Министерство образования Республики Беларусь Учреждение образования БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления Кафедра Интеллектуальных информационных технологий

ОТЧЕТ

к лабораторной работе 2 по дисциплине «Проектирование программного обеспечения интеллектуальных систем»

Выполнил: И.Д.Кондратьева Проверил: С. В. Бутрин Цель: Изучить построение графического пользовательского интерфейса с использование библиотеки Kivy (https://kivy.org/)

Задача: разработать оконное приложение с одним главным окном и несколькими дочерними диалогами. Вызов диалогов осуществляется через соответствующие пункты меню. Команды меню должны дублироваться на панели инструментов.

Вариант:

Вариант 13

Номер поезда	Станция отправления	Дата и время отправления	Дата и время прибытия	Время в пути
				(не вводится руками, а вычисляется автоматически)

Условия поиска и удаления:

- по номеру поезда или дате отправления;
- по времени отправления (задается верхний и нижний предел) или времени прибытия (задается верхний и нижний предел);
- по станции отправления или станции прибытия;
- по времени в пути

В приложении имеются 4 главных экрана, позволяющие выполнять следующие операции:

- посмотреть таблицу
- найти отдельные записи таблицы по отдельным параметрам (по номеру поезда, станции отправления, станции прибытия, минимальной и максимальной дате и времени отправления, минмальной и максимальной дате и времени прибытия времени в пути)
- удаление по отдельным параметрам (по номеру поезда, станции отправления, станции прибытия, минимальной и максимальной дате и времени отправления, минмальной и максимальной дате и времени прибытия времени в пути)
- добавление новых записей (номер поезда, станция отправления, станция прибытия, дата и время отправления, дата и время прибытия)

Переключение между экранами осуществляется с помощью панели инструментов, реализованной с помощью объекта класса ActionBar. Примеры работы приложения:

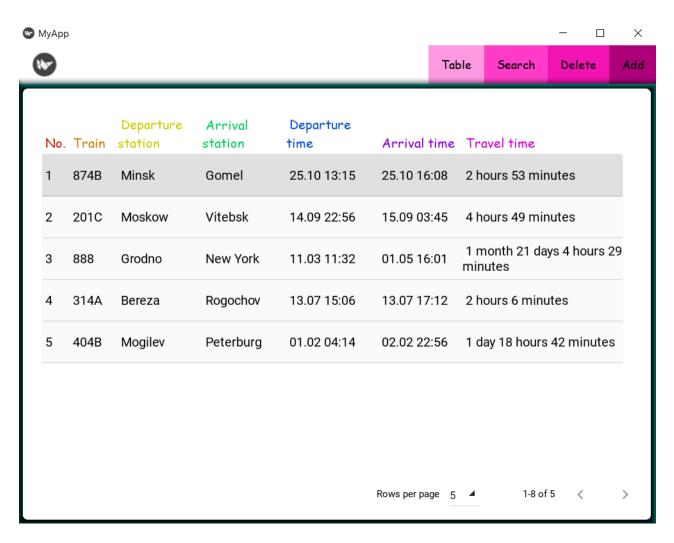


Рис. 1: Экран с таблицей



Рис. 2: Экран поиска



Рис. 3: Экран добавления новых записей в таблицу

Возможность переключения между экранами была реализована с помощью объекта класса ScreenManager.

Таблица с данными выводится с помощью объекта класса MDDataTable. Записи в таблице записаны на разных страницах, которые можно листать.

Рис. 4: Описание таблицы

Данные таблицы хранятся в xml файле, чтение осуществляется с помощью SAX парсера.

```
cLass TableHandler(sax.ContentHandler):
    def __init__(self):
    super(TableHandler,self).__init__()
           self.union = []
           self.train = [self.num]
     def startElement(self, name, attrs):
           self.current = nam
    def characters(self, content):
    if self.current == "number_of_train":
           self.current == "number_or_train :
    self.number_of_train = content
elif self.current == "departure":
    self.departure = content
elif self.current == "arrival":
    self.arrival = content
elif self.current == "departure_data_time":
                 self.departure_data_time = conten
           elif self.current == "arrival_data_time":
                 self.arrival_data_time = content
    def endElement(self, name):
    if self.current == 'number_of_train':
           self.train.append(self.number_of_train)
elif self.current == 'departure':
    self.train.append(self.departure)
           elif self.current == 'arrival':
           self.train.append(self.arrival)
elif self.current == 'departure_data_time':
           self.train.append(self.departure_data_time)
elif self.current == 'arrival_data_time':
                 self.train.append(self.arrival_data_time)
                  travel_time = time.time_of_travel(time.to_date(self.departure_data_time), time.to_date(self.arrival_data_time))
                 self.train.append(time.from_time_to_string(travel_time))
           self.current =
            if len(self.train) == 7:
    self.union.append(self.train)
    self.num = str(int(self.num)+1)
    self.train=[self.num]
```