# Hammer-IO

**An online platform to build, deploy, and monitor microservice applications in Node.js**

Official Website
https://hammer-io.github.io

Take it for a spin!
http://hammer-io-test.ece.iastate.edu


**Team**
**sdmay18-19**
Erica Clark
Nathan De Graaf
Nathan Karasch
Jack Meyer
Nischay Venkatram

**Advisor / Client**
Lotfi Ben-Othmane


## Problem

The development and deployment of microservice applications to the cloud is a complex process that requires significant resources and domain expertise. Students and small startups with limited knowledge, resources, or time are faced with a significant barrier when beginning a microservice application.

## Solution

Hammer-IO provides an online platform to build, deploy, and monitor microservice applications in Node.js.

## Users

- Small Teams
- Startups
- Students
- Developers with limited time & resources

## Usages

### Initialize the DevOps Pipeline

- Source Control
- Continuous Integration
- Testing
- Containerization
- Deployment

### Application Monitoring

- Application Uptime
- Memory Consumption
- Server Response Time & Server Status
- URL Usage
- Issues
- Build Statuses

## Components

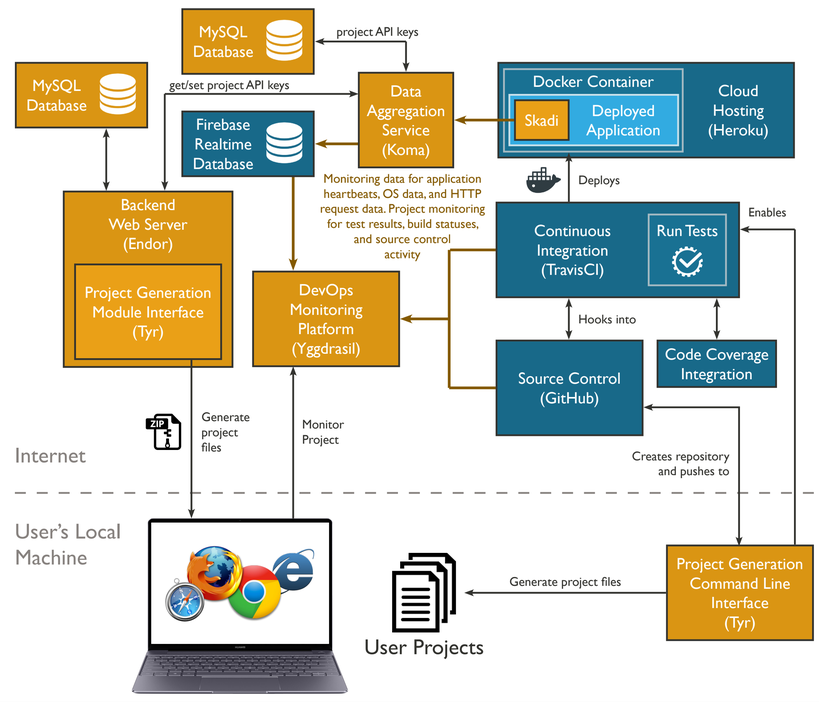
**Tyr** — A Node.Js application generator with automated DevOps initialization

**Yggdrasil** — A web interface to monitor the health and status of deployed Tyr applications

**Endor** — The backend server

**Koma** — A microservice to collect and aggregate monitoring data sent by user applications

**Skadi** — A Node.Js module acting as middleware in an Express application to stream monitoring data to Koma




## Testing

### Strategy

Write unit and integration tests for new features where possible. Where not possible, use manual testing. Tests and linting run in TravisCI upon push to GitHub. Code must pass review, pass lint, and pass tests before being merged.

### Verification Activities

**Unit Testing**
Individual components tested using Mocha

**Acceptance Testing**
Developers ensured all features aligned with client requirements

**Code Review**
Code written in feature branches and reviewed by another dev before merging to master

**Integration Testing**
Components tested after integration using Mocha and Chai

**Manual Testing**
Required for most UI features and third-party integration testing

**Static Analysis**
ESLint used for static analysis. Extended linting rules used by Airbnb.

## Project Goals
*(Design Requirements)*

### Functional Requirements

**Node.js Application Generation with Fully-Configured DevOps Pipeline**
A tool that sets up the services (such as source control, continuous integration, and deployment) involved in maintaining and delivering our user's Node.js application.

**Monitoring Interface**
A way to view build and test histories, uptime and health statistics, and other reports for one of our user's applications.

### Non-Functional Reqs

**Usability**
Simple, polished, and easy to use.

**Supportability**
Deployed instance support for Linux with Node.js 8. Web application is cross-platform.

**Reliability**
Application uptime > 99%

**Security**
Ensure user passwords and keys are handled safely.

### Operational Environment

**Automated DevOps Tools**
Our tools are published to npm, which can be installed and ran on our user's machines.

**Server Instances**
Developed cross-platform. Deployed on an Ubuntu 16.04 virtual machine on ISU's VPN.

**Source code**
GitHub, as an open source project, which allows users to build our project from source.

## Technical Details

### Programming Languages

| | |
|---|---|
| All Applications | Javascript |

### Libraries

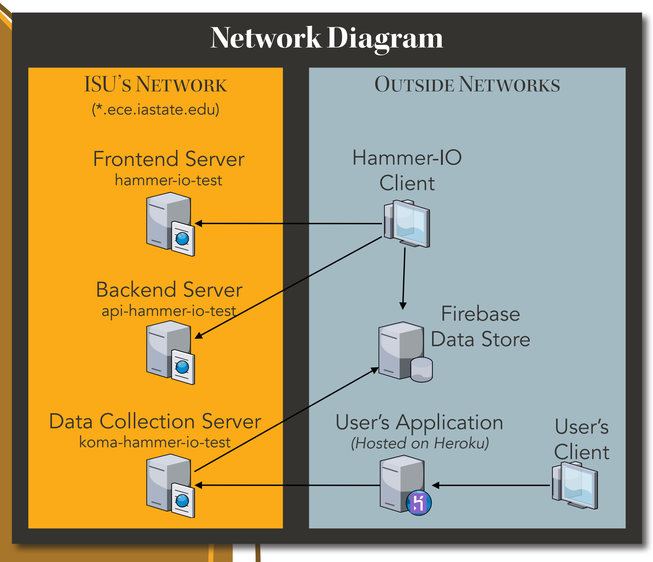
| | |
|---|---|
| Frontend | React |
| Backend | Node.js, Express.js |
| Database | MySQL, Sequelize ORM, Firebase |

### Development Tools

| | |
|---|---|
| IDEs | IntelliJ, WebStorm, DataGrip |
| Source Control | Git / GitHub |
| Project Management | GitHub |
| Continuous Integration | TravisCI |
| Deployment | Docker |
| Dependency Mngmt | NPM |

### Operating Environment

| | |
|---|---|
| Endor, Yggdrasil, Koma | Docker containers within a single virtual machine on ISU's VPN. An NGINX reverse proxy routes requests to the appropriate container. |
| Tyr | CLI installed via NPM. |
| Skadi | NPM module used as a dependency in a user's Node application. |




### By the Numbers
*(as of 4/20/2018)*

| | |
|---|---|
| Developers: | 5 |
| Systems: | 6 |
| Issues Opened: | 360 |
| Issues Closed: | 313 |
| Git Commits: | 926 |
| Lines of Code for Tracked Files: | 189,608 |