**Security analysis – Sqills Group 36**

During this project we valued the security of the app very high. Although the app will mostly be used in house, the schedule might still contain some private data about an important meeting. That's why we took the following measures to prevent any unwanted activity.

Since we rely on SQL during this project, one of the dangers that occurs is SQL injection. The app contains a couple of fields where user input is required, for example when making a reservation. To prevent a malicious user from abusing this we used 'prepared statements'. One example of this technique can be found in the file: 'main/java/…/sqills/dao/ReservationDao.java'. On line 105 a prepared statement is used for inserting a new reservation into the system. With a prepared statement we are able to insert values later into the SQL query with the question marks, this makes it impossible for a malicious user to insert any dangerous code.

Another possible weak spot is cross-site scripting, or XSS. When a user gets the chance to input anything manually by himself, the danger of XSS will always be there. That's why we decided to limit the input fields where the user can type something himself. However in some places this is still necessary, like at the reservation form. Here a first name, last name, email-address, etc. need to be filled in. When a form is submitted all input is checked used 'regular expression', or regex. This checks for any irregular patterns that could be malicious.

Since there is a login feature on the administrator page, we need to worry about another vulnerability. A malicious user could use brute force, or a dictionary attack to crack a password and get access to the administrator functions. To prevent this we decided to hash the passwords, which makes it much harder to crack a password. The function we used for this purpose is PBKDF2, or password-based key derivation function. The advantage of this function is that it is purposefully slow, this is advantageous for us because someone who wants to infiltrate would need a lot more computation power to crack the password.

Overall we believe our web-app is secure. We have no issues that can be exploited. And although we made sure the whole app is secure, we believe that there won't be a lot of cases where someone will try to breach the security of the app. But of course it still is important to make sure the app is completely secure.