Akademia Techniczno-Humanistyczna w Bielsku-Białej

Projekt

Cyberbezpieczeństwo

Sprawozdanie nr 3

Ochrona przed botami, spamem i wprowadzaniem niepożądanych treści

GRUPA:2a / SEMESTR:7 / ROK: 2022

NAZWISKO I IMIĘ / NR INDEKSU

1.Aleksander Michalec / 055932

2. Jakub Wykręt / 055837

3. Szymon Białek / 055872

1. Zadania do zrealizowania:

   Udoskonal program utworzony w ćwiczeniu 2, który wprowadza następujące zasady dla systemu bezpieczeństwa:

   1. Dodaj mechanizm Captcha według zadania indywidualnego.

   2. Dodaj mechanizm Google ReCaptcha.

   Zadanie indywidualne:

   - Podanie odpowiedzi na wyświetlone pytanie (np. Co jest stolicą Wielkiej Brytanii?)

```
public class Captcha
{
    public int Id { get; set; }
    public String Question { get; set; }
    public String Answer { get; set; }

    public Captcha(int id, string q, string a)
    {
        Id = id;
        Question = q;
        Answer = a;
    }

    public static Captcha GenerateCaptchaQuestion()
    {
        var questions = new List<Captcha>()
        {
        new Captcha
        (
            1,
            "Stolica Wielkiej Brytanii to...?",
            "Londyn"
        ),
        new Captcha
        (
            10,
            "Jak nazywa się najmroźniejszy kontynent świata...?",
            "Antarktyda"
        ),
        };

        var rand = new Random();
        return questions.ToArray()[rand.Next(1, 10)];
    }

}
```

Rys. 1

Rys. 1 przedstawia klasę, która zawiera pola na pytanie oraz odpowiedź. Posiada ona również bazę kilku pytań, z których wybierane jest ono później do weryfikacji.

```
12      public bool CaptchaEnabled { get; set; }
13      public bool ReCaptchaEnabled { get; set; }
```

Rys. 2

Natomiast ta część kodu pokazuje model SecuritySettings, który został rozszerzony o pole Captcha oraz ReCaptcha.

```
if (settings is not null)
{
    captchaEnabled = settings.Value.First().CaptchaEnabled;
    reCaptchaEnabled = settings.Value.First().ReCaptchaEnabled;
    if (reCaptchaEnabled)
        style = "display: block";

    var captcha = Captcha.GenerateCaptchaQuestion();
    question = captcha.Question;
    answer = captcha.Answer;
}
```

Rys. 3

Na Rys. 3 część kodu odpowiada za sprawdzanie wartości ustawień podczas renderowania ekranu.

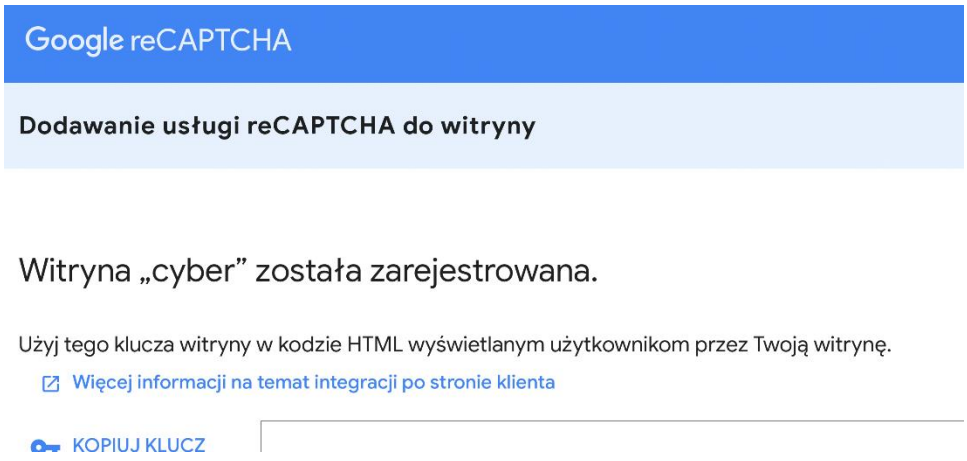
```
@if (captchaEnabled)
{
    <div class="mb-3">
        <label>@question</label>
        <i class="fa-solid fa-arrows-rotate" @onclick="RollQuestion"></i>
        <InputText @bind-Value="@model.Answer" class="form-control" placeholder="Answer" />
    </div>
}
<div id="recaptcha" style="@style"></div>
@if (wrongCaptcha)
{
    <p style="color: red">Weryfikacja nie powiodła się</p>
}
```

Rys. 4

Na Rys. 4 implementujemy w widoku dodatkowe pole formularza.

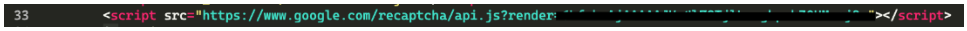
Rys. 5

Natomiast Rys. 5 prezentuje logowanie, które zostało rozszerzone o pole weryfikacji. Warto zauważyć, że użytkownik ma opcję wylosowania innego pytania z bazy.



Rys. 6

Rys. 6 pokazuje wycinek ze strony, na której zarejestrowaliśmy naszą witrynę w celu dostępu do API Google, które zapewni nam widżet ReCaptchy.



Rys. 7

Zaprezentowaliśmy tutaj wycinek z głównego layoutu aplikacji, gdzie dodaliśmy referencję. (Rys.7)

```
14  function googleRecaptcha(dotNetObject, selector, sitekeyValue) {
15      return grecaptcha.render(selector, {
16          "sitekey": "[redacted]",
17          "callback": (response) => { dotNetObject.invokeMethodAsync("CallbackOnSuccess", response); },
18          "expired-callback": () => { dotNetObject.invokeMethodAsync("CallbackOnExpired", response); }
19      });
20  }
21
22  function getResponse(response) {
23      return grecaptcha.getResponse(response);
24  }
```

Rys. 8

Rys. 8 prezentuje implementację skryptu renderującego widżet oraz wysyłającego odpowiedź o statusie weryfikacji.

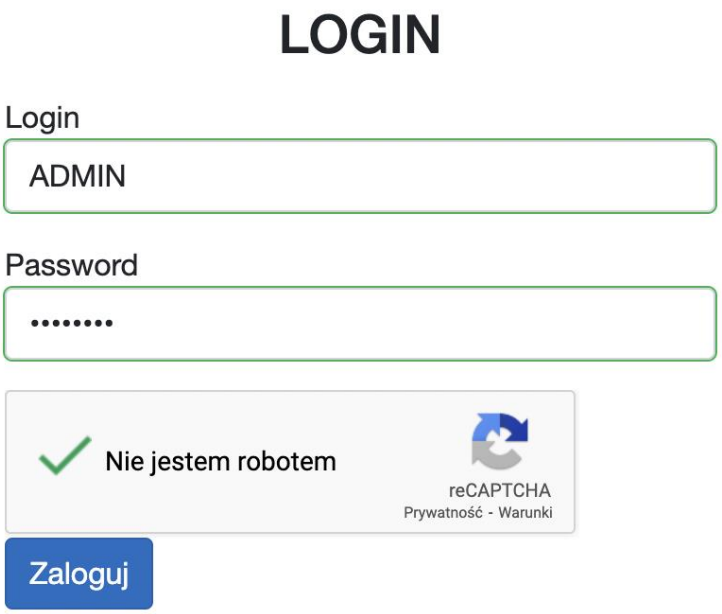
```
[JSInvokable, EditorBrowsable(EditorBrowsableState.Never)]
public void CallbackOnSuccess(string response)
{
    captchaResponse = "success";
}

[JSInvokable, EditorBrowsable(EditorBrowsableState.Never)]
public void CallbackOnExpired(string response)
{
    captchaResponse = "fail";
}

private void ShowResponse()
{
    captchaResponse = $"Odpowiedź: {captchaResponse}";
}
```

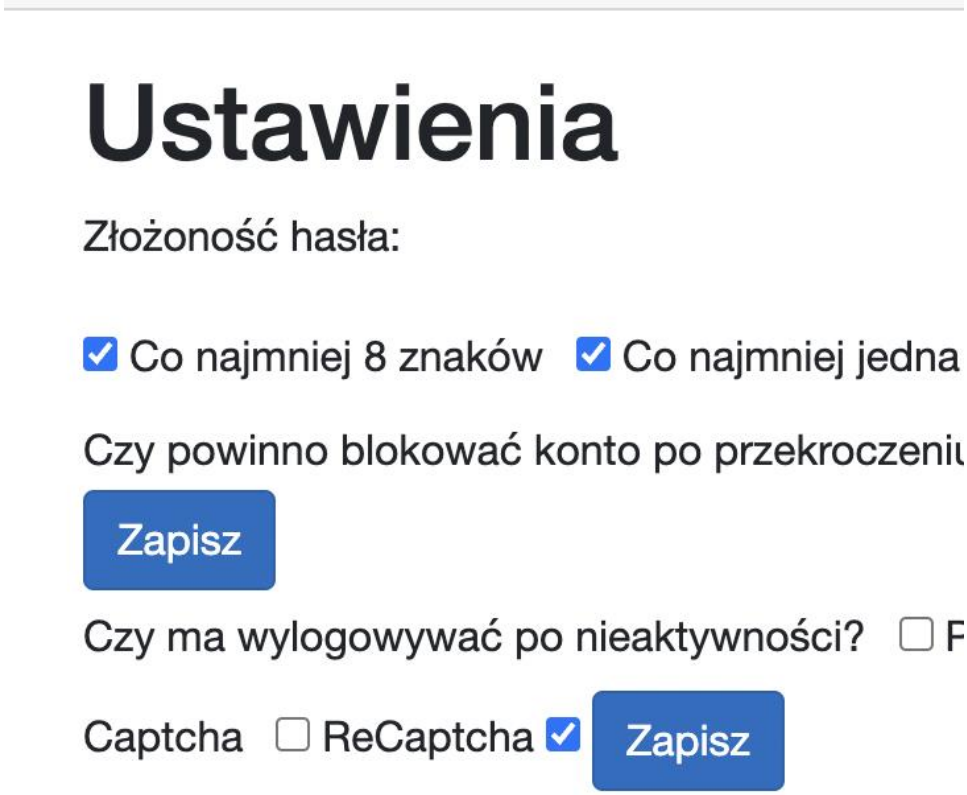
Rys. 9

Możemy tutaj zauważyć, że funkcja ta dostała dostęp do innych, które odpowiadają za przekazanie odpowiedzi do weryfikacji w widoku ekranu logowania. (Rys.9)



Rys. 10

Rys. 10 nowy widok ekranu logowania z włączoną ReCaptchą.



Rys. 11

Rys. 11 prezentuje nowe ustawienia bezpieczeństwa, które wybiera Administrator.