

```

1  """The module that contains all controller classes that are common to criminal
2  cases (criminal includes traffic). """
3  import os
4  import pathlib
5  from docxtpl import DocxTemplate
6  from loguru import logger
7
8  from PyQt5.QtWidgets import QDialog
9  from PyQt5 import QtCore
10 from PyQt5.QtCore import QDate
11 from PyQt5.QtWidgets import QLabel, QPushButton, QMessageBox, QComboBox, QLineEdit
12 from PyQt5.QtSql import QSqlDatabase, QSqlQuery
13
14 from models.case_information import (
15     CaseInformation,
16     CriminalCharge,
17     AmendOffenseDetails,
18     LicenseSuspensionTerms,
19     CommunityControlTerms,
20     CommunityServiceTerms,
21     OtherConditionsTerms,
22 )
23 from views.add_conditions_dialog_ui import Ui_AddConditionsDialog
24 from views.amend_offense_dialog_ui import Ui_AmendOffenseDialog
25 from resources.db.DatabaseCreation import create_offense_list, create_statute_list
26
27
28 PATH = str(pathlib.Path().absolute())
29 TEMPLATE_PATH = PATH + "\\resources\\Templates\\"
30 SAVE_PATH = PATH + "\\resources\\Saved\\"
31 DB_PATH = PATH + "\\resources\\db\\"
32 CHARGES_DATABASE = DB_PATH + "\\charges.sqlite"
33
34
35 class BaseCriminalDialog(QDialog):
36     """This class is a base class to provide methods that are used by some or
37     all class controllers that are used in the application. This class is never
38     instantiated as its own dialog, but the init contains the setup for all
39     inherited class controllers."""
40
41     def __init__(self, parent=None):
42         super().__init__(parent)
43         self.setupUi(self)
44         self.doc = None
45         self.docname = None
46         self.pay_date_dict = {
47             "forthwith": 0,
48             "within 30 days": 30,
49             "within 60 days": 60,
50             "within 90 days": 90,
51         }
52
53     def close_window(self):
54         """Function connected to a button to close the window. Can be connected
55         to any button press/click/release to close a window."""
56         self.close()
57
58     @logger.catch
59     def add_charge_process(self, bool):
60         """The order of functions that are called when the add_charge_Button is
61         clicked(). The order is important to make sure the informaiton is
62         updated before the charge is added and the data cleared from the fields.
63
64         The bool is passed as an argument through clicked() but not used."""
65         self.criminal_charge = CriminalCharge()
66         self.add_charge()
67         self.clear_charge_fields()

```

```

68
69 @logger.catch
70 def create_entry_process(self):
71     """The order of functions that are called when the create_entry_Button is pressed()
72     on the MinorMisdemeanorDialog. The order is important to make sure the information is
73     updated before the entry is created."""
74     self.update_case_information()
75     self.create_entry()
76     self.close_event()
77
78 @logger.catch
79 def create_entry(self):
80     """The standard function used to create an entry when a create entry
81     button is press/click/released."""
82     self.doc = DocxTemplate(self.template.template_path)
83     self.doc.render(self.case_information.get_case_information())
84     self.set_document_name()
85     self.doc.save(SAVE_PATH + self.docname)
86     os.startfile(SAVE_PATH + self.docname)
87
88 def set_document_name(self):
89     """Sets document name based on the case number and name of the template
90     must include '.docx' to make it a Word document."""
91     self.docname = (
92         self.case_information.case_number + "_" + self.template.template_name + ".docx"
93     )
94
95 def set_case_information_banner(self):
96     """Sets the banner on a view of the interface. It modifies label
97     widgets on the view to text that was entered."""
98     self.defendant_name_label.setText(
99         "State of Ohio v. {defendant_first_name} {defendant_last_name}".format(
100             defendant_first_name = self.case_information.defendant.first_name,
101             defendant_last_name = self.case_information.defendant.last_name
102         )
103     )
104     self.case_number_label.setText(self.case_information.case_number)
105     if self.case_information.defendant_attorney_name is not None:
106         self.defendant_attorney_name_label.setText(
107             "Attorney: " + self.case_information.defendant_attorney_name
108         )
109     else:
110         self.defendant_attorney_name_label.setText("Attorney: None")
111
112 def update_case_information(self):
113     """Placeholder for future use to refactor out of other dialogs and reduce
114     code reuse."""
115     pass
116
117
118 class AddConditionsDialog(BaseCriminalDialog, Ui_AddConditionsDialog):
119     """The AddConditionsDialog is created when the addConditionsButton is clicked on
120     the MinorMisdemeanorDialog. The conditions that are available to enter information
121     for are based on the checkboxes that are checked on the MMD screen."""
122
123 @logger.catch
124 def __init__(self, minor_misdemeanor_dialog, parent=None):
125     super().__init__(parent)
126     self.case_information = minor_misdemeanor_dialog.case_information
127     self.modify_view()
128     self.community_service = (
129         minor_misdemeanor_dialog.community_service_checkBox.isChecked()
130     )
131     self.license_suspension = (
132         minor_misdemeanor_dialog.license_suspension_checkBox.isChecked()
133     )
134     self.community_control = (
135         minor_misdemeanor_dialog.community_control_checkBox.isChecked()

```

```

136 )
137 self.other_conditions = (
138     minor_misdemeanor_dialog.other_conditions_checkBox.isChecked()
139 )
140 self.other_conditions = (
141     minor_misdemeanor_dialog.other_conditions_checkBox.isChecked()
142 )
143 self.enable_condition_frames()
144
145 @logger.catch
146 def modify_view(self):
147     """Modifies the view of AddConditionsDialog that is created by the UI
148     file.
149     Gets the total number of charges from the charges in charges_list then
150     loops through the charges_list and adds parts of each charge to the
151     view. CLEAN UP?"""
152     index_of_charge_to_add = 0
153     column = self.charges_gridLayout.columnCount() + 1
154     total_charges_to_add = len(self.case_information.charges_list)
155     while index_of_charge_to_add < total_charges_to_add:
156         charge = vars(self.case_information.charges_list[index_of_charge_to_add])
157         if charge is not None:
158             self.charges_gridLayout.addWidget(
159                 QLabel(charge.get("offense")), 0, column
160             )
161             self.charges_gridLayout.addWidget(
162                 QLabel(charge.get("statute")), 1, column
163             )
164             self.charges_gridLayout.addWidget(
165                 QLabel(charge.get("finding")), 2, column
166             )
167             column += 1
168             index_of_charge_to_add += 1
169
170 @logger.catch
171 def enable_condition_frames(self):
172     """Enables the frames on the AddConditionsDialog dialog if the condition is checked
173     on the MinorMisdemeanorDialog screen. Also creates an instance of the object for
174     each condition."""
175     if self.other_conditions is True:
176         self.other_conditions_frame.setEnabled(True)
177         self.other_conditions_details = OtherConditionsTerms()
178     if self.license_suspension is True:
179         self.license_suspension_frame.setEnabled(True)
180         self.license_suspension_details = LicenseSuspensionTerms()
181         self.license_suspension_date_box.setDate(QtCore.QDate.currentDate())
182     if self.community_service is True:
183         self.community_service_frame.setEnabled(True)
184         self.community_service_terms = CommunityServiceTerms()
185         self.community_service_date_to_complete_box.setDate(
186             QtCore.QDate.currentDate()
187         )
188     if self.community_control is True:
189         self.community_control_frame.setEnabled(True)
190         self.community_control_terms = CommunityControlTerms()
191
192 @logger.catch
193 def add_conditions(self):
194     """The method is connected to the pressed() signal of continue_Button on the
195     Add Conditions screen."""
196     if self.community_service is True:
197         self.add_community_service_terms()
198     if self.community_control is True:
199         self.add_community_control_terms()
200     if self.license_suspension is True:
201         self.add_license_suspension_details()
202     if self.other_conditions is True:
203         self.add_other_condition_details()

```

```

204 @logger.catch
205
206 def add_community_control_terms(self):
207     """The method adds the data entered to the CommunityControlTerms object
208     that is created when the dialog is initialized. Then the data is transferred
209     to case_information."""
210     self.community_control_terms.type_of_community_control = (
211         self.type_of_community_control_box.currentText()
212     )
213     self.community_control_terms.term_of_community_control = (
214         self.term_of_community_control_box.currentText()
215     )
216     self.case_information.community_control_terms = self.community_control_terms
217
218 @logger.catch
219 def add_community_service_terms(self):
220     """The method adds the data entered to the CommunityServiceTerms object
221     that is created when the dialog is initialized. Then the data is transferred
222     to case_information."""
223     self.community_service_terms.hours_of_service = (
224         self.community_service_hours_ordered_box.value()
225     )
226     self.community_service_terms.days_to_complete_service = (
227         self.community_service_days_to_complete_box.currentText()
228     )
229     self.community_service_terms.due_date_for_service = (
230         self.community_service_date_to_complete_box.date().toString("MMMM dd, yyyy")
231     )
232     self.case_information.community_service_terms = self.community_service_terms
233
234 @logger.catch
235 def add_license_suspension_details(self):
236     """The method adds the data entered to the LicenseSuspensionTerms object
237     that is created when the dialog is initialized. Then the data is transferred
238     to case_information."""
239     self.license_suspension_details.license_type = (
240         self.license_type_box.currentText()
241     )
242     self.license_suspension_details.license_suspended_date = (
243         self.license_suspension_date_box.date().toString("MMMM dd, yyyy")
244     )
245     self.license_suspension_details.license_suspension_term = (
246         self.term_of_suspension_box.currentText()
247     )
248     if self.remedial_driving_class_checkBox.isChecked():
249         self.license_suspension_details.remedial_driving_class_required = True
250     else:
251         self.license_suspension_details.remedial_driving_class_required = False
252     self.case_information.license_suspension_details = (
253         self.license_suspension_details
254     )
255
256 @logger.catch
257 def add_other_condition_details(self):
258     """The method allows for adding other conditions based on free form text
259     entry."""
260     self.other_conditions_details.other_conditions_terms = (
261         self.other_conditions_plainTextEdit.toPlainText()
262     )
263     self.case_information.other_conditions_details = (
264         self.other_conditions_details
265     )
266
267 @logger.catch
268
269 def set_service_date(self, days_to_complete):
270     """Sets the community_service_date_to_complete_box based on the number
271     of days chosen in the community_service_date_to_complete_box."""

```

```

272     days_to_complete = int(
273         self.community_service_days_to_complete_box.currentText()
274     )
275     self.community_service_date_to_complete_box.setDate(
276         QDate.currentDate().addDays(days_to_complete)
277     )
278
279
280 class AmendOffenseDialog(BaseCriminalDialog, Ui_AmendOffenseDialog):
281     """The AddOffenseDialog is created when the amendOffenseButton is clicked on
282     the MinorMisdemeanorDialog screen. The case information is passed in
283     order to populate the case information banner.
284
285     The set_case_information_banner is an inherited method from BaseCriminalDialog."""
286     @logger.catch
287     def __init__(self, case_information, parent=None):
288         super().__init__(parent)
289         self.case_information = case_information
290         self.amend_offense_details = AmendOffenseDetails()
291         self.set_case_information_banner()
292         self.modify_view()
293
294     @logger.catch
295     def modify_view(self):
296         """The modify view method updates the view that is created on init.
297         Place items in this method that can't be added directly in QtDesigner
298         so that they don't need to be changed in the view file each time pyuic5
299         is run."""
300         offense_list = create_offense_list()
301         self.original_charge_box.addItems(offense_list)
302         self.amended_charge_box.addItems(offense_list)
303
304     @logger.catch
305     def amend_offense(self):
306         """Adds the data entered for the amended offense to the AmendOffenseDetails
307         object then points the case_information object to the AmendOffenseDetails
308         object."""
309         self.amend_offense_details.original_charge = (
310             self.original_charge_box.currentText()
311         )
312         self.amend_offense_details.amended_charge = (
313             self.amended_charge_box.currentText()
314         )
315         self.amend_offense_details.motion_disposition = (
316             self.motion_decision_box.currentText()
317         )
318         self.case_information.amend_offense_details = self.amend_offense_details

```