```java
package com.ProjectClassTask;

import java.awt.AWTException;

public class BaseClass {
    public static WebDriver driver;
    // WEBELEMENT METHODS

    public static void userInput(WebElement element, String value) {
        element.sendKeys(value);
    }

    public static void clickTheElement(WebElement element) {
        element.click();
    }

    public static void booleanOptions(String name, WebElement element) {

        if (name.equalsIgnoreCase("display")) {
            boolean displayed = element.isDisplayed();
            System.out.println(displayed);
        } else if (name.equalsIgnoreCase("enable")) {
            boolean enabled = element.isEnabled();
            System.out.println(enabled);
        } else if (name.equalsIgnoreCase("select")) {
            boolean selected = element.isSelected();
            System.out.println(selected);
        } else {
            System.out.println("Invalid Element");
        }
    }

    public static void textOntheElement(WebElement element) {
        String text = element.getText();
        System.out.println(text);
    }

    public static void attributeOfTheElement(WebElement element, String value) {
        String attribute = element.getAttribute(value);
        System.out.println(attribute);
    }

    public static void clearOfTheElement(WebElement element) {
        element.clear();
    }

    public static void singleDropDown(String name, WebElement element, String value) {

        Select s = new Select(element);
        if (name.equalsIgnoreCase("text")) {
            s.selectByVisibleText(value);
        } else if (name.equalsIgnoreCase("value")) {
            s.selectByValue(value);
        } else if (name.equalsIgnoreCase("index")) {
            int val = Integer.parseInt(value); // string convert into integer
            s.selectByIndex(val);
        } else if (name.equalsIgnoreCase("deSelectValue")) {
```

```
 86             s.deselectByValue(value);
 87         } else if (name.equalsIgnoreCase("deSelectIndex")) {
 88             int vale = Integer.parseInt(value);
 89             s.deselectByIndex(vale);
 90         } else if (name.equalsIgnoreCase("deSelectText")) {
 91             s.deselectByVisibleText(value);
 92         } else if (name.equalsIgnoreCase("deSelectAll")) {
 93             s.deselectAll();
 94         } else {
 95             System.out.println("Invalid Selection");
 96         }
 97 
 98     }
 99 
100     public static void mouseActions(WebElement element, String name) {
101 
102         Actions a = new Actions(driver);
103 
104         if (name.equalsIgnoreCase("click")) {
105             a.click(element).perform();
106             ;
107         } else if (name.equalsIgnoreCase("doubleClick")) {
108             a.doubleClick(element).perform();
109             ;
110         } else if (name.equalsIgnoreCase("rightClick")) {
111             a.contextClick(element).perform();
112             ;
113         } else if (name.equalsIgnoreCase("move")) {
114             a.moveToElement(element).perform();
115         } else {
116             System.out.println("invalid action");
117         }
118     }
119 
120     public static void dragAndDrop(String name, WebElement source, WebElement desti) {
121         Actions aa = new Actions(driver);
122         if (name.equalsIgnoreCase("dragAndDrop")) {
123             aa.dragAndDrop(source, desti).perform();
124         } else if (name.equalsIgnoreCase("clickAndHold")) {
125             aa.clickAndHold(source).moveToElement(desti).release(desti).perform();
126         }
127     }
128 
129     public static void multipleDropDownOptions(WebElement element, String name) {
130 
131         Select ss = new Select(element);
132 
133         boolean multiple = ss.isMultiple();
134         System.out.println(multiple);
135 
136         if (name.equalsIgnoreCase("optios")) {
137             List<WebElement> options = ss.getOptions();
138 
139             for (int i = 0; i < options.size(); i++) {
140                 String text = options.get(i).getText();
141                 System.out.println(text);
142             }
```

```
143         } else if (name.equalsIgnoreCase("allOptions")) {
144             List<WebElement> allSelectedOptions = ss.getAllSelectedOptions();
145 
146             for (WebElement allOptionText : allSelectedOptions) {
147 
148                 String text1 = allOptionText.getText();
149                 System.out.println(text1);
150             }
151         } else if (name.equalsIgnoreCase("firstSelected")) {
152             WebElement firstSelectedOption = ss.getFirstSelectedOption();
153             System.out.println(firstSelectedOption.getText());
154         }
155     }
156 
157     // WEBDRIVER METHOD
158 
159     public static WebDriver browserLaunch(String browser) {
160 
161         if (browser.equalsIgnoreCase("chrome")) {
162             System.setProperty("webdriver.chrome.driver",
163                     "D:\\ecl\\javatask\\ProjectClassTask\\driver\\chromedriver.exe");
164             driver = new ChromeDriver();
165         } else if (browser.equalsIgnoreCase("firefox")) {
166             System.setProperty("webdriver.gecko.driver",
167                     ".\\driver\\geckodriver.exe");
168             driver = new FirefoxDriver();
169         } else if (browser.equalsIgnoreCase("edge")) {
170             String property = System.getProperty("user.dir");
171             System.out.println(property);
172             System.setProperty("webdriver.edge.driver",""+property +"\\driver\
    \msedgedriver.exe");
173             driver = new EdgeDriver();
174         } else {
175             System.out.println("ivalid browser");
176         }
177         driver.manage().window().maximize();
178         return driver;
179     }
180 
181     public static void launchURL(String option, String url) {
182 
183         if (option.equalsIgnoreCase("getURL")) {
184             driver.get(url);
185         }
186 
187         else if (option.equalsIgnoreCase("navigateURL")) {
188             driver.navigate().to(url);
189         } else {
190             System.out.println("invalid url");
191         }
192 
193     }
194 
195     public static void textPage(String option) {
196         if (option.equalsIgnoreCase("getTitle")) {
197             String title = driver.getTitle();
198             System.out.println(title);
```

```
199         } else if (option.equalsIgnoreCase("getTitle")) {
200             String urlText = driver.getCurrentUrl();
201             System.out.println(urlText);
202         }
203 
204     }
205 
206     public static void navigatePage(String option) {
207         if (option.equalsIgnoreCase("forward")) {
208             driver.navigate().forward();
209         } else if (option.equalsIgnoreCase("back")) {
210             driver.navigate().back();
211         } else if (option.equalsIgnoreCase("refresh")) {
212             driver.navigate().refresh();
213         }
214     }
215 
216     public static void sleepPage(int sec) throws InterruptedException {
217         Thread.sleep(sec);
218     }
219 
220     public static void implicitThePage(int sec) {
221 
222         driver.manage().timeouts().implicitlyWait(sec, TimeUnit.SECONDS);
223     }
224 
225     public static void explicitWaitOfThePage(int sec, String option, WebElement element) {
226 
227         WebDriverWait wait = new WebDriverWait(driver, 20);
228 
229         if (option.equalsIgnoreCase("visible")) {
230             wait.until(ExpectedConditions.visibilityOf(element));
231         } else if (option.equalsIgnoreCase("clickable")) {
232             wait.until(ExpectedConditions.elementToBeClickable(element));
233 
234         }
235     }
236 
237     public static void takesScreen(String fileName) throws IOException {
238 
239         TakesScreenshot ts = (TakesScreenshot) driver;
240 
241         File source = ts.getScreenshotAs(OutputType.FILE);
242 
243         File desti = new File("D:\\ecl\\javatask\\SeleniumTask\\screenshot\\" + fileName +
    ".png");
244         // FileUtils.copyFile(source, desti);
245         FileHandler.copy(source, desti);
246     }
247 
248     public static void keyBord(String option) throws AWTException {
249 
250         Robot r = new Robot();
251 
252         if (option.equalsIgnoreCase("clickControl")) {
253             r.keyPress(KeyEvent.VK_CONTROL);
254         } else if (option.equalsIgnoreCase("releaseControl")) {
```

```
255             r.keyRelease(KeyEvent.VK_CONTROL);
256         } else if (option.equalsIgnoreCase("clickEnter")) {
257             r.keyPress(KeyEvent.VK_ENTER);
258         } else if (option.equalsIgnoreCase("releaseEnter")) {
259             r.keyRelease(KeyEvent.VK_ENTER);
260         } else if (option.equalsIgnoreCase("clickA")) {
261             r.keyPress(KeyEvent.VK_A);
262         } else if (option.equalsIgnoreCase("releaseA")) {
263             r.keyRelease(KeyEvent.VK_A);
264         }
265     }
266 
267     public static void alertPage(String option) {
268 
269         Alert alert = driver.switchTo().alert();
270 
271         if (option.equalsIgnoreCase("accept")) {
272             alert.accept();
273         }
274 
275         else if (option.equalsIgnoreCase("dismiss")) {
276             alert.dismiss();
277         } else if (option.equalsIgnoreCase("text")) {
278 
279             String text = alert.getText();
280             System.out.println();
281         }
282 
283     }
284 
285     public static void alertSendkey(String value) {
286 
287         Alert alert1 = driver.switchTo().alert();
288 
289         alert1.sendKeys(value);
290         alert1.accept();
291     }
292 
293     public static void javaScriptScroll(WebElement element, String option) {
294 
295         try {
296             JavascriptExecutor js = (JavascriptExecutor) driver;
297 
298             if (option.equalsIgnoreCase("scroll")) {
299                 js.executeScript("arguements[0].scrollIntoView();", element);
300             }
301 
302             else if (option.equals("click")) {
303                 js.executeScript("arguments[0].click();", element);
304             }
305 
306         } catch (Exception e) {
307             e.printStackTrace();
308             System.out.println(e);
309         }
310     }
311 
```

```
312     public static void jsScrollPixel(String xDirection, String yDirection) {
313         try {
314             JavascriptExecutor js1 = (JavascriptExecutor) driver;
315             js1.executeScript("window.scrollBy(" + xDirection + "," + yDirection + ")");
316         }
317 
318         catch (Exception e) {
319             e.printStackTrace();
320             System.out.println(e);
321         }
322     }
323 
324     public static void scrollBottomPage(String option) {
325 
326         try {
327             JavascriptExecutor js2 = (JavascriptExecutor) driver;
328             if (option.equalsIgnoreCase("scrollDown")) {
329                 js2.executeScript("window.scrollBy(0,document.body.ScrollHeight);");// entire
    page scroll down
330             } else if (option.equalsIgnoreCase("scrollUp")) {
331                 js2.executeScript("window.scrollBy(0,-document.body.ScrollHeight);"); //
    entire page scroll up
332             }
333         } catch (Exception e) {
334             e.printStackTrace();
335             System.out.println(e);
336         }
337     }
338 
339     public static String singleWindowHandle() {
340 
341         String windowHandle = driver.getWindowHandle();
342 
343         System.out.println(windowHandle);
344 
345         return windowHandle;
346 
347     }
348 
349     public static void multipleWindowHandle(String firstWindowId) {
350 
351         Set<String> windowHandles = driver.getWindowHandles();
352 
353         System.out.println(windowHandles);
354 
355         System.out.println(windowHandles.size());
356 
357         for (String allID : windowHandles) {
358 
359             if (!(allID == firstWindowId)) {
360                 driver.switchTo().window(allID);
361                 break;
362             }
363         }
364     }
365 
366     public static void frameSize(String value) {
```

```
367
368         List<WebElement> findElements = driver.findElements(By.tagName(value));
369
370         System.out.println(findElements.size());
371
372     }
373
374     public static void frameSelection(String option,String value) {
375
376         if (option.equalsIgnoreCase("index")) {
377             int indexVal = Integer.parseInt(value);
378             driver.switchTo().frame(indexVal);
379         }
380         else if (option.equalsIgnoreCase("id")) {
381             driver.switchTo().frame(value);
382
383         }
384     }
385     public static void frameSelectionElement(WebElement element) {
386
387             driver.switchTo().frame(element);
388         }
389     public static void frameParentDefault(String option,String value) {
390
391         if (option.equalsIgnoreCase("parent")) {
392             driver.switchTo().parentFrame();
393         }
394         else if (option.equalsIgnoreCase("defaultContenet")) {
395             driver.switchTo().defaultContent();
396         }
397     }
398
399     public static void multiWindow2() {
400         Set<String> windowHandles = driver.getWindowHandles();
401         Iterator iterator = windowHandles.iterator();
402         String firstWin = (String) iterator.next();
403         String secondWin = (String) iterator.next();
404         driver.switchTo().window(secondWin);
405     }
406
407 }
408
409
```