PUSP174214

v.1.0

STLDD - HÖGNIVÅDESIGN
ETSF20 GRUPP 2

Ansvarig Grupp: (SG)Systemgrupp
Uppgjord Av: (UG)Utvecklingsgrupp

8 mars 2017

Innehåll

Dokumenthistorik

Ver.	Date	Resp.	Description
0.1	2017-02-24	SG	STLDD skapad
0.2	2017-02-25	SG	Översikt skapad över vy- och kontroller-nivå
0.3	2017-02-27	SG	Översikt kompletterad med modell-nivå
0.4	2017-02-27	SG	Översikt kompletterad med övriga klasser på modellnivå, samt skapande av databas , sekvensdiagram, routning och appendix A

1 Introduktion

Detta dokument beskriver högnivådesignen för ett tidrapporteringssystem, E-Kyss. Systemet i fråga är en utveckling utav ett redan befintligt system, ”BaseBlockSystem”. Systemet innehåller flera olika funktioner såsom att skapa tidsrapporter, hålla reda på rapporterad tid utav mindre grupper i projektgruppen. Projektledare har även möjlighet att se en sammanställning utav all rapporterad tid.

2 Referensdokument

- SRS PUSP174212 version: 0.2
- BaseBlockSystem STLDD: PUSS1204 version: 1.0 gäller för alla punkter om det inte är specificerat i respektive underrubrik att det som står i PUSS412004 version: 1.0 utgår för specifik del.

3 Terminologi

- **View**: Vyer i systemet har till uppgift att presentera modeller i det format klintens anrop önskar.
- **Applikationsserver**: En applikationsserver är ett mjukvaruramverk som tillhandahåller möjligheter att skapa webbapplikationer, samt servermiljö att exekvera dom på.
- **JSP**: JSP är Java-teknologi för att skapa dynamisk vy. JSP agerar i systemet som vy enligt MVC.
- **MVC** (Model View Controller): Är ett mjukvarudesignmönster för skapande av bland annat grafiska gränssnitt, men har under senare år även börjat användas inom webbutveckling. Tanken bakom konceptet är att dela upp en applikation i tre delar, för att kunna separera interna representationer av data, och och logiken bakom för hur den datan handskas ifrån hur den informationen och datan presenteras för slutanvändaren. Grundtanken är den att återanvändning av kod ska bli enklare och att en mer parallell utveckling ska möjliggöras.
- **Servlet**:Servlet är små program som exekveras på serversidan av en web-anslutning. En servlet utökar dynamiskt funktionaliteten hos en webbserver. För att använda eller skapa servlets, krävs tillgång till en servlet-container/server, två populära alternativ är Tomcat, som E-KYSS är utvecklat för, eller Glassfish.
- **Java Bean**: En mjukvarukomponent som har blivit designad för att kunna återanvändas i ett varierat antal miljöer. Det finns inga restriktioner på vad en Java Bean kan göra eller användas till, men grundtanken bakom Java Bean-teknologin, är introspektion vid runtime. Används här som datastruktur samt transport av data mellan modell och vy-domän.

4 Översikt

Systemet är utvecklad för applikationsservern Apache Tomcat, där Java Servlet, JavaServer Pages (JSP) samt Java Expression Language (JSP EL)-teknologier tillämpas enligt JavaEE:s 2:a MVC-modellarkitektur. Systemets huvudfunktion är att fungera som ett tidrapporteringssystem.

4.1 Vynivå: JSP (JavaServer Pages)

Nedan förklaras vad varje fil för JSP:n har för betydelse.

4.1.1 login.jsp

Denna fil innehåller vy-design som representerar inloggningsformuläret på förgrunds nivå, d.v.s. för den webbsida som är publikt tillgänglig utan inloggning. Denna webbsida används för att komma åt bakgrundsnivån för tidrapporteringsapplikationen. Designen består utav en inloggningsfält för användarnamn, ett för lösenord samt en rullista över val av projektgrupper. Kontrollen för denna vy är **class LoginServlet**.

4.1.2 groupmanagement.jsp

Filen innehåller vy-design som representerar projektgrupphanteringen för webbapplikationen som används av administratören. Här finns ett formulärfält för att lägga till en ny projektgrupp, samt en lista över vilka projektgrupper som existerar i systemet. I denna listan finns funktionen att ta bort aktuell projektgrupp från systemet. Kontrollen för denna vy är **class GroupManagementSevlet**.

4.1.3 usermanagement.jsp

Vy-designen för användarhantering representeras av denna fil. Vyn används av administratör och projektledare där vyn anpassas utifrån användarens behörigheter utifrån dess roll i systemet. Kontrollen för denna vy är **class UserManagementServlet**.

4.1.4 reportmanagement.jsp

Filen innehåller vy-designen som representerar hantering av veckorapporter. Endast projektledaren har tillgång till denna webbsida. Kontrollen för denna vy är **class ReportManagementServlet**.

4.1.5 report.jsp

Filen innehåller vy-designen som representerar skapandet av veckorapporter. Projektledare och användare kommer använda denna webbsida för veckorapportering. Automatiserad signering av

projektledarens veckorapportering sker på modellnivå när veckorapporten skickas in till servern. Kontrollen för denna vy är **class ReportServlet**.

4.1.6 dashboard.jsp

Vy-design som representerar sammanställning av tidrapporter som renderas ut på sidan. I denna vy väljs även hur sammanställningen ska presenteras genom att välja olika renderingsformer av sammanställningen. Kontrollen för denna vy är **class DashBoardServlet**.

4.1.7 user.jsp

I denna vy-design ser användaren sin personliga information, och kan ändra sitt lösenord. Kontrollen för denna vy är **class UserServlet**.

4.2 Kontrollernivå: Servlet

Klasserna som specificeras nedan agerar som kontroller i systemet.

4.2.1 LoginServlet

Denna kontrollern tar hand om kommunikationen mellan **login.jsp** och modellen **LoginBean**, som innehåller information om en given projektgrupp. Denna **LoginBean** skickas som en lista till **login.jsp** för att rendera ut valen av projektgrupp vid inloggning.

URL-pattern: ./

4.2.2 GroupManagementServlet

Tar hand om kommunikationen mellan **groupmanagement.jsp** och bönan **GroupManagementBean**.

URL-pattern: ./management/groups

4.2.3 UserManagementServlet

Tar hand kommunikationen mellan **usermanagement.jsp** och bönan **UserManagementBean**.

URL-pattern: ./management/users

4.2.4 ReportManagementServlet

Tar hand om kommunikationen mellan **reportmanagment.jsp** och bönan **ReportManagementBean**.

URL-pattern: ./management/reports

4.2.5 ReportServlet

Tar hand om kommunikationen mellan **report.jsp** och bönan **ReportBean**.

URL-pattern: ./reports

4.2.6 DashboardServlet

Tar hand om kommunikation mellan **dashboard.jsp** och bönan **DashboardBean**.

URL-pattern: ./dashboard

4.2.7 UserServlet

Tar hand om kommunikationen mellan **user.jsp** och bönan **UserBean**.

URL-pattern: ./settings/user

4.3 Modellnivå: JavaBeans

Klasserna som specificeras nedan kommer skickas till vynivån i systemet.

4.3.1 LoginBean

En get/set-klass innehållandes data som krävs av **login.jsp** för att rendera vyn.

4.3.2 GroupManagementBean

En get/set-klass innehållandes data som krävs av **groupmanagement.jsp** för att rendera vyn.

4.3.3 UserManagementBean

En get/set-klass innehållandes data som krävs av **usermanagement.jsp** för att rendera vyn.

4.3.4 ReportManagementBean

En get/set-klass innehållandes data som krävs av **reportmanagement.jsp** för att rendera vyn.

4.3.5 ReportBean

En get/set-klass innehållandes data som krävs av **report.jsp** för att rendera vyn.

4.3.6 DashboardBean

En get/set-klass innehållandes data som krävs av **dashboard.jsp** för att rendera vyn.

4.3.7 UserBean

En get/set-klass innehållandes data som krävs av **user.jsp** för att rendera vyn.

4.4 Övriga klasser på modellnivå

Klasser som specificeras nedan skickas aldrig till vynivån.

4.4.1 Database

Hanterar databasuppkopplingen genom designmönstret singleton.

4.4.2 DatabaseHandler

Hanterar SQL-förfrågningar direkt mot databasen.

4.4.3 BeanFactory

Matar in information från databasen till bönorna.

4.4.4 BeanUtilities

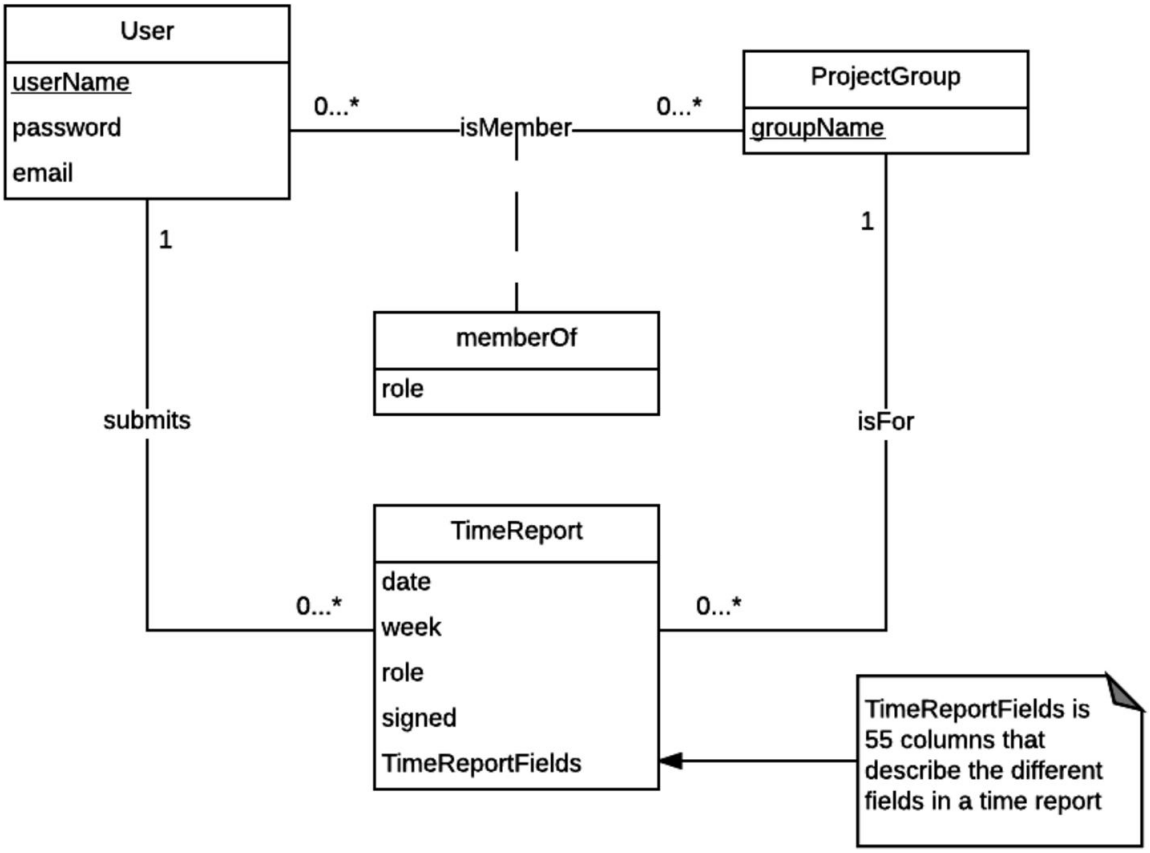
Matar in information från HTTP-förfrågan (“request”) till bönorna.

4.4.5 BeanTransaction

Extraherar bönans information och utför databaskommunikation genom **class DatabaseHandler**.

5 Databas

Figur 1: Databasen består av tre huvudsakliga tabeller Users, ProjectGroups och TimeReports. Den har även en relationstabell memberOf som beskriver relationen mellan Users och ProjectGroups.



Users(<u>userName</u>, password, email)

ProjectGroups(<u>groupName</u>)

TimeReports("field 1", ... , "field 65")

memberOf(*groupName*, *member*, role)

5.1 Users

Users(<u>userName</u>, password, email)

Users är en tabell som innehåller användare som är inlagda i systemet. Den består av tre kolumner: userName, password och email. userName är primary key vilket innebär att det endast kan finnas en användare med det användarnamnet.

Figur 2: Tabellen har följande utseende:

```
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| userName | varchar(10) | NO   | PRI | NULL    |       |
| password | varchar(6)  | NO   |     | NULL    |       |
| email    | varchar(100)| NO   |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
```

Listing 1: Tabellen kan konstrueras med följande SQL-satser:

```
mysql> CREATE TABLE Users (
    -> userName varChar(10) NOT NULL,
    -> password varChar(6) NOT NULL,
    -> email varChar(100) NOT NULL,
    -> PRIMARY KEY (userName));
```

5.2 ProjectGroups

ProjectGroups(<u>groupName</u>)

ProjectGroups är en tabell som innehåller de olika projektgrupperna i systemet. Den innehåller endast en kolumn, groupName, som är satt som "primary key" vilket innebär att alla projektgrupper måste ha unika namn.

Figur 3: Tabellen har följande utseende:

```
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| groupName | varchar(100)| NO   | PRI | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
```

Listing 2: Tabellen kan konstrueras med följande SQL-satser:

```
mysql> CREATE TABLE ProjectGroups (
    -> groupName varChar(100),
    -> PRIMARY KEY (groupName));
```

5.3 TimeReports

TimeReports är en tabell som innehåller all information som rör tidrapporterna. Då tabellen har många kolumner (65 st) visas den inte här. För att få ut relevant information ur tidrapporterna så finns det en hjälp-procedure som tar fram relevant information för tillfällen då detta kan kräva många rader kod. För enklare fall så räcker det med en vanlig SELECT. Det finns även två triggers, en för när man lägger in en ny rad, och en när man uppdaterar en existerande rad, som väljer rätt roll för användaren, uppdaterar de totala värdena för aktiviteterna med subaktiviteter (11-19), uppdaterar den totala tiden spenderad för subaktiviteterna (d, i, f och r) och den totala tiden rapporten avser.

5.4 memberOf

memberOf(*groupName*, *member*, role)

memberOf är en relationstabell som beskriver relationen mellan användare och de olika projektgrupperna. Den har tre kolumner: groupName, member och role. Kolumnerna groupName och member är tillsammans satta som UNIQE, vilket resulterar i att en medlem endast kan vara medlem i samma grupp en gång (en roll per projektgrupp). *groupName* är en "foreign key" som refererar till **ProjectGroup(groupName)** och *member* är en "foreign key" som refererar till **Users**(userName).

Figur 4: Tabellen har följande utseende:

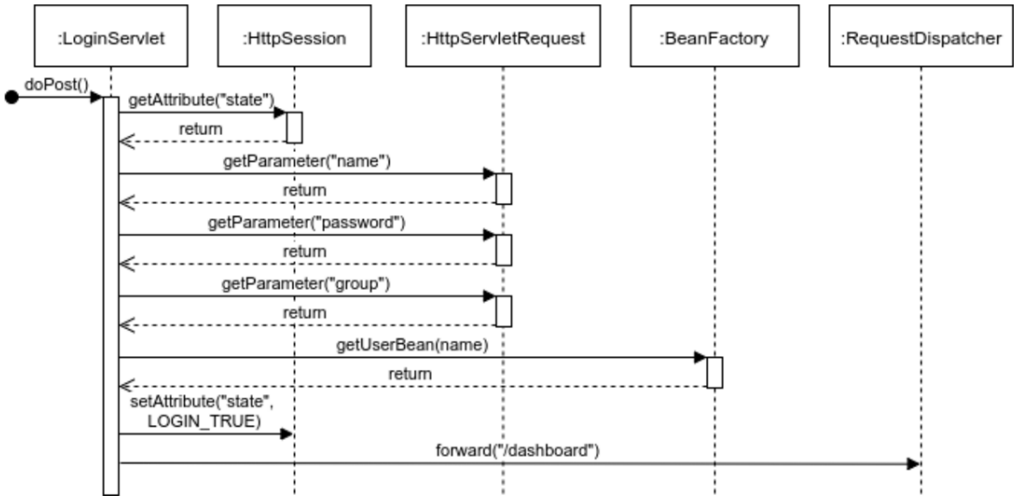
```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| groupName | varchar(100) | YES  | MUL | NULL    |       |
| userName  | varchar(10)  | YES  | MUL | NULL    |       |
| role      | varchar(10)  | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
```

Listing 3: Tabellen kan konstrueras med följande SQL-satser:

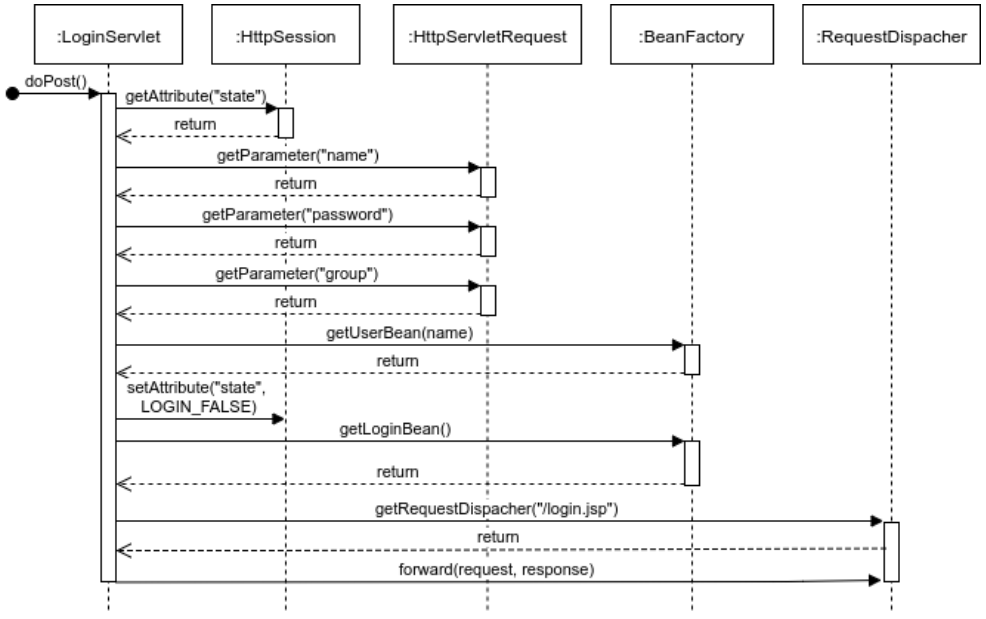
```
mysql> CREATE TABLE memberOf (
    -> groupName varChar(100),
    -> member varChar(10),
    -> role varChar(10),
    -> UNIQUE (groupName, member),
    -> FOREIGN KEY (groupName) references ProjectGroups(groupName)
    -> ON UPDATE CASCADE ON DELETE CASCADE ,
    -> FOREIGN KEY (member) references users(userName)
    -> ON UPDATE CASCADE ON DELETE CASCADE);
```

6 Sekvensdiagram

6.1 Class LoginServlet

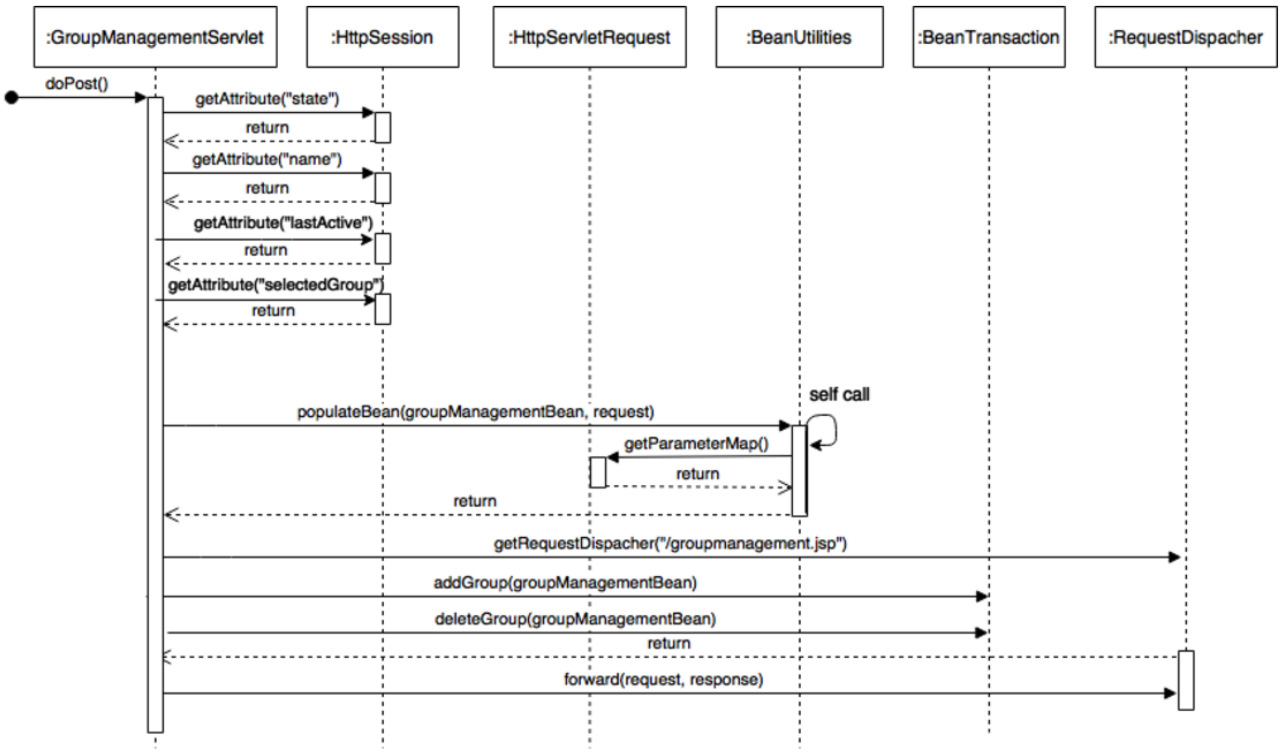


Figur 5: Sekvensdiagram som hanterar en lyckad inloggningsförfrågan

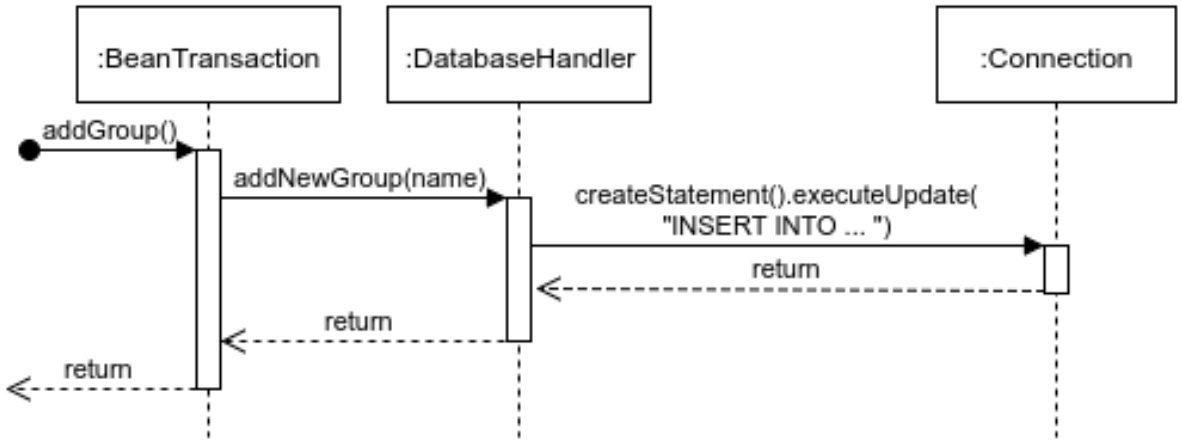


Figur 6: Sekvensdiagram som hanterar en icke-lyckad inloggningsförfrågan.

6.2 Class GroupManagementServlet

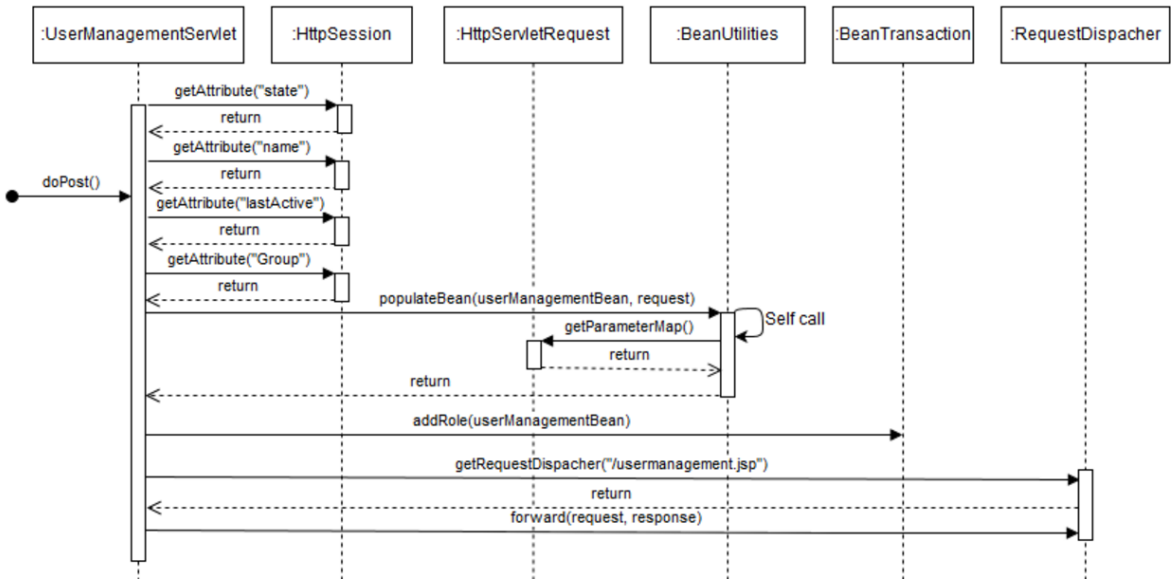


Figur 7: Sekvensdiagram som hanterar tilläggande av ny användare samt borttagande av existerande användare.



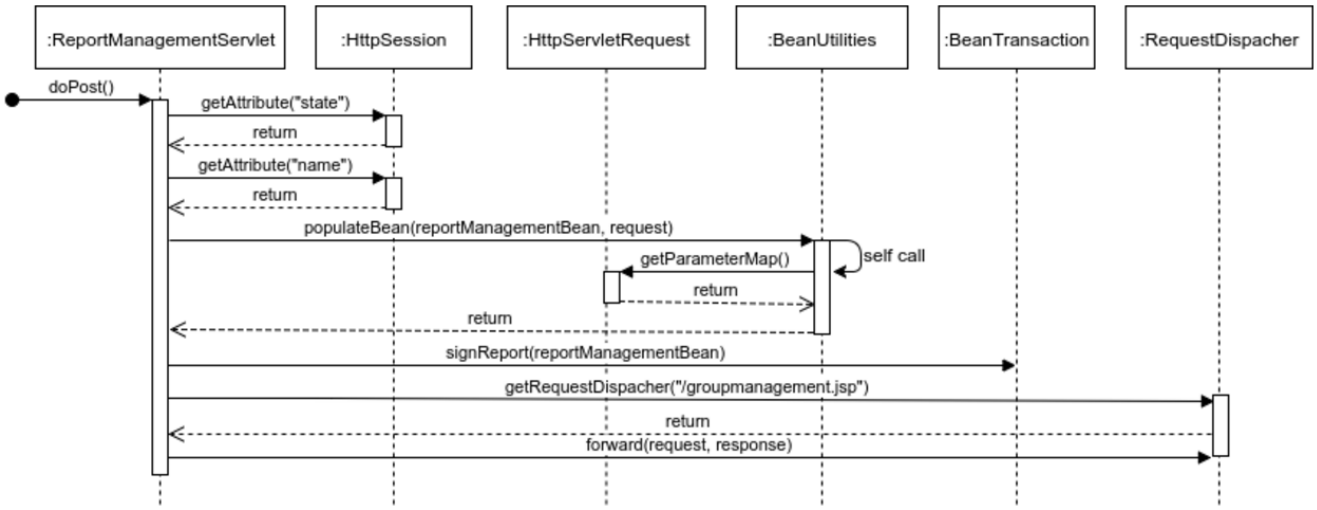
Figur 8: Sekvensdiagram som visar vad klassen BeanTransaction gör vid metoden addGroup().

6.3 UserManagementServlet

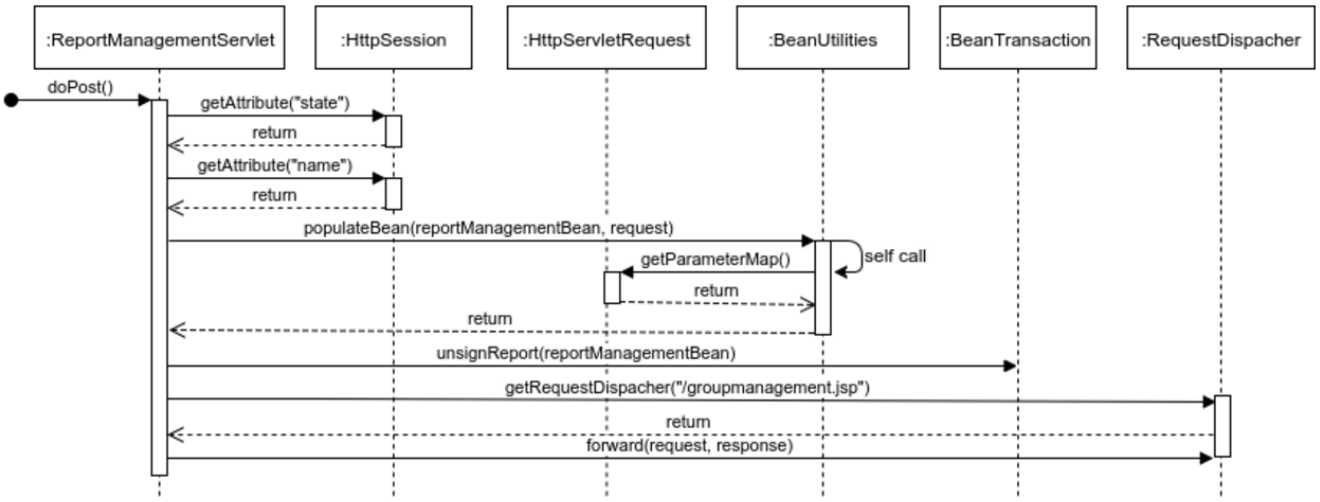


Figur 9: Sekvensdiagram som beskriver rolltilldelning av en projektmedlem.

6.4 Class ReportManagementServlet

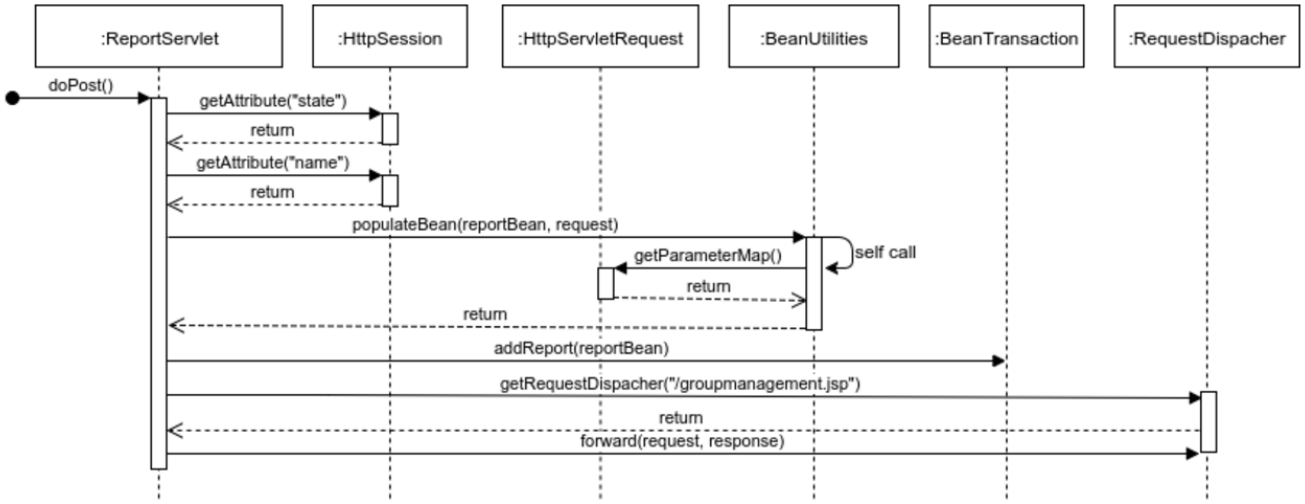


Figur 10: Sekvensdiagram som beskriver hur en veckorapport signeras av en projektledare.



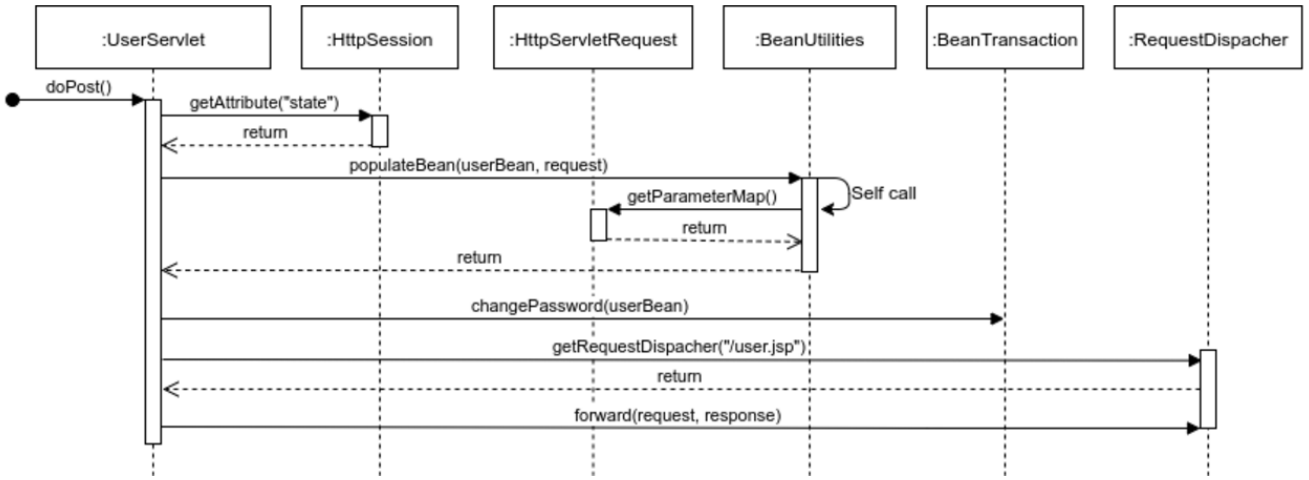
Figur 11: Sekvensdiagram som beskriver hur en eller flera veckorapporters signering tas bort av en projektledare.

6.5 ReportServlet

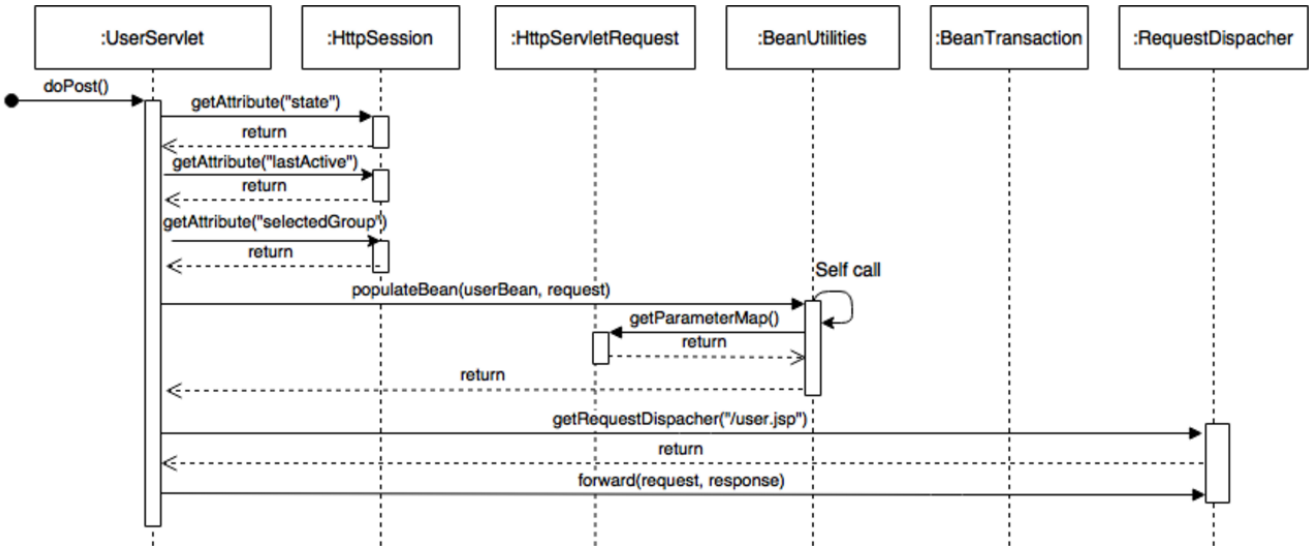


Figur 12: Sekvensdiagram som beskriver hur en projektmedlems tidrapportering sparas.

6.6 UserServlet

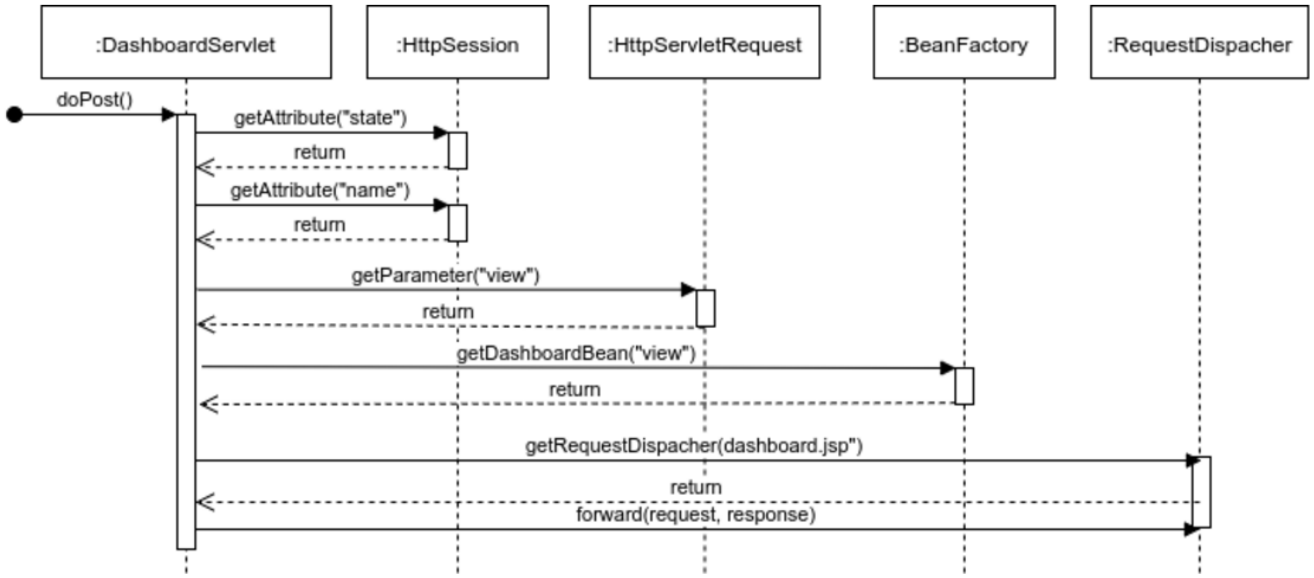


Figur 13: Sekvensdiagram som beskriver en lyckad ändring av en projektmedlems lösenord.



Figur 14: Sekvensdiagram som beskriver en misslyckad ändring av en projektmedlems lösenord.

6.7 DashBoardServlet



Figur 15: Sekvensdiagram som beskriver användandet av DashBoardServlet

7 Routing

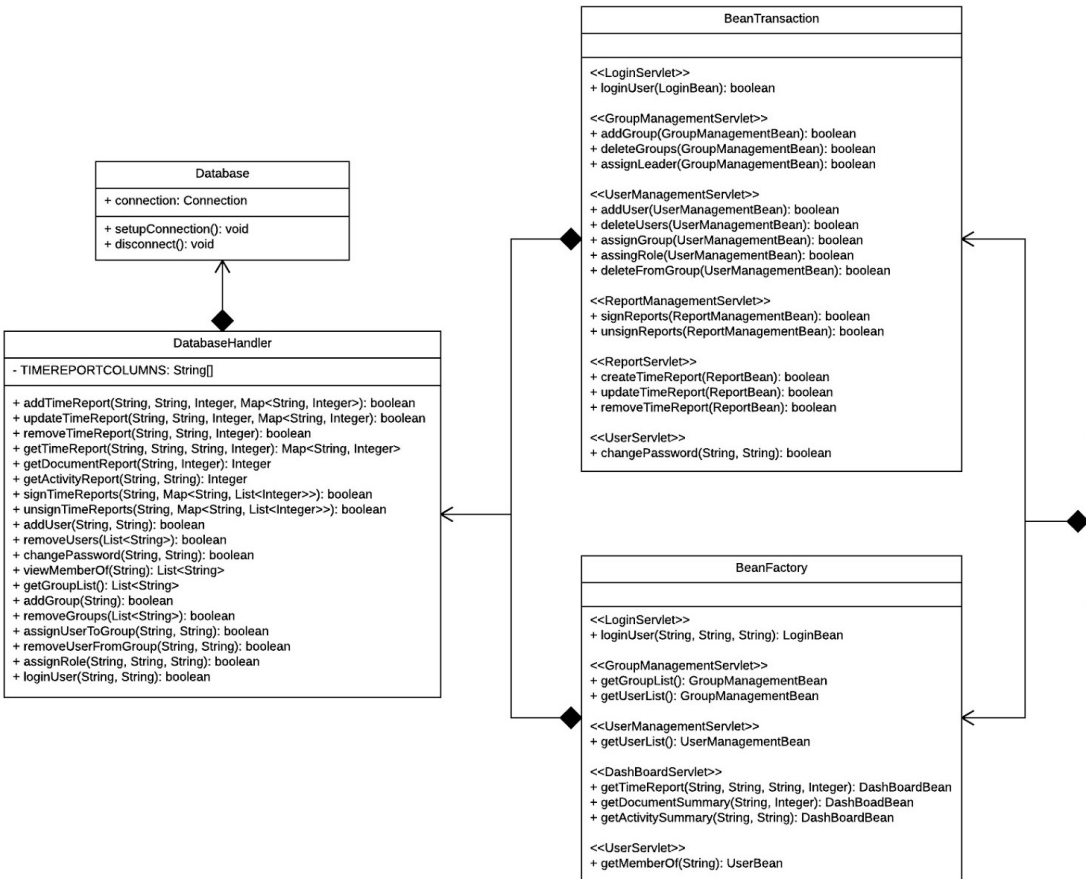
Servlets konfigureras till att lyssna på specificerat HTTP URL-mönster, genom användning av @Webservlet-annotering inuti klassfilen. Denna annotering bearbetas av Tomcat vid så kallad deployment

Servlet-Class	URL-Pattern
LoginServlet	./ ./login ./logout
GroupManagementServlet	./management/groups
UserManagementServlet	./management/users
ReportmanagementServlet	./management/reports
ReportServlet	./report
DashBoardServlet	./dashboard
UserServlet	./settings/user

Tabell 1: Routing

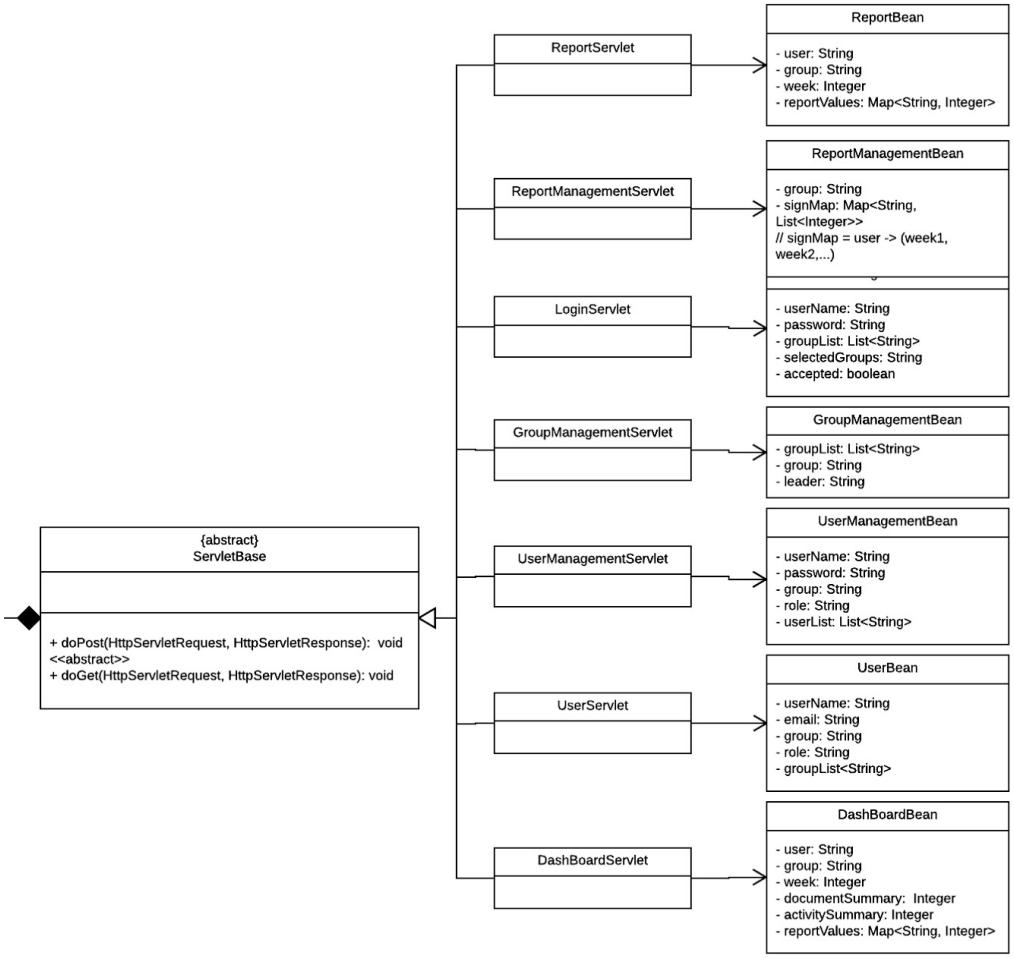
8 Appendix A

Klassdiagrammet för Java-klasser.
Diagrammet är delat på mitten och fortsätter i figur 17 på sida 18.



Figur 16: Första halvan i klassdiagrammet.

Klassdiagrammet för Java-klasser.
Diagrammet är delat på mitten och fortsätter från figur 16 på sida 17.



Figur 17: Andra halvan i klassdiagrammet.