



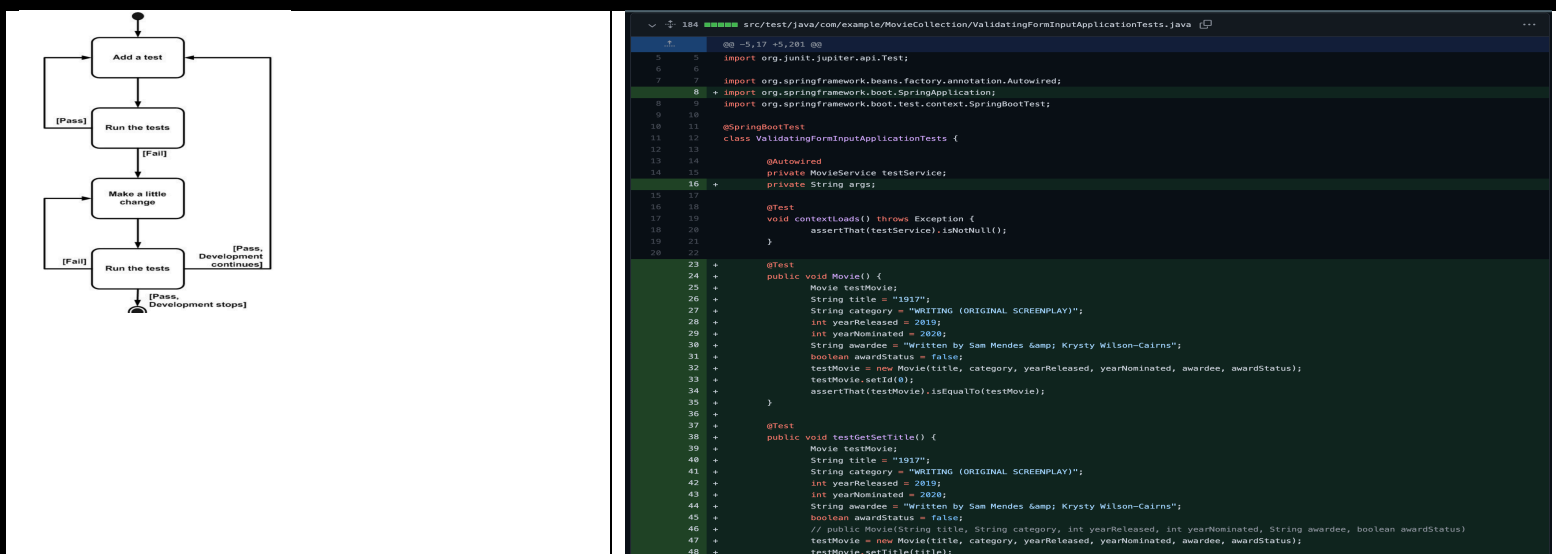
Test Driven Development (TDD) Documentation

Figure 1's UML activity diagram depicts the processes of test first development (TFD).

The first step is to quickly add a test that just has enough code to fail. Following that, we run project tests, which are generally the whole test suite, but for the purpose of speed, we have chosen to run only a subset to ensure that the new test fails.

The project's functional code is then modified to ensure that it passes the new tests. The fourth step is to run your tests again. We had to update and retest our functional code if they failed. When the tests are successful, the process is restarted (you may first need to refactor any duplication out of your design as needed, turning TFD into TDD).

Unit Test Diagram and Code Performance:



```

49 +         assertThat(testMovie.getTitle()).contains(title);
50 +     }
51 + }
52 +
53 + @test
54 + public void testGetSetCategory() {
55 +     Movie testMovie;
56 +     String title = "1917";
57 +     String category = "WARING (ORIGINAL SCREENPLAY)";
58 +     int yearReleased = 2019;
59 +     int yearNominated = 2020;
60 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
61 +     boolean awardStatus = false;
62 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
63 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
64 +     testMovie.setCategory(category);
65 +     assertThat(testMovie.getCategory()).contains(category);
66 + }
67 +
68 + @test
69 + public void testGetSetYearReleased() {
70 +     Movie testMovie;
71 +     String title = "1917";
72 +     String category = "WARING (ORIGINAL SCREENPLAY)";
73 +     int yearReleased = 2019;
74 +     int yearNominated = 2020;
75 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
76 +     boolean awardStatus = false;
77 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
78 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
79 +     testMovie.setYearReleased(yearReleased);
80 +     assertThat(testMovie.getYearReleased()).isEqualTo(yearReleased);
81 + }
82 +
83 + @test
84 + public void testGetSetYearNominated() {
85 +     Movie testMovie;
86 +     String title = "1917";
87 +     String category = "WARING (ORIGINAL SCREENPLAY)";
88 +     int yearReleased = 2019;
89 +     int yearNominated = 2020;
90 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
91 +     boolean awardStatus = false;
92 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
93 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
94 +     testMovie.setYearNominated(yearNominated);
95 +     assertThat(testMovie.getYearNominated()).isEqualTo(yearNominated);
96 + }

```

```

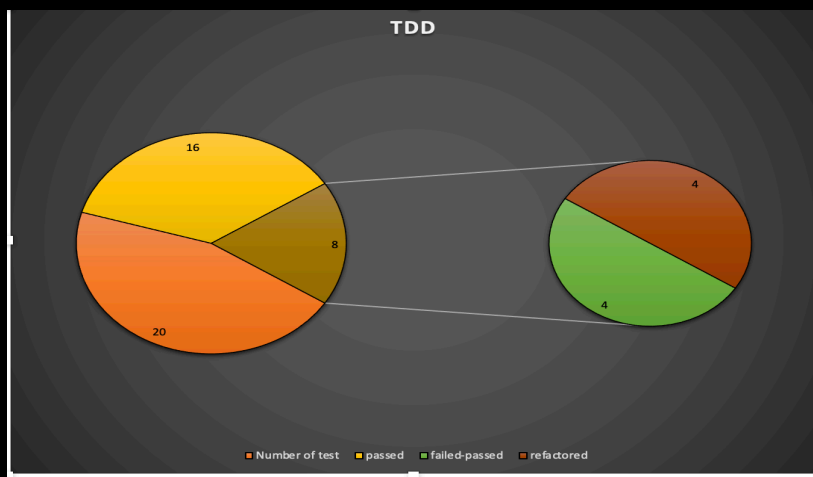
97 + @test
98 + public void testGetSetAwardStatus() {
99 +     Movie testMovie;
100 +     String title = "1917";
101 +     String category = "WARING (ORIGINAL SCREENPLAY)";
102 +     int yearReleased = 2019;
103 +     int yearNominated = 2020;
104 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
105 +     boolean awardStatus = false;
106 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
107 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
108 +     testMovie.setAwardStatus(awardStatus);
109 +     assertThat(testMovie.getAwardStatus()).contains(awardStatus);
110 + }
111 +
112 + @test
113 + public void testGetSetImageLink() {
114 +     Movie testMovie;
115 +     String title = "1917";
116 +     String category = "WARING (ORIGINAL SCREENPLAY)";
117 +     int yearReleased = 2019;
118 +     int yearNominated = 2020;
119 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
120 +     boolean awardStatus = false;
121 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
122 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
123 +     testMovie.setImageLink(imageLink);
124 +     assertThat(testMovie.getImageLink()).isEqualTo(imageLink);
125 + }
126 +
127 + @test
128 + public void testGetSetImageLink() {
129 +     Movie testMovie;
130 +     String title = "1917";
131 +     String category = "WARING (ORIGINAL SCREENPLAY)";
132 +     int yearReleased = 2019;
133 +     int yearNominated = 2020;
134 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
135 +     boolean awardStatus = false;
136 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
137 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
138 +     testMovie.setImageLink(imageLink);
139 +     assertThat(testMovie.getImageLink()).isEqualTo(imageLink);
140 + }

```

```

141 + @test
142 + public void testGetSetImageLink() {
143 +     Movie testMovie;
144 +     String title = "1917";
145 +     String category = "WARING (ORIGINAL SCREENPLAY)";
146 +     int yearReleased = 2019;
147 +     int yearNominated = 2020;
148 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
149 +     boolean awardStatus = false;
150 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
151 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
152 +     testMovie.setImageLink(imageLink);
153 +     assertThat(testMovie.getImageLink()).isEqualTo(imageLink);
154 + }
155 +
156 + @test
157 + public void testGetSetImageLink() {
158 +     Movie testMovie;
159 +     String title = "1917";
160 +     String category = "WARING (ORIGINAL SCREENPLAY)";
161 +     int yearReleased = 2019;
162 +     int yearNominated = 2020;
163 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
164 +     boolean awardStatus = false;
165 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
166 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
167 +     testMovie.setImageLink(imageLink);
168 +     assertThat(testMovie.getImageLink()).isEqualTo(imageLink);
169 + }
170 +
171 + @test
172 + public void testGetSetImageLink() {
173 +     Movie testMovie;
174 +     String title = "1917";
175 +     String category = "WARING (ORIGINAL SCREENPLAY)";
176 +     int yearReleased = 2019;
177 +     int yearNominated = 2020;
178 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
179 +     boolean awardStatus = false;
180 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
181 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
182 +     testMovie.setImageLink(imageLink);
183 +     assertThat(testMovie.getImageLink()).isEqualTo(imageLink);
184 + }
185 +
186 + @test
187 + public void testGetSetImageLink() {
188 +     Movie testMovie;
189 +     String title = "1917";
190 +     String category = "WARING (ORIGINAL SCREENPLAY)";
191 +     int yearReleased = 2019;
192 +     int yearNominated = 2020;
193 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
194 +     boolean awardStatus = false;
195 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
196 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
197 +     testMovie.setImageLink(imageLink);
198 +     assertThat(testMovie.getImageLink()).isEqualTo(imageLink);
199 + }
200 +
201 + @test
202 + public void testGetSetImageLink() {
203 +     Movie testMovie;
204 +     String title = "1917";
205 +     String category = "WARING (ORIGINAL SCREENPLAY)";
206 +     int yearReleased = 2019;
207 +     int yearNominated = 2020;
208 +     String awardee = "Written by Sam Mendes & Krysty Wilson-Cairns";
209 +     boolean awardStatus = false;
210 +     // public Movie(String title, String category, int yearReleased, int yearNominated, String awardee, boolean awardStatus)
211 +     testMovie = new Movie(title, category, yearReleased, yearNominated, awardee, awardStatus);
212 +     testMovie.setImageLink(imageLink);
213 +     assertThat(testMovie.getImageLink()).isEqualTo(imageLink);
214 + }

```



Data Analysis		
Name of Unit Test	Status of passed or fail	Refactoring
public void Movie ()	Passed	-
public void testGetSetTitle()	Passed	-
public void testGetSetCategory()	Passed	-
public void testGetSetYearReleased()	Passed	-
public void testGetSetYearNominated()	Passed	-
public void testGetSetAwardee()	Passed	-
public void testGetSetIsAwardStatus()	Passed	-
public void testGetSetImageLink()	Passed	-

public void testGetSetLink()	Fail-Passed	Refactored
public void testGetSetPlot()	Fail - Passed	Refactored
public void testGetSetTmdbId()	Passed	Refactored
public void testToString()	Passed	-
public void main()	Passed	-
Number of Commits for Unit test and Validation	5	