Online Test Application:

Technologies, Libraries and Frameworks used:

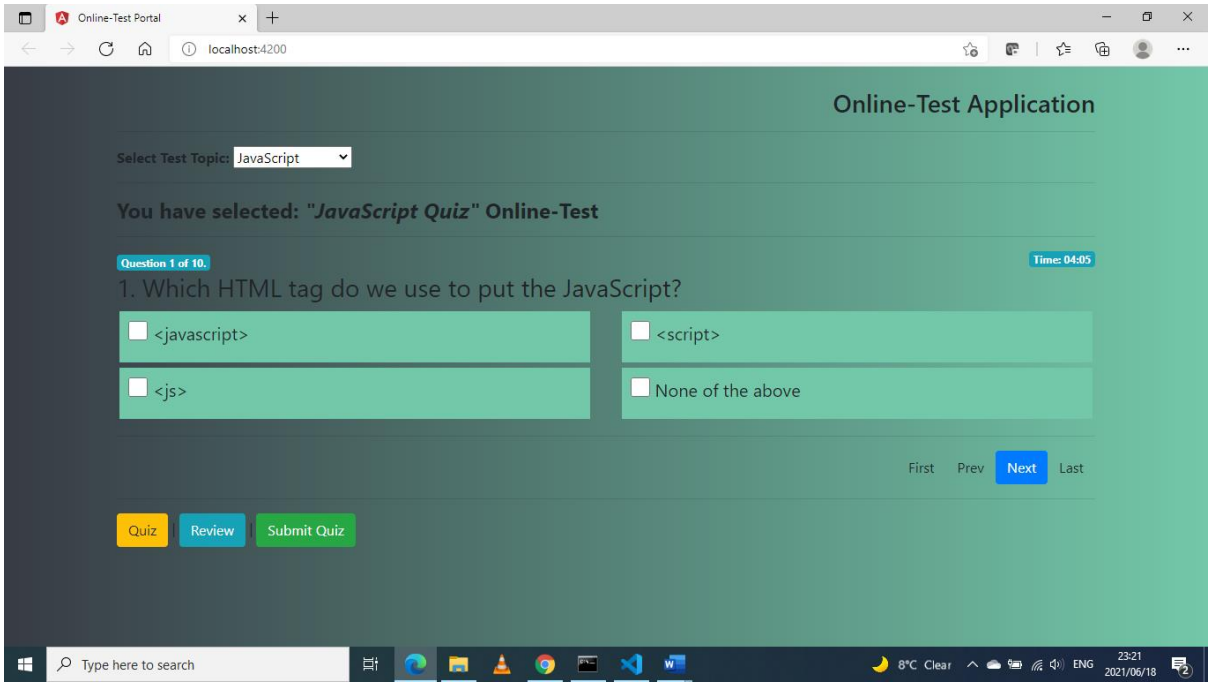
- **Visual Studio Code**
- **Angular**
- **NodeJS**
- **JSON**
- **CSS and HTML**
- **Git and Github (https://github.com/luandz89/online-test-app.git)**

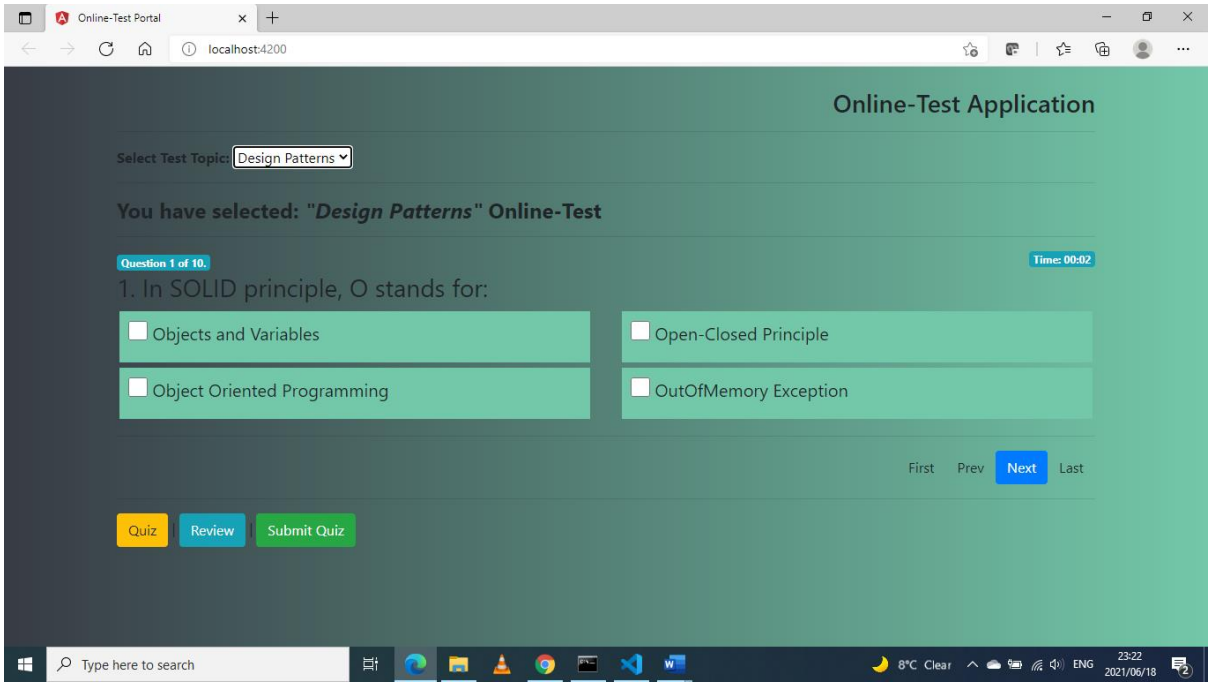
The application allows the test-takers to take the test by selecting their appropriate technology suite. Its in the form of quiz in 4 different programming languages.

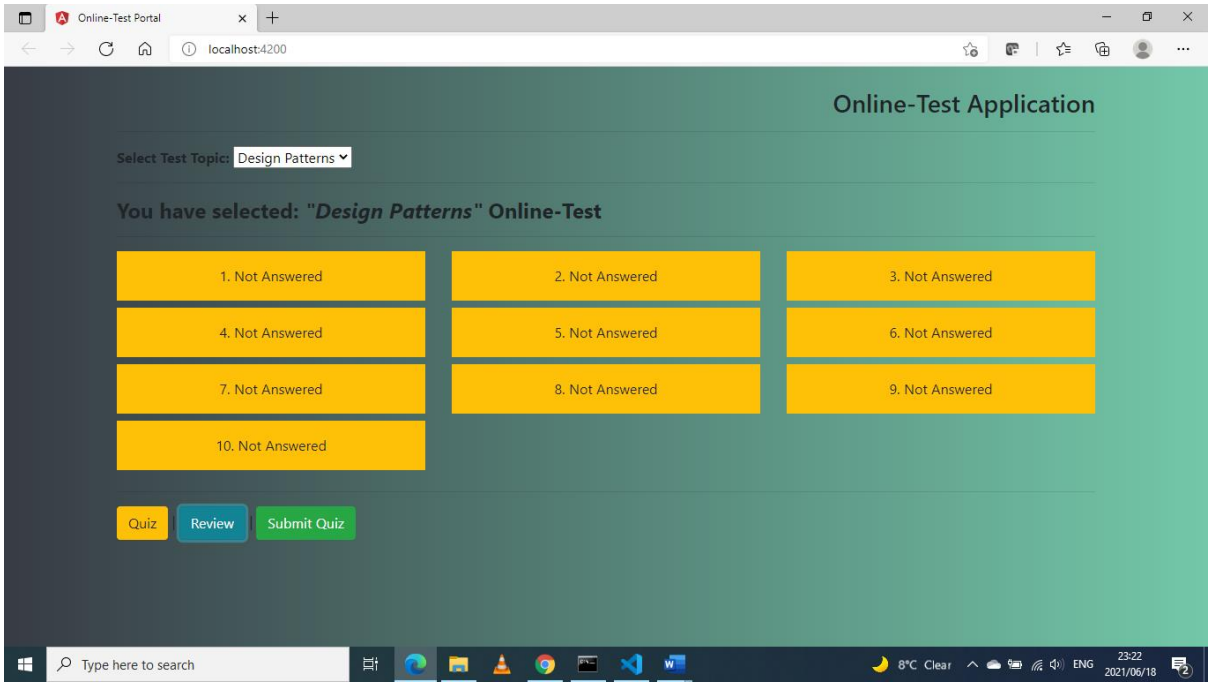
After a user finishes the test, the results are shown immediately when you click the submit button.

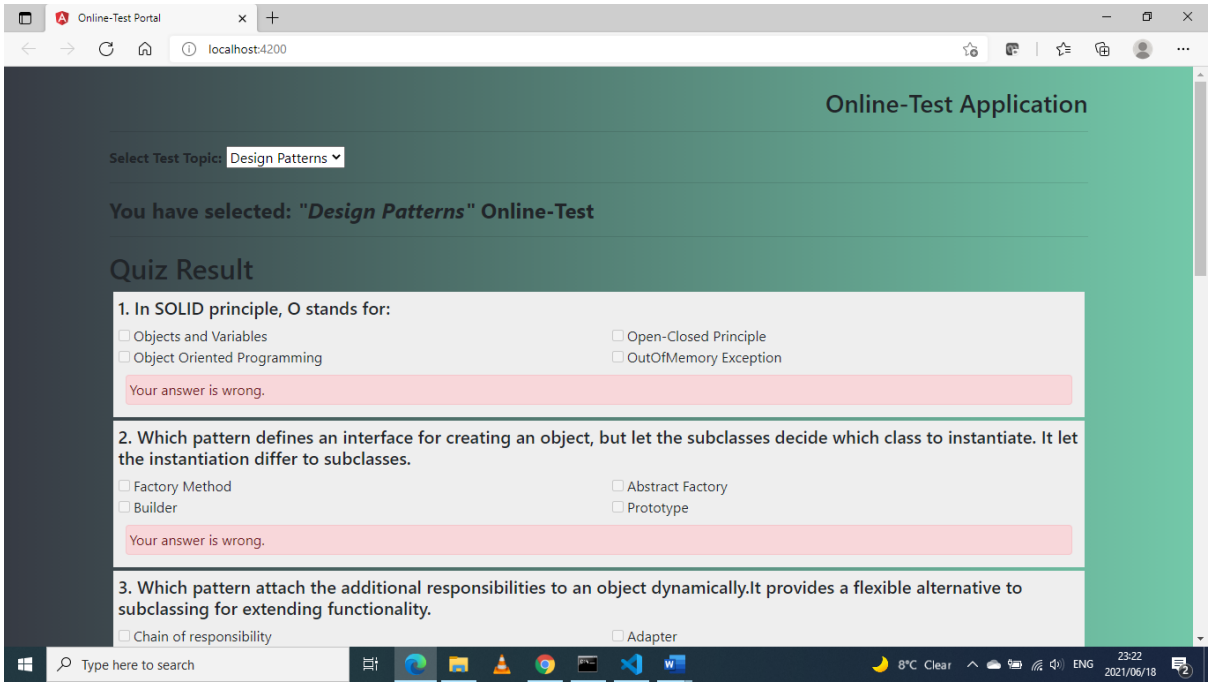
A user can also review answered and unanswered questions by clicking the review button.

Application Screenshots:






Online-Test Portal
localhost:4200
Online-Test Application
Select Test Topic: Design Patterns
You have selected: "Design Patterns" Online-Test
1. Not Answered
2. Not Answered
3. Not Answered
4. Not Answered
5. Not Answered
6. Not Answered
7. Not Answered
8. Not Answered
9. Not Answered
10. Not Answered
Quiz
Review
Submit Quiz
Type here to search
8°C Clear
ENG
23:22
2021/06/18


Online-Test Portal
localhost:4200
Online-Test Application
Select Test Topic: Design Patterns
You have selected: "Design Patterns" Online-Test
Quiz Result
1. In SOLID principle, O stands for:
Objects and Variables
Open-Closed Principle
Object Oriented Programming
OutOfMemory Exception
Your answer is wrong.
2. Which pattern defines an interface for creating an object, but let the subclasses decide which class to instantiate. It let the instantiation differ to subclasses.
Factory Method
Abstract Factory
Builder
Prototype
Your answer is wrong.
3. Which pattern attach the additional responsibilities to an object dynamically.It provides a flexible alternative to subclassing for extending functionality.
Chain of responsibility
Adapter
Type here to search
8°C Clear
ENG
23:22
2021/06/18

Project Source codes:

quiz.component.ts

```
<div class="row">
  <div class="col-12">
    <h3 class="text-right">Online-Test Application</h3>
    <hr />
  </div>
<br><br>
  <div class="col-12 text-left">

    <strong> Select Test Topic:  </strong>

    <select [(ngModel)]="quizName" (change)="loadQuiz(quizName)">
      <option *ngFor="let quiz of quizes" [value]="quiz.id">{{quiz.name}}</opt
ion>
    </select>
    <hr />
  </div>
</div>
<div id="quiz">
  <h4 class="text-
left"> <strong>You have selected: <i>"{{quiz.name}}"</i> Online-
Test</strong></h4>
  <hr />

  <div *ngIf="mode=='quiz' && quiz">
    <div *ngFor="let question of filteredQuestions;">
      <div class="badge badge-
info">Question {{pager.index + 1}} of {{pager.count}}.</div>
      <div *ngIf="config.duration" class="badge badge-info float-
right">Time: {{ellapsedTime}}</div>
      <h3 class="font-weight-normal">{{pager.index + 1}}.
        <span [innerHTML]="question.name"></span>
      </h3>
      <div class="row text-left options">
        <div class="col-6" *ngFor="let option of question.options">
          <div class="option">
            <label class="font-weight-normal" [attr.for]="option.id">
              <input id="{{option.id}}" type="checkbox" [(ngModel)]="option.se
lected" (change)="onSelect(question, option);" /> {{option.name}}
            </label>
          </div>
        </div>
      </div>
    </div>
    <hr />
    <div class="text-right">
```

```
      <button class="btn btn-
default" *ngIf="config.allowBack" (click)="goTo(0);">First</button>
      <button class="btn btn-
default" *ngIf="config.allowBack" (click)="goTo(pager.index - 1);">Prev</butto
n>
      <button class="btn btn-
primary" (click)="goTo(pager.index + 1);">Next</button>
      <button class="btn btn-
default" *ngIf="config.allowBack" (click)="goTo(pager.count - 1);">Last</butto
n>
      </div>
  </div>

  <div class="row text-center" *ngIf="mode=='review'">
    <div class="col-4 cursor-
pointer" *ngFor="let question of quiz.questions; let index = index;">
      <div (click)="goTo(index)" class="p-3 mb-
2 {{ isAnswered(question) == 'Answered'? 'bg-info': 'bg-
warning' }}">{{index + 1}}. {{ isAnswered(question) }}</div>
    </div>
  </div>
  <div class="result" *ngIf="mode=='result'">
    <h2>Quiz Result</h2>
    <div *ngFor="let question of quiz.questions; let index = index">
      <div class="result-question">
        <h5>{{index + 1}}. {{question.name}}</h5>
        <div class="row">
          <div class="col-6" *ngFor="let Option of question.options">
            <input id="{{Option.id}}" type="checkbox" disabled="disabled" [(ng
Model)]="Option.selected" /> {{Option.name}}
          </div>
        </div>
        <div class="p-1 m-2 alert {{ isCorrect(question) == 'correct'? 'alert-
success': 'alert-danger'}}">Your answer is {{isCorrect(question)}}.</div>
      </div>
    </div>

  </div>
  <hr />
  <div *ngIf="mode!='result'">
    <button class="btn btn-warning" (click)="mode = 'quiz'">Quiz</button> |
    <button class="btn btn-info" (click)="mode = 'review'">Review</button> |
    <button class="btn btn-success" (click)="onSubmit();">Submit Quiz</button>
  </div>
</div>
```

Quiz.component.spec.ts

```
/* tslint:disable:no-unused-variable */
import { async, ComponentFixture, TestBed } from '@angular/core/testing';
import { By } from '@angular/platform-browser';
import { DebugElement } from '@angular/core';

import { QuizComponent } from './quiz.component';

describe('QuizComponent', () => {
  let component: QuizComponent;
  let fixture: ComponentFixture<QuizComponent>;

  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [ QuizComponent ]
    })
    .compileComponents();
  }));

  beforeEach(() => {
    fixture = TestBed.createComponent(QuizComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

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

Quiz.component.ts

```ts
import { Component, OnInit } from '@angular/core';

import { QuizService } from '../services/quiz.service';
import { HelperService } from '../services/helper.service';
import { Option, Question, Quiz, QuizConfig } from '../models/index';

@Component({
  selector: 'app-quiz',
  templateUrl: './quiz.component.html',
  styleUrls: ['./quiz.component.css'],
  providers: [QuizService]
})
export class QuizComponent implements OnInit {
  quizes: any[];
  quiz: Quiz = new Quiz(null);
  mode = 'quiz';
  quizName: string;
  config: QuizConfig = {
    'allowBack': true,
    'allowReview': true,
    'autoMove': false,
    'duration': 300,
    'pageSize': 1,
    'requiredAll': false,
    'richText': false,
    'shuffleQuestions': false,
    'shuffleOptions': false,
    'showClock': false,
    'showPager': true,
    'theme': 'none'
  };

  pager = {
    index: 0,
    size: 1,
    count: 1
  };
  timer: any = null;
  startTime: Date;
  endTime: Date;
  ellapsedTime = '00:00';
  duration = '';
```

```
  constructor(private quizService: QuizService) { }

  ngOnInit() {
    this.quizes = this.quizService.getAll();
    this.quizName = this.quizes[0].id;
    this.loadQuiz(this.quizName);
  }

  loadQuiz(quizName: string) {
    this.quizService.get(quizName).subscribe(res => {
      this.quiz = new Quiz(res);
      this.pager.count = this.quiz.questions.length;
      this.startTime = new Date();
      this.ellapsedTime = '00:00';
      this.timer = setInterval(() => { this.tick(); }, 1000);
      this.duration = this.parseTime(this.config.duration);
    });
    this.mode = 'quiz';
  }

  tick() {
    const now = new Date();
    const diff = (now.getTime() - this.startTime.getTime()) / 1000;
    if (diff >= this.config.duration) {
      this.onSubmit();
    }
    this.ellapsedTime = this.parseTime(diff);
  }

  parseTime(totalSeconds: number) {
    let mins: string | number = Math.floor(totalSeconds / 60);
    let secs: string | number = Math.round(totalSeconds % 60);
    mins = (mins < 10 ? '0' : '') + mins;
    secs = (secs < 10 ? '0' : '') + secs;
    return `${mins}:${secs}`;
  }

  get filteredQuestions() {
    return (this.quiz.questions) ?
      this.quiz.questions.slice(this.pager.index, this.pager.index + this.page
r.size) : [];
  }

  onSelect(question: Question, option: Option) {
    if (question.questionTypeId === 1) {
      question.options.forEach((x) => { if (x.id !== option.id) x.selected = f
alse; });
    }
```

```
    if (this.config.autoMove) {
      this.goTo(this.pager.index + 1);
    }
  }

  goTo(index: number) {
    if (index >= 0 && index < this.pager.count) {
      this.pager.index = index;
      this.mode = 'quiz';
    }
  }

  isAnswered(question: Question) {
    return question.options.find(x => x.selected) ? 'Answered' : 'Not Answered
';
  };

  isCorrect(question: Question) {
    return question.options.every(x => x.selected === x.isAnswer) ? 'correct'
: 'wrong';
  };

  onSubmit() {
    let answers = [];
    this.quiz.questions.forEach(x => answers.push({ 'quizId': this.quiz.id, 'q
uestionId': x.id, 'answered': x.answered }));

    console.log(this.quiz.questions);
    this.mode = 'result';
  }
}
```

App.component.html

```
<div class="container">
  <app-quiz></app-quiz>
</div>
```

App.component.spec.ts

```
import { TestBed, async } from '@angular/core/testing';
import { AppComponent } from './app.component';
```

```
describe('AppComponent', () => {
  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [
        AppComponent
      ],
    }).compileComponents();
  }));
  it('should create the app', async(() => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.debugElement.componentInstance;
    expect(app).toBeTruthy();
  }));
  it(`should have as title 'app'`, async(() => {
    const fixture = TestBed.createComponent(AppComponent);
    const app = fixture.debugElement.componentInstance;
    expect(app.title).toEqual('app');
  }));
  it('should render title in a h1 tag', async(() => {
    const fixture = TestBed.createComponent(AppComponent);
    fixture.detectChanges();
    const compiled = fixture.debugElement.nativeElement;
    expect(compiled.querySelector('h1').textContent).toContain('Welcome to app
!');
  }));
});
```

App.component.ts

```
import { Component } from '@angular/core';

import { QuizComponent } from './quiz/quiz.component';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
}
```

App.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';

import { AppComponent } from './app.component';
import { QuizComponent } from './quiz/quiz.component';
import { HttpClientModule } from '@angular/common/http';

@NgModule({
  declarations: [
    AppComponent,
    QuizComponent
  ],
  imports: [
    BrowserModule,
    FormsModule,
    HttpClientModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Online-Test Portal</title>
  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="icon" type="image/x-icon" href="favicon.ico">
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

Main.ts

```
import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-browser-dynamic';

import { AppModule } from './app/app.module';
import { environment } from './environments/environment';

if (environment.production) {
  enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.log(err));
```

Styles.css

```
.btn{
    border-radius: 3;
}

.option{
    background-color:#73C8A9;
    font-size:20px;
    padding:10px;
    margin:3px;
}
    #quiz .options input[type=checkbox] {
        background-color: burlywood;
        height:22px;
        width:22px;
    }

.cursor-pointer{
    cursor: pointer;
}

.result-question{
    background-color:#eee;
    margin:4px;
    padding:6px;
```

```
}

@media only screen and (max-width: 480px) {
    h1,.h1{
        font-size:22px;
    }
    h2,.h2{
        font-size:20px;
    }
    h3,.h3{
        font-size:18px;
    }
    .option {
        font-size: 16px;
        padding: 6px;
    }
}

body {
    background: #73C8A9;  /* fallback for old browsers */
background: -webkit-linear-
gradient(to right, #373B44, #73C8A9);  /* Chrome 10-25, Safari 5.1-6 */
background: linear-
gradient(to right, #373B44, #73C8A9); /* W3C, IE 10+/ Edge, Firefox 16+, Chrom
e 26+, Opera 12+, Safari 7+ */

}
```

Angular.json

```
{
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
  "cli": {
    "analytics": false
  },
  "version": 1,
  "newProjectRoot": "projects",
  "projects": {
    "online-test-app": {
      "root": "",
      "sourceRoot": "src",
      "projectType": "application",
      "prefix": "app",
      "schematics": {},
      "architect": {
```

```
      "build": {
        "builder": "@angular-devkit/build-angular:browser",
        "options": {
          "outputPath": "dist/online-test-app",
          "index": "src/index.html",
          "main": "src/main.ts",
          "polyfills": "src/polyfills.ts",
          "tsConfig": "src/tsconfig.app.json",
          "assets": [
            "src/favicon.ico",
            "src/data",
            "src/assets"
          ],
          "styles": [
            "node_modules/bootstrap/dist/css/bootstrap.css",
            "src/styles.css"
          ],
          "scripts": [
            "node_modules/jquery/dist/jquery.min.js",
            "node_modules/bootstrap/dist/js/bootstrap.bundle.js"
          ]
        },
        "configurations": {
          "production": {
            "fileReplacements": [
              {
                "replace": "src/environments/environment.ts",
                "with": "src/environments/environment.prod.ts"
              }
            ],
            "optimization": true,
            "outputHashing": "all",
            "sourceMap": false,
            "extractCss": true,
            "namedChunks": false,
            "aot": true,
            "extractLicenses": true,
            "vendorChunk": false,
            "buildOptimizer": true
          }
        }
      },
      "serve": {
        "builder": "@angular-devkit/build-angular:dev-server",
        "options": {
          "browserTarget": "online-test-app:build"
        },
        "configurations": {
```

```
          "production": {
            "browserTarget": "online-test-app:production"
          }
        }
      },
      "extract-i18n": {
        "builder": "@angular-devkit/build-angular:extract-i18n",
        "options": {
          "browserTarget": "online-test-app:build"
        }
      },
      "test": {
        "builder": "@angular-devkit/build-angular:karma",
        "options": {
          "main": "src/test.ts",
          "polyfills": "src/polyfills.ts",
          "tsConfig": "src/tsconfig.spec.json",
          "karmaConfig": "src/karma.conf.js",
          "styles": [
            "styles.css"
          ],
          "scripts": [],
          "assets": [
            "src/favicon.ico",
            "src/assets"
          ]
        }
      },
      "lint": {
        "builder": "@angular-devkit/build-angular:tslint",
        "options": {
          "tsConfig": [
            "src/tsconfig.app.json",
            "src/tsconfig.spec.json"
          ],
          "exclude": [
            "**/node_modules/**"
          ]
        }
      }
    }
  },
  "online-test-app-e2e": {
    "root": "e2e/",
    "projectType": "application",
    "architect": {
      "e2e": {
        "builder": "@angular-devkit/build-angular:protractor",
```

```
          "options": {
            "protractorConfig": "e2e/protractor.conf.js",
            "devServerTarget": "online-test-app:serve"
          }
        },
        "lint": {
          "builder": "@angular-devkit/build-angular:tslint",
          "options": {
            "tsConfig": "e2e/tsconfig.e2e.json",
            "exclude": [
              "**/node_modules/**"
            ]
          }
        }
      }
    }
  },
  "defaultProject": "online-test-app"
}
```

.gitignore

```
# dependencies
/node_modules
/.pnp
.pnp.js

# testing
/coverage

# production
/build

# misc
.DS_Store
.env.local
.env.development.local
.env.test.local
.env.production.local

npm-debug.log*
yarn-debug.log*
yarn-error.log*
```