TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Informaatikainstituut

IDU40LT
Eerik Mägi 112156IAPB

ERALDISEISEV ISETEENINDUSLIK PAROOLIDE VAHETAMISE VEEBITARKVARA OLEMASOLEVATE RAKENDUSTE JAOKS

bakalaureusetöö

Juhendaja: Erki Eessaar
Doktorikraad
Dotsent

Tallinn 2016

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Eerik Mägi

23.05.2016

Annotatsioon

Käesoleva töö eesmärgiks on disainida ning realiseerida veebipõhise kasutajaliidesega rakendus, mida saab kasutada olemasolevate süsteemidega ning mis võimaldab nende süsteemide kasutajatel iseteeninduslikult vahetada oma paroole ilma administraatori sekkumiseta sõltumata sellest, kas nad teavad oma hetkel kehtivat parooli või mitte. Tarkvaral on ka eesmärk aidata ja julgustada kasutajaid kasutama tugevaid paroole. Selleks peab rakendus olema võimeline mõõtma kasutaja sisestatud parooli tugevust ning vastavalt administraatori seadistusele paroolide minimaalse tugevuse nõudeid jõustama. Rakenduse idee pärineb faktist, et TTÜ Informaatikainstituudi ruumide broneerimise süsteemi kasutajatel puudub võimalus oma ununenud parooli vahetada.

Töö tulemusena loodi kasutusvalmis veebirakendus, mis täidab eelpool mainitud eesmärke. See veebirakendus avaldati GitHub-i keskkonnas avatud lähtekoodiga ning on alla laetav järgneval aadressil.

https://github.com/Pisi-Deff/PasswordManager

Sellel leheküljel asub ka rakenduse lühike ingliskeelne tutvustus ja kasutusjuhend. Veebirakenduse lähtekood avaldati MIT litsentsi all. Rakendus seati üles mitme reaalse süsteemi jaoks ning töötas vastavalt eesmärkidele.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 42 leheküljel, 7 peatükki, 11 joonist.

Abstract

A Standalone Self Service Web Application for Changing Passwords in Existing Applications

The goal of this thesis is to design and build an application with a web based user interface, that one can use together with existing systems. It must enable the users of those systems to change their passwords in said systems by themselves without the involvement of administrators. It must be possible regardless of whether they remember their currently active password or not. The software also has the goal of helping and encouraging users to use strong password. To accomplish this, the application has to be able to measure the strength of passwords entered by the user and enforce a minimal password strength depending on the configuration of the application. The idea for such an application stems from the fact that the room reservation system of the Informatics Institute of Tallinn University of Technology does not provide its users with the ability to change their forgotten passwords.

As a result of this thesis, a complete and ready to use web application was created, which fulfills the goals described above. The author published this software on the GitHub service as open source under the MIT license. It is downloadable from the following address.

https://github.com/Pisi-Deff/PasswordManager

The page also contains a short introduction and guide in English. The software was set up for multiple real systems and worked as expected, fulfilling the goals of this thesis.

The thesis is in Estonian and contains 42 pages of text, 7 chapters, 11 figures.

Lühendite ja mõistete sõnastik

AJAX	**Asynchronous JavaScript and XML / Asünkroonne JavaScript ja XML** Brauseris JavaScripti koodis XMLHttpRequest objekti kasutamine, et asünkroonselt andmeid vahetada serveril töötavate skriptidega [1]. Seda kasutades saab näiteks lehekülje infot uuendada või leheküljele infot juurde lisada ilma lehekülge täielikult taasavamata.
Avatekst	**Plaintext** „Sõnum enne krüpteerimist või pärast dekrüpteerimist, s. t. tavaline tekst, mida igaüks võib lugeda. Avateksti vastandiks on šiffertekst.“ [2]
Eessüsteem	**Front end** „Veebilehele ilmuv kasutajaliides, mis võimaldab veebisaidi külastajal kahepoolselt suhelda saidi dünaamiliste osadega nagu andmebaasid, ostukorviprogrammid ja onlain-ostutöötlustarkvara.“ [2]
Hostinimi	**Host name** „Võrkuühendatud arvutile omistatud unikaalne nimi, mille järgi võrk selle arvuti e-posti, uudisgruppide jms elektroonilise infovahetuse puhul ära tunneb.“ [2] Näiteks „apex.ttu.ee“ või erijuht „localhost“, mis vastab alati kohalikule arvutile [2].
PHP	**PHP: Hypertext Preprocessor** Populaarne üldotstarbeline skriptimiskeel, mis on eriti sobiv veebiarenduseks [3].
SQL	**Structured Query Language / Struktureeritud päringukeel**

	Relatsioonilisel andmemudelil põhinev andmebaasikeel andmestruktuuride, neid ümbritsevate andmebaasiobjektide, andmete ja pääsuõiguste haldamiseks [2].
Sõrestikmudel	**Wireframe**
	„Kest, mis näitab loodavaid veebilehti. Iga leht on joonistatud informatsiooniplokkidena, mis näitavad kus ja kuidas paiknevad erinevad komponendid. Selliselt esitatakse objektide visuaalne paigutus (layout) kuid mitte lõplik välimus.“ [4]
Süntaks	**Syntax**
	„Mingis keeles moodustatud lause struktuuri määravad reeglid. Süntaks määrab ära, kuidas sõnad ja sümbolid peavad lauses olema kokku pandud.“ [2]
Teek	**Library**
	„Teek on kollektsioon funktsioone, makrosid, klasse vms komponente, mis on mõeldud korduvkasutuseks programmides.“ [5]

Sisukord

Jooniste loetelu

1 Sissejuhatus

Paroolid kipuvad ununema ning nende vahetamine on kasutajate haldurite jaoks lõppematu ülesanne. Tarkvara, mis võimaldab paroole iseteeninduslikult vahetada, aitab haldurite töökoormust suurel määral vähendada.

See probleem esineb näiteks TTÜ Informaatikainstituudi ruumide broneerimise rakendusel ja TTÜ Informaatikainstituudi õppematerjalide süsteemil Maurus. Kumbki rakendus ei võimalda kasutajal taastada oma ununenud parooli. Autor otsis paroolide vahetust abistavat tarkvara ja leidis lisamooduleid konkreetsetele süsteemidele, kuid mitte head üldist lahendust. Kuna sellist funktsionaalsust vajavaid tarkvarasüsteeme on teisigi, siis tekkis mõte probleemi üldistada ning luua tarkvara, mida saaks kasutada koos erinevate teiste tarkvaradega.

Lõputöö eesmärgiks on luua eraldiseisev tarkvara, mis võimaldab olemasoleva rakenduse kasutajatel vahetada oma parool sõltumata sellest, kas nad mäletavad oma kehtivat parooli või mitte. See tarkvara peab olema võimalikult universaalne, lubades end seadistada erinevate rakenduste toetamiseks. Selleks peab arvestama näiteks parooli kaitsmise meetodiga, andmebaasi tüübiga, andmebaasisüsteemiga ja andmemudeli eripäraga. Turvalisuse tõstmiseks peab tarkvara olema seadistatav laskmaks kasutajatel valida ainult tugevaid paroole. Oluline on ka, et tarkvara ei laseks end kuritarvitada, lubades nullida suvaliste kasutajate paroole. Tarkvara kasutamise eeldus on, et parool salvestatakse mingil kujul andmebaasisüsteemi abil loodud andmebaasi.

Töö elluviimiseks uuritakse turvalisuse ja paroolide temaatikal kirjandust ning analüüsitakse ja võrreldakse nende tulemusi, et välja selgitada kuidas määrata parooli tugevust, mis omadused tagavad parooli tugevuse ja milliseid turvalisust tagavaid omadusi on võimalik luua. Analüüsi tulemusel leitakse parim viis kuidas luua probleemi lahendav tarkvara. Samuti uuritakse olemasolevaid paroolide vahetamise mooduleid, et leida nendest ideid loodava tarkvara funktsionaalsuse jaoks. Tarkvara realiseeritakse ning seda võrreldakse ka juba loodud tarkvaradega. Tarkvara luuakse avatud lähtekoodiga ning tehakse maailmale avalikuks GitHubi keskkonna vahendusel.

2 Kasutajatuvastus

Inimestel tuleb tihti ette olukordi, kus on vaja piirata tundliku informatsiooni levikut või ligipääsu tundlikele esemetele ainult nendele isikutele, kellele seda tahetakse anda või kellele seda on vaja. Kui selliste isikute arv on väike ja kõik isikud on teada, siis piisab vaid isiku nägemisest, et teada, kas teda võib usaldada või mitte. Kui aga usaldatavaid isikuid on väga palju või ei ole kõik need isikud teada, siis on vaja välja mõelda abivahendid, mille abil saab isikut tuvastada.

2.1 Ajalugu

Paroole ja võtmesõnu on kasutatud juba vähemalt mitu tuhat aastat, et tuvastada, kas isik on sõber või vaenlane. Varaseim teadaolev kirjalik allikas selliste paroolide kohta on kirjutatud Kreeklasest ajaloolase Polybiuse poolt ning pärineb 2. sajandist e.m.a [6]. Selles kirjeldatakse muuhulgas kuidas valvurid sõnumeid vahetasid ja kuidas ühe paroolsõna ütlemise viisiga saab eristada sõpra ja vaenlast. Varaseim teadaolev kirjalik allikas, mis mainib sõnumite krüpteerimist kasutades parooli võtmena pärineb 16. sajandist. Selle krüpteeringu nimi oli Vigenère Šiffer [6].

Varajased arvutid ei kasutanud paroole, vaid toetusid füüsilisele turvalisusele [6]. Lühidalt võib sellest aru saada nii, et kui kellelgi oli füüsiline ligipääs ruumile, kus arvuti asus, siis oli tal ka õigus arvutit kasutada. Esimene arvuti, mis kasutas paroole oli ajaloolaste sõnul USA ülikooli Massachusettsi Tehnoloogiainstituut (ingl *Massachusetts Institute of Technology,* või lühidalt *MIT*) ajajagamise arvuti nimega *CTSS* [7]. Selle süsteemiga toimus ka esimene arvuti kasutajatuvastuse murdmine. Kuna kõikide kasutajate paroole hoiti ühes kaitsmata failis avatekstina, siis oli lihtne see fail välja printida ja teiste kasutajatena süsteemi siseneda [7]. Esimesed katsed arvutis hoitud paroole krüptida tehti 1970. aastatel, algul USA õhuvägede privaatsetes süsteemides ja hiljem UNIX-i operatsioonisüsteemis [7].

Kuna digitaalselt hoitavate ja käsitlevate andmete hulk aina suureneb, siis on arenenud nii digitaalsed andmete turvamise tehnoloogiad kui ka turvameetmeid murda tahtvate ründajate töövahendid.

2.2 Digitaalsed kasutajatuvastuse liigid

Järgnevalt kirjeldatakse põhilisi kasutajatuvastuse vahendeid, mida kasutatakse digitaalseks autentimiseks.

2.2.1 Kasutajanimi ja parool

Nagu eelnevalt mainitud on parool üks ajalooliselt vanimaid viise, kuidas kasutajat tuvastada. Digitaalses maailmas käib parooli juurde enamasti ka kasutajat unikaalselt identifitseeriv nimetus ehk kasutajanimi. Kasutajanimi on tihtipeale võrdlemisi lühike kasutaja poolt valitud kirjamärgijada, ent olenevalt süsteemi looja tahtest võib selleks olla ka näiteks kasutaja meiliaadress, süsteemi administraatori poolt määratud kirjamärgijada või süsteemi poolt genereeritud kirjamärgijada.

Ainult kasutajanime ja parooliga autentimine on pealtnäha kasutaja jaoks lihtne ja mugav, meeles peab pidama vaid kahte ühikut informatsiooni. Samas tähendab see ka seda, et ründajal on sissemurdmiseks vaja omandada vaid kaks ühikut informatsiooni. Kui süsteeme on palju, siis muutub ka nende kahe informatsiooniühiku meelespidamine kasutajale suureks probleemiks. See võib kasutajat julgustada taaskasutada paroole mitmetes süsteemides ja/või kasutada liiga lihtsaid paroole (vt jaotis 3.2). Parool on kergesti jagatav kirjalikus vormis, mis on nii hea kui ka halb omadus. Mõnikord on väga vaja parooli teistega jagada, ent samas võivad üleskirjutatud parooli näha kõrvalised isikud.

2.2.2 Mitmekordne autentimine

Mitmekordse autentimise puhul küsitakse autentimiseks mitut erinevat kasutajat tuvastavat tõendit [8]. Primaarseks autentimisvahendiks on enamasti tavaline parooli ja kasutajanime kombinatsioon. Teiseseks autentimisvahendiks võib olla näiteks kasutaja meilile saadetav kood, kasutaja telefonile saadetud SMS, kasutaja telefonil töötava rakenduse poolt genereeritud kood või spetsiaalne koode genereeriv riistvaraline seade.

Mitmekordset autentimist kasutav süsteem on turvalisem kui ainult kasutajanime ja parooli kasutav süsteem, sest ründajal on sissemurdmiseks vaja rohkem andmeid [8]. Samas peab ka kasutaja autentimiseks sisestama rohkem informatsiooni, mis vähendab autentimise mugavust ja kiirust [8]. Suur puudus on ka tagavaravahendi vajadus autentimiseks juhuks kui kasutajal kaob ligipääs teisesele autentimisvahendile. Näiteks kui kasutaja kaotab telefoni, millel on koode genereeriv rakendus. Kui puudub tagavaravahend, siis võib see kasutaja jaoks tähendada tema väljalukustamist süsteemist.

Mitmekordset autentimist kasutab näiteks Eesti ID-kaart [9] ning on seadistatav näiteks Google-i kontode [10] ja Steami kontode [11] jaoks.

2.2.3 Privaatse ja avaliku võtme paar

Avaliku võtme krüptograafia (ingl *Public Key Cryptoraphy*) on 1970. aastatel leiutatud ja hiljem täiendatud krüptograafiline meetod [12]. Kasutatakse kahte võtit – üks avalik ja üks privaatne võti. Need võtmed genereeritakse korraga ning nad on omavahel seotud [13]. Avaliku võtme võib omanik muretsemata välja jagada, kuna seda ei ole võimalik kurjasti ära kasutada [13]. Privaatne võti peab aga olema ainult oma omanikule teada. Avalikku võtit kasutatakse krüpteerimiseks ja vastava privaatse võtmega antud signatuuride verifitseerimiseks. Privaatse võtmega saab vastava avaliku võtmega krüpteeritud infot dekrüpteerida ja infot allkirjastada.

Kui kasutatakse tugevaid võtmeid, siis on see väga võimas autentimisviis. Kahjuks ei ole aga loodud eriti palju tehnoloogiaid, mis võimaldaks tavakasutajatele sellise autentimisviisi mugavat kasutamist tavalistel veebilehekülgedel. Kõige lähemale jõuab sellele sihtmärgile Eesti riigi ID-kaart. ID-kaardiga autentimine on mugav: vajab vaid kaardilugemisseadet, füüsilist kaarti ja PIN koodi [9], mis on mitmekordse autentimise vahendina kasutatav lühike parool. ID-kaardiga autentimine ei ole anonüümne, vaid kaardil olevad võtmed vastavad kindlale Eesti riigi kodanikule, Eestis püsivalt eleavale välismaalasele või e-residendile [9], mis tähendab, et sellega autentides ei ole võimalik jääda süsteemis anonüümseks kasutajaks.

2.2.4 Integratsiooni kasutamine

Integratsiooni kasutamine kasutajate autentimiseks tähendab, et kasutaja autentimise kohustus antakse edasi teisele süsteemile. Sellist teenust pakuvad veebilehtede jaoks suurtest firmadest näiteks Google [14] ja Facebook [15]. Integratsiooni kasutamise puhul

tekib süsteemi kasutajal võimalus ennast autentida integratsiooni teenuse pakkuja kaudu ning tolle kohustus on siis süsteemile kinnitada, et kasutaja on tõesti see, kes ta väidab end olevat. Google-i integratsioon pakub süsteemile veel rohkem mugavusi. Näiteks võib süsteem kasutaja loal kasutaja Google-i kontol faile kirjutada ja lugeda [14].

Kasutaja peab selle autentimisvahendi kasutamise puhul meeles hoidma vähem autentimise andmeid, mistõttu on see talle väga mugav.

Selle autentimisvahendi suurim puudus on, et süsteem sõltub integratsiooni teenuse pakkujast ning peab seda usaldama. Kui keegi murrab teenusepakkuja süsteemi sisse, siis saadakse ligipääs ka kõigile seda kasutatavatele süsteemidele. Paljude süsteemide autentimise eest vastutavad teenused on ka olulisemad ründesihtmärgid ründajatele. Probleeme nii kasutajatele kui ka süsteemile võib tekitada ka teenusepakkuja otsus mingil põhjusel lõpetada integratsiooni teenuse pakkumine. Eriti kui seda kasutatakse ainsa autentimisvahendina.

3 Turvaline parool

Järgnevas peatükis kirjeldatakse tugeva parooli omadusi.

3.1 Parooli entroopia

Entroopia on informatsiooniteoorias statistiline parameeter, mille defineeris Claude E. Shannon. Sellega mõõdetakse, kui palju informatsiooni iga tähemärk tekstis toodab [16], ehk teisisõnu kui mitu bitti iga sümboli kohta on minimaalselt vaja, et informatsiooni kodeerida. Selle parameetri ühikuks on bitt. Shannon defineeris entroopia valemiga (1) [17].

$$H(X) = -\sum_{i=1}^{n} p(x_i) log_b\, p(x_i) \quad (1)$$

Valemis on H entroopia tähemärgi kohta bittides, X on võimalike sümbolite hulk, x_i on i-s sümbol hulgas X, $p(x_i)$ on i-nda sümboli esinemise tõenäosus informatsioonis ning b on baas [17]. Terve sõna entroopia saamiseks tuleb entroopia tähemärgi kohta korrutada tähemärkide arvuga sõnas. Kahendarvude puhul on baasiks 2. Sümboli esinemise tõenäosus arvutatakse jagades selle esinemiste arv sõnas sõna pikkusega. Näiteks on Shannoni entroopia paroolil „password“ 24 bitti ja paroolil „subtext-thickly-ambergris-coincident“ 180 bitti. Parooli ära arvamiseks kuluks jõhkra jõu ründe puhul maksimaalselt 2^n katset, kus n on entroopia bittides, ja keskmiselt poole vähem katseid [17].

Parooli äraarvatavus ei sõltu alati aga ainult sellest, kui kiiresti arvuti kõik võimalikud kombinatsioonid läbi käia suudab, vaid ka kontekstuaalsetest omadustest. Näiteks on parool „password“ nii tihti kasutatud ja hästi tuntud, et arvatavasti on see üks esimesi, mida ründaja proovib. Seetõttu on selle parooli reaalne entroopia arvutatust palju väiksem. Seoses sellega peab ka mainima, et selles töös väljatoodud näidisparoolide reaalne entroopia on üles märgitust väiksem, kuna neid on spetsiifiliselt mainitud. Seetõttu pole soovitatav neid kasutada paroolidena.

USA Riiklik Standardite ja Tehnoloogia Instituut (ingl *National Institute of Standards and Technology*, või lühidalt *NIST*) soovitas 2004. aastal paroolide minimaalseks entroopiaks kasutada 80 bitti [16].

3.2 Nõrga parooli omadused

Nõrk parool on parool, millel on väike entroopia. Järgnevalt tuuakse välja põhilised nõrga parooli omadused.

Parool on liiga lühike [18]. Mida lühem on parool, seda vähem informatsiooni saab temas sisalduda.

Paroolis sisalduvad tähenduslikud kuupäevad [19], [18]. Näiteks sünnikuupäevad. Kuna niisugused kuupäevad on enamasti seotud kasutajaga, siis on nad võimalik kasutajat uurides teada saada. See vähendab ründajale mitteteadaoleva informatsiooni hulka paroolis.

Lihtsate täheasenduse kasutamine [19]. Näiteks „password“ asemel „p455w0rd“.

Järjestikused klahvid klaviatuuril [19]. Näiteks „qwertyuiop“ või „asdfghjkl“.

Kasutajanime või süsteemi/veebilehe nime sisaldumine paroolis [19], [18]. Sellise nime kasutamine paroolis tähendab, et ründajale on osa parooli informatsioonist samahea kui teada.

Hariliku parooli kasutamine [18]. Uurijad on avaldanud suuri nimekirju populaarsetest paroolidest, mis enamasti ei ole nõrgad ainult populaarsuse pärast, vaid ka oma lihtsuse poolest. Näiteks leiti umbes 15 miljoni lekkinud parooli uurimisel, et 0,6% nendest paroolidest on „123456“ ja 10 kõige populaarsema parooli abil saaks ära arvata keskmiselt 16 igast 1000 paroolist [20].

Sõnaraamatus esineva parooli kasutamine [19]. Paroolide häkkimise vahendid kasutavad sõnaraamatuid, et selliseid paroole kiiremini murda.

Parool, mida on kasutatud teistes süsteemides [18]. Ühe parooli murdmisega pääseb ründaja sel juhul mitmesse süsteemi.

3.3 Tugeva parooli omadused

Tugeval paroolil on võimalikult suur entroopia. Entroopia suurendamiseks saab kasutada järgnevaid omadusi [16]:

- Parooli suurem pikkus
- Erinevate kirjamärkide (sealhulgas ka sümbolite) kasutamine
- Genereerida parool juhuarvude abil

Esimesed kaks neist suurendavad parooli võimalike kombinatsioonide ja permutatsioonide arvu [16]. See tähendab, et ründajal on parooli ära arvamiseks vaja potentsiaalselt ära proovida suuremat hulka võimalusi. Juhuarvude abil genereeritud parooli puhul puudub oht, et parool arvatakse lihtsalt ära kasutaja kohta välja uuritud isikliku info ja ajaloo järgi.

Samas on pikemat ja keerulisemat parooli kasutajal enamasti palju raskem meelde jätta kui lühikest, mis sunnib kasutajat eelistama lühemaid ja lihtsamaid paroole. Jõhkra jõu rünnet kasutaval ründajal sellist probleemi ei ole. Ta lihtsalt katsetab ühekaupa kõik võimalikud paroolid läbi ja meeldejäävus ei mängi mingit rolli [19]. Mittemeeldejääv parool toob kasu ainult ründajale.

Probleemi saab lahendada kasutades paroolsõna asemel paroollauset (ingl *passphrase*) [21]. Paroollauseks võib olla on näiteks „minu imeline parool“. Paroollause on pikem kui paroolsõna ning kuna ta koosneb tavalistest sõnadest, siis on inimesel seda kergem meelde jätta [21]. Meeldejätmist võib lihtsustada ka näiteks riimuvate sõnade kasutamine [22]. See lihtne näide ei ole aga ideaalne. Paroolis kasutatud lause väljendab mõtet, mis tähendab, et lause osade informatsioon on teiste osadega seotud ja parool on seetõttu nõrgem. Paroollause on palju tugevam, kui paroollause genereerida suvaliselt valitud sõnadest suurest sõnastikust [21].

4 Parooli turvaliselt käsitlemine

Järgnevalt kirjeldatakse, mis tehnoloogilisi võtteid peaks rakendama, et vähendada paroolide lekkimise ja murdmise riski.

4.1 Kliendipoolselt

Veebilehel oleval vormil peab paroolide sisestamise tekstiväljadel olema sisendi tüüp „password“ (vt Joonis 1). See tagab, et brauser kuvab nendele väljadele sisestatud kirjamärkide asemel tärnid [23]. Kui parool ei ole peidetud, siis võib kõrvaline isik seda pealt näha, mis tähendab, et parool ei ole enam teada ainult sisestajale ja enam ei ole turvaline.

```
<label>
  Password
  <input type="password" name="password1" />
</label>
<label>
  Confirm Password
  <input type="password" name="password2" />
</label>
```

Joonis 1. Parooli tüüpi sisendiga tekstiväljad.

Vormi saatmiseks veebiserverile tuleks kasutada *HTTP* protokolli meetodit *POST* (vt Joonis 2).

```
<form method="POST" action="target.php">
      […]
</form>
```

Joonis 2. POST meetodit kasutav vorm.

GET meetodi kasutamise puhul lisab veebibrauser kõigi vormi väljade väärtused väljakutsutava lehekülje aadressile [24], mis tähendab, et kasutaja kõrval asuvad isikud võivad brauseri aadressiriba pealt näha sisestatud infot avatekstina (vt Joonis 3).

Joonis 3. GET meetodiga kasutatud vormi ärasaatmisel aadressiribal tekkivad väärtused.

POST meetodiga saadetud infot hoitakse lehekülje aadressist eraldi [24]. Lisaks võib mainida, et otsingumootorite ämblikprogrammid (ingl *spider*), mis nende jaoks veebilehti

indekseerivad, üldjuhul väldivad *POST* meetodiga päringuid, et veebilehtedele infot mitte sisestada.

4.2 Serveripoolselt

Turvamata HTTP ühenduse korral saadetakse kasutaja brauserist serverisse kasutaja poolt sisestatud info, sealhulgas paroolid, tavaliselt avateksti kujul. See tähendab, et igaüks, kes brauseri ja veebiserveri vahelist infovahetust pealt kuulab, on võimeline lugema kasutaja sisestatud paroole [25]. Pealtkuulamine võib toimuda näiteks häkitud ruuteriga või WiFi ühendust jälgides.

HTTPS protokoll on HTTP protokolli turvaline versioon [25]. HTTPS kasutab privaatse ja avaliku võtme paari (vt jaotis 2.2.3), et kaitsta pealtkuulamise eest brauseri ja veebiserveri vahelist andmevahetust [25]. HTTPS protokolli kasutamine eeldab, et serveri administraator on serverile installeerinud vajalikud sertifikaadid ning seadistanud veebiserveri neid kasutama.

4.3 Andmebaasis

Paroole andmebaasis avatekstina hoida on ohtlik. Kui ründaja pääseb ligi andmebaasi andmetele, siis on tal koheselt teada kõigi kasutajate paroolid. Seetõttu tuleks parool enne andmebaasi talletamist mõne ühesuunalise räsialgoritmiga räsida [7]. Räsialgoritmi ühesuunalisus tähendab, et parooli räsitud kujust ei ole enam võimalik välise informatsioonita algset parooli taastada.

Varased räsialgoritmid nagu DES, MD4, MD5, SHA-1, jne on tänapäeva tingimustes nõrkadeks jäänud ning nende kasutamine ei ole enam soovitatav [26]. Räsimiseks tuleks kasutada tugevaid tuntuid algoritme, mille tugevuses on teadlased veendunud [27]. Nendeks soovitatakse töö kirjutamise ajal (2016. aasta kevadel) näiteks bcrypt algoritmi [28], mille toetus on PHP keelde sisse ehitatud [29]. Oma isikliku räsialgoritmi loomine ei ole hea idee, sest tugeva algoritmi loomine on väga keeruline ning vigade tegemine, mis muudavad algoritmi kergesti murtavaks, on väga lihtne [27].

Räsialgoritm võiks töötada võimalikult aeglaselt, et ründajal kuluks iga parooli äraarvamiskatse peale rohkem aega [28]. Kui räsialgoritm on liiga kiire, siis saab seda

teha aeglasemaks kasutades seda mitu korda järjest, räsides eelmise iteratsiooni tulemust uuesti.

Räsimisel tuleks kindlasti kasutada soola. Soola termini all on mõeldud kirjamärgijada, mis lisatakse kasutaja paroolile (algusesse või lõppu) enne räsimist [7]. Soola kasutamine aitab vältida vikerkaaretabeli (ingl *Rainbow Table*) tüüpi rünnakuid, kus ründajal on ettevalmistatud suur hulk paroolidele vastavaid räsiväärtusi, mida on vaja vaid andmebaasis olevate räsiväärtustega võrrelda [7]. Sool peaks eelistatavalt olema unikaalne iga kasutaja jaoks, et ei saaks luua süsteemi jaoks ühte vikerkaaretabelit.

5 Olemasolevad paroolide muutmisega ja/või nullimisega seotud rakendused ja komponendid

Autor otsis töö jaoks püstitatud probleemi lahendamiseks olemasolevaid rakendusi või komponente. Järgnevalt tuuakse leitute seest välja need rakendused või komponendid, mille funktsionaalsus oli töö eesmärkidega kõige sarnasem. Selliste rakenduste ja moodulite leidmiseks kasutati põhiliselt *Google* otsingut järgmiste võtmesõnadega ja nendega sarnaste variatsioonidega:

- password change application
- forgot password application
- password reset module
- password recovery module
- password change component

5.1 SMS|Passcode Password Reset Module

http://www.smspasscode.com/what-we-do/password-reset-module/

Tasuline teenus Active Directory tarkvara kasutavate võrkude jaoks, mis tuvastab, kui kasutaja on sisestanud vale parooli ning saadab kasutajale SMS-iga juhised parooli vahetamiseks [30]. SMS-is sisaldub ka ühekordselt kasutatav kood, millega kasutaja tuvastatakse [30]. See eeldab, et süsteemis hoitakse kõikide kasutajate telefoninumbreid. Kuna teenus on mõeldud Active Directory võrkude jaoks, siis ei saa seda kasutada käesoleva töö probleemi lahendamiseks.

5.2 Password Reset module for ProcessWire

https://github.com/plauclair/PasswordReset

Avatud lähtekoodiga paroolide nullimise moodul ProcessWire sisuhaldussüsteemi jaoks [31]. Saadab kasutajale meili veebiaadressiga, kus kasutaja saab määrata uue parooli [31]. Veebiaadress aegub 24 tunni pärast [31]. Uue parooli tugevuse määramiseks

kontrollitakse ainult selle pikkust [31]. Kuna moodul on kirjutatud spetsiifilise tarkvara jaoks, siis ei saa seda antud töö probleemi lahendamiseks kasutada.

5.3 Forgot Password for User Management System

http://codecanyon.net/item/forgot-password-for-user-management-system/2516786

Tasuline moodul, mida saab integreerida eksisteerivatesse rakendustesse [32]. Moodul laseb kasutajatel nullida oma parool kui nad on selle unustanud [32]. Nullimiseks peab kasutaja sisestama oma kasutajanime ning seejärel meili teel saadetud veebiaadressil, mis on aktiivne vaid lühikese aja, määrama uue parooli [32]. Parooli räsimiseks on toetatud ainult MD5 ja SHA1 räsifunktsioonid [32], mis on mõlemad tänapäeva tingimustes liiga nõrgad, ning ei kasutata soola [32], mis on ebaturvaline. Moodul toetab ainult MySQL andmebaasisüsteemi [32], mis tähendab, et rakendust pole võimalik kasutada näiteks ruumide broneerimise süsteemiga, mis kasutab PostgreSQL andmebaasisüsteemi.

5.4 ServiceNow Password Reset

http://wiki.servicenow.com/?title=Password_Reset

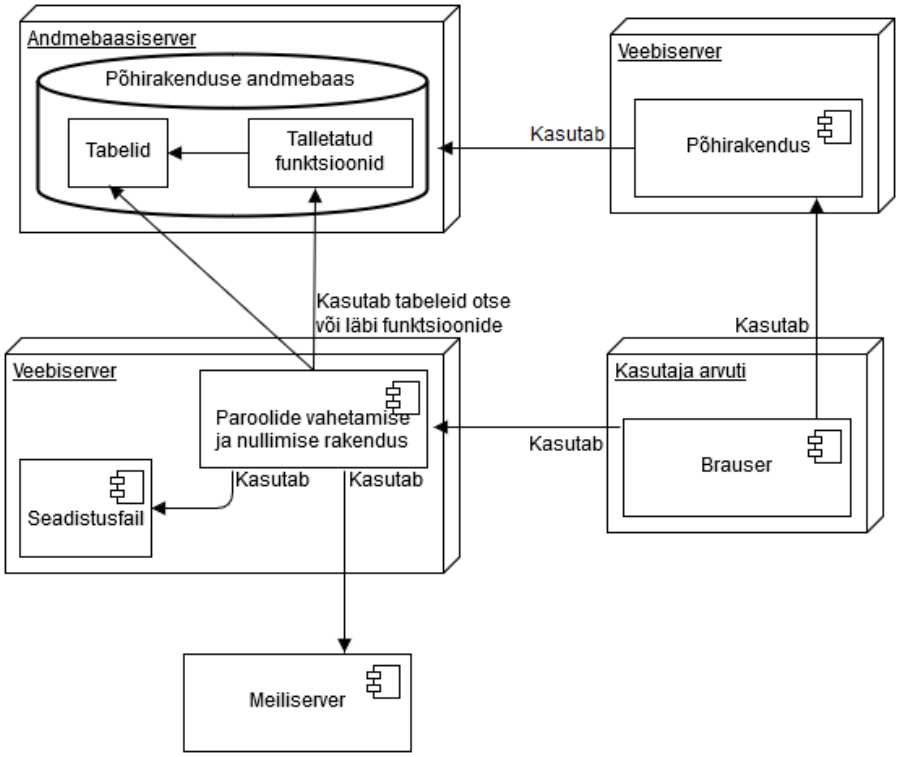
ServiceNow haldustarkvara moodul, mis laseb kasutajatel nullida oma paroole kas SMS-i teel saadetud koodi sisestamisega või kasutajat identifitseerivatele küsimustele vastamisega [33]. Nendeks küsimusteks võivad olla kasutajate poolt varem sisestatud kontrollküsimused või süsteemis kasutaja kohta hoitava informatsiooni põhjal genereeritud küsimused [33]. See eeldab, et süsteemis hoitakse informatsiooni, mille põhjal saab kasutajat unikaalselt ja kindlalt tuvastada. Kuna moodul on kirjutatud spetsiifilise kommertsiaalse produkti jaoks, siis ei saa seda töös esitatud probleemi lahendamiseks kasutada.

5.5 Järeldus

Autor ei suutnud leida turvalist rakendust või komponenti, mille abil saaks kasutajatel iseteeninduslikult võimaldada oma paroole vahetada näiteks Informaatikainstituudi ruumide broneerimise süsteemis või Mauruses. Autor järeldab, et on tugev põhjus luua tarkvara, mis lahendaks käesolevas töös püstitatud probleemi.

6 Universaalne paroolide vahetamise ja taastamise rakendus

Järgnevalt kirjeldatakse käesolevas töös loodud rakendust.



Joonis 4. Rakenduse evitusskeem.

Rakenduse evitusskeemil (vt Joonis 4) on välja toodud kõik rakendusega seotud komponendid. Skeem toob välja ühe võimaluse rakenduse keskkonna ülesehitusest. See ei tähenda, et rakenduse keskkond peab olema just selline nagu on joonisel kujutatud. Kõik komponendid võivad olla jaotatud mitmete erinevate riistvarade peale või ühe riistvara peal töötada. Näiteks võivad põhirakendus ja paroolide vahetamise ja nullimise rakendus asuda samal veebiserveril.

6.1 Rakenduse eesmärgid

- Administraatoritel peab olema võimalik rakendust seadistada töötamiseks võimalikult paljude erinevate süsteemidega.

- Rakendus peab võimaldama kasutajatel vahetada ja nullida oma paroole ilma süsteemi administraatorite sekkumiseta sõltumata sellest, kas nad mäletavad oma kehtivat parooli või mitte. Parooli nullimise eeldus on, et kasutaja on registreeritud süsteemis, kus paroole vahetatakse, enda toimiva meiliaadressi.

- Administraatoritel peab olema võimalik seadistada, mis minimaalse tugevusega paroole rakendus kasutajatel valida lubab.
- Võimaldada kasutajatel nupuvajutusega genereerida tugev ja meeldejääv parool.
- Rakendus peab kaitsma end väärkasutamise eest. See tähendab, et rakendus ei tohi lubada kellelgi vahetada neile mittekuuluvate kasutajate paroole või õngitseda infot süsteemi kasutajate kohta.

6.2 Funktsionaalsus

Rakendusel on kaks põhilist funktsiooni: kasutaja parooli vahetamine ja selle erijuht, kasutaja parooli nullimine.

6.2.1 Parooli vahetamine

Kui kasutaja soovib vahetada oma parooli, siis peab ta eelkõige sisestama oma kasutajanime ja kehtiva parooli ning seejärel kaks korda uue parooli. Juhul kui kasutaja ei tea oma kehtivat parooli, peab ta suunduma parooli nullimise protsessi leheküljele. Süsteem kontrollib kasutaja sisestatud uue parooli tugevust vastavalt rakenduse parooli tugevuse seadistusele. Vormi lubatakse ära saata vaid siis, kui sisestatud uus parool on piisavalt tugev. Kui vorm on ära saadetud, kontrollib süsteem andmebaasis, kas kasutaja sisestatud kasutajanimi ja kehtiv parool vastavad eksisteerivale kasutajale. Kui sellist kasutajat ei eksisteeri, siis matkib süsteem siiski parooli vahetumist. Kui kasutaja eksisteerib, siis vahetatakse parool andmebaasis uue vastu.

6.2.2 Parooli nullimine

Parooli nullimine on parooli vahetamise erijuht, kus kasutaja ei tea oma kehtivat parooli. Kasutaja sisestab kasutajaliideses oma meiliaadressi. Süsteem kontrollib, kas see meil on mõne eksisteeriva kasutajaga seotud. Kui selline kasutaja eksisteerib, siis saadetakse aadressile meil viitega leheküljele, kus ta saab määrata uue parooli. Viite aadressis on unikaalne kood, mis identifitseerib seda parooli nullimise protsessi. Seda hoitakse koos protsessi alustamise aja, kasutaja IP aadressiga ja kasutajanimega veebiserveril ajutistes failides. Koodi loetakse aegunuks ning parooli vahetada ei lubata, kui protsessi alustamisest on möödunud rohkem kui üks tund. Kui kasutaja avab meilile saadetud parooli nullimise lingi, siis kontrollitakse, et kasutaja IP aadress vastaks sellele, mis see

protsessi alguses oli. See aitab ära hoida rakenduse väärkasutamist. Leheküljel peab kasutaja sisestama uue parooli, mille tugevust kontrollitakse samamoodi nagu tavalisel parooli muutmisel. Vormi ära saatmisel muudetakse kasutaja parool tema poolt määratuks.

Alternatiivne lahendus oleks, et kasutaja määrab uue parooli protsessi alguses ja meilile saadetakse link, mis uue parooli aktiveerib. See on aga potentsiaalselt liiga ohtlik. Ründaja võib proovida nullida kasutaja parooli ja kasutaja võib kogemata või uudishimust lingi avada, põhjustades oma parooli vahetumise ründaja poolt määratud parooli vastu.

6.3 Eessüsteem

Kuna loodav rakendus peab olema võimalikult universaalne, siis tehti otsus teha kasutajaliides eelkõige inglise keeles. Ingliskeelse rakenduse puhul on võrreldes eestikeelsega suurem tõenäosus, et see on teistele arendajatele kasulik. Siiski on rakenduse rahvusvahelikustamise võimaldamine üks oluline edasiarenduspunkt (vt jaotis 6.8).

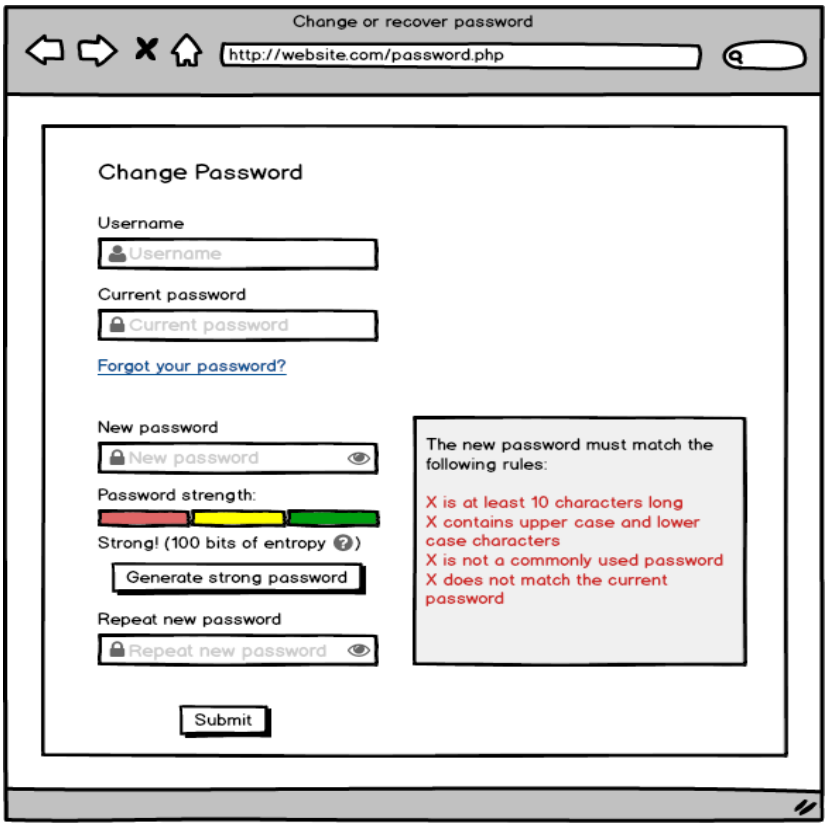
Enne rakenduse programmeerimise alustamist otsustati luua kasutajaliidese disaini prototüüp. Prototüübi tegemine on abiks otsustamisel, kuidas hakkab kasutajaliides reaalselt välja nägema ja töötama.

Prototüüp loodi sõrestikmudelina (ingl *wireframe*) ning esitatakse järgnevalt staatiliste piltidena. Tegemist ei ole funktsionaalse prototüübiga, millega kasutaja rakenduse töötamist proovida saab, vaid staatilise prototüübina, mis annab vaid ülevaate kõikide oluliste kasutajaliidese elementide paigutusest. Sõrestikmudeli eesmärk ei ole kujunduse detailne kuvamine, vaid lubada arendajal eksperimenteerida uute kasutajaliidese disaini ideedega ja neid testida ning sobitada [4]. Seega ei kajasta prototüüp kasutajaliidese elementide lõplikke kujusid, värve, suurusi ja sõnastust.

Prototüüp sai tehtud tarkvara Balsamiq Mockups versiooniga 3.3.14 [34], mis oli töö tegemise ajal (2016. aasta kevadel) selle tarkvara värskeim versioon.

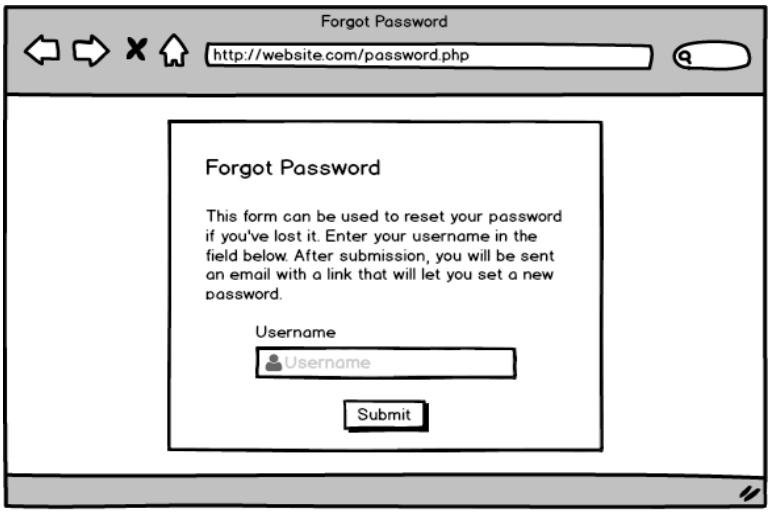
6.3.1 Prototüüp

Rakenduse avaleht on parooli vahetamise leht (vt Joonis 5). Lehel saab kasutaja vahetada endale teadaoleva kehtiva parooli uue vastu. Selle jaoks peaks kasutaja sisestama oma kasutajanime, kehtiva parooli ning kaks korda uue parooli. Rakendus näitab kasutajale, kui tugev on sisestatud uus parool ning kas see on vastavuses seadistuses määratud parooli tugevusreeglitega. Kasutajal on võimalus nupuvajutusega lasta rakendusel genereerida tugev parool. Kehtiva parooli välja all asub viide parooli nullimise protsessi esimesele leheküljele, juhuks kui kasutaja ei tea oma parooli.



Joonis 5. Parooli vahetamise lehekülje staatiline prototüüp.

Parooli nullimise esimesel leheküljel (vt Joonis 6) on lühike kirjeldustekst ning kasutajanime väli.

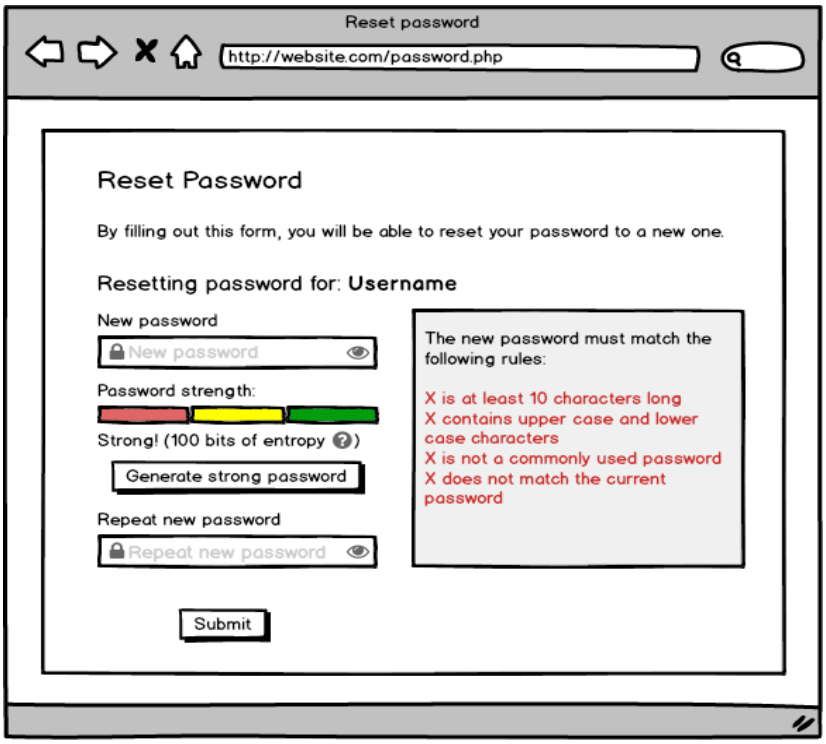


Joonis 6. Parooli nullimise esimese vormi staatiline prototüüp.

Pärast parooli nullimise protsessi esimese vormi täitmist ja ära saatmist saadetakse kasutajaga seotud meiliaadressile kiri (vt Joonis 7), milles teavitatakse kasutajat parooli nullimise protsessi käivitamisest ning mis sisaldab viidet parooli nullimise teisele leheküljele.

Joonis 7. Parooli nullimisel kasutajale saadetud meili staatiline prototüüp.

Parooli nullimise teisel leheküljel (vt Joonis 8) on kasutajal võimalik määrata uus parool. Leheküljel asuvad sarnased elemendid, mis parooli vahetamise leheküljelgi (vt Joonis 5). Peale selle vormi ära saatmist vahetatakse kasutaja parool andmebaasis ära.



Joonis 8. Parooli nullimise protsessi uue parooli määramise vormi staatiline prototüüp.

Lisas 4 on välja toodud mõned ekraanitõmmised rakenduse lõplikust väljanägemisest.

6.3.2 Kasutajaliidese ehitamiseks kasutatavad abivahendid

Peamine kasutatav tööriist kasutajaliidese dünaamilisemaks muutmiseks on JavaScripti teek jQuery. See teek lihtsustab paljusid tihti kasutatavaid võtteid nagu näiteks elementide valimist lehekülje dokumendiobjektide mudelist (ingl *Document Object Model*, lühidalt DOM) ja AJAX päringute tegemist.

Veel otsustati kasutada Twitter Bootstrap raamistiku töö kirjutamise ajal uusimat versiooni 3.3.6. Bootstrap on eessüsteemi raamistik, mis annab mugavad vahendid lehekülje kiireks disainimiseks ja lehekülje dünaamiliste elementide loomiseks. Selle raamistiku kasutamise eeldus on jQuery teegi kasutamine.

6.3.3 Mallid

Mallide ülesanne on neile antud sisendi põhjal genereerida kasutajale näidatav vaade. Rakendus kasutab mallidena PHP funktsioone, mille sisend on malli sisendinfo ja väljund on genereeritud vaade. Sisendinfo antakse mallidele lehekülgede klassides. Iga mall asub omaette failis. Mallid võivad olla kokku pandud mitmetest teistest mallidest. Rakendus kasutab malle kahel eesmärgil.

Esiteks kasutatakse malle kasutaja veebibrauserile tagastatavate HTML dokumentide moodustamiseks. Kõik lehekülgede mallid kasutavad baasmalle, mis määravad lehekülgede üldise välimuse, sealhulgas päise ja jaluse. Baasmallidesse saavad lehekülgede mallid baasmallis määratud asukohta oma sisu sisestada. Peale selle on eraldi mallideks veel elemendid, mida kasutatakse mitmel leheküljel (näiteks uue parooli määramise väljad).

Teiseks kasutatakse malle parooli nullimise protsessi käigus saadetud meili pealkirja, avateksti ja HTML formaadis versiooni genereerimiseks. Meili HTML formaadis versioon kasutab lehekülgede mallidele sarnaselt baasmalli, mida võib tulevikus kasutada teiste meilide mallide loomiseks.

6.3.4 Parooli genereerimine

Genereeritud paroolid vastavad jaotises 3.2 mainitud tugeva parooli omadustele. Täpsemalt sõltub genereeritav parool ka jaotises 0 kirjeldatud parooli tugevusega seotud seadistuse parameetritest. Parooli genereerimiseks valitakse seadistuses määratud sõnaraamatust juhuslikult seadistuses määratud arv sõnu ning seotakse nad kriipsuga.

Sõnaraamatu faili struktuur on lihtne (vt Joonis 9). Iga sõna on eraldi real ning sõnad koosnevad vaid väiketähtedest ja kriipsudest. Rakendust kasutava süsteemi administraatorid võivad kasutada mistahes sõnaraamatut, mis nõutud struktuurile vastab.

```
19647   obesity
19648   obey
19649   obfuscate
19650   obfuscation
19651   obit
19652   obituary
19653   object
19654   object-oriented
```

Joonis 9. Sõnaraamatu faili näide.

Rakendusega lähtekoodiga kaasas olev sõnaraamat põhineb ingliskeelsete sõnade kollektsioonil 12Dicts, mille koostaja on Alan Beale [35]. Kollektsioonis olevad sõnad on tavapärased ja mitte eriala spetsiifilised, mistõttu sobib see väga hästi meeldejääva parooli genereerimiseks. Sõnad on põhjalikult kontrollitud, et neis ei oleks vigu [35].

Täpsemalt on kasutatud 12Dicts kollektsiooni töö kirjutamise ajal uusima versiooni 6.0.1 nimekirja 2+2+3frq, milles on sõnad grupeeritud ning sõnade grupid on järjestatud

kasutussageduse järgi kahanevas järjekorras [35]. Nimekirjaga on teostatud järgnevad muudatused:

- Eemaldatud on kõik kommenteerivad sümbolid !, (,), *
- Eemaldatud on kõik sõnad, mis on lühemad kui 4 kirjamärki
- Eemaldatud on kõik vandesõnad
- Kõik suurtähed on vahetatud väiketähtedeks

Muudatustele järgnevas nimekirjas on 33248 sõna. Selle sõnaraamatu abil genereeritud paroolidel, milles kasutatakse nelja juhuslikku sõna, on enamasti Shannoni valemi järgi arvutatud entroopia üle 100 biti.

6.3.5 Parooli tugevuse määramine

Parooli entroopia määramiseks kasutatakse jaotises 3.1 kirjeldatud Shannoni entroopia valemi käsoleva töö autori poolt loodud realisatsiooni PHP keeles. Parooli muutumisel kasutajaliidese väljal saadab brauser AJAX päringu serverile sisestatud parooliga. Server arvutab selle parooli entroopia bittides ning tagastab väärtuse brauserile, kus seda siis kasutajale näidatakse. Kui entroopia bittide arv ületab seadistuses määratud minimaalse väärtuse, siis näidatakse kasutajaliidesel, et parooli tugevus on „*Okay*“, ehk piisav. Kui entroopia bittide arv ületab seadistuses määratud tugeva parooli bittide väärtuse, siis näidatakse, et parooli tugevus on „*Good*“, ehk hea. Soovituslik minimaalne entroopia bittide arv 60 on valitud subjektiivselt võrreldes erinevate genereeritud paroolide arvutatud entroopiat.

6.4 Serveripoolne osa

Rakenduse serveripoolse osa loomiseks sai valitud PHP programmeerimiskeel [36], sest selles keeles kirjutatud rakendusi on lihtne veebiservereile paigaldada ning kuna see keel on töö autorile tuttav. Kuna rakendusel on ainult kolm erinevat lehekülge ning üks AJAX päringutega tegelev kontroller, siis otsustati mitte kasutada PHP raamistikke. Raamistikud on väga suureks abiks suurte rakenduste kirjutamisel, kus on palju lehekülgi. Sellise võrdlemisi väikese rakenduse puhul on aga lihtsam kontrollerite loogika ise

kirjutada. Raamistiku mittekasutamine vähendab ka rakenduse sõltuvust teistest projektidest.

Rakenduse loomisel kasutatakse objektorienteeritud programmeerimist, mis tähendab, et enamus loogikast organiseeritakse klassidesse.

6.4.1 Andmebaasid

Andmebaasidega suhtlemiseks sai valitud andmebaasi abstraktsiooni kihi Doctrine DBAL [37] töö kirjutamise ajal uusim stabiilne versioon 2.5.4. Otsus seda tarkvara kasutada tehti, sest see toetab paljusid erinevaid andmebaasitüüpe, näiteks PostgreSQL, Oracle Database, Microsoft SQL Server, MySQL [37]. See aitab teha rakendust universaalsemaks, sest võimaldab rakendust väheste muudatuste või ilma muudatusteta kasutada koos erinevate andmebaasisüsteemide peale ehitatud rakendustega.

Doctrine DBAL on kergekaaluline ja õhuke kiht PHP keele ametliku laienduse PDO (*PHP Data Objects*) ümber [37]. Enamuste andmebaasisüsteemidega (näiteks PostgreSQL, MySQL) suhtlemiseks kasutab Doctrine PDO draivereid. Osade jaoks (näiteks Oracle) kasutatakse teisi draivereid, sest vastav PDO draiver puudub või ei ole piisavalt hästi töötav. Kõik need draiverid tehakse kättesaadavaks ühise ja võimsa liidese kaudu [38]. Doctrine nõuab töötamiseks vähemalt PHP tarkvara versiooni 5.3.2 [39].

Kui seadistusfailis määratud andmebaasiga ei saa ühendust, siis teatatakse kasutajale, et süsteemis tekkis viga ning nad peaksid ühendust võtma süsteemi administraatoritega.

Ettevalmistatud lausete (ingl *prepared statement*) kasutamine hoiab ära SQL-i süstimise rünnakud [40]. Kõik SQL laused saadetakse esiteks ilma sisendinfota andmebaasiserverile. Andmebaasiserver sõelub lause ning märgib ära, milliseid sisendparameetreid mis kohtades lause nõuab. Seejärel saadab rakendus sisendid ning andmebaasiserver paigutab need ise vastavatesse kohtadesse. See välistab SQL-i süstimise rünnakud.

6.4.2 Meilide saatmine

PHP keelde on meilide saatmise võimalus sisseehitatud *mail* funktsiooni abil [41]. Sellel on aga suur hulk puuduseid. Kõige olulisemad neist on välja toodud lisas 3.

Nende puuduste vältimiseks ning et meilide saatmist oleks mugavam seadistada otsustati meilide saatmiseks kasutada vabavaraliselt kättesaadava PHPMailer komponendi töö kirjutamise ajal (2016. aasta kevadel) uusimat väljalastud versiooni 5.2.14 [42]. PHPMailerisse on integreeritud SMTP tugi [42], mis tähendab, et meilide saatmiseks ei pea toetuma rakendusega samal serveril jooksvale meiliserverile.

Meili mudeli hoidmiseks on loodud klass Email. See hoiab meili saatja nime ja aadressi, saaja nime ja aadressi, meili pealkirja ning meili sisu avateksti ja HTML formaadis. Abstraktne baasklass EmailSender defineerib meili saatmise liidese ning selle laiendus EmailSenderPHPMailer realiseerib meili saatmise PHPMailerit kasutades. Abstraktse baasklassi kasutamine teeb lihtsaks vajaduse korral sama funktsiooni ühe realisatsiooni vahetamise teise vastu. Meilisaatmise seadistamist kirjeldatakse jaotises 0.

6.4.3 Paroolide räsimine

Kui vastavalt seadistusele ei tehta paroolide räsimist andmebaasis olevates funktsioonides, vaid on see rakenduse ülesanne, siis kasutatakse räsimiseks PHP keelde sisseehitatud *crypt* funktsiooni [29]. Töö kirjutamise hetkel toetab see funktsioon räsialgoritme DES, MD5, Blowfish (bcrypt), SHA256 ja SHA512 [29]. DES ja MD5 on teistega võrreldes nõrgemad, mistõttu otsustati neid rakenduses mitte toetada. SHA256 ja SHA512 algoritmid on piisavalt tugevad kui nendega kasutatav räsimisiteratsioonide arv on kõrge. Seega on rakendus võimeline paroole räsima Blowfish, SHA256 ja SHA512 algoritmidega. Algoritmi valik toimub rakenduse seadistuses (vt jaotis 6.4.1).

Räsialgoritmi iteratsioonide arvu seadistuse kaudu muuta pole rakenduses esialgu võimalik. Määratud iteratsioonide arv on piisavalt suur, et testserveril võtaks paroolide räsimine keskmiselt 0,3 sekundit. Iteratsioonide arvu automaatne määramine vastavalt serveri jõudlusele on rakenduse potentsiaalne täiendus.

6.4.4 Logimine

Logide kirjutamise eesmärk on anda süsteemi administraatorile parem ülevaade, kas rakendus töötab, kuidas rakendust kasutatakse ja kas keegi proovib rakenduse kaudu süsteemi rünnata. Sellel eesmärgil kirjutab rakendus logisse järgmised tegevused:

- Vahetati parool kasutades teadaolevat parooli
- Prooviti parooli vahetada vale kehtiva parooliga

- Alustati parooli nullimise protsess
- Prooviti alustada parooli nullimine tundmatu kasutajaga
- Parooli nullimise protsess lõpetati uue parooli määramisega
- Andmebaasiga suhtlemisel või meili saatmisel tekkis viga

Iga tegevusega logitakse aeg ning kasutaja IP aadress. Kui see on saadaval, siis logitakse ka kasutajanimi.

Logifailide asukoht on määratud rakenduse seadistuses. Iga päeva logide jaoks tehakse uus logifail, mille nimi algab kuupäevaga. Logifaili sisu näidet võib vaadata Lisast 6.

6.5 Seadistus

Vastavalt tööle püstitatud eesmärkidele peab loodav rakendus olema võimalikult universaalne. Selleks peab rakendus olema seadistatav toetamaks erinevaid süsteeme.

Lihtsuse mõttes hoitakse seadistust tavalises PHP koodifailis olevas assotsiatiivses jadas. Jada kirje võtmeks on seadistuse parameetri nimi ning jada kirje väärtus on parameetri väärtus (vt Joonis 10). Kui failis defineeritakse sama parameetrit mitu korda, siis säilib neist ainult kõige viimase definitsiooni väärtus. Esialgu ei kontrolli rakendus seadistusfailis määratud parameetrite väärtuste õigsust. See on rakendusele võimalik hilisem täiendus.

Seadistuse parameetrid on jaotatud järgnevatesse gruppidesse:

- Andmebaas – kuidas rakendus suhtleb andmebaasiga
- Parooli tugevuse kontroll – millistele reeglitele peab kasutaja uus parool vastama
- Parooli genereerimine – kuidas genereeritakse tugev parool
- Meilide saatmine – kuidas rakendus meile saadab
- Üldine

Iga grupi (välja arvatud üldise) parameetritel on neid teistest eristav eesliide. Täpsemalt on rakenduse seadistuse parameetreid kirjeldatud lisas 1. Rakenduse lähtekoodiga on näidisena kaasas seadistusfail, kus on välja toodud ja inglise keeles kommenteeritud kõik lisas 1 kirjeldatud parameetrid (vt Joonis 10).

```
/**************************
 **** Generic settings ****
 **************************/

/**
 * Unique identifier for this instance of Password Manager.
 * Needed when multiple instances of this application are running on the same
 * web server to avoid the separate instances interfering with eachother's data.
 *
 * The value has to be a string of only alphanumeric characters and may be case-insensitive
 * depending on the host operating system.
 * The value may be used as part of filenames, so it should not be too long to avoid
 * file system limitations. Around 10 characters would be fine.
 * Example: 'sitename'
 */
$cfg['instanceIdentifier'] = 'sitename';

/**
 * The name of the system, the users and passwords of which this application deals with.
 * Used for example in sent emails and titles of pages.
 *
 * Example: 'Site Name'
 */
$cfg['applicationName'] = 'Site Name';
```

Joonis 10. Lõik rakendusega näidisena kaasasolevast seadistusfailist.

Kuna seadistusfail sisaldab sensitiivset infot nagu näiteks andmebaasi kasutajanimi ja parool, siis peab rakendama meetmeid, mis vähendavad riski, et faili sisu on loetav kõrvalistele isikutele.

Kasutatakse *.htaccess* faili, et veebiserveri tasemel keelata ligipääs sensitiivset infot sisaldavatele failidele (vt Joonis 11). Seda faili kasutab Apache veebiserveritarkvara kaustapõhise seadistuse määramiseks [43]. Joonis 11 näidatud lõiguga keelatakse igasugune ligipääs seadistusfailile juhul kui ta on rakenduse kaustas. Peab mainima, et selle meetme töötamine sõltub sellest, kas veebiserveri seadistuses on sisse lülitatud parameeter *AllowOverride* [44].

```
<Files "config.php">
  order allow,deny
  deny from all
</Files>
```

Joonis 11. Lõik failist .htaccess, mis keelab ligipääsu config.php failile.

Rakenduse ülesseadjaile soovitatakse hoida tegelikku seadistusfaili väljaspool veebiserveri juurkausta. Veebiserveri juurkaust on kaust, milles olevatele failidele veebiserver lubab päringuid teha [45]. Kui veebiserveril peaks mingil põhjusel PHP

tarkvara lakkama töötamast, siis on võimalik, et rakenduse PHP failide sisu saadetakse brauserile avateksti kujul. Sellises olukorras oleks võimalik brauseri kaudu lugeda seadistusfaili sisu. Kui seadistusfaili hoitakse aga väljaspool veebiserveri avalikku kausta, siis ei ole brauseriga võimalik sellele päringuid teha, ent PHP on siiski võimeline seda rakenduse jaoks lugema.

Kuna veebiserveril peab olema seadistusfailile lugemisõigus, siis pole võimalik seadistusfaili peita kasutajate eest, kellel on õigused samal veebiserveril PHP faile luua ja veebiserveri kaudu käivitada. Nad saavad lihtsalt luua PHP skripti, mis veebiserveril veebiserveri õigustega käivitudes loeb seda seadistusfaili.

6.6 Rakenduse paigaldamine

Rakenduse kasutamiseks on vajalik, et sellega täiendatav süsteem S ning rakenduse käitamise keskkond täidaks järgnevad minimaalsed nõuded:

- S kasutab paroolide räsimiseks Blowfish, SHA256 või SHA512 algoritme.
- S hoiab andmebaasis kasutaja meiliaadressi ja parooli räsiväärtust.
- Veebiserverile on paigaldatud PHP tarkvara, mille versioon on vähemalt 5.3.2. Seda versiooni nõuab Doctrine DBAL tarkvara (vt jaotis 0).
- Veebiserveril peab olema installeeritud kasutatava andmebaasisüsteemi jaoks vajalik PHP lisamoodul (vt jaotis 0).
- Paroolide nullimise protsessi käigus meilide saatmise töötamiseks peab rakendusel olema võimalik ühenduda meiliserveriga.

Rakenduse paigaldamiseks tuleb teostada järgmised sammud:

- Rakenduse failide alla laadimine GitHub-i keskkonnast (vt jaotis 6.9).
- Rakendusega kaasasolevas *config.php* failis oleva seadistuse muutmine vastavalt sihtsüsteemi parameetritele (vt jaotis 6.5).
- Kui vaja, andmevahetuse funktsioonide loomine süsteemi andmebaasis (vt jaotis 0).

- Rakenduse failide üleslaadimine veebiserverile.
- Faili *config.php* liigutamine väljaspoole veebiserveri avalikku kausta (vt jaotis 6.5) ja failis *configlink.php* faili *config.php* asukoha määramine.

6.7 Võrdlus sarnase eesmärgiga rakenduste või moodulitega

Järgnevalt võrreldakse loodud tarkvara peatükis 5 leitud sarnase eesmärgiga rakenduste või moodulitega.

6.7.1 SMS|Passcode Password Reset Module

Võrreldes selle rakendusega, on loodud rakendus võimeline toetama erinevaid süsteeme ja mitte ainult Active Directory tarkvara kasutavaid süsteeme. Loodud rakendus ei kasuta paroolide vahetamiseks SMS-iga saadetud koodi, vaid kontrollib kasutaja olemasolevat parooli või tuvastab kasutaja temale meili saatmisega, mis ei eelda, et süsteem teab kasutajate telefoninumbreid.

6.7.2 Password Reset module for ProcessWire

Loodud rakendus kasutab võrreldes selle mooduliga palju tõhusamaid vahendeid uue parooli tugevuses veendumiseks. Sarnaselt eelnevaga ei ole loodud rakendus mõeldud ainult ühe spetsiifilise tarkvaraga kasutamiseks.

6.7.3 Forgot Password for User Management System

Erinevalt sellest moodulist on loodud rakendus võimeline toetama erinevaid andmebaasisüsteeme. Loodud rakendus kasutab tänapäeva tingimustes tugevaid räsialgoritme, see rakendus seevastu kasutab nõrkasid MD5 and SHA1 algoritme.

6.7.4 ServiceNow Password Reset

Ka see moodul on erinevalt loodud tarkvarast ehitatud vaid ühe kindla tarkvara toetamiseks. Loodud tarkvaral peab olema ligipääs vaid kasutajate kasutajanimedele, meiliaadressidele ja parooli räsiväärtustele. Sellel tarkvaral on kasutaja tuvastamiseks vaja teada ka kasutaja telefoninumbrit või muud informatsiooni kontrollküsimuste esitamiseks.

6.8 Probleemid ja võimalikud edasiarendused

Järgnevalt tuuakse välja mõned probleemid, mida käesoleva tööga ei lahendatud, ning ideed, mida mingil põhjusel ei realiseeritud.

Rakenduse rahvusvahelikustamine. Selleks, et rakendust oleks võimalik kasutada süsteemidega, mis ei ole ingliskeelsed, tuleks võimaldada kasutajaliidese tõlkimine erinevatesse keeltesse.

Seadistuse parameetrite väärtuste kontroll. Rakendus võiks osata kontrollida, et seadistusfailis on kõik vajalikud parameetrid väärtustatud ja nende väärtused vastavad nõudmistele. See aitaks administraatoritel vigasest seadistusest kiiremini teada saada.

Testidega katmine. Kui rakenduse loogika kaetaks testidega, siis saaks hilisema arenduse käigus vähendada või vältida programmivigade tekkimist. Testide kirjutamiseks saaks kasutada näiteks populaarset PHPUnit raamistikku [46].

Kasutajate registreerimine. Kui rakendusele lisataks ka kasutajate registreerimise võimalus, siis saaks tugeva parooli kasutamist jõustada samade reeglite järgi ka konto loomisel. Seda ideed ei realiseeritud kuna see on rakenduse teiste funktsioonidega võrreldes palju mahukam ning rakendus vajaks kasutatava süsteemi arhitektuuri kohta palju rohkem informatsiooni. Pealegi ei luba kõik rakendused kasutajatel endil konto registreerida, vaid selle loob neile administraator.

Räsialgoritmide iteratsioonide arvu automaatne seadistus. Erinevate serverite jõudlused ei ole samad, mistõttu peaks räsialgoritmide iteratsioonide arvu määrama vastavalt serveri jõudlusele. Selleks tuleks luua eraldi skript, mida saaks veebiserveril tööle panna ja mis määraks iteratsioonide arvu arvestades serveri jõudlust.

6.9 Lähtekoodi avaldamine

Kuna rakendus võib olla kasulik ka teistele arendajatele ja süsteemidele, siis otsustati see luua avatud lähtekoodiga vabavarana ning avaldada GitHubi keskkonnas. Üle 14 miljoni kasutaja ja üle 35 miljoni projektiga (2016. aasta andmed) on GitHub maailma suurim avatud lähtekoodiga tarkvara arendamise keskkond [47]. GitHubis on palju vahendeid, mis aitavad kaasa arendustegevusele ning toetavad koostööd arendajate vahel [47]. Projekti lisamine sinna tagab, et rakendus on kergesti kättesaadav kõigile, kes sellest

huvitatud on. Rakendust oma süsteemiga kasutavad isikud saavad kergesti teatada probleemidest või soovitada uusi ideid. Pole ka välistatud, et projektiga liituvad teised arendajad.

Lähtekood on avalikult kättesaadaval aadressil:

https://github.com/Pisi-Deff/PasswordManager

Lisaks lähtekoodile on GitHubi projektile lisatud ka ingliskeelne kirjeldus rakenduse olemusest ning juhend rakenduse kasutamiseks.

Lähtekood otsustati avaldada *MIT* litsentsiga [48]. See litsents on lühikese ja väga arusaadava tekstiga ning lubab vabalt kasutada rakendust ja selle lähtekoodi nii vabavaralistes, isiklikes, kui ka kommertsiaalsetes projektides, eeldusel, et autorile viidatakse. Litsents ei anna kasutajale ühtegi garantiid ning autor ei vastuta võimalike kahjude eest.

6.10 Rakenduse kasutamise näited

Loodud rakendus seati üles TTÜ Informaatikainstituudi ruumide broneerimise süsteemiga ning uue versiooniga TTÜ tarkvara jaotuse keskkonnast. Rakendus töötas mõlema süsteemiga. Nende rakenduste jaoks andmebaasides kasutajate, õiguste ja funktsioonide seadistamiseks käivitatud SQL laused on välja toodud lisas 5.

Mõlemad süsteemid kasutavad PostgreSQL andmebaasisüsteemi. Need süsteemid kasutasid erinevaid paroolide räsimise ja rakendustele juurdepääsu pakkumise lähenemisi (paroolide räsimine andmebaasi *vs* rakenduse tasemel; rakendused pöörduvad otse tabelite poole *vs* rakendused kasutavad andmebaasi läbi rutiinide kihi) ning see näitab, et loodud tarkvara saab mõlema juhuga hakkama.

Kuna käesoleva töö käigus oli autoril võimalus rakendust katsetada ainult PostgreSQL andmebaasisüsteemi kasutavate süsteemidega ei saa välistada, et teiste andmebaasisüsteemide toetamiseks pole vaja teha rakendusega mõningaid muudatusi.

7 Kokkuvõte

Käesoleva töö eesmärgiks oli disainida ning realiseerida veebipõhise kasutajaliidesega rakendus, mida saab kasutada olemasolevate süsteemidega ning mis võimaldab nende süsteemide kasutajatel iseteeninduslikult vahetada oma paroole ilma administraatori sekkumiseta ja sõltumata sellest, kas nad teavad oma hetkel kehtivat parooli. Tarkvaral oli ka eesmärk aidata ja julgustada kasutajatel kasutada tugevaid paroole. Selleks pidi rakendus olema võimeline mõõtma kasutaja sisestatud parooli tugevust ning vastavalt administraatori seadistusele paroolide minimaalse tugevuse nõudeid jõustama. Loodava lahenduse näol pidi olema tegemist veebipõhise kasutajaliidesega. Loodud rakendus pidi olema piisavalt paindlik, et seda saaks kasutada paljude erinevate süsteemidega.

Kõigi nende eesmärkide täitmiseks oli vaja uurida millised parooli omadused tugevdavad ja nõrgendavad parooli ning kuidas genereerida tugevaid paroole. Lisaks oli vaja teada, kuidas on võimalik genereerida kasutaja jaoks tugevat parooli. Kuna rakendus tegeleb paroolide käsitlemisega, siis pidi ka uurima, kuidas seda saab teha turvaliselt. Loodud rakendus seati üles Informaatikainstituudi ruumide broneerimise süsteemiga ning TTÜ uue tarkvara allalaadimise süsteemiga ning töötas vastavalt ootustele. Seega võib käesoleva eesmärke lugeda täidetuks.

Töö tulemusena loodi kasutusvalmis veebirakendus, mis täidab eelpool mainitud eesmärke. See veebirakendus avaldati GitHub-i keskkonnas avatud lähtekoodiga ning on alla laetav järgneval aadressil.

https://github.com/Pisi-Deff/PasswordManager

Sellel leheküljel asub ka rakenduse lühike ingliskeelne tutvustus ja kasutusjuhend. Veebirakenduse lähtekood avaldati MIT litsentsiga.

Töös kirjeldati, milliseid arhitektuurilisi otsuseid võeti vastu rakenduse loomiseks. Töös on välja toodud ka loodud rakenduse võimalikke täiendusi ja edasiarendusi, mis suurendaksid rakenduse kasutusmugavust ja sobivust erinevate süsteemidega. Loodud rakendust võrreldi teiste leitud sarnaste eesmärkidega rakenduste ja moodulitega.

Kasutatud kirjandus

[1] „Mozilla Developer Network – AJAX – Getting Started,“ [WWW]. https://developer.mozilla.org/en-US/docs/AJAX/Getting_Started. [11 Mai 2016].

[2] „E-teatmik,“ [WWW]. http://www.vallaste.ee. [12 Märts 2006].

[3] „PHP,“ [WWW]. http://php.net/. [16 Mai 2016].

[4] A. Rinde, „Veebi kujunduse elutsükkel,“ [WWW]. http://www.cs.tlu.ee/~rinde/www_materjal/veebi_elutsykkel.pdf. [14 Aprill 2016].

[5] „Wikipedia – Teek,“ [WWW]. https://et.wikipedia.org/wiki/Teek. [14 Mai 2016].

[6] A. Crenshaw, „Of History & Hashes: A Brief History of Password Storage, Transmission, & Cracking,“ Mai 2015. [WWW]. https://www.trustedsec.com/may-2015/passwordstorage/. [30 Märts 2016].

[7] G. Khalil, „Password Security-- Thirty-Five Years Later,“ 2014. [WWW]. http://www.sans.org/reading-room/whitepapers/basics/password-security-thirty-five-years-35592. [04 Aprill 2016].

[8] S. Rosenblatt ja J. Cipriani, „Two-factor authentication: What you need to know (FAQ),“ [WWW]. http://www.cnet.com/news/two-factor-authentication-what-you-need-to-know-faq/. [10 Mai 2016].

[9] „ID-kaardi elektrooniline kasutamine,“ [WWW]. http://www.id.ee/index.php?id=30108. [15 Mai 2016].

[10] „Google 2-Step Verification,“ [WWW]. https://www.google.com/landing/2step/. [12 Mai 2016].

[11] „Steam Support – Account Security Recommendations,“ [WWW]. https://support.steampowered.com/kb_article.php?ref=1266-OAFV-8478. [12 Mai 2016].

[12] M. Scott, „A Brief History of Authentication,“ [WWW]. http://www.miracl.com/blog/a-brief-history-of-authentication. [18 Aprill 2016].

[13] „PKI ehk avaliku võtme infrastruktuur,“ [WWW]. https://courses.cs.ut.ee/2013/infoturve/fall/Main/PKIEhkAvalikuV%C3%B5tmeInfrastruktuur. [12 Aprill 2016].

[14] „Google Sign-In,“ [WWW]. https://developers.google.com/identity/. [12 Mai 2016].

[15] „Facebook Login,“ [WWW]. https://developers.facebook.com/docs/facebook-login. [12 Mai 2016].

[16] W. E. Burr, D. F. Dodson ja W. T. Polk, „NIST Special Publication 800-63 – INFORMATION SECURITY,“ Juuni 2004. [WWW]. http://www.usda.gov/egov/egov_redesign/intranet/eauth/SP800-63V6.pdf. [6 Märts 2016].

[17] D. Eastlake, J. Schiller ja S. Crocker, „RFC4086 – Randomness Requirements for Security,“ Juuni 2005. [WWW]. https://tools.ietf.org/html/rfc4086. [28 Aprill 2016].

[18] „Passwords - the why and how?,“ [WWW]. http://www.infosec.qmul.ac.uk/good-practice/77136.pdf. [14 Mai 2016].

[19] „Password Guidelines,“ [WWW]. http://www.lockdown.co.uk/?pg=password_guide. [15 Mai 2016].

[20] „Unmasked: What 10 million passwords reveal about the people who choose them,“ [WWW]. http://wpengine.com/unmasked/. [15 Mai 2016].

[21] „Password Vs Passphrase: Here’s 5 Reasons to Use Passphrase,“ [WWW]. http://www.passworddragon.com/password-vs-passphrase. [15 Mai 2016].

[22] M. Ghazvininejad ja K. Knight, „How to Memorize a Random 60-Bit String,“ [WWW]. http://www.isi.edu/natural-language/mt/memorize-random-60.pdf. [18 Mai 2016].

[23] „Mozilla Developer Network – <input>,“ [WWW]. https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input. [05 Aprill 2016].

[24] „Hypertext Transfer Protocol – HTTP/1.1 – Method Definitions,“ [WWW]. https://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html. [01 Mai 2016].

[25] „What is HTTPS?,“ [WWW]. https://www.instantssl.com/ssl-certificate-products/https.html. [12 Aprill 2016].

[26] „Krüptograafiliste algoritmide elutsükli uuring,“ Cybernetica AS, 3 Juuni 2015. [WWW]. https://www.ria.ee/public/RIA/Kruptograafiliste_algoritmide_uuring_2015.pdf. [21 Mai 2016].

[27] B. Schneier, „Cryptography: The Importance of Not Being Different,“ [WWW]. https://www.schneier.com/essays/archives/1999/03/cryptography_the_imp.html. [15 Mai 2016].

[28] „Salted Password Hashing - Doing it Right,“ [WWW]. https://crackstation.net/hashing-security.htm. [15 Mai 2016].

[29] „PHP Documentation – crypt,“ [WWW]. http://php.net/manual/en/function.crypt.php. [14 Mai 2016].

[30] „SMS|Passcode Password Reset Module,“ [WWW]. http://www.smspasscode.com/what-we-do/password-reset-module/. [18 Mai 2016].

[31] „Password Reset module for ProcessWire,“ [WWW]. https://github.com/plauclair/PasswordReset. [18 Mai 2016].

[32] „Forgot Password for User Management System,“ [WWW]. http://codecanyon.net/item/forgot-password-for-user-management-system/2516786. [18 Mai 2016].

[33] „ServiceNow Password Reset,“ [WWW]. http://wiki.servicenow.com/?title=Password_Reset. [18 Mai 2016].

[34] „Balsamiq,“ [WWW]. https://balsamiq.com/. [14 Aprill 2016].

[35] „12Dicts,“ [WWW]. http://wordlist.aspell.net/12dicts/. [12 Mai 2016].

[36] „PHP Documentation,“ [WWW]. http://php.net/manual/en/. [18 Aprill 2016].

[37] „Doctrine DBAL 2 documentation – Introduction,“ [WWW]. http://docs.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/introduction.html. [12 Mai 2016].

[38] „Doctrine DBAL 2 documentation – Configuration,“ [WWW]. http://docs.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/configuration.html#driver. [12 Mai 2016].

[39] „Doctrine DBAL – composer.json,“ [WWW]. https://github.com/doctrine/dbal/blob/v2.5.4/composer.json. [15 Mai 2016].

[40] „Doctrine DBAL 2 documentation – Security,“ [WWW]. http://docs.doctrine-project.org/projects/doctrine-dbal/en/latest/reference/security.html. [10 Mai 2016].

[41] „PHP Documentation – mail,“ [WWW]. http://php.net/manual/en/function.mail.php. [04 Aprill 2016].

[42] „PHPMailer,“ [WWW]. https://github.com/PHPMailer/PHPMailer. [18 Aprill 2016].

[43] „Apache HTTP Server Version 2.4 Documentation,“ [WWW]. https://httpd.apache.org/docs/current/howto/htaccess.html. [01 Mai 2016].

[44] „Apache HTTP Server 2.4 Documentation – AllowOverride,“ [WWW]. https://httpd.apache.org/docs/2.4/mod/core.html#allowoverride. [15 Mai 2016].

[45] „Apache HTTP Server 2.4 Documentation – Mapping URLs to Filesystem Locations,“ [WWW]. https://httpd.apache.org/docs/2.4/urlmapping.html#documentroot. [15 Mai 2016].

[46] „PHPUnit,“ [WWW]. https://phpunit.de/. [10 Mai 2016].

[47] G. –. About, „https://github.com/about,“ [WWW]. [12 Aprill 2016].

[48] „Choose a License – MIT License,“ [WWW]. http://choosealicense.com/licenses/mit/. [02 Mai 2016].

[49] M. Burnett, „Today I Am Releasing Ten Million Passwords,“ 10 Veebruar 2015. [WWW]. https://xato.net/today-i-am-releasing-ten-million-passwords-b6278bbe7495. [12 Mai 2016].

Lisa 1 – Seadistuse parameetrite kirjeldused

Tabelis 1 kirjeldatud parameetrid on seotud andmebaasiga, milles hoitavate kasutajate ja paroolidega haldamisega rakendus tegeleb.

Tabel 1. Andmebaasiga seotud seadistuse parameetrid.

Parameeter	Kirjeldus	Näiteväärtus(ed)
db_type	Andmebaasisüsteem.	„postgresql“, „oracle“, „mssql“
db_host	Andmebaasiserveri hostinimi.	„localhost“, „db.ttu.ee“
db_port	Andmebaasiserveri võrgupordi number. Kui väärtus on *null*, siis kasutatakse valitud tüüpi andmebaasisüsteemi vaikimisi võrguporti.	*null*, „1234“
db_database	Andmebaasi nimi, kus hoitakse põhirakenduse kasutajaandmeid.	„maindb“
db_user	Kasutajanimi, mida rakendus kasutab andmebaasisüsteemis enda autentimiseks. Kui andmete käsitlemiseks kasutatakse funktsioone (parameeter *db_useDBFunctions* on tõese väärtusega), siis peab antud kasutajal olema õigus vastavaid funktsioone välja kutsuda. Kui ei kasutata funktsioone, siis peab kasutajal olema õigus lugeda ja uuendada parameetriga *db*_userTable defineeritud kasutajate tabelit.	„kasutaja“

Parameeter	Kirjeldus	Näiteväärtus(ed)
db_password	Parool, mida rakendus kasutab andmebaasisüsteemis enda autentimiseks.	„parool“
db_userTable	Tabeli nimi andmebaasis, kus hoitakse kasutajate andmeid ja milles on järgnevate parameetritega määratud veerud. Kui tabelile pääseb ligi ainult kindla skeemi kaudu, siis tuleb selle skeemi nimi lisada tabeli nime algusesse ja eraldada tabeli nimest punktiga. Kasutatakse ainult siis kui andmete käsitlemiseks ei kasutata funktsioone (parameeter *db_useDBFunctions* on väärtusega *false*). Rakendus eeldab, et kasutajate andmed (kasutajanimi, meiliaadress, parool) on koos ühes tabelis. Kui need on laiali mitmes tabelis, siis tuleb paroolivahetuse tööle saamiseks kasutada andmebaasiserveris talletatud rutiine, millega paroolide vahetamise rakendus suhtleb.	„users“, „schema1.users“
db_username Column	Veeru nimi kasutajate tabelis, kus hoitakse kasutajanimesid. Kui rakendus kasutab meiliaadressi kasutajanimena, siis peavad selle ja järgneva parameetri väärtused kokku langema. Kasutatakse ainult siis kui andmebaasiga suhtlemisel ei kasutata funktsioone (parameeter *db_useDBFunctions* on väära väärtusega).	„username“

Parameeter	Kirjeldus	Näiteväärtus(ed)
db_email Column	Veeru nimi kasutajate tabelis, kus hoitakse kasutaja meiliaadressi. Kasutatakse ainult siis kui andmebaasiga suhtlemisel ei kasutata funktsioone (parameeter *db_useDBFunctions* on väära väärtusega).	„email“
db_password Column	Veeru nimi kasutajate tabelis, kus hoitakse kasutaja parooli. Kasutatakse ainult siis kui andmebaasiga suhtlemisel ei kasutata funktsioone (parameeter *db_useDBFunctions* on väära väärtusega).	„password“
db_password HashMethod	Meetod, mida kasutatakse parooli räsimiseks. Räsimist teostatakse ainult siis, kui parool salvestatakse otse andmebaasi *db_userTable* parameetriga määratud tabelisse või kui parooli vahetamiseks kasutatakse andmebaasi funktsiooni (vt. parameeter *db_changePasswordFunction*) ja parameeter *db_useHashedPasswordForFunctions* on tõese väärtusega. Vajadusel on toetatud ka süsteemid, mis hoiavad paroole avatekstina. Selleks tuleb kasutada väärtust *null*.	„bcrypt“, „sha256“, „sha512“, *null*

Parameeter	Kirjeldus	Näiteväärtus(ed)
db_password Changed EventFunction	Andmebaasis oleva funktsiooni nimi, mis kutsutakse välja siis kui kasutaja parool on muudetud. Kasulik kui andmebaasis on vaja pärast parooli muutmist mingit äriloogikat käivitada. Näiteks kustutada kasutaja sessioone. Selle funktsiooni kasutamine ei sõltu parameetri *db_useDBFunctions* väärtusest. Funktsiooni ainus sisend on kasutajanimi, millega seotud kasutaja parool vahetati. Funktsioonil ei ole väljundit. Funktsiooni näidisrealisatsioon on nähtav joonisel Joonis 12.	„fn_ userPassChanged“, „schema1.fn_ pwChangeEvent“
db_useDB Functions	Kas kasutada kasutaja autentimiseks, kasutaja meiliaadressi lugemiseks ja parooli vahetamiseks andmebaasi funktsioone (*true*) või informatsiooni otse tabelist lugeda ja tabelisse kirjutada (*false*).	*true, false*
db_useHashed PasswordFor Functions	Kas parooli vahetamise ja autentimise funktsioonidele antakse parool üle algkujul (*false*) või räsitakse see enne kasutades *db_passwordHashMethod* parameetriga määratud räsimeetodit (*true*). Parooli algkujul üleandmine võimaldab soovi korral räsimist teha veebiserveri asemel andmebaasiserveris. Kasutatakse ainult siis kui andmebaasiga suhtlemisel kasutatakse funktsioone (parameeter *db_useDBFunctions* on tõese väärtusega).	*true, false*

Parameeter	Kirjeldus	Näiteväärtus(ed)
db_getUser Email Function	Andmebaasis oleva funktsiooni nimi, mis kutsutakse välja antud kasutajanimega kasutaja meiliaadressi kättesaamiseks. Käitub ka kasutaja olemasolu kontrollina. Kui funktsioon ei tagasta väärtust, siis võetakse seda kui antud kasutajanimega kasutaja puudumist andmebaasis. Funktsiooni ainus sisend on kasutajanimi. Funktsiooni väljundiks on kasutaja meiliaadress. Funktsiooni näidisrealisatsioon on nähtav joonisel Joonis 13. Kasutatakse ainult siis kui andmebaasiga suhtlemisel kasutatakse funktsioone (parameeter *db_useDBFunctions* on tõese väärtusega).	„fn_getUser Email“, „schema1. fn_getUserEmail“
db_change Password Function	Andmebaasis oleva funktsiooni nimi, mis kutsutakse välja antud kasutajanimega kasutaja parooli vahetamiseks. Funktsiooni esimene sisend on kasutajanimi ja teine sisend on uus parool. Funktsioonil puudub väljund. Funktsiooni näidisrealisatsioon on nähtav joonisel Joonis 14. Kasutatakse ainult siis kui andmebaasiga suhtlemisel kasutatakse funktsioone (parameeter *db_useDBFunctions* on tõese väärtusega).	„fn_changePass“, „schema1. fn_changePass“

Parameeter	Kirjeldus	Näiteväärtus(ed)
db_user Authenticate Function	Andmebaasis oleva funktsiooni nimi, mis kutsutakse välja kontrollimaks, kas antud kasutajanimi ja parool vastavad eksisteerivale kasutajale. Kasutatakse siis, kui kasutaja tahab endale teadaolevat parooli vahetada. Funktsiooni esimene sisend on kasutajanimi ja teine sisend on parool. Funktsiooni väljundiks on tõene kui autentimine õnnestus ning väär vastasel juhul. Funktsiooni näidisrealisatsioon on nähtav joonisel Joonis 15. Kasutatakse ainult siis kui andmebaasiga suhtlemisel kasutatakse funktsioone (parameeter *db_useDBFunctions* on tõese väärtusega).	„fn_userAuth“, „schema1. fn_userAuth“

Tabelis 2 kirjeldatud seadistuse parameetritega määratakse, kuidas kasutajaliidesel genereeritakse paroole (vt jaotis 6.3.4).

Tabel 2. Parooli genereerimisega seotud seadistuse parameetrid.

Parameeter	Kirjeldus	Näiteväärtus(ed)
pwgen_dictionary FilePath	Absoluutne aadress failisüsteemis sõnaraamatu failini. Väärtus on kohustuslik. Faili struktuur on kirjeldatud jaotises 6.3.4.	„/home/sitename/dict.txt“
pwgen_wordsNumber	Mitu sõna parooli loomiseks kasutatakse. Väärtus peab olema vähemalt 1. Soovituslik väärtus on 4, sest selle väärtusega genereeritud paroolide Shannoni entroopia on katsetuste kohaselt enamasti üle 100 biti.	4

Tabelis 3 kirjeldatud seadistuse parameetritega määratakse, mis reegleid rakendatakse kontrollimaks, kas kasutaja poolt valitud parool on piisavalt tugev.

Tabel 3. Parooli tugevuse reeglitega seotud seadistuse parameetrid.

Parameeter	Kirjeldus	Näiteväärtus(ed)
pw_minLength	Parooli minimaalne lubatud pikkus tähemärkides.	8
pw_maxLength	Parooli maksimaalne lubatud pikkus tähemärkides. Soovituslikult võimalikult suur number. Kui väärtus on number null, siis ei piirata parooli maksimaalpikkust. Vajalik näiteks räsialgoritmide jaoks, millel on parooli pikkus piiratud.	0, 64
pw_minEntropyBits	Parooli minimaalne entroopia (vt jaotis 3.1) bittides. Soovituslik väärtus on 60.	60, 80
pw_strongEntropyBits	Parooli soovituslik entroopia (vt jaotis 3.1) bittides. Peaks olema suurem kui minimaalne entroopia bittide arv. Soovituslik väärtus on 100.	100

Tabelis 4 kirjeldatud parameetritega seadistatakse meilide väljasaatmise funktsionaalsus.

Tabel 2. Meilide saatmisega seotud seadistuse parameetrid.

Parameeter	Kirjeldus	Näiteväärtus(ed)
email_host	Meiliserveri hostinimi.	„mail.ttu.ee“
email_port	Meiliserveri võrguport. Kui väärtus on *null*, siis kasutatakse valitud tüüpi meiliprotokolli vaikimisi võrguporti.	*null*, 25
email_type	Meilisaatmiseks kasutatav protokoll.	„smtp“, „pop3“
email_use Authentication	Kas kasutada meiliserveriga suhtlemisel autentimist või mitte.	*true, false*
email_authUsername	Kasutajanimi meiliserveriga autentimisel.	„mailuser“
email_authPassword	Parool meiliserveriga autentimisel.	„password“
email_encryption	Millist krüpteeringut kasutada meiliserveriga suhtlemisel. Märge: PHPMailer toetab krüpteeringut ainult SMTP protokolli kasutades [42].	„ssl“, „tls“, *null*
email_mailerAddress	Rakenduse poolt väljasaadetud meilide saatja meiliaadress.	„noreply@example.com“

Tabelis 5 kirjeldatud seadistuse parameetrid on seotud rakenduse üldise töötamisega ning ei sobi ühegi eelneva parameetrite grupi koosseisu.

Tabel 3. Rakenduse üldise seadistuse parameetrid.

Parameeter	Kirjeldus	Näiteväärtus(ed)
applicationName	Süsteemi nimi, mille kasutajate paroolidega rakendus tegeleb. Kasutatakse näiteks saadetud meilides ja lehekülgede pealkirjades.	„TTÜ Informaatikainstituudi ruumide broneerimise süsteem“, „Maurus“
instanceIdentifier	Rakenduse instantsi unikaalselt identifitseeriv tähemärgijada. Kui samal veebiserveril töötab mitu selle rakenduse instantsi, siis on vajalik neid selle parameetri väärtusega eristada, et nad üksteise salvestatud andmeid ei puutuks. Väärtus peab koosnema vaid tähtedest ja numbritest. Olenevalt serveri operatsioonisüsteemist võib selle väärtus olla tundlik suur- ja väiketähtede suhtes, mistõttu on soovituslik vältida identifikaatoreid, mis erinevad ainult suur- ja väiketähtede poolest. Kuna väärtust kasutatakse failinimede moodustamiseks, siis ei tohiks see olla liiga pikk, et vältida failisüsteemi piiranguid.	„sitename“
logsFolder	Absoluutne failisüsteemi aadress kaustani, kuhu kirjutatakse rakenduse logid (vt jaotis 6.4.4). Soovitatav on see kaust asetada väljaspoole veebiserveri avalikku kausta. Kui väärtus on *null*, siis ei kirjuta rakendus logisid.	„/home/sitename/logs“

Lisa 2 – Andmebaasiga suhtlemiseks kasutatud funktsioonide näidisimplementatsioonid PostgreSQL-i andmebaasisüsteemi süntaksiga

Järgnevad näidisfunktsioonid realiseerivad jaotises 0 kirjeldatud funktsioone. Funktsioonide ja parameetrite nimed ei ole olulised. Parameetrite tüübid ja järjekord on olulised.

```
CREATE OR REPLACE FUNCTION fn_pwChangeEvent
  (username TEXT)
  RETURNS VOID
  AS $$
    DELETE FROM sessions s
      WHERE s.user_id = (
        SELECT u.id FROM users u
          WHERE u.username = $1
      )
  $$
  LANGUAGE SQL SECURITY DEFINER;
```

Joonis 12. Kasutaja parooli muutmisele reageerimise funktsiooni lihtne näidis PostgreSQL-i andmebaasisüsteemi süntaksit kasutades.

```
CREATE OR REPLACE FUNCTION fn_getUserEmail
  (username TEXT)
  RETURNS TEXT
  AS $$
    SELECT u.email FROM users u
      WHERE u.username = $1
  $$
  LANGUAGE SQL SECURITY DEFINER;
```

Joonis 13. Kasutaja meiliaadressi kättesaamise funktsiooni lihtne näidis PostgreSQL-i andmebaasisüsteemi süntaksit kasutades.

```
CREATE OR REPLACE FUNCTION fn_setPass
  (username TEXT, password TEXT)
  RETURNS VOID
  AS $$
    UPDATE users u
      SET u.password = $2
      WHERE u.username = $1
  $$
  LANGUAGE SQL SECURITY DEFINER;
```

Joonis 14. Kasutaja parooli määramise funktsiooni lihtne näidis PostgreSQL-i andmebaasisüsteemi süntaksit kasutades.

```
CREATE OR REPLACE FUNCTION fn_userAuth
  (username TEXT, password TEXT)
  RETURNS BOOLEAN
  AS $$
    SELECT u.password = $2 FROM users u
      WHERE u.username = $1
  $$
  LANGUAGE SQL SECURITY DEFINER;
```

Joonis 15. Kasutaja autentimise funktsiooni lihtne näidis PostgreSQL-i andmebaasisüsteemi süntaksit kasutades.

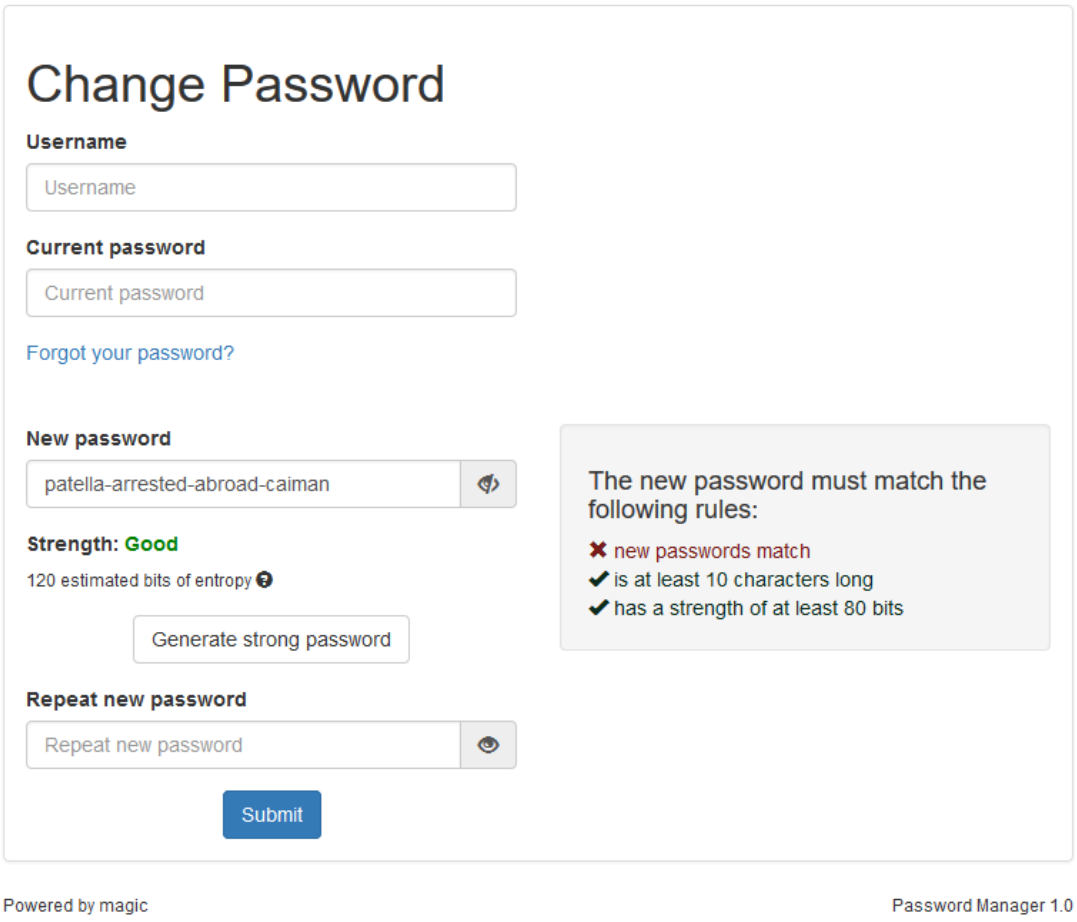
Lisa 3 – PHP-sse sisseehitatud meilide saatmise funktsiooni puudused

Järgnevalt tuuakse välja mõned PHP-sse sisseehitatud *mail* funktsiooni olulised puudused [41]:

- Meili saatmiseks kasutatakse PHP seadistuses ja veebiserveril oleva *sendmail* rakenduse seadistuses määratud meiliserverit. Rakenduses endas on meilide saatmise konfigureerimine väga piiratud.
- Meili päised (ingl *header*) tuleb kõik käsitsi üles seada. Kuna meilide koostamisel on väga palju standardiseeritud reegleid, siis teeb see meilide saatmise keerulisemaks.
- Keeruline on saata meile, millel on nii HTML versioon kui ka avatekstina versioon.
- Windowsi operatsioonisüsteemi kasutavatel serveritel tuleb mõnikord funktsiooni töölesaamiseks teha keerulisi muudatusi PHP seadistuses.
- Funktsiooni töö käigus tekkinud vigu ei ole lihtne programselt kinni püüda ja käsitleda.

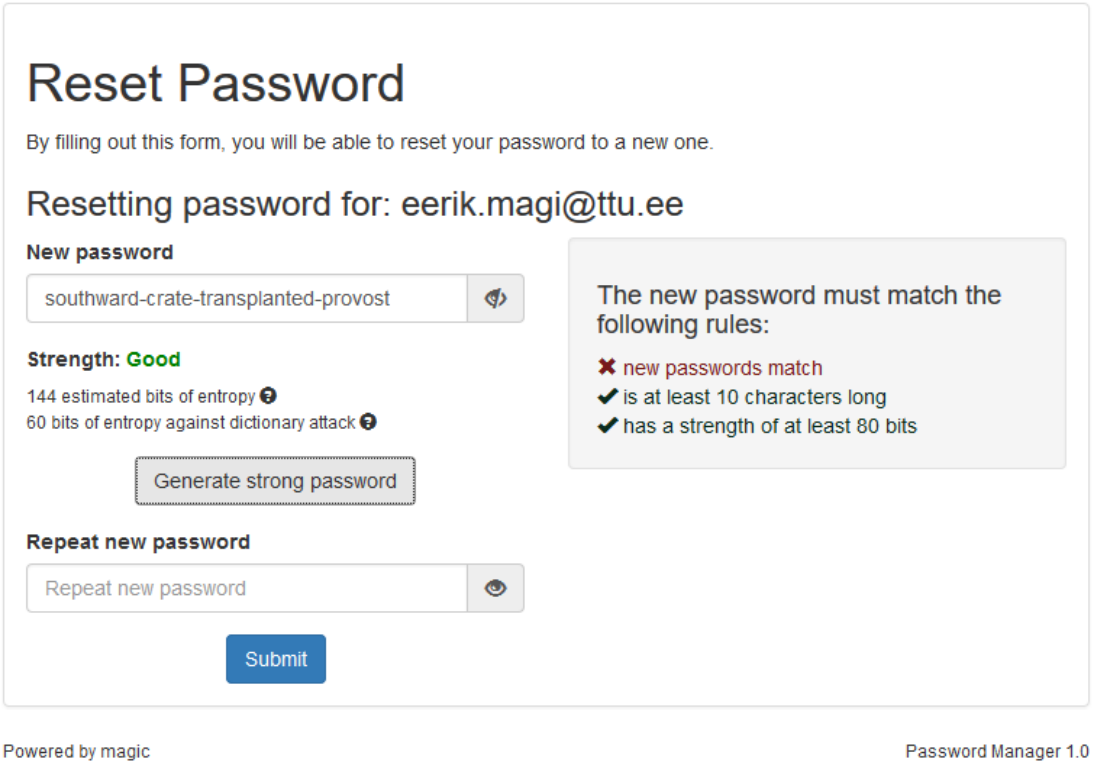
Lisa 4 – Ekraanitõmmised rakenduse välimusest

Järgnevalt on välja toodud mõned ekraanitõmmised töö käigus loodud rakenduse välimusest. Joonisel 16 on välja toodud paroolide vahetamise lehekülg.



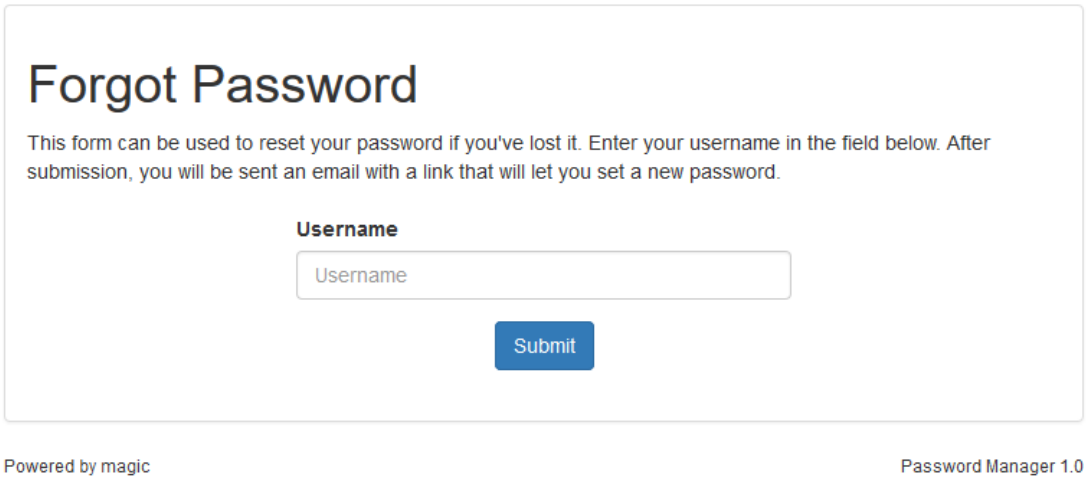
Joonis 16. Ekraanitõmmis parooli vahetamise lehekülje välimusest.

Joonisel 17 on näidatud parooli nullimise lehekülg.



Joonis 17. Ekraanitõmmis parooli nullimise lehekülje välimusest.

Joonisel 18 on näidatud ununenud parooli lehekülg.



Joonis 18. Ekraanitõmmis ununenud parooli lehekülje välimusest.

Lisa 5 – Kasutajate, õiguste ja funktsioonide ülesseadmiseks reaalsete süsteemide andmebaasides kasutatud SQL laused

Järgnevalt on välja tootud reaalsete süsteemide, mille puhul loodud tarkvara üles seati, andmebaasides käivitatud SQL laused, millega loodi rakenduse jaoks kasutajad, määrati kasutajate õigused ning loodi funktsioonid. Nendes lausetes on ära muudetud kasutajate, andmebaaside ja tabelite nimed ning paroolid. Õiguste andmisel on lähtutud minimaalsuse põhimõttest ehk on kasutajatele antud vaid need õigused, mida rakendus töötamiseks vajab.

Süsteem 1 (paroolivahetuse rakendus loeb/muudab andmeid otse tabelis):

```
CREATE USER kasutajanimi WITH PASSWORD 'parool';
REVOKE CONNECT, TEMP ON DATABASE andmebaasi_nimi FROM PUBLIC;
REVOKE CREATE, USAGE ON SCHEMA public FROM PUBLIC;
REVOKE USAGE ON LANGUAGE plpgsql FROM PUBLIC;

GRANT CONNECT ON DATABASE andmebaasi_nimi TO kasutajanimi;
GRANT USAGE ON SCHEMA public TO kasutajanimi;

GRANT SELECT, UPDATE (email, password_hash) ON tabeli_nimi TO kasutajanimi;
```

Süsteem 2 (paroolivahetuse rakendus loeb/muudab andmeid andmebaasis loodud funktsioonide kaudu):

```
CREATE USER kasutajanimi WITH PASSWORD 'parool';
REVOKE CONNECT, TEMP ON DATABASE andmebaasi_nimi FROM PUBLIC;
REVOKE CREATE, USAGE ON SCHEMA public FROM PUBLIC;
REVOKE USAGE ON LANGUAGE plpgsql FROM PUBLIC;

CREATE OR REPLACE FUNCTION f_leia_isiku_email (tabeli_nimi.email%TYPE)
RETURNS TEXT
AS $$
 SELECT email FROM tabeli_nimi WHERE email = $1;
$$
LANGUAGE SQL SECURITY DEFINER
SET search_path = public, pg_temp;
```

```
CREATE OR REPLACE FUNCTION f_muuda_isiku_parooli(tabeli_nimi.email%TYPE,
tabeli_nimi.parool%TYPE)
RETURNS VOID
AS $$
 UPDATE tabeli_nimi SET parool = public.crypt($2, public.gen_salt('bf', 11))
WHERE email = $1
$$
LANGUAGE SQL SECURITY DEFINER
SET search_path = public, pg_temp;

REVOKE EXECUTE ON FUNCTION
f_leia_isiku_email (tabeli_nimi.email%TYPE),
f_muuda_isiku_parooli(tabeli_nimi.email%TYPE, tabeli_nimi.parool%TYPE),
f_identifitseeri_kasutaja(
 s_email tabeli_nimi.email%TYPE,
 s_parool VARCHAR)
FROM PUBLIC;

GRANT CONNECT ON DATABASE andmebaasi_nimi TO kasutajanimi;
GRANT USAGE ON SCHEMA public TO kasutajanimi;

GRANT EXECUTE ON FUNCTION
f_leia_isiku_email (tabeli_nimi.email%TYPE),
f_muuda_isiku_parooli(tabeli_nimi.email%TYPE, tabeli_nimi.parool%TYPE),
f_identifitseeri_kasutaja(
 s_email tabeli_nimi.email%TYPE,
 s_parool VARCHAR
)
TO kasutajanimi;
```

Lisa 6 – Logifaili näidissisu

Järgnev lõik on võetud ühe süsteemi, millega koos loodud rakendust kasutatakse, logifailist. Teksti on muudetud, et eemaldada kasutajate kasutajanimed ja IP aadressid.

```
01:54:20 (EEST)|256.123.123.123|> Failed password change attempt due to
invalid password and/or username with username "".
01:54:36 (EEST)|256.123.123.123|> Password reset initiated for user
"kasutaja@example.com".
01:55:36 (EEST)|256.123.123.123|> Password reset successfully finished for
user "kasutaja@example.com".
21:25:04 (EEST)|256.123.123.123|> Password reset initiated for user
"kasutaja@example.com".
21:26:27 (EEST)|256.123.123.123|> Password reset successfully finished for
user "kasutaja@example.com".
21:26:32 (EEST)|256.123.123.123|> Attempt to reset password with invalid key.
```