Angular Application for Dynamic Question Ordering

# Objective

The objective is to build a screen that allows users to set a list of parent questions which can be ordered from 1st to last. Based on the selected response (parent question may include radio button, textbox, singleselect), the screen will display a set of child questions. The logic for displaying parent-child behavior is consistent across all question sets. Display selected results in summary page.

# Project structure

src

├── app

│ ├── components

│ │ ├── questionnaire

│ │ │ ├── questionnaire.component.ts

│ │ │ ├── questionnaire.component.html

│ │ │ ├── questionnaire.component.css

│ │ │ └── questionnaire.component.spec.ts

│ │ ├── child-question

│ │ │ ├── child-question.component.ts

│ │ │ ├── child-question.component.html

│ │ │ ├── child-question.component.css

│ │ │ └── child-question.component.spec.ts

│ │ ├── question

│ │ │ ├── question.component.ts

│ │ │ ├── question.component.html

│ │ │ ├── question.component.css

│ │ │ └── question.component.spec.ts

│ │ ├── Summary

│ │ │ ├── sumamry.component.ts

│ │ │ ├── sumamry.component.html

│ │ │ ├── sumamry.component.css

│ │ │ └── sumamry.component.spec.ts

│ ├── services

│ │ ├── question.service.ts

│ ├── models

│ │ ├── question.model.ts

├── assets

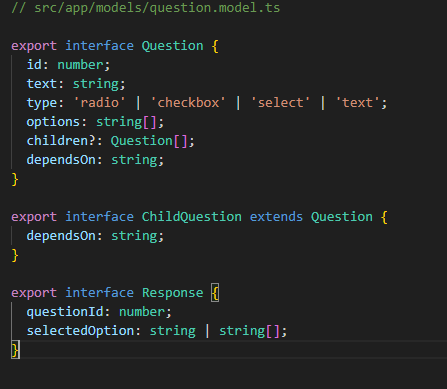
│ ├── questions.json

│ ├── app.component.ts

│ ├── app.module.ts

│ └── app-routing.module.ts

# Model



# Services



# Components

### Questionnaire component

// src/app/components/questionnaire/questionnaire.component.ts

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

import { Router } from '@angular/router';

import { QuestionService } from '../../services/question.service';

import { Question } from '../../models/question.model';

@Component({

  selector: 'app-questionnaire',

  templateUrl: './questionnaire.component.html',

  styleUrls: ['./questionnaire.component.css']

})

export class QuestionnaireComponent implements OnInit {

  questions: Question[] = [];

  constructor(private questionService: QuestionService, private router: Router) { }

  ngOnInit(): void {

    // Subscribe to questions from service

    this.questionService.questions$.subscribe(questions => {

      this.questions = questions;

    });

  }

  /\*\*

   \* Method to handle response changes from child components

   \* @param response The response object containing questionId and selectedOption

   \*/

  onResponseChange(response: any) {

    this.questionService.setResponse(response); // Subscribe to questions from service

  }

  /\*\*

   \* Navigates to the summary page with collected responses

   \*/

  goToSummary() {

    // Pass responses to summary page via service

    this.router.navigate(['/summary']); // Navigate to SummaryComponent

  }

  /\*\*

   \* Moves a question up in the list based on direction

   \* @param index The question to be moved based on index

   \*/

  moveUp(index: number) {

    if (index > 0) {

      [this.questions[index], this.questions[index - 1]] = [this.questions[index - 1], this.questions[index]];

      this.questionService.setQuestions(this.questions);

    }

  }

  /\*\*

   \* Moves a question down in the list based on direction

   \* @param index The question to be moved based on index

   \*/

  moveDown(index: number) {

    if (index < this.questions.length - 1) {

      [this.questions[index], this.questions[index + 1]] = [this.questions[index + 1], this.questions[index]];

      this.questionService.setQuestions(this.questions);

    }

  }

}

### Child-question component

// src/app/components/child-questions/child-questions.component.ts

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

import { Question } from '../../models/question.model';

import { QuestionService } from '../../services/question.service';

import { Response } from '../../models/question.model';

@Component({

  selector: 'app-child-questions',

  templateUrl: './child-questions.component.html',

  styleUrls: ['./child-questions.component.css']

})

export class ChildQuestionsComponent implements OnInit {

  @Input() parentQuestion!: Question; // Input property for parent question

  childQuestions: Question[] = [];

  constructor(private questionService: QuestionService) { }

  ngOnInit(): void {

    this.questionService.responses$.subscribe(responses => {

      const response = responses.find(r => r.questionId === this.parentQuestion.id);

      if (response) {

        this.childQuestions = this.parentQuestion.children?.filter(child => child.dependsOn === response.selectedOption) || [];

      } else {

        this.childQuestions = [];

      }

    });

  }

  /\*\*

   \* Method to handle response changes from child components

   \* @param response The response object containing questionId and selectedOption

   \*/

  onResponseChange(response: Response) {

    this.questionService.setResponse(response);

  }

}

### Question component

// src/app/components/question/question.component.ts

import { Component, Input, Output, EventEmitter } from '@angular/core';

import { Question, Response } from '../../models/question.model';

@Component({

  selector: 'app-question',

  templateUrl: './question.component.html',

  styleUrls: ['./question.component.css']

})

export class QuestionComponent {

  @Input() question!: Question; // Input property for parent question

  @Output() responseChange = new EventEmitter<Response>(); // Event emitter for response changes

  selectedOption: string | string[] = '';

  /\*\*

   \* Emits the response change event when the option is selected or text is input

   \* @param selectedOption The selected option or text input

   \*/

  onOptionChange(event: Event) {

    if (this.question.type === 'radio' || this.question.type === 'select' || this.question.type === 'text') {

      const target = event.target as HTMLInputElement | HTMLSelectElement;

      this.selectedOption = target.value;

    } else if (this.question.type === 'checkbox') {

      const target = event.target as HTMLInputElement;

      if (target.checked) {

        this.selectedOption = Array.isArray(this.selectedOption) ? [...this.selectedOption, target.value] : [target.value];

      } else {

        this.selectedOption = Array.isArray(this.selectedOption) ? this.selectedOption.filter(item => item !== target.value) : [];

      }

    }

    this.responseChange.emit({ questionId: this.question.id, selectedOption: this.selectedOption });

  }

}

### Summary component

// src/app/components/summary/summary.component.ts

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

import { QuestionService } from '../../services/question.service';

import { Response } from '../../models/question.model';

@Component({

  selector: 'app-summary',

  templateUrl: './summary.component.html',

  styleUrls: ['./summary.component.css']

})

export class SummaryComponent implements OnInit {

  responses: Response[] = [];

  constructor(private questionService: QuestionService) { }

  ngOnInit(): void {

    // Get responses from service initially

    this.questionService.responses$.subscribe(responses => {

      this.responses = responses; // Update responses on change

      console.log("this.questions::", this.responses);

    });

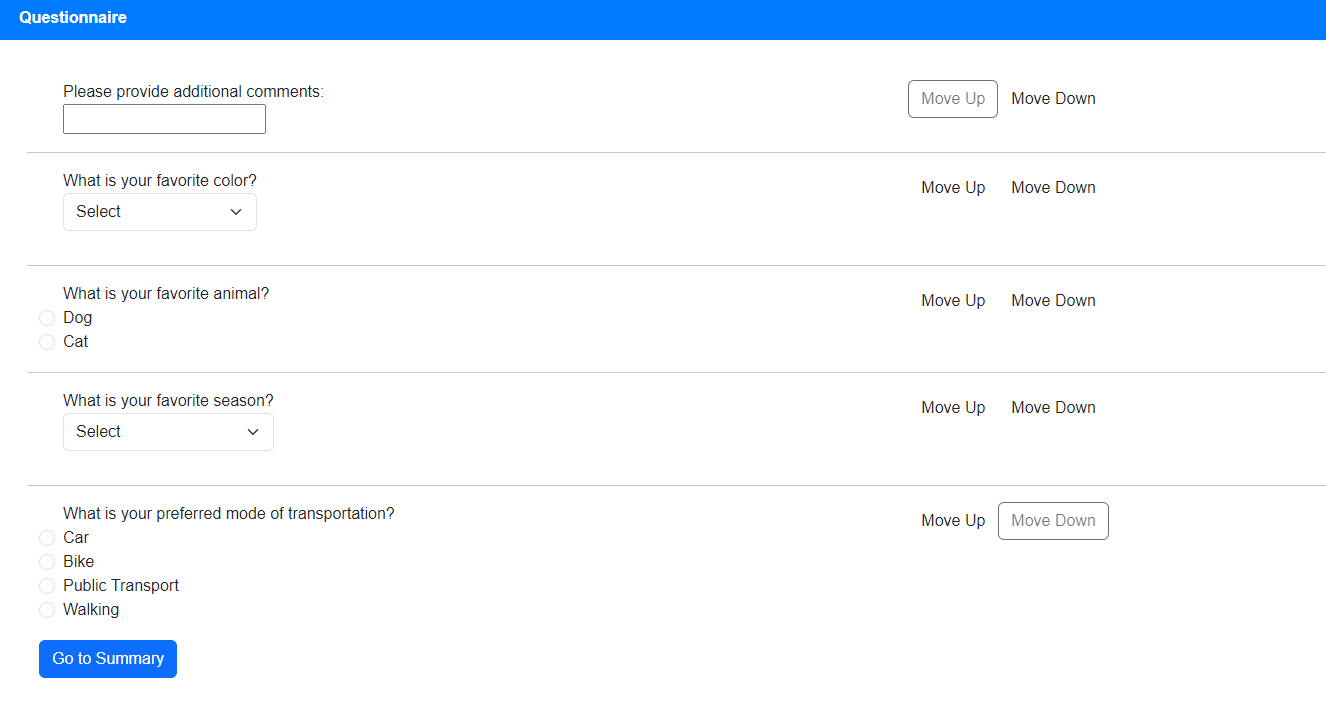
  }

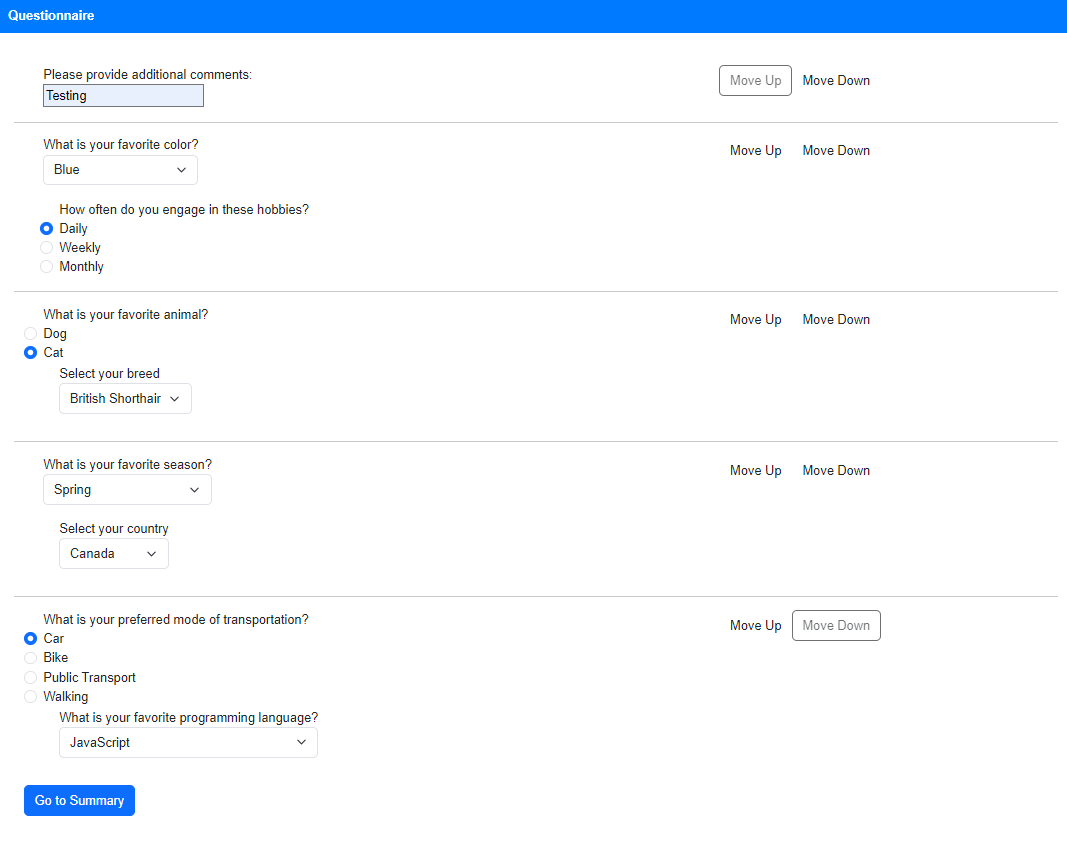
}

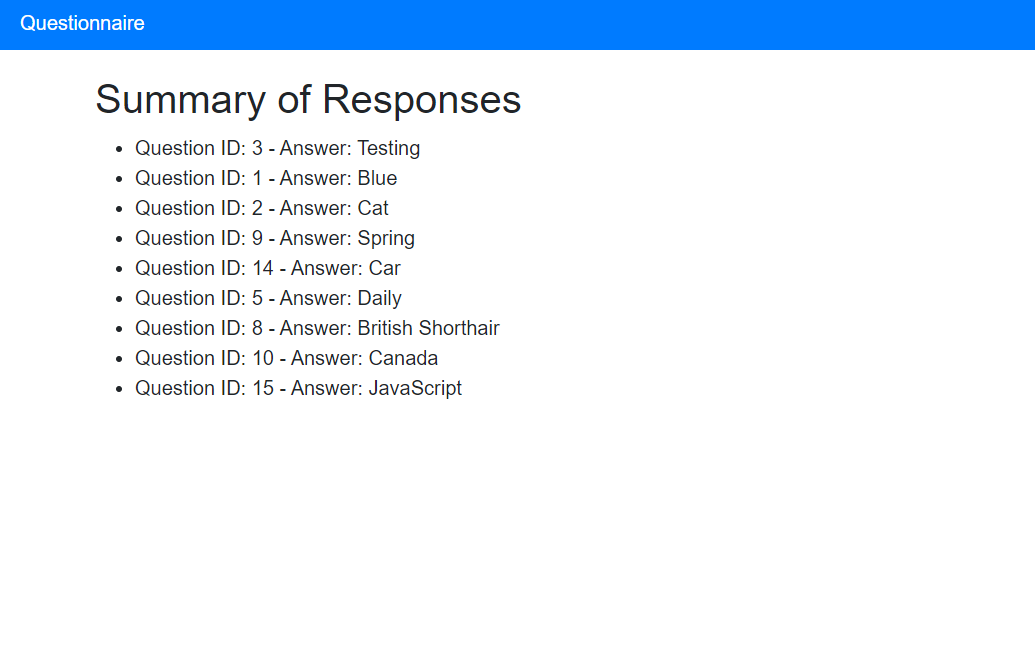
# Implementation

I have used the BehaviorSubject, Input and output. Using bootstrap implemented the simple UI screen for form and summary.

# UI screens







# Questions json object

[

  {

    "id": 3,

    "text": "Please provide additional comments:",

    "type": "text"

  },

  {

    "id": 1,

    "text": "What is your favorite color?",

    "type": "select",

    "options": ["Red", "Blue", "Green"],

    "children": [

      {

        "id": 4,

        "text": "Why do you like this color?",

        "type": "select",

        "options": ["It's calming", "It's vibrant", "It's unique"],

        "dependsOn": "Red"

      },

      {

        "id": 5,

        "text": "How often do you engage in these hobbies?",

        "type": "radio",

        "options": ["Daily", "Weekly", "Monthly"],

        "dependsOn": "Blue"

      },

      {

        "id": 6,

        "text": "Why do you like Green?",

        "type": "text",

        "dependsOn": "Green"

      }

    ]

  },

  {

    "id": 2,

    "text": "What is your favorite animal?",

    "type": "radio",

    "options": ["Dog", "Cat"],

    "children": [

      {

        "id": 7,

        "text": "Why do you like Dogs?",

        "type": "text",

        "dependsOn": "Dog"

      },

      {

        "id": 8,

        "text": "Select your breed",

        "type": "select",

        "options": ["Siamese", "British Shorthair", "Persian"],

        "dependsOn": "Cat"

      }

    ]

  },

  {

    "id": 9,

    "text": "What is your favorite season?",

    "type": "select",

    "options": ["Spring", "Summer", "Fall", "Winter"],

    "children": [

      {

        "id": 10,

        "text": "Select your country",

        "type": "select",

        "options": ["USA", "Canada", "Mexico"],

        "dependsOn": "Spring"

      },

      {

        "id": 11,

        "text": "Why do you like Summer?",

        "type": "text",

        "dependsOn": "Summer"

      },

      {

        "id": 12,

        "text": "Why do you like Fall?",

        "type": "text",

        "dependsOn": "Fall"

      },

      {

        "id": 13,

        "text": "Why do you like Winter?",

        "type": "text",

        "dependsOn": "Winter"

      }

    ]

  },

  {

    "id": 14,

    "text": "What is your preferred mode of transportation?",

    "type": "radio",

    "options": ["Car", "Bike", "Public Transport", "Walking"],

    "children": [

      {

        "id": 15,

        "text": "What is your favorite programming language?",

        "type": "select",

        "options": ["JavaScript", "Python", "Java", "C++"],

        "dependsOn": "Car"

      },

      {

        "id": 16,

        "text": "Why do you prefer Bike?",

        "type": "text",

        "dependsOn": "Bike"

      },

      {

        "id": 17,

        "text": "Why do you prefer Public Transport?",

        "type": "text",

        "dependsOn": "Public Transport"

      },

      {

        "id": 18,

        "text": "Why do you prefer Walking?",

        "type": "text",

        "dependsOn": "Walking"

      }

    ]

  }

]