Projektrapport

Deltagande:

Linus Renström

Linus.renstrom@icloud.com

Emre Özata

emre10909@gmail.com

Brandon Adams Nkata

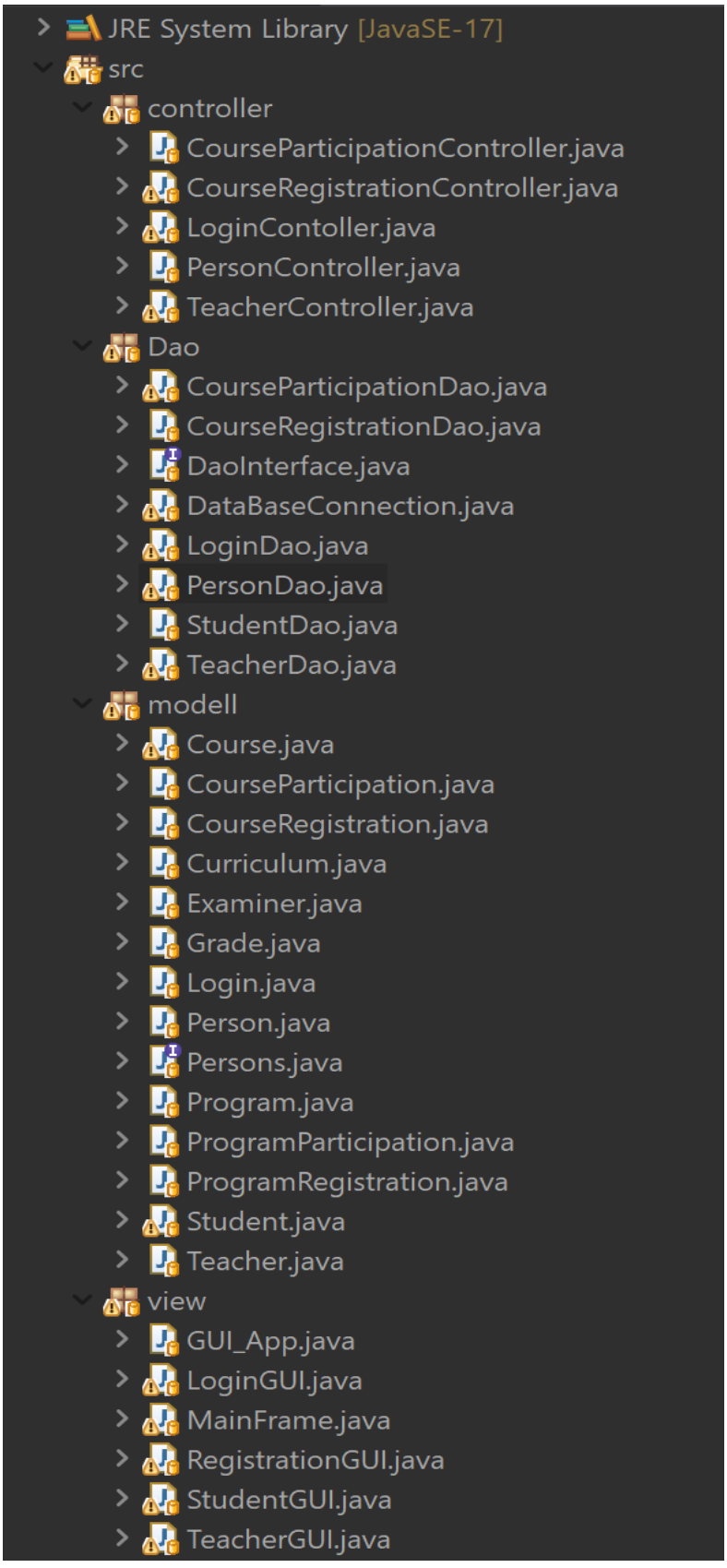
brandondalima@gmail.com

2024-10-31

Innehåll

1. Förklaring av design och tydliga beskrivningar av användningsfall

I projektet har paket och klasser delats upp strukturerat (se figur 1).



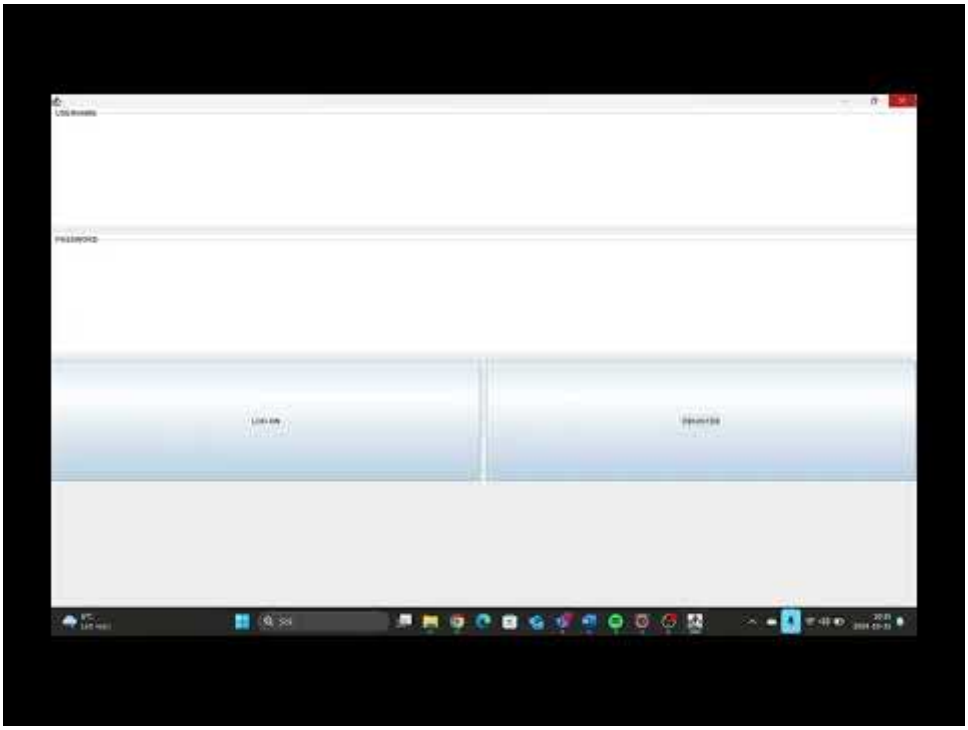
Figur 1 - figuren visar uppdelningen av klasser och paket.

1.2 Modell, Dao, controller, view design och CRUD

I projektet separeras modell-klasser, Dao-klasser, kontroller-klasser och vy-klasser i olika paket. Modell-klasserna skapades med hjälp av entiteterna från databasen som mall. Sedan tar Dao-klasserna emot vissa objekt och objektifierar hämtat data från databasen. Det vill säga att en person som hämtas från databasen skapas som objekt i Dao-klassen. Efter detta tar kontroller-klasser emot Dao-objekt och använder Dao-klassens metoder för att hämta, lägga in, visa och ta bort data från databasen (CRUD). I vyn anropas kontroller-klassernas metoder för att kunna uppdatera basen, samt så visar de resultat om vad som utfördes. Detta efterliknar MVC strukturen förutom att modellen ersätts med Dao-klasserna, vilket leder till att det blir Modell-Dao-Kontroller-View design-pattern. Men i vissa fall så måste kontroller-klasser och gui-klasser känna igen vissa modeller för att kunna objektifiera det hämtade datat.

1.3 Användningsfall

I koden beskriv de skapade anvädningsfallen.



1.3.1 Registrering och få en anpassad vy beroende på roll och konto

Detta användningsfall har de nödvändiga stegen som en användare av systemet måste utföra i systemets grafiska användargränssnitt för att kunna ha tillgång till systemets övriga funktionaliteter. Det är dels möjligt att registrera nya användare som inte finns i verksamheten, men det är även möjligt att sedan logga in som redan befintlig användare. Om en användare är ny till systemet och inte har ett inloggningskonto så finns det möjlighet att registrera sig. När användaren registrerar sig skall de fylla i förnamn, efternamn, personnummer, användarnamn och lösenord. De ska också markera om det vill registrera sig som en lärare eller student. Under registreringen så kommer den inmatade datan att hämtas från respektive fält och skickas till vår databas genom kontrollern och DAO klassen. Förnamn, efternamn och personnummer kommer att insertas till en person tabel i databasen. Användarnamn, lösenord och personnummer kommer att hamna i login tabellen. Sedan kommer valet av att registrera sig som en lärare respektive student att hantera kring vilken tabellpersonnumret hamnar i.

När en användare har ett konto så kan den logga in och komma åt de funktionaliteter den skall ha tillgång till. När användaren har fyllt i inloggningsdetaljerna. Användarnamnet och lösenordet skrivs i en JTextField och JPasswordField respektive.
Sedan bekräfta genom att klicka på en ”login” knapp så hämtas dessa uppgifter och bearbetas med hjälp an ett ”loginController” objekt.

Vid inloggning kontrollera systemet först om inloggningsdetaljerna är rätt genom att kolla med databasen via Dao-klassen som har hand om inloggning. Om inloggningen är kopplad till en befintlig person i systemet så utförs nästa steg där systemet kontrollera om användaren som logga in är en student eller en lärare. Detta görs också genom att kontrollera lämpliga tabeller i databasen med hjälp lämpliga Dao-klasser.
Om inloggningsuppgifterna stämmer så se vyn till att presentera ett anpassad grafiskt gränssnitt genom att kontakta och manipulera modellen via ”loginController”.
”loginController” få i sin tur flera objekt i sin konstruktör. Dessa används för att bland annat kommunicera med databasen och för att kommunicera modellklasserna.

1.3.2 Student ser alla tillgängliga kurser och anmäler sig till kurs

Detta anvädningsfall består av att en student som loggar in i systemet får möjlighet till att till att se de kurser som verksamheten erbjuder. Då användaren har angett ett användarnamn och lösenord som är korrekt och har en studentroll får den upp en unik student-vy. I denna vy har studenten flera olika knappar som den kan klicka på och en av dem är att se tillgängliga kurser (”Get courses”) inom verksamheten. Då användaren markerar en tillgänglig kurs på vyns JList och sedan klickar på att registrera sig på vald kurs (”Register to selected course”) får användaren upp ett meddelande att den har blivit antagen till kursen. Användaren kan inte endast klicka på ”registrera sig på vald kurs” knappen utan då blir användaren ombedd att välja en kurs ur listan. När användaren registrerar sig på en kurs så kommer en insert sats i vår skapade DAO klass att anropas där genom en kontroller klass. I DAO klassen så kommer information likt student-ID, vecka för kursstart, år för kursstart, en unik registreringskod och kurs koden att infogas i databasen.

1.3.3 Student ser alla kurser den är registrerad på

Detta användningsfall finns i studentvyn och med hjälp av att vi sparar just det student-id som loggar in kan vi enkelt få upp en JList av alla kurser studenten är registrerad på. Detta medför till att det sker et select statement i en sql-sats i en av våra Dao-klasser.

1.3.4 Student hoppas av från kurs

Detta användningsfall finns i studentvyn och tar bort den inloggande studenten från en av sina antagna kurser. Med hjälp av en JButton så kan studenten få upp sina registrerade kurser i en JList, vilket leder till att en JList-lyssnare kan läggas till. Detta medför till att en kurs kan väljas ur listan och därefter, med hjälp av ytterligare en JButton, kan en elev välja att hoppa av den kursen genom att klicka på den knappen. Detta medför till att det sker ett Delete statement i en sql-sats i en av Dao-klasserna.

1.3.5 Lärare kan se vilka kursen den undervisar

Detta användningsfall finns i lärarvyn och visar kurser som läraren undervisar. Genom att trycka på en JButton så fylls en JList med all kursinfo som vi hämta från databasen med en select-sats i en av våra Dao-klasser.

1.3.6 Lärare kan se vilka elever som finns på kursen

Detta användningsfall finns i lärarvyn och har ett användarvänligt gränssnitt. En inloggad lärare kan trycka på en JButton för att få upp sina kurser som i tidigare användningsfall (1.3.7) och därifrån klicka på ytterligare en JButton för att få upp antagna studenter på kursen.

1.3.7 Lärare kan sätta betyg på en elev på specifik kurs

Detta användningsfall finns i lärarvyn och lägger till en student i grades-tabellen i vår databas. Gränssnittet är användarvänligt då lärare kan se vilka elever som finns med på en kurs som just den inloggade läraren undervisar. Då kan den med hjälp av en JList-lyssnare klicka på en elev och får då upp en JOptionPane som hanterar inputs och där skriver man in betyg i formatet (1–5). Därefter läggs all data in i databasen i vår tabell ”grades”.

2. UML-diagram

Till detta projekt har det skapats ett UML-diagram (Se bilagor). I detta UML-diagram ser man associationer samt kompositioner bland klasser. UML-diagrammet visar också systemet i sin helhet samt hur allt hör ihop med varandra.

3. Förbättringar av projektet

En förbättringspunkt som vi kom på är att vi kunde ha programmerat mer mot ett interface för att minska beroenden mellan vissa klasser. Att programmera mot ett interface skulle också medföra till ett robustare system, polymorfism och bättre återanvändning av kod. Anledningen till att vi inte använde interfaces var för att vi gick efter vår databas och utifrån dess attribut, vilket gjorde att modell-klasserna gick fort att skapa.

Vi kunde även ha strukturerat det bättre genom att använda en cardlayout och få in alla vyer i mainframe istället för att låta vårt login-GUI skapa alla nya vyer.

4. Bilagor

Obj-orientering-UML.drawio (1).pdf