# BookKeeper – Správca študijných materiálov

## Zámer projektu

Keďže pri štúdiu používame veľa odbornej literatúry, rozhodol som sa spraviť aplikáciu, ktorá by zaznamenávala počet požičaných kníh, kde boli tieto knihy požičané; ak by sa jednalo o e-book, aplikácia by poskytovala odkazy kde sa daný e-book nachádza. Ak by užívateľ mal záujem vyhľadať nejakú knihu, mohol by použiť vyhľadávanie v aplikácii, ktoré mu napovie, v ktorej knižnici sa kniha nachádza, či je dostupná, resp. koľko kusov je ešte k dispozícií. Vyhľadávač bude reagovať na kľúčové slová, poprípade na tému danej knihy, a poskytne používateľovi najlepšie zhody.

Dostupnosť, nové tituly a správu „databázy“ kníh má na starosti administrátor, ktorý je vlastne unikátny používateľ (s unikátnymi právami meniť záznamy o knihách). Technicky to bude riešené cez dedenie. V prípade, že nejaká kniha nebude k dispozícii, môže sa užívateľ prihlásiť na odoberanie noviniek o dostupnosti tejto knihy (observer), čo ho vlastne upozorní, že je kniha znova k dispozícií.

Na zaznamenávania požičaných kníh študenta by slúžilo vlastné používateľské konto, kam sa bude prihlasovať svojim loginom a heslom. Samozrejme nebude chýbať registrácia nového užívateľa. Plánom je aj vytvorenie nejakého bazáru (marketplace), kam by mohli používatelia pridávať inzeráty, ak chcú niekomu ponúknuť knihu, poprípade majú záujem o kúpu.

## Návod na spustenie/používanie

Iniciálne okno programu je LoginWindow, pri ktorom má užívateľ registrovať sa, alebo sa prihlásiť (tlačidlá *Log in* a *Register*). Po úspešnej registrácii/prihlásení sa zobrazí hlavné menu, v ktorom si užívateľ môže pozrieť svoj profil a svoje vlastnené materiály, a druhá možnosť je vyhľadať iné materiály vo vyhľadávacom okne. V prípade, ak je prihlásený Administrátor je dostupná voľba *Administration.* Pri zvolení tejto možnosti sa otvorí správcovské okno, v ktorom má administrátor možnosť využiť testovacie prvky a editáciu zoznamu učebných materiálov.

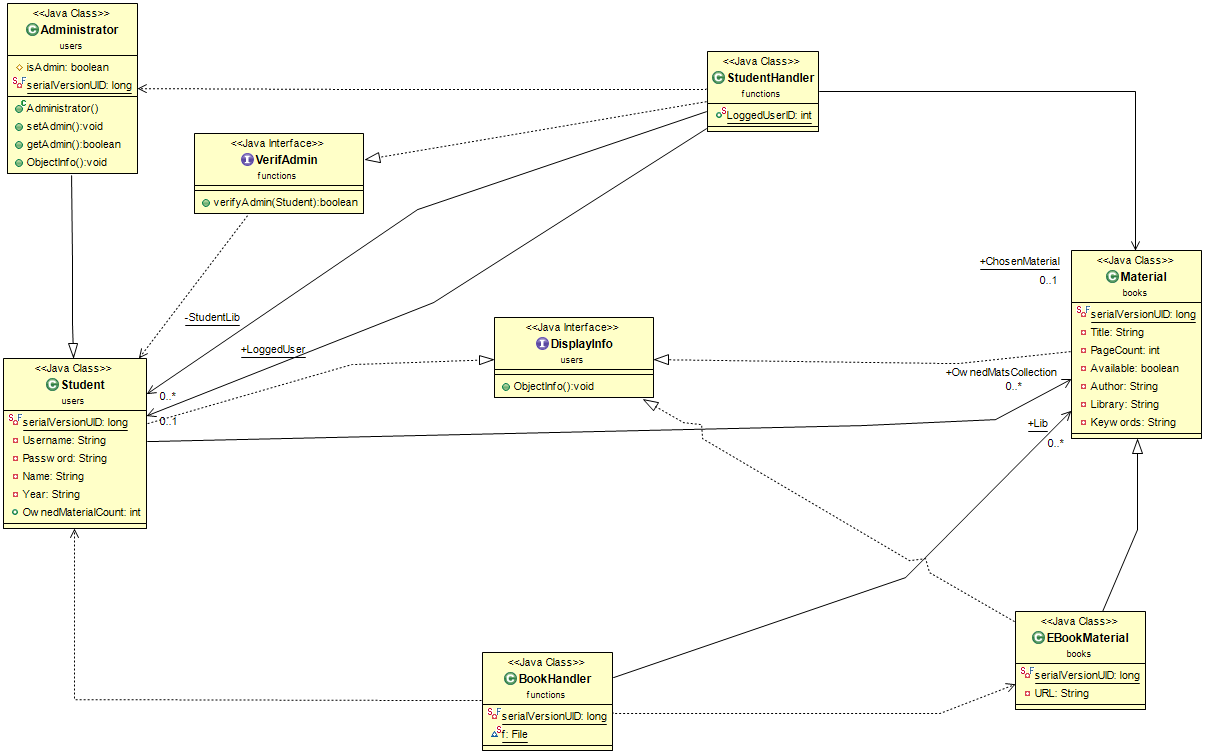
## Štruktúra tried

Hlavné triedy programu sú trieda Student, ktorá zastupuje normálneho užívateľa, a od nej odvodená trieda Administrator, ktorá disponuje atribútom, ktorý jej určuje administrátorskú rolu. Všetky detaily obsahuje diagram tried, ktorý je priložený v repozitári.

Kniha, resp. učebný materiál je zastúpená triedou Material, a od nej odvodená trieda EbookMaterial, ktorá sa odlišuje atribútom URL. Obe skupiny tried sa serializujú do svojich súborov; prihlasovacie údaje Studenta do súboru *CredentialsJ.ser* a údaje o materiáloch do *Materials.txt*.

Spracovanie týchto tried majú na starosti Handler triedy, konkrétne StudentHandler a BookHandler. Obe obsahujú záznamy typu ArrayList, ktoré sa serializujú/deserializujú podľa potreby. Obsahujú tiež vnútornú logiku programu a jednotlivé metódy, ktoré spracovávajú záznamy materiálov a užívateľov.

Diagram tried je nižšie, pre bližší pohľad sa nachádza aj v repozitári.



## Kritéria hodnotenia

Splnenie zámeru, funkčnosť programu

Zámer sa mi podarilo do väčšej časti splniť, okrem vytvorenia bazáru a systém odoberania noviniek o dostupnosti kníh. Vytvorenie bazáru by bolo časovo a všeobecne náročné, a systém odoberania nemá opodstatnenie, pretože sa materiály ukladajú do samostatného záznamu typu *ArrayList* u každého užívateľa. Mojim plánom tiež bolo vytvorenie auto-complete vyhľadávania, avšak toto by vyžadovalo importovanú knižnicu, čo by bránilo spusteniu programu v prostredí Eclipse bez tejto knižnice.  
Program je možné preložiť v základnom prostredí Eclipse.

### Dedenie, zapuzdrenie, polymorfizmus

Vetvy dedenia sú 2, a to medzi spomínanými triedami *Student* – *Administrator* a *Material* – *EbookMaterial*. Tieto objekty sa prekonávajú vo viacerých metódach, taktiež pridávanie do serializovaných *ArrayListov* spĺňa podmienku polymorfizmu (*StudentHandler*, *BookHandler* – pridávanie do *ArrayListu*). Zapuzdrenie je použité tiež v hierarchiách tried *Material* a *Student*, každá trieda je vybavená getter/setter metódami a ich vlastnosti sú atribútu private.

### Agregácia, rozhrania

Pri triede *Student* sa „vlastnené“ materiály ukladajú do *ArrayListu* typu *Material*, čo je vlastne agregácia typu *Material* v triede *Student*; každý *Material* v *ArrayListe* *OwnedMatsCollection* má svoje vlastné atribúty, čím spĺňa definíciu agregácie. Rozhranie ObjectInfo je použité pri testovaní, kde sa vypíšu hodnoty objektu (*Student* alebo *Material*). Rozhranie *verifyAdmin* obsahuje metódu, ktorá po implementácii overí, či je užívateľ *Administrator*.

### Organizovanie programu, oddelenie aplikačnej logiky, ostatné kritériá

Balík gui obsahuje len okná, a metódy, ktoré súvisia s grafickým rozhraním a s „vyhadzovaním“ výnimočných stavov (rozhodovanie, či program vyhodí Exception). Balík *functions* obsahuje všetky triedy, ktoré majú za úlohu riadiť užívateľov a ich záznamy, a takisto aj materiály a ich správu, a vlastné druhy Exceptions, ktoré sú použité v rôznych častiach programu (bližšie v rozšírených kritériách). Týmto je aplikačná logika výrazne oddelená od GUI časti.

Balíky *users* a *books* obsahujú jednotlivé typy používateľov a študijných materiálov.

Každá podstatná metóda má aj svoj komentár, kde je vysvetlené použitie danej metódy. Konkrétne príklady je možné vidieť v dokumentácii Javadoc priloženej v repozitári.

## Ďalšie kritériá

### Ošetrenie výnimočných stavov vlastnými výnimkami

Pri registrovaní nového užívateľa sa vyskytnú výnimky, ako napr. zlé vyplnenie registračného formulára, vtedy sa vyhodia vlastné výnimočné stavy (*Exceptions*) z balíka *functions*. Ďalšie takéto ošetrenie je napríklad pri prihlasovaní, kedy sa vypíše správa, ak nenašlo daného užívateľa, ktorý sa snaží prihlásiť. Podobne sú ošetrené výnimočné stavy v hľadaní materiálov, pridávaní a pod. Overené sú aj výnimočné stavy v textových poliach, ktoré spracúvajú čísla (ošetrené výnimky *NumberFormatException*).

### RTTI

RTTI je využité pri metóde verifyAdmin v triede StudentHandler; pri overovaní užívateľa je skontrolované, či je užívateľ inštanciou triedy Administrator. Program si týmto overí objektový *type*.

### Poskytnutie užívateľského rozhrania oddelene od aplikačnej logiky

Aplikačná logika je oddelená v triedach *StudentHandler* a *BookHandlerod* grafického rozhrania, kde sú spracované metódy na pridávanie do záznamov, rôzne operácie s objektami typov *Student* a *Material*.

Pri vyhľadávaní materiálov je vytvorená metóda *SearchForMaterial*, ktorá nájde tituly podľa zadaného textu, a tieto tituly slúžia ako obsah Modelu, ktorý sa vypíše do JListu vo vyhľadávacom okne, čím je spracovaný vstup a aj výstup, ktorý slúži len k nevyhnutnej tvorbe okna. Na takom istom princípe funguje aj zoznam vlastnených materiálov v profilovom okne (*ProfileWindow*). Pri úprave, pridávaní, a iných operáciach s učebnými materiálmi sú metódy na prácu s objektami uložené v triede *BookHandler.*

### Lambda výrazy

V triede *StudentHandler* sa pri vymazávaní zo záznamu vlastnených materiálov objektu typu Student využíva lambda výraz na hľadanie zvoleného titulu na vymazanie. Pokiaľ iterátor nájde objekt, ktorý spĺňa podmienku na vymazanie, tento objekt sa vymaže z užívateľského *ArrayListu* vlastnených materiálov.

### Generickosť

Generické metódy sa využívajú v pridávaní používateľa do serializovaného zoznamu používateľov, kedy sa automaticky vyberie metóda, ktorá prislúcha daným parametrom. Tiež *ArrayList* typu *Material* v triede *BookHandler* ktorý ukladá materiály oboch typov je generický tým, že automaticky rozoznáva typ daného parametra.

### Multithreading

Pri otvorení nového okna sa automaticky vytvorí vlastná niť (*thread*) v ktorej prebieha program, tým sú okná ochránené pred runtime chybou, keďže logické časti programu prebiehajú mimo nite rozhrania.

### Návrhové vzory

Vzor Composite je použitý pri ukladaní používateľov do serializovaného záznamu, keďže každý užívateľ má svoj vlastný ArrayList, ktorý slúži ako ďalšia úroveň v hierarchií záznamov.

## 2. Odovzdané verzie programu

1. Pracovná verzia 10.04.2018  
   Program obsahujúci základné prvky zámeru, základy funkcionality. Program obsahoval základné okná, fundamentálnu logiku na prácu so serializovanými súbormi, vyhľadavánie bolo implementované na testovacie účely.
2. Prezentovateľný program 14.05.2018  
   Program pripravený na prezentáciu a na používanie, pokrývajúci hodnotiace kritériá. Program obsahuje všetky časti programu na vyhľadávanie a prácu s učebnými materiálmi, užívateľské rozhranie, kompletné prihlasovanie, registrovanie užívateľov, užívateľské profily. Vytvorenie dokumentácie Javadoc a diagramu tried.

Konkrétne zmeny sa nachádzajú v histórií commitov na GitHub Classroom repozitári.