# John Henry Mahr

**UI Development**

## SUMMARY

I am a Front End UI developer working with modern technologies and frameworks to help build the next generation of browser deployed applications. I am excited about the fusion of technology, design, copy and UX to deliver experiences that are functional and satisfying to use.

## SKILLS AND TECHNOLOGIES

### Foundational

- Html5 markup with a special focus on semantic structure and content hierarchy.
- CSS3, LESS, SASS, utilizing BEM methodology
- Javascript: ES5, ES2015, ES7
- Node.js
- Responsive, RESS and Adaptive approaches to multi-device, multi-format sites.
- Javascript modules: AMD, CommonJS and ES6 using Webpack, Require.js.
- Image manipulation, optimization and graphics using Adobe Illustrator/Photoshop.
- General Unix/OSX CLI/Bash skills.

### Frameworks and Libraries

- React.js, with Redux Flux implementation
- React Router
- Babel transpiler
- Backbone and Backbone.Marionette JS framework
- Bootstrap CSS and UI Component framework
- Dust.js client side templating
- jQuery (of course)
- Freemarker Java based templating language
- Bluebird and Q Promise polyfills
- Modernizr for feature detection.
- Underscore / Lodash
- Es Lint / Standard JS style guide

### Testing

- Jest (preferred)
- Mocha with Chai using Karma and Phantom.js
- Selenium webdriver IO for functional and integration test automation.

### Tooling, infrastructure and management

- Git version control
- NPM package/dependency management
- Webpack/Gulp/Grunt JS Tasking/Deployment
- Maven/Artifactory infrastruture
- Jenkins configuration and management.
- Docker Containers

### Server Side

- Node/ Express
- Php

1326 Jefferson St NE • Minneapolis, MN 55413
(612) 481.8114 • www.johnhenrymahr.com • john@johnhenrymahr.com
www.linkedin.com/in/johnhenrymahr • github.com/johnhenrymahr

John Henry Mahr

## WORK HISTORY

| | |
|---|---|
| *2014-present* | **Best Buy** Contract developer (via **Valere**) |
| *2009-2014* | **Bizideo LLC**: Contract developer |
| *2007-2008* | **HomeShowVR**: PHP developer |
| *2001-2009* | **Digital Access:** Art department manger. |
| *1999-2001* | **Graf-X**: Prepress/Production |
| *1997-1999* | **The Publishing Group**: Production/Design. |

## SAMPLE PROJECTS

## BEST BUY

### Customer Self Service

I was part of a new initiative at Best Buy to allow customers more options to modify their orders post-purchase without having to call customer service. The primary business objective was to cut costs and lower overall load on the customer service department. A secondary advantage was to allow post purchase sales of protection plans which offered a new profit center. I was provided with business requirements, a UX document, and design files for large and small view. I worked with backend developers to develop an FE/BE integration contract. The application was developed with Backbone Marionette with client side routing and service calls to REST endpoints. I had to work with BE devs and QA to get the project through integration testing with the Order Management Enterprise System. Both small view and large view versions of the app needed to be created (BBY uses adaptive RESS methodology in place of responsive design). Other considerations were: guard logic so self service pages could not inadvertently be accessed for non-qualified orders or orders beyond the modification grace period, triggering analytics events for certain user behavior based on business requirements, feature flag control of certain aspects of the application so the Operations Department has some options to manage application if load was to get too great and response times suffer during peak holiday season. Additionally, I was responsible for the integration of new links and updated statues into the existing Order Details page.

I has some amount of help, when possible, from other Front-End devs, but I was the UI point person and the one who was ultimately responsible for delivering UI code on scheduled target dates.

#### Self Service: Modify Payment

Customers have the ability to retry the Credit Card attached to the order, add a new Credit Card or use one of their existing cards saved in their profile (for registered users). Utilized card type detection, Luhn validation for card numbers, address standardization and front end input validation with a host of special requirements for specific card types. Phase II of this self service added real time authorization to provide immediate feedback to the user if there was an issue with the new transaction.

1326 Jefferson St NE • Minneapolis, MN 55413
(612) 481.8114 • www.johnhenrymahr.com • john@johnhenrymahr.com
www.linkedin.com/in/johnhenrymahr • github.com/johnhenrymahr

**Self Service: Shipping Address / Level of Service**

This self service gives users the ability to change a store pickup order to a shipping order, change the shipping address of an existing order to a new address or one from their profile (registered users), change the level of service (shipping priority) of an existing shipment, and see any additional charges or refunds that may be incurred.

**Self Service: Add Geek Squad Protection Plan / Apple Care**

This self service focuses on the ability for a customer to add a protection plan (either a Geek Squad variety or Apple Care for an Apple product) post purchase within a designated time period.

**Self Service: Return Center**

The newest self service launching for 2016 holiday season. This application allows the customer to initiate a return as a self service option. Depending on the sku details they may be able to download a pre-filled return form (generated with css) or a pre-filled return form that also contains a prepaid shipping label. The app includes the ability to list all items that are returnable, initiate a new return, re-print shipping labels/return forms where appropriate, and track a return in progress. That app was built with Backbone Marionette using server side routing and REST api calls.

## Manage Credit Cards / Financial Services Page

Ported an existing app that was written in a JSP based server rendered page to a Backbone Marionette app. Re-implemented existing UI logic using new technology so app could be moved from the legacy system to the new dedicated profile application. Supported existing features, but also added modularity to support additional features as requested by financial services team for specific programs such as Best Buy Visa, Amex Pay with Points, and others. Original app did not support mobile, so I worked with design to create mobile view for the application.

## Manage Shipping Addresses

The first app in the profile stack to be ported to the new Facebook React framework utilizing the Redux Flux implementation. I worked on conversion from Backbone/Require.js to a React/Redux/Webpack based application. Project requirements were to preserve functionality and to not introduce and new regressions with the port.

## BIZIDEO LLC

### Video CMS: uploaded and Transcoding Service  Components

Client maintained a Video CMS system for their clients to upload, manage, and publish media to playlists.  I helped them make the transition from using the high transcoding system provided by their CDN to the more affordable and robust service provided by a third party provider. I made use of the new providers xml api to give the user real-time feedback on the progress of the uploading/transcoding job. I also re-wrote their uploader to move from a flash based system with file size limitations to a pure JavaScript / html5 one that allowed for multiple uploads of virtually any size. Additionally, I added support for FTP Uploads, or more specifically, the processing of files uploaded via FTP. I also helped them move the server side (php) upload and transcoding component from their over-burdened dedicated server to a cloud worker that offered the potential to be replicated easily as user demand increased.

### Playlist API and Feed Generator

I centralized a variety of code snippets located all over the clients system to a API that could generate playlist feeds in xml, xsfp, or JSON formats depending on parameters passed through _GET variables using data from the clients mySQL server. Over time I had to extend it several times to meet new requirements, including adding support for rtmp streaming, multiple source formats, creating playlists based on keywords from the clients existing keyword system and playlists based on statistics from clients stat tables. I also abstracted the CDN based parsing required into drivers so it would be easier to load content from multiple CDN sources.

### Subscription Service Manager

I designed and implemented a subscription service provider that would allow users to purchase access to content for a designated period of time. It was built on top of both the playlist generator mentioned above as well as an existing user system. The system included capability for administrators to manage subscriptions of their users, enforcement of security on items 'managed' by a subscription, and handlers that accepted postbacks from various shopping cart systems (implemented ecwid and oneshoppingcart.com)to generate subscriptions based on user purchases.

### Event-based Analytics for Media Player

Client had need for a statistic  tracking system that offered more granularity for their media player. They had in place a simple 'hit counter' type stat system that essentially was able to track when a piece of media was 'played' and that was about it. They wanted to be able to achieve a much more detailed view of user behavior when using the media player. Essentially they needed to track certain user events such as pause, play, seek, stop, and get a general sense of how much of a piece of content was watched by a particular user.