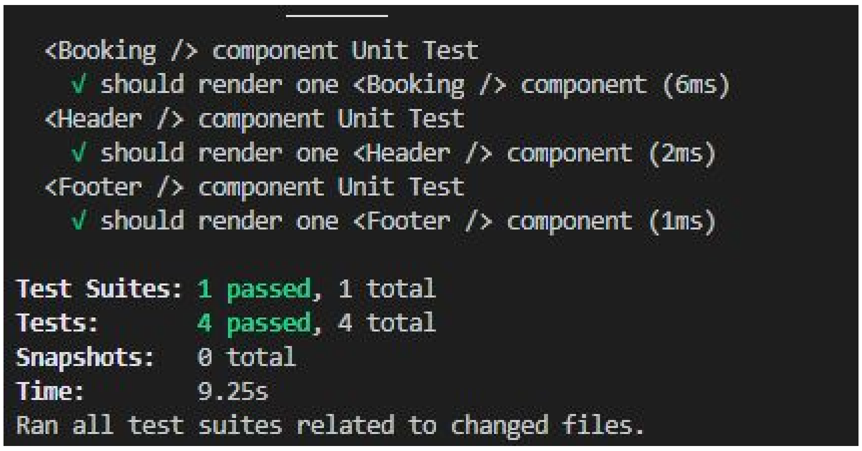
Milestone 2 - Unit Tests

Front-End

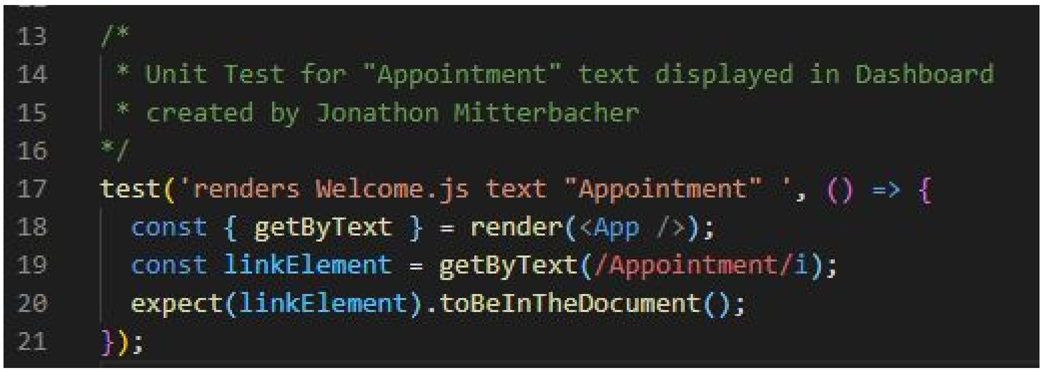
‘App.test.js’ test suite



```
  <Booking /> component Unit Test
    √ should render one <Booking /> component (6ms)
  <Header /> component Unit Test
    √ should render one <Header /> component (2ms)
  <Footer /> component Unit Test
    √ should render one <Footer /> component (1ms)

Test Suites: 1 passed, 1 total
Tests:       4 passed, 4 total
Snapshots:   0 total
Time:        9.25s
Ran all test suites related to changed files.
```

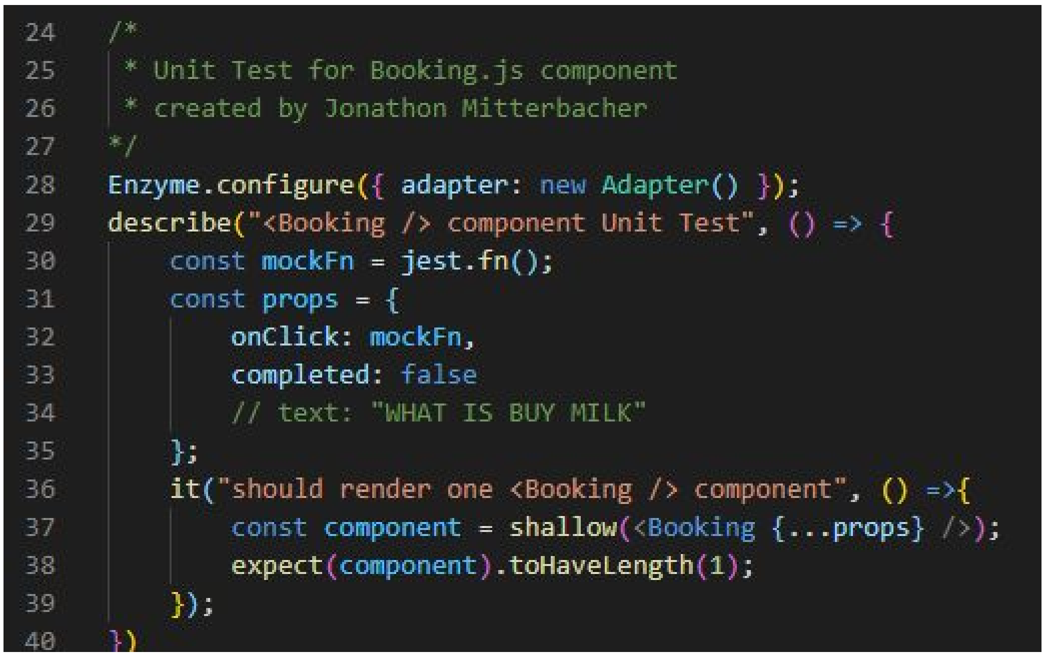
Dashboard.js unit test (s3784464)



```
/*
 * Unit Test for "Appointment" text displayed in Dashboard
 * created by Jonathon Mitterbacher
*/
test('renders Welcome.js text "Appointment" ', () => {
  const { getByText } = render(<App />);
  const linkElement = getByText(/Appointment/i);
  expect(linkElement).toBeInTheDocument();
});
```

This unit test checks if the word “Appointment” is on the /Dashboard page.

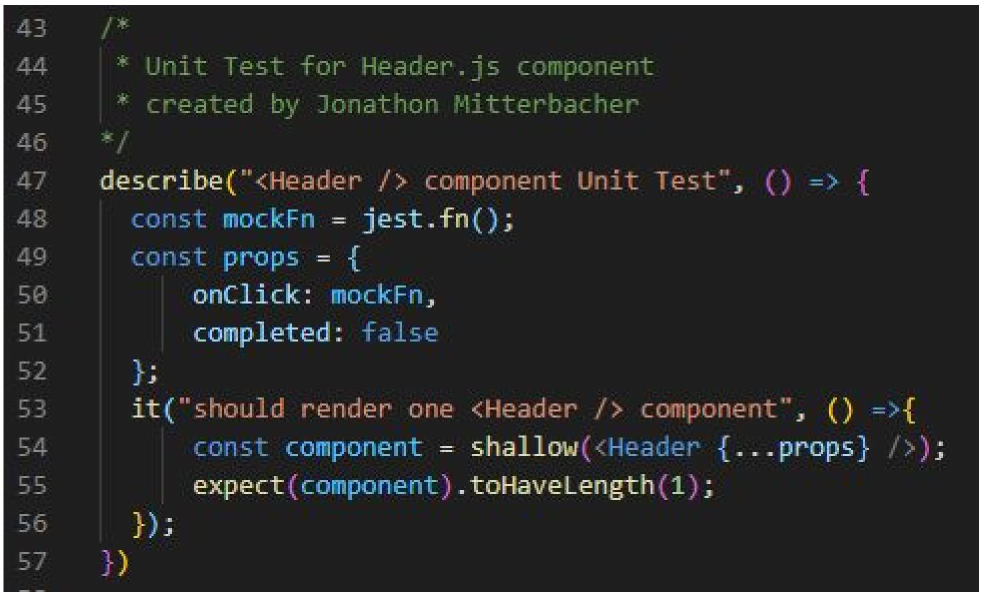
Booking.js unit test (s3784464)



```
/*
 * Unit Test for Booking.js component
 * created by Jonathon Mitterbacher
*/
Enzyme.configure({ adapter: new Adapter() });
describe("<Booking /> component Unit Test", () => {
    const mockFn = jest.fn();
    const props = {
        onClick: mockFn,
        completed: false
        // text: "WHAT IS BUY MILK"
    };
    it("should render one <Booking /> component", () =>{
        const component = shallow(<Booking {...props} />);
        expect(component).toHaveLength(1);
    });
})
```

This unit test checks whether the Booking.js component is rendered exactly once onto the /booking page.

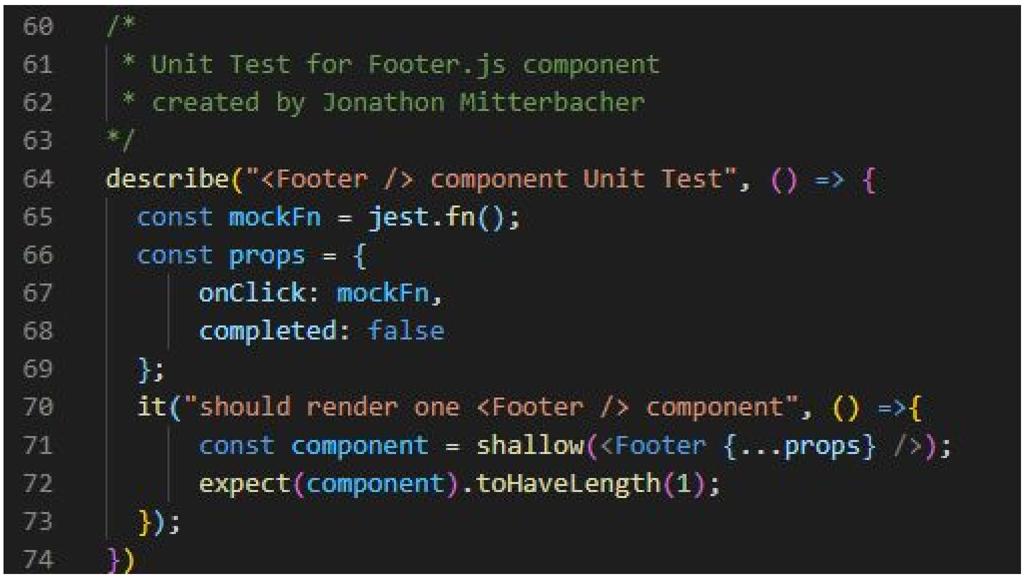
Header.js unit test (s3784464)



```
/*
 * Unit Test for Header.js component
 * created by Jonathon Mitterbacher
*/
describe("<Header /> component Unit Test", () => {
  const mockFn = jest.fn();
  const props = {
      onClick: mockFn,
      completed: false
  };
  it("should render one <Header /> component", () =>{
      const component = shallow(<Header {...props} />);
      expect(component).toHaveLength(1);
  });
})
```

This unit test checks whether the Header.js component is rendered on all pages exactly once.

Footer.js unit test (s3784464)



```
/*
 * Unit Test for Footer.js component
 * created by Jonathon Mitterbacher
*/
describe("<Footer /> component Unit Test", () => {
  const mockFn = jest.fn();
  const props = {
      onClick: mockFn,
      completed: false
  };
  it("should render one <Footer /> component", () =>{
      const component = shallow(<Footer {...props} />);
      expect(component).toHaveLength(1);
  });
})
```

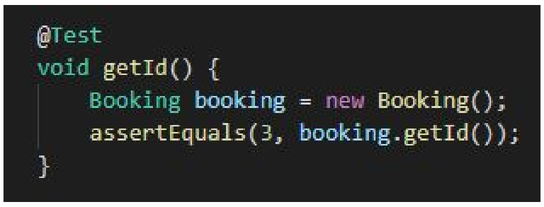
This unit test checks whether the Footer.js component is rendered on all pages exactly once.

Back-End

Steven Hoang(s3600563)

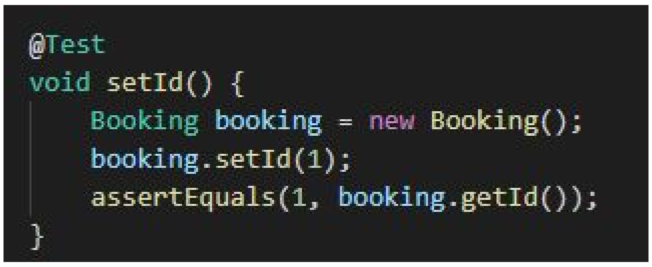
BookTest.java test suite for the backend

Testing for the getId() method:



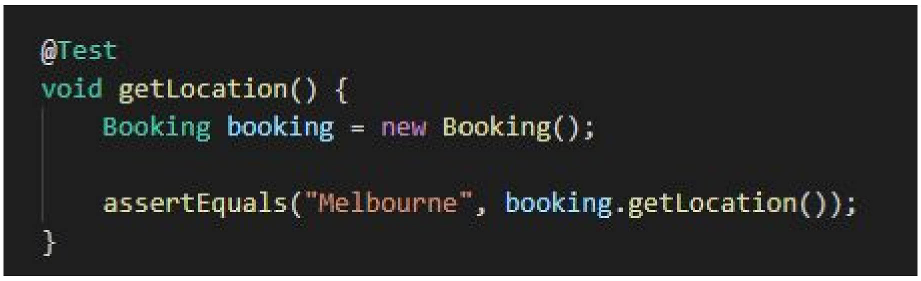
```
@Test
void getId() {
    Booking booking = new Booking();
    assertEquals(3, booking.getId());
}
```

Testing for the setId() method:



```
@Test
void setId() {
    Booking booking = new Booking();
    booking.setId(1);
    assertEquals(1, booking.getId());
}
```

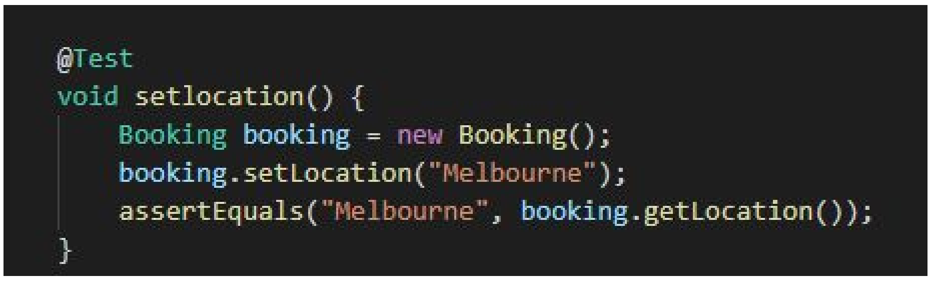
Testing the booking method for whether it grabs the correct location.



```
@Test
void getLocation() {
    Booking booking = new Booking();

    assertEquals("Melbourne", booking.getLocation());
}
```

Testing for the setLocation() method in the Booking class to see if it sets the location correctly

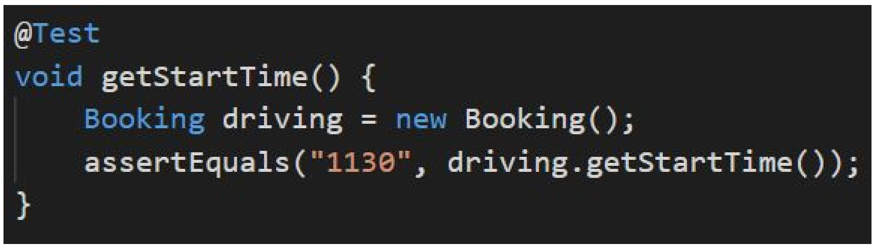


```
@Test
void setlocation() {
    Booking booking = new Booking();
    booking.setLocation("Melbourne");
    assertEquals("Melbourne", booking.getLocation());
}
```

Jonathan H(s3722092)

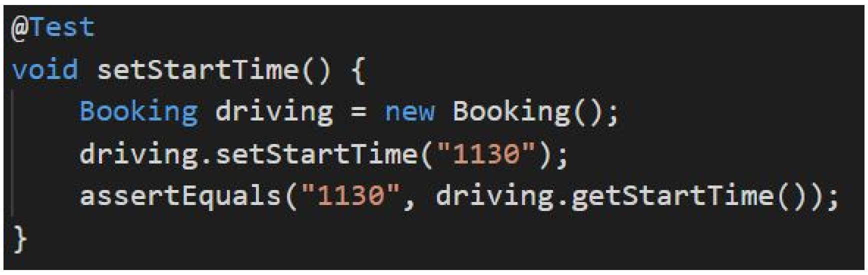
BookingTimeTest.java

Testing for the getStartTime method to see if it gets the correct time from DB



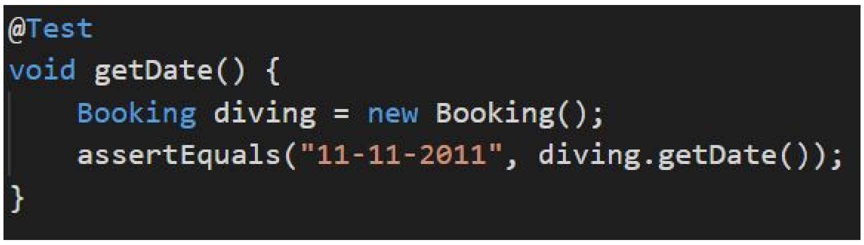
```java
@Test
void getStartTime() {
    Booking driving = new Booking();
    assertEquals("1130", driving.getStartTime());
}
```

Testing the setStartTime method to see if the time is correctly set



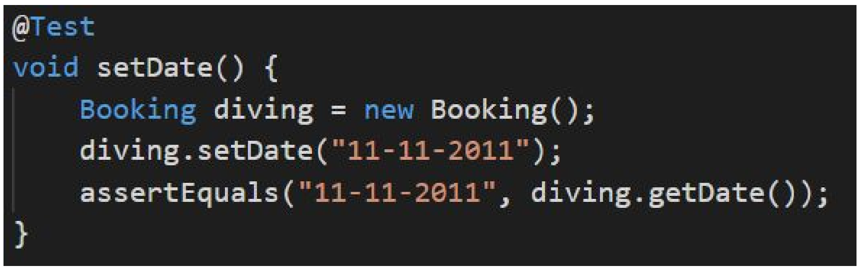
```java
@Test
void setStartTime() {
    Booking driving = new Booking();
    driving.setStartTime("1130");
    assertEquals("1130", driving.getStartTime());
}
```

Testing the getDate method to see if it gets the correct date from DB



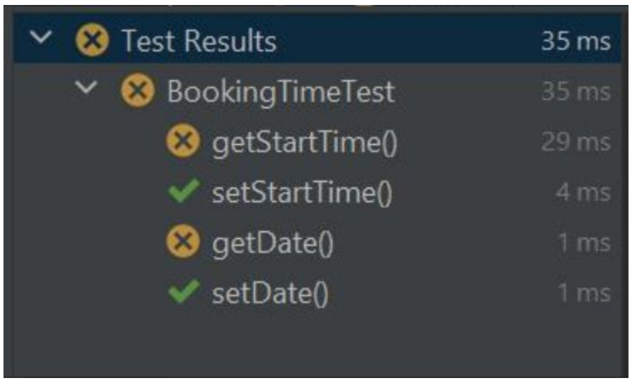
```java
@Test
void getDate() {
    Booking diving = new Booking();
    assertEquals("11-11-2011", diving.getDate());
}
```

Testing the setDate method to see if the date is correctly set



```java
@Test
void setDate() {
    Booking diving = new Booking();
    diving.setDate("11-11-2011");
    assertEquals("11-11-2011", diving.getDate());
}
```

Test results:



```
Test Results             35 ms
  BookingTimeTest        35 ms
    getStartTime()       29 ms
    setStartTime()        4 ms
    getDate()             1 ms
    setDate()             1 ms
```