SOFTWARE ENGINEERING PROJECT

NOTIFICATION ALERT SYSTEM

GROUP MEMBERS:

SAHIL SONKER	(202251115)
TANUJ SAINI	(202251141)
SAPNA MEHAR	(202252336)
VIVEK YADAV	(202251159)
RUCHIR KUMAR	(202252334)

PROJECT DESCRIPTION

NOTIFICATION ALERT SYSTEM is a web application designed to streamline your task management process while providing additional features such as weather forecasting and task reminders. The application consists of several key components:

Login and Signup Page: Users can create accounts or log in securely to access the application's features.

TO-DO List: The TO-DO List page allows users to input tasks categorized as indoor or outdoor activities. Users can add, edit, and delete tasks as needed.

Task Reminder Alarm: Users can set reminders for their tasks. When the time arrives, the application will notify the user through an alarm or notification.

Calendar Integration: The application includes a calendar feature where users can schedule tasks by setting specific dates. This calendar syncs with the task list for better organization.

Weather Tab: The Weather tab provides users with current weather conditions and forecasts for their location. Users can also search for weather forecasts in any location worldwide.

Map Integration: The Weather tab includes a map feature that displays the user's location and weather information. Users can interact with the map to explore weather conditions in different areas.

Stop Watch: The stop watch will be helpful to set a target and complete the task within the time frame.

Key Features:

User authentication and authorization for secure access.

Task management with categorization (indoor/outdoor), editing, and deletion capabilities.

Reminders/alerts for upcoming tasks.

Calendar integration for scheduling tasks.

Weather forecast display for the user's location and worldwide search functionality.

Interactive map for visualizing weather conditions.

Technologies Used:

Frontend: HTML, CSS, JavaScript

Backend: Firebase, JS

Database: FirerbaseDB for storing user data and task information

Weather API: Integration with a weather API like OpenWeatherMap for real-time weather data

Map API: Integration with a mapping API like Google Maps for displaying weather information on the map

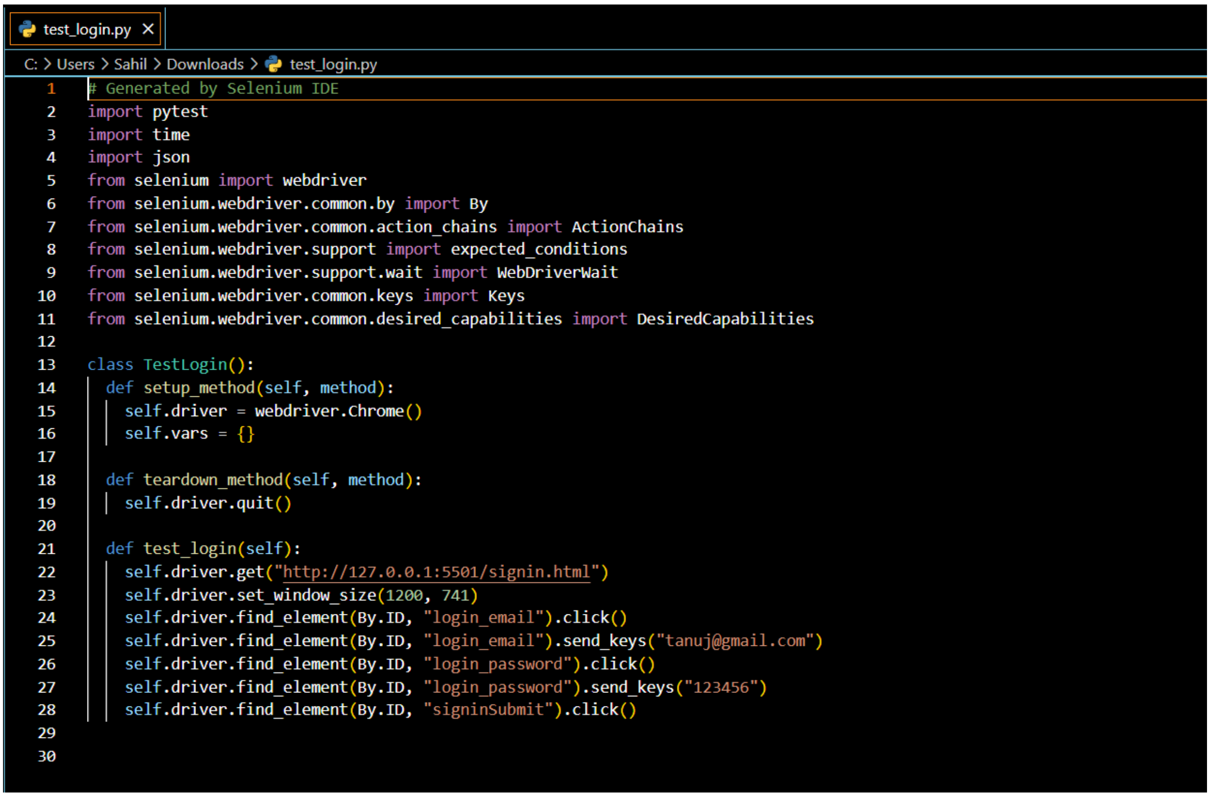
Additional Considerations:

Error handling and validation to ensure data integrity and a smooth user experience.

Testing to identify and address any bugs or issues.

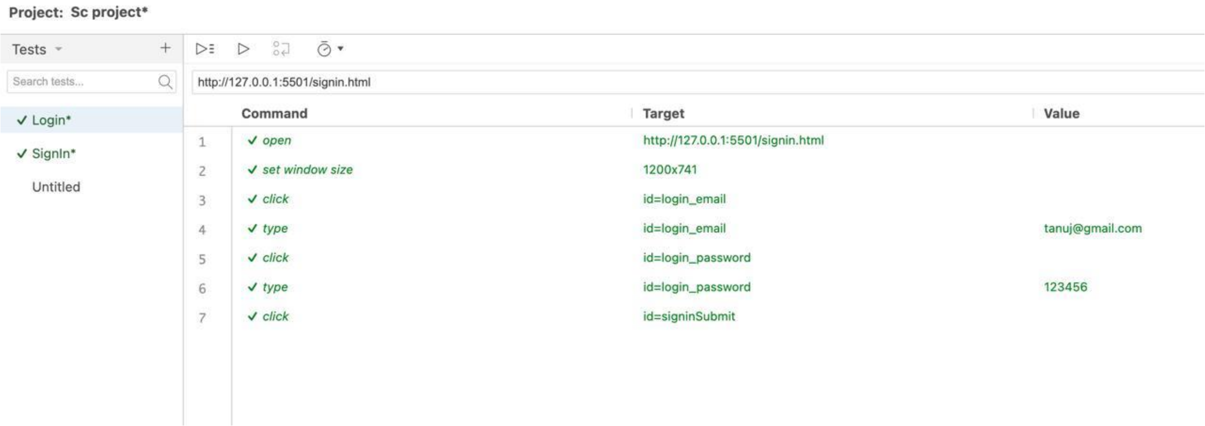
This project aims to provide users with a comprehensive task management solution enhanced by weather forecasting functionality, ultimately improving productivity and organization in daily activities.

Login:



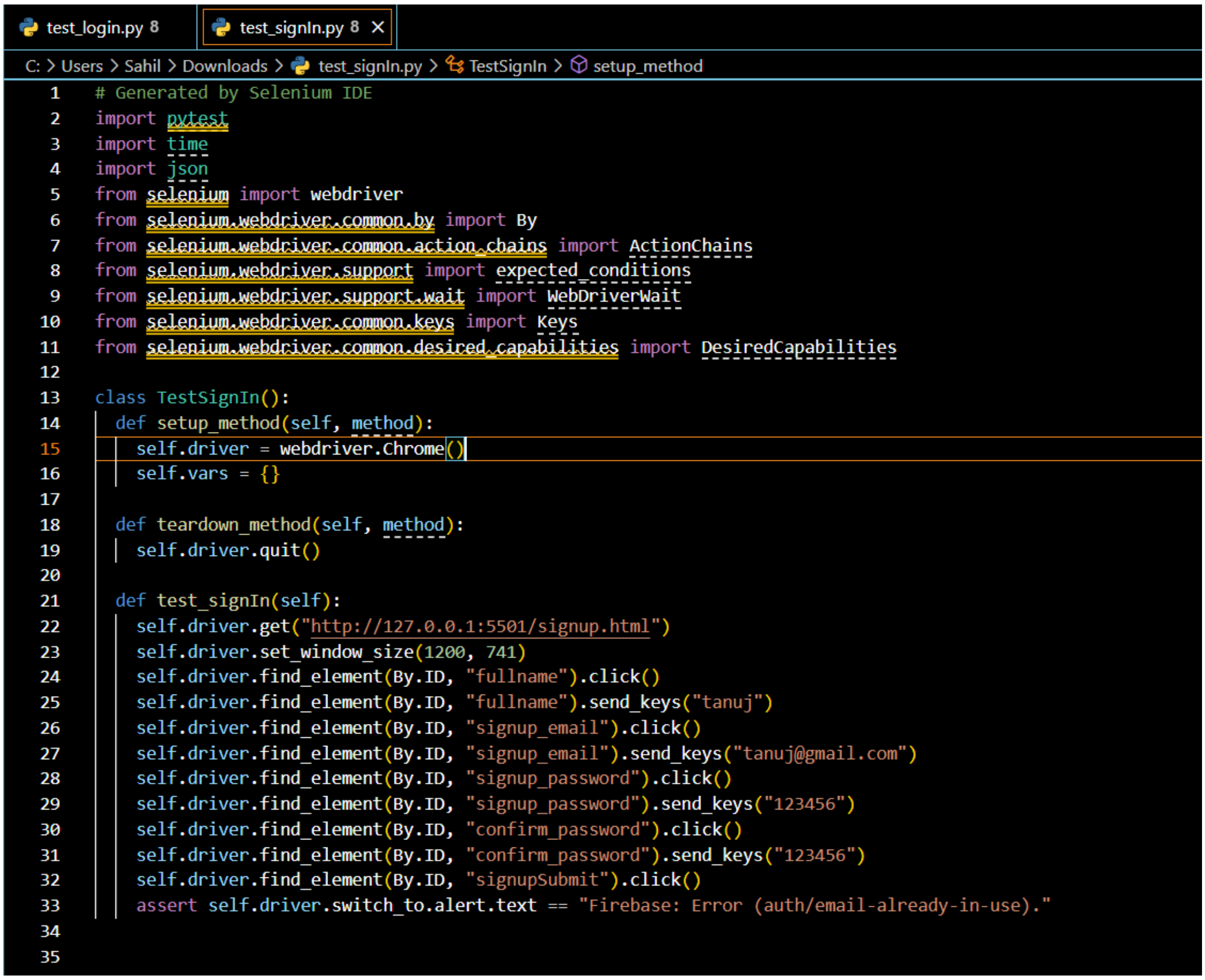
```python
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

class TestLogin():
  def setup_method(self, method):
    self.driver = webdriver.Chrome()
    self.vars = {}

  def teardown_method(self, method):
    self.driver.quit()

  def test_login(self):
    self.driver.get("http://127.0.0.1:5501/signin.html")
    self.driver.set_window_size(1200, 741)
    self.driver.find_element(By.ID, "login_email").click()
    self.driver.find_element(By.ID, "login_email").send_keys("tanuj@gmail.com")
    self.driver.find_element(By.ID, "login_password").click()
    self.driver.find_element(By.ID, "login_password").send_keys("123456")
    self.driver.find_element(By.ID, "signinSubmit").click()
```

Output:



	Command	Target	Value
1	✓ open	http://127.0.0.1:5501/signin.html	
2	✓ set window size	1200x741	
3	✓ click	id=login_email	
4	✓ type	id=login_email	tanuj@gmail.com
5	✓ click	id=login_password	
6	✓ type	id=login_password	123456
7	✓ click	id=signinSubmit	

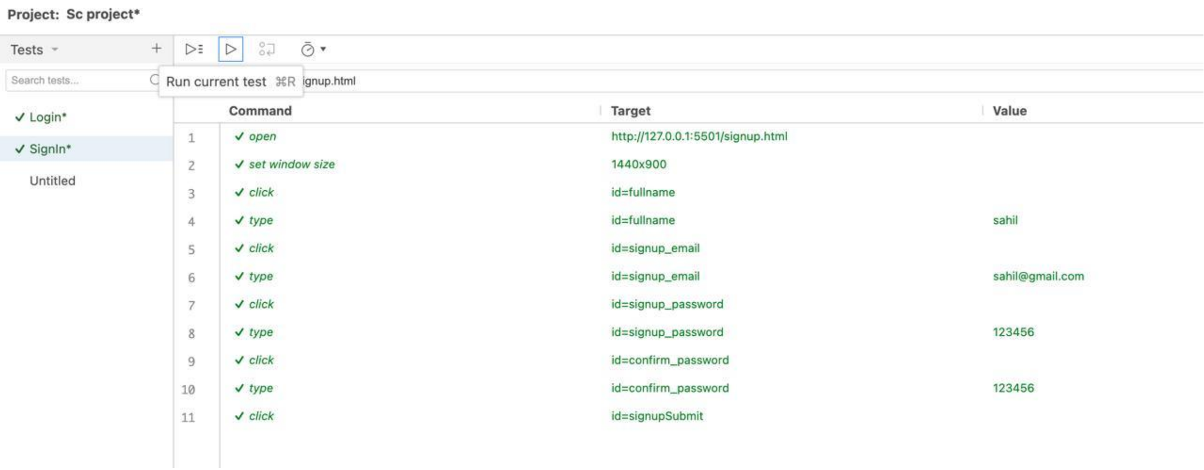
SignIn:



test_login.py 8
test_signIn.py 8 ×
C: > Users > Sahil > Downloads > test_signIn.py > TestSignIn > setup_method

```python
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

class TestSignIn():
  def setup_method(self, method):
    self.driver = webdriver.Chrome()
    self.vars = {}

  def teardown_method(self, method):
    self.driver.quit()

  def test_signIn(self):
    self.driver.get("http://127.0.0.1:5501/signup.html")
    self.driver.set_window_size(1200, 741)
    self.driver.find_element(By.ID, "fullname").click()
    self.driver.find_element(By.ID, "fullname").send_keys("tanuj")
    self.driver.find_element(By.ID, "signup_email").click()
    self.driver.find_element(By.ID, "signup_email").send_keys("tanuj@gmail.com")
    self.driver.find_element(By.ID, "signup_password").click()
    self.driver.find_element(By.ID, "signup_password").send_keys("123456")
    self.driver.find_element(By.ID, "confirm_password").click()
    self.driver.find_element(By.ID, "confirm_password").send_keys("123456")
    self.driver.find_element(By.ID, "signupSubmit").click()
    assert self.driver.switch_to.alert.text == "Firebase: Error (auth/email-already-in-use)."
```

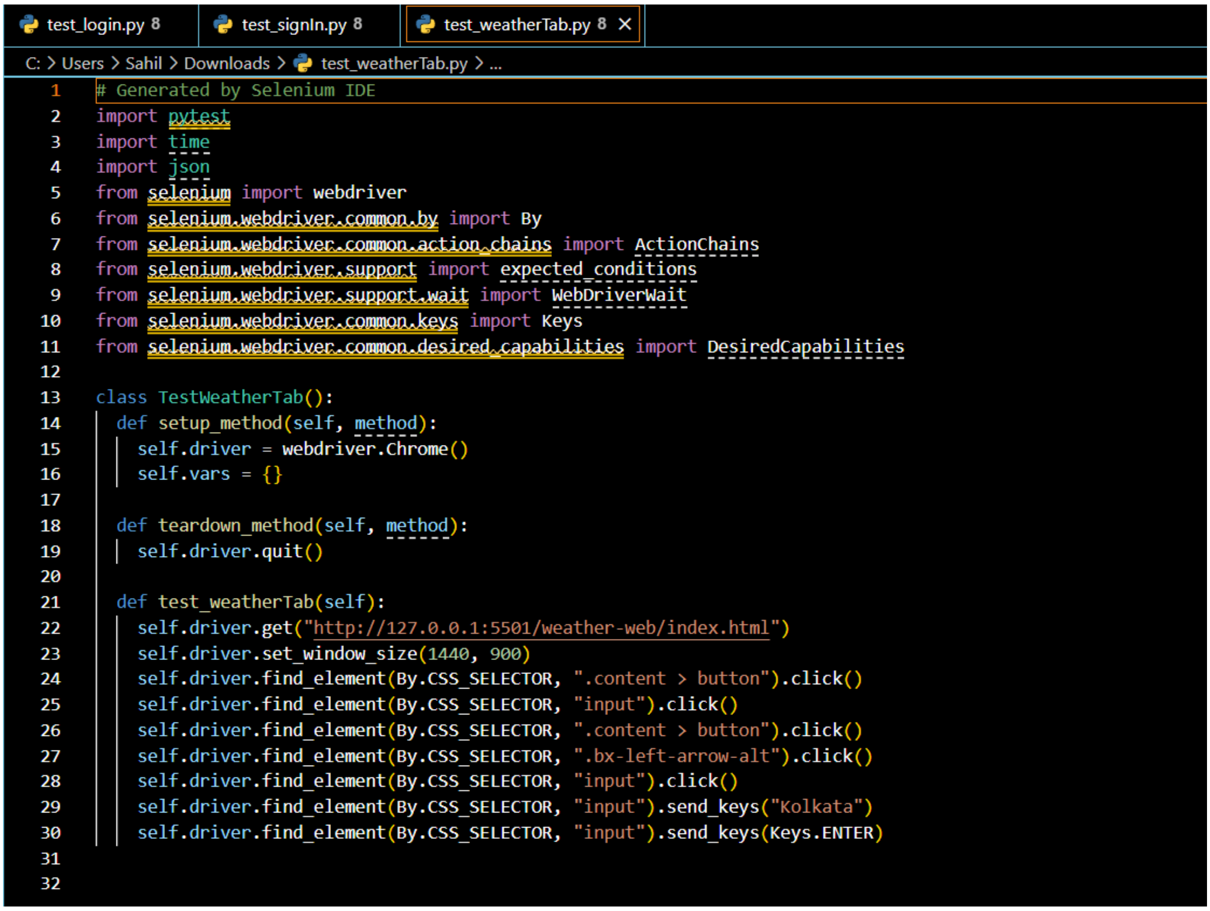
Output:



✓ Login*
✓ SignIn*
Untitled

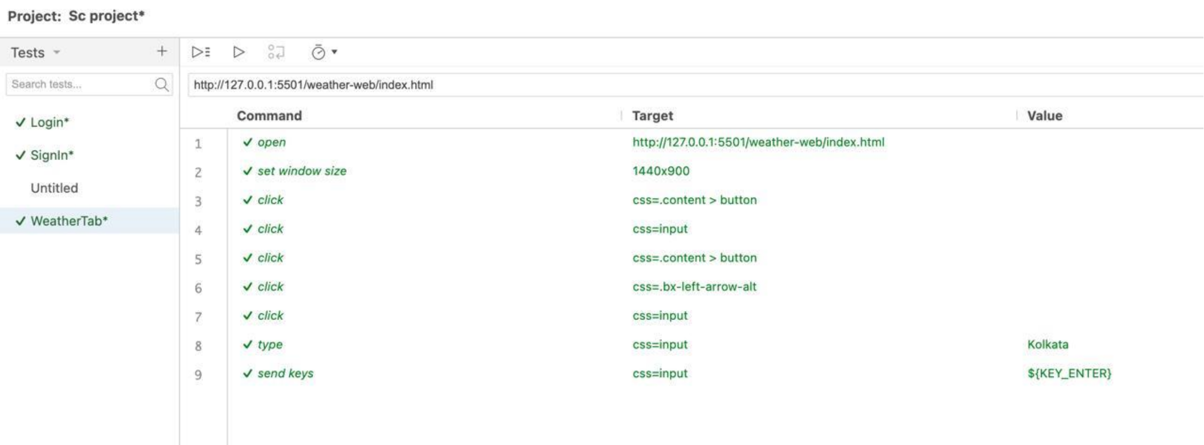
	Command	Target	Value
1	✓ open	http://127.0.0.1:5501/signup.html	
2	✓ set window size	1440x900	
3	✓ click	id=fullname	
4	✓ type	id=fullname	sahil
5	✓ click	id=signup_email	
6	✓ type	id=signup_email	sahil@gmail.com
7	✓ click	id=signup_password	
8	✓ type	id=signup_password	123456
9	✓ click	id=confirm_password	
10	✓ type	id=confirm_password	123456
11	✓ click	id=signupSubmit	

Weather Tab:



```python
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

class TestWeatherTab():
  def setup_method(self, method):
    self.driver = webdriver.Chrome()
    self.vars = {}

  def teardown_method(self, method):
    self.driver.quit()

  def test_weatherTab(self):
    self.driver.get("http://127.0.0.1:5501/weather-web/index.html")
    self.driver.set_window_size(1440, 900)
    self.driver.find_element(By.CSS_SELECTOR, ".content > button").click()
    self.driver.find_element(By.CSS_SELECTOR, "input").click()
    self.driver.find_element(By.CSS_SELECTOR, ".content > button").click()
    self.driver.find_element(By.CSS_SELECTOR, ".bx-left-arrow-alt").click()
    self.driver.find_element(By.CSS_SELECTOR, "input").click()
    self.driver.find_element(By.CSS_SELECTOR, "input").send_keys("Kolkata")
    self.driver.find_element(By.CSS_SELECTOR, "input").send_keys(Keys.ENTER)
```

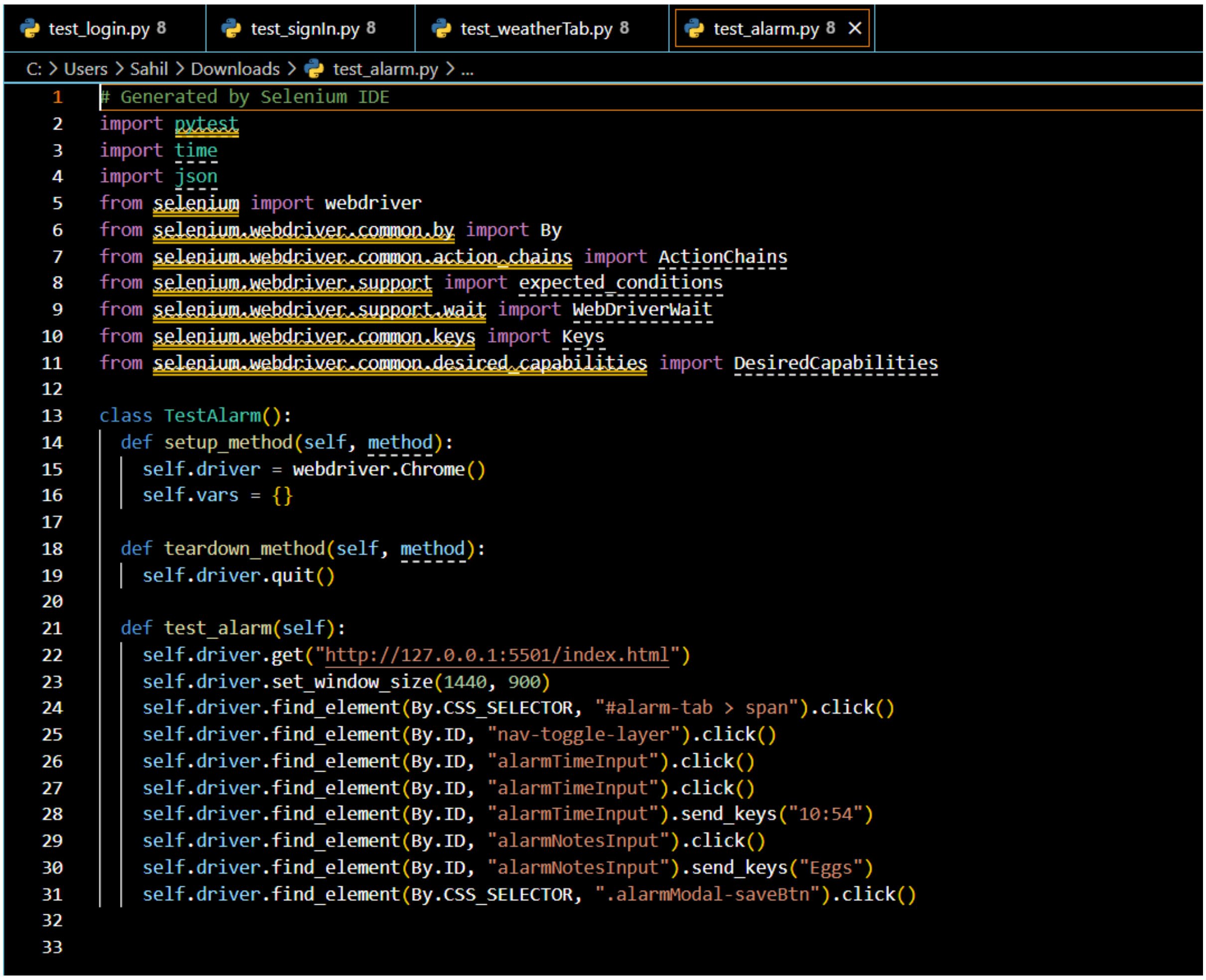
Output:



http://127.0.0.1:5501/weather-web/index.html

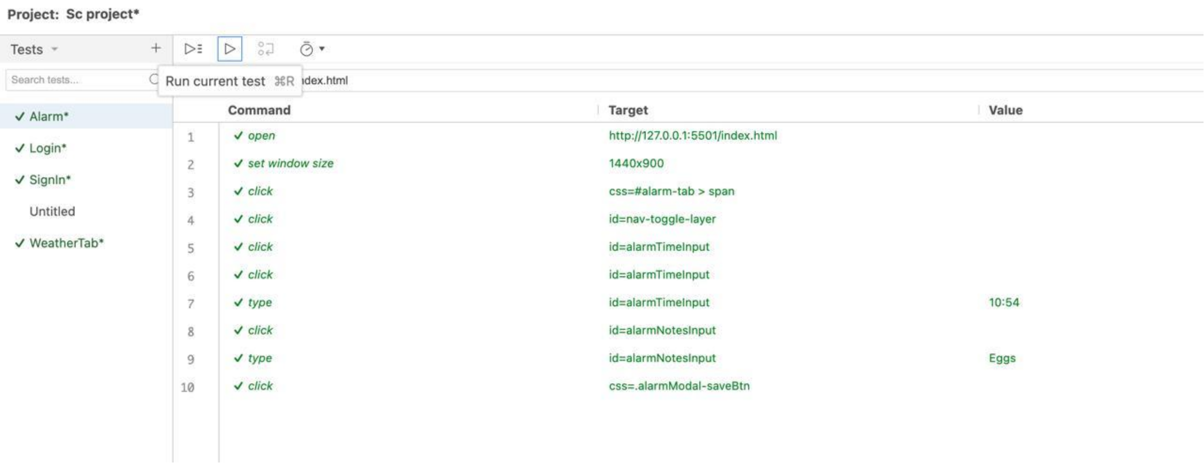
	Command	Target	Value
1	✓ open	http://127.0.0.1:5501/weather-web/index.html	
2	✓ set window size	1440x900	
3	✓ click	css=.content > button	
4	✓ click	css=input	
5	✓ click	css=.content > button	
6	✓ click	css=.bx-left-arrow-alt	
7	✓ click	css=input	
8	✓ type	css=input	Kolkata
9	✓ send keys	css=input	${KEY_ENTER}

Alarm:



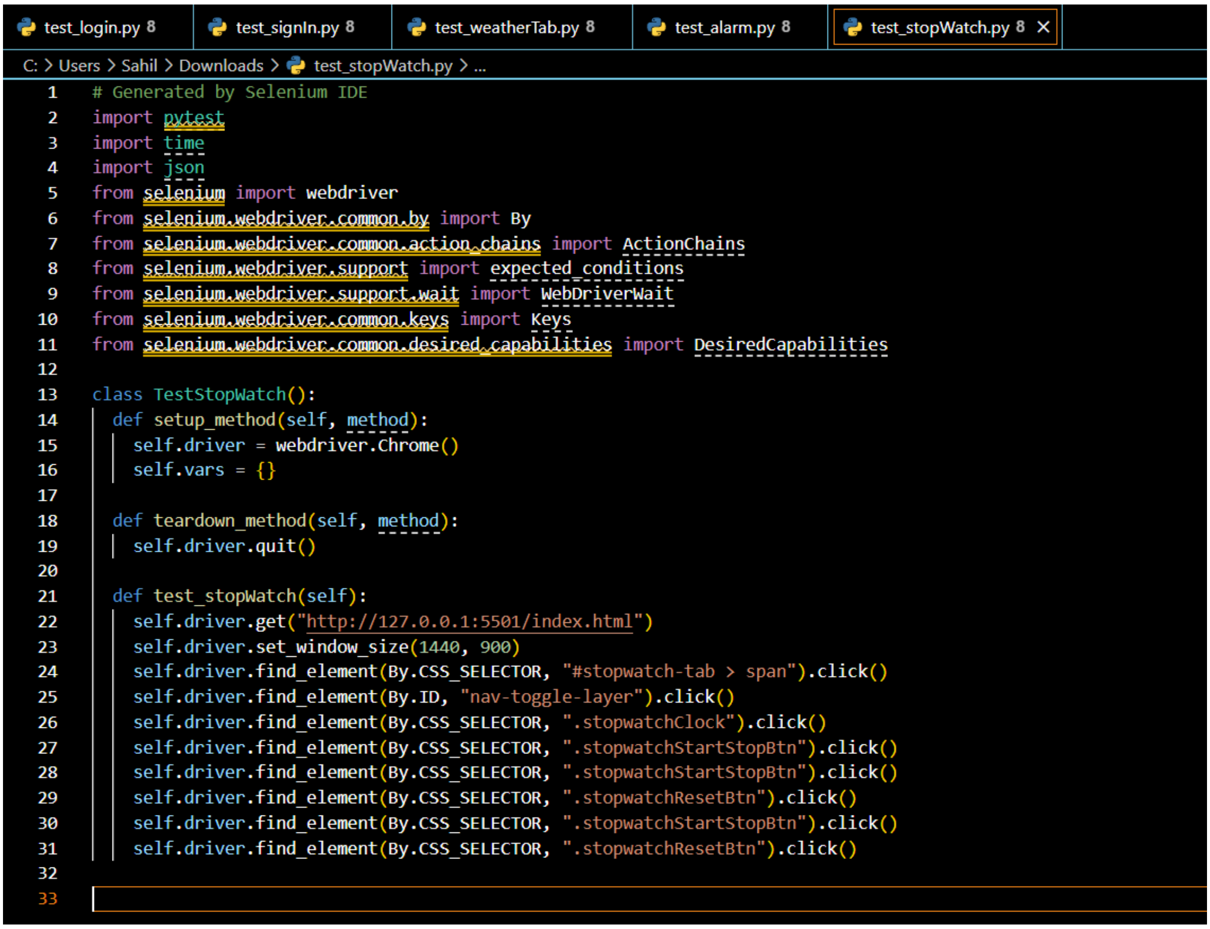
```python
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

class TestAlarm():
  def setup_method(self, method):
    self.driver = webdriver.Chrome()
    self.vars = {}

  def teardown_method(self, method):
    self.driver.quit()

  def test_alarm(self):
    self.driver.get("http://127.0.0.1:5501/index.html")
    self.driver.set_window_size(1440, 900)
    self.driver.find_element(By.CSS_SELECTOR, "#alarm-tab > span").click()
    self.driver.find_element(By.ID, "nav-toggle-layer").click()
    self.driver.find_element(By.ID, "alarmTimeInput").click()
    self.driver.find_element(By.ID, "alarmTimeInput").click()
    self.driver.find_element(By.ID, "alarmTimeInput").send_keys("10:54")
    self.driver.find_element(By.ID, "alarmNotesInput").click()
    self.driver.find_element(By.ID, "alarmNotesInput").send_keys("Eggs")
    self.driver.find_element(By.CSS_SELECTOR, ".alarmModal-saveBtn").click()
```

Output:



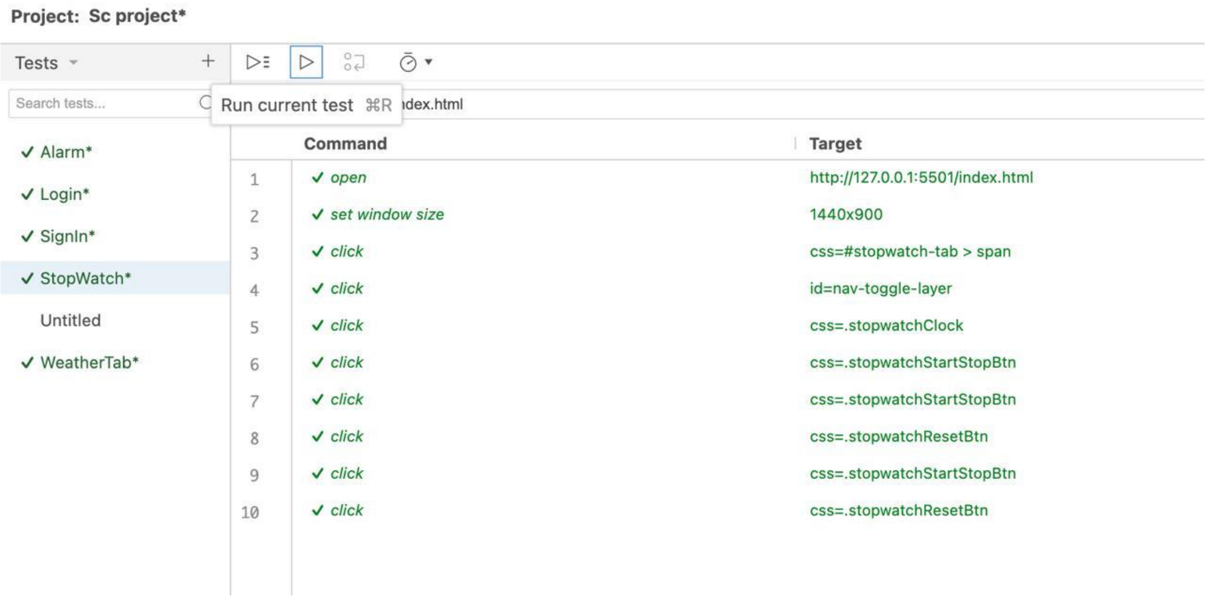
	Command	Target	Value
1	✓ open	http://127.0.0.1:5501/index.html	
2	✓ set window size	1440x900	
3	✓ click	css=#alarm-tab > span	
4	✓ click	id=nav-toggle-layer	
5	✓ click	id=alarmTimeInput	
6	✓ click	id=alarmTimeInput	
7	✓ type	id=alarmTimeInput	10:54
8	✓ click	id=alarmNotesInput	
9	✓ type	id=alarmNotesInput	Eggs
10	✓ click	css=.alarmModal-saveBtn	

Stop Watch:



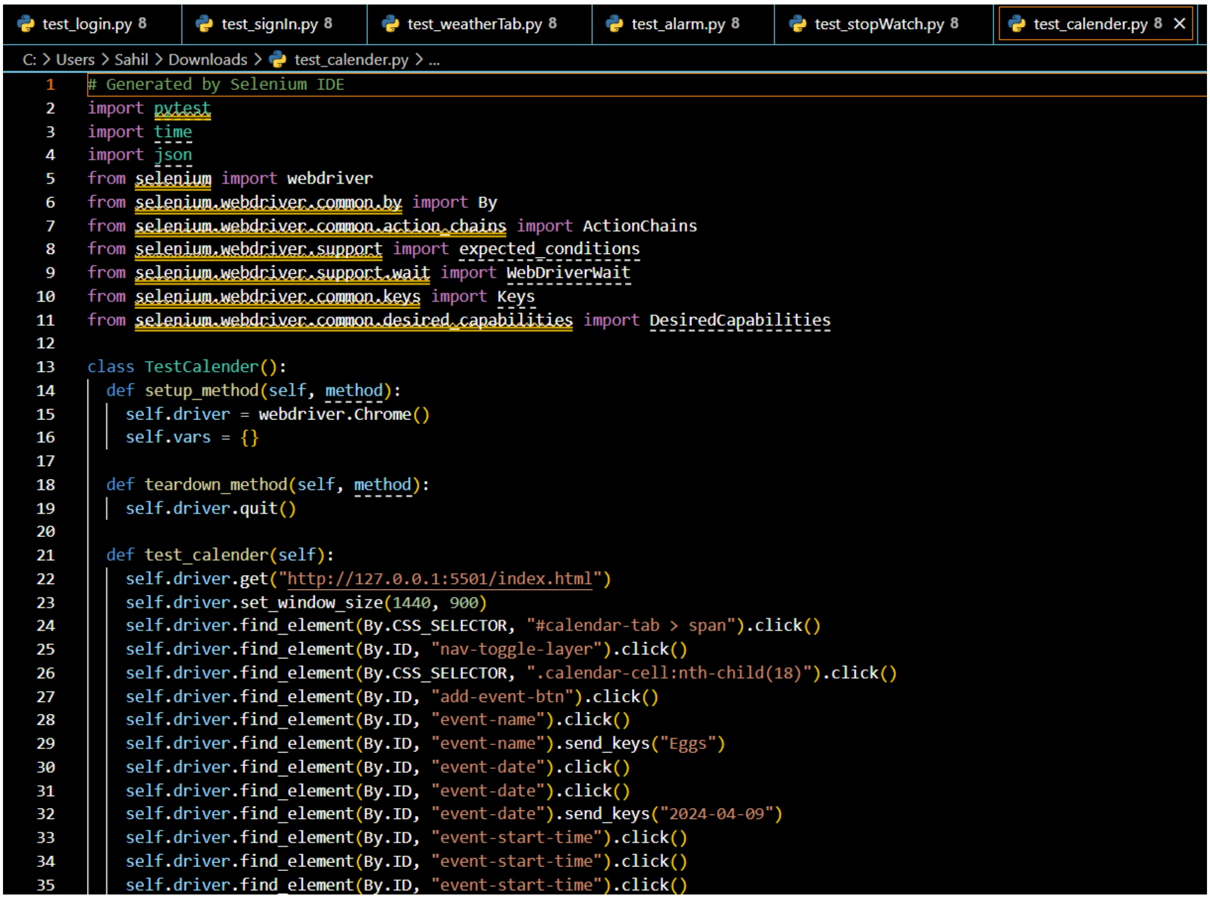
```python
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

class TestStopWatch():
    def setup_method(self, method):
        self.driver = webdriver.Chrome()
        self.vars = {}

    def teardown_method(self, method):
        self.driver.quit()

    def test_stopWatch(self):
        self.driver.get("http://127.0.0.1:5501/index.html")
        self.driver.set_window_size(1440, 900)
        self.driver.find_element(By.CSS_SELECTOR, "#stopwatch-tab > span").click()
        self.driver.find_element(By.ID, "nav-toggle-layer").click()
        self.driver.find_element(By.CSS_SELECTOR, ".stopwatchClock").click()
        self.driver.find_element(By.CSS_SELECTOR, ".stopwatchStartStopBtn").click()
        self.driver.find_element(By.CSS_SELECTOR, ".stopwatchStartStopBtn").click()
        self.driver.find_element(By.CSS_SELECTOR, ".stopwatchResetBtn").click()
        self.driver.find_element(By.CSS_SELECTOR, ".stopwatchStartStopBtn").click()
        self.driver.find_element(By.CSS_SELECTOR, ".stopwatchResetBtn").click()

```

Output:

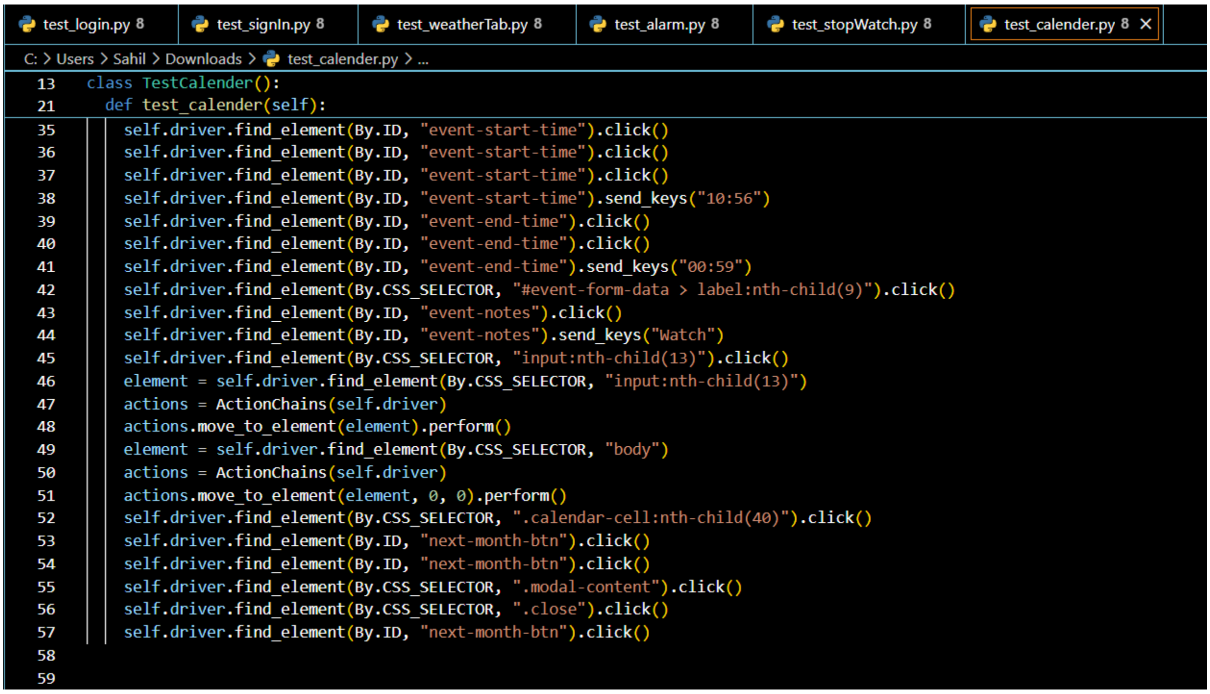


	Command	Target
1	✓ open	http://127.0.0.1:5501/index.html
2	✓ set window size	1440x900
3	✓ click	css=#stopwatch-tab > span
4	✓ click	id=nav-toggle-layer
5	✓ click	css=.stopwatchClock
6	✓ click	css=.stopwatchStartStopBtn
7	✓ click	css=.stopwatchStartStopBtn
8	✓ click	css=.stopwatchResetBtn
9	✓ click	css=.stopwatchStartStopBtn
10	✓ click	css=.stopwatchResetBtn

Calender:

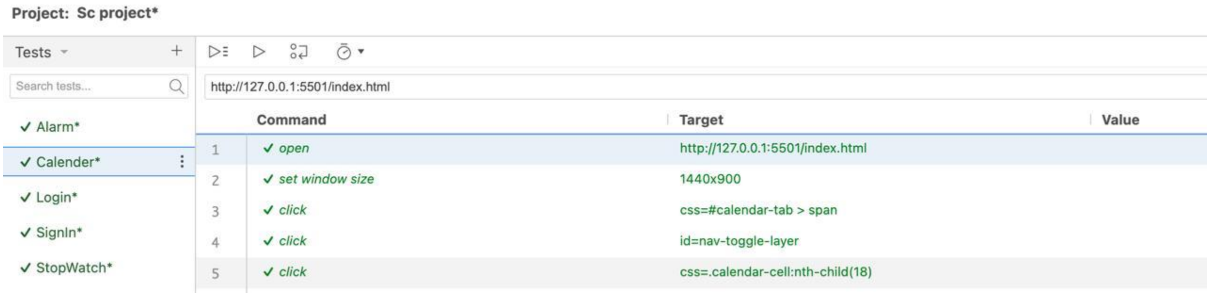


```python
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

class TestCalender():
    def setup_method(self, method):
        self.driver = webdriver.Chrome()
        self.vars = {}

    def teardown_method(self, method):
        self.driver.quit()

    def test_calender(self):
        self.driver.get("http://127.0.0.1:5501/index.html")
        self.driver.set_window_size(1440, 900)
        self.driver.find_element(By.CSS_SELECTOR, "#calendar-tab > span").click()
        self.driver.find_element(By.ID, "nav-toggle-layer").click()
        self.driver.find_element(By.CSS_SELECTOR, ".calendar-cell:nth-child(18)").click()
        self.driver.find_element(By.ID, "add-event-btn").click()
        self.driver.find_element(By.ID, "event-name").click()
        self.driver.find_element(By.ID, "event-name").send_keys("Eggs")
        self.driver.find_element(By.ID, "event-date").click()
        self.driver.find_element(By.ID, "event-date").click()
        self.driver.find_element(By.ID, "event-date").send_keys("2024-04-09")
        self.driver.find_element(By.ID, "event-start-time").click()
        self.driver.find_element(By.ID, "event-start-time").click()
        self.driver.find_element(By.ID, "event-start-time").click()
```



```python
class TestCalender():
    def test_calender(self):
        self.driver.find_element(By.ID, "event-start-time").click()
        self.driver.find_element(By.ID, "event-start-time").click()
        self.driver.find_element(By.ID, "event-start-time").click()
        self.driver.find_element(By.ID, "event-start-time").send_keys("10:56")
        self.driver.find_element(By.ID, "event-end-time").click()
        self.driver.find_element(By.ID, "event-end-time").click()
        self.driver.find_element(By.ID, "event-end-time").send_keys("00:59")
        self.driver.find_element(By.CSS_SELECTOR, "#event-form-data > label:nth-child(9)").click()
        self.driver.find_element(By.ID, "event-notes").click()
        self.driver.find_element(By.ID, "event-notes").send_keys("Watch")
        self.driver.find_element(By.CSS_SELECTOR, "input:nth-child(13)").click()
        element = self.driver.find_element(By.CSS_SELECTOR, "input:nth-child(13)")
        actions = ActionChains(self.driver)
        actions.move_to_element(element).perform()
        element = self.driver.find_element(By.CSS_SELECTOR, "body")
        actions = ActionChains(self.driver)
        actions.move_to_element(element, 0, 0).perform()
        self.driver.find_element(By.CSS_SELECTOR, ".calendar-cell:nth-child(40)").click()
        self.driver.find_element(By.ID, "next-month-btn").click()
        self.driver.find_element(By.ID, "next-month-btn").click()
        self.driver.find_element(By.CSS_SELECTOR, ".modal-content").click()
        self.driver.find_element(By.CSS_SELECTOR, ".close").click()
        self.driver.find_element(By.ID, "next-month-btn").click()
```

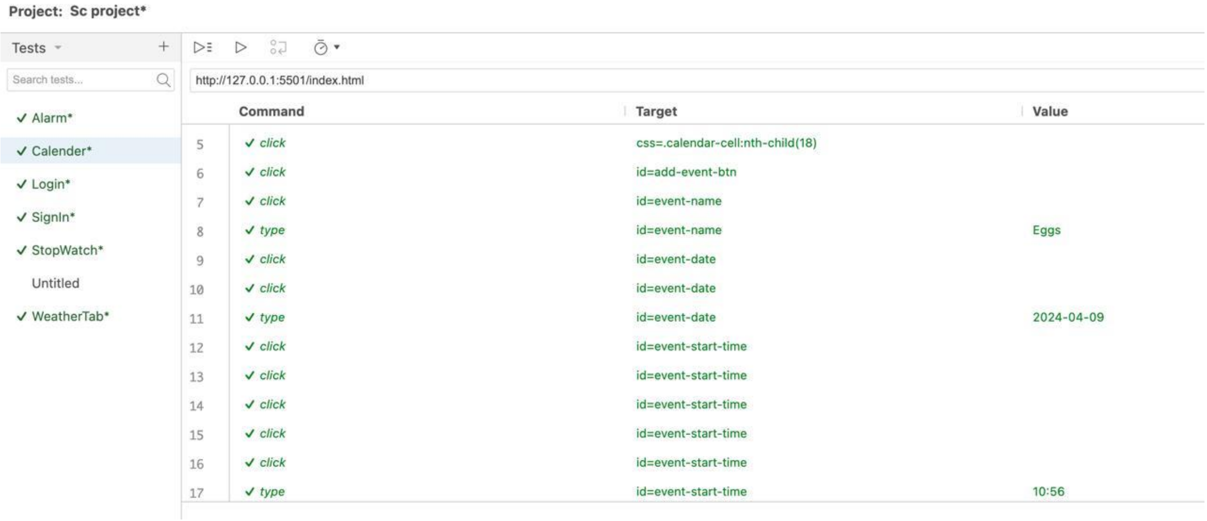
Output:



Project: Sc project*

http://127.0.0.1:5501/index.html

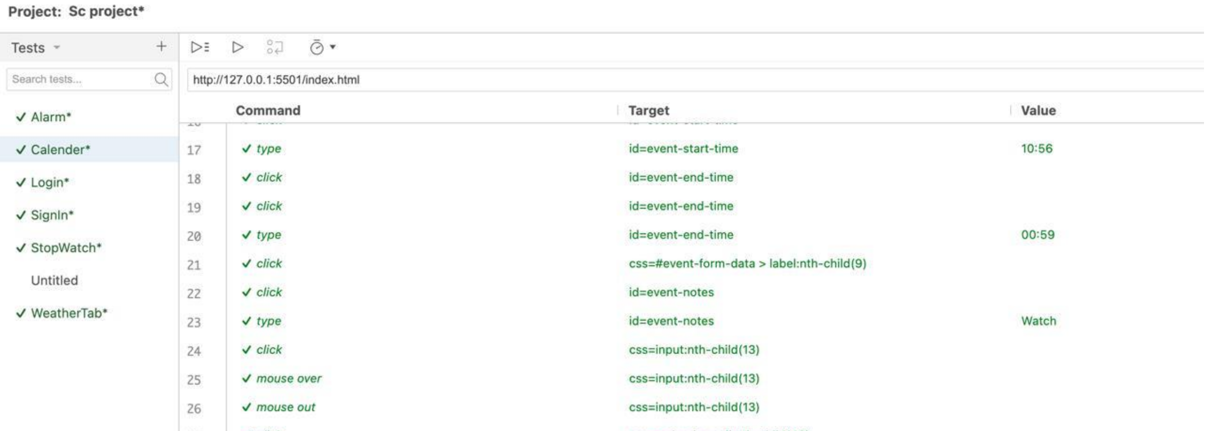
	Command	Target	Value
1	✓ open	http://127.0.0.1:5501/index.html	
2	✓ set window size	1440x900	
3	✓ click	css=#calendar-tab > span	
4	✓ click	id=nav-toggle-layer	
5	✓ click	css=.calendar-cell:nth-child(18)	



Project: Sc project*

http://127.0.0.1:5501/index.html

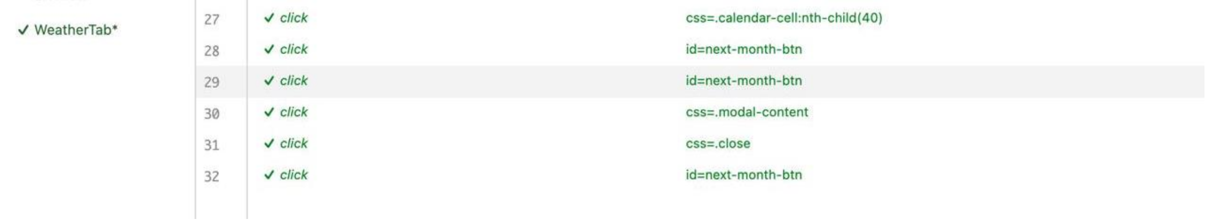
	Command	Target	Value
5	✓ click	css=.calendar-cell:nth-child(18)	
6	✓ click	id=add-event-btn	
7	✓ click	id=event-name	
8	✓ type	id=event-name	Eggs
9	✓ click	id=event-date	
10	✓ click	id=event-date	
11	✓ type	id=event-date	2024-04-09
12	✓ click	id=event-start-time	
13	✓ click	id=event-start-time	
14	✓ click	id=event-start-time	
15	✓ click	id=event-start-time	
16	✓ click	id=event-start-time	
17	✓ type	id=event-start-time	10:56



Project: Sc project*

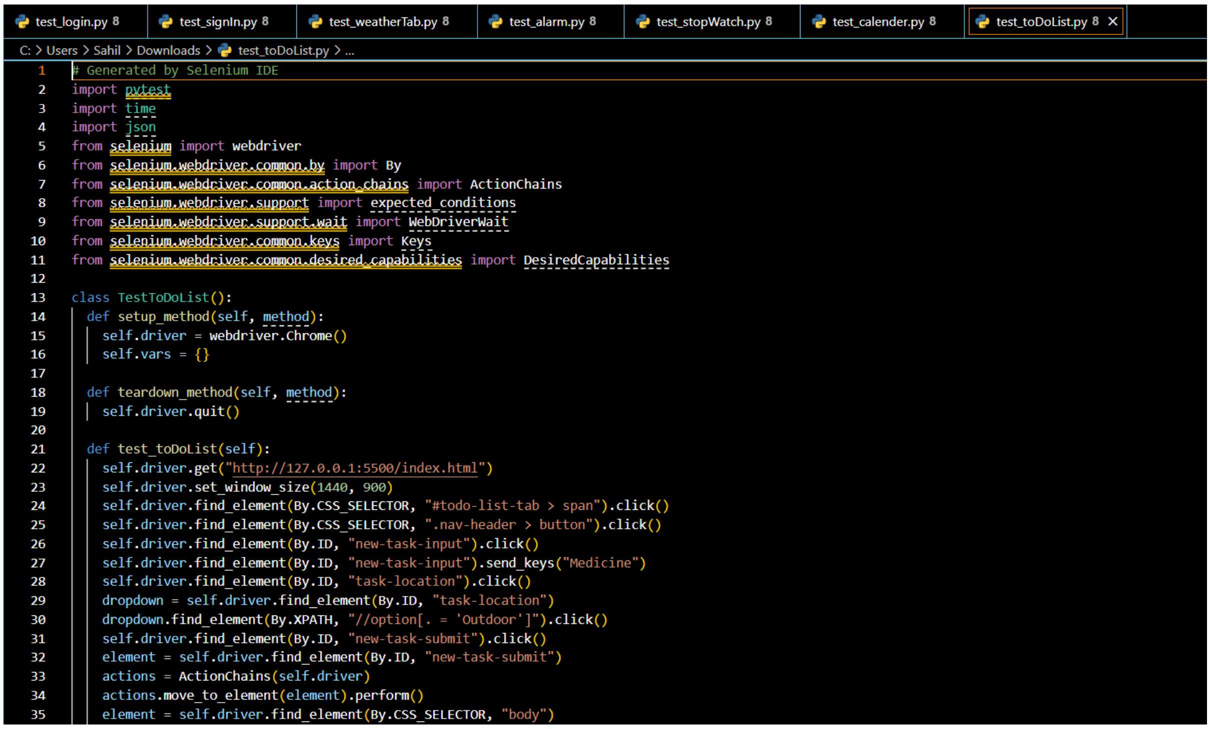
http://127.0.0.1:5501/index.html

	Command	Target	Value
17	✓ type	id=event-start-time	10:56
18	✓ click	id=event-end-time	
19	✓ click	id=event-end-time	
20	✓ type	id=event-end-time	00:59
21	✓ click	css=#event-form-data > label:nth-child(9)	
22	✓ click	id=event-notes	
23	✓ type	id=event-notes	Watch
24	✓ click	css=input:nth-child(13)	
25	✓ mouse over	css=input:nth-child(13)	
26	✓ mouse out	css=input:nth-child(13)	

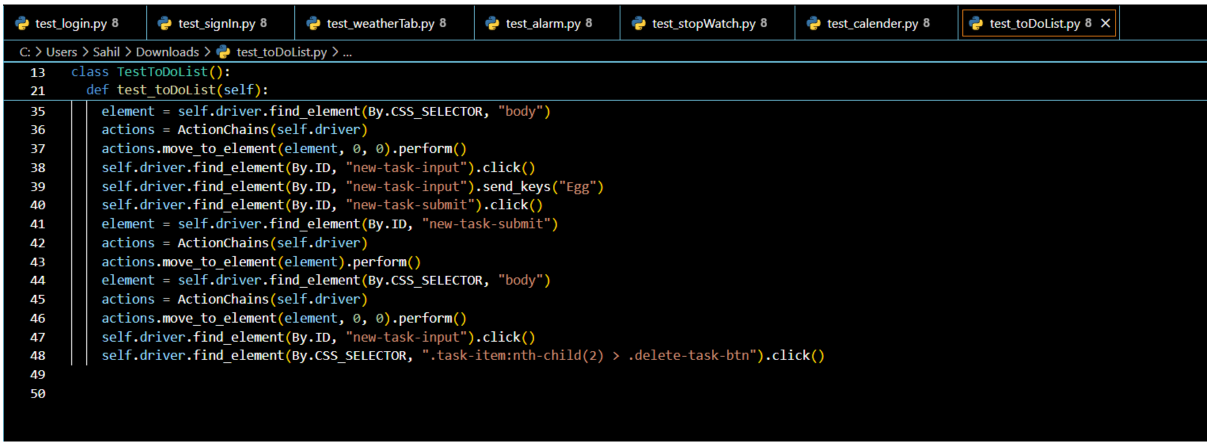


	Command	Target	Value
27	✓ click	css=.calendar-cell:nth-child(40)	
28	✓ click	id=next-month-btn	
29	✓ click	id=next-month-btn	
30	✓ click	css=.modal-content	
31	✓ click	css=.close	
32	✓ click	id=next-month-btn	

TO-DO List:

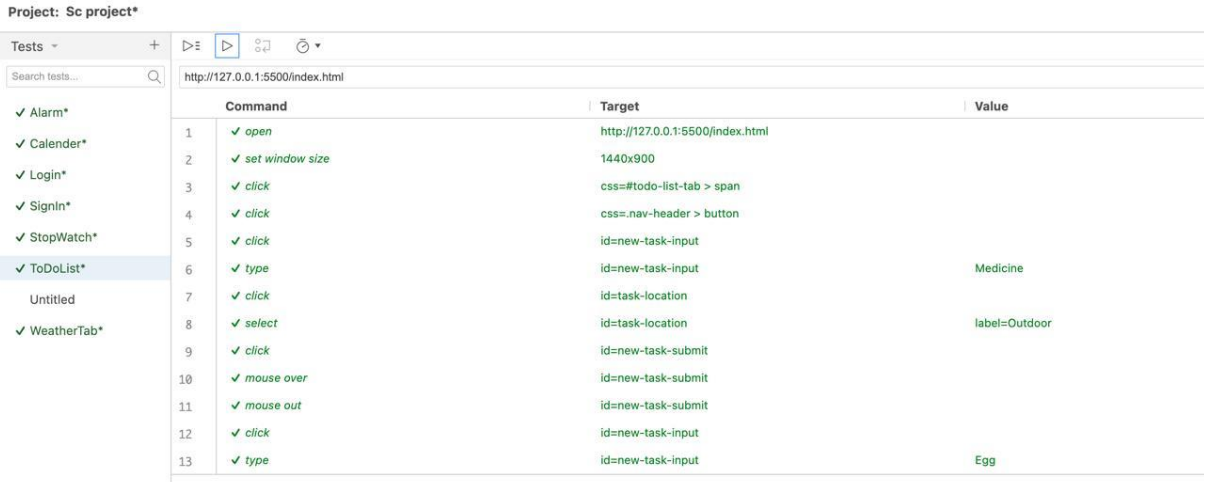


```python
# Generated by Selenium IDE
import pytest
import time
import json
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.action_chains import ActionChains
from selenium.webdriver.support import expected_conditions
from selenium.webdriver.support.wait import WebDriverWait
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.desired_capabilities import DesiredCapabilities

class TestToDoList():
    def setup_method(self, method):
        self.driver = webdriver.Chrome()
        self.vars = {}

    def teardown_method(self, method):
        self.driver.quit()

    def test_toDoList(self):
        self.driver.get("http://127.0.0.1:5500/index.html")
        self.driver.set_window_size(1440, 900)
        self.driver.find_element(By.CSS_SELECTOR, "#todo-list-tab > span").click()
        self.driver.find_element(By.CSS_SELECTOR, ".nav-header > button").click()
        self.driver.find_element(By.ID, "new-task-input").click()
        self.driver.find_element(By.ID, "new-task-input").send_keys("Medicine")
        self.driver.find_element(By.ID, "task-location").click()
        dropdown = self.driver.find_element(By.ID, "task-location")
        dropdown.find_element(By.XPATH, "//option[. = 'Outdoor']").click()
        self.driver.find_element(By.ID, "new-task-submit").click()
        element = self.driver.find_element(By.ID, "new-task-submit")
        actions = ActionChains(self.driver)
        actions.move_to_element(element).perform()
        element = self.driver.find_element(By.CSS_SELECTOR, "body")
```



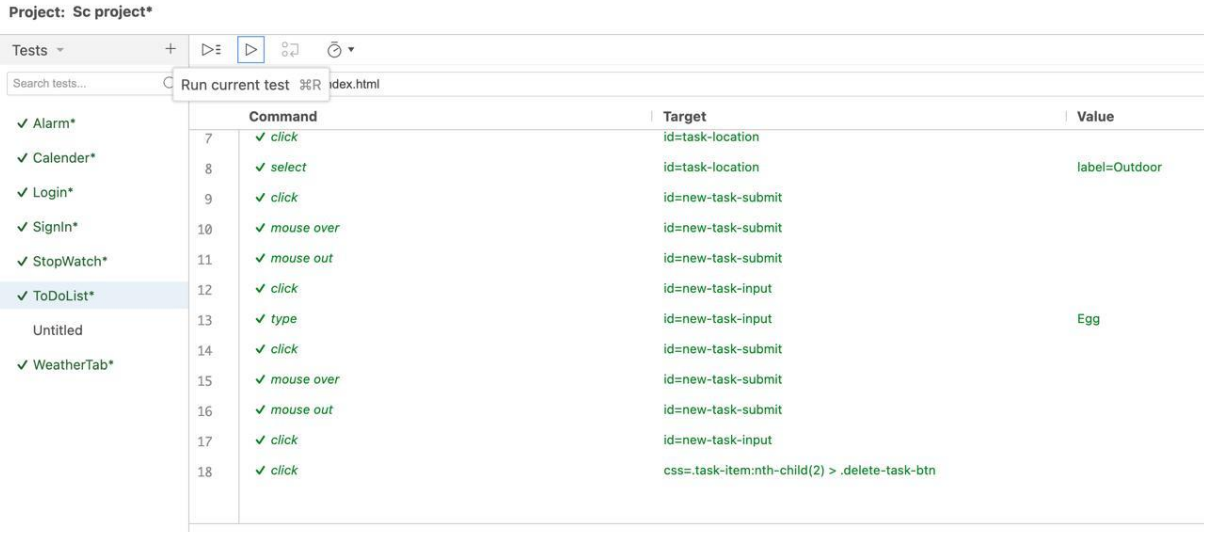
```python
class TestToDoList():
    def test_toDoList(self):
        element = self.driver.find_element(By.CSS_SELECTOR, "body")
        actions = ActionChains(self.driver)
        actions.move_to_element(element, 0, 0).perform()
        self.driver.find_element(By.ID, "new-task-input").click()
        self.driver.find_element(By.ID, "new-task-input").send_keys("Egg")
        self.driver.find_element(By.ID, "new-task-submit").click()
        element = self.driver.find_element(By.ID, "new-task-submit")
        actions = ActionChains(self.driver)
        actions.move_to_element(element).perform()
        element = self.driver.find_element(By.CSS_SELECTOR, "body")
        actions = ActionChains(self.driver)
        actions.move_to_element(element, 0, 0).perform()
        self.driver.find_element(By.ID, "new-task-input").click()
        self.driver.find_element(By.CSS_SELECTOR, ".task-item:nth-child(2) > .delete-task-btn").click()
```

Output:



http://127.0.0.1:5500/index.html

	Command	Target	Value
1	✓ open	http://127.0.0.1:5500/index.html	
2	✓ set window size	1440x900	
3	✓ click	css=#todo-list-tab > span	
4	✓ click	css=.nav-header > button	
5	✓ click	id=new-task-input	
6	✓ type	id=new-task-input	Medicine
7	✓ click	id=task-location	
8	✓ select	id=task-location	label=Outdoor
9	✓ click	id=new-task-submit	
10	✓ mouse over	id=new-task-submit	
11	✓ mouse out	id=new-task-submit	
12	✓ click	id=new-task-input	
13	✓ type	id=new-task-input	Egg



ıdex.html

	Command	Target	Value
7	✓ click	id=task-location	
8	✓ select	id=task-location	label=Outdoor
9	✓ click	id=new-task-submit	
10	✓ mouse over	id=new-task-submit	
11	✓ mouse out	id=new-task-submit	
12	✓ click	id=new-task-input	
13	✓ type	id=new-task-input	Egg
14	✓ click	id=new-task-submit	
15	✓ mouse over	id=new-task-submit	
16	✓ mouse out	id=new-task-submit	
17	✓ click	id=new-task-input	
18	✓ click	css=.task-item:nth-child(2) > .delete-task-btn	