Project 02- Online Test Application

This project is also hosted on GitHub repository

Link : https://github.com/zvdas/online-test-app-angular.git

https://github.com/zvdas/online-test-app-angular

To create a new angular project in a specific folder, open terminal inside that folder and type

ng new OnlineTestApp

Would you like to add Angular routing? (y/N) ***y***

Which stylesheet format would you like to use? (Use arrow keys)

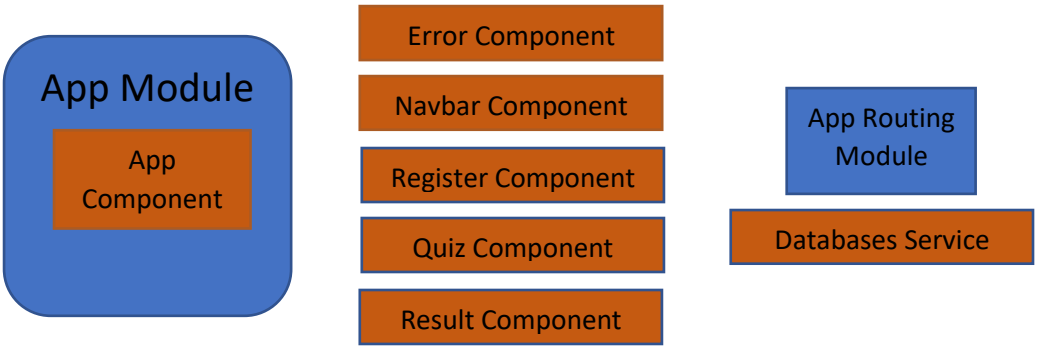
> CSS

to execute an angular project

ng serve

An angular app page will be created on localhost:4200. Open in browser.

Image shows how all the components are composed together to form the online test app.



A json server is created with 10 questions. (A json linter can be used or a csv file to json online converter)

quiz.json
{ "Quiz": [{

```
  "id": 1,
  "question": "HTML stands for?",
  "options": [
   "Hyper Text Markup Language",
   "High Text Markup Language",
   "Hyper Tabular Markup Language",
   "None of these"
  ],
  "answer": "A",
  "ansindex": 0
 },
 {
  "id": 2,
  "question": "Which of the following tag is used to mark a beginning of paragraph ?",
  "options": [
   "<TD>",
   "<br>",
   "<P>",
   "<TR>"
  ],
  "answer": "C",
  "ansindex": 2
 },
 {
  "id": 3,
  "question": "From which tag descriptive list starts ?",
  "options": [
   "<LL>",
   "<DD>",
   "<DL>",
   "<DS>"
  ],
  "answer": "C",
  "ansindex": 2
 },
 {
  "id": 4,
  "question": "Correct HTML tag for the largest heading is",
  "options": [
   "<head>",
   "<h6>",
   "<heading>",
   "<h1>"
  ],
  "answer": "D",
  "ansindex": 3
 },
 {
```

```
  "id": 5,
  "question": "The attribute of <form> tag",
  "options": [
   "Method",
   "Action",
   "Both (a)&(b)",
   "None of these"
  ],
  "answer": "C",
  "ansindex": 2
 },
 {
  "id": 6,
  "question": "Markup tags tell the web browser",
  "options": [
   "How to organise the page",
   "How to display the page",
   "How to display message box on page",
   "None of these"
  ],
  "answer": "B",
  "ansindex": 1
 },
 {
  "id": 7,
  "question": "www is based on which model?",
  "options": [
   "Local-server",
   "Client-server",
   "3-tier",
   "None of these"
  ],
  "answer": "B",
  "ansindex": 1
 },
 {
  "id": 8,
  "question": "What are Empty elements and is it valid?",
  "options": [
   "No, there is no such terms as Empty Element",
   "Empty elements are element with no data",
   "No, it is not valid to use Empty Element",
   "None of these"
  ],
  "answer": "B",
  "ansindex": 1
 },
 {
```

```
    "id": 9,
    "question": "Which of the following attributes of text box control allow to limit the maximum character?",
    "options": [
     "size",
     "len",
     "maxlength",
     "all of these"
    ],
    "answer": "C",
    "ansindex": 2
   },
   {
    "id": 10,
    "question": "Web pages starts with which of the following tag?",
    "options": [
     "<Body>",
     "<Title>",
     "<HTML>",
     "<Form>"
    ],
    "answer": "C",
    "ansindex": 2
   }
  ]
}
```


The components are created as shown in the diagram above. Create a pipe for indexing the quiz to display individual results per page. Also install a json server to host the databases for participants, the quiz questions and answers and participant answers. In the quiz ts file, while posting the participant answers to the database, a formula will be used to check whether the answers are correct or incorrect for each question.

ng g component register

ng g c navbar

ng g c quiz

ng g c result

ng g c error

ng g service services/databases

ng g pipes pipes/StringToNumber

npm install --save json-server

json-server --watch --port 3000 database-files /participant.json

json-server --watch –port 4000 database-files/quiz.json

json-server --watch --port 5000 database-files /answer.json

ng add @angular/material

npm install --save bootstrap

npm install ngx-countdown –save

adding some styles to the sheet to make it aesthetically pleasing and interactive.

styles.css

```
input.ng-invalid.ng-dirty{
   border-bottom-color : red !important;
   box-shadow: 0 1px 0 0 #e91e63 !important;
}
button.btn-submit{
   width: 100%;
}
ul.answer li:hover{
   background-color: #26a69a;
   color: #eafaf9;
   cursor: pointer;
}
ul li.selected{
   background-color: #26a69a !important;
   color: #eafaf9 !important;
}
nav{
   position: sticky;
   top: 0;
   z-index: 100;
}
body {
   background-color: #607d8bff;
}
```

index.html

```
  <!-- Compiled and minified CSS -->
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/css/materialize.min.css">
  <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
</head>
<body class="mat-typography">
  <app-root></app-root>
  <!-- Compiled and minified JavaScript -->
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/materialize/1.0.0/js/materialize.min.js"></script>
```

Add routes to the app routing module so that links clicked can redirect to required pages as well as blank and wildcard for empty pages. Export the app routing module to app module so that the routes function properly. Also add router outlet tag to the app html component so that all the children components can be added to the same page.

app-routing.module.ts

```
const routes: Routes = [
  {path:'register',component:RegisterComponent},
  {path:'quiz',component:QuizComponent},
  {path:'result',component:ResultComponent},
  {path:'review',component:RegisterComponent},
  {path:'',redirectTo:'/register',pathMatch:'full'},
  {path:'**',component:ErrorComponent}
];
@NgModule({
  imports: [
    RouterModule.forRoot(routes),
    HttpClientModule
  ],
  exports: [RouterModule]
})
```

app.module.ts

```
@NgModule({
  declarations: [
    AppComponent,
    RegisterComponent,
    NavbarComponent,
    QuizComponent,
    ResultComponent,
    ReviewComponent,
    PageNotFoundComponent,
    StringToNumberPipe
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    BrowserAnimationsModule,
    MatIconModule,
    MatListModule,
    MatButtonModule,
    CountdownModule,
    HttpClientModule,
    FormsModule,
    ReactiveFormsModule
  ],
  exports:   [NavbarComponent],
  providers: [DbConnectService],
  bootstrap: [AppComponent]
})

export class AppModule { }
```

app.component.html

```
<router-outlet></router-outlet>
```

Create a ts (TypeScript) file for each of the json files to be used as databases, so that the classes to be exported can be mapped to the object (i.e.) define name, email, username and password for participant.

participant.ts

```
export class participant
{
   name:string;
   email:string;
   username:any;
   password:any;

   constructor(name:string,email:string,username:any,password:any)
   {
      this.name=name;
      this.email=email;
      this.username=username;
      this.password=password;
   }
}
```

quiz.ts

```
export class quiz
{
   static answer(answer: any) {
     throw new Error('Method not implemented.');
   }
   id:number;
   question:string;
   options:string[]=[];
   answer:string;
   ansindex:number;

   constructor(id: number,question: string,options:string[],answer: string,ansindex:number)
   {
      this.id=id;
      this.question=question;
      this.options=options;
      this.answer=answer;
      this.ansindex=ansindex;
   }
}
```

answer.ts

```
export class answer
{
   selectedOption:string;
   answerStatus:string;
   answerScore:number;

   constructor(selectedOption:string,answerStatus:string,answerScore:number)
   {
      this.selectedOption =  selectedOption;
      this.answerStatus   =  answerStatus;
      this.answerScore    =  answerScore;
   }
}
```

Define functions to send and retrieve fields to and from the databases using Http client.

databases.service.ts

```
export class DatabasesService {
  participantServer='http://localhost:3000/Participant';
  quizServer='http://localhost:4000/Quiz';
  answerServer='http://localhost:5000/Answer';
  constructor(private hc:HttpClient) { }
  sendParticipantDetails(newParticipant:participant)
  {
    this.hc.post(this.participantServer,newParticipant).subscribe(
      (response)=>  console.log(response),
      (error)=>     console.log(error),
      ()=>          console.log("completed")
      )
  }

  getParticipants()
  {
    return this.hc.get<participant[]>(this.participantServer);
  }

  getQuestions()
  {
    return this.hc.get<quiz[]>(this.quizServer);
  }

  sendParticipantAnswers(newParticipantAnswer:answer)
  {
    return this.hc.post(this.answerServer,newParticipantAnswer).subscribe(
      (response)=>  console.log(response),
      (error)=>     console.log(error),
      ()=>          console.log("completed")
    )
  }

  getParticipantAnswers()
  {
    return this.hc.get<answer[]>(this.answerServer);
  }
}
```

Create the register component using reactive and angular forms to enable one way communication from model to view so that data fed into the registration form can be sent to the database and create a function to route the webpage to quiz once the “Start quiz” button is clicked.

register.component.ts

```
export class RegisterComponent implements OnInit {

 form!: FormGroup;

 constructor(private db:DatabasesService, private router: Router) { }

 ngOnInit(): void {

  this.form=new FormGroup(
   {
    'fullname':new FormControl("",Validators.required),
    'emailid':new FormControl("",Validators.email),
    'uname':new FormControl("",Validators.required),
    'pword':new FormControl("",Validators.required)
   }
  );

 }

 onClickSubmit()
 {
  this.db.sendParticipantDetails(this.form.value);
  localStorage.clear();
  localStorage.setItem('participant',JSON.stringify(this.form.value));
  this.router.navigate(['/quiz']);
 }

}
```

register.component.html

```
<div class="row">
   <div class="col s6 offset-s3">
      <div class="card">
         <div class="col s6">
            <h2>Quiz Registration</h2>
         </div>
         <div class="col s6">
            <img class="offset-s3" src="/assets/images/logo.png" style="width:250px;height:250px;">
         </div>
         <div class="card-content">
            <form [formGroup]="form" (ngSubmit)="onClickSubmit()">
               <div class="row">
                  <div class="input-field col s12">
                     <mat-icon class="material-icons prefix">account_circle</mat-icon>
```

```
                <input type="text" id="fullname" formControlName="fullname">
                <label>Full Name</label>
              </div>
            </div>
            <div class="row">
              <div class="input-field col s12">
                <mat-icon class="material-icons prefix">mail_outline</mat-icon>
                <input type="text" id="emailid" formControlName="emailid">
                <label>Email Address</label>
              </div>
            </div>
            <div class="row">
              <div class="input-field col s12">
                <mat-icon class="material-icons prefix">mail_outline</mat-icon>
                <input type="text" id="uname" formControlName="uname">
                <label>Username</label>
              </div>
            </div>
            <div class="row">
              <div class="input-field col s12">
                <mat-icon class="material-icons prefix">mpassword</mat-icon>
                <input type="text" id="pword" formControlName="pword">
                <label>Password</label>
              </div>
            </div>
            <div class="row">
              <div class="input-field col s12">
                <button class="btn-large btn-submit" type="submit" [disabled]="form.invalid" >Start
Quiz</button>
              </div>
            </div>
          </form>
        </div>
      </div>
    </div>
</div>
```

Create a navbar component which will display the name of the person taking the quiz (fed from registration page), the date and time.

navbar.component.ts

```
export class NavbarComponent implements OnInit {

  pDetails: any;
  pDetailsparsed: any;

  constructor(private db:DatabasesService, private router:Router) { }

  ngOnInit(): void {
```

```
    this.pDetails = localStorage.getItem('participant');
    this.pDetailsparsed = JSON.parse(this.pDetails);

  }

  curDate = new Date();

  Logout()
  {
    localStorage.clear();
    this.router.navigate(['/register']);
  }

}
```

navbar.component.html

```
<nav class="light-red">
    <div class="nav-wrapper">
        <a href="#" class="brand-logo center">Quiz</a>
        <span style="position: absolute; left: 10px;">Hi,
{{pDetailsparsed.fullname|titlecase}}</span>
        <span style="position: absolute; left: 200px;">{{curDate|date:"longDate"}}</span>
        <span style="position: absolute; right: 200px;">{{curDate|date:"fullTime"}}</span>
        <ul id="nav-mobile" class="right hide-on-med-and-down">
            <li>
                <a (click)="Logout()">Logout</a>
            </li>
        </ul>
    </div>
</nav>
```

string-to-number.pipe.ts

```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
  name: 'stringToNumber'
})
export class StringToNumberPipe implements PipeTransform {

  transform(value: unknown, ...args: unknown[]): unknown {
    var numeric = Number(value);
    return numeric;
  }

}
```

Create functions in the quiz component to get the quiz data from the database and display in the webpage, one question at a time using for loop with indexing. Create an index staring with 0 with end 9 (10 questions) and use increments with (click) to increment the questions one at a time. While sending the answers clicked by the participant, a formula will check whether the participant answer matches the answer in the quiz database to determine whether it is correct or incorrect. The result will be sent to the database along with the answer selected.

quiz.components.ts

```
export class QuizComponent implements OnInit {

  quiz: quiz[] = [];
  pDetails: any;
  pDetailsparsed: any;
  currentIndex: number = 0;
  endIndex:number = 9;

  constructor(private db:DatabasesService,private router:Router) { }

  ngOnInit(): void {
    this.db.getQuestions().subscribe(
      (response)=>  {this.quiz = response;},
      (error)=>     console.log(error),
      ()=>          console.log("completed")
    )

    this.countdown.begin();

  }


  @ViewChild('cd', { static: false }) private countdown!: CountdownComponent;

  onSubmit(quizForm:NgForm)
  {
    if(quizForm.value.selectedOption[0] ===
this.quiz[this.currentIndex].options[this.quiz[this.currentIndex].ansindex]){
      this.db.sendParticipantAnswers({
        "selectedOption": quizForm.value.selectedOption[0],
        "answerStatus":"Correct",
        "answerScore":1
      })
    }else{
      this.db.sendParticipantAnswers({
        "selectedOption": quizForm.value.selectedOption[0],
        "answerStatus": "Incorrect",
        "answerScore": 0
      })
    };
  }

  finishQuiz()
  {
    this.router.navigate(['/result']);
  }

}
```

quiz.components.html

```
<app-navbar></app-navbar>
<form (ngSubmit)="onSubmit(quizForm)" #quizForm="ngForm">
   <div class="row" *ngIf="quiz">
      <div class="col s8 offset-s2">
         <div class="card blue-grey darken-1">
            <div class="card-content white-text">
               <div class="card-action" *ngFor="let item of
quiz.slice(currentIndex,currentIndex+1); let i = index" name="item">
                  <h4>{{item.id}}. {{item.question}}</h4><br><br>
                  <mat-selection-list class="collection answer white blue-grey-text" #options
[multiple]="false" name="selectedOption" ngModel #selectedOption="ngModel">
                     <mat-list-option class="blue-grey-text" value="{{opt}}" *ngFor="let opt of
item.options">{{opt}}</mat-list-option>
                     <span *ngIf="selectedOption.untouched">* Please click one of the
options</span>
                  </mat-selection-list>
                  <p>Option selected: {{options.selectedOptions.selected[0]?.value}}</p>
               </div>
            </div>
         </div>
      </div>
   </div>
<countdown #cd [config]="{leftTime:3600}" class="col right white blue-grey-text"
style="position:fixed;right:3%;top:18%;z-index:100;font-size: 36px;"></countdown>
<button type="button" (click)="currentIndex=currentIndex+1" *ngIf="currentIndex<endIndex"
style="position:fixed;right:5%;top:50%;" mat-fab color="primary"><mat-
icon>arrow_forward_ios</mat-icon></button>
<button class="right blue-grey-text" type="submit" [disabled]="quizForm.untouched"
style="height:40px;position:fixed;right:50%;bottom:5%;">Submit Answer</button>
<button class="right blue-grey-text" type="button" (click)="finishQuiz()"
*ngIf="currentIndex==endIndex"
style="height:40px;position:fixed;right:10%;bottom:5%;">Finish Quiz</button>
</form>
```

The result will be displayed on the screen with a review section below so that each question can be reviewed one at a time, scrolled using the same concept of for loop with indexing mentioned earlier. There will also be an option to retake the quiz.

result.component.ts

```
export class ResultComponent implements OnInit {

 quiz: quiz[]=[];
 answers: answer[]=[];
 currentIndex: number = 0;
 endIndex:number = 9;

 constructor(private db:DatabasesService, private router:Router) { }

 ngOnInit(): void {

  this.db.getQuestions().subscribe(
   (response)=>  {this.quiz = response;},
   (error)=>    console.log(error),
   ()=>         console.log("completed")
  )

  this.db.getParticipantAnswers().subscribe(
   (response)=>  {this.answers = response;},
   (error)=>    console.log(error),
   ()=>         console.log("completed")
  )

 }

 sum()
 {
 return this.answers.reduce((sum,item) => sum + item.answerScore,0);
 }

 testStatus()
 {
  if(this.answers.reduce((sum,item) => sum + item.answerScore,0) >= (this.answers.length*0.70)){
   return "\nYou have passed the Quiz.\nCongratulations!"
  }else{
   return "\nYou have failed the Quiz.\nPlease try again!"
  };
 }

 retakeQuiz()
 {
  this.router.navigate(['/quiz']);
 }
```

}

result.component.html

```
<app-navbar></app-navbar>
<div class="row">
  <div class="col s4 offset-s4">
    <div class="card white border border-blue-grey border-5">
      <div class="card-content center blue-grey-text">
<div class="center middle white blue-grey-text" style="font-size: xx-large;">
          <h4>You're Score is : {{sum()}} / {{answers.length}}</h4><br>
          <h4>{{testStatus()}}</h4>
          <button mat-raised-button class="bottom" type="submit" (click)="retakeQuiz()">Take Quiz
Again</button>
      </div>
    </div>
  </div>
</div>
<div class="row">
  <div class="col s4 offset-s4">
    <div class="card white border border-blue-grey border-5">
      <div class="card-content blue-grey-text">
        <div class="row" *ngFor="let item1 of quiz.slice(currentIndex,currentIndex+1);let i=index"
name="item1">
          <h4>{{item1.id}}. {{item1.question}}</h4>
          <p>Correct Answer: {{item1.options[item1.ansindex]}}</p>
        </div>
        <div class="row" *ngFor="let item2 of answers.slice(currentIndex,currentIndex+1); let i =
index" name="item2">
          <p>Your Answer: {{ item2.selectedOption }}</p><br>
          <p>Your answer is {{ item2.answerStatus }}</p>
        </div>
      </div>
    </div>
  </div>
</div>
<button type="button" (click)="currentIndex=currentIndex-1" *ngIf="currentIndex>0"
style="position:fixed;;left:5%;top:50%;" mat-fab color="primary"><mat-
icon>arrow_back_ios_new</mat-icon></button>
<button type="button" (click)="currentIndex=currentIndex+1" *ngIf="currentIndex<endIndex"
style="position:fixed;right:5%;top:50%;" mat-fab color="primary"><mat-
icon>arrow_forward_ios</mat-icon></button>
```

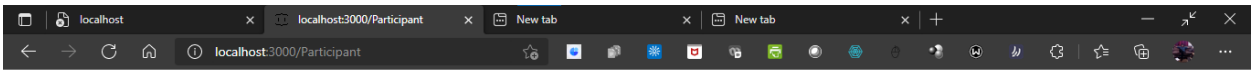
In case of an error while typing the url, the page will route to an information page displaying the error and suggesting workarounds.

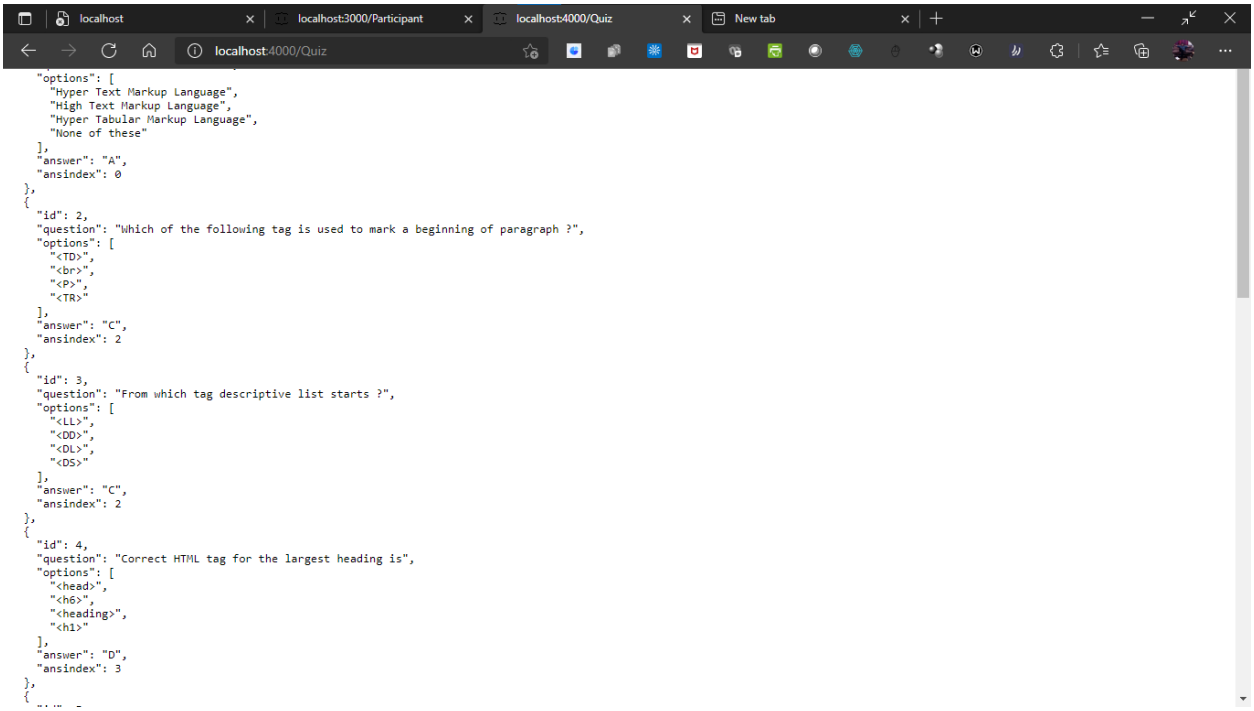
error.component.html

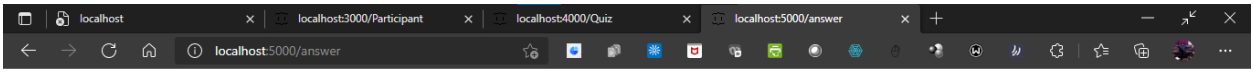
```
<div class="container">
  <div class="row">
    <div class="col s4 offset-s4">
      <div class="card white blue-grey-text center">
        <h1>404</h1>
        <h2>Page Cannot Be Found</h2>
        <p>It appears you have entered an incorrect web adress. Please check the address once
again!</p>
      </div>
    </div>
  </div>
</div>
```

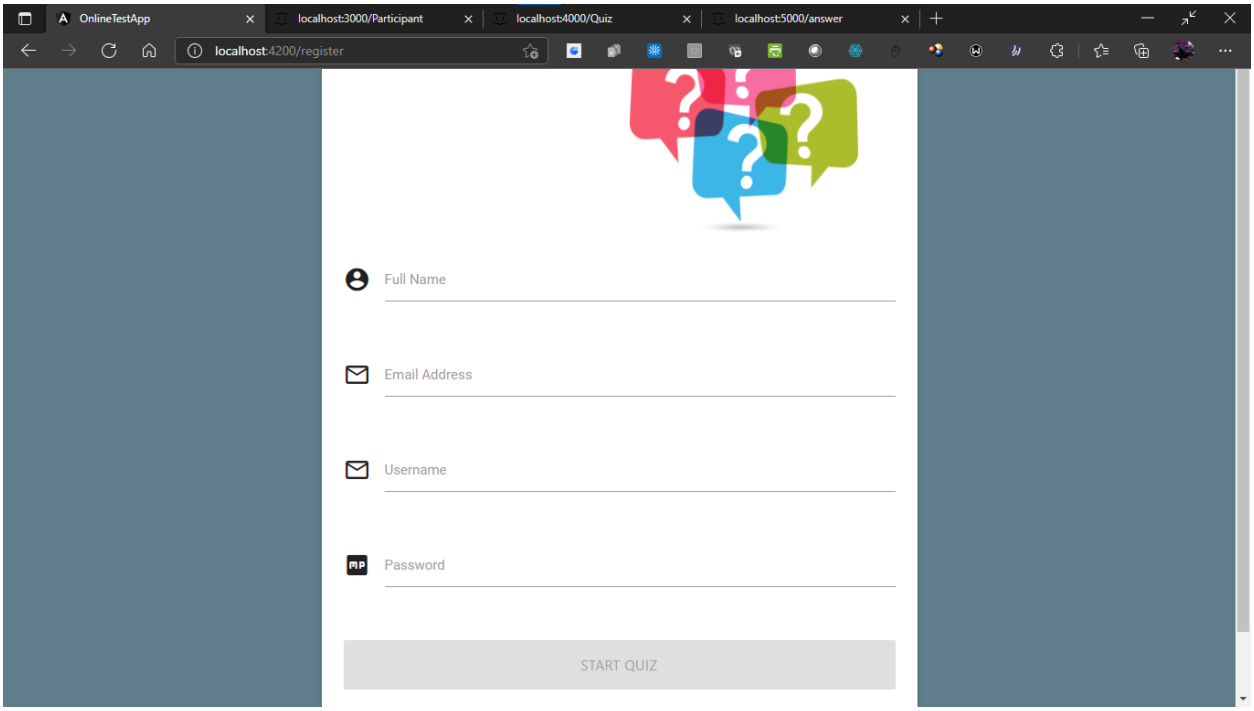
Screenshots of the trial are shown below. The first scenario is with passed quiz and the second, failed quiz. Screenshots of the databases before and after taking the quiz are also below for comparison.

Screenshots






localhost
localhost:3000/Participant
localhost:4000/Quiz
localhost:5000/answer
localhost:5000/answer


OnlineTestApp
localhost:3000/Participant
localhost:4000/Quiz
localhost:5000/answer
localhost:4200/register
Full Name
Email Address
Username
Password
START QUIZ

Quiz Registration

Full Name

Email Address

Username

Password

START QUIZ

Quiz Registration

Full Name

Terry

Email Address

McKormak@aol.com

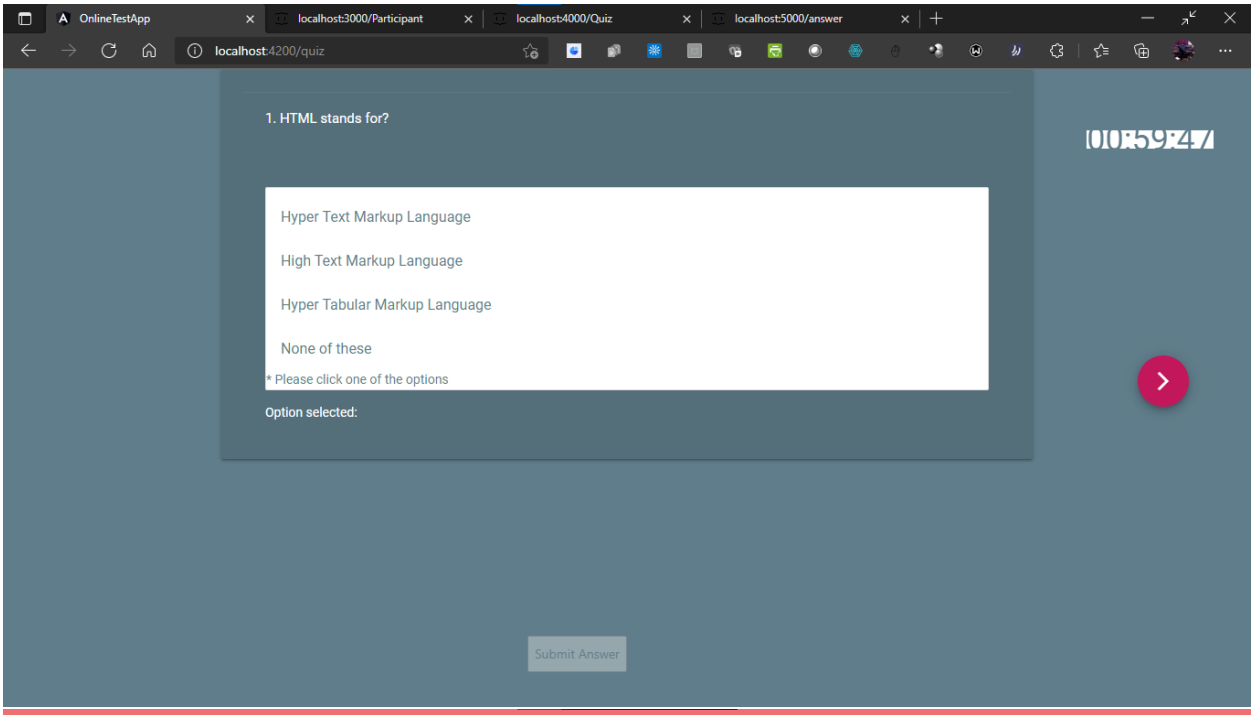
Username

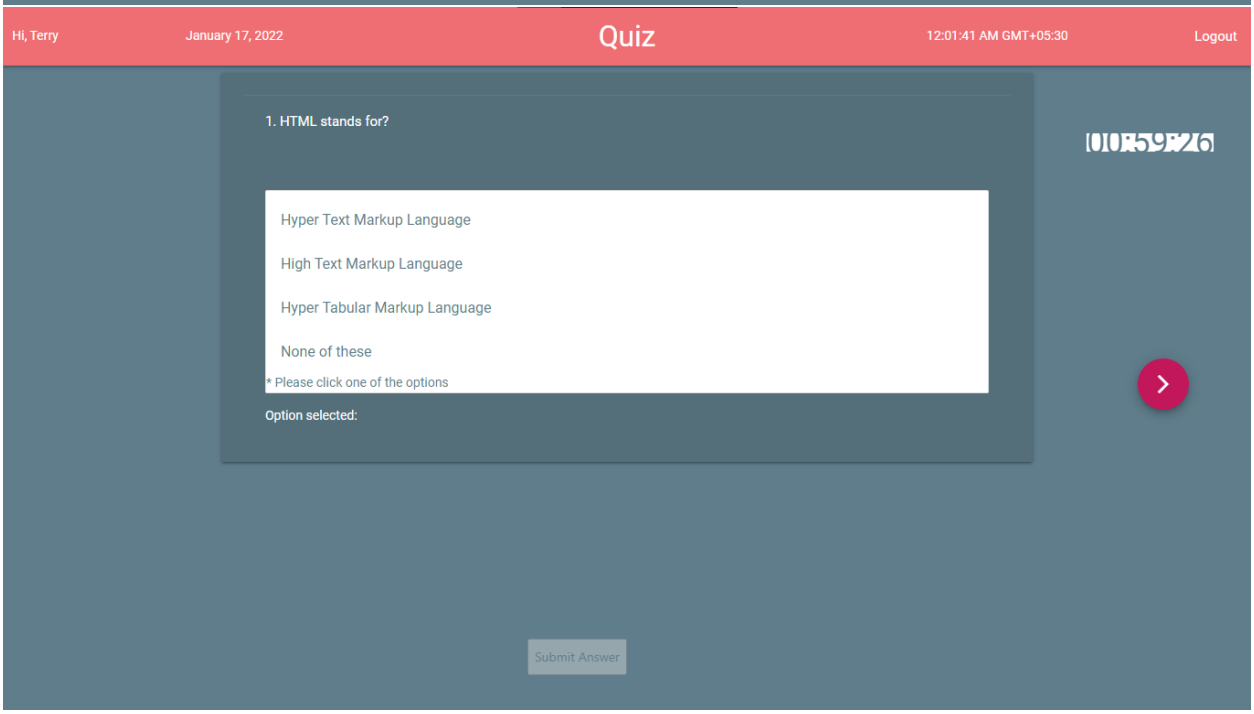
tmckormak

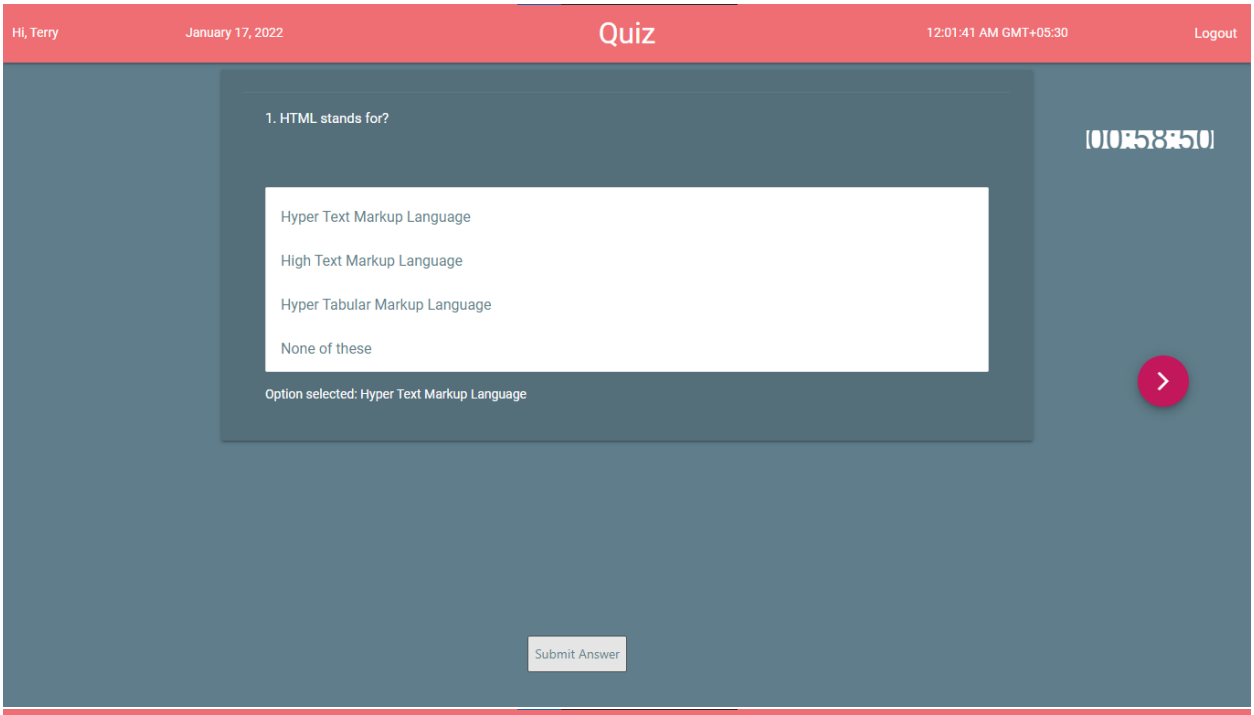
Password

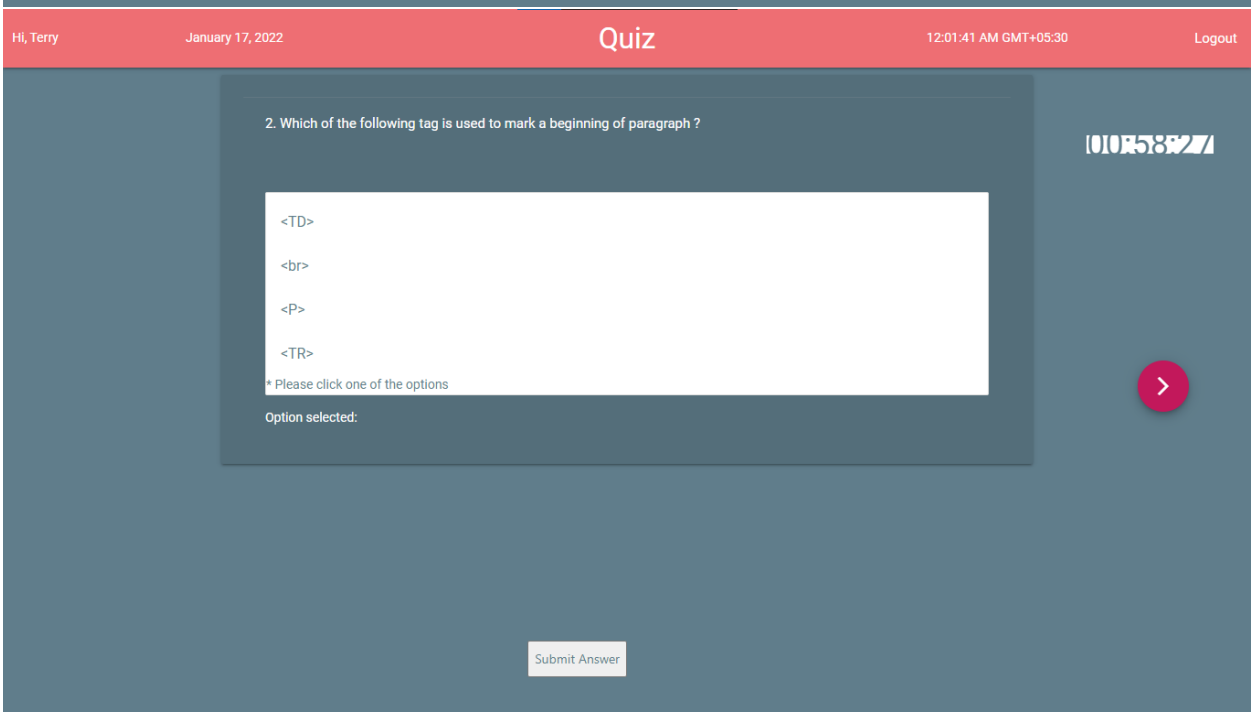
tmck01234S

START QUIZ

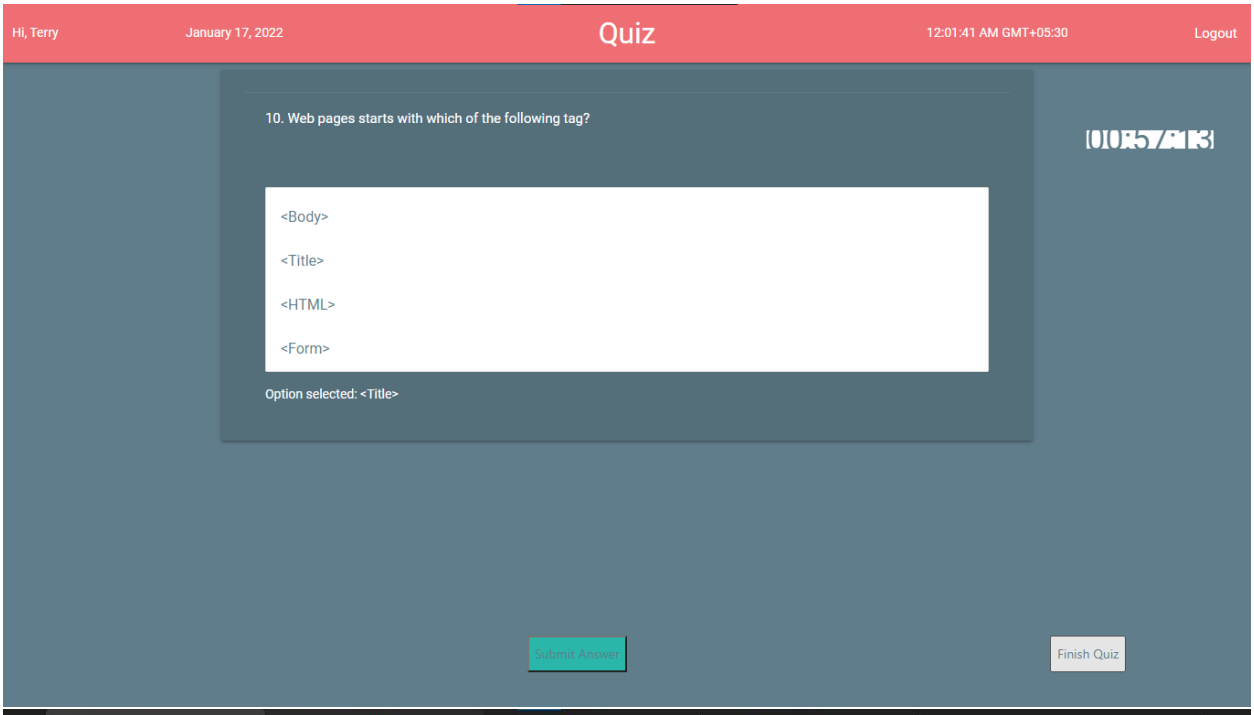

OnlineTestApp
localhost:3000/Participant
localhost:4000/Quiz
localhost:5000/answer
localhost:4200/quiz
1. HTML stands for?
00:59:47
Hyper Text Markup Language
High Text Markup Language
Hyper Tabular Markup Language
None of these
* Please click one of the options
Option selected:
Submit Answer

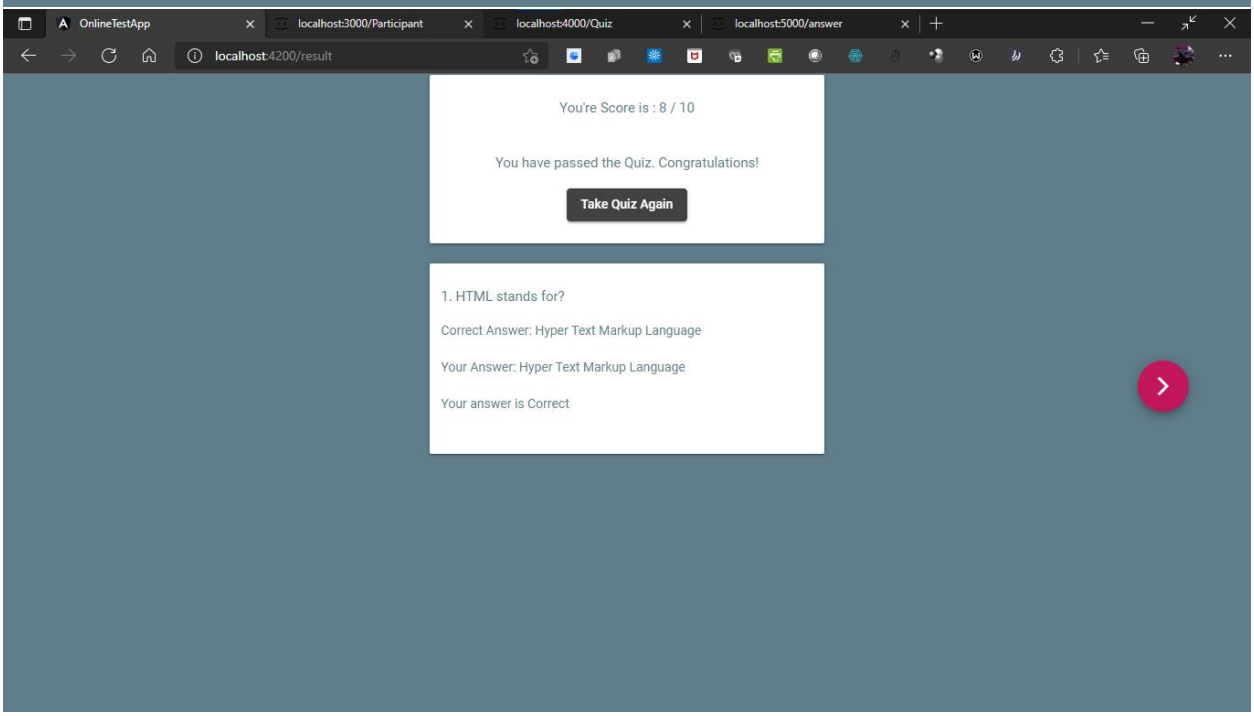

Hi, Terry
January 17, 2022
Quiz
12:01:41 AM GMT+05:30
Logout
1. HTML stands for?
00:59:26
Hyper Text Markup Language
High Text Markup Language
Hyper Tabular Markup Language
None of these
* Please click one of the options
Option selected:
Submit Answer

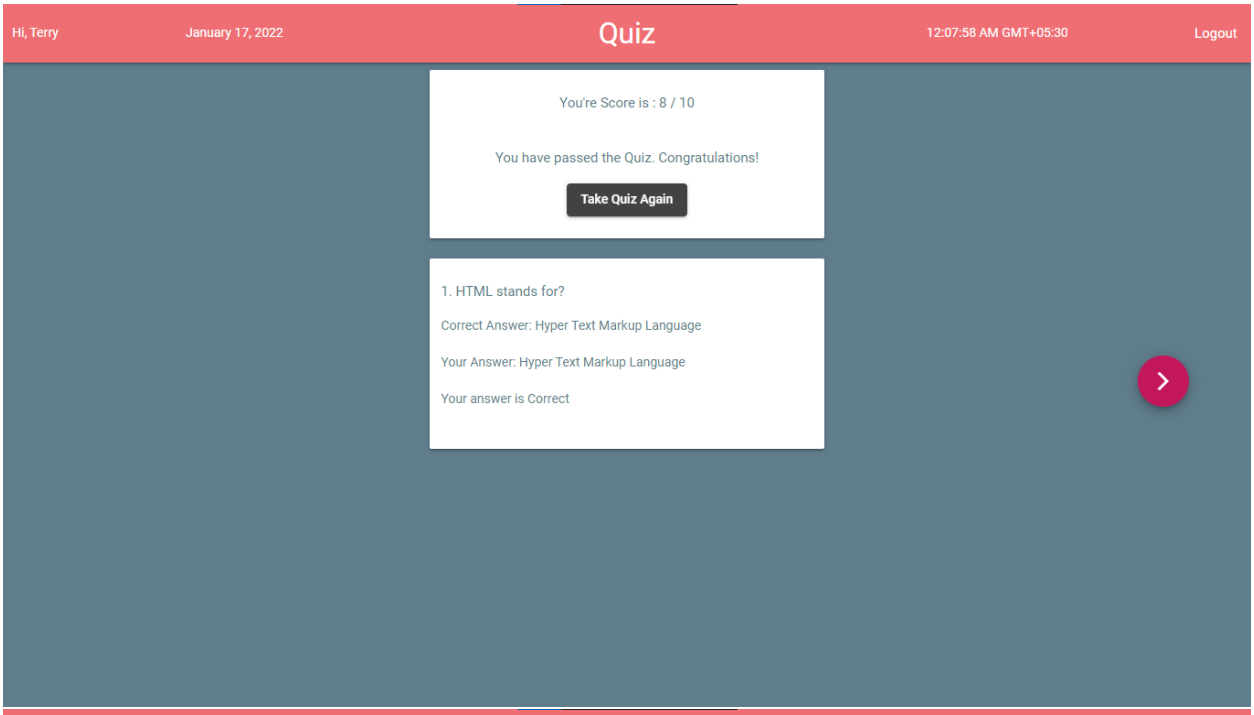

Hi, Terry
January 17, 2022
Quiz
12:01:41 AM GMT+05:30
Logout
1. HTML stands for?
Hyper Text Markup Language
High Text Markup Language
Hyper Tabular Markup Language
None of these
Option selected: Hyper Text Markup Language
Submit Answer

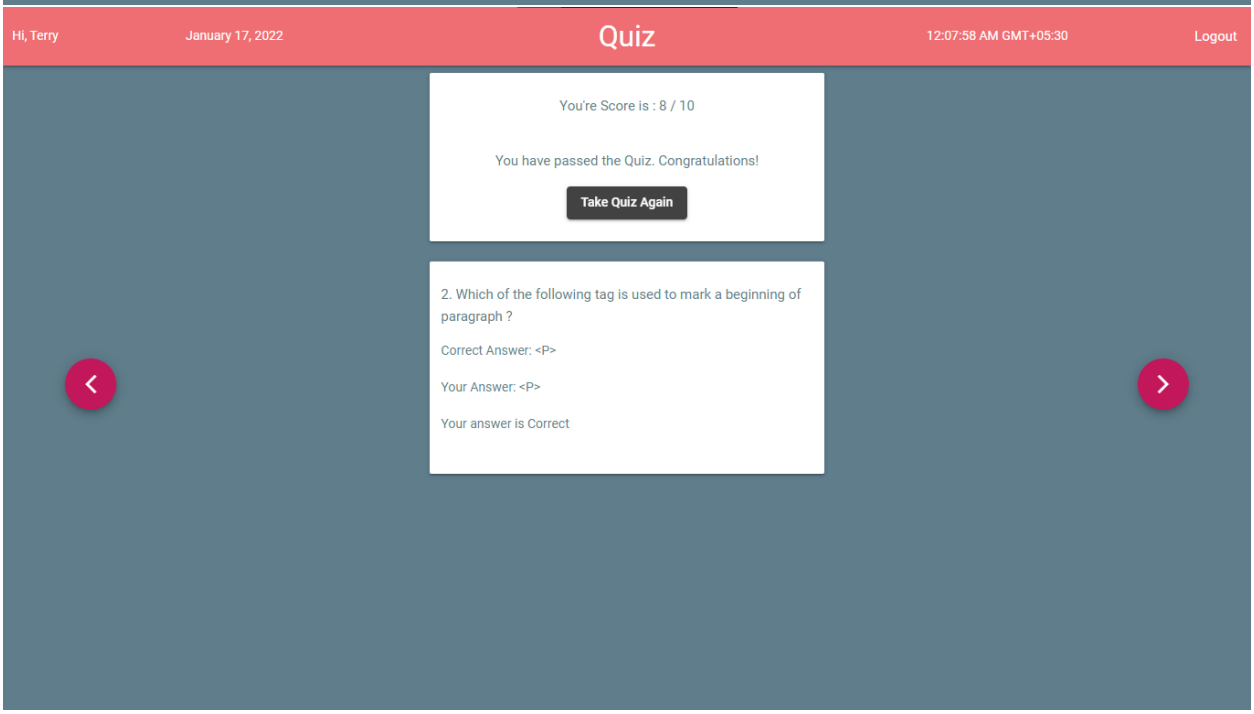

Hi, Terry
January 17, 2022
Quiz
12:01:41 AM GMT+05:30
Logout
2. Which of the following tag is used to mark a beginning of paragraph ?
<TD>

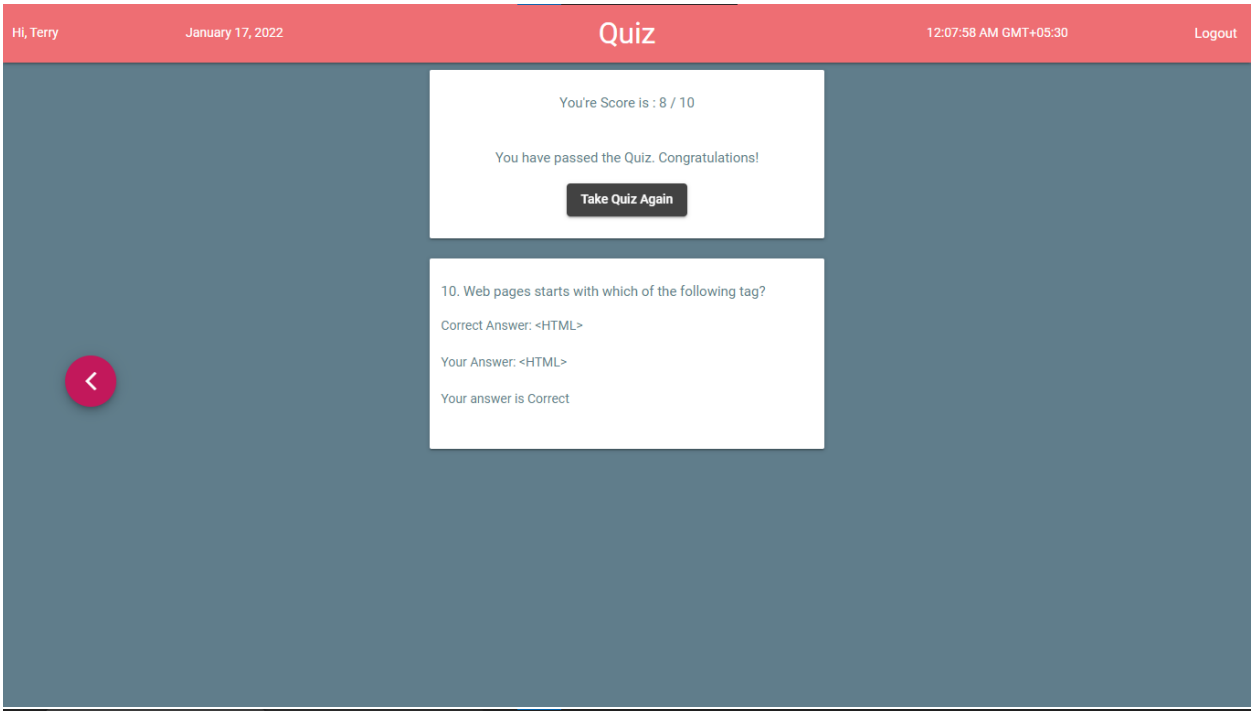
<P>
<TR>
* Please click one of the options
Option selected:
Submit Answer

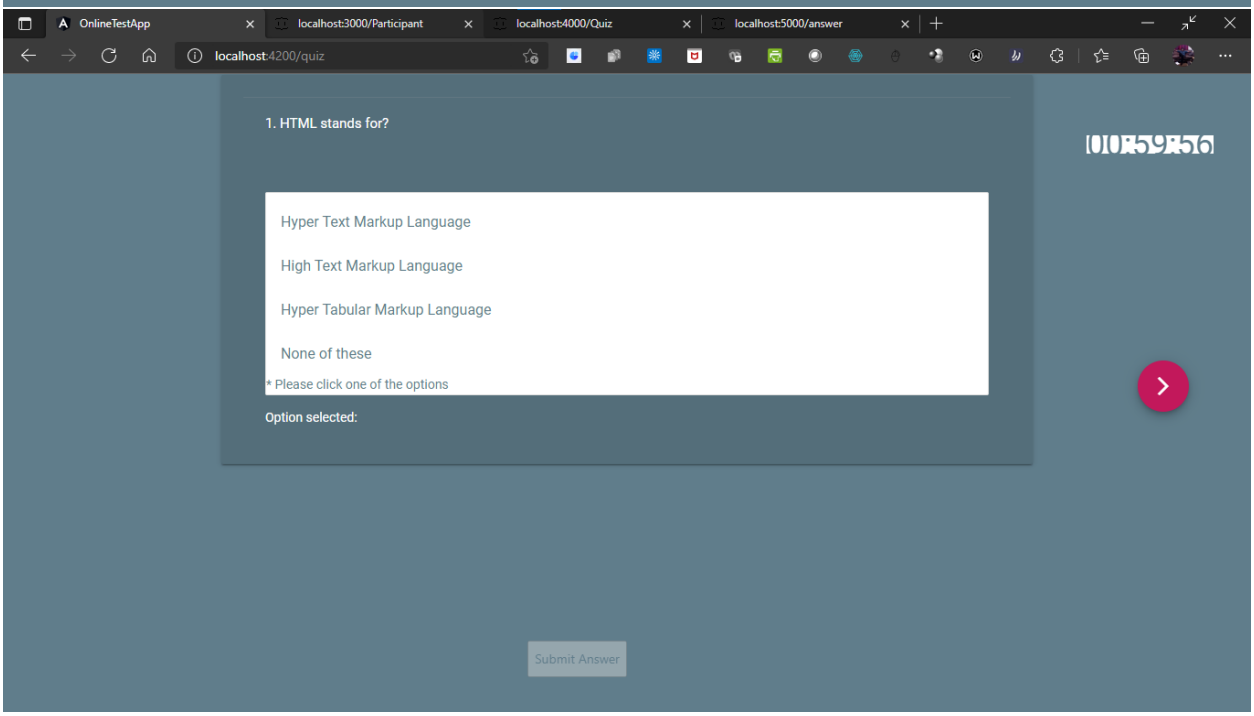

Hi, Terry
January 17, 2022
Quiz
12:01:41 AM GMT+05:30
Logout
10. Web pages starts with which of the following tag?
00:57:13
<Body>
<Title>
<HTML>
<Form>
Option selected: <Title>
Submit Answer
Finish Quiz

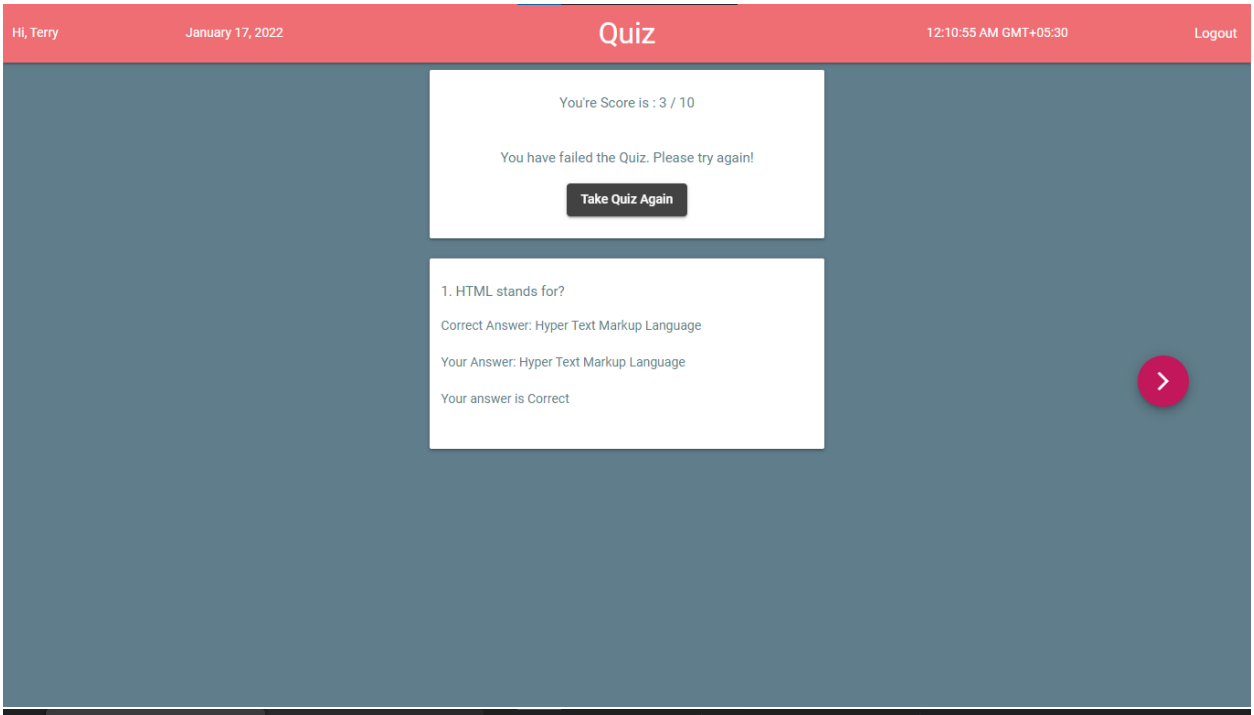

OnlineTestApp
localhost:3000/Participant
localhost:4000/Quiz
localhost:5000/answer
localhost:4200/result
You're Score is : 8 / 10
You have passed the Quiz. Congratulations!
Take Quiz Again
1. HTML stands for?
Correct Answer: Hyper Text Markup Language
Your Answer: Hyper Text Markup Language
Your answer is Correct

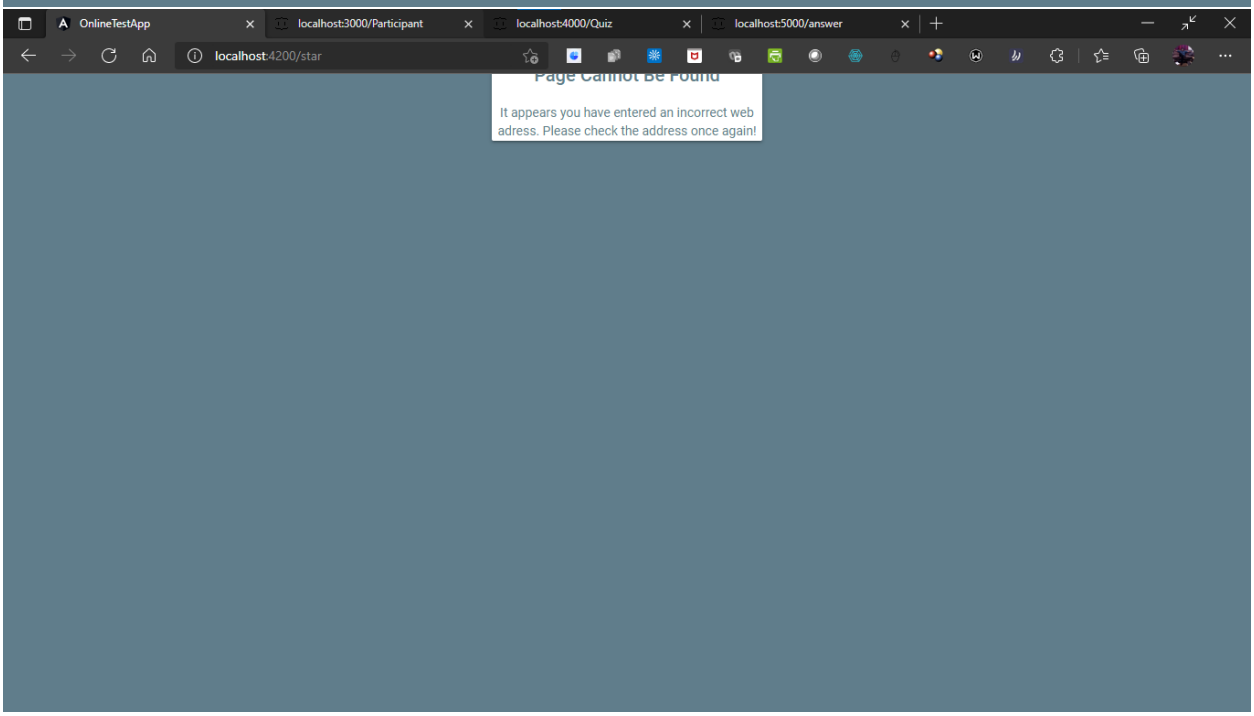

Hi, Terry
January 17, 2022
Quiz
12:07:58 AM GMT+05:30
Logout
You're Score is : 8 / 10
You have passed the Quiz. Congratulations!
Take Quiz Again
1. HTML stands for?
Correct Answer: Hyper Text Markup Language
Your Answer: Hyper Text Markup Language
Your answer is Correct

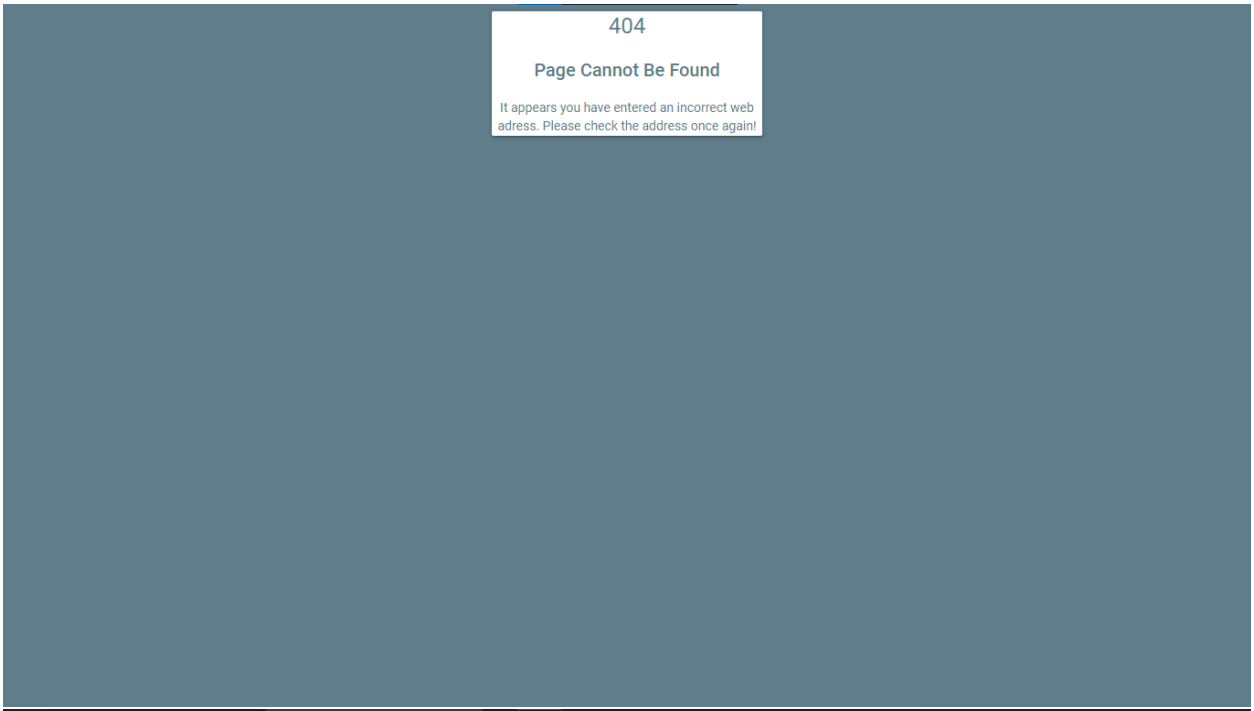

Hi, Terry
January 17, 2022
Quiz
12:07:58 AM GMT+05:30
Logout
You're Score is : 8 / 10
You have passed the Quiz. Congratulations!
Take Quiz Again
2. Which of the following tag is used to mark a beginning of paragraph ?
Correct Answer: <P>
Your Answer: <P>
Your answer is Correct


Hi, Terry
January 17, 2022
Quiz
12:07:58 AM GMT+05:30
Logout
You're Score is : 8 / 10
You have passed the Quiz. Congratulations!
Take Quiz Again
10. Web pages starts with which of the following tag?
Correct Answer: <HTML>
Your Answer: <HTML>
Your answer is Correct


OnlineTestApp
localhost:3000/Participant
localhost:4000/Quiz
localhost:5000/answer
localhost:4200/quiz
1. HTML stands for?
Hyper Text Markup Language
High Text Markup Language
Hyper Tabular Markup Language
None of these
* Please click one of the options
Option selected:
Submit Answer

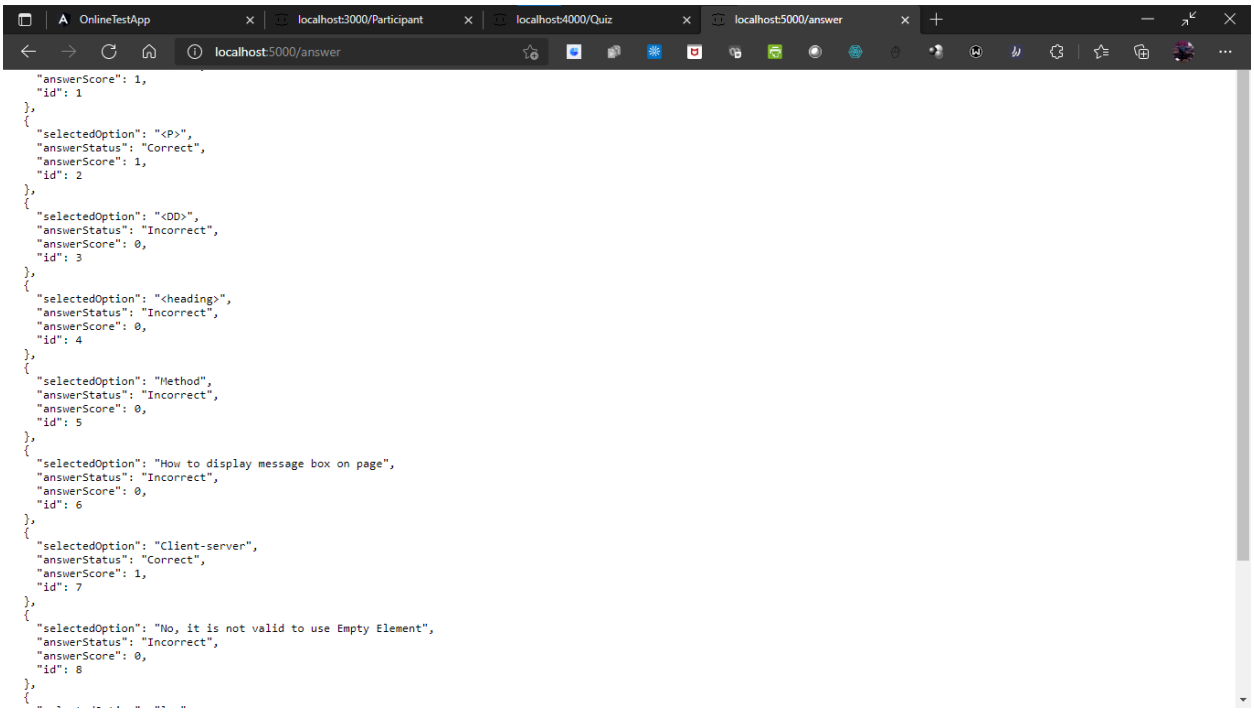

Hi, Terry
January 17, 2022
Quiz
12:10:55 AM GMT+05:30
Logout
You're Score is : 3 / 10
You have failed the Quiz. Please try again!
Take Quiz Again
1. HTML stands for?
Correct Answer: Hyper Text Markup Language
Your Answer: Hyper Text Markup Language
Your answer is Correct


OnlineTestApp
localhost:3000/Participant
localhost:4000/Quiz
localhost:5000/answer
localhost:4200/star
It appears you have entered an incorrect web adress. Please check the address once again!




```
    "uname": "tmckormak",
    "pword": "tmck01234S",
    "id": 1
  }
]
```

```
[
  {
    "fullname": "Terry",
    "emailid": "McKormak@aol.com",
    "uname": "tmckormak",
    "pword": "tmck01234S",
    "id": 1
  }
]
```


```
  "answerScore": 1,
  "id": 1
},
{
  "selectedOption": "<P>",
  "answerStatus": "Correct",
  "answerScore": 1,
  "id": 2
},
{
  "selectedOption": "<DD>",
  "answerStatus": "Incorrect",
  "answerScore": 0,
  "id": 3
},
{
  "selectedOption": "<heading>",
  "answerStatus": "Incorrect",
  "answerScore": 0,
  "id": 4
},
{
  "selectedOption": "Method",
  "answerStatus": "Incorrect",
  "answerScore": 0,
  "id": 5
},
{
  "selectedOption": "How to display message box on page",
  "answerStatus": "Incorrect",
  "answerScore": 0,
  "id": 6
},
{
  "selectedOption": "Client-server",
  "answerStatus": "Correct",
  "answerScore": 1,
  "id": 7
},
{
  "selectedOption": "No, it is not valid to use Empty Element",
  "answerStatus": "Incorrect",
  "answerScore": 0,
  "id": 8
},
{
```

```
[
  {
    "selectedOption": "Hyper Text Markup Language",
    "answerStatus": "Correct",
    "answerScore": 1,
    "id": 1
  },
  {
    "selectedOption": "<P>",
    "answerStatus": "Correct",
    "answerScore": 1,
    "id": 2
  },
  {
    "selectedOption": "<DD>",
    "answerStatus": "Incorrect",
    "answerScore": 0,
    "id": 3
  },
  {
    "selectedOption": "<heading>",
    "answerStatus": "Incorrect",
    "answerScore": 0,
    "id": 4
  },
  {
    "selectedOption": "Method",
    "answerStatus": "Incorrect",
    "answerScore": 0,
    "id": 5
  },
  {
    "selectedOption": "How to display message box on page",
    "answerStatus": "Incorrect",
    "answerScore": 0,
    "id": 6
  },
  {
    "selectedOption": "Client-server",
    "answerStatus": "Correct",
    "answerScore": 1,
    "id": 7
  },
  {
    "selectedOption": "No, it is not valid to use Empty Element",
    "answerStatus": "Incorrect",
    "answerScore": 0,
    "id": 8
  },
  {
```


END OF PROJECT