HANOI UNIVERSITY

FACULTY OF INFORMATION TECHNOLOGY

----------&----------

JAVA SOFTWARE DEVELOPMENT

TIMETABLE SCHEDULER

Instructor: Mr. Ngo Van Quyen
Group members: Nguyễn Công Tuấn - 2101040201
Vũ Thị Hường - 2101040100
Mai Hồng Hạnh - 2101040082
Nguyễn Đức Toàn - 2001040206
Vũ Đặng Anh Quân - 20010401072

November 5, 2024

1. Introduction

Most of the educational institutions and students are facing problems in maintaining the schedules, which are mostly manual, labor-intensive, and highly prone to errors. In such a perspective, the project aims at designing a timetable management system; it would be a desktop-based application using Java Swing. The core objective is to offer a clear user-oriented tool for efficient organizing of scheduling classes, avoidance of clashes, and further smoothing the whole process of scheduling.

This reduces by a great level the amount of work needed where timetables are scheduled manually; hence, automating this will save on that. Since the development of the system requires interactivity and adaptability in graphical user interface creation, the potential of Java Swing was picked for usability even by unsophisticated users.

The proposal will further allow for future development in functionality of the system: for example, inclusion of room reservations and lesson planning. In brief, the system shall reduce time wastage by users and minimize cases of conflict and errors in maintaining schedules.

2. Related Work

Microsoft Teams

Compared to the calendar in **Microsoft Teams (2017)**, which focuses on general event scheduling and real-time collaboration, this timetable system is specialized for academic environments. Microsoft Teams handles recurring meetings but lacks features tailored for academic schedules, like automatic conflict detection and detailed management of recurring events. The timetable system addresses these needs, preventing double-booking and automating conflict detection, making it more suitable for educational use than Teams' calendar.

A Greedy-based Algorithm in Optimizing Student's Recommended Timetable Generator with Semester Planner

The use of a greedy algorithm in timetable optimization is explored in the work of Khyrina Airin Fariza Abu Samah et al., titled **A Greedy-based Algorithm in Optimizing Student's Recommended Timetable Generator with Semester Planner**. This study presents a solution to student scheduling problems during the COVID-19 pandemic by using a greedy algorithm to optimize the allocation of tasks in a student's schedule. The greedy approach, which makes locally optimal choices in the hope of finding a globally optimal solution, offers a straightforward yet effective method for solving scheduling problems.

In the context of the timetable management system developed in this thesis, the greedy algorithm presents an alternative to the brute-force method currently used for conflict detection. While the current system checks every possible occurrence of an event to detect conflicts, the greedy algorithm could optimize this process by prioritizing tasks based on certain criteria, such as time or resource availability. The studies reviewed in this section provide valuable insights into various approaches for optimizing timetable generation and conflict detection. The use of greedy algorithms presents a viable alternative to the brute-force approach, offering a way to enhance the system's performance in generating optimal timetables. By drawing on these works, the system developed in this thesis can continue to evolve, incorporating more advanced algorithms to meet the growing needs of educational institutions.

3. **Methodology**:

In developing the timetable management system using Java Swing, a structured and straightforward approach was followed to ensure the project was completed efficiently. The development process was divided into several key phases: problem analysis, system design, implementation, and testing. These stages were essential in addressing scheduling conflicts, providing an easy-to-use interface, and optimizing the system's performance.

Tools and Frameworks

- IDE: The project was developed using IntelliJ IDEA Ultimate, a comprehensive integrated development environment (IDE) that supports Java development with features such as code assistance, refactoring, and Maven integration.
- Maven: Apache Maven was used to manage dependencies and the project's build configuration. Maven's declarative nature helped streamline the project's lifecycle and manage necessary libraries efficiently.

Libraries and Dependencies

- MySQL Connector: The system uses the `mysql-connector-java` library to connect to a MySQL database. This connector allows for smooth interaction between the Java application and the database, facilitating the storage and retrieval of timetable data.
- JCalendar: The system employs JCalendar, which provides a user-friendly graphical interface for date selection. This enhances the usability of the system by allowing users to quickly select dates from a calendar instead of manually entering them.
- iText PDF Library: The system integrates the itextpdf library, enabling the creation and manipulation of PDF documents. This library is crucial for generating reports or exporting timetables in a PDF format, providing a reliable and flexible way to produce professional-quality documents directly from the Java application.

Algorithm Design

The conflict detection algorithm is responsible for ensuring that no two events overlap, particularly when handling recurring events. The algorithm uses a brute-force approach to compare each new event with existing events. For each occurrence, it compares the start and end times, ensuring that no two events are scheduled at the same time.

While this approach works effectively for smaller datasets, more advanced algorithms, such as genetic algorithms or greedy algorithms, could be introduced in future versions to improve performance when dealing with larger schedules. By leveraging these advanced techniques, the system will be able to handle more complex scheduling scenarios efficiently.

By integrating powerful tools like Java 8, MySQL, and JCalendar, the system provides a reliable and efficient solution for managing timetables, detecting conflicts early, and ensuring a smooth user experience.

This methodology provided a clear and organized approach to developing the timetable management system. By focusing on reducing manual effort, minimizing errors, and providing a simple, user-friendly interface, the project successfully met its objectives. Future improvements will focus on optimizing the conflict detection algorithm and expanding the system's functionality to include advanced features like room reservations and detailed lesson planning.This would reduce the computational complexity of the system, making it faster and more efficient in scenarios with large numbers of events. The integration of a greedy algorithm into the system would enhance its ability to generate optimal schedules quickly, especially in dynamic environments where schedules need frequent adjustments.

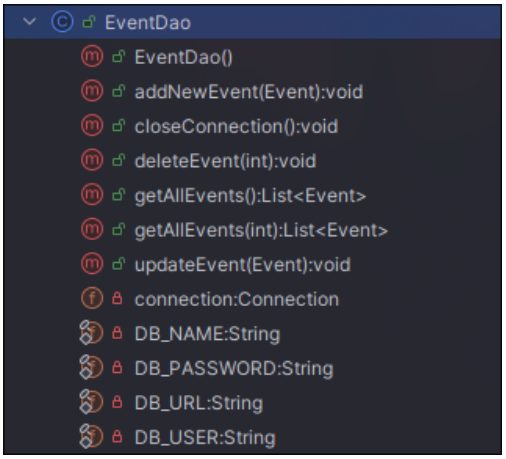
4. Implementation

In building the timetable management system, several Java techniques learned throughout the course were applied, with a particular emphasis on the **Model-View-Controller (MVC)** design pattern. The MVC pattern is a widely-used architectural framework that separates the application's logic into three interconnected components: the Model, View, and Controller. This division enhances modularity, maintainability, and scalability, making the system easier to develop and extend. Below, our group describes how each component of MVC was applied in the timetable management system using Java.

1. Model: Representing the Data Layer

The Model is responsible for handling the data and business logic of the system. In the timetable management system, the Model includes the classes and interfaces that represent the data entities, such as events, and handle interactions with the database. For instance:

EventDao.java: This class serves as the Data Access Object (DAO) for interacting with the database. It implements CRUD (Create, Read, Update, Delete) operations for managing events in the MySQL database. The DAO pattern separates the data layer from the rest of the application, promoting a clean separation of concerns.



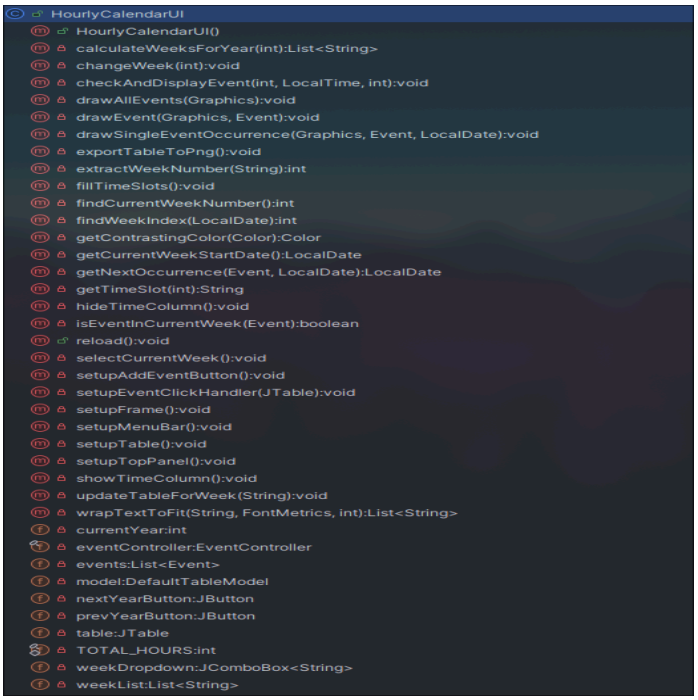
DatabaseInitializer.java: This class is responsible for initializing the database connection and ensuring the proper structure is in place. It manages the connection to the MySQL database using **JDBC** (Java Database Connectivity), providing the system with persistent storage for event data such as classes, rooms, and time slots.

By using JDBC, the system can efficiently handle the storage and retrieval of event data, ensuring that all schedules are saved and can be accessed whenever needed.

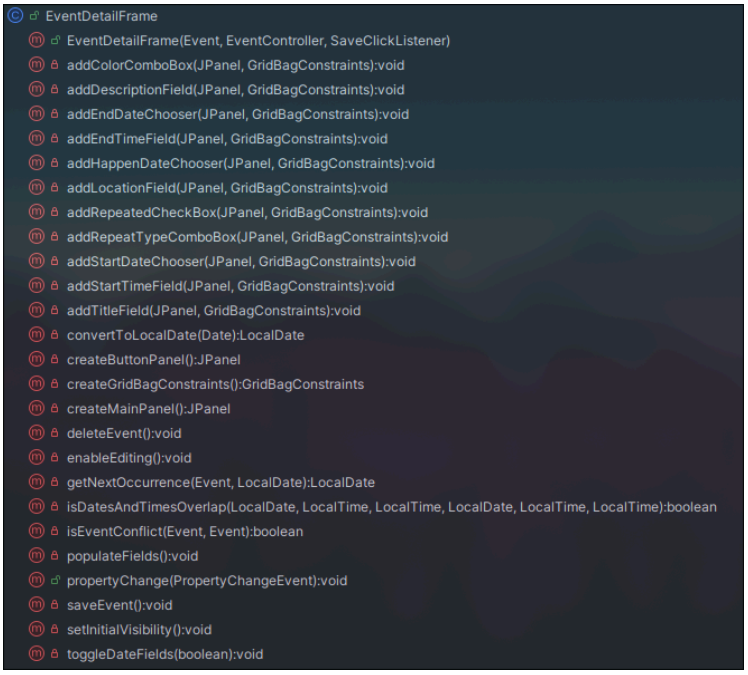
2. View: The User Interface

The View is responsible for presenting the data to the user and capturing user input. In this system, Java Swing was used to create the graphical user interface (GUI), making the system user-friendly and interactive. Various frames were developed to handle different parts of the scheduling process:

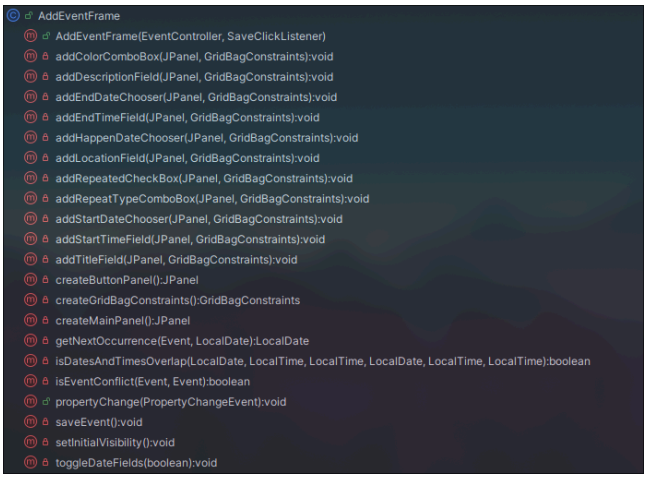
HourlyCalendarUI.java: This class implements the visual calendar view, showing the events in an hourly format. It allows users to see the entire schedule at a glance and visually spot potential conflicts. This component is crucial for users managing large, complex timetables, as it provides a comprehensive overview of the day's schedule.



EventDetailFrame.java: This frame is used to display detailed information about a particular event. It shows the user all relevant data, such as the event's start time, end time, and location, in a clean and accessible format.



AddEventFrame.java: This class is a Swing-based window that allows users to input details for new events. It provides input fields for event names, dates, times, and locations. The user-friendly design ensures that even users with little technical experience can easily add new classes or events to the schedule.



3. Controller: Handling User Actions

The Controller is the intermediary between the Model and the View. It processes user inputs, communicates with the Model to update data, and refreshes the View. In the timetable management system, the Controller coordinates between the GUI and the underlying data logic:

EventController.java: This class serves as the central controller, managing the flow of data between the UI (View) and the data (Model). It listens for user actions, such as adding or modifying events, and updates the data accordingly through the EventDao class. For example, when the user adds a new event, the EventController validates the input, checks for conflicts, and then updates the database via EventDao.

```java
package org.example.controllers;

import org.example.databases.EventDao;
import org.example.models.Event;

import java.util.List;

public class EventController {  9 usages  StebenC
    private final EventDao eventDao;  6 usages

    public EventController() { eventDao = new EventDao(); }

    public void addNewEvent(Event event) { eventDao.addNewEvent(event); }

    public void updateEvent(Event event) { eventDao.updateEvent(event); }

    public void deleteEvent(int eventId) { eventDao.deleteEvent(eventId); }

    public List<Event> getAllEvents(int weekNumber) {  1 usage  StebenC
        return eventDao.getAllEvents(weekNumber);
    }

    public List<Event> getAllEvents() { return eventDao.getAllEvents(); }
}
```

By separating the business logic from the UI logic, the **Controller** ensures that the system follows the MVC pattern, making it easier to maintain and update the codebase in the future.

Application of Course Techniques

Throughout the implementation, several Java techniques learned in the course were applied:

- **JDBC**: To connect and interact with the MySQL database, **JDBC** was used extensively. This allowed for seamless CRUD operations and persistent data management, ensuring that events are stored safely and can be retrieved efficiently.
- **Java Swing**: The **Swing** framework was used for creating an interactive and responsive GUI. This included various features such as drag-and-drop for adding events, input validation, and real-time updates to the calendar.
- **Event-Driven Programming**: The use of listeners such as SaveClickListener demonstrated how Java's event-driven programming model can be utilized to capture and respond to user actions, ensuring the system reacts instantly to user input.
- **MVC Pattern**: By applying the **Model-View-Controller** design pattern, the project was structured in a way that separates concerns, ensuring the system is modular, easier to maintain, and scalable for future developments.

5. Results

The development of the timetable management system using Java Swing resulted in a functional application capable of managing complex schedules for educational institutions. The system successfully meets the initial project goals of automating the scheduling process, detecting conflicts between events, and providing a user-friendly interface for managing timetables.

Key Outcomes

Functional Timetable System: The system provides users with an interactive interface where they can add, update, and view events in a calendar format. The hourly view provides an overview of daily schedules, allowing users to visualize how events align throughout the day. The system's graphical user interface, built using Java Swing, was well-received by potential users during initial testing, who noted its ease of use and clean layout.

Conflict Detection: One of the most critical features of the system is its ability to detect scheduling conflicts. The conflict detection algorithm efficiently identifies overlaps between events, particularly handling recurring events. Users are warned of conflicts during the scheduling process, preventing issues such as double-booking rooms or overlapping classes.

6. Conclusion

Challenges Encountered

Handling Recurring Events: One of the more complex issues encountered during development was the handling of recurring events. Managing schedules with repeated occurrences required additional logic to ensure that conflicts were correctly identified across multiple days. Initially, the brute-force approach caused performance bottlenecks when dealing with large sets of recurring events. To address this, optimizations were made in how the algorithm iterates through event occurrences, improving the system's efficiency in detecting conflicts without sacrificing accuracy.

Database Performance: Interfacing with the MySQL database initially caused performance lags when loading and saving large amounts of data. This issue was particularly noticeable when adding or retrieving multiple events at once. To resolve this, optimizations were made by batching database queries and leveraging efficient indexing strategies to speed up the interaction between the system and the database.

Lessons Learned

The development of this system reinforced the importance of early testing and iterative design. By frequently testing the system with potential users and incorporating their feedback, many usability and performance issues were identified and addressed early in the development cycle. Furthermore, adopting the Model-View-Controller (MVC) architecture from the outset proved crucial in maintaining a clean codebase that could be easily modified and expanded. The project demonstrated the power of using Java Swing for building graphical interfaces and the value of leveraging database management through JDBC for scalable, real-time data handling.

Future Improvements

While the current system performs well in managing small to medium-sized datasets, future iterations could incorporate more sophisticated algorithms, such as genetic algorithms, to optimize timetables further. Additionally, enhancing the system to include features such as room reservations or integrated lesson planning could significantly increase its functionality and value to educational institutions. Finally, improving the system's scalability by integrating advanced database optimization techniques would ensure the system remains performant as the number of events grows.

7. Contributions

The Project was developed and completed with 5 member in group 7, specifically each person takes on different parts of the job and has equivalent contributions to the project:

- Vũ Đặng Anh Quân:
    + Project Manager
    + Conceptualization
    + Algorithm Design
    + Project design
    + Developer
- Nguyễn Đức Toàn:
    + Conceptualization
    + Requirement analysis
    + Developer
    + Project design
    + Debugging and testing
- Nguyễn Công Tuấn:
    + Conceptualization
    + User Interface (UI) design
    + Developer
    + Debugging and testing
    + User Experience (UX) design
- Vũ Thị Hường:

    + Conceptualization
    + Developer
    + User Experience (UX) design
    + Report maker
    + Powerpoint maker
- Mai Hồng Hạnh:
    + Conceptualization
    + Developer
    + User Experience (UX) design
    + Report maker
    + Powerpoint maker

8. Reference

Khyrina Airin Fariza Abu Samah, Siti Qamalia Thusree, Ahmad Firdaus Ahmad Fadzil, Lala Septem Riza, Shafaf Ibrahim and Noraini Hasan, "A Greedy-based Algorithm in Optimizing Student's Recommended Timetable Generator with Semester Planner" International Journal of Advanced Computer Science and Applications(IJACSA), 13(1), 2022. http://dx.doi.org/10.14569/IJACSA.2022.0130146