

# Politechnika Lubelska

Wydział Elektrotechniki i Informatyki

Kierunek Informatyka



Programowanie full-stack w chmurze obliczeniowej

**Student**

Adrian Duwer

nr. albumu: 90902

grupa lab: I2N 1.1/1

**Prowadzący**

dr. inż. Sławomir Przyłucki

Lublin 2023

## Spis treści

Zadanie 1 .....	3
Zadanie 2 .....	5
Zadanie 3 .....	7

# Zadanie 1

Zadanie podzielone jest na dwie części : obowiązkową i dodatkową (nieobowiązkową)

## CZĘŚĆ OBOWIĄZKOWA

### 1. (max. 20%)

Proszę napisać program serwera (dowolny język programowania), który realizować będzie następującą funkcjonalność:

- po uruchomieniu kontenera, serwer pozostawia w logach informację o dacie uruchomienia, imieniu i nazwisku autora serwera (imię i nazwisko studenta) oraz porcie TCP, na którym serwer nasłuchuje na zgłoszenia klienta.
- na podstawie adresu IP klienta łączącego się z serwerem, w przeglądarce powinna zostać wyświetlona strona informująca o adresie IP klienta i na podstawie tego adresu IP, o dacie i godzinie w jego strefie czasowej.

W sprawozdaniu proszę umieścić kod oprogramowania wraz z niezbędnymi komentarzami.

```
index.php X
index.php
1 <?php
2 // Pobierz IP klienta
3 $clientIP = $_SERVER['REMOTE_ADDR'];
4
5 // Pobierz datę i godzinę w strefie czasowej serwera
6 $serverTimeZone = new DateTimeZone(date_default_timezone_get());
7 $serverDateTime = new DateTime('now', $serverTimeZone);
8 $serverDateTimeString = $serverDateTime->format('l, d-m-Y, H:i:s');
9
10 // Pobierz informacje o strefie czasowej na podstawie adresu IP klienta
11 $ipInfoUrl = "https://ipapi.co/{$clientIP}/json/";
12 $ipInfo = json_decode(file_get_contents($ipInfoUrl));
13
14 // Sprawdź, czy informacja o strefie czasowej została pobrana poprawnie
15 if (isset($ipInfo->timezone) && !empty($ipInfo->timezone)) {
16     $clientTimeZone = new DateTimeZone($ipInfo->timezone);
17 } else {
18     $clientTimeZone = $serverTimeZone; // Użyj domyślnej strefy czasowej serwera
19 }
20
21 // Pobierz datę i godzinę w strefie czasowej klienta
22 $clientDateTime = new DateTime('now', $clientTimeZone);
23 $clientDateTimeString = $clientDateTime->format('l, d-m-Y, H:i:s');
24
25 // Zapisz informacje do logów:
26 $logMessage = sprintf("Serwer uruchomiony dnia: %s \nprzez: Adrian Duwer \nna porcie: %s \n", $serverDateTimeString, $_SERVER['SERVER_PORT']);
27 file_put_contents('log.txt', $logMessage . PHP_EOL, FILE_APPEND);
28
29 // Wyświetl stronę informacyjną dla klienta
30 echo "<html>
31 <head>
32 <title>Informacje o kliencie</title>
33 </head>
34 <body>
35 <h1>Adres IP klienta: $clientIP</h1>
36 <h2>Data i godzina: $clientDateTimeString</h2>
37 </body>
38 </html>";
39 ?>
```

Rys. 1.1. Kod programu dotyczący zadania 1.

### Opis działania kodu:

Na początku kodu zdefiniowana jest zmienna \$clientIP, która przechowuje adres IP klienta. Następnie kod pobiera bieżącą datę i godzinę w strefie czasowej serwera i zapisuje je w zmiennej \$serverDateTimeString. Wykorzystuje do tego obiekt DateTime i DateTimeZone, które pozwalają na operacje na czasie i strefach czasowych. Wykorzystujemy usługę IPAPI (<https://ipapi.co/>), aby uzyskać informacje o strefie czasowej na podstawie adresu IP klienta. Tworzony jest URL z adresem IP klienta, a następnie pobierane są dane w formacie JSON przy użyciu funkcji file\_get\_contents(). Otrzymane dane są dekodowane i przechowywane w zmiennej \$ipInfo. Po pobraniu informacji o strefie czasowej klienta, kod sprawdza, czy informacja została pobrana

poprawnie, sprawdzając, czy zmienna \$ipInfo zawiera pole timezone i czy nie jest puste. Jeśli informacja o strefie czasowej jest dostępna, tworzona jest nowa strefa czasowa \$clientTimeZone na podstawie otrzymanych danych. Jeśli jednak nie udało się pobrać informacji o strefie czasowej, kod użyje domyślnej strefy czasowej serwera. Następnie pobieramy bieżącą datę i godzinę w strefie czasowej klienta przy użyciu obiektu DateTime i \$clientTimeZone. Wynik jest zapisywany w zmiennej \$clientDateTimeString. Kolejnym krokiem jest zapisanie informacji o uruchomieniu serwera do pliku logu. Tworzony jest komunikat zawierający datę uruchomienia serwera, autora i port serwera. Komunikat ten jest zapisywany w pliku "log.txt" za pomocą funkcji file\_put\_contents(). Na końcu generowana jest strona HTML, która wyświetla informacje o kliencie. Cała struktura strony HTML jest generowana dynamicznie przy użyciu funkcji echo.



```
{
  "ip": "31.135.25.5",
  "network": "31.135.24.0/22",
  "version": "IPv4",
  "city": "Swidnik",
  "region": "Lublin",
  "region_code": "06",
  "country": "PL",
  "country_name": "Poland",
  "country_code": "PL",
  "country_code_iso3": "POL",
  "country_capital": "Warsaw",
  "country_tld": ".pl",
  "continent_code": "EU",
  "in_eu": true,
  "postal": "21-040",
  "latitude": 51.202,
  "longitude": 22.6468,
  "timezone": "Europe/Warsaw",
  "utc_offset": "+0200",
  "country_calling_code": "+48",
  "currency": "PLN",
  "currency_name": "Zloty",
  "languages": "pl",
  "country_area": 312685.0,
  "country_population": 37978548,
  "asn": "AS56983",
  "org": "SWIDMAN S.C. Mariusz Bzowski, Dariusz Drelich"
}
```

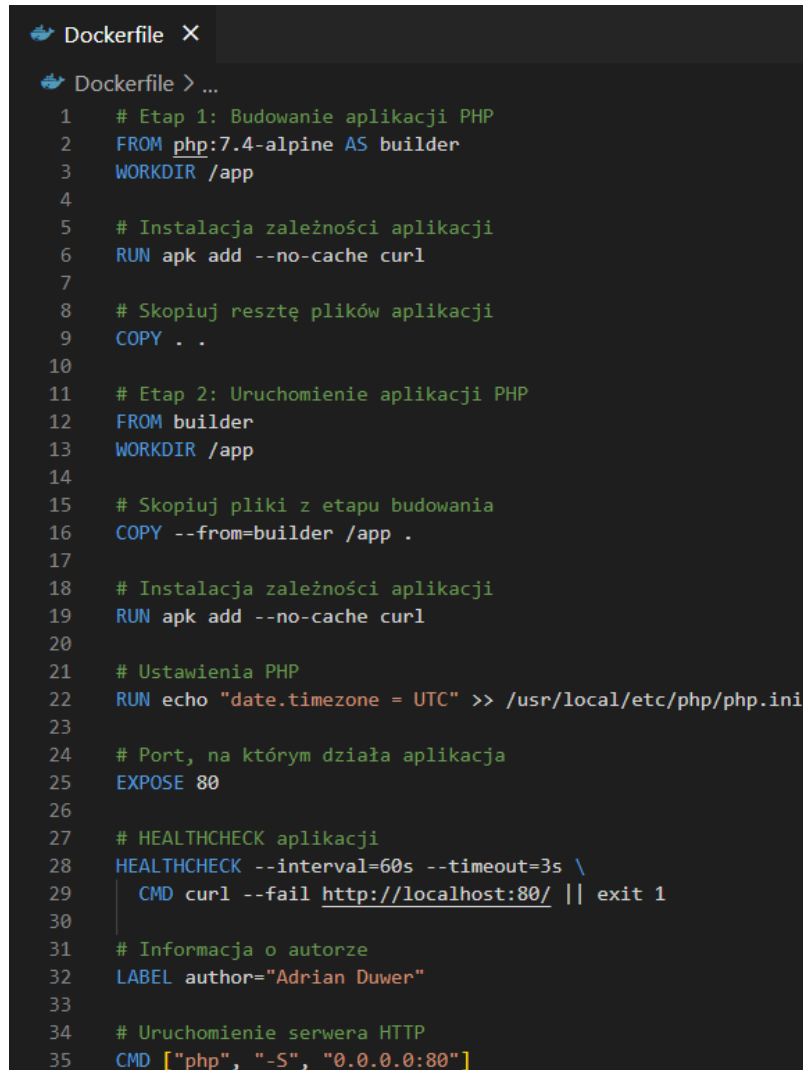
*Rys. 1.2. Dane otrzymane na podstawie IP przy użyciu IPAPI.CO*

## Zadanie 2

### 2. (max. 50%)

Opracować plik Dockerfile, który pozwoli na zbudowanie obrazu kontenera realizującego funkcjonalność opisaną w punkcie 1. Przy ocenie brane będzie sposób opracowania tego pliku (wieloetapowe budowanie obrazu, ewentualne wykorzystanie warstwy scratch, optymalizacja pod kątem funkcjonowania cache-a w procesie budowania, optymalizacja pod kątem zawartości i ilości warstw, healthcheck itd.). Dockerfile powinien również zawierać informację o autorze tego pliku (ponownie imię oraz nazwisko studenta).

*W sprawozdaniu proszę umieścić plik Dockerfile wraz z niezbędnymi komentarzami.*



```
Dockerfile X
Dockerfile > ...
1 # Etap 1: Budowanie aplikacji PHP
2 FROM php:7.4-alpine AS builder
3 WORKDIR /app
4
5 # Instalacja zależności aplikacji
6 RUN apk add --no-cache curl
7
8 # Skopiuj resztę plików aplikacji
9 COPY . .
10
11 # Etap 2: Uruchomienie aplikacji PHP
12 FROM builder
13 WORKDIR /app
14
15 # Skopiuj pliki z etapu budowania
16 COPY --from=builder /app .
17
18 # Instalacja zależności aplikacji
19 RUN apk add --no-cache curl
20
21 # Ustawienia PHP
22 RUN echo "date.timezone = UTC" >> /usr/local/etc/php/php.ini
23
24 # Port, na którym działa aplikacja
25 EXPOSE 80
26
27 # HEALTHCHECK aplikacji
28 HEALTHCHECK --interval=60s --timeout=3s \
29 | CMD curl --fail http://localhost:80/ || exit 1
30
31 # Informacja o autorze
32 LABEL author="Adrian Duwer"
33
34 # Uruchomienie serwera HTTP
35 CMD ["php", "-S", "0.0.0.0:80"]
```

*Rys. 1.3. Plik Dockerfile*

Poniżej znajduje się tekstowy opis poszczególnych sekcji i instrukcji w pliku Dockerfile:

FROM php:7.4-alpine AS builder: Instrukcja FROM określa bazowy obraz, który zostanie użyty jako podstawa dla obrazu kontenera. W tym przypadku używamy obrazu php:7.4-alpine.

WORKDIR /app: Instrukcja WORKDIR ustawia katalog roboczy (working directory) dla wszystkich kolejnych instrukcji w kontekście tego obrazu.

RUN apk add --no-cache curl: Instrukcja RUN wykonuje polecenie wewnątrz kontenera podczas budowania obrazu. Tutaj instalujemy pakiet curl przy użyciu menedżera pakietów Alpine Linux.

COPY . .: Instrukcja COPY kopiuje pliki z lokalnego systemu plików do kontenera. Tutaj kopiujemy wszystkie pliki i foldery z obecnego katalogu (kontekstu) do katalogu roboczego w kontenerze.

FROM builder: Ta linijka rozpoczyna drugi etap budowania obrazu. Odnosi się do wcześniej zdefiniowanego etapu oznaczonego jako builder.

WORKDIR /app: Ponownie ustawiamy katalog roboczy dla drugiego etapu.

COPY --from=builder /app .: Kopiujemy pliki z etapu budowania (builder) do bieżącego katalogu roboczego w drugim etapie.

RUN apk add --no-cache curl: Instalujemy zależności aplikacji w drugim etapie, tak jak to zostało wykonane w pierwszym etapie.

RUN echo "date.timezone = UTC" >> /usr/local/etc/php/php.ini: Dodajemy wpis do pliku php.ini w celu ustawienia domyślnej strefy czasowej PHP na UTC.

EXPOSE 80: Instrukcja EXPOSE określa port, na którym aplikacja w kontenerze będzie nasłuchiwać.

HEALTHCHECK --interval=60s --timeout=3s CMD curl --fail http://localhost:80/ || exit 1: Instrukcja HEALTHCHECK definiuje mechanizm sprawdzania stanu zdrowia aplikacji wewnątrz kontenera. Tutaj wykonujemy zapytanie HTTP do lokalnego adresu przy użyciu curl. Jeśli zapytanie zwróci błąd, proces sprawdzania zakończy się niepowodzeniem.

LABEL author="Adrian Duwer": Etykieta LABEL umożliwia dodanie metadanych do obrazu. Tutaj dodajemy informację o autorze.

CMD ["php", "-S", "0.0.0.0:80"]: Instrukcja CMD definiuje domyślne polecenie, które zostanie wykonane podczas uruchamiania kontenera. Tutaj uruchamiamy serwer PHP, który nasłuchuje na adresie IP 0.0.0.0 i porcie 80

## Zadanie 3

### 3. (max. 30%)

Należy podać polecenia niezbędne do:

- zbudowania opracowanego obrazu kontenera,
- uruchomienia kontenera na podstawie zbudowanego obrazu,
- sposobu uzyskania informacji, które wygenerował serwer w trakcie uruchamiania kontenera (patrz: punkt 1a),
- sprawdzenia, ile warstw posiada zbudowany obraz.

*W sprawozdaniu należy podać treść poleceń (punkty a – d) w raz w ewentualnymi komentarzami oraz zrzut ekranu okna przeglądarki, potwierdzający poprawne działanie systemu.*

a) docker build -t aplikacja-pfswcho .

```
root@DESKTOP-00TNOHE:~/PFSwCh0-1# ls
Dockerfile index.php
root@DESKTOP-00TNOHE:~/PFSwCh0-1# docker build -t aplikacja-pfswcho .
Sending build context to Docker daemon 4.608kB
Step 1/13 : FROM php:7.4-alpine AS builder
--> 0e3af06bf90e
Step 2/13 : WORKDIR /app
--> Using cache
--> 78b9c8086915
Step 3/13 : RUN apk add --no-cache curl
--> Using cache
--> 7cfe627cf684
Step 4/13 : COPY . .
--> 00ebf37c58f4
Step 5/13 : FROM builder
--> 00ebf37c58f4
Step 6/13 : WORKDIR /app
--> Running in db3520bb659b
Removing intermediate container db3520bb659b
--> 7fc93a5c50b4
Step 7/13 : COPY --from=builder /app .
--> 4595e1d24778
Step 8/13 : RUN apk add --no-cache curl
--> Running in 935a99b8091e
fetch https://dl-cdn.alpinelinux.org/alpine/v3.16/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.16/community/x86_64/APKINDEX.tar.gz
OK: 14 MiB in 33 packages
Removing intermediate container 935a99b8091e
--> 949b21137d21
Step 9/13 : RUN echo "date.timezone = UTC" >> /usr/local/etc/php/php.ini
--> Running in 071aee7b36a6
Removing intermediate container 071aee7b36a6
--> a4e8d90ef51b
Step 10/13 : EXPOSE 80
--> Running in 45d665900c22
Removing intermediate container 45d665900c22
--> a6c1d90ec752
Step 11/13 : HEALTHCHECK --interval=60s --timeout=3s CMD curl --fail http://localhost:80/ || exit 1
--> Running in 3ec2791a241e
Removing intermediate container 3ec2791a241e
--> 2c03ea602611
Step 12/13 : LABEL author="Adrian Duwer"
--> Running in e6e08d6d4944
Removing intermediate container e6e08d6d4944
--> cdd082b0227a
Step 13/13 : CMD ["php", "-S", "0.0.0.0:80"]
--> Running in 974caa4987d7
Removing intermediate container 974caa4987d7
--> e5026fc76adb
Successfully built e5026fc76adb
```

*Rys. 1.4. Budowa opracowanego obrazu kontenera*

b) docker run -p 8080:80 aplikacja-pfswcho

```
root@DESKTOP-00TNOHE:~/PFSwCh0-1# docker run -p 8080:80 aplikacja-pfswcho
[Wed May 24 16:11:43 2023] PHP 7.4.33 Development Server (http://0.0.0.0:80) started
```

*Rys. 1.5. Uruchomienie kontenera na podstawie wygenerowanego obrazu*

c) docker exec nazwa\_obrazu cat log.txt

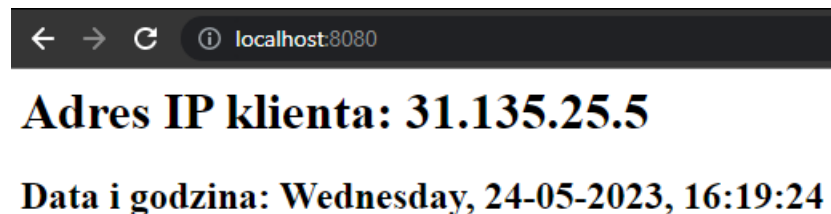
```
root@DESKTOP-00TNOHE:~/PFSwCh0-1# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
7a946e1a3589   aplikacja-pfswo  "docker-php-entrypoin..."  42 seconds ago  Up 40 seconds (health: starting)  0.0.0.0:8080->80/tcp, :::8080->80/tcp  upbeat_goldberg

root@DESKTOP-00TNOHE:~/PFSwCh0-1# docker exec upbeat_goldberg cat log.txt
Serwer uruchomiony dnia: Wednesday, 24-05-2023, 16:19:24
przez: Adrian Duwer
na porcie: 80

Serwer uruchomiony dnia: Wednesday, 24-05-2023, 16:20:19
przez: Adrian Duwer
na porcie: 80

root@DESKTOP-00TNOHE:~/PFSwCh0-1#
```

Rys. 1.6. Wyświetlenie informacji wygenerowanych przez serwer w logach



Rys. 1.7. Wyświetlenie informacji z poziomu przeglądarki

d) docker history aplikacja-pfswocho

```
root@DESKTOP-00TNOHE:~/PFSwCh0-1# docker history aplikacja-pfswocho
IMAGE          CREATED          CREATED BY          SIZE      COMMENT
e5026fc76adb   16 minutes ago  /bin/sh -c #(nop)  CMD ["php" "-S" "0.0.0.0:8080"]  0B
cdd082b0227a   16 minutes ago  /bin/sh -c #(nop)  LABEL author=Adrian Duwer  0B
2c03ea602611   16 minutes ago  /bin/sh -c #(nop)  HEALTHCHECK &{"CMD-SHELL" "curl -f http://localhost:8080/health || exit 1"}  0B
a6c1d90ec752   16 minutes ago  /bin/sh -c #(nop)  EXPOSE 80  0B
a4e8d90ef51b   16 minutes ago  /bin/sh -c echo "date.timezone = UTC" >> /usr/local/etc/php/conf.d/docker-php-conf.d  20B
949b21137d21   16 minutes ago  /bin/sh -c apk add --no-cache curl  51.5kB
4595e1dd24778   16 minutes ago  /bin/sh -c #(nop)  COPY dir:ce5e7d56bd8df6e36...  2.15kB
7fc93a5c50b4   16 minutes ago  /bin/sh -c #(nop)  WORKDIR /app  0B
00ebf37c58f4   16 minutes ago  /bin/sh -c #(nop)  COPY dir:23194db0d04921376...  2.15kB
7cfe627cf684   46 minutes ago  /bin/sh -c apk add --no-cache curl  51.5kB
78b9c8086915   46 minutes ago  /bin/sh -c #(nop)  WORKDIR /app  0B
0e3af06bf90e   6 months ago    /bin/sh -c #(nop)  CMD ["php" "-a"]  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENTRYPOINT ["docker-php-entrypoin..."]  0B
<missing>       6 months ago    /bin/sh -c docker-php-ext-enable sodium  51.5kB
<missing>       6 months ago    /bin/sh -c #(nop)  COPY multi:6edd033b037aa2d...  7.2kB
<missing>       6 months ago    /bin/sh -c set -eux; apk add --no-cache --v...  63.8MB
<missing>       6 months ago    /bin/sh -c #(nop)  COPY file:ce57c04b70896f77...  587B
<missing>       6 months ago    /bin/sh -c set -eux; apk add --no-cache --v...  10.5MB
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHP_SHA256=924846abf9...  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHP_URL=https://www.p...  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHP_VERSION=7.4.33  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENV GPG_KEYS=42670A7FE4D0...  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHP_LDFLAGS=-Wl,-O1 -r...  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHP_CPPFLAGS=-fstack-...  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHP_CFLAGS=-fstack-pr...  0B
<missing>       6 months ago    /bin/sh -c set -eux; mkdir -p "$PHP_INI_DIR...  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHP_INI_DIR=/usr/loca...  0B
<missing>       6 months ago    /bin/sh -c set -eux; adduser -u 82 -D -S -G...  4.68kB
<missing>       6 months ago    /bin/sh -c apk add --no-cache ca-certifica...  3.57MB
<missing>       6 months ago    /bin/sh -c #(nop)  ENV PHPIZE_DEPS=autoconf ...  0B
<missing>       6 months ago    /bin/sh -c #(nop)  CMD ["/bin/sh"]  0B
<missing>       6 months ago    /bin/sh -c #(nop)  ADD file:ceeb6e8632fafc657...  5.54MB

root@DESKTOP-00TNOHE:~/PFSwCh0-1#
```

Rys. 1.8. Wyświetlenie warstw zbudowanego obrazu