Node.js site Guestbook -application (6 p.)

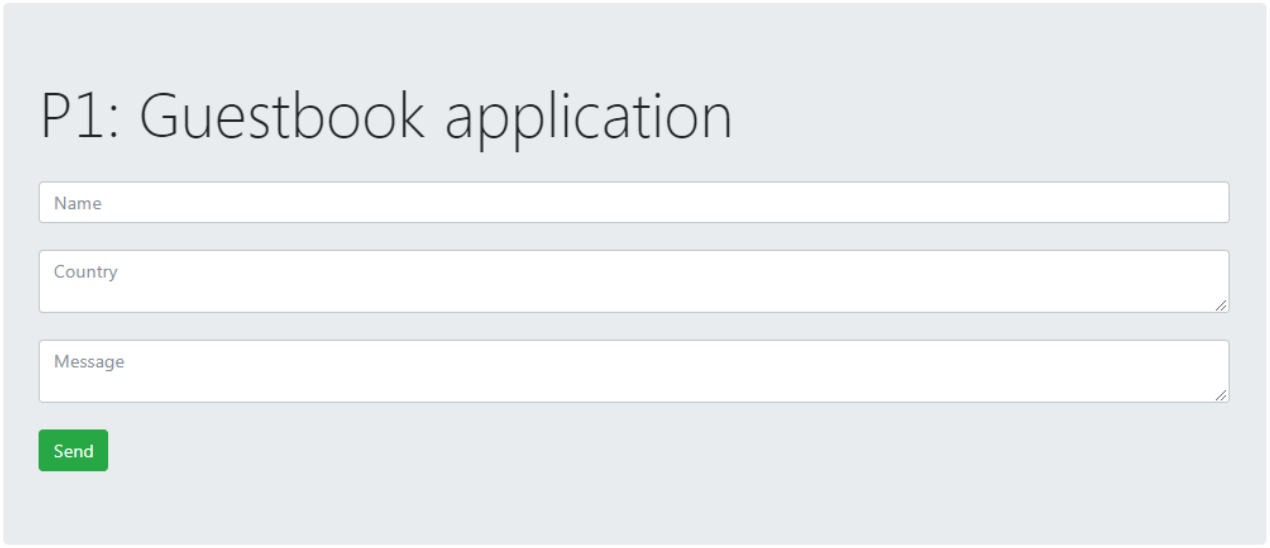
Create a Guestbook app which will allow users to add messages to a public guestbook page. Messages will be saved to a JSON file. Use Bootstrap of Pure.css to add styling.

1. Create a web server with Node.js and Express. Add four routes to your application: "/", "/guestbook", "newmessage" and "/ajaxmessage".

2. / -route should display the frontpage of the site. You can make up this content yourself, it can be a Company / greetings page etc. The page should have navigation links which lead to other routes in the app. Below is a sample page.

3. /guestbook -route should load a JSON-file and parse it nicely on the page as a formatted table (use either Bootstrap or Pure.css to beautify the table). The demo JSON file is available here; http://pastebin.com/VpbJqSic.

4. /newmessage -route should add an input form to the page. Form should have fields for the following data: username, country and message and a button which enables user to send it. Don't allow empty fields to be submitted.



5. When the form is submitted, the collected data should be saved as JSON data to the data file. Sample dataset can be seen below:

```
{
  "id": "2",
  "username": "Sofia",
  "country": "Macau",
  "date": "Mon Apr 05 1993 09:12:01 GMT+0300 (FLE Daylight Time)",
  "message": "Hello world!"
}
```

   Use JavaScripts Date() -function to store the timestamp of the submission. Id is not necessary in our app, but if you wish, you can create one for each item as well.

6. /ajaxmessage should display similar form to the user**. The only difference** is, that the Submit-button does not post the form, but rather it runs a Javascript-code on page which will collect the data from the text fields and send them to the backend as AJAX-call. It will also return all the messages as a response to the page, under the form.

   You can use native Javascript or jQuery to implement the code.

7. Publish the application in your GitHub repository. Deploy to Heroku. Return links to both Heroku app and GitHub-repo.

Evaluation:

Node-server with all 4 routes present	4 p.
Home page (/): actual content and structure, not only text.	3 p.
Guestbook (/guestbook): load, parse, format using style librares	**3 p.**
Form (/newmessage): form with styling, check for errors, post data	**3 p.**
Save the data to a JSON file	**2 p.**
Ajaxform (/ajaxmessage): check for errors, post ajax data, parse & display results with style	**3 p.**
Publish the app in Heroku using GitHub	**3 p.**
Total	**21 p.**