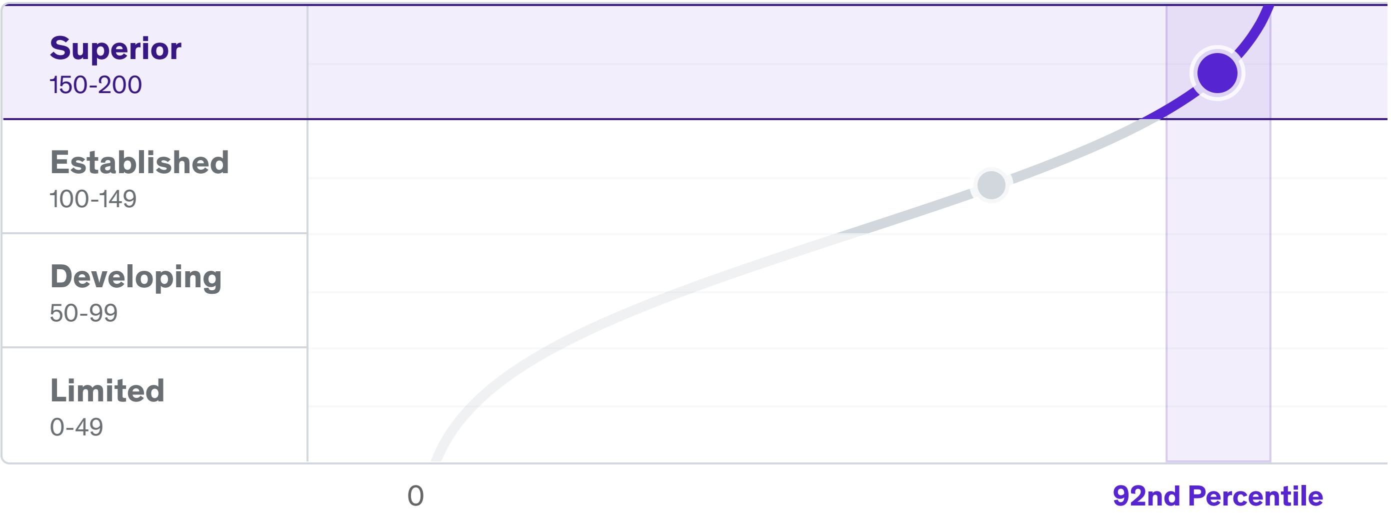
Flask **Beta** Save and exit

Flask

31 questions

Your score: 170

What do these numbers mean?



Your performance was better than **92%** of Udemy learners who have completed this assessment.

Your answers

Review your answers. Learn from these explanations of correct and incorrect response options.

All domains ⌄ All answers ⌄ **25** correct **6** incorrect **0** skipped

Retake assessment

✓ Question 1 Correct

What choice would you have made with the debug option to initialize the Flask application for a production environment?

You selected **Answer 4 Correct**

FALSE

Answer explanation: The option should be turned off (FALSE) for the production mode. This will help avoiding outputting errors to the users.

Send feedback

✓ Question 2 Correct

Which of the following Jinja templates correctly implements a simple if-else statement?

You selected **Answer 1 Correct**

```
{% if name %}
    <h1>Hello, {{name}}!</h1>
{% else %}
    <h1>What is your name?</h1>
{% endif %}
```

Answer explanation: Conditional statements should be enclosed in {% ... %}. Variables should be enclosed in double curly brackets {{ ... }}. {% endif %} should be used to signify the end of the if-else block.

Send feedback

✗ Question 3 Incorrect

What functions from Flask-Login can be used as decorators to check if the user is logged in and authenticated before calling the view function? Select all that apply.

login_enabled()

Answer explanation: login_enabled() is not a function or variable available in Flask-Login. The login_required() decorator will only let logged and authenticated users call the view function.

You selected **Answer 3** **Correct**

fresh_login_required()

Answer explanation: The fresh_login_required() function can be used a decorator to check if a user is logged, authenticated, and "fresh" before a view function is called. A user is determined to be "fresh" if they logged in via the "remember me" functionality. In some cases such as changing profile information (address, password, etc.) you may want to have a user relogin if their login isn't "fresh".

You selected **Answer 4** **Correct**

login_required()

Answer explanation: login_required() can decorate a view to check if a user is logged in and authenticated before the view is called. It checks the current_user.is_authenticated, which is set when a user logs in and logs out.

Send feedback

✓ **Question 4** **Correct**

Which of the following errorhandler decorator functions renders a custom template when an HTTP 404 error occurs?

You selected **Answer 1** **Correct**

```
@app.errorhandler(404)
def page_not_found(e):
    return render_template('404.html'), 404
```

Answer explanation: The errorhandler() decorator allows the user to create a custom error page that will be displayed to the user when an error occurs. In this example, when a 404 error occurs, the 404.html template will be rendered and displayed to the user.

Send feedback

✓ **Question 5 Correct**

What event occurs immediately after a user creates and sends a text query in the search bar of the web site?

You selected **Answer 1 Correct**

The query is sent from the client to the application server.

Answer explanation: Immediately after a query is created from user input, it must be sent from the client to the application server. Once the application receives the query, it can produce results that are sent back to the client.

Send feedback

✓ **Question 6 Correct**

How can the user session be accessed in Flask?

You selected **Answer 1 Correct**

The session variable

Answer explanation: The session variable refers to the user session. It is used to store values that need to be remembered in between requests.

Send feedback

✓ Question 7 Correct

Which line of code correctly adds the below Foo class as a resource to a Flask application?

```
class Foo(Resource):
    def get(self, foo_id):
        return {foo_id: foos[foo_id]}

    def put(self, foo_id):
        foos[foo_id] = request.form['data']
        return {foo_id: foos[foo_id]}
```

× Question 8 Incorrect

In order to validate a form with Flask-WTF, what HTTP request method must be used to submit the form? Select all that apply.

Answer 2 Correct

PUT

Answer explanation: The PUT HTTP method is an acceptable request method for submitting a form and validating it. The is_submitted() function checks the HTTP request method. The validate_on_submit() function calls the is_submitted() function.

You selected **Answer 3 Correct**

POST

Answer explanation: POST is an acceptable and common HTTP method used to submit forms. The quick_form() function in Flask-WTF creates a POST request when the form is submitted.

Answer 4 Correct

PATCH

Answer explanation: PATCH is an accepted HTTP method for submitting a form if it is to be validated using the validate_on_submit() function in Flask-WTF. The is_submitted() function checks the type of request.

Send feedback

✓ **Question 9 Correct**

The jsonify() function call below will wrap which of the following in a Response?

```
jsonify(
    student_id =3287,
    course="physics"
)
```

You selected **Answer 1 Correct**

```
{
    "student_id " : 3287,
    "course" : "physics"
}
```

Answer explanation: The jonsify() function serializes given data, converting it to a JSON object. The JSON is then wrapped in a Response Flask object.

Send feedback

✓ **Question 10 Correct**

Which of the following url_for() function calls will produce "htt//localhost/home"?

You selected **Answer 4 Correct**

```
@app.route('/home')
def index():
    return '<p>You are home.</p>'

with app.app_context():
    url_for('index')
```

Answer explanation: The value "index" is referring to the endpoint and view function index()

Send feedback

✓ **Question 11** **Correct**

Which of the following Python frameworks has an admin panel built in?

You selected **Answer 3** **Correct**

Django

Answer explanation: The basic Django framework has a built-in admin panel. The provided admin panel is a UI that devs can interact with and customize to easily handle administrative tasks.

Send feedback

✓ **Question 12** **Correct**

What best explains the relationship between REST and RESTful?

You selected **Answer 4** **Correct**

REST is an architectural pattern for creating web services. A service or an API is considered to be RESTful when it implements that pattern.

Answer explanation: The REST design pattern is widely used across the industry to create stateless web services often known as RESTful APIs or services. RESTful APIs implement a REST architecture through the service.

Send feedback

✓ **Question 13** **Correct**

What is the default behavior when the "Submit" button is clicked on a web form? Select all that apply.

The webpage is refreshed.

Answer explanation: The webpage will by default refresh when the submit button is clicked and the form is successfully validated. There may be times where the desired behavior is to redirect the user to another page rather than refreshing the webpage.

You selected **Answer 3** **Correct**

The data is validated.

Answer explanation: Data in the form is checked to ensure that it is valid. This occurs before a request is sent to the server.

You selected **Answer 4** **Correct**

A GET request is sent.

Answer explanation: By default, forms use GET requests. This is often not the preferred method because POST requests are more secure and easier to handle when modifying data.

Send feedback

✓ Question 14 Correct

Analyze the template below. Which of the following routes correctly passes the "name" variable into the template?

```
<div class="container">
        <div class="page-header">
                <h1>Hello, {{name}}!</h1>
        </div>
</div>
```

You selected **Answer 2** **Correct**

```
@app.route('/user/<name>')
def user(name):
```

Answer explanation: A variable can be passed from URL to a template via the view function. The function retrieves the variable from the URL and passes it to the template.

Send feedback

✓ **Question 15** **Correct**

If an expired token attempts to access a protected endpoint, what JSON response will be received?

You selected **Answer 4** **Correct**

{"msg": "Token has expired"} and a 401 status code

Answer explanation: {"msg": "Token has expired"} is a JSON response that will be returned along with a 401 status code when an expired token attempts to access an endpoint. The JSON object is in the form of a key value pair within curly braces. The message can be modified to a custom message.

Send feedback

✓ **Question 16** **Correct**

How many requests can a Flask application process at a time?

You selected **Answer 4** **Correct**

Flask applications process one request per thread.

Answer explanation: One request can be handled per thread. By default, the app is single threaded, but this can easily be changed with some configuration adjustments. Multiple threads can be working on different requests at the same time.

Send feedback

✕ Question 17 Incorrect

What would the user argument be in the url_for() function in the HTML snippet <a href={{ url_for('user') }}>User</a>?

You selected **Answer 2** **Incorrect**

a view

Answer explanation: This is a proper way to refer to a route. The argument is referring to the @app.route('/user').

Answer 4 **Correct**

a route

Answer explanation: The user argument does not refer to a template. The argument is referring to the @app.route('/user').

Send feedback

✓ Question 18 Correct

Which of the following lines can be used in a template to inherit another template that provides a basic uniform layout?

You selected **Answer 2** **Correct**

{% extends "layout.html" %}

Answer explanation: An extends statement enclosed in {% ... %} can be used to inherit a parent template. This can be used to create a uniform layout across multiple webpages and simplify the child templates.

Send feedback

✓

Question 19 Correct

Which of the following snippets will correctly display flashed messages?

You selected **Answer 4 Correct**

```
{% with messages = get_flashed_messages() %}
   <ul class="flashes">
    {% for message in messages %}
      <li>{{ message }}</li>
    {% endfor %}
   </ul>
{% endwith %}
```

Answer explanation: The get_flashed_messages() function retrieves messages from storage and displays them to the user. It returns a list of messages even if only one message is retrieved.

Send feedback

✓

Question 20 Correct

When developing a Flask application, what vulnerabilities need to be considered regarding Cross-Site Scripting and Cross-Site Request Forgery? Select all that apply.

You selected **Answer 1 Correct**

Jinja2 not auto-escaping all HTML

Answer explanation: Jinja has built-in autoescape capabilities, but autoescape is currently not enabled by default. The select_autoescape() function can be used to explicitly define autoescaping behavior.

You selected **Answer 2 Correct**

Storing authentication information in cookies

Answer explanation: Storing authentication information in cookies is standard. However, it does make users vulnerable to third party requests. Extra care and security should be taken for requests that modify data on the server.

You selected **Answer 3** **Correct**

Calling Markup() on user input

Answer explanation: The Markup class wraps text without escaping. Only text/values that you can guarantee are safe should be passed to the Markup constructor. You cannot ensure user input is non-malicious.

You selected **Answer 4** **Correct**

Using the | safe filter

Answer explanation: The | safe filter explicitly marks everything passed to it as "safe". This allows autoescaping to be bypassed.The | safe filter should only be used with trusted content which is not from users and which you can ensure is indeed safe.

Send feedback

✓ **Question 21** **Correct**

What is generate_password_hash used for?

You selected **Answer 2** **Correct**

It takes a plaintext password, a hashing method, and a salt length as an input to produce a hashed password. By default, it produces a salt string with length 16.

Answer explanation: The main use of the generate_password_hash() is that it takes a plaintext password, hashing method, and salt length as an input to produce hashed passwords. By default, it produces a salt string with length 16.

Send feedback

× Question 22 Incorrect

What is required in order to initialize all Flask applications?

You selected **Answer 2 Incorrect**

Define a route() decorator

Answer explanation: A route() decorator is commonly used in Flask applications to define and handle routes between URLs and functions. However, route decorators are not an absolute requirement for a Flask application.

Answer 3 Correct

Create an instance of the Flask class

Answer explanation: An application instance is required in order for the application to run. This is typically done by passing "__name__" into the constructor of the Flask class. Python sets __name__ to the path and name of the current file.

Send feedback

✓ Question 23 Correct

How could you register an error handler in Flask?

You selected **Answer 1 Correct**

By decorating a function with errorhandler()

Answer explanation: You register an error handler in Flask by decorating a function with a function. The correct function is errorhandler()

Send feedback

✓ Question 24 Correct

Which of the following performs a query on a MongoDB instance and will return the first occurrence of a Book with the title "Dune"?

You selected **Answer 2** **Correct**

results = Book.objects(title="Dune").first()

Answer explanation: The object's attribute of a Document is a QuerySetManager. The first() method is used to return only the first occurence of the results meeting the specified criteria.

Send feedback

× Question 25 Incorrect

How can you access session data in a template?

Answer 1 **Correct**

{{ session['email'] }}

Answer explanation: Like with any variable, in template they must be surrounded by {{ }} unless they are inside of a statement. If they are inside of a statement, the entire statement is contained in {% %}. The session object is global and can be used to store various data.

You selected **Answer 2** **Incorrect**

{{ session.email }}

Answer explanation: The "email" attribute in session should be inside square brackets. It is referred to like this: {{ session['email'] }}.

Send feedback

You selected **Answer 4** **Correct**

Provide a variable in the URL path, wrapped with two opposing arrows <>. Then the variable is passed into the routes function as an argument in the views.py

Answer explanation: Square brackets < > are used to signify a variable in the URL that is passed into the route() decorator function. It's then passed to the view function associated with that particular route.

Send feedback

✓ Question 27 Correct

When using MongoEngine, which method in a QuerySet or modify() and save() is used to update documents atomically?

You selected **Answer 2** **Correct**

update(), update_one(), modify()

Answer explanation: The update function performs an atomic update on matches from a query. The modify method performs an atomic update and reloads the updated document. update_one does an atomic update on the first field that matches a query.

Send feedback

✓ Question 28 Correct

What is the main purpose of adding the NGINX as a dedicated server?

You selected **Answer 4** **Correct**

It manages web traffic, proxies requests to the WSGI server, and serves static content itself.

Answer explanation: NGINX is a type of HTTP web server that can be paired with a WSGI server, such as uWSGI or Gunicorn. It manages web traffic and proxies.

Send feedback

✓ **Question 29 Correct**

What packages should be installed in order to use Flask-Mail? Select all that apply.

You selected **Answer 1 Correct**

Flask

Answer explanation: Flask is required in order to use Flask-Mail. Flask-Mail contains Flask-specific code, so Flask must be installed.

You selected **Answer 2 Correct**

Flask-Mail

Answer explanation: Flask-Mail provides functionality to send emails to users. Flask-Mail and Flask must be installed in order to utilize this functionality.

Send feedback

✓ **Question 30 Correct**

Given the user() function below, which template correctly references the "name" variable?

```
@app.route('/user/<name>')
def user(name):
    return render_template('user.html', name=name)
```

You selected **Answer 2 Correct**

```
<div class="container">
        <div class="page-header">
                <h1>Hello, {{name}}!</h1>
        </div>
```

Answer explanation: When using a variable within a Jinja2 template, the variable name should be inside double curly brackets {{ }}. The variable should match the variable on the left side of the equals sign in the render_template() function call.

Send feedback

Question 31 Incorrect

Given the Greet class and the add_resource() method call below, what is the endpoint? class Greet(Resource): def get(self, id): return {'hello':'world'} api.add_resource(Greet, '/greetings', '/hello')

Answer 2 Correct

greet

Answer explanation: When using the add_resource() method, an endpoint can be specified in the parameters. When not specified, it defaults to the name of the Resource class in lowercase.

You selected

Answer 3 Incorrect

/greetings

Answer explanation: /greetings is a URL, as is /hello. Multiple URLs can be added to a single resource. These URLs are not endpoints. An endpoint can be specified in an argument of the add_resource() method. When not specified, it defaults to the name of the resource class in all lowercase, so the endpoint in the example would be "greet".

Send feedback

Show less