

## 15. Online Polls Platform

Creating an Online Polls platform using the MEAN stack (MongoDB, Express.js, Angular, Node.js) is an engaging project. Below is a high-level overview of how to build such a platform, along with some code snippets to guide you:

### Project Setup and Structure

Set up a new project folder and structure for your Online Polls platform. Install the required Node.js packages and create a basic Angular application.

#### # Create a new Angular application

```
ng new online-polls-app
```

### - Backend (Node.js & Express.js)

Create the backend of your Online Polls platform using Node.js and Express.js.

### Installation of Packages

Install the necessary packages for Express.js, Mongoose (for MongoDB), and other dependencies.

```
npm install express mongoose cors
```

## Setting up Express.js

Create your Express.js server, set up middleware, and handle routes.

- javascript

**// server.js**

```
const express = require('express');  
const mongoose = require('mongoose');  
const cors = require('cors');
```

```
const app = express();
```

**// Middleware**

```
app.use(express.json());  
app.use(cors());
```

**// Database connection**

```
mongoose.connect('mongodb://localhost/online-polls-app', {  
  useNewUrlParser: true,  
  useUnifiedTopology: true,  
  useCreateIndex: true,  
});
```

## **// Define Mongoose models for Poll and Vote data**

```
const Poll = mongoose.model('Poll', {  
  question: String,  
  options: [String],  
  votes: [Number],  
});  
  
const Vote = mongoose.model('Vote', {  
  pollId: mongoose.Schema.Types.ObjectId,  
  optionIndex: Number,  
  voter: String,  
});
```

## **// Routes for managing polls and votes**

```
app.post('/api/polls', async (req, res) => {  
  // Create a new poll  
  // Save the poll to the database  
});
```

```
app.get('/api/polls', async (req, res) => {
```

```
// Retrieve a list of polls

});

app.get('/api/polls/:id', async (req, res) => {

  // Retrieve a specific poll and its options

});

app.post('/api/vote', async (req, res) => {

  // Create a new vote

  // Update the poll's vote count

  // Save the vote to the database

});
```

## - Frontend (Angular)

Create the frontend of your Online Polls platform using Angular. Design the user interface for creating and participating in polls.

### Design and UI

Design the user interface for your Online Polls using Angular components, templates, and styles.

## Poll Management

Create components and forms for users to create new polls, including specifying the question and answer options.

## Poll Participation

Design components for users to view and participate in polls by selecting options.

- **typescript**

**// poll-participation.component.ts**

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

```
import { PollService } from '../poll.service';
```

```
@Component({
```

```
  selector: 'app-poll-participation',
```

```
  templateUrl: '../poll-participation.component.html',
```

```
})
```

```
export class PollParticipationComponent {
```

```
  poll: any;
```

```
  selectedOption: number;
```

```
  constructor(private pollService: PollService) {}
```

```
loadPoll(pollId: string) {  
  this.pollService.getPoll(pollId).subscribe((data) => {  
    this.poll = data;  
  });  
}  
  
vote() {  
  this.pollService.vote(this.poll._id, this.selectedOption);  
}  
}
```

## MongoDB

Create a MongoDB database to store poll and vote data.

## Putting It All Together

Integrate the frontend and backend by making API requests from Angular components to Node.js routes. Ensure that you handle poll creation, participation, and vote counting properly.

Building an Online Polls platform is a fun and interactive project. You can expand it with additional features such as user accounts, real-time updates, poll sharing, and analytics for poll creators.