Device Authorization Flow 101

Logging in From Input Constrained Devices

Jess Temporal

she/her
DevRel @ Auth0 by Okta

jtemporal.com
/socials

Input Constrained Devices

Input Constrained Devices

Input Constrained Devices

- Smart TVs

Input Constrained Devices

- Smart TVs
- Smart Fridges

Input Constrained Devices

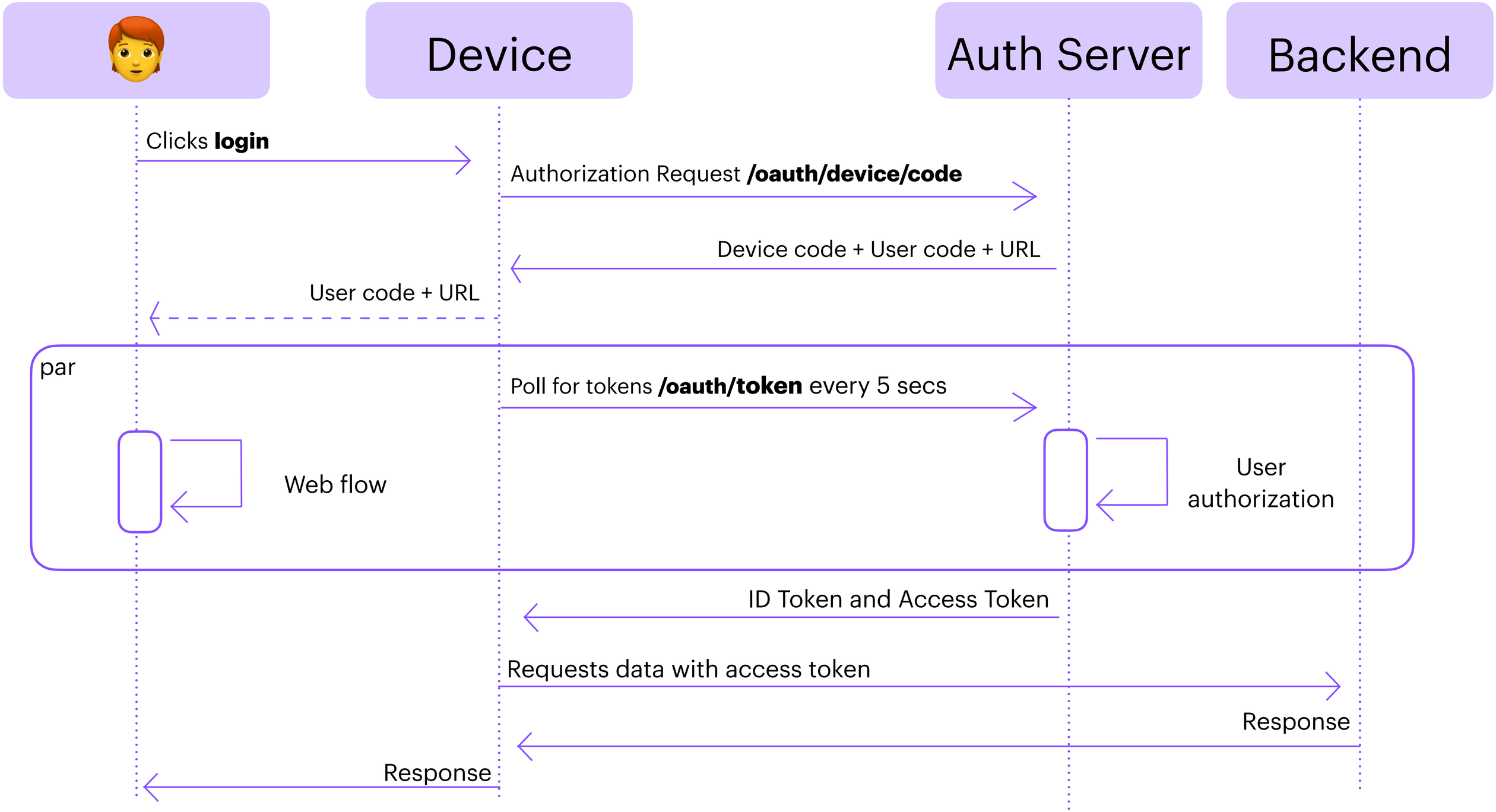
- Smart TVs
- Smart Fridges
- Handheld consoles

Virtual Keyboard

Virtual Keyboard

- not that great for typing
- touch screens?
- password managers?

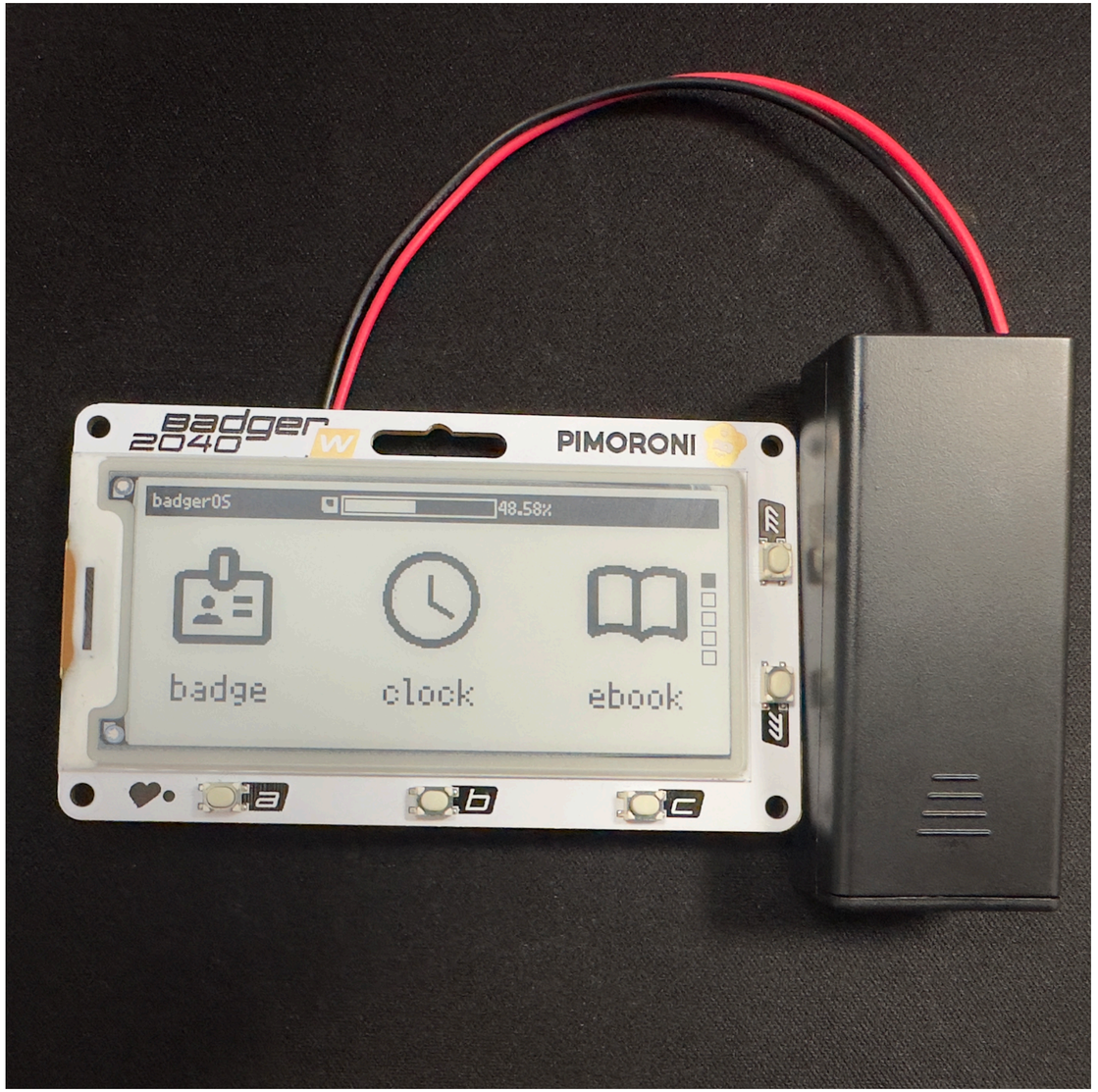
Device Authorization Flow


Device
Auth Server
Backend
Clicks **login**
Authorization Request **/oauth/device/code**
Device code + User code + URL
User code + URL
par
Poll for tokens **/oauth/token** every 5 secs
Web flow
User authorization
ID Token and Access Token
Requests data with access token
Response
Response

Building is committing to memory

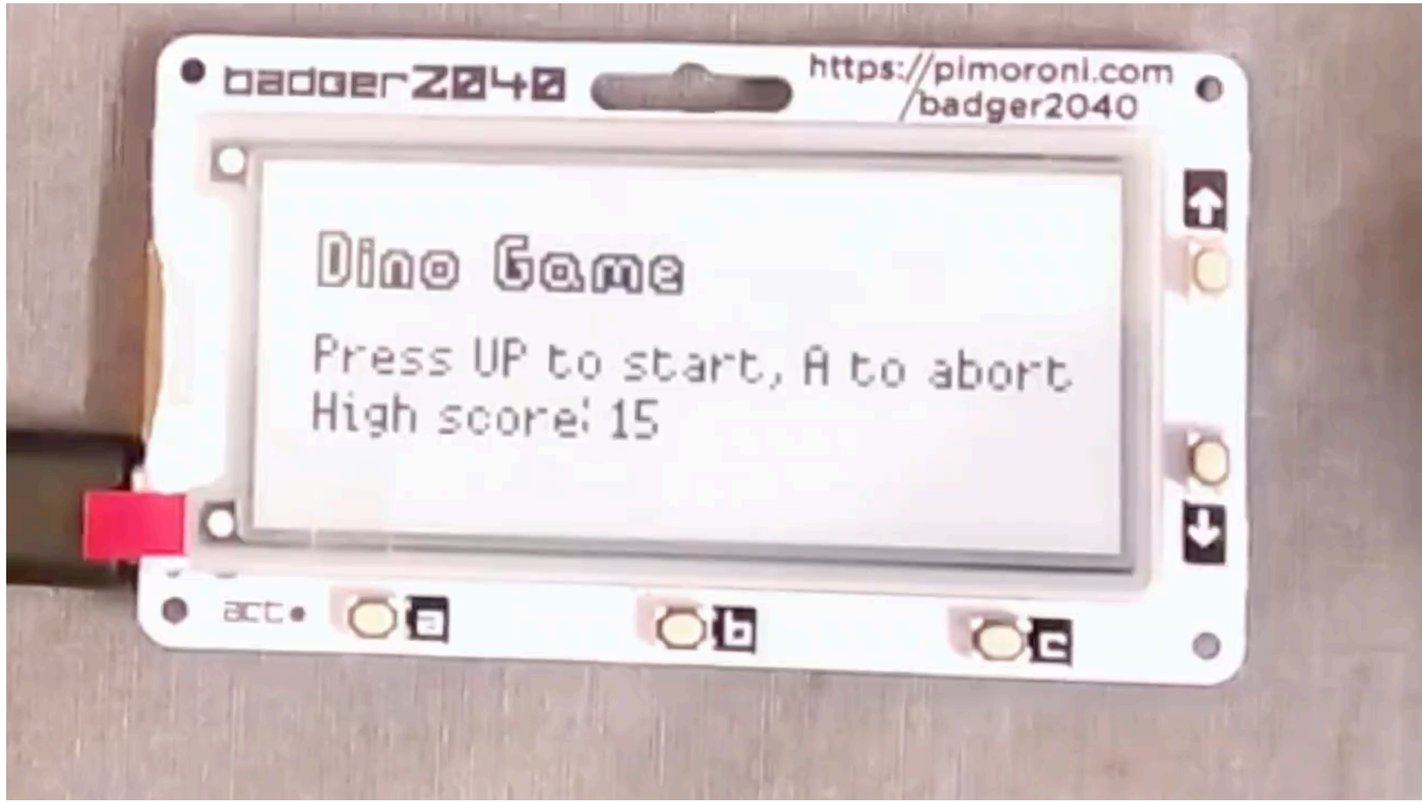
The device

- Badger 2040 W from Pimoroni
- Wifi capable
- e-ink display
- Micropython

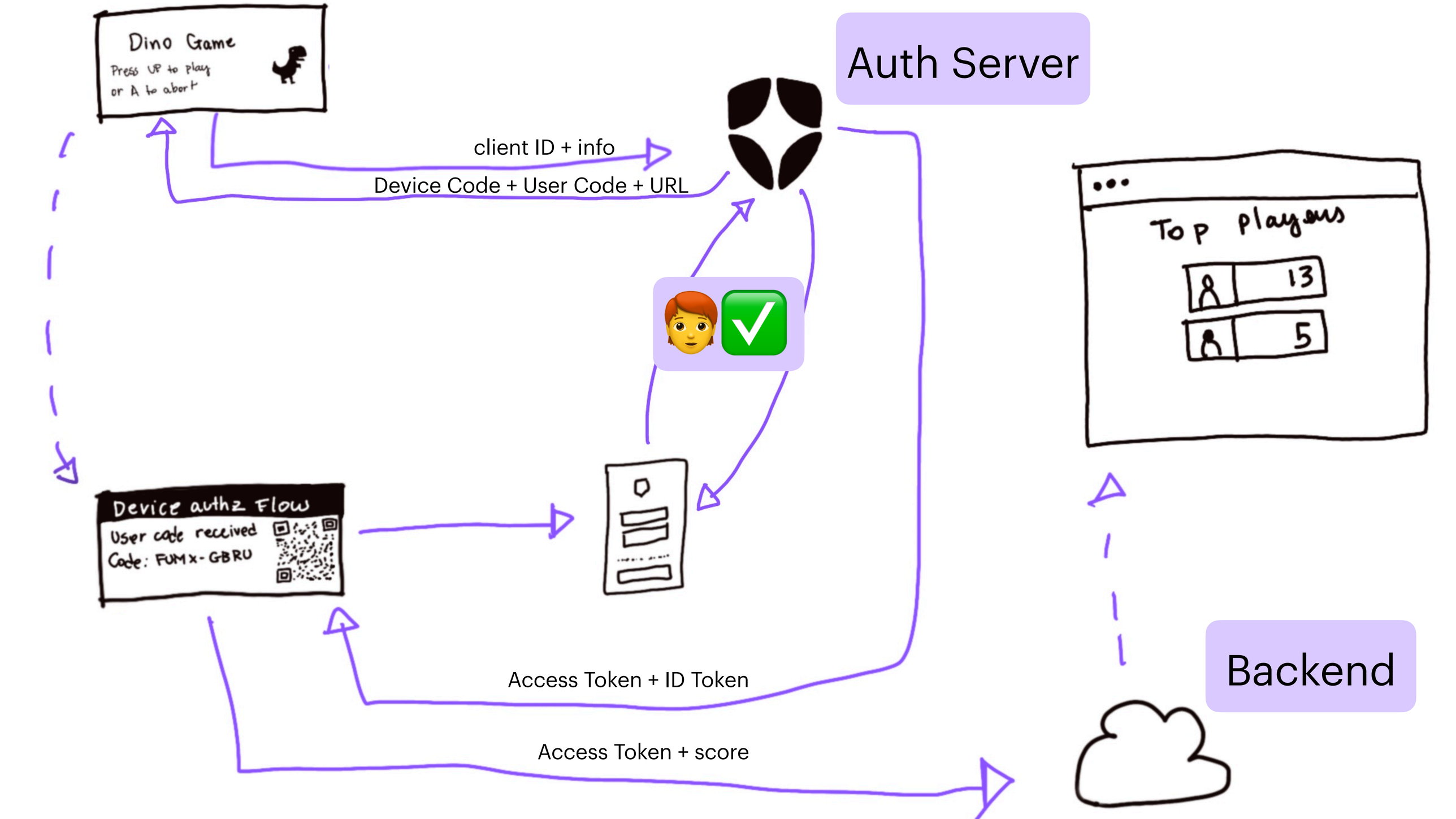


Dino Game

- Made for badger2040 with Micropython
- implements the game logic
- game ends when the Dino hits a obstacle



source


Dino Game
Press UP to play
or A to abort
Auth Server
client ID + info
Device Code + User Code + URL
Top players
13
5
Device authz Flow
User code received
Code: FUMX-GBRU
Access Token + ID Token
Access Token + score
Backend

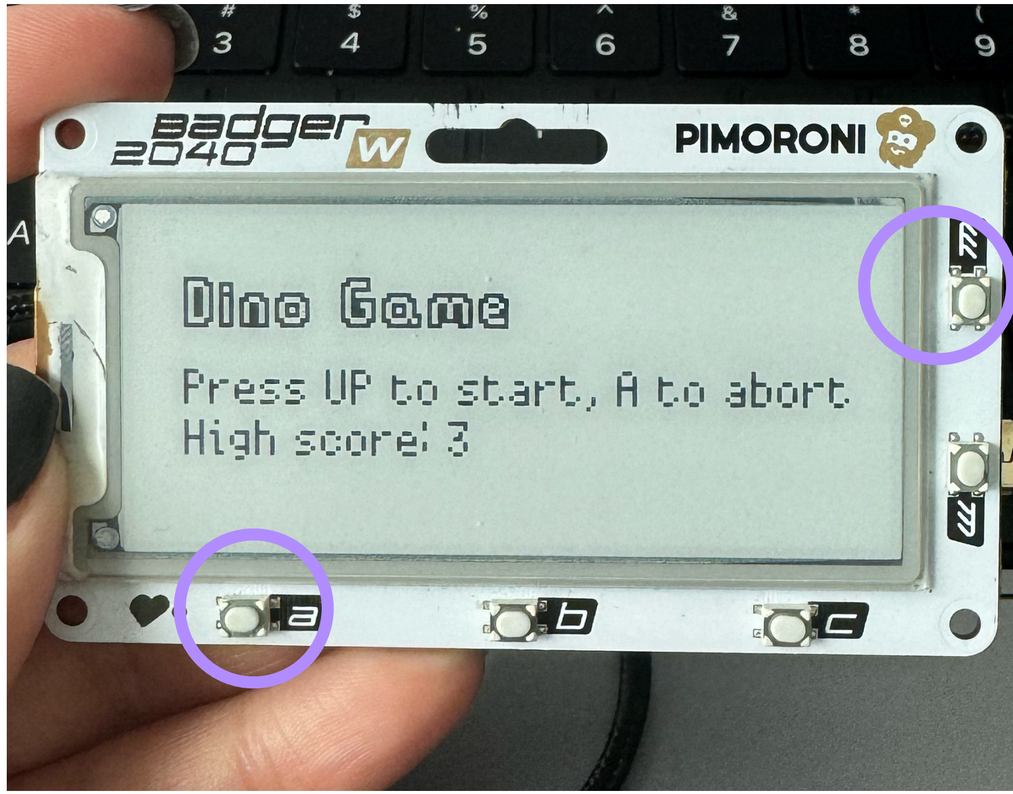
Dino Game

- Updated the game for badger 2040 w
- no changes to the game engine

source

Dino Game

```
def main():
    global high_score
    start_text()
    while True:
        if display.pressed(BUTTON_UP):
            game_loop()
            # . . .  saves score
            endgame_text()
        elif display.pressed(BUTTON_A):
            gc.collect()
            clear_screen()
            display.update()
            return
```



Dino Game

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

Dino Game

```
# . . . 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()
```

Dino Game

```
# . . . 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)
```

Dino Game

```
# . . . 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)
        start_text()
```

Dino Game

```
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)
        start_text()
```

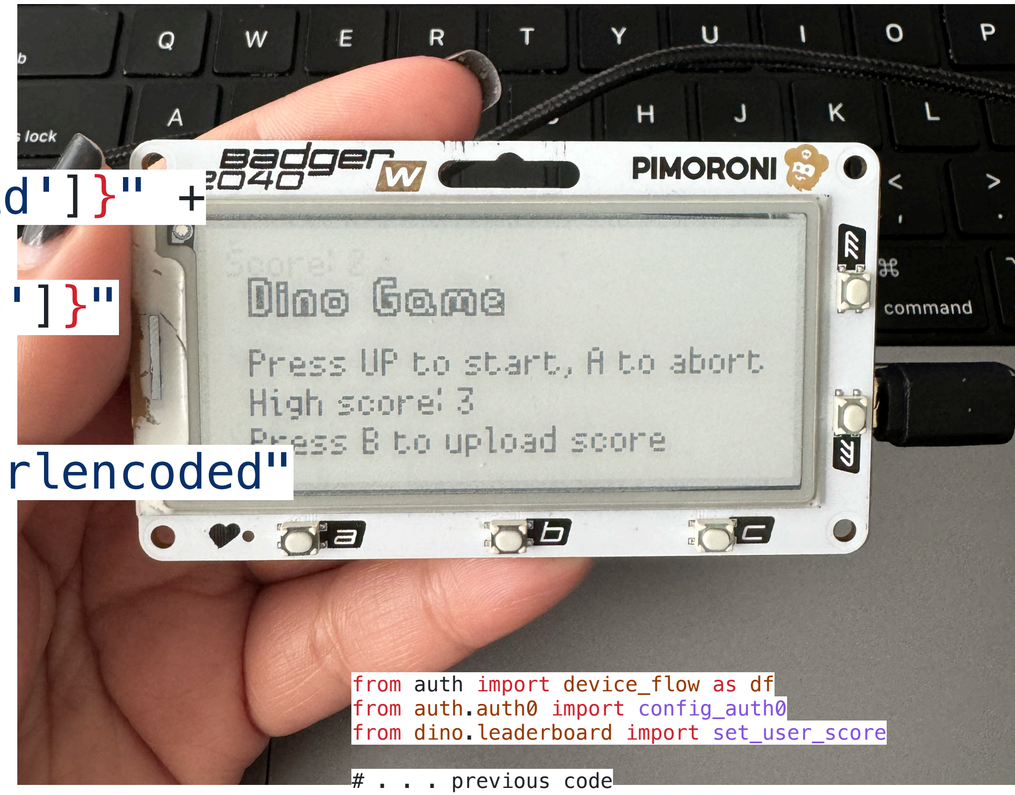
Device Flow

```
def config_auth0():
    auth0_config = open(AUTH0_CONFIG_PATH, "r")

    AUTH0_DOMAIN = auth0_config.readline()
    AUTH0_CLIENT_ID = auth0_config.readline()
    ALGORITHMS = auth0_config.readline()
    AUDIENCE = auth0_config.readline()

    auth0_request_data = {
        "domain": AUTH0_DOMAIN.strip('\n'),
        "client_id": AUTH0_CLIENT_ID.strip('\n'),
        "algorithms": [ALGORITHMS.strip('\n')],
        "audience": AUDIENCE.strip('\n')
    }

    auth0_config.close()
    return auth0_request_data
```


```
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...")
```

Device Flow

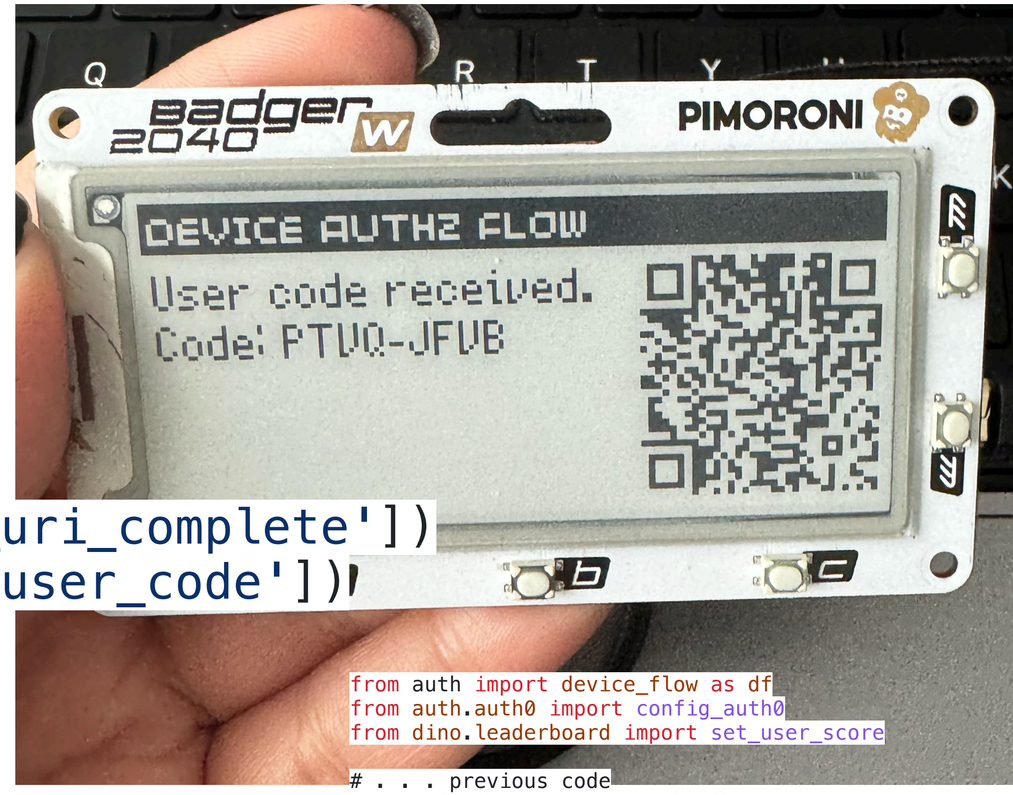
```python
def login(auth0_request_data):

    device_code_data = (
        f"client_id={auth0_request_data['client_id']}" +
        f"&scope=openid profile" +
        f"&audience={auth0_request_data['audience']}"
    )
    headers = {
        "Content-Type": "application/x-www-form-urlencoded"
    }
    url = "https://{}/oauth/device/code".format(
        auth0_request_data['domain'])
    device_code_response = urequests.post(
        url,
        headers=headers,
        data=device_code_data
    )
    # . . .
```




```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

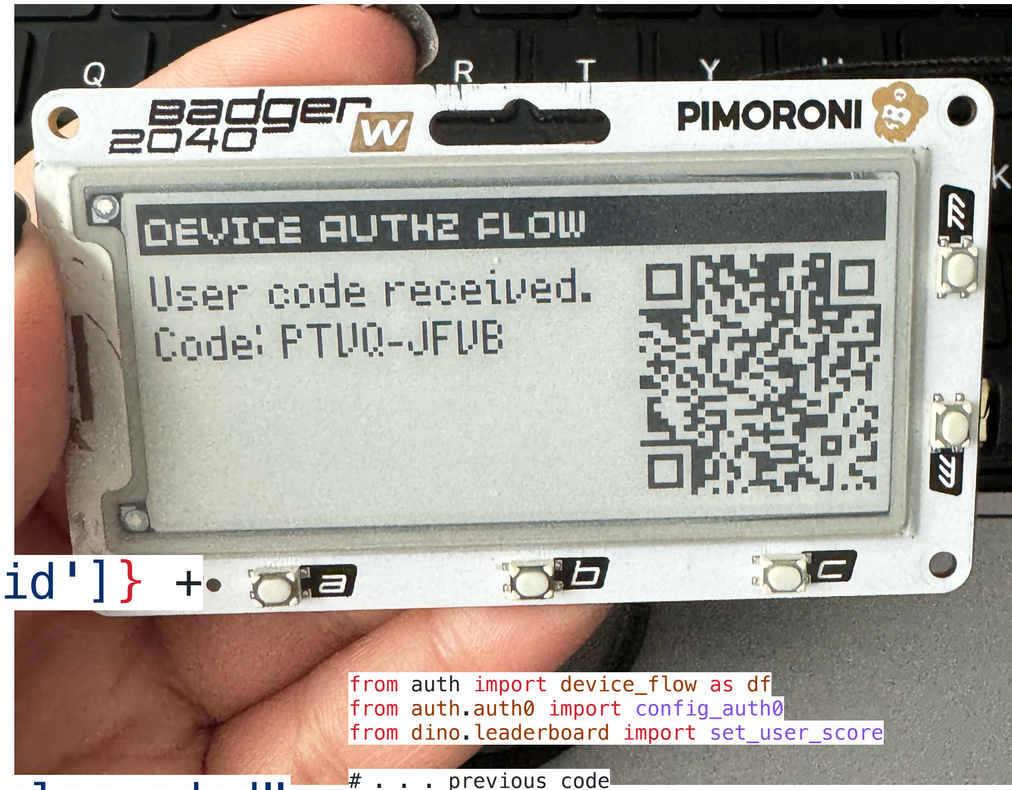
```python
def login(auth0_request_data):
    # . . .
    if device_code_response.status_code != 200:
        print('Error generating the device code')
        return

    print('Device code successful')
    data = device_code_response.json()

    print('1. Navigate to: ', data['verification_uri_complete'])
    print('2. Enter the following code: ', data['user_code'])
    draw_page(data)
    # . . .
```




```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...")
```

Device Flow

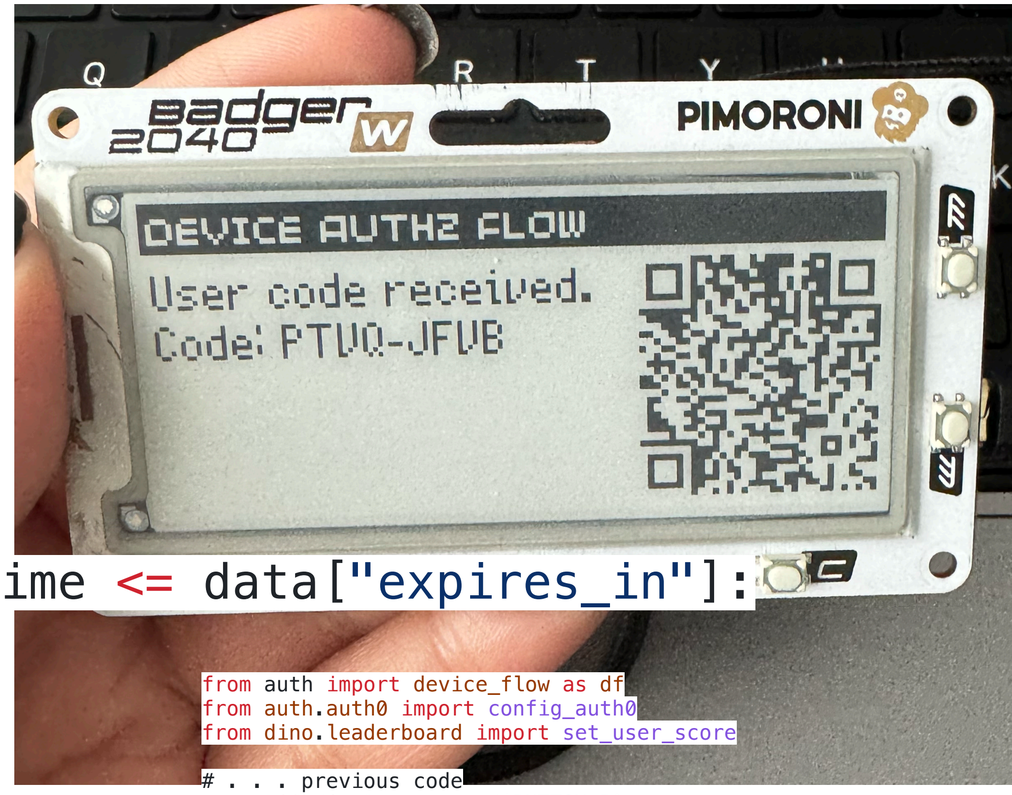
```python
def login(auth0_request_data):
    # . . .
    user_confirmation = False
    elapsed_time = 0
    grant_type = (
        'urn:ietf:params:oauth:' +
        'grant-type:device_code'
    )
    device_code_data = (
        f"device_code={data['device_code']} +
        f"&client_id={auth0_request_data['client_id']} +
        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...")
```

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
```



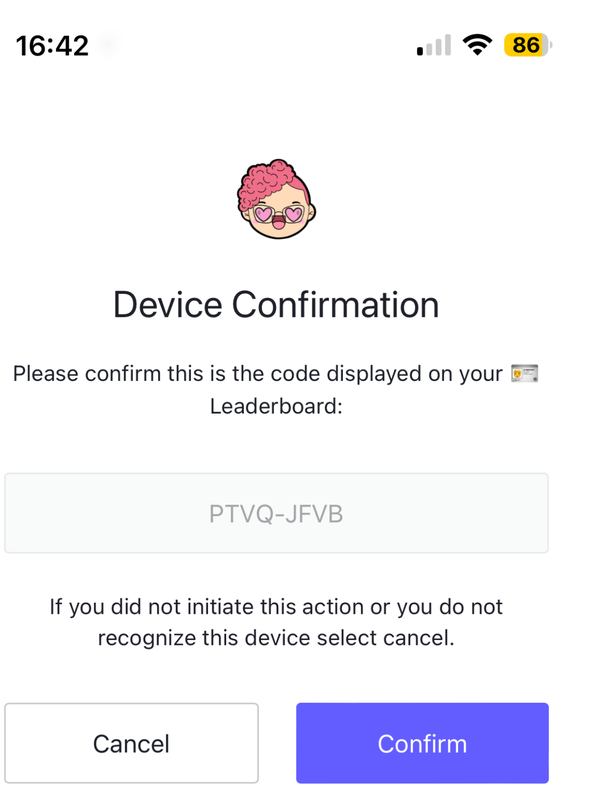

```python
from auth import device_flow as df
from auth.auth0 import config_auth0
from dino.leaderboard import set_user_score

# . . . 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...")
```

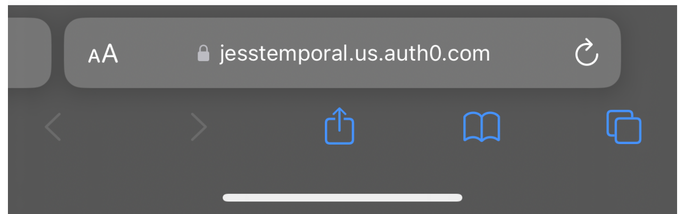
Web Flow

- User scans QR Code
- Note the User code being displayed

Web Flow

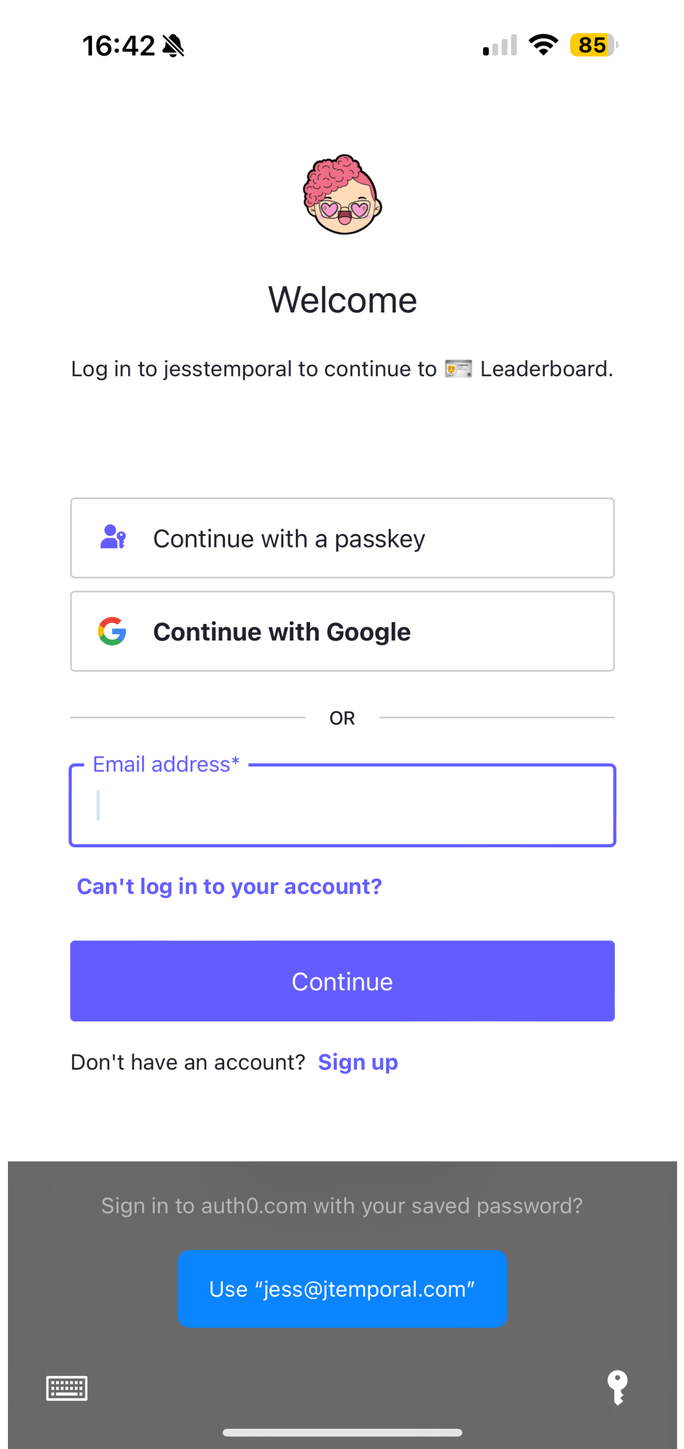
- The QR Code pulls up the confirmation page hosted on Auth0





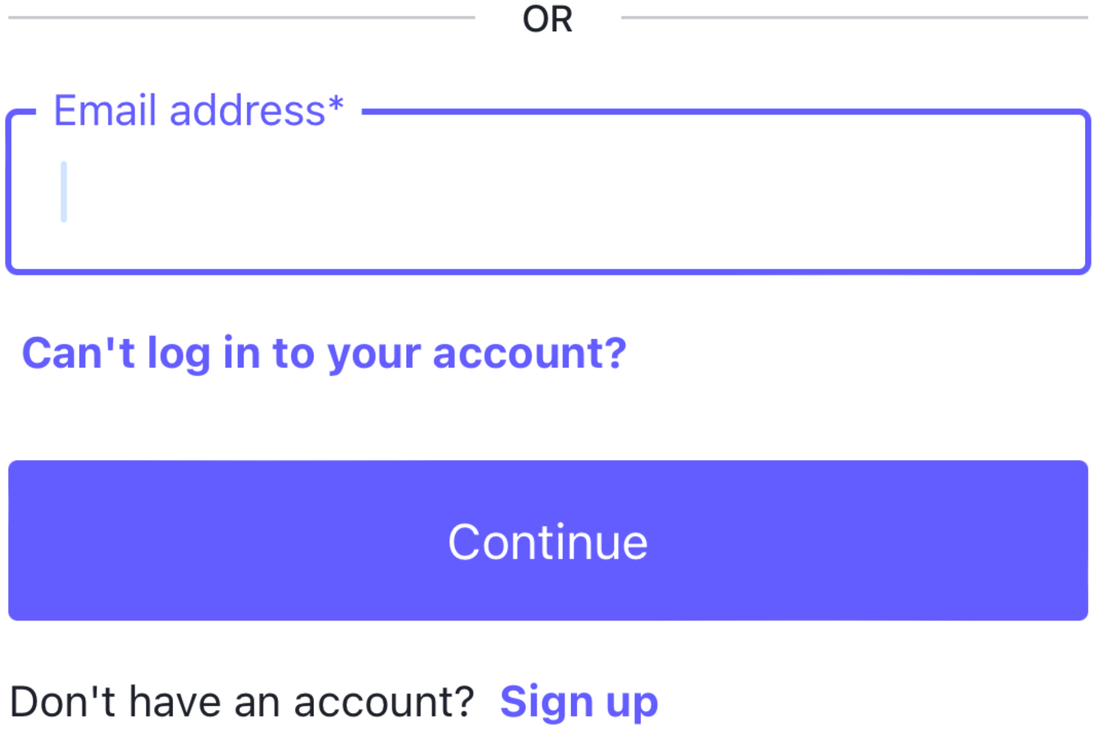
Web Flow

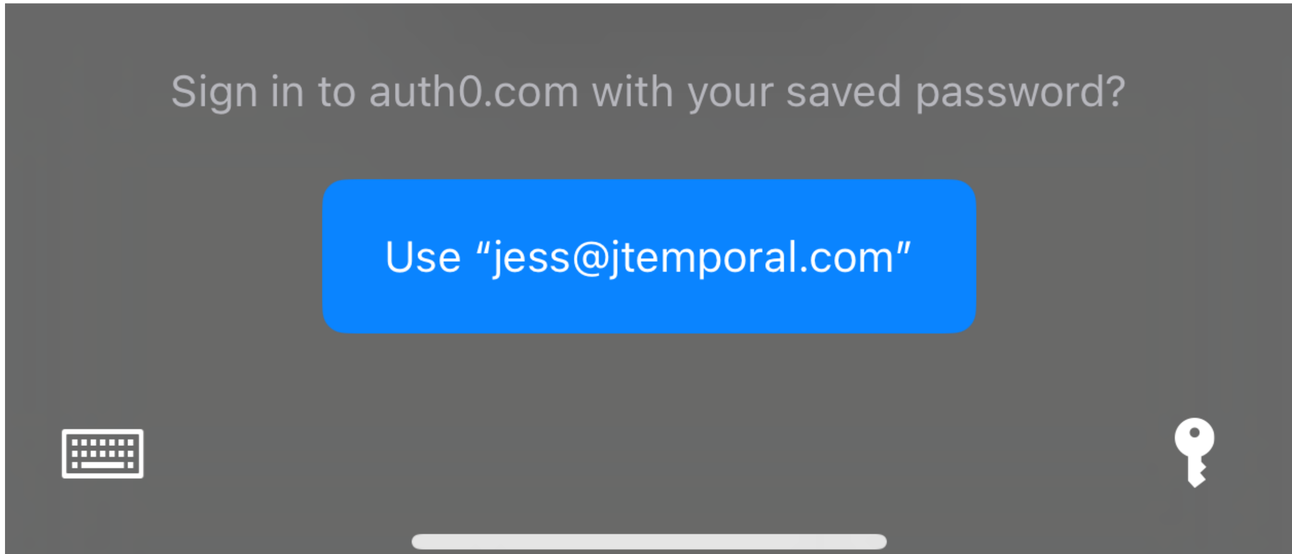
- Then the login/sign up happens



Web Flow

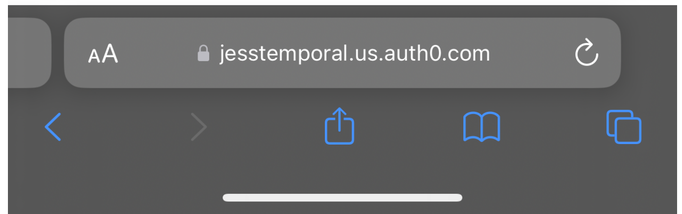
- Then the login/sign up happens
- Password manager prompt





Web Flow

- Feedback on everything going well
- Under the hood: Auth0 sends the ID token and Access token to the device



External API Call

```
def set_user_score(access_token, score):

    headers = {
        "Authorization": f"Bearer {access_token}"
    }
    url = (
        f"https://{api_url}/scores"+
        f"?score={str(score)}"
    )

    response = urequests.post(
        url,
        headers=headers
    )
    print("Score set with API")
```

Leaderboard

```python
@app.post("/scores", dependencies=[Depends(bearer)])
async def set_score(request: Request, score: int):

    data = database()
    score = int(request.query_params['score'])

    auth_token = request.headers.get('Authorization').split()[1]
    payload = JsonWebToken(auth_token).validate()

    if not data.get(payload['sub']):  # new user
        data[payload['sub']] = [score,]
    else:  # pre-existing user
        data[payload['sub']].append(score)

    with open('database.json', 'w') as f:
        f.write(json.dumps(data))

    return {"status": "ok", "message": "Data written successfully."}
```

Leaderboard

```
@app.get("/", include_in_schema=False)
async def top_players(request: Request):
    data = database()

    if data.__len__() < 10:
        top_players = sorted(
            data, key=lambda x: max(data[x]),
            reverse=True)[:data.__len__()]
    else:
        top_players = sorted(
            data, key=lambda x: max(data[x]),
            reverse=True)[:10]

    if top_players == []:
        return templates.TemplateResponse(
            "leaderboard.html",
            {"request": request, "top_players": {}}
        )
    # . . .
```

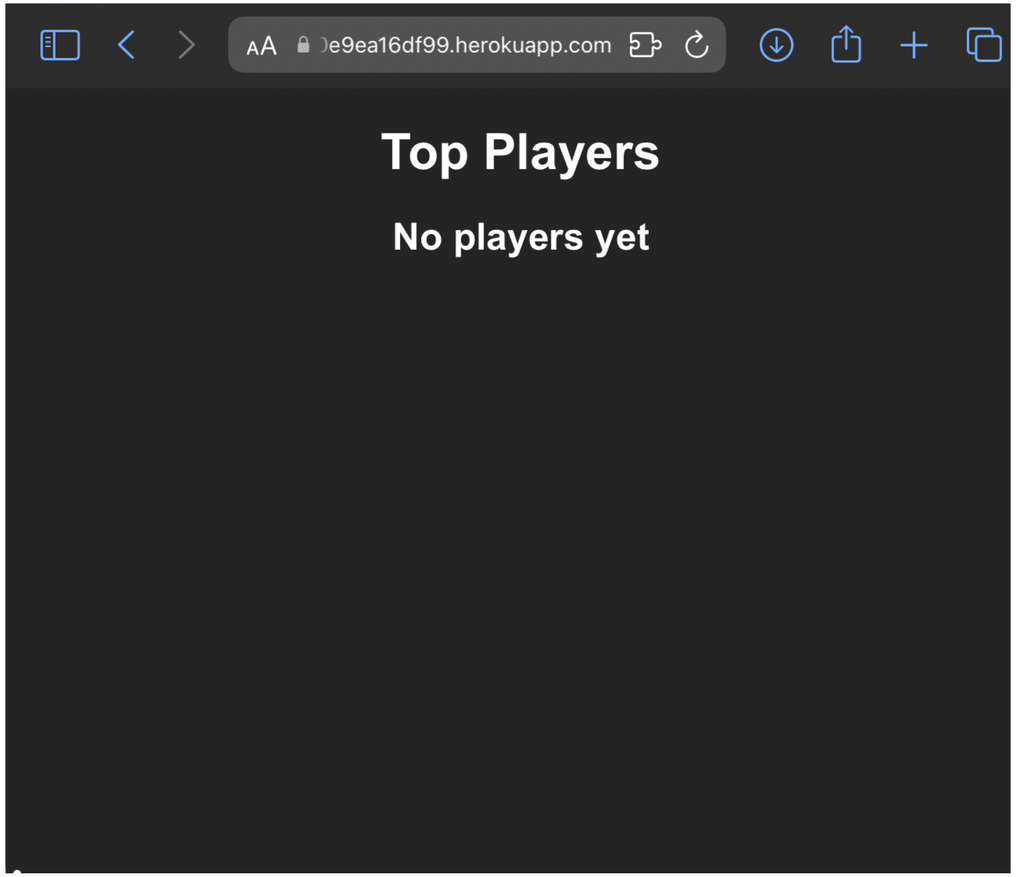
Leaderboard

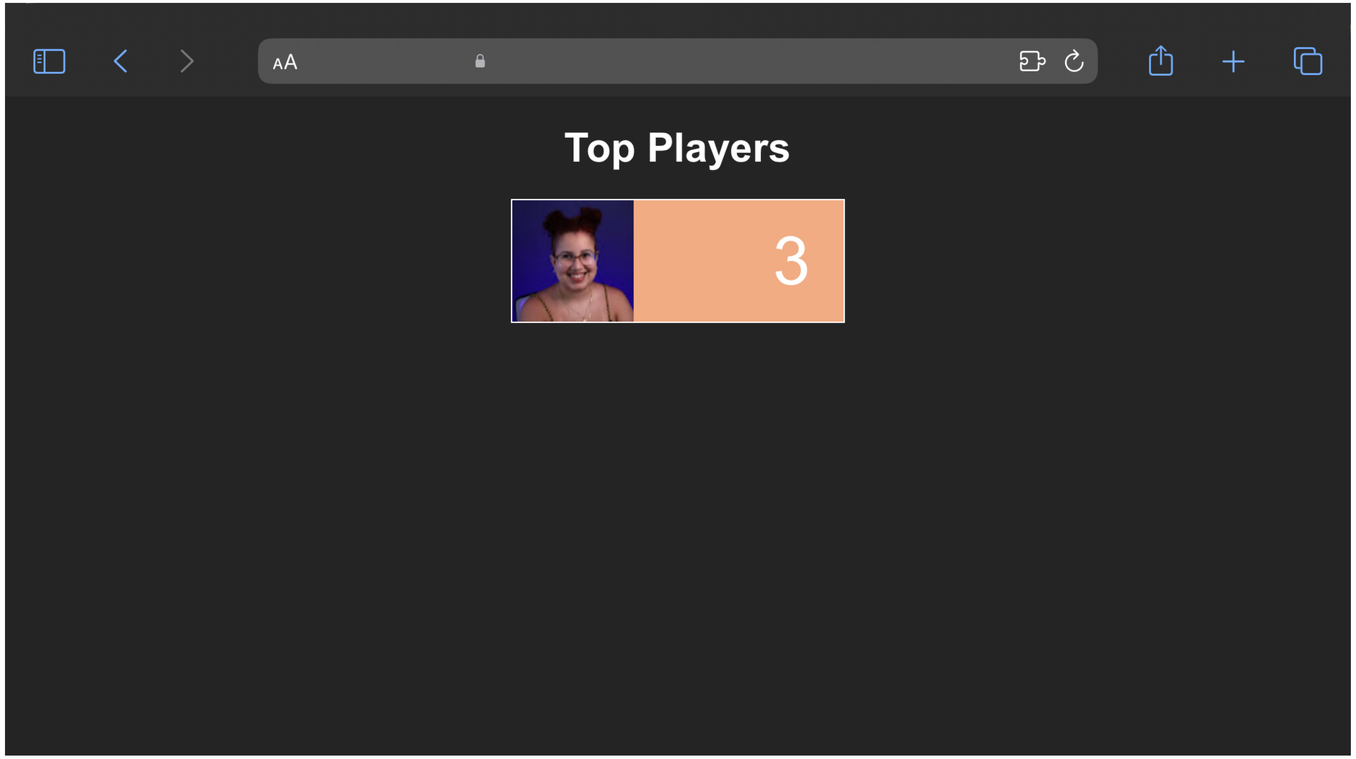
```python
@app.get("/", include_in_schema=False)
async def top_players(request: Request):

    # . . .
    # querying Auth0 for user data of top 10 players
    query = " OR ".join(top_players)
    users = auth0.users.list(
        per_page=10, include_fields=True,
        search_engine='v3', q=f'user_id:{query}'
    )

    # orders players by their highest score
    data = {k: max(data[k]) for k in top_players}

    # . . .
```

Leaderboard

```
@app.get("/", include_in_schema=False)
async def top_players(request: Request):

    # . . .
    # prepares data for rendering
    for user in users["users"]:
        score = data[user["user_id"]]
        data[user["user_id"]] = {
            "score": score,
            "photo": user["picture"]
        }

    return templates.TemplateResponse(
        "leaderboard.html",
        {"request": request, "top_players": data})
```

Leaderboard





Demo?


badger 2040 W
PIMORONI
Dino Game
Press UP to start, A to abort
High score: 0
Press B to upload score
a
b
c

Takeaways

Takeaways

- Device Authorization Flow can provide a better experience for users
- It is 2 flows put together: Device Flow + Web Flow
- Login does not need to be painful if you don’t have a keyboard
- You can also use Device Authorization Flow for other things like CLIs
- Try implementing the concepts at least once

Questions?

jtemporal.com/pyohio24


dev_day