Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Slack plugin pre podporu vedenia záverečných prác

Bakalárska práca

2020
Weiwei Chen

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Slack plugin pre podporu vedenia záverečných prác

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	RNDr. Richard Ostertág, PhD.

Bratislava, 2020
Weiwei Chen

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Weiwei Chen
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Slack plugin pre podporu vedenia záverečných prác.
Slack plugin for management of thesis progress.

Anotácia: Zosúladiť pravidelné týždenné stretávanie vedúceho práce so študentami je niekedy ťažké. Preto môže byť zaujímavé prejsť na stretávanie sa v „elektronickej podobe“ cez komunikačnú platformu Slack. Cieľom práce je vyvinúť rozšírenie pre túto platformu, ktoré bude automaticky realizovať týždenné reporty o postupe študentov a sumarizovať ich vedúcemu prác.

Vedúci: RNDr. Richard Ostertág, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: prof. RNDr. Martin Škoviera, PhD.

Spôsob sprístupnenia elektronickej verzie práce:
bez obmedzenia

Dátum zadania: 29.10.2019

Dátum schválenia: 30.10.2019

doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Chcel by som sa poďakovať svojmu školiteľovi RNDr. Richardovi Ostertágovi, PhD. za odbornú pomoc, vedenie a cenné rady pri písaní mojej práce.

Abstrakt

Zosúladiť pravidelné týždenné stretávanie vedúceho práce so študentami je niekedy ťažké. Preto môže byť zaujímavé prejsť na stretávanie sa v „elektronickej podobe“ cez komunikačnú platformu Slack. Cieľom práce je vyvinúť rozšírenie pre túto platformu, ktoré bude automaticky realizovať týždenné reporty o postupe študentov a sumarizovať ich vedúcemu prác.

Kľúčové slová: Slack, ChatBot, Webová aplikácia

Abstract

Sometimes is difficult to reconcile the regular weekly meeting of the supervisor with the students. Therefore, it maybe interesting to switch to meeting in “electronic form” through the Slack communication platform.

Keywords: Slack, ChatBot, Web application

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Vďaka rýchlemu rozvoju informačných technológií sa v posledných rokoch menia aj pracovné metódy a spôsoby v rôznych oblastiach. Je viditeľne, že nové pracovné metódy s použitím informačných technológií zvyšujú produktivitu a umožňujú rýchle spätné väzby v rámci spolupráce. Napríklad písanie záverečných prác vyžaduje spoluprácu medzi študentom a školiteľom. V niektorých špeciálnych prípadoch nie je možne alebo efektívne stretávať sa pravidelne, možno by bolo dobré nájsť alternatívne spôsoby na stretávanie, napríklad „v elektronickej forme“. Účelom elektronickej platformy Slack je poskytovať online služby a prostredie pre tímovú spoluprácu. Počas konzultácii „v elektronickej forme“ školiteľ chce byť informovaný postupe záverečných prác. Školiteľ môže viesť viac projektov naraz. Preto by bolo zaujímavé tento proces riešiť algoritmicky a zosumarizovať všetky aktuálne stavy o postupe záverečných prác od študentov na jednom mieste.

Práca je rozdelená na tri hlavné časti. Prvá časť sa zoberá popisom problematiky a návrhom a implementáciou Slack aplikácie. Druhá návrhom a implementáciou webovej aplikácie a návrhom databázových systémov. Posledná časť popisuje bezpečnostné problémy pri tvorbe webových aplikácií.

Prvá a druhá kapitola vysvetľuje základne informácie o platforme Slack, ako funguje Slack na základe posielania okamžitých správ a aké sú jeho výhody proti iným podobným technológiám. Následne stručne popisuje spôsob poskytovania služby Slack a vybrané vlastnosti Slack API. Potom popisuje vybrané technológie a nástroje na vytváranie Slack aplikácie, komunikácie medzi Slack API a lokálnou aplikáciou. Nakoniec druhej kapitoly sú spísané funkčné požiadavky, návrh a implementácia Slack aplikácie.

Tretia kapitola popisuje použitý softvér a požiadavky webových aplikácií, stručný popis stránok, use-case diagram používania webových stránok a riešenia bezpečnosti webových aplikácií.

Štvrtá kapitola vysvetľuje základne pojmy v oblasti databázových systémov. Ako fungujú transakcie a ich požiadavky. Následné popisuje postupy a normalizácie pri návrhu databázových systémov. Pri mojom návrhu nakoniec popisujem jednotlivé tabuľky.

Posledná kapitola obsahuje bezpečnostné opatrenia na tvorbu webových aplikácií. Analyzuje niektoré vybrané útoky, ako fungujú a aké sú bezpečnostné rizika pri jed-

notlivých útokoch. Popisuje stručne aj riešenia proti útokom. Nasleduje stručný popis zabezpečenia hesiel pri úniku databázy.

Kapitola 1

Slack

V tejto kapitole si povieme niečo o platforme Slack a prečo je výhodne používať túto platformu v dnešnej dobe. Ukážeme hlavné funkcionality aj rozšírené funkcionality, ktoré poskytuje Slack.

1.1 Úvod do Slacku

Systémy okamžitých správ sa objavili v osemdesiatych rokoch a odvtedy ich ľudia začali používať v rôznych oblastiach. Napríklad v dnešnej dobe elektronická komunikácia s priateľmi je takmer len pod takými systémami. Systémy okamžitých správ sú výhodné aj pre skupinové komunikácie. Predstavme si takú situáciu, že potrebujeme komunikovať s partnerskou spoločnosťou v iných krajinách, v niektorých prípadoch nestačia len E-mailové komunikácie. Síce E-maily sú výhodne pre krátke a zriedkavé komunikácie, v prípade, že účastníci tejto komunikácie sú viacerí a diskutujú o rôznych témach, potom E-mail už nie je efektívne riešenie.

E-mail je nevyhnutný, keď účastníci vytvoria veľké množstvo správ na pôvodnom E-maily v skupinových komunikáciách, ktoré sa musia pri každom čítaní otvoriť aj s prečítanými kontextmi a čitatelia musia nájsť správne odpovede k daným otázkam.
V prípade, že sa vyskytne viac diskusií na raz, možno je ťažké okamžite určiť o ktorej téme sa rozpráva.

E-mail je celkom neefektívny na hľadanie nejakých konverzácii. Pretože E-mailoví klienti sa môžu rozhodnúť na vymazanie alebo archivovanie nejakej časti správ, alebo obsah v tomto E-maile je príliš široký a nemohol efektívne vyhľadávať. Riešenie týchto problémov mohol byť systémy okamžitých sprav, ktoré obmedzujú každú komunikáciu na jedno miesto, oddeľujú konverzácie na rôznych témach.

Čo je Slack?

Prvé spustenie Slacku bolo v auguste 2013. Slack je komunikačná platforma na základe systému okamžitých sprav. V nedávnom prieskume správcov Slacku sa ukázalo, že 32% respondentov uviedlo, že Slack zvýšil produktivitu a znížil interný E-mail o 48.6% a stretnutia o 25,1%. 62.4% respondentov si myslí, že Slack im uľahčil vyhľadávanie informácií. [8] Slack poskytuje služby rôznych cenách, v tejto práci používame len bezplatnú verziu.

1.1.1 Funkcionality

Hlavné funkcionality

Platforma Slack poskytuje mnoho užitočných funkcionalít.
Prístupnosť. Slack má aplikáciu na všetkých operačných systémoch (Windows, MacOS, Linux) aj na mobilných OS (IOS, Android) má aj webovú aplikáciu. Čo nám umožňuje prístup z hocijakého zariadenia na jeho platformu.
Zjednodušené komunikácie. Komunikácia sa podobá na klasické chatové rozhranie, teda nepotrebujeme sa naučiť žiadne novú zručnosť počas používania. Obsahy komunikácií môžu byt hocičo (texty, obrázky, videá, audia, atď.)
Organizácie obsahu. Slack rozdeľuje kanály na skupinové a súkromné. Skupinový kanál je dostupný pre všetkých účastníkov tejto skupiny. Súkromný kanál je určený len pre komunikácie medzi dvoma účastníkmi. Účastník má právo poslať správu, rôzne typy súborov a prípadne prečítať históriu chatu a stiahnuť súbory na danom kanály.
Zálohovanie správ. Všetky správy si zálohuje Slack pre neskoršie čítanie a zaručuje, aby sa nestratili dôležité správy.
Editovanie správ. Upravovanie odoslaných správ ak sa vyskytli nejaké preklepy alebo iné písomné chyby.
Odvolanie správ. Odvolanie odoslaných správ používame v prípade, že posielame nejakú správu omylom, ale len v nejakom časovom limite.
Odstránenie sprav. Odstránenie odoslaných správ používame vtedy, keď nie je možne odvolať omylom odoslanú správu kvôli časovému limitu.
Formátovanie textu. Okrem bežných formátovaní, Slack poskytuje formát pre zdrojový kód, čo zlepšuje skúsenosti s čítaním kódu.

Rozšírene funkcionality

Okrem hore uvedených funkcií, Slack poskytuje aj dôležité funkcionality a to Slack aplikácie. Slack aplikácie umožňujú pridanie rôznych rozlíšení okolo Slacku. Inštalácie Slack aplikácií sú jednoduché, stačí ich nájsť na stránke Apps a jedinom klikom na tlačidlo Add. Existuje dostatočne veľa Slack aplikácií rôznych typov od rôznych firiem,

napríklad Google Drive, Google Calendar, GitHub, Twitter, Slack for Outlook, Dropbox, atď. Všetky Slack aplikácie sú dostupné pre všetky druhy zariadení, nezáleží od spôsobu prihlásenia ani od operačných systémov. V prípade, že nám nestačí existujúce Slack aplikácie, je možnosť si vytvoriť vlastnú. Slack poskytuje aj rozhranie pre programovanie aplikácií (API), ktoré umožňuje komunikáciu medzi svojou aplikáciou a informáciou zo Slacku.

1.1.2 Slack aplikácie a bezpečnosť

Funkcionality Slack aplikácie

- Prichádzajúce WebHooky (mcoming WebHooks) – Je jednoduchý spôsob posielania správ z externého zdroja do Slacku. S použitím HTTP požiadaviek vo formáte JSON, ktoré obsahujú okrem samotnej správy aj iné nepovinné detaily. Dokáže poslať aj súbory ako prílohy v správach.
- Príkazy lomky (Slash Commands) – Príkazy umožňujú používateľom komunikovať so svojou aplikáciou zvnútra Slacku.
- Roboty (Bots) – umožňujú používateľom vymeniť správy so svojou aplikáciou.
- Odbery udalosti (Event Subscriptions) – Slack aplikácia môže odberať nejaké udalosti z platformy Slack, dostáva notifikáciu pri zmene udalosti (napr. keď používateľ posielal správu na skupinový kanál, alebo nahral nejaký súbor).
- Oprávnenie (Permissions) – konfigurovanie povolenia, aby Slack aplikácia mohla pracovať so Slackom API.

Bezpečnosť Slack aplikácia

Slack automaticky generuje QAuth prístupový kód (QAuth Access Token) na overenie svojich Slack aplikácií a je potrebné ho používať počas každej komunikácie. V prípade, že Slack aplikacia pošle správu bez QAuth kódu, Slack automaticky túto správu zahodí a nedoručí prijímateľovi.

1.2 Robot

Robot (Bot) je počítačový softvér, ktorý pracuje v komunikačnom prostredí, simuluje konverzáciu medzi ľuďmi na základe prednastavených pravidiel. Jednoduchý robot reaguje na kľúčové slova. Ak počas spustenia programu dostáva nejakú vetu obsahujúcu dane slová, potom len odpovie preddefinovanú odpoveď. Existuje však aj zložitejší robot, ktorý pracuje na základne umelej inteligencie, spracovania ľudského jazyka, atď.

Robot môže komunikovať s veľkým počtom používateľov naraz a je schopný odpovedať kedykoľvek počas spustenia. Cieľom tejto práce je vytvoriť jednoduchého robota na automatické posielanie správ a zber informácií z platformy Slack do lokálnej databázy.

Interakcie zo Slack robota

Slack robot je rovnaký ako bežná Slack aplikácia, môže pristupovať k rovnakému rozsahu Slack API. To znamená, že je rovnako silný ako bežná Slack aplikácia. Výhodou použitia Slack robota je znížiť náročnosť pracovanie používateľa so Slack aplikáciou. Slack robot môže poslať každému používateľovi priamo správy, avšak aj na skupinové kanály. Používatelia môžu spomenúť robota v správe a pozývať ho do nejakého kanálu, atď.
Keď sa rozhodneme používať Slack robot, tak je potrebne si najprv vytvoriť používateľa robota (Bot User). Pridávame ho cez stránku nastavenia svojej aplikácie a zapneme funkciu „Bot Users“ v navigačnej ponuke, je potrebné mu dať meno na identifikovanie robota. Najprv treba nastaviť Udalosti API (Events API), a potom si ho už môžeme nainštalovať do Slacku, aby sme zistili, aké správy a udalosti dostane náš robot.
Vybavovania udalosti (handle event) už sú samotné lokálne implementácie Slack aplikácie. Keď prídu udalosti, aplikácia môže odpovedať na danú otázku, poslať súbory alebo zapísať niečo do databázy, atď. Všetko, čo Slack aplikácia posiela je pod menom používateľa robota.

1.2.1 Knižnice pre Slack API

Existujú rôzné knižnice pre rôzne programovacie jazyky.

- C++ - engine, matterbot, slacking, SlackRtm, ...
- C# Cake.Slack, Log4Slack, SlackConnector, ...
- Java – Jbot, SlackMC, jslack, AcraSlack, ...
- JavaScript – alex-slack, Botkit, nodebot-slack, ...
- PHP – alfred-slack, GifBot , Laravel, botman, ...
- atď.[1]

1.2.2 Definícia problematiky

Hlavná implementačná úloha je vytvoriť Slack aplikáciu, s ktorou sa dokážeme opýtať všetkých študentov na aktuálny stav ich projektu, a zároveň uložiť zozbierané odpovede

študentov do databázy. Ďalej na zobrazenie a manažovanie projektov treba vytvoriť webové rozhranie.

Skôr než začneme navrhovať Slack aplikáciu pre danú problematiku, musíme sa rozhodnúť v akom programovacom jazyku budem pokračovať. Keďže cieľom našej implementačnej práce je vytvoriť webovú aplikáciu, tak na implementovanie tejto práce budeme používať knižnicu BotMan v programovacom jazyku PHP.

Kapitola 2

BotMan

V tejto kapitole si povieme základné informácie o použitých technológiách, o knižnici BotMan a o návrhu Slack aplikácie spojením s BotManom.

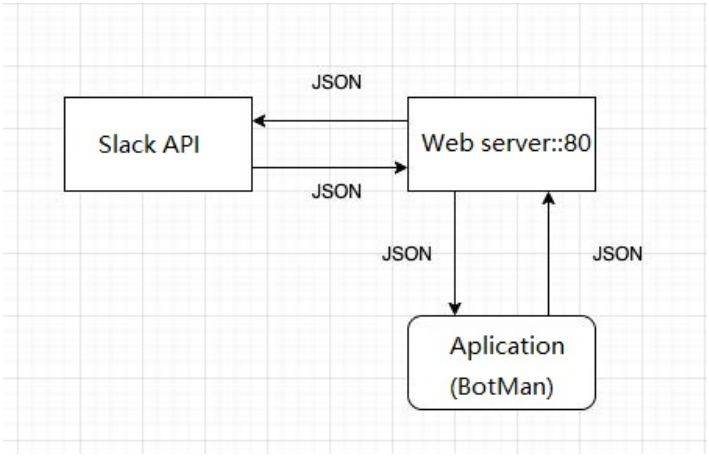
2.1 Použité technológie

PHP

je skriptovací programovací jazyk, ktorý je určený pre webovú stránku a beží na webovom server. Syntax jazyka PHP bola inšpirovaná niekoľkými jazykmi Perl, C, Pascal a Java. Preto je jednoduchý na pochopenie. PHP je dynamický skriptovací jazyk. To znamená, že PHP dokáže dynamicky generovať obsah webovej stránky. Je možné ho spustiť aj cez príkazový riadok alebo integrované vývojové prostredie (IDE). Jazyk PHP môže byť nasadený na rôznych webových serveroch a operačných systémoch. Dokáže sa pripojiť skoro so všetkými relačnými databázami. Mnoho známych informačných systémov bolo napísaných v PHP, napríklad WordPress, Moodle, MediaWiki, Facebook, atď. [4]

BotMan

BotMan je knižnica programovacieho jazyka PHP, ktorá zjednodušuje prácu pri vyvíjaní chatovacích robotov. Autorom tejto knižnice je Marcel Pociot. Knižnica BotMan nie je určená len pre platformu Slack, podporuje aj iné platformy, napríklad Microsoft Bot Framework (MS Teams, Skype), Facebook messenger, HipChat, Telegram, Web, WeChat, atď. To znamená, že naša aplikácia bude fungovať aj pre iné platformy s malou úpravou zdrojového kódu.



Obr. 2.1: Komunikácia s Slackom API a frameworkom BotMan

Inštalácia knižnice BotMan

BotMan používa Composer na správu jeho závislosť (dependency), teda je potrebné inštalovať najprv Composer. Po inštalácii Composer spustíme nasledovný príkaz na inštaláciu BotMan: *Composer require botman/botman.* [6]

2.1.1 Komunikácia medzi Slack API a aplikáciou

Komunikácia je celkom jednoduchá, v Slacku prednastavíme adresu webového servera v časti odberu udalostí, kam má Slack posielať notifikácie o udalostiach. Keď nastane jedna z udalosti, pošle danú udalosť vo formante JSON obr. 2.1 na túto adresu. Teda do nastaveného webového servera, ktorý počúva na porte 80, ak server dostáva správy, posúva ďalej správu do BotManu (aplikácie). BotMan (aplikácia) spracuje udalosť a posiela nejaké odpovede naspäť JSON do Slacku, pripadne neposiela nič podľa nastavenia aplikácie.

2.1.2 Jednoduchý príklad

Prvý pokus je vytvoriť jednoduchý príklad na komunikáciu medzi nimi. Implementácia na strane aplikácie takú jednoduchú funkciu, ktorá čaká správu zo Slack. V prípade, že dostane správu s obsahom „hi“, len odpovie naspäť „Hey, whats up?“, ak prídu iné správy dané vstupy ignoruje, pozri obr. 2.2 2.3. JSON súbor od Slacku obsahuje dosť informácií, avšak v tomto pokuse stačí všimnúť na type, user, text a channel_type. Obr. 2.4

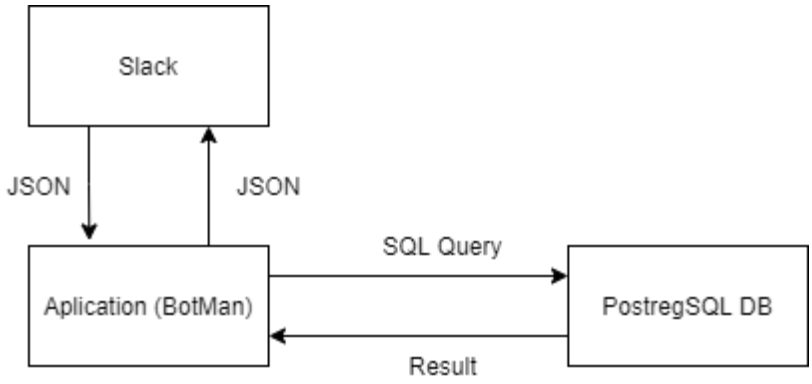
```
$botman->hears('hi', function(BotMan $bot){
   $bot->reply('Hey, whats up?');
});
```

Obr. 2.2: Implementácia jednoduchého príkladu

Obr. 2.3: Ukážka zobrazenia správ v Slack


```
{
    "token": "YOUR_TOKEN_HERE",
    "team_id":"TPSGXX6X2",
    "api_app_id":"APWXEVX7",
    "event":{
        "client_msg_id": "37fe19d9-8eaa-418d-90x8-fa3ce001c63x",
        "type": "message",
        "text": "hi",
        "user": "UPX2012XA",
        "ts": "1573676189.015010",
        "team": "TPSXXX6H2",
        "channel": "DPYMLX898",
        "event_ts": "153676259.0150010",
        "channel_type": "im"
    },
    "type": "event_callback",
    "event_id": "EvQh112XUA0",
    "event_time": 1573654189,
    "authed_users":[
        "UQ9RBLGB7"
    ]

}
```


Obr. 2.4: Ukážková správa od Slack vo formáte JSON



Obr. 2.5: Diagram pripojenia Slack, DB a Aplikácie (BotMan)

2.2 Funkcionálne požiadavky

Základné otázky. Aplikácia by sa mala opýtať jednotlivých študentov 3 základné otázky: čo robil, čo bude robiť a aké má problémy.
Periodické reporty. Aplikácia dostáva a usporiada odpovede od študentov, následne vloží ich do databázového systému.
Získanie id kanálu. Aplikácia poskytuje získanie svojho identifikátora kanála pomocou príkazov lomky (Slash Command) pre každého študenta individuálne, aby aplikácia vedela s kým sa rozpráva, tento identifikátor treba vedieť už pri registrácii.
Upravovanie odpovede. Aplikácia povolí študentom upraviť svoje odpovede na otázky pomocou príkazov lomky (Slash Command), ale len za posledný týždeň.
Pripojenie do PostgreSQL databázy. Aplikácia pracuje s databázou, overuje platnosť HTTP požiadavky a uloží eeriodické reporty do databázy. Obr. 2.5
Automatické posielanie správ. Aplikácia dokáže spustiť posielanie správ pravidelne automatické pomocou linuxového plánovača CRON.
Registrácia nových študentov a zmena stavu projektu. Aplikácia nemá priamo prístup do týchto nastaveniach. Treba požiadať zmenu vedúceho, ktorý má prístup do webovej aplikácie.

2.3 Inštalácia Slack aplikácie

Slack aplikácia a webové rozhranie

V prípade, že nasadíme Slack aplikácie na localhoste, treba stiahnuť a používať program Ngrok alebo iný alternatívny program, ktorý vystaví lokálny webový server na internete (Ngrok podporuje aj HTTPS protokol) a Ngrok generuje verejnú webovú adresu. Nainštalujeme potrebné závislosti (dependecy) pre Slack aplikáciu použitím Composer: *php composer.phar install*. Ak nasadíme Slack aplikáciu a webové rozhranie

na online prostredie, netreba mať program Ngrok.

Slack

Treba nastaviť verejnú webovú adresu na Slacku (event subscriptions a slash commands) a prekopírujeme OAuth Access token od Slacku naspať do aplikácie.

PostgreSQL

Slack Aplikácia vyžaduje spojenie do PostgreSQL databázových systémov, je potrebné doplniť do súboru config (súbor je uložený v elektronickej prílohe) pri spustení aplikácie. Potom spustíme súbor db.sql (v prílohe) na vytváranie potrebných tabuliek.

CRON

Pre posielanie správ pravidelne automatické treba pridávať sender.php (v prílohe) do linuxového plánovača CRON a prideľujeme ho nejaké časové periódy.

Kapitola 3

Webové rozhranie

V tejto kapitole si ukážeme niektoré základne pojmy v tvorbe webových aplikácii a použite softvéry aj narvh pri implementácii.

3.1 Použité softvéry

Smarty je PHP šablónový nástroj (template engine), ktorý slúži na zlepšenie čitateľnosť kódu PHP a HTML. Pretože samotný PHP kódy môže obsahovať nielen funkčne logické časti, ale aj HTML kód. Ak ide o rozsiahlejšie PHP kódy, niekedy sú ťažko čitateľné a čitatelia sa strácajú. Preto riešenie s použitim šablónovacieho nástroja zlepšuje čitateľnosť a šetrí čas pre budúci vývoj. [2]

PHP je skriptovací programovací jazyk, ktorý slúži hlavne na programovanie dynamických webových stránok. PHP generuje dynamický obsah vo formáte HTML na strane servera. Následne webový server pošle vygenerovane HTML dokument klientovi.

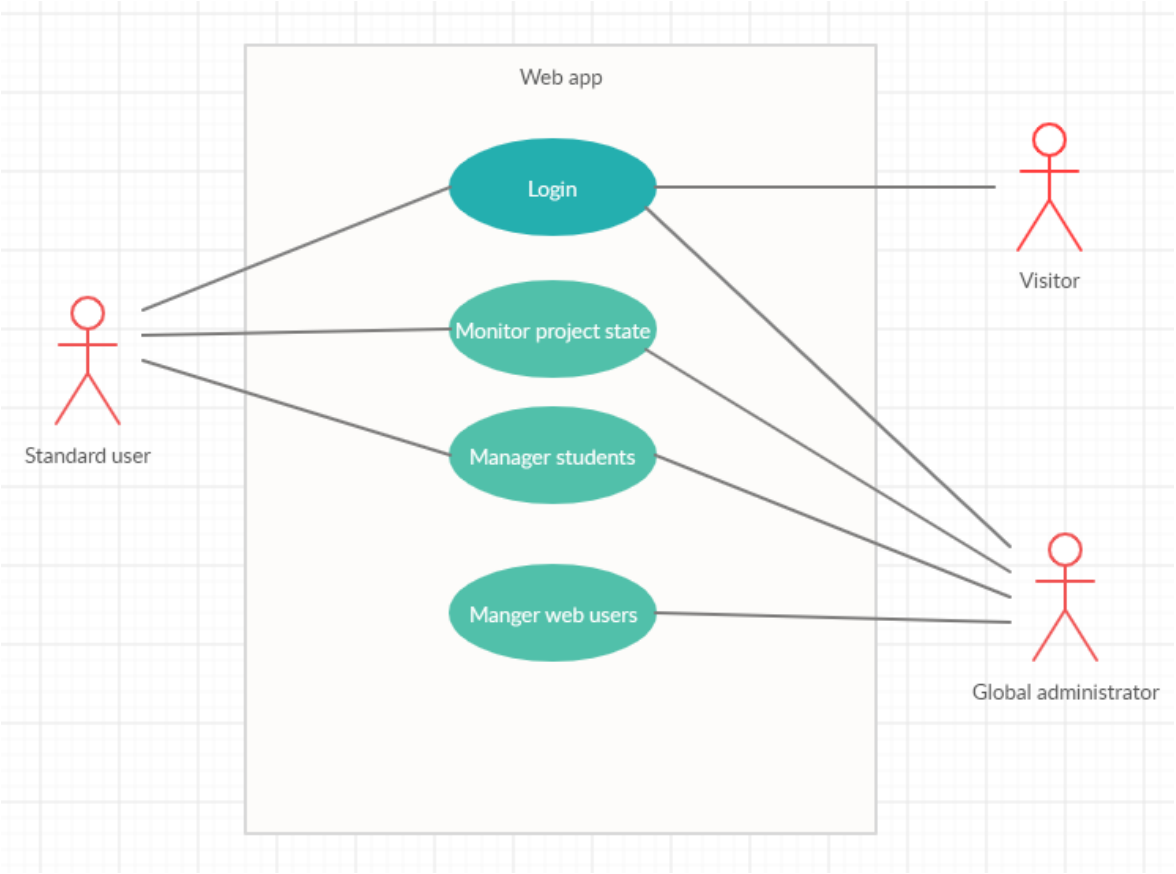
PostgreSQL je relačný databázový systém, ktorý poskytuje voľne šíriteľné svoje služby a využíva SQL jazyk na manipuláciu s databázou.

3.2 Návrh webovej aplikácie

Cieľom tohto návrhu je vytvoriť webovú aplikáciu, v ktorej sa zobrazuje aktuálny stav študentov a tiež manažujú projekty jednotlivých študentov, ako aj prebieha registrácia nových webových používateľov.

3.2.1 Používatelia webovej aplikácie

Štandardný používateľ – má právo pozerať týždňové reporty svojich študentov manažovať ich a ich projekty. Prípadne môže zastaviť automatické posielanie otázok BotMan študentovi v platforme Slack.



Obr. 3.1: Use case diagram

Globálny administrátor – špeciálny typ používateľ a ktorý okrem štandardných funkcií dokáže resetovať heslo jednotlivých používateľov. Prípadne môže odstrániť aj nejakého zaregistrovaného používateľa a pridať nového používateľa. Obr. 3.1.

3.2.2 Webové stránky

Administratívne rozhranie

Okrem štandardného používateľského rozhrania, globálny administrátor dokáže odstrániť a pridať webových používateľov. Aplikácia nepovolí registráciu do systému, jediný spôsob na registráciu je sa kontaktovať s globálnym administrátorom. Štandardný webový používateľ nemôže odstrániť zo systému žiadneho používateľa.

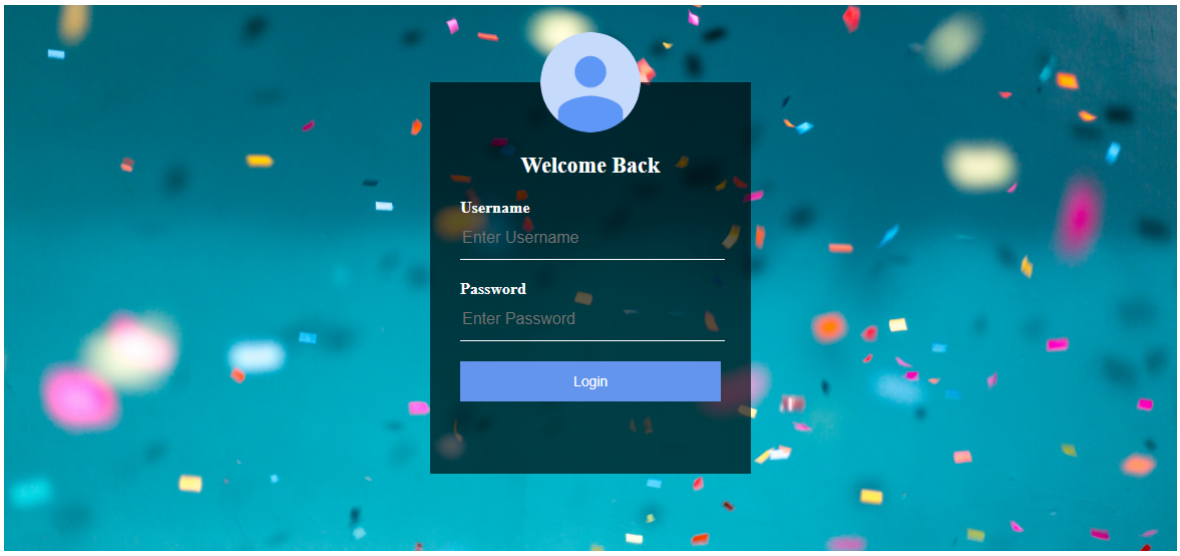
3.2.3 Bezpečnosť

1. Webová aplikácia presne kontroluje vstupy od používateľa použitím JS kódu na strane klienta a PHP funkcie (pg_escape_string()) proti SQL injection, aby nevznikol útok typu injection.

2. Webová aplikácia používa protokol HTTPS na komunikáciu medzi klientom serverom, aby útočník nemohol odchytiť nešifrovanú komunikáciu.

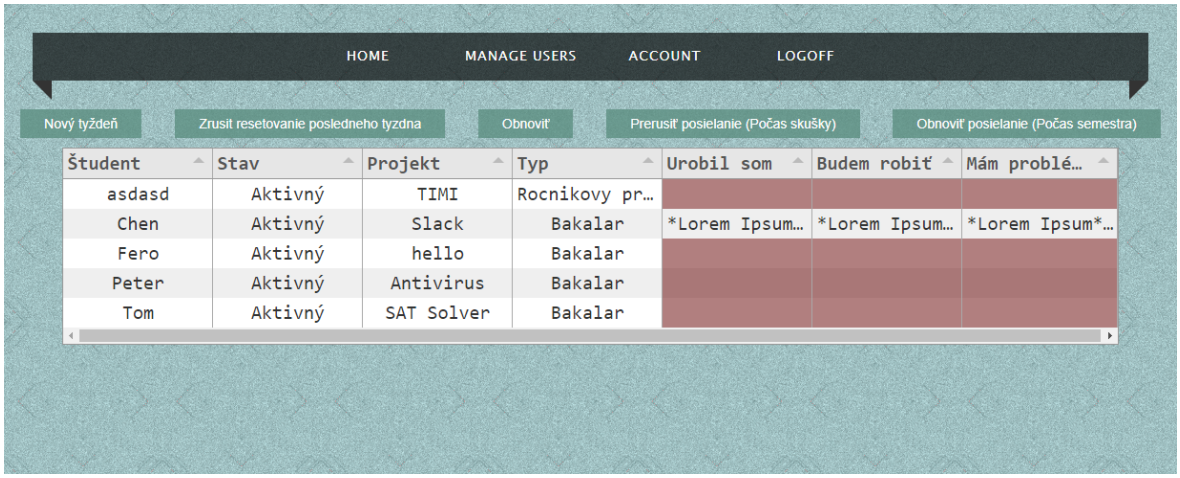
3. Webová aplikácia ukladá heslo do databázy s náhodnou soľou (salt) s využitím hešovacej funkcie SHA512, aby útočník nemohol v prípade úniku databázy získavať ani odhadnúť heslo.

3.2.4 Popis stránky



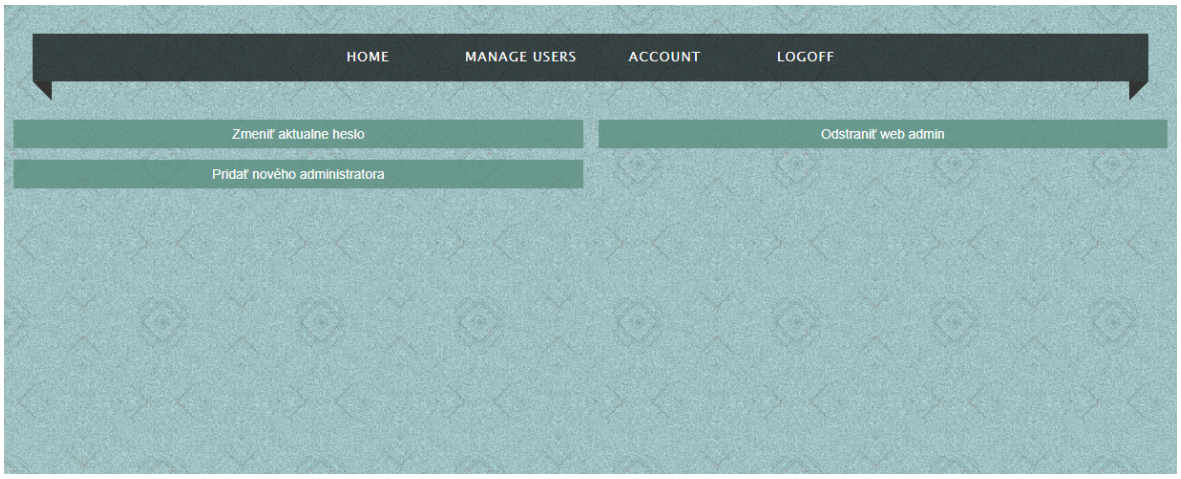
Obr. 3.2: Screenshot /index.php

/index.php – Hlavná stránka, na ktorej sa kontroluje stav prihlasovania, ak používateľ nie je prihlásený tak ho presmeruje na prihlasovací formulár, v opačnom prípade presmeruje na home.php.



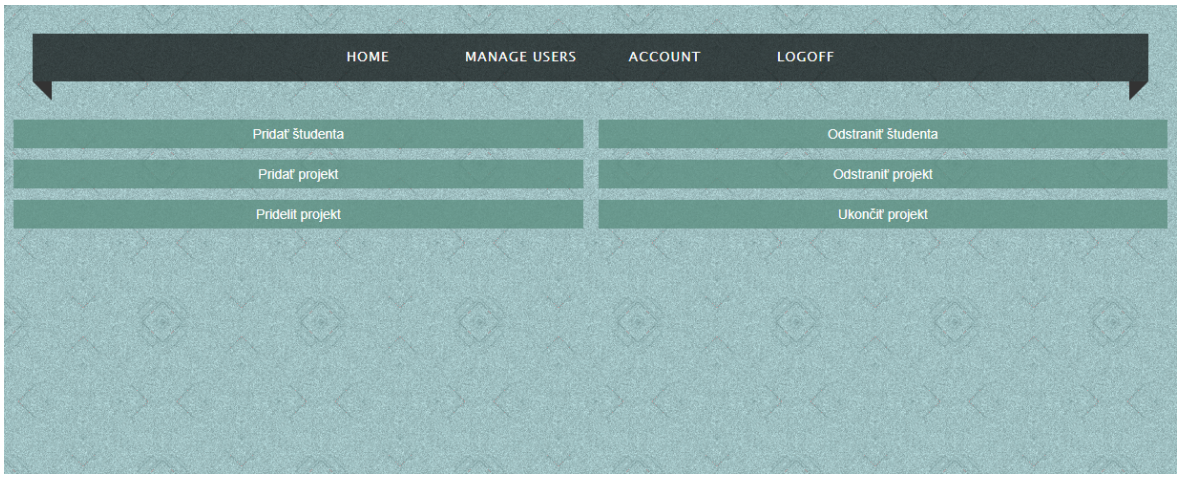
Obr. 3.3: Screenshot /home.php

/home.php – Obsahuje tabuľky o stavoch projektu študentov a štatistiky jednotlivých študentov. Zastavuje a obnovuje posielanie správ študentom a manuálne posielanie správ.



Obr. 3.4: Screenshot /ucet.php

/ucet.php – Manažovanie webových používateľov. Túto stránku má povolenie navštíviť len globálny administrátor.



Obr. 3.5: Screenshot /users.php

/users.php – Manažovanie študentov a projektov.

/config – Obsahuje konfiguráciu pripojenia do databázových systémov, vypíše chybové hlášky, keď sa nepodarí pripojiť. Nemá grafické rozhranie.

Kapitola 4

Databáza

V tejto kapitole definujeme základné pojmy z databázových systémov, ukážeme, čo môžu databázy urobiť pre nás a tiež nejaké princípy na navrhovanie efektívnych databáz.

4.1 Základne pojmy

Databáza je množina štruktúrovaných dát uložených v počítačovom systéme.
Databázový systém je počítačový systém, ktorý poskytuje efektívne manipulačné prostredie s veľkým objemom štruktúrovaných dát. Musí splniť nasledovné podmienky

1. Uložené dát musí byť bezpečné aj v nečakaných situáciách
2. Umožňuje viacerým užívateľom paralelne manipulovať dáta a spracovať dátové schémy
3. Podporuje manipuláciu aj s veľkými a zložitými dátami

Relačné databázy sú založené na relačnom dátovom modeli, ktoré používajú tabuľkové štruktúry na reprezentovanie dát a dátových vzťahov. Existujú aj iné dátové modely ako napríklad stromové a grafové, ale pri vyhľadávaní zložitejších dát potrebujú programovací jazyk vyššej úrovne. Najpopulárnejšie relačné databázy sú PostgreSQL server, MySQL server, MS SQL server, atď.
SQL (štruktúrovaný vyhľadávací jazyk) je jazyk, ktorý poskytuje množiny tabuľkových operácii ako napríklad na manipuláciu (INSERT, DELETE, SELECT, UPDATE, ...) a definíciu údajov (CREATE, ALTER, DROP, ...) v relačných databázach.

4.2 Transakcia

Transakcia je postupnosť operácií, ktorá sa vykoná atomicky poradie a zároveň neovplyvňuje iné transakcie. Pritom každá postupnosť musí začať s operáciou START

a operácia START môže sa vyskytnúť práve raz v postupnosti (systém abortuje celú transakciu v prípade sa vyskytne START viac ako raz). Každá postupnosť musí končiť operáciou COMMIT alebo ABORT, daná operácia sa vyskytne pravé raz v postupnosti.

4.2.1 Operácie transakcie

- START: začína novú transakciu
- READ: číta záznam z databázy
- WRITE: zapisuje záznam do databázy
- INSERT: vkladá záznam do databázy
- DELETE: odstráni záznam z databázy
- COMMIT: označí úspešne ukončenie transakciu
- ABORT: označí neúspešne ukončenie transakciu

4.2.2 Požiadavky na transakčný databázový systém: ACID

- Atomicky (Atomicity): transakcia sa buď úspešne vykoná s operáciou COMMIT alebo nevykoná vôbec.
- Konzistencia (Consistency): databázový systém musí ostať v jednoznačnom stave aj po vykonaní hocijakej transakcie.
- Izolovanosť (Isolation): databázový systém poskytuje možnosť paralelne vykonávať viacero transakcií súčasne, výsledok transakcie nemôže byť ovplyvnený od iných transakcií.
- Trvanlivosť (Durability): ak daná transakcia skončí úspešne s operáciou COMMIT, musí byť uchovaná v databáze.

Okrem podmienky konzistencií sa má o to starať programátor, databázový systém sa stará o ostatných podmienok automatické.
Rozvrh je postupnosť, ktorá vznikne premiešaním operácií niekoľkých transakcií.

Plánovač (Scheduler)

Databázový systém používa plánovač na transformáciu vstupného rozvrhu do výstupného. Ideálny výstupný rozvrh je sériový, teda nepremiešame operácie niekoľkých transakcií ale spustí transakcie po poradí.

Uviaznutie (Deadlock)

Databázový systém poskytuje možnosti paralelne vykonávať viac transakcií naraz. Preto je pravdepodobne, že vznikne problém uviaznutia. Teda rôzne transakcie požiadajú rovnaké dáta alebo čakajú navzájom. Systém rieši uviaznutie algoritmicke, abortuje niektoré transakcie z uviaznutia, ak nájde uviaznutie.

Obnova

Okrem spracovania a uchovávania dát, databáza musí zaručiť aj odolnosť voči výpadkom ako napríklad výpadok elektriny.

4.3 Normalizácia databázy

Normalizácia databázy je proces štruktúrovania relačných databáz, ktorý transformuje návrh databázy do normálnej formy. Cieľom je odstrániť redundancie, nekonzistencie a duplikácie dát. [7]

Atribút – reprezentuje stĺpec relácie a má svoj unikátny názov v rámci relácie.
Relácia – je podmnožina kartézskeho súčinu množín atribútov relácie, ktorá je reprezentovaná ako dvojrozmerná tabuľka.
Zložený atribút – atribút, ktorý sa obsahuje viac hodnôt v sebe.
Duplikát – záznam, ktorý nie je jediný v danej relácii.

Primárny kľúč – Primárny kľúč je jedinečný atribút, ktorý jednoznačne identifikuje každý záznam. V prípade, že existuje viac kľúčov, databázový systém vyberie jeden kľúč z nich ako primárny.
Cudzí kľúč – Cudzí kľúč je integritné obmedzenie relačných databázach, ktoré spojí atribúty viacerých tabuliek. Atribúty, ktoré spoje pomocou cudzieho kľúča, musia mať primárny kľúč v relácii.

Funkčná závislosť – Množina atribútov Y funkčne závisí od množiny atribútov X relácii r, ak pre každú platnú inštanciu X, daná hodnota z inštancie X jednoznačne určuje hodnotu inštancie Y.

Nadkľúč relácie je podmnožina atribútov relácie, z ktorej sa dajú odvodiť všetky atribúty relácie.
Kľúč relácie je taký nadkľúč, z ktorého keď sa odstráni ľubovoľný atribút, prestane byť nadkľúčom danej relácie. Inak povedane kľúč relácie je minimálny nadkľúč.

4.3.1 Normálne formy

Prvá normálna forma (1NF)

Relačná schéma je v prvej normálnej forme, ak žiaden atribút nie je zložený a databáza neobsahuje duplikáty.

Druhá normálna forma (2NF)

Relačná schéma je v druhej normálnej forme vtedy, ak je v prvej normálnej forme, všetky atribút patri do nejakého kľúča alebo nepatri žiaden a zároveň je funkčne závislý od primárneho kľúča.

Tretia normálna forma (3NF)

Relačná schéma je v tretej normálnej forme vtedy, ak je v druhej normálnej forme. Pre každú platnú funkčnú závislosť $X \rightarrow Y$ platí, že buď X je nadkľúč v relácii alebo Y je súčasť nejakého kľúča v relácii.

Boyce-Codd normálna forma (BCNF)

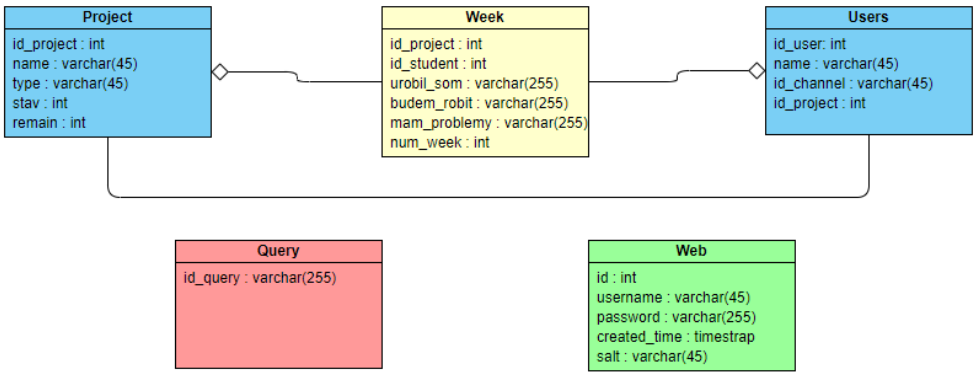
Relačná schéma je v Boyce-Codd normálnej forme, ak pre každú platnú netriviálnu funkčnú závislosť $X \rightarrow Y$ platí, že X je nadkľúč.

4.4 Návrh databázy

Návrh databázy je proces popisovanie štruktúr požiadaviek aplikácie do databázových modeli. Je veľmi dôležité udržovať náš návrh v normálnej forme. To nám odstráni veľa problémov pri navrhovaní databázy. Návrh databázy sa rozdeľuje do troch fáz, koncepčná, logická a fyzická.

4.4.1 Koncepčný návrh

Koncepčný návrh je vizualizácia dátového modelu pomocou vytvorenia UML diagramu alebo ER diagramu, ktorý popisuje aké veci vstupujú do diagramu a ich súvislosti, pozri obr. 4.1.



Obr. 4.1: UML diagram

4.4.2 Používatelia webovej aplikácie

4.4.3 Logický návrh

Logický návrh je proces, ktorý prevedie koncepčný návrh do dátového relačného modelu. Tento návrh nezávisí od hardwaru či softwaru, cieľom je generovať relácie, typy atribútov, atď. V tomto procese treba rozmýšľať nad normalizáciou.

Popis jednotlivých tabuliek

Users. Tabuľka Users udržiava základne informácie o študentoch ako napríklad meno študenta, identifikačne číslo študenta a cudzí kľúč identifikačné číslo projektu sa spája s tabuľkou Project. Predpokladáme, že každý študent má práve jeden aktívny projekt. Identifikačné číslo kanálu je špeciálna požiadavka od frameworku BotMan, Bez toho BotMan nevie, kam má odosielať správy. Študent môže získať id_channel v platforme Slack pomocou prikazu lomky (Slash Command).

Project. Tabuľka Project udržiava informácie o projektoch, s ktorými sa jednoznačne určí daný projektu pomocou identifikačného čísla projektu. Obsahuje aj meno projektu, typ projektu. Atribút stav, označí či daný projekt je aktívny. Ak nie je, potom BotMan nebude študentovi posielať správy. Atribút remain, označí počet zostávajucich týždeň na dokončenie projektov, je možne ho modifikovať vo webovom rozhranií.

Period. Tabuľka Week udržiava týždňový stav projektov a študentov. Cudzie kľúče id_project a id_student sú identifikátory tejto tabuľky. Atribút numweek označí, ktorý je týždeň. V atribúte urobil_som, budem_robit, mam_problemy sú uložene odpovede dané otázky.

Query. Tabuľka Query udržiava spracované požiadavky od platformy Slack. Slack alebo iné webové aplikácie s technológiou HTTP, zvyčajne posiela raz požiadavky do prednastaveného servera či klienta. V prípade, že webové aplikácie nedostávajú alebo dostavajú neskoršie spätné väzby z druhého konca. Tak predpokladajú, že dané požiadavky sa stratili počas prenosu a znovu posielajú rovnaké požiadavky. Aby aplikácia nepracovala znova spracovane požiadavky, je potrebne určiť a zapamätať, ktoré požiadavky sú vykonané. V mojom prípade si zapíšem identifikačné číslo požiadavky do databázy po úspešnom vykonaní a pred vykonaním požiadavky ich skontrolujem čí existuje dane identifikačne číslo od databázy.

Web. Tabuľka Web udržiava prihlasovacie údaje do webového rozhrania. V atribúte salt sú uložené náhodné znaky, ktorým sa zvyšuje bezpečnosť hesla. Je to riešenie proti slabým heslám.

4.4.4 Fyzický návrh

Fyzický návrh je mapovanie logického návrhu na konkrétny databázový systém pomocou jazykov SQL DDL (Data Definition Language). Cieľom je vytvoriť tabuľky, primárne, cudzie a sekundárne kľúče, indexy, atď. SQL DDL dotazy sú v prílohe aplikácie. Ak prejdú všetky tri fázy, tak sme úspešne navrhli databázu s požiadavkami svojej aplikácie, a ak navyše spĺňa normalizáciu databázy, potom je zaručená efektívnosť tohto návrhu.

Kapitola 5

Bezpečnosť

V tejto kapitole si ukážeme niektoré bezpečné problémy pri implementácii webových aplikácií. Bez ohľadu na to, aký softvér ideme zrealizovať, treba sa dopredu pozrieť na bezpečne riziká. Väčšina dynamických webových aplikácií je navrhovaných v architektúre klient – server, preto musíme chrániť stranu klienta, stranu servera aj prenos dát medzi nimi. [5]

5.1 Typické útoky

5.1.1 Útoky Injection

JavaScript injection

JavaScript injection je druh útoku, ktorý nastáva pri spracovaní neočakávaných vstupných dát. Ak aplikácia dostáva dáta, ktoré obsahujú spustiteľné JavaScript príkazy a následne si ich vloží do databázy. V prípade, ak aplikácia potrebuje zobraziť tento vstupný obsah v prehliadači, v skutočnosti prehliadač nezobrazuje žiadny obsah, ale vykonáva príkaz keďže daný obsah je spustiteľný JS príkaz. Pozri obr. 5.1 5.2

Obr. 5.1: Príklad vloženia JavaScripta kódu do HTML stránky

Takému útoku je možné sa vyhnúť kontrolou vstupných dát. Teda zahodí taký vstup alebo premení spustiteľne príkazy vo vstupných dátach, aby aplikácia naďalej bezchybné spracoval. Používam JavaScript kód na kontrolu vstupu používateľa, aby nemohol napísať spustiteľný JS príkaz.

Obr. 5.2: Výsledky príkladu

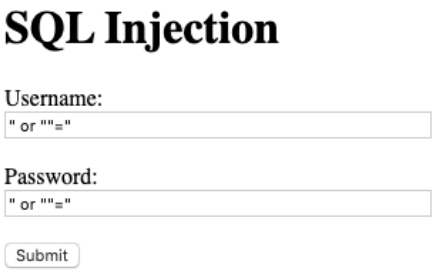
SQL injection

SQL injection je druh útoku, ktorý sa snaží zmeniť SQL dotaz z prednastavených dotazov pomocou vloženia kľúčových znakov do vstupu. Zvyčajne SQL dotazy sú prednastavené na strane servera a inicializujú niekoľko premene, ktoré požiadajú hodnoty od používateľov. Teda útočník môže zadávať aj špeciálne znaky SQL jazyka, ktoré pokazia prednastavené dotazy. [3]

SQL injection je vážny bezpečnostný problém pre (nielen) webovú aplikáciu. Ak útočník útočí takým spôsobom, je možné pozmeniť dáta, získavať neprístupne dáta, dokonca môže ovládnuť celý systém. Avšak útočník musí dopredu uhádnuť alebo získať štruktúru dotazu a štruktúru databázového systému. Pozri obrázky 5.3 5.4 5.5 5.6.

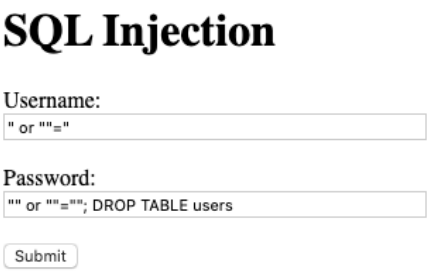
```php
$user = $_POST("username");
$Pass = $_POST("userpassword");

$sql = 'SELECT * FROM users WHERE username ="' + $user + '" AND password ="' + $Pass + '"';

//Expected SQL query
$sql = 'SELECT * FROM users WHERE username ="myUsername" AND password ="myPass"';
```

Obr. 5.3: Príklad SQL injection na stráne servera PHP code



Obr. 5.4: Príklad SQL injection - vždy pravdivé prihlásenie

Riešenie takého útoku môže byť podobné ako riešenie pri JavaScript injection, to kontrolou vstupných dát pred vložením do databázového systému. V prípade, že sa



Obr. 5.5: Príklad SQL injection - vymazanie tabuľky

```
//Result SQL query with SQL injection
//Always true login
$sql = 'SELECT * FROM users WHERE username ="" or ""="" AND password ="" or ""="";

//True login and drop user table
$sql = 'SELECT * FROM users WHERE username ="" or ""="" AND password ="" or ""=""; DROP TABLE users';
```

Obr. 5.6: Príklad SQL injection - vysledné dotazy

vyskytujú špeciálne znaky (napr. úvodzovky, apostrofy, ...) z SQL jazyka, je dobré si ich nahradiť bezpečnými znakmi a vložiť do databázového systém. Existujú aj knižnice, ktoré poskytujú hotové funkcie na ošetrenie SQL injection v niektorých moderných programovacích jazykoch (napr. Java, Python, PHP, ...). V tejto webovej aplikácie používam PHP funkcie (pg_escape_string()) na ošetrenie SQL injection.

HTTP header injection

HTTP header injection je druh útoku, ktorý nastáva vtedy, ak útočník môže zapísať ľubovoľné znaky do HTTP hlavičky. Akékoľvek webové stránky, ktoré sa dajú zobraziť cez internetový prehliadač (bez ohľadu na to, aké technológie a frameworky používajú dane stránky) obsahujú protokol HTTP. V HTTP protokole existujú špeciálne znaky <CRLF> (Carriage Return a Line Feed) medzi hlavičkou a obsahom, ktorý označí koniec hlavičky a začiatok obsahu. obr. 5.7 5.8

```
http://example.com/login?page=http%3A%2F%2Fexample%2Findex

HTTP/1.1 302 Moved Temporarily
Date: Tue, 17 Aug 2010 20:00:29 GMT
Server: Apache /1.3.3.6 (unix)
Location: http://example/index
```

Obr. 5.7: Príklad HTTP Header pre odpoveď

```
http://example.com/login?page=http%3A%2F%2Fexample%2Findex%0D%0A%0D%0A%3Cscript%3Ealert%28%27hello%27%29%3C%2Fscript%3E
HTTP/1.1 302 Moved Temporarily
Date: Tue, 17 Aug 2010 20:00:29 GMT
Server: Apache mod_fcgid/2.3.5 mod_auth_passthrough/2.1 mod_bwlimited/1.4 FrontPage/5.0.2.2635
Location: http://localhost/index<CRLF>
<CRLF>
<script>alert('Hello! JavaScript code')</script>
```

Obr. 5.8: Príklad HTTP CRLF injection

Riešenie HTTP heads injection spočíva vo filtrovaní pri prijatí žiadosti od klienta, aby webový server nespracoval nepovolené znaky, najmä CRLF. Týmto spôsobom sa zabráni znečisteniu hlavičiek HTTP od vstupných údajov.

5.1.2 Útoky typu Hijacking

Sniffing attack

Sniffing attack je druh útoku, ktorý sa snaží dostať k informáciám medzi internetovým prehliadačom a webovým serverom. Útočník používa paketový analyzátor (napr. Wireshark) na odpočúvanie komunikácie medzi klientom a serverom. Ak sa útočník dostane ku komunikácii, následne by mohol získavať dosť citlivé údaje, ako napríklad prihlasovacie meno a heslo, ktoré sa zvyčajné posiela od klienta do servera, avšak server môže posielať naspäť výsledky z nejakého SQL dotazu. Okrem úniku citlivých dát, útočník môže používať aj MITM (Man-in-the-middle) útok. V tomto prípade útočník sa stane aktívnym prostredníkom. Klient posiela dáta útočníkovi namiesto servera, útočník spracuje dáta a preposiela do servera. Následne server odpovedá útočníkovi s dátami, útočník môže opraviť odpoveď alebo rovno preposiela odpoveď klientovi. Teda útočník má prístup k dátam aj od servera.
Riešenie takýchto útokov prechádza z oblasti kryptológie, teda zašifrovaním komunikácie medzi internetovým prehliadačom a webovým serverom. známym šifrovacím webovým komunikačným protokolom je HTTPS (Zabezpečený hypertextový prenosový protokol), ktorý prenesie dáta v šifrovacom kanáli namiesto jednoduchej textovej komunikácie. HTTPS používa protokol SSL (Secure Socket Layer) alebo zlepšený protokol TLS (Transport Layer Security) na šifrovanie dát. V prvej fáze tejto komunikácie prebehne SSL alebo TLS Handshake, vtedy keď klient sa dohodne so serverom na šifrovacom algoritme, server posiela klientovi svoj certifikát a šifrovací kľúč na šifrovanie správ. Potom ich komunikácie prebiehajú len v šifrovacom kanáli. HTTPS protokol zabráni úniku dát pri útoku MITM, pretože útočník sa dostane len k šifrovaciemu dátam a neprečíta ich bez šifrovacieho kľúča a algoritmu. Ale klient musí pravidelne skontrolovať platnosť a vydavateľa certifikátu od servera, lebo útočník môže vytvoriť falošný certifikát. V tejto aplikácie používam protokol HTTPS na šifrovanie komunikácie.

Session hijacking attack

Session hijacking attack je druh útoku, pri ktorom útočník získava session od klienta. Zvyčajne sa session používa na uchovávanie identifikátora prihláseného klienta do webovej stránky. Útočník sa snaží získať od neho session a čím sa prihlási do webovej stránky pod jeho menom. Na získavanie Session je možné používať útok JavaScript Injection, HTTP Header injection, Sniffing attack, atď.

- Session fixation. Útočník dopredu zistí zo servera nejaký platný session a používa klientov identifikátor session pri návšteve stránky. Po prihlásení klienta, útočník dostáva rovnaké právo ako daný klient do systému.
  Riešenie je vygenerovať nový identifikátor session po počiatočnej autentifikácii, to znamená, že sa identifikátor session zmení okamžite po autentifikácii. Aj keď útočník uhádol počiatočný identifikátor session, nebude mať prístup do stránky.

- Cookie hijacking alebo direct access. Ak útočník inštaluje škodlivý softvér na zistenie session cookie v klientskom počítači a pošle ich útočníkovi, tak útočník získava priamo prístup k súboru cookie v dočasnom úložisku prehliadača klienta pomocou škodlivého softvéru alebo útočník môže mať lokálny alebo vzdialený prístup do systému.
  Riešenie je používať antivírusový softvér vo svojom systéme a neposkytovať nikomu vzdialený prístup do svojho počítača.

- Dictionary Attack. Ak systém generuje len predvídateľne a krátke identifikátory session, útočník môže skúšať uhádnuť identifikátor session aktívneho klienta.
  Pre riešenie tohto útoku musí systém generovať nepredvídateľne a dostatočne dlhé identifikátory session.

5.2 Bezpečnosť hesiel

5.2.1 Slabé heslo

I keď slabé heslo sa nepovažuje za útok od útočníka, útočník môže používať dictionary attack na zistenie prihlasovacieho hesla, ak je heslo slabé, útočník získava heslo v rozumnom čase. Niektoré aplikácie nevynútia klientov, aby používali zložitejšie heslá, dôsledku toho klienti môžu používať jednoduché a ľahko uhádnuteľné heslá, ako napríklad 123456, dátum narodenia, telefónne číslo, meno, atď.
Riešenie je, nepovoliť zadanie slabé heslo od klienta v aplikácii. Teda aplikácia skontroluje zadané heslo pri registrácii alebo pri zmene heslá, aby nové heslo obsahovalo kombináciu malých a veľkých písmen, číslic a iných znakov a zároveň dĺžka musí byť dostatočne dlhá. Ideálne je nechať strojovo generovať nejaké náhodne postupnosti znakov

s uvedenými podmienkami. Silné heslo je ťažko zapamätateľné, a preto niektorí klienti majú rovnaké heslo všade, čo je dosť nebezpečné. Existujú také počítačové programy, ktoré automaticky generujú silné heslá a následne ich spravujú (Napríklad 1password). Namiesto nezapamätateľných silných hesiel si stačí zapamätať jedno prístupové heslo do programu.
Pre riešenie toho útoku server po niekoľkých neúspešných pokusoch prihlásenie môže používateľovi zamietnuť prihlásenie do systému na určitý čas, aby útočník nemohol preťažiť server a rýchlo testovať heslo podľa slovníku.

5.2.2 Únik databázy

Aj napriek tomu, že SQL databázový systém je zabezpečený proti úniku dát, útočník môže získať prístup do databázového systému pomocou útokov SQL injection alebo útokom Sniffing. Preto je nebezpečné ukladať heslá do databázy vo formáte otvoreného textu. Je dobre používať hašovaciu funkciu pri registrácii, a vkladať len zahašované heslá do databázy. V prípade, že používame známe hašovacie funkcie so slabým heslom, útočník má vysokú pravdepodobnosť vypočíta vstupný reťazec z zahašovaceho reťazca.
Riešenie je používať zložitejšiu hašovaciu funkciu ako napríklad SHA512 a pridať náhodnú kryptografickú soľ (salt). Pri hašovaní hesla pridávame do pôvodného reťazca aj náhodne generované reťazce znakov, tým sa zvyšuje bezpečnosť hesiel. A zároveň klienti, ktorí majú rovnaké heslo budú mať iný zahašovaný reťazec v databáze.

V tejto aplikácie síce povolí zadávať aj slabé heslo, avšak rieši problém slabého hesla a úniku databázy použitím zložitejšej hašovacej funkcie SHA512 s náhodnou kryptografickou soľou (salt).

Záver

Cieľom tejto bakalárskej práce bolo navrhovať a implementovať Slack rozšírenie na zbieranie týždňových reportov o postupe študentov a webové rozhranie na monitorovanie a manažovanie jednotlivých študentov. Teda implementačnú prácu môžeme rozdeľovať do dvoch časti.

V prvej časti práce by mohol čitateľ získať základnú predstavu o fungovaní platformy Slacku a spôsob komunikácie so svojou implementáciou so Slackom API. Moja implementácia spĺňa všetky funkčné požiadavky. Túto časť považujem za splnenú.

Mojou úlohou v ďalšom časti bolo predstaviť tvorbu webového rozhrania. V tejto časti sa nachádza návrh databázových systémov a možné spôsoby o ochranie webovej aplikácie proti útokom. Čitateľ môže získať základnú predstavu v oblasti databázových systémov a bezpečnosti tvorby webovej aplikácie. Pri implementácii webovej aplikácie je potrebné dopredu rozmýšľať nad bezpečnosťou a efektivitou navrhovanej databázy. Moja implementácia je testovaná s typickými útokmi a je odolne proti nim. Zároveň sú splnená všetky funkčné požiadavky vzhľadom na webové rozhranie.

Literatúra

[1] Slack api library, 2019. [Citované 2019.12.20] Dostupné z `https://api.slack.com/community`.

[2] Smarty documentation, 2019. [Citované 2020.04.23] Dostupné z `https://www.smarty.net/docs/en`.

[3] Chris Anley. Advanced sql injection in sql server applications. 2002.

[4] Leon Atkinson. *Core PHP programming: using PHP to build dynamic Web sites*. Pearson Education, 2000.

[5] Dharmaraj Rajaram Patil and JB Patil. Survey on malicious web pages detection techniques. *International Journal of u-and e-Service, Science and Technology*, 8(5):195–206, 2015.

[6] Marcel Pociot. Botman installation, 2019. [Citované 2019.12.20] Dostupné z `https://botman.io/2.0/installation`.

[7] Abraham Silberschatz, Henry F Korth, Shashank Sudarshan, et al. *Database system concepts*, volume 5. McGraw-Hill New York, 1997.

[8] Slack. Slack survey results, 2019. [Citované 2019.12.20] Dostupné z `https://a.slack-edge.com/7b00/img/survey/slack_survey_results.pdf`.

Príloha A: obsah elektronickej prílohy

V elektronickej prílohe priloženej k práci sa nachádza zdrojový kód programu a konfiguračne súbory pri spustení programu.