

Bezpieczeństwo systemów informatycznych	Data: 24.11.2020
Projekt: koko-kola.xyz	
Autorzy: Mateusz Mikrut, Grzegorz Lipski	

Odnośnik do repozytorium na platformie github:

<https://github.com/KISiM-AGH/projekt-semestralny-trasownicy>

## Aspekty aplikacji nakierowane na bezpieczeństwo:

### 1. Hashowanie hasła

W aplikacji zastosowano hashowanie hasła z wykorzystaniem bcrypt z 9 rundami.

Bcrypt to funkcja skrótu kryptograficznego, która powstała w oparciu o szyfr blokowy Blowfish. Została stworzona głównie w celu hasowania hasła statycznych, a nie jak inne znane funkcje do hashowania dowolnych danych binarnych. Pozwala sterować jego złożonością obliczeniową poprzez zmianę ilości rund w procesie hasowania (tzw. work factor). Daje nam to dużą elastyczność przeciwko atakom w przyszłości.

```

userSchema.pre('save', function (next) {
  if (!this.isModified('password')) return next()
  const rounds = 9
  bcrypt.hash(this.password, rounds).then((hash) => {
    this.password = hash
    next()
  }).catch(next)
})

userSchema.methods = {
  view (full) {
    let view = {}
    let fields = ['id', 'name', ]
    if (full) {
      fields = [...fields, 'role', 'email']
    }
    fields.forEach((field) => { view[field] = this[field] })
    return view
  },
  authenticate (password) {
    return bcrypt.compare(password, this.password).then((valid) =>
      valid ? this : false)
  }
}

userSchema.statics = {
  roles
}

const model = mongoose.model('User', userSchema)
module.exports = {model, userSchema}

```

## 2. Logowanie za pomocą tokena jwt

API naszej aplikacji stanowi prosta aplikacja napisana w Node.js wykorzystująca Express.js do obsługi żądań oraz tokeny JWT.

JSON Web Token (JWT) to otwarty standard (RFC 7519), który określa kompaktowy i niezależny sposób bezpiecznego przesyłania informacji między stronami jako obiekt JSON.

### 3. ReCaptcha podczas logowania

W projekcie wykorzystaliśmy reCAPTCHA v2, które weryfikuje czy użytkownik nie jest botem. Zastosowanie reCAPTCHA w formularzu logowania umożliwia ochronę serwisu przed różnymi atakami np brute force.

Google reCAPTCHA

←

Ustawienia

Etykieta

koko-kola

9/50

Typ reCAPTCHA:

Pole wyboru „wersja 2”

Klucze reCAPTCHA

▼

Domeny

×


koko-kola.xyz

+

Dodaj domenę, np. example.com

email address

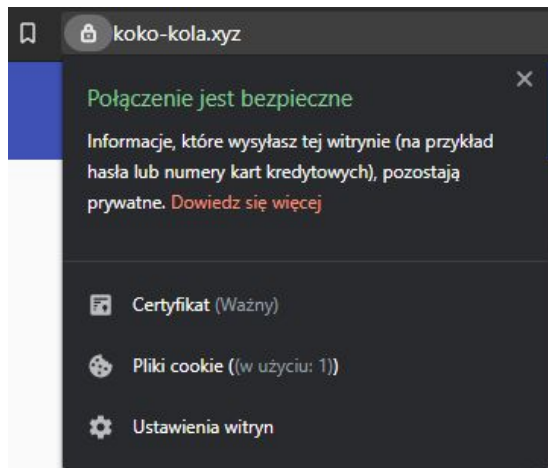
☐ Nie jestem robotem

  
reCAPTCHA  
Prywatność - Warunki

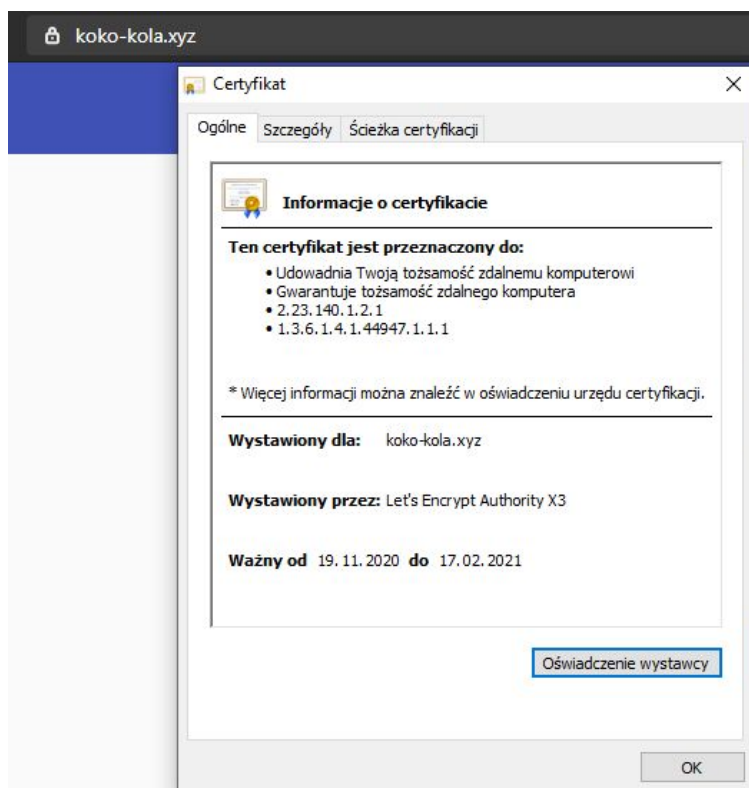
LOGIN

#### 4. Https zamiast http

Połączenie z witryną koko-kola.xyz jest szyfrowane z wykorzystaniem protokołu ssl.



Darmowy certyfikat ssl został wygenerowany za pomocą *let's encrypt*:



#### 5. Autoryzacja

W zależności od pomyślnej identyfikacji i uwierzytelnienia użytkownika, otrzymuje on dostęp do zasobów strony internetowej. W przypadku próby wejścia niezalogowanego użytkownika na wybraną podstronę zostanie on automatycznie przekierowany na stronę logowania. Podobnie część opcji na stronie jest na wstępie zablokowana jak np. menu.

## 6. Ograniczenie dostępu do bazy danych

Ograniczono dostęp do bazy danych poprzez stworzenie whitelisty adresów IP, które mogą się z nią połączyć. Wykorzystano narzędzia do zarządzania dostępem MongoDB Atlas. Na liście znajduje się adres IP komputera administratora, adres sieci, w której znajdują się wirtualne maszyny wysyłające dane do bazy oraz adres instancji EC2 AWS, gdzie znajduje się część backendowa aplikacji.

### Network Access

IP Access List

Peering

Private Endpoint

+ ADD IP ADDRESS

You will only be able to connect to your cluster from the following list of IP Addresses:

IP Address	Comment	Status	Actions
34.203.217.5/32	backend	<div><div></div>Active</div>	<div><div>⚙ EDIT</div><div>🗑 DELETE</div></div>
149.156.124.1/32 (includes your current IP address)	administracja	<div><div></div>Active</div>	<div><div>⚙ EDIT</div><div>🗑 DELETE</div></div>
149.156.96.169/32	VM	<div><div></div>Active</div>	<div><div>⚙ EDIT</div><div>🗑 DELETE</div></div>

## 7. Ograniczenie dostępu do backendu

W konsoli AWS zamknięto dostęp do wszystkich portów wirtualnej maszyny, na której znajduje się backend, za wyjątkiem portu 9000, na którym nasłuchuje serwer Express backendu obsługujący zapytania do API, oraz portu 22, który jest wykorzystywany do połączeń SSH. Ustawiono reguły tak, aby port SSH był dostępny wyłącznie z określonego adresu.

### ▼ Inbound rules

<input type="text" value="Filter rules"/>			
Port range	Protocol	Source	Security groups
9000	TCP	0.0.0.0/0	launch-wizard-2
22	TCP	149.156.124.1/32	launch-wizard-2

Dodatkową warstwę zabezpieczeń stanowi autoryzacja kluczem prywatnym przy połączeniu SSH.

## 8. Obsługa CORS

Aby umożliwić obsługę zapytań przez frontend znajdujący się na innym serwerze, w części backendowej zaimplementowano obsługę CORS. Wykorzystano middleware cors dla serwera Express. Ograniczono origin wyłącznie do witryny frontendowej.

```
const expressConfig = (apiRoot, routes) => {  
  const app = express()  
  const corsOptions = {  
    origin: 'https://koko-kola.xyz/',  
    optionsSuccessStatus: 200  
  }  
  
  app.use(cors(corsOptions));  
}
```

## 9. Security headers

Kolejną warstwę zabezpieczeń stanowią odpowiednie headery zapytań HTTP. Nagłówki przykładowej odpowiedzi:

```
Access-Control-Allow-Origin: https://koko-kola.xyz  
Vary: Origin  
Content-Security-Policy: default-src 'self';base-uri  
'self';block-all-mixed-content;font-src 'self' https: data:;frame-ancestors  
'self';img-src 'self' data:;object-src 'none';script-src 'self';script-src-attr  
'none';style-src 'self' https: 'unsafe-inline';upgrade-insecure-requests  
X-DNS-Prefetch-Control: off  
Expect-CT: max-age=0  
X-Frame-Options: SAMEORIGIN  
Strict-Transport-Security: max-age=15552000; includeSubDomains  
X-Download-Options: noopen  
X-Content-Type-Options: nosniff  
X-Permitted-Cross-Domain-Policies: none  
Referrer-Policy: no-referrer  
X-XSS-Protection: 0  
Content-Type: application/json; charset=utf-8  
Content-Length: 272  
ETag: W/"110-jv9qt/4d1651zBZkzZoJxuaSXq8"  
Date: Tue, 24 Nov 2020 15:33:27 GMT  
Connection: keep-alive
```

Konfigurację nagłówków wykonano wykorzystując middleware Helmet do serwera Express. Najważniejsze funkcje nagłówków:

- **Content Security Policy**  
Uniemożliwia ładowanie dodatkowych zasobów, przeciwdziała cross-site scripting i innym atakom typu code injection.
- **X-XSS Protection**  
Spełnia tę samą funkcję co CSP w starszych przeglądarkach.

- **HTTP Strict Transport Security**  
Wymusza korzystanie z HTTPS
- **X-Frame Options**  
Zapewnia ochronę przed clickjackingiem poprzez ograniczenie możliwości renderowania strony w ramce
- **Expect CT**  
Wymusza stosowanie się do wymagań Certificate Transparency.
- **X-Frame Content Type Options**  
Zabezpiecza przed sniffingiem

Przetestowano bezpieczeństwo headerów na stronie [securityheaders.com](https://securityheaders.com), gdzie strona otrzymała wynik A. Jedynym brakującym nagłówkiem był Permissions-Policy, jednak zrezygnowano z jego implementacji ze względu na to, że nie jest jeszcze wspierany przez zdecydowaną większość przeglądarek.

## Scan your site now



☐ Hide results ☒ Follow redirects

### Security Report Summary



Site: <https://34.203.217.5:9000/>

IP Address: 34.203.217.5

Report Time: 22 Nov 2020 18:12:18 UTC

Headers: ✔ Content-Security-Policy ✔ X-Frame-Options ✔ Strict-Transport-Security ✔ X-Content-Type-Options  
✔ Referrer-Policy ✖ Permissions-Policy

### Supported By

Probely

Great grade! Perform a deeper security analysis of your website and APIs:

### Raw Headers

HTTP/1.1	404 Not Found
Access-Control-Allow-Origin	https://koko-kola.xyz
Vary	Origin
Content-Security-Policy	default-src 'none'
X-DNS-Prefetch-Control	off
Expect-CT	max-age=0
X-Frame-Options	SAMEORIGIN
Strict-Transport-Security	max-age=15552000; includeSubDomains
X-Download-Options	noopen
X-Content-Type-Options	nosniff
X-Permitted-Cross-Domain-Policies	none
Referrer-Policy	no-referrer
X-XSS-Protection	0
Content-Type	text/html; charset=utf-8
Content-Length	139
Date	Sun, 22 Nov 2020 18:12:18 GMT
Connection	keep-alive

### Missing Headers

**Permissions-Policy** [Permissions Policy](#) is a new header that allows a site to control which features and APIs can be used in the browser.

### Additional Information

<b>Access-Control-Allow-Origin</b>	The Access-Control-Allow-Origin header is used to configure CORS.
<b>Content-Security-Policy</b>	<a href="#">Content Security Policy</a> is an effective measure to protect your site from XSS attacks. By whitelisting sources of approved content, you can prevent the browser from loading malicious assets. <a href="#">Analyse</a> this policy in more detail. You can sign up for a free account on <a href="#">Report URI</a> to collect reports about problems on your site.
<b>Expect-CT</b>	<a href="#">Expect-CT</a> allows a site to determine if they are ready for the upcoming Chrome requirements and/or enforce their CT policy.
<b>X-Frame-Options</b>	<a href="#">X-Frame-Options</a> tells the browser whether you want to allow your site to be framed or not. By preventing a browser from framing your site you can defend against attacks like clickjacking.
<b>Strict-Transport-Security</b>	<a href="#">HTTP Strict Transport Security</a> is an excellent feature to support on your site and strengthens your implementation of TLS by getting the User Agent to enforce the use of HTTPS.
<b>X-Content-Type-Options</b>	<a href="#">X-Content-Type-Options</a> stops a browser from trying to MIME-sniff the content type and forces it to stick with the declared content-type. The only valid value for this header is "X-Content-Type-Options: nosniff".
<b>Referrer-Policy</b>	<a href="#">Referrer Policy</a> is a new header that allows a site to control how much information the browser includes with navigations away from a document and should be set by all sites.
<b>X-XSS-Protection</b>	<a href="#">X-XSS-Protection</a> sets the configuration for the XSS Auditor built into older browsers. The recommended value was "X-XSS-Protection: 1; mode=block" but you should now look at <a href="#">Content Security Policy</a> instead.