Software Technology Engineering

Semester 1 - Class Y

Timetable Planning Tool
Project Report

Group1
Christian Foyer (315200)
Martin Rosendahl (315201)
Nina Wrona (315202)
Robert Barta (315242)
Kamil Fischbach (315273)

Supervisors:
Mona Wendel Andersen
Steffen Vissing Andersen

Date of completion: 16/12/2021

The number of characters in the main text: 47541

Table of Contents

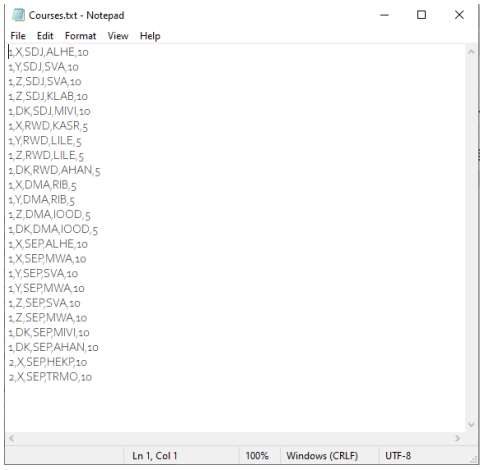
Abstract

The Timetable Planning Tool is a system that helps with creating a schedule for schools and Universities.The tool decreases the amount of time spent on inserting basic information about teachers, students and courses. Other objectives included avoiding overlapping the sessions, duplicating a test week, and editing course participants. The resulting timetable is displayed on the website.

The program was implemented in Java, while the website was built in HTML5, CSS3 and JavaScript. In the process, information is read from the XML file and sessions are added for the previously chosen class. The result is converted into an XML file. Although improvements can be made to the website for the end users, this project significantly decreases the difficulty of planning schedules from the current method.

1. *Introduction*

At VIA University College in Horsens, Bob oversees making the timetable for the courses software engineering students must attend. As this is one of the most populous areas of study at the school, it can be a very stressful task.



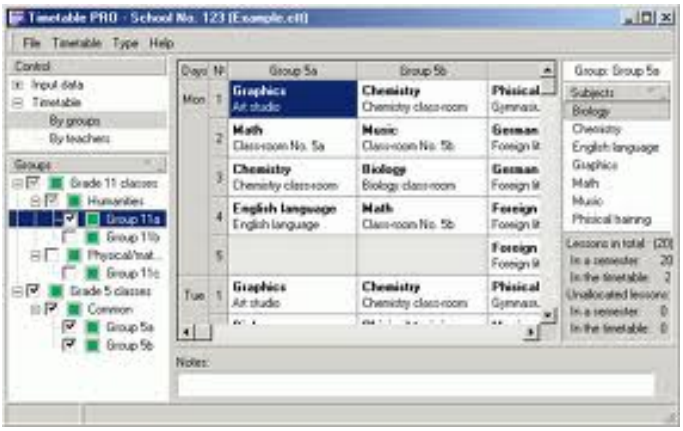
First, Bob receives a document from the head of the department, which includes classes with the students enrolled, teachers, courses, and the ECTS value of each course. Different types of students must be added or removed to individual courses based on the information in their transcripts. Then, he manually plans a timetable that collects and implements wishes sent by the teachers and combines them with the students' best interests.

Subsequently, he assigns courses to rooms but must check for overlaps by hand. Bob then sends out the test week he has created to be reviewed by the teachers. Once Bob has incorporated the suggestions of the teachers, he publishes the timetable as a spreadsheet on Studienet and removes the holidays. However, during the semester, updates must be made, which takes plenty of time.

In the current method, all teachers and students can view all of the timetables without logging in. With the intricacies of how the timetable is made now, it is not possible to find a replacement if Bob is not able to do it. Due to the shortfalls of the current method, Bob is not able to easily and efficiently create and publish a

timetable for both teachers and students to view. Finally, students can access individual timetables.

To add to the challenges that stand in the way, the university college just moved to a new campus in the center of town with limited rooms; each room is now precious. In a review conducted in 2014, a group of students at the Worcester Polytechnic Institute analyzed and optimized the scheduling of operating rooms to increase their usage and efficiency. They implemented a new scheduling software that used linear programming. Consequently, the time the operating rooms were in use increased. “According to our liaison at UMMMC, Dr. Tze Chiam, Ph.D. , the external benchmark for OR utilization is between 70 and 80%; UMMMC is currently operating at approximately 65% utilization for their operating rooms” (Carrol, Juers, and Kent, 2014). Similarly, the current method is underutilizing the limited rooms at the new campus.



There are current market[1] solutions to help with scheduling, but they are short of the ideal. Bob's schedule is special because some rooms can open into each other, and some courses have multiple teachers assigned. Coupled with the inability to account for those features of the timetable, many current tools available have outdated user interfaces that directly make using and reading the timetable

[1] Pictured: Timetable Pro from (Top 10 Free Scheduling Software For Schools And Colleges, 2021)

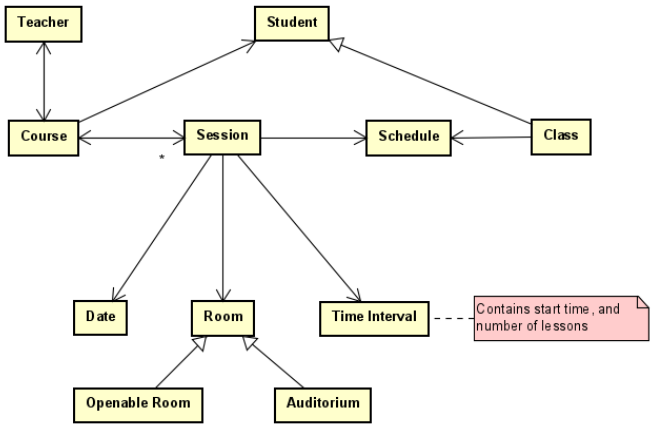
difficult (Top 10 Free Scheduling Software For Schools And Colleges, 2021). As these older user interfaces make the software difficult to use, the lack of flexibility in older software compounds the issues.

In a systematic review of timetables systems available for different higher education institutions, Vrielink and other researchers found that many timetable tools used today do not effectively cater to the flexibility needed for higher education institutions (Oude Vrielink, Jansen, Hans, and van Hillegersberg, 2017).

To summarize, Bob’s current method requires too much manual data entry and creates obstacles to an efficient timetable.

2. *Analysis*

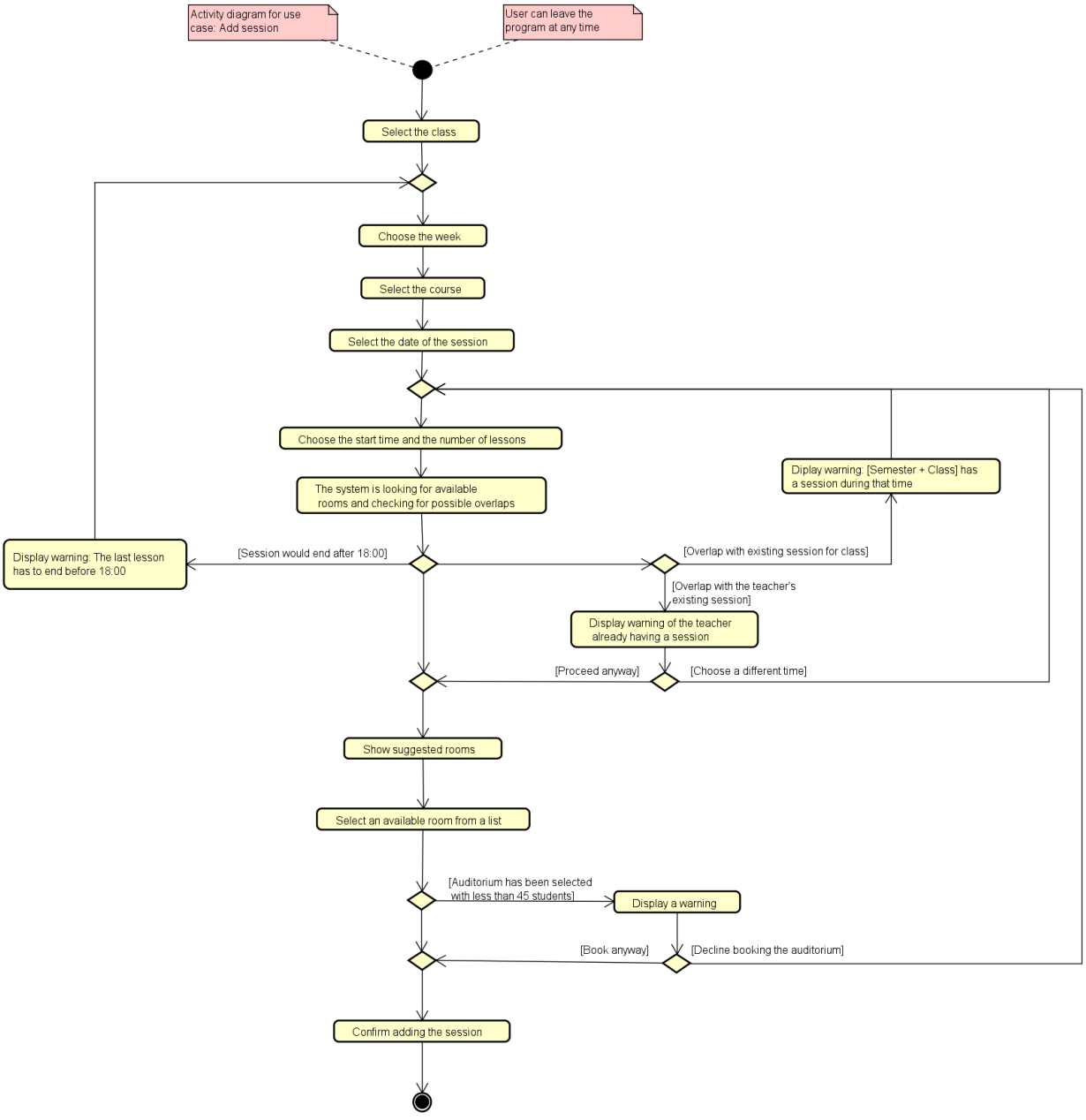
To deliver a piece of software, it must first be decided what to center and how to relate it to the other parts. Since a schedule of sessions is the end goal, it is the sessions that have the most associations in the following domain model.



Four very simple questions can be asked of each session: when will it take place, what will be taught, who will be in attendance, and where will it be held. First, we can assign a date using a simple date class and the length of the session is then determined by the number of lessons. Lessons at VIA have a standard 45 minutes of teaching and a 10-minute break, totaling 55 minutes per lesson. Accordingly, a session is a block of these 55-minute lessons. Second, we can link what will be taught and who will be in attendance as being properties of the course. Not only does the course have information about the subject matter, but it also has information about who needs to be assigned to teach and who is assigned to attend. This information is predetermined by the information from the head of the department but must also be flexible for change during the semester. Lastly, a session must either be assigned to a room or be scheduled for an online meeting. The list of rooms available to the department will also be predetermined by the head of the

department. However, keeping track of sessions and only allowing the planner to book sessions in rooms that are available is a critical piece of the software. In the implementation section, the methods to do this will be dissected and presented.

With the four basic questions asked, the sessions can be added piece by piece to the schedule. As the addition of sessions is so crucial, the basic structure of the activity is shown in the diagram below.


Activity diagram for use case: Add session
User can leave the program at any time
Select the class
Choose the week
Select the course
Select the date of the session
Choose the start time and the number of lessons
The system is looking for available rooms and checking for possible overlaps
Diplay warning: [Semester + Class] has a session during that time
Display warning: The last lesson has to end before 18:00
[Session would end after 18:00]
[Overlap with existing session for class]
[Overlap with the teacher's existing session]
Display warning of the teacher already having a session
[Proceed anyway]
[Choose a different time]
Show suggested rooms
Select an available room from a list
[Auditorium has been selected with less than 45 students]
Display a warning
[Book anyway]
[Decline booking the auditorium]
Confirm adding the session

Functional requirements

Critical priority:

1. As a planner, I want to create an individual timetable for every teacher and student, which they can find using their VIA ID so that they can see the schedule of their courses for the semester.
2. As a planner, I want to import the document from the head of the department so that I can use this information directly without manual entry.
3. As a planner, I want to add, remove and move sessions so that the requests made by the teachers are fulfilled.
4. As a planner, I want to book an available room for a session so that teachers have a room reserved to teach their courses.
5. As a planner, I want to be the only one to use the planning tool so that no one else can make changes.
6. As a planner, I want to create a test week that I can send to teachers so that they can review it and suggest changes before it is spread across the semester.
7. As a teacher, I would like to view my individual schedule so that I can view the courses I am going to teach.
8. As a student, I would like to view the timetable so that I know when to go to school to attend classes.
9. As a planner, I want to remove the holidays from the schedule so that the timetable accurately reflects the semester schedule made by VIA.

High priority:

10. As a planner, I want to manually select students so that I can add or remove students from the course.
11. As a planner, I want to assign multiple teachers to one course so that they can teach the course together and see it in their timetables.
12. As a planner, I want to be notified of gaps between lessons so that the time of a room that is booked for the lesson is as short as possible.
13. As a planner, I want to be able to reschedule a session of one class so that I can use a room with foldable walls to increase the capacity to 90 students if needed.

Low priority:

14. As a planner, I want to book the same room for each course of the same class so that first-semester students can be in the same room for their courses.
15. As a planner, I want to see the week numbers on the timetable so that I can navigate it using the week number, as is custom in Danish culture.
16. As a planner, I want to see a suggested room so that a room can be selected quicker with less input.
17. As a Student, I want to select the week that I am viewing so that I can see my schedule in advance.
18. As a planner, I want to be warned when booking the auditorium for less than 45 people so that it can remain available for larger groups.

19. As a planner, I want to have a planning tool displaying students enrolled in courses, ECTS points per course, and the teachers teaching each course so that I can compare the timetable with the information provided by the head of the department.
20. As a student or a teacher, I would like to be able to view the timetable in either English or Danish so that I can understand it better.

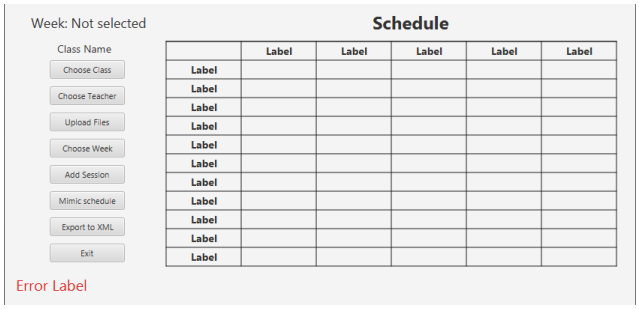
Non-functional requirements

21. The most recent changes to the timetable are published and visible to the users.

3. *Design*

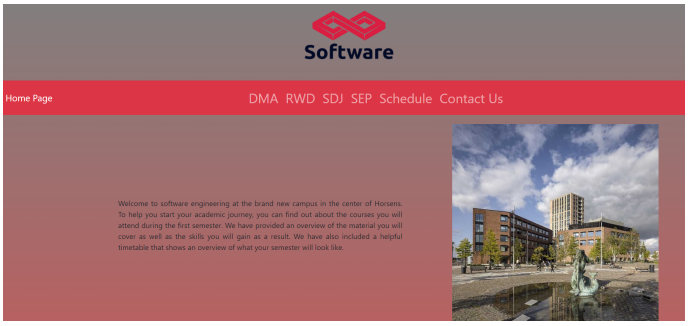
A basic pattern of software design can be described as a model-view-controller. The most basic classes are the ones represented in the domain model. Those basic classes have the blueprints for the objects like the rooms and students. Additionally, the model has the list classes that are collections of the above mentioned objects.

To create the controllers that manipulate the model, the JavaFX library was used to give the GUI functionality. 11 controllers in all were designed to accomplish the tasks needed for different scenes. Connections between windows were conceptualized by finding the buttons needed to accomplish the critical requirements but still have intuitive navigation. The windows with the least amount of buttons and text were designed first to establish a basic design through line to be followed for the rest of the design. The scenes in this case are the separate windows of the application that are opened according to the events that occur.

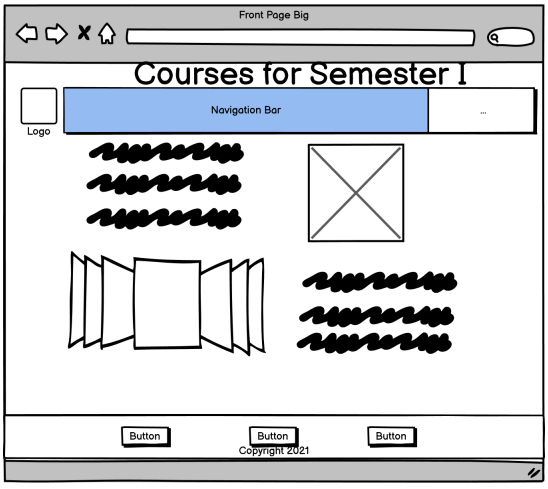


Although the user utilizes the classes from the model, they are instead directly interacting with the controllers and being shown information from the view package. Having a strong visual component to the GUI was critical for streamlining the complex task of scheduling sessions at VIA. With the visual component in hand, the methods that affect the features of the GUI were written. When the main program is executed, the

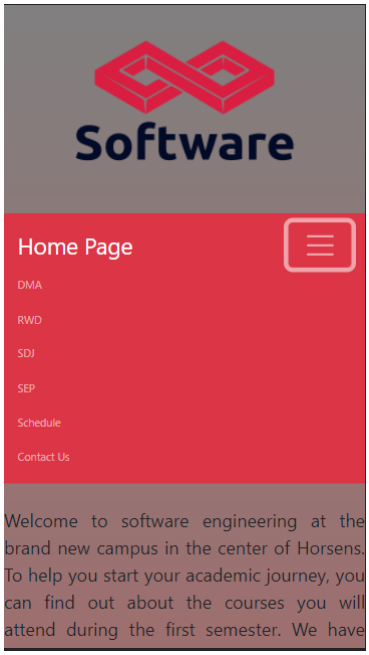
schedule window is opened and the grid is populated with sessions belonging to the class and week chosen. The session view model was designed to harvest properties from the session objects for the grid pane to display. As such, key information about a session for a user includes the course name, date, and room.



The website has a front page and pages for each of the four courses first semester students are assigned. It also has a schedule page and a contact us page. The four course pages give information on the courses and include pictures of the teacher. The contact us page has contact information for each of the creators of the website. The purpose of the schedule page is to allow the user to view a time table for a specific class or teacher.



The layout of the page was kept as simple as possible. The navigation bar is placed at the top and the relevant social media links are located in the footer. In the original wireframe pictured, there was a carousel of photos. However it unnecessarily cluttered the front page, so it was scrapped. Instead, the carousel was used in the various pages for the courses to describe the knowledge, skills, and competences.



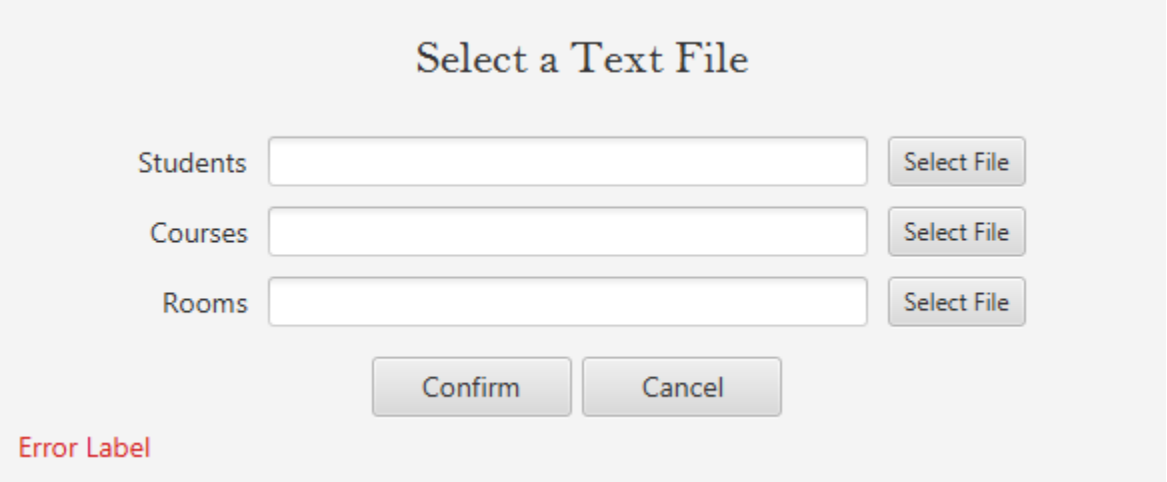
Because of the vertical orientation of most mobile devices, special considerations were made to change the layout for smaller view ports. In particular, the navigation bar collapses for small screens. The overarching objective was to remove the need for horizontal scrolling.

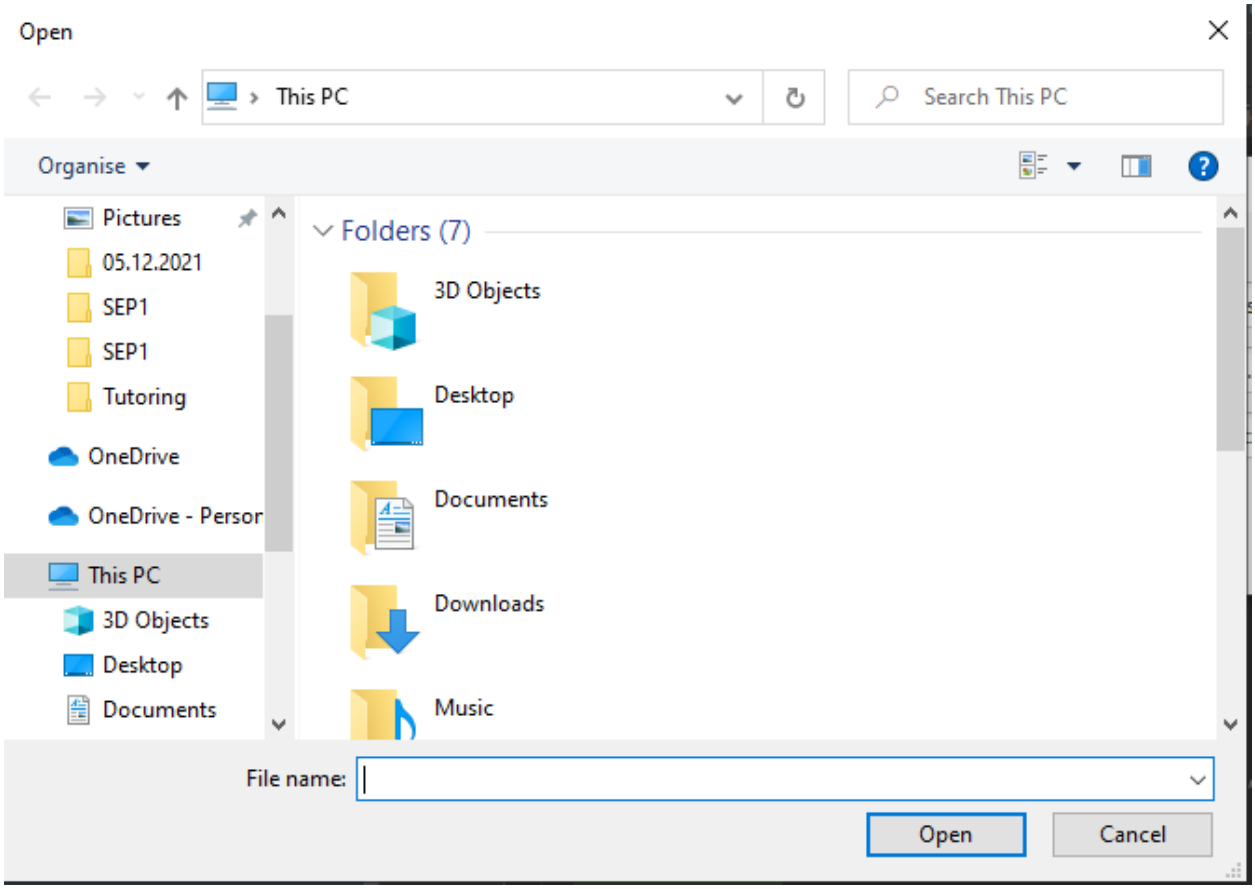
Mobile devices were a critical consideration for the schedule, because students in particular use mobile devices very frequently. In addition, a schedule is a piece of information that must be widely accessible at any time in any place.

4. *Implementation*

First of all, we started with implementing the basic classes such as Date, Time, Student, and Teacher. These basic classes were structured to hold much of the information needed in instance variables that would be assigned to the individual classes. For example, the Via ID and name of a student were held in an instance variable. Because of encapsulation, these instance variables could only be accessed using methods when outside of the class.

For the planner, the process of creating the schedule starts when the head of the department sends out a set of three text files for the following semester: students, courses, and rooms. In order for the program to take in these text files an upload window was created.



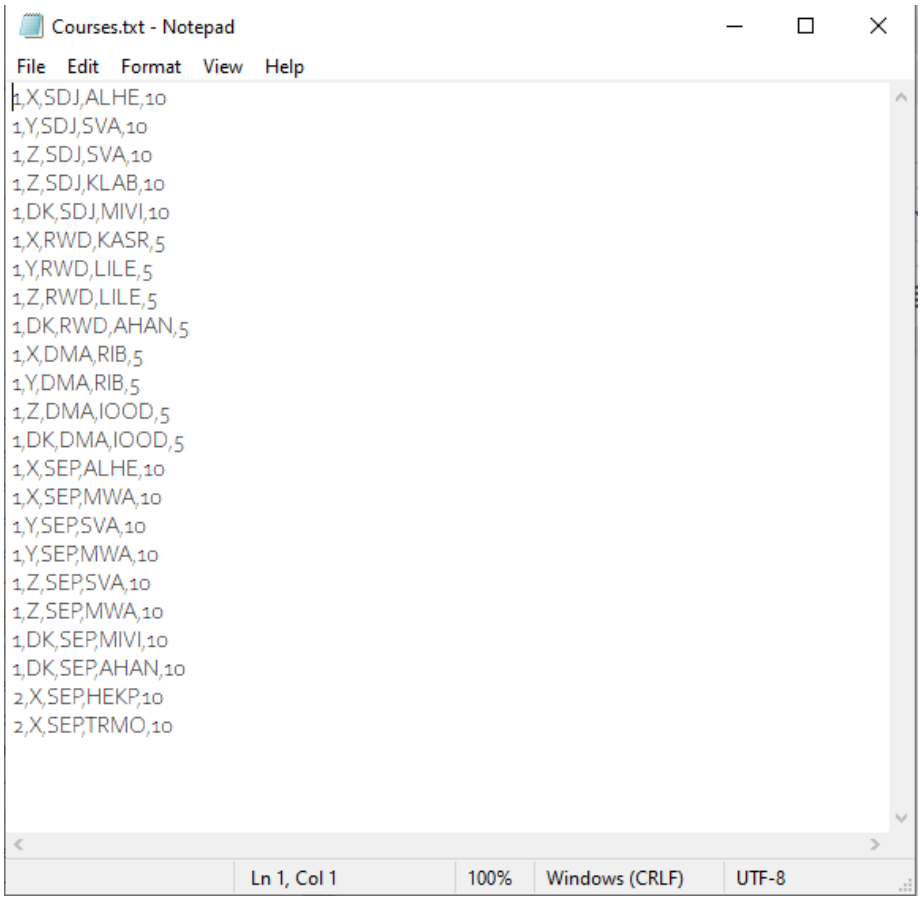


Using the FileChooser class from the JavaFX library, the inputted file gets fed into the program. From there, the text files are read using a manual parsing method that provides a group of lists to the model to be used when the planner is crafting the schedule. The student text file passes student objects into a predefined list of classes. The courses file creates different course objects and holds the information about teachers teaching as well as the assigned class. Lastly, the room text file creates a list of different rooms on campus as well as

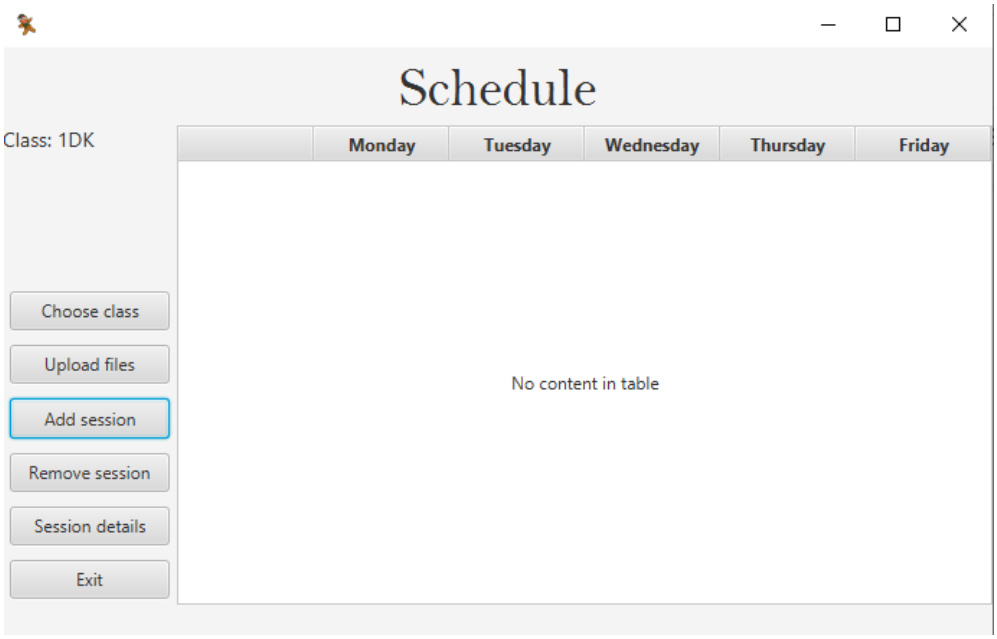
their capacity and ability to be opened into other rooms. Although the XML files are created later in the process, it is illustrative to show how the parser reads and organizes the lines into objects for later use.

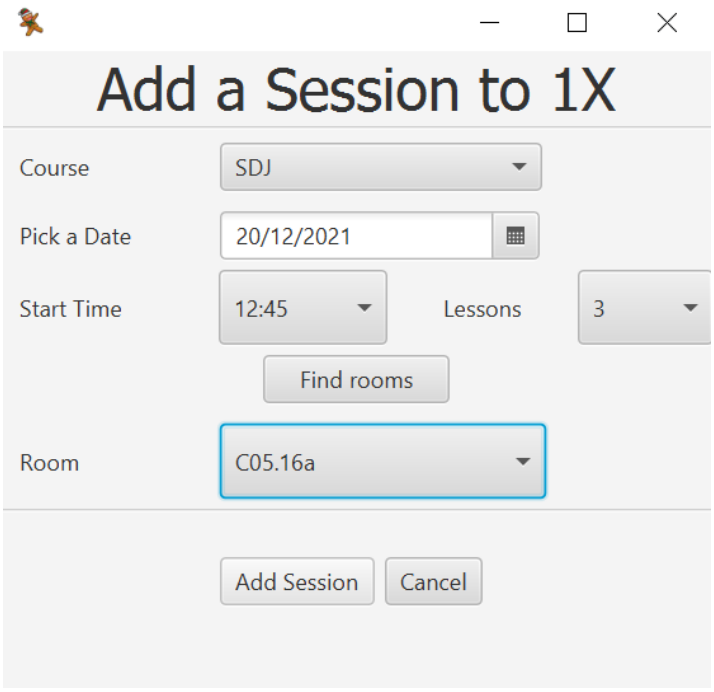
```
//Reads in a file of Courses and turns it into a CourseList
public static CourseList manualReadCourses(File file)
{

  CourseList courses = new CourseList();
  TeacherList masterTeacherList = new TeacherList();
  try
  {
    Scanner in = new Scanner(file);
    Course course;
    int semesterTaught = 0;
    ClassGroup classGroup;
    String courseName = null;
    int ECTS = 0;
    String[] parts;

    while (in.hasNext())
    {
      String line = in.nextLine();

      if (line.contains(","))
      {

        parts = line.split( regex: ",");
        if (parts.length == 5)
        {
          semesterTaught = Integer.parseInt(parts[0]);
```

The method shown above creates a course object with default or null values for the instance variables. Through the process of reading the lines of the text files, the instance variables are assigned accordingly.



With the files uploaded, the model has plenty of information about the sessions that need to be created to populate the schedule. After choosing the class for the schedule, sessions can be added using the add session window.





In order to create a session object, four options must be selected from choice boxes: the course, the date, the start time, and the number of lessons. From there, the available rooms can be found using the “find rooms button”. Not only does this button create the session object, but it also populates the drop-down menu for rooms by using the “suggest rooms” method. It is within this method that only available rooms are returned for that particular session.

```
@FXML private void findRoomsButton()
{
  roomsChoiceBox.getItems().removeAll(roomsArray);
  try
  {
    session = new Session(courseChoiceBoxInAddSession.getValue(),
        getDateFromDatePicker(), startTimeChoiceBox.getValue(),
        numberOfLessonsChoiceBox.getValue());
    System.out.println("I just created the following session:\n");
    System.out.println(session);
    loadRoomArray();
  }
  catch (Exception e)
  {
    errorLabel.setText(e.getMessage());
  }

}
```

```
public RoomList suggestRooms(Session session) {
    if (session == null) {
        throw new IllegalArgumentException("Parameter cannot be null!");
    }

    RoomList suggestedRoomList = new RoomList();

    for (int i = 0; i < roomList.size(); i++) {
        if (isRoomAvailable(roomList.get(i), session.getStartTime(),
                session.getNumberOfLessons(), session.getDate())) {
            if (roomList.get(i).getCapacity() >= session.getCourse().getClassGroup()
                    .getStudents().size()) {
                suggestedRoomList.addRoom(roomList.get(i));
            }
        }
    }
    if (suggestedRoomList.size() > 0) {
        return suggestedRoomList;
    }
    throw new NullPointerException(
            "There are no rooms found fulfilling given values!");
```

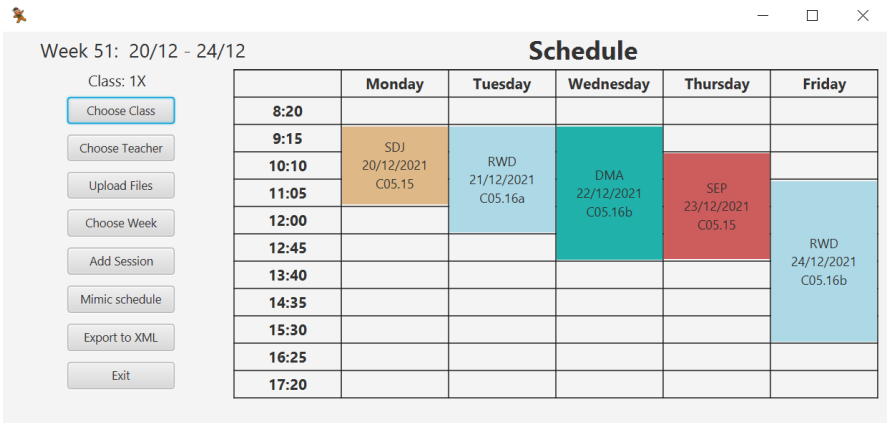
After the session is added to the model, the sessions are added to the schedule grid view model. This populates the grid in the main window of the planning tool.

```
for (int i = 0; i < scheduleViewModel.getList().size(); i++)
{
  if (scheduleViewModel.getList().get(i).getTeacher()
      .contains(model.getChosenTeacher().toString()))
  {
    StringProperty courseName = scheduleViewModel.getList().get(i)
        .getCourseProperty();
    Label labelTest = new Label();
    labelTest.setText(courseName.get());
    labelTest.setId("session" + i);

    // Adds a background color to the session on the grid
    String backColor = "lavender";
    String courseHolder =
        "" + scheduleViewModel.getList().get(i).getCourseProperty();
```

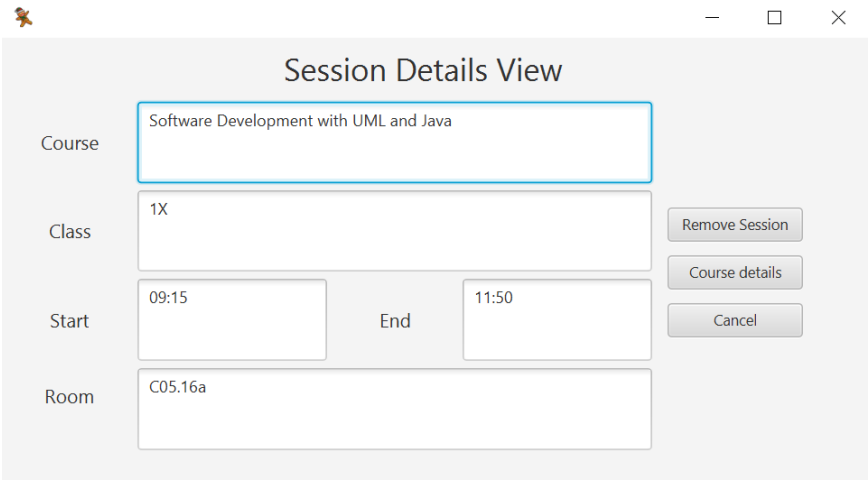
```
int startTimeInt = scheduleViewModel.getList().get(i)
    .getStartTimeIntProperty();
int numberOfLessonsInt = scheduleViewModel.getList().get(i)
    .getNumberOfLessonsProperty().intValue();
int dayOfWeek = scheduleViewModel.getList().get(i)
    .getDayOfWeekProperty().getValue();

labelTest.setMinHeight(
    (double) (numberOfLessonsInt * 21) + (4.6 * numberOfLessonsInt
        - 5));
labelTest.setTextOverrun(OverrunStyle.CLIP);
labelTest.setMinWidth(98.5);
labelTest.setTextAlignment(TextAlignment.CENTER);
labelTest.setAlignment(Pos.CENTER);

gridPane
    .add(labelTest, dayOfWeek, startTimeInt,  i2: 1, numberOfLessonsInt);
```

These different properties of a session are harvested within the session view model. Inside of the view model, different methods are applied to the model to determine certain properties that are needed to correctly display the session onto the table of the week.

```
public SessionViewModel(Session session)
{
  courseProperty = new SimpleStringProperty(session.shortString());

  LocalDate localDate = LocalDate
      .of(session.getDate().getYear(), session.getDate().getMonth(),
          session.getDate().getDay());

  // Set day of week as an int. 1 is Monday, 7 is Sunday
  teacherProperty = new SimpleStringProperty(
      session.getTeachers().toString());
  dayOfWeekProperty = new SimpleIntegerProperty(
      localDate.getDayOfWeek().getValue());
  startTimeProperty = new SimpleStringProperty(session.getStartTimeString());
  numberOfLessonsProperty = new SimpleIntegerProperty(
      session.getNumberOfLessons());
  fullStringProperty = new SimpleStringProperty(session.toString());
```



From the grid pane, individual sessions can be selected to view details and further move to the course details. The session can also be removed from the session details window. These functions are controlled in the session view controller. The methods called in the controllers are present in the schedule model, which is an interface. The functions are then implemented in the schedule model manager. In the case of removing the session, the function itself is in the session list.

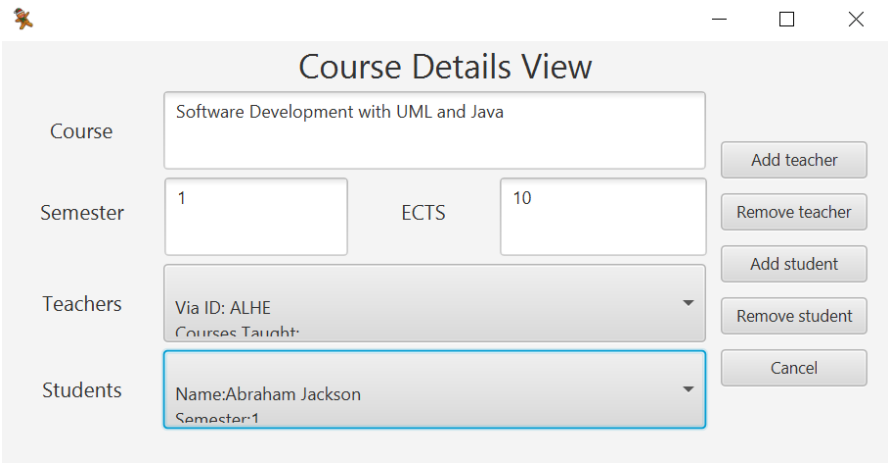



```
@FXML private void removeSessionButton()
{
  model.removeSession(chosenSession);
  viewHandler.openView( name: "schedule");
}
```

```java
public void removeSession(Session session) {
    sessionList.removeSession(session);
}
```

```java
public void removeSession(Session session) {
    if (size() == 0) {
        throw new NullPointerException(
                "The list is empty! You cannot remove anything!");
    }
    if (session == null) {
        throw new IllegalArgumentException("Session cannot be null!");
    }

    sessions.remove(session);
}
```

From inside the session details view, the course details window is where the planner can add and remove participants: both teachers and students. Using the selection from the choice box, the participant is found in the model and then removed. Because the choicebox is reset each time the window is opened, the choice is instantly reflected.

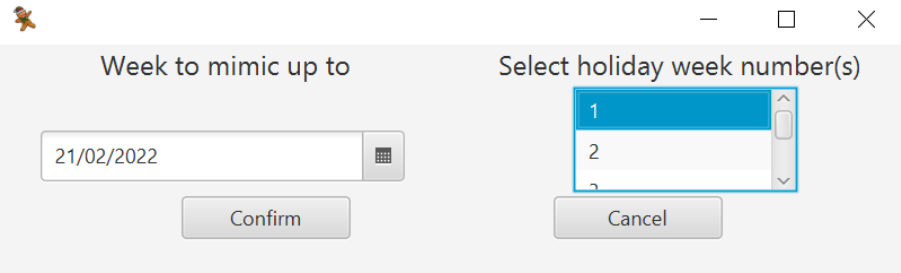


```
@FXML
private void removeStudentButton() {
    try {
        model.getChosenSession().getCourse().removeStudent(studentChoice.getValue());
        reset();
    } catch (IllegalArgumentException e) {
        errorLabel.setText(e.getMessage());
    }
```

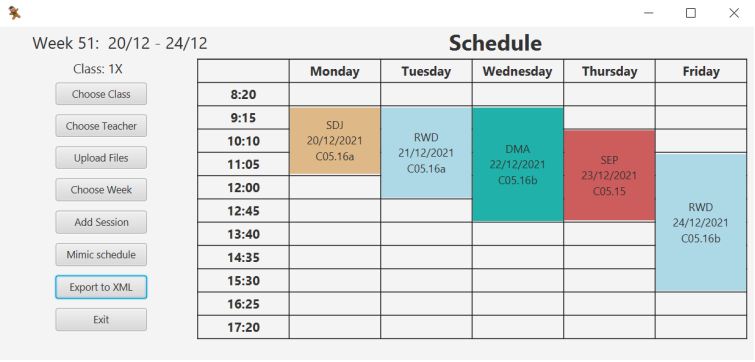
When the first week is created the planner can then mimic that week through whatever week they want. Holiday weeks can be selected in the same window. Because it was implemented using a list view from JavaFX, multiple weeks can be selected. The selected weeks are then placed in a holder variable in the model.

```java
@FXML private void confirmButton()
{
  // Do the holidays here
  for (int i = 0;
       i < holidayPicker.getSelectionModel().getSelectedIndices().size(); i++)
  {
    weekHoliday.add(Integer.parseInt(
        holidayPicker.getSelectionModel().getSelectedIndices().get(i)
            .toString()) + 1);
  }
  System.out.println(weekHoliday);
  model.setHolidayWeeks(weekHoliday);

  // Do the mimicking here
  LocalDate date = datePicker.getValue();
  chosenWeekNumber = date.get(ChronoField.ALIGNED_WEEK_OF_YEAR);
  LocalDate mondayHolder = date;
  while (mondayHolder.getDayOfWeek().getValue() != 1)
  {
    System.out.println(mondayHolder);
    mondayHolder = mondayHolder.minusDays(1);
    System.out.println("I just moved back one day");
    System.out.println("Holder is now " + mondayHolder);
  }
  // Monday Holder is now the monday of the week that is targeted to end the mimic
  Date monday = new Date(mondayHolder.getDayOfMonth(),
      mondayHolder.getMonthValue(), mondayHolder.getYear());
  System.out.println("I just set the Monday to " + monday);
  Date endMonday = monday.copy();
  endMonday.stepForward( days: 7);
  Date startMonday = model.getChosenMonday().copy();
  startMonday.stepForward( days: 7);
  Date dateHolder = model.getChosenMonday().copy();
```

```
while (!startMonday.equals(endMonday))
{
  boolean isHoliday = false;
  if (startMonday.getWeekday().equals("MONDAY"))
  {
    for (int j = 0; j < weekHoliday.size(); j++)
    {
      if (startMonday.getWeekNumber() == weekHoliday.get(j))
      {
        startMonday.stepForward( days: 7);
        isHoliday = true;
        break;
      }
    }
  }
  if (!isHoliday)
  {
    SessionList sessionListForDay = model.getSessionsByDateAndClassGroup(
        dateHolder, model.getChosenClassGroup());
    for (int i = 0; i < sessionListForDay.size(); i++)
    {
      Session sessionCopy = sessionListForDay.get(i)
          .copySessionToDate(startMonday);
      model.addSession(sessionCopy, sessionCopy.getRoom());
    dateHolder.stepForwardOneDay();
    startMonday.stepForwardOneDay();
```



The mimicking occurs by keeping track of a few dates. First, the sessions from the first day of the origin week are copied to the first day of the target week. After doing the same for the other days in the week, the origin date moves back to the first Monday, whereas the target progresses to the next Monday. This mimicking occurs until the Friday of the target date chosen is reached. The mimicking does not occur during chosen holiday weeks. With the sessions mimicked, the planner can then export the schedule into an XML so the website can easily read and display the session objects.




```
<SessionList>
<Session>
    <Course>SDJ</Course>
    <Class>1X</Class>
    <Teachers>ALHE,</Teachers>
    <Date>20/12/2021</Date>
    <WeekDay>MONDAY</WeekDay>
    <StartTime>9:15</StartTime>
    <NumberOfLessons>3</NumberOfLessons>
    <EndTime>11:50</EndTime>
    <Room>C05.16a</Room>
</Session>
<Session>
    <Course>RWD</Course>
    <Class>1X</Class>
    <Teachers>KASR,</Teachers>
    <Date>21/12/2021</Date>
    <WeekDay>TUESDAY</WeekDay>
    <StartTime>9:15</StartTime>
    <NumberOfLessons>4</NumberOfLessons>
    <EndTime>13:30</EndTime>
    <Room>C05.16a</Room>
</Session>
```

For the website to display the session objects, the XML file is then read on the schedule.html. Each row in the table on the schedule in the website is treated as an array. These arrays are then collected into a larger array called timelist. Placing each session only requires a brisk jaunt through 2000 lines of JavaScript.

The website first loads the empty table without sessions. The row on the top is displaying the name of the weekdays whereas the column contains start hours from 8:20 until 17:20.

```
<!--8:20, 9:15, 10:10, 11:05, 12:00, 12:45, 13:40, 14:35, 15:30, 16:25 and 17:20.-->
<div class="container-fluid w-100 m-2">
    <table class="table d-xs-table text-center ">
        <tr>
            <th class="table-secondary w-table">Time </th>
            <th class="table-secondary ">Monday</th>
            <th class="table-secondary ">Tuesday</th>
            <th class="table-secondary ">Wednesday</th>
            <th class="table-secondary ">Thursday</th>
            <th class="table-secondary ">Friday</th>
        </tr>
        <tr id="8:20">
            <td class="table-secondary w-table"><b>08:20-09:05</b></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
        </tr>
        <tr id="9:15">
            <td class="table-secondary w-table"><b>09:15-10:00</b></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
        </tr>
        <tr id="10:10">
            <td class="table-secondary w-table"><b>10:10-11:00</b></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
            <td class="table-light"></td>
        </tr>
        <tr id="11:05">
            <td class="table-secondary w-table"><b>11:05-11:50</b></td>
```

The screenshot below shows how it is displayed on the website. Using the class “container-fluid w-100 m-2”, the element covers the whole width of the website while also having margins on the right and left. This functionality is possible by the usage of Bootstrap 5: a framework for building websites.

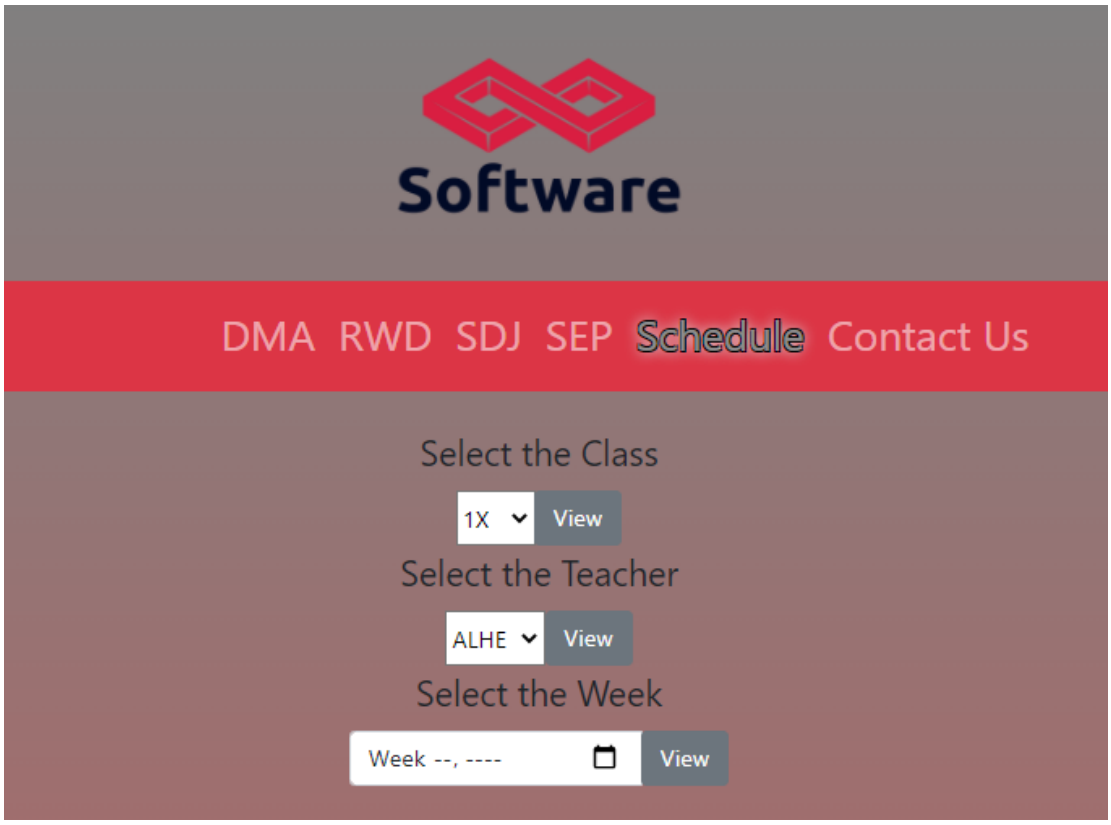
Time	Monday	Tuesday	Wednesday	Thursday	Friday
08:20-09:05					
09:15-10:00					
10:10-11:00					
11:05-11:50					
12:00-12:45					
12:45-13:30					
13:40-14:25					
14:35-15:20					
15:30-16:15					
16:25-17:10					
17:20-18:05					

Next, the dropboxes were added to select the class or teacher. In addition, a date picker was added to choose the week of the schedule. The chosen week needs improvements to be more dynamic to the sessions that should be displayed. However, the pickers for both class and teacher have proven functionality. The class and teacher can be chosen only once as when done more, the table is being overwritten. This requires the user to refresh the page.

```
<div class="container  justify-content-center" >
<div class="col-xs-3 d-flex justify-content-center">
<form class=form>
    <legend>Select the Teacher</legend>
        <div class="input-group d-flex justify-content-center">
            <select id="selectedTeacher" name="teachers" >
                <option value="ALHE" selected>ALHE</option>
                <option value="SVA">SVA</option>
                <option value="MIVI">MIVI</option>
                <option value="LILE">LILE</option>
                <option value="AHAN">AHAN</option>
                <option value="RIB">RIB</option>
                <option value="IOOD">IOOD</option>
                <option value="MWA">MWA</option>
                <option value="HEKP">HEKP</option>
                <option value="TRMO">TRMO</option>
            </select>
        </div>
      <span class="input-group-btn">
        <button id="teacherBtn" class="btn btn-secondary" type="button">View</button>
      </span>
</form>
</div>
</div>
```

```
<div class=" container d-flex justify-content-center " >
    <div class="col-xs-3 d-flex justify-content-center">
     <form class= form >
        <legend>Select the Class</legend>
        <div class="input-group d-flex justify-content-center">

        <select id="selectedClass" name="classGroups" >
            <option value="1X">1X</option>
            <option value="1Y" >1Y</option>
            <option value="1Z" >1Z</option>
            <option value="1DK">1DK</option>
            <option value="2X">2X</option>
            <option value="2Y" >2Y</option>
            <option value="2Z" >2Z</option>
            <option value="2DK">2DK</option>
            <option value="3X">3X</option>
            <option value="3Y" >3Y</option>
            <option value="3Z" >3Z</option>
            <option value="3DK">3DK</option>
            <option value="4X">4X</option>
            <option value="4Y" >4Y</option>
            <option value="4Z" >4Z</option>
            <option value="4DK">4DK</option>
        </select>
         <span class="input-group-btn">
           <button id="classBtn" type="button" class="btn btn-secondary">View</button>
         </span>
       </div>
     </form>
```

The “d-flex justify-content-center” Bootstrap 5 class for selection menus aligns the content to the horizontal center of the page. The result is shown below.



After the button is clicked the value of either chosen class or teacher is read and assigned to the variable in the JavaScript code. In order to achieve it, jQuery was used.
Subsequently, either the readXMLSessionListClass() or the readXMLSessionListTeacher() function is called.

```
$("#classBtn").click(function () {
    chosenClass = $("#selectedClass").val();
    readXMLSessionListClass();
});
$("#teacherBtn").click(function () {
    chosenTeacher = $("#selectedTeacher").val();
    readXMLSessionListTeacher();
});
```

Those two methods both have a switch case depending on the chosen value. The accurate XML file is opened and the showData() function is called.

```
function readXMLSessionListClass() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function () {
        if (this.readyState == 4 && this.status == 200) {
            showData(xhttp);
        }
    }
    if (chosenClass != null) {
        switch (chosenClass) {
            case ("1X"): xhttp.open("GET", "../../SessionList1X.xml", true);
                break;
            case ("1Y"): xhttp.open("GET", "../../SessionList1Y.xml", true);
                break;
            case ("1Z"): xhttp.open("GET", "../../SessionList1Z.xml", true);
                break;
            case ("1DK"): xhttp.open("GET", "../../SessionList1DK.xml", true);
                break;
            case ("2X"): xhttp.open("GET", "../../SessionList2X.xml", true);
                break;
            case ("2Y"): xhttp.open("GET", "../../SessionList2Y.xml", true);
                break;
            case ("2Z"): xhttp.open("GET", "../../SessionList2Z.xml", true);
                break;
            case ("2DK"): xhttp.open("GET", "../../SessionList2DK.xml", true);
                break;
            case ("3X"): xhttp.open("GET", "../../SessionList3X.xml", true);
                break;
            case ("3Y"): xhttp.open("GET", "../../SessionList3Y.xml", true);
                break;
            case ("3Z"): xhttp.open("GET", "../../SessionList3Z.xml", true);
                break;
            case ("3DK"): xhttp.open("GET", "../../SessionList3DK.xml", true);
                break;
            case ("4X"): xhttp.open("GET", "../../SessionList4X.xml", true);
                break;
```

```
function readXMLSessionListTeacher() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function () {
        if (this.readyState == 4 && this.status == 200) {
            showData(xhttp);
        }
    };
    if (chosenTeacher != null) {
        switch (chosenTeacher) {
            case ("ALHE"): xhttp.open("GET", "../../SessionListALHE.xml", true);
                break;
            case ("SVA"): xhttp.open("GET", "../../SessionListSVA.xml", true);
                break;
            case ("KLAB"): xhttp.open("GET", "../../SessionListKLAB.xml", true);
                break;
            case ("MIVI"): xhttp.open("GET", "../../SessionListMIVI.xml", true);
                break;
            case ("KASR"): xhttp.open("GET", "../../SessionListKASR.xml", true);
                break;
            case ("LILE"): xhttp.open("GET", "../../SessionListLILE.xml", true);
                break;
            case ("AHAN"): xhttp.open("GET", "../../SessionListAHAN.xml", true);
                break;
            case ("RIB"): xhttp.open("GET", "../../SessionListRIB.xml", true);
                break;
            case ("IOOD"): xhttp.open("GET", "../../SessionListIOOD.xml", true);
                break;
            case ("MWA"): xhttp.open("GET", "../../SessionListMWA.xml", true);
                break;
            case ("HEKP"): xhttp.open("GET", "../../SessionListHEKP.xml", true);
                break;
            case ("TRMO"): xhttp.open("GET", "../../SessionListTRMO.xml", true);
                break;

        }
    }
    xhttp.send();
}
```

The program loops through the FXML file by the tag name “Session”. Later on, showData() function correctly gets the values for all the sessions using the following JavaScript syntax on the below screenshot.

```
function showData(xml) {
    var xmlDoc = xml.responseXML;
    var x = xmlDoc.getElementsByTagName("Session");
    var listLength = x.length;


    for (var p = 0; p < listLength; p++) {

        var course = x[p].getElementsByTagName("Course")[0].childNodes[0].nodeValue;
        var room = x[p].getElementsByTagName("Room")[0].childNodes[0].nodeValue;
        var teachers = x[p].getElementsByTagName("Teachers")[0].childNodes[0].nodeValue.split(",");
        var numberOfLessonsString = x[p].getElementsByTagName("NumberOfLessons")[0].childNodes[0].nodeValue;
        var numberOfLessons = parseInt(numberOfLessonsString);
        var startTime = x[p].getElementsByTagName("StartTime")[0].childNodes[0].nodeValue;
        var weekDay = x[p].getElementsByTagName("WeekDay")[0].childNodes[0].nodeValue;
```

Consequently, those variables are used to assign them to empty “td” tags inside HTML using “.innerHTML”. It is possible to access the table data due to the below variables. Each variable represents one row with 6 “td” tag elements inside. Specifically, “hour820[a]” represents the table data for a lesson that starts on Monday 8:20.

```
//All the variables rows with our. We are gonna get different days by->
//hour820[1] = 8:20 Monday and hour820[5] = 8:20 Friday.
var hour820 = document.getElementById("8:20").getElementsByTagName("td");
var hour915 = document.getElementById("9:15").getElementsByTagName("td");
var hour1010 = document.getElementById("10:10").getElementsByTagName("td");
var hour1105 = document.getElementById("11:05").getElementsByTagName("td");
var hour1200 = document.getElementById("12:00").getElementsByTagName("td");
var hour1245 = document.getElementById("12:45").getElementsByTagName("td");
var hour1340 = document.getElementById("13:40").getElementsByTagName("td");
var hour1435 = document.getElementById("14:35").getElementsByTagName("td");
var hour1530 = document.getElementById("15:30").getElementsByTagName("td");
var hour1625 = document.getElementById("16:25").getElementsByTagName("td");
var hour1720 = document.getElementById("17:20").getElementsByTagName("td");
```

This is how the data from the XML is parsed inside the table.

```
switch (startTime) {
    case ("8:20"):
        {
            switch (weekDay) {
                case ("MONDAY"):
                    {
                        hour820[a].innerHTML = course + "<br>" + room + "<br>";
                        for (var j = 0; j < teachers.length; j++) {
                            hour820[a].innerHTML += teachers[j] + "  ";
                        }
                        switch (course) {
                            case ("SDJ"):
                                {
                                    hour820[a].classList.add("bg-danger");
                                }
                                break;
                            case ("DMA"):
                                {
                                    hour820[a].classList.add("bg-info");
                                }
                                break;
                            case ("RWD"):
                                {
                                    hour820[a].classList.add("bg-primary");
                                }
                                break;
                            case ("SEP"):
                                {
                                    hour820[a].classList.add("bg-warning");
                                }
                                break;
                        }
                        hour820[a].classList.add("pt-" + numberOfLessons);
                        hour820[a].rowSpan = "" + numberOfLessons;

                        for (var k = numberOfLessons - 1; k >= 1; k--) {
                            timeList[k][a].remove();
```

This is how the session details are displayed on the website. However, sometimes the blank spots outside of the table occurred due to the usage of rowSpan to show the actual length of the sessions. Although there were multiple attempts to fix them, it did not work out. Below there is an example of a table both for a class and a teacher.

Time	Monday	Tuesday	Wednesday	Thursday	Friday
08:20-09:05	SDJ C05.15 ALHE				
09:15-10:00		SEP C05.15 ALHE			
10:10-11:00			RWD C05.15 ALHE		
11:05-11:50					
12:00-12:45					
12:45-13:30					
13:40-14:25					
14:35-15:20					
15:30-16:15					
16:25-17:10					
17:20-18:05					

Time	Monday	Tuesday	Wednesday	Thursday	Friday
08:20-09:05				RWD C05.15 LILE	RWD C05.15 LILE
09:15-10:00					
10:10-11:00					
11:05-11:50					
12:00-12:45					
12:45-13:30					
13:40-14:25					
14:35-15:20					
15:30-16:15					
16:25-17:10					
17:20-18:05					

Here is the approach for removing the “td” tags in JavaScript.

```
for (var k = numberOfLessons - 1; k >= 1; k--) {
    timeList[k][a].remove();

}
```

Where the “timeList” variable is:

```
var timeList = [hour820, hour915, hour1010, hour1105, hour1200, hour1245, hour1340, hour1435,
    hour1530, hour1625, hour1720];
```

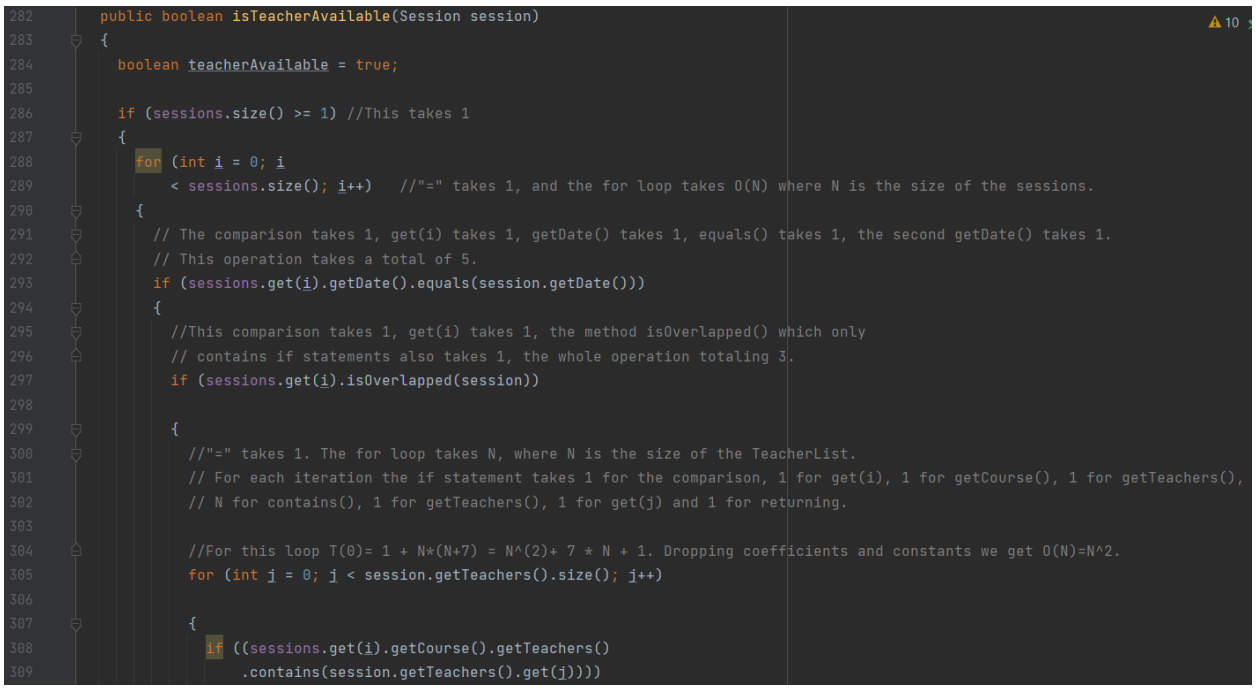
Considering the implemented java methods, with so many classes being utilized, it is important to analyze and test individual pieces to ensure their functionality. We chose two methods to analyze based on their time complexities: isRoomAvailable(), and isTeacherAvailable(). Both of these methods contain two for loops and call several different methods. The commented code snippets can be viewed here:

```
205     public boolean isRoomAvailable(Room room, Time timeStart, int numberOfLessons,
206         Date date)
207     {
208       SessionList list = new SessionList(); //Creation of a SessionList object takes 1.
209       if (sessions.size() != 0) //Comparison takes 1, size() takes 1.
210       {
211         // "=" takes 1, size() takes 1 and the for loop takes O(N).
212         //The whole loop takes 2 + (N*9)
213         //Dropping constants and coefficients, the time complexity is O(N).
214         for (int i = 0; i < sessions.size(); i++)
215
216         {
217           // The comparison takes 1, get(i) takes 1, getRoom() takes 1,
218           // get(i) takes 1, getDate() takes 1, equals(date) takes 1.
219           if (sessions.get(i).getRoom().equals(room) && sessions.get(i).getDate()
220               .equals(date))
221           {
222             // addSession() takes 1, get(i) takes 1, getRoom() takes 1.
223             list.addSession(sessions.get(i), sessions.get(i).getRoom());
224           }
225         }
226         //"=" takes 1, size() takes 1 and the for loop takes O(N).
227         //The whole loop takes 2 + (N*3) + 1.
228         //Dropping constants and coefficients, the time complexity is O(N).
229         for (int i = 0; i < list.size(); i++)
230
231         {
```

isRoomAvailable() 1/2 with O(N) time complexity

```
226         //"=" takes 1, size() takes 1 and the for loop takes O(N).
227         //The whole loop takes 2 + (N*3) + 1.
228         //Dropping constants and coefficients, the time complexity is O(N).
229         for (int i = 0; i < list.size(); i++)
230 
231         {
232             // the comparison takes 1, get(i) takes 1,
233             // isOverlapped takes 1 because it only contains if statements.
234             if (list.get(i).isOverlapped(timeStart, numberOfLessons))
235             {
236                 // takes 1
237                 return false;
238             }
239         }
240     }
241     return true;
242 }
243 // We chose this method because it contains two for loops and calls different methods.
244 // T(O)= 3 + N + N = 2*N + 3.
245 // Dropping constants and coefficients the time complexity of this method id O(N).
246 
```

isRoomAvailable() 2/2 with O(N) time complexity



```
282   public boolean isTeacherAvailable(Session session)                                    ⚠ 10
283   {
284     boolean teacherAvailable = true;
285
286     if (sessions.size() >= 1) //This takes 1
287     {
288       for (int i = 0; i
289           < sessions.size(); i++)   //"=" takes 1, and the for loop takes O(N) where N is the size of the sessions.
290       {
291         // The comparison takes 1, get(i) takes 1, getDate() takes 1, equals() takes 1, the second getDate() takes 1.
292         // This operation takes a total of 5.
293         if (sessions.get(i).getDate().equals(session.getDate()))
294         {
295           //This comparison takes 1, get(i) takes 1, the method isOverlapped() which only
296           // contains if statements also takes 1, the whole operation totaling 3.
297           if (sessions.get(i).isOverlapped(session))
298
299           {
300             //"=" takes 1. The for loop takes N, where N is the size of the TeacherList.
301             // For each iteration the if statement takes 1 for the comparison, 1 for get(i), 1 for getCourse(), 1 for getTeachers(),
302             // N for contains(), 1 for getTeachers(), 1 for get(j) and 1 for returning.
303
304             //For this loop T(0)= 1 + N*(N+7) = N^(2)+ 7 * N + 1. Dropping coefficients and constants we get O(N)=N^2.
305             for (int j = 0; j < session.getTeachers().size(); j++)
306
307             {
308               if ((sessions.get(i).getCourse().getTeachers()
309                   .contains(session.getTeachers().get(j))))
```

isTeacherAvailable() 1/2 with O(N^3) time complexity

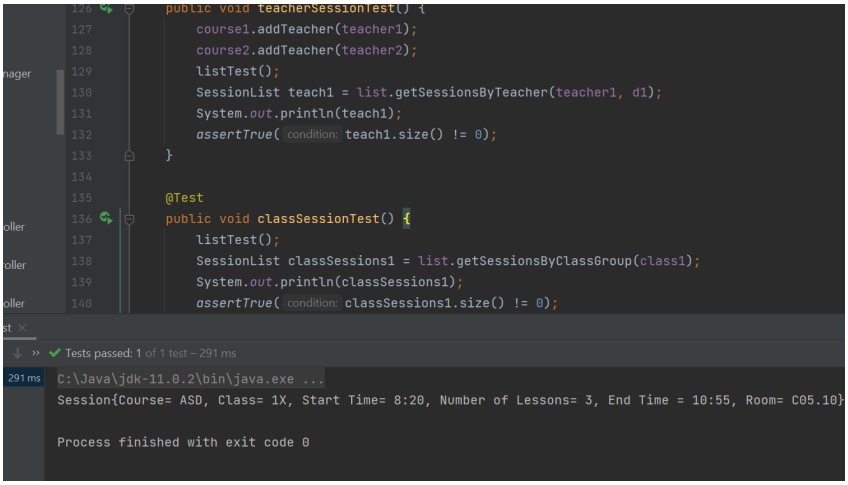
```
310
311                 {
312                     teacherAvailable = false;
313                 }
314                 else
315                 {
316                     teacherAvailable = true;
317                 }
318             }
319         }
320         else
321         {
322             return teacherAvailable;
323         }
324       }
325       else
326       {
327         return teacherAvailable;
328       }
329     }
330   }
331   return teacherAvailable; // O(1)
332   //Returns take 1
333 }
334 // We chose this method because it contains two loops and multiple if statements,
335 // which we find complex compared to the rest of our methods.
336 // For the whole method O(N)= 2 + N*( N^2 + 8) + 1 = N^3 + 8*N + 3.
337 // Dropping the coefficients and constants the time complexity is N^3.
```

isTeacherAvailable() 2/2 with O(N^3) time complexity

5. *Test*

To make sure the requirements were fulfilled, JUnit was utilized in an external branch on the build from the 13th of December. The external branch was used because the main branch contained project-level bugs when the libraries for JUnit were connected. JUnit allows for a very simple overview of tests using a unit test model. By testing individual use cases, the smaller pieces of source code can be examined separately.

Test Case	Result
1. Find the name of the course from a created session	Pass
2. Find the start time of a session	Pass
3. Add a group of sessions to a list of sessions	Pass
4. Get the list of class groups from a text file	Pass
5. Get the list of courses from a text file	Pass
6. Get all the sessions of a teacher on a date	Pass
7. Get all the sessions of a class	Pass
8. Choose the class and read the file accordingly	Pass
9. Display the schedule accordingly to the file	Fail
10. Display a timetable for the chosen week	Fail



JUnit has a very simple execution in this picture. Simple tests are passed or failed with details shown in the console.

6. *Results and Discussion*

Functional requirements

Critical priority:

1. As a planner, I want to create an individual timetable for every teacher and student, which they can find using their VIA ID so that they can see the schedule of their courses for the semester.
    a. The planner is able to create schedules for teachers and students and teachers, but they are not able to locate them using their VIA ID.
2. As a planner, I want to import the document from the head of the department so that I can use this information directly without manual entry.
    a. This function works quite well by using the file chooser from JavaFX. The planner is able to browse their computer for the text files. This seamlessly creates the student objects, course objects, and room objects.
3. As a planner, I want to add, remove and move sessions so that the requests made by the teachers are fulfilled.
    a. Using the GUI, this requirement is fulfilled from the different controllers used for the windows. The sessions can be added and removed. To move a session, it must be removed and then re added.
4. As a planner, I want to book an available room for a session so that teachers have a room reserved to teach their courses.

   a. The suggest rooms function creates a list of rooms that are available for a session. The session that is used as a parameter has the information of the time and date.

5. As a planner, I want to be the only one to use the planning tool so that no one else can make changes.

   a. Currently, changes can only be made to the schedule for the planning tool. Since no changes can be made to the schedule from the website, the planner has sole control of the schedule.

6. As a planner, I want to create a test week that I can send to teachers so that they can review it and suggest changes before it is spread across the semester.

   a. The planner can do this in the main window of the GUI. The teachers can view the week on the website and then send input via email.

7. As a teacher, I would like to view my individual schedule so that I can view the courses I am going to teach.

   a. The teacher's individual schedule for a class can be viewed on the planning tool. However, the same can not currently be said for the website. The different classes, as well as teachers, have different XML files.

8. As a student, I would like to view the timetable so that I know when to go to school to attend classes.

   a. This was accomplished by showing the sessions the class the student belongs to has. The schedule of sessions is published on the website and held in the XML file for their respective class.

9. As a planner, I want to remove the holidays from the schedule so that the timetable accurately reflects the semester schedule made by VIA.
    a. The holidays are removed from the session in the mimic schedule window. The selected weeks are not only skipped during the copying of sessions, they are also given large labels that can be seen in the planning tool when a holiday week is selected to view.

<u>High priority:</u>

10. As a planner, I want to manually select students so that I can add or remove students from the course.
    a. The planner can manually add and remove students from courses in the course details window in the GUI.
11. As a planner, I want to assign multiple teachers to one course so that they can teach the course together and see it in their timetables.
    a. The teachers for a course are set to an ArrayList so they can be added and removed with ease. When multiple teachers are assigned to a course, it is reflected in the schedule.
12. As a planner, I want to be notified of gaps between lessons so that the time of a room that is booked for the lesson is as short as possible.
    a. An alert window pops up during the process of adding a session if there is a gap between lessons. This notifies the planner of gaps but still lets them book the session if needed.

13. As a planner, I want to be able to reschedule a session of one class so that I can use a room with foldable walls to increase the capacity to 90 students if needed.

    a. While reading in a rooms list from a text file a FoldableRoom object is created when a letter is read at the end. This functions as a subclass of Room. However there is currently no functionality for opening the rooms into each other.

Low priority:

14. As a planner, I want to book the same room for each course of the same class so that first-semester students can be in the same room for their courses.

    a. Because the suggested rooms function is able to determine precisely when a room is available, consecutive sessions can be booked in the same room. This is particularly important for first-semester students.

15. As a planner, I want to see the week numbers on the timetable so that I can navigate it using the week number, as is custom in Danish culture.

    a. On both the website and planning tool, the datepickers have been configured to show the week number for the selected date. This makes navigation easier for a Danish planner and Danish students.

16. As a planner, I want to see a suggested room so that a room can be selected quicker with less input.

    a. The suggest rooms function only returns rooms that are available for a session, so it is not possible to book a room that is unavailable.

17. As a Student, I want to select the week that I am viewing so that I can see my schedule in advance.

a. Although users of the website can select a week to view the schedule, it does not affect which sessions are displayed on the page.

18. As a planner, I want to be warned when booking the auditorium for less than 45 people so that it can remain available for larger groups.

    a. The planning tool opens a confirmation window when the planner attempts to book the auditorium for fewer than 45 students. This confirmation window still allows them to book with a lower amount if they choose.

19. As a planner, I want to have a planning tool displaying students enrolled in courses, ECTS points per course, and the teachers teaching each course so that I can compare the timetable with the information provided by the head of the department.

    a. In the planning tool, it is possible to see this information by clicking on a session and viewing the course details.

20. As a student or a teacher, I would like to be able to view the timetable in either English or Danish so that I can understand it better.

    a. On the one hand, the full name of the course in the planning tool is shown in the language of the class group selected. When a session of SDJ is booked for the Danish class, the full Danish title appears in the session and course details. However, the website does not display in Danish.

Non-functional requirements

21. The most recent changes to the timetable are published and visible to the users.

a. The planner can export the schedule they are editing to xml in the planning tool. This holds the most recent changes and is read directly by the website.

7. *Conclusions*

This Timetable Planning Tool fulfills most of the requirements. We decided to display the timetable for classes and that's how students should access them instead of inserting their Via Id. On the website sessions are not displayed correctly. We haven't covered requirement number 13. Currently we can not add a session with the same teachers to two different classes in order to use the foldable rooms. From low priority requirements we started to implement the feature of displaying the chosen week of the schedule on the website, however we have not completed the task. Neither is our website displayed in Danish.

Nevertheless we are able to create a schedule for every class and see individual timetables for teachers in the system. We can add and remove both students and teachers. We are able to see all details that are imported from the files. What is more, we can even choose those files from private libraries on one's computer. Duplication of the tested week is possible, so is scheduling the holidays. We also notify about the time gaps in the schedule, prevent scheduling two lessons at the same time for the specified class and check the teacher's availability.

Our system is effective and intuitive. All possible doubts are cleared by the User Guide. Consequently everyone is able to take advantage of our planning tool.

8. *Project Future*

This system is not ready to be launched. The website is not finished, it does not display the imported schedule correctly and the timetable creates redundant white spaces. We would like to also add Class Group names on the teachers' schedules. Moreover we would like to create a schedule, on the website, using a content management system, which would require less technical knowledge to configure and customize. We also plan to add direct links from the schedule to the course description on StudieNet and course pages in itsLearning. We are still missing the danish version of the website which we would like to still create.

In the future we would like to take account of electives from 6th and 7th semester. We plan to make the GUI windows responsive. To be more precise, windows are supposed to have the same size and also be able to display session and course information next to the schedule simultaneously, so that planner can compare information and view the timetable or add a session at the same time. In addition we aim for a menu bar on the main window with the help button which would include instructions on how to use the system as well as the most popular questions.

Allowing teachers to make basic edits is another function we would like to incorporate in our programm. It would be beneficial for teachers to change rooms, times and remove the sessions. Instead of sending an email to the planner we would like to have it in the system and integrate those using the confirmation, that would be only available for the administrator. Adding a notification, when the request is made, would be another useful feature.

9. *Sources of Information*

Carroll, C., Juers, M. and Kent, S., 2014. Improving the Scheduling of Operating Rooms at UMass Memorial Medical Center. [ebook] Worcester, MA: Worcester Polytechnic Institute. Available at: <https://web.wpi.edu/Pubs/E-project/Available/E-project-030614-144337/unrestricted/Final_OR_Paper.pdf> [Accessed 5 October 2021].

Gaddis, T., 2015. Starting Out with Java: Early Objects. 5th ed. Boston [etc.]: Pearson.

Oude Vrielink, R., Jansen, E., Hans, E. and van Hillegersberg, J., 2017. Practices in timetabling in higher education institutions: a systematic review. Annals of Operations Research, [online] 275(1), pp.145-160. Available at: <https://www.researchgate.net/publication/320675938_Practices_in_timetabling_in_higher_education_institutions_a_systematic_review> [Accessed 5 October 2021].

Vagueware.com. 2021. Top 10 Free Scheduling Software For Schools And Colleges. [online] Available at: <http://www.vagueware.com/free-scheduling-software-for-schools-and-colleges/> [Accessed 4 October 2021].

10. *Appendices*

Appendix A: Project Description

Timetable for Software Engineering

Christian Foyer, 315200

Martin Rosendahl, 315201

Nina Wrona, 315202

Robert Barta, 315242

Software Technology Engineering

Semester 1Y

6th of October, 2021

Table of content

1. Background Description

At VIA University College in Horsens, Bob oversees making the timetable for the courses software engineering students must attend. As this is one of the most populous areas of study at the school, it can be a very stressful task.

First, Bob receives a document from the head of department, which includes classes with the students enrolled, teachers, courses, and the ECTS value of each course. Different types of students must be added or removed to individual courses based on the information in their transcripts. Then, he manually plans a timetable that collects and implements wishes sent by the teachers and combining them with the students' best interest.

Subsequently, he assigns courses to rooms, but must check for overlaps by hand. Bob then sends out the test week he has created to be reviewed by the teachers. Once Bob has incorporated the suggestions of the teachers, he publishes the timetable as a spreadsheet on Studienet and removes the holidays. However, during the semester, updates must be made, which takes plenty of time.

In the current method, all teachers and students can view all of the timetables without logging in. With the intricacies of how the timetable is made now, it is not possible to find a replacement if Bob is not able to do it. Due to the shortfalls of the current method, Bob is not able to easily and efficiently create and publish a timetable for both teachers and students to view. Finally, students can access individual timetables.

To add to the challenges that stand in the way, the university college just moved to a new campus in the center of town with limited rooms; each room is now

precious. In a review conducted in 2014, a group of students at the Worcester Polytechnic Institute analyzed and optimized the scheduling of operating rooms to increase their usage and efficiency. They implemented a new scheduling software that used linear programming. Consequently, the time the operating rooms were in use increased. “According to our liaison at UMMMC, Dr. Tze Chiam, Ph.D., the external benchmark for OR utilization is between 70 and 80%; UMMMC is currently operating at approximately 65% utilization for their operating rooms” (Carrol, Juers, and Kent, 2014). Similarly, the current method is underutilizing the limited rooms at the new campus.

There are current market solutions to help with scheduling, but they are short of ideal. Bob's schedule is special because some rooms can open into each other, and some courses have multiple teachers assigned. Coupled with the inability to account for those features of the timetable, many current tools available have outdated user interfaces that directly make using and reading the timetable difficult (Top 10 Free Scheduling Software For Schools And Colleges, 2021). As these older user interfaces make the software difficult to use, the lack of flexibility in older software compounds the issues.

In a systematic review of timetabling available for different higher education institutions, Vrielink and other researchers found that many timetable tools used today do not effectively cater to the flexibility needed for higher education institutions (Oude Vrielink, Jansen, Hans and van Hillegersberg, 2017).

Summarizing, Bob’s current method requires too much manual data entry and creates obstacles to an efficient timetable.

2. Problem Statement

Main problem

Bob does not have an effective tool to make and publish a timetable for software engineering students and teachers at VIA University College, causing underutilization of rooms and overbooking.

1. What information is needed to avoid double booking?

2. What information do we need to take into consideration while creating a timetable?

3. What information do we need to update the timetable for Bob?

4. What should students and teachers be able to see?

3. Definition of purpose

The purpose of this project is to develop a system that will assist with Bob's scheduling of software engineering courses and publish an optimal timetable that meets the needs of students and teachers.

4. Delimitation

1. We will not create individual timetables.
2. We will not schedule holidays automatically.

5. Methodology

During the first semester, students are limited to using the waterfall methodology: a primitive and iterative development cycle that depends on the previous step to complete the next task. The process consists of a rigid sequence of analysis, design, implementation, and testing. During the final weeks of the project, the waterfall methodology will not be strictly implemented.

6. Time schedule

Total hours: 1100 hours split between 4 group members (275 per person)

Week	Date	Number of Hours (per member)	Tasks
1 (41)	11/10 – 15/10	23	Project description revision
	12/10 6PM		Project Description 1st Draft
	13/10 6PM		Partner Group Feedback + Mid Sem Evaluation
	15/10 4PM		Project Description Final

2 (43)	25/10 – 29/10	23	Analysis
3 (44)	1/11 – 5/11	23	Analysis
4 (45)	8/11 – 12/11	23	Analysis
5 (46)	15/11 – 19/11	23	Designing

6 (47)	22/11 – 26/11	23	Implementation
7 (48)	29/11 – 3/12	45	Java implementation/Documentation
8 (49)	6/12 – 10/12	45	Java/Website design/Documentation
9 (50)	13/12 – 17/12	45	Documentation[SVA(|V2]

	17/12 1PM		Project Deadline

7. Risk assessment

Risks	Likelihood Scale: 1-5 5 = high	Severity Scale: 1-5	Product of likelihood and seve	Risk mitigation e.g. Preventive- & Responsive actions	Identifiers	Responsible

	risk	5 = high risk	rity			
Unreadable timetable	2	3	6	Agree on the color scheme. Asking random person to read the timetable.	Unreadable text, mixing days with months and weeks etc.	Robert
Invalid document received from the Head	2	4	8	Rewrite the document, convert it into a different doc type or	Java doesn't read the docum	Chris

Department				rewrite code to match the doc type.	ent pro perl y.	
GUI doesn't function properly	3	4	12	Work on the GUI in a timely manner so there is plenty of time to debug/fix errors if needed Get Help if needed	Wh eth er or not the GU I dis pla ys pro perl y	Nina
Java doesn't compile due to syntax errors	3	5	15	Work on the code in a timely manner so there is plenty of time to debug/fix errors	Err or is sho wn in IDE that doe s not	Mart in

				if needed Get Help if needed	allow code to compile	

8. Sources of Information

Carroll, C., Juers, M. and Kent, S., 2014. Improving the Scheduling of Operating Rooms at UMass Memorial Medical Center. [ebook] Worcester, MA: Worcester Polytechnic Institute. Available at: <https://web.wpi.edu/Pubs/E-project/Available/E-project-030614-144337/unrestricted/Final_OR_Paper.pdf> [Accessed 5 October 2021].

Gaddis, T., 2015. Starting Out with Java: Early Objects. 5th ed. Boston [etc.]: Pearson.

Oude Vrielink, R., Jansen, E., Hans, E. and van Hillegersberg, J., 2017. Practices in timetabling in higher education institutions: a systematic review. Annals of Operations Research, [online] 275(1), pp.145-160. Available at: <https://www.researchgate.net/publication/320675938_Practices_in_timetabling_in_higher_education_institutions_a_systematic_review> [Accessed 5 October 2021].

Vagueware.com. 2021. Top 10 Free Scheduling Software For Schools And Colleges. [online] Available at:

<http://www.vagueware.com/free-scheduling-software-for-schools-and-colleges/> [Accessed 4 October 2021].

9. Appendices

	Group Contract
Project Name:	SEP PROJECT 1
Due Date:	17 dec, 6PM

Group 1:

1.1.1.1 Name	1.1.1.2 Surname
Nina Anna	Wrona
Martin	Rosendahl
Robert	Barta
Christian “Chris”	Foyer

Weekly meeting’s rules

- We accept 3 absences.
- Discussing previous meeting minutes at the beginning of every meeting.
- Meeting minutes will be taken.

- Group members will be respectful to each other.
- We accept 15 minutes window to arrive.
- Meetings take place at Campus.
- Communicating delays and absences in a timely manner.

Expected schedule

- Tuition period: 1 times a week (obligatory) – Wednesday afternoon.
- Project period: 9.00-12.00 and 13.00-16.00 from Monday to Friday.

Supervisor meetings

- All members attend supervisor’s meetings.

Additionally:

- We agree to help each other with our knowledge gaps.

Consequences:

1. Reading the contract as a group + warning.
2. Round discussing the issue.
3. Evaluation of self and peers regarding issue.
4. Supervisor meeting.

All people signed down agree to all the rules written down in this document.

Nina Wrona

Christian Foyer

Martin Rosendahl

Robert Barta

Appendix B: User Guide

Contents

---

1. Importing the files

2. Adding a Session

3. Mimicking the schedule (spreading out a week throughout the whole semester)

4. View Teacher’s Schedule

5. View Session Details

6. Remove Session

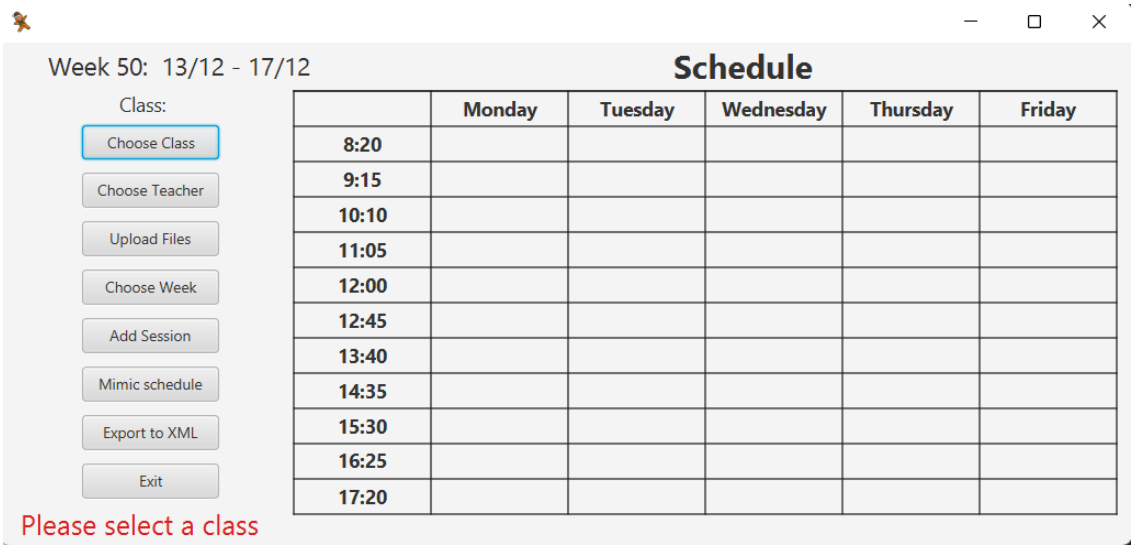
7. View Course Details

8. Add/Remove Teacher/Student

9. Publish the timetable

1. Importing the files

When starting the program, you will see a grid view of the current week (see Screenshot 1). You have to start by importing the text files by following the next steps:

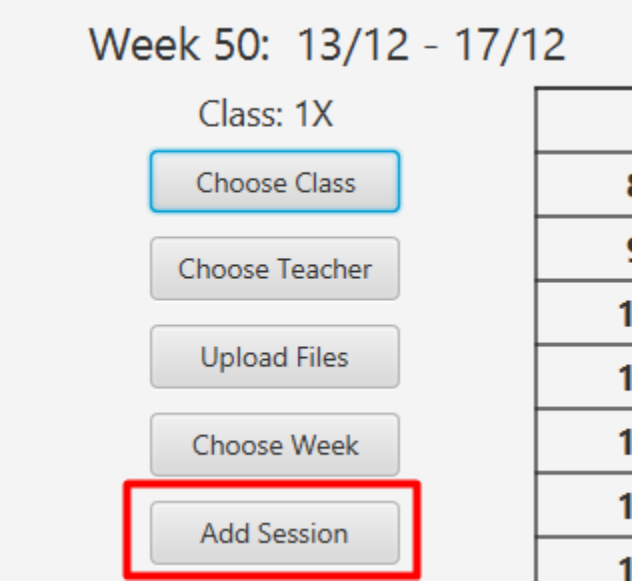


Screenshot 1

1. Click on the Upload Files button.
2. Click on the Select File button separately for browsing the Student, the Course, and the Rooms text files on your computer.
3. Make sure you have selected the corresponding files and click on the “Confirm” button. Your files have now been uploaded.

2. Adding a Session

1. Click on the Choose Class Button (Screenshot 1).
2. Click on the dropdown bar and select the Class you want to add a session.
3. Confirm your selection by clicking on the Confirm button.
4. You can now see which class you have selected above the Choose Class button (Screenshot 2).
5. Click on the Add Session Button (Screenshot 2).



Screenshot 2

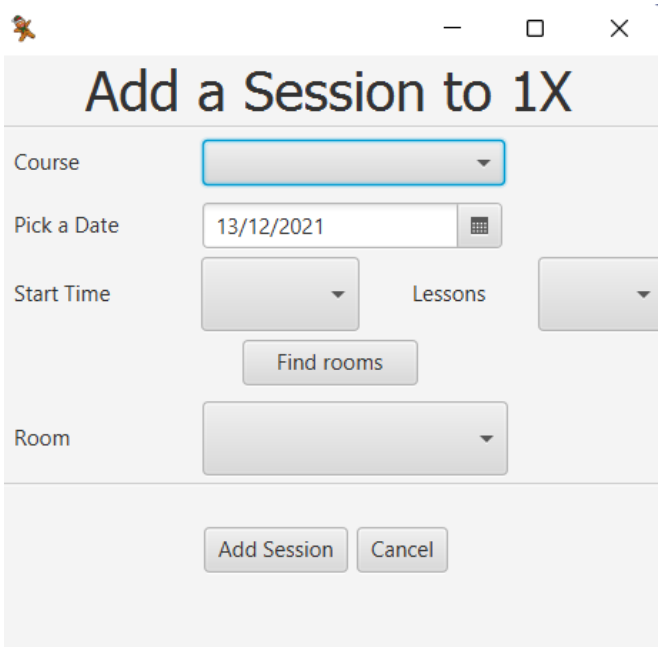
6. In the Add Session Window (Screenshot 3) click on the dropdown bar, then on the course you would like to add to the session.
7. Select the date for the Session.
8. Select the Start Time of the Session by choosing from the values inside the dropdown bar.
9. From the Lessons dropdown bar, select the number of lessons you want the Session to have.
10. Click on the Find rooms button
11. The dropdown bar Room now contains the available rooms for this time. Choose the room you would like to book for the Session.
12. Your session is now booked and displayed on the schedule.

You may run into the following warnings while adding a session:

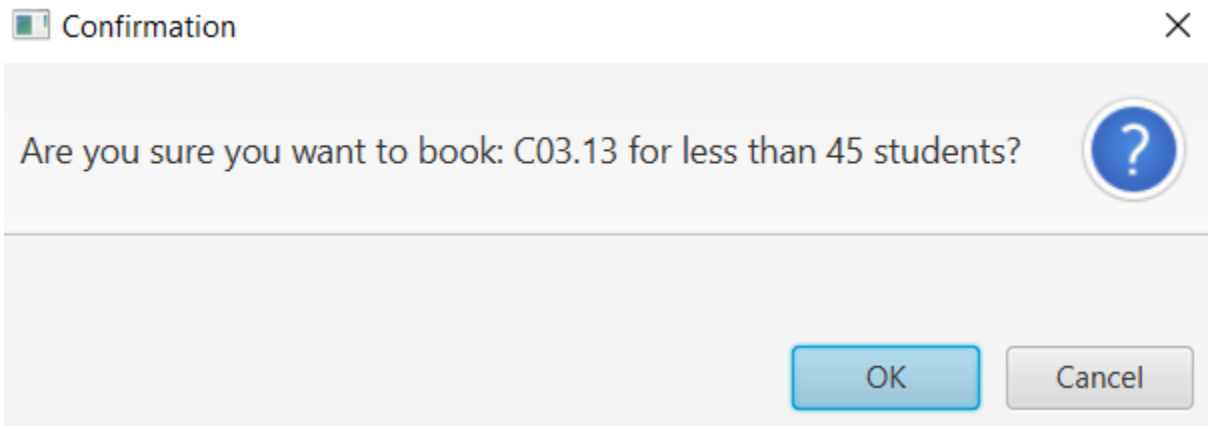
1. You cannot book a session with a number of lessons that would end the session after 18:00.
2. You cannot book a session during the holidays.
3. You cannot book a session that would overlap another session of the class'.
4. You cannot book a session that would overlap another session of the teacher's.

5. If you are booking a room with a capacity of more than 100 for less than 45 students a confirmation window will show up (Screenshot 4). If you would like to book the room anyway, just click OK to book the room and add the session, or Cancel to start all over.
6. If you are adding a session with an empty lesson slot between two sessions (excluding the lunch break), a confirmation window will show up (Screenshot 5). Click OK to book the room and add the session, or Cancel to start all over.

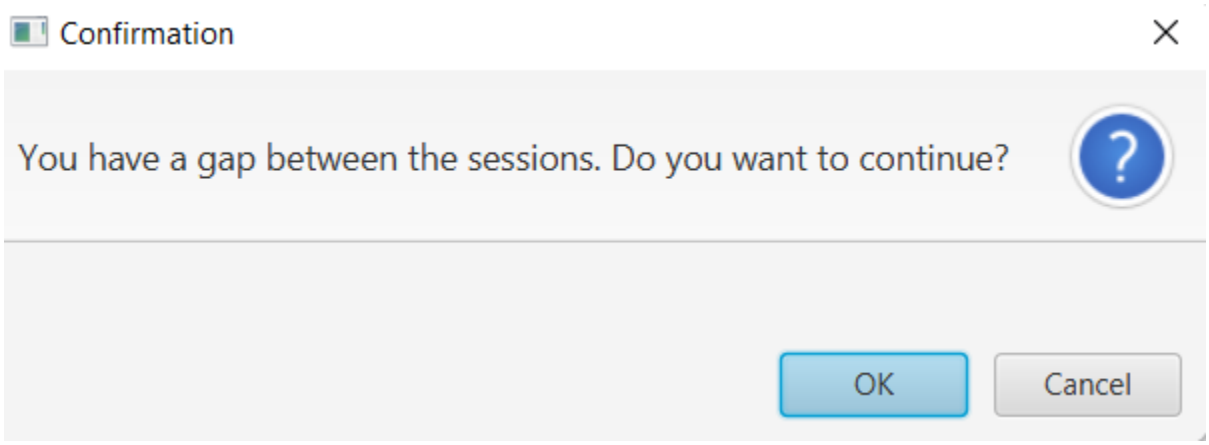
Should any of the 4 first warnings show up, you will have to choose another date/ start time/ number of lessons for the session, depending on which warning showed up.



Screenshot 3



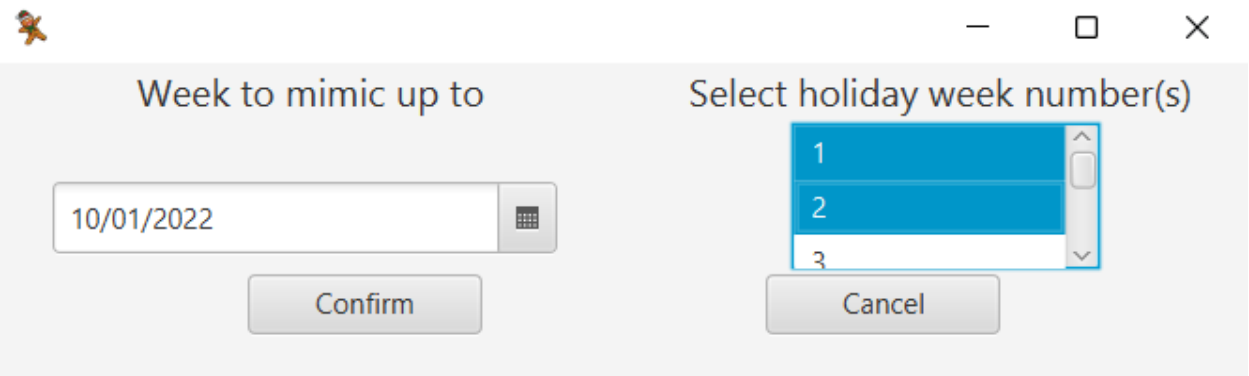
Screenshot 4



Screenshot 5

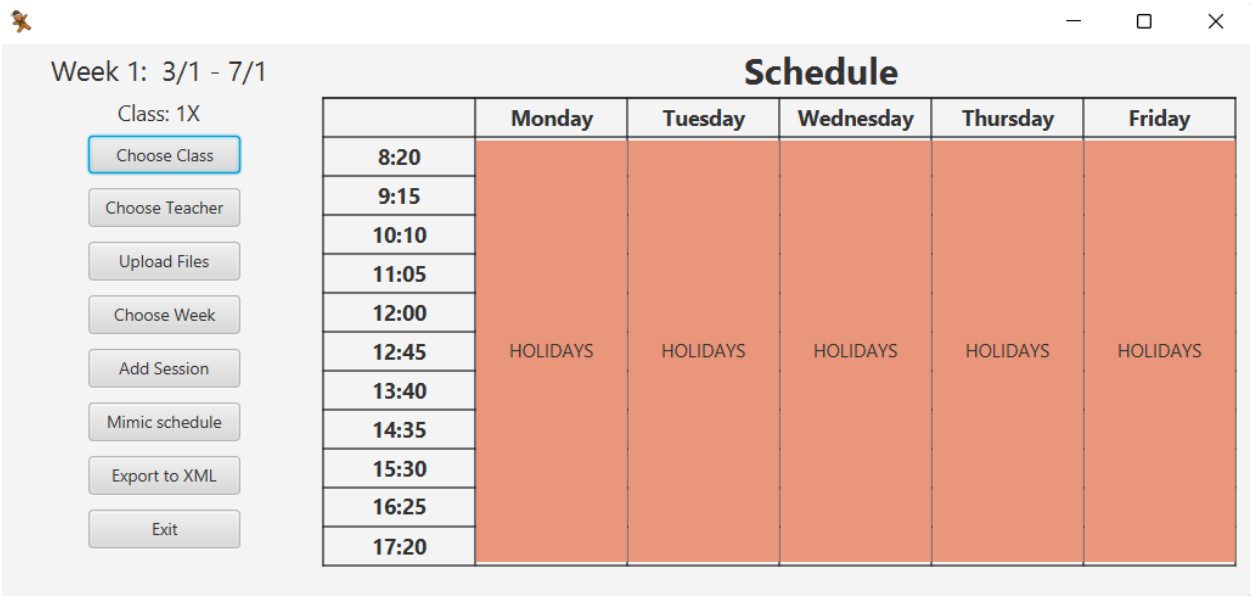
3. **Mimicking the schedule (spreading out a week throughout the whole semester)**

1. You must have at least one Session to be able to mimic the schedule.
2. Click on the Mimic schedule button below the Add Session button.



Screenshot 6

3. Select the week until which (and including) you wish to replicate the weekly schedule.
4. Select the week(s) you want to set as holidays (no sessions will be booked on these weeks and you will be unable to add sessions to these weeks).
5. Press the Confirm button.
6. The sessions have been mimicked up to the chosen date, and the holiday weeks have been added (Screenshot 7).



Screenshot 7

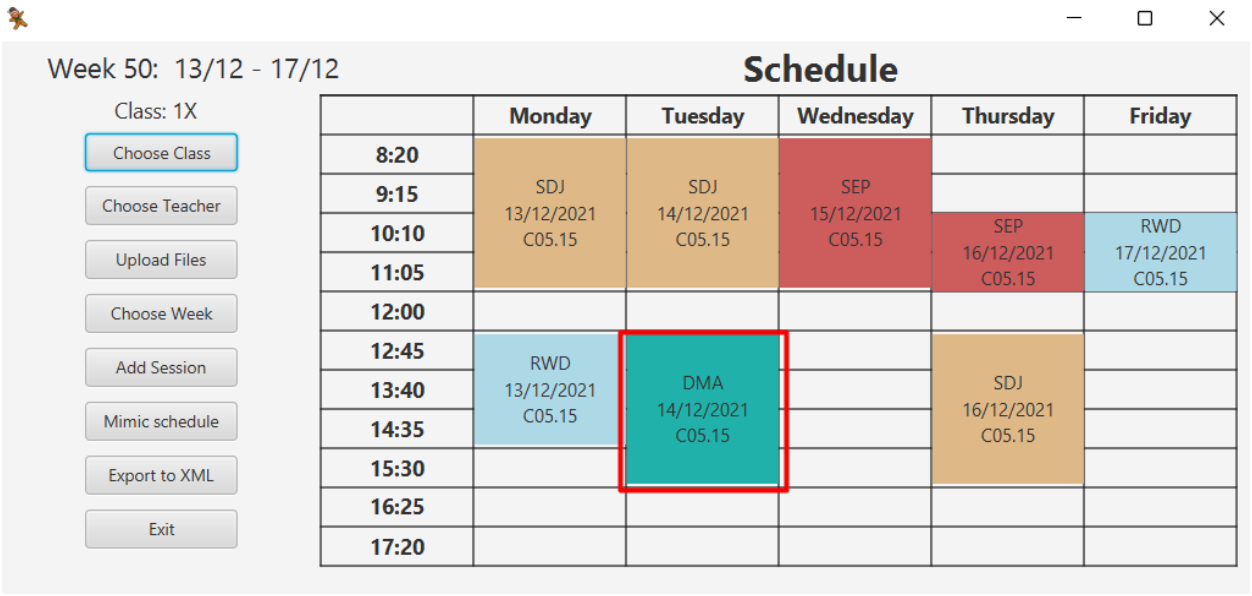
4. View Teacher's Schedule

After adding the sessions, you will be able to see the schedule of a teacher by following the next steps:

1. Click on the Choose Teacher button.

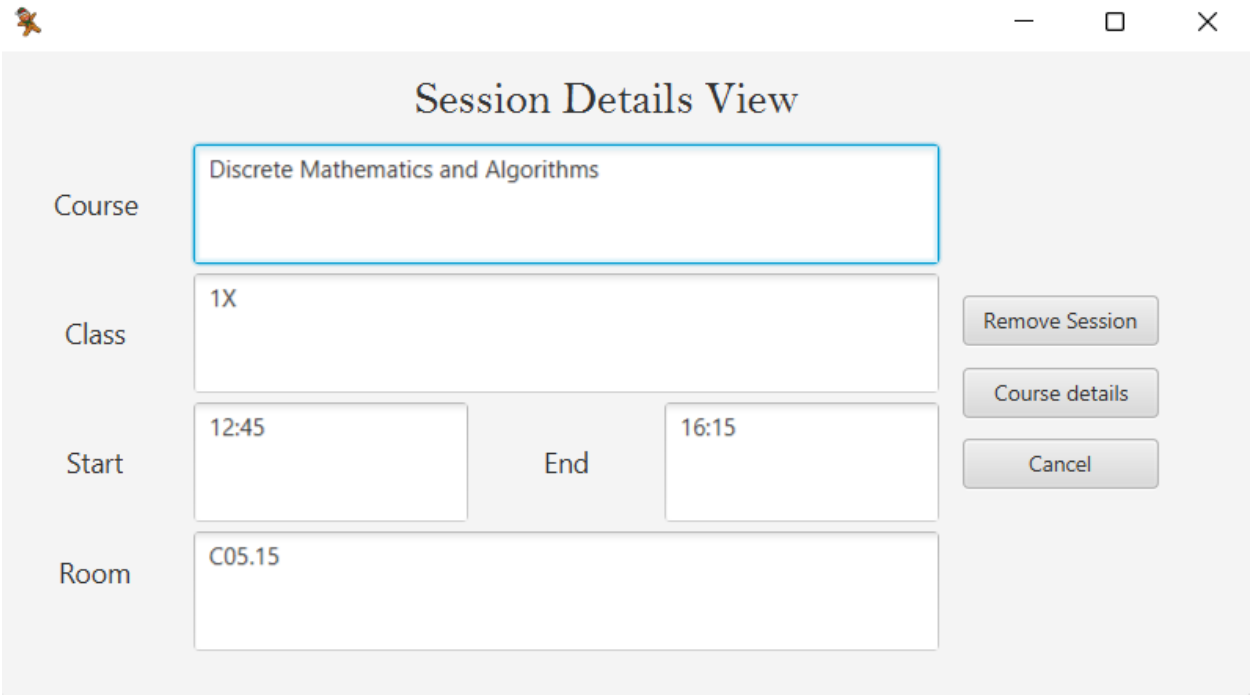
2. Select the teacher for whom you would like to view the schedule.
3. Click the Confirm button.
4. The sessions of the selected teacher are now displayed.

5. View Session Details



Screenshot 8

1. Click on the session of which details you would like to view (DMA in the example on Screenshot 8)
2. A new window will open (Screenshot 9) with the full name of the course (in Danish if it is for a Danish class), the class, the start time, the end time, and the room.



Screenshot 9

6. **Remove Session**

1. Choose the Session you would like to remove (Screenshot 8).
2. Click on the Remove Session button.
3. The session has been removed.

7. View Course Details

1. Choose the Session of the course which you would like to view (Screenshot 8).
2. Click on the Course Details button.
3. The Course Details View will open. Here you can see the full name of the course, the semester in which it is taught, its ECTS value, and two dropdown bars: one where the list with the teachers from this course can be viewed, and one where the list of students from this class can be viewed.

8. Add/Remove Teacher/Student

For the above actions you need to enter the Course Details View by clicking on a Session, then Course Details.

1. To add a Teacher to the course click on Add Teacher, then enter the VIA ID of the teacher to be added to the course.

2. To remove a Teacher from the course click on the dropdown bar and click on the teacher you would like to remove. After selecting the teacher, click on the Remove teacher button. The teacher has now been removed from the course.

3. To add a student to the course click on Add student. In the new window, enter the full name of the student, a 6-digit VIA ID, and the class you want to add the student to. Finalize the addition by clicking on the Confirm button.

4. To remove a student from the course click on the dropdown bar and click on the student you would like to remove. After selecting the student, click on the Remove student button. The student has now been removed from the course.

9. Publish the timetable

1. You either have to buy a domain for the website or open it from a Live Server using Visual Studio Code.

2. Click on the Export to XML button (Screenshot 1). The website will use this XML file to display the timetable on the website.

Appendix C: Source Code and Documentation

SEP1 is the root folder of the project.

Java source code: SEP1Group1Y/Schedule/src/

Website: SEP1Group1Y/Website/

JavaDoc: SEP1Group1Y/JavaDoc/

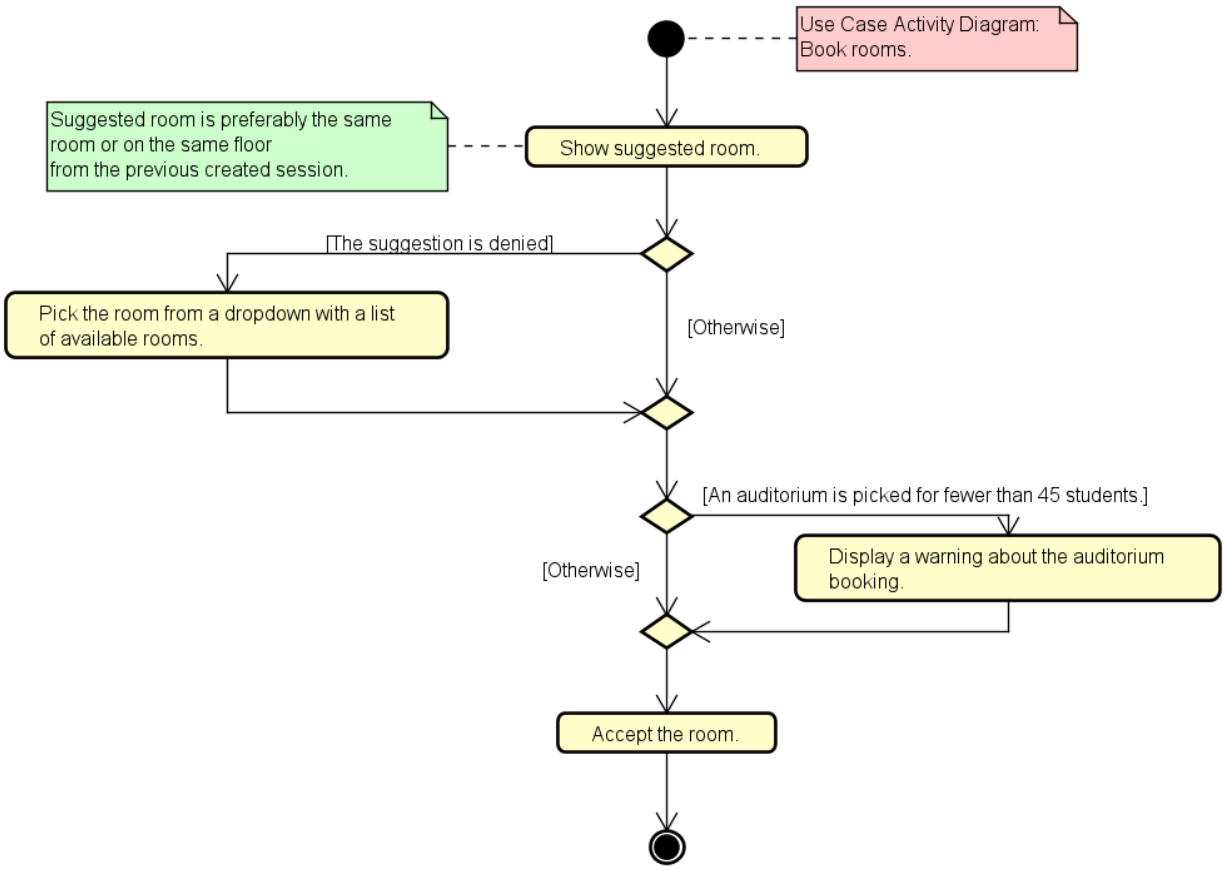
XML files that are read by the website: SEP1Group1Y/

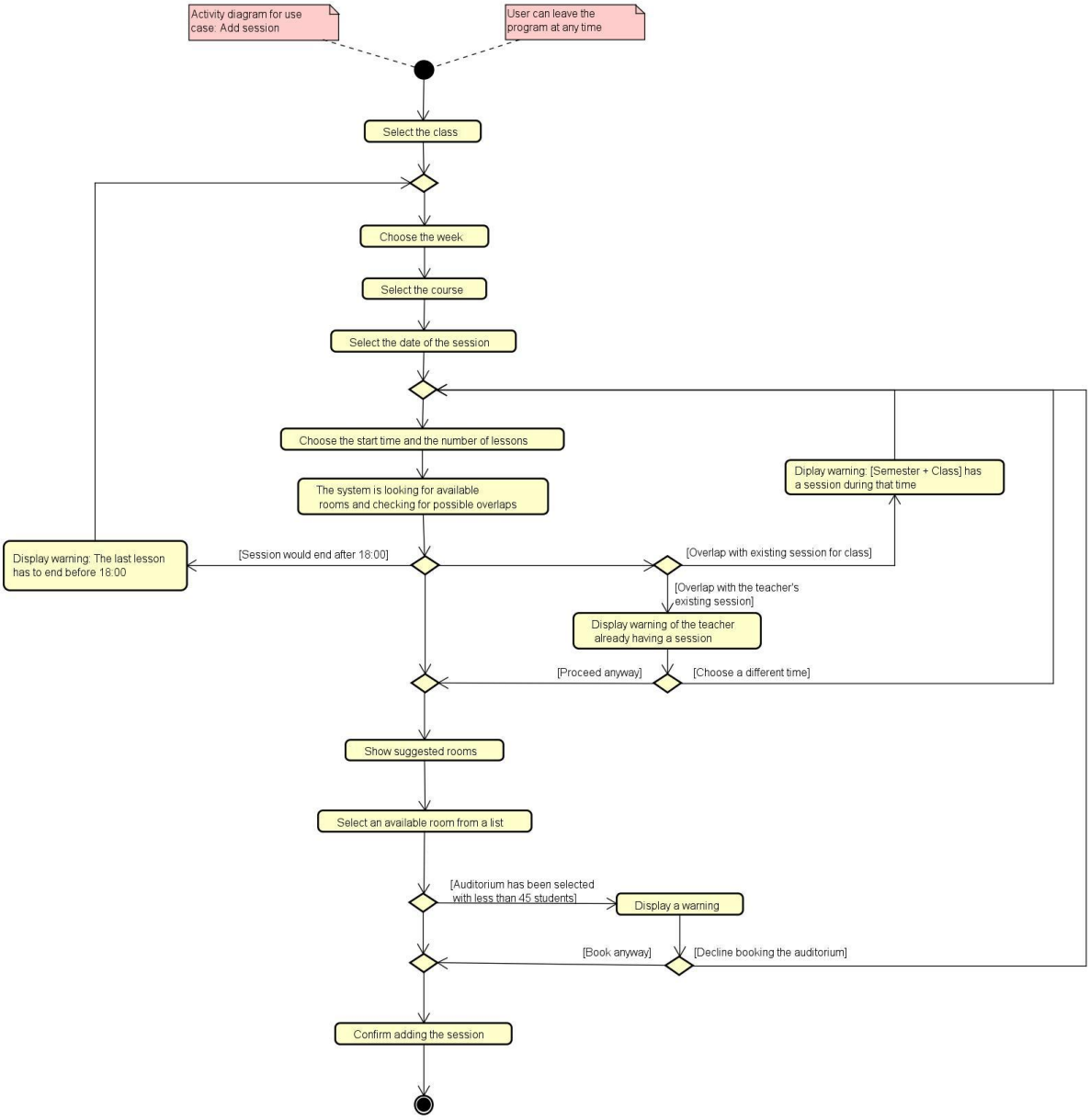
Methods with time complexity analysis in big O notation in Java:
SEP1Group1Y/src/model/list/SessionList.java

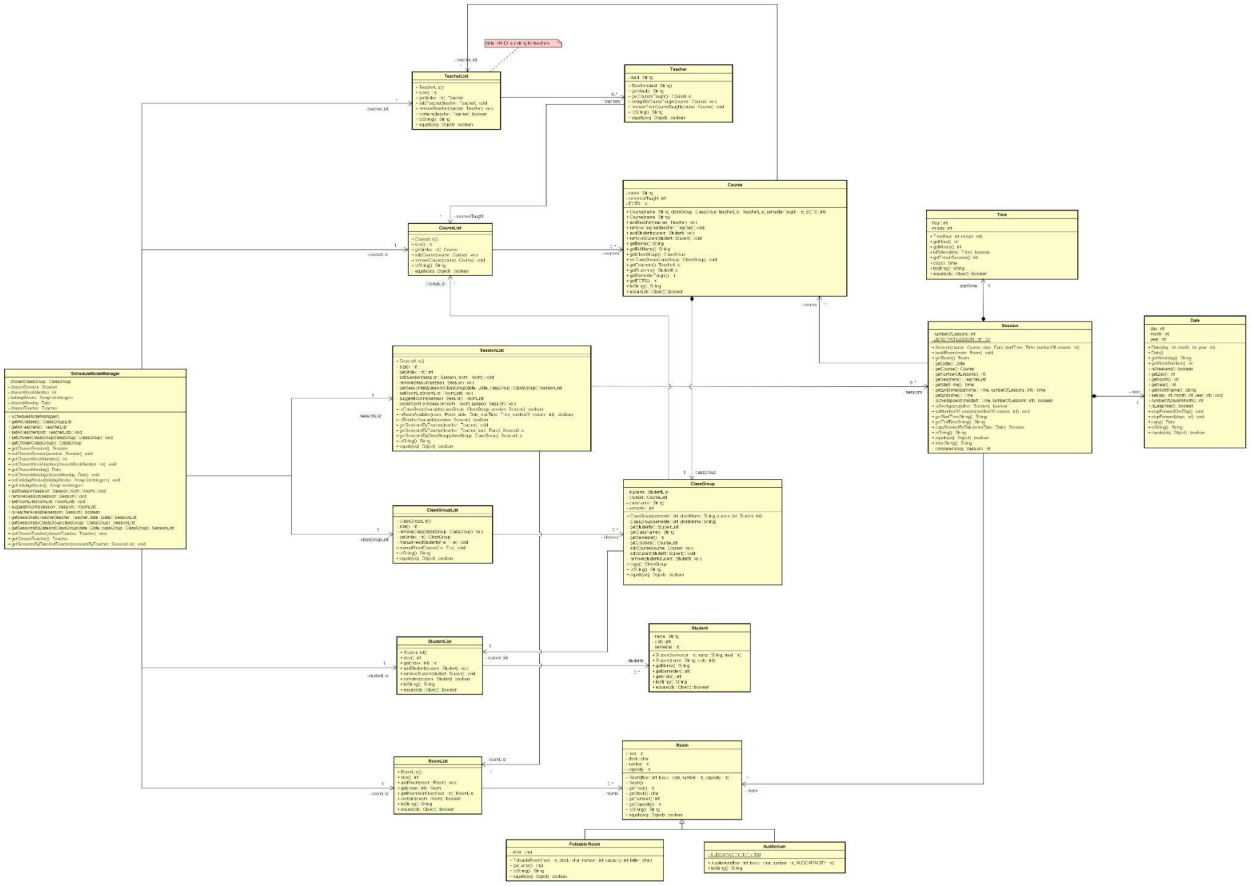
- isRoomAvailable()
- isTeacherAvailable()

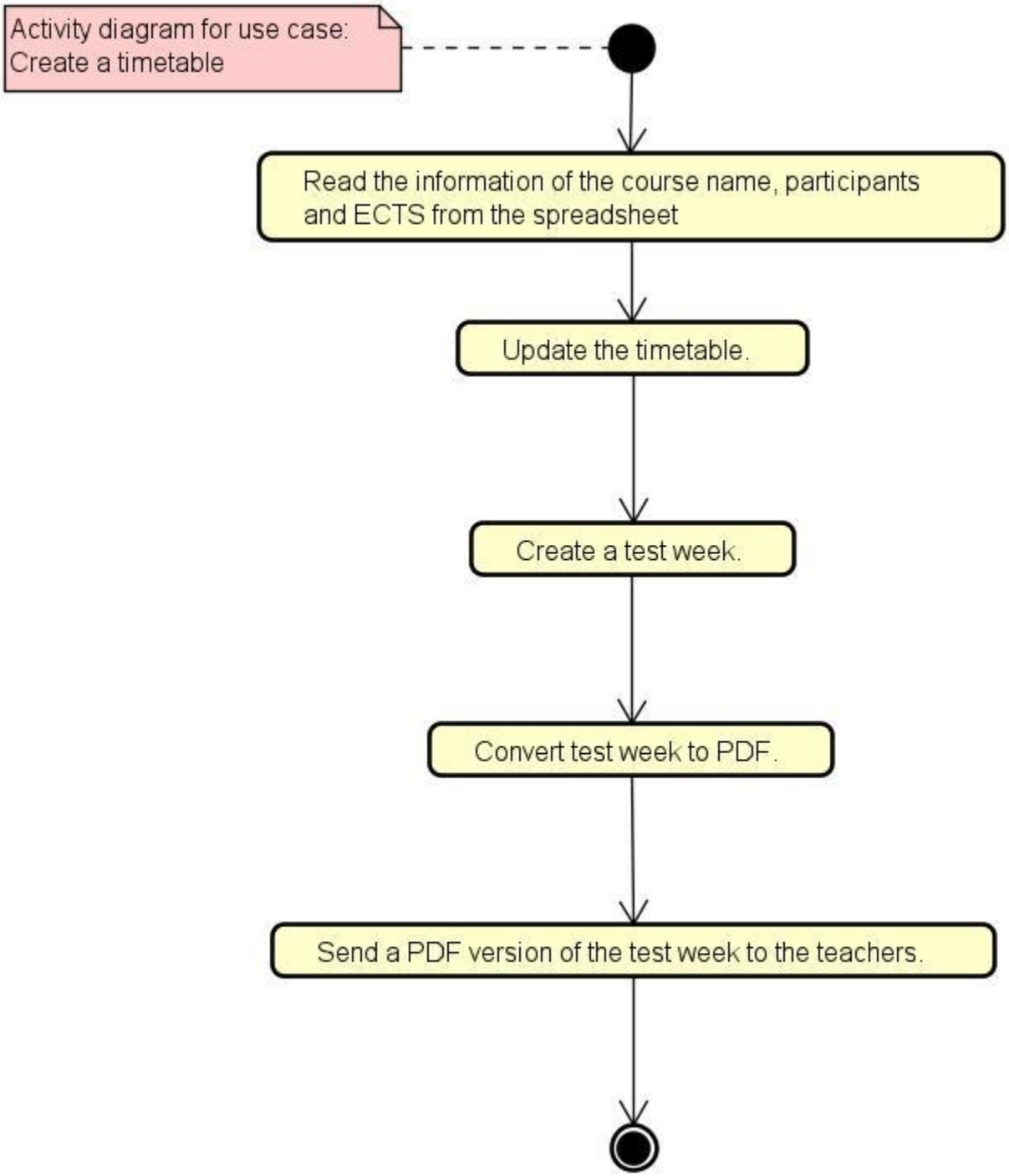
The code snippets with the time complexity analysis are in the implementation section of the report.

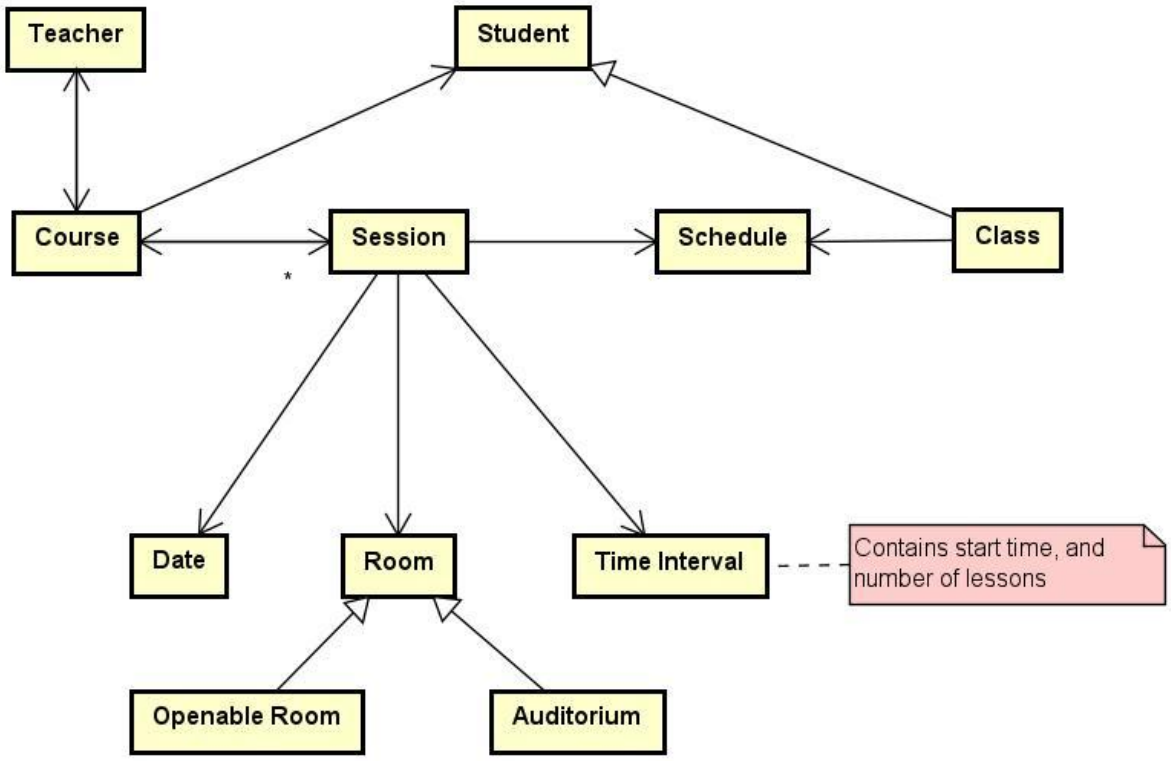
Appendix D: Diagrams

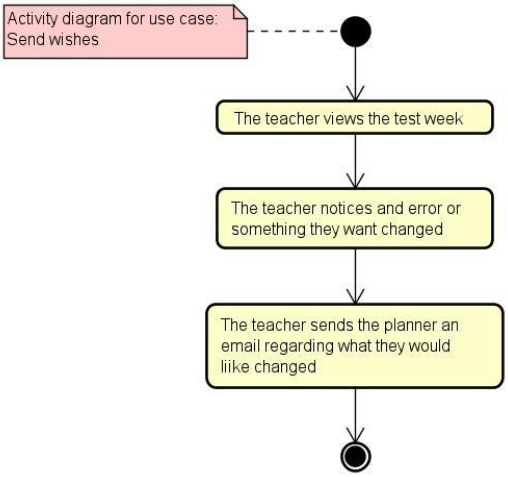


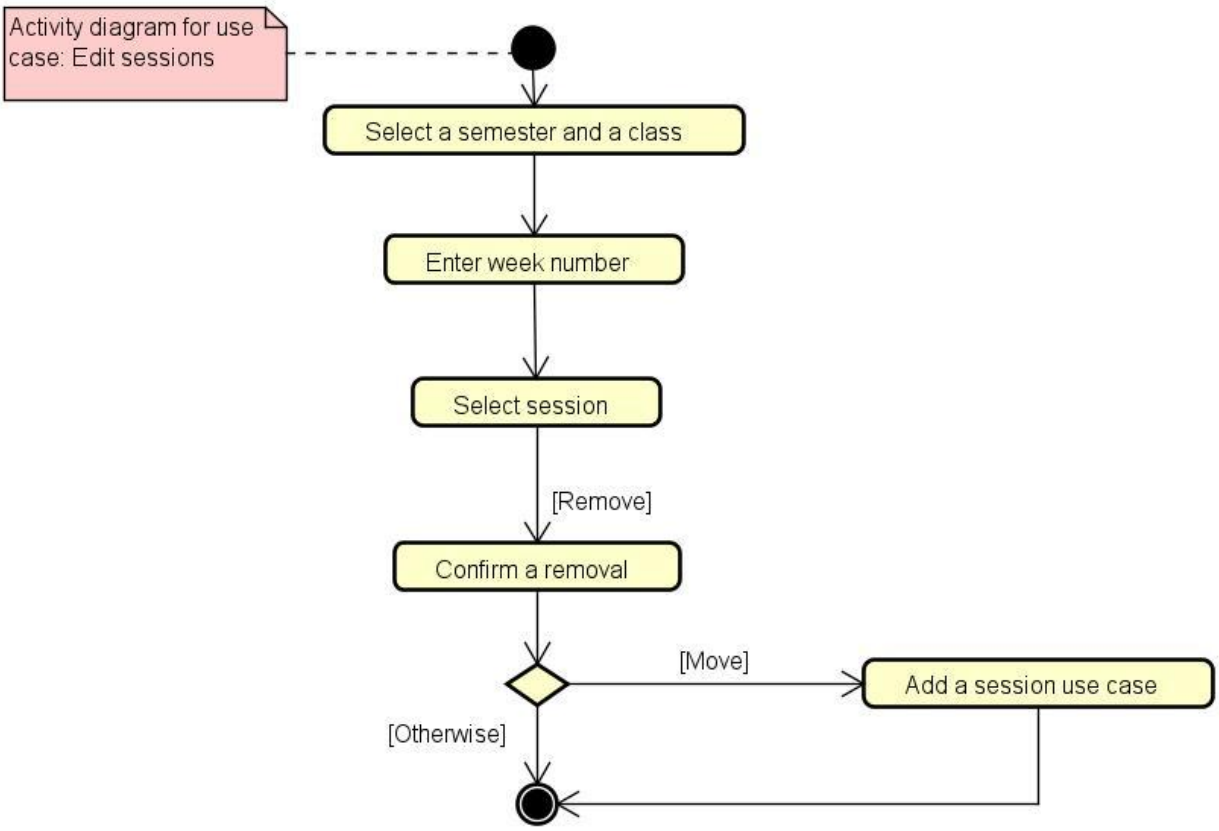

Activity diagram for use case: Add session
User can leave the program at any time
Select the class
Choose the week
Select the course
Select the date of the session
Choose the start time and the number of lessons
The system is looking for available rooms and checking for possible overlaps
Diplay warning: [Semester + Class] has a session during that time
Display warning: The last lesson has to end before 18:00
[Session would end after 18:00]
[Overlap with existing session for class]
[Overlap with the teacher's existing session]
Display warning of the teacher already having a session
[Proceed anyway]
[Choose a different time]
Show suggested rooms
Select an available room from a list
[Auditorium has been selected with less than 45 students]
Display a warning
[Book anyway]
[Decline booking the auditorium]
Confirm adding the session

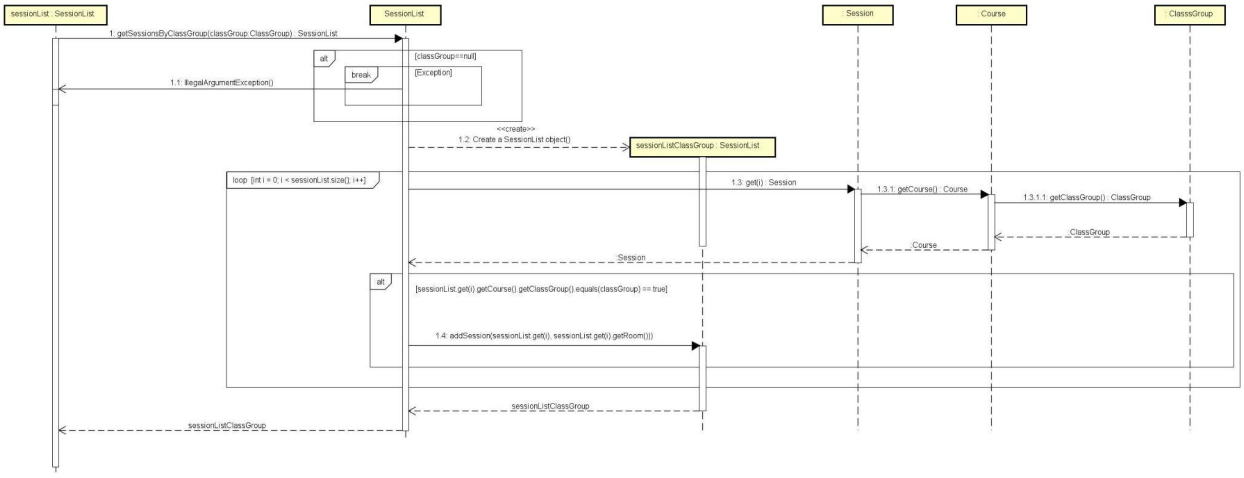


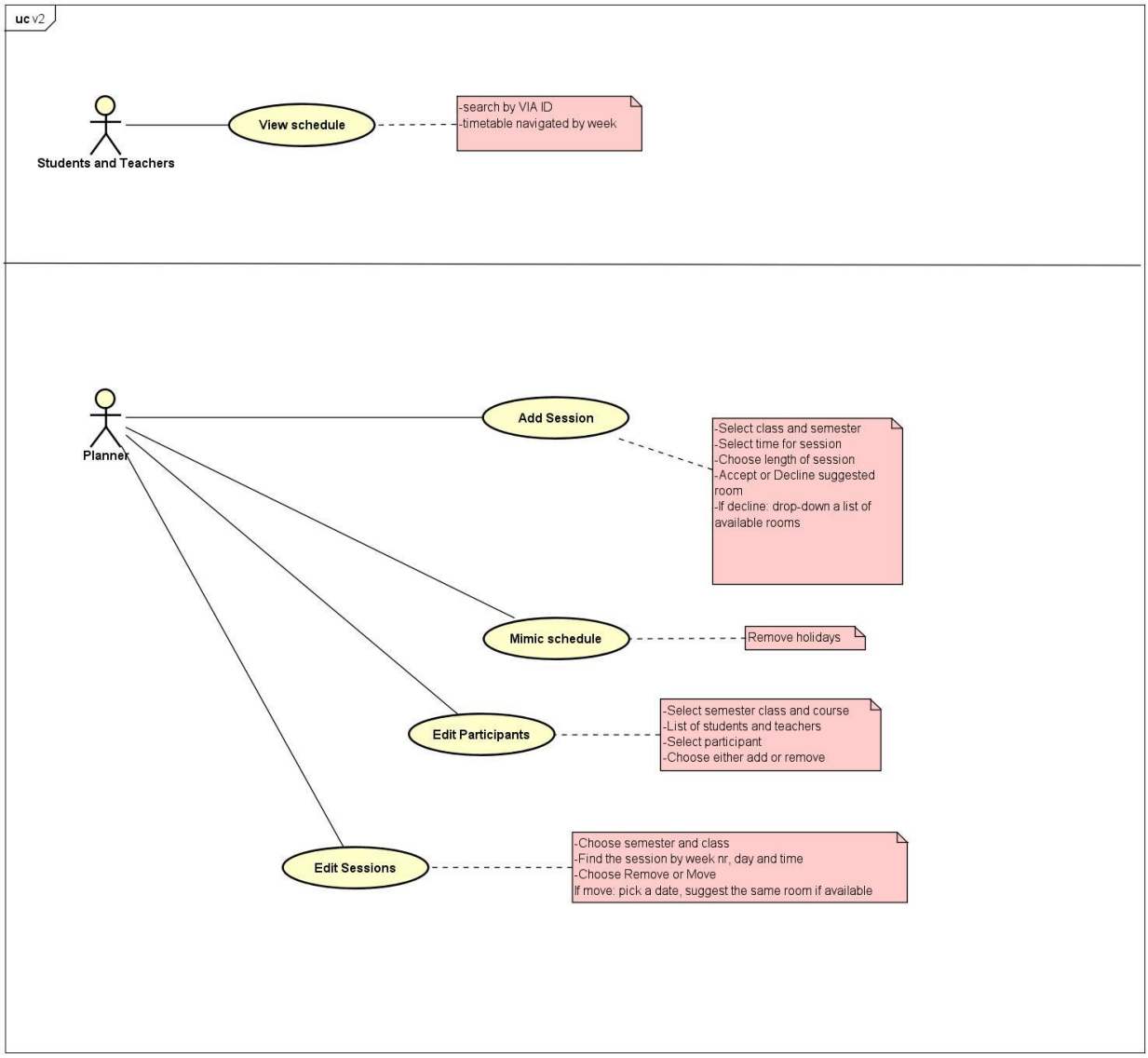

Activity diagram for use case:
Create a timetable
Read the information of the course name, participants and ECTS from the spreadsheet
Update the timetable.
Create a test week.
Convert test week to PDF.
Send a PDF version of the test week to the teachers.

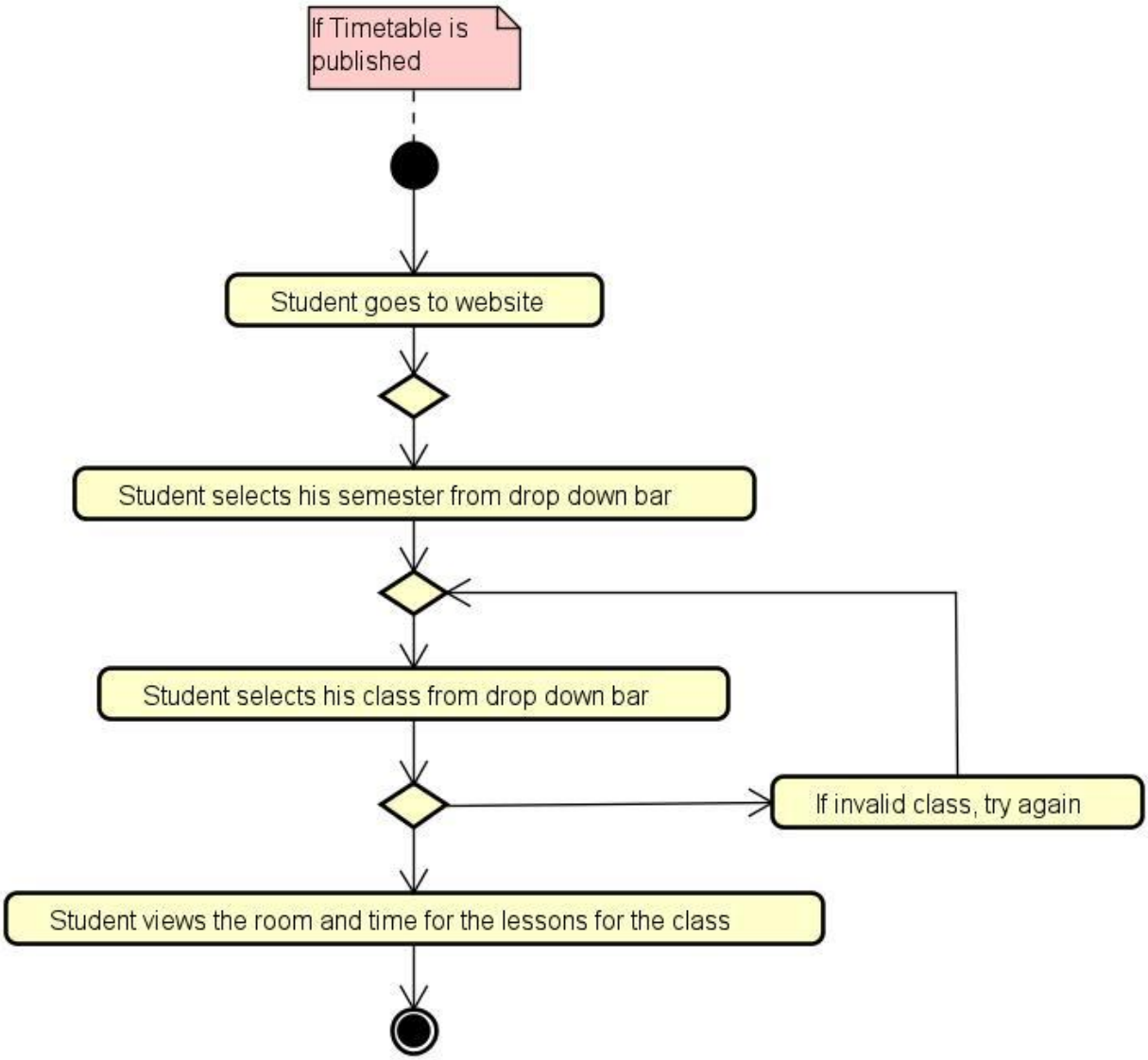

Teacher
Student
Course
Session
Schedule
Class
*
Date
Room
Time Interval
Contains start time, and number of lessons
Openable Room
Auditorium

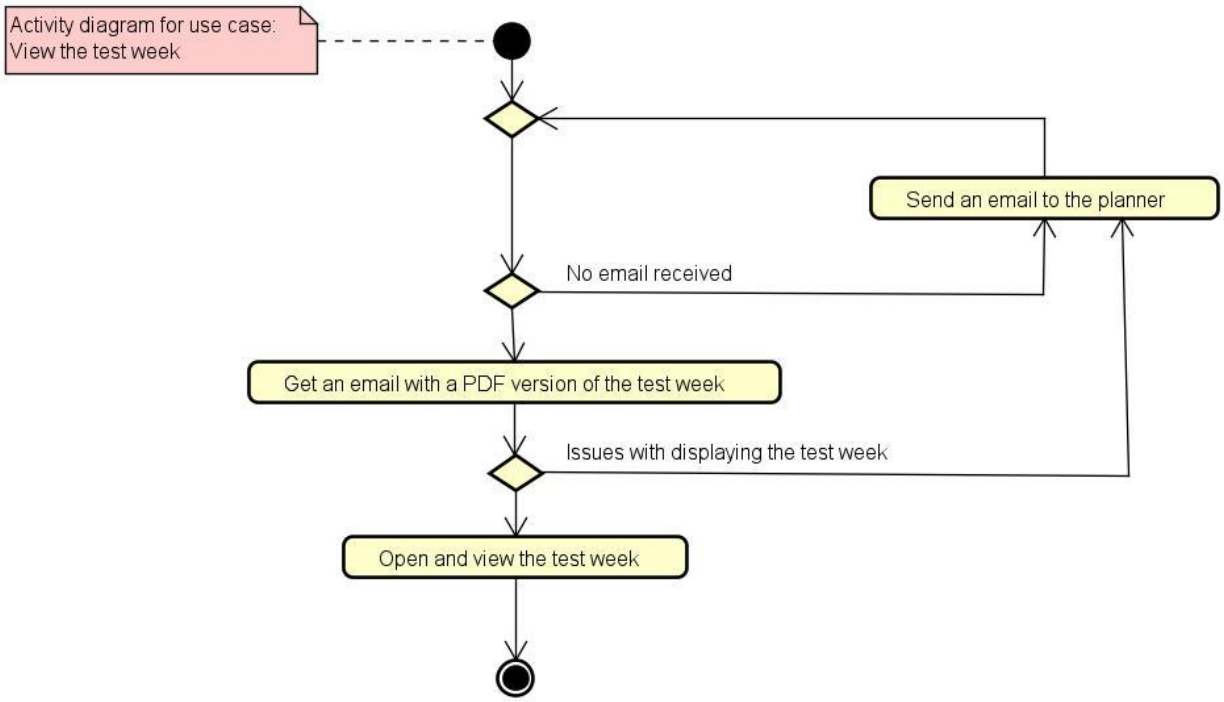

Activity diagram for use case:
Send wishes
The teacher views the test week
The teacher notices and error or something they want changed
The teacher sends the planner an email regarding what they would liike changed

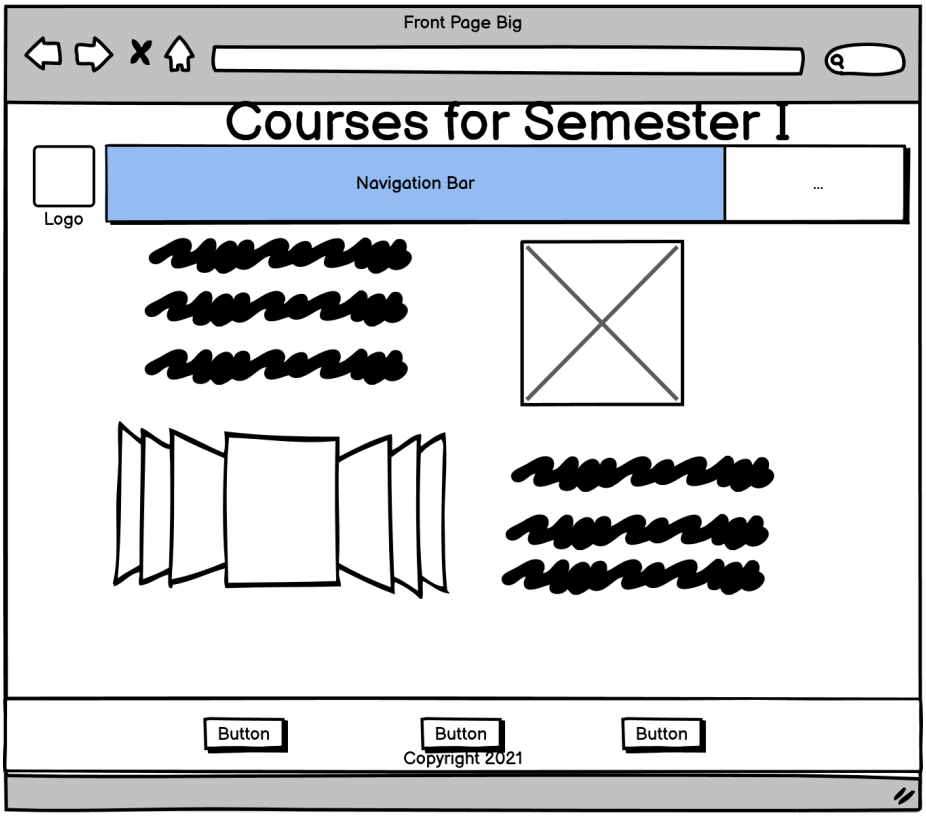

Activity diagram for use case: Edit sessions
Select a semester and a class
Enter week number
Select session
[Remove]
Confirm a removal
[Move]
Add a session use case
[Otherwise]

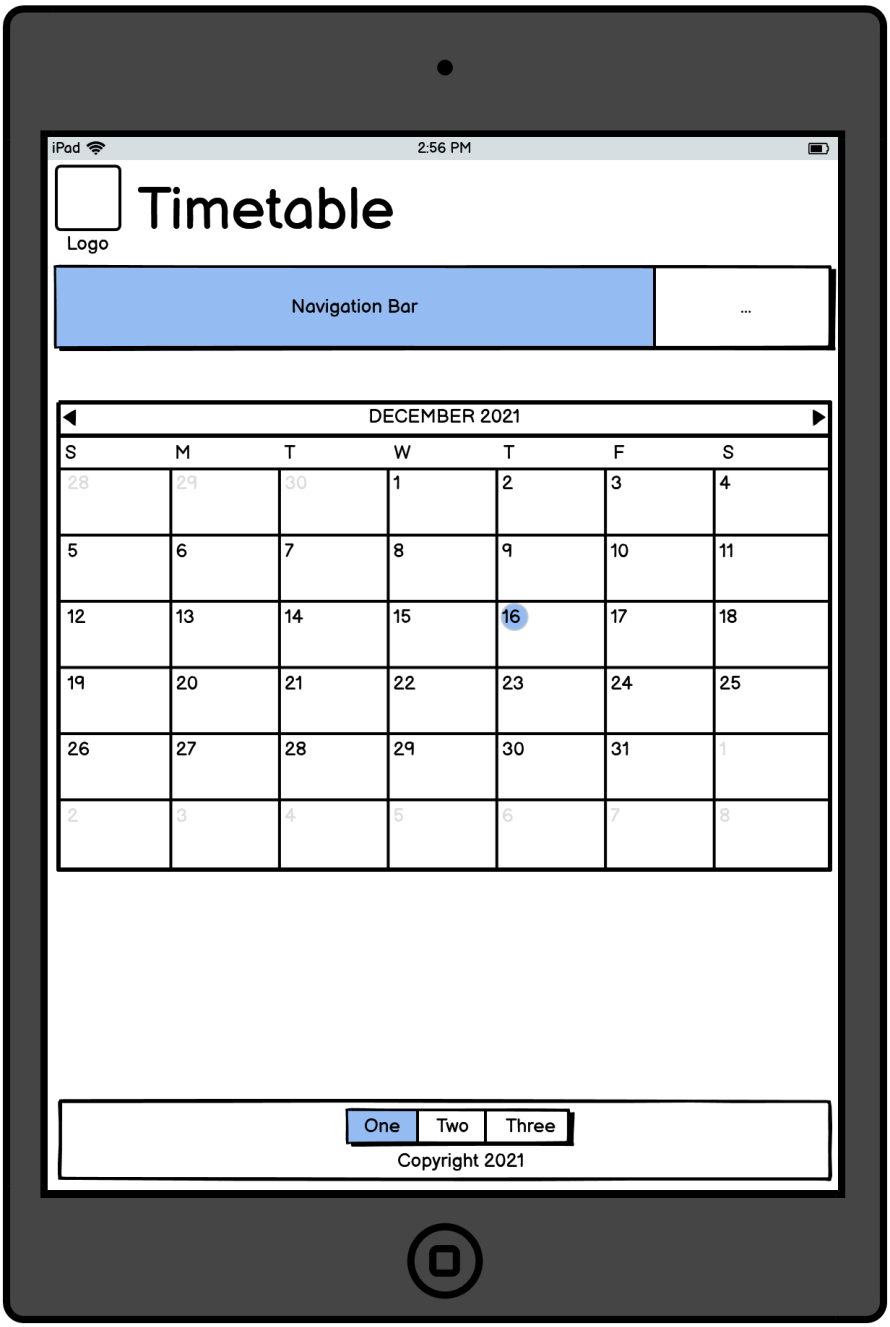

sessionList : SessionList
SessionList
: Session
: Course
: ClasssGroup
1: getSessionsByClassGroup(classGroup:ClassGroup) : SessionList
alt
[classGroup==null]
break
[Exception]
1.1: IllegalArgumentException()
<<create>>
1.2: Create a SessionList object()
sessionListClassGroup : SessionList
loop [int i = 0; i < sessionList.size(); i++]
1.3: get(i) : Session
1.3.1: getCourse() : Course
1.3.1.1: getClassGroup() : ClassGroup
:ClassGroup
:Course
:Session
alt
[sessionList.get(i).getCourse().getClassGroup().equals(classGroup) == true]
1.4: addSession(sessionList.get(i), sessionList.get(i).getRoom())
sessionListClassGroup
sessionListClassGroup


uc v2
Students and Teachers
View schedule
-search by VIA ID
-timetable navigated by week
Planner
Add Session
-Select class and semester
-Select time for session
-Choose length of session
-Accept or Decline suggested room
-If decline: drop-down a list of available rooms
Mimic schedule
Remove holidays
Edit Participants
-Select semester class and course
-List of students and teachers
-Select participant
-Choose either add or remove
Edit Sessions
-Choose semester and class
-Find the session by week nr, day and time
-Choose Remove or Move
If move: pick a date, suggest the same room if available


If Timetable is published
Student goes to website
Student selects his semester from drop down bar
Student selects his class from drop down bar
If invalid class, try again
Student views the room and time for the lessons for the class


Activity diagram for use case:
View the test week
Send an email to the planner
No email received
Get an email with a PDF version of the test week
Issues with displaying the test week
Open and view the test week


Front Page Big
Courses for Semester I
Navigation Bar
...
Logo
Button
Button
Button
Copyright 2021


iPad
2:56 PM
Timetable
Logo
Navigation Bar
...
DECEMBER 2021
S M T W T F S
28 29 30 1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31 1
2 3 4 5 6 7 8
One Two Three
Copyright 2021

Software Technology Engineering

Semester 1 - Class Y

Timetable Planning Tool
Process Report

Group1
Christian Foyer (315200)
Martin Rosendahl (315201)
Nina Wrona (315202)
Robert Barta (315242)
Kamil Fischbach (315273)

Supervisors:

Mona Wendel Andersen

Steffen Vissing Andersen

Date of completion:

17/12/2021

The number of characters in the main text: 55653

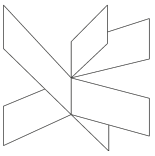
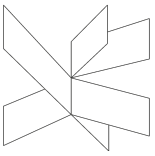


Table of content



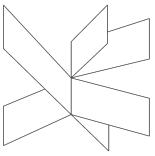
1 Introduction

This document contains Group 1 reflections on collaboration during the first Semester Project. It contains our personal judgment as well as the team's point of view regarding each member’s contribution, learning process, issues we have faced while working on the system, advantages, and disadvantages of decisions we have made.

The process of creating the “Timetable Planning Tool” involved group meetings, consultations with supervisors, and individual work. We were writing down meeting minutes systematically before the start of the project period, during which we took meeting minutes once a week. They captured both most of our milestone decisions and setbacks.

Relying on them we can state that only at the end of the second week of the project period we have had a slight slowdown in the work progress, however, we got through it and managed to successfully motivate one another to continue.

We collected all of our thoughts and reflections in this document in order to improve the group work on upcoming semesters.



2 Group Description

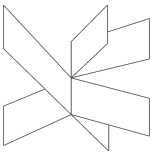
Our group contains 5 members Nina Wrona, Kamil Fischbach, Robert Barta, Martin Rosendahl and Christian Foyer. We are all from different countries, there is an age gap between us however we have started our project with similar coding skills.

Nina and Kamil are the youngest contributants in the group. They both come from Poland and they received comparable education before coming to VIA University. They studied extended mathematics and physics, and as a result, they haven’t had any skills in programming beforehand.

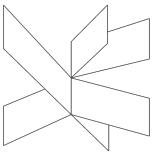
Christian and Martin are both half Danish and half American. On the contrary, they are the oldest members of the group. Neither Christian nor Martin have studied anything related to Software Engineering. Martin was studying math and physics at HF in Odense. On the other hand, Christian, after graduating high school in Minnesota has taken several jobs including sommelier, goatherd, and pollster. Therefore, Christian has gained a lot of knowledge in the area of group working and time management, that he was wisely using during the project work.

Finally, Robert is a group connector. He comes from Romania and he studied Bachelor of Economy of Commerce, Tourism, and Services. All knowledge regarding Software he gained from the current program, like all of us.

Regarding our background we faced several setbacks while preparing the project, however, they were small and irrelevant regarding the whole process, we haven't paid much attention to them.



On the bright side, we learned a lot about each other's cultures, traditions, and tools to work. Robert introduced us to Google Documents, which made our relevant documents writing much easier. We worked on several parts of the document at the same time and thanks to that we were able to complete them quicker. Nina taught us how to use GitHub, which in return made our coding-related work much faster. We could share our progress quickly and follow all the proceedings.

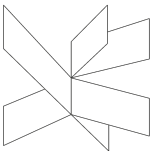


3 Project Initiation

The topic of our project is “Timetable Planning Tool” and was dictated by our educational program. Before we were formed into groups our supervisors made an exercise when they put us in a team with completely random people from our class. After a few days, we received a task to form groups with people of our own choices. In our case, we stuck to the pairing that occurred during the above-mentioned exercise. We found that our work approach is similar and after reaching success in our first task we were very excited to keep up the good work.

In the beginning, we created a 4-person team with Nina, Christian, Martin, and Robert. We were able to choose our group members at the beginning of the semester project. Kamil joined us after the autumn after he was dissatisfied with his previous group. The choice of our group members was based on our motivations and goals. According to self determination theory, we all relied on internal motivation most of the time (Niemiec and Ryan, 2009). We were driven by curiosity and willingness to learn more. Martin stated he had a lot of fun while programming, Kamil was feeling accomplished whenever his program followed the intended path. Chris has willingly devoted a lot of his leisure time to exploring the code and its issues.

Naturally, we were aware that during the whole project we might lose this motivation and experience frustration, sadness, and discouraging failure. Therefore we decided to keep friendly, after-work meetings to keep reminding ourselves of our goals and motivations in a non-stressful environment.

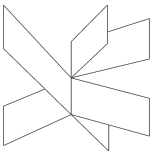


On top of that, the whole group saw a huge opportunity to prove the personal competencies of each member and wisely use the autonomy given to us to improve our skills in fields such as coding, working in a group, time management, and academic writing.

While forming the group we made sure to align our expectations and goals while writing the group contract. That way, when we started, we aimed for goals together as a group.

According to the Lean Construction Institute in their article about new group dynamics, a newly created group should discuss conditions of satisfaction, design vision, team structure, and team culture at the beginning of the collaboration (Lean Construction Institute, 2021). The result of our agreements is stated in the Group Contract. It included our schedule, working rules, and conflict management. We agreed on meeting every Monday and Wednesday during the tuition part of the semester and from Monday to Friday during the project period. We accepted a 15 minutes window to arrive and 3 absences. We didn't face any disagreements during that phase but we were aware that they may occur in the future, so as a part of the conflict management preparation we decided to face every issue together by rereading the contract, discussing the problem, and if needed, scheduling a conversation with a supervisor. Later Kamil joined our group and agreed on our rules without exceptions, so we could follow them from the very beginning till the end.

As emphasized in the above mentioned article "Lean/IPD projects benefit from a focused effort on team building, dynamics, role definition, and vision-casting activities which are repeated whenever a new team is formed, new participants enter teams, and whenever there are significant changes in team operations" (Lean Construction Institute, 2021). We focused on activities that have built a



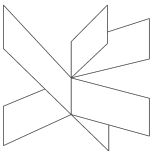
bond and mutual understanding in our team. Firstly we worked together on smaller assignments over the past few months, getting to know each other's work styles. Additionally, we gained knowledge about the strong and weak points of every group member. We used it later while assigning tasks.

Afterward, we went to several informal meetings, for example to the Bone's Restaurant, in order to improve the overall atmosphere of our meetings.

Regarding team structure we haven't conducted personal assessments, however, Chris naturally became a leader of our group, very expressional personalities of Nina and Kamil were balanced by calmer members Martin and Robert.

We planned our project using the Waterfall project management methodology. As stated in the online article "The Waterfall methodology—also known as the Waterfall model—is a sequential development process that flows like a waterfall through all phases of a project (analysis, design, development, and testing, for example), with each phase completely wrapping up before the next phase begins" (Adobe Workfront, 2021). This means that we first finished the analysis part of the project, then moved to the design, and never came back to correct mistakes we made in already closed phases.

Following this ordered list of tasks made us understand how every phase should be covered. We took enough time to make sure we understand the purpose and importance of each stage and never leave anything to the end of the project. However, the disadvantage of this approach is that when we discovered any wrongly designed methods we could not go back to change them.

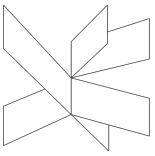


4 Project Description

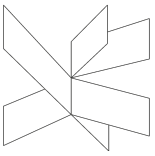
Project description was the first official document we wrote together. We started with watching a project description video to get a brief overall view of what we are going to do. Then we worked through all chapters that the project description should contain and we divided them between us, saying that each person needs to cover one or two of them. We felt that this way we can spend less time on preparing single parts of the document and focus on the overall outcome instead. For example we took time to make sure that everyone understands the Waterfall project management. This allowed us to save our group meetings for clarifying any doubts instead of making the document from scratch.

Therefore, when the first draft was finished we discussed all relevant sections. We summarized and categorized answers for the customer that were relevant for us in order to finish the background description. We agreed on milestones and deadlines and thoroughly analyzed the problem statement. We planned approximately what we will work on during specific weeks and scheduled deadlines to make sure we have time for each part of the project.

With the last one, we had some minor issues. We weren't sure how to ask questions and avoid referring to the design part of the project. We had to rewrite them several times to finally get 4 that were specific enough.



We have to admit that we also struggled with the background description. Another challenge was to collect suitable sources of information and fade them into the document. Also, in our opinion, making the bullet points out of the interview with a customer was difficult. The reason was that he was talking very chaotic and provided both important and redundant information, so we had to first write down and then decide on which ones we should include. We had several disagreements in that part of the project, however every time we have come to a compromise.

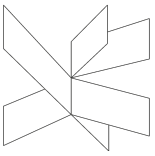


5 Project Execution

In the project execution phase, we began with creating GUI windows and in accordance with the Analysis document, we started coding. We followed the project plan by focusing on the Java implementation and creation of the system. At the same time we were adding first information to the Project Report accordingly. Subsequently in the middle of the second week we worked on finishing necessary methods and started merging the Java with HTML of the website. We were working on both Java and Html until Tuesday of the last week of the project period. Then we put greater emphasis on relevant documentation.

In our first class diagram, regarding the basic and the list packages, we planned to have toString, getters to every instance variable and setter methods in every class, which was unnecessary in most cases. However, it resulted in being faster while coding more complicated methods, because we knew those basic ones that were likely to be used later. Sometimes we even discovered that those pre-prepared methods were inaccurate, for example, getters in the SessionList Class were taking inaccurate parameters, therefore we had to correct them. And even if cleaning the code was a bit more time taking before handing in the project, we still believe it was worth it.

Moving to the view package, we found those classes more challenging to implement because identical exceptions were thrown out every time we tried to specify the root for the window and the issue that caused this exception was always different.

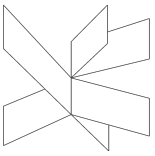


We also placed too many methods in the ScheduleModel interface that were already written in other classes and accessed through getters written in the ScheduleModel. That was additional and redundant work that we made, however, it taught us that the purpose of the ScheduleModelManager is to have access to instance variables, or its copies, from other classes and through them use methods that were linked to them.

We haven't finished our project as we planned. We are missing the Danish version of the website, teachers' individual schedules still require improvements to display all of the sessions one is teaching, because right now we manage to only display teachers' lessons in one class. Students are not accessing the timetable through VIA ID, instead, we can search for it using the name of the class. However, we are still very happy with the project results.

We identify that merging our changes on GitHub was a huge risk. Many times when someone added changes to our code, previous progress was being deleted and that resulted in wasting time to recreate lost improvements. When it irritated us long enough we searched for a solution on the internet and discovered that when we have any conflicts while importing a new piece of code, we can compare it to the previous code and only add changes without canceling other members' progress.

If we were to start coding our system all over again we would definitely first focus on critical priority requirements and then move to high priority and leave low priority until the end. We did not pay enough attention to the order in which we implemented methods, that is why in the end we are left with some critical priority requirements touched, but not fulfilled.



6 Personal Reflections

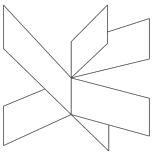
Robert:

I was very optimistic about my group at the beginning of the semester. When drafting the group contract, we all had similar goals: attending all classes together (not just SEP), meeting every week at least once, helping each other with our knowledge gaps. By now, two days before the deadline of the project, I can confirm that I am more than satisfied with my group's performance.

We have been meeting up after every SEP class to work together on diagrams and reports, DMA and RWD assignments, and never left anything to be done last-minute. There were some absences but each group member made up for it on the previous or next days. On my behalf, I was working one day per week; I managed to book most of my shifts for the weekends, but whenever there was a day with no classes or no in-class attendance I took advantage of it and went to work instead. In such cases, we have agreed with my group to be flexible and assign tasks to work on when one of us would not come to the meetings.

I think the sense of responsibility was visible in all of us: all group members were proactive and assigned tasks for themselves, and proudly presented their completed tasks at the beginning of the next meetings. I was mostly focused on documentation, more complex java methods, and helping the other members whenever needed.

In terms of communication, we were engaged with each other on a daily basis. We have used Facebook messenger to communicate absences in a timely

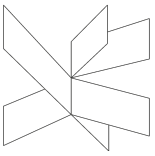


manner or discuss meetings outside of university (going out together for dinner or drinks, cooking together, playing board games). From a professional point of view, we had several text channels on Discord for different courses, topics, tasks, booked rooms, SEP documents on GoogleDocs, and repositories for each RWD assignment, SEP project, and SDJ exam preparation on GitHub. These tools have really increased our effectiveness in communication, however, I believe that there is still room for improvement. None of us had any experience using GitHub, and we did have some difficulties, for example accidentally deleting 4 hours' worth of group work in the Project Period.

I have never been part of such a motivated study group - I believe we have often positively impacted each other's motivation. None of us felt motivated every day - it has occurred several times that we raised each other's spirits and thus increased productivity and created a supportive and pleasant atmosphere in the group.

When it comes to the study atmosphere, I have found it to be really enjoyable and productive. Working together every day in the project period made us bond much better. We made lots of jokes, and whenever things got out of control, the most mature member of our group, Chris, helped us get back on track to focus on the tasks at hand. Regarding cultural differences, I have learned many interesting facts about both Danish and American culture from Chris and Martin. On the other hand, we could relate very easily to each other with Nina and Kamil thanks to the similarities between Polish and Romanian cultures.

In my opinion, there are several advantages to group work and problem-based learning. If I had to work on my own I would not have been able to deliver such a complex system. We did not individually work on all parts of the project but we explained our parts to each other such that we have gained an understanding of

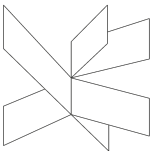


different parts of the project. For me, this was a great way of learning new things, as when it comes to the View Classes in Java, or displaying table elements in JavaScript it seemed a bit foggy to me. To be honest, JavaScript still does, so that is something I still have to look into in the future.

Having had to solve a real-life problem helped me understand why the project description, analysis, and design are an essential part of the project, especially during the project period. I would like to put more time and effort into the Project Design for my next semester project, as we had some parts that had to be designed during our implementation.

The supervisor meetings were very helpful as we gained new insight into our project that helped simplify a lot of things in our implementation. In the future, I would like to prepare better as a group for these meetings by structuring our questions and using the time more efficiently.

I am definitely looking forward to collaborating with the same team in the next semester. Hopefully, already knowing each other will help us further improve our group dynamics.



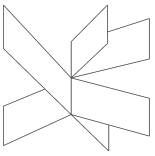
Martin:

My experience with this SEP group has been great because everyone did their part and we all got along. Whenever we disagreed on something, we would settle it with a vote. I think the SEP project was a great way to meet new people and get valuable experience working with others.

The content of the group contract that our group created was about group expectations. The expectations were that we did not have more than 3 absences, we meet one time a week, and help each other with knowledge gaps. I think our group did a great job following the expectations laid out in our group contract because no one had too many absences. Everyone was always on time for our group meetings and we also ended up meeting two times instead of the one time that was in the contract. We also helped fill in each other's knowledge gaps whenever someone was stuck on something.

I felt responsible for the group project because I had to write the code that read in the text files for the students, courses, and rooms. This was a key part of our assignment because we had to convert these files into room objects, course objects, and student objects which we would need to create our timetable. Without this code, we would not be able to make a functioning timetable. This put a lot of pressure on me to get the work done in a timely manner and to get it done right because the whole project depended on it.

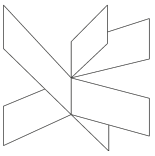
I believe that the group contract has had a direct impact on the group's cooperation. Our group contract helped set some ground rules on how we should work together and what to do if there were any disagreements or problems and we relied on it to get through any obstacles we had. One of the



biggest impacts was our time requirements that we had in our group contract. In our group we followed it really well and no one was unreasonably late or skipped any of our meetings without a good reason and no one broke the three absence rule. Some adjustments that I would suggest for the next group contract would be to put in a requirement to have to merge in GitHub in a predetermined way to avoid losing code. However, it was a really helpful tool that we all learned to use so that we could work together more effectively and I believe that it should be a requirement.

The teamwork in our group went really well. We all worked together with almost no issues and if there was one we would settle it with a group vote. We worked together by dividing into teams so that we could work on multiple parts of the project at the same time and then when we were done we would group up and go over what we did and explain it to the rest of the group. Everyone in the group contributed satisfactorily. Everyone completed their tasks and had a big impact on our project. We could not have done it without each member's help. Everyone did their best and delivered to a maximum to the group and often went the extra length to learn something new that we could use for the assignment. An example of this was when they learned how to use Gridpane from JavaFX for the schedule or when we all learned how to use Github so we could share our work with each other. Everyone's expertise was utilized when we assigned tasks.

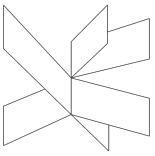
Throughout these two and a half months of project work our motivations have varied but they have always been high. We were motivated to do well on this project because we wanted to create a working product and it was a team assignment so our work had a direct impact on others. When we first started the project we were highly motivated and also overly ambitious which led to us



making a lot of requirements for our time table. This became a challenge to meet all of the requirements we set and was a little demotivating since some of the requirements were not finished as well as we would have liked. An example of this is how we wanted to make both the planning tool and the website have the ability to display in Danish but we were only able to get it on the planning tool.

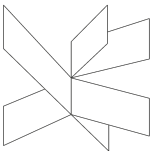
Our group was multicultural which I personally found really cool. I benefited from this because I got to learn a lot of things about their cultures and the countries they come from. One issue I had with being in a multicultural group was that even though they all spoke English, they were not as used to it as I was. This was a small problem at first because I often had to repeat myself because I spoke too fast for them to follow but by the end of our project period they were all able to understand me easily. During this assignment I learned that I am able to work well in a group context. Being split into different teams and working on tasks with different people really helped me learn how to be a team player.

There were many advantages to being in a group. Having a group that worked together helped us get many different views on how to solve a problem as well as letting us divide the tasks so that we could finish in a timely manner. Group work went really well with problem-based learning because we all sat down together and analyzed the problem and figured out how we could solve it. Then we broke it down into tasks to assign each of us. A disadvantage of group work that I experienced was that sometimes you can not move forward with a task until another member of the group finishes their tasks since all of our code works independently. We did a great job with the problem formulation. Chris



turned Bob's video of him talking into text in a google doc so that we could go over it thoroughly and figure out all the features that Bob wants in the timetable.

For this project we had Steffen as our supervisor. I was very satisfied with the cooperation from him. Every meeting we had with him he made sure to answer all the questions we have and to clarify anything we were unsure of. He always had time for us and was a great help. We used our supervisor for determining which features the client wanted in the timetable, how to format a sequence diagram, and to go over our class diagram. We handled the cooperation and communication with our supervisor through two meetings here at Via University and two meetings online through discord.



Nina:

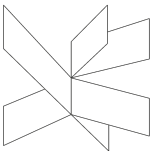
Generally, I felt responsible for the atmosphere of our group meetings and for the tasks assigned to me during discussions. As an example, I have built most of the windows in Scene builder.

I find the group contract necessary to state the rules in the group, however, after creating it I don't remember a day when we were going back to it. I believe that we were so devoted to doing the best job we can, we simply followed the rules without hesitating. For the next group contract, I would only like to add that everyone can work from home as well or work in different hours than previously stated as long as he finishes all his tasks.

Our group worked great together. Each member made significant input into the project. Robert, for example, did amazing work with the documentation, Chris was helping with every issue regarding coding and documentation we had. Kamil created a great website and Martin did a huge part of the Java code. These were all fields that lay in each member's strongest skills.

The motivation that drove our group was to get a 12 on the SEP exam. Curiosity played an important role as well. However, it was the willingness to pass the exam with a high grade that kept us going when failures demotivated us. I found out that I was not as good at cooperating with other people as I thought. During this project, I discovered that sometimes it is much easier to solve problems along with a colleague instead of struggling alone. Solutions come faster and both work and success are much more enjoyable.

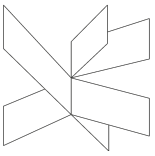
In my opinion, the greatest advantage of problem-based learning is widening the area of work and discovering different solutions to one problem. This



increases interest in the field because you do not need to follow any patterns and instead you are free to discover new, different ways to reach the goal.

The only disadvantage I can think of is when you are stuck in the early phase of the task and struggle to move forward. You can easily get lost in the sea of information and don't find the answer to your question.

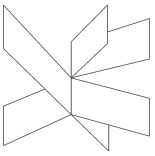
All in all, it was a pleasant experience to work with those easy-going and hard-working men. I learned a lot from them and I am looking forward to collaborating with them in the future.



Kamil:

I joined this group after the autumn break because I was not satisfied with the performance of my previous group. I just felt like I wanted more and my expectations were on a different level. Luckily, the idea of changing a group came to my mind. Due to being friends with Robert, it was possible to ask and convince the other group members to accept me inside this hard-working and ambitious team. I am quite sure that both being active in the lessons and easy-going nature helped it. I knew that I would probably have to work harder than I did, but at least I also knew that there are people that I can rely on and there is no need to check their work. I believe that I brought some more energy to the group and, what is even more important, increased the quality of our work. Besides that, throwing some jokes or comments into the conversation is truly my thing which I did during the semester. Hopefully, this made our meetings more cheerful and enjoyable without the loss of workflow and focus. The spirits have been lightened up!

All of the rules written down in the group contract are completely relevant and during this project, they were obeyed by all of us. This includes helping each other with knowledge gaps, for instance, when someone missed a DMA lesson we were eager to explain to him what it was about and help to understand the way that this is done. Moreover, we listened to each other's opinions and considered them later on. Everyone was treated with respect and care. The delays were accepted in a good manner. If there was some personal issue, staying at home or not showing up was not a problem. All meetings were held on the campus as agreed. To my mind, it is a great idea because in that way we distinguish our home life from studying. However, there were some days that we

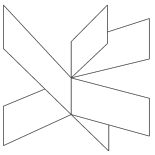


met on Discord after agreeing to it. For example, there was this one day when it was raining in the morning so badly that leaving your house and getting to the university seemed like either an impossible or a “suicidal” mission. In the group contract, we also agreed that during the semester project period we will meet every day from 9 to 16 and that was fulfilled. I am really grateful for doing that because it made the work just less stressful. It also helped my soul when everything was more or less planned and every day we made small but visible progress.

Considering the work, everyone did their best in order to contribute to our final result. When one of the members had a bad day we were trying to help and motivate him. The team’s motivation came from the desire to deliver the possibly best product and later get the highest grade. Everyone had it in their mind during this semester. Our tasks were divided equally and we tried to give everyone a chance to work at the specific field of studies at one point or another. After finishing crucial parts the progress and work done had been presented and explained. It had a great impact on allowing us to keep up to date with all the stuff that was happening.

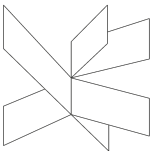
Group work is beneficial in a way that we get to cooperate with other people from different backgrounds, cultures, and work attitudes. This makes the time more interesting and enjoyable. To give an example, I just loved the stories told by our Danish-American friend Chris about life in Colorado, serving as a highly respected waiter in great restaurants or some other “American-dream” related stuff. In the end, having a chance to meet people from all over the world just made me realize that we are all, of course in a positive way, the same.

Talking about the courses themselves, I would definitely present the design part of SEP much earlier. It would have allowed us to start the work on class



diagrams earlier so we would have more time both to implement and analyze them more deeply. The programming part of the DMA seemed too challenging, to my mind some of the exercises should be easier or even much easier. There should also be some time allocated for showing the correct way to do them and explaining them. Besides that, SEP learning paths were too long sometimes.

To summarize things up, I am truly grateful to be in this group and this project allowed me to gain knowledge and practice material learned throughout the semester. I am sure that we will stay in the same team next semester, at least I hope for the best!



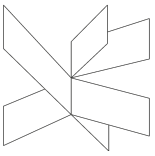
Chris:

When I get a chance to talk to other students from different departments in VIA and we talk about projects, one question is often asked of me: you get to choose your groups? Being able to craft a group for the project is a great privilege that I really enjoyed. As the first group to be added to the list when teams were forming, this had a large impact on us.

Throughout the year before arriving in Horsens, I went through the course descriptions for the courses we would need to take during the first semester. While reading through the material VIA has to offer on their website a few things became clear. Project based learning has a large impact on what is taught and how students are evaluated. As part of it, groups would deliver projects every single semester. As such, drafting team members would be critical to success and secretly began during the study start.

From the study start, I managed to rope Martin into the nonsense that eventually became Group 1. He had a noticeable Californian accent and we were able to connect on a surface level on the basis of nationality. Throughout the start of SDJ, we worked together on coding and I noticed that we work well off of each other. Martin is great at writing the base code, and then when something doesn’t work, we work well together in figuring out what to try. Most importantly we both work on the principle of isolating the problem.

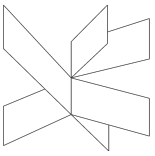
Although it takes two to tango, it takes four to make a semester project. I first met Nina and Robert when I was grouped with them as we worked on the UN Sustainable Development Goals. To oversimplify the dynamic for the sake of brevity, Nina provided an alternative angle, then Robert expanded and applied. I



will be the first to admit that all of my ideas are not always excellent. A brief stroll through my old photos is evidence enough. However, Nina always has the gumption to challenge an idea if she visualizes something differently. This has been a great boon to the group as our democratic ideals allow us to consider alternatives with consensus being important for what is implemented. On the other hand, Robert is able to take those decisions and predict roadblocks that can occur five steps down the line. He has been great at seeing the big picture, and at the same time, understands how each of the smaller cogs fit into the larger machine.

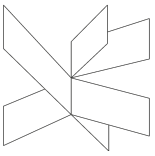
Kamil asked to join the group shortly after the autumn break and I remember being conflicted. I knew he would make for an excellent member from the times I had talked to him when I had bugs in class, but I had also just told another person that our group was full. I decided to again follow group consensus, and we unanimously wanted him to join. After joining, he had this look of a great weight being lifted off of his shoulders. From my understanding, it seemed he knew that he wouldn’t be alone in doing the project, and we were all happy to have him.

With a larger group than before, I tried to reconsider my position in the group. Through my years of work in different industries, I had always been happy to take the roles other people did not want. In the group, I wanted to take this role again, but since everyone was so ambitious there were rarely tasks that were absolutely avoided. So instead, I tried to apply some lessons I have learned from leadership positions that I had at some jobs. I wanted to make sure that we made good use of our time by reigning in the discussion when we would get stuck. I also wanted to make sure that group members felt that their hard work was recognized.



At some point during my education at VIA, I would be greatly honored to take on the challenge of being a project manager. I find the challenges of leadership to be enticing and enjoyable. As such, I would like to improve some aspects that make for a good manager. First, I think that a manager should be flawless in attendance to show dedication. Although I didn't go over the contract's limit, I could have been at the meeting each time. In addition, I missed a few classes during the tuition period that I regret not being present for. Secondly, managers should make themselves available to assist others. I could have dropped what I was doing in order to help those struggling with a task.

All in all, I am very proud of the work the group has made. It brings me great joy to be a part of a group that I am certain will continue to succeed through the coming semesters.

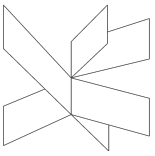


7 Supervision

We had several meetings with supervisors, 3 of them were held during the project period. They were usually very helpful and we were definitely the most efficient when we had prepared the agenda for the encounter before. We definitely agree that they have always helped us to get us back on track when we felt lost during the process. The guidance was necessary but we also felt that supervision wasn't overwhelming. Supervisors gave us a lot of autonomy which positively affected our teamwork.

The approach they have taken was clearly inspired by the Laissez-faire leadership management style. According to the article Team Forming Team Initiation (Kick Off), "While 'laissez-faire' implies a completely hands-off approach, many leaders still remain open and available to group members for consultation and feedback. They might provide direction at the beginning of a project, but then allow group members to do their jobs with little oversight" (Lean Construction Institute, 2021). This approach has allowed us to make our own decisions without pausing the work to wait for approval.

We were satisfied with the cooperation with our supervisors mostly as a result of receiving feedback before handing in necessary documents and being able to consult our code issues during the project period.

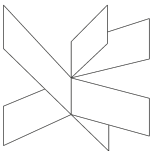


8 Conclusions

In Summary, we believe that having a fixed working schedule has been beneficial for the whole group in a way that kept us motivated and consistent. We will still keep working on GitHub because it fastens the process of sharing files we are working on, however now we will make sure that only one person is working on a class or a file at a time. We incorporated a workflow that was based on dividing our team into sub-groups and dividing tasks between them. Additionally, it requires regular meetings to present, discuss and merge the work. This approach made us learn from each other and improve our overall skills. We felt that this suits us the best regarding time-effectiveness and quality of the work.

At the beginning of the project, we ordered a list of tasks to do during the implementation phase. However, as mentioned in project execution we failed in following the hierarchy. We have to pay more attention to that during our next assignment.

Regarding activities we had to do apart from coding itself were written down in the excel file, nevertheless, we have not paid much attention to that. This time it hasn't affected our outcome, because we covered every relevant task looking through all the guidelines uploaded on the VIA University College website. Regardless, we strongly believe that preparing a checklist at the beginning of the project with all requirements we have to cover would result in a calmer and less chaotic documentation phase.

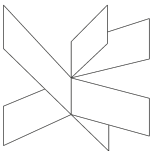


9 Resources

Adobe Workfront. 2021. *Waterfall Methodology - A Complete Guide*. [online] Available at: <https://www.workfront.com/project-management/methodologies/waterfall> [Accessed 15 December 2021].

Lean Construction Institute. 2021. *Team Forming Team Initiation (Kick Off)*. [online] Available at: <https://leanconstruction.org/uploads/wp/media/learning_laboratory/Project_Kickoff/Team_Forming_Team_Initiation.pdf> [Accessed 15 December 2021].

Niemiec, C. and Ryan, R., 2009. Autonomy, competence, and relatedness in the classroom. Theory and Research in Education, 7(2), pp.133-144.



Appendices

Meeting Minutes 1

04/10/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

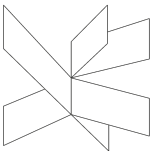
Agenda

Main topics for the meeting

- Generate questions for Bob
- Start the problem description
- Schedule time with supervisor to forward questions to Bob
    - Check in tomorrow with Steffen and see if there is time in class

Discussion

- Questions for Bob



- Access modifier?
- Wishes incorporated?
- How are classes found and defined?
- Danish version?
- Should it read the file directly?
- All rooms and combinations given?
- How do students find which class they are in?

· Refining questions

- Questions were grouped together in categories
- Robert wrote the questions on his computer and we discussed
- Formatted the file so answers can be inputted

· Problem description

- Watched the intro to the video
- Looked at a few alternatives
- Brainstormed some ideas
    - § Robert wrote down

Tasks

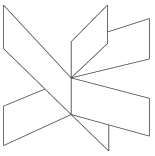
· Schedule supervisor meeting for project period

- Nina to Mona

· Ask Steffen the questions about Bob

- Everyone

· 2 sources each for background description



- Everyone

· Upload meeting minutes

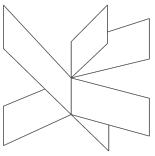
- Chris

· Finish video

- Everyone

· Do the problem description together on Wednesday

- Everyone, next meeting



Meeting Minutes 2

06/10/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

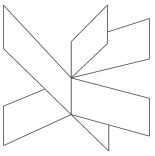
Agenda

Main topics for the meeting

- Generate questions for Bob
- Start the problem description
- Schedule time with supervisor to forward questions to Bob
    - Check in tomorrow with Steffen and see if there is time in class

Discussion

- Questions for Bob
    - Access modifier?



- Wishes incorporated?
- How are classes found and defined?
- Danish version?
- Should it read the file directly?
- All rooms and combinations given?
- How do students find which class they are in?

· Refining questions

- Questions were grouped together in categories
- Robert wrote the questions on his computer and we discussed
- Formatted the file so answers can be inputted

· Problem description

- Watched the intro to the video
- Looked at a few alternatives
- Brainstormed some ideas
  - § Robert wrote down

Tasks

· Schedule supervisor meeting for project period

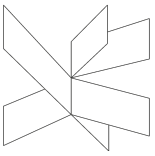
- Nina to Mona

· Ask Steffen the questions about Bob

- Everyone

· 2 sources each for background description

- Everyone



- Upload meeting minutes
    - Chris
- Finish video
    - Everyone
- Do the problem description together on Wednesday
    - Everyone, next meeting

Meeting Minutes 3

08/10/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

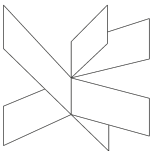
Agenda

Main topics for the meeting

- Watch project description video
- Work through chapters of project description
    - Assign
- Review Sources that we found
- Summarize and categorize answers from Bob

Discussion

- Bob’s answers



- Collect info about rooms
    - § Property if rooms can open
- Research file input java
- Agreed to do website at the end of the project

· Background Description

- Made bullet points to write into draft
- Defined our in-text citations
- Avoid solution based writing
- Discussed current market solutions
- Discussed sources

· Problem Statement

- Summarized the issue
- Avoided solution based writing

Tasks

· Ask for milestone deadlines in class Thursday

- Everyone

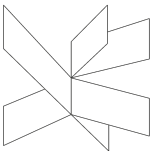
· Schedule supervisor meeting for project period

- Nina to Mona

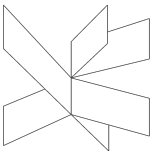
· Work on the Project Description Thursday in class

- Everyone

· Chapters divided



- Delimitation
    - Nina
- Risk Assessment
    - Martin
- 1st Draft of background description
    - Chris
- Definition of purpose
    - Robert
- Refine the problem description together on Wednesday
    - Everyone, next meeting
- Research waterfall, Kanban, and SCRUM (optional)
    - Everyone
- Add appendix in class
    - Group contract
    - Everyone
- Reformat sources
    - Everyone
- Upload meeting minutes and 1st draft of project description
    - Chris



Meeting Minutes 4

10/10/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

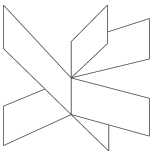
Agenda

Main topics for the meeting

- Refine and combine html documents
- Distill layout requirements
- Check on group 6 status
- Learn more about GitHub
- Improve html documentation

Discussion

- Previous class during RWD



- Created wireframe
- Reviewed functional and non-functional requirements for assignment 1
- Started creating web pages

· Group 6 project description status

- Not sent to email yet

· Reworking layout of DMA page

- Placing picture to the right
- Teacher below
- Fixing borders

· Contact us page

- Adding web optimized images
- Using table to position things

· Committing changes to GitHub

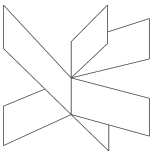
- Learning to utilize pull requests
- Uploading files
- Version control

Tasks

· Ask for milestone deadlines in class Thursday

- Everyone

· Schedule supervisor meeting for project period



- Nina to Mona

· Add footer to each page

- Martin

· Multiply DMA to other classes

- Nina

· Touch up front page layout

- Chris

· Work on GitHub

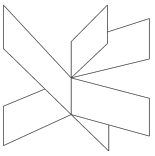
- Everyone

· Check requirements to current page

- Robert

· Validate current code

- Chris



Meeting Minutes 5

15/10/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

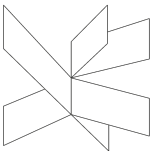
Agenda

Main topics for the meeting

- Squash issues from GitHub
- Refine assignment 1 layout
- Review project description

Discussion

- Bugs
    - Copyright in footer
    - Multiple teacher layout



    - Subtitle issues
    - Validator errors
    - Color scheme of pages
- Project Description
    - Trim background
    - Update risks
    - Delimit delimitations
    - Update table of contents
- Hand in
    - Project description
    - Assignment 1

Tasks

- Ask for milestone deadlines in class Thursday
    - Everyone
- Schedule supervisor meeting for project period
    - Nina to Mona
- Watch GitHub videos
    - Everyone
- Enjoy break
    - Everyone
- Prepare for Java midterm
    - Everyone

Meeting Minutes 6

27/10/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

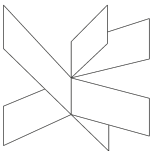
Agenda

Main topics for the meeting

- Define requirements
- Making the use case diagram

Discussion

- Define the actors
    - Planner, teachers, students
- Nina reads the transcript
    - Requirements are listed



- Chris takes over the reading
- Discussing the requirements as they arrive

· Reading the questions, which we asked, again

- Individual timetables for teachers
- Danish version
- Acceptable hours

· Writing the requirements formally

- Meet the template
- Order in priority

· Make use case diagram

- Grouping items together
- Linking different items
- Managing layout

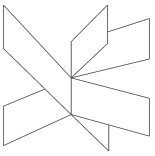
Tasks

· Schedule supervisor meeting for project period

- Nina to Steffen

· Arrive tomorrow at 9 am

- Everyone



Meeting Minutes 7

28/10/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

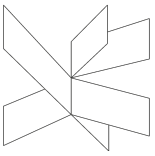
Agenda

Main topics for the meeting

- Write use case descriptions
- Relating

Discussion

- Write use case descriptions
    - Defining what needs to happen for different actions
    - Defining pre and post conditions
- Relating use cases to requirements

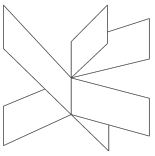


- Going through to refine
- Updating the use cases

.

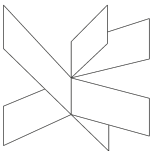
Tasks

- Schedule supervisor meeting for project period
  - Nina to Steffen
- Catch up on Discord during the weekend
  - Everyone
  - Sunday evening 6PM
- View schedule use case diagram
  - Martin
- Update the timetable diagram
  - Robert
- Update the asta diagram
  - Robert
- Rewrite update description
  - Robert
- Create a timetable diagram + view the test week
  - Nina
- Send wishes + publish the timetable
  - Chris
- Domain model



    - Martin and Chris

- Book room for Monday at 2:30 PM
    - Robert



Meeting Minutes 8

1/11/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

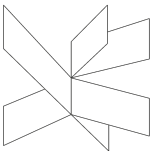
Agenda

Main topics for the meeting

- Going over requirements of Assignment 2
- Making wireframes for different sizes

Discussion

- Do we remake the whole website?
    - Bootstrap the table
    - Placing the nav bar over the other pages
- Requirements
    - Requirements from assignment 1



- Collapsible nav bar
- More than 1 breakpoint
- Carousel
- Grid on Bootstrap

· Wireframes on Balsamiq

- Each page first
- Then, medium and small
- Making footer smaller
- Including the nav bar on each page
- Make nav bar sticky on small screens
- Only one day on the page

Tasks

· Schedule supervisor meeting for project period

- Nina to Steffen

· Course page

- Nina

· Schedule

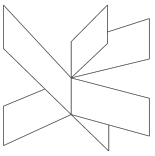
- Robert

· Front page

- Chris

· Contact us

- Martin



- Pages
    - First draft is due Wednesday
- Read and comment on the other group's analysis
    - Everyone

Meeting Minutes 9

11/11/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

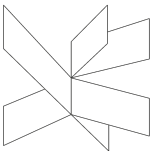
Agenda

Main topics for the meeting

- Discuss team reflexivity assignment
- Meet with Mona
- Collaborate on DMA assignment

Discussion

- Team reflexivity
    - Discuss again in person
- Upload motivation document to resources upload page



    - Looks completed, send email to Mona
- DMA
    - Talk about Hash set
    - Implement retain all
- Review SEP analysis
    - Add domain model to table of contents
    - Line up black circles

Tasks

- Finish DMA
    - Everyone
- Revise SEP analysis
    - Everyone

Meeting Minutes 10

14/11/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

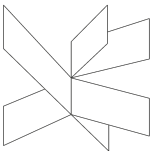
Agenda

Main topics for the meeting

- Discuss team reflexivity assignment
- Meet with Mona
- Collaborate on DMA assignment

Discussion

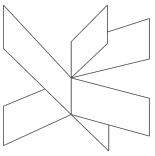
- Team reflexivity
    - Discuss again in person
- Upload motivation document to resources upload page



    - Looks completed, send email to Mona
- DMA
    - Talk about Hash set
    - Implement retain all
- Review SEP analysis
    - Add domain model to table of contents
    - Line up black circles

Tasks

- Finish DMA
    - Everyone
- Revise SEP analysis
    - Everyone



Meeting Minutes 11

23/11/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

Kamil Fischbach

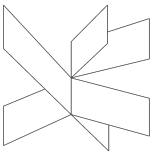
Agenda

Main topics for the meeting

- Discuss current SEP progress
- Structure project period
- Prepare supervisor meeting
- Turn in Secret Garden

Discussion

- Current SEP progress



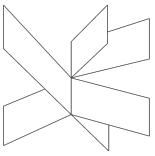
- Working through class diagram
- Making associations proper
  - § Deleting variable in attribute compartments
- Catching up on current work
- Splitting up to tasks below

· Schedule project period

- Major tasks remaining
  - § Process report
  - § Project report
  - § Make the UI
    - · Conceptualize it
    - · Diagram it
    - · Write the controllers
  - § Make the website
  - § Make the Javadoc for each of the classes
  - § Finish model classes
    - · Compare to UML
  - § Redo the structure diagram
    - · Include view

Tasks

· Coding (Session, TimeInterval)



    - Kamil, Martin
- Diagram
    - Nina
- Book room sequence diagram
    - Delayed
- Secret upload
    - Robert

Minutes 12

24/11/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

Kamil Fischbach

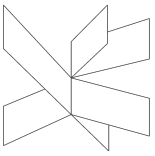
Agenda

Main topics for the meeting

- Fix RWD Assignment 3 bugs
- Start DMA Hand-in 4

Discussion

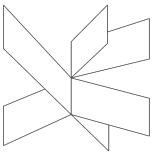
- Squashing bugs in the garden
    - Apples don’t load on the crown of the tree for different screen sizes
    - Sun doesn’t load correctly



- Net can’t be uncharged

Tasks

- Requirement rewording
  - Nina
- Domain Model revamp
  - Martin
- Add session use case
  - Kamil
- Add session diagram
  - Robert
- Revisions and assistance
  - Chris



Meeting Minutes 13

29/11/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Martin Rosendahl

Kamil Fischbach

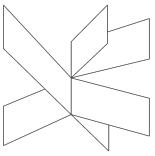
Agenda

Main topics for the meeting

- Discuss RWD and DMA
- Structure Diagram
- Class Diagram

Discussion

- Discuss RWD and DMA
    - Talk about comments
    - Update issues on GitHub



    - Getting charge button to work
- Structure
    - Converting from domain model
    - Looking at presentation
    - Boxing different packages
- Class Diagram
    - Attributes
    - Operations
        - § Which class should do this?
        - § Return types
        - § Parameters
    - Associations
    - Generalizations
        - § Sub and super classes

Tasks

- File Reader and Writer (attempt to start, finish not needed)
    - Martin
- Model and Manager
    - Nina
- Lists and Sub classes to room
    - Chris
    - Can reassign

Meeting Minutes 14

30/11/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

Kamil Fischbach

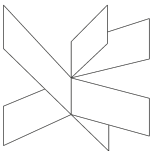
Agenda

Main topics for the meeting

- Discuss current SEP progress
- Structure project period
- Prepare supervisor meeting
- Turn in Secret Garden

Discussion

- Current SEP progress



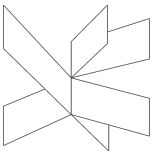
- Working through class diagram
- Making associations proper
  - § Deleting variable in attribute compartments
- Catching up on current work
- Splitting up to tasks below

· Schedule project period

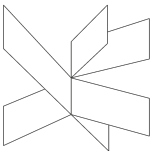
- Major tasks remaining
  - § Process report
  - § Project report
  - § Make the UI
    - · Conceptualize it
    - · Diagram it
    - · Write the controllers
  - § Make the website
  - § Make the Javadoc for each of the classes
  - § Finish model classes
    - · Compare to UML
  - § Redo the structure diagram
    - · Include view

Tasks

· Coding (Session, TimeInterval)



    - Kamil
- Diagram
    - Nina
- Book room sequence diagram
    - Delayed
- Secret upload
    - Robert



Meeting Minutes 15

06/12/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

Kamil Fischbach

Agenda

Main topics for the meeting

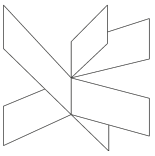
· Update on progress

· Debuggin

Discussion

· Project progress

- Schedule shows some text!
  - § However, we need to switch from table view to grid pane

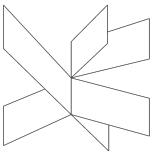


        - Unable to select individual session in table
    - Reader works, and upload files view works
        - § The two need to be connected

- Debugging
    - Adding in Martin's branch
        - § Making Main open again
        - § Making parameters match
            - File input
    - Resetting the add session page
    - Courses not connected to classgroup
        - § Set up a switch to read in the classgroup to assign it
        - § Avoid creating a new classgroup
    - Gridpane not showing courses
        - § Added constructor
        - § Added a label to add to the gridpane

Tasks

- None



Meeting Minutes 16

07/12/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

Kamil Fischbach

Agenda

Main topics for the meeting

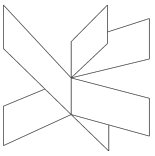
· Update on progress

· Debuggin

Discussion

· Project progress

- Schedule shows some text!
  - § However, we need to switch from table view to grid pane



        - Unable to select individual session in table
    - Reader works, and upload files view works
        - § The two need to be connected

- Debugging
    - Adding in Martin's branch
        - § Making Main open again
        - § Making parameters match
            - File input
    - Resetting the add session page
    - Courses not connected to classgroup
        - § Set up a switch to read in the classgroup to assign it
        - § Avoid creating a new classgroup
    - Gridpane not showing courses
        - § Added constructor
        - § Added a label to add to the gridpane

Tasks

- Non

Meeting Minutes 17

10/12/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

Kamil Fischbach

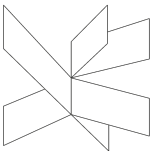
Agenda

Main topics for the meeting

.

Discussion

- Project progress
    - Schedule shows some text!
        - § However, we need to switch from table view to grid pane
            - Unable to select individual session in table



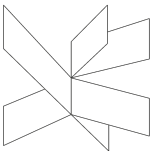
- Reader works, and upload files view works
  - § The two need to be connected

· Debugging

- Adding in Martin's branch
  - § Making Main open again
  - § Making parameters match
    - · File input
- Resetting the add session page
- Courses not connected to classgroup
  - § Set up a switch to read in the classgroup to assign it
  - § Avoid creating a new classgroup
- Gridpane not showing courses
  - § Added constructor
  - § Added a label to add to the gridpane

Tasks

· Non



Meeting Minutes 18

13/12/2021

Meeting Notes

Attending

Christian Foyer

Nina Wrona

Robert Barta

Martin Rosendahl

Kamil Fischbach

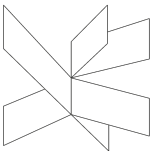
Agenda

Main topics for the meeting

- Reconsider diagrams
- Fix color of boxes
- Read through requirements

Discussion

- Requirements
    - Students should not search by VIA ID



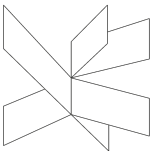
- Exception for when room is not available
- Finish teacher
- Teacher not available should show as error label instead of halting program
- Set the first error label to choosing a teacher or a class
- Place xml files manually for website
    - § Clarify in user guide
- Fix color for holidays on schedule
- Foldable rooms
    - § Set Boolean for opened
- Extend footer
- Comments for website from Line
- Integrate Danish
- Change error label to empty after files are uploaded

· Website comments

- Gradient borders
- Remove Halloween photos
- Margin on right of carousel for courses
- Mobile phone timetable
- Move css to bootstrap

· Nina and Martin get teacher to display sessions

- Make errors, fix errors, ya know?



Tasks

- Nina
    - Do the whole project
    - Non-negotiable (Signed 13/12)