

Test Scenario	Test Case	Precondition(s)	Steps	Expected Result(s)	Actual Result	Status
Rendering Svelte Components with Props	Should render with title and value	1. The Card.svelte component is available. 2. Props are correctly defined and accessible within the component.	1. Render Card with defined props. 2. Verify Card component.	1. The Card component renders correctly without errors. 2. Both title and value props are displayed correctly.	1. The Card component renders correctly without errors. 2. Props are displayed correctly.	Pass
Rendering Svelte Components with Props	Should render footer text	The Footer.svelte component is available.	1. Render the Footer component. 2. Verify that the footer text is rendered correctly.	The footer text "© 2024 Your Website. All rights reserved." is displayed within the Footer component.	Footer text rendered as expected.	Pass
Rendering Svelte Components with Props	Should render navigation links	1. The Navbar.svelte component is available. 2. Props are correctly defined and accessible within the component.	1. Render the Navigation component with defined props. 2. Verify that the navigation links are rendered correctly.	The navigation links for "Home", "History", "Github", and "Contact" are displayed within the Navigation component, along with the title "WeatherSense".	The Navbar component renders correctly without errors.	Pass
Rendering Svelte Components with Props	Should have correct href attributes	The Navbar.svelte component has been rendered correctly.	Verify Navbar href.	The Navbar component has the correct href attributes for each navigation link.	The Navbar component has the correct href attributes for each navigation link.	Pass
Rendering Svelte Components with Props	Should render formatted timestamp with prefix	1. The Timestamp.svelte component is available. 2. Props are correctly defined and accessible within the component.	1. Render Timestamp with defined props. 2. Verify Timestamp component.	1. The Timestamp component renders correctly without errors. 2. Both timestamp and prefix props are displayed correctly.	1. The Timestamp component renders correctly without errors. 2. Both timestamp and prefix props are displayed correctly.	Pass

Test Scenario	Test Case	Precondition(s)	Steps	Expected Result(s)	Actual Result	Status
Rendering Svelte Components with Props	Should render formatted timestamp with link	<ol style="list-style-type: none"> <li>1. The TimestampHistory.svelte component is available.</li> <li>2. Props are correctly defined and accessible within the component.</li> </ol>	<ol style="list-style-type: none"> <li>1. Render TimestampHistory with defined props.</li> <li>2. Verify TimestampHistory component.</li> </ol>	<ol style="list-style-type: none"> <li>1. The TimestampHistory component renders correctly without errors.</li> <li>2. Both timestamp props are displayed correctly.</li> <li>3. The href attribute contains a correct link.</li> </ol>	<ol style="list-style-type: none"> <li>1. The TimestampHistory component renders correctly without errors.</li> <li>2. Both timestamp props are displayed correctly.</li> <li>3. The href attribute contains a correct link.</li> </ol>	Pass
HTTP Requests to Every Web Page Route	GET http request to "/"	The front-end server must be running.	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/"</li> <li>2. Verify response status code</li> </ol>	Status 200	Status 200	Pass
HTTP Requests to Every Web Page Route	GET request to "/history"	The front-end server must be running.	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/history"</li> <li>2. Verify response status code</li> </ol>	Status 200	Status 200	Pass
HTTP Requests to Every Web Page Route	GET request to "/weather/id" with <b>valid</b> id	<ol style="list-style-type: none"> <li>1. The front-end server is running.</li> <li>2. Back-end server is running.</li> <li>3. Define valid id to test</li> </ol>	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/weather/id"</li> <li>2. Verify response status code</li> </ol>	Status 200	Status 200	Pass
HTTP Requests to Every Web Page Route	GET request to "/weather/id" with <b>invalid</b> id	<ol style="list-style-type: none"> <li>1. The front-end server is running.</li> <li>2. Back-end server is running.</li> <li>3. Define invalid id to test</li> </ol>	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/weather/id"</li> <li>2. Verify response status code</li> </ol>	Status 404	Status 404	Pass
Verifies that the website functions correctly and as expected from the user's perspective	Users can go to the main page and can click all the button navigation bars.	All website services are running.	<p>Go to route "localhost:xxxx/"</p> <ol style="list-style-type: none"> <li>1. Click all Navigation bar buttons.</li> </ol>	All buttons on navigation bars can click and redirect.	<ol style="list-style-type: none"> <li>1. Card component rendered correctly without errors.</li> <li>2. Props displayed correctly.</li> </ol>	Pass

Test Scenario	Test Case	Precondition(s)	Steps	Expected Result(s)	Actual Result	Status
Verifies that the website functions correctly and as expected from the user's perspective	Users can go to the History page and View old weather data and go back to home.	All website services are running.	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/"</li> <li>2. Click the button History on the Navigation bar.</li> <li>3. Click view on some data.</li> <li>4. Click the Home button on the Navigation bar.</li> </ol>	Users can go to the history page and view old data and can go back to home.	Users can go to the history page and view old data and can go back to home.	Pass
Verifies that the website functions correctly and as expected from the user's perspective	In the home page all data Card components are rendered.	All website services are running.	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/"</li> <li>2. Verify the page.</li> </ol>	All data displayed.	All data displayed.	Pass
Verifies that the website functions correctly and as expected from the user's perspective	In the history page the View button must be loaded.	All website services are running.	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/"</li> <li>2. Click the History button.</li> <li>3. Verify that the view button appears.</li> </ol>	In each old data view button must be rendered.	In each old data view button must be rendered.	Pass
Verifies that the website functions correctly and as expected from the user's perspective	In another weather data board (history) Data card must be loaded.	All website services are running.	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/"</li> <li>2. Click the History button.</li> <li>3. Click the View button on any data.</li> <li>4. Verify that all data cards must be rendered all.</li> </ol>	In the old weather data dashboard all data cards must be loaded.	In the old weather data dashboard all data cards must be loaded.	Pass
Verifies that the website functions correctly and as expected from the user's perspective	Visualize page must render all component.	All website services are running.	<ol style="list-style-type: none"> <li>1. Go to route "localhost:xxxx/"</li> <li>2. Click the History button.</li> <li>3. Click the View button on any data.</li> <li>4. Verify that all data cards must be rendered all.</li> </ol>	User can see all visualized data components.	User can see all visualized data components.	Pass

Test Scenario	Test Case	Precondition(s)	Steps	Expected Result(s)	Actual Result	Status
Backend API Endpoint Validation	GET HTTP Request to "/latest"	Back-end Server is running.	1. HTTP GET request to "localhost:3000/". 2. Verify the response.	Response status and json are the latest weather data.	Response status and json are the latest weather data.	Pass
Backend API Endpoint Validation	GET HTTP Request to "/history"	Back-end Server is running.	1. HTTP GET request to "localhost:3000/history". 2. Verify the response.	Response status and json are all the weather data ordered by DESC.	Response status and json are all the weather data ordered by DESC.	Pass
Backend API Endpoint Validation	HTTP GET Request to "/weather/id"	1. Back-end Server is running 2. Defined id	1. HTTP GET request to "localhost:3000/weather/id". 2. Verify the response.	Response status and json are the weather data specific id.	Response status and json are the weather data specific id.	Pass
ModelAPI-Server Endpoint Validation	HTTP POST Request to "/predict"	1. ModelAPI-Server is running 2. Defined data to send to predict a model.	1. HTTP GET request to "localhost:5000/predict". 2. Verify the response.	Response status and json are the same data but have new attribute name "weather_pred" and its value.	Response status and json are the same data but have new attribute name "weather_pred" and its value.	Pass