```
 1 import sys
 2 from http.client import responses
 3
 4 import requests
 5 from PyQt5.QtWidgets import (QApplication, QWidget,
   QLabel, QLineEdit, QPushButton,QVBoxLayout)
 6
 7 from PyQt5.QtCore import Qt
 8 from urllib3.exceptions import HTTPError
 9
10
11 class WeatherApp(QWidget):
12     def __init__(self):
13         super().__init__()
14         self.city_label = QLabel("Enter City Name:",
   self)
15         self.city_input = QLineEdit(self)
16         self.get_weather_button = QPushButton("Get
   Weather", self)
17         self.temperature_label = QLabel(self)
18         self.emoji_label = QLabel(self)
19         self.description_label = QLabel(self)
20         self.initUI()
21
22     def initUI(self):
23         self.setWindowTitle("Weather App")
24
25
26         vbox = QVBoxLayout()
27
28         vbox.addWidget(self.city_label)
29         vbox.addWidget(self.city_input)
30         vbox.addWidget(self.get_weather_button)
31         vbox.addWidget(self.temperature_label)
32         vbox.addWidget(self.emoji_label)
33         vbox.addWidget(self.description_label)
34
35
36         self.setLayout(vbox)
37
38         self.city_label.setAlignment(Qt.AlignCenter)
```

```
39          self.city_input.setAlignment(Qt.AlignCenter)
40          self.temperature_label.setAlignment(Qt.
    AlignCenter)
41          self.emoji_label.setAlignment(Qt.AlignCenter)
42          self.description_label.setAlignment(Qt.
    AlignCenter)
43
44
45          self.city_label.setObjectName("city_label")
46          self.city_input.setObjectName("city_input")
47          self.get_weather_button.setObjectName("
    get_weather_button")
48          self.temperature_label.setObjectName("
    temperature_label")
49          self.emoji_label.setObjectName("emoji_label")
50          self.description_label.setObjectName("
    description_label")
51
52          self.setStyleSheet("""
53              QLabel, QPushButton{
54                  font-family: calibri;
55
56              }
57              QLabel#city_label{
58                  font-size: 40px;
59                  font-style: italic;
60              }
61              QLineEdit#city_input{
62                  font-size: 40px;
63              }
64              QPushButton#get_weather_button{
65                  font-size: 30px;
66                  font-weight: bold;
67              }
68              QLabel#temperature_label{
69                  font-size: 75px;
70
71              }
72              QLabel#emoji_label{
73                  font-size: 100px;
74                  font-family: Segoe UI emoji;
```

```
75              }
76              QLabel#description_label{
77                  font-size: 50px;
78
79              }
80          """)
81
82          self.get_weather_button.clicked.connect(self
    .get_weather)
83
84      def get_weather(self):
85
86          api_key = "7ebed01f53b7af8405cf4eb2358f0b2b"
87          city = self.city_input.text()
88          url = f"https://api.openweathermap.org/data/
    2.5/weather?q={city}&appid={api_key}"
89
90          try:
91              response = requests.get(url)
92              response.raise_for_status()
93              data = response.json()
94
95              if data["cod"] == 200:
96                  self.display_weather(data)
97
98          except requests.exceptions.HTTPError as
    http_error:
99              match response.status_code:
100                 case 400:
101                     self.display_error("Bad request:
    \nPlease check your input")
102                 case 401:
103                     self.display_error("Unauthorized
    :\nInvalid API Key")
104                 case 403:
105                     self.display_error("Forbidden:\n
    Access is Denied")
106                 case 404:
107                     self.display_error("Not Found:\n
    City not found")
108                 case 500:
```

```
109                     self.display_error("Internal
    server error:\nPlease try again later")
110                 case 502:
111                     self.display_error("Bad Gateway:
    \nInvalid response from server")
112                 case 503:
113                     self.display_error("Service
    Unavailable:\nServer is down")
114                 case 504:
115                     self.display_error("Gateway
    Timeout:\nNo response from the server")
116                 case 505:
117                     self.display_error(f"HTTP Error
    occured:\n{http_error}")
118
119         except requests.exceptions.ConnectionError:
120             self.display_error("Connection Error:\n
    Check Your internet connection")
121         except requests.exceptions.Timeout:
122             self.display_error("Timeout Error:\nThe
    request timed out")
123         except requests.exceptions.TooManyRedirects:
124             self.display_error("Too Many Redirects:\
    nCheck the URL")
125         except requests.exceptions.RequestException
    as req_error:
126             self.display_error(f"Request Error:\n{
    req_error}")
127
128     def display_error(self, message):
129         self.temperature_label.setStyleSheet("font-
    size: 30px;")
130         self.temperature_label.setText(message)
131         self.emoji_label.clear()
132         self.description_label.clear()
133
134     def display_weather(self, data):
135         self.temperature_label.setStyleSheet("font-
    size: 75px;")
136         temperature_k = data["main"]["temp"]
137         temperature_c = temperature_k - 273.15
```

```
        temperature_f = (temperature_k * 9/5 ) - 459
.67
        weather_id = data["weather"][0]["id"]
        weather_description = data["weather"][0]["
description"]


        self.temperature_label.setText(f"{
temperature_c:.0f}°C")
        self.emoji_label.setText(self.
get_weather_emoji(weather_id))
        self.description_label.setText(
weather_description)

    @staticmethod
    def get_weather_emoji(weather_id):

        if 200 <= weather_id <= 232:
            return "⛈️"
        elif 300 <= weather_id <= 321:
            return "🌦️"
        elif 500 <= weather_id <= 531:
            return "🌧️"
        elif 600 <= weather_id <= 622:
            return "❄️"
        elif 701 <= weather_id <= 741:
            return "🌫"
        elif weather_id == 762:
            return "🌋"
        elif weather_id == 781:
            return "🌪️"
        elif weather_id == 800:
            return "☀️"
        elif 801 <= weather_id <= 804:
            return "☁️"
        else:
            return ""

if __name__ == "__main__":
        app = QApplication(sys.argv)
        weather_app = WeatherApp()
```

```
174         weather_app.show()
175         sys.exit(app.exec_())
176
177
178
179
180
181
182
183
184
```