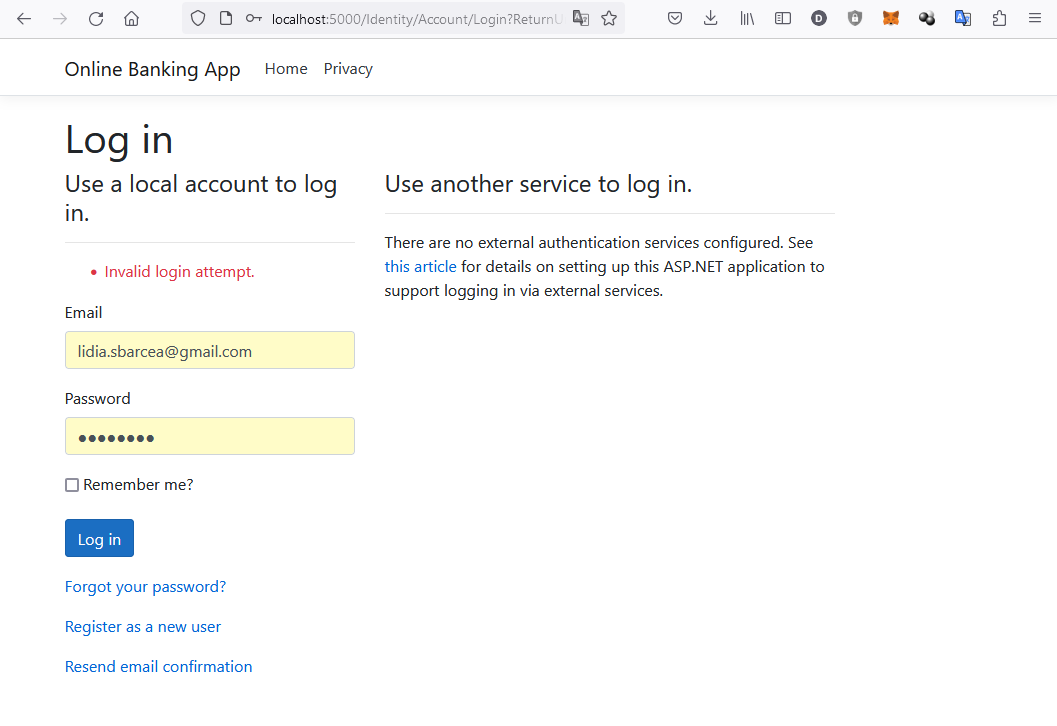
## VULNERABILITĂȚI DE AUTENTIFICARE

ABSENȚA RESTRICȚIILOR CU PRIVIRE LA NUMĂRUL DE AUTENTIFICĂRI INCORECTE

Un adversar va încerca întotdeauna să obțină acces la sistem și să păcălească mecanismul de autentificare. Cel mai prevalent mod de a face acest lucru este prin utilizarea credențialelor compromise, colectate din surse ilicite sau prin folosirea unei liste de parole comune. O aplicație web ASP.NET Core trebuie să reziste la acest tip de exploatare prin implementarea măsurilor defensive cum ar fi respingerea încercărilor excesive de autentificare. Pentru a gestiona utilizatorii, parolele, datele de profil și rolurile a fost folosit framework-ul ASP.NET Core Identity.

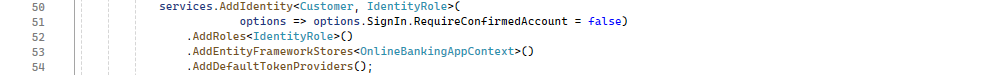
Inițial, după mai multe încercări eșuate contul nu este blocat, ceea ce face ca aplicația să fie vulnerabilă la atacuri brute-force.



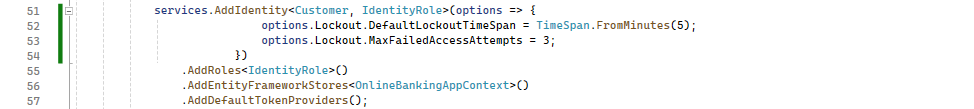


Pentru a mitiga această vulnerabilitate, în metoda PasswordSignInAsync trebuie ca argumentul lockoutOnFailure să fie true. În ASP.NET Core Identity, metoda PasswordSignInAsync este utilizată pentru a autentifica un utilizator cu o parolă. Când argumentul lockoutOnFailure este setat ca true, ASP.NET Core Identity va urmări numărul de încercări eșuate de autentificare pentru un utilizator și va bloca utilizatorul pentru o perioadă de timp dacă numărul de încercări eșuate depășește numărul maxim configurat de încercări.

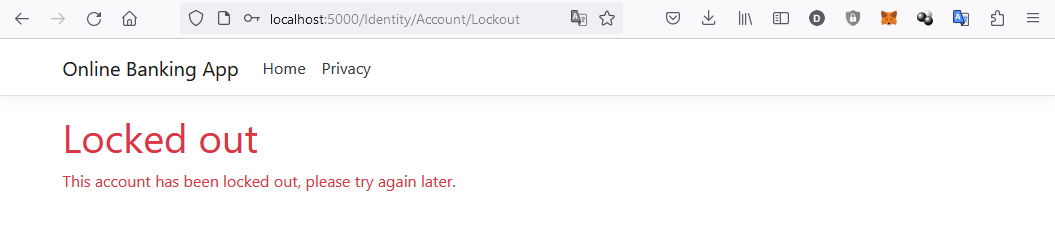




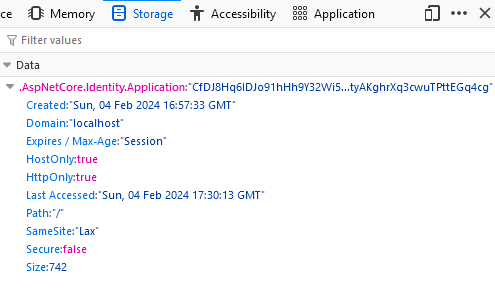
Setările de blocare, cum ar fi numărul maxim de încercări eșuate și perioada de timp de blocare, sunt configurate în LockoutOptions din opțiunile serviciului Identity. Acest lucru va împiedica reîncercările automate și excesive de autentificare de către un actor rău intenționat. Numărul de încercări eșuate poate fi definit folosind MaxFailedAccessAttempts. DefaultLockoutTimeSpan setează timpul pentru cât timp utilizatorul este blocat.



Astfel, după trei încercări eșuate de autentificare, utilizatorul este blocat.



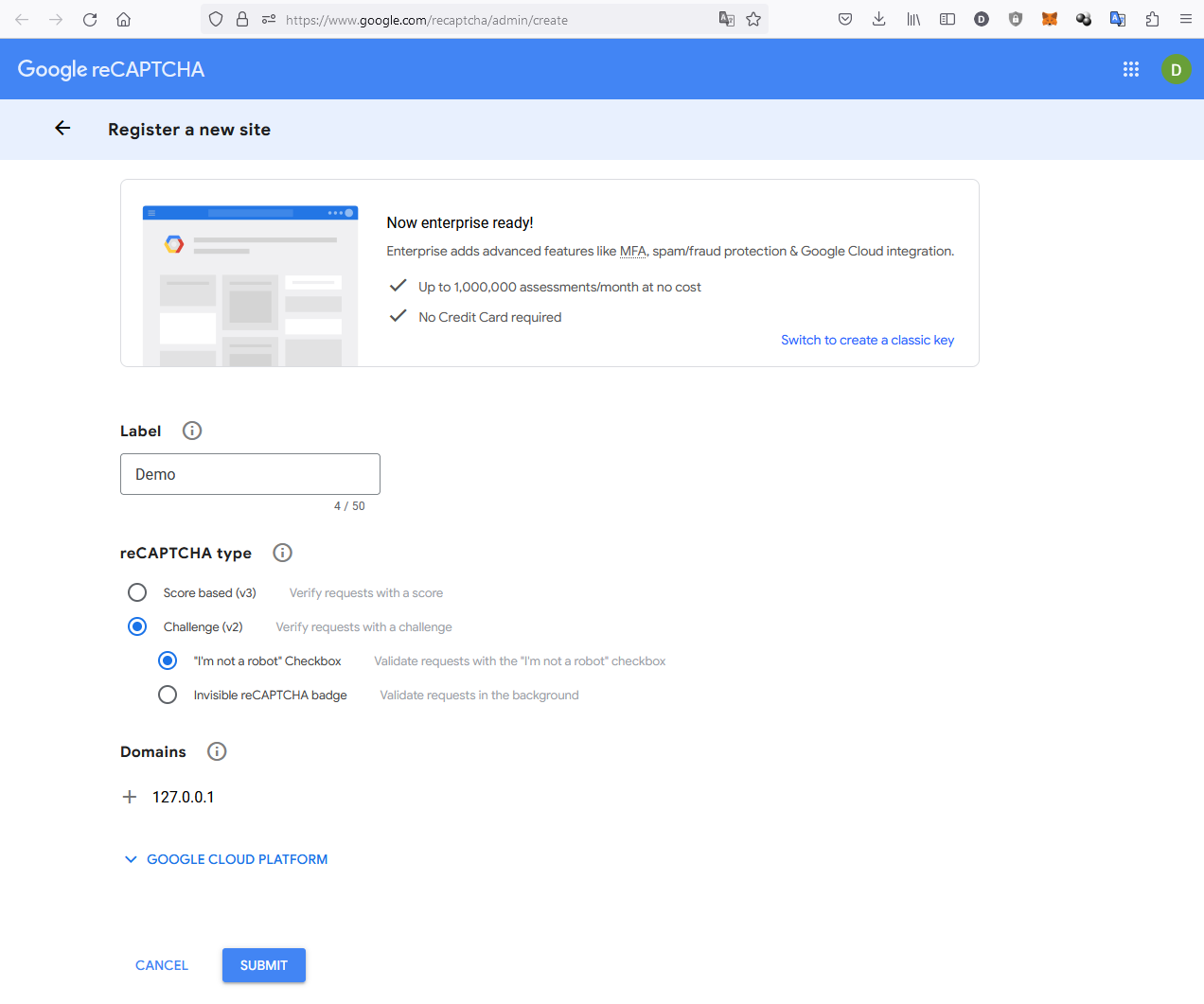
Metoda PasswordSignInAsync verifică dacă utilizatorul cu e-mailul specificat există și dacă parola se potrivește. Dacă credențialele sunt corecte și contul nu este blocat sau nu este necesar să treacă prin autentificarea cu doi factori, utilizatorul este autentificat. În timpul procesului de conectare, ASP.NET Core Identity creează un cookie de autentificare dacă aceasta are succes. Acest cookie este apoi trimis către browserul utilizatorului, permițând serverului să recunoască utilizatorul în solicitările ulterioare. În momentul redirecționării utilizatorului către returnUrl, cookie-ul de autentificare a fost deja setat de framework-ul Identity. Dacă sunt necesari pași suplimentari, cum ar fi autentificarea cu doi factori, procesul gestionează acele cazuri înainte de a seta orice cookie-uri de autentificare.

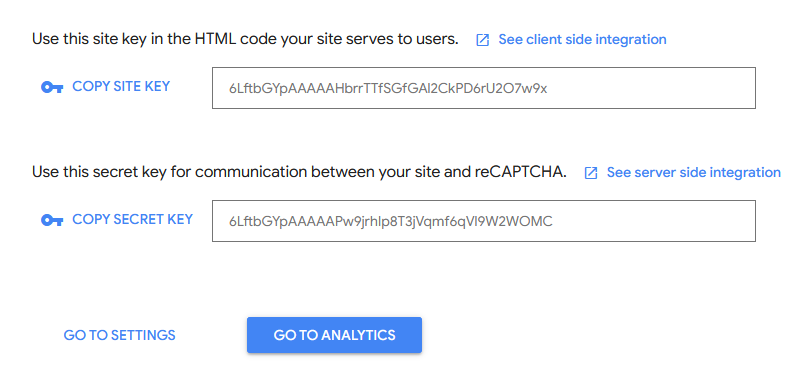


Deși limitarea numărului de încercări de autentificare rezolvă problema atacurilor de tip brute-force, acest lucru facilitează atacuri de tip Denial of Service în care atacatorii blochează în mod deliberat utilizatorii legitimi.

O alternativă mai bună constă în utilizarea serviciului de tip reCAPTCHA furnizat de Google care oferă o balansare mai bună a confortului utilizatorilor și securitate. Mulți utilizatori pot trece testul cu un singur clic, în timp ce comportamentele suspecte declanșează provocări suplimentare. Acesta utilizează instrumente avansate de analiză și machine learning pentru a detecta comportamentele similare roboților, făcându-l mai efectiv asupra atacurilor automatizate.

Pentru utilizarea serviciului reCAPTCHA a fost realizată o configurare a versiunii v2 I am not a robot, în urma căreia au fost obținute cheile pentru integrarea la nivelul clientului și a serverului.





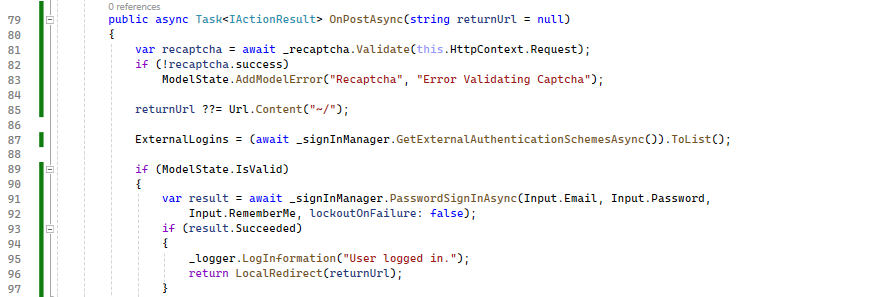


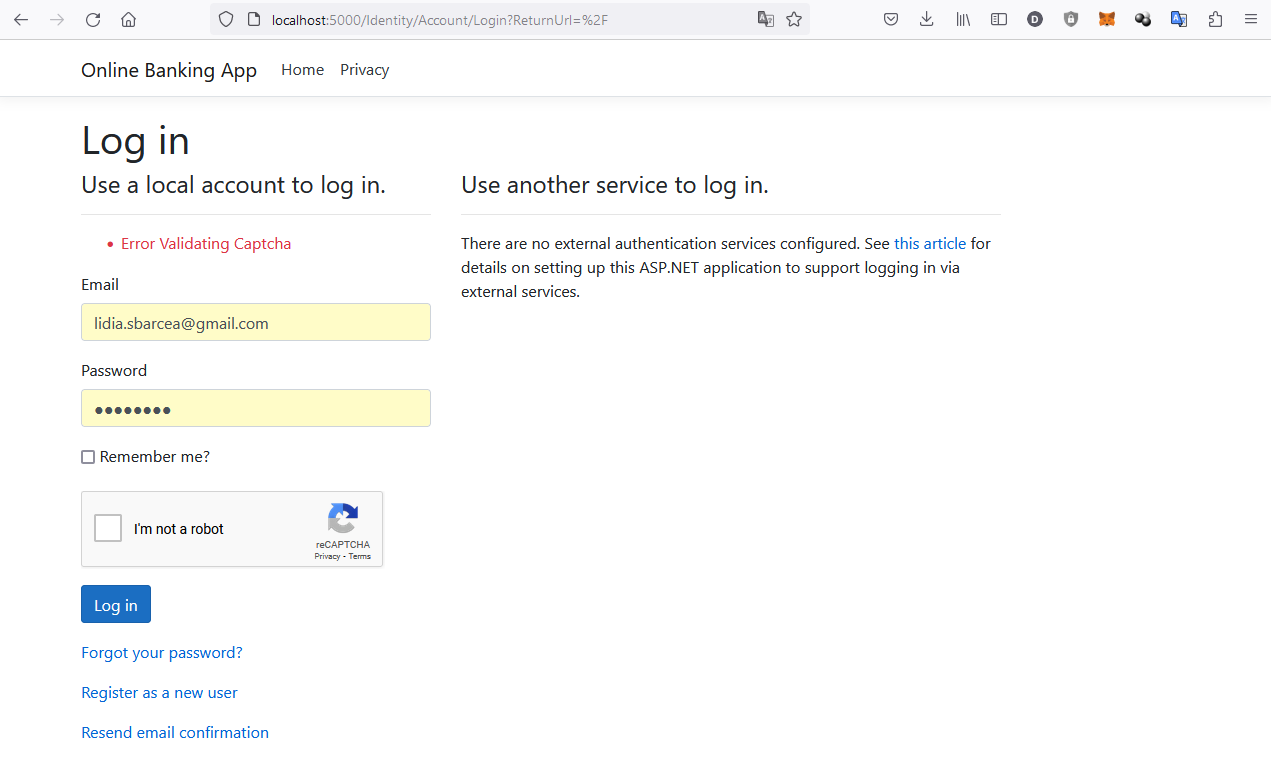
Directiva @inject IOptions<RecaptchaSettings> este folosită pentru a injecta setările reCAPTCHA, respectiv cele două chei, în pagina Razor, făcând posibilă accesarea acestora într-o maniera strongly-typed.

De asemenea la nivelul formularului, a fost inclus un form-group cu scopul de a randa căsuța de verificare a widget-ului reCAPTCHA.



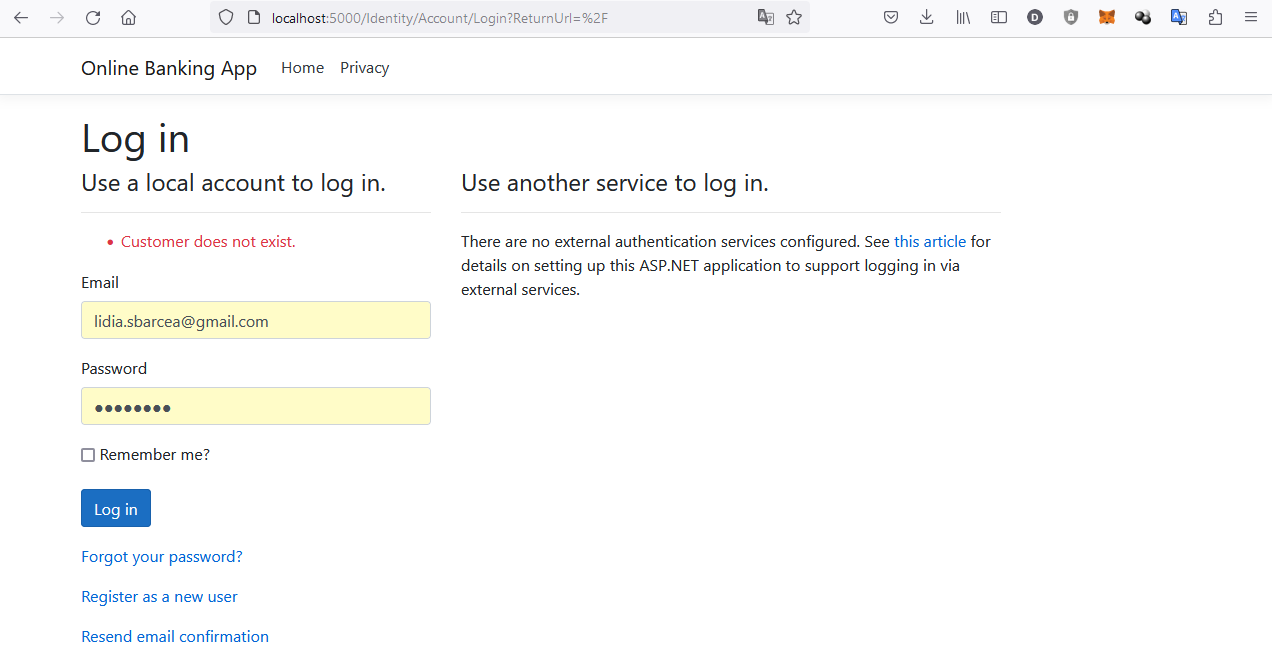
Secvența de cod de pe server asociată serviciului reCAPTCHA trimite o cerere de validare către Google reCAPTCHA API, preluând din cererea HTTP token-ul de răspuns al utilizatorului inclus odată cu trimiterea formularului. După primirea rezultatului, dacă validarea nu s-a efectuat cu succes, variabilei ModelState îi este adăugată o eroare pentru a semnala la nivelul interfeței acest lucru. Procesul de autentificare continuă doar dacă token-ul din cerere este valid.





### MESAJE DETALIATE ÎN CAZUL AUTENTIFICĂRILOR NEREUȘITE

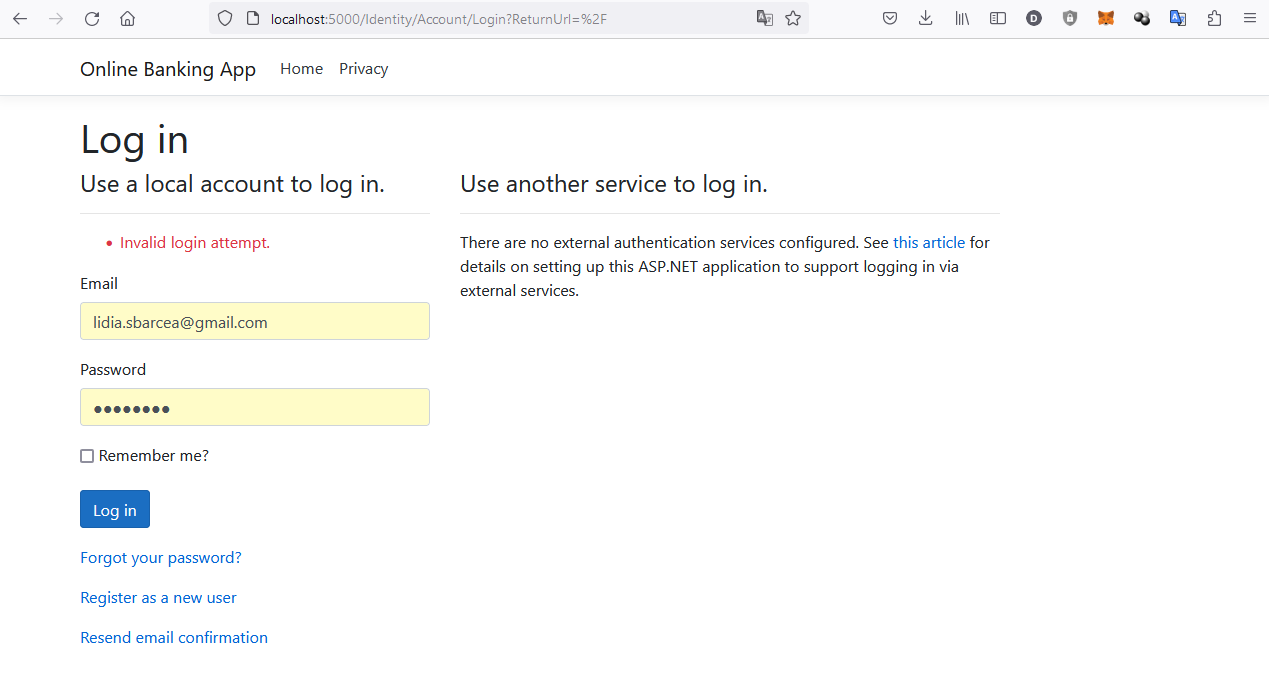
Fiecare informație este vitală pentru un actor rău intenționat. A ști dacă un utilizator există în baza de date a unei aplicații web poate reprezenta informația necesară pentru a executa un atac mai dăunător. Aplicația web trebuie protejată prin nefurnizarea acestor informații și afișarea unor mesaje mai generice în timpul erorilor de autentificare.



Mesajul curent de eroare furnizează ca informație atacatorului faptul că o adresă de email particulară nu există în sistem. Atacatorul poate folosi această informație pentru a colecta adresele de email care au cont.



În consecință, va fi schimbat codul care afișează mesajul de eroare pentru a preveni atacurile de enumerare a utilizatorilor. De asemenea, va fi prevenită și determinarea existenței sau inexistenței conturilor.



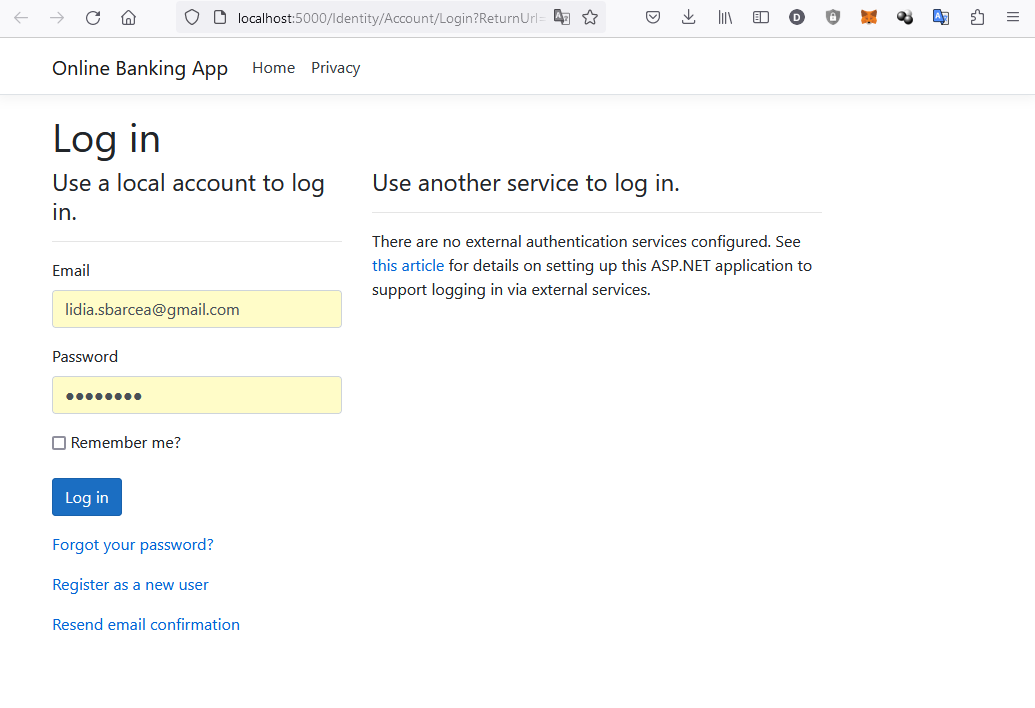


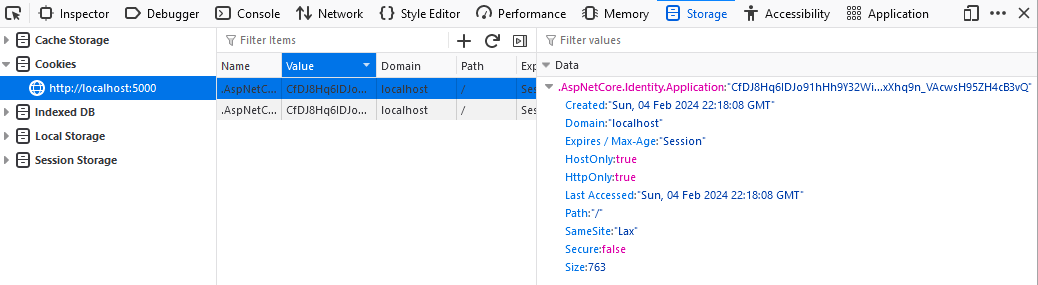
### EXPIRAREA TÂRZIE A UNEI SESIUNI

Pentru autentificarea bazată pe cookie-uri, cum ar fi atunci când se utilizează ASP.NET Identity cu configurarea implicită, expirarea sesiunii de autentificare este determinată de setările de expirare ale cookie-ului. Opțiunea "Remember me" poate face cookie-ul persistent sau nu.

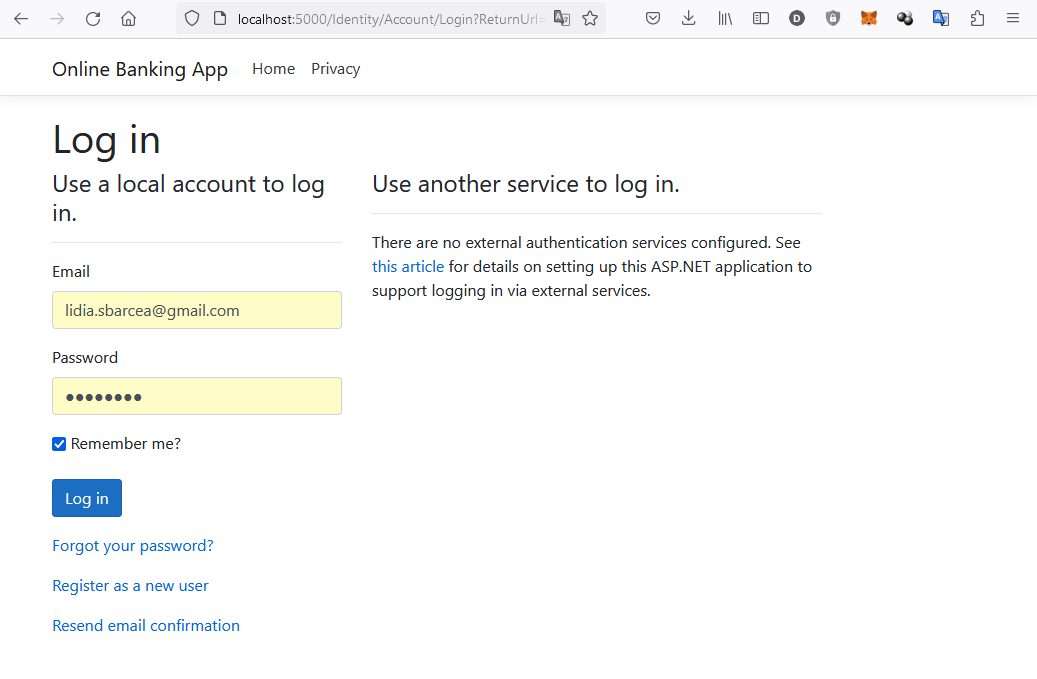
Dacă nu este setat în mod explicit, durata implicită poate varia în funcție de versiunea ASP.NET folosită, dar un standard comun este de 14 zile pentru cookie-urile persistente. Pentru cookie-urile nepersistente (unde cookie-ul nu este destinat să dureze între sesiuni de browser), cookie-ul durează de obicei până când sesiunea de browser se încheie.

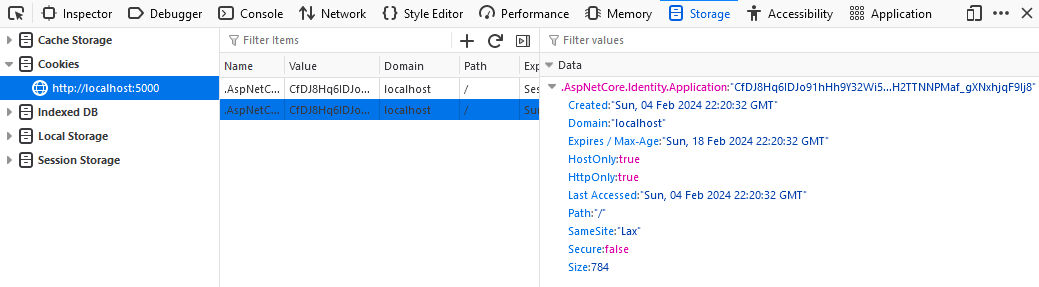
Cookie-urile nepersistente, cunoscute și ca cookie-uri de sesiune sau cookie-uri tranzitorii, sunt temporare și stocate doar în memoria browserului. Ele sunt șterse atunci când utilizatorul închide browserul sau încheie sesiunea. Aceste cookie-uri nu au o dată de expirare setată și durează doar pentru durata sesiunii de browser.



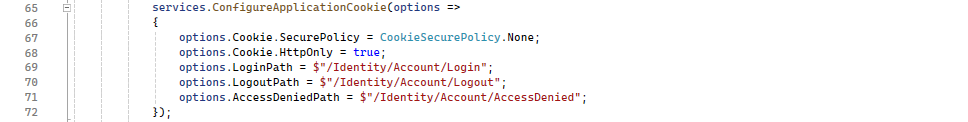


Cookie-urile persistente, cunoscute și ca cookie-uri permanente sau stocate, sunt salvate pe dispozitivul utilizatorului între sesiunile de browser. Acestea permit site-urilor web să rețină informații și setări despre un utilizator pentru viitoarele vizite. Acestea vin cu o dată de expirare specificată de atributul Expires sau o vârstă maximă specificată de atributul Max-Age. Până când această dată sau timpul trece, cookie-ul va rămâne pe dispozitivul utilizatorului.





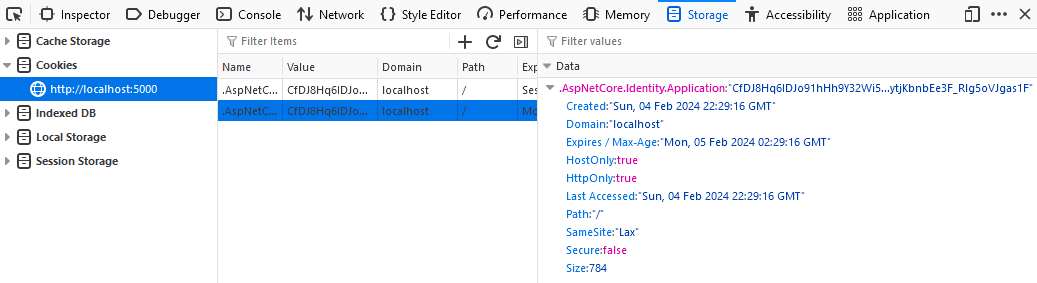
Pentru aplicațiile ASP.NET Core care folosesc middleware-ul de autentificare bazat pe cookie-uri, dacă nu este setată în mod explicit proprietatea ExpireTimeSpan, durata de expirare implicită pentru un cookie persistent este, de obicei, de 14 zile.

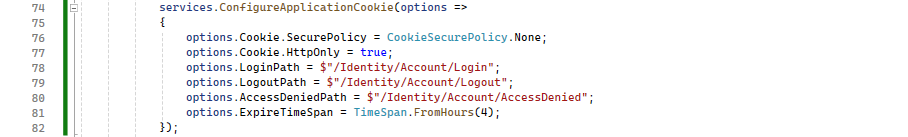


Timpul optim de expirare pentru un cookie de sesiune variază semnificativ în funcție de tipul de aplicație, cerințele de securitate, considerentele experienței utilizatorului și conformitatea cu reglementările. Stabilirea unui timp de expirare adecvat pentru cookie-urile de sesiune este crucială pentru echilibrarea securității și comodității.

Aplicațiile de e-commerce au durata unei sesiuni de la câteva ore la câteva zile. Acest lucru permite utilizatorilor să păstreze articole în coșul de cumpărături între sesiuni fără a compromite securitatea. Aplicațiile bancare și financiare pot avea o sesiune ce durează de la 5 la 15 minute de inactivitate. Aplicațiile financiare necesită o securitate ridicată, astfel că timpurile scurte de expirare ajută la protejarea conturilor utilizatorilor. Rețelele de socializare și forumurile au o sesiune ce poate varia de la câteva zile la câteva săptămâni. Aceste aplicații prioritizează angajarea utilizatorilor și comoditatea, permițând utilizatorilor să rămână autentificați între sesiuni pentru perioade extinse.

În consecință, va fi scurtată valabilitatea unei sesiuni pentru a atenua riscul unor atacuri bazate pe sesiune.



UTILIZAREA UNUI ALGORITM INSUFICIENT DE SIGUR PENTRU HASH-UL PAROLELOR