

## **Flask Report**

### **What does this technology (library/framework/service) accomplish for you?**

Using Flask allows us to implement certain web framework features in a simple way that would have otherwise been much more complex to implement. In our specific case, we use Flask in our Python files to redirect users to different paths on the site using `redirect()` and `route()`.

### **How does this technology accomplish what it does?**

Flask Initialization:

<https://github.com/pallets/flask/blob/3c9b85469ef4abda1e996a5ec0e7bef0f1a0b394/src/flask/app.py#L100>

The first thing we do when utilizing Flask is initialize our Flask object in `app.py` with the line

```
app = Flask(__name__)
```

This tells Flask that we want to create a new WSGI application with the name “app” since “\_\_name\_\_” is equivalent to `app`. The name given to `Flask()` is important because it tells Flask everything that needs to be included in the object. The Flask object is initialized with empty variables, but the ones that are used in this project are `url_rule_class`, `session_interface`, `request_class` and `response_class`, `config_class`, and `secret_key`. The `url_rule_class` is what we use for creating URL rules when routing in `app_url_rule`, which is discussed below. `Session_interface` is used by Flask sessions which is how we keep track of the users who are logged in. The request and response classes are also used by the `route()` and `redirect()` functions that are described below. The `config_class` and `secret_key` are both used for the encryption of our sessions when we are storing user login data in the session.

Route:

<https://github.com/pallets/flask/blob/03db9194d8d229d04e6a37b14bbe2499d23922d5/src/flask/scaffold.py#L370>

Flask’s `route` function works by using a decorator to decorate a view function. The decorator function in `route` calls `add_url_rule` which is used for registering a rule for routing incoming requests and building URLs. The string that gets passed into `route` is used as the endpoint that `route` will send the user to when it gets called. The function

that gets called beneath the route() call is the view function, and inside the view function is the code of whatever the server wants to happen at that particular endpoint. A mapping of endpoints to view functions is stored in self.view\_functions so that the system knows which page to show the user upon calling route.

#### Redirect:

(No function definition is given in the Flask GitHub repository)

Returns a response that redirects the user to the target location that is passed into the redirect function as a string parameter. The string that gets passed into redirect represents the endpoint that is created by the add\_url\_rule function where the endpoint is mapped to a view function.

#### Session:

<https://github.com/pallets/flask/blob/3c9b85469ef4abda1e996a5ec0e7bef0f1a0b394/src/flask/sessions.py#L47>

We use Flask sessions in our project to keep track of who is logged in and who is logged out on our site. Flask sessions act like a python dictionary where a key is mapped to a value. Since a session is almost exactly like a dictionary, we just add the user to the session when they login, and upon logging out we pop the user's information off the session, therefore removing them from our list of people who are online. To encrypt the data being stored, we utilize the Flask object's secret\_key which we set to be a random number in login.py. Having this user data stored in the session also allows us to access their information that is stored in our MongoDB database if they are logged in.

### **What license(s) or terms of service apply to this technology?**

#### Flask License:

Copyright 2010 Pallets

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

In our project, this license means that the copyright holder (Pallets) is not responsible for any damages that may come as a result of our project. They are also not responsible for any loss of data that may happen in our project as a result of using Flask.