Part 0: We are planning to programing in Java. Some of the members use macOS and others use Windows. We have talked to Prof. Kaiser and she said she was fine with it. For API, we will use JDBC for communication with database, JavaMail for mailing, and potentially Sphinx4, a voice recognition API.

Please see below for the proposal.

Schedule++

Advanced Software Engineering Project Proposal

Team: ScoreOverflow

Cindy Le(xl2738), Henry Xing(hx2209), David Wan(dw2735), Zichuan Wang(zw2395)

Synopsis:

Schedule++ is a multi-platform desktop software that automates the scheduling of public facility usages (e.g., university classrooms). Administrators can input a list of events (e.g. class sessions) with their respective time and location requirements. The system will assign a room and time slot for each of them while optimizing for these constraints. Individual users can also view available rooms and book venues for private events on this platform. Additional features include specifying the equipments (e.g. projector) available in a classroom and prioritizing certain user groups (e.g. professors) for room booking. The software can also potentially add a voice control function for accessibility concerns. The system will be a smart and integrated tool for universities and other facility providers.

Target Users:

- Educational institutions (e.g., universities and high schools).
- Middle to large companies with more than several meeting rooms and people to use them.
- Public event space providers or sports stadiums with available courts.

User Categories:

- Normal users (e.g., students and normal employees): They can book individual events in available rooms
- High priority users (e.g., professors and managers): They can book events like the normal users. They also have priority booking over normal users, able to book rooms that were already taken by normal users.
- Program supervisors (e.g., member of the administration office): They are very similar to high priority users, but they can schedule events in batch.
- Administrators (e.g., Building maintenance personnel): They input the facility information including operating hours and available equipments. They can optionally shut down a facility for maintenance purposes.

Use Cases:

< 0 >: As an administrator, I want the ability to book a new room or change the availability of an existing room so that I can be sure the information in the system is update to date. I am satisfied if I can add a new room with its availability/facility information (e.g., projectors) to the system and that I can change the availability/facility information of a room.

< 1 >: As a normal user, I want the ability to book a room or cancel my booking. My conditions of satisfaction are I can search for rooms whose availability/facility information matches my request and I can cancel my booking. If my booked room is overridden by a high priority user, I want to get a notification about it, and get an alternative room that matched my preferences, or, if there are no rooms available that matched all my preferences, the best matched room. If there is a person who wants to book rooms that I have booked, I want to get a notification, and an option to accept, decline, or ignore.

< 2 >: As a high priority user, I want the ability to book a room with high priority or cancel my booking. I am satisfied if I can search for rooms whose availability and facility information matches my request, I can cancel my booking and I can book a room even if it has been booked by a normal user (under certain constraints).

< 3 >: As a program supervisor, I want the ability to schedule meeting places and venues in batch. I am satisfied if I can input a list of meeting requests with specified time/location/equipment/capacity preferences and schedule all of them without pairwise conflict and satisfy the most requirements.

< 4 >: (Optional) As a user with disability, I want the ability to book rooms with speech input. I am satisfied if I can get get information and book meeting places with voice control.

< 5 >: (Optional) As a user, I want the ability to see booked rooms and contact him/her to negotiate. I am satisfied if I can get contact information of the person who booked my desired room.

Workflows/ Usage Scenarios:

Case users:

1. Users sign up with UNI/employee ID
2. Select room size, priority, and other preferences (devices, etc.)
3. Get lists of rooms
    a. normal users can only see unbooked rooms
    b. high priority users can see both unbooked rooms and rooms booked by normal users
    c. Same user groups can see each others' booked rooms for negotiation
4. Select the room the user wants to book and confirm the time period and location

5. (Optional) Voice input for differently abled people
6. If a high priority user overwrites the room (book a room that a normal user has booked), it will automatically send an email to notify the normal user

Case Administrator:

1. Sign up with UNI/employee ID
2. Input new classroom information and available technologies
3. Change existing classroom information and available technologies as necessary

Case program supervisor:

1. Users sign up with UNI/employee ID
2. Import a list of events with associated room size, priority, and other preferences to request rooms
3. Get the room assignments for these events, automatically scheduled by the system, notify any normal users if a room is overwritten
4. (Optional) Voice input for differently abled people

Technologies

Development Framework: Java

Static Analysis tool: PMD and IDEs (Eclipse, IntelliJ)

Unit testing tool: JUnit

Build tool: Maven

Data Storage: MySQL, jOOQ/JDBC, (potentially) online storage platform: OCI

Continuous integration tool: Travis CI

API: JDBC, JavaMail, Sphinx4 (Optional)

Acceptance Testing

- Login
  - Input: Username and password.
  - Output: If the username and password is valid, return a success message and show the main page of the program.
  - (Special case) If the username and passport is not valid, present an error message and ask the user to enter again by returning to the original page.
- Search for rooms

    - Input: A list of requirements and preferences, including desired room capacity, equipments (e.g. projectors), time slots, building/location preferences and desired room name.
    - Output: A list of rooms that match the preference. At the end of the resulting list, there is a button that asks if a normal user wants to see other booked rooms for negotiations. For high priority users, we also show rooms that he/she can potentially override.
    - (Special case) No room that satisfies all requirements is available: show a list of rooms that satisfy some of user's conditions, ordered by the number of matching preferences in descending order.
- Book rooms
    - Input: Select the desired room (from a list of items returned from the search for rooms function).
    - Output: Try to modify the corresponding database entry. Display a message indicating whether successful or not.
    - (Special case) Two users booked at the same time: block the second user.
- Override bookings
    - Input: (1) High priority user selects the room, and (2) confirmation that she or he wants to override.
    - Output: (1) Displays a warning message and a confirmation dialog; and (2) displays success for high priority user, changes the booking information to the high priority user; searches the best room alternative for the normal user, and sends a notification to the normal user who previously had the room. Changes database accordingly.
    - (Special case) Two users override at the same time: Assign the room to whoever came first. Send failure message for the other user, display other rooms to book.
- Cancel bookings
    - Input: User selects a specific room that he/she has booked to be canceled.
    - Output: Show a confirmation that the booking is cancelled. Modify the database item and notify the user.
- Batch input
    - Input: User inputs a file (csv, txt) with the given format.
    - Output: Book all the rooms that satisfy the requirements if possible.
    - (Special case) If some of the room requirements cannot be satisfied, display a list of all the rooms that cannot be booked. Ask the user to select the alternative rooms one by one.
    - (Special case) If the file format is invalid, display an error message.
- Replace rooms for normal users
    - Input: The room preference entered previously.

    - Output: Select the best alternative room and send information to the normal user.
- (Optional) Speech input
    - Input: Voice.
    - Output: the corresponding feature. If the demand was unclear, ask the user to speak again.
- (Optional) Negotiation
    - Input: (1) the room booking preference entered previously, selecting the option at the end of the search; (2) selecting the room for negotiation; and (3) A user written short message for the current room holder.
    - Output: (1) a list of normal user booked rooms that satisfy the preference; (2) send a customizable email message to the current room booker.
- (Optional) Change room owner (after negotiation step)
    - Input: (1) Room owner accepts; (2) owner declines; and (3) owner ignores.
    - Output: (1) Confirmation for the new user, change the database; (2) decline message for the new user; and (3) nothing, the message goes away from the owner for this session.