VŠB TECHNICKÁ UNIVERZITA OSTRAVA | FAKULTA ELEKTROTECHNIKY A INFORMATIKY | KATEDRA INFORMATIKY

Ticket Systém

(c) 2024, verze 1.5, 6.5.2024

Patrik Mintěl

Katedra informatiky

https://dbedu.cs.vsb.cz

Obsah

1 Specifikace zadání 3
2 Datová analýza 5
3 Funkční analýza 8

1 Specifikace zadání

Opravy:

– Přidáno upřesnění druhé vazby mezi tabulkou Users a Tickets
– Vysvětlení tabulky s historickými daty (Ticket_Categories) – Při vytvoření ticketu a zvolení kategorie, do které ticket spadá hráčem, se do tabulky vloží záznam, s ID kategorie, aktuálním datumem a cizí klíč odkazující na tabulku Users bude Null (Upravena vazba v konceptuálním a relačním modelu). Pokud poté moderátor změní kategorii, tak se do této tabulky vloží záznam s novou kategorií a nyní v cizím klíči odkazujícím na tabulku Users bude id moderátora. Aktuální kategorie ticketu je poté ta, která má datum změny jako nejnovější.
– Odebrán sloupec types z datového modelu, který tam minule neměl už být.

VIZE:

Potřebujeme ticketový systém, kde budou hráči na Minecraft serveru řešit problémy, ptát se na dotazy. Prostřednictvím tohoto systému můžou hráči zakládat tickety s jejich dotazem a kategorizovat ho do určité kategorie. Moderátor mu poté na dotaz může odpovědět, případně k němu napsat komentář jinému moderátorovi a provést přiřazení.

ROLE:

Rolí s nejvyšší pravomocí bude **moderátor**, který bude moci procházet cizí tickety a na ně opovídat, přiřadit je jinému moderátorovi, upravovat kategorii ticketu (pokud jí hráč určí špatně) a uzavírat ticket. Poté zde bude role **hráč** pro hráče na serveru, kteří budou moci vytvářet tickety.

VSTUPY:

U **ticketů (tickets)** budeme evidovat jeho název, status, typ, kdy byl vytvořen a kým byl vytvořen. **Kategorie ticketu** je uložen v separátní tabulce (categories), kde je uložen jeho název a popis. Tento typ může mít více různých ticketů. Nový ticket může vytvořit hráč, moderátor bude poté moci ticket přiřadit nějakému jinému moderátorovi, nebo sobě (druhá vazba mezi tabulkou Users a Tickets), **změnit mu kategorii (ticket_categories)**, pokud ji hráč uvedl špatně a změnit mu status. Každý ticket bude mít svoje **zprávy** (messages), které bude moci hráč, nebo moderátor napsat. K ticketu si poté moderátoři mohou psát **poznámky (comments).**

Uživatel (users) bude mít povinně zadanou přezdívku, heslo a roli. Optimálně si může poté v nastavení zadat email. Uživatel si může svoje heslo a email kdykoliv změnit.

VÝSTUPY:

Z databáze se primárně bude vypisovat seznam **ticketů (tickets)** do seznamu na webu. Zde si poté může buď hráč, nebo moderátor specifický ticket otevřít a zde uvidí název, typ a status ticketu, pod tím zprávy seřazené od nejnovější po nejstarší a vpravo **poznámky** (comments) napsané jinými moderátory. Dále na stránce s daným ticketem se může ticketu upravit **kategorii**, změnit mu **status**, nebo ho **někomu přiřadit**.

Dále si můžou moderátoři na separátní stránce přidávat, případně odebírat **kategorie ticketů (categories)**.

Jako poslední výstup bude u uživatele (hráč + moderátor), kdy si v nastavení bude moci zobrazit svoji aktuální **přezdívku** a **email**.

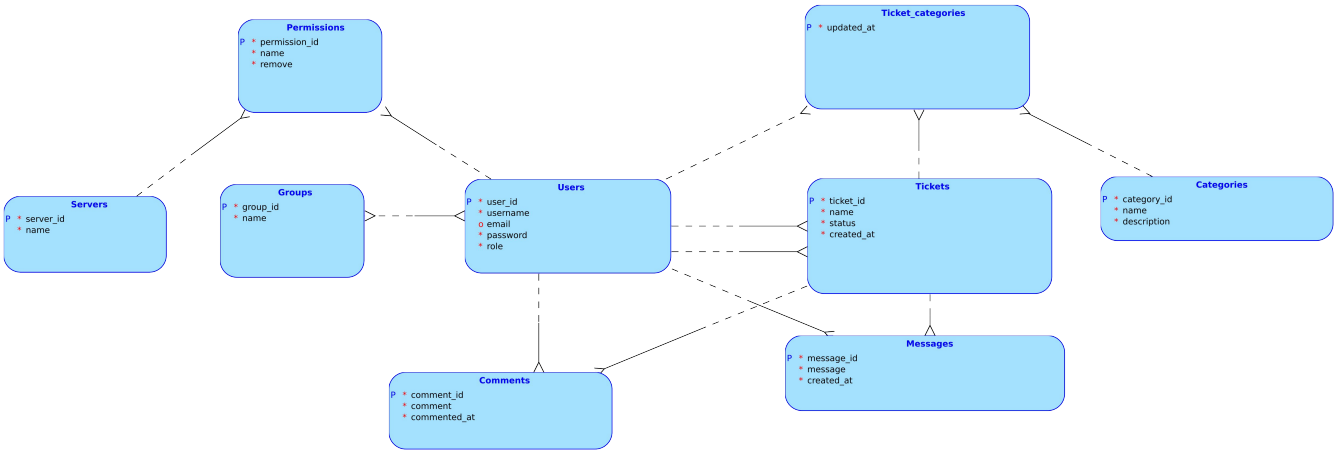
FUNKCE:

Jakmile hráč, nebo moderátor odepíše na ticket, tak se automaticky změní status ticketu, buď na „Čeká na odpověď hráče“, nebo „Čeká na odpověď moderátora“, tím se poté budou dané tickety řadit v seznamu ticketů výše.

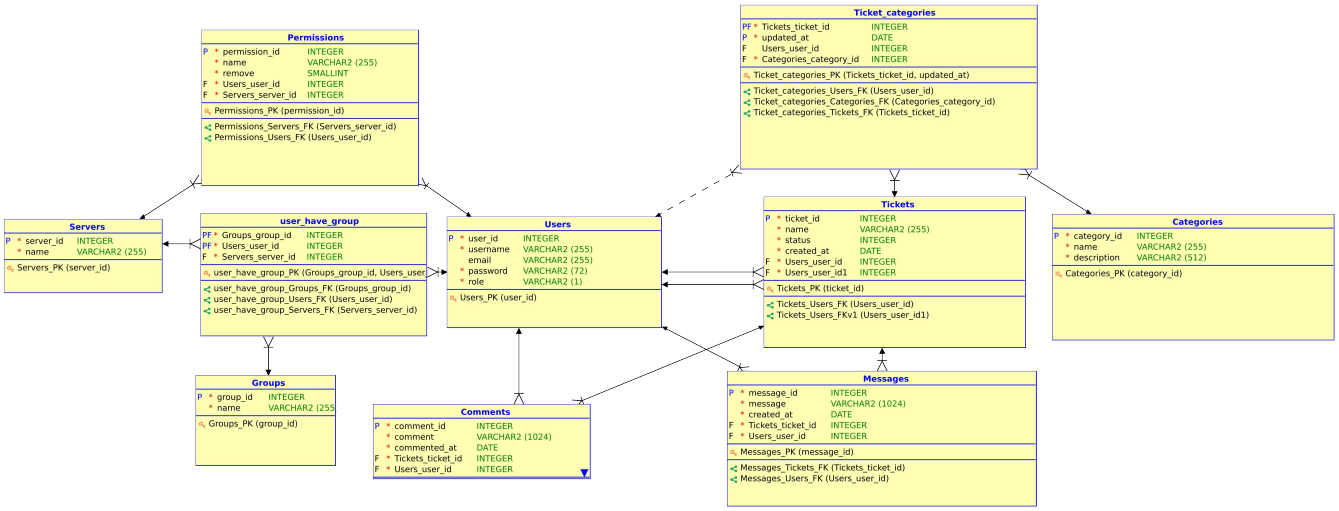
Automatický systém bude uzavírat tickety, které nebudou aktivní více jak měsíc, tedy pokud bude ticket ve stavu „Čeká na odpověď hráče“ a hráč od té doby 30 dní neodpověděl.

2 Datová analýza

Konceptuální datový model



Relační datový model



Datový slovník

Tabulka **Users**

Název atributu	Dat. typ	Délka	Klíč	Null	IO	Popis
user_id	INTEGER		Primární	Ne		Automatický inkrementovaný PK
username	VARCHAR	255		Ne		Uživatelské jméno, pro přihlášení uživatele
email	VARCHAR	255		Ano		Emailová adresa
password	VARCHAR	72		Ne		Heslo uživatele zašifrované algoritmem BCRYPT
role	VARCHAR	1		Ne	1	Role uživatele

Tabulka **Tickets**

Název atributu	Dat. typ	Délka	Klíč	Null	IO	Popis
ticket_id	INTEGER		Primární	Ne		Automatický inkrementovaný PK
name	VARCHAR	255		Ne		Název ticketu
status	INTEGER		Cizí (Statuses)	Ne		Stav ticketu
created_at	DATE			Ne		Datum vytvoření ticketu
Users_user_id	INTEGER		Cizí (Users)	Ne		Hráč, který vytvořil ticket
Users_user_id2	INTEGER		Cizí (Users)	Ne		Moderátor, který je přiřazen k ticketu

Tabulka **Messages**

Název atributu	Dat. typ	Délka	Klíč	Null	IO	Popis
message_id	INTEGER		Primární	Ne		Automatický inkrementovaný PK
message	VARCHAR	1024		Ne		Zpráva
created_at	DATE			Ne		Datum vytvoření zprávy
Tickets_ticket_id	INTEGER		Cizí (Tickets)	Ne		Ticket, ke kterému je zpráva přiřazena
Users_user_id	INTEGER		Cizí (Users)	Ne		Hráč, nebo moderátor, který zprávu napsal

Tabulka **Comments**

Název atributu	Dat. typ	Délka	Klíč	Null	IO	Popis
comment_id	INTEGER		Primární	Ne		Automatický inkrementovaný PK
comment	VARCHAR	1024		Ne		Komentář
commented_at	DATE			Ne		Datum, kdy byl komentář napsán
Tickets_ticket_id	INTEGER		Cizí (Tickets)	Ne		Ticket, ke kterému je komentář přiřazen
Users_user_id	INTEGER		Cizí (Users)	Ne		Moderátor, který komentář napsal

Tabulka **Categories**

Název atributu	Dat. typ	Délka	Klíč	Null	IO	Popis
category_id	INTEGER		Primární	Ne		Automatický inkrementovaný PK
name	VARCHAR	255		Ne		Název kategorie ticketu
description	VARCHAR	512		Ne		Popis kategorie ticketu

Tabulka **Ticket_categories**

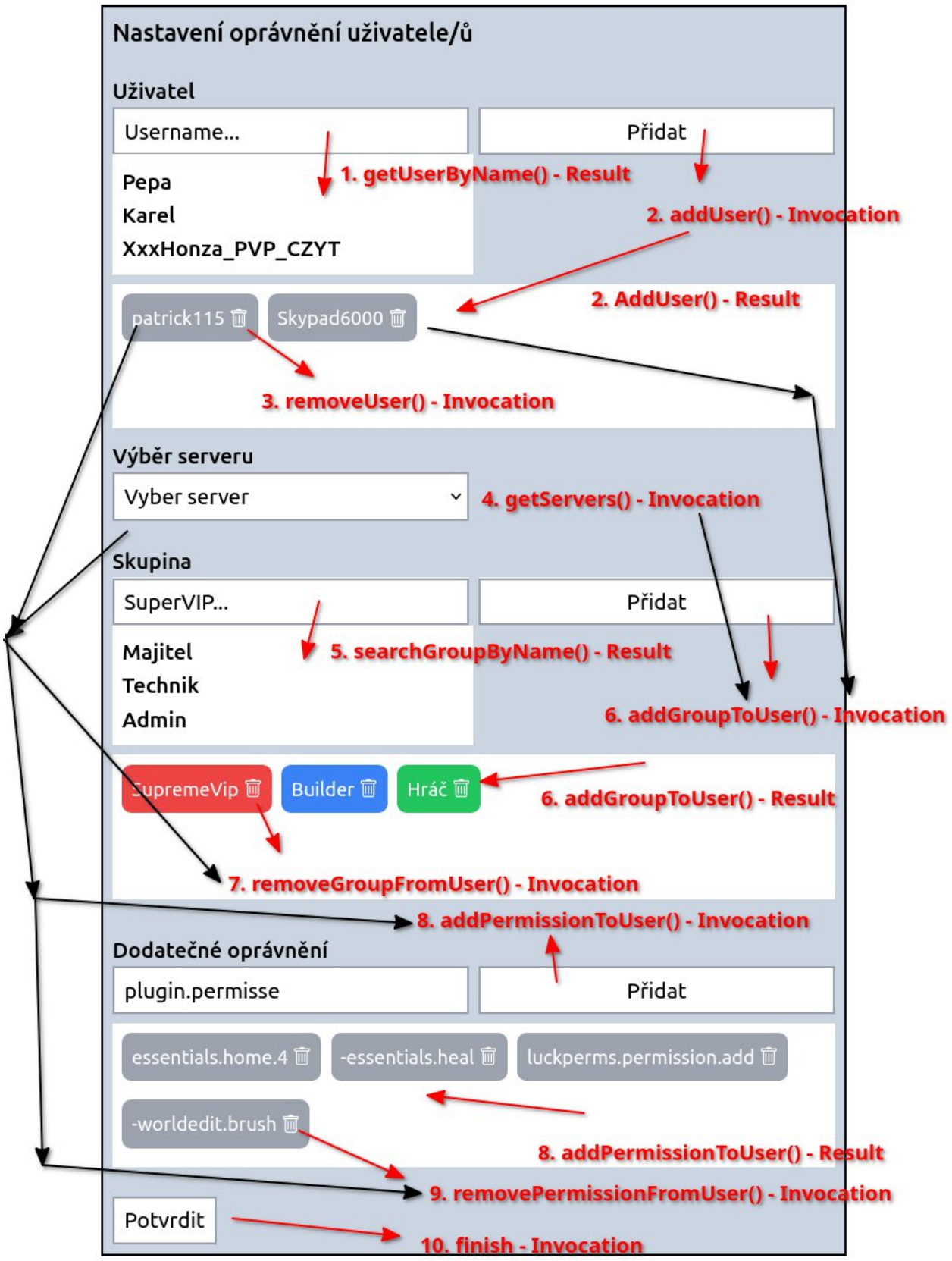
Název atributu	Dat. typ	Délka	Klíč	Null	IO	Popis
Tickets_ticket_id	INTEGER		Primární, Cizí (Tickets)	Ne		Ticket, ke kterému je kategorie přiřazena
updated_at	DATE		Primární	Ne		Datum, kdy byla kategorie přiřazena k ticketu
Users_user_id	INTEGER		Cizí (Users)	Ano		Moderátor, který kategorii přiřadil k ticketu. (Null při vytvoření ticketu, kdy si kategorii vybírá Hráč)
Categories_category_id	INTEGER		Cizí (Categories)	Ne		Kategorie, která je přiřazena k ticketu

Integritní omezení:

1. Role musí mít hodnotu buď "H" pro hráče, nebo "M" pro moderátora.

3 Funkční analýza

Návrh formuláře



Popis Formuláře:

Formulář slouží k nastavení skupin a oprávnění hráči/ům. Formulář může použít jen přihlášený uživatel, který na to má dané oprávnění. Po otevření formuláře se do proměnné vf_currentUserId uloží aktuální ID uživatele. Uživatel poté začne psát jméno (nickname) hráče a pomocí funkce 1) getUsersByName získá hráče s textem obsahující toto jméno, uživatel poté klikne na jméno a po kliknutí na tlačítko a funkce 2) addUser ho přidá do seznamu vf_selectedUsers. Pomocí ikonky koše u daného jména ho může pomocí funkce 3) removeUser odebrat ze seznamu. Pomocí funkce 4) getServers se do selectu vyberou dostupné servery a uživatel si poté server může vybrat a ten se uloží do proměnné vf_serverId. Dále uživatel může vyhledat skupinu, kterou chce uživateli/ům přidat, kdy při psaní se volá funkce 5) searchGroupByName a ta vyhledá skupiny, jejíž jména obsahují daný text. Uživatel poté může skupinu vybrat a pomocí funkce 5) addGroupToUsers ji přidá do tabulky user_groups danému uživteli/ům. Skupinu lze po kliknutím na ikonku koše a zavoláním funkce 7) removeGroupFromUsers odstranit. Poté může uživatel zadat specifické oprávnění, které má uživatel obdržet, nebo o něho přijít, pokud začíná ‚-', po zadání oprávnění klikne na tlačíko a funkcí 8) jí přidá do databáze, pokud teda začíná ‚-', tak do hodnoty remove dá 1 (true), jinak 0 (false). Kliknutím na ikonku u koše a pomocí funkce 9) removePermissionFromUsers . Na konci uživatel klikne na tlačítko potvrdit, které dokončí operaci a do tabulky log zapíše informaci o tom, že daný uživatel změnil oprávnění daným hráčům.

Proměnné formuláře:

a) vf_currentUserId – Id aktuálního uživatele, který formulář vyplňuje
b) vf_selectedUsers – Vybrání uživatelé funkcí 2) addUser, pro které se přidává skupina/oprávnění
c) vf_serverId – Id serveru, pro který přidává uživatelům skupinu/oprávnění

Funkce formuláře:

1) getUsersByName(varchar text) – vyhledá v databázi hráče, jejichž jméno obsahuje text a ty vrátí
2) addUser(varchar name) – přidá hráče do vf_selectedUsers
3) removeUser(varchar name) – odebere hráče z vf_selectedUsers
4) getServers() – vrátí seznam serverů, které existují
5) searchGroupByName(varchar text) – vyhledá skupiny, které obsahují text a ty vrátí
6) addGroupToUser(int groupId) - Podívá se na skupiny aktuálního uživatele na daném vybraném serveru a zkontroluje, jestli daná skupina není výše (má větší oprávnění), než kterou má on, aby nedošlo k tomu, že si uživatel sám sobě dá vyšší oprávňění. Pokud toto ověření neprojde, tak se ještě funkce podívá, jestli na daném serveru neobsahuje oprávnění, buď ‚*', která odpovídá maximálním oprávněním, nebo jestli nemá oprávnění na to, udělovat ostatním hráčům vyšší skupiny. Pokud ani přes to nemá dané oprávnění, funkce hodí chybu, že není možné přidat tuto skupinu, jinak vloží záznamy do vazební tabulky mezi user <-> group s parametrem pro daný server
7) removeGroupFromUsers(int groupId) – Odebere skupinu hráčům pro daný server
8) addPermissionToUsers(varchar permission) – Přidá oprávnění pro daný server všem uživatelům, prvně zkontroluje, jestli oprávnění začíná ‚-', či nikoliv, pokud ano, tak do parametru ‚remove' dá 1 (bráno jako true), jinak do něho dá 0.
9) removePermissionFromUser(varchar permission) – Odebere dané oprávnění vybraným uživatelům

10) finish() – Před ukončením do tabulky log zapíše, id daného uživatele a id uživatelů, kterým upravoval oprávnění a po dokončení resetuje formulář pro opětovné použití.

Minispecifikace:

addGroupToUser(int groupId)

Začátek transakce:

```
BEGIN TRANSACTION;
```

Zjištění maximální váhy skupiny uživatele (skupina s id 1 má nejvyšší váhu)

```
SELECT MIN(group_id) INTO p_group_id INTO FROM user_have_group
   WHERE user_id = vf_currentUserId AND server_id = vf_serverId;
```

Pokud je váha požadované skupiny větší, než má uživatel (chce přiřadit skupinu s větším oprávněním) tak zkontroluju zkontroluju, jestli nevlastní **oprávnění *,** nebo oprávnění pro přidávání těchto **skupin (permission.group.addhigher).**

```
SELECT COUNT(*) INTO p_perm_count FROM permissions WHERE
      user_id = vf_currentUserId AND server_id = vf_serverId AND
      (name = '*' OR name = 'permission.group.addhigher') AND remove = 0;
```

Pokud nemá ani toto oprávnění, tak funkce skončí s chybou a vypíše hlášku.

Pokud má uživatel oprávnění, tak pro každého uživatele, pro kterého chce přidat danou skupinu jí přidá

```
FORALL user IN vf_selectedUsers.FIRST..vf_selectedUsers.LAST
     INSERT INTO user_have_group (group_id, user_id, server_id)
     VALUES (f_groupId, user, vf_serverId);
```

A funkce skončí úspěchem.

```
COMMIT;
```