PIA

Semestrální práce

Kivbook

Jakub Váverka A17N0095P

Úvod

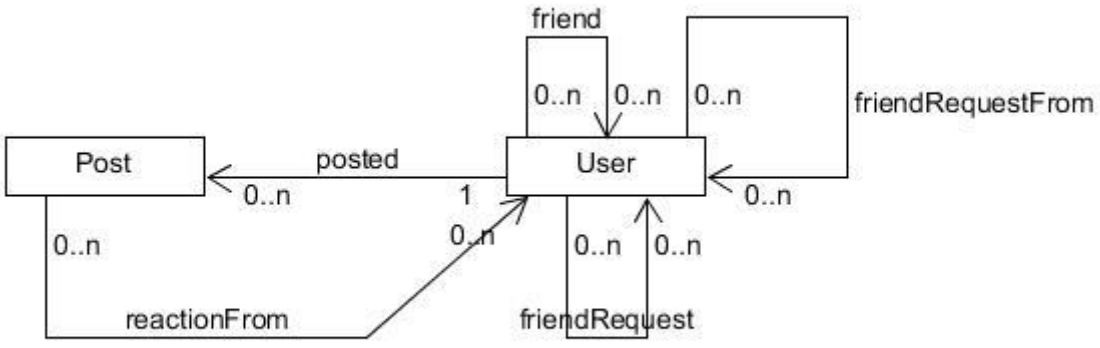
Toto je dokumentace k semestrální práci z předmětu PIA. Cílem bylo vytvoření sociální sítě kivbook a implementace volitelných funkcí.

Popis architektury

Aplikace je postavená na šabloně ze cvičení od profesora Daňka. Využívá MVC. Pro model používá Java Persistance API. Webové požadavky jsou obsluhovány pomocí servletů psaných v programovcím jazyku Java a generování html je prováděno přes technologii JavaServer Pages.

Datový model

Celý datový model je velice simplistický. Využívá jen 2 objekty. Během vývoje bylo objektů více, ale po analýze jsem zjistil že jsou poměrně zbytečné. Prvním objektem je objekt User, který v sobě uchovává informace o uživateli, jako je datum narození, přihlašovací jméno, pohlaví, a název profilového obrázku. Je dvakrát v relaci ManyToMany sám se sebou a tím tvoří přátelství a žádosti o přátelství. Zároveň je v relaci OneToMany s objektem Post. Objekt Post obsahuje informace o příspěvku, jako je čas vytvoření, obsah a také odkaz zpět na uživatele, který ho vytvořil. Teto relace je tedy obousměrná. Poté má v sobě ještě 3 relace ManyToMany na uživatele, které uchovávají jaký uživatel na jaký příspěvek reagoval.



Popis souborů

Při inicializaci se nejprve vytvoří aplikační kontext, který je ve složce main/ApplicationContext, poté se inicializuje /web/listener/ApplicationStartListener. Ten nastaví na jakých adresách poslouchá jaký servlet, inicializuje je a zároveň jim dodá potřebné objekty z aplikačního kontextu.

Servlety:

Nachází se ve složce /web/servlet/, vždy obsouží požadavky, které nan ě mohou přijít a poté odešlou řízení příslušnému *.jsp souboru, který vygeneruje html kód pro tazatele.

Main: Hlavní servlet, pokud je uživatel nepřihlášený přepošle ho do servletu Login, jinak zobrazí homepage

Login: Servlet, který obsluhuje přihlašování

Register: Servlet, který obsluhuje registrace

Profile: Obsluhuje uživatelův profil

Requests: Slouží k obsloužení žádostí o přátelství

Friends: Zobrazuje současná přátelství

Search: Obsluhuje požadavky na vyhledávání uživatelů a následné přidání do přátelství

JavaServer Pages:

Tyto soubory generují html kód pro webový prohlížeč.

homePage: Nepřihlášenému uživateli zobrazí všechny příspěvky od všech uživatelů. Pokud je uživatel přihlášený zobrazí příspěvky jeho a jeho přátel. Při prvním přehlášení je tedy prázdná

loginPage: V tomto projektu je použita jako index. První stránka co uživatel uvidí. Obsahuje jen jednoduchý přihlašovací formulář

registerPage: Na této stránce je možné provádět registrace.

profilePage: Stránka která v levém sloupci zobrazí náhled profilu a v prvém uživatelovi příspěvky. Slouží k zobrazení profilu přihlášeného uživatele i uživatelů ostatních

friendsPage: Zobrazí seznam přátelství přihlášeného uživatele. Je nepřístupná bez přihlášení

requestsPage: Zobrazí seznam žádostí o přátelství přihlášeného uživatele. Je nepřístupná bez přihlášení

searchPage: Zobrazuje výsledek hledání, který uživatel provedl

Managers and Services:

Manipulace s modelem je prováděna skrze 2 manažery, které mají definovaná rozhraní.

PostManager: operace nad příspěvky

UserManager: operace nad uživateli

Zároveň jsou využívány 2 služby

AuthenticationService: ověřuje zda je uživatel přihlášen

FormService: zpracovává požadavky, které chodí na servlety a volá vyžádané funkce od manažerů

DAO:

Ve složce dao jsou objekty pro práci s databází. Jsou zde jak rozhraní tak konkrétní implementace v pro JPA.

Závěr

Byl jsem velice nemile překvapen tím jak dlouho mi trvala implementace funkčnosti. Při programování jsem naplánoval velké množství funkcí, které jsem chtěl implementovat a poté co jsem zjistil, že se blíží řádný termín pro odevzdánil, jsem byl nucen dost funkcí odstranit. Nejvíce si myslím že utrpěl databázový model. Měl jsem vytvořeny objekty pro likování, přátelství a žádosti o přátelství. Poté co jsem si ale uvědomil že objekty v sobě nebudou mít nic víc než odkaz na jiný objekt, vznikla z nich vlastne jen přechodová tabulka, nebyl důvod je uchovávat a tak jsem přepsal většinu relací na ManyToMany a nechal JPA ať udělá relační tabulky za mě.