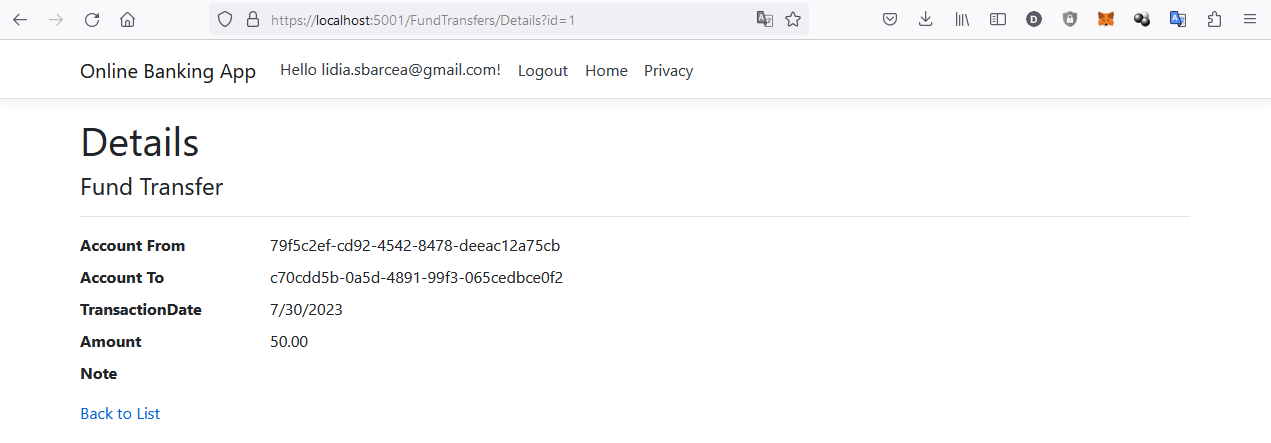
Autorizarea este la fel de semnificativă și esențială ca și autentificarea, definind ce poate să efectueze și să execute un utilizator autentificat. Resursele și paginile web trebuie să aibă privilegii definite pentru a limita accesul neautorizat. Lipsa sau controlul accesului necorespunzător implementat sunt unele dintre vulnerabilitățile de control acces.

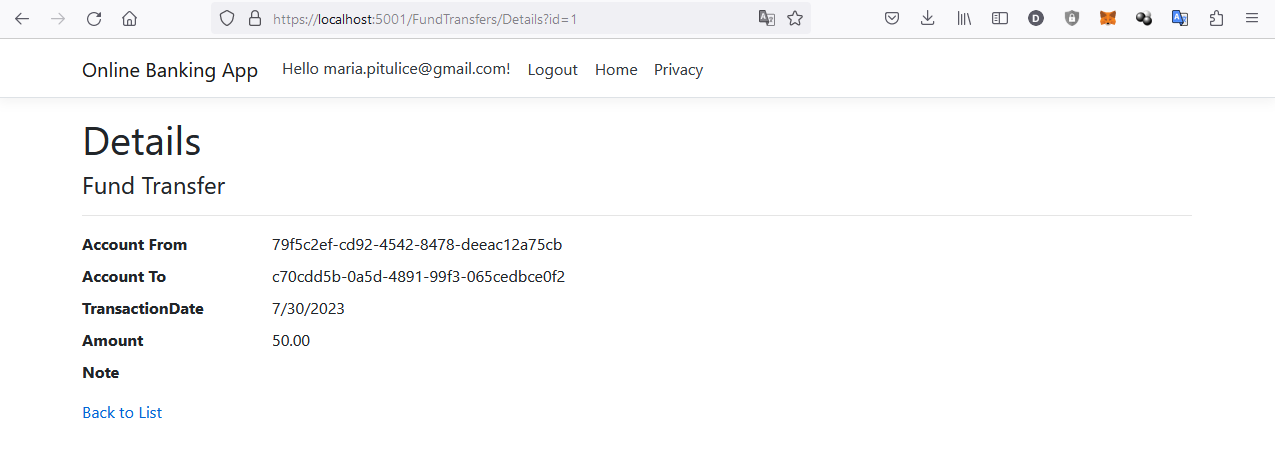
Într-un sistem de autorizare, conceptul de "cerințe" și "resurse" este central pentru controlul accesului. O cerință este un criteriu specific care trebuie îndeplinit pentru ca autorizarea să fie reușită. De exemplu, o cerință ar putea fi ca un utilizator să fie proprietarul unui cont pentru a transfera fonduri, sau că un utilizator trebuie să aibă un anumit rol sau permisiune pentru a accesa o resursă. O resursă, pe de altă parte, reprezintă obiectul sau datele la care se controlează accesul.

Pentru a se obține o înregistrare dintr-o bază de date este folosit adesea un identificator unic. Design-ul și structura bazei de date se bazează pe această cheie, iar uneori acestea pot fi ușor de ghicit sau enumerate. Atacatorii pot găsi acești identificatori în cererile către server-ul web. Dacă nu sunt protejate corespunzător cu controale de acces, un utilizator rău intenționat poate vizualiza, modifica sau, în cel mai rău caz, șterge aceste înregistrări. Această vulnerabilitate este reprezentată de referințele directe către obiectele nesigure.

Prezența referințelor directe către obiecte nesigure (IDOR)

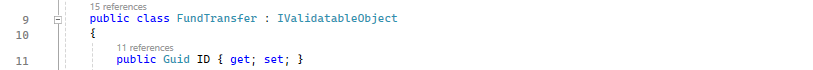
Când un utilizator se autentifică cu succes în contul său, poate naviga către pagina sa de transferuri de fonduri pentru a vizualiza detalii ale tranzacțiilor specifice, cum ar fi detaliile unui transfer cu ID-ul 1, ID identificat din URL-ul paginii. Un alt utilizator, cu intenții malițioase, observă faptul că ID-ul transferului este vizibil în URL, așa că poate încerca să acceseze detaliile transferurilor celorlalți utilizatori. Acest scenariu indică o vulnerabilitate de securitate unde controalele de acces nu sunt aplicate corespunzător, permițând unui utilizator să acceseze detalii private ale tranzacției altui utilizator fără autorizație. Această încălcare a confidențialității subliniază necesitatea critică a mecanismelor de autentificare și autorizare securizate pentru a proteja datele și tranzacțiile utilizatorilor.

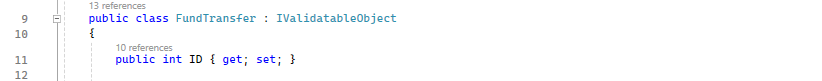




Testul precedent arată că această pagină este susceptibilă la o vulnerabilitate de securitate de tip IDOR. Pentru a remedia vulnerabilitatea IDOR din cod va fi adăugată o verificare a utilizatorului pentru a stabili dacă un anumit utilizator poate vedea pagina cu detaliile transferului de fonduri.

Primul pas pentru a mitiga această vulnerabilitate este schimbarea proprietății ID de la tipul int la tipul Guid pentru ca identificatorii să fie mult mai dificil de ghicit de către atacatori. Guid-urile sunt recunoscute pentru unicitatea și complexitatea lor, reducând astfel riscul vulnerabilităților IDOR prin ghicirea sau enumerarea secvențială a ID-urilor.

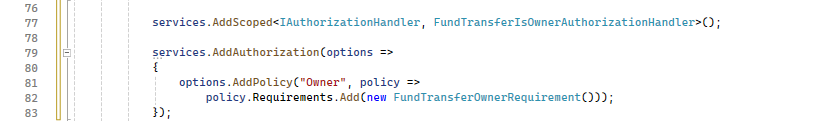




Pentru a implementa o politică de autorizare cu cerința ca cererea să fie rezolvată dacă utilizatorul este proprietarul înregistrării transfer de fonduri, a fost creată o clasă de tip Authorization Handler, a cărei metodă HandleRequirement determină dacă ID-ul clientului asociat transferului de fonduri corespunde cu ID-ul utilizatorului care a trimis cererea. Pentru fiecare cerință a unei politici trebuie definită o astfel de clasă. Dacă unica cerință definită în acest caz este îndeplinită, va fi apelată metoda Succeed ce indică acest lucru, iar accesul la resursa FundTransfer va fi permis. Altfel, accesul va fi refuzat. Prin apelarea metodei Succeed pentru o anumită cerință, se declară explicit că această cerință specifică a fost îndeplinită. Acest lucru este esențial pentru ca sistemul să știe care cerințe au fost îndeplinite și care nu.



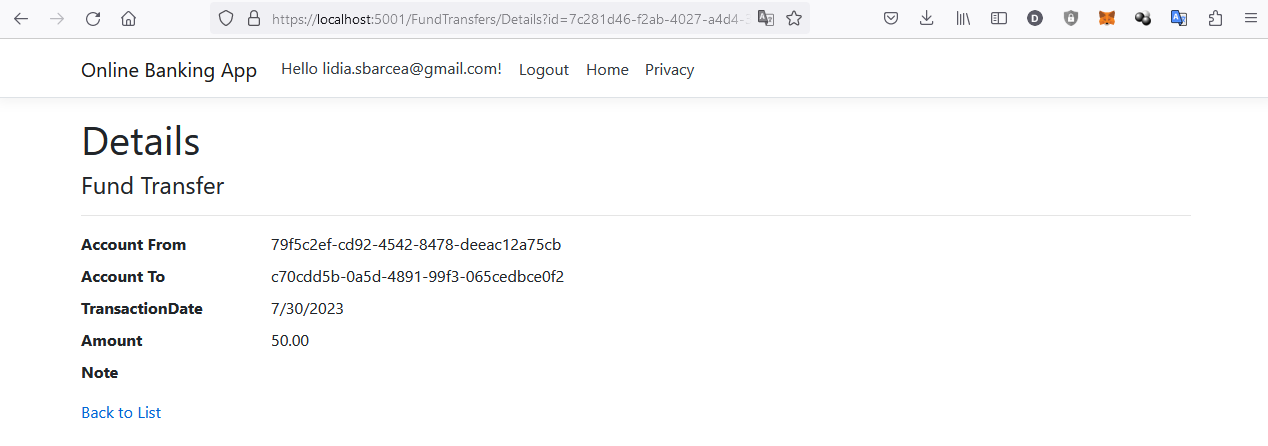
Pentru ca acest Authorization Handler să poată fi utilizat de sistemul de autorizare prin DI pe durata de viață a cererii și pentru a adăuga o politică preconfigurată, au fost folosite metodele AddScoped și AddAuthorization, respectiv AddPolicy în metoda ConfigureServices. Adăugarea politicii ”Owner” definește cerința specifică care trebuie îndeplinită pentru autorizarea accesului la resurse.

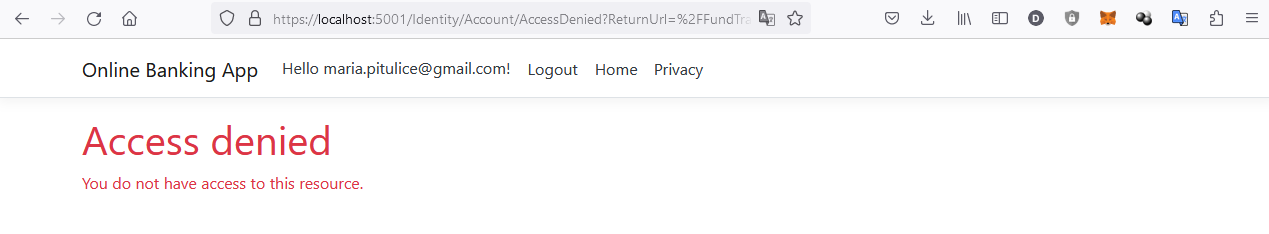


Pentru a utiliza serviciul de autorizare în pagina de detalii acesta trebuie injectat în constructorul paginii. Acesta poate fi folosit ulterior pentru a verifica dacă utilizatorul curent îndeplinește cerințele politicii de autorizare și a acorda acces la resursa solicitată. Când este apelată metoda AuthorizeAsync, serviciul de autorizare va procesa utilizatorul dat în funcție de politica specificată și resursa furnizată. Acesta va rula toate cerințele configurate pentru politica "Owner". Dacă toate cerințele sunt îndeplinite, rezultatul va avea proprietatea Succeeded setată pe true. Dacă vreo cerință nu este îndeplinită, Succeeded va fi false, și accesul la resursă nu va fi acordat.



Simulând din nou scenariul descris inițial se poate observa că a fost implementată o metodă robustă de autorizare folosind abordarea bazată pe politici.

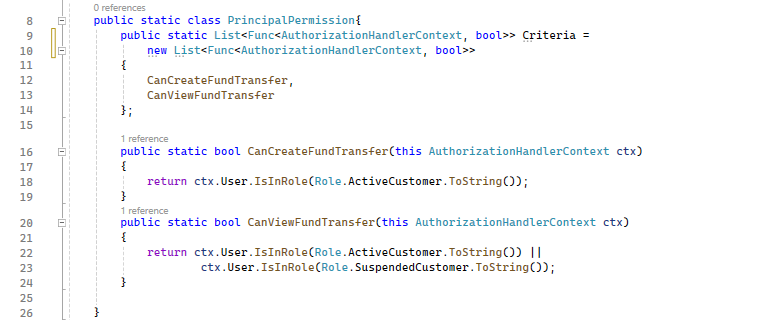




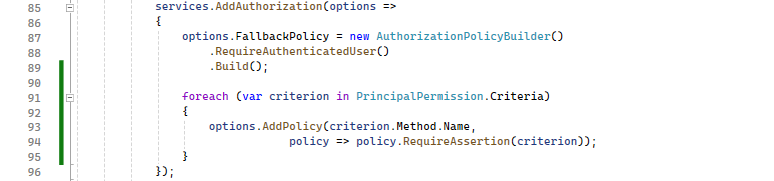
Absența autorizării

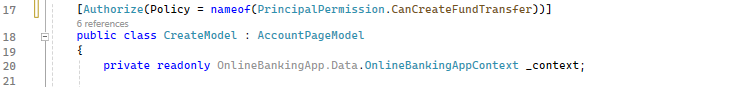
Orice utilizator autentificat poate crea un transfer de fonduri. Pentru a preveni accesul neautorizat la resurse în aplicație, remedierea lipsei controlului de acces poate consta în implementarea unei autorizări bazate pe politici cu criterii definite pe baza rolurilor pe care le are utilizatorul. Pentru acest lucru, trebuie creat un nou model care să poată reprezenta permisiunile pentru un principal (adică un utilizator sau o altă entitate capabilă să dețină roluri).

A fost definită o clasă PrincipalPermission cu o listă Criteria care conține funcții utilizate pentru a determina dacă un utilizator îndeplinește anumite criterii pe baza rolurilor sale. În această clasă au fost adăugate două metode statice: CanCreateFundTransfer, ce verifică dacă utilizatorul are rolul ActiveCustomer și CanViewFundTransfer, ce verifică dacă utilizatorul are rolul ActiveCustomer sau SuspendedCustomer.



Pentru fiecare criteriu parcurs din lista PrincipalPermission.Criteria este adăugată o politică de autorizare, iar numele politicii este derivat din numele metodei criteriului, politica fiind configurată să necesite îndeplinirea acelui criteriu.

În acest fragment de cod, atributul Authorize este aplicat clasei CreateModel, asociată paginii de creare a unui transfer de fonduri. Atributul Authorize specifică faptul că accesul la această pagină este controlat de o politică, iar politica la care se referă este PrincipalPermission.CanCreateFundTransfer.



Dacă înainte de utilizarea acestei politici, atât utilizatorii activi cât și cei suspendați puteau crea un transfer de fonduri, în cazul curent doar utilizatorii care sunt activi mai pot face acest lucru.

