Technická dokumentácia pre autentifikáciu používateľov, správu profilu - Administračný portál

Verzia 06.05.2016

Tabuľka 1. Autori

Autor	Rola
Peter Vrana	

Tabuľka 2. História zmien

Verzia	Dátum	Autor	Popis
1.0	17.11.2015	Peter Vrana	Vytvorenie dokumentu
1.1	17.11.2015	Peter Halaš	Pridanie zmeny a resetu hesla
1.2	02.03.2016	Peter Vrana	Autentifikácia a autorizácia v administračnom portáli
1.3	06.05.2016	Peter Halaš	Správa aliasov

Obsah

1 Úvod 1
2 Analýza 2
3 Návrh riešenia 3
3.1 Návrh prihlasovania 3
3.2 Návrh zmeny hesla 5
3.3 Návrh resetu hesla 5
3.4 Návrh úpravy profilu 5
3.5 Návrh autentifikácie a autorizácie 5
3.6 Návrh spravovania aliasov a služieb 6
4 Implementácia 7
4.1 Implementácia prihlasovania 7
4.2 Implementácia zmeny hesla 8
4.3 Implementácia resetu hesla 9
4.4 Implementácia úpravy profilu 9
4.5 Implementácia autentifikácie a autorizácie 10
4.6 Implementácia spravovania aliasov a služieb 10
5 Testovanie 12

1 Úvod

V tomto dokumente je opísaná funkcionalita týkajúca sa autentifikácie používateľov a správy profilu v Administračnom portáli.

2 Analýza

Architektúra projektu PerConIK pozostáva z množstva modulov, z hľadiska zvýšenia bezpečnosti je potrebné, aby k niektorým modulom mohli pristupovať autentifikovaný používatelia. Niektoré moduly obsahujú vlastné mechanizmy na správu používateľov. Pre zabezpečenie konzistentnosti a zníženie redundancie kódu je potrebné vytvoriť centrálny administračný portál, kde by bola sústredená celková správa používateľov. Framework .Net MVC poskytuje rôzne podporné mechanizmy, ako zabezpečiť autentifikáciu používateľov napr. Identity Framework, Forms Autentifikácia, autentifikačné atribúty.... Je potrebné zvoliť vhodný mechanizmus vyhovujúci naším potrebám a použiť ho pri autentifikácii používateľov. Pri autentifikácii je požadovaná nasledujúca funkcionalita:

- Verifikácia prihlasovacích údajov cez AIS LDAP
- Možnosť zmeny hesla
- Možnosť resetu hesla na pôvodné AIS heslo

Súčasťou tohto modulu je aj správa profilu používateľov, v rámci správy profilu má používateľ možnosť upraviť:

- Meno
- Priezvisko
- Email

Používateľ si môže spravovať aliasy. Alias je tvorený dvojicou: host domain časť URL adresy a email používateľa. Takto definovaný alias musí byť unikátny pre všetkých používateľov systému. Používateľ má možnosť pridať do doménovej adresy aj celú URL adresu, z ktorej sa do databázy uloží len host domain danej URL adresy.

Vystavené sú dve služby, ktoré:

- Vrátia meno používateľa na základe url a emailu.
- Vrátia zoznam emailov pre daného používateľa a danú url.

3 Návrh riešenia

3.1 Návrh prihlasovania

Aby bola zabezpečená požadovaná funkcionalita opísaná v analýze, rozhodli sme sa prihlasovanie navrhnúť nasledovne. Používateľ zadá prihlasovacie meno a heslo – prihlasovacie údaje do AIS, následne sa overí, či používateľ so zadaným menom existuje v databáze.

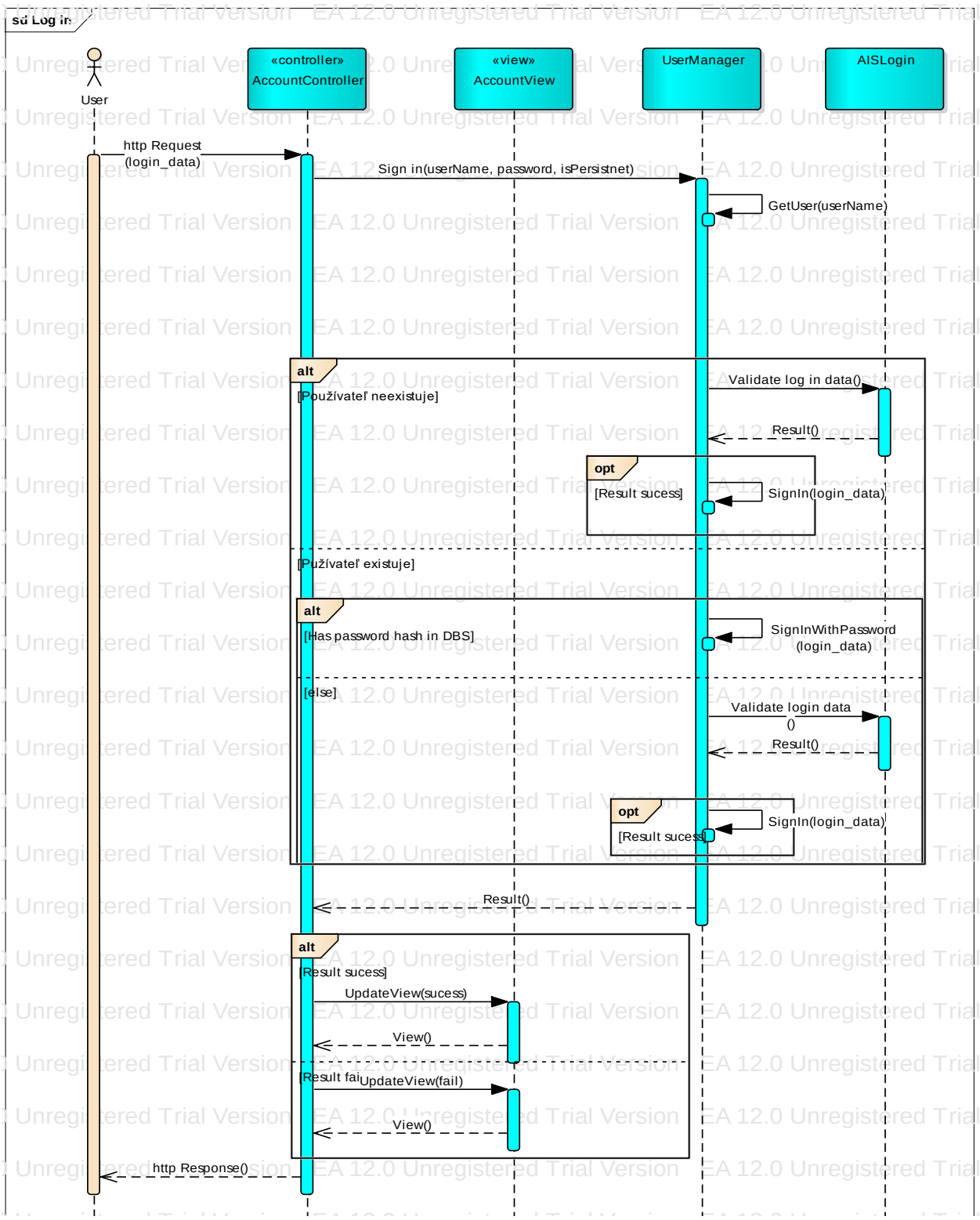
Ak používateľ existuje v databáze overí sa, či je v databáze uložený hash jeho hesla:

- Ak nie overenie prihlasovacích údajov prebieha prostredníctvom AIS LDAP.
    - Ak overenie prebehne korektne používateľ je v stave prihlásený, čo mu umožňuje pristupovať k vybraným funkciám systému a k úprave profilu.
    - Ak nie dôjde k výpisu chybovej hlášky a presmerovaniu na prihlasovaciu obrazovku.
- Ak áno overenie prebieha porovnaním zadaného hesla a hesla v databáze, pričom opäť platí, že ak používateľ zadal platné heslo bude prihlásený, ak nie bude presmerovaný späť na prihlasovaciu obrazovku s chybovou hláškou.

Ak používateľ neexistuje v databáze overia sa prihlasovacie údaje voči AIS LDAP

- Ak používateľ zadal správne údaje, vytvorí sa preň v databáze záznam a zároveň bude prihlásený.
- V prípade, že overenie voči AIS LDAP zlyhá používateľ je presmerovaný na prihlasovaciu obrazovku a zobrazí sa chybová správa.

Na obrázku 3.1 je zachytený proces prihlasovania prostredníctvom sekvenčného diagramu.



Obrázok 3.1 Sekvenčný diagram – prihlásenie používateľa

3.2 Návrh zmeny hesla

Používateľ si na stránke svojho profilu môže nastaviť lokálne heslo v prípade, že sa prihlasuje pomocou údajov do AIS, pretože v takomto prípade nemá v databáze uložené heslo. Táto funkcionalita je dôležitá a odporúča sa každému novému používateľovi, nastaviť si lokálne heslo, aby sa predišlo problémom spojeným s nedostupnosťou služby AIS LDAP. Vytvorenie lokálneho hesla môže vykonať len prihlásený používateľ a je potrebné aby zadal iba nové heslo.

Používateľ si na stránke svojho profilu môže zmeniť lokálne heslo. Podmienky pre vykonanie tejto akcie sú: používateľ musí byť prihlásený a musí mať nastavené lokálne heslo. Od používateľa sa vyžaduje zadať aktuálne lokálne heslo a nové lokálne heslo.

Na obrázku 3.2 sú zachytené triedy: *SetPasswordModel* a *ChangePasswordModel*, ktoré súvisia s vytvorením a zmenou lokálneho hesla.

3.3 Návrh resetu hesla

Používateľovi sa po prvom neúspešnom prihlásení zobrazí možnosť resetovania hesla. Po zvolení tejto možnosti používateľ zadá prihlasovacie meno a heslo k AIS. Podľa toho či zadá správne alebo nesprávne údaje bude používateľ informovaný o úspechu / neúspechu operácie.

3.4 Návrh úpravy profilu

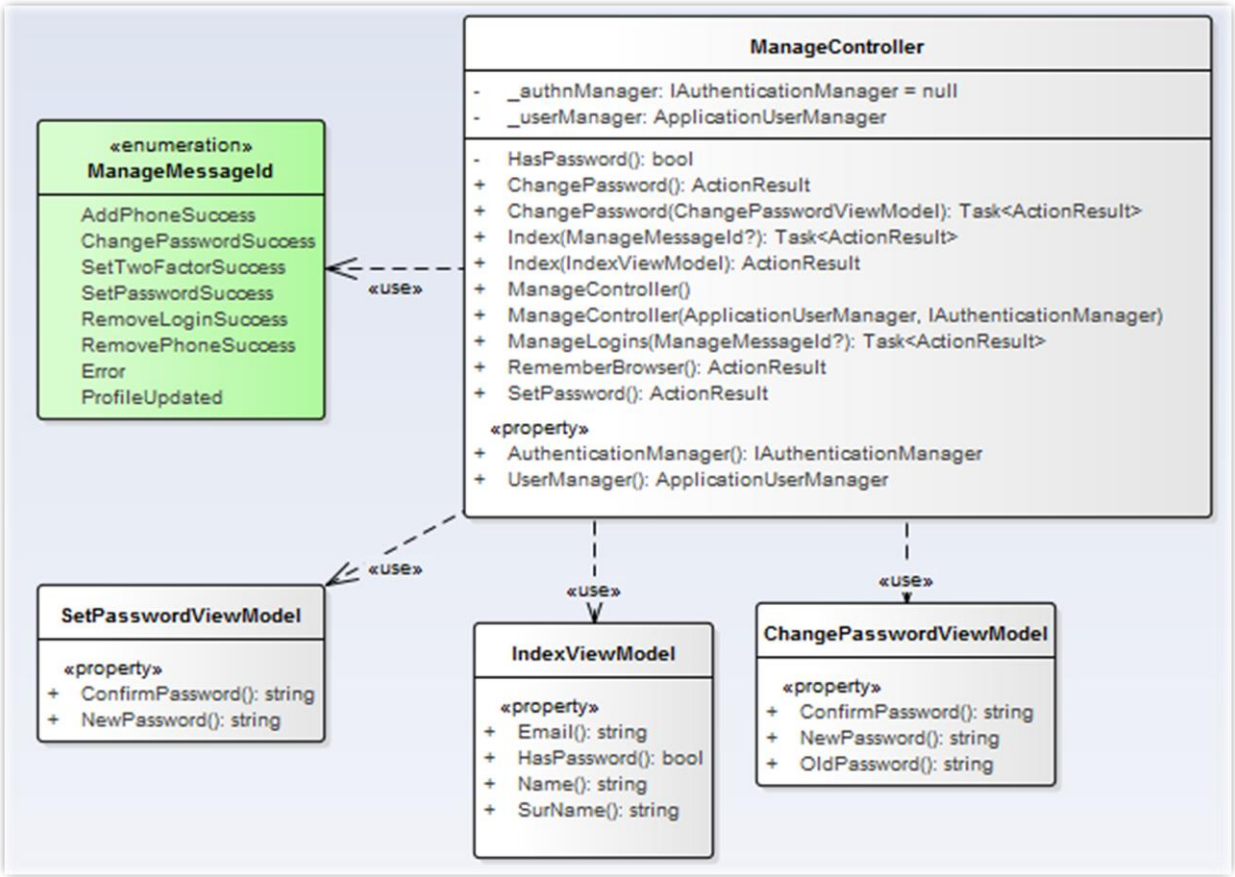
Úprava profilu používateľa bola navrhnutá nasledovne. Používateľ po prihlásení do systému uvidí v pravej hornej časti obrazovky svoje prihlasovacie meno. Po kliknutí na toto meno sa používateľovi sprístupní obrazovka na ktorej sa zobrazí jeho aktuálne meno, priezvisko a e-mail v rámci systému. Používateľ má možnosť tieto položky upravovať. Po dokončení úprav používateľ potvrdí svoje úpravy tlačidlom a následne sa tieto úpravy uložia do databázy. Ak pri ukladaní dôjde k chybe používateľ je oboznámený o tejto chybe prostredníctvom chybovej hlásky. V prípade, že uloženie zmien prebehne bez problémov používateľ je oboznámený o úspechu operácie prostredníctvom správy na obrazovke. Na obrázku 3.2 sú zachytené najdôležitejšie triedy, ktoré sa využívajú na úpravu profilu a úpravu hesla.

3.5 Návrh autentifikácie a autorizácie

Na autentifikáciu a autorizáciu používateľa bol použitý upravený autorizačný atribút, nebol použitý štandardný autorizačný atribút pretože, bolo potrebné zapracovať autorizáciu do existujúceho návrhu kde sa roly ukladali na špeciálne miesto v rámci kolekcie používateľov, preto nebolo možné použiť štandardný .Net MVC RoleProvider. Autorizácia bola navrhnutá tak, že používateľ v role „admin“ má prístup ku všetkým funkciám aplikácie, používateľ v role „read“ má

prístup k zmene vlastných údajov a používateľ v role „deny“ má rovnaké práva ako neprihlásený používateľ.

Návtr



Obrázok 3.2 Diagram tried pre úpravu profilu a hesla

3.6 Návrh spravovania aliasov a služieb

Najväčšou výzvou bolo zabezpečenie unikátnosti aliasov pomocou atomických operácií Monga. Pôvodný návrh bolo pridanie poľa dvojíc host domain z url a email. No pre zabezpečenie unikátnosti sa nakoniec vytvorila nová kolekcia *aliasses*, ktorá obsahuje trojicu hodnôt:

- Host domain z url
- Email
- Meno používateľa

Používateľ má mať možnosť pridávať a mazať vlastné aliasy.

Je potrebné vystaviť dve služby, ktoré:

- Vrátia meno používateľa na základe url a emailu.
- Vrátia zoznam emailov pre daného používateľa a danú url.

Pričom url môže byť URL adresa alebo priamo host domain.

4 Implementácia

4.1 Implementácia prihlasovania

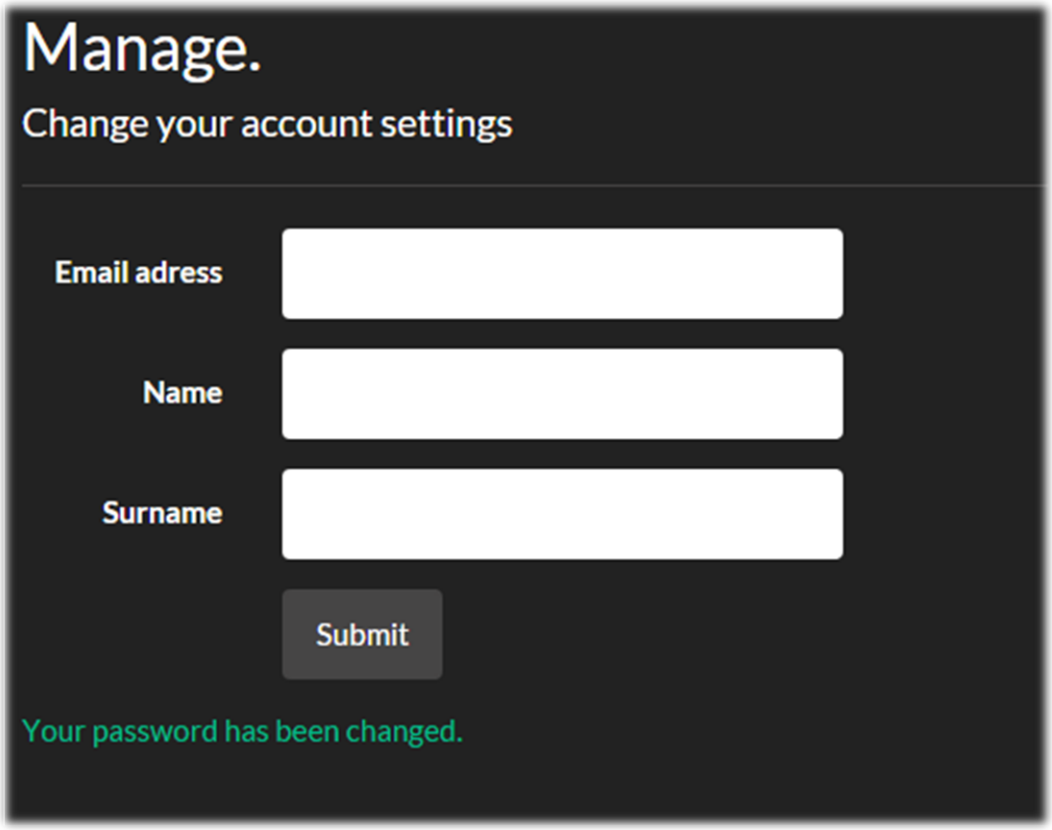
Pri implementácii prihlasovania do systému sme postupovali podľa návrhu. Pre prihlásenie sa využíva Identity framework, ktorý je súčasťou .Net MVC. Využitie tohto frameworku značne uľahčilo implementáciu prihlasovania, ale zároveň spôsobilo drobné implementačné komplikácie. Vzhľadom na to, že Identity framework ako väčšina architektúry .Net MVC je prispôsobená najmä na podporu relačných databáz, bolo potrebné vykonať značné úpravy, aby bolo možné dáta ukladať do Mongo databázy. Na obrázku 4.1 je zachytená najdôležitejšia metóda PasswordSignIn, v ktorej sa nachádza najväčšia časť logiky prihlasovania. Pre overenie používateľov sa ďalej v aplikácii budú používať autorizačné atribúty.

```
public async Task<SignInStatus> PasswordSignIn(string userName, string password, bool isPersistent, bool shouldLockout)
{
    var user = await UserManager.FindByNameAsync(userName);
    if (user == null)
    {
        if (aisVerificator.AisVerification(userName, password))
        {
            user = new ApplicationUser { UserName = userName };
            var identityResult = await UserManager.CreateAsync(user);

            if (identityResult != IdentityResult.Success)
            {
                return SignInStatus.Failure;
            }

            return await SignInOrTwoFactor(user, isPersistent);
        }

        return SignInStatus.Failure;
    }

    if (await UserManager.IsLockedOutAsync(user.Id))
    {
        return SignInStatus.LockedOut;
    }

    if (user.PasswordHash == null)
    {
        if (aisVerificator.AisVerification(user.UserName, password))
        {
            return await SignInOrTwoFactor(user, isPersistent);
        }
    }
    else if (await UserManager.CheckPasswordAsync(user, password))
    {
        return await SignInOrTwoFactor(user, isPersistent);
    }
    return SignInStatus.Failure;
}
```

Obrázok 4.1 Metóda PasswordSignIn v triede ApplicationUserManager

4.2 Implementácia zmeny hesla

Táto implementácie nebola príliš náročná, keďže .Net MVC obsahoval metódy pre vykonanie potrebných vecí. Bolo potrebné prispôsobiť grafické rozhranie pre dané potreby. Na obrázku 4.2 je zobrazené používateľské rozhranie po úspešnej zmene hesla.



Obrázok 4.2 Používateľské rozhranie pre zmenu hesla

4.3 Implementácia resetu hesla

Prebehla bez väčších komplikácií. Rovnako aj v tejto úlohe bola nachystaná metóda vďaka .Net MVC. Bolo potrebné zmeniť logiku tejto metódy, aby sa lokálne heslo resetovalo overenými prihlasovacími údajmi do AIS.

4.4 Implementácia úpravy profilu

Implementácia úpravy profilu prebehla podľa návrhu bez komplikácií. Táto implementácia bola do veľkej miery zjednodušená, pretože .Net MVC už obsahoval niektoré prvky potrebné pre úpravu profilu, tie bolo potrebné iba jednoducho upraviť. Na obrázku 4.3 je zachytené používateľské rozhranie pre úpravu profilu.

Obrázok 4.3 Používateľské rozhranie pre úpravu profilu

4.5 Implementácia autentifikácie a autorizácie

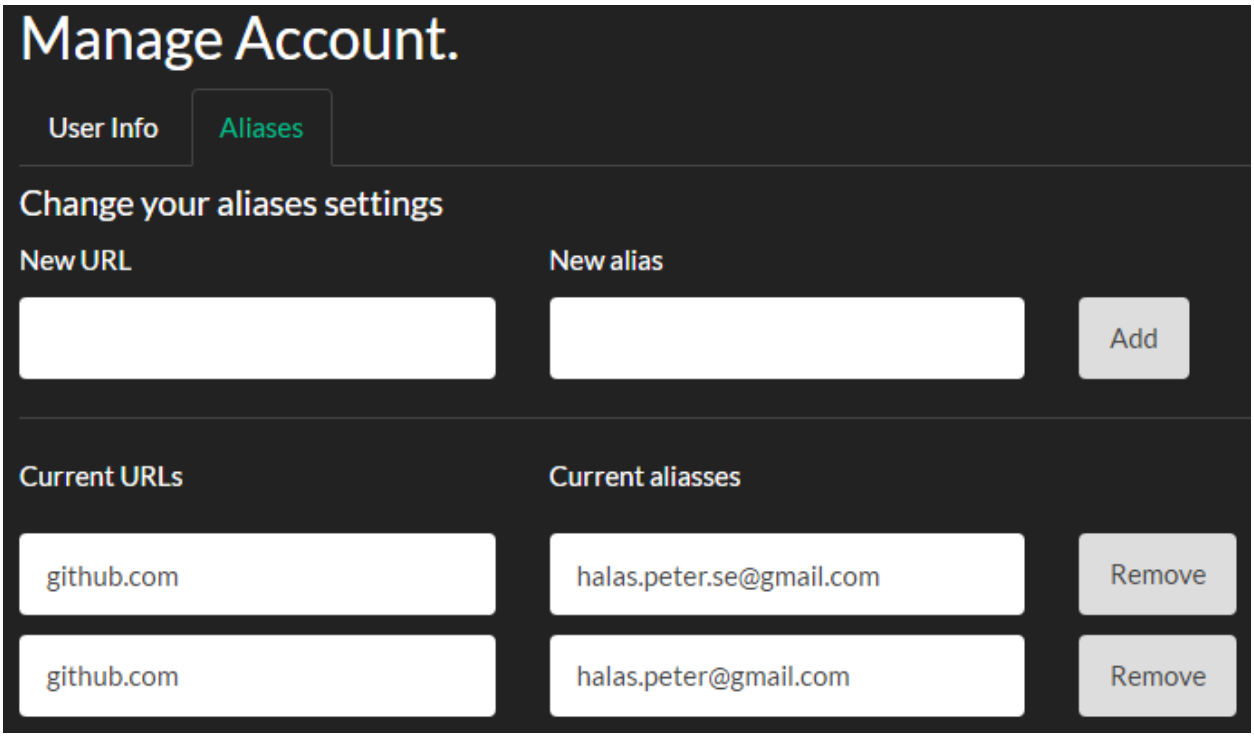
Na autentifikáciu používateľov bol použitý štandardný autorizačný atribút [*Authorize*]. Na autorizáciu používateľov bol vytvorený vlastný autorizačný atribút [*DevActsAdminPortalAuthorizationAttribute*], jadro samotnej autorizácie je zachytené na obrázku 4.4. funkcia je samo opisná a nevyžaduje si ďalšie vysvetlenie... Všetky funkcie vyžadujúce prístup autorizovaného používateľa boli následne týmto atribútom dekorované spolu so zoznam rolí, ktoré majú k danej funkcionalite prístup. Pridanie default admina bolo vyriešené prostredníctvom dátovej migrácie, ktorá pridala do databázy používateľa s rolou admin vzťahujúcou sa na tento modul trieda *Migration5*.

```
protected override bool AuthorizeCore(HttpContextBase httpContext)
{
    if (httpContext.User.Identity.IsAuthenticated)
    {
        var user = UserManager.FindByName(httpContext.User.Identity.Name);

        if (user == null)
            return false;

        var repos = _repositoryManager.UserManagmentContex.GetAllRepositories();

        string role = _repositoryManager.UserManagmentContex.GetRepositoryRightForUser(user.UserName, "DevActs");

        return Roles.Split(',').Contains(role);
    }
    return false;
}
```

Obrázok 4.4 Jadro autorizačného atribútu DevActsAdminPortalAuthorizationAttribute.

4.6 Implementácia spravovania aliasov a služieb

Implementácia rozhrania na správu používateľov je spravená pomocou jazyka Javascript. Po kliknutí používateľa na svoj profil, kde si môže meniť osobné údaje, má možnosť prepnúť sa na správu aliasov. Na obrázku 4.4 je znázornené používateľské rozhranie, kde si môže používateľ pridať nový alias alebo vymazať aktuálne.



Obrázok 4.4 Používateľské rozhranie pre správu používateľov

Api controler poskytuje dve REST služby:

- GetAliasesByUserNameAndUrl – vráti všetky aliasy pre daného používateľa a danú url (buď URL adresa alebo host domain adresa).
- GetUserByAlias – vráti meno používateľa podľa zadaného url a emailu (url parameter je buď URL adresa alebo host domain adresa).

5 Testovanie

Boli vytvorené unit testy, ktoré testujú či sa *AccountController* správa korektne pri prihlasovaní používateľa, pričom sa testovalo prihlásenie s použitím AIS LDAP verifikácie aj prihlásenie s overením hesla voči databáze. Taktiež boli vytvorené testy pre úspešný a neúspešný reset hesla. Všetky vymenované testy sa nachádzajú v projekte AdministrationPortal.Tests v triede *AccountControllerTest* v metódach *LogInWithLDAP*, *LogInWithoutLDAP*, *ResetPasswordSucessfully*, *ResetPasswordFailed*. Pre autentifikáciu a autorizáciu boli vytvorené testy *AuthorizeCoreRoleCheckTest*, *AuthorizeCoreNotAuthenticatedTest*, *AuthorizeCoreUserIsNull*. Pre spravovanie aliasov boli vytvorené testy: *GetAlliassesByUserAndUrl*, *GetAlliassesByUserAndUrlFail*, *GetUserByAlias* a *GetUserByAliasFail*.