James W. Gary

Software Engineer | New York City | 646.549.0648 | jim@jamesgary.com | github.com/giacomo9999 | linkedin.com/in/james-gary/

Technical Skills

Strong — JavaScript (ES6+), React.js, Node.js, Node Express, REST APIs, NoSQL/Mongoose, SQL/PostgreSQL, Webpack, Git/Github, HTML/CSS, OOP, Agile/Scrum, Adobe CS (InDesign, Photoshop, Illustrator), Lightwave3D, Form•Z.
Experienced — AWS, GraphQL, TDD (Jest, Enzyme, Spectron, Chai, Cypress), Ruby, Ruby On Rails, Redux, Typescript.

Experience

Aesop for Storybook | Software Engineer

- Implemented VSCode API to build an extension porting Storybook UI that allows developers to test components directly in the IDE, thereby leading to better developer experience and convenience while creating, viewing and testing.
- Integrated WebView API with native VS Code API, allowing user to instantiate a view panel that renders StorybookUI with full functionality, providing the developer same experience as viewing Storybook in the browser.
- Invoked Node.js child process module and used its ability to asynchronously spawn child processes and execute commands in the console based on Aesop's logic and the developer's workspace configuration for launching Storybook.
- Used Typescript to help streamline the debugging experience, reduce errors, enforce predictable functions and classes with consistent static data types, assertions and interfaces, and ensure consistency of future code contributions through the use of static typing.
- Applied Node.js event emitters to ensure seamless execution of events, manage asynchronous events, and monitor running processes.
- Explored Storybook repository to build stories in React and test components in isolation to ensure full compatibility with the extension.
- Utilized PS-Node and Node-Netstat modules to look up, interact and manipulate node processes stdin and stdout to extract data— ports, pids, arguments, local and network addresses—and aid in implementing the logic that Aesop depends on to render Storybook within VS Code IDE.
- Configured Webpack integration to optimize performance with hot module reloading for dynamic component previews upon file system changes, without constraining existing support for client-defined configurations.
- Product developed and maintained as part of OSLabs.

American Association For Artificial Intelligence and **IEEE Computer Society** | 3D Digital Artist | 1993-present

- Used 2D and 3D software (Lightwave3D using Python with Lightwave SDK, plus Adobe CS: Photoshop, Illustrator; also Form Z and 3DCoat) to conceptualize and execute 3D-renderings.
- Based on project needs and resolution of final render, used Python with Lightwave SDK to create polygonal or parametric OBJ files using hand-modeled or imported data, shaders and image/normal maps (linked asset dependencies in LWO scene files; executed final renders).
- Produced rough renders for client approval; designed and created 3D assets (models, maps, scene files) for implementation of final rendering; executed final render and added typography.

Open-Source Products

Whiteboard | Shared digital-whiteboarding app

- Developed React Konva canvas in HTML5 that gave access to the DOM, enabling app to gather data on specific user inputs through modular and custom components while preserving the declarative code and modularity afforded by React.
- Implemented BCrypt to hash passwords for one-way encryption, improving user security while maintaining ease of authentication for user. Used NoSQL to store and manage user data remotely, allowing for the possibility of sharding for scalability as the number of users grows.

Chaturanga | URL-based Websocket chat functionality.

- Used Websockets to provide full-duplex communication channels by establishing a handshake over a single TCP connection to provide real-time messaging updates between multiple users, reducing the number of HTTP requests necessary for the app to function.
- Built Chrome extension to display user comments in a pop-up, allowing comment windows to exist independently of browser tabs.
- Used Node Express to optimize throughput and scalability via a web application capable of handling multiple http requests at a specific URL .

OSPaperTrader | Real-time stock tracker that allows users to dynamically track the value of a mock portfolio.

- Used Node.js/Express server to maintain a database of user login and portfolio info and to query the AlphaVantage API of stock prices and update users' portfolio values accordingly.
- Designed a PostgreSQL database schema to store relational data, connecting users with their portfolio and login information.
- Built React front end to create a responsive view layer using modular components to be responsive and scalable when displaying user data.
- Engineered Cypress end-to-end tests to insure that all user-facing facets of the app are functional before pushing code to production.
- Used Redux to create predictable state container for React data and facilitate debugging through easily testable "pure" reducer functions.

Public Talks

"*GraphQL: An Alternative To RESTful APIs*" – Build With Code NYC (Ethiq speaker series) — January 2020

Education

Bachelor of Fine Arts - Art Center College of Design, Pasadena , CA

Interests

Distance running (finished NYC Marathon in 3:45) | Cooking (particularly Indian and Chinese cuisine) | Old-school role-playing games (AD&D 1st ed., Metamorphosis Alpha, Gamma World) | Collecting books about art, design, and cookery | Exploring Brooklyn.