```
 1  using System;
 2  using System.Diagnostics;
 3  using System.Windows.Forms;
 4
 5  namespace ShutdownLibrary
 6  {
 7      public class ShutdownManager
 8      {
 9          private System.Timers.Timer countdownTimer;
10          private int countdownSeconds;
11          private Label statusLabel;
12          private RadioButton shutdownRadioButton;
13          private RadioButton hibernateRadioButton;
14          private RadioButton netflixRadioButton;
15          private TextBox timeTextBox;
16          private Button startButton;
17          private Button stopButton;
18          private RadioButton formatRadioButton;
19          private RadioButton selectedHourRadioButton;
20          private bool countdownStarted = false;
21
22          public ShutdownManager(Label statusLabel, RadioButton
              shutdownRadioButton, RadioButton hibernateRadioButton,
              RadioButton netflixRadioButton, TextBox timeTextBox, Button
              startButton, Button stopButton, RadioButton
              formatRadioButton, RadioButton selectedHourRadioButton)
23          {
24              this.statusLabel = statusLabel;
25              this.shutdownRadioButton = shutdownRadioButton;
26              this.hibernateRadioButton = hibernateRadioButton;
27              this.netflixRadioButton = netflixRadioButton;
28              this.timeTextBox = timeTextBox;
29              this.startButton = startButton;
30              this.stopButton = stopButton;
31              this.formatRadioButton = formatRadioButton;
32              this.selectedHourRadioButton = selectedHourRadioButton;
33
34              InitializeCountdownTimer();
35              AttachEventHandlers();
36          }
37
38          private void InitializeCountdownTimer()
39          {
40              countdownTimer = new System.Timers.Timer(1000);
41              countdownTimer.Elapsed += CountdownTimer_Tick;
42          }
43
44          private void AttachEventHandlers()
45          {
46              startButton.Click += StartButton_Click;
47              stopButton.Click += StopButton_Click;
48          }
49
```

```
50          private void StartButton_Click(object sender, EventArgs e)
51          {
52              try
53              {
54                  // Sprawdzenie, czy jakaś opcja została wybrana
55                  if (!shutdownRadioButton.Checked && !
                      hibernateRadioButton.Checked && !
                      netflixRadioButton.Checked)
56                  {
57                      ShowErrorMessage("Proszę wybrać jedną z opcji:
                      Shutdown, Hibernate lub Netflix.");
58                      return; // Przerwij działanie metody, jeśli żadna
                      opcja nie została wybrana
59                  }
60
61                  if (selectedHourRadioButton.Checked)
62                  {
63                      string[] timeParts = timeTextBox.Text.Split(':');
64
65                      if (timeParts.Length != 2)
66                      {
67                          ShowErrorMessage("Nieprawidłowy format czasu.
                      Wprowadź czas w formacie hh:mm.");
68                          return;
69                      }
70
71                      int targetHour, targetMinute;
72
73                      if (!int.TryParse(timeParts[0], out targetHour)
                      || !int.TryParse(timeParts[1], out targetMinute))
74                      {
75                          ShowErrorMessage("Nieprawidłowy format czasu.
                      Wprowadź czas w formacie hh:mm.");
76                          return;
77                      }
78
79                      // Sprawdzenie poprawności wprowadzonej godziny i
                      minuty
80                      if (targetHour < 0 || targetHour > 23 ||
                      targetMinute < 0 || targetMinute > 59)
81                      {
82                          ShowErrorMessage("Nieprawidłowa godzina lub
                      minuta.");
83                          return;
84                      }
85
86                      DateTime now = DateTime.Now;
87                      DateTime targetTime = new DateTime(now.Year,
                      now.Month, now.Day, targetHour, targetMinute, 0);
88
89                      if (targetTime <= now)
90                      {
91                          targetTime = targetTime.AddDays(1);
```

```
92                      }
93
94                      countdownSeconds = (int)(targetTime -
                    now).TotalSeconds;
95                  }
96                  else if (formatRadioButton.Checked)
97                  {
98                      countdownSeconds = ParseTimeInSeconds();
99                  }
100                 countdownStarted = true; // Ustawiamy flagę na true,
                      aby oznaczyć rozpoczęcie odliczania
101                 DisableRadioButtons(); // Wyłączamy przyciski radiowe
102
103                 countdownTimer.Start();
104             }
105             catch (FormatException)
106             {
107                 ShowErrorMessage("Nieprawidłowy format czasu. Wprowadź
                      czas w formacie hh:mm.");
108             }
109         }
110
111
112
113         private void StopButton_Click(object sender, EventArgs e)
114         {
115             countdownTimer.Stop();
116             UpdateStatusLabel("Odliczanie zatrzymane.");
117
118             countdownStarted = false; // Po zatrzymaniu odliczania
                  ustawiamy flagę z powrotem na false
119             EnableRadioButtons(); // Odblokowujemy przyciski radiowe
120         }
121
122         private int ParseTimeInSeconds()
123         {
124             string timeText = timeTextBox.Text;
125             int hours = 0, minutes = 0, seconds = 0;
126
127             // Sprawdź, czy tekst zawiera dwukropki
128             if (timeText.Contains(":"))
129             {
130                 // Podziel tekst na części przed i po dwukropku
131                 string[] timeParts = timeText.Split(':');
132
133                 // Sprawdź, czy podano odpowiednią liczbę części czasu
134                 if (timeParts.Length == 2 || timeParts.Length == 3)
135                 {
136                     // Obsługa formatu mm:ss
137                     if (timeParts.Length == 2)
138                     {
139                         if (!int.TryParse(timeParts[0], out minutes)
                        || !int.TryParse(timeParts[1], out seconds))
```

```
140                         {
141                             ShowErrorMessage("Nieprawidłowy format czasu. Wprowadź czas w formacie hh:mm:ss lub mm:ss lub ss.");
142                             return 0;
143                         }
144                     }
145                     // Obsługa formatu hh:mm:ss
146                     else if (timeParts.Length == 3)
147                     {
148                         if (!int.TryParse(timeParts[0], out hours) || !int.TryParse(timeParts[1], out minutes) || !int.TryParse(timeParts[2], out seconds))
149                         {
150                             ShowErrorMessage("Nieprawidłowy format czasu. Wprowadź czas w formacie hh:mm:ss lub mm:ss lub ss.");
151                             return 0;
152                         }
153                     }
154                 }
155                 else
156                 {
157                     ShowErrorMessage("Nieprawidłowy format czasu. Wprowadź czas w formacie hh:mm:ss lub mm:ss lub ss.");
158                     return 0;
159                 }
160             }
161             // Jeśli nie ma dwukropków
162             else
163             {
164                 // Obsługa formatu ss
165                 if (!int.TryParse(timeText, out seconds))
166                 {
167                     ShowErrorMessage("Nieprawidłowy format czasu. Wprowadź czas w formacie hh:mm:ss lub mm:ss lub ss.");
168                     return 0;
169                 }
170             }
171
172             // Sprawdź poprawność wartości
173             if (hours < 0 || minutes < 0 || seconds < 0 || minutes >= 60 || seconds >= 60)
174             {
175                 ShowErrorMessage("Nieprawidłowy czas. Wprowadź czas w formacie hh:mm:ss lub mm:ss, gdzie mm i ss to liczby od 0 do 59.");
176                 return 0;
177             }
178
179             // Zwróć czas w sekundach
```

```
180             return hours * 3600 + minutes * 60 + seconds;
181         }
182
183         private void CountdownTimer_Tick(object sender, EventArgs e)
184         {
185             if (countdownSeconds > 0)
186             {
187                 UpdateRemainingTimeStatusLabel();
188                 countdownSeconds--;
189             }
190             else
191             {
192                 ExecuteAction();
193             }
194         }
195
196         private void UpdateRemainingTimeStatusLabel()
197         {
198             string mode = GetSelectedMode();
199             TimeSpan remainingTime = TimeSpan.FromSeconds
                  (countdownSeconds);
200             UpdateStatusLabel($"{mode} nastąpi za: \n {FormatTime
                  (remainingTime)}");
201         }
202
203         private void ExecuteAction()
204         {
205             string mode = GetSelectedMode();
206             if (shutdownRadioButton.Checked)
207             {
208                 ShutdownComputer();
209                 UpdateStatusLabel($"Akcja wykonana. \n Komputer
                      zostanie wyłączony");
210             }
211             else if (hibernateRadioButton.Checked)
212             {
213                 HibernateComputer();
214                 UpdateStatusLabel($"Akcja wykonana. \n Komputer
                      zostanie zahibernowany");
215             }
216             else if (netflixRadioButton.Checked)
217             {
218                 ShutdownNetflix();
219                 UpdateStatusLabel($"Akcja wykonana. \n Aplikacja
                      Netflix zostanie zamknięta.");
220             }
221             else
222             {
223                 UpdateStatusLabel("Żadna akcja nie została wybrana.");
224             }
225
226             countdownTimer.Stop();
227         }
```

```
228
229         private string GetSelectedMode()
230         {
231             if (shutdownRadioButton.Checked)
232             {
233                 return "Zamknięcie systemu";
234             }
235             else if (hibernateRadioButton.Checked)
236             {
237                 return "Hibernacja systemu";
238             }
239             else if (netflixRadioButton.Checked)
240             {
241                 return "Zamknięcie aplikacji Netflix";
242             }
243             else
244             {
245                 return "Nieznana akcja";
246             }
247         }
248
249         private string FormatTime(TimeSpan timeSpan)
250         {
251             if (timeSpan.Hours > 0)
252             {
253                 return $"{timeSpan.Hours} godzin {timeSpan.Minutes}     ↵
                      minut {timeSpan.Seconds} sekund";
254             }
255             else if (timeSpan.Minutes > 0)
256             {
257                 return $"{timeSpan.Minutes} minut {timeSpan.Seconds}    ↵
                      sekund";
258             }
259             else
260             {
261                 return $"{timeSpan.Seconds} sekund";
262             }
263         }
264
265         private void ShutdownComputer()
266         {
267             Process.Start("shutdown", "/s /f /t 1");
268         }
269
270         private void HibernateComputer()
271         {
272             Process.Start("shutdown", "/h");
273         }
274
275         private void ShutdownNetflix()
276         {
277             Process.Start("taskkill", "/IM ApplicationFrameHost.exe");
278         }
```

```
279
280         private void UpdateStatusLabel(string message)
281         {
282             if (statusLabel.InvokeRequired)
283             {
284                 statusLabel.Invoke(new Action<string>
                      (UpdateStatusLabel), message);
285             }
286             else
287             {
288                 statusLabel.Text = message;
289             }
290         }
291
292         // Metoda do blokowania przycisków radiowych
293         private void DisableRadioButtons()
294         {
295             shutdownRadioButton.Enabled = false;
296             hibernateRadioButton.Enabled = false;
297             netflixRadioButton.Enabled = false;
298             selectedHourRadioButton.Enabled = false;
299             formatRadioButton.Enabled = false;
300         }
301
302         // Metoda do odblokowywania przycisków radiowych
303         private void EnableRadioButtons()
304         {
305             shutdownRadioButton.Enabled = true;
306             hibernateRadioButton.Enabled = true;
307             netflixRadioButton.Enabled = true;
308             selectedHourRadioButton.Enabled = true;
309             formatRadioButton.Enabled = true;
310         }
311
312         private void ShowErrorMessage(string message)
313         {
314             MessageBox.Show(message, "Błąd", MessageBoxButtons.OK,
                  MessageBoxIcon.Warning);
315         }
316     }
317 }
318
319
```