**Adding a Cookie to a Website**

The project focuses on implementing cookie functionality in a Flask web application. It involves creating, displaying, and removing cookies using Python's Flask framework. The application demonstrates how cookies can be used to store and retrieve information between the client's browser and the server. The tasks involve adding a cookie with a specific value, retrieving and displaying the cookie's value, and removing the cookie. Screenshots are provided to showcase the successful execution of each task and validate the presence or absence of cookies in the browser settings. This project serves as a practical introduction to working with cookies in web development and emphasizes the use of Flask to handle cookies effectively.

**Steps:**

First, **app.py** file is opened in the provided Flask project.

A screen shot of a computer program

Description automatically generated

The Flask application is run, and upon navigating to "[http://localhost:88](http://localhost:88/)", the web browser displays the message "Practicing Cookies!".

A screenshot of a computer

Description automatically generated

In the **addCookie()** method, a cookie named "myFirstCookie" is added to the response with the value "Hello World - my first cookie!".

A screen shot of a computer program

Description automatically generated

The web browser is pointed to "<http://localhost:88/addCookie>", and the browser displays the message "Cookie added!" after the cookie is successfully added.

A screenshot of a computer

Description automatically generated

After looking at the web browser settings, and it is verified that the new cookie "myFirstCookie" is present.

A screenshot of a computer

Description automatically generated

In the **displayCookieValue()** method, the cookie value for "myFirstCookie" is retrieved from the request's cookies, and the browser displays "Found the cookie: " followed by the cookie value.

A screen shot of a computer program

Description automatically generated

The web browser is pointed to "<http://localhost:88/displayCookieValue>" after adding the cookie. The browser displays "Found the cookie: Hello World - my first cookie!".

A screenshot of a computer

Description automatically generated

The cookie "myFirstCookie" is removed from the browser settings.

A screenshot of a computer

Description automatically generated

The web browser is pointed to "<http://localhost:88/displayCookieValue>" again. Since the cookie has been removed, the browser displays "Cookie not found!".

A screenshot of a computer

Description automatically generated

In the **removeCookie()** method, the cookie "myFirstCookie" is removed. In this case, the cookie named "myFirstCookie" is being set with an empty value (**""**), meaning it will be stored with no specific content. Setting an empty value for a cookie is used when you want to remove the cookie from the client's browser. By setting an empty value and setting the **max\_age** attribute to 0, the cookie will be effectively deleted from the client's browser. The browser will remove the cookie from its storage since it has a max\_age of 0, which means it has already expired.

A screenshot of a computer screen

Description automatically generated

First it is made sure that the cookie is added.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Then the web browser is pointed to "<http://localhost:88/removeCookie>", and it is verified that the cookie has been removed from the browser settings. The browser will display "Cookie removed!".

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

In conclusion, the project successfully demonstrates the implementation of cookie functionality in a Flask web application, enabling the creation, display, and removal of cookies to store and manage data between the client and the server.