PRADITA University

IF140303-Web Application Development

Session-07: Introduction to the Phoenix Framework

PRU/SPMI/FR-BM-18/0222

Alfa Yohannis

Introduction to the Phoenix Framework

- Phoenix is a web development framework for building scalable and maintainable applications in Elixir.
- It is known for its performance, fault tolerance, and real-time capabilities.
- Phoenix requires installation of Phoenix itself, Node.js for asset management, and PostgreSQL for database management.

Why Install Node.js and PostgreSQL?

- **Node.js**: Handles assets like JavaScript and CSS, making it essential for the front-end part of the Phoenix application.
- **PostgreSQL**: A robust, open-source relational database system that Phoenix uses via Ecto for managing data persistence.

Phoenix Workflow Overview

- **Incoming Request**: Phoenix receives a request from the client.
- **Ensure HTML Request**: Phoenix checks if the request is for an HTML response.
- **Session Check**: Verifies if the request has an active session.
- **Security Check**: Performs necessary security validations.
- **HTTP Headers**: Adds appropriate headers for browser compatibility.
- **Access Request**: Determines what resource the request is trying to access.
- **Formulate Response**: Phoenix formulates the response and sends it back to the client.

Generating a Phoenix Project

- To create a new Phoenix project, use the command:

```
mix phx.new <name>
```

- Navigate to the project directory and create the database:

```
mix ecto.create
```

- Configure the database by editing `dev.exs`:

Database Configuration in `dev.exs`

```
config :reddit, Reddit.Repo,
adapter: Ecto.Adapters.Postgres,
username: "postgres",
password: "postgres",
database: "reddit_dev",
hostname: "localhost",
pool_size: 10
```

- Adjust the `username` and `password` according to your PostgreSQL setup.
- Start the Phoenix server with:

```
mix phx.server
```

Project Overview: Reddit-like Discussion Forum

- The project is a discussion forum similar to Reddit.
- Users can sign in with GitHub, view posts, create discussions, and comment.
- Users can also edit or delete their posts.

Server-Side Templating vs Single Page Apps (SPA)

- **Server-Side Templating**: The server generates HTML dynamically and sends it to the client.
- **Single Page Apps**: The client dynamically updates content without reloading the page, often using frameworks like React or Angular.
- Phoenix supports both approaches, allowing flexibility in web application design.

Phoenix Layout and MaterializeCSS

- Phoenix uses layout files to define the overall structure of HTML pages.
- MaterializeCSS is a modern front-end framework similar to Bootstrap, providing ready-to-use components and a responsive grid system.
- Add MaterializeCSS to your project by including its link tag in your layout file.
- Location at: web > templates > layout > app.html.eex

Implementing the Header with MaterializeCSS

```
<body>
<nav class="light-blue">
<div class="nav-wrapper container">
<a href="/" class="brand-logo">Logo</a>
<ul class="right">
<%= if @conn.assigns[:user] do %>
<li>
<%= link "Logout", to: session_path(@conn, :signout) %>
</li>
```

Implementing the Header with MaterializeCSS

```
      <% else %>
      <li>
      <%= link "Sign in with Github", to: session_path(@conn, :request, "github") %>
      </li>
      <% end %>
      </ul>
      </div>
      </nav>
```

Summary

- We introduced the Phoenix framework and discussed the necessary installations.
- Covered the Phoenix request-response workflow and project generation.
- Reviewed the project requirements, server-side templating vs. SPA, and implemented a MaterializeCSS header.