Token-Based Authentication With Flask

by Real Python 37 Comments advanced flask web-dev

Tweet Share Email

Table of Contents

- Objectives
- Introduction
- Getting Started
  - Project Setup
  - Database Setup
  - Migrations
- Sanity Check
- JWT Setup
  - Encode Token
  - Decode Token
- Route Setup
- Register Route
- Login Route
  - Registered user login
  - Non-Registered user login
- User Status Route
- Logout Route Tests
- Blacklist
- Logout Route Handler
- Refactoring
- Code Smell
- Refactor

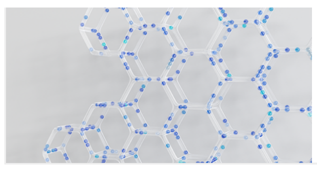
Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

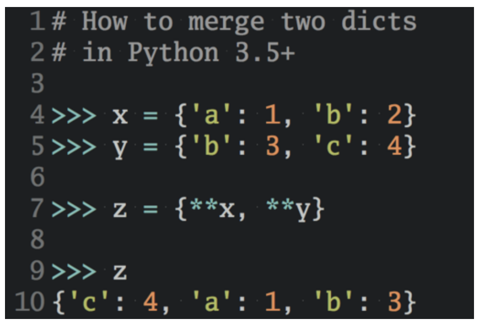
Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application



The Most Open and Secure Public Cloud for Business

Ad by IBM



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick
code snippet every couple of days:

Email Address

Send Python Tricks »

Objectives

By the end of this tutorial, you will be ab

1. Discuss the benefits of using JWTs
2. Implement user authentication wi
3. Blacklist user tokens when necess
4. Write tests to create and verify JWTs and user authentication
5. Practice test-driven development

Free Bonus: Click here to get access to a free Flask + Python video tutorial that shows you how to build Flask web app, step-by-step.

Introduction

JSON Web Tokens (or JWTs) provide a means of transmitting information from the client to the server in a stateless, secure way.

On the server, JWTs are generated by signing user information via a secret key, which are then securely stored on the client. This form of auth works well with modern, single page applications. For more on this, along with the pros and cons of using JWTs vs. session and cookie-based auth, please review the following articles:

1. Cookies vs Tokens: The Definitive Guide
2. Token Authentication vs. Cookies
3. How do sessions work in Flask?

NOTE: Keep in mind that since a JWT is signed rather than encrypted it should never contain sensitive information like a user's password.

Getting Started

Enough theory, let's start implementing some code!

Project Setup

Start by cloning the project boilerplate and then create a new branch:

Shell

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

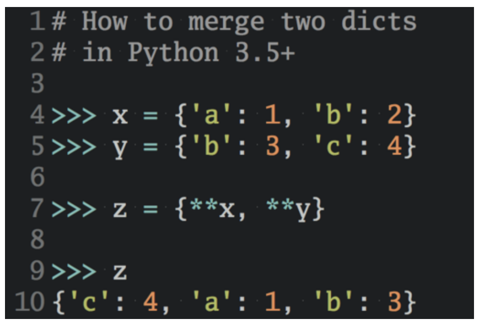
- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

Database Setup

Let's set up Postgres.

> **NOTE**: If you're on a Mac, check out F

Once the local Postgres server is runnin
project name:



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

SQL

```
(env)$ psql
# create database flask_jwt_auth;
CREATE DATABASE
# create database flask_jwt_auth_test;
CREATE DATABASE
# \q
```

> **NOTE**: There may be some variation on the above commands, for creating a database, based upon your version of Postgres. Check for the correct command in the Postgres documentation.

Before applying the database migrations we need to update the config file found in *project/server/config.py*. Simply update the `database_name`:

Python

```python
database_name = 'flask_jwt_auth'
```

Set the environment variables in the terminal:

Shell

```
(env)$ export APP_SETTINGS="project.server.config.DevelopmentConfig"
```

Update the following tests in *project/tests/test__config.py*:

Python

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

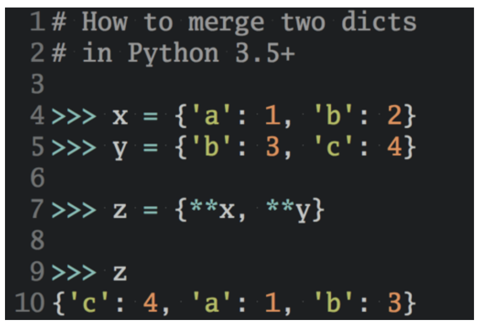
› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
    def create_app(self):
        app.config.from_object('project.server.config.TestingConfig')
        return app

    def test_app_is_testing(self):
        self.assertTrue(app.config['
        self.assertTrue(
            app.config['SQLALCHEMY_D
        )
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

Run them to ensure they still pass:

Shell

```
(env)$ python manage.py test
```

You should see:

Shell

```
test_app_is_development (test__config.TestDevelopmentConfig) ... ok
test_app_is_production (test__config.TestProductionConfig) ... ok
test_app_is_testing (test__config.TestTestingConfig) ... ok

----------------------------------------------------------------------
Ran 3 tests in 0.007s

OK
```

Software to help turn customers into products into employees into and brands into

REQUEST DEMO

Remove ads

Migrations

Add a *models.py* file to the “server” directory:

Python

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

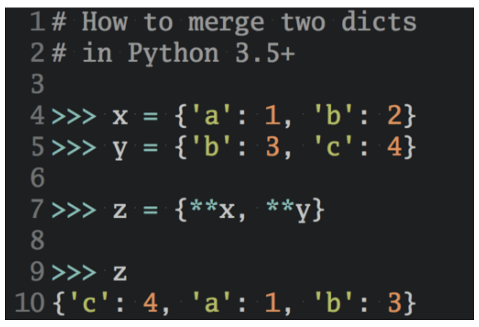
```
        self.email = email
        self.password = bcrypt.generate_password_hash(
            password, app.config.get
        ).decode()
        self.registered_on = datetim
        self.admin = admin
```

In the above snippet, we define a basic u

Install psycopg2 to connect to Postgres:

Shell

```
(env)$ pip install psycopg2==2.6.2
(env)$ pip freeze > requirements.txt
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Within *manage.py* change-

Python

```python
from project.server import app, db
```

To-

Python

```python
from project.server import app, db, models
```

Apply the migration:

Shell

```
(env)$ python manage.py create_db
(env)$ python manage.py db init
(env)$ python manage.py db migrate
```

Sanity Check

Did it work?

SQL

```
(env)$ psql
# \c flask_jwt_auth
You are now connected to database "flask_jwt_auth" as user "michael_herman".
```

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

- Server then verifies that email and password are correct and responds with an auth token
- Client stores the token and sends it along with all subsequent requests to the API
- Server decodes the token and valic

This cycle repeats until the token expire:

The tokens themselves are divided into

- Header
- Payload
- Signature

We'll dive a bit deeper into the payload,
[Introduction to JSON Web Tokens](#) article.

```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

×

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

To work with JSON Web Tokens in our app, install the [PyJWT](#) package:

Shell

```
(env)$ pip install pyjwt==1.4.2
(env)$ pip freeze > requirements.txt
```

Remove ads

Encode Token

Add the following method to the `User()` class in *project/server/models.py*:

Python

```
def encode_auth_token(self, user_id):
    """
    Generates the Auth Token
    :return: string
    """
    try:
```

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. [Learn More](#) [Continue to site](#)

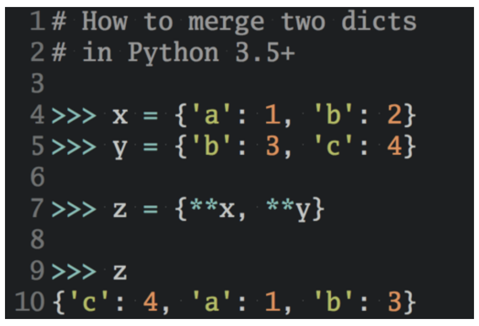
Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

So, given a user id, this method creates
file. The payload is where we add metad
to as JWT Claims. We utilize the followin

- `exp`: expiration date of the token
- `iat`: the time the token is generate
- `sub`: the subject of the token (the u

The secret key *must* be random and only



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Python

```
>>> import os
>>> os.urandom(24)
b"\xf9'\xe4p(\xa9\x12\x1a!\x94\x8d\x1c\x99l\xc7\xb7e\xc7c\x86\x02MJ\xa0"
```

Set the key as an environment variable:

Shell

```
(env)$ export SECRET_KEY="\xf9'\xe4p(\xa9\x12\x1a!\x94\x8d\x1c\x99l\xc7\xb7e\xc7c\x86\x02MJ\xa0"
```

Add this key to the `SECRET_KEY` within the `BaseConfig()` class in *project/server/config.py*:

Python

```
SECRET_KEY = os.getenv('SECRET_KEY', 'my_precious')
```

Update the tests within *project/tests/test__config.py* to ensure the variable is set correctly:

Python

```
def test_app_is_development(self):
    self.assertFalse(app.config['SECRET_KEY'] is 'my_precious')
    self.assertTrue(app.config['DEBUG'] is True)
    self.assertFalse(current_app is None)
    self.assertTrue(
        app.config['SQLALCHEMY_DATABASE_URI'] == 'postgresql://postgres:@localhost/flask_jwt_auth'
    )
```

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

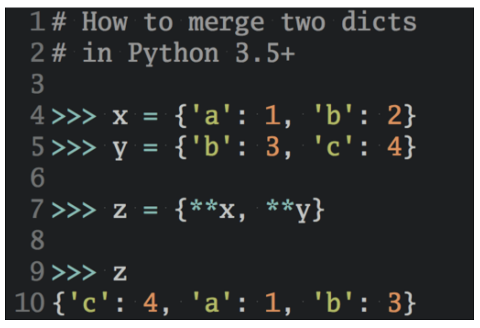
```
import unittest

from project.server import db
from project.server.models import Us
from project.tests.base import BaseT


class TestUserModel(BaseTestCase):

    def test_encode_auth_token(self)
        user = User(
            email='test@test.com',
            password='test'
        )
        db.session.add(user)
        db.session.commit()
        auth_token = user.encode_auth_token(user.id)
        self.assertTrue(isinstance(auth_token, bytes))

if __name__ == '__main__':
    unittest.main()
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Run the tests. They all should pass.

Decode Token

Similarly, to decode a token, add the following method to the `User()` class:

Python

```
@staticmethod
def decode_auth_token(auth_token):
    """
    Decodes the auth token
    :param auth_token:
    :return: integer|string
    """
    try:
        payload = jwt.decode(auth_token, app.config.get('SECRET_KEY'))
        return payload['sub']
    except jwt.ExpiredSignatureError:
        return 'Signature expired. Please log in again.'
    except jwt.InvalidTokenError:
```

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

Add a test to *test_user_model.py*:

Python

```
def test_decode_auth_token(self):
    user = User(
        email='test@test.com',
        password='test'
    )
    db.session.add(user)
    db.session.commit()
    auth_token = user.encode_auth_tc
    self.assertTrue(isinstance(auth_
    self.assertTrue(User.decode_auth
```

```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick
code snippet every couple of days:

Email Address

Send Python Tricks »

Make sure the tests pass before moving on.

NOTE: We will handle invalid tokens by blacklisting them later.

IBM Cloud

Ad by IBM

See More

Remove ads

Route Setup

Now we can configure the auth routes using a test-first approach:

- `/auth/register`
- `/auth/login`
- `/auth/logout`
- `/auth/user`

Start by creating a new folder called "auth" in "project/server". Then, within "auth" add two files, *__init__.py* and *views.py*. Finally, add the following code to *views.py*:

Python

```
# project/server/auth/views.py

from flask import Blueprint, request, make_response, jsonify
from flask.views import MethodView

from project.server import bcrypt, db
from project.server.models import User

auth_blueprint = Blueprint('auth', __name__)
```

To register the new Blueprint with the app, add the following to the bottom of *project/server/__init__.py*:

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

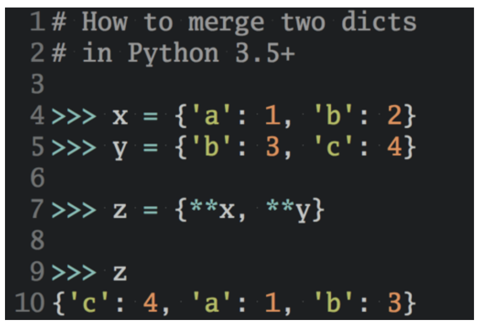
› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
class TestAuthBlueprint(BaseTestCase
    pass


if __name__ == '__main__':
    unittest.main()
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

Register Route

Start with a test:

Python

```
def test_registration(self):
    """ Test for user registration """
    with self.client:
        response = self.client.post(
            '/auth/register',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            )),
            content_type='application/json'
        )
        data = json.loads(response.data.decode())
        self.assertTrue(data['status'] == 'success')
        self.assertTrue(data['message'] == 'Successfully registered.')
        self.assertTrue(data['auth_token'])
        self.assertTrue(response.content_type == 'application/json')
        self.assertEqual(response.status_code, 201)
```

Make sure to add the import:

Python

```
import json
```

Run the tests. You should see the following error:

Python

```
raise JSONDecodeError("Expecting value", s, err.value) from None
json.decoder.JSONDecodeError: Expecting value: line 1 column 1 (char 0)
```

Now, let's write the code to get the test to pass. Add the following to *project/server/auth/views.py*:

Python

```
class RegisterAPI(MethodView):
    """
```

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

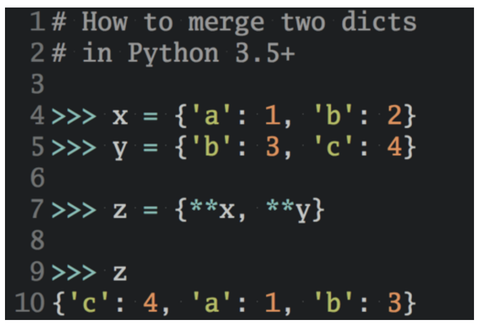
Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
                # generate the auth token
                auth_token = user.encode_auth_token(user.id)
                responseObject = {
                    'status': 'succe
                    'message': 'Succ
                    'auth_token': au
                }
                return make_response
            except Exception as e:
                responseObject = {
                    'status': 'fail'
                    'message': 'Some
                }
                return make_response
        else:
            responseObject = {
                'status': 'fail',
                'message': 'User already exists. Please Log in.',
            }
            return make_response(jsonify(responseObject)), 202

# define the API resources
registration_view = RegisterAPI.as_view('register_api')

# add Rules for API Endpoints
auth_blueprint.add_url_rule(
    '/auth/register',
    view_func=registration_view,
    methods=['POST']
)
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Here, we register a new user and generate a new auth token for further requests, which we send back to the client.

Run the tests to ensure they all pass:

Shell

```
Ran 6 tests in 0.132s

OK
```

Next, let's add one more test to ensure the registration fails if the user already exists:

Python

```
def test_registered_with_already_registered_user(self):
    """ Test registration with already registered email"""
    user = User(
        email='joe@gmail.com',
        password='test'
    )
    db.session.add(user)
    db.session.commit()
    with self.client:
        response = self.client.post(
            '/auth/register',
```

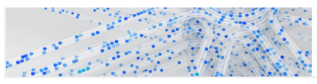
Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

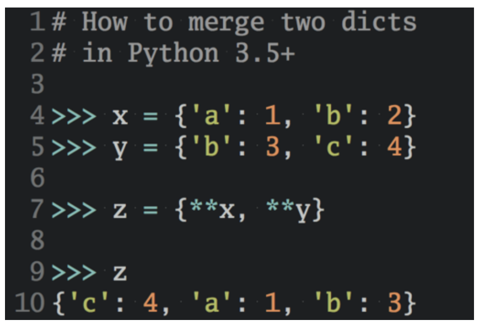


Ad by IBM See More

Login Route

Again, start with a test. To verify the logi

1. Registered user login
2. Non-registered user login



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

Registered user login

Python

```
def test_registered_user_login(self):
    """ Test for login of registered-user login """
    with self.client:
        # user registration
        resp_register = self.client.post(
            '/auth/register',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            )),
            content_type='application/json',
        )
        data_register = json.loads(resp_register.data.decode())
        self.assertTrue(data_register['status'] == 'success')
        self.assertTrue(
            data_register['message'] == 'Successfully registered.'
        )
        self.assertTrue(data_register['auth_token'])
        self.assertTrue(resp_register.content_type == 'application/json')
        self.assertEqual(resp_register.status_code, 201)
        # registered user login
        response = self.client.post(
            '/auth/login',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            ))
```

Ads help us run this site

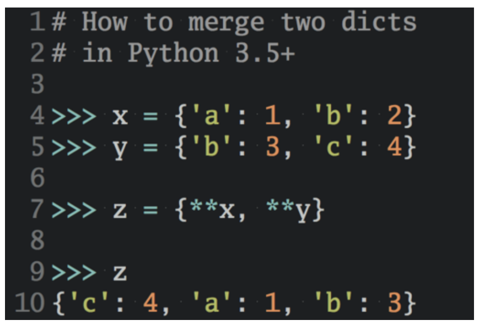
When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
    User Login Resource
    """
    def post(self):
        # get the post data
        post_data = request.get_jsor
        try:
            # fetch the user data
            user = User.query.filter
                email=post_data.get(
              ).first()
            auth_token = user.encode
            if auth_token:
                responseObject = {
                    'status': 'succe
                    'message': 'Succ
                    'auth_token': au
                }
                return make_response(jsonify(responseObject)), 200
        except Exception as e:
            print(e)
            responseObject = {
                'status': 'fail',
                'message': 'Try again'
            }
            return make_response(jsonify(responseObject)), 500
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Don't forget to convert the class to a view function:

Python

```
# define the API resources
registration_view = RegisterAPI.as_view('register_api')
login_view = LoginAPI.as_view('login_api')

# add Rules for API Endpoints
auth_blueprint.add_url_rule(
    '/auth/register',
    view_func=registration_view,
    methods=['POST']
)
auth_blueprint.add_url_rule(
    '/auth/login',
    view_func=login_view,
    methods=['POST']
)
```

Run the tests again. Do they pass? They should. Don't move on until all tests pass.

Non-Registered user login

Add the test:

Python

```
def test_non_registered_user_login(self):
```

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

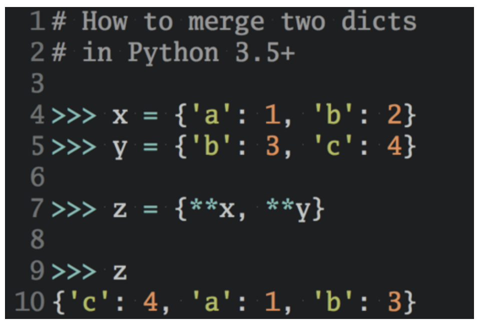
› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

Run the tests, and then update the code:

Python



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

```
class LoginAPI(MethodView):
    """
    User Login Resource
    """
    def post(self):
        # get the post data
        post_data = request.get_json()
        try:
            # fetch the user data
            user = User.query.filter_by(
                email=post_data.get('email')
            ).first()
            if user and bcrypt.check_password_hash(
                user.password, post_data.get('password')
            ):
                auth_token = user.encode_auth_token(user.id)
                if auth_token:
                    responseObject = {
                        'status': 'success',
                        'message': 'Successfully logged in.',
                        'auth_token': auth_token.decode()
                    }
                    return make_response(jsonify(responseObject)), 200
            else:
                responseObject = {
                    'status': 'fail',
                    'message': 'User does not exist.'
                }
                return make_response(jsonify(responseObject)), 404
        except Exception as e:
            print(e)
            responseObject = {
                'status': 'fail',
```

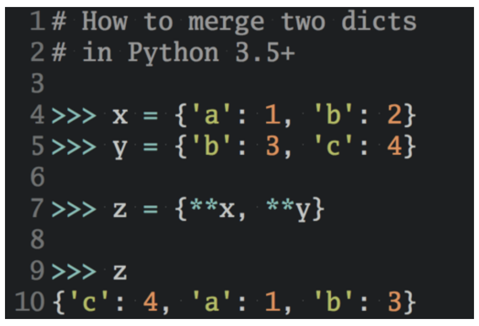
Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

```python
def test_user_status(self):
    """ Test for user status """
    with self.client:
        resp_register = self.client.post(
            '/auth/register',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            )),
            content_type='application/json'
        )
        response = self.client.get(
            '/auth/status',
            headers=dict(
                Authorization='Bearer ' + json.loads(
                    resp_register.data.decode()
                )['auth_token']
            )
        )
        data = json.loads(response.data.decode())
        self.assertTrue(data['status'] == 'success')
        self.assertTrue(data['data'] is not None)
        self.assertTrue(data['data']['email'] == 'joe@gmail.com')
        self.assertTrue(data['data']['admin'] is 'true' or 'false')
        self.assertEqual(response.status_code, 200)
```

The test should fail. Now, in the handler class, we should:

- extract the auth token and check its validity
- grab the user id from the payload and get the user details (if the token is valid, of course)

Python

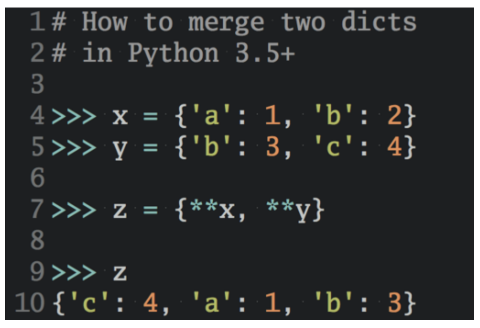
Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

```
class UserAPI(MethodView):
    """
    User Resource
    """
    def get(self):
        # get the auth token
        auth_header = request.headers.get('Authorization')
        if auth_header:
            auth_token = auth_header.split(" ")[1]
        else:
            auth_token = ''
        if auth_token:
            resp = User.decode_auth_token(auth_token)
            if not isinstance(resp, str):
                user = User.query.filter_by(id=resp).first()
                responseObject = {
                    'status': 'success',
                    'data': {
                        'user_id': user.id,
                        'email': user.email,
                        'admin': user.admin,
                        'registered_on': user.registered_on
                    }
                }
                return make_response(jsonify(responseObject)), 200
            responseObject = {
                'status': 'fail',
                'message': resp
            }
            return make_response(jsonify(responseObject)), 401
        else:
            responseObject = {
                'status': 'fail',
                'message': 'Provide a valid auth token.'
            }
            return make_response(jsonify(responseObject)), 401
```

So, if the token is valid and not expired, we get the user id from the token's payload, which is then used to get the user data from the database.

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

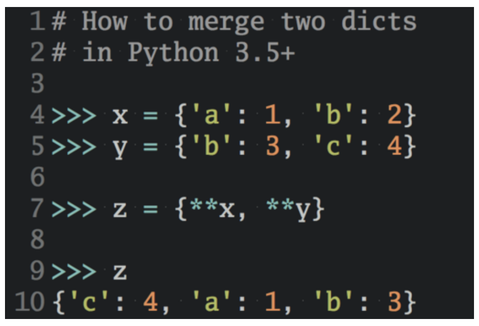
```
        methods=['GET']
)
```

The tests should pass:

Shell

```
Ran 10 tests in 0.240s

OK
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

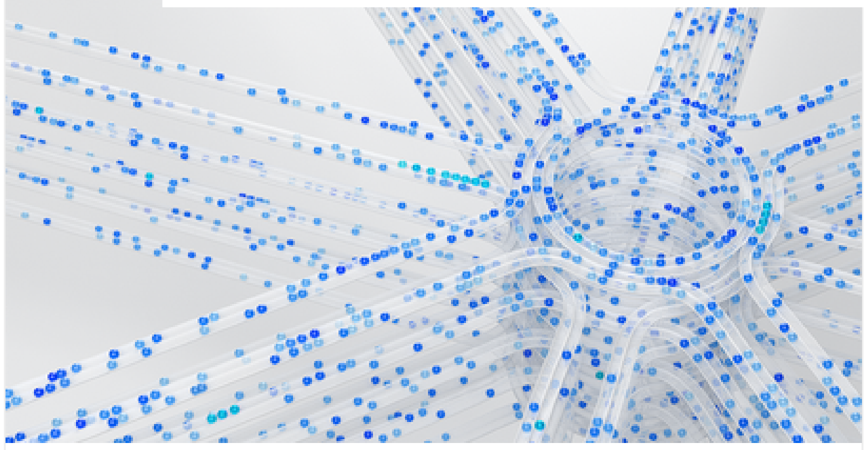
Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

One more route to go!



IBM Cloud

Protect your business with the resiliency and flexibility of IBM Cloud.

Ad by IBM See More

Remove ads

Logout Route Tests

Tests valid logout:

Python

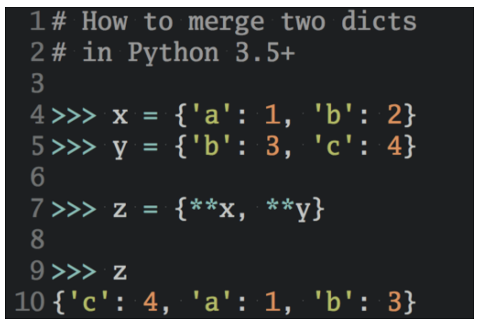
Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

```
def test_valid_logout(self):
    """ Test for logout before toker
    with self.client:
        # user registration
        resp_register = self.client.
            '/auth/register',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            )),
            content_type='application/json',
        )
        data_register = json.loads(resp_register.data.decode())
        self.assertTrue(data_register['status'] == 'success')
        self.assertTrue(
            data_register['message'] == 'Successfully registered.')
        self.assertTrue(data_register['auth_token'])
        self.assertTrue(resp_register.content_type == 'application/json')
        self.assertEqual(resp_register.status_code, 201)
        # user login
        resp_login = self.client.post(
            '/auth/login',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            )),
            content_type='application/json'
        )
        data_login = json.loads(resp_login.data.decode())
        self.assertTrue(data_login['status'] == 'success')
        self.assertTrue(data_login['message'] == 'Successfully logged in.')
        self.assertTrue(data_login['auth_token'])
        self.assertTrue(resp_login.content_type == 'application/json')
        self.assertEqual(resp_login.status_code, 200)
        # valid token logout
        response = self.client.post(
            '/auth/logout',
            headers=dict(
                Authorization='Bearer ' + json.loads(
                    resp_login.data.decode()
                )['auth_token']
            )
        )
        data = json.loads(response.data.decode())
        self.assertTrue(data['status'] == 'success')
        self.assertTrue(data['message'] == 'Successfully logged out.')
        self.assertEqual(response.status_code, 200)
```

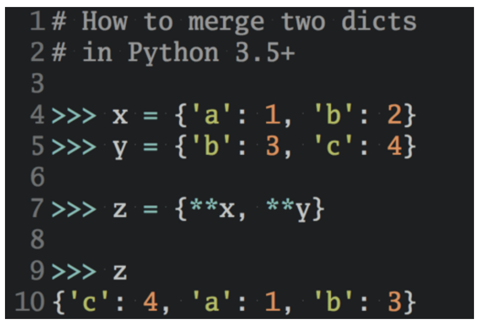
Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

```
def test_invalid_logout(self):
    """ Testing logout after the tok
    with self.client:
        # user registration
        resp_register = self.client.
            '/auth/register',
            data=json.dumps(dict(
                email='joe@gmail.com
                password='123456'
            )),
            content_type='application/json',
        )
        data_register = json.loads(resp_register.data.decode())
        self.assertTrue(data_register['status'] == 'success')
        self.assertTrue(
            data_register['message'] == 'Successfully registered.')
        self.assertTrue(data_register['auth_token'])
        self.assertTrue(resp_register.content_type == 'application/json')
        self.assertEqual(resp_register.status_code, 201)
        # user login
        resp_login = self.client.post(
            '/auth/login',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            )),
            content_type='application/json'
        )
        data_login = json.loads(resp_login.data.decode())
        self.assertTrue(data_login['status'] == 'success')
        self.assertTrue(data_login['message'] == 'Successfully logged in.')
        self.assertTrue(data_login['auth_token'])
        self.assertTrue(resp_login.content_type == 'application/json')
        self.assertEqual(resp_login.status_code, 200)
        # invalid token logout
        time.sleep(6)
        response = self.client.post(
            '/auth/logout',
            headers=dict(
                Authorization='Bearer ' + json.loads(
                    resp_login.data.decode()
                )['auth_token']
            )
        )
        data = json.loads(response.data.decode())
        self.assertTrue(data['status'] == 'fail')
        self.assertTrue(
            data['message'] == 'Signature expired. Please log in again.')
        self.assertEqual(response.status_code, 401)
```

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

Blacklist

Add the following code to *project/server,*

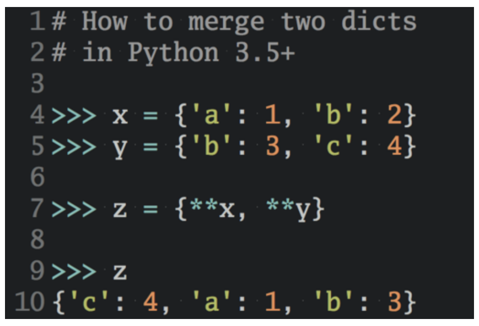
Python

```
class BlacklistToken(db.Model):
    """
    Token Model for storing JWT toke
    """
    __tablename__ = 'blacklist_toker

    id = db.Column(db.Integer, prima
    token = db.Column(db.String(500), unique=True, nullable=False)
    blacklisted_on = db.Column(db.DateTime, nullable=False)

    def __init__(self, token):
        self.token = token
        self.blacklisted_on = datetime.datetime.now()

    def __repr__(self):
        return '<id: token: {}'.format(self.token)
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Then create and apply the migrations. Once done, your database should have the following tables:

SQL

```
Schema |          Name           |   Type   |  Owner
--------+-------------------------+----------+----------
public | alembic_version         | table    | postgres
public | blacklist_tokens        | table    | postgres
public | blacklist_tokens_id_seq | sequence | postgres
public | users                   | table    | postgres
public | users_id_seq            | sequence | postgres
(5 rows)
```

With that, we can add the logout handler…

Logout Route Handler

Update the views:

Python

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

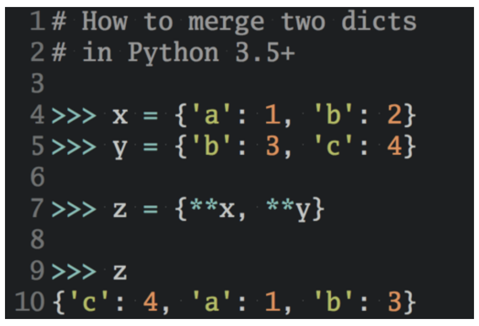
Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
class LogoutAPI(MethodView):
    """
    Logout Resource
    """
    def post(self):
        # get auth token
        auth_header = request.header
        if auth_header:
            auth_token = auth_header
        else:
            auth_token = ''
        if auth_token:
            resp = User.decode_auth_
            if not isinstance(resp,
                # mark the token as
                blacklist_token = Bl
                try:
                    # insert the token
                    db.session.add(blacklist_token)
                    db.session.commit()
                    responseObject = {
                        'status': 'success',
                        'message': 'Successfully logged out.'
                    }
                    return make_response(jsonify(responseObject)), 200
                except Exception as e:
                    responseObject = {
                        'status': 'fail',
                        'message': e
                    }
                    return make_response(jsonify(responseObject)), 200
            else:
                responseObject = {
                    'status': 'fail',
                    'message': resp
                }
                return make_response(jsonify(responseObject)), 401
        else:
            responseObject = {
                'status': 'fail',
                'message': 'Provide a valid auth token.'
            }
            return make_response(jsonify(responseObject)), 403


# define the API resources
registration_view = RegisterAPI.as_view('register_api')
login_view = LoginAPI.as_view('login_api')
user_view = UserAPI.as_view('user_api')
logout_view = LogoutAPI.as_view('logout_api')

# add Rules for API Endpoints
auth_blueprint.add_url_rule(
    '/auth/register',
    view_func=registration_view,
    methods=['POST']
)
auth_blueprint.add_url_rule(
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

Python

```python
from project.server.models import Us
```

When a users logs out, the token is no lo

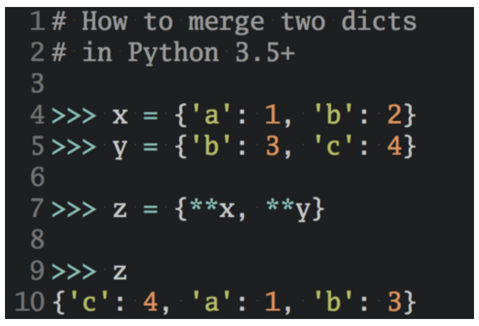
> **NOTE:** Often, larger applications hav
> does not run out of valid tokens.

Run the tests:

Shell

```
Ran 12 tests in 6.418s

OK
```



```python
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

Remove ads

Refactoring

Finally, we need to ensure that a token has not been blacklisted, right after the token has been decoded - `decode_auth_token()` - within the logout and user status routes.

First, let's write a test for the logout route:

Python

Ads help us run this site

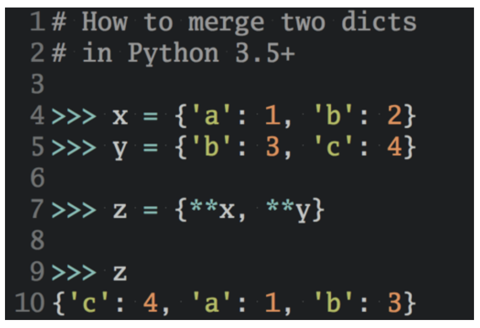
When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
                password='123456'
            )),
            content_type='applicatic
        )
        data_register = json.loads(r
        self.assertTrue(data_registe
        self.assertTrue(
            data_register['message']
        self.assertTrue(data_registe
        self.assertTrue(resp_registe
        self.assertEqual(resp_regist
        # user login
        resp_login = self.client.pos
            '/auth/login',
            data=json.dumps(dict(
                email='joe@gmail.com',
                password='123456'
            )),
            content_type='application/json'
        )
        data_login = json.loads(resp_login.data.decode())
        self.assertTrue(data_login['status'] == 'success')
        self.assertTrue(data_login['message'] == 'Successfully logged in.')
        self.assertTrue(data_login['auth_token'])
        self.assertTrue(resp_login.content_type == 'application/json')
        self.assertEqual(resp_login.status_code, 200)
        # blacklist a valid token
        blacklist_token = BlacklistToken(
            token=json.loads(resp_login.data.decode())['auth_token'])
        db.session.add(blacklist_token)
        db.session.commit()
        # blacklisted valid token logout
        response = self.client.post(
            '/auth/logout',
            headers=dict(
                Authorization='Bearer ' + json.loads(
                    resp_login.data.decode()
                )['auth_token']
            )
        )
        data = json.loads(response.data.decode())
        self.assertTrue(data['status'] == 'fail')
        self.assertTrue(data['message'] == 'Token blacklisted. Please log in again.')
        self.assertEqual(response.status_code, 401)
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

In this test, we blacklist the token just before the logout route gets hit which makes our valid token unusable.

Update the imports:

Python

```python
from project.server.models import User, BlacklistToken
```

The test should fail with the following exception:

Ads help us run this site

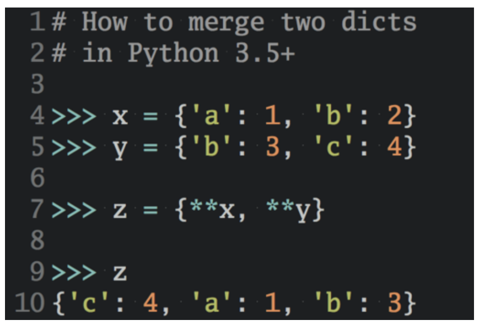
When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
    """
    try:
        payload = jwt.decode(auth_tc
        is_blacklisted_token = Black
        if is_blacklisted_token:
            return 'Token blackliste
        else:
            return payload['sub']
    except jwt.ExpiredSignatureError
        return 'Signature expired. F
    except jwt.InvalidTokenError:
        return 'Invalid token. Pleas
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Finally, add the `check_blacklist()` funct

Python

```python
@staticmethod
def check_blacklist(auth_token):
    # check whether auth token has been blacklisted
    res = BlacklistToken.query.filter_by(token=str(auth_token)).first()
    if res:
        return True
    else:
        return False
```

Before you run the test, update `test_decode_auth_token` to convert the bytes object to a string:

Python

```python
def test_decode_auth_token(self):
    user = User(
        email='test@test.com',
        password='test'
    )
    db.session.add(user)
    db.session.commit()
    auth_token = user.encode_auth_token(user.id)
    self.assertTrue(isinstance(auth_token, bytes))
    self.assertTrue(User.decode_auth_token(
        auth_token.decode("utf-8") ) == 1)
```

Run the tests:

Shell

```
Ran 13 tests in 9.557s

OK
```

In a similar fashion, add one more test for the user status route.

Python

Ads help us run this site

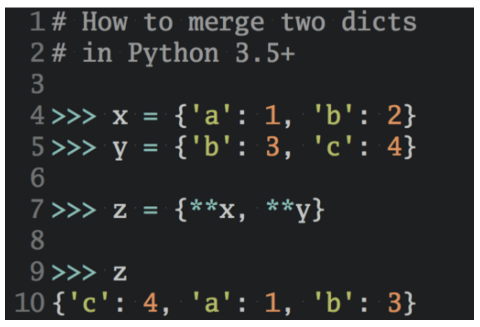
When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
        db.session.add(blacklist_token)
        db.session.commit()
        response = self.client.get(
            '/auth/status',
            headers=dict(
                Authorization='Beare
                    resp_register.da
                )['auth_token']
            )
        )
        data = json.loads(response.c
        self.assertTrue(data['status
        self.assertTrue(data['messag
        self.assertEqual(response.st
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

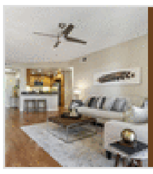
Similar to the last test, we blacklisted the token before the user status route gets hit.

Run the tests for one final time:

Shell

```
Ran 14 tests in 10.206s

OK
```



Communication Hill Charmer
Asking Price $729,888

ADWERX

COMPASS

DRE #01915526 and 1925793
Sandy & Joe 408-394-5312

Remove ads

Code Smell

Finally, take a look at *test_auth.py*. Notice the duplicate code? For example:

Python

```
self.client.post(
    '/auth/register',
    data=json.dumps(dict(
        email='joe@gmail.com',
        password='123456'
    )),
    content_type='application/json',
)
```

There are eight occurrences of this. To fix, add the following helper at the top of the file:

Python

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

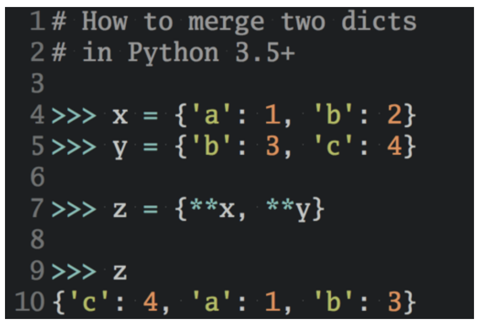
Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

How about logging in a user? Refactor it on your own. What else can you refactor? Comment below.

Refactor

For the PyBites Challenge, let's refactor
following test to *test_auth.py*:



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Python

```
def test_user_status_malformed_beare
    """ Test for user status with ma
    with self.client:
        resp_register = register_use
        response = self.client.get(
            '/auth/status',
            headers=dict(
                Authorization='Bearer' + json.loads(
                    resp_register.data.decode()
                )['auth_token']
            )
        )
        data = json.loads(response.data.decode())
        self.assertTrue(data['status'] == 'fail')
        self.assertTrue(data['message'] == 'Bearer token malformed.')
        self.assertEqual(response.status_code, 401)
```

Essentially, an error is thrown if the `Authorization` header is formatted incorrectly - e.g., no space between `Bearer` and the token value. Run the tests to ensure they fail, and then update the `UserAPI` class in *project/server/auth/views.py*:

Python

```
class UserAPI(MethodView):
    """
    User Resource
```

Ads help us run this site

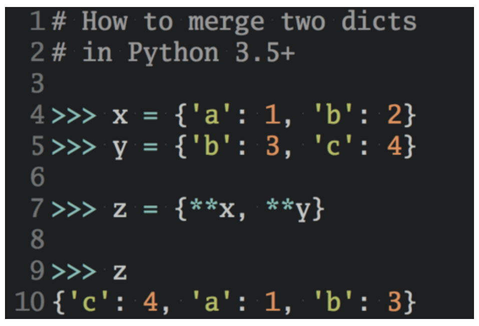
When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

```
                user = User.query.filter_by(id=resp).first()
                responseObject = {
                    'status': 'succe
                    'data': {
                        'user_id': u
                        'email': use
                        'admin': use
                        'registered_
                    }
                }
                return make_response
            responseObject = {
                'status': 'fail',
                'message': resp
            }
            return make_response(jsc
        else:
            responseObject = {
                'status': 'fail',
                'message': 'Provide a valid auth token.'
            }
            return make_response(jsonify(responseObject)), 401
```



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Run the tests one final time.

Conclusion

In this tutorial, we went through the process of adding authentication to a Flask app with JSON Web Tokens. Turn back to the objectives from the beginning of this tutorial. Can you put each one into action? What did you learn?

What's next? How about the client-side. Check out Token-Based Authentication With Angular for adding Angular into the mix.

To see how to build a complete web app from scratch using Flask, check out our video series:

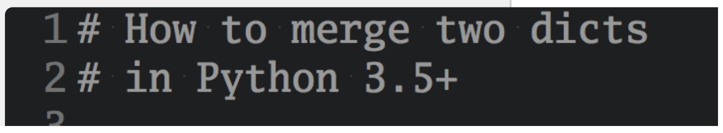
Free Bonus: Click here to get access to a free Flask + Python video tutorial that shows you how to build Flask web app, step-by-step.

Feel free to share your comments, questions, or tips in the comments below. The full code can be found in the flask-jwt-auth repository.

Cheers!

Python Tricks

Get a short & sweet **Python Trick** delivered to your inbox every couple of days. No spam ever. Unsubscribe any time. Curated by the Real Python



```
# How to merge two dicts
# in Python 3.5+
```

Ads help us run this site

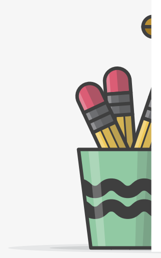
When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

Maste
With Un



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

Email Address

Send Python Tricks »

Join us and get access to hundreds of tutorials, hands-on video courses, and a community of expert Pythonistas:

Level Up Your Python Skills »

What Do You Think?

Tweet Share Email

Real Python Comment Policy: The most useful comments are those written with the goal of learning from or helping out other readers—after reading the whole article and all the earlier comments. Complaints and insults generally won't make the cut here.

What's your #1 takeaway or favorite thing you learned? How are you going to put your newfound skills to use? Leave a comment below and let us know.

Featured Comment

wrinkledcheese • 2 years ago
I've been picking at this tutorial from time to time, on a daily basis. Half-

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application

configured alembic.ini created in the root of the repo, as opposed to in the alembic directory as is provided

After this I was able to follow th tutorial is old, and there's proba plan on integrating Google Autl

1 ^ | ˅ • Share ›

Improve Your Python

...with a fresh Python Trick code snippet every couple of days:

```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Email Address

Send Python Tricks »

37 Comments Real Python Disc

Recommend 8 Tweet Shar

Join the discussion…

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Ingmar • 3 years ago

I thought the main advantage of JWT was being able to validate them without accessing the database (for various reasons). Doesn't adding a database driven blacklist check completely invalidate any advantages JWT has?

4 ^ | ˅ • Reply • Share ›

Keep Learning

Related Tutorial Categories: advanced flask web-dev

— FREE Email Series —

Python Tricks

```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

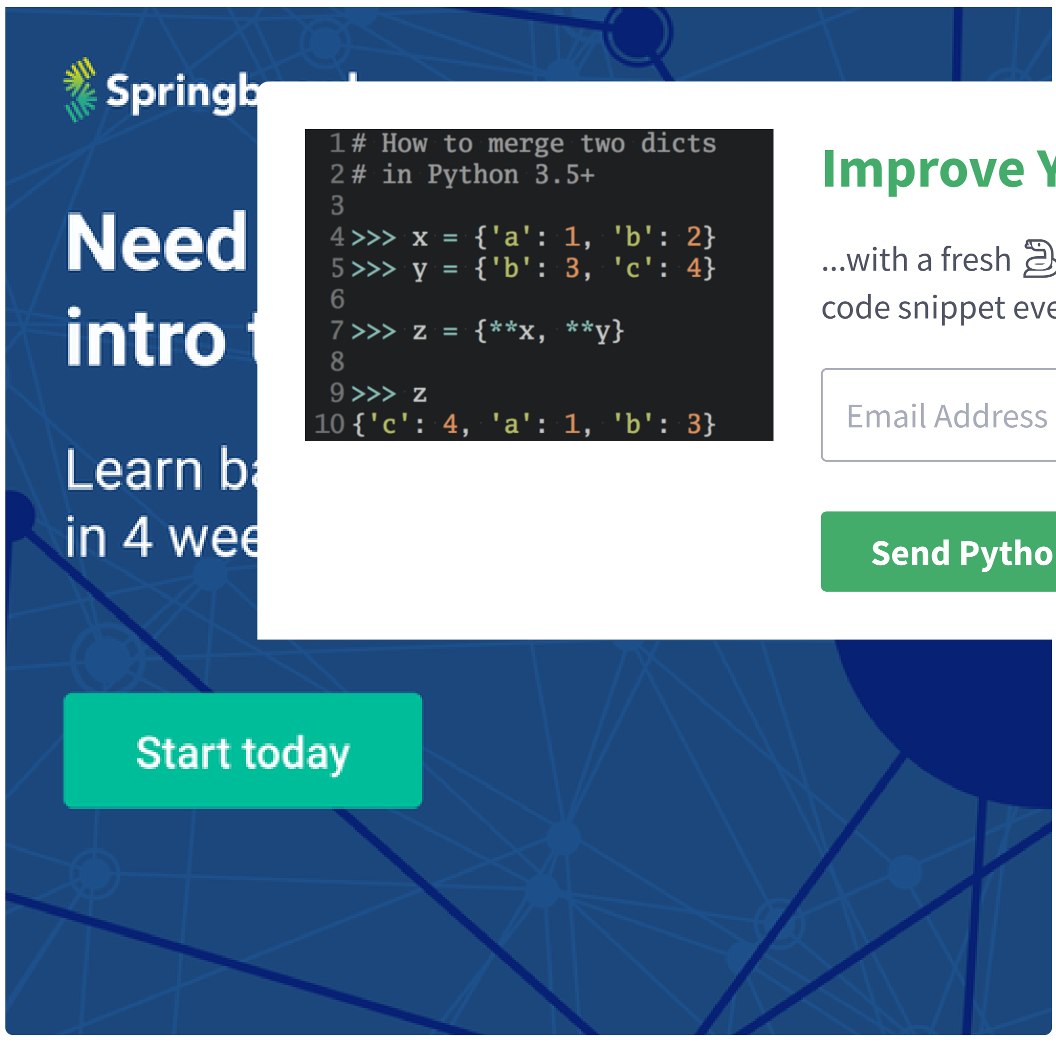
Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site

Information that may be used:

- Type of browser and its settings
- Information about the device's operating system
- Cookie information
- Information about other identifiers assigned to the device
- The IP address from which the device accesses a client's website or mobile application
- Information about the user's activity on that device, including web pages and mobile apps visited or used
- Information about the geographic location of the device when it accesses a website or mobile application



```
# How to merge two dicts
# in Python 3.5+

>>> x = {'a': 1, 'b': 2}
>>> y = {'b': 3, 'c': 4}

>>> z = {**x, **y}

>>> z
{'c': 4, 'a': 1, 'b': 3}
```

Improve Your Python

...with a fresh **Python Trick** code snippet every couple of days:

Email Address

Send Python Tricks »

Table of Contents

- Objectives
- Introduction
- → Getting Started
- Sanity Check
- JWT Setup
- Route Setup
- Register Route
- Login Route
- User Status Route
- Logout Route Tests
- Blacklist
- Logout Route Handler
- Refactoring
- Code Smell
- Refactor
- Conclusion

Tweet Share Email

High Quality Python Video Courses

Watch Now »

Ads help us run this site

When you visit our site, pre-selected companies may access and use certain information on your device to serve relevant ads or personalized content.

› Information that may be used. › Purposes for storing information. Learn More Continue to site