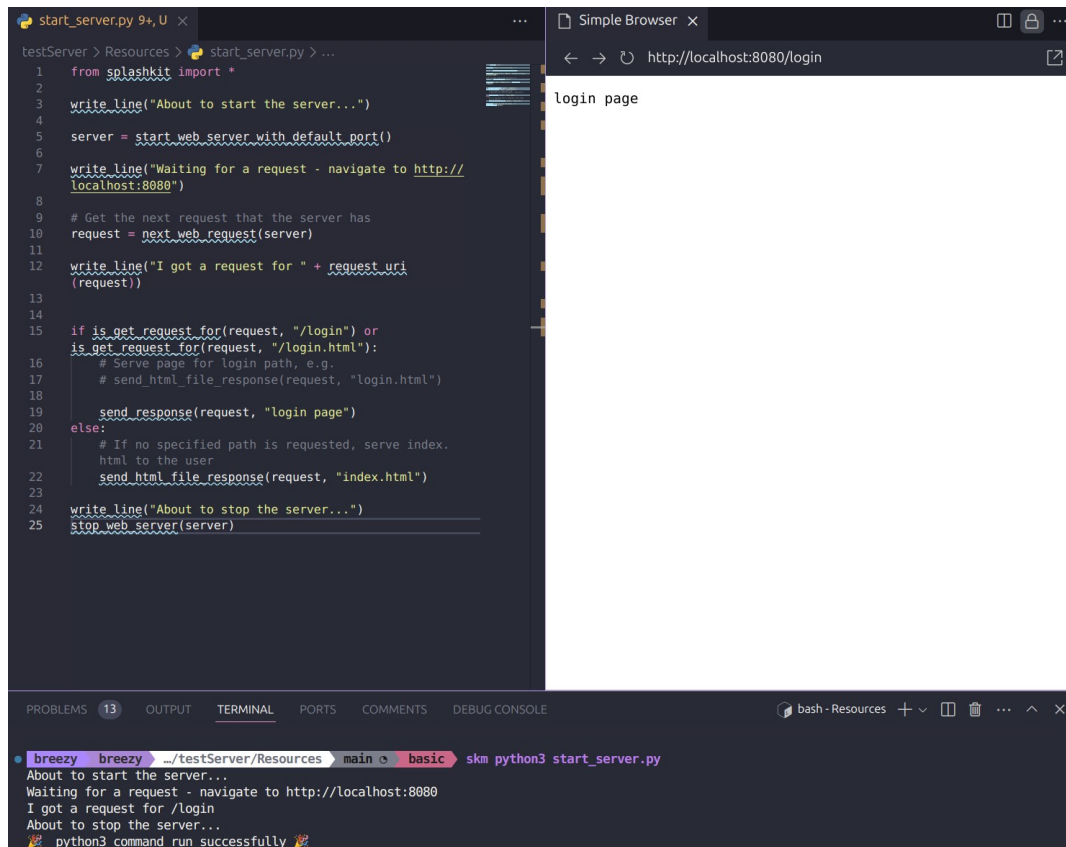


Python:

Request login:



The screenshot shows a Python web server running on localhost:8080. The server is implemented using the `splashkit` library. The code in `start_server.py` starts a web server, waits for a request, and serves the `login.html` file when the request is for `/login`. The terminal output shows the server starting, waiting for a request, receiving a request for `/login`, and serving the `login page`. The browser window shows the login page at `http://localhost:8080/login`.

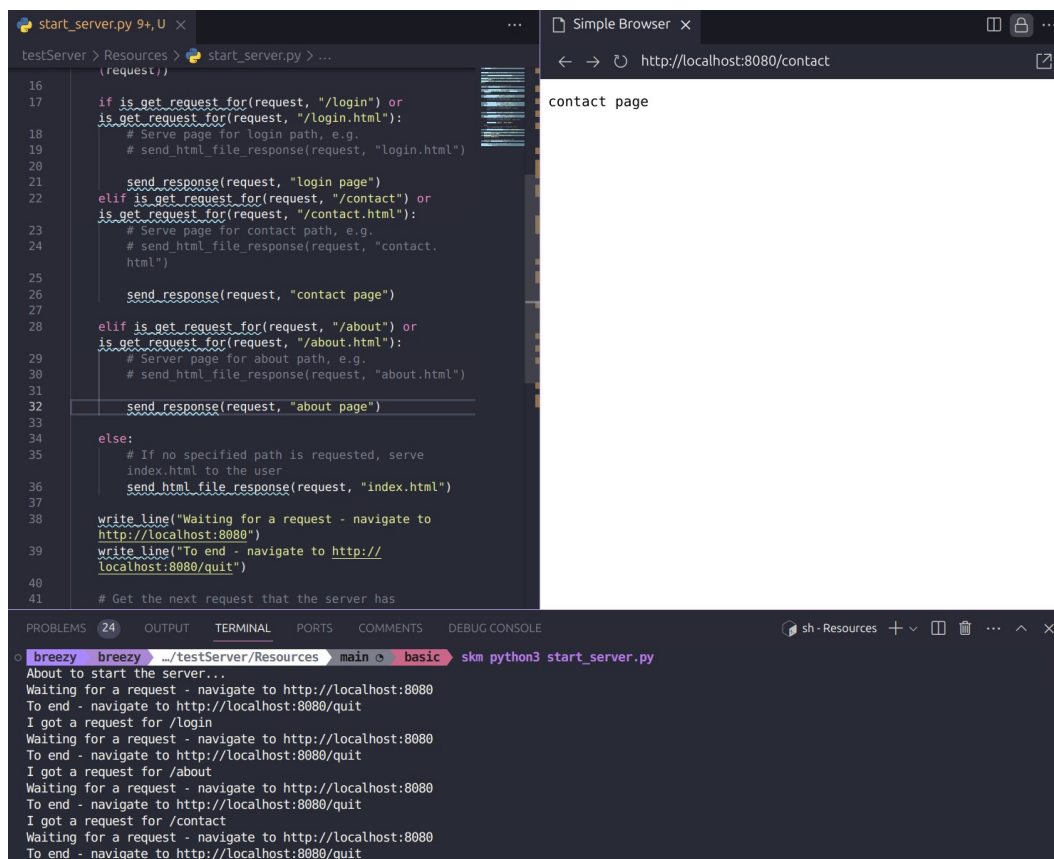
```
1 from splashkit import *
2
3 write_line("About to start the server...")
4
5 server = start_web_server_with_default_port()
6
7 write_line("Waiting for a request - navigate to http://localhost:8080")
8
9 # Get the next request that the server has
10 request = next_web_request(server)
11
12 write_line("I got a request for " + request.uri
13 (request))
14
15 if is_get_request_for(request, "/login") or
16 is_get_request_for(request, "/login.html"):
17     # Serve page for login path, e.g.
18     # send_html_file_response(request, "login.html")
19     send_response(request, "login page")
20 else:
21     # If no specified path is requested, serve index.
22     # html to the user
23     send_html_file_response(request, "index.html")
24
25 write_line("About to stop the server...")
26 stop_web_server(server)
```

terminal output:

```
breazy breezy ~/testServer/Resources main basic skm python3 start_server.py
About to start the server...
Waiting for a request - navigate to http://localhost:8080
I got a request for /login
About to stop the server...
python3 command run successfully
```

Simple Browser window: `http://localhost:8080/login`
login page

Handle multiple requests:



The screenshot shows the same Python web server, but now it is handling multiple requests. The code in `start_server.py` has been updated to serve `login.html`, `contact.html`, and `about.html` files when the request is for `/login`, `/contact`, or `/about` respectively. The terminal output shows the server starting, waiting for a request, receiving a request for `/login`, serving the `login page`, waiting for a request, receiving a request for `/about`, serving the `about page`, waiting for a request, receiving a request for `/contact`, serving the `contact page`, and so on. The browser window shows the contact page at `http://localhost:8080/contact`.

```
16 (request))
17
18 if is_get_request_for(request, "/login") or
19 is_get_request_for(request, "/login.html"):
20     # Serve page for login path, e.g.
21     # send_html_file_response(request, "login.html")
22     send_response(request, "login page")
23 elif is_get_request_for(request, "/contact") or
24 is_get_request_for(request, "/contact.html"):
25     # Serve page for contact path, e.g.
26     # send_html_file_response(request, "contact.
27     html")
28     send_response(request, "contact page")
29
30 elif is_get_request_for(request, "/about") or
31 is_get_request_for(request, "/about.html"):
32     # Server page for about path, e.g.
33     # send_html_file_response(request, "about.html")
34     send_response(request, "about page")
35
36 else:
37     # If no specified path is requested, serve
38     # index.html to the user
39     send_html_file_response(request, "index.html")
40
41 write_line("Waiting for a request - navigate to
42 http://localhost:8080")
43 write_line("To end - navigate to http://
44 localhost:8080/quit")
45
46 # Get the next request that the server has
```

terminal output:

```
breazy breezy ~/testServer/Resources main basic skm python3 start_server.py
About to start the server...
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /login
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /about
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /contact
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
```

Simple Browser window: `http://localhost:8080/contact`
contact page

C++:

Request Login:

The screenshot shows a C++ web server running on localhost:8080. The browser window displays the login page. The terminal window shows the server's output, including the request for the login page and the response sent back to the browser.

```
testServerCPP > Resources > C++ start_server.cpp > main()
3 int main()
13 request = next_web_request(server);
14 write_line("I got a request for " + request_uri
15 (request));
16
17 if (is_get_request_for(request, "/login") or
18 is_get_request_for(request, "/login.html"))
19 {
20 // Serve page for login path, e.g.
21 // send_html_file_response(request, "login.
22 html");
23
24 send_response(request, "login page");
25
26 }
27 else
28 {
29 //If no specified path is requested, serve
30 index.html to the user
31 send_html_file_response(request, "index.html");
32
33 }
34
35 write_line("About to stop the server...");
36 stop_web_server(server);
37
38 return 0;
39 }
```

login page

```
bash - Resources
breezy breezy ~/testServerCPP/Resources main basic g++ start_server.cpp -l SplashKit -o start_server
breezy breezy ~/testServerCPP/Resources main basic ./start_server
About to start the server...
Waiting for a request - navigate to http://localhost:8080
I got a request for /login
About to stop the server...
```

Handling Multiple Requests:

The screenshot shows a C++ web server running on localhost:8080. The browser window displays the about page. The terminal window shows the server's output, including the request for the about page and the response sent back to the browser.

```
testServerCPP > Resources > C++ start_server.cpp > main()
3 int main()
33
34 else if (is_get_request_for(request, "/about")
35 or is_get_request_for(request, "/about.html"))
36 {
37 // Server page for about path, e.g.
38 // send_html_file_response(request, "about.
39 html");
40
41 send_response(request, "about page");
42
43 }
44 else
45 {
46 //If no specified path is requested, serve
47 index.html to the user
48 send_html_file_response(request, "index.
49 html");
50
51 }
52
53 write_line("Waiting for a request - navigate to
54 http://localhost:8080");
55 write_line("To end - navigate to http://
56 localhost:8080/quit");
57
58 //Get the next request that the server has
59 request = next_web_request(server);
60
61 }
62
63 write_line("About to stop the server...");
64 stop_web_server(server);
65
66 return 0;
67 }
```

about page

```
./start_server - Resources
breezy breezy ~/testServerCPP/Resources main basic ./start_server
About to start the server...
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /login
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /contact
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /about
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
```

C#:

Request Login:

The screenshot shows a Visual Studio Code editor with a C# file named `Program.cs` and a Simple Browser window. The browser is at `http://localhost:8080/login` and displays "login page". The terminal shows the output of the program.

```
class Program
{
    static void Main()
    {
        // Get the next request that the server has
        request = SplashKit.NextWebRequest(server);

        SplashKit.WriteLine("I got a request for " +
            SplashKit.RequestURI(request));

        if (SplashKit.IsGetRequestFor(request, "/" +
            "login") || SplashKit.IsGetRequestFor(request, "/" +
            "login.html"))
        {
            // Serve page for login path, e.g.
            // SplashKit.SendHtmlFileResponse(request,
            // "login.html");

            SplashKit.SendResponse(request, "login
            page");
        }
        else
        {
            // If no specified path is requested, serve
            index.html to the user
            SplashKit.SendHtmlFileResponse(request,
            "index.html");
        }

        SplashKit.WriteLine("About to stop the server...
        ");
        SplashKit.StopWebServer(server);
    }
}
```

```
bash - Server-CSharp
breezy breezy ~/Resources/Server-CSharp main basic dotnet run build
About to start the server...
Waiting for a request - navigate to http://localhost:8080
I got a request for /login
About to stop the server...
```

Handling Multiple Requests:

The screenshot shows a Visual Studio Code editor with a C# file named `Program.cs` and a Simple Browser window. The browser is at `http://localhost:8080/about` and displays "about page". The terminal shows the output of the program.

```
while (!SplashKit.IsGetRequestFor(request, "/quit"))
{
    else if (SplashKit.IsGetRequestFor(request, "/" +
        "about") || SplashKit.IsGetRequestFor(request, "/" +
        "about.html"))
    {
        // Serve page for about path, e.g.
        // SendHtmlFileResponse(request, "about.html");

        SplashKit.SendResponse(request, "about page");
    }
    else
    {
        // If no specified path is requested, serve
        index.html to the user
        SplashKit.SendHtmlFileResponse(request, "index.
        html");
    }

    SplashKit.WriteLine("Waiting for a request -
    navigate to http://localhost:8080");
    SplashKit.WriteLine("To end - navigate to http://
    localhost:8080/quit");

    // Get the next request that the server has
    request = SplashKit.NextWebRequest(server);
}

SplashKit.WriteLine("About to stop the server...");
SplashKit.StopWebServer(server);
```

```
dotnet - Server-CSharp
breezy breezy ~/Resources/Server-CSharp main basic dotnet run build
About to start the server...
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /login
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /contact
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
I got a request for /about
Waiting for a request - navigate to http://localhost:8080
To end - navigate to http://localhost:8080/quit
```