

Software Testing (CS-731)

Client-side web applications testing (bypass testing)

SANANDAN SHARMA MT2021114

SRIJAN PRAKASH MT2021139

# Objective:

Projects that involve testing of client-side code of a web application by designing test cases that bypass client-side validation and sending changed/corrupt input to the server.

# Code Description:

The code for testing that we have used consists of an Advertisement posting website. The website uses React Js as its frontend and Nodejs as its backend and uses MongoDB database. The website basically makes the complicated process of posting ads by a commoner, an easy and seamless task. Our code consists of JavaScript component files representing the various frontend components of the website and we have mainly done testing on the Login and Signup components of the website. Hence, we bypass the client end of our code to see the response of our backend for abnormal inputs i.e. We do bypass testing.

# Testing Strategy Used:

The World Wide Web gives software developers a new way to deploy sophisticated, interactive programs with complex GUIs and large numbers of back-end software components that are integrated into a novel and interesting ways. Web applications are constructed from heterogeneous software components that interact with each other and with users in novel ways. Web software components are distributed across multiple computers and organizations, are often created and integrated dynamically, are written in diverse languages and run-on diverse hardware platforms, and must satisfy very high requirements for reliability, availability, and usability. These characteristics offer powerful new abilities and also present new problems to software developers.

Hence, it’s important to do proper testing for every website on the internet. One such testing is bypass testing which basically checks the response of the server when client-side code is forcefully bypassed. Bypass testing is a unique and novel way to create test cases that is available only because of the unusual mix of client -server, HTML GUI, and JavaScript technologies that are used in Web applications.

To apply bypass testing, the HTML form elements are analyzed and each input element is modeled as a parameter. Values are chosen to violate each of the restrictions that the HTML and JavaScript impose. At present, the values are chosen according to simple rules, but we expect to describe the restrictions via regular expressions, then use mutation-style modifications to the expressions to generate invalid inputs.

# Tools used for Testing:

We have used the **VS code** to run and test our code and **Selenium** as the testing framework for designing and creating basic test cases. We used npm as package manager and pip for python dependencies. Figure 1. shows the Selenium dependency.

# Bypass Testing Steps:

1. Download the selenium using the command.
   * Pip install selenium
2. Our project consists of signup and login page, where we used bypass testing strategy to bypass the html script. We used Selenium testing tool to automate the testing process. Figure 2 shows the signup page script for which we generated the test cases.

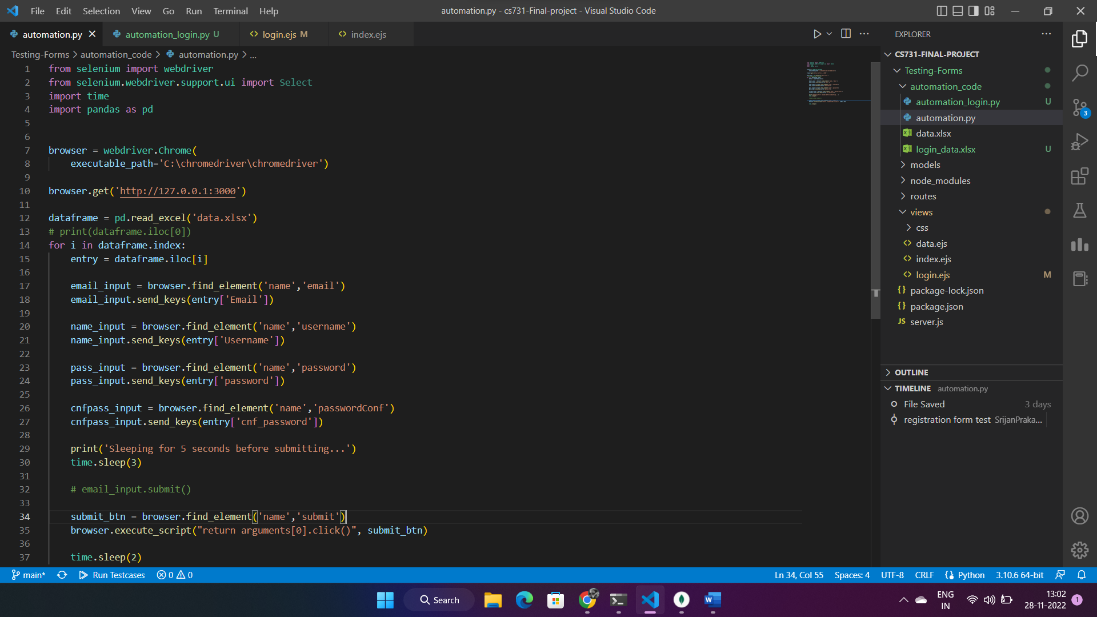


Figure 2: signup script

1. In the bypass testing strategy, we bypass the client side scripts, and send the inputs to the server and records how server responds. For the signup script we have written data into an excel file from which a input is read separately everytime.

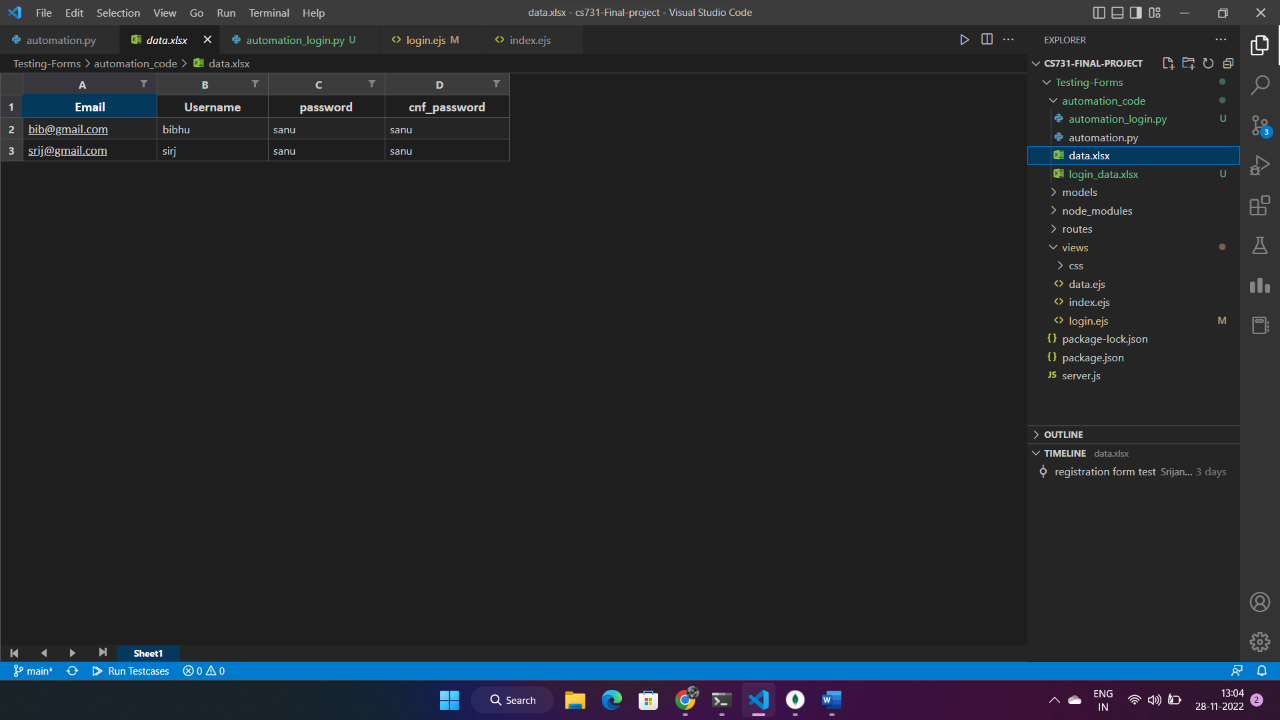


Figure 3: signup form entries valid testcases

1. Selenium automates the testing process. We wrote the two types of test cases, which is invalid input and valid input. And we recorded the responses of the server in both the cases and accordingly the testcase result is displayed on the terminal.
2. Two automation files are given one is for the login automation and validation and the other one checks for registration automation and validation.
3. The login automation code is given below, it opens up the browser fills the data from the excel sheets and displays the result on the terminal according to the pass or fail cases.

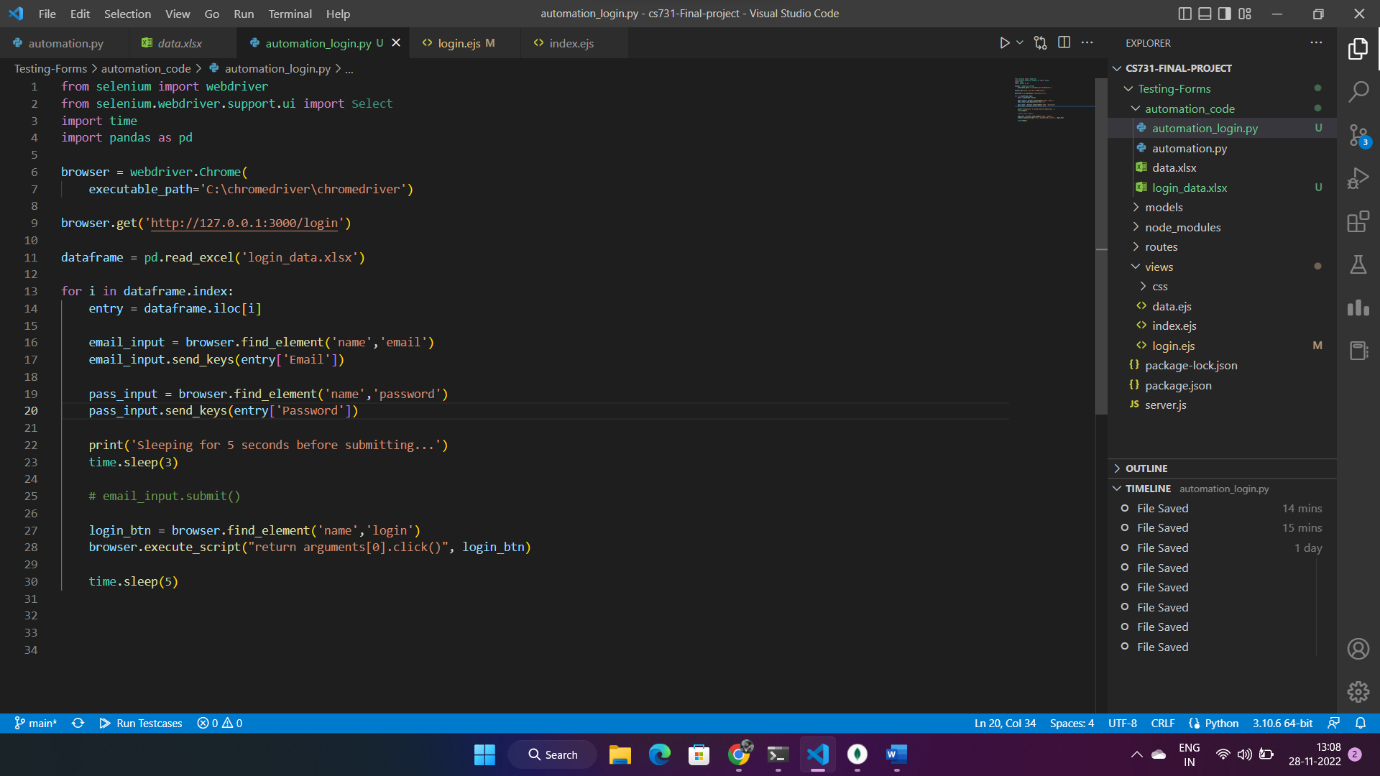


Figure 4 : login automation script

# References:

1. https://selenium-python.readthedocs.io/
2. Jeff Offutt, Ye Wu, Xiaochen Du and Hong Huang. 2014. Bypass testing of web applications. Information and Software Engineering.