구현 Class

ButtonPanel.class

```
package org.example.view;

import javax.swing.*;
import java.awt.*;

public class ButtonPanel extends JPanel {
    public JButton showListButton = new JButton("Show List");
    public JButton addButton = new JButton("Add");
    public JButton deleteButton = new JButton("Delete");
    public JButton loadButton = new JButton("Load");
    public JButton saveButton = new JButton("Save");

    public ButtonPanel() {
        setLayout(new GridLayout(5, 1, 5, 5));
        add(showListButton);
        add(addButton);
        add(deleteButton);
        add(loadButton);
        add(saveButton);
    }
}
```

- **역할**:
  사용자 인터페이스 오른쪽에 위치하며, 다섯 개의 주요 기능 버튼(`Show List` , `Add` , `Delete` , `Load` , `Save`)을 수직으로 배치함.
- **구성요소**:
  - `JButton showListButton` : 저장된 과목 리스트 출력
  - `JButton addButton` : 과목 추가 기능 예정
  - `JButton deleteButton` : 과목 삭제 기능 예정
  - `JButton loadButton` : 파일에서 로드

CourseListPanel.class

```
package org.example.view;

import java.awt.BorderLayout;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;
import org.example.Course;
import org.example.CourseSchedule;

public class CourseListPanel extends JPanel {

    CourseListPanel(Object[][] data) {
        JTable table = new JTable();
        String[] columns = {"과목명", "학점", "요일", "시작교시", "시간"};

        table.setModel(new DefaultTableModel(data, columns));

        JScrollPane scrollPane = new JScrollPane(table);
```

```
        setLayout(new BorderLayout());
        add(scrollPane, BorderLayout.CENTER);
    }

    public static Object[][] getTableData(CourseSchedule courseSchedule) {
        Object[][] data = new Object[courseSchedule.getCourses().size()][5];
        for (int i = 0; i < courseSchedule.getCourses().size(); i++) {
            Course course = courseSchedule.getCourses().get(i);
            data[i][0] = course.getName();
            data[i][1] = course.getCredit();
            data[i][2] = course.getDay();
            data[i][3] = course.getStart();
            data[i][4] = course.getTime();
        }
        return data;
    }
}
```

- **역할**:
  과목 리스트를 표 형태로 화면에 출력하는 역할을 담당. 과목 정보를 테이블 형태로 받아 시각적으로 표현함.
- **기능**:
  - `JTable`과 `JScrollPane`을 활용하여 과목명을 포함한 5개 속성을 열로 출력
  - 열 이름: `과목명`, `학점`, `요일`, `시작교시`, `시간`
- **고려 사항**:
  - 해당하는 과목 정보를 보여주는 Panel로, CourseSchedule이 들어왔을 때, 이를 Object로 변환하여 Table로 보여줄 수 있도록 하는 static 메서드를 생성하였다.
  - 외부에서 해당 메서드를 사용하면 object로 바꿀 수 있도록 static 메서드로 설정하였다.

InputPanel.class

```
package org.example.view;

import javax.swing.*;
import java.awt.*;

public class InputPanel extends JPanel {
    public JTextField subjectField = new JTextField();
    public JTextField creditField = new JTextField();
    public JComboBox<String> dayComboBox = new JComboBox<>(new String[]{"Mon", "Tue", "Wed",
"Thu", "Fri"});
    public JTextField startField = new JTextField();
    public JTextField timeField = new JTextField();

    public InputPanel() {
        setLayout(new GridLayout(2, 5, 5, 5));

        add(new JTextField("과목명"));
        add(new JTextField("학점"));
        add(new JTextField("요일"));
        add(new JTextField("시작 교시"));
        add(new JTextField("시간"));

        add(subjectField);
        add(creditField);
        add(dayComboBox);
        add(startField);
        add(timeField);
    }
```

```
    // 필요하면 선택된 요일 가져오기
    public String getSelectedDay() {
        return (String) dayComboBox.getSelectedItem();
    }
}
```

- **역할**:
  사용자로부터 새로운 과목 정보를 입력받기 위한 텍스트 필드 입력창
- **기능**:
  - 위쪽 행에는 입력 필드를 설명하는 텍스트 필드(과목명, 학점 등)를 표시
  - 아래쪽 행에는 실질적인 입력 필드 (`JTextField` 변수들)를 구성
  - Day의 경우 Combobox를 이용하여 요일 선택 가능

ScheduleTablePanel.class

```
package org.example.view;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;

public class ScheduleTablePanel extends JPanel {
    public ScheduleTablePanel(Object[][] courseData) {
        setLayout(new BorderLayout());

        String[] days = {"Mon", "Tue", "Wed", "Thr", "Fri"};
        Object[][] schedule = toScheduleTable(courseData);
        JTable table = new JTable(new DefaultTableModel(schedule, days));

        JTable rowHeader = new JTable(new DefaultTableModel(
                new Object[][]{
                        {"1"}, {"2"}, {"3"}, {"4"}, {"5"}, {"6"}, {"7"}, {"8"}, {"9"}
                }, new String[]{""}
        ));
        rowHeader.setEnabled(false);
        rowHeader.setPreferredScrollableViewportSize(new Dimension(30, 0));

        JScrollPane scrollPane = new JScrollPane(table);
        scrollPane.setRowHeaderView(rowHeader);
        add(scrollPane, BorderLayout.CENTER);
    }

    public static Object[][] toScheduleTable(Object[][] courseData) {
        Object[][] table = new Object[9][5]; // 9교시 × 5요일

        for (Object[] course : courseData) {
            String name = (String) course[0];
            String day = (String) course[2];
            int start = Integer.parseInt(course[3].toString()) - 1;
            int time = Integer.parseInt(course[4].toString());

            int dayIdx = switch (day) {
                case "Mon" -> 0;
                case "Tue" -> 1;
                case "Wed" -> 2;
                case "Thu" -> 3;
                case "Fri" -> 4;
                default -> -1;
            };
```

```
            if (dayIdx == -1 || start < 0 || start + time > 9) continue;

            for (int i = 0; i < time; i++) {
                table[start + i][dayIdx] = name;
            }
        }
        return table;
    }

}
```

- **역할**:
  주간 시간표 형태로 과목이 배치된 스케줄 테이블을 화면에 출력
- **기능**:
  - `JTable`을 이용해 월 ~~ 금 요일(열)과 1 ~~ 9교시(행)에 과목을 배치
  - `scrollPane.setRowHeaderView()`를 사용해 좌측에 교시를 명시하는 row header 구성
  - 예시로 `"Java"`, `"CC"`, `"DS"` 과목이 일부 셀에 미리 배치됨
  -
- **고려사항**:
  - 기존의 시간표 내용에 담겨있는 정보들이 넘어오는 것을 염두하여 toScheduleTable이라는 메서드를 정의하였다. 이를 통하여 외부에서 넘어온 Course 정보를 이용하여 테이블을 구성할 수 있도록 하였다.

CourseScheduleSection.class

```
package org.example.view;

import javax.swing.*;
import java.awt.*;
import org.example.CourseSchedule;

public class CourseScheduleSection extends JPanel {
    private CourseSchedule schedule;
    private InputPanel inputPanel;
    private JTextField myCoursesField;
    private CourseListPanel listPanel;
    private ScheduleTablePanel tablePanel;
    private ButtonPanel buttonPanel;

    public CourseScheduleSection(CourseSchedule schedule, ButtonPanel buttonPanel) {
        this.schedule = schedule;
        this.buttonPanel = buttonPanel;

        setLayout(new BorderLayout());

        initializeTopPanel();
        updatePanels();
    }

    private void initializeTopPanel() {
        inputPanel = new InputPanel();
        myCoursesField = new JTextField("My Courses");
        myCoursesField.setEditable(false);
    }

    public void updatePanels() {
        removeAll();

        Object[][] data = CourseListPanel.getTableData(schedule);
        listPanel = new CourseListPanel(data);
```

```
        tablePanel = new ScheduleTablePanel(data);

        // 입력 + 리스트를 왼쪽에 묶기
        JPanel inputAndListPanel = new JPanel();
        inputAndListPanel.setLayout(new BoxLayout(inputAndListPanel, BoxLayout.Y_AXIS));
        inputAndListPanel.add(inputPanel);
        inputAndListPanel.add(myCoursesField);
        inputAndListPanel.add(listPanel);

        // 위쪽 전체 묶기: 입력+리스트 + 버튼
        JPanel topPanel = new JPanel(new BorderLayout());
        topPanel.add(inputAndListPanel, BorderLayout.CENTER); // 왼쪽 입력
        topPanel.add(buttonPanel, BorderLayout.EAST);         // 오른쪽 버튼

        // 상단 패널 (입력 + 버튼), 하단 시간표
        add(topPanel, BorderLayout.NORTH);
        add(tablePanel, BorderLayout.CENTER);

        revalidate();
        repaint();
    }

    public void setSchedule(CourseSchedule schedule) {
        this.schedule = schedule;
        updatePanels();
    }

    public CourseSchedule getSchedule() {
        return schedule;
    }

    public InputPanel getInputPanel() {
        return inputPanel;
    }

    public CourseListPanel getListPanel() {
        return listPanel;
    }

    public ScheduleTablePanel getTablePanel() {
        return tablePanel;
    }
}
```

- **역할**:
  - `CourseScheduleSection` 은 사용자로부터 새로운 과목 정보를 입력받고, 현재 등록된 과목 목록과 시간표를 한 화면에 종합적으로 보여주는 **중앙 패널** 역할을 한다.
  - 상단에는 과목 추가를 위한 **입력 인터페이스**(`InputPanel`)를 제공하고, 하단에는 현재 시간표와 등록된 과목을 표 형식으로 나타낸다.

CourseScheduleView.class

```
package org.example.view;

import java.net.InetAddress;
import java.net.UnknownHostException;
import java.time.LocalDateTime;
import javax.swing.*;
import java.awt.*;
import org.example.CourseSchedule;
```

```
public class CourseScheduleView {
    public static void main(String[] args) throws Exception {
        JFrame frame = new JFrame("Planning Course Schedule");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(800, 1200);

        ButtonPanel buttonPanel = new ButtonPanel();
        CourseSchedule dummySchedule = new CourseSchedule("myschedule-dump.data");

        CourseScheduleSection courseSection = new CourseScheduleSection(dummySchedule,
buttonPanel);
        frame.add(courseSection); // CENTER로 자동 배치

        buttonPanel.showListButton.addActionListener(e -> {
            try {
                CourseSchedule schedule = new CourseSchedule("myschedule-norm.data");
                JFrame frame2 = new JFrame("Course List");
                frame2.setSize(500, 300);
                frame2.setLayout(new BorderLayout());
                frame2.add(new JTextField("List of Courses - Click to Select"),
BorderLayout.NORTH);
                frame2.add(new CourseListPanel(CourseListPanel.getTableData(schedule)),
BorderLayout.CENTER);
                frame2.setVisible(true);
            } catch (Exception ex) {
                ex.printStackTrace();
                JOptionPane.showMessageDialog(frame, "Error loading schedule: " +
ex.getMessage());
            }
        });

        frame.setVisible(true);
    }
}
```

- **역할**:
  전체 GUI 애플리케이션의 메인 클래스이며, 프레임을 초기화하고 전체 UI를 구성함
- **기능**:
  - `JFrame` 객체를 생성하여 기본 크기와 종료 동작을 설정하고, `BorderLayout` 을 적용하여 전체 UI를 레이아웃한다
  - **중앙 영역(`CENTER`)** 에는 `CourseScheduleSection` 을 배치하여 입력창, 과목 리스트, 시간표를 통합 구성한다.
  - `CourseScheduleSection` 은 내부적으로 `InputPanel` , `CourseListPanel` , `ScheduleTablePanel` 을 포함함
  - **오른쪽 영역(`EAST`)**에는 `ButtonPanel` 을 추가하여 버튼 UI(`Show List` , `Add` , `Delete` 등)를 제공한다.
  - `"Show List"` 버튼 클릭 시:
    - 외부 파일(`myschedule-norm.data`)에서 `CourseSchedule` 객체를 불러와
    - 새로운 팝업 프레임(Frame 2)에 `CourseListPanel` 을 테이블로 출력한다

고려사항
본 프로젝트에서는 사용자 인터페이스를 두 개의 화면(Frame 1, Frame 2)으로 구분하여 설계하였다. Frame 1은 메인 화면으로, 사용자로부터 과목 정보를 입력받고 시간표를 확인할 수 있는 중심 인터페이스 역할을 한다.

화면 상단에는 과목 정보를 입력받는 텍스트 입력창과 "My Courses"라는 제목 표시줄, 그리고 현재 입력된 과목들을 테이블로 보여주는 패널이 순서대로 배치된다. 이러한 구성 요소들은 CourseScheduleSelection과 Course 객체가 따로 만들어진 뒤 이를 통해 화면이 구현될 수 있도록 하였다.

현재는 임시 데이터를 기반으로 테이블이 구성되어 있으나, 추후에는 `Course` 객체와 연동하여 실제 입력된 과목 정보를 반영할 수 있도록 확장할 예정이다.

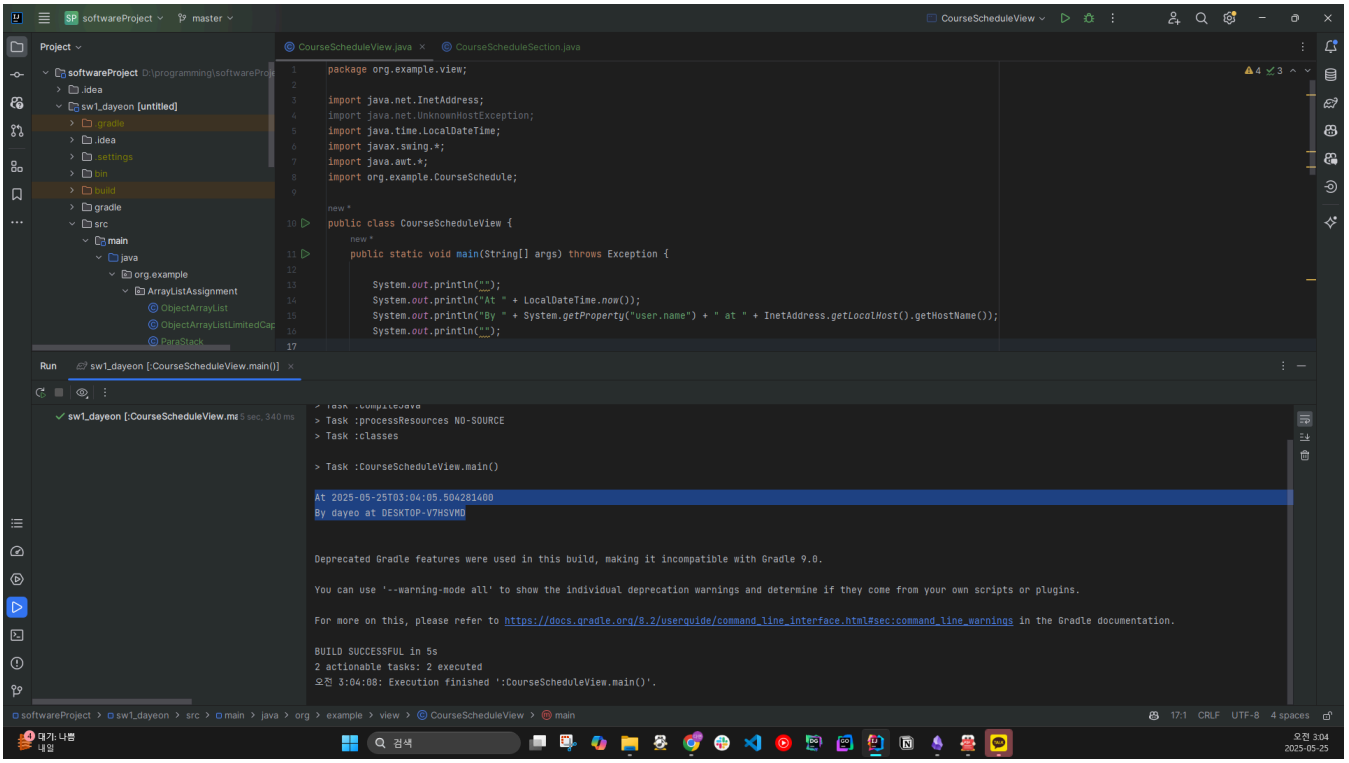
한편 Frame 2는 "Show List" 버튼 클릭 시 별도로 생성되는 서브 화면으로, 저장된 과목 목록을 표 형태로 출력한다. 이 화면은 추후 사용자가 과목을 선택하거나 세부 정보를 확인할 수 있도록 기능이 확장될 수 있는 구조로 고려되었다.

myschedule-dump.data 파일

```
OS:3:Wed:2:2
//
Network:2:Thu:4:2
//
AI:3:Fri:1:3
//
DataMining:2:Mon:5:2
//
Security:3:Wed:6:2
//
Compiler:3:Thu:1:3
//
ML:3:Fri:4:2
//
```

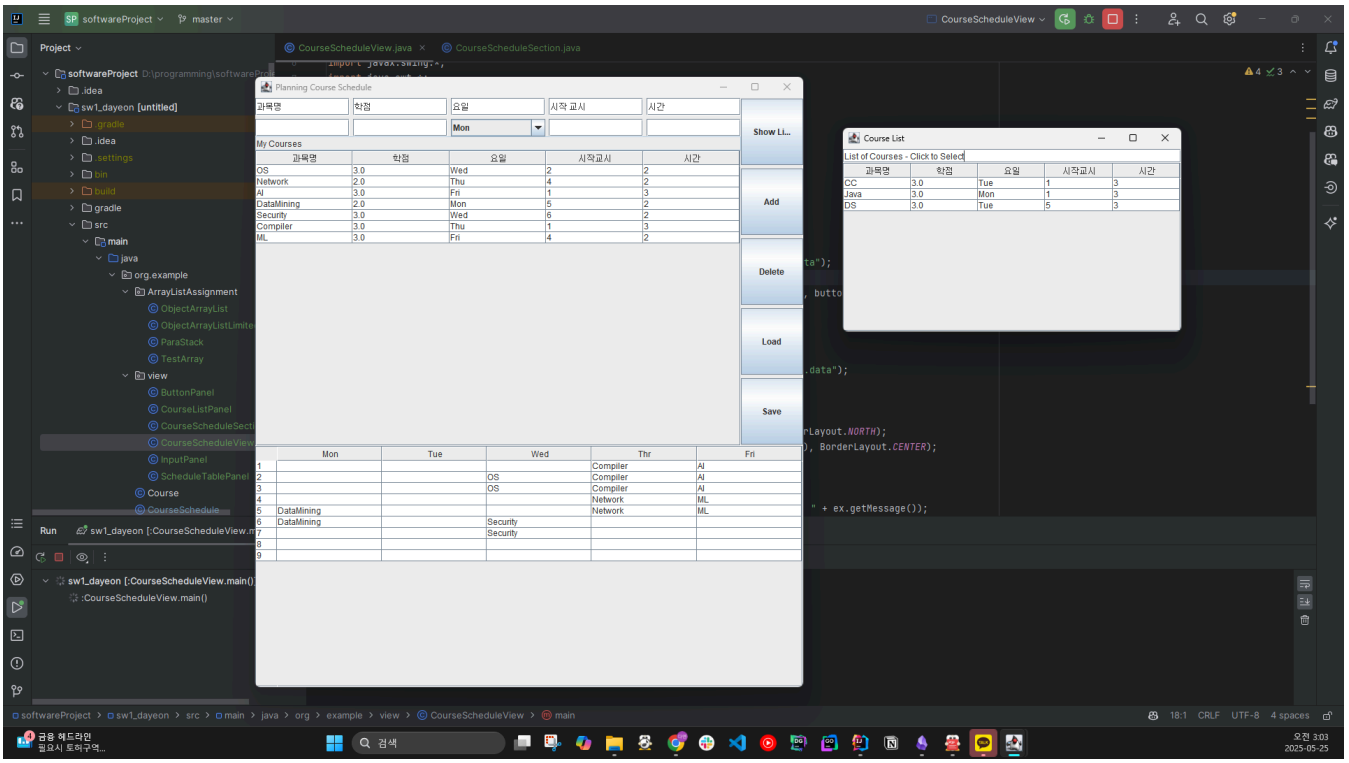
위와 같이 구성한 data 파일을 사용하였다.

본인인증



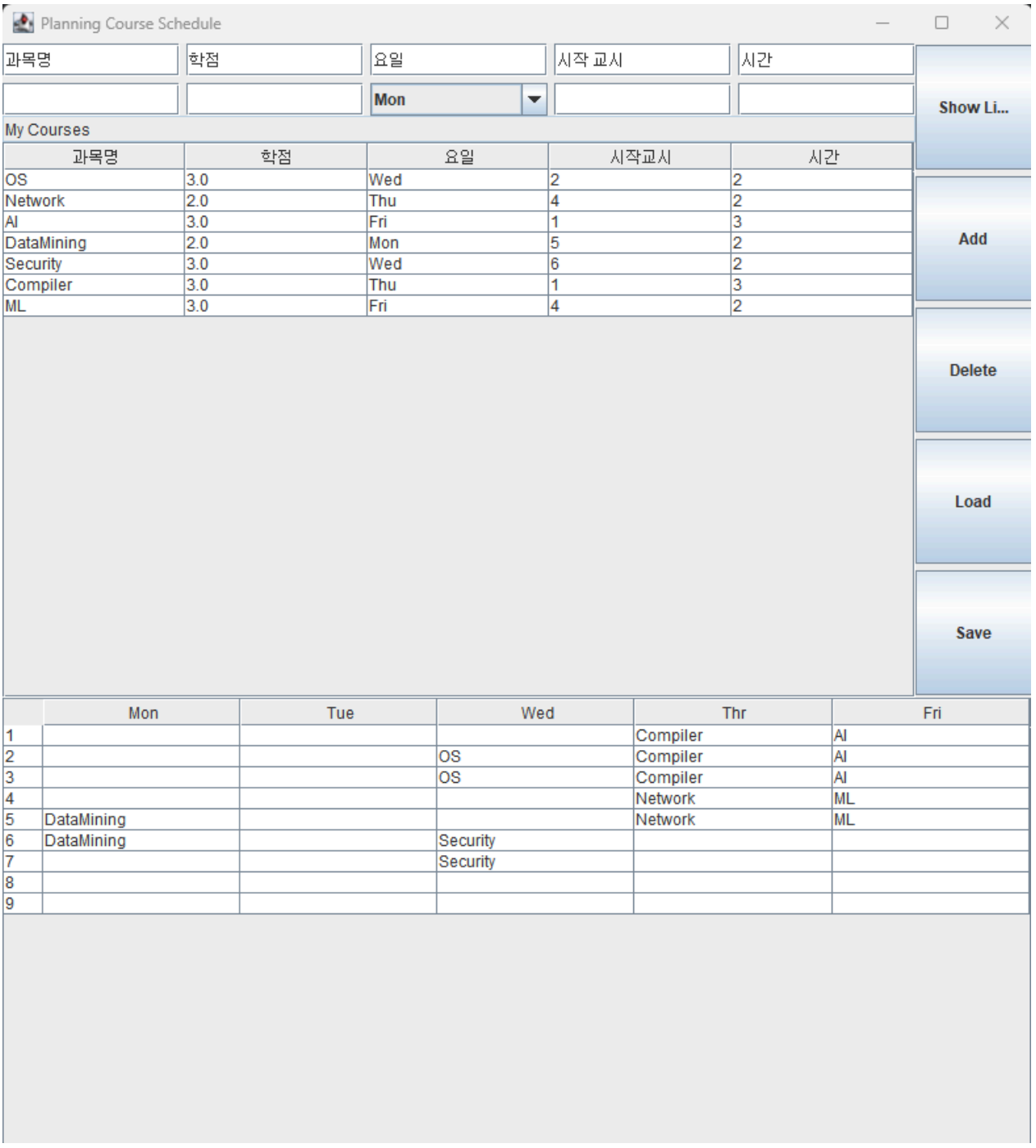
다음과 같은 환경에서 수행하였다.

화면 구현 내역

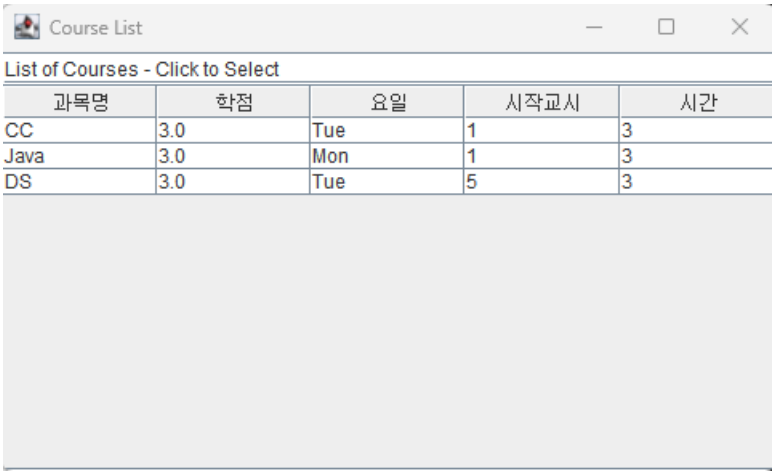


세부 화면

Planning Course Schedule



Course List



평가 표

평가표 (즉, 리포트 작성시 체크 사항)

평가 항목	학생 자체 평가 (리포트 해당 부분 표시 및 간단한 의견)	평가 (빈칸)	점수 (빈칸)
완성도 (동작 여부) - 두 GUI 화면 완성도 - "ShowList" 클릭하면, 과목리스트 창이 떠야 함. - 과제 3 코드의 통합	제시된 화면과 동일하게 GUI 화면을 완성 ShowList 클릭 시 과목리스트 창이 제대로 표현됨 과목리스트 연계시 CourseSchedule 코드와 통합		
설계 노트 - GUI 설계 시 착안 사항 - 클래스 구조 (특히, 과목 리스트 테이블 처리) - 프로젝트 3의 클래스와의 관계 - 시행착오 내용 및 해결 방안	CourseListPanel을 사용 하여 과목명에 따른 테 이블이 보여지도록 코드 작성 CourseListPanel과 CourseScheduleSection 에서 기존 Course, CourseSchedule과 연계되도록 작성 메서드 접근 제어자, static 등을 고려하여 작성		
리포트 - 평가자 시각으로 리포트 검토 - 위의 평가 요소들이 명확하게 기술되었는가? - 본인 인증? 자신만의 과목 정보?	레포트에 위와 관련된 내용을 모두 기재 본인인증 완료 자신만의 과목 정보 ML, Compiler 등 사용		
총평/계			

* 학생 자체 평가는 점수에 반영되지 않음.
* 학생 스스로 자신의 보고서를 평가하면서, 체계적으로 프로젝트를 마무리하도록 유도하는 것이 목적임.