from kivymd.app import MDApp

from kivymd.uix.dialog import MDDialog

from kivymd.uix.boxlayout import MDBoxLayout

from kivymd.uix.pickers import MDDatePicker

from kivymd.uix.list import TwoLineAvatarIconListItem, ILeftBodyTouch

from kivymd.uix.selectioncontrol import MDCheckbox

from datetime import datetime

# To be added after creating the database

from database import Database

# Initialize db instance

db = Database()

class DialogContent(MDBoxLayout):

"""OPENS A DIALOG BOX THAT GETS THE TASK FROM THE USER"""

def \_\_init\_\_(self, \*\*kwargs):

super().\_\_init\_\_(\*\*kwargs)

self.ids.date\_text.text = str(datetime.now().strftime('%A %d %B %Y'))

def show\_date\_picker(self):

"""Opens the date picker"""

date\_dialog = MDDatePicker()

date\_dialog.bind(on\_save=self.on\_save)

date\_dialog.open()

def on\_save(self, instance, value, date\_range):

date = value.strftime('%A %d %B %Y')

self.ids.date\_text.text = str(date)

# After creating the database.py

class ListItemWithCheckbox(TwoLineAvatarIconListItem):

'''Custom list item'''

def \_\_init\_\_(self, pk=None, \*\*kwargs):

super().\_\_init\_\_(\*\*kwargs)

# state a pk which we shall use link the list items with the database primary keys

self.pk = pk

def mark(self, check, the\_list\_item):

'''mark the task as complete or incomplete'''

if check.active == True:

the\_list\_item.text = '[s]' + the\_list\_item.text + '[/s]'

db.mark\_task\_as\_complete(the\_list\_item.pk) # here

else:

the\_list\_item.text = str(db.mark\_task\_as\_incomplete(the\_list\_item.pk)) # Here

def delete\_item(self, the\_list\_item):

'''Delete the task'''

self.parent.remove\_widget(the\_list\_item)

db.delete\_task(the\_list\_item.pk) # Here

class LeftCheckbox(ILeftBodyTouch, MDCheckbox):

'''Custom left container'''

# Main App class

class MainApp(MDApp):

task\_list\_dialog = None

def build(self):

# Setting theme to my favorite theme

self.theme\_cls.primary\_palette = "Red"

# Showing the task dialog to add tasks

def show\_task\_dialog(self):

if not self.task\_list\_dialog:

self.task\_list\_dialog = MDDialog(

title="Create Task",

type="custom",

content\_cls=DialogContent(),

)

self.task\_list\_dialog.open()

def on\_start(self):

# Load the saved tasks and add them to the MDList widget when the application starts

try:

incompleted\_tasks, completed\_tasks = db.get\_tasks()

if incompleted\_tasks != []:

for task in incompleted\_tasks:

add\_task = ListItemWithCheckbox(pk=task[0], text=str(task[1]), secondary\_text=task[2])

self.root.ids.container.add\_widget(add\_task)

if completed\_tasks != []:

for task in completed\_tasks:

add\_task = ListItemWithCheckbox(pk=task[0], text='[s]' + str(task[1]) + '[/s]',

secondary\_text=task[2])

add\_task.ids.check.active = True

self.root.ids.container.add\_widget(add\_task)

except Exception as e:

print(e)

pass

def close\_dialog(self, \*args):

self.task\_list\_dialog.dismiss()

def add\_task(self, task, task\_date):

'''Add task to the list of tasks'''

# print(task.text, task\_date)

created\_task = db.create\_task(task.text, task\_date)

# return the created task details and create a list item

self.root.ids['container'].add\_widget(

ListItemWithCheckbox(pk=created\_task[0], text='[b]' + str(created\_task[1]) + '[/b]',

secondary\_text=created\_task[2]))

task.text = ''

if \_\_name\_\_ == '\_\_main\_\_':

app = MainApp()

app.run()