2) Broken authentication and session management

Korisnici na sistem se prijavljuju upotrebom korisničkog imena tj. email-a i lozinke koje su uneli prilikom registracije. Kako bi se sprečilo korišćenje jednostavnih lozinki, korisnik pri registraciji mora da unese niz od osam karaktera pri čemu su dozvoljene samo cifre i mala i velika slova. Takođe mora uneti bar jedno veliko slovo i bar jednu cifru. Prilikom registrovanja, korisnik mora dva puta uneti lozinku kako bi potvrdio da želi baš tu lozinku.

Lozinke korisnika se čuvaju u XML bazi pri čemu su one transformisane upotrebom hash i salt metoda.

Implementirana web aplikacija za komunikaciju izmdu svog frontend-a i backend-a koristi HTTPS protokol tj. TLS čime je obezbeđena tajnost podataka koji se razmenjuju na datoj relaciji. Time je obezbeđena sigurnost korisničkog imena i lozinke koji se prenose prilikom registrovanja i prijavljivanja na sistem.

Radi identifikovanja prijavljenog korisnika se ne koristi sesija već JWT (Java Web Token) mehanizam. Time su izbegnuti problemi koje sesija za sobom nosi (upotrebom XSS-a se može ukrasti sesija). Token koji browser dobija od servera se čuva u samom browser-u (Local Storage). Taj token se prilikom odjavljivanja korisnika briše.

1) Injection

Opasnost od injection napada prilikom pretrage po metapodacima (datumu kreiranja i datumu usvajanja) je odklonjena pošto se na serverskoj strani vrši provera da li uneti podaci za datum kreiranja formiraju datum i da li uneti podaci za datum usvajanja formiraju validan datum.

Prilikom popunjavanja formi za kreiranje akta ili amandmana veoma lako može doći do injection napada, ako se naravno ne izvrši zaštita. Problem pretstavlja mogućnost korisnika da unese neke xml tagove i time promeni strukturu rezultujućeg dogumenta. U te tagove može ubaciti neku skriptu koja će se izvršiti prilikom preuzimanja (od strane korisnika) odgovarajućeg dokumenta iz baze podataka. Stoga bi XML dokumenat bilo potrebno validirati. Jedan od načina je uz pomoć odgovarajuće XML šeme pre slanja u bazu podataka. Međutim u aplikciji je implementirano upotrebom biblioteke Jsoup-1.9.2.jar i koričćenjem izraza oblika String cleanPostPayload = Jsoup.*clean*(postPayload, Whitelist.*basic*()); gde je cleanPostPayload prešićeni string iz promenjive postPayload. Iz stringa će biti uklonjeni svi elementi koji mogu izazvati injection.