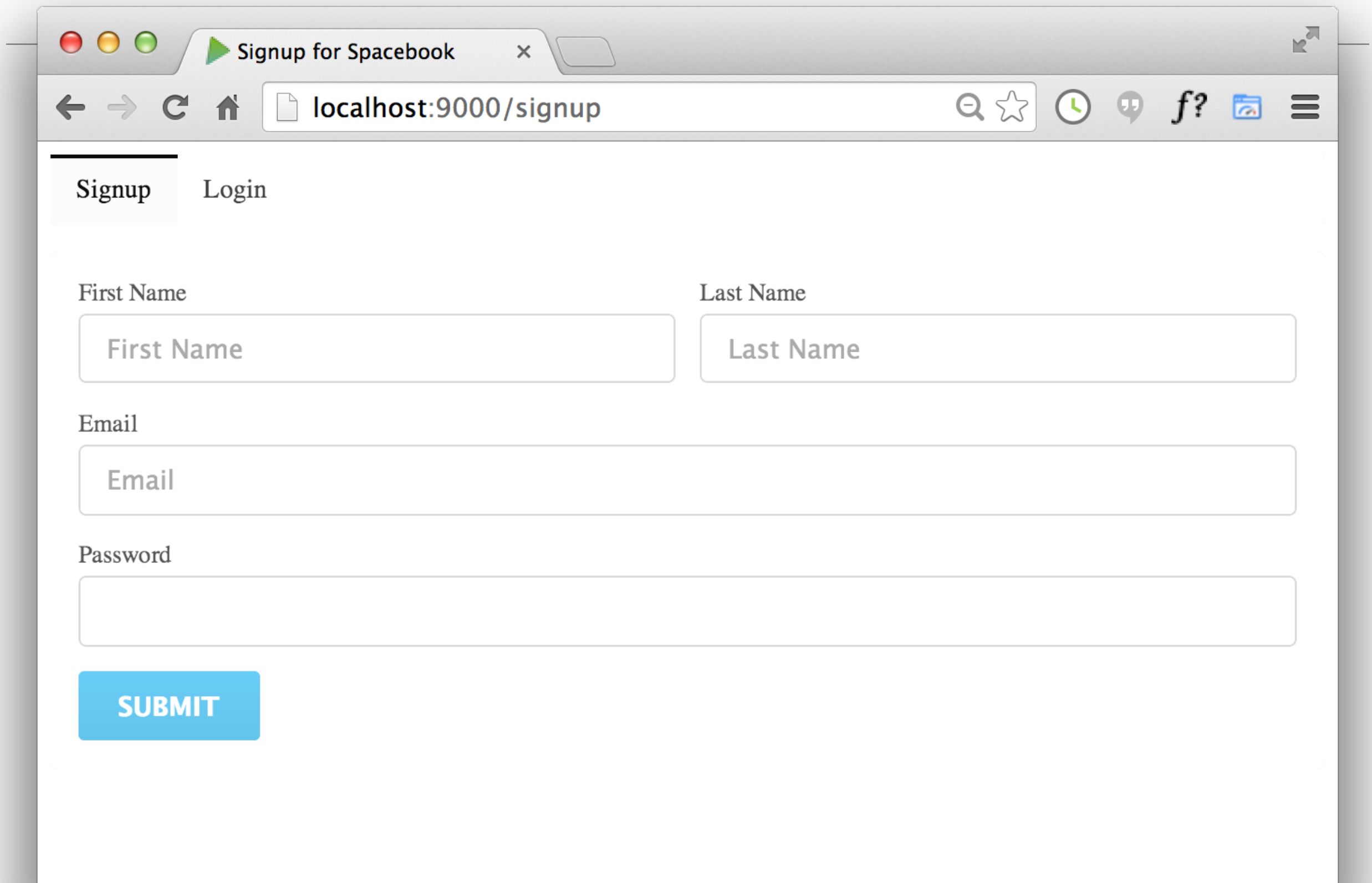
```
{extends 'main.html' /}
{set title:'Welcome to Spacebook' /}

{include 'nav/open.html' /}
  <li> <a href="/signup"> Signup </a></li>
  <li> <a href="/login"> Login </a></li>
{include 'nav/close.html' /}

<p> Signup or Log in to the Spacebook Service </p>
```



Signup View



The image shows a web browser window with a single tab titled "Signup for Spacebook". The address bar displays "localhost:9000/signup". The page features a navigation bar with "Signup" (active) and "Login" links. Below this, the form consists of four input fields: "First Name", "Last Name", "Email", and "Password". A blue "SUBMIT" button is positioned at the bottom left of the form area.

Signup for Spacebook x

localhost:9000/signup

Signup Login

First Name

Last Name

Email

Password

SUBMIT

Signup <form>

```
<form action="/register" method="POST">
  <div class="two fields">
    <div class="field">
      <label>First Name</label>
      <input placeholder="First Name" type="text" name="firstName">
    </div>
    <div class="field">
      <label>Last Name</label>
      <input placeholder="Last Name" type="text" name="lastName">
    </div>
  </div>
  <div class="field">
    <label>Email</label>
    <input placeholder="Email" type="text" name="email">
  </div>
  <div class="field">
    <label>Password</label>
    <input type="password" name="password">
  </div>
  <button class="ui blue submit button">Submit</button>
</form>
```

Signup HTML

- `<form>` tag introduces a form that a user can fill out

```
<form action="/register" method="POST">
```

```
</form>
```

<label>

- <label> : string associated with input fields

```
<form action="/register" method="POST">
```

```
    <label>First Name</label>
```

```
    <label>Last Name</label>
```

```
    <label>Email</label>
```

```
    <label>Password</label>
```

```
</form>
```

<input>

- Allow a user to provide input

```
<form action="/register" method="POST">
```

```
  <input placeholder="First Name" type="text" name="firstName">
```

```
  <input placeholder="Last Name" type="text" name="lastName">
```

```
  <input placeholder="Email" type="text" name="email">
```

```
  <input type="password" name="password">
```

```
</form>
```

<input> type="text"

- Allow a user to provide textual input
- Text us displayed as user types

```
<form action="/register" method="POST">
```

```
<input placeholder="First Name" type="text" name="firstName">
```

```
<input placeholder="Last Name" type="text" name="lastName">
```

```
<input placeholder="Email" type="text" name="email">
```

```
</form>
```

<input> type="password"

- Allow a user to provide textual input
- Display '*' for each character to keep password private

```
<form action="/register" method="POST">
```

```
<input type="password" name="password">
```

```
</form>
```


Text <input>, type="submit"

- Displays a Button
- Value contains string to be displayed on Button surface

```
<form action="/register" method="POST">
```

```
<button class="ui blue submit button">Submit</button>
```

```
</form>
```

Form Action

- When the button is pressed by the user:
- All of the text the user typed is put together into a single “Request”
- This Request is “POST”ed to the web application
- It is posted to the “/register” action

```
<form action="/register" method="POST">
```

```
</form>
```

Register Route

POST /register

Accounts.register

- register() action will be responsible for:
 - Recovering all of the fields “POST”ed by the user
 - Save all these fields in a database
 - Display the start screen again.

```
public class Accounts extends Controller
{
    //...

    public static void index()
    {
        render();
    }

    public static void register()
    {
        index();
    }
}
```

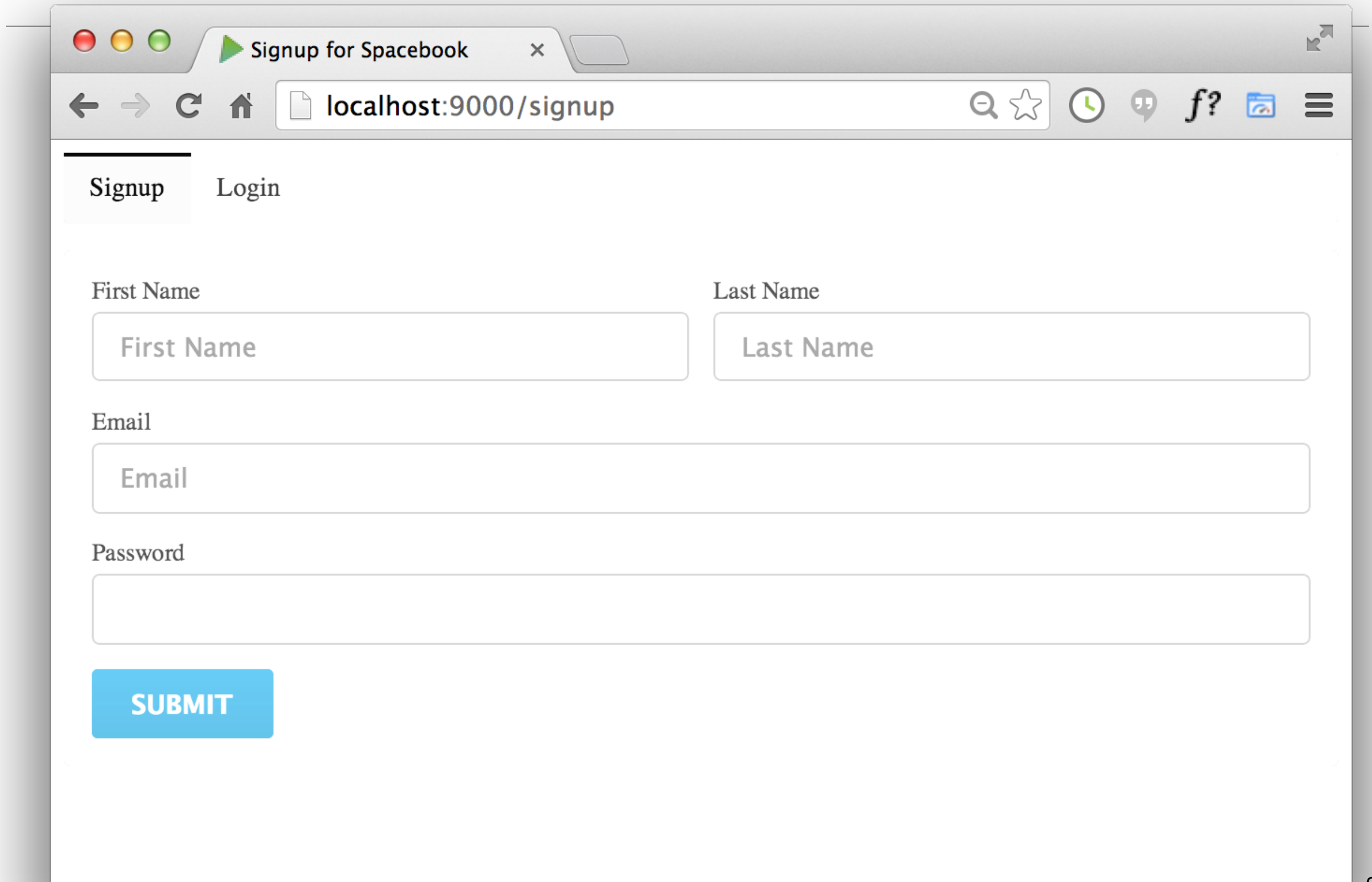
Styling the Signup View

Semantic UI styles..

```
<form action="/register" method="POST">
  <div class="two fields">
    <div class="field">
      <label>First Name</label>
      <input placeholder="First Name" type="text" name="firstName">
    </div>
    <div class="field">
      <label>Last Name</label>
      <input placeholder="Last Name" type="text" name="lastName">
    </div>
  </div>
  <div class="field">
    <label>Email</label>
    <input placeholder="Email" type="text" name="email">
  </div>
  <div class="field">
    <label>Password</label>
    <input type="password" name="password">
  </div>
  <button class="ui blue submit button">Submit</button>
</form>
```

Semantic
UI styles..

views/accounts/signup.html



The image shows a web browser window with a single tab titled "Signup for Spacebook". The address bar displays "localhost:9000/signup". The page has a navigation bar with "Signup" (active) and "Login" links. Below this, there are four input fields: "First Name", "Last Name", "Email", and "Password". Each field has a placeholder text matching its label. At the bottom left, there is a blue "SUBMIT" button.

Signup for Spacebook

localhost:9000/signup

Signup Login

First Name

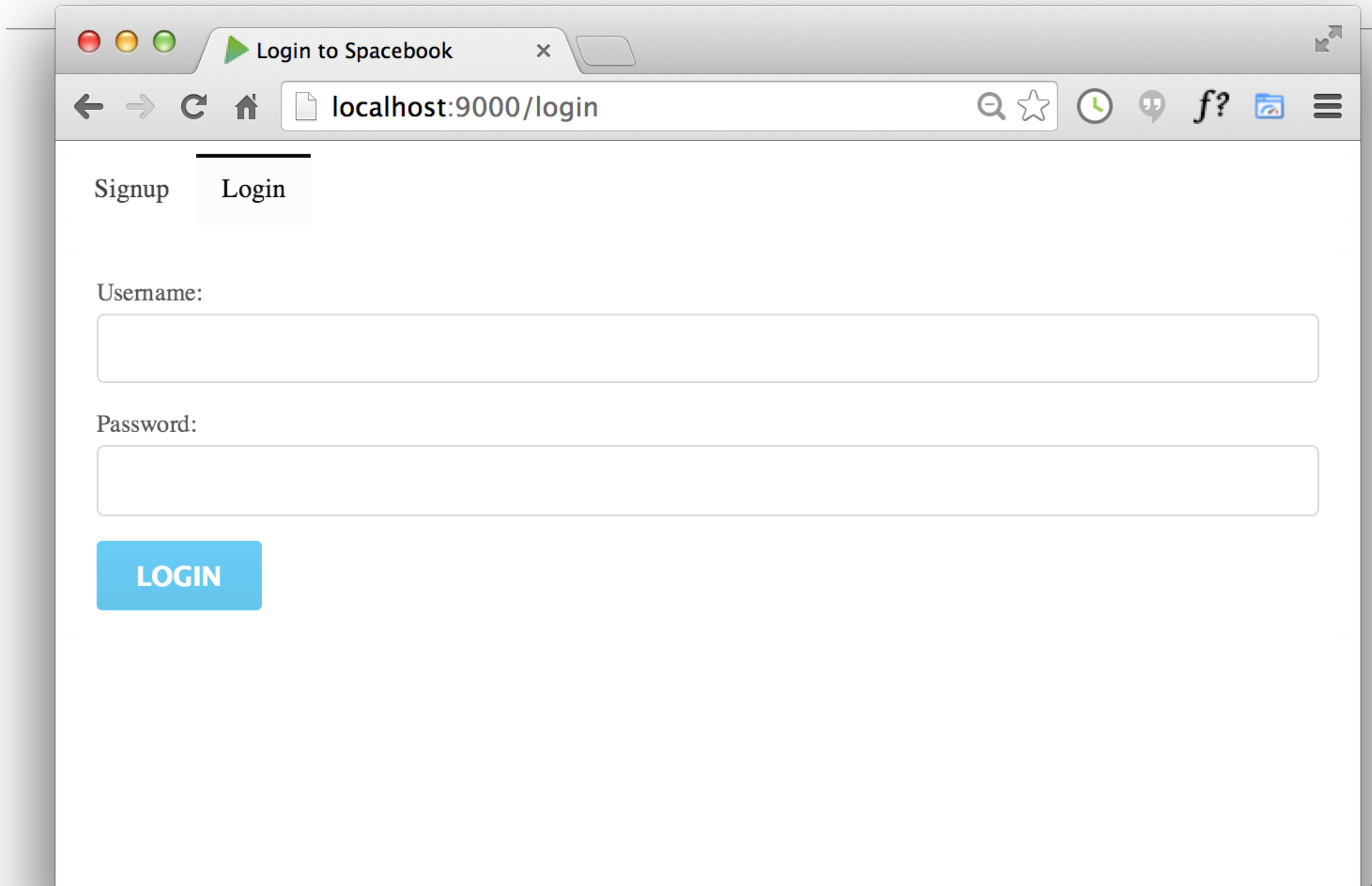
Last Name

Email

Password

SUBMIT

views/accounts/login.html



The screenshot shows a web browser window with a single tab titled "Login to Spacebook". The address bar displays "localhost:9000/login". The page features two tabs: "Signup" and "Login", with "Login" being the active tab. Below the tabs, there are two input fields: "Username:" and "Password:". A blue "LOGIN" button is positioned below the password field. The browser's toolbar includes back, forward, refresh, and home buttons, as well as search, star, clock, and other utility icons.

Signup Login

Username:

Password:

LOGIN

```
<form action="/authenticate" method="POST">
  <div class="field">
    <label> Username: </label>
    <input type="text" name="email">
  </div>
  <div class="field">
    <label> Password: </label>
    <input type="password" name="password">
  </div>
  <button class="ui blue submit button">Login</button>
</form>
```

login form - action implementation

POST /authenticate	Accounts.authenticate
-----------------------	-----------------------

- Authenticate should:
 - recover username + password POSTed by the user
 - Check to see if these are valid for some user in the system
 - Redirect to the user home page if valid
 - if not, redirect to start page (index)

```
public class Accounts extends Controller
{
    //...

    public static void index()
    {
        render();
    }

    public static void authenticate()
    {
        Home.index();
    }
}
```



```

# Routes
# This file defines all application routes (Higher priority routes first)
# ~~~~

# Landing page
GET    /                               Accounts.index

# Accounts

GET    /signup                       Accounts.signup
GET    /login                      Accounts.login
GET    /logout                    Accounts.logout
POST   /authenticate                 Accounts.authenticate
POST   /register                    Accounts.register

# Home page
GET    /home                      Home.index

# Home page
GET    /home                    Home.index
GET    /home/drop/{name}        Home.drop

# Members page
GET    /members                 Members.index
GET    /members/follow/{name}   Members.follow

# Profile page
GET    /profile                 Profile.index

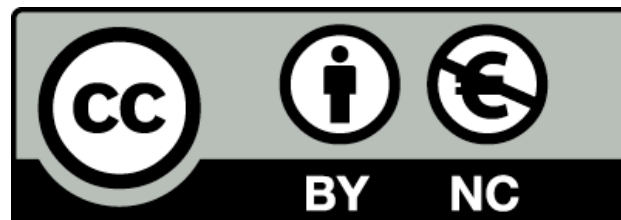
# Public Profiles
GET    /publicprofile/{name}     PublicProfile.visit

# Ignore favicon requests
GET    /favicon.ico             404

# Map static resources from the /app/public folder to the /public path
GET    /public/                 staticDir:public

# Catch all
*      /{controller}/{action}    {controller}.{action}

```



Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

