WiFi data usage document

Introduction

This document describes the usage of iO Digital's proprietary employee Wi-Fi data. How it's fetched, how it's used and how it's being stored in the context of ***TrackIO*** mobile app that is being developed by Group 3 Semester 3 students at Fontys University of applied sciences.

Mobile app

TrackIO is a mobile app for locating employees within the company. Since the pandemic silently ended, some employees have hybrid working schedule so they might be working from home and some employees have issues with locating them. It could also be an issue that some iO Digital campuses and buildings have multiple floors therefore finding certain employees may be inefficient and time-intensive.

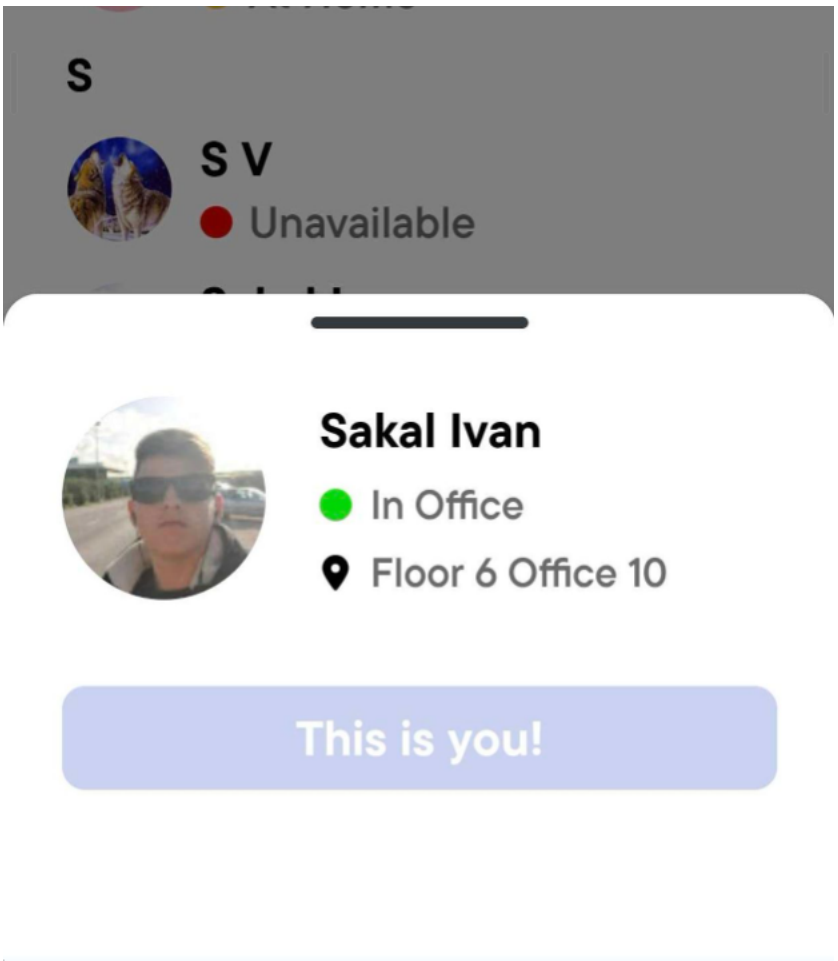

Image 1: Mobile application location demonstration

The core app functionality is this: users log in using iO's preferred identity provider, and once they have logged in they have all the features unlocked in the application. Those include: status setting, user browsing, pinning favorite/most frequented users, looking for (pinging) certain users.

One of the features that is requested is automatic location switching based on Wi-Fi connection. IO Digital campuses have a Wi-Fi network reserved only for its employees and since iO campuses can be quite large, that network is spread out between multiple routers. Each router is placed in a specific location and connection data is being gathered in real time and being stored in company's AWS S3 storage solution.

Usage

This data is going to be used for automatic location switching per user. Users can opt in or out of that feature at will. Wi-Fi storage solution groups users per router(location). This location will be displayed in your profile view as demonstrated in Image 1.

Architecture

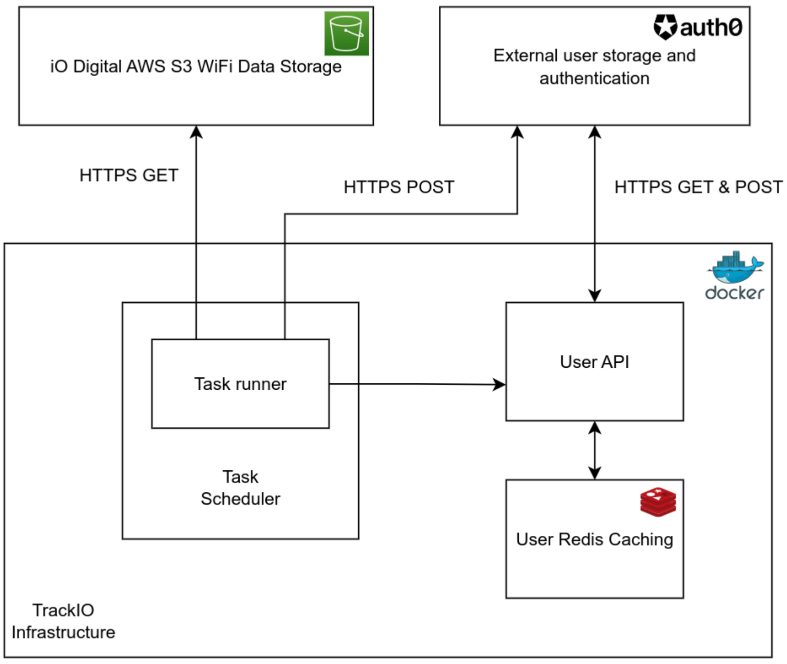

Image 2: Project architecture demonstration

Explanation

To give context to the explanation, I'll go over the TrackIO architecture which is demonstrated in Image 2. We chose Auth0 as our main way of user authentication and data storage. Auth0 is a developer-friendly, cost and time-effective way of integrating authentication and authorization into the application, and they provide storage for user data that we utilize in own our way to store any data related to user. We are using data storage to store any other data that is crucial for mobile app to work, such as: user statuses, FCM tokens (tokens required for push notifications) and MAC addresses. We have a Redis database instance in place to cache user data from Auth0. Reason behind caching data is sometimes Auth0 responses can be particularly slow. Data we are caching is everything that Auth0 stores on their end about users and they claim to be GDPR compliant. But caching system doesn't play any significant part in Wi-Fi data usage, in terms of application logic. Its access if very limited, so if we were to remove it, the application would work as before, but slower.

Component	Purpose	Location	Data storage	Data Transfer protocol
iO Digital AWS S3 Wi-Fi Data Storage	Managed by iO Digital, stores the data about Wi-FI router connections	Local	✅	HTTPS
Auth0	User authentication & authorization, data storage	External/Cloud	✅	HTTPS
User API	Middleware between Auth0 and mobile application, sends notifications to devices	Local	❌	HTTPS
User Redis Caching	Caches responses from Auth0	Local	✅	HTTPS
Task Scheduler	Schedules tasks	Local	❌	N/A
Task runner	Executes tasks (In our case, automatic Wi-Fi location setting task)	Local	❌	HTTPS

Security claims

Our main storage solution is Auth0 and they claim to be GDPR compliant.

We make sure all of our inter-component communication is done on a secure channel, such as HTTPS. We cache Auth0's API response which contains non-sensitive user data. These are all the security claims by Auth0:

https://auth0.com/docs/secure/data-privacy-and-compliance/gdpr#auth0-features-aiding-gdpr-compliance

https://auth0.com/docs/secure/security-guidance/data-security/user-data-storage

https://auth0.com/docs/secure/data-privacy-and-compliance/data-processing

https://auth0.com/blog/how-we-store-data-in-the-cloud-at-auth0/

Data usage

Architecture

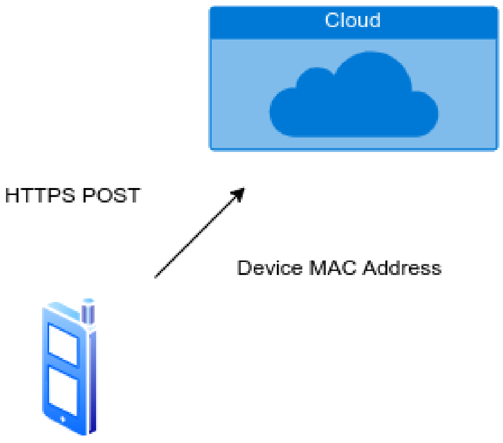

Image 3: Data transfer demonstration on mobile application start

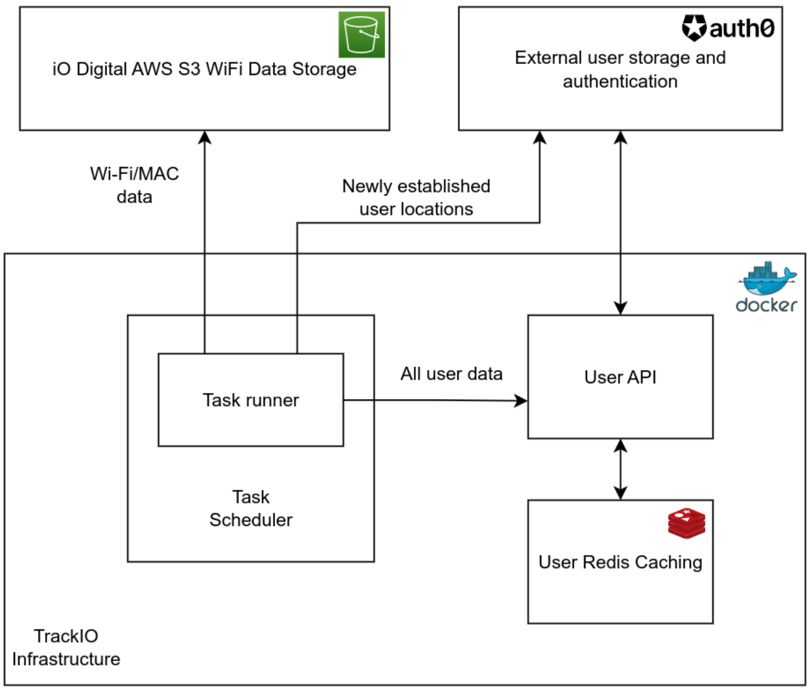

Image 4: Data flow flow demonstration throughout whole architecture

Explanation

Before task scheduler & runner are explained, we first need to take a look at data flow and actual scheduling. IO Digital's S3 bucket is periodically updated with latest data every X minutes. Task scheduler will be configured to match the S3 refresh rate so it runs through the Wi-Fi data only one time per data upload. That way we don't waste resources traversing through same data more than once.

```
{
    "created_at": "2023-03-22T22:30:43.944Z",
    "email": "sakalivan4@gmail.com",
    "email_verified": true,
    "family_name": "Sakal",
    "given_name": "Ivan",
    "identities": [
        {
            "provider": "google-oauth2",
            "user_id": "104188134538566592581",
            "connection": "google-oauth2",
            "isSocial": true
        }
    ],
    "locale": "en",
    "name": "Ivan Sakal",
    "nickname": "sakalivan4",
    "picture": "https://lh3.googleusercontent.com/a/AGNmyxbBiaMTYW0pr95iEjs2CeGWHa3N207mDFFUwbOoJQ=s96-c",
    "updated_at": "2023-04-26T15:13:52.799Z",
    "user_id": "google-oauth2|104188134538566592581",
    "user_metadata": {
        "fcmToken": "cwbai45cTty8URJR0qAKSG:APA91bHwQVzTB4ypuGc8L9iWabBc84iAdyaH-bouK8Rf43dNIt-g1bcgkGVeL-OpI6axvWYdEZgtyKY_mQRm_9H0UGDSFrh
        "favorites": [
            "google-oauth2|105561884367336807374",
            "windowslive|21d59fc461d2f1fa"
        ],
        "macAddress": "00:1B:44:11:3A:B7",
        "status": "Unavailable",
        "location": "Floor 1"
    },
    "last_ip": "31.217.8.203",
    "last_login": "2023-04-26T12:14:21.966Z",
    "logins_count": 14,
    "blocked_for": [],
    "guardian_authenticators": []
}
```

Once task is ran, it fetches Wi-Fi data data and user data. Wi-Fi data is fetched from iO Digital’s S3 bucket, and user data is fetched from our API. Both endpoints are called using HTTPS protocol, therefore data won’t leak or get malformed on its way. Wi-Fi data is saved to memory, just like user data fetched from TrackIO user API. Wi-Fi data should contain which devices are connected to Wi-Fi connection in which position / on which floor of the building. Every device has its own unique MAC Address which we keep in user data. Read more about MAC addresses here. Task loops over each MAC address, links the unique address to a user, and then changes user’s location to the one listed in Wi-Fi data. Once the task loops over every MAC address in Wi-Fi data, every new location change is updated by means of sending multiple HTTP requests to TrackIO user API. The sensitive data doesn’t leave the task environment, only new locations gets sent through TrackIO API (*TrackIO API doesn’t store any data by itself*) and TrackIO API forwards locations to Auth0 data storage. Task is then finished, memory is cleared which means any previously kept data is erased.