Logs-React Project Proposal

Version 1.0

Matthew Belford

May 22, 2020

1. Problem Definition

As of the composition of this document the OpenRPC tooling repository does not contain any software designed to assist in logging JSON-RPC calls. Users that are making JSON-RPC calls rely on using their browsers Network Development Tools (DevTools) to monitor JSON-RPC traffic. Browser Network DevTools are inefficient when monitoring JSON-RPC calls as filtering is not obvious and little information regarding the call is provided other than the contents of the call itself.

1.1. System Requirements

This section will outline the functional requirements of the proposed system as of the composition of this document and are subject to change throughout the project's development.

1.1.1. REQ-1 Logging of JSON-RPC Calls in Web Applications

The system should provide a clear log of all JSON-RPC requests and responses that are made in a web application in which the system has been implemented. The log should contain the time of the call, if the call is a request or a response, the method that relates to the call, and the payload.

1.1.2. REQ-2 Analysis of Provided Logs

The system should be able to accept an array of JSON-RPC call data and display the system as if it were logged in a web application. All functionalities of the system should be usable on the provided log. The log should be able to be uploaded directly in the application or can be passed into the component as an argument.

1.1.3. REQ-3 Time Series Log

The log should be a time series dataset that is displayed in chronological order.

1.1.4. REQ-4 JSON Syntax Highlighting

The system should use proper JSON formatting and syntax highlighting when displaying the payload.

1.1.5. REQ-5 Importability

The system should be packageable and able to be integrated into a large variety of applications as a component.

1.1.6. REQ-6 Replay JSON-RPC Calls

The system should be able to replay a JSON-RPC call that exists in the log and the system should be able to replay a selected sequence of logs in order.

1.1.7. REQ-7 OpenRPC Document Generation

The system should be able to generate a valid OpenRPC document based on the history of the JSON-RPC calls that the system has logged.

1.1.8. REQ-8 Server/Client Generation

The system should be able to generate a client and server that uses the OpenRPC document that can be generated from the log history.

1.1.9. REQ-9 Filter Specific Methods

The system should be able to filter JSON-RPC calls by their method. A separate tab should be present for all ‘rpc’ methods. The user should be able to make combinations of methods they wish to filter.

2. Proposed Solution

The proposed system is a React component written in TypeScript called Logs-React. The component should be packaged using npm allowing for users to install and import that component into their projects. Logs-React will act as a messenger application between the client and the server displaying all JSON-RPC calls between the two entities. Users of Logs-React should be able to filter JSON-RPC calls by their methods. Furthermore, the history of the JSON-RPC calls should be able to be used to generate an OpenRPC document as well as a client/server environment using the corresponding OpenRPC document. The client/server environment will be created by implementing OpenRPC’s generator tool.

2.1 Wireframes

This section contains three wireframes representing a potential GUI for the proposed system.

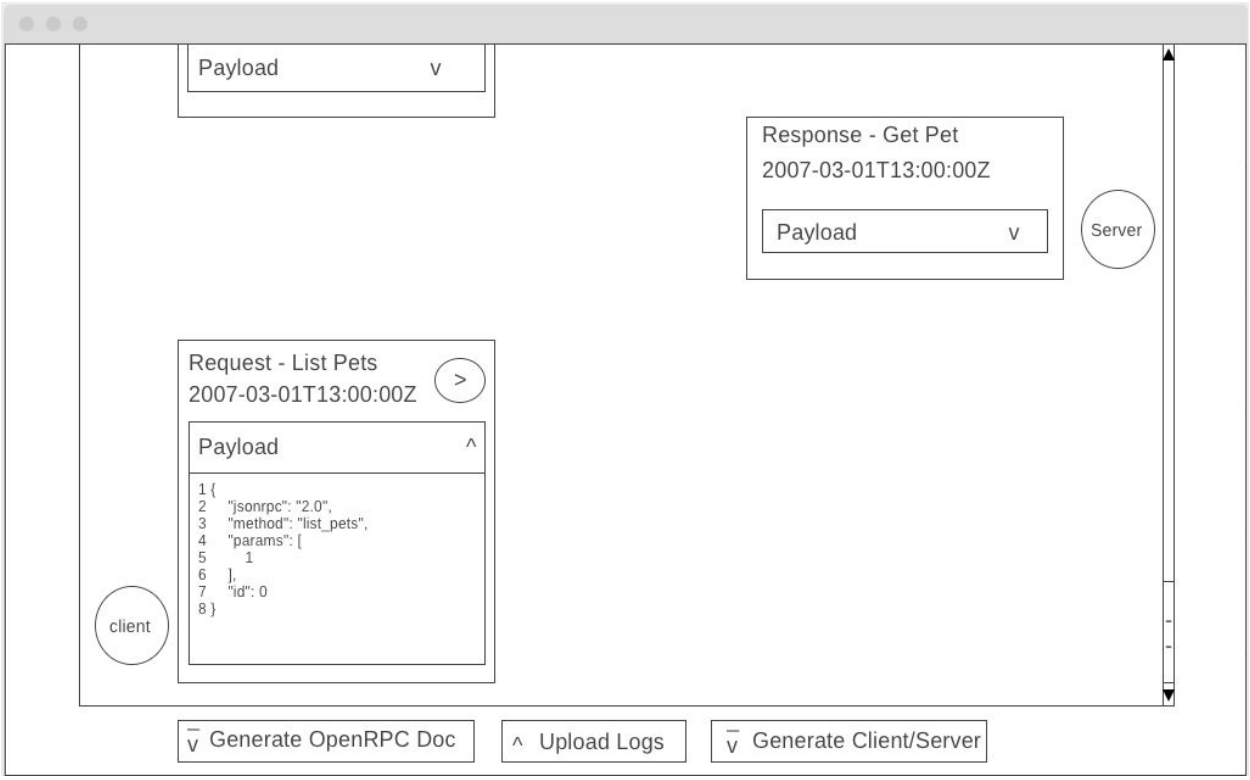

Figure 2.1.1. Logs-React main view with multiple JSON-RPC calls

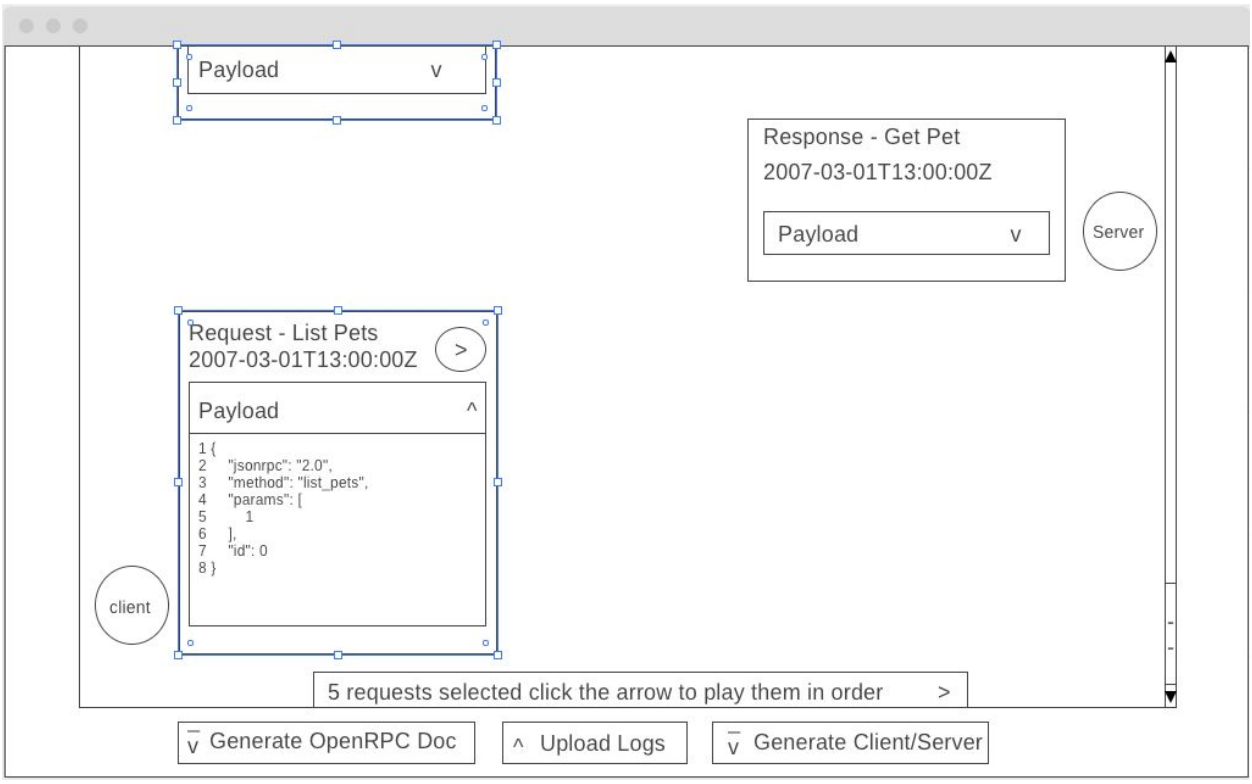

Figure 2.1.2. Logs-React with multiple requests selected to be replayed in sequence

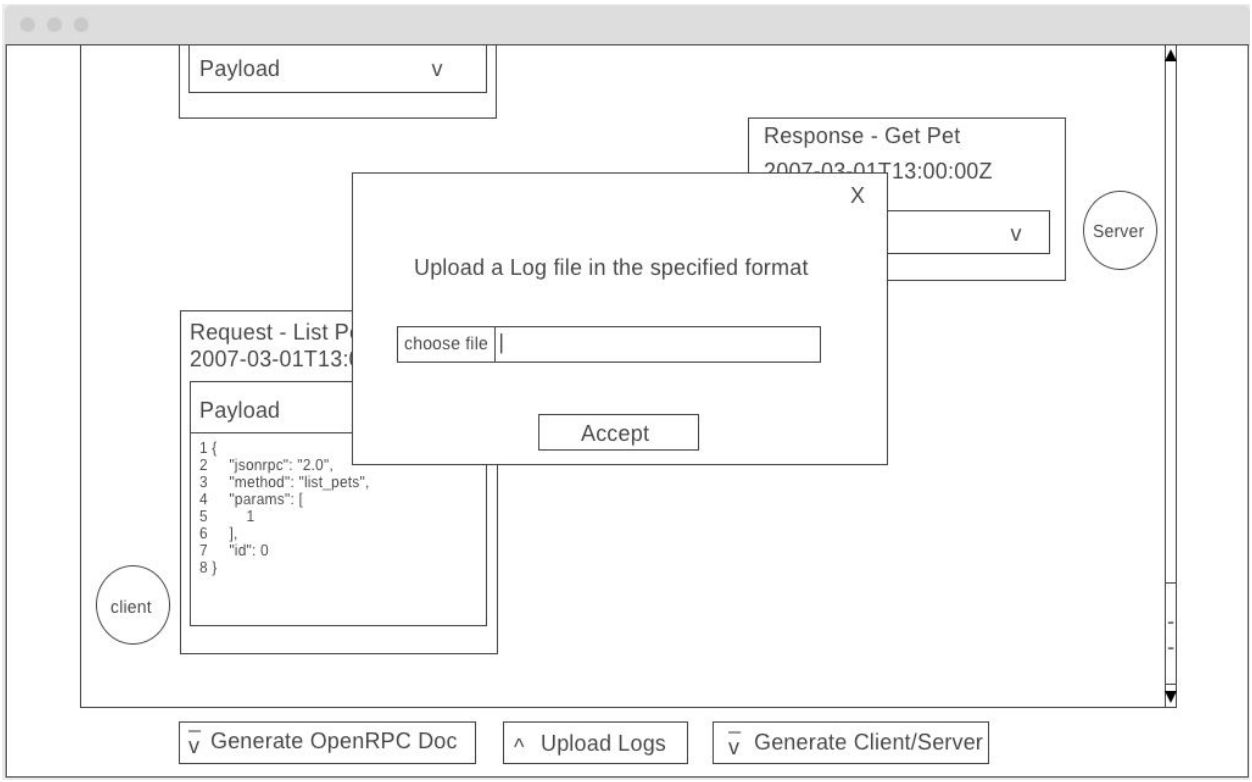

Figure 2.1.3. Logs-React upload logs dialogue

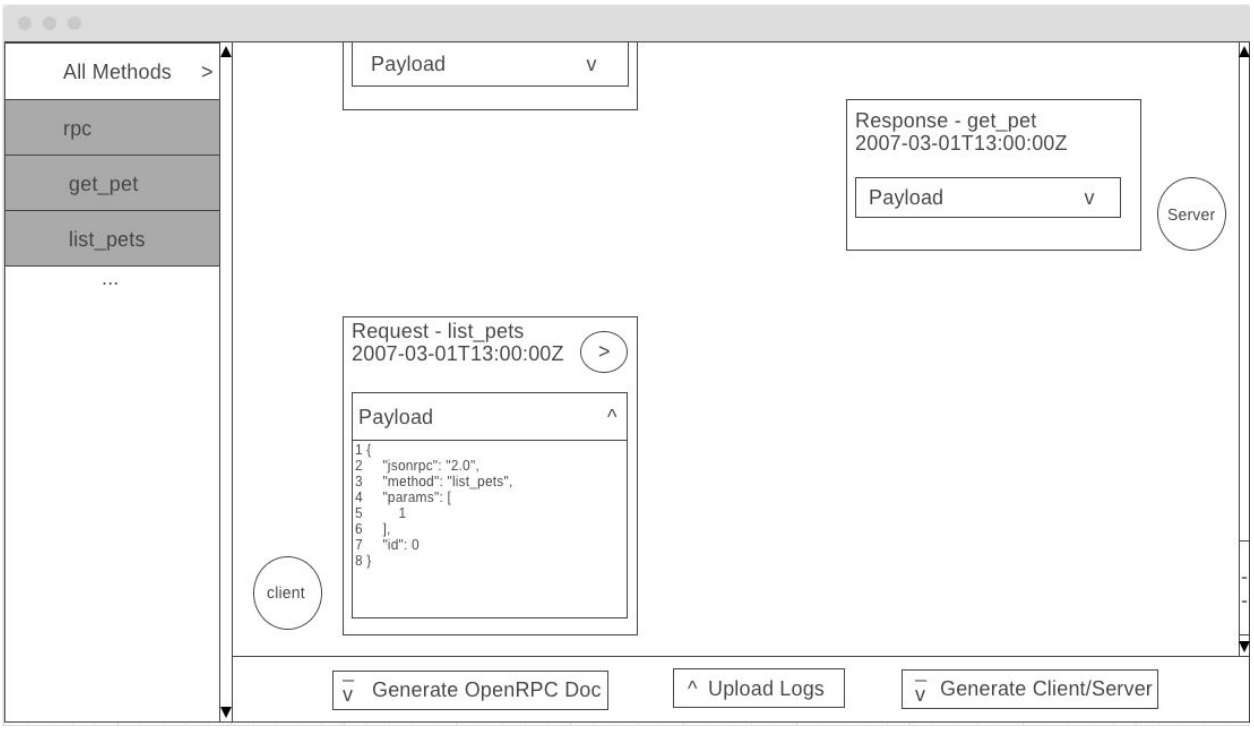

Figure 2.1.4. Logs-React main view with filter tab

3. Milestones

This section identifies project milestones and the sequence in which their development should occur. The achievement of each milestone is reliant on the satisfaction of one or more requirement(s).

3.1. M1 Logs-React Messenger GUI

Milestone one is the completion of the messenger style GUI element of the component including the chronological display of time series data, proper JSON syntax highlighting, and method filters. This milestone is complete with the satisfaction of REQ-1, REQ-3, REQ-4, and REQ-9. Currently M1 is approximately 35% complete and should be achieved by June 5th.

3.2. M2 Replay Requests with Sequencing

Milestone two is achieved with the integration of JSON-RPC request replays. The user should be able to select one or more requests and replay the requests in their selected sequence. M2 is complete with the satisfaction of REQ-6.

3.3. M3 Generate OpenRPC Document

Milestone three is achieved when the user has the ability to generate a valid OpenRPC document using the history of their JSON-RPC calls. M3 is completed with the satisfaction of REQ-7

3.4. M4 Generate Client/Server

Milestone four is achieved when the user has the ability to generate a client/server environment. This feat will be achieved by implementing OpenRPC's generator as a component. M4 is completed with the satisfaction of REQ-8.

3.5. M5 Provided Logs

Milestone five is achieved when the Logs-React component can accept pre-recorded logs by uploading a file or by passing an array of log objects into the component using props. M5 is completed with the satisfaction of REQ-2.

3.6. M6 Testing, Packaging, and Deploying

Milestone six is achieved when Logs-React has been thoroughly tested, packaged, and deployed to npm. M6 is completed with the satisfaction of REQ-5.

4. Resources

Wireframes: https://wireframe.cc/46upvE