**ePoll application**

ePoll is the latest invention to gather up opinions from friends and alike. It's quite like https://www.polly.ai/ but much more simple way.

A poll is created by posting the question and its answer options to the backend. Polls can be listed and fetched individually. A vote can be cast to single option of an event, the backend will not record the identity of the voter and will just count individual votes per option.

The assignment instructions given in this document are general. In case you were provided alternate instructions for assignment with the message, in case of conflict those will overrule these instructions.

**Backend API**

Simple reference implementation of backend server is provided (see server.js).

**Requirements**

Requires Node.js installed with npm.

**Install**

To install required packages run

npm install

**Run**

To run the server run

npm start

**List all existing polls**

Endpoint: /polls

**Request**

Method: GET

**Response**

Body:

{

"polls": [

{

"id": 1,

"title": "What is your favorite drink?"

},

{

"id": 2,

"title": "Is this a cool question?"

}

]

}

**Get a poll**

Endpoint: /polls/{id}

**Request**

Method: GET Parameters: id

**Response**

Body:

{

"id": 2,

"title": "Is this a cool question?",

"options": [

{

"id": 1,

"title": "Yes",

"votes": 0

},

{

"id": 2,

"title": "No",

"votes": 0

},

{

"id": 3,

"title": "Cool, another option",

"votes": 0

}

]

}

**Vote**

Endpoint: /polls/{id}/vote/{option}

**Request**

Method: POST Parameters: id (id of the poll to vote in) option (id of the option to vote for)

**Response**

Body:

{

"id": 2,

"title": "Is this a cool question?",

"options": [

{

"id": 1,

"title": "Yes",

"votes": 0

},

{

"id": 2,

"title": "No",

"votes": 1

},

{

"id": 3,

"title": "Cool, another option",

"votes": 0

}

]

}

**Create new poll**

Endpoint: /polls/add

**Request**

Method: POST Body:

{

"title": "Test qestion?",

"options":[

"Option 1?",

"Option 2?"

]

}

**Response**

{

"id": 3,

"title": "Test qestion?",

"options": [

{

"id": 1,

"title": "Option 1?",

"votes": 0

},

{

"id": 2,

"title": "Option 2?",

"votes": 0

}

]

}

**Front-end assignment**

Create a web application that allows users to: \* List existing polls \* View poll options and their current vote counts \* Vote an option for a poll \* Create a new poll with title and options

The application must use backend server api that is described above (an example server implementation is provided).

You can select the frameworks and components to use freely, a modern js application is preferred. The example server application can be used as a backend, but in case you know that you can write a better one you can do so (please follow the instructions and requirements for backend assignment).

The deliverable should contain application sources and README file containing at least a guide how to build and run the application.

**Backend assignment**

Create a backend server application that provides REST API as described above. An example server implementation is provided, but the implementation is very basic and has several flaws. You should not repeat the flaws but follow the API declaration.

The implementation must \* Implement the described API \* Persist all the polls and their voting data so that the data is available even after server restarts \* Authentication or identification for the users is not required in any way. API is free to use by anonymous users.

The deliverable should contain application sources and README file containing at least a guide how to set up and run the application. In case the persistent store requires some schema (link in database) please include scripts to build that as well.