Name: Nandhakumaran H

Employee Id: 12225

W3h Analysis:

Case Study: Online Charity Management System	
1. What?	2. How?
What are the modules required for online charity management System? **Ans:** 1. Admin Module 2. Employee Module 3. User Module **What are the functionalities needed for Admin module?** **Ans:** 1. Register and Login to the portal 2. Do CRUD operations on All Data **What are the functionalities needed for Employee Module?** **Ans:** 1. Register and Login to the portal 2. Do CRUD operation on Employee Data 3. Handle requests of users and response accordingly **What are the functionalities needed for User Module?** **Ans:** 1. Register and Login to the portal 2. Do CRUD operation on User Data 3. Raise Request	**Admin:** **Register and Login:** **Method 1:** Register and Login by Admin Id and password **Method 2:** Register and Login by email id and password **Method 3:** Register and Login by Admin Name and Password **CRUD:** **Method 1:** Do Crud operations by unique values **Method 2:** Do Crud Operations by Duplicable values **Method 3:** Do Crud Operations by Manually search the Users and Employees **Method 4:** Do Crud Operations by Selecting the Employee and User in Dropdown Menu **Employee:** **Register and Login:** **Method 1:** Register and Login by Employee Name and password **Method 2:** Register and Login by email id and password

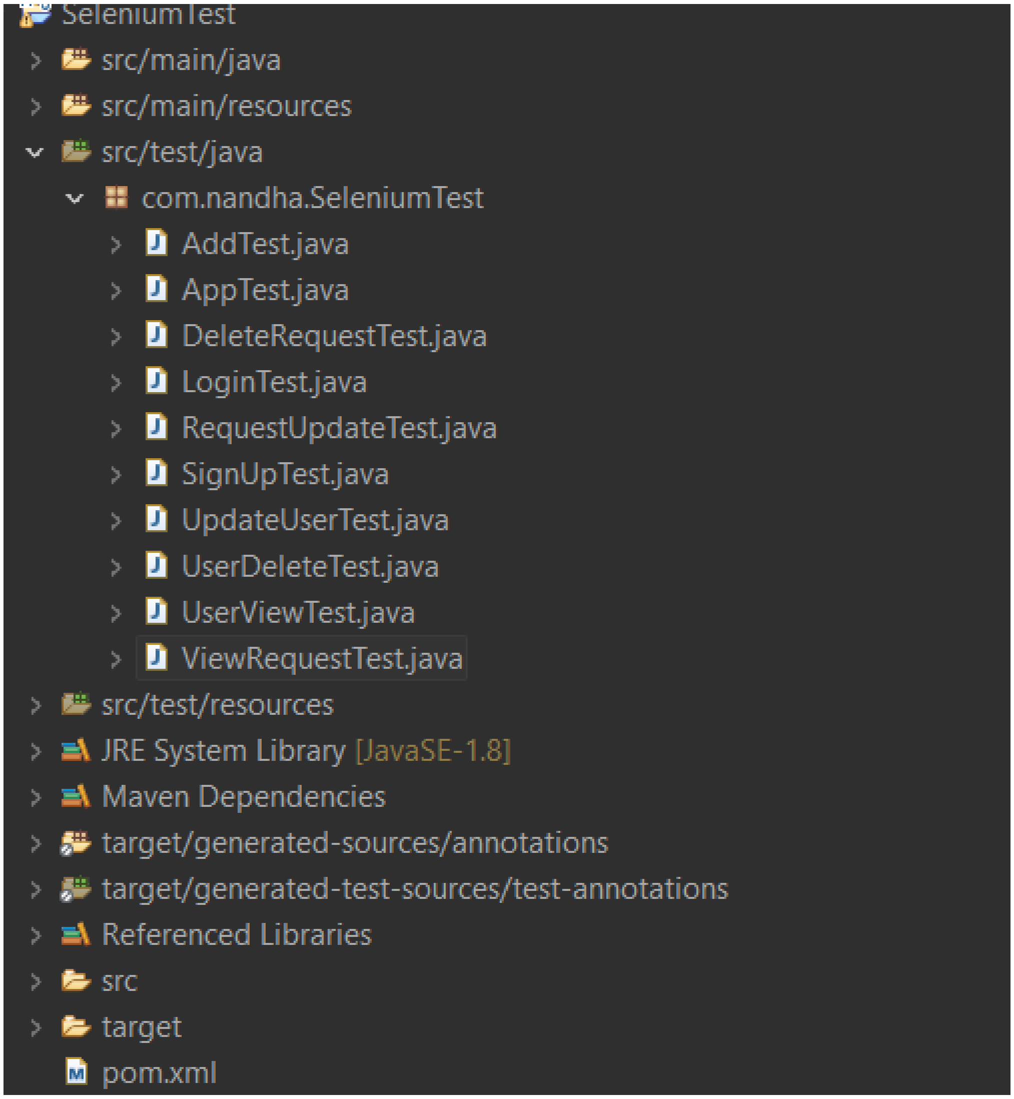
	Method 3: Register and Login by Employee id and password. **Method 4:** Register and Login by Social Media Accounts **CRUD:** **Method 1:** Do Crud operations by unique values like primary keys **Method 2:** Do Crud Operations by Duplicable values **Method 3:** Do Crud Operations by Manually search the Requests **Method 4:** Do Crud Operations by Selecting the Request in Dropdown Menu **Handle requests and Response:** **Method 1:** Handle the requests and responses of users by their User Id and donation type **Method 2:** Handle the requests and responses of users by their addresses **Method 3:** Handle the requests and responses by users by their frequent donation history and any other third-party data. **User:** **Register and Login:** **Method 1:** Register and Login by Username and password **Method 2:** Register and Login by email id and password **Method 3:** Register and Login by User Id and password **CRUD:**

	Method 1: Do Crud operations by unique values like User Ids. **Method 2:** Do Crud Operations by Duplicable values like Addresses. **Raise Request:** **Method 1:** Raise Requests by typing what type of donation they want to donate **Method 2:** Raise requests by selecting the existing donation formats **Method 3:** Raise Requests by copying other user's donation type or sample donation type formats
Admin: **Register and Login:** **Method 1:** Register and Login by Admin Id and password **(**Unique values like Admin Id is most preferrable for Register and Login for Admin part of side) **CRUD:** **Method 1:** Do Crud operations by unique values (Unique values are the best way handle data anywhere) **Employee:** **Register and Login:** **Method 1:** Register and Login by Employee Name and password **(**Usernames are most preferable for Register and Login for Employee side)	**Admin, Employee and User:** • Register and Login by email id and social media account is not a best approachable way for the trustable website like charity websites. • CRUD Operation by duplicable values will lead to Loss of data • CRUD Operations by manually searching the data without using primary key is not an effective way to do crud operations. • CRUD Operations by Dropdown menu by selecting the data is not an efficient way to find or select the data. • Handling requests and responses by duplicable values is the inappropriate way to handle the data. • Handling requests and responses by previous donation history to any

CRUD: **Method 1:** Do Crud operations by unique values like Employee Ids. (Unique values are the best way handle data anywhere) **Handle requests and Response:** **Method 1:** Handle the requests and responses of users by their User Id and donation types (Unique values like User Ids will avoid the confusion among user during updating the donation status) **User:** **Register and Login:** **Method 1:** Register and Login by Username and password (Unique values like usernames are most preferable for Register and Login for Users Side) **CRUD:** **Method 1:** Do Crud operations by unique values like primary keys (Unique values are the best way handle data anywhere) **Raise Request:** **Method 1:** Raise Requests by typing what type of donation they want to donate **(**Type the donation type will be the most efficient way to raise a request**)**	other third-party data is the inappropriate way to handle the data. • Raise requests by selecting the existing donation formats or copying other user’s donation formats will make the user lose their genuine interest to select the things which they want to mention or donate

3. Why?	4. Why Not?

WebDriver:



```
SeleniumTest
  > src/main/java
  > src/main/resources
  v src/test/java
    v com.nandha.SeleniumTest
      > AddTest.java
      > AppTest.java
      > DeleteRequestTest.java
      > LoginTest.java
      > RequestUpdateTest.java
      > SignUpTest.java
      > UpdateUserTest.java
      > UserDeleteTest.java
      > UserViewTest.java
      > ViewRequestTest.java
  > src/test/resources
  > JRE System Library [JavaSE-1.8]
  > Maven Dependencies
  > target/generated-sources/annotations
  > target/generated-test-sources/test-annotations
  > Referenced Libraries
  > src
  > target
    pom.xml
```

package com.nandha.SeleniumTest;

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

import static org.junit.Assert.assertEquals;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class AddTest {

private WebDriver driver;

@BeforeEach
void start() throws InterruptedException {
driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("http://localhost:3000/login");

WebElement usernameInput = driver.findElement(By.name("name"));
usernameInput.sendKeys("Nandhakumaran");
WebElement passwordInput = driver.findElement(By.name("password"));
passwordInput.sendKeys("123");

WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login')]"));
loginButton.submit();

Thread.sleep(5000);

String currentUrl = driver.getCurrentUrl();
if (currentUrl.equals("http://localhost:3000/main")) {
System.out.println("Positive login test passed for employee!");
} else if (currentUrl.equals("http://localhost:3000/usermain")) {
System.out.println("Positive login test passed for donator!");
} else {
System.out.println("Positive login test failed. Unexpected redirection.");
}

WebElement addButton =
driver.findElement(By.xpath("//button[contains(text(),'Donate')]"));
addButton.click();
}


@AfterEach
```

```
void close() {
driver.quit();
}

@Test
public void testAddUserWithValidData() throws InterruptedException {

// Locate the input fields
WebElement nameField = driver.findElement(By.name("name"));
nameField.sendKeys("Vashanth");

WebElement emailField = driver.findElement(By.name("emailid"));
emailField.sendKeys("vasi@example.com");

WebElement addressField = driver.findElement(By.name("address"));
addressField.sendKeys("Mandelanagar");

WebElement phoneField = driver.findElement(By.name("phoneNumber"));
phoneField.sendKeys("1234567890");

WebElement donationField = driver.findElement(By.name("donation"));
donationField.sendKeys("100");

WebElement submitButton =
driver.findElement(By.xpath("//button[contains(text(),'Submit')]"));
submitButton.submit();
```

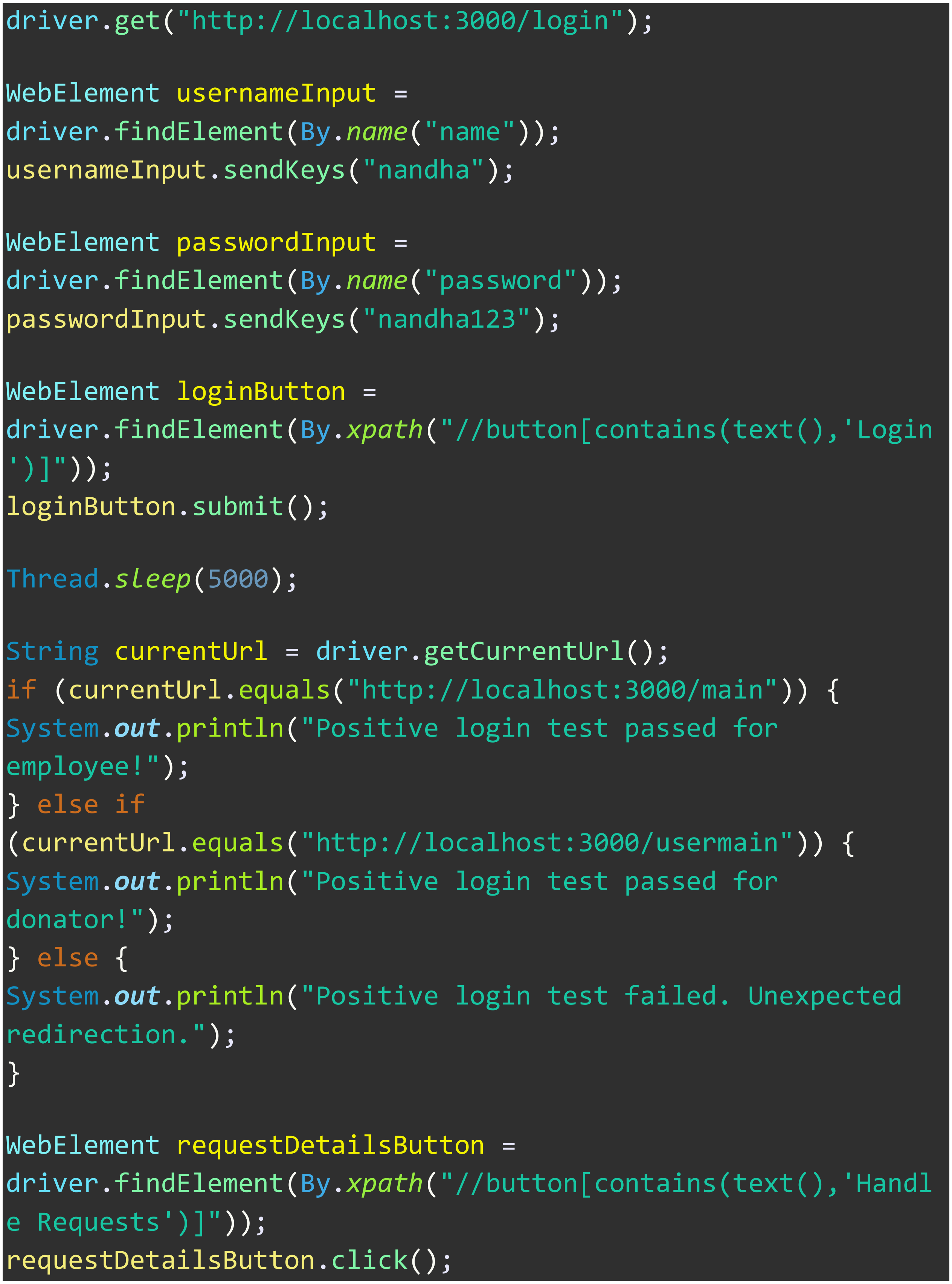
```
Thread.sleep(5000);

assertEquals("Data Added Successfully", driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

Thread.sleep(5000);

}

@Test
public void testAddUserWithEmptyFields() throws InterruptedException {

WebElement submitButton =
driver.findElement(By.xpath("//button[contains(text(),'Submit')]"));

submitButton.submit();

Thread.sleep(5000);

}

@Test
public void testAddUserWithInvalidEmail() throws InterruptedException {

WebElement nameField = driver.findElement(By.name("name"));
```

```
nameField.sendKeys("Vashanth");

WebElement emailField = driver.findElement(By.name("emailid"));
emailField.sendKeys("InvalidEmailId");

WebElement addressField = driver.findElement(By.name("address"));
addressField.sendKeys("Mandelanagar");

WebElement phoneField = driver.findElement(By.name("phoneNumber"));
phoneField.sendKeys("1234567890");

WebElement donationField = driver.findElement(By.name("donation"));
donationField.sendKeys("100");

WebElement submitButton =
driver.findElement(By.xpath("//button[contains(text(),'Submit')]"));
submitButton.submit();

Thread.sleep(5000);

}

@Test
public void testAddUserWithInvalidPhoneNumber() throws InterruptedException {

WebElement nameField = driver.findElement(By.name("name"));
```

```
nameField.sendKeys("Vashanth");

WebElement emailField = driver.findElement(By.name("emailid"));
emailField.sendKeys("vasi@example.com");

WebElement addressField = driver.findElement(By.name("address"));
addressField.sendKeys("Mandelanagar");

WebElement phoneField = driver.findElement(By.name("phoneNumber"));
phoneField.sendKeys("Invalid-Phone-Number");

WebElement donationField = driver.findElement(By.name("donation"));
donationField.sendKeys("100");

WebElement submitButton =
driver.findElement(By.xpath("//button[contains(text(),'Submit')]"));
submitButton.submit();

Thread.sleep(5000);

assertEquals("Failed to add data. Please try again.",
driver.switchTo().alert().getText());

driver.switchTo().alert().accept();

Thread.sleep(5000);
```

```
}

}
```

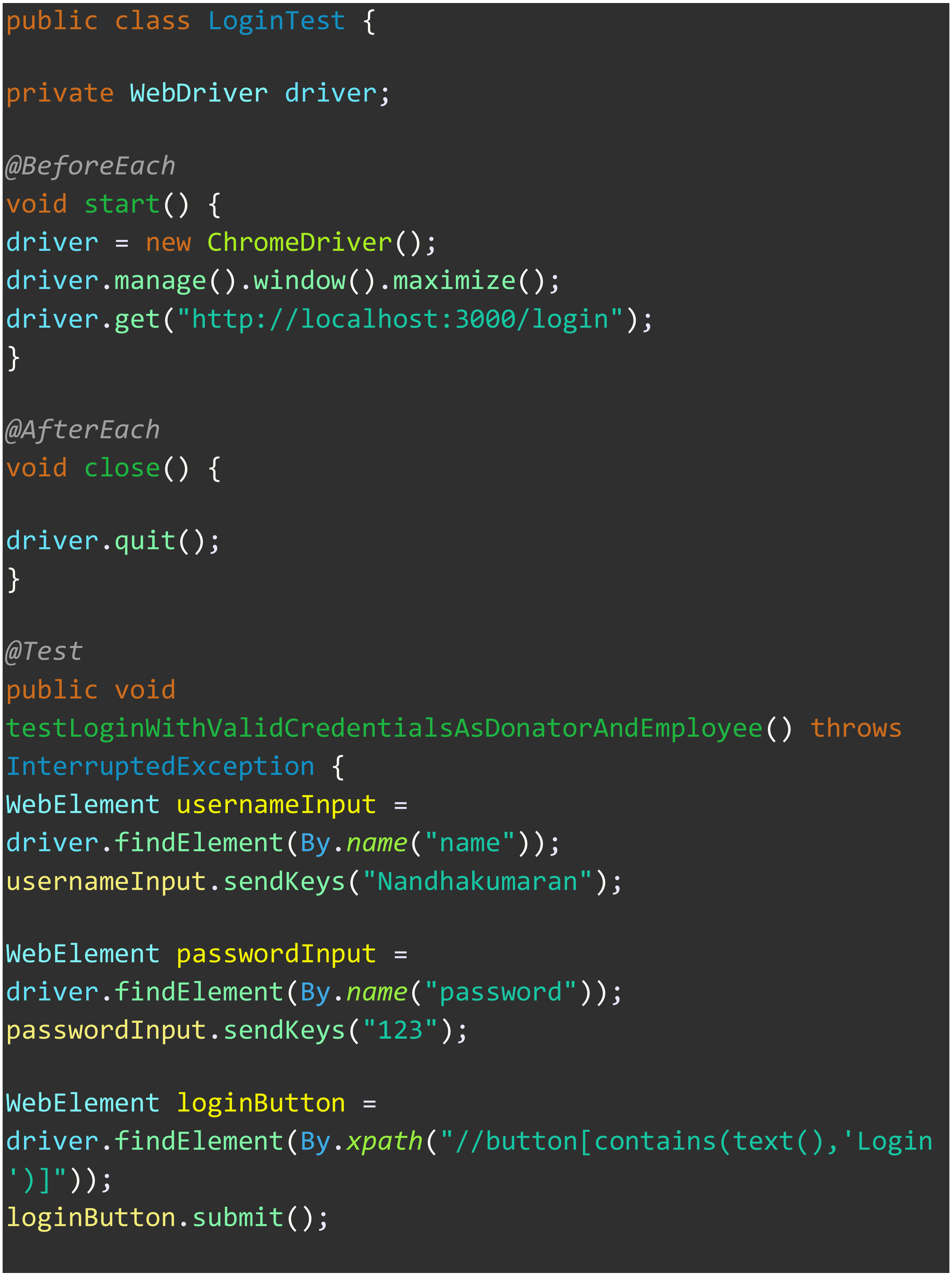
```
package com.nandha.SeleniumTest;

import static org.junit.Assert.assertEquals;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class DeleteRequestTest {

private WebDriver driver;

@AfterEach
void close() {
driver.quit();
}

@Test
void testDeleteRequest() throws InterruptedException {

driver = new ChromeDriver();

driver.manage().window().maximize();
```



```java
driver.get("http://localhost:3000/login");

WebElement usernameInput = driver.findElement(By.name("name"));
usernameInput.sendKeys("nandha");

WebElement passwordInput = driver.findElement(By.name("password"));
passwordInput.sendKeys("nandha123");

WebElement loginButton = driver.findElement(By.xpath("//button[contains(text(),'Login')]"));
loginButton.submit();

Thread.sleep(5000);

String currentUrl = driver.getCurrentUrl();
if (currentUrl.equals("http://localhost:3000/main")) {
System.out.println("Positive login test passed for employee!");
} else if (currentUrl.equals("http://localhost:3000/usermain")) {
System.out.println("Positive login test passed for donator!");
} else {
System.out.println("Positive login test failed. Unexpected redirection.");
}

WebElement requestDetailsButton = driver.findElement(By.xpath("//button[contains(text(),'Handle Requests')]"));
requestDetailsButton.click();
```

```
WebElement requestDeleteButton =
driver.findElement(By.xpath("//*[@id=\"root\"]/div/table/tbody/tr[3]/td[9]/button"));
requestDeleteButton.click();

Thread.sleep(5000);
assertEquals("Do you want to delete?",
driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

Thread.sleep(5000);
assertEquals("Record has been deleted",
driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

Thread.sleep(5000);

}

}
```

```
package com.nandha.SeleniumTest;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
```

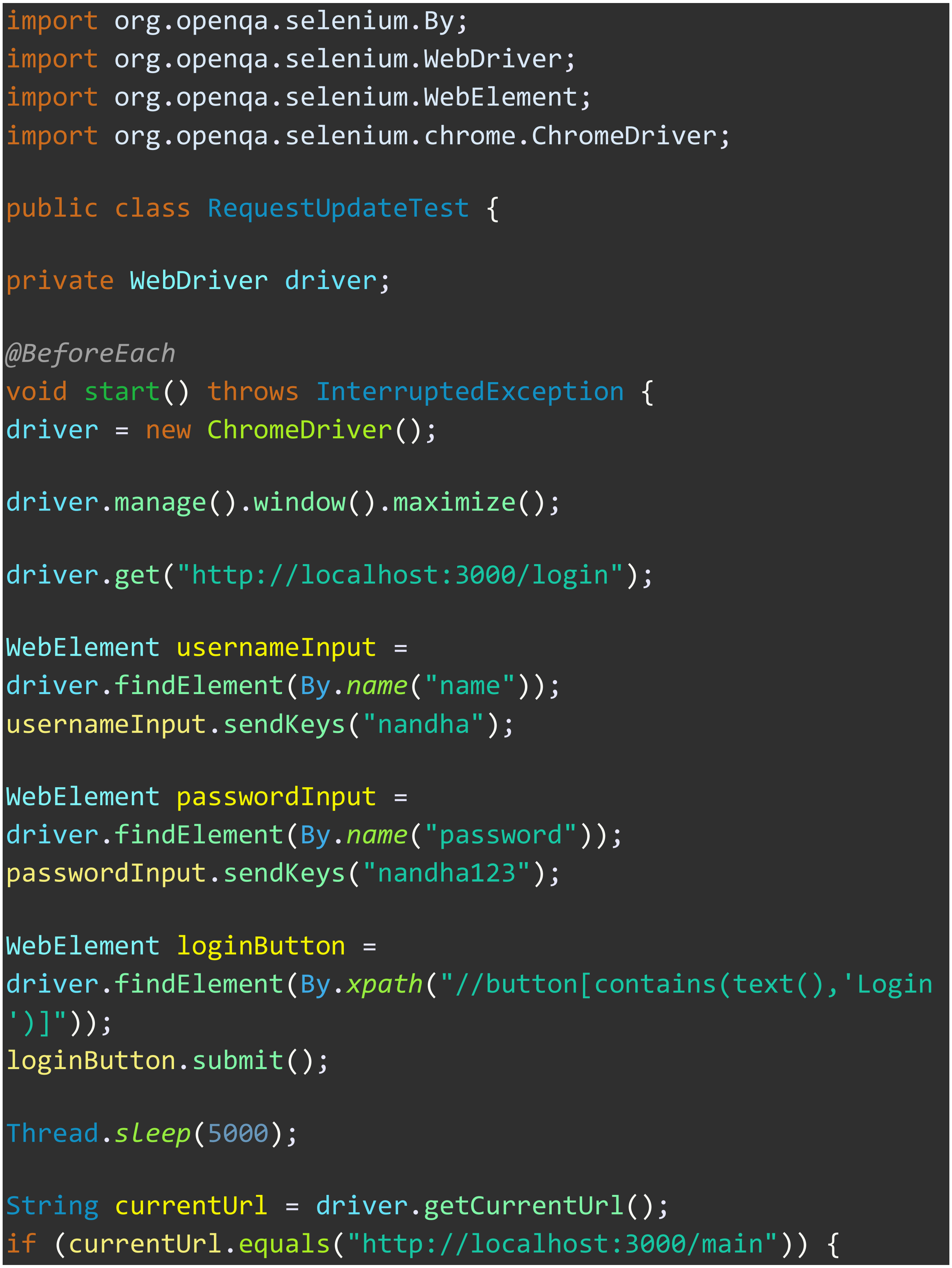


```
public class LoginTest {

private WebDriver driver;

@BeforeEach
void start() {
driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("http://localhost:3000/login");
}

@AfterEach
void close() {

driver.quit();
}

@Test
public void
testLoginWithValidCredentialsAsDonatorAndEmployee() throws
InterruptedException {
WebElement usernameInput =
driver.findElement(By.name("name"));
usernameInput.sendKeys("Nandhakumaran");

WebElement passwordInput =
driver.findElement(By.name("password"));
passwordInput.sendKeys("123");

WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login
')]"));
loginButton.submit();
```

```
Thread.sleep(5000);

String currentUrl = driver.getCurrentUrl();
if (currentUrl.equals("http://localhost:3000/main")) {
System.out.println("Positive login test passed for
employee!");
} else if
(currentUrl.equals("http://localhost:3000/usermain")) {
System.out.println("Positive login test passed for
donator!");
} else {
System.out.println("Positive login test failed. Unexpected
redirection.");
}

Thread.sleep(10000);

}

@Test
public void testLoginWithInvalidCredentials() throws
InterruptedException {

WebElement usernameInput =
driver.findElement(By.name("name"));
usernameInput.sendKeys("InvalidUser");

WebElement passwordInput =
driver.findElement(By.name("password"));
passwordInput.sendKeys("InvalidPassword");

WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login
')]"));
```

```
loginButton.submit();

WebElement errorMessage = null;
try {
errorMessage = driver.findElement(By.className("text-red-500"));
} catch (Exception e) {

}
if (errorMessage != null && errorMessage.isDisplayed()) {
System.out.println("Negative login test passed for invalid credentials!");
} else {
System.out.println("Negative login test failed. Expected error message not found.");
}

Thread.sleep(10000);

}

}
```

```
package com.nandha.SeleniumTest;

import static org.junit.Assert.assertEquals;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
```



```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class RequestUpdateTest {

private WebDriver driver;

@BeforeEach
void start() throws InterruptedException {
driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("http://localhost:3000/login");

WebElement usernameInput =
driver.findElement(By.name("name"));
usernameInput.sendKeys("nandha");

WebElement passwordInput =
driver.findElement(By.name("password"));
passwordInput.sendKeys("nandha123");

WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login
')]"));
loginButton.submit();

Thread.sleep(5000);

String currentUrl = driver.getCurrentUrl();
if (currentUrl.equals("http://localhost:3000/main")) {
```

```
System.out.println("Positive login test passed for
employee!");
} else if
(currentUrl.equals("http://localhost:3000/usermain")) {
System.out.println("Positive login test passed for
donator!");
} else {
System.out.println("Positive login test failed. Unexpected
redirection.");
}

WebElement donatorDetailsButton =
driver.findElement(By.xpath("//button[contains(text(),'Handl
e Requests')]"));
donatorDetailsButton.click();
}

@AfterEach
void close() {
driver.quit();
}

@Test
void testUpdateRequestValid() throws InterruptedException {

WebElement requestUpdateButton = driver
.findElement(By.xpath("//*[@id=\"root\"]/div/table/tbody/tr[
1]/td[9]/a/button"));
requestUpdateButton.click();

driver.findElement(By.name("status")).clear();
WebElement statusField =
driver.findElement(By.name("status"));
statusField.sendKeys("Pending");
```

```
WebElement submitButton =
driver.findElement(By.xpath("//*[@id=\"edit\"]/div/form/button"));
submitButton.submit();

Thread.sleep(5000);

assertEquals("Data Updated Successfully",
driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

Thread.sleep(5000);

}

}
```

package com.nandha.SeleniumTest;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import static org.junit.Assert.assertEquals;

import org.junit.jupiter.api.AfterEach;

```
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

public class SignUpTest {

private WebDriver driver;

@BeforeEach
void start() {
driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("http://localhost:3000/signup");
}

@AfterEach
void close() {
driver.quit();
}

@Test
public void testSignupWithValidData() throws InterruptedException {

WebElement nameInput = driver.findElement(By.name("name"));
nameInput.sendKeys("nandha");

WebElement typeInput = driver.findElement(By.name("type"));
```

```
typeInput.sendKeys("donator");

WebElement emailInput = driver.findElement(By.name("email"));
emailInput.sendKeys("nandha@example.com");

WebElement passwordInput = driver.findElement(By.name("password"));
passwordInput.sendKeys("nandha64");

WebElement signupButton =
driver.findElement(By.xpath("//button[contains(text(),'Sign Up')]"));
signupButton.submit();

Thread.sleep(5000);
assertEquals("Data Added Successfully", driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

assertEquals("http://localhost:3000/login", driver.getCurrentUrl());

Thread.sleep(10000);

}

@Test
public void testSignupWithEmptyFields() throws InterruptedException {

WebElement nameInput = driver.findElement(By.name("name"));
```

```
nameInput.sendKeys("");

WebElement typeInput = driver.findElement(By.name("type"));
typeInput.sendKeys("Employee");

WebElement emailInput = driver.findElement(By.name("email"));
emailInput.sendKeys("sena@example.com");

WebElement passwordInput = driver.findElement(By.name("password"));
passwordInput.sendKeys("sena123");

WebElement signupButton =
driver.findElement(By.xpath("//button[contains(text(),'Sign Up')]"));
signupButton.submit();

Thread.sleep(5000);
assertEquals("Please enter Valid Name!!!", driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

Thread.sleep(5000);
assertEquals("Please Enter the Valid Inputs!!!", driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

assertEquals("http://localhost:3000/signup", driver.getCurrentUrl());

Thread.sleep(10000);
```

```
}

}

package com.nandha.SeleniumTest;

import static org.junit.Assert.assertEquals;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class UpdateUserTest {

private WebDriver driver;

@BeforeEach
void start() throws InterruptedException {
driver = new ChromeDriver();
```

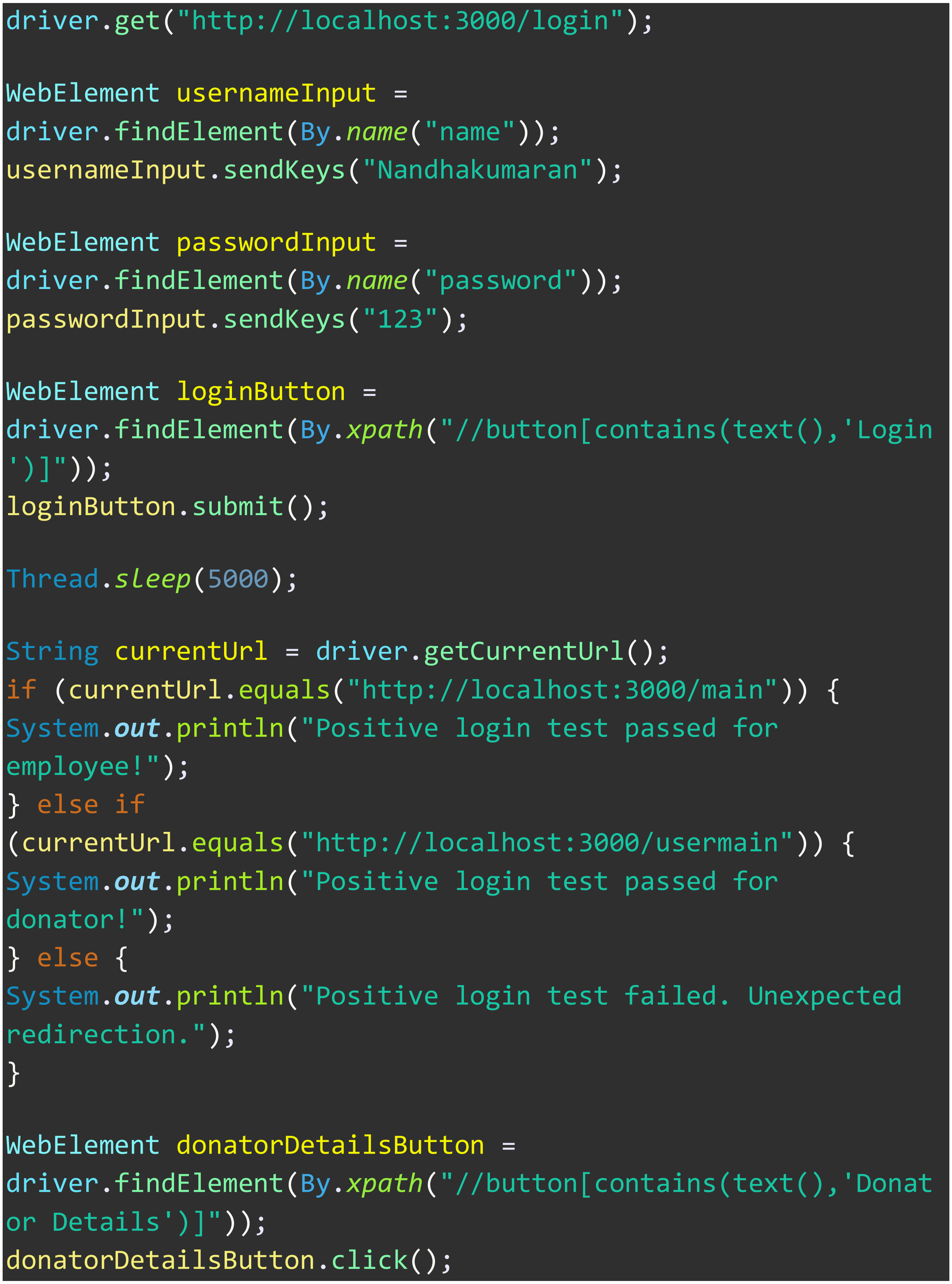
```
driver.manage().window().maximize();

driver.get("http://localhost:3000/login");

WebElement usernameInput = driver.findElement(By.name("name"));
usernameInput.sendKeys("Nandhakumaran");

WebElement passwordInput = driver.findElement(By.name("password"));
passwordInput.sendKeys("123");

WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login')]"));
loginButton.submit();

Thread.sleep(5000);

String currentUrl = driver.getCurrentUrl();
if (currentUrl.equals("http://localhost:3000/main")) {
System.out.println("Positive login test passed for employee!");
} else if (currentUrl.equals("http://localhost:3000/usermain")) {
System.out.println("Positive login test passed for donator!");
} else {
System.out.println("Positive login test failed. Unexpected redirection.");
}
```

```
WebElement donatorDetailsButton =
driver.findElement(By.xpath("//button[contains(text(),'Donator Details')]"));

donatorDetailsButton.click();

}


@AfterEach

void close() {

driver.quit();

}


@Test

void testUpdateUserValid() throws InterruptedException {


WebElement donatorUpdateButton = driver

.findElement(By.xpath("//*[@id=\"root\"]/div/table/tbody/tr[3]/td[8]/a/button"));

donatorUpdateButton.click();


driver.findElement(By.name("name")).clear();

WebElement nameField = driver.findElement(By.name("name"));

nameField.sendKeys("Ponraj Marikannan");


driver.findElement(By.name("emailid")).clear();

WebElement emailField = driver.findElement(By.name("emailid"));

emailField.sendKeys("vasi@example.com");


driver.findElement(By.name("address")).clear();
```

```
WebElement addressField = driver.findElement(By.name("address"));
addressField.sendKeys("Mandelanagar");

driver.findElement(By.name("phoneNumber")).clear();
WebElement phoneField = driver.findElement(By.name("phoneNumber"));
phoneField.sendKeys("0987654321");

driver.findElement(By.name("donation")).clear();
WebElement donationField = driver.findElement(By.name("donation"));
donationField.sendKeys("100");

WebElement submitButton =
driver.findElement(By.xpath("//*[@id=\"edit2\"]/div/form/button"));
submitButton.submit();

Thread.sleep(5000);

assertEquals("Data Updated Successfully", driver.switchTo().alert().getText());

driver.switchTo().alert().accept();
Thread.sleep(5000);

}
```

}

```
package com.nandha.SeleniumTest;

import static org.junit.Assert.assertEquals;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class UserDeleteTest {

private WebDriver driver;

@AfterEach
void close() {
driver.quit();
}

@Test
void testDeleteUser() throws InterruptedException {

driver = new ChromeDriver();

driver.manage().window().maximize();
```



```java
driver.get("http://localhost:3000/login");

WebElement usernameInput =
driver.findElement(By.name("name"));
usernameInput.sendKeys("Nandhakumaran");

WebElement passwordInput =
driver.findElement(By.name("password"));
passwordInput.sendKeys("123");

WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login')]"));
loginButton.submit();

Thread.sleep(5000);

String currentUrl = driver.getCurrentUrl();
if (currentUrl.equals("http://localhost:3000/main")) {
System.out.println("Positive login test passed for
employee!");
} else if
(currentUrl.equals("http://localhost:3000/usermain")) {
System.out.println("Positive login test passed for
donator!");
} else {
System.out.println("Positive login test failed. Unexpected
redirection.");
}

WebElement donatorDetailsButton =
driver.findElement(By.xpath("//button[contains(text(),'Donator Details')]"));
donatorDetailsButton.click();
```

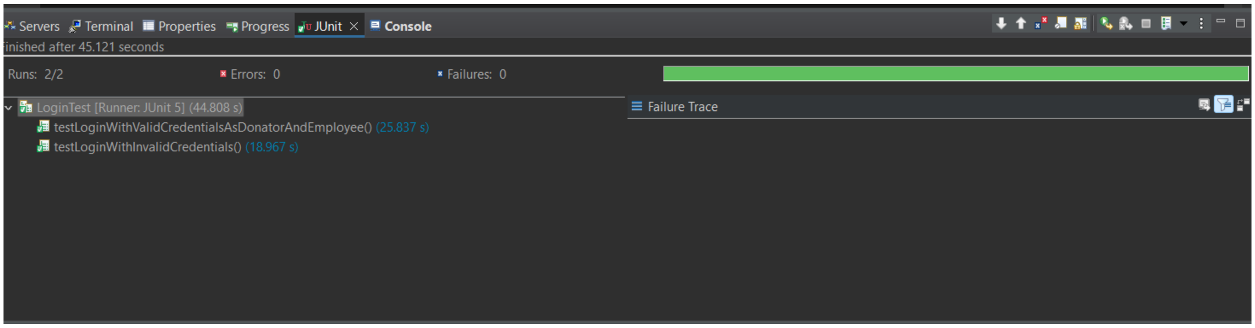
```
WebElement donatorDeleteButton = driver
.findElement(By.xpath("//*[@id=\"root\"]/div/table/tbody/tr[
4]/td[8]/button"));
donatorDeleteButton.click();

Thread.sleep(5000);
assertEquals("Do you want to delete?",
driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

Thread.sleep(5000);
assertEquals("Record has been deleted",
driver.switchTo().alert().getText());
driver.switchTo().alert().accept();

Thread.sleep(5000);

}

}
```

```
package com.nandha.SeleniumTest;


import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class UserViewTest {

private WebDriver driver;

@AfterEach
void close() {
driver.quit();
}

@Test
void testViewUser() throws InterruptedException {

driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("http://localhost:3000/login");

WebElement usernameInput = driver.findElement(By.name("name"));
usernameInput.sendKeys("Nandhakumaran");

WebElement passwordInput = driver.findElement(By.name("password"));
passwordInput.sendKeys("123");
WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login')]"));

loginButton.submit();


Thread.sleep(5000);


String currentUrl = driver.getCurrentUrl();

if (currentUrl.equals("http://localhost:3000/main")) {

System.out.println("Positive login test passed for employee!");

} else if (currentUrl.equals("http://localhost:3000/usermain")) {

System.out.println("Positive login test passed for donator!");

} else {

System.out.println("Positive login test failed. Unexpected redirection.");

}


WebElement donatorDetailsButton =
driver.findElement(By.xpath("//button[contains(text(),'Donator Details')]"));

donatorDetailsButton.click();


Thread.sleep(5000);


}


}
```

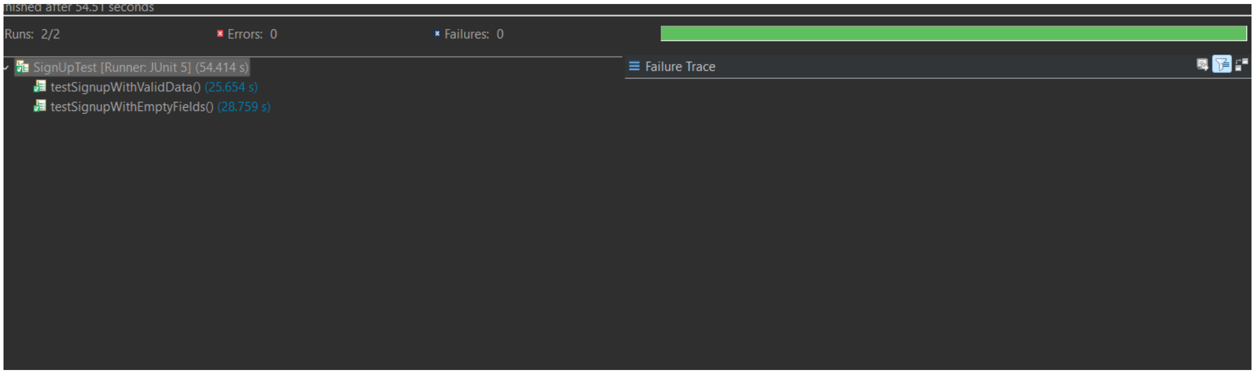
```java
package com.nandha.SeleniumTest;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class ViewRequestTest {

private WebDriver driver;

@AfterEach
void close() {
driver.quit();
}

@Test
void testViewRequest() throws InterruptedException {

driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("http://localhost:3000/login");

WebElement usernameInput =
driver.findElement(By.name("name"));
usernameInput.sendKeys("nandha");
WebElement passwordInput =
driver.findElement(By.name("password"));
passwordInput.sendKeys("nandha123");

WebElement loginButton =
driver.findElement(By.xpath("//button[contains(text(),'Login
')]"));
loginButton.submit();

Thread.sleep(5000);

String currentUrl = driver.getCurrentUrl();
if (currentUrl.equals("http://localhost:3000/main")) {
System.out.println("Positive login test passed for
employee!");
} else if
(currentUrl.equals("http://localhost:3000/usermain")) {
System.out.println("Positive login test passed for
donator!");
} else {
System.out.println("Positive login test failed. Unexpected
redirection.");
}

WebElement donatorDetailsButton =
driver.findElement(By.xpath("//button[contains(text(),'Handl
e Requests')]"));
donatorDetailsButton.click();


Thread.sleep(5000);

}
```

```
}
```

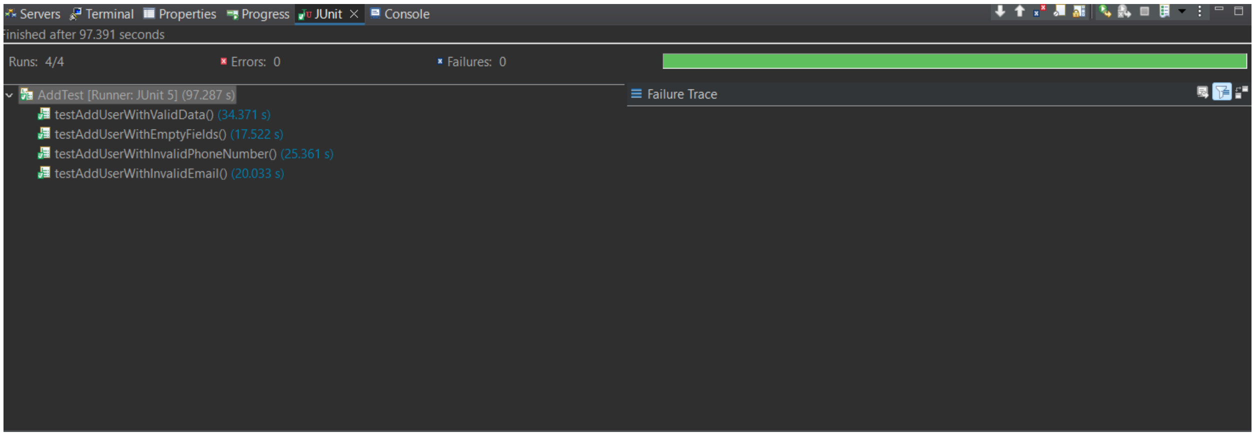
ScreenShots:



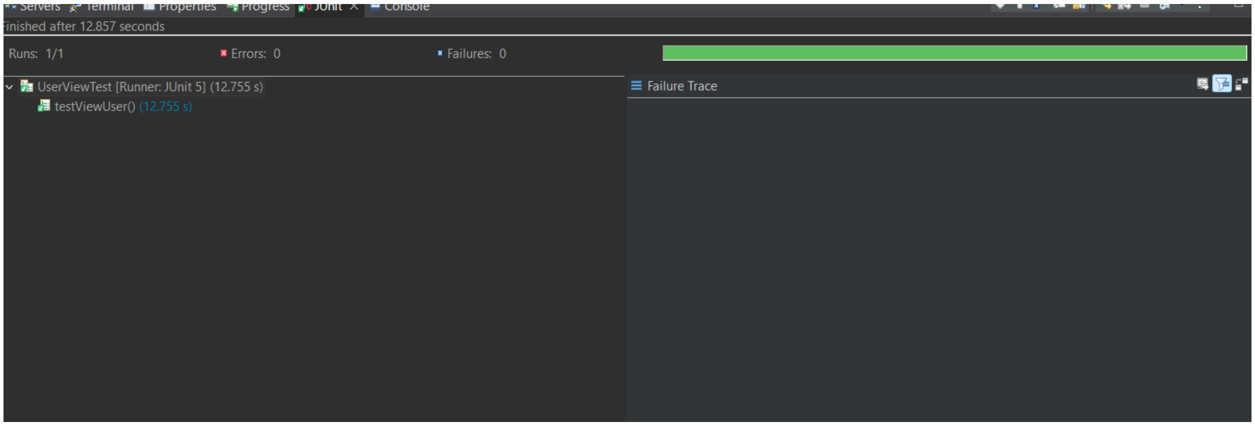
```
inished after 45.121 seconds
Runs:  2/2          Errors:  0          Failures:  0

LoginTest [Runner: JUnit 5] (44.808 s)
    testLoginWithValidCredentialsAsDonatorAndEmployee() (25.837 s)
    testLoginWithInvalidCredentials() (18.967 s)
```



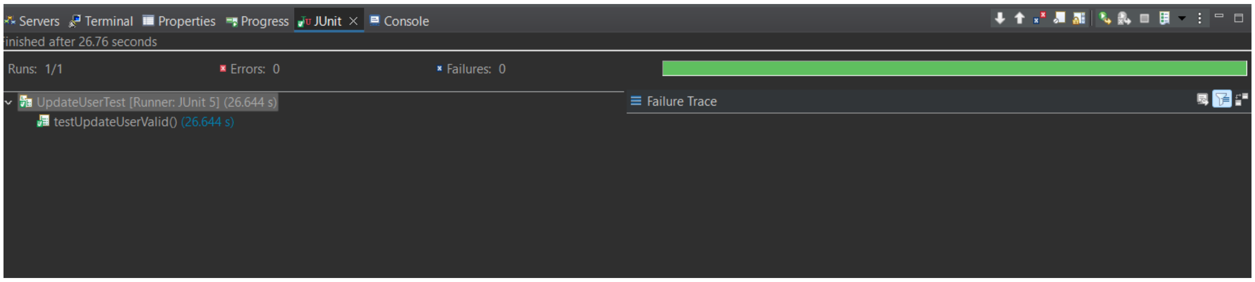
```
nished after 54.51 seconds
Runs:  2/2          Errors:  0          Failures:  0

SignUpTest [Runner: JUnit 5] (54.414 s)
    testSignupWithValidData() (25.654 s)
    testSignupWithEmptyFields() (28.759 s)
```



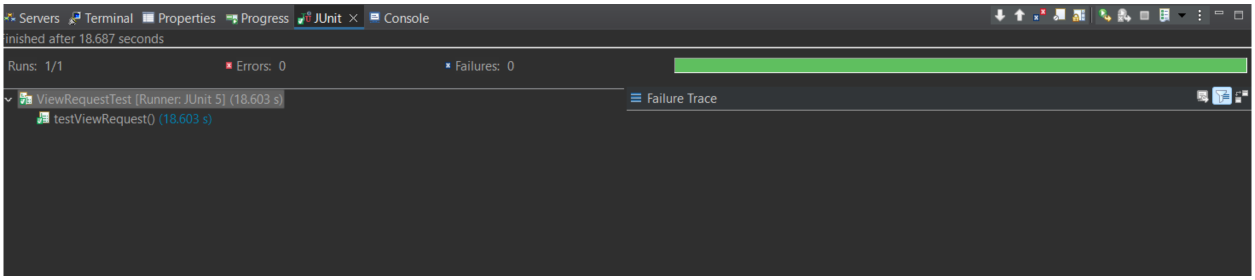
```
inished after 97.391 seconds
Runs:  4/4          Errors:  0          Failures:  0

AddTest [Runner: JUnit 5] (97.287 s)
    testAddUserWithValidData() (34.371 s)
    testAddUserWithEmptyFields() (17.522 s)
    testAddUserWithInvalidPhoneNumber() (25.361 s)
    testAddUserWithInvalidEmail() (20.033 s)
```



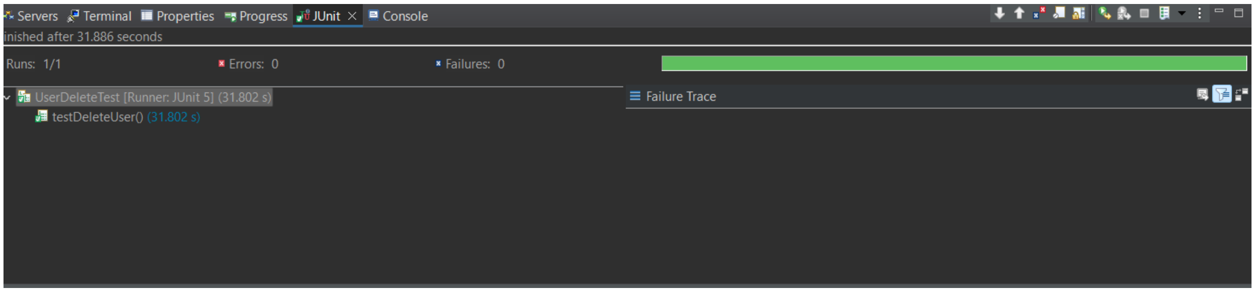
```
inished after 12.857 seconds
Runs:  1/1          Errors:  0          Failures:  0

UserViewTest [Runner: JUnit 5] (12.755 s)
    testViewUser() (12.755 s)
```



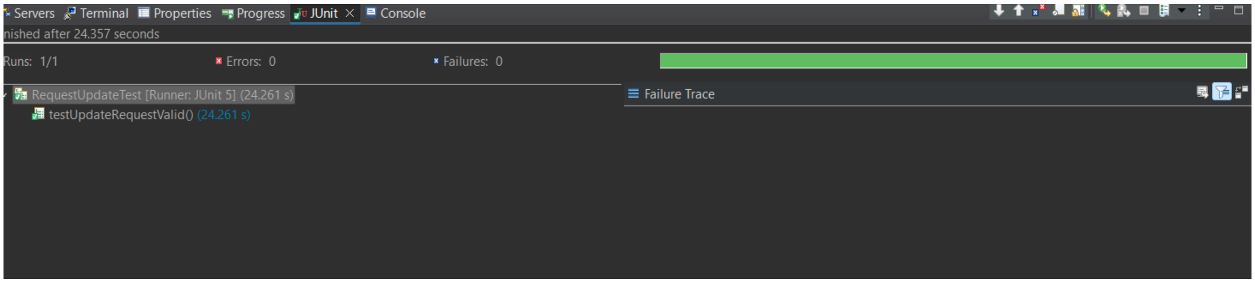
```
inished after 26.76 seconds
Runs:  1/1          Errors:  0          Failures:  0

UpdateUserTest [Runner: JUnit 5] (26.644 s)
    testUpdateUserValid() (26.644 s)
```



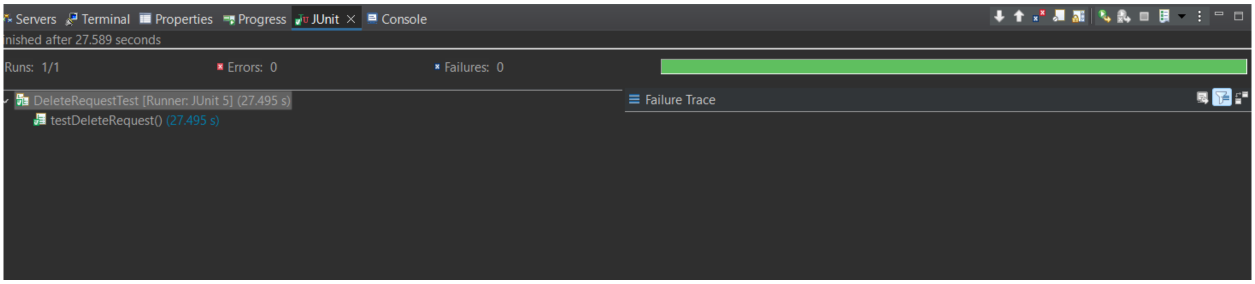
```
inished after 18.687 seconds
Runs:  1/1          Errors:  0          Failures:  0
ViewRequestTest [Runner: JUnit 5] (18.603 s)
    testViewRequest() (18.603 s)
```



```
inished after 31.886 seconds
Runs:  1/1          Errors:  0          Failures:  0
UserDeleteTest [Runner: JUnit 5] (31.802 s)
    testDeleteUser() (31.802 s)
```

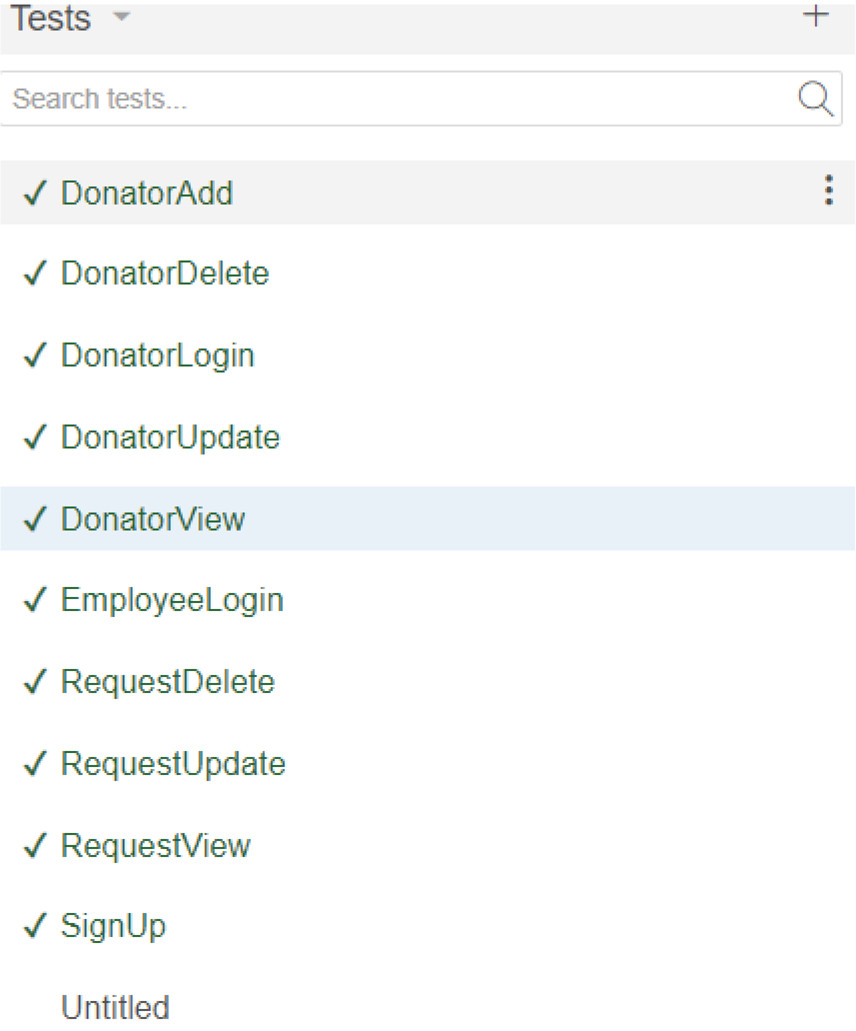


```
nished after 24.357 seconds
Runs:  1/1          Errors:  0          Failures:  0
RequestUpdateTest [Runner: JUnit 5] (24.261 s)
    testUpdateRequestValid() (24.261 s)
```



```
inished after 27.589 seconds
Runs:  1/1          Errors:  0          Failures:  0
DeleteRequestTest [Runner: JUnit 5] (27.495 s)
    testDeleteRequest() (27.495 s)
```

Selenium IDE Scripts:



✓ DonatorAdd

✓ DonatorDelete

✓ DonatorLogin

✓ DonatorUpdate

✓ DonatorView

✓ EmployeeLogin

✓ RequestDelete

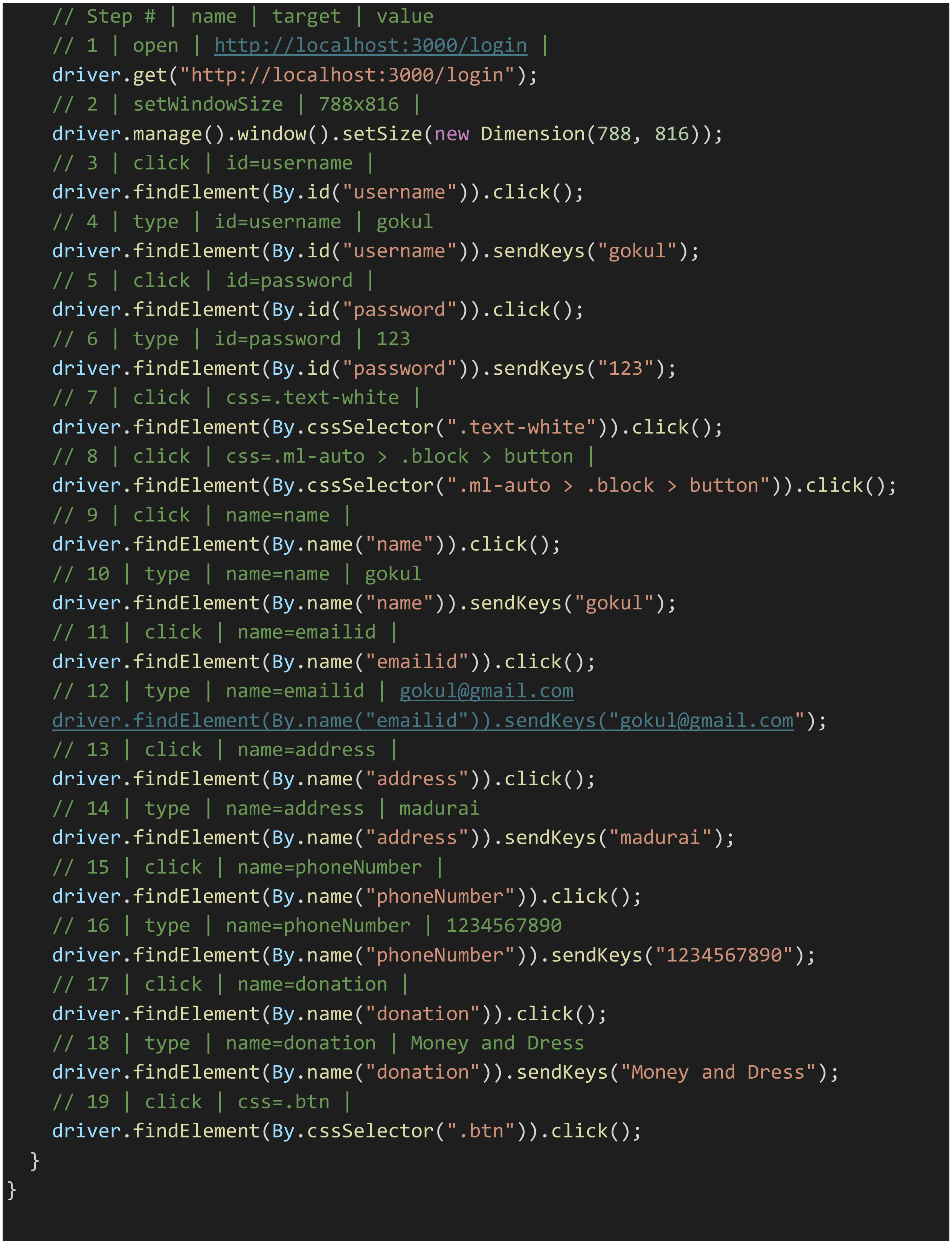
✓ RequestUpdate

✓ RequestView

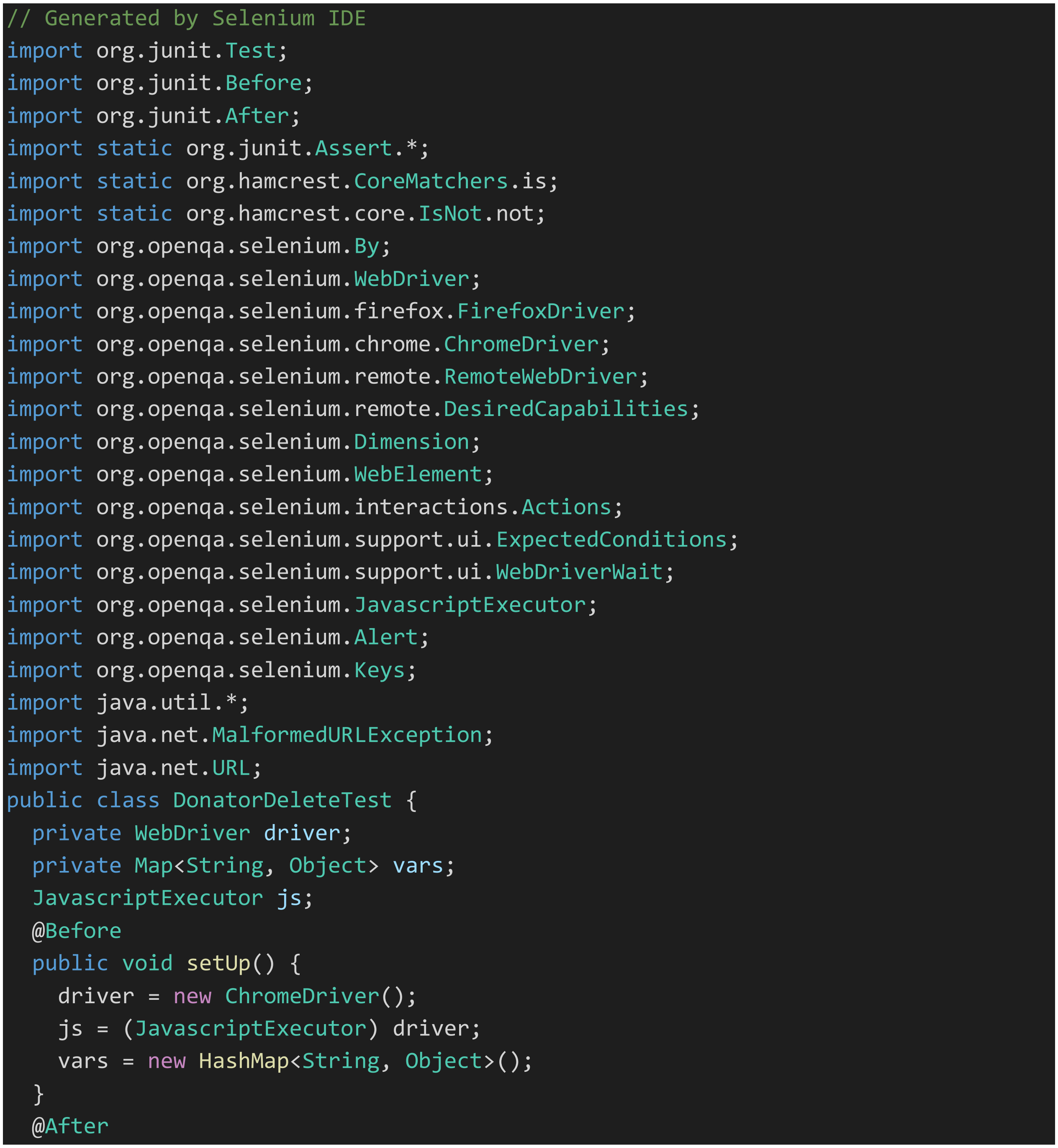
✓ SignUp

Untitled

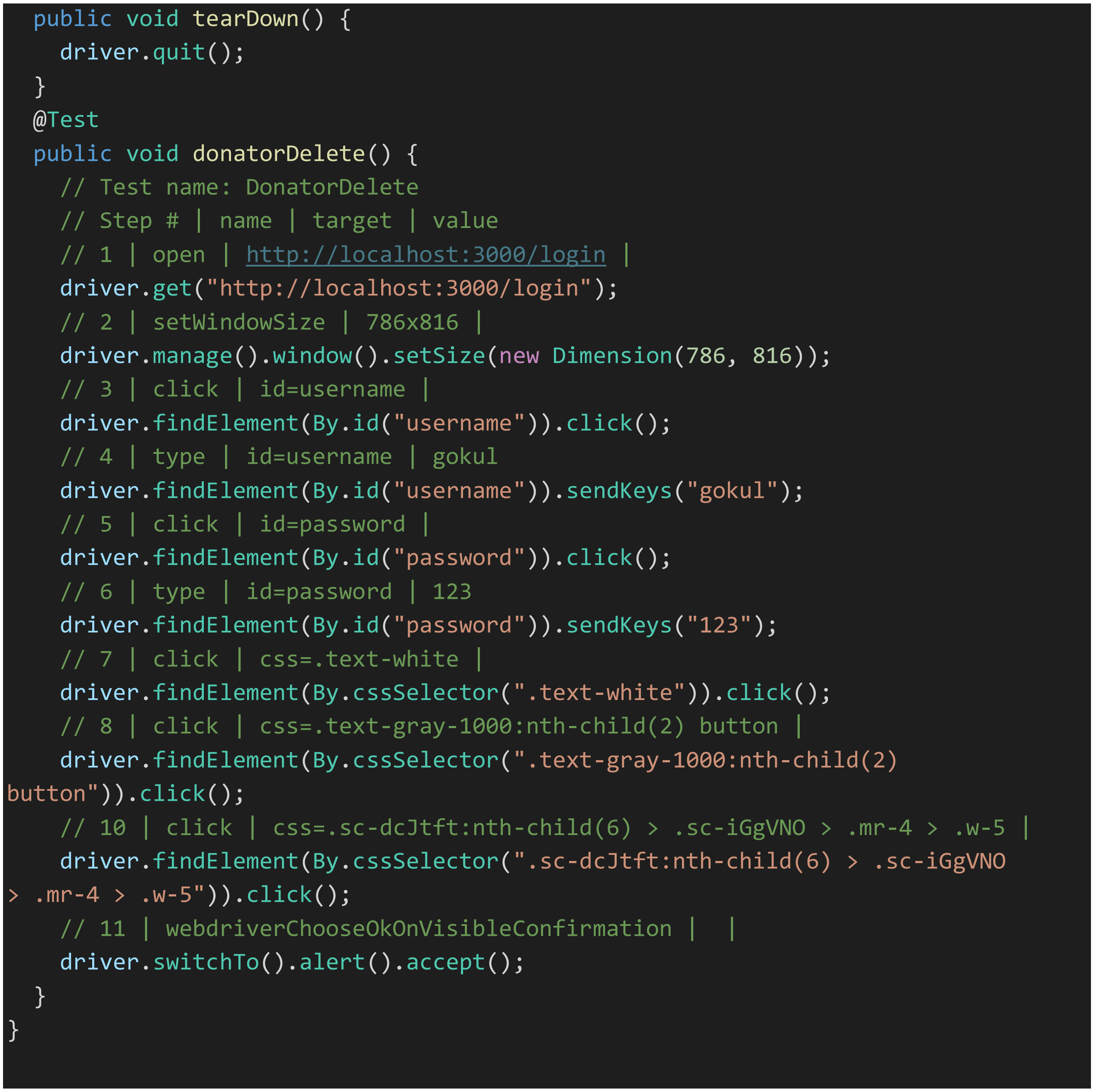
```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class DonatorAddTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void donatorAdd() {
    // Test name: DonatorAdd
```



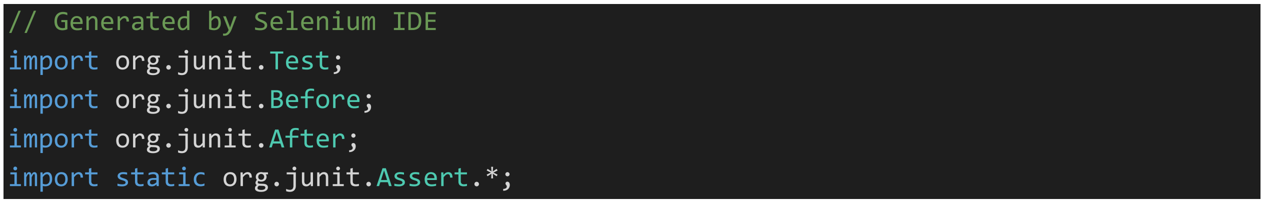
```
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 788x816 |
    driver.manage().window().setSize(new Dimension(788, 816));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | gokul
    driver.findElement(By.id("username")).sendKeys("gokul");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | 123
    driver.findElement(By.id("password")).sendKeys("123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
    // 8 | click | css=.ml-auto > .block > button |
    driver.findElement(By.cssSelector(".ml-auto > .block > button")).click();
    // 9 | click | name=name |
    driver.findElement(By.name("name")).click();
    // 10 | type | name=name | gokul
    driver.findElement(By.name("name")).sendKeys("gokul");
    // 11 | click | name=emailid |
    driver.findElement(By.name("emailid")).click();
    // 12 | type | name=emailid | gokul@gmail.com
    driver.findElement(By.name("emailid")).sendKeys("gokul@gmail.com");
    // 13 | click | name=address |
    driver.findElement(By.name("address")).click();
    // 14 | type | name=address | madurai
    driver.findElement(By.name("address")).sendKeys("madurai");
    // 15 | click | name=phoneNumber |
    driver.findElement(By.name("phoneNumber")).click();
    // 16 | type | name=phoneNumber | 1234567890
    driver.findElement(By.name("phoneNumber")).sendKeys("1234567890");
    // 17 | click | name=donation |
    driver.findElement(By.name("donation")).click();
    // 18 | type | name=donation | Money and Dress
    driver.findElement(By.name("donation")).sendKeys("Money and Dress");
    // 19 | click | css=.btn |
    driver.findElement(By.cssSelector(".btn")).click();
  }
}
```



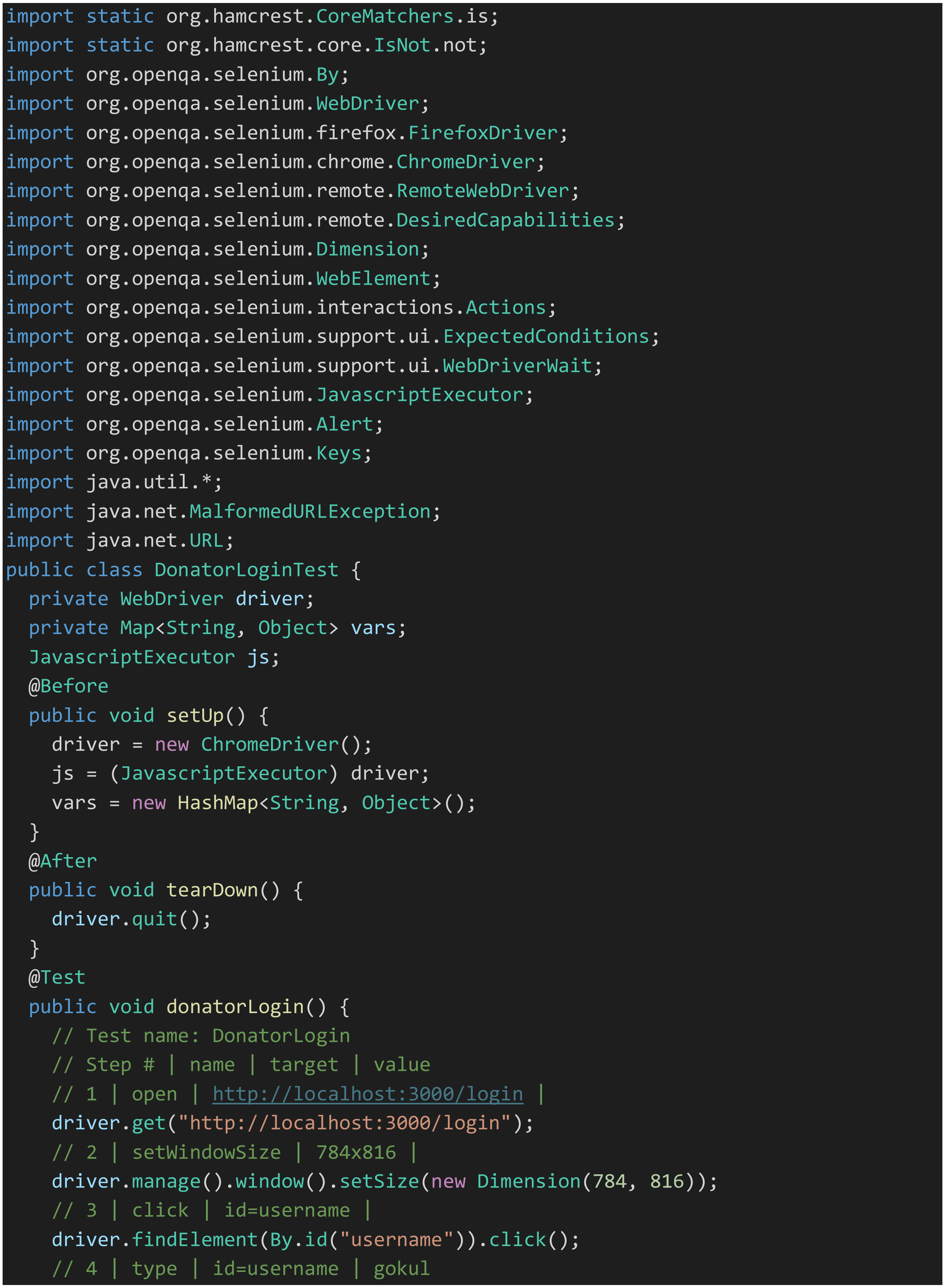
```java
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class DonatorDeleteTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
```



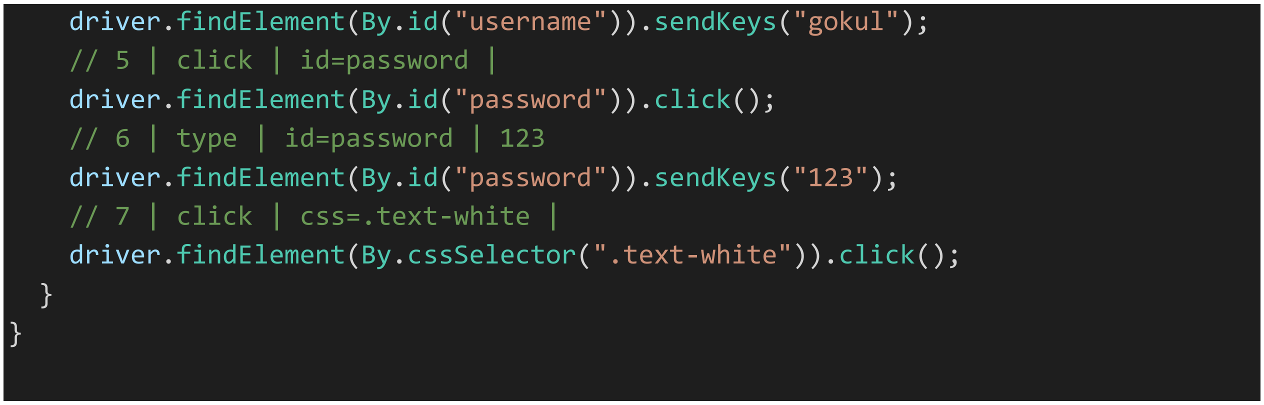
```java
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void donatorDelete() {
    // Test name: DonatorDelete
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 786x816 |
    driver.manage().window().setSize(new Dimension(786, 816));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | gokul
    driver.findElement(By.id("username")).sendKeys("gokul");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | 123
    driver.findElement(By.id("password")).sendKeys("123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
    // 8 | click | css=.text-gray-1000:nth-child(2) button |
    driver.findElement(By.cssSelector(".text-gray-1000:nth-child(2)
button")).click();
    // 10 | click | css=.sc-dcJtft:nth-child(6) > .sc-iGgVNO > .mr-4 > .w-5 |
    driver.findElement(By.cssSelector(".sc-dcJtft:nth-child(6) > .sc-iGgVNO
> .mr-4 > .w-5")).click();
    // 11 | webdriverChooseOkOnVisibleConfirmation |  |
    driver.switchTo().alert().accept();
  }
}
```



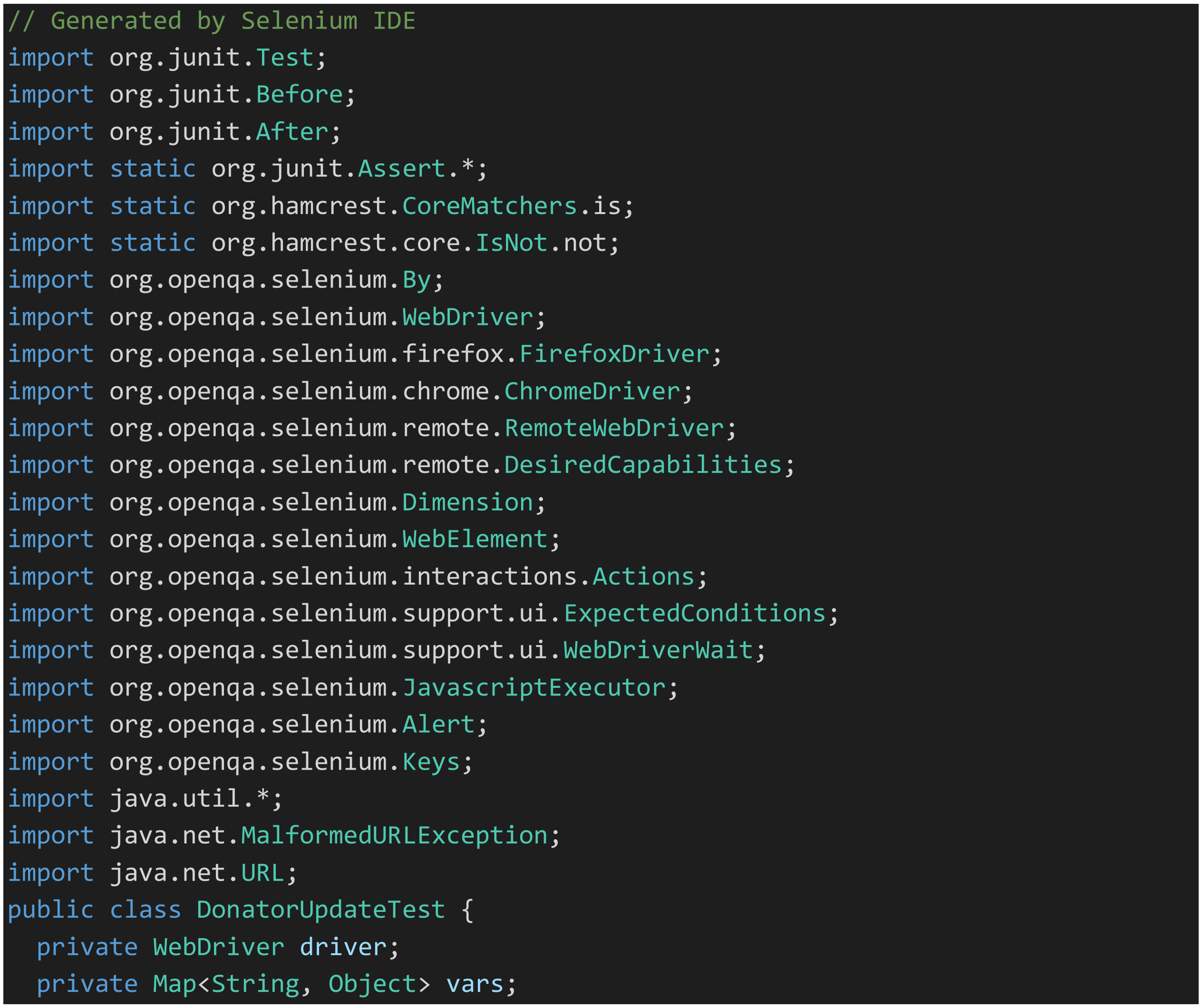
```java
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
```



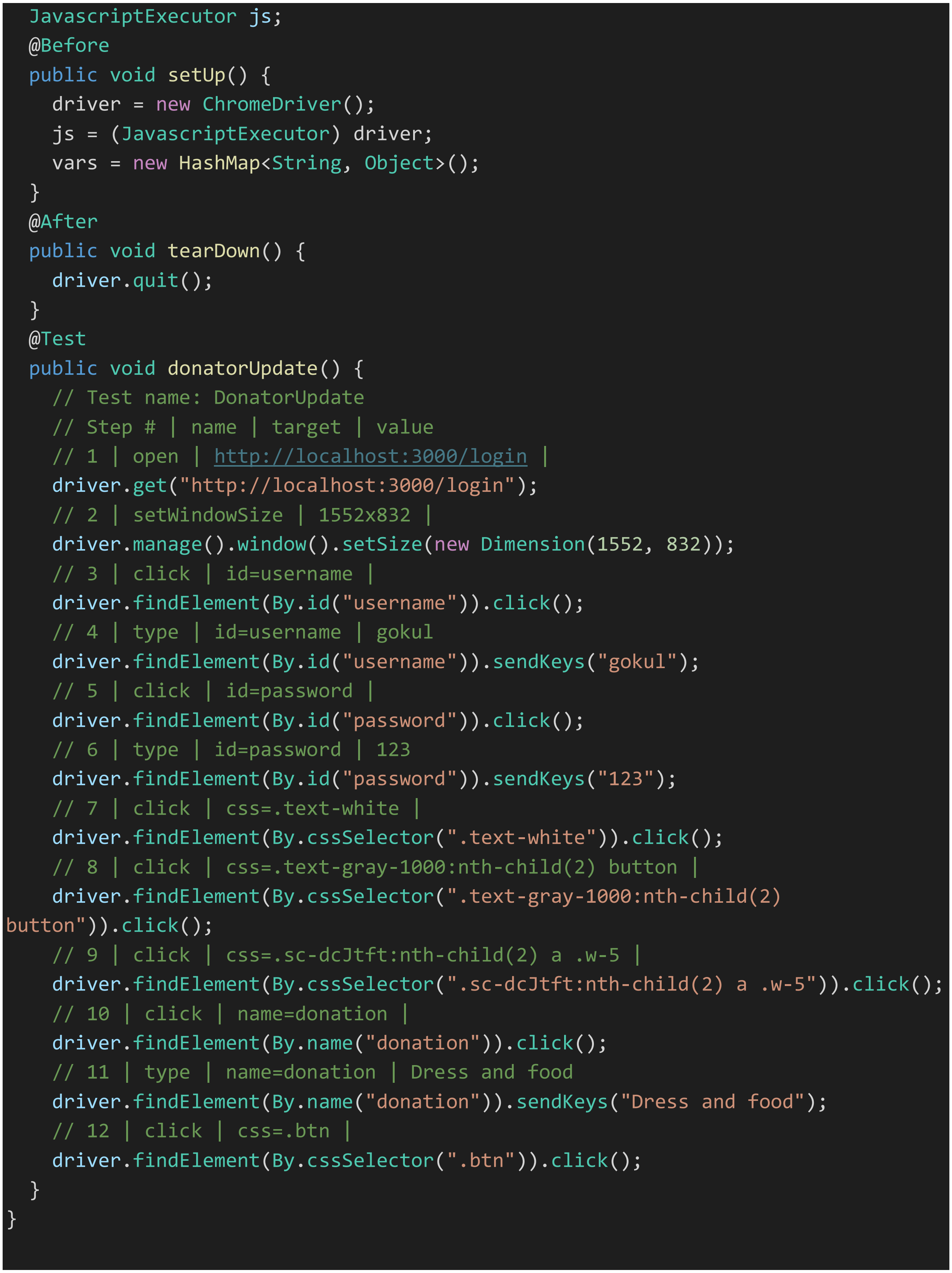
```
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class DonatorLoginTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void donatorLogin() {
    // Test name: DonatorLogin
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 784x816 |
    driver.manage().window().setSize(new Dimension(784, 816));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | gokul
```



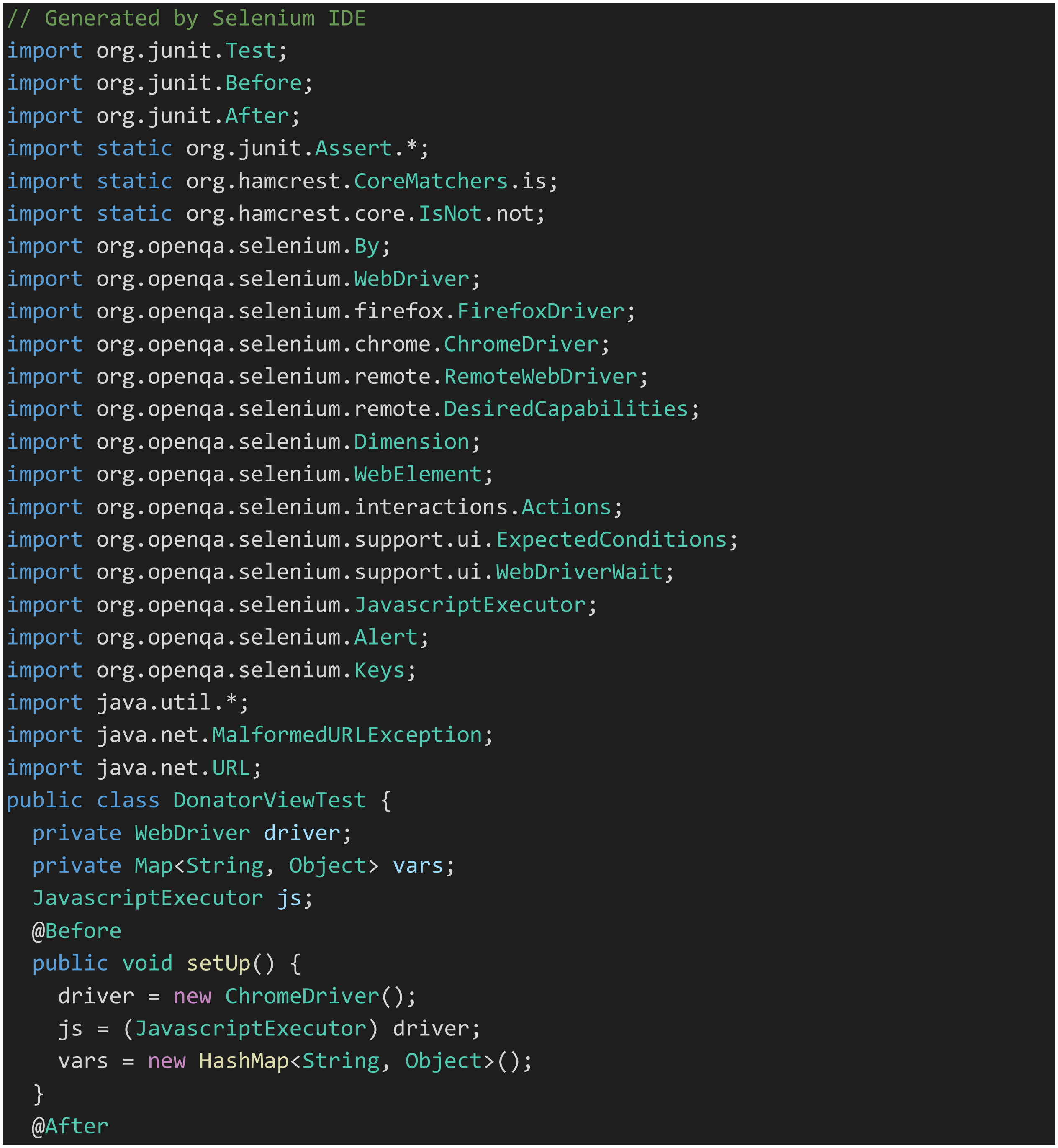
```
    driver.findElement(By.id("username")).sendKeys("gokul");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | 123
    driver.findElement(By.id("password")).sendKeys("123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
  }
}
```



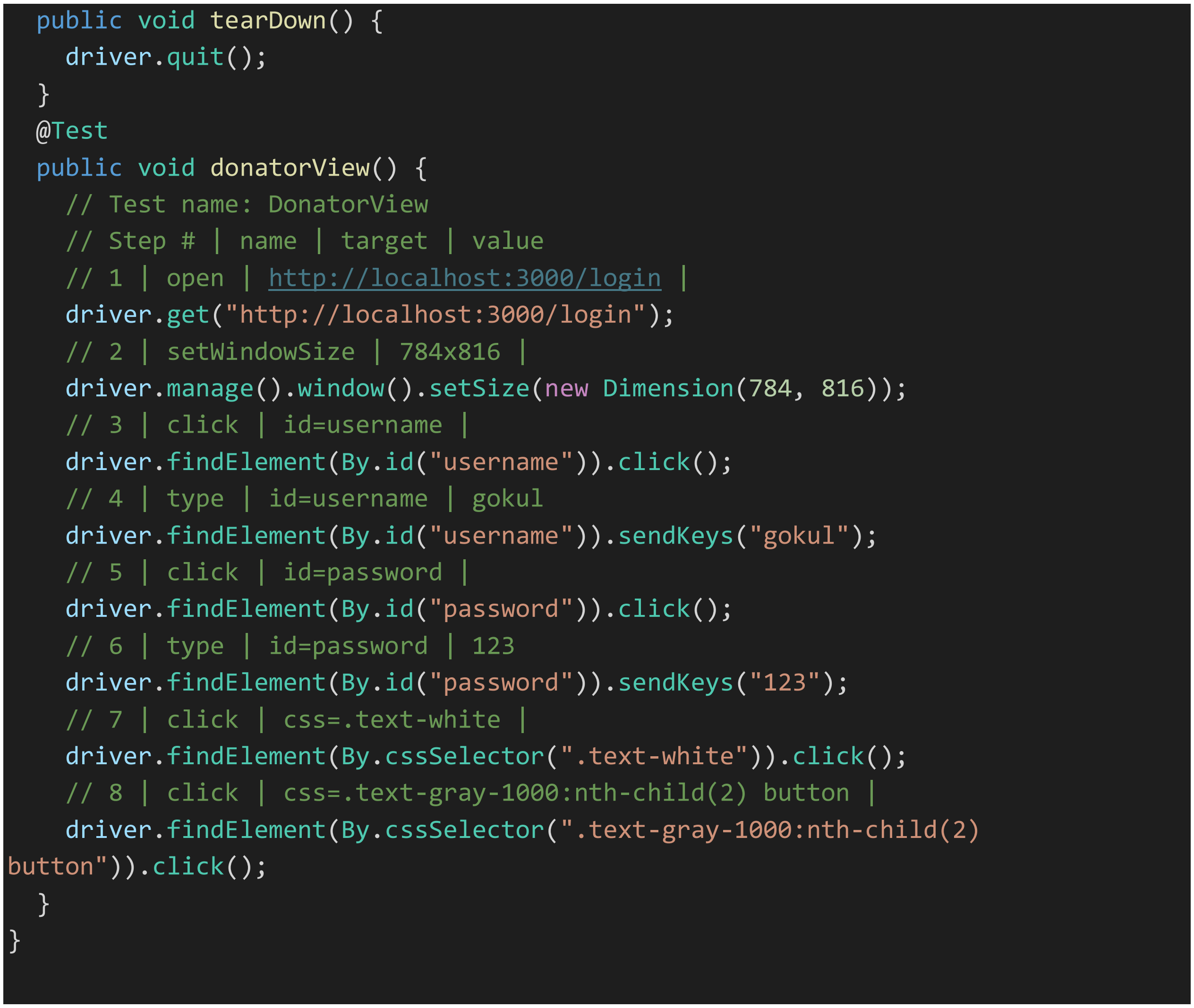
```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class DonatorUpdateTest {
  private WebDriver driver;
  private Map<String, Object> vars;
```



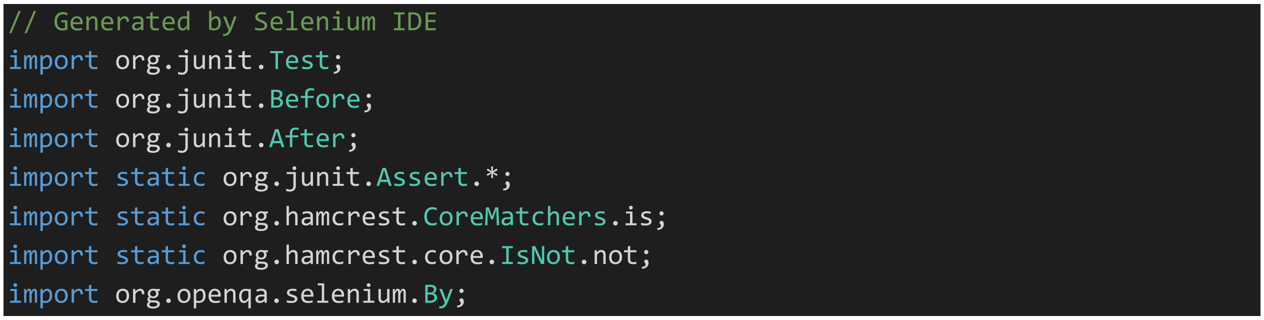
```
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void donatorUpdate() {
    // Test name: DonatorUpdate
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 1552x832 |
    driver.manage().window().setSize(new Dimension(1552, 832));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | gokul
    driver.findElement(By.id("username")).sendKeys("gokul");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | 123
    driver.findElement(By.id("password")).sendKeys("123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
    // 8 | click | css=.text-gray-1000:nth-child(2) button |
    driver.findElement(By.cssSelector(".text-gray-1000:nth-child(2)
button")).click();
    // 9 | click | css=.sc-dcJtft:nth-child(2) a .w-5 |
    driver.findElement(By.cssSelector(".sc-dcJtft:nth-child(2) a .w-5")).click();
    // 10 | click | name=donation |
    driver.findElement(By.name("donation")).click();
    // 11 | type | name=donation | Dress and food
    driver.findElement(By.name("donation")).sendKeys("Dress and food");
    // 12 | click | css=.btn |
    driver.findElement(By.cssSelector(".btn")).click();
  }
}
```



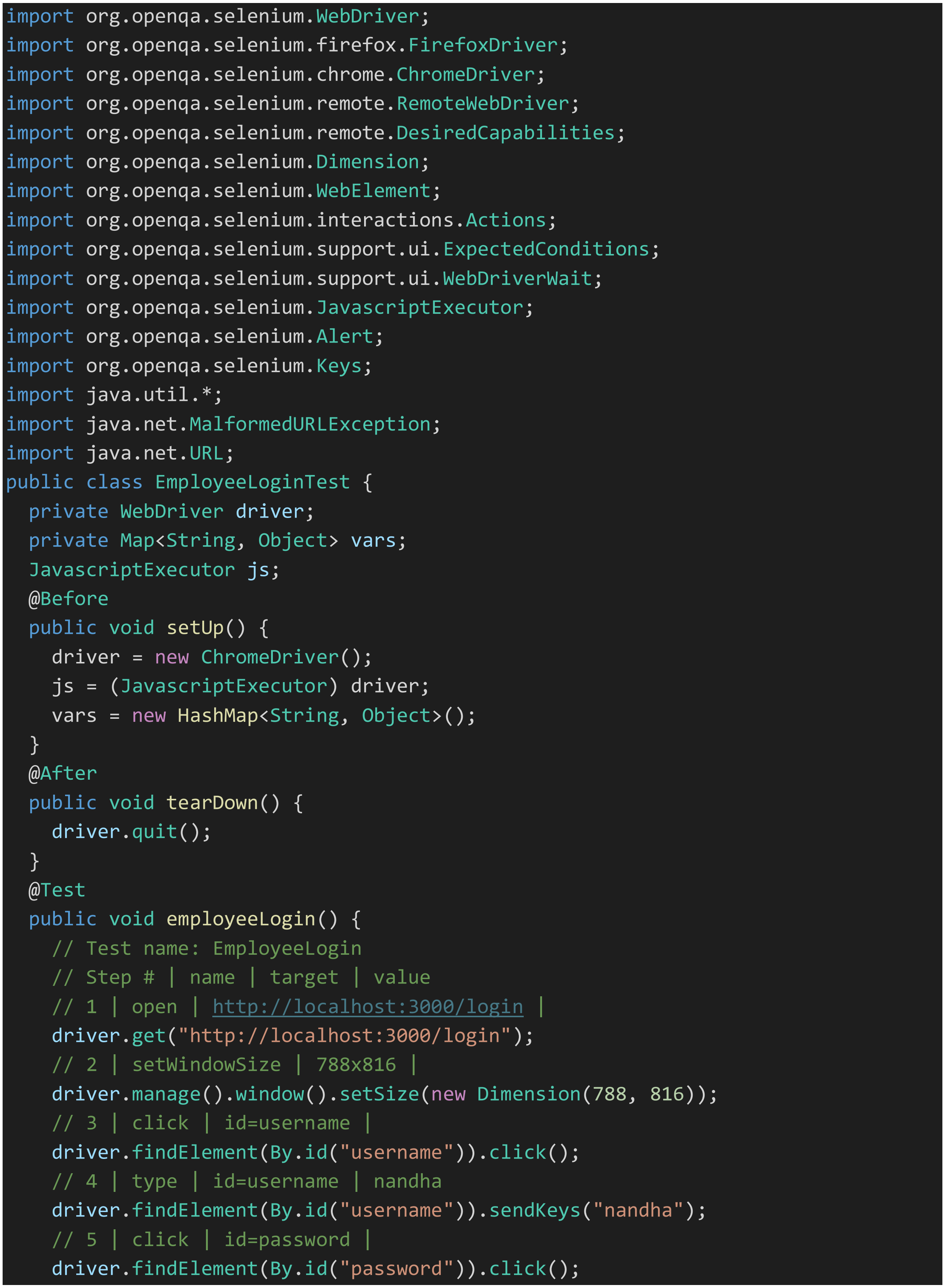
```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class DonatorViewTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
```



```
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void donatorView() {
    // Test name: DonatorView
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 784x816 |
    driver.manage().window().setSize(new Dimension(784, 816));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | gokul
    driver.findElement(By.id("username")).sendKeys("gokul");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | 123
    driver.findElement(By.id("password")).sendKeys("123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
    // 8 | click | css=.text-gray-1000:nth-child(2) button |
    driver.findElement(By.cssSelector(".text-gray-1000:nth-child(2)
button")).click();
  }
}
```

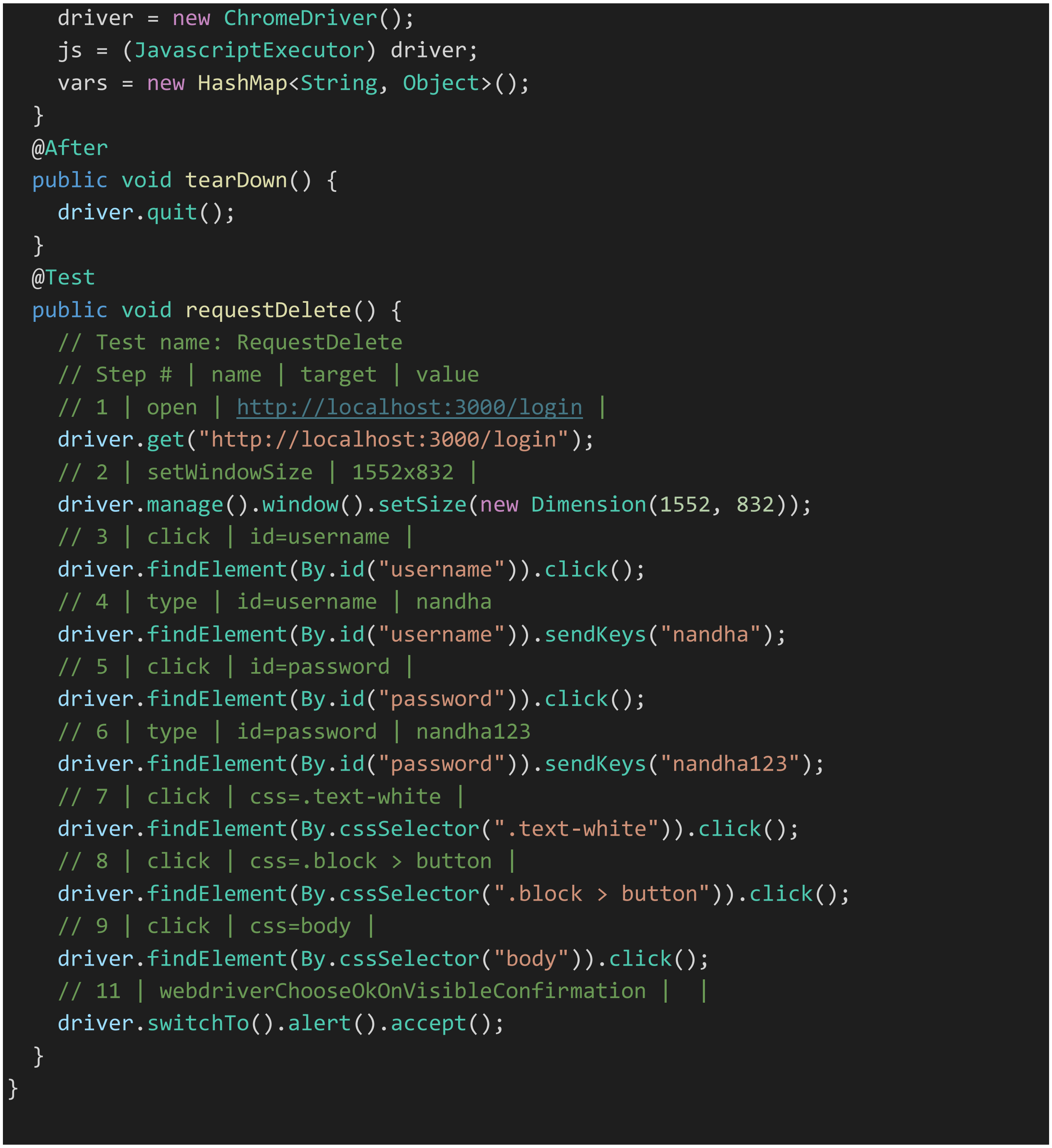


```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
```



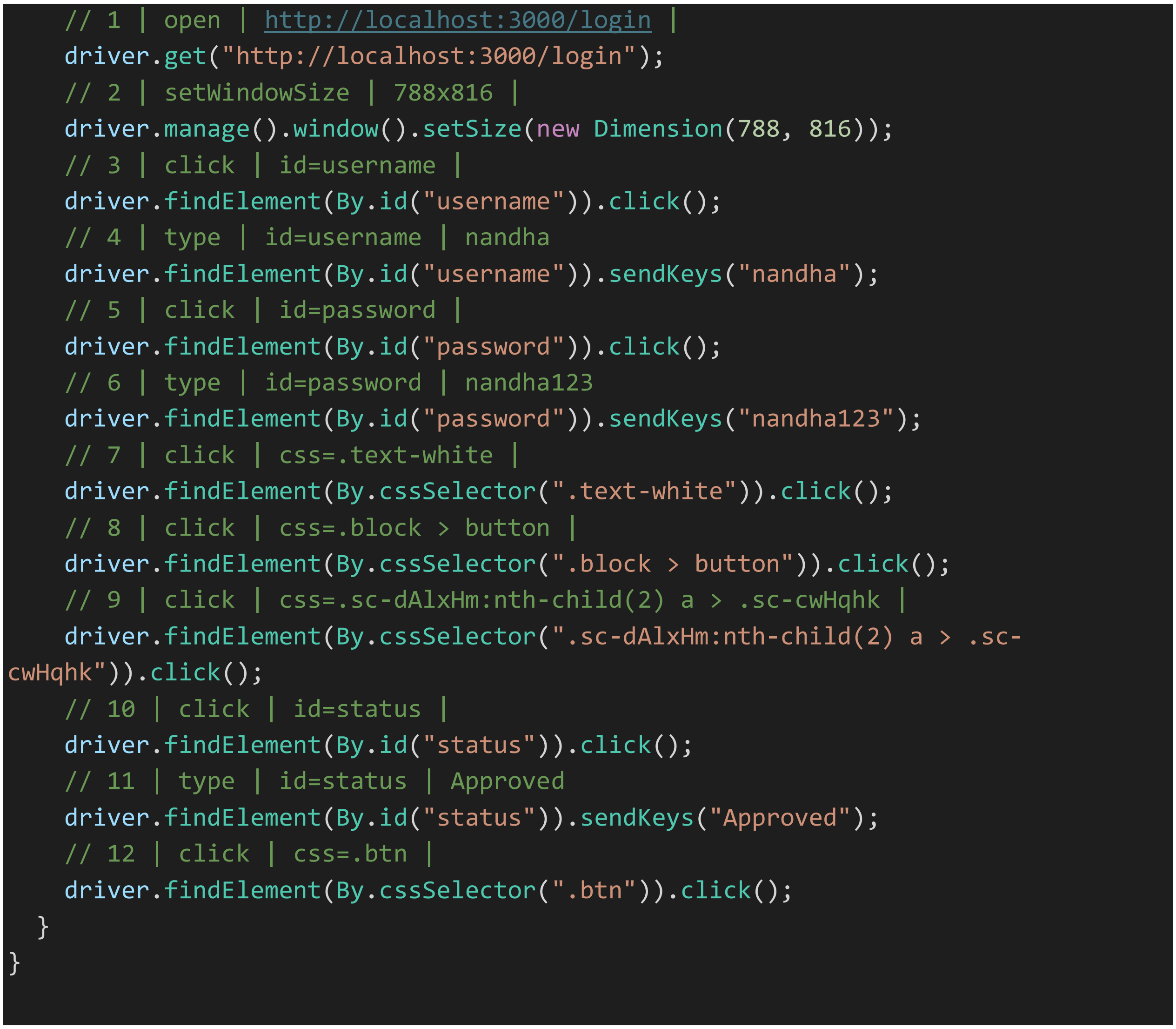
```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class EmployeeLoginTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void employeeLogin() {
    // Test name: EmployeeLogin
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 788x816 |
    driver.manage().window().setSize(new Dimension(788, 816));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | nandha
    driver.findElement(By.id("username")).sendKeys("nandha");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | nandha123
    driver.findElement(By.id("password")).sendKeys("nandha123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
  }
}
```

```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class RequestDeleteTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
```

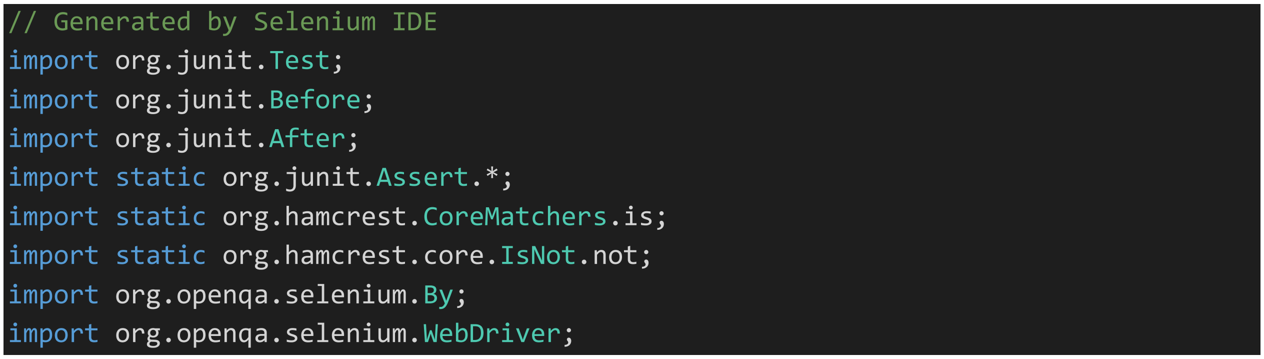


```
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void requestDelete() {
    // Test name: RequestDelete
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 1552x832 |
    driver.manage().window().setSize(new Dimension(1552, 832));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | nandha
    driver.findElement(By.id("username")).sendKeys("nandha");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | nandha123
    driver.findElement(By.id("password")).sendKeys("nandha123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
    // 8 | click | css=.block > button |
    driver.findElement(By.cssSelector(".block > button")).click();
    // 9 | click | css=body |
    driver.findElement(By.cssSelector("body")).click();
    // 11 | webdriverChooseOkOnVisibleConfirmation |  |
    driver.switchTo().alert().accept();
  }
}
```

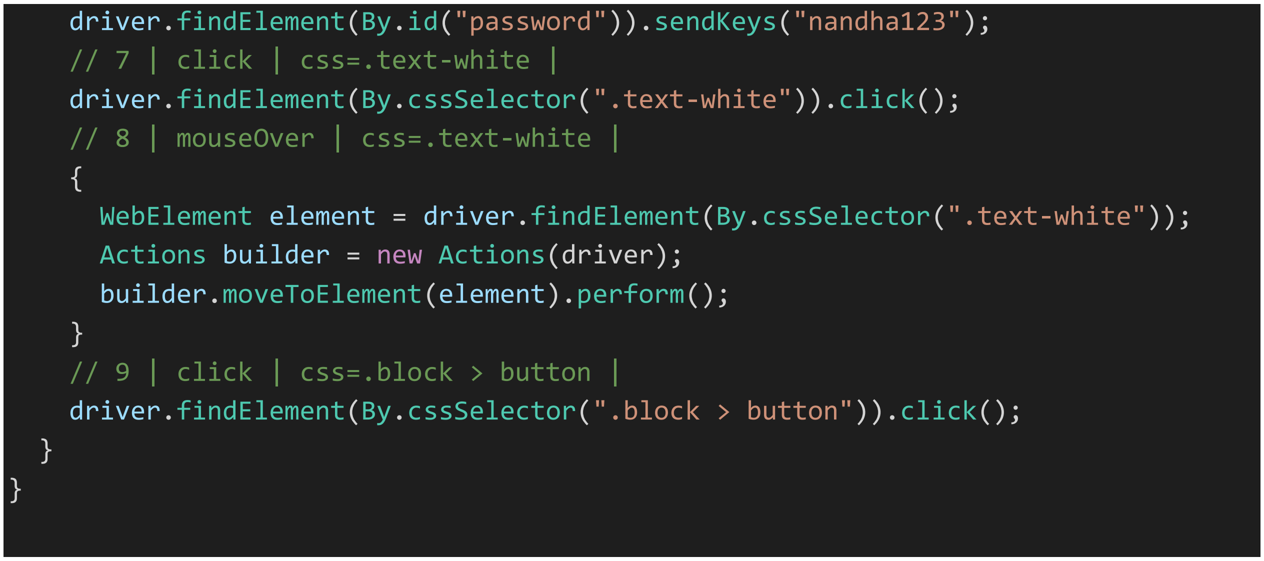
```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class RequestUpdateTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void requestUpdate() {
    // Test name: RequestUpdate
    // Step # | name | target | value
```



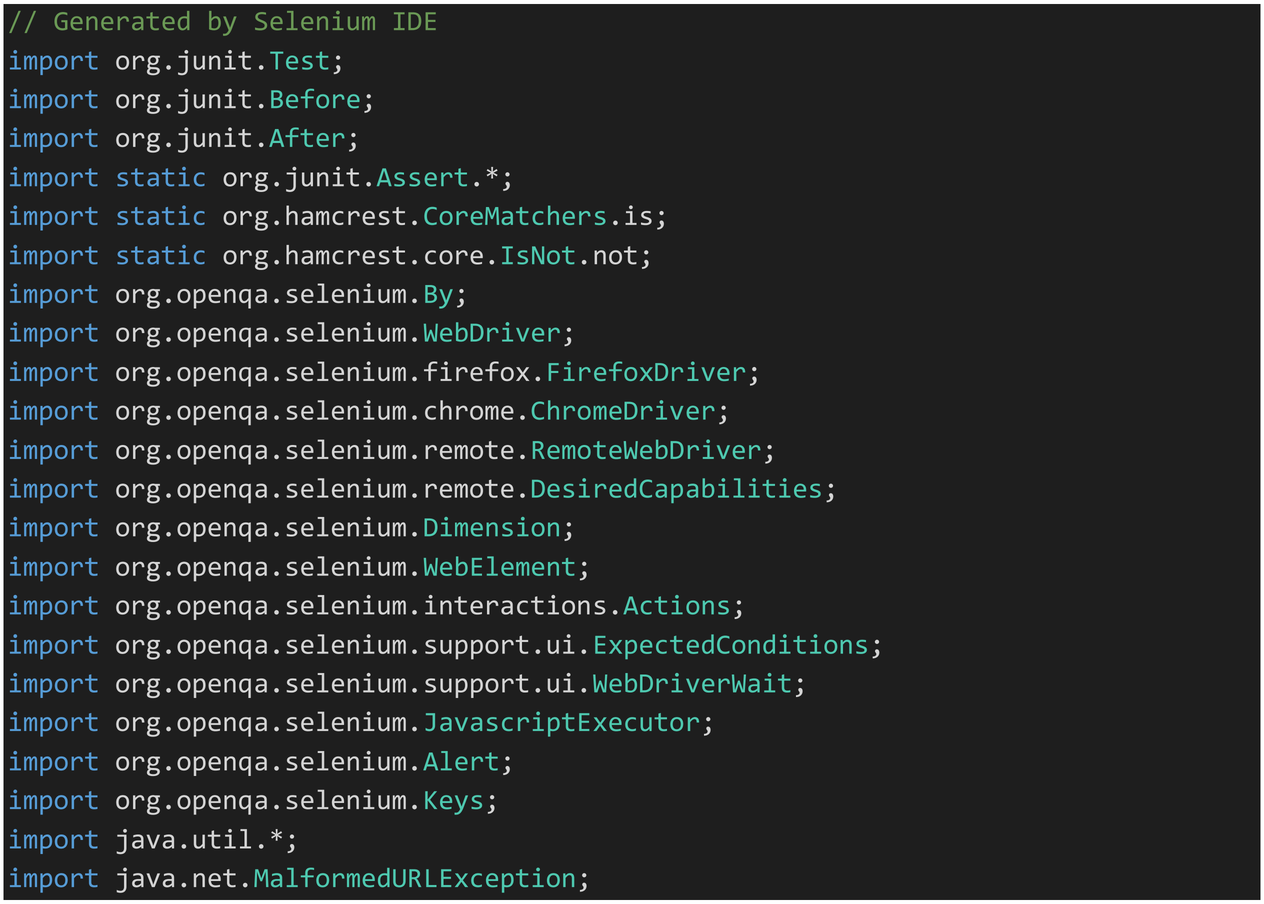
```
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 788x816 |
    driver.manage().window().setSize(new Dimension(788, 816));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | nandha
    driver.findElement(By.id("username")).sendKeys("nandha");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | nandha123
    driver.findElement(By.id("password")).sendKeys("nandha123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
    // 8 | click | css=.block > button |
    driver.findElement(By.cssSelector(".block > button")).click();
    // 9 | click | css=.sc-dAlxHm:nth-child(2) a > .sc-cwHqhk |
    driver.findElement(By.cssSelector(".sc-dAlxHm:nth-child(2) a > .sc-
cwHqhk")).click();
    // 10 | click | id=status |
    driver.findElement(By.id("status")).click();
    // 11 | type | id=status | Approved
    driver.findElement(By.id("status")).sendKeys("Approved");
    // 12 | click | css=.btn |
    driver.findElement(By.cssSelector(".btn")).click();
  }
}
```



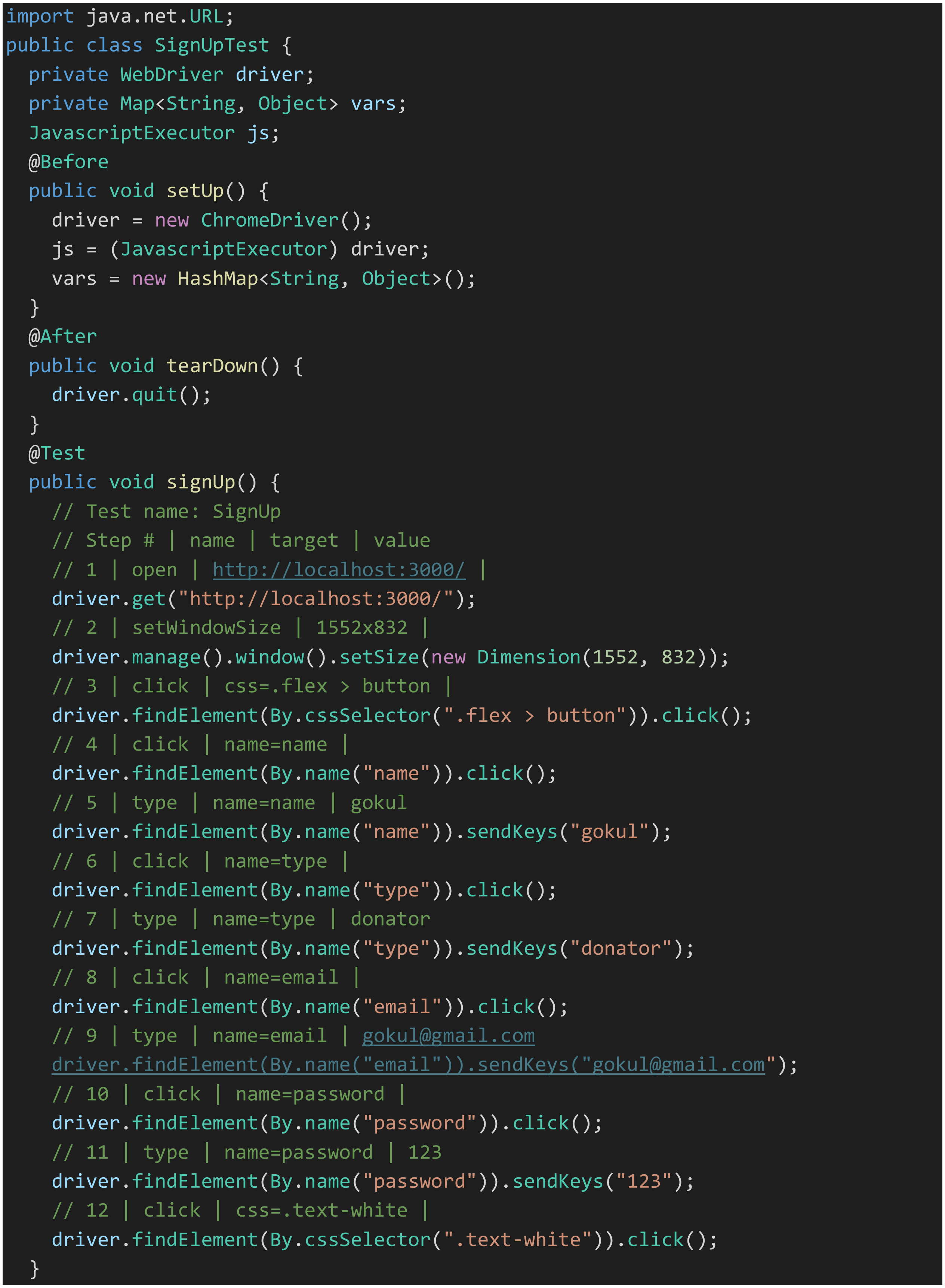
```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
import java.net.URL;
public class RequestViewTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void requestView() {
    // Test name: RequestView
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/login |
    driver.get("http://localhost:3000/login");
    // 2 | setWindowSize | 790x816 |
    driver.manage().window().setSize(new Dimension(790, 816));
    // 3 | click | id=username |
    driver.findElement(By.id("username")).click();
    // 4 | type | id=username | nandha
    driver.findElement(By.id("username")).sendKeys("nandha");
    // 5 | click | id=password |
    driver.findElement(By.id("password")).click();
    // 6 | type | id=password | nandha123
```



```
    driver.findElement(By.id("password")).sendKeys("nandha123");
    // 7 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
    // 8 | mouseOver | css=.text-white |
    {
      WebElement element = driver.findElement(By.cssSelector(".text-white"));
      Actions builder = new Actions(driver);
      builder.moveToElement(element).perform();
    }
    // 9 | click | css=.block > button |
    driver.findElement(By.cssSelector(".block > button")).click();
  }
}
```

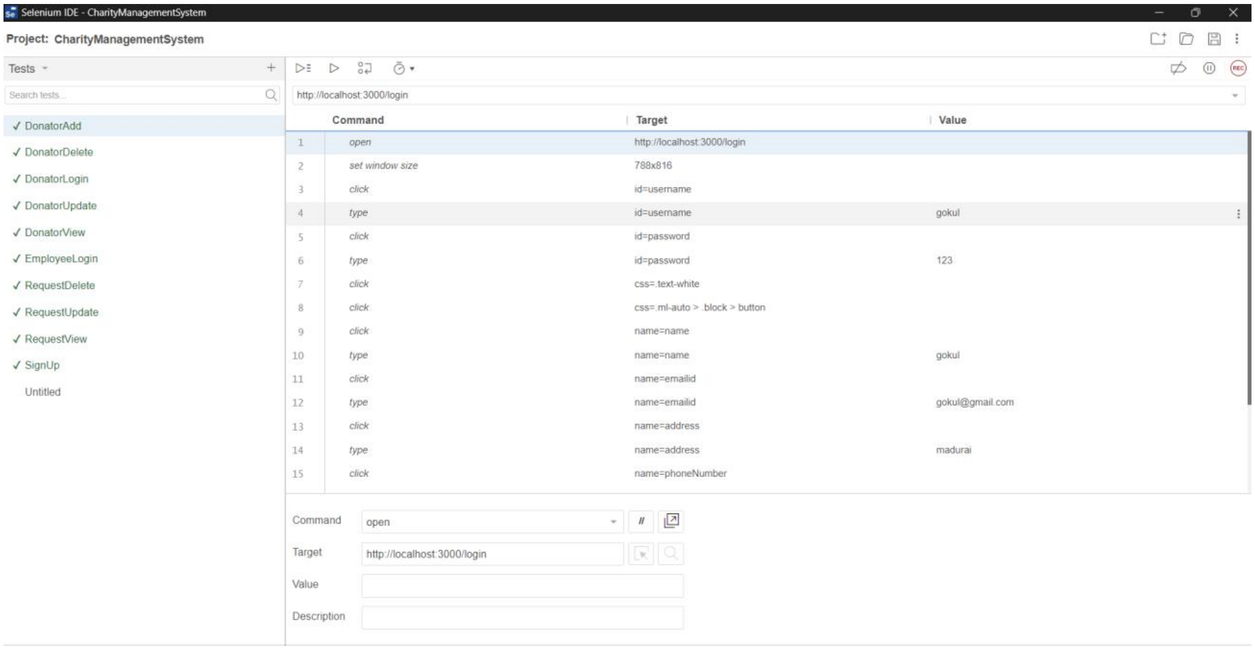


```
// Generated by Selenium IDE
import org.junit.Test;
import org.junit.Before;
import org.junit.After;
import static org.junit.Assert.*;
import static org.hamcrest.CoreMatchers.is;
import static org.hamcrest.core.IsNot.not;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.remote.RemoteWebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.Dimension;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.interactions.Actions;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.Alert;
import org.openqa.selenium.Keys;
import java.util.*;
import java.net.MalformedURLException;
```



```java
import java.net.URL;
public class SignUpTest {
  private WebDriver driver;
  private Map<String, Object> vars;
  JavascriptExecutor js;
  @Before
  public void setUp() {
    driver = new ChromeDriver();
    js = (JavascriptExecutor) driver;
    vars = new HashMap<String, Object>();
  }
  @After
  public void tearDown() {
    driver.quit();
  }
  @Test
  public void signUp() {
    // Test name: SignUp
    // Step # | name | target | value
    // 1 | open | http://localhost:3000/ |
    driver.get("http://localhost:3000/");
    // 2 | setWindowSize | 1552x832 |
    driver.manage().window().setSize(new Dimension(1552, 832));
    // 3 | click | css=.flex > button |
    driver.findElement(By.cssSelector(".flex > button")).click();
    // 4 | click | name=name |
    driver.findElement(By.name("name")).click();
    // 5 | type | name=name | gokul
    driver.findElement(By.name("name")).sendKeys("gokul");
    // 6 | click | name=type |
    driver.findElement(By.name("type")).click();
    // 7 | type | name=type | donator
    driver.findElement(By.name("type")).sendKeys("donator");
    // 8 | click | name=email |
    driver.findElement(By.name("email")).click();
    // 9 | type | name=email | gokul@gmail.com
    driver.findElement(By.name("email")).sendKeys("gokul@gmail.com");
    // 10 | click | name=password |
    driver.findElement(By.name("password")).click();
    // 11 | type | name=password | 123
    driver.findElement(By.name("password")).sendKeys("123");
    // 12 | click | css=.text-white |
    driver.findElement(By.cssSelector(".text-white")).click();
  }
}
```

ScreenShots:



http://localhost:3000/login

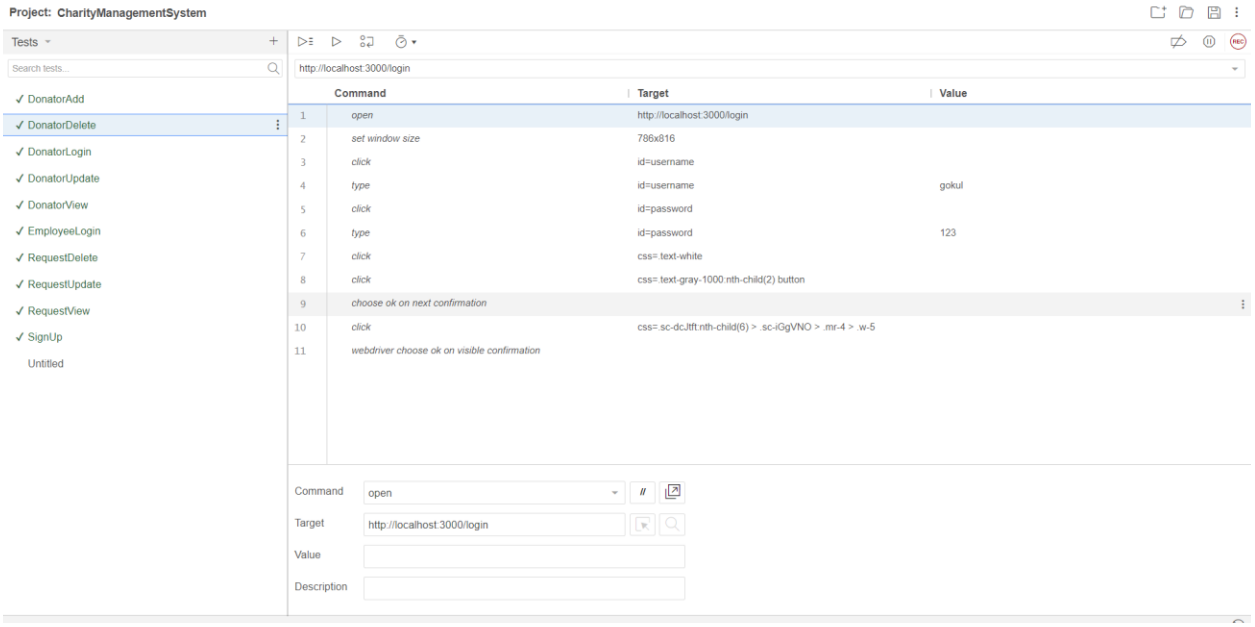
	Command	Target	Value
1	*open*	http://localhost:3000/login	
2	*set window size*	788x816	
3	*click*	id=username	
4	*type*	id=username	gokul
5	*click*	id=password	
6	*type*	id=password	123
7	*click*	css=.text-white	
8	*click*	css=.ml-auto > .block > button	
9	*click*	name=name	
10	*type*	name=name	gokul
11	*click*	name=emailid	
12	*type*	name=emailid	gokul@gmail.com
13	*click*	name=address	
14	*type*	name=address	madurai
15	*click*	name=phoneNumber	

Command: open

Target: http://localhost:3000/login

Value:

Description:



http://localhost:3000/login

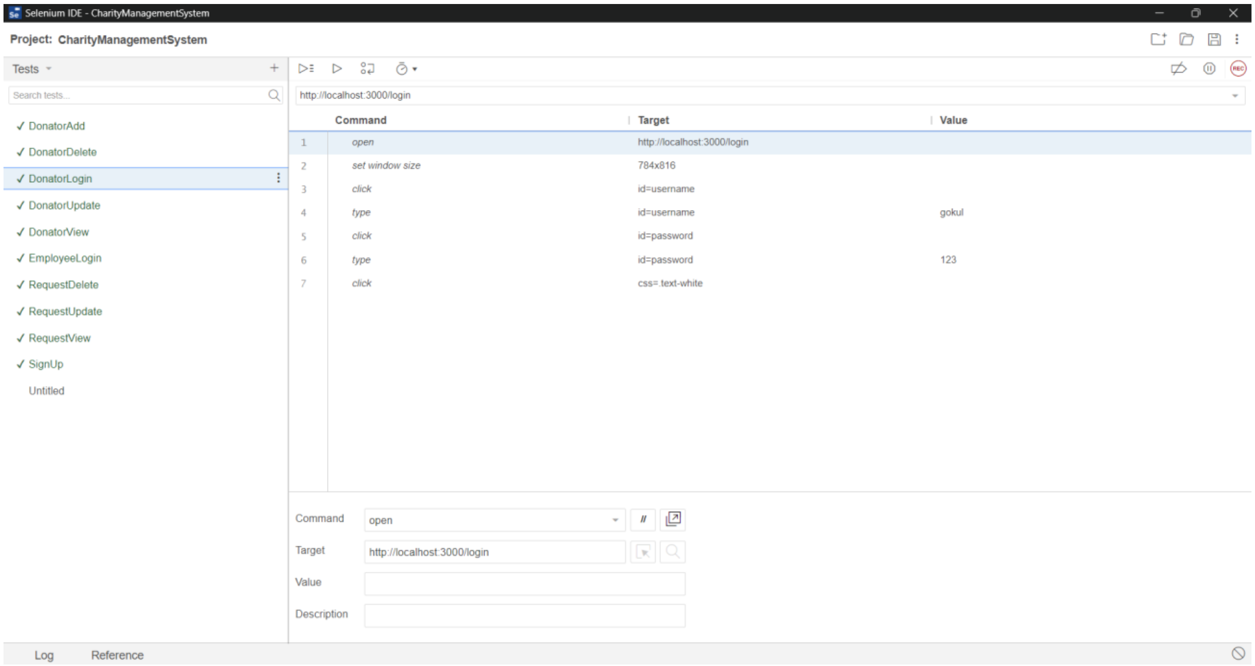
	Command	Target	Value
1	*open*	http://localhost:3000/login	
2	*set window size*	786x816	
3	*click*	id=username	
4	*type*	id=username	gokul
5	*click*	id=password	
6	*type*	id=password	123
7	*click*	css=.text-white	
8	*click*	css=.text-gray-1000:nth-child(2) button	
9	*choose ok on next confirmation*		
10	*click*	css=.sc-dcJtft:nth-child(6) > .sc-iGgVNO > .mr-4 > .w-5	
11	*webdriver choose ok on visible confirmation*		

Command: open

Target: http://localhost:3000/login

Value:

Description:



http://localhost:3000/login

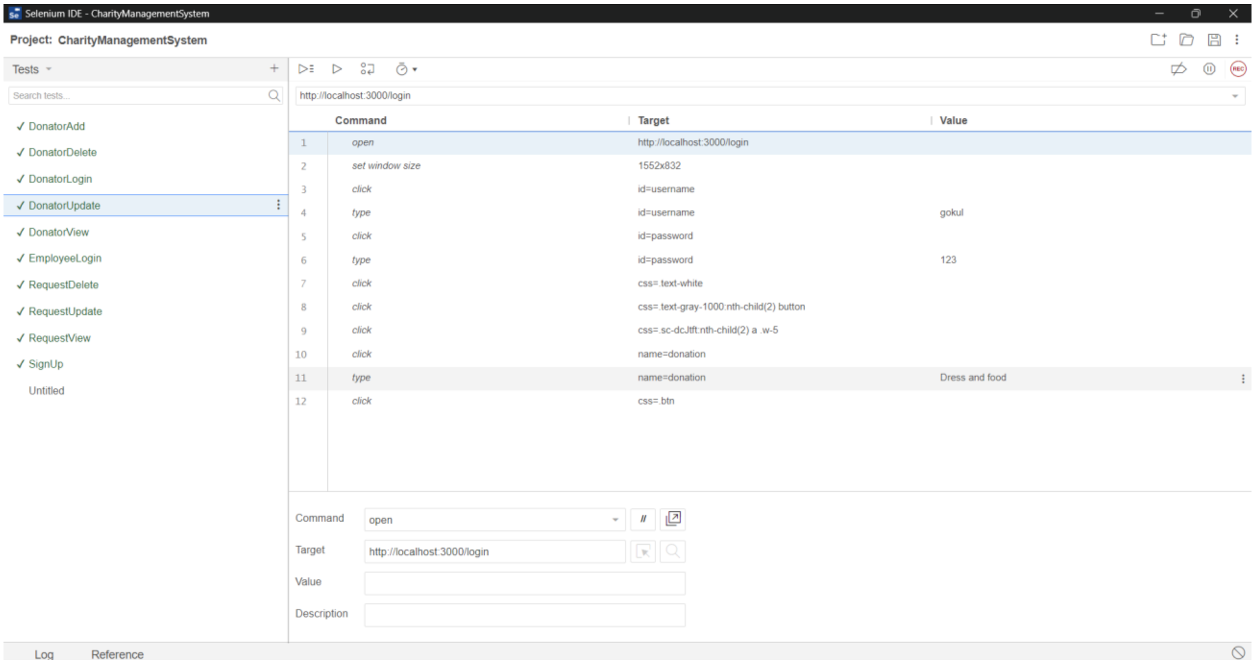
	Command	Target	Value
1	*open*	http://localhost:3000/login	
2	*set window size*	784x816	
3	*click*	id=username	
4	*type*	id=username	gokul
5	*click*	id=password	
6	*type*	id=password	123
7	*click*	css=.text-white	

Command: open

Target: http://localhost:3000/login

Value:

Description:



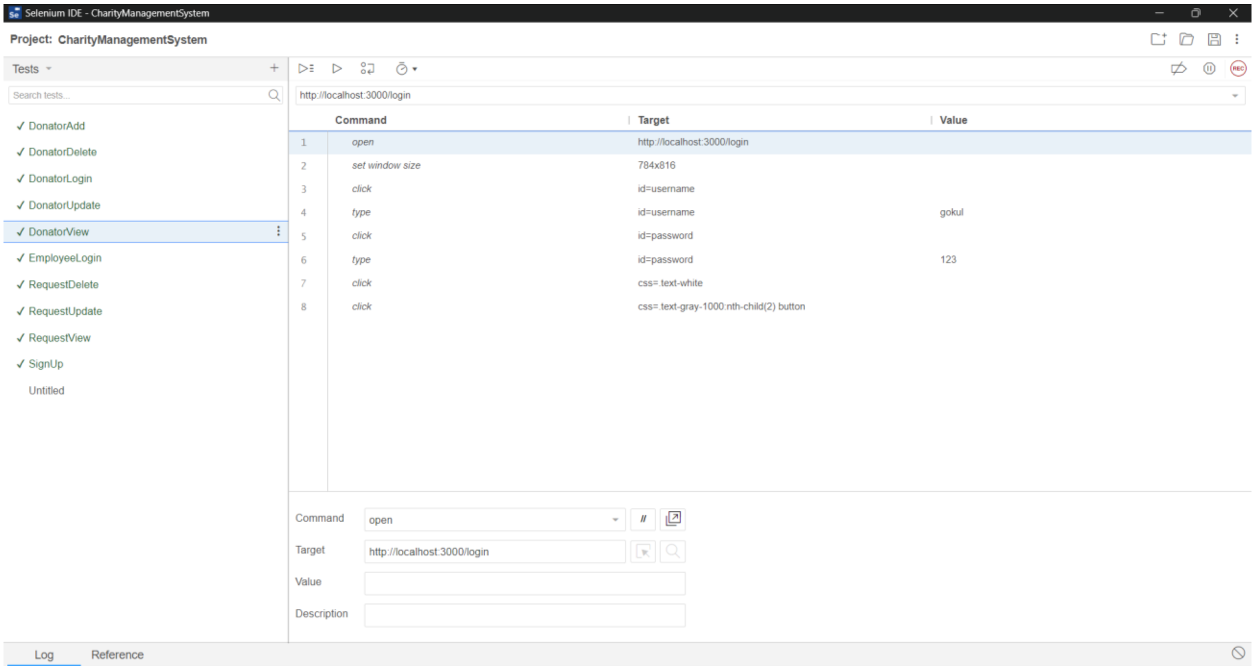
Project: CharityManagementSystem

✓ DonatorAdd
✓ DonatorDelete
✓ DonatorLogin
✓ DonatorUpdate
✓ DonatorView
✓ EmployeeLogin
✓ RequestDelete
✓ RequestUpdate
✓ RequestView
✓ SignUp
Untitled

http://localhost:3000/login

	Command	Target	Value
1	*open*	http://localhost:3000/login	
2	*set window size*	1552x832	
3	*click*	id=username	
4	*type*	id=username	gokul
5	*click*	id=password	
6	*type*	id=password	123
7	*click*	css=.text-white	
8	*click*	css=.text-gray-1000:nth-child(2) button	
9	*click*	css=.sc-dcJtft:nth-child(2) a .w-5	
10	*click*	name=donation	
11	*type*	name=donation	Dress and food
12	*click*	css=.btn	

Command: open
Target: http://localhost:3000/login
Value:
Description:



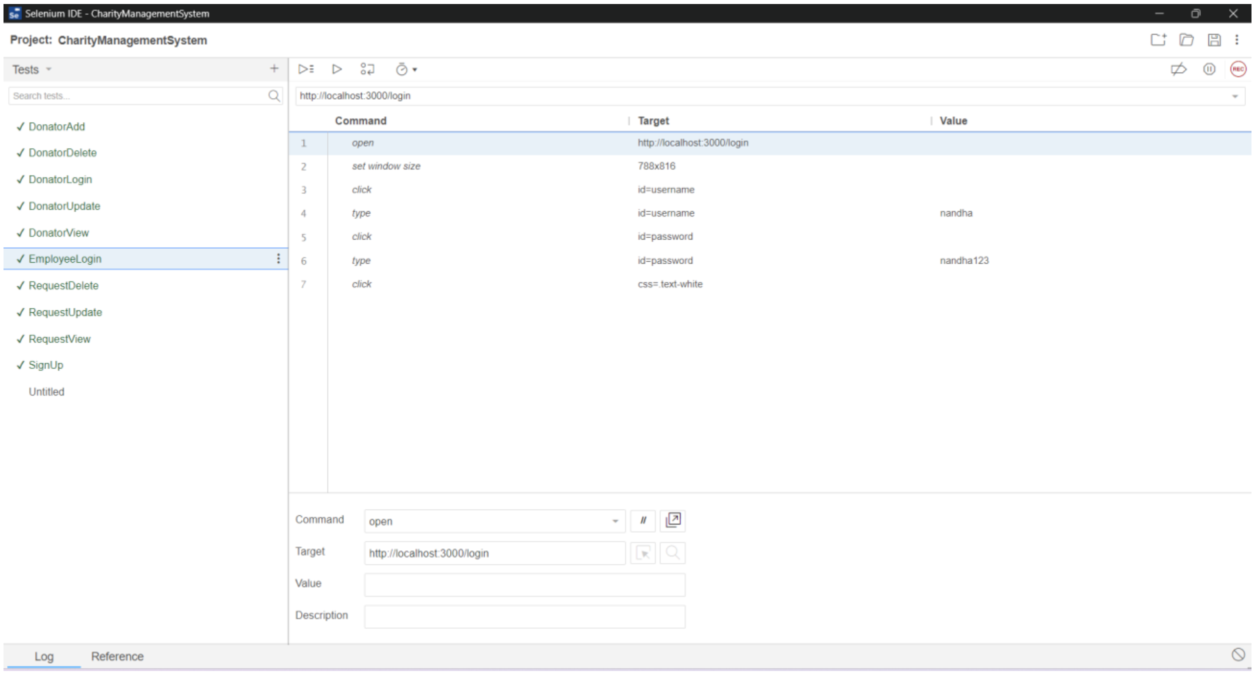
Project: CharityManagementSystem

✓ DonatorAdd
✓ DonatorDelete
✓ DonatorLogin
✓ DonatorUpdate
✓ DonatorView
✓ EmployeeLogin
✓ RequestDelete
✓ RequestUpdate
✓ RequestView
✓ SignUp
Untitled

http://localhost:3000/login

	Command	Target	Value
1	*open*	http://localhost:3000/login	
2	*set window size*	784x816	
3	*click*	id=username	
4	*type*	id=username	gokul
5	*click*	id=password	
6	*type*	id=password	123
7	*click*	css=.text-white	
8	*click*	css=.text-gray-1000:nth-child(2) button	

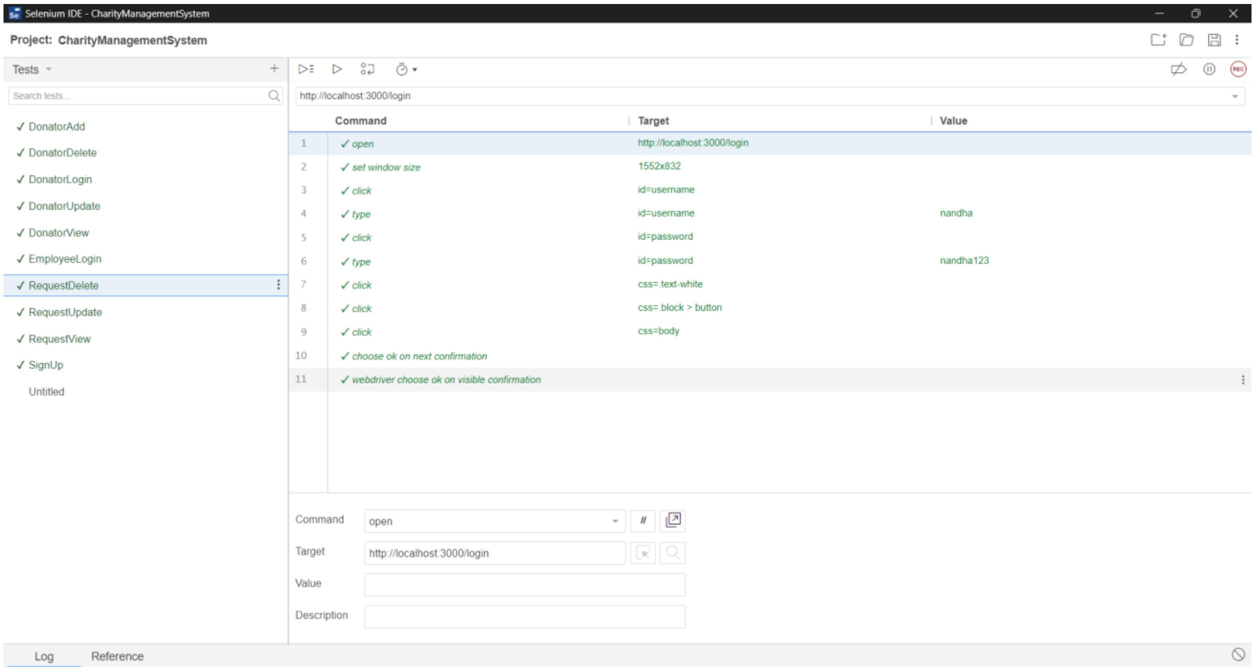
Command: open
Target: http://localhost:3000/login
Value:
Description:



http://localhost:3000/login

	Command	Target	Value
1	*open*	http://localhost:3000/login	
2	*set window size*	788x816	
3	*click*	id=username	
4	*type*	id=username	nandha
5	*click*	id=password	
6	*type*	id=password	nandha123
7	*click*	css=.text-white	

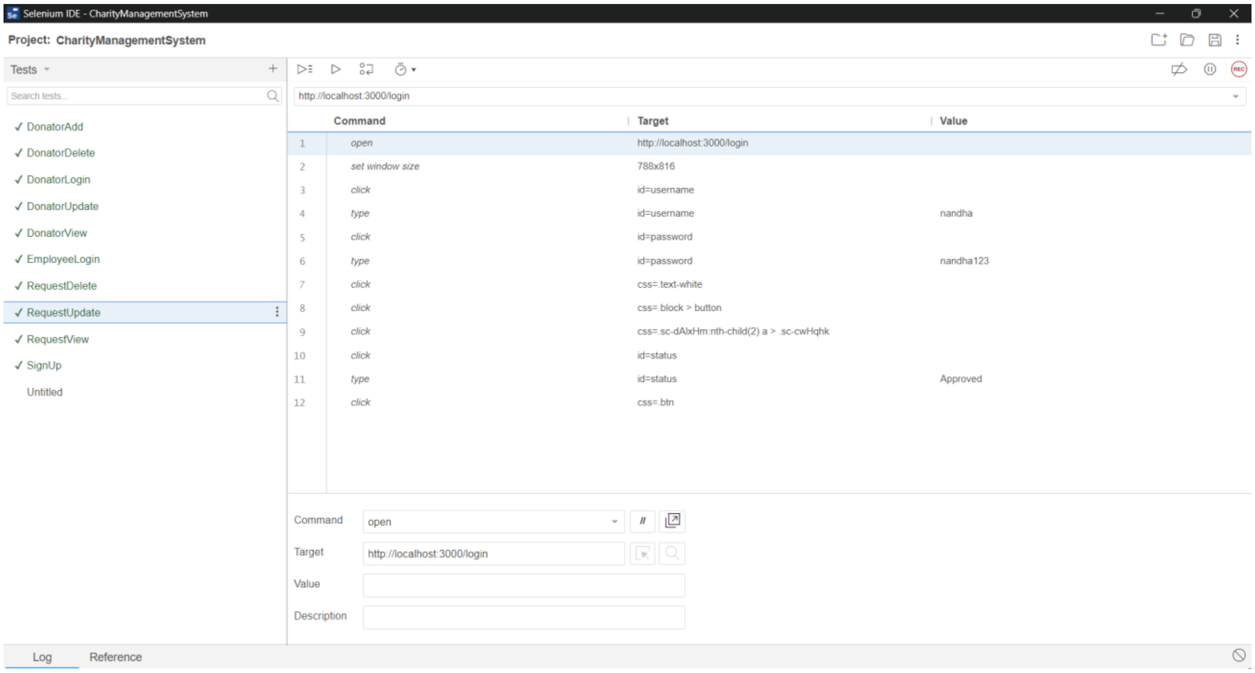
Command: open
Target: http://localhost:3000/login
Value:
Description:



http://localhost:3000/login

	Command	Target	Value
1	✓ *open*	http://localhost:3000/login	
2	✓ *set window size*	1552x832	
3	✓ *click*	id=username	
4	✓ *type*	id=username	nandha
5	✓ *click*	id=password	
6	✓ *type*	id=password	nandha123
7	✓ *click*	css=.text-white	
8	✓ *click*	css=.block > button	
9	✓ *click*	css=body	
10	✓ *choose ok on next confirmation*		
11	✓ *webdriver choose ok on visible confirmation*		

Command: open
Target: http://localhost:3000/login
Value:
Description:



Selenium IDE - CharityManagementSystem

Project: CharityManagementSystem

Tests

- ✓ DonatorAdd
- ✓ DonatorDelete
- ✓ DonatorLogin
- ✓ DonatorUpdate
- ✓ DonatorView
- ✓ EmployeeLogin
- ✓ RequestDelete
- ✓ RequestUpdate
- ✓ RequestView
- ✓ SignUp
- Untitled

http://localhost:3000/login

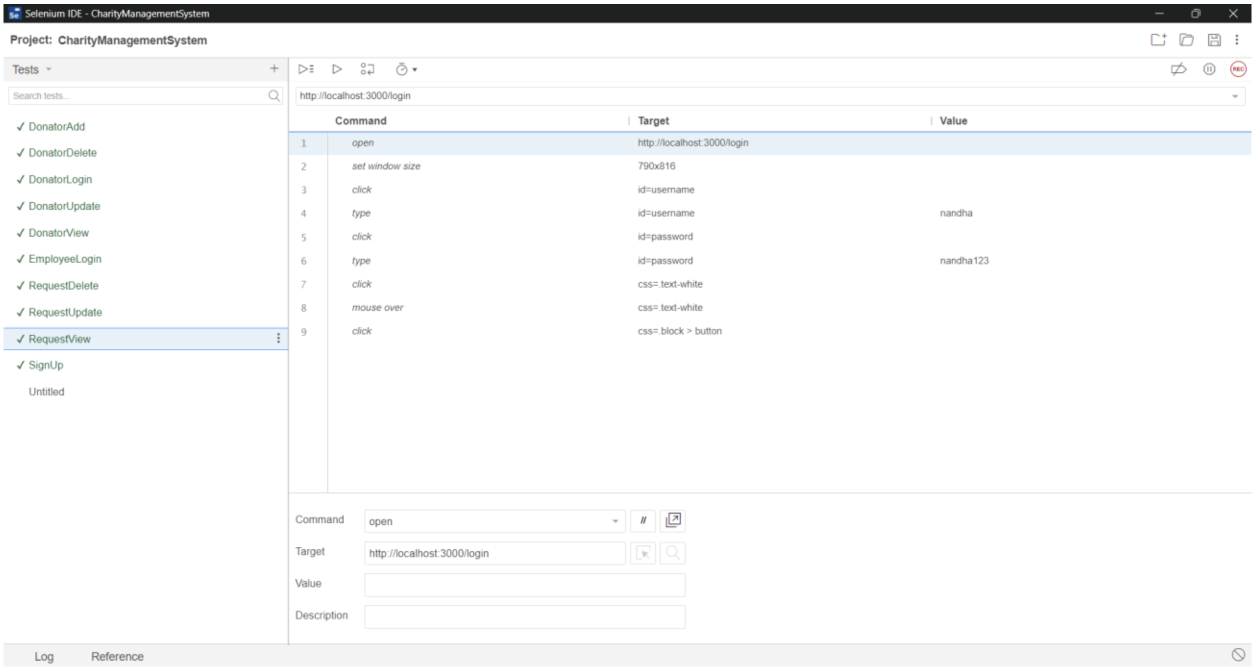
	Command	Target	Value
1	open	http://localhost:3000/login	
2	set window size	788x816	
3	click	id=username	
4	type	id=username	nandha
5	click	id=password	
6	type	id=password	nandha123
7	click	css=.text-white	
8	click	css=.block > button	
9	click	css=.sc-dAlxHm:nth-child(2) a > .sc-cwHqhk	
10	click	id=status	
11	type	id=status	Approved
12	click	css=.btn	

Command: open

Target: http://localhost:3000/login

Value:

Description:



Selenium IDE - CharityManagementSystem

Project: CharityManagementSystem

Tests

- ✓ DonatorAdd
- ✓ DonatorDelete
- ✓ DonatorLogin
- ✓ DonatorUpdate
- ✓ DonatorView
- ✓ EmployeeLogin
- ✓ RequestDelete
- ✓ RequestUpdate
- ✓ RequestView
- ✓ SignUp
- Untitled

http://localhost:3000/login

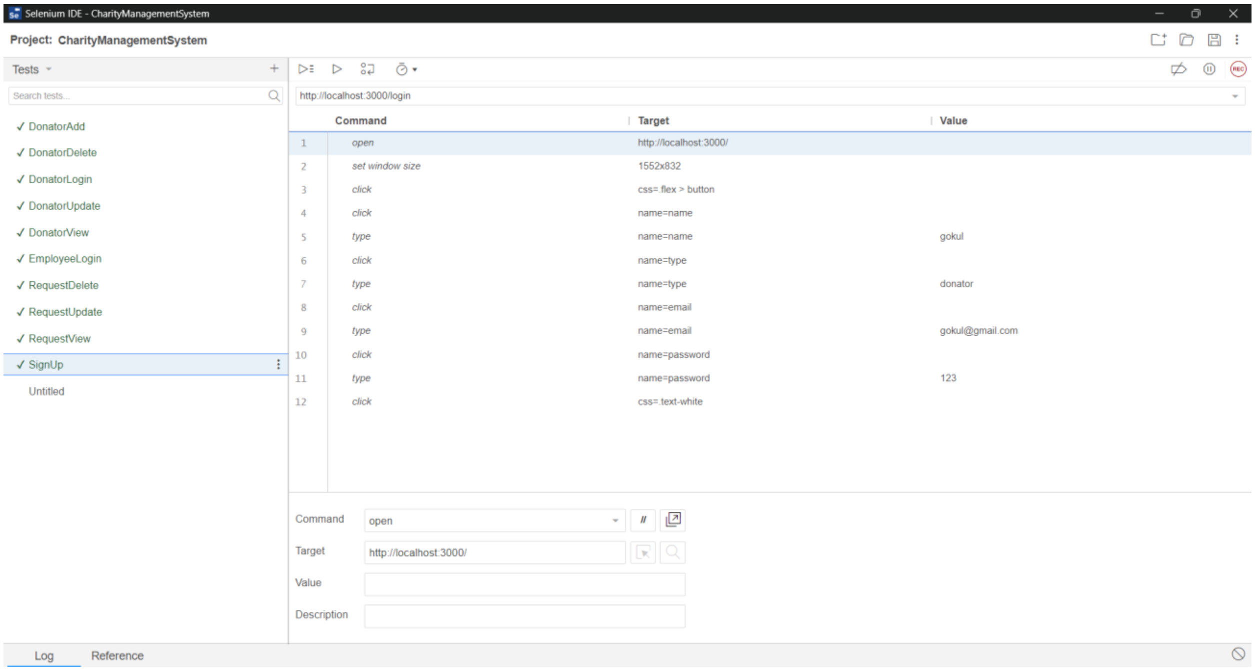
	Command	Target	Value
1	open	http://localhost:3000/login	
2	set window size	790x816	
3	click	id=username	
4	type	id=username	nandha
5	click	id=password	
6	type	id=password	nandha123
7	click	css=.text-white	
8	mouse over	css=.text-white	
9	click	css=.block > button	

Command: open

Target: http://localhost:3000/login

Value:

Description:



Project: CharityManagementSystem

- ✓ DonatorAdd
- ✓ DonatorDelete
- ✓ DonatorLogin
- ✓ DonatorUpdate
- ✓ DonatorView
- ✓ EmployeeLogin
- ✓ RequestDelete
- ✓ RequestUpdate
- ✓ RequestView
- ✓ SignUp
- Untitled

http://localhost:3000/login

	Command	Target	Value
1	*open*	http://localhost:3000/	
2	*set window size*	1552x832	
3	*click*	css=.flex > button	
4	*click*	name=name	
5	*type*	name=name	gokul
6	*click*	name=type	
7	*type*	name=type	donator
8	*click*	name=email	
9	*type*	name=email	gokul@gmail.com
10	*click*	name=password	
11	*type*	name=password	123
12	*click*	css=.text-white	

Command: open

Target: http://localhost:3000/

Value:

Description: