

| Business Template  **Test Automation & Selenium Basics** |
| --- |
| Task 1 **Provide the Python code based on the** [**api\_task.py**](https://github.com/AntonLazarchik/DQE_LAB_2024_CODE_SAMPLES/blob/main/api_task.py) **which will test 2 scenarios:**  **1. Let’s pretend that you’ve loaded the test data for the user with Id=3 and generated 10 valid posts. Verify that 10 posts were created for user 3.**  **Use:** [**https://jsonplaceholder.typicode.com/**](https://jsonplaceholder.typicode.com/) **(Resources section)**  **2. You have some data project on which the client used previously GCP cloud for storing the data for forecasts. Now the client wants to build some analytics around this data in AWS. So, we need to verify that the data for some specific date is presented on the source (GCP) and the target staging bucket in AWS. You need to verify in the test that both buckets for specific dates are not empty as a basic smoke test.1. Using the automatic approach (Selenium Manager or Driver Management Software) for Chrome.**  **Expected result: Python script, Conftest,py file with fixtures, a screenshot of the allure report with 2 tests.**       Task 2 **Provide the Python code which will open google.com in Chrome and Firefox browsers and print the title of the page.**  **1. Using the automatic approach (Selenium Manager or Driver Management Software) for Chrome.**  **2. Using the manual approach (PATH variable or Hard Coded Location) for Firefox.**  **Expected result: Python script**    **Link to documentation:**  [**https://www.selenium.dev/documentation/webdriver/getting\_started/install\_drivers/**](https://www.selenium.dev/documentation/webdriver/getting_started/install_drivers/) Task 3 **Using the attached list of screenshots and links to websites provide a list of the locators for elements in the red area. For each type of locator below prepare 2 examples. 2 locators by id, 2 locators by class name, etc. Be sure that your locator is leading to an exact single element, not a group of them.**  **You can provide different locators for the same element, like 1 locator by ID and 1 by name for a button element. But the maximum number of locators for the same element mustn’t be more than 2.**  **1. class name**  **Example: find\_element(By.CLASS\_NAME, "information")**  **2. id**  **Example: find\_element(By.ID, "lname")**  **3. name**  **Example: find\_element(By.NAME, "newsletter")**  **4. CSS selector**  **Example: find\_element(By.CSS\_SELECTOR, "#fname")**  **5. XPath**  **Example: find\_element(By.XPATH, "//input[@value='f']")**  **6\*. Relative locators**  **Example: locate\_with(By.TAG\_NAME, "input").above({By.ID: "password"})**    **Expected result: Python script. Screenshot with the execution results.**  **The result must be provided as the Python code and be fully executable. Example:**    **Links to documentation:**  [**https://www.selenium.dev/documentation/webdriver/elements/locators/**](https://www.selenium.dev/documentation/webdriver/elements/locators/)       Task 4. **Prepare a Python script. Use implicit wait and explicit wait inside the script:**  **1. Open the google.com.**  **2. Specify “Selenium” as the search field.**  **3. Open the first link in the results.**  **Expected result: Python script. Screenshot with the execution results.** |