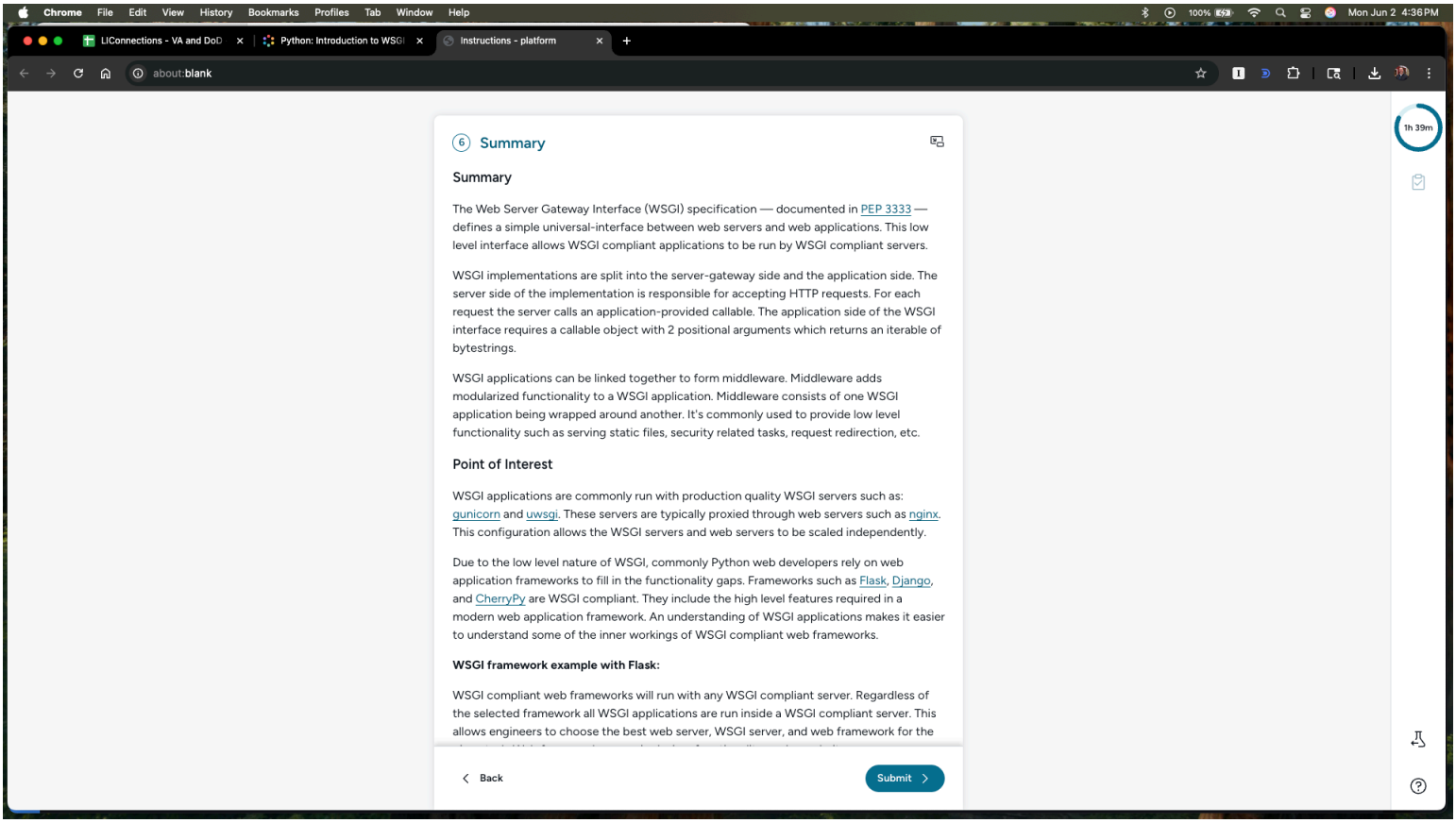
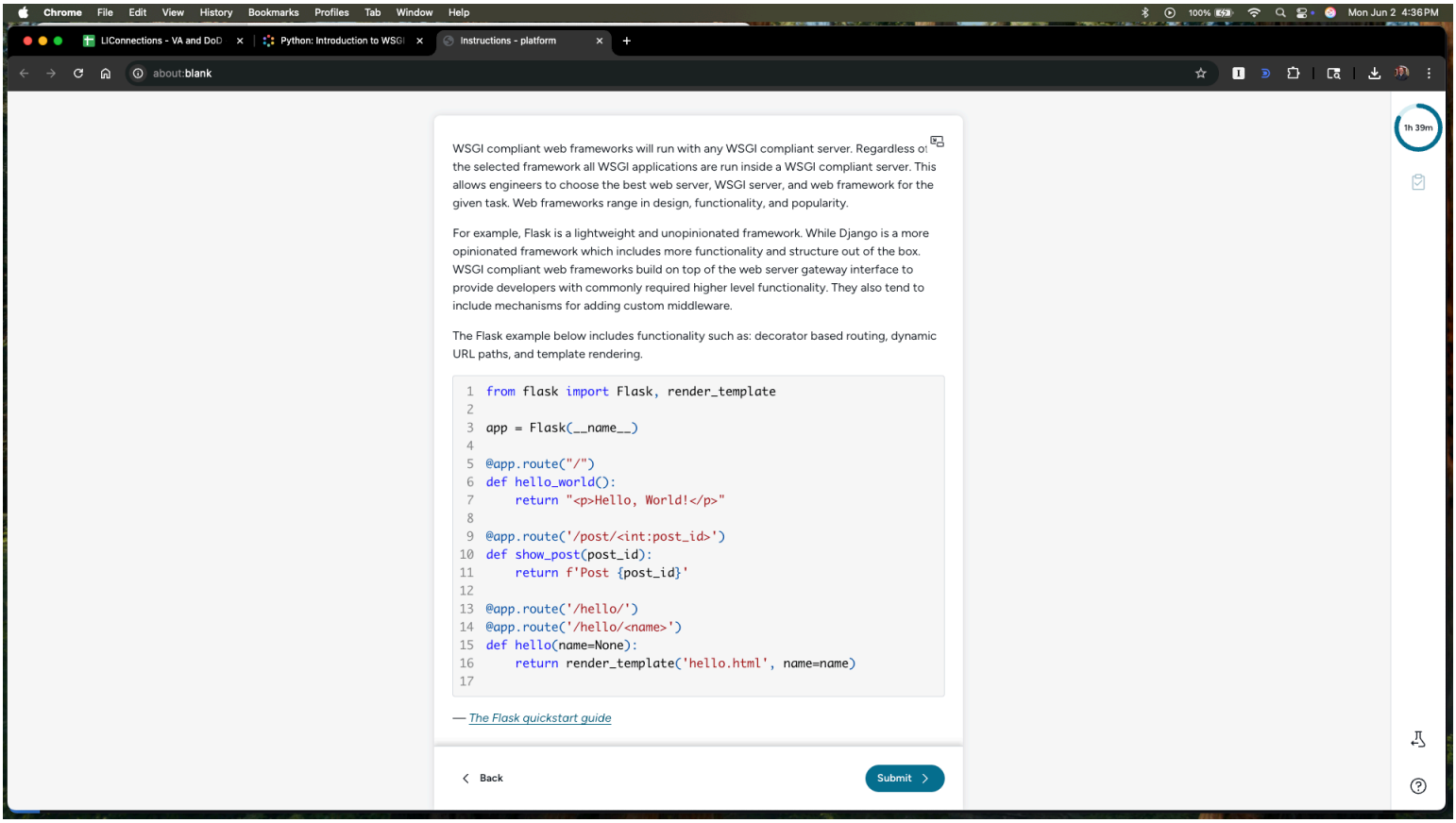

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon Jun 2 4:36 PM
LIConnections - VA and DoD
Python: Introduction to WSGI
Instructions - platform
about:blank
6 Summary
Summary
Summary
The Web Server Gateway Interface (WSGI) specification — documented in PEP 3333 — defines a simple universal-interface between web servers and web applications. This low level interface allows WSGI compliant applications to be run by WSGI compliant servers.
WSGI implementations are split into the server-gateway side and the application side. The server side of the implementation is responsible for accepting HTTP requests. For each request the server calls an application-provided callable. The application side of the WSGI interface requires a callable object with 2 positional arguments which returns an iterable of bytestrings.
WSGI applications can be linked together to form middleware. Middleware adds modularized functionality to a WSGI application. Middleware consists of one WSGI application being wrapped around another. It's commonly used to provide low level functionality such as serving static files, security related tasks, request redirection, etc.
Point of Interest
WSGI applications are commonly run with production quality WSGI servers such as: gunicorn and uwsgi. These servers are typically proxied through web servers such as nginx. This configuration allows the WSGI servers and web servers to be scaled independently.
Due to the low level nature of WSGI, commonly Python web developers rely on web application frameworks to fill in the functionality gaps. Frameworks such as Flask, Django, and CherryPy are WSGI compliant. They include the high level features required in a modern web application framework. An understanding of WSGI applications makes it easier to understand some of the inner workings of WSGI compliant web frameworks.
WSGI framework example with Flask:
WSGI compliant web frameworks will run with any WSGI compliant server. Regardless of the selected framework all WSGI applications are run inside a WSGI compliant server. This allows engineers to choose the best web server, WSGI server, and web framework for the
Back
Submit
1h 39m


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Mon Jun 2 4:36 PM
LIConnections - VA and DoD
Python: Introduction to WSGI
Instructions - platform
about:blank
1h 39m
WSGI compliant web frameworks will run with any WSGI compliant server. Regardless of the selected framework all WSGI applications are run inside a WSGI compliant server. This allows engineers to choose the best web server, WSGI server, and web framework for the given task. Web frameworks range in design, functionality, and popularity.
For example, Flask is a lightweight and unopinionated framework. While Django is a more opinionated framework which includes more functionality and structure out of the box. WSGI compliant web frameworks build on top of the web server gateway interface to provide developers with commonly required higher level functionality. They also tend to include mechanisms for adding custom middleware.
The Flask example below includes functionality such as: decorator based routing, dynamic URL paths, and template rendering.
1 from flask import Flask, render_template
2
3 app = Flask(__name__)
4
5 @app.route("/")
6 def hello_world():
7 return " Hello, World! "
8
9 @app.route('/post/<int:post_id>')
10 def show_post(post_id):
11 return f'Post {post_id}'
12
13 @app.route('/hello/')
14 @app.route('/hello/<name>')
15 def hello(name=None):
16 return render_template('hello.html', name=name)
17
— The Flask quickstart guide
Back
Submit