Dokumentácia bezpečného testovacieho systému

Jozef Komáromy

April 29, 2024

1 Prehľad problému

Účelom tohto projektu je vytvoriť testovací systém pre školu, ktorý zabezpečí počítač testovaného študenta tak, aby nemohol podvádzať. Vo väčšine prípadov prebieha digitálne testovanie prostredníctvom web-aplikácie v prehliadači (napr. Google Forms, Moodle, a pod.). Tieto spôsoby testovania sú často nezabezpečené proti najjednoduchším spôsobom podvádzania. Testovaná osoba si napríklad môže jednoducho otvoriť ďalší tab a vyhladávať správne odpovede na Googli. Aby sme vyriešili problém podvádznia bez priameho zásahu do testovacieho prostredia, potrebujeme vyvinúť aplikáciu, ktorá nahradí úlohu tradičného webového prehliadaču pri testovaní. Testovacie prostredie teda bude spustené cez túto aplikáciu. Možnosti testovanej osoby sú obmedzené tak, že nemôže na počítači zobrazovať iné aplikácie, a prechádzať na web-stránky ktoré mu neboli explicitne povolené.

2 Riešenie

Mojím riešením tohto problému je sada softvéru pozostavájúca z 4 častí:

1. *Client*
2. *Vanguard*
3. *Server*
4. *Supervisor*

2.1 Client

Aplikácia spustená na počítači testovanej osoby (ďalej študent), v ktorej je spustené webové testovacie prostredie. Po otvorení aplikácie má študent možnosť pripojiť sa do miestnosti vytvorenej dohliadajúcou osobou (ďalej učiteľ). Po pripojení do miestnosti sa načíta konfigurácia pre túto miestnosť a zobrazí sa testovacie prostredie. Taktiež sa spustí program Vanguard, ktorý upozorňuje na podozrivé správanie študenta. Informácie o podozrivom správaní sa ďalej posielajú na Server.

2.2 Vanguard

Pomocný program pre testovaciu aplikáciu, ktorý sleduje akcie študenta vrámci celého operačného systému. Ak vyhodnotí že správanie študenta je podozrivé, posúva túto informáciu do aplikácie Client. Zahŕňa viacero služieb, ktoré sledujú napr. aké okno je v popredí systému.

2.3 Server

Softvér bežiaci na vzdialenom zariadení, slúži zväčša ako komunikačná vrstva medzi aplikáciami Client a Supervisor. Taktiež sa tu uchovávajú informácie o miestnostiach, a študentoch v nich. Poskytuje aj bezpečné prihlásenie učiteľa do systému, a pripojenie študenta do miestnosti.

2.4 Supervisor

Webová aplikácia ktorá učiteľovy poskytuje zobrazenie informácií získaných zo Serveru. Učiteľ sa prihlási do systému a následne môže vytvárať miestnosti, monitorovať študentov pripojených do miestnosti, a diaľkovo ovládať ich klientskú aplikáciu.

3 Špecifikácia

3.1 Client

3.2 Vanguard

3.3 Server

3.3.1 Technológie

Server je implementovaný v programovacom jazyku *PHP* pre webový server *Apache* a databázu *MySQL/MariaDB*. Pre komunikáciu s aplikáciou klienta a applikáciou Supervisor sa používajú *POST* requesty cez protokol *HTTP(S)* kde aj klient a server posielajú dáta v formáte *JSON*.

3.3.2 Konfigurácia

3.3.3 Databáza

Databázová schéma obsahuje nasledovné entity:

3.3.4 Vendor

PHP programy v adresáry */vendor* niesu prístupné verejnosti, slúžia ako knižnica pre endpointy.

autoload.php - Automaticky načítaný pre každý request na endpoint, načíta premenné prostredia zo súboru *env.ini*, a includne *util.php*.

Endpoint.php (*abstract class*) - Je abstrakciou pre koncový bod servera. Automaticky dekoduje request body z formátu *JSON* do php array. Taktiež zachytáva chyby, aby neboli viditelné pre klienta. Vždy vracia výsledok vo formáte *JSON*, ktorý vždu obsahuje hodnotu *code*, ktorá reprezentuje úspešnosť požiadavky. Ak *code = 0*, znamená to, že operácia prebehla úspešne, nenulové hodnoty znamenajú chybu. Ak nastala chyba jej popis sa nachádza v hodnote *message*.

```
// príklad výsledku úspešnej operácie
{
  "code": 0,
  "data": {...}
}

// príklad výsledku operácie s chybou
{
  "code": 1,
  "message": "Input argument username not provided"
}
```

Response.php (*interface*) - Reprezentuje odpoveď servera, obsahuje kód odpovede a dáta odpovede.

> **ResponseSuccess** - má kód odpovede 0, a dáta reprezentujúce výsledok operácie
>
> **ResponseError** - má kód odpovede 0, a dáta obsahujúce iba správu (*message*), ktorá popisuje chybu ktorá nastala

Database.php (*class*) - Pripojenie na databázu

Auth.php - Kryptografické funkcie na generovanie bezpečných hesiel, kontrola hesla, hashovanie, a taktiež interakcie s databázou týkajúce sa autentikácie používateľa.

UserEndpoint.php (*abstract class*) - Endpoint ktorý najskôr vykoná autentikáciu používateľa, a ak vyhodnotí že je prihlásený, vykoná operáciu. V request body (*JSON*) musí byť poskytnutá hodnota *session*.

```
// príklad vstupu
{ "session": "QWBUP9WcuBt6zVjPo4YOwG3d4SLBzehN" }

// výstup ak nebol poskytnutý session
{ code: 1, message: "Not logged in" }

// výstup ak je session nesprávny
{ code: 1, message: "Invalid session" }
```

3.3.5 Endpointy

3.4 Supervisor