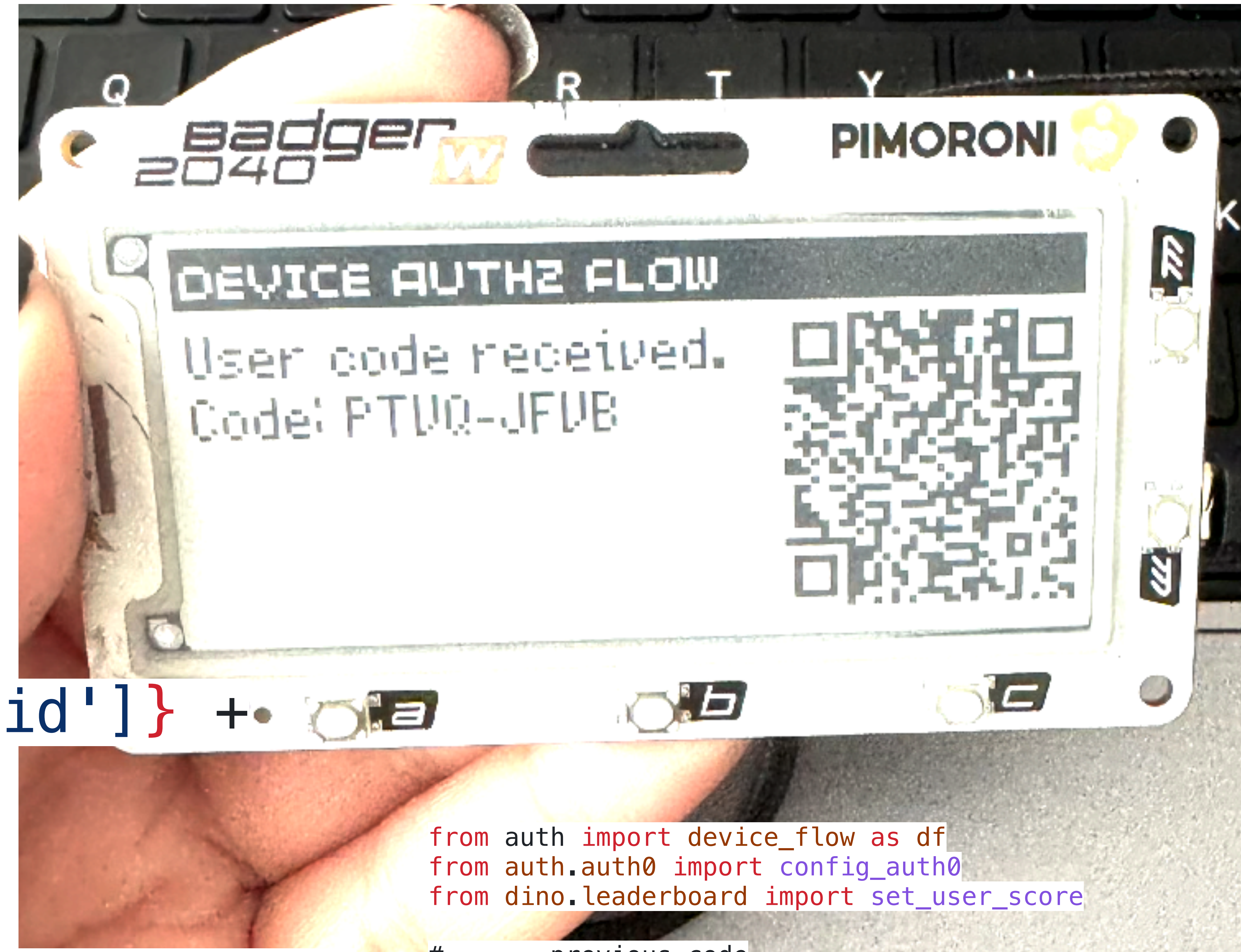
Device Flow

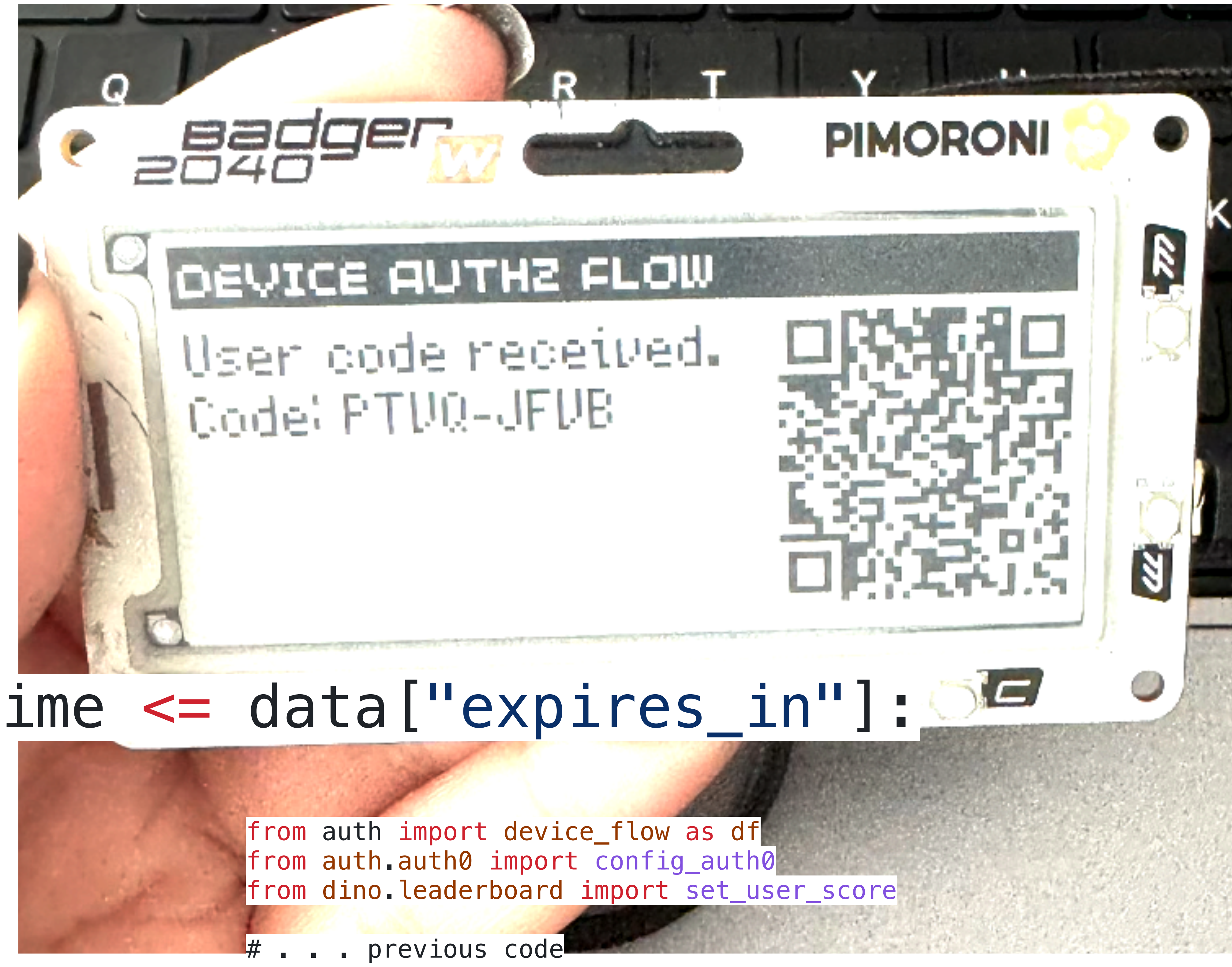
```python
def login(auth0_request_data):
    # . . .
    user_confirmation = False
    elapsed_time = 0
    grant_type = (
        'urn:ietf:params:oauth:' +
        'grant-type:device_code'
    )
    device_code_data = (
        f"device_code={data['device_code']} +
        f"&client_id={auth0_request_data['client_id']} +
        f"&grant_type={grant_type}"
    )
    headers = {
        "Content-Type": "application/x-www-form-urlencoded"
    }
    url = "https://{}/oauth/token".format(
        auth0_request_data['domain'])
    # . . .
```



```python
from auth import device_flow as df
from auth.auth0 import config_auth0
from dino.leaderboard import set_user_score

# . . . previous code
    elif display.pressed(BUTTON_B):
        print("Connecting to the internet...")
        if not display.isconnected():
            display.connect()
        clear_screen()

        print("Config Auth0")
        auth0_request_data = config_auth0()

        print("Requesting tokens")
        tokens = df.login(auth0_request_data)

        print("Setting user score with API...")
        set_user_score(tokens.get('access_token', 'or
```

Device Flow

```python
def login(auth0_request_data):
    # . . .
    while not user_confirmation:
        token_response = urequests.post(
            url,
            headers=headers,
            data=device_code_data
        )
        tokens = token_response.json()

        if 'error' in tokens.keys() and elapsed_time <= data["expires_in"]:
            elapsed_time += data["interval"]
            time.sleep(data["interval"])
        else:
            user_confirmation = True

    # Return tokens to application
    return tokens
```



```python
from auth import device_flow as df
from auth.auth0 import config_auth0
from dino.leaderboard import set_user_score

# . . . previous code
    elif display.pressed(BUTTON_B):
        print("Connecting to the internet...")
        if not display.isconnected():
            display.connect()
        clear_screen()

        print("Config Auth0")
        auth0_request_data = config_auth0()

        print("Requesting tokens")
        tokens = df.login(auth0_request_data)

        print("Setting user score with API...")
        set_user_score(tokens.get('access_token', 'error'), score
```