

```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Drawing;
5 using System.Data;
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10
11
12 using Nihulon2.Model;
13 using System.Collections;
14 using DGVPrinterHelper;
15 using System.IO;
16
17 namespace Nihulon2.SupervisorsAdministration
18 {
19     /*
20      * The class of view that displays the data of exams
21      * ISupervisorsAdministrationView - interface that provides methods to the ↗
22      * controller
23      * for displaying the data
24      */
25     public partial class SupervisorAdministration_View : UserControl, ↗
26     {
27         ISupervisorsAdministrationView
28
29         // Instance of the controller that manages this view
30         private SupervisorAdministration_Controller _controller;
31
32         public SupervisorAdministration_View()
33         {
34             InitializeComponent();
35
36             radioButtonShowAll.Checked = true; // set "show all exams" as a ↗
37             default filter
38         }
39
40         #region Form Events
41
42         // Adding a new exam
43         private void btnAddExam_Click(object sender, EventArgs e)
44         {
45             // If all required fields at the form are filled
46             if(cboCourse_NewExam.SelectedIndex != -1 &&
47                 cboDivision_NewExam.SelectedIndex != -1 && ↗
48                 cboRoom_NewExam.SelectedIndex != -1)
49             {
50                 // Create a new exam and fill its fields
51                 Exam newExam = new Exam();
52                 newExam.course = cboCourse_NewExam.Text;
53                 newExam.Date = DatePicker.Text;
54                 newExam.DisciplineName = txtDiscipline_NewExam.Text;
55                 newExam.division = cboDivision_NewExam.Text;
56                 newExam.EndingTime = ToTimePicker.Text;
57                 newExam.GroupName = txtGroup_NewExam.Text;
```

```
53         newExam.hasExtraTime = cbExtraTime_NewExam.Checked;
54         newExam.room = cboRoom_NewExam.Text;
55         newExam.StartTime = FromTimePicker.Text;
56         newExam.SupervisorName = txtSupervisor_NewExam.Text;
57         newExam.dateOfCreation = DateTime.Today.ToString  ↗
            ("dd.MM.yyyy");
58
59         // Add the new exam to the list of exams and insert into the  ↗
            DB
60         _controller.addExam(newExam);
61
62         clearAddingForm();
63     }
64 }
65
66 // Clear the adding form
67 private void btnClearAddingForm_Click(object sender, EventArgs e)
68 {
69     clearAddingForm();
70     // Disable the button of clearing the form
71     btnClearAddingForm.Enabled = false;
72 }
73
74 // Create a copy of selected exam
75 private void btn_CopyRow_Click(object sender, EventArgs e)
76 {
77     // get selected exam as an instance of Exam
78     Exam selectedExam = dgvExamTable.CurrentRow.DataBoundItem as Exam; ↗
79
80     if(selectedExam != null)
81         _controller.addExam(selectedExam); // Add exam
82 }
83
84 // Delete an exam
85 private void btnDeleteExam_Click(object sender, EventArgs e)
86 {
87     int examId;
88     Exam exam;
89     // Go through all selected exams and delete them
90     foreach (DataGridViewRow row in dgvExamTable.SelectedRows)
91     {
92         exam = row.DataBoundItem as Exam;
93         if(exam != null)
94         {
95             examId = exam.Id; // get exam id from selected exam
96             _controller.deleteExam(examId); // delete
97         }
98     }
99     // Get the updated data from the DB
100     _controller.fillTable();
101 }
102
103 // Open the form of changing an exam by click
104 // on the button "change exam"
105 private void btnChangeExam_Click(object sender, EventArgs e)
106 {
```

```
106         this.callFormOfExamChanging();
107     }
108     // Open the form of changing an exam by double click on the exam at the table
109     private void dgvExamTable_CellDoubleClick(object sender, DataGridViewCellEventArgs e)
110     {
111         this.callFormOfExamChanging();
112     }
113
114     // When one of the checkboxes, "canceled" or "has extra time" changes its state,
115     // change the exam state and save into the DB
116     private void dgvExamTable_CellValueChanged(object sender, DataGridViewCellEventArgs e)
117     {
118         Exam changedExam = dgvExamTable.CurrentRow.DataBoundItem as Exam;
119         if(changedExam != null)
120             _controller.changeExam(changedExam);
121     }
122
123     // When a row in the table selected, fill textbox with the comments of the selected exam
124     // If there is multi-selected rows, disable the buttons of changing and copying
125     private void dgvExamTable_SelectionChanged(object sender, EventArgs e)
126     {
127         // If there is selected row at the table, fill the text box with comments
128         if (dgvExamTable.CurrentRow != null)
129         {
130             Exam exam = dgvExamTable.CurrentRow.DataBoundItem as Exam;
131             if (exam != null)
132                 txt_Comments.Text = exam.Comments;
133         }
134
135         // If there are more than one selected row, disable the buttons of changing and copying exams
136         if(dgvExamTable.SelectedRows.Count != 1)
137         {
138             btnChangeExam.Enabled = false;
139             btn_CopyRow.Enabled = false;
140         }
141         else
142         {
143             btnChangeExam.Enabled = true;
144             btn_CopyRow.Enabled = true;
145         }
146     }
147
148     // Save changes after editing the comments
149     private void txt_Comments_Leave(object sender, EventArgs e)
150     {
151         // get selected exam
152         Exam changedExam = (dgvExamTable.CurrentRow.DataBoundItem as
```

```
        Exam);
155
156        // If casted successful
157        if(changedExam != null)
158        {
159            changedExam.Comments = txt_Comments.Text; // set new comments
160            _controller.changeExam(changedExam); // change into the DB
161        }
162    }
163
164    /*
165     * When one of the radiobuttons at the time filters was checked,
166     * enable the needed option and disable others
167     */
168    // Show all
169    private void radioButtonShowAll_CheckedChanged(object sender, EventArgs e)
170    {
171        dateTimePickerOneDay_Filter.Enabled = false;
172        dateTimePickerFrom_Filter.Enabled = false;
173        dateTimePickerTo_Filter.Enabled = false;
174    }
175    // Period
176    private void radioButtonPeriod_CheckedChanged(object sender, EventArgs e)
177    {
178        dateTimePickerOneDay_Filter.Enabled = false;
179        dateTimePickerFrom_Filter.Enabled = true;
180        dateTimePickerTo_Filter.Enabled = true;
181    }
182    // One day
183    private void radioButtonOneDay_CheckedChanged(object sender, EventArgs e)
184    {
185        dateTimePickerOneDay_Filter.Enabled = true;
186        dateTimePickerFrom_Filter.Enabled = false;
187        dateTimePickerTo_Filter.Enabled = false;
188    }
189
190    // When the button "show exams" has been pressed,
191    // set all values of the filters at the controller and ask to get the needed data
192    private void btnShowExams_Click(object sender, EventArgs e)
193    {
194        /* Set the filters of period of time */
195
196        // show all or show new
197        if (radioButtonShowAll.Checked == true || radioButtonShowNew.Checked == true)
198        {
199            _controller.dateFromFilter = "";
200            _controller.dateToFilter = "";
201        }
202        // show exams for one day
203        else if (radioButtonOneDay.Checked == true)
204        {
```

```
205         _controller.dateFromFilter = dateTimePickerOneDay_Filter.Text;
206         _controller.dateToFilter = dateTimePickerOneDay_Filter.Text;
207     }
208     // show exams for a period
209     else if (radioButtonPeriod.Checked == true)
210     {
211         _controller.dateFromFilter = dateTimePickerFrom_Filter.Text;
212         _controller.dateToFilter = dateTimePickerTo_Filter.Text;
213     }
214
215     // Set other filters from the form to the controller
216     _controller.divisionFilter = cboDivisionFilter.Text;
217     _controller.courseFilter = cboCourseFilter.Text;
218     _controller.roomFilter = cboRoomFilter.Text;
219     _controller.showDisabledFilter = cbShowDisabledExams.Checked;
220     _controller.showNewFilter = radioButtonShowNew.Checked;
221     _controller.showOverlaps = false;
222
223     // Ask the controller to fill the table according to the filters
224     _controller.fillTable();
225 }
226
227 // If one of the dates at the filter form was changed,
228 // check if the "From" date is earlier than the "To" date.
229 // If not, set the "To" date equal to the "From" date
230 private void dateTimePickerFilter_ValueChanged(object sender,      ↗
    EventArgs e)
231 {
232     this.checkAndSetFilterDates(dateTimePickerTo_Filter,          ↗
        dateTimePickerFrom_Filter);
233 }
234
235 // The method that is called when something has been changed at the  ↗
    form
236 // of adding a new exam. It makes the button "clear the form" being  ↗
    enabled
237 private void creationFormChanged(object sender, EventArgs e)
238 {
239     btnClearAddingForm.Enabled = true;
240 }
241
242 // Printing the Exam table
243 // The method uses DGVPainter module that serves
244 // the printing of data Grid Views
245 private void btn_Print_Click(object sender, EventArgs e)
246 {
247     DGVPainter printer = new DGVPainter();
248
249     // Settings of the printing
250     printer.HeaderCellAlignment = StringAlignment.Near;
251     printer.PageSettings.Landscape = true;
252     printer.PageSettings.Color = false;
253     printer.PageSettings.Margins.Left = 30;
254     printer.RowHeight = DGVPainter.RowHeightSetting.CellHeight;
255
256     // Print
```

```
257         printer.PrintDataGridView(dgvExamTable);
258     }
259
260     // Create a form for taking the name of Excel file
261     private void btn_ExportToExcel_Click(object sender, EventArgs e)
262     {
263         frmGetNameOfExcelFile getNameForm = new frmGetNameOfExcelFile();
264         // Event called when user entered the name of the file
265         getNameForm.onFileNameConfirm += excelNameEntered;
266
267         getNameForm.StartPosition = FormStartPosition.CenterParent;
268         getNameForm.ShowDialog();
269     }
270     // Loads exams from excel
271     private void btn_ImportFromExcel_Click(object sender, EventArgs e)
272     {
273         // Create a folder for Excel files if not exists
274         string location = System.Windows.Forms.Application.StartupPath;
275         location += @"\Excel files";
276         if (!Directory.Exists(location))
277         {
278             Directory.CreateDirectory(location);
279         }
280
281         // Open the file dialog from the folder with Excel files
282         OpenFileDialog openFileDialog = new OpenFileDialog();
283         openFileDialog.InitialDirectory = location;
284         openFileDialog.Filter = "Excel files (*.xlsx)|*.xlsx|All files (*.*)|*.*";
285
286         if (openFileDialog.ShowDialog() == DialogResult.OK && openFileDialog.CheckFileExists)
287         {
288             _controller.loadFromExcel(openFileDialog.FileName);
289         }
290     }
291
292     // Checks each exam if it has time overlaps and paint its row red
293     private void dgvExamTable_DataBindingComplete(object sender, DataGridViewBindingCompleteEventArgs e)
294     {
295         foreach (DataGridViewRow dgvr in dgvExamTable.Rows)
296         {
297             if (Convert.ToBoolean(dgvr.Cells["hasOverlap"].Value))
298                 dgvr.DefaultCellStyle.BackColor = Color.IndianRed;
299         }
300     }
301
302     // Show only the exams that have time overlaps
303     private void btnOverlaps_Click(object sender, EventArgs e)
304     {
305         _controller.showOverlaps = true;
306
307         _controller.fillTable();
308     }
309
```

```
310 // Runs searching of exams with time overlaps
311 private void btnCheckOverlaps_Click(object sender, EventArgs e)
312 {
313     _controller.checkOverlaps();
314 }
315
316 #endregion
317
318 #region Interface
319 // binds the view with its controller
320 public void setController(SupervisorsAdministration_Controller  ↗
321     controller)
322 {
323     _controller = controller;
324 }
325
326 // Binds the list of exams at the controller with the table at the  ↗
327     view
328 public void bindExamsWithTable(ref SortableBindingList<Exam>  ↗
329     examsList)
330 {
331     dgvExamTable.DataSource = examsList; // bind
332     this.formatTable(); // set column sizes
333 }
334
335 // Fill the comboboxes of Divisions at the form of filters and at the  ↗
336     form of adding
337 public void fillDivisionsCbo(string[] divisionsNames)
338 {
339     cboDivisionFilter.Items.Add("הכול");
340     // Fill the comboboxes with names
341     for (int i = 0; i < divisionsNames.Length; i++)
342     {
343         cboDivisionFilter.Items.Add(divisionsNames[i]);
344         cboDivision_NewExam.Items.Add(divisionsNames[i]);
345     }
346     // Set "show all exams" as selected
347     cboDivisionFilter.SelectedIndex = 0;
348 }
349
350 // Fill the comboboxes of Courses at filters form and at the adding  ↗
351     form
352 public void fillCoursesCbo(string[] coursesNames)
353 {
354     cboCourseFilter.Items.Add("הכול");
355     // Fill the comboboxes with names
356     for (int i = 0; i < coursesNames.Length; i++)
357     {
358         cboCourseFilter.Items.Add(coursesNames[i]);
359         cboCourse_NewExam.Items.Add(coursesNames[i]);
360     }
361     // Set "show all exams" as selected
362     cboCourseFilter.SelectedIndex = 0;
363 }
364
365 // Fill the comboboxes of Rooms at filters form and at the adding form
366 public void fillRoomsCbo(string[] roomsNames)
367 {
```

```
361         cboRoomFilter.Items.Add("רונן");
362         // Fill the comboboxes with names
363         for (int i = 0; i < roomsNames.Length; i++)
364         {
365             cboRoomFilter.Items.Add(roomsNames[i]);
366             cboRoom_NewExam.Items.Add(roomsNames[i]);
367         }
368         // Set "show all exams" as selected
369         cboRoomFilter.SelectedIndex = 0;
370     }
371
372     // Clear all combo boxes
373     public void clearComboBoxes()
374     {
375         cboCourseFilter.Items.Clear();
376         cboDivisionFilter.Items.Clear();
377         cboRoomFilter.Items.Clear();
378         cboCourse_NewExam.Items.Clear();
379         cboDivision_NewExam.Items.Clear();
380         cboRoom_NewExam.Items.Clear();
381     }
382
383     // Get updated data from the DB and reload the view
384     public void reloadView()
385     {
386         _controller.reload();
387     }
388
389     // Set visible or hide the button and the label that show the overlaps ↗
390     // at the table
391     public void showOverlapsWarning(bool hasOverlaps)
392     {
393         if (hasOverlaps == true && btnOverlaps.Visible == false)
394         {
395             btnOverlaps.Visible = true;
396             labelOverlaps.Visible = true;
397         }
398         else if (hasOverlaps == false)
399         {
400             btnOverlaps.Visible = false;
401             labelOverlaps.Visible = false;
402         }
403     }
404     #endregion
405
406     #region Private methods
407
408     // Clear txt fields and checkbox at the form for adding exams
409     private void clearAddingForm()
410     {
411         txtDiscipline_NewExam.Clear();
412         txtGroup_NewExam.Clear();
413         txtSupervisor_NewExam.Clear();
414         cboRoom_NewExam.SelectedIndex = -1;
415         cboCourse_NewExam.SelectedIndex = -1;
```



```
416         cboDivision_NewExam.SelectedIndex = -1;
417
418         cbExtraTime_NewExam.Checked = false;
419     }
420
421     // Set column sizes and hide columns that user doesn't need
422     // like Id, date of creation etc.
423     private void formatTable()
424     {
425         // hide the columns that user doesn't need
426         dgvExamTable.Columns["comments"].Visible = false;
427         dgvExamTable.Columns["id"].Visible = false;
428         dgvExamTable.Columns["dateOfCreation"].Visible = false;
429         dgvExamTable.Columns["hasOverlap"].Visible = false;
430
431         // set fixed size for not textual columns
432         dgvExamTable.Columns["Date"].Width = 80;
433         dgvExamTable.Columns["isCanceled"].Width = 70;
434         dgvExamTable.Columns["hasExtraTime"].Width = 70;
435         dgvExamTable.Columns["startTime"].Width = 70;
436         dgvExamTable.Columns["endingTime"].Width = 70;
437         dgvExamTable.Columns["GroupName"].Width = 70;
438         dgvExamTable.Columns["room"].Width = 100;
439         dgvExamTable.Columns["division"].Width = 100;
440
441         // fill the entire space with textual columns
442         dgvExamTable.Columns["SupervisorName"].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;
443         //dgvExamTable.Columns["division"].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;
444         dgvExamTable.Columns["course"].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;
445         //dgvExamTable.Columns["GroupName"].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;
446         dgvExamTable.Columns["DisciplineName"].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;
447         //dgvExamTable.Columns["room"].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;
448
449         // forbid editing for all columns excepting "canceled" and "has
         extra time"
450         dgvExamTable.Columns["Date"].ReadOnly = true;
451         dgvExamTable.Columns["startTime"].ReadOnly = true;
452         dgvExamTable.Columns["endingTime"].ReadOnly = true;
453         dgvExamTable.Columns["SupervisorName"].ReadOnly = true;
454         dgvExamTable.Columns["division"].ReadOnly = true;
455         dgvExamTable.Columns["course"].ReadOnly = true;
456         dgvExamTable.Columns["GroupName"].ReadOnly = true;
457         dgvExamTable.Columns["DisciplineName"].ReadOnly = true;
458         dgvExamTable.Columns["room"].ReadOnly = true;
459     }
460
461     // Checks if the date "From" is earlier than the "To" date
462     // If not, sets the "To" date equal to the "From" date
463     private void checkAndSetFilterDates(DateTimePicker dateTimePickerTo, DateTimePicker
        dateTimePickerFrom)
```

```
464     {
465         if (dateTimePickerTo.Value < dateTimePickerFrom.Value)
466         {
467             dateTimePickerTo.Value = dateTimePickerFrom.Value;
468         }
469     }
470
471     // Event handler that is called when the user enter
472     // the name of the excel file for saving the table of exams
473     private void excelNameEntered(string excelName)
474     {
475         _controller.createExcel(excelName);
476     }
477
478
479     // Called when one of the time pickers at the form of creating
480     // new exams has been changed.
481     // Checks if the "from" date is earlier than "to" date
482     private void TimePicker_ValueChanged(object sender, EventArgs e)
483     {
484         checkAndSetFilterDates(ToTimePicker, FromTimePicker);
485     }
486
487     // Get changed exam and save the changes into the DB
488     private void saveChangedExam(Exam changedExam)
489     {
490         _controller.changeExam(changedExam);
491     }
492
493     // Creates the form of changing exams,
494     // fills its controls and set the exam
495     // that is needed to be changed
496     private void callFormOfExamChanging()
497     {
498         // get selected exam as an instance of Exam
499         Exam selectedExam = (Exam)dgvExamTable.CurrentRow.DataBoundItem;
500
501         // Show form for changing exams
502         frmChangeExam changeForm = new frmChangeExam();
503         // Subscribe on exam changes
504         changeForm.onSaveChanges += saveChangedExam;
505         // Fill the comboBoxes of the changeForme with related items
506         changeForm.fillComboBoxes(cboDivision_NewExam.Items,
507                                   cboCourse_NewExam.Items, cboRoom_NewExam.Items);
508         // Set the exam that needed to be changed
509         changeForm.setExam(selectedExam);
510
511         // Show the form at the center
512         changeForm.StartPosition = FormStartPosition.CenterParent;
513         changeForm.ShowDialog();
514     }
515     #endregion
516 }
517 }
518
```