Scalable Nodes Team Audition Project

Objective

Spend 3-5 hours developing a project which implements a simple REST backend service (details below). Present your project to some developers on our team (it's pretty informal and the developers are nice :-)).

Presentation

Bring your laptop and do a short presentation which includes the following:

- a demo of your service running.
- show us the github/bitbucket repo where your project is stored.
- show us a block diagram which describes your project's architecture.
- be ready to walk through the code with us and answer some questions about your project.

Let us know if bringing a laptop is an issue, we'll figure something out. You can use something like Postman (https://www.getpostman.com/) to demo your service.

Project Details

Create a simple application which exposes a simple REST API that allows users to:

- Submit/post messages
- Lists received messages
- Retrieves a specific message on demand, and determines if it is a palindrome.
- Allows users to delete specific messages

Bonus Points

If you finish early (or you are having so much fun you want more :-)) you could add one or two of the following for bonus points.

- Deploy to a Cloud Provider (e.g., AWS). Launch your application via a public DNS record.
- Provide a simple UI to interact with the service:
  - Shows the list of messages posted by the users.
  - Allows to post new messages.
  - Allows to select a given message to see extra details.
- Add Functional and/or System Tests.
- Try out Docker. Deploy and run your app using Docker.

More details

- Please submit your code to a github or bitbucket repository.
- If you have any questions, don't hesitate to ask us.
- Use your favorite programming language.
- Include a README.md which has the following:
  - Brief description of the architecture (including a block diagram).
  - How to build, deploy and access your app.
  - Some REST API documentation.
- Here are some of the things we are looking for:
  - Code quality, ie: Style, complexity, good practices.
  - Application architecture, ie: Design patterns, modularity.
  - API design quality, ie: Follows standards and good practices.
  - Documentation quality, ie: Content quality, completeness and accuracy.