

IT-314 Software Engineering

GUI Testing



Project Title: QuestAI

Group 12

1. Initial Authentication Pages - Script-Based Tests with Assertions

The following pages were tested using manual scripting + assertions, because they required precise validation:

- **Sign-Up & OTP**
- **Successful Login**
- **Forgot Password_Login_ResetPassword**

For these pages:

We used assertElementPresent, assertEquals, and toast checks
We verified input fields, password fields, and page transitions
Selenium steps were written manually (not recorded)
Assertions were necessary to confirm correctness

These tests were written first, because OTP-related flows cannot be captured reliably through recording.

2. Remaining Application Pages — Recording Method

Once the login-related pages were finished, all normal in-app navigation pages were tested using the Selenium IDE record & playback feature as shown and explained in the demo video:

- **Navbar (Logged Out)**
- **Navbar (Logged In)**
- **About**
- **Edit Profile**
- **Create Story Page & Chatbox**
- **Public Stories Page**
- **Home Page(manually)**

For these pages:

Only clicks + navigation flow

Goal: validate user journey

Everything created through recording for natural flow testing

Testing of each page is shown below.

Sign Up and OTP page

The screenshot shows the Selenium IDE interface with the following details:

- Project:** SE_QuestAI_GUI*
- Test:** Sign_up&OTP*
- Playback base URL:** https://quest-ai-frontend.vercel.app/
- Test Steps:**

Command	Target	Value
✓ execute script	window.localStorage.clear();	
✓ open	https://quest-ai-frontend.vercel.app/	
✓ wait for element visible	id=name	30000
✓ type	id=name	Tester
✓ assert value	id=name	Tester
- Log:**
 - open https://quest-ai-frontend.vercel.app/ OK
 - waitForElementVisible on id=name with value 30000 OK
 - type on id=name with value Tester OK
 - assertValue on id=name with value Tester OK
 - type on id=email with value 202301119@dau.ac.in OK
 - assertValue on id=email with value 202301119@dau.ac.in OK
 - type on id=password with value 123456 OK

The screenshot displays two separate test runs in the Selenium IDE interface, both titled "Project: SE_QuestAI_GUI".

Test Run 1 (Top):

- Test Name:** ✓ Sign_up&OTP*
- Playback base URL:** (not explicitly shown)
- Log:**
 - 9. assertValue on id=password with value 123456 OK
 - 10. click on xpath=//button[contains(text(),'SHOW') or contains(text(),'HIDE')] OK
 - 11. click on xpath=//button[contains(text(),'SIGN-UP')] OK
 - 12. Trying to find css=.toast... OK
 - echo: OTP sent to your email! Redirecting
 - 14. waitForElementVisible on xpath=/h2[contains(text(),'VERIFY OTP')] with value 30000 OK
- Summary:** 'Sign_up&OTP' completed successfully

Test Run 2 (Bottom):

- Test Name:** ✓ Sign_up&OTP*
- Playback base URL:** (not explicitly shown)
- Log:**
 - 9. assertValue on id=password with value 123456 OK
 - 10. click on xpath=//button[contains(text(),'SHOW') or contains(text(),'HIDE')] OK
 - 11. click on xpath=//button[contains(text(),'SIGN-UP')] OK
 - 12. Trying to find css=.toast... OK
 - echo: OTP sent to your email! Redirecting
 - 14. waitForElementVisible on xpath=/h2[contains(text(),'VERIFY OTP')] with value 30000 OK
- Summary:** 'Sign_up&OTP' completed successfully

The test confirmed that the Sign-Up page loads properly, takes all the inputs properly, and successfully sends the OTP on the provided email. A toast appears, and the app correctly redirects to the OTP page, so the Sign-Up - OTP flow works as expected.

Since Selenium IDE can't read real email OTPs, the OTP submission and "Resend OTP" features were tested manually. These worked fine, and after entering the correct OTP, the sign up was successful.

Also, Google Sign-In can't be automated because Google blocks automated OAuth popups, so this feature was manually tested as well.

Successful Login

The screenshot displays two separate instances of the Selenium IDE interface, both titled "Extension: (Selenium IDE) - Selenium IDE - SE_QuestAI_GUI — Mozilla Firefox".

Test Case 1 (Top): This test case, named "Successful_Sign_in", contains the following steps:

Command	Target	Value
✓ open	https://quest-al-frontend.vercel.app/Login	
✓ wait for element visible	id=email	30000
✓ assert element present	id=password	
✓ type	id=email	202301119@dau.ac.in
✓ assert value	id=email	202301119@dau.ac.in

Test Case 2 (Bottom): This test case, also named "Successful_Sign_in", contains the following steps:

Command	Target	Value
✓ click	xpath=/button[contains(text(),'SHOW') or contains(text(),'HIDE')]	
✓ click	xpath=/form//button[contains(text(),'LOGIN')]	
✓ wait for element present	css=.toast	30000
✓ store text	css=.toast	toastText
✓ echo		\$(toastText)

Both instances show a log of executed commands at the bottom, with timestamps from 15:57:02 to 15:57:05. The logs indicate successful execution of each step.

The screenshot shows the Selenium IDE interface with the following details:

- Project:** SE_QuestAI_GUI
- Test:** Successful_Sign_in
- URL:** https://quest-ai-frontend.vercel.app/
- Test Steps:**

Command	Target	Value
12. ✓ wait for element present	xpath=/h2	30000
13. ✓ execute script	return window.localStorage.getItem("accessToken");	accessToken
14. ✓ echo		\${accessToken}
- Log:**

```

echo: ✓ Sign in successful!                                         15:57:05
12. waitForElementPresent on xpath://h2 with value 30000 OK        15:57:05
13. executeScript on return window.localStorage.getItem("accessToken"); with value accessToken OK      15:57:06
echo: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJfaWQiOiI2OTI0MjcZDExMmE5M2VmYzJlMml5NmYjLCJibWFpbCI6IjwMjMwMTExOUBkYXUuYWMuaw4iLCJ1c2VybmtZS16IIRlc3RiclsImhdCl6MTc2Mzk
'Successful_Sign_in' completed successfully                         15:57:06

```

This test checks that the Login page loads correctly and that both the email and password fields are present and can accept the correct input. After entering the correct credentials, the test clicks the Login button and waits for the success toast message. It then confirms the login and the user has successfully logged in.

ForgotPassword_Login_ResetPassword

The screenshot shows the Selenium IDE interface with the following details:

- Project:** SE_QuestAI_GUI
- Test:** ForgotPassword_Login_ResetPassword
- URL:** https://quest-ai-frontend.vercel.app/
- Test Steps:**

Command	Target	Value
1. ✓ execute script	window.localStorage.clear();	
2. ✓ open	https://quest-ai-frontend.vercel.app/Login	
3. ✓ wait for element visible	xpath=/button[contains(text(),'Forgot password')]	30000
4. ✓ click	xpath=/button[contains(text(),'Forgot password')]	
- Log:**

```

Running 'ForgotPassword_Login_ResetPassword'
1. executeScript on window.localStorage.clear(); OK                16:52:28
2. open on https://quest-ai-frontend.vercel.app/Login OK        16:52:29
3. waitForElementVisible on xpath=/button[contains(text(),'Forgot password')] with value 30000 OK 16:52:29
4. click on xpath=/button[contains(text(),'Forgot password')] OK   16:52:29
5. waitForElementVisible on id=email with value 30000 OK       16:52:30

```

Extension: (Selenium IDE) - Selenium IDE - SE_QuestAI_GUI — Mozilla Firefox

Project: SE_QuestAI_GUI

Tests + ✓ ForgetPassword_Login_ResetPassword ✓ Sign_up&OTP ✓ Successful_Login

Search tests... ✓

https://quest-ai-frontend.vercel.app/ ✓

Command	Target	Value
4 ✓ click	xpath=//button[contains(text(),'Forgot password')]	
5 ✓ wait for element visible	id=email	30000
6 ✓ assert element present	id=email	
7 ✓ type	id=email	202301119@dau.ac.in

Command ✓ // ✓ ✓

Target ✓ ✓ ✓

Value ✓

Description ✓

Log Reference

```

7. type on id=email with value 202301119@dau.ac.in OK
16:52:31
8. assertValue on id=email with value 202301119@dau.ac.in OK
16:52:31
9. click on xpath=/button[contains(.,'SEND OTP') or contains(text(),[' SEND OTP '])] OK
16:52:31
10. waitForElementPresent on css=.toast with value 30000 OK
16:52:31
11. storeText on css=.toast with value fpToast OK
16:52:34
echo: ✓ OTP sent to your email!
16:52:34
    
```

Extension: (Selenium IDE) - Selenium IDE - SE_QuestAI_GUI — Mozilla Firefox

Project: SE_QuestAI_GUI

Tests + ✓ ForgetPassword_Login_ResetPassword ✓ Sign_up&OTP ✓ Successful_Login

Search tests... ✓

https://quest-ai-frontend.vercel.app/ ✓

Command	Target	Value
10 ✓ wait for element present	aids(text(),[' SEND OTP '])	
11 ✓ store text	css=.toast	30000
12 ✓ echo	css=.toast	fpToast
13 ✓ wait for element visible	xpath=/h2[contains(text(),'VERIFY OTP')]	30000

Command ✓ // ✓ ✓

Target ✓ ✓ ✓

Value ✓

Description ✓

Log Reference

```

13. waitForElementVisible on xpath=/h2[contains(text(),'VERIFY OTP')] with value 30000 OK
16:52:34
14. waitForElementVisible on xpath=/h2[contains(text(),'SET NEW PASSWORD')] with value 30000 OK
16:52:34
15. assertElementPresent on id=password OK
16:52:51
16. assertElementPresent on id=confirmPassword OK
16:52:51
17. type on id=password with value 111111 OK
16:52:51
18. assertValue on id=password with value 111111 OK
16:52:51
    
```

The screenshot displays two separate runs of the 'ForgetPassword_Login_ResetPassword' test case within the Selenium IDE interface. Both runs show the following sequence of steps:

- assertValue on id=password with value 111111 OK
- click on xpath=/button[contains(text(),'SET PASSWORD') or contains(text(),'SUBMIT')] OK
- waitForElementVisible on css=.toast with value 30000 OK
- storeText on css=.toast with value resetToast OK
- echo: Password has been reset!
- 'ForgetPassword_Login_ResetPassword' completed successfully

The second run (bottom) includes additional steps:

- type on id=password with value 111111
- assert value on id=password with value 111111
- type on id=confirmPassword with value 111111
- assert value on id=confirmPassword with value 111111
- click on xpath=/button[contains(text(),'SET PASSWORD') or contains(text(),'SUBMIT')]
- wait for element visible on css=.toast with value 30000
- store text on css=.toast with value resetToast
- echo \${resetToast}

This test covers the full Forgot Password flow starting from the Login page. First, it checks that clicking the “Forgot password?” button correctly opens the Forgot Password page and that the email field accepts a valid email. When the backend sends an OTP, the test also verifies the toast message that appears. After that, it confirms that the user is navigated to the OTP page and then automatically redirects to the Reset Password page.

Since Selenium IDE cannot read or enter real OTPs, the OTP step was tested manually. From the Reset Password page onward, Selenium continues the flow by making sure the New Password and Confirm Password fields are visible and working. Finally, it enters matching passwords, submits the form, verifies the success toast, and confirms that the password reset is completed properly.

Navbar_LoggedOut

The screenshot shows the Selenium IDE interface with the following details:

Project: SE_QuestAI_GUI

Test Name: Navbar_LoggedOut

Target URL: https://quest-ai-frontend.vercel.app/

Test Log:

Step	Action	Target	Value	Timestamp
1	✓ open	https://quest-ai-frontend.vercel.app/		21:42:16
2	✓ set window size	1204x680		21:42:16
3	✓ click	css=.hidden > .form-button:nth-child(1)		21:42:16
4	✓ mouse over	css=.hidden > .form-button:nth-child(1)		21:42:17
5	✓ click	css=.hidden > .form-button:nth-child(2)		21:42:17

Log:

```
Running 'Navbar_LoggedOut'
1. open on https://quest-ai-frontend.vercel.app/ OK
2. setWindowSize on 1204x680 OK
3. click on css=.hidden > .form-button:nth-child(1) OK
4. mouseOver on css=.hidden > .form-button:nth-child(1) OK
5. click on css=.hidden > .form-button:nth-child(2) OK
```

Test 2: Run all tests

Test Name: Navbar_LoggedOut

Target URL: https://quest-ai-frontend.vercel.app/

Test Log:

Step	Action	Target	Value	Timestamp
7	✓ mouse over	css=.hidden > .form-button:nth-child(2)		21:42:17
8	✓ mouse out	css=.hidden > .form-button:nth-child(2)		21:42:17
9	✓ click	css=.hidden > .form-button:nth-child(3)		21:42:17
10	✓ mouse over	css=.hidden > .form-button:nth-child(3)		21:42:18
11	✓ mouse over	css=.hidden > .form-button:nth-child(3)		21:42:18

Log:

```
6. mouseOver on css=.hidden > .form-button:nth-child(2) OK
7. mouseOver on css=.hidden > .form-button:nth-child(2) OK
8. mouseOut on css=.hidden > .form-button:nth-child(2) OK
9. click on css=.hidden > .form-button:nth-child(3) OK
10. mouseOver on css=.hidden > .form-button:nth-child(3) OK
11. mouseOver on css=.hidden > .form-button:nth-child(3) OK
12. mouseOut on css=.hidden > .form-button:nth-child(3) OK
```

The screenshot shows the Selenium IDE interface with the following details:

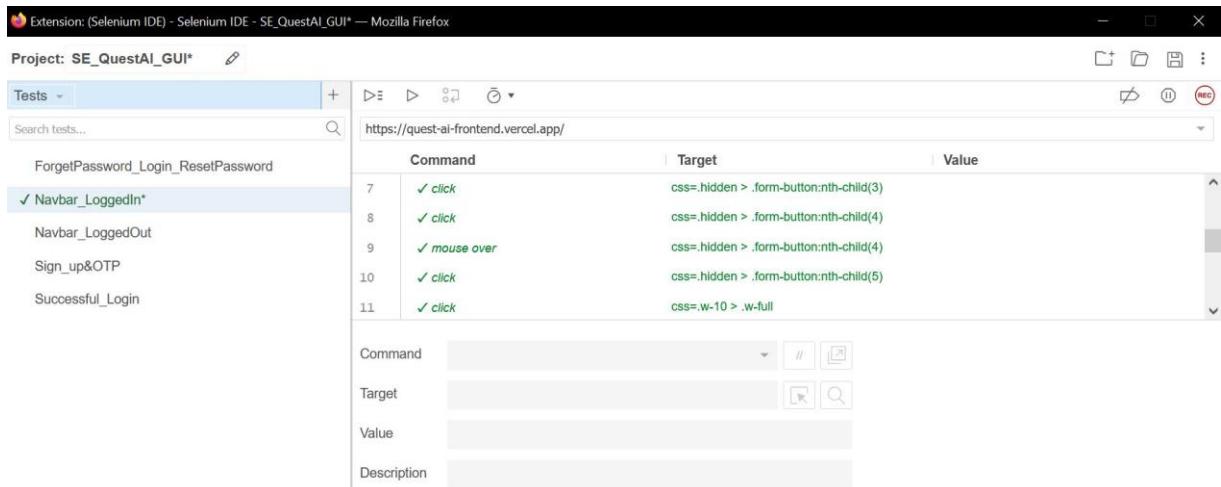
- Project:** SE_QuestAI_GUI
- Tests:** ForgetPassword_Login_ResetPassword, Navbar_LoggedOut, Sign_up&OTP, Successful_Login
- Selected Test:** Navbar_LoggedOut
- Test Log:**
 - 14. click on css=.hidden > .form-button:nth-child(5) OK
 - 15. mouseOver on css=.hidden > .form-button:nth-child(5) OK
 - 16. click on css=.hidden > .form-button:nth-child(5) OK
 - 17. click on css=h-[70px] OK
- Log:** 'Navbar_LoggedOut' completed successfully

This test checked whether all the Navbar buttons work properly when the user isn't logged in. Starting from the Login page, the about, sign up, login and public stories buttons each opened the pages they were supposed to, and all the expected elements loaded fine on those pages. The theme toggle also worked without any issues, and clicking the Quest AI logo brought me back to the Login page as it should. Overall, the navigation flow for a logged-out user works smoothly.

Navbar_LoggedIn

The screenshot shows the Selenium IDE interface with the following details:

- Project:** SE_QuestAI_GUI*
- Tests:** ForgetPassword_Login_ResetPassword, Navbar_LoggedIn*, Navbar_LoggedOut, Sign_up&OTP, Successful_Login
- Selected Test:** Navbar_LoggedIn*
- Test Log:**
 - 1. open on https://quest-ai-frontend.vercel.app/Home/Tester OK
 - 2. setWindowSize on 1204x680 OK
 - 3. click on css=.hidden > .form-button:nth-child(1) OK
 - 4. mouseOver on css=.hidden > .form-button:nth-child(1) OK
 - 5. click on css=.hidden > .form-button:nth-child(2) OK
 - 6. click on css=.hidden > .form-button:nth-child(3) OK



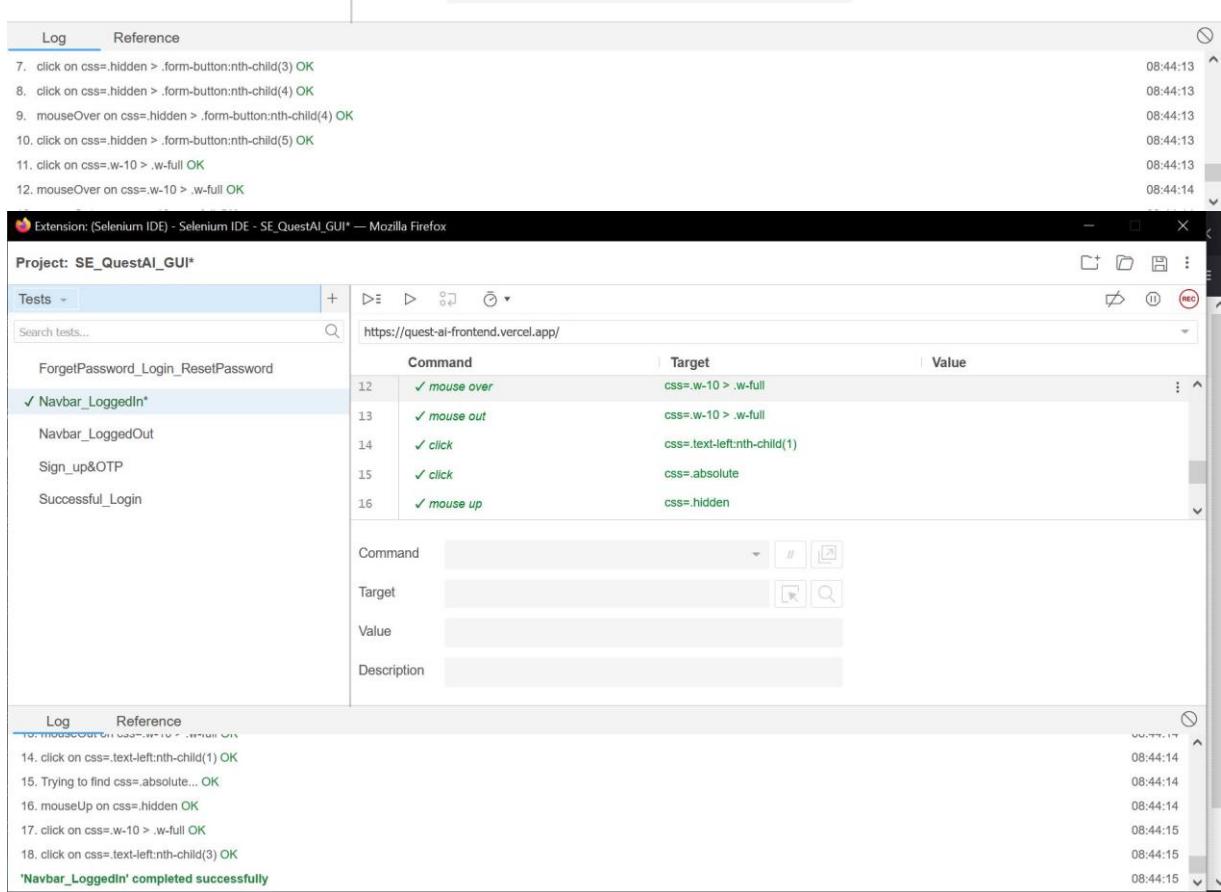
The screenshot shows the Selenium IDE interface with the project "SE_QuestAI_GUI" open. The URL is set to "https://quest-ai-frontend.vercel.app/". The left sidebar lists test cases: "ForgetPassword_Login_ResetPassword", "✓ Navbar_LoggedIn*", "Navbar_LoggedOut", "Sign_up&OTP", and "Successful_Login". The main area displays a table of test steps:

	Command	Target	Value
7	✓ click	css=.hidden > .form-button:nth-child(3)	
8	✓ click	css=.hidden > .form-button:nth-child(4)	
9	✓ mouse over	css=.hidden > .form-button:nth-child(4)	
10	✓ click	css=.hidden > .form-button:nth-child(5)	
11	✓ click	css=.w-10 > .w-full	

Below the table are input fields for "Command", "Target", "Value", and "Description". The "Log" tab at the bottom shows the execution history:

```

7. click on css=.hidden > .form-button:nth-child(3) OK
8. click on css=.hidden > .form-button:nth-child(4) OK
9. mouseOver on css=.hidden > .form-button:nth-child(4) OK
10. click on css=.hidden > .form-button:nth-child(5) OK
11. click on css=.w-10 > .w-full OK
12. mouseOver on css=.w-10 > .w-full OK
    
```



The second screenshot shows the same Selenium IDE interface with the "✓ Navbar_LoggedIn*" test case selected. The log table now includes additional steps:

	Command	Target	Value
12	✓ mouse over	css=.w-10 > .w-full	
13	✓ mouse out	css=.w-10 > .w-full	
14	✓ click	css=.text-left:nth-child(1)	
15	✓ click	css=absolute	
16	✓ mouse up	css=.hidden	

The log table at the bottom shows the execution history with the last step being successful:

```

10. mouseUp on css=.hidden > .w-full OK
14. click on css=.text-left:nth-child(1) OK
15. Trying to find css=absolute... OK
16. mouseUp on css=.hidden OK
17. click on css=.w-10 > .w-full OK
18. click on css=.text-left:nth-child(3) OK
'Navbar_LoggedIn*' completed successfully
    
```

The logged-in navbar was tested using simple click-based Selenium IDE commands. Each navbar item (Home, Create Story, Public Stories, Theme Toggle, Profile Panel, Logout) responded correctly during the test.

Navigation transitions worked smoothly without failures.

Overall, the logged-in navbar flow is functioning as expected.

About Page

The About page was successfully reached through the Navbar, and navigation worked correctly. There are no interactive buttons or inputs on the About page, so no further testing is needed.

Edit Profile Page

The screenshot displays two separate Selenium IDE sessions, each showing a test case for the 'Edit_Profile' feature.

Test Case 1 (Top):

- Project:** SE_QuestAI_GUI*
- URL:** https://quest-ai-frontend.vercel.app/
- Commands:**

Command	Target	Value
✓ open	https://quest-ai-frontend.vercel.app/Home/Tester	
✓ set window size	1204x680	
✓ click	css=.hidden	
✓ click	css=w-10 > .w-full	
✓ mouse over	css=w-10 > .w-full	
- Log:**
 - 'Navbar_Loggedin' completed successfully
 - Running 'Edit_Profile'
 - 1. open on https://quest-ai-frontend.vercel.app/Home/Tester OK
 - 2. setWindowSize on 1204x680 OK
 - 3. click on css=.hidden OK
 - 4. click on css=w-10 > .w-full OK

Test Case 2 (Bottom):

- Project:** SE_QuestAI_GUI*
- URL:** https://quest-ai-frontend.vercel.app/
- Commands:**

Command	Target	Value
✓ click	css=.text-left:nth-child(1)	
✓ click	css=flex-col > .p-2	
✓ type	css=flex-col > .p-2	Tester1
✓ click	css:flex:nth-child(4) > .w-full > .p-2	
✓ type	css:flex:nth-child(4) > .w-full > .p-2	123456
- Log:**
 - 5. mouseOver on css=w-10 > .w-full OK
 - 6. mouseOut on css=w-10 > .w-full OK
 - 7. click on css=.text-left:nth-child(1) OK
 - 8. Trying to find css=flex-col > .p-2... OK
 - 9. type on css=flex-col > .p-2 with value Tester1 OK
 - 10. click on css:flex:nth-child(4) > .w-full > .p-2 OK
 - 11. type on css=flex:nth-child(4) > .w-full > .p-2 with value 123456 OK

The screenshot displays two separate sessions in the Selenium IDE interface, both titled "Project: SE_QuestAI_GUI*".

Session 1 (Top):

- URL: <https://quest-ai-frontend.vercel.app/>
- Commands (Listed in Log):

 - 12. click on css=.right-2 OK
 - 13. click on css=.flex:nth-child(4) > .mt-6 OK
 - 14. mouseOver on css=.flex:nth-child(4) > .mt-6 OK
 - 15. mouseOut on css=.flex:nth-child(4) > .mt-6 OK
 - 16. click on css=.w-10 > .w-full OK
 - 17. mouseOver on css=.w-10 > .w-full OK

Session 2 (Bottom):

- URL: <https://quest-ai-frontend.vercel.app/>
- Commands (Listed in Log):

 - 19. click on css=text-left:nth-child(1) OK
 - 20. click on css=.mt-8 .p-2 OK
 - 21. type on css=.mt-8 .p-2 with value gsk_lYpMoz6M9bIPYpBLJSSKGdyb3FYF6CuKDXKmHjlX0JKaAJeJfnv OK
 - 22. click on css=.mt-8 > .mt-6 OK
 - 23. click on css=.absolute OK

Log:

'Edit_Profile' completed successfully

The Edit Profile flow was tested successfully using Selenium IDE through the recording method.

The profile-picture change feature was also triggered in Selenium, but since Selenium IDE cannot verify image previews or file rendering, the profile-picture update was manually tested and confirmed to work properly.

Overall, all Edit Profile functionalities behaved as expected.

Story form and Chatbox

Extension: (Selenium IDE) - Selenium IDE - SE_QuestAI_GUI* — Mozilla Firefox

Project: SE_QuestAI_GUI*

Tests + Run current test Ctrl+R 3.vercel.app/

	Command	Target	Value
13	✓ type	name=character	Iron Man
14	✓ click	name=genre	
15	✓ send keys	name=genre	\$(KEY_DOWN)
16	✓ type	name=genre	Sci-fi
17	✓ click	css=.mt-6	

✓ Edit_Profile
ForgetPassword_Login_ResetPassword
✓ Navbar_LoggedIn
Navbar_LoggedOut
Sign_up&OTP
✓ Story_Form*
Successful_Login

Log Reference
15. sendKeys on name=genre with value \$(KEY_DOWN) OK 09:34:49
16. type on name=genre with value Sci-fi OK 09:34:49
17. click on css=.mt-6 OK 09:34:49
18. mouseOver on css=.mt-6 OK 09:34:49
19. mouseOut on css=.mt-6 OK 09:34:50
20. click on css=.w-10 > .w-full OK 09:34:50
21. mouseOver on css=.w-10 > .w-full OK 09:34:50

Extension: (Selenium IDE) - Selenium IDE - SE_QuestAI_GUI* — Mozilla Firefox

Project: SE_QuestAI_GUI*

Tests + Run current test Ctrl+R 3.vercel.app/

	Command	Target	Value
17	✓ click	css=.mt-6	
18	✓ mouse over	css=.mt-6	
19	✓ mouse out	css=.mt-6	
20	✓ click	css=.w-10 > .w-full	

✓ Edit_Profile
ForgetPassword_Login_ResetPassword
✓ Navbar_LoggedIn
Navbar_LoggedOut
Sign_up&OTP
✓ Story_Form*
Successful_Login

Log Reference
10. mouseover on css=.mt-6 OK 09:34:50
19. mouseOut on css=.mt-6 OK 09:34:50
20. click on css=.w-10 > .w-full OK 09:34:50
21. mouseOver on css=.w-10 > .w-full OK 09:34:50
22. mouseOut on css=.w-10 > .w-full OK 09:34:50
23. click on css=.inset-0 OK 09:34:50
'Story_Form' completed successfully 09:34:51

The screenshot shows the Selenium IDE interface with the following details:

- Project:** SE_QuestAI_GUI*
- Test Name:** Story_Form*
- Test Steps:** The test consists of four steps:
 - Step 20: ✓ click on element with CSS selector .w-10 > .w-full
 - Step 21: ✓ mouse over on element with CSS selector .w-10 > .w-full
 - Step 22: ✓ mouse out on element with CSS selector .w-10 > .w-full
 - Step 23: ✓ click on element with CSS selector .inset-0
- Log:** The log shows the recorded actions and their timestamps:
 - 19. mouseOut on css=.mt-6 OK (09:34:50)
 - 20. click on css=.w-10 > .w-full OK (09:34:50)
 - 21. mouseOver on css=.w-10 > .w-full OK (09:34:50)
 - 22. mouseOut on css=.w-10 > .w-full OK (09:34:50)
 - 23. click on css=.inset-0 OK (09:34:50)
- Message:** "'Story_Form' completed successfully" (09:34:51)

The Story Form page was tested successfully using Selenium IDE through the recording method.

Entering the title, setting, character, and genre worked perfectly, and the form submission flow was captured without failures.

However, the Chatbox page could not be automated in Selenium, since it requires AI responses, dynamic message rendering, and scrolling—elements that Selenium IDE cannot reliably detect.

Therefore, the Chatbox was manually tested end-to-end, including sending prompts, receiving responses, copying messages, and completing/continuing a story.

All Chatbox functionalities worked correctly during manual testing.

Public Story

The screenshot shows the Selenium IDE interface with the project 'SE_QuestAI_GUI*' selected. A test named 'Public_Story*' is currently running, indicated by the green progress bar at the top of the test list. The test details pane on the right shows the following steps:

Step	Command	Target	Value
1	✓ open	https://quest-ai-frontend.vercel.app/	
2	✓ set window size	1204x680	
3	✓ mouse over	css=.hidden > .form-button:nth-child(5)	
4	✓ click	css=.hidden > .form-button:nth-child(2)	
5	✓ click	css=.fa-play > path	

The log pane at the bottom shows the execution of these steps:

- 2. setWindowSize on 1204x680 OK
- 3. mouseOver on css=.hidden > .form-button:nth-child(5) OK
- 4. click on css=.hidden > .form-button:nth-child(2) OK
- 5. Trying to find css=.fa-play > path... OK
- 6. Trying to find css=span... OK
- 7. click on css=.hidden > .form-button:nth-child(2) OK
- 8. mouseOver on css=.hidden > .form-button:nth-child(2) OK

The screenshot shows the Selenium IDE interface with the project 'SE_QuestAI_GUI*' selected. A test named 'Public_Story*' is currently running, indicated by the green progress bar at the top of the test list. The test details pane on the right shows the following steps:

Step	Command	Target	Value
7	✓ click	css=.hidden > .form-button:nth-child(2)	
8	✓ mouse over	css=.hidden > .form-button:nth-child(2)	
9	✓ click	css=.fa-download > path	
10	✓ click	css=.gap-6	

The log pane at the bottom shows the execution of these steps:

- 6. Trying to find css=.fa-download > path... OK
- 7. click on css=.hidden > .form-button:nth-child(2) OK
- 8. mouseOver on css=.hidden > .form-button:nth-child(2) OK
- 9. Trying to find css=.fa-download > path... OK
- 10. click on css=.gap-6 OK

'Public_Story' completed successfully

The Public Stories page was tested using Selenium IDE through the recording method, and all recorded interactions worked correctly.

Clicking on a Public Story successfully navigates to the Chatbox in public mode, confirming that story viewing without login works as intended.

The Download Story button was also tested and verified to generate and download the text file properly.

Hover effects, navigation, and icon interactions (Play, Download) were captured and executed successfully in Selenium.

Overall, the Public Stories feature works smoothly.

Home Page

The screenshot shows the Selenium IDE interface with a recorded test for the 'Home_Page' scenario. The test steps are:

Step	Command	Target	Value
1	✓ open	https://quest-ai-frontend.vercel.app/Home/Emporer	
2	✓ set window size	1204x680	
3	✓ click	css=.fa-lock > path	
4	✓ click	css=.fa-lock-open > path	
5	✓ click	css=.fa-user-plus > path	

The Log section shows the execution details:

```
Running 'Home_Page'
1. open on https://quest-ai-frontend.vercel.app/Home/Emporer OK
2. setWindowSize on 1204x680 OK
3. Trying to find css=.fa-lock > path... OK
4. Trying to find css=.fa-lock-open > path... OK
5. click on css=.fa-user-plus > path OK
```

The screenshot shows the Selenium IDE interface with a recorded test for the 'Home_Page' scenario. The test steps are:

Step	Command	Target	Value
7	✓ type	css=input	dharva.1010@gmail.com
8	✓ click	css=.mt-4 > .form-button	
9	✓ click	css=.fa-play > path	
10	✓ click	css=.form-button:nth-child(1) > span	
11	✓ mouse over	css=.hidden > .form-button:nth-child(4)	

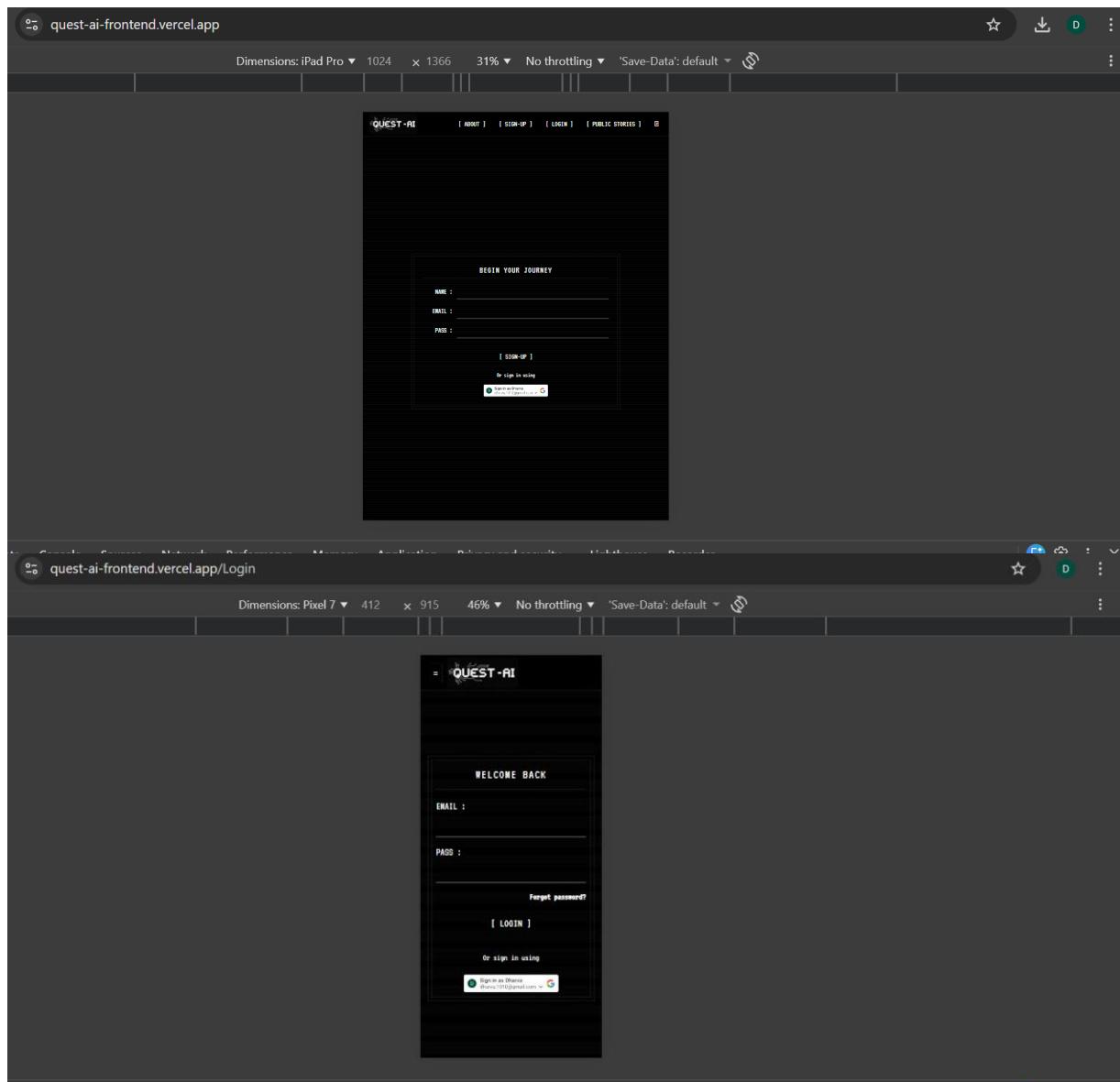
The Log section shows the execution details:

```
0. click on css=button OK
1. type on css=input with value dharva.1010@gmail.com OK
2. click on css=.mt-4 > .form-button OK
3. click on css=.fa-play > path OK
4. Trying to find css=.form-button:nth-child(1) > span... OK
5. mouseOver on css=.hidden > .form-button:nth-child(4) OK
'Home_Page' completed successfully
```

Some buttons on the Home page, especially the ones that open the ChatBox, had to be tested manually because Selenium wasn't able to detect or interact with the ChatBox properly. The rest of the Home page buttons and flows that worked with Selenium were tested using the normal recording method. Both manual and recorded tests were used to make sure the whole Home page works correctly.

Responsive testing

The responsiveness of the application was tested manually using the browser's Inspect → Device Toolbar. All the main pages were checked across multiple device to ensure proper layout, alignment, and usability.. Screenshots of key pages on different device sizes have been captured and included. The pages are shown below:



The image displays three separate screenshots of the QUEST-AI web application, each showing a different page or view of the platform.

Screenshot 1: About Page (Samsung Galaxy A51/71)

Dimensions: Samsung Galaxy A51/71 ▾ 412 x 914 46% ▾ No throttling ▾ 'Save-Data': default ▾

ABOUT QUEST-AI

Welcome to the collaborative dreaming engine. Here, you don't just read the story — you hack the narrative. Set your parameters, initialize your character, and watch a new reality compile itself in real-time.

Why We Built This

→ Back to the Blank Page. Traditional tabletop RPGs are a pain to set up and "Setup fatigue." We built Quest to be a traditionalistic storytelling drop-in-a-garage, pick-a-persona, and let the engine handle the heavy lifting of world-building instantly.

The AI Twist

→ The Dungeon Master Is Alive. We don't use rigid scripts. By leaning into the chaotic creativity of LLMs, we ensure no two sessions are identical. The AI acts as an improvisational partner that reacts to your

Screenshot 2: Public Stories Page (Nest Hub Max)

Dimensions: Nest Hub Max ▾ 1280 x 800 50% ▾ No throttling ▾ 'Save-Data': default ▾

QUEST-AI [ABOUT] [PUBLIC STORIES] [HOME] [CREATE STORY]

Public Stories

harry porter and the Philosopher's stone ► ↴
Owner: dmcwylde1000@gmail.com
Character: Harry Porter
Description: harry porter, with his goes to hogwarts a place where all the Wizards live and where the wizardry is taught.

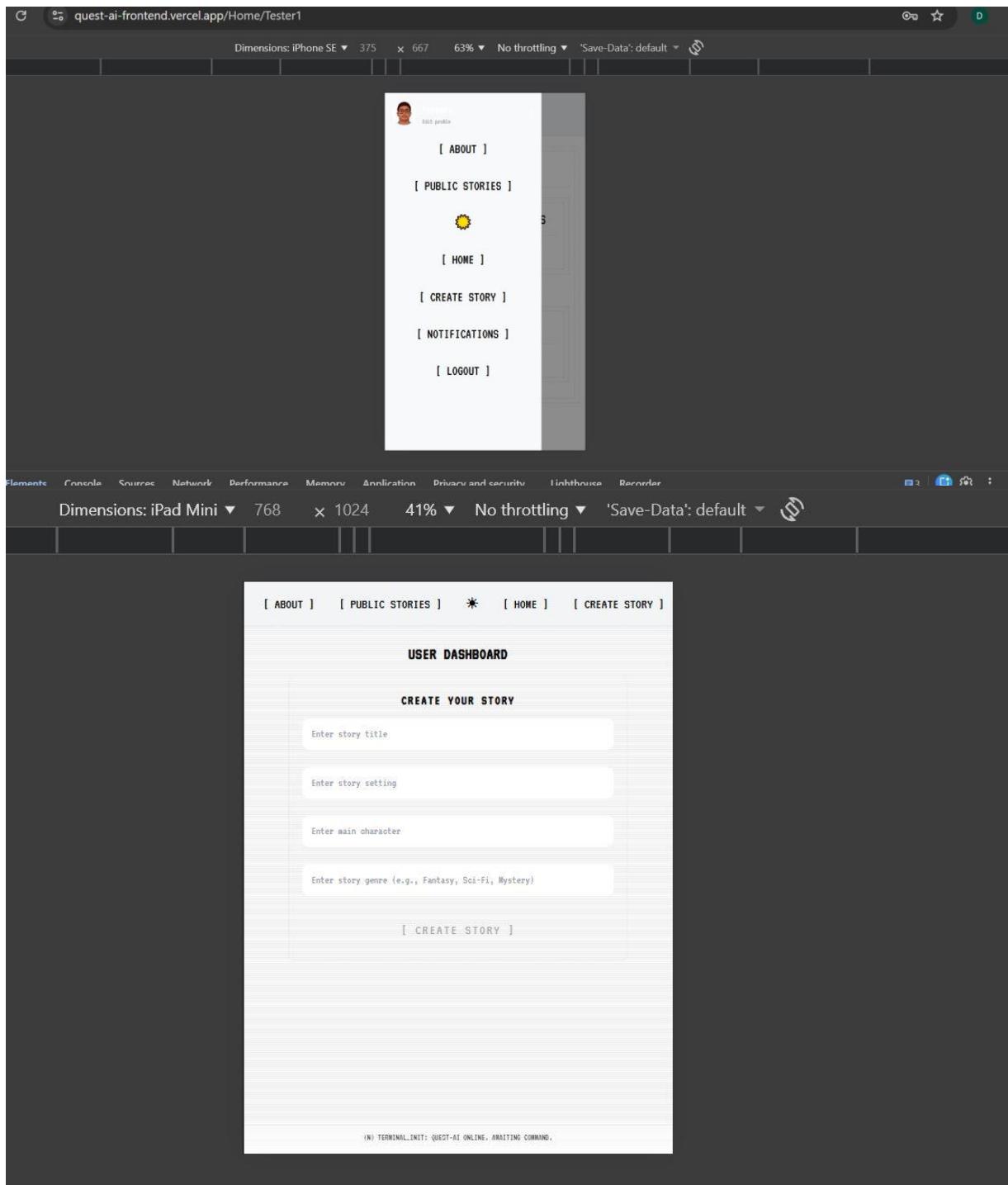
Screenshot 3: User Dashboard (iPhone SE)

Dimensions: iPhone SE ▾ 375 x 667 63% ▾ No throttling ▾ 'Save-Data': default ▾

USER DASHBOARD

COMPLETED STORIES
No completed stories.

ONGOING STORIES
No ongoing stories.



quest-ai-frontend.vercel.app/Edit/Tester1

Dimensions: iPad Pro ▾ 1024 × 1366 31% ▾ No throttling ▾ 'Save-Data': default ▾

QUEST-AI [ABOUT] [PUBLIC STORIES] * [HOME] [CREATE STORY]

Edit Profile

Update profile picture
New Disease (optional)
New Password (optional)

[SAVE CHANGES]

Update API Key

Your API Key

[SAVE API KEY]

quest-ai-frontend.vercel.app/ChatBox/Tester1/6914d022c048c7660141fbe2%20public%20true

Dimensions: iPad Pro ▾ 1024 × 1366 31% ▾ No throttling ▾ 'Save-Data': default ▾

The Gentry Kid barely touches your until fridges before it shudders "Bleeding!"—the falter, wavy, mostly drowsy, but a sample of older products trade looks like they've just snatched their salts. At the long-suffering, you can't help but feel like you're getting off easy. And g'fathers who clean the sun-rot tunnels, and a lady with blonde, Raquel, who's already cracked these people-pie gelato and is lactating, she's been applying. They've both left—years, too, and the boy's back at the barbershop. You can't help but feel like you're a rock polka still hanging with mouthy, and Raquel's entry that continues in the outside-in-square part saving staircase and a passenger with either no or selling out your mom in a voice like you'd never heard. The boy's back at the barbershop. You can't help but feel like the boy's back at the barbershop. You can't help but feel like the boy's back at the barbershop. You can't help but feel like the boy's back at the barbershop.

The Great Wall's morning laughter rings you back—Raquel's twin brothers just showed your partner up to spread peanut butter and jelly on the boy's back at the barbershop. You can't help but feel like the boy's back at the barbershop. You can't help but feel like the boy's back at the barbershop. You can't help but feel like the boy's back at the barbershop. You can't help but feel like the boy's back at the barbershop.

RESET YOUR PASSWORD

Please enter your email to receive a new password (OTP).

EMAIL : Enter your registered email

[SEND OTP]