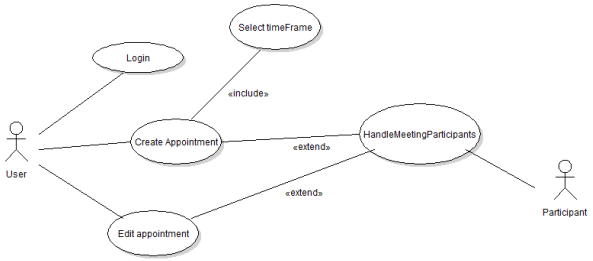
1 Use case



Figur 1: class diagram

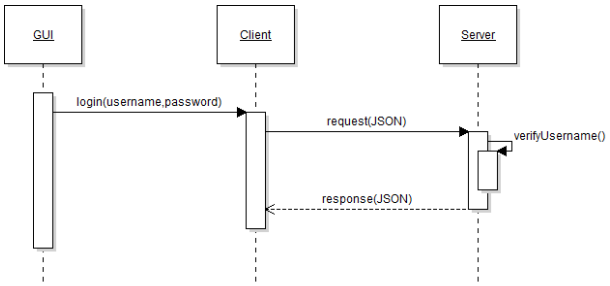
2 Text use case

Use case	Create appointment
Preconditions	Logged in
Steps	1. User activates create appointment 2. Title 3. From time (date and time) 4. To time (date and time) 5. Optional: Location 6. Optional: Description 7. Optional: Select room 8. Optional: Select participants 9. Optional: Select groups 10. Optional: Alarm 11. Commit
Expected result	1. The appointment is created and shown in the calendar. 2. All participants invited gets notified.
Possible errors	Users input a invalid value in from or to time. Reset from time and to time.

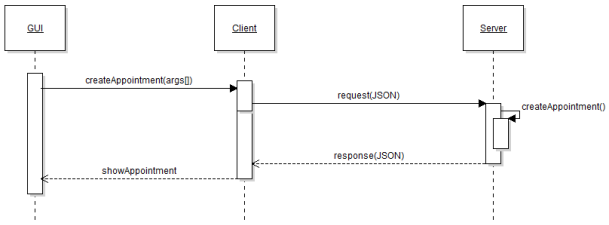
Use case	Manage meeting participants
Preconditions	Logged in and appointment selected
Steps	1. User activates create appointment 2. Title 3. From time (date and time) 4. To time (date and time) 5. Optional: Location 6. Optional: Description 7. Optional: Select room 8. Optional: Select participants 9. Optional: Select groups 10. Optional: Alarm 11. Commit
Expected result	1. The appointment is created and shown in the calendar. 2. All participants invited gets notified.
Possible errors	Users input a invalid value in from or to time. Reset from time and to time.

Use case	Edit appointment
Preconditions	Logged in and an appointment the user has created is selected
Steps	1. User edits some of the fields in the appointment. (E.g. from time, end time, title, location, description, room) 2. User clicks Commit
Expected result	1. The appointment is changed as expected. 2. The invited users get notified and can choose to accept or decline.
Possible errors	Users input a invalid value in from or to time. »Reset from time and to time. Users changes timeperiod, but the meeting room is occupied. » Automatically assign a new meeting room for the user.

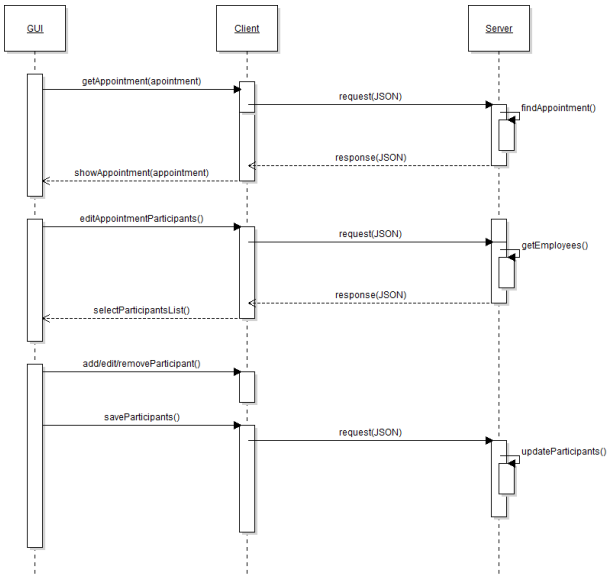
3 Sequence diagram



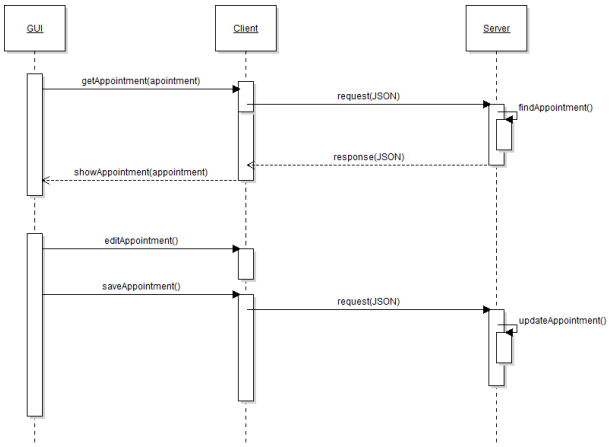
Figur 2: class diagram



Figur 3: class diagram



Figur 4: class diagram



Figur 5: class diagram

4 System structure

The system will be implemented in a server/client configuration where each user need their own client on their own computer and the server is hosted on its own computer. The server-side is the only piece of the software that interacts with the database. All the information from every client is saved in the database. Between the client and the server, the information is sent formatted as JSON. The frontend will be written in Java Swing.

4.0.1 User

The User class is one of the more important classes in the system, it defines the way our users are handled. Every user will have a username, password, name and email. They also have a list of alarms that is related to the user.

4.0.2 Appointment

The Appointment class is where we implement how every appointment is handled. An appointment has a id, title, description, timeframe, owner, participants and room/location. The room/location is mainly based on the need for a meeting room.

4.0.3 MeetingRoom

The MeetingRoom class is where we handle the meeting rooms. It is a basic class which only holds information about id, room, capacity and reservations.

4.0.4 Group

Group is a class what collects users in groups, it only contains an id, groupname and a list of users.

4.0.5 Alarm

Alarm is a little class that holds the date and time information about when a alarm should be executed.

4.0.6 TimeFrame

TimeFrame is a class that holds the duration of time. It has a startdate and an enddate, it can provide the duration of itself. It is mainly used to set time for appointments and reserve meeting rooms.