

Documentation tool

We used IntelliSense as our documentation tool. It is intelligent and powered by a language service.

language service:

A language service provides intelligent code documentation based on language semantics and an analysis of your source code. If a language service knows possible completions, the IntelliSense suggestions will pop up as you type. If you continue typing characters, the list of members (variables, methods, etc.)

The way to use IntelliSense:

Here is the video tutorial regarding it's use: <https://www.youtube.com/watch?v=OxWap-qdlcQ>


Sample of some code Documentations:

```
1.js X
> Ishma > Desktop > DT > JS Untitled-1.js > [0] addMovie

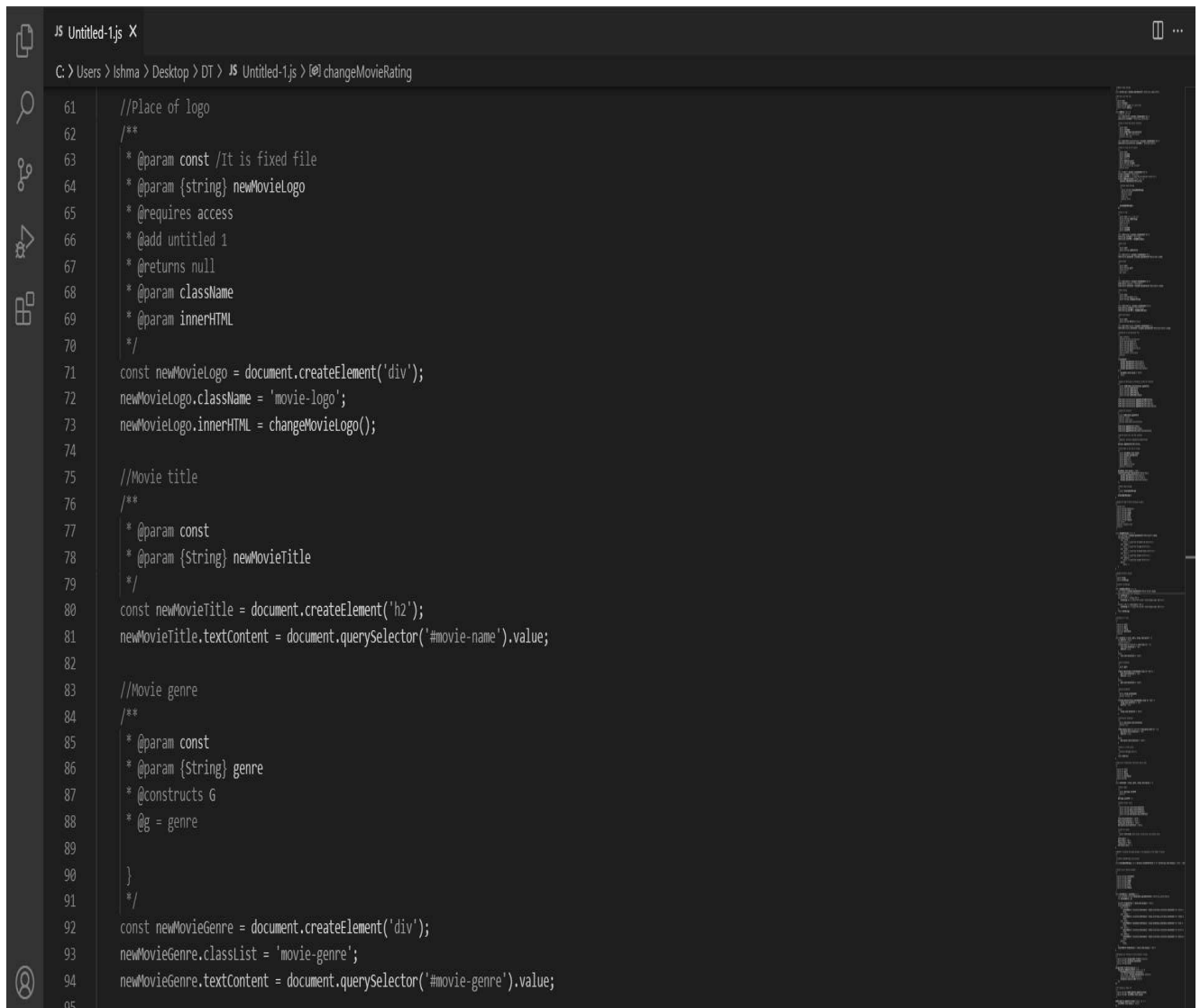
Removing the empty message display
*
  @action clear all
  @Return empty message
/
const emptyMessage = document.querySelector('.movie-list__empty-item');

Add movie into the list
*
  @param const
  @param className
  @param {string} movie-list__movie-item
  @return @const addMovie
/
const addMovie = () => {
  //Create a new item
  const newMovieItem = document.createElement('div');
  newMovieItem.className = 'movie-list__movie-item';

  //Place of movie description container
  /**
   * @param const
   * @param className
   * @const newMovieDescriptionContainer
   * @constant New movie description
   * @returns class name
   */
  const newMovieDescriptionContainer = document.createElement('div');
  newMovieDescriptionContainer.className = 'movie-description';
```

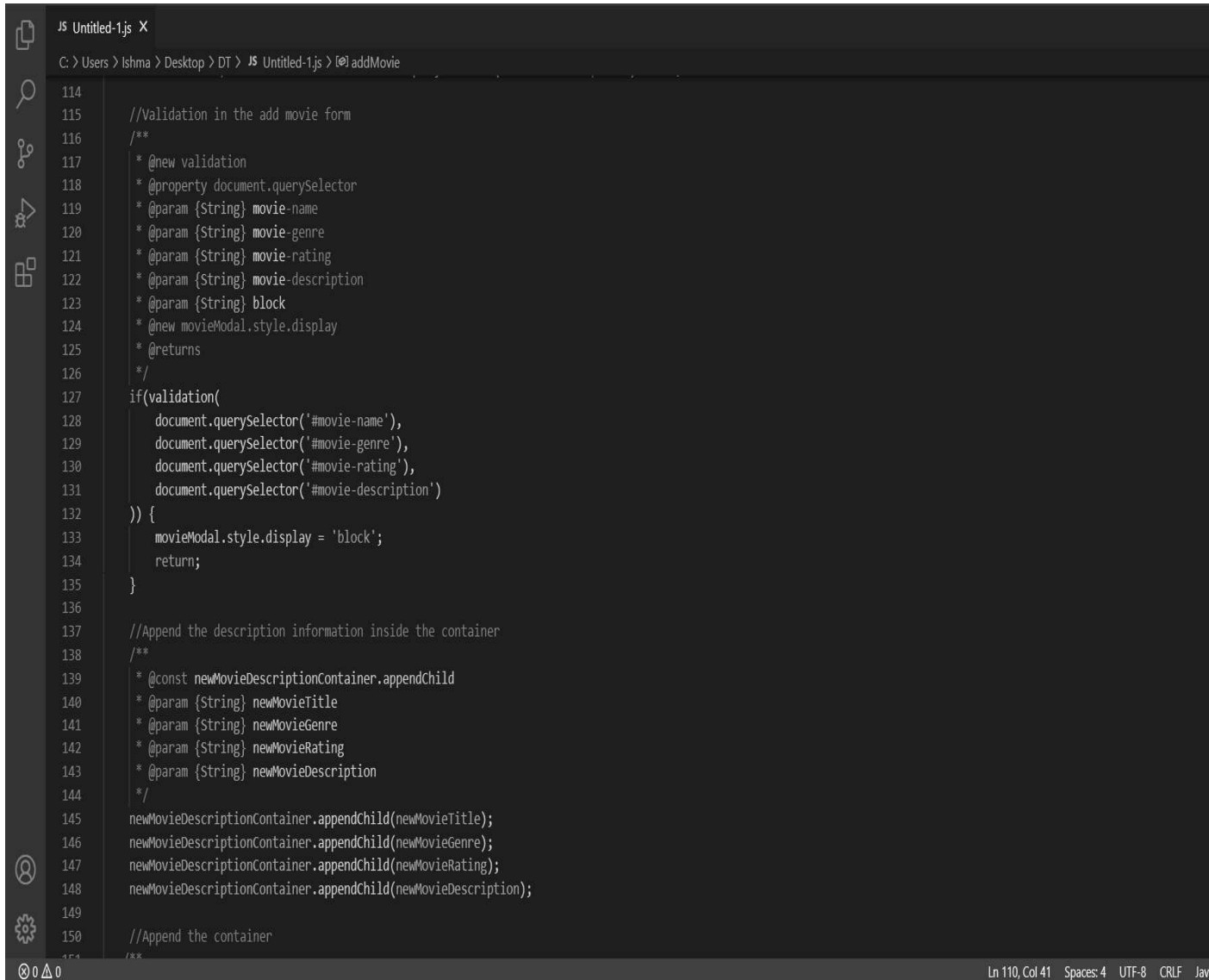


```
31
32 //Place of trash bin for delete
33 /**
34  * @param const
35  * @param className
36  * @param innerHTML
37  * @add trash-icon
38  * @param addEventListener
39  * @param {String} trashBin
40  * @new add const{string} trashBin
41  * @action delete
42  */
43 const trashBin = document.createElement('div');
44 trashBin.className = 'trash-icon';
45 trashBin.innerHTML = '<i class="fas fa-trash-alt fa-2x"></i>';
46 trashBin.addEventListener('click', () => {
47   movieList.removeChild(newMovieItem);
48
49   //Display empty message
50   /**
51    * @param {string} displayEmptyMessage
52    * @private messages
53    * @requires access
54    * @type text
55    * @action delete
56    */
57
58   displayEmptyMessage();
59 });
60
61 //Place of logo
62 /**
63  * @param const /It is fixed file
64  * @param {string} newMovieLogo
65  * @requires access
66  * @add untitled 1
67  * @returns null
68  * @param className
```



```
JS Untitled-1.js X
C:\Users\Ishma\Desktop\DT\JS Untitled-1.js | changeMovieRating

61 //Place of logo
62 /**
63  * @param const /It is fixed file
64  * @param {string} newMovieLogo
65  * @requires access
66  * @add untitled 1
67  * @returns null
68  * @param className
69  * @param innerHTML
70  */
71 const newMovieLogo = document.createElement('div');
72 newMovieLogo.className = 'movie-logo';
73 newMovieLogo.innerHTML = changeMovieLogo();
74
75 //Movie title
76 /**
77  * @param const
78  * @param {String} newMovieTitle
79  */
80 const newMovieTitle = document.createElement('h2');
81 newMovieTitle.textContent = document.querySelector('#movie-name').value;
82
83 //Movie genre
84 /**
85  * @param const
86  * @param {String} genre
87  * @constructs 6
88  * @eg = genre
89  */
90 }
91
92 const newMovieGenre = document.createElement('div');
93 newMovieGenre.classList = 'movie-genre';
94 newMovieGenre.textContent = document.querySelector('#movie-genre').value;
95
```

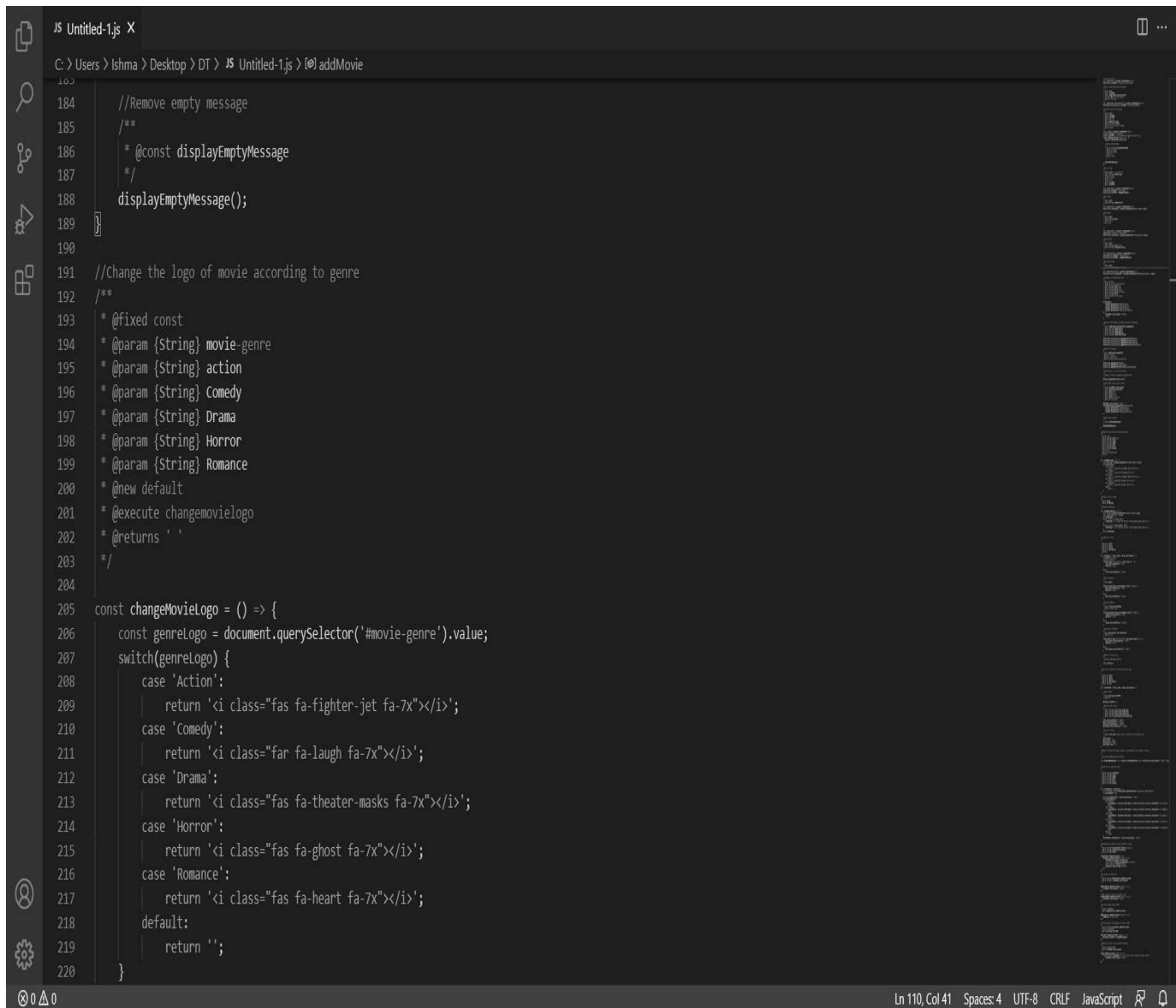
```
114 //Validation in the add movie form
115 /**
116  * @new validation
117  * @property document.querySelector
118  * @param {String} movie-name
119  * @param {String} movie-genre
120  * @param {String} movie-rating
121  * @param {String} movie-description
122  * @param {String} block
123  * @new movieModal.style.display
124  * @returns
125  */
126
127 if(validation(
128     document.querySelector('#movie-name'),
129     document.querySelector('#movie-genre'),
130     document.querySelector('#movie-rating'),
131     document.querySelector('#movie-description')
132 )) {
133     movieModal.style.display = 'block';
134     return;
135 }
136
137 //Append the description information inside the container
138 /**
139  * @const newMovieDescriptionContainer.appendChild
140  * @param {String} newMovieTitle
141  * @param {String} newMovieGenre
142  * @param {String} newMovieRating
143  * @param {String} newMovieDescription
144  */
145 newMovieDescriptionContainer.appendChild(newMovieTitle);
146 newMovieDescriptionContainer.appendChild(newMovieGenre);
147 newMovieDescriptionContainer.appendChild(newMovieRating);
148 newMovieDescriptionContainer.appendChild(newMovieDescription);
149
150 //Append the container
151 /**
```

Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript



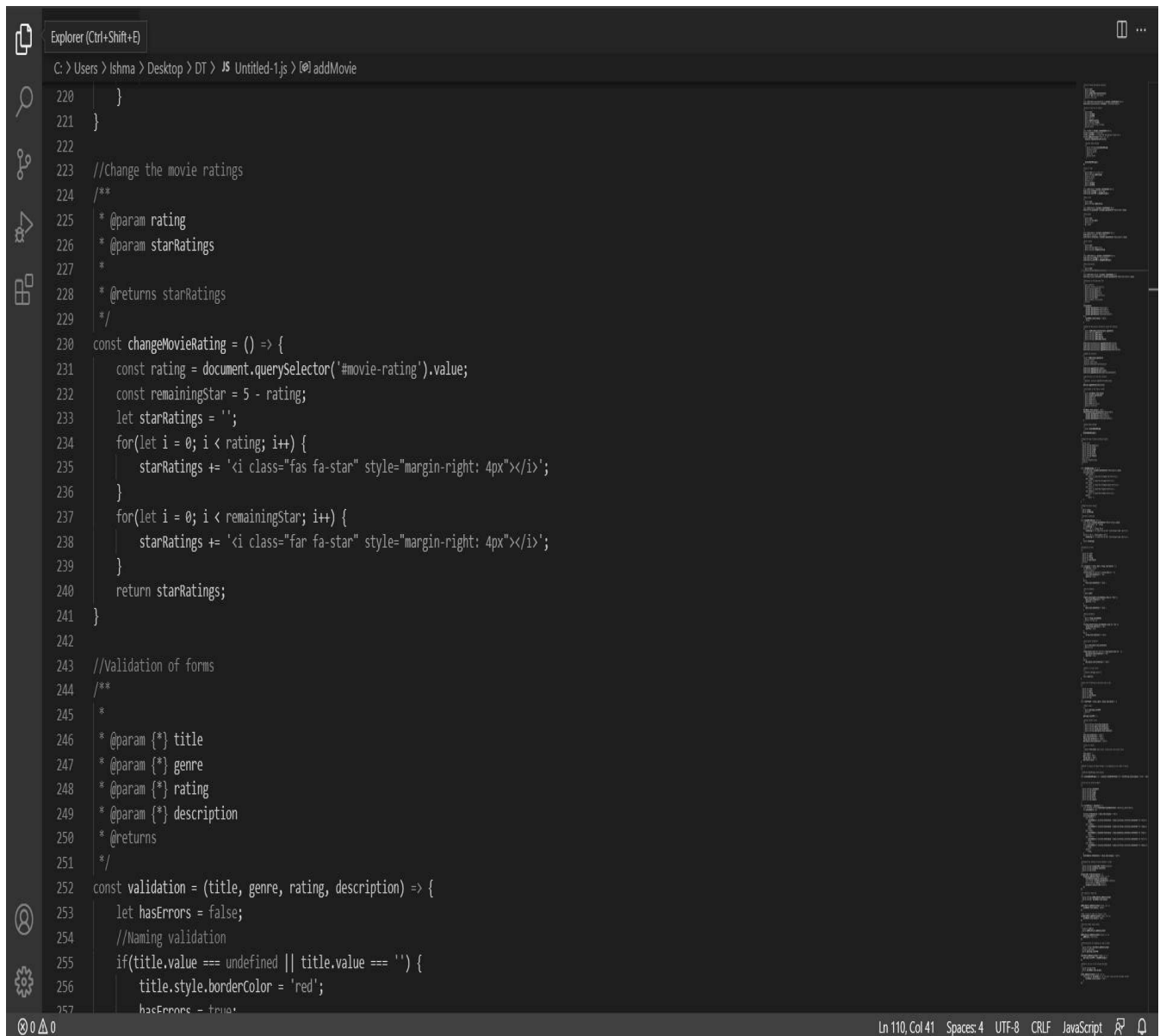
```
JS Untitled-1.js X
C:\Users\Ishma\Desktop\DT\JS Untitled-1.js addMovie
150 //Append the container
151 /**
152  * @const newMovieItem.appendChild
153  * @execute trashbin
154  * @execute newMovieLogo
155  * @execute newMovieDescriptionContainer
156  */
157 newMovieItem.appendChild(trashBin);
158 newMovieItem.appendChild(newMovieLogo);
159 newMovieItem.appendChild(newMovieDescriptionContainer)
160
161 //Add the movie list into the container
162 /**
163  * @execute movieList.appendChild(newMovieItem)
164  */
165 movieList.appendChild(newMovieItem);
166
167 //Close modal as the item is listed
168 /**
169  * @const movieModal.style.display
170  * @param document.querySelector
171  * @param movie-name
172  * @param movie-genre
173  * @param movie-rating
174  * @param movie-description
175  * @execute clearFormat
176  */
177 movieModal.style.display = 'none';
178 clearFormat(document.querySelector('#movie-name'),
179   document.querySelector('#movie-genre'),
180   document.querySelector('#movie-rating'),
181   document.querySelector('#movie-description')
182 );
183
184 //Remove empty message
185 /**
186  * @const displayEmptyMessage
187  */
```

Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript



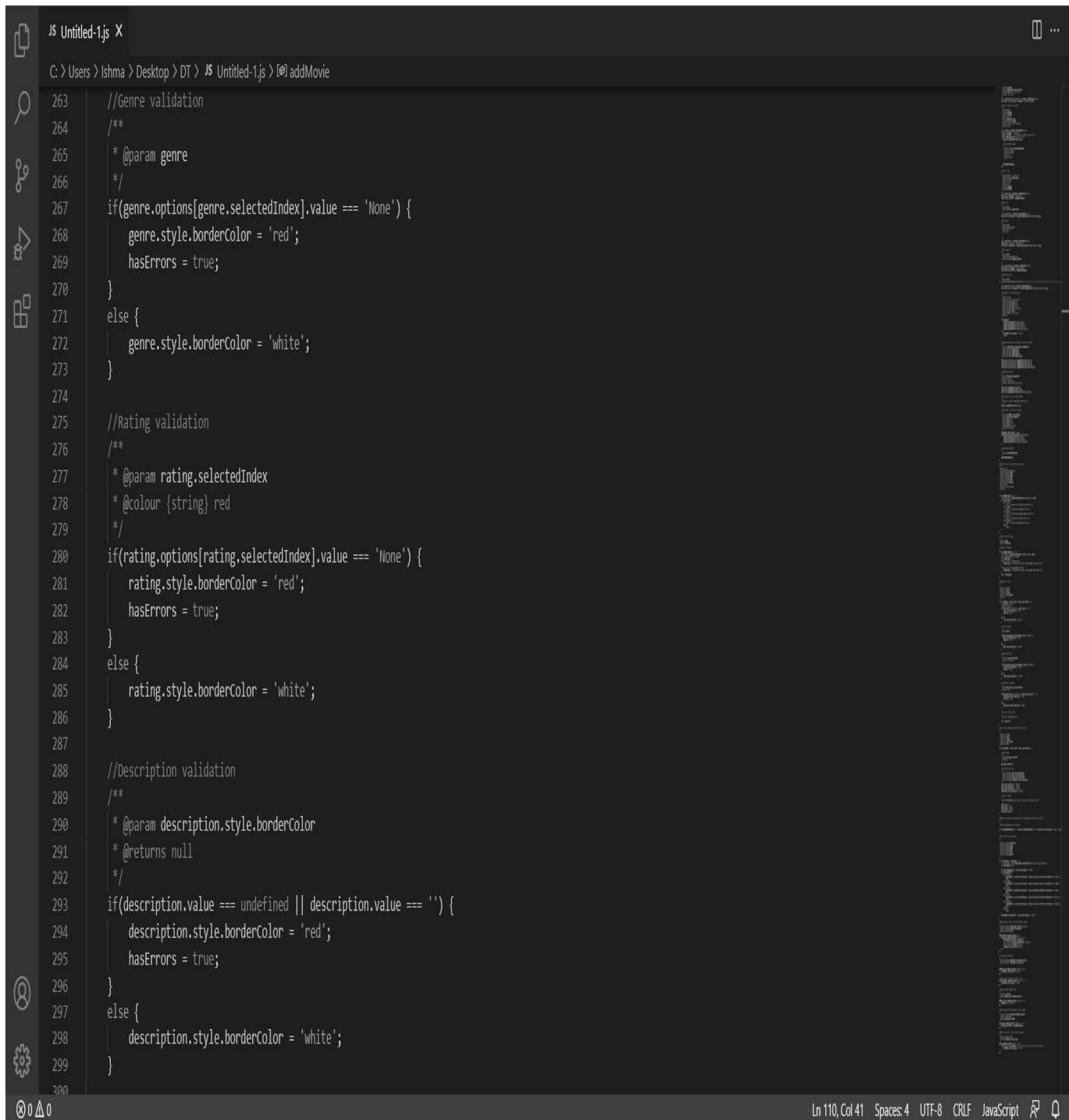
```
JS Untitled-1.js X
C:\Users\Ishma\Desktop>DT> JS Untitled-1.js > [0] addMovie
184 //Remove empty message
185 /**
186  * @const displayEmptyMessage
187  */
188 displayEmptyMessage();
189
190
191 //Change the logo of movie according to genre
192 /**
193  * @fixed const
194  * @param {String} movie-genre
195  * @param {String} action
196  * @param {String} Comedy
197  * @param {String} Drama
198  * @param {String} Horror
199  * @param {String} Romance
200  * @new default
201  * @execute changemovieLogo
202  * @returns ' '
203  */
204
205 const changeMovieLogo = () => {
206   const genreLogo = document.querySelector('#movie-genre').value;
207   switch(genreLogo) {
208     case 'Action':
209       return 'i class="fas fa-fighter-jet fa-7x"></i>';
210     case 'Comedy':
211       return 'i class="far fa-laugh fa-7x"></i>';
212     case 'Drama':
213       return 'i class="fas fa-theater-masks fa-7x"></i>';
214     case 'Horror':
215       return 'i class="fas fa-ghost fa-7x"></i>';
216     case 'Romance':
217       return 'i class="fas fa-heart fa-7x"></i>';
218     default:
219       return '';
220   }
}
```

Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript



```
220 }
221 }
222
223 //Change the movie ratings
224 /**
225  * @param rating
226  * @param starRatings
227  *
228  * @returns starRatings
229  */
230 const changeMovieRating = () => {
231   const rating = document.querySelector('#movie-rating').value;
232   const remainingStar = 5 - rating;
233   let starRatings = '';
234   for(let i = 0; i < rating; i++) {
235     starRatings += '<i class="fas fa-star" style="margin-right: 4px"></i>';
236   }
237   for(let i = 0; i < remainingStar; i++) {
238     starRatings += '<i class="far fa-star" style="margin-right: 4px"></i>';
239   }
240   return starRatings;
241 }
242
243 //Validation of forms
244 /**
245  *
246  * @param {*} title
247  * @param {*} genre
248  * @param {*} rating
249  * @param {*} description
250  * @returns
251  */
252 const validation = (title, genre, rating, description) => {
253   let hasErrors = false;
254   //Naming validation
255   if(title.value === undefined || title.value === '') {
256     title.style.borderColor = 'red';
257     hasErrors = true;
```

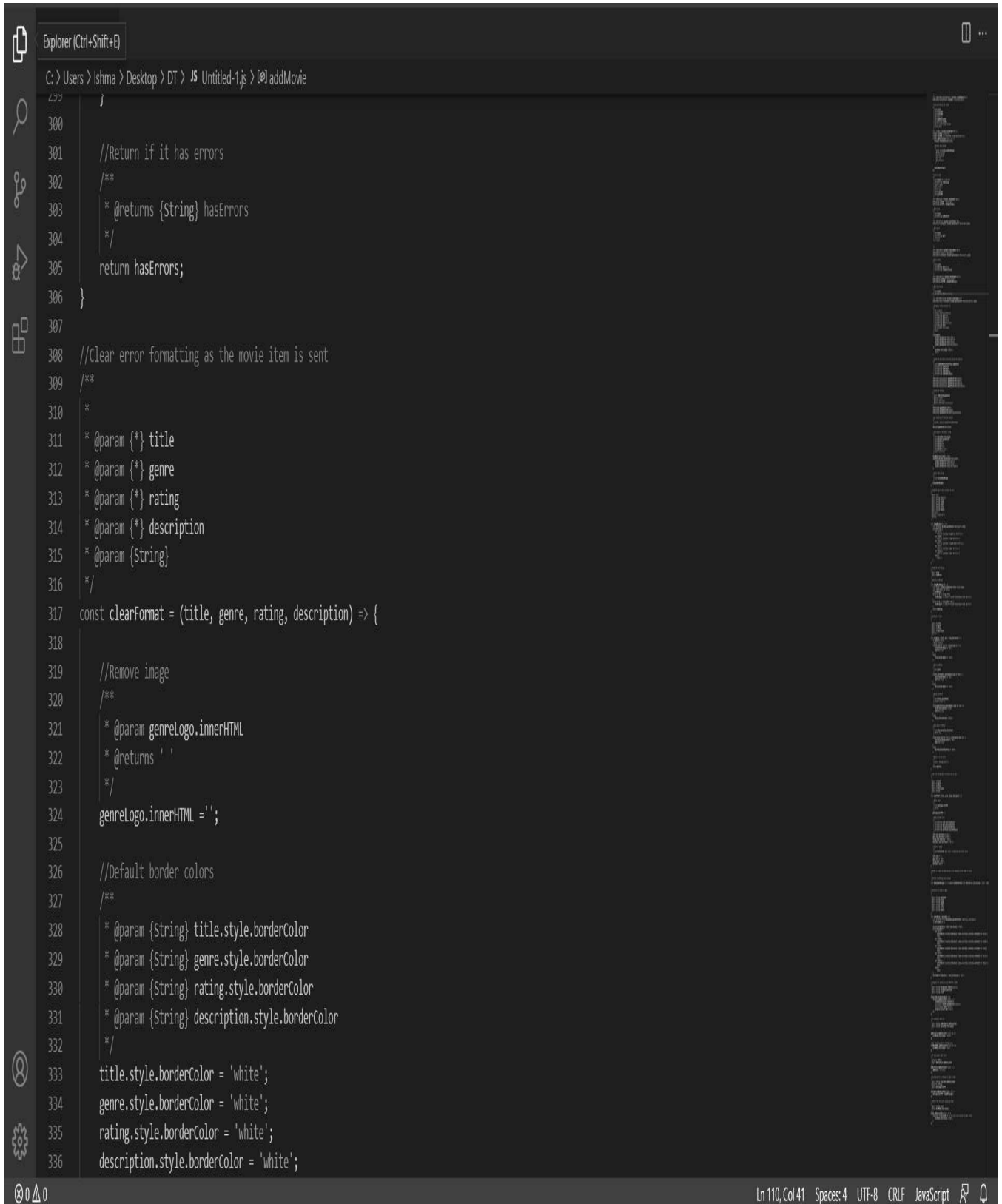
Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript



The image shows a screenshot of a VS Code editor window titled "JS Untitled-1.js X". The file path in the breadcrumb is "C:\> Users > Ishma > Desktop > DT > JS Untitled-1.js > [9] addMovie". The editor contains JavaScript code for validating movie input fields. The code is as follows:

```
263 //Genre validation
264 /**
265  * @param genre
266  */
267 if(genre.options[genre.selectedIndex].value === 'None') {
268     genre.style.borderColor = 'red';
269     hasErrors = true;
270 }
271 else {
272     genre.style.borderColor = 'white';
273 }
274
275 //Rating validation
276 /**
277  * @param rating.selectedIndex
278  * @colour {string} red
279  */
280 if(rating.options[rating.selectedIndex].value === 'None') {
281     rating.style.borderColor = 'red';
282     hasErrors = true;
283 }
284 else {
285     rating.style.borderColor = 'white';
286 }
287
288 //Description validation
289 /**
290  * @param description.style.borderColor
291  * @returns null
292  */
293 if(description.value === undefined || description.value === '') {
294     description.style.borderColor = 'red';
295     hasErrors = true;
296 }
297 else {
298     description.style.borderColor = 'white';
299 }
300
```

The status bar at the bottom indicates "Ln 110, Col 41", "Spaces: 4", "UTF-8", "CRLF", "JavaScript", and icons for search, run and debug, and a refresh icon.



```
Explorer (Ctrl+Shift+E)
C:\Users\Ishma\Desktop\DT\JS\Untitled-1.js @ addMovie

300
301 //Return if it has errors
302 /**
303  * @returns {String} hasErrors
304  */
305 return hasErrors;
306 }
307
308 //Clear error formatting as the movie item is sent
309 /**
310  *
311  * @param {*} title
312  * @param {*} genre
313  * @param {*} rating
314  * @param {*} description
315  * @param {String}
316  */
317 const clearFormat = (title, genre, rating, description) => {
318
319 //Remove image
320 /**
321  * @param genreLogo.innerHTML
322  * @returns ' '
323  */
324 genreLogo.innerHTML = '';
325
326 //Default border colors
327 /**
328  * @param {String} title.style.borderColor
329  * @param {String} genre.style.borderColor
330  * @param {String} rating.style.borderColor
331  * @param {String} description.style.borderColor
332  */
333 title.style.borderColor = 'white';
334 genre.style.borderColor = 'white';
335 rating.style.borderColor = 'white';
336 description.style.borderColor = 'white';

```

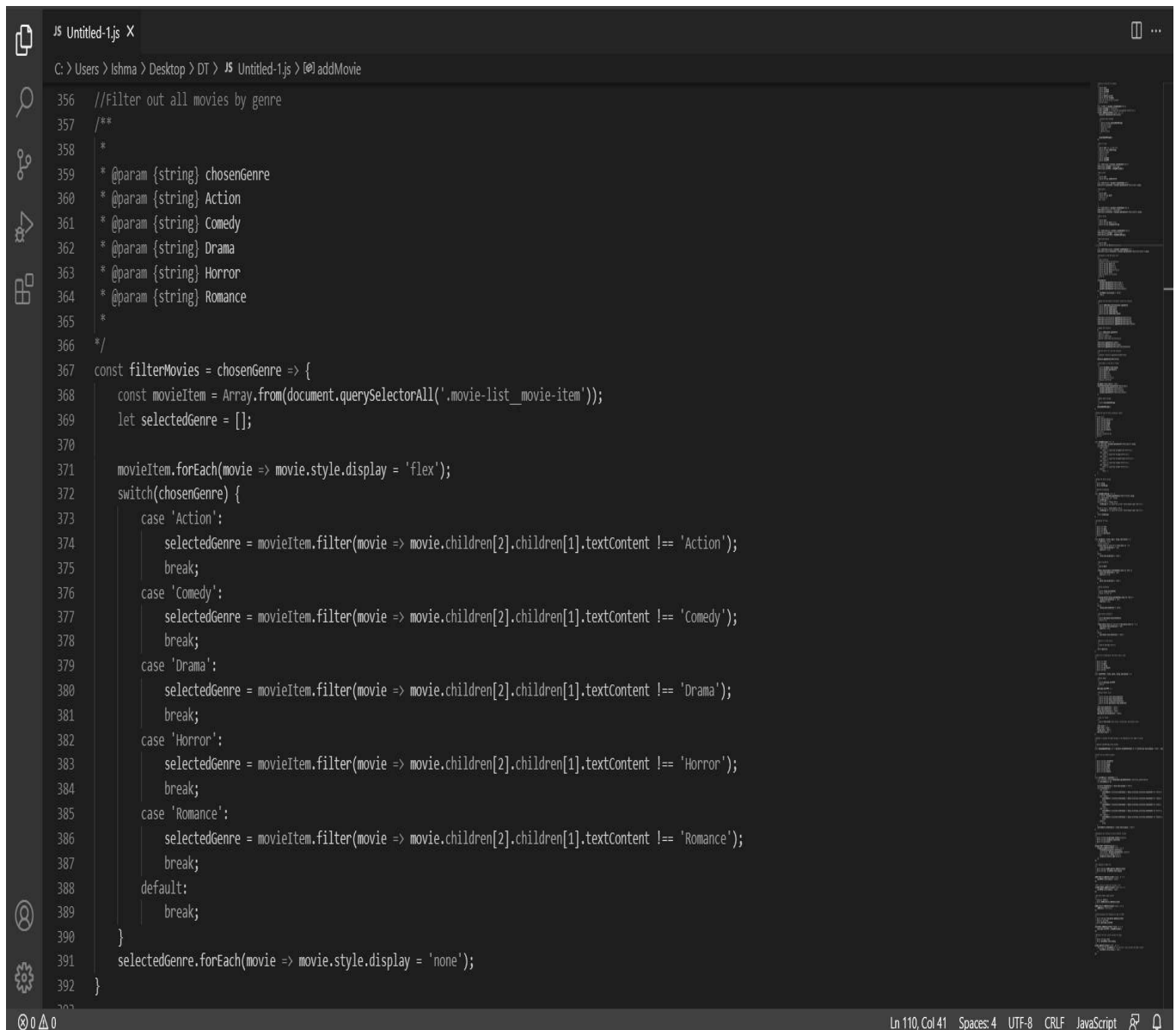
Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript

```

326 //Default border colors
327 /**
328  * @param {String} title.style.borderColor
329  * @param {String} genre.style.borderColor
330  * @param {String} rating.style.borderColor
331  * @param {String} description.style.borderColor
332  */
333 title.style.borderColor = 'white';
334 genre.style.borderColor = 'white';
335 rating.style.borderColor = 'white';
336 description.style.borderColor = 'white';
337
338 //Clear all values
339 /**
340  * @param title.value; genre.value; rating.value; description.value;
341  */
342 title.value = '';
343 genre.value = 'None';
344 rating.value = 'None';
345 description.value = '';
346 }
347
348 //Whether to display the empty message or not depending on the number of movies
349 /**
350  *
351  * @returns emptyMessage.style.display
352  */
353 const displayEmptyMessage = () => movieList.childElementCount <= 2 ? emptyMessage.style.display = 'block' : emptyMessage.style.display = 'none';
354
355
356 //Filter out all movies by genre
357 /**
358  *
359  * @param {string} chosenGenre
360  * @param {string} Action
361  * @param {string} Comedy
362  * @param {string} Drama
363  * @param {string} Horror

```

Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript



```
356 //Filter out all movies by genre
357 /**
358 *
359 * @param {string} chosenGenre
360 * @param {string} Action
361 * @param {string} Comedy
362 * @param {string} Drama
363 * @param {string} Horror
364 * @param {string} Romance
365 *
366 */
367 const filterMovies = chosenGenre => {
368   const movieItem = Array.from(document.querySelectorAll('.movie-list_movie-item'));
369   let selectedGenre = [];
370
371   movieItem.forEach(movie => movie.style.display = 'flex');
372   switch(chosenGenre) {
373     case 'Action':
374       selectedGenre = movieItem.filter(movie => movie.children[2].children[1].textContent !== 'Action');
375       break;
376     case 'Comedy':
377       selectedGenre = movieItem.filter(movie => movie.children[2].children[1].textContent !== 'Comedy');
378       break;
379     case 'Drama':
380       selectedGenre = movieItem.filter(movie => movie.children[2].children[1].textContent !== 'Drama');
381       break;
382     case 'Horror':
383       selectedGenre = movieItem.filter(movie => movie.children[2].children[1].textContent !== 'Horror');
384       break;
385     case 'Romance':
386       selectedGenre = movieItem.filter(movie => movie.children[2].children[1].textContent !== 'Romance');
387       break;
388     default:
389       break;
390   }
391   selectedGenre.forEach(movie => movie.style.display = 'none');
392 }
```



```
JS Untitled-1.js X
C:\Users\Ishma\Desktop\DT\JS Untitled-1.js [9] addMovie
392 }
393
394 //Navigation bar setting to active whenever clicked
395 /**
396  * @param {string} navigationBar.forEach(navigation)
397  * @param {string} navigation.textContent
398  * @param {string} Active
399  */
400 navigationBar.forEach(navigation => {
401   navigation.addEventListener('click', e => {
402     filterMovies(navigation.textContent);
403     const active = document.querySelector('.active');
404     active.classList.remove('active');
405     navigation.classList.add('active');
406   });
407 });
408
409 //For opening of modal box
410 /**
411  * @param {String} addMovieButton.addEventListener
412  * @param {String} movieModal.style.display
413  */
414
415 addMovieButton.addEventListener('click', () => {
416   movieModal.style.display = 'block';
417 });
418
419 //When closing of modal box through x icon
420 closeMovieModal.addEventListener('click', () => {
421   movieModal.style.display = 'none';
422 });
423
424 //Add movie modal submit button
425 /**
426  * @requires addMovie
427  * @param addMovieToList.addEventListener
428  */
429 addMovieToList.addEventListener('click', e => {
```

Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript



```
JS Untitled-1.js X
C:\Users\Ishma\Desktop>DT>JS Untitled-1.js>[!@]addMovie

422 });
423
424 //Add movie modal submit button
425 /**
426  * @requires addMovie
427  * @param addMovieToList.addEventListener
428  */
429 addMovieToList.addEventListener('click', e => {
430   addMovie(); //Add movies
431 });
432
433 //Selecting genre and changing its logo in modal
434 /**
435  * @param {String} selectGenre.addEventListener
436  * @requires genreLogo
437  * @param genreLogo.innerHTML
438  */
439 selectGenre.addEventListener('change', e => {
440   genreLogo.innerHTML = changeMovieLogo();
441 });
442
443 //Whenever the user clicks outside the modal
444 /**
445  * @param {string} click
446  * @const movieModal.style.display
447  */
448 window.addEventListener('click', e => {
449   if(e.target === movieModal) { //If the user clicks outside the modal content
450     movieModal.style.display = 'none';
451   }
452 });
```

Ln 110, Col 41 Spaces: 4 UTF-8 CRLF JavaScript