Nunit + Jasmine + Karma Assignment :- 1

Write brief description about unit testing and functional testing and it's benefit in project, as a developer perspective?

Unit Testing :-

Unit Testing is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

Benefits of Unit Testing as developer perspective :-

- Makes the Process Agile
- Quality of Code
- Finds Software Bugs Early
- Facilitates Changes and Simplifies Integration
- Provides Documentation
- Debugging Process
- Design
- Reduce Costs

Functional Testing :-

Functional testing verifies that the software performs its stated functions in a way that the users expect.The process of functional testing involves a series of tests: Smoke, Sanity, Integration, Regression, Interface, System and finally User Acceptance Testing. Tests are conducted on each feature of the software to determine its behavior, using a combination of inputs simulating normal operating conditions, and deliberate anomalies and errors.

Benefits of Functional Testing as developer perspective :-

- It ensures that the customer or end- user is satisfied.
- It produces a defect-free product/software.

Nunit + Jasmine + Karma Assignment :- 1

- It ensures the all the requirements should be met.
- It ensures the proper working of all the functionalities of an application/software/product.
- It ensures that the software/product works as expected.
- It ensures security and safety.
- It improves the quality of the product.
- The risks and loss associated with the product/software reduced.

Where and why do you need unit testing in your project, give me 10 example and code snap?

1 :-

```
it('should create the app', () => {
  const fixture = TestBed.createComponent(AppComponent);
  const app = fixture.componentInstance;
  expect(app).toBeTruthy();
});
```

2 :-

```
it(`should have as title 'AngularTraining'`, () => {
   const fixture = TestBed.createComponent(AppComponent);
   const app = fixture.componentInstance;
   expect(app.title).toEqual('AngularTraining');
});
```

3 :-

```
 it('should render title', () => {
   const fixture = TestBed.createComponent(AppComponent);
   fixture.detectChanges();
   const compiled = fixture.nativeElement;
  expect(compiled.querySelector('.content          span').textContent).toContain('AngularTraining
   app is running!');
});
```

4 :-

```
  it('should change title to Angular Training', async(() => {
   const fixture = TestBed.createComponent(AppComponent);
   fixture.nativeElement.querySelector('button').click();
   fixture.detectChanges();
```

Nunit + Jasmine + Karma Assignment :- 1

```
  expect(fixture.nativeElement.querySelector('h1').textContent).toEqual('Unit Test App');
});
```

5 :-

```
 it("should create a post in an array", () => {
   const qouteText = "This is my new post";
        service.addNewQuote(qouteText);
        expect(service.quoteList.length).toBeGreaterThanOrEqual(1);
      });
```

6 :-

```
it("should remove a created post from the array of posts", () => {
    service.addNewQuote("This is my new post");
     service.removeQuote(0);
 expect(service.quoteList.length).toBeLessThan(1);
});
```

7 :-

```
  it("should create Quote component", () => {
   expect(component).toBeTruthy();
  });
```

8 :-

```
  it("should use the quoteList from the service", () => {
    const quoteService = fixture.debugElement.injector.get(QuoteService);
    fixture.detectChanges();
    expect(quoteService.getQuote()).toEqual(component.quoteList);
 });
```

9 :-

```
 it("should create a new post", () => {
    component.quoteText = "I love this test";
   fixture.detectChanges();
   const compiled = fixture.debugElement.nativeElement;
   expect(compiled.innerHTML).toContain("I love this test");
});
```

10:-

```
  it("should disable the button when textArea is empty", () => {
   fixture.detectChanges();
   const button = fixture.debugElement.query(By.css("button"));
   expect(button.nativeElement.disabled).toBeTruthy();
 });
```

Nunit + Jasmine + Karma Assignment :- 1