Adeptus Astartes Implementation for

Master of Science in Information Technology

Software Design and Programming

James McKenna and Estell Moore

University of Denver University College

10/17/2024

Faculty: Greg O'Toole, PhD

Director: Gregory Reese, PhD

Dean: Michael J. McGuire, MLS

Abstract

In the development of our Adeptus Astartes Web Application Project. We decided to build out a front page framework that will tie into all our aspects of our project. Using Figma for collaboration and design purposes. This allowed us to stage out a framework that would tie into other tools and applications we would use later in the project; to fully build out the functionality of our Web App. HTML and CSS were also used after Figma to build the structure and styling of our web pages. Visual Studio Code was our integrated development environment that we used to code and help with our development of the web app. Through these tools and implementation we were able to build a very small prototype that we will continue to integrate and grow as the class progresses.

TABLE OF CONTENTS

Chapter

1. FRONT-END IMPLEMENTATION 3

2. PROCESSING IMPLEMENTATION 5

3. DEPLOYMENT AND THE CLOUD 6

4. THE PROTOTYPE APPROACH 8

REFERENCES 12

Front-End Implementation

To start with development, the group will utilize Figma for a wireframe, then will optimize the code through HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) and VSCode (Visual Studio Code). After partaking in discussions in this class, the group has found that Figma could be an incredibly useful tool for prototyping a basic layout for each website landing page.

Figma is a UX design tool that has useful collaborative capabilities where a group of people could simultaneously develop website pages (Figma. 2024b.). It has the capability to many layouts including desktop, mobile iOS/Android, smart watch displays, and more. This tool was chosen primarily because Figma can offer a UX template wireframe design that can be utilized as a reference for the actual UI design. One of the difficulties of developing a user-friendly website is positioning of UI elements. This tool can eliminate any issues with positioning, as it presents the possibility of manually dragging-and-dropping shapes to their desired areas, then can be later refined in development.

There are some drawbacks with designing a website through Figma, as it is all UX design, but no UI implementations. Element shapes can be utilized to mock a layout for a website, but there still needs to be work done after the UX portion of the design. This tool allows an export of CSS and HTML scripts to be utilized as a wireframe for the actual development. Once the export is performed, VSCode can be utilized to optimize the code with actual elements instead of simple UX shapes.

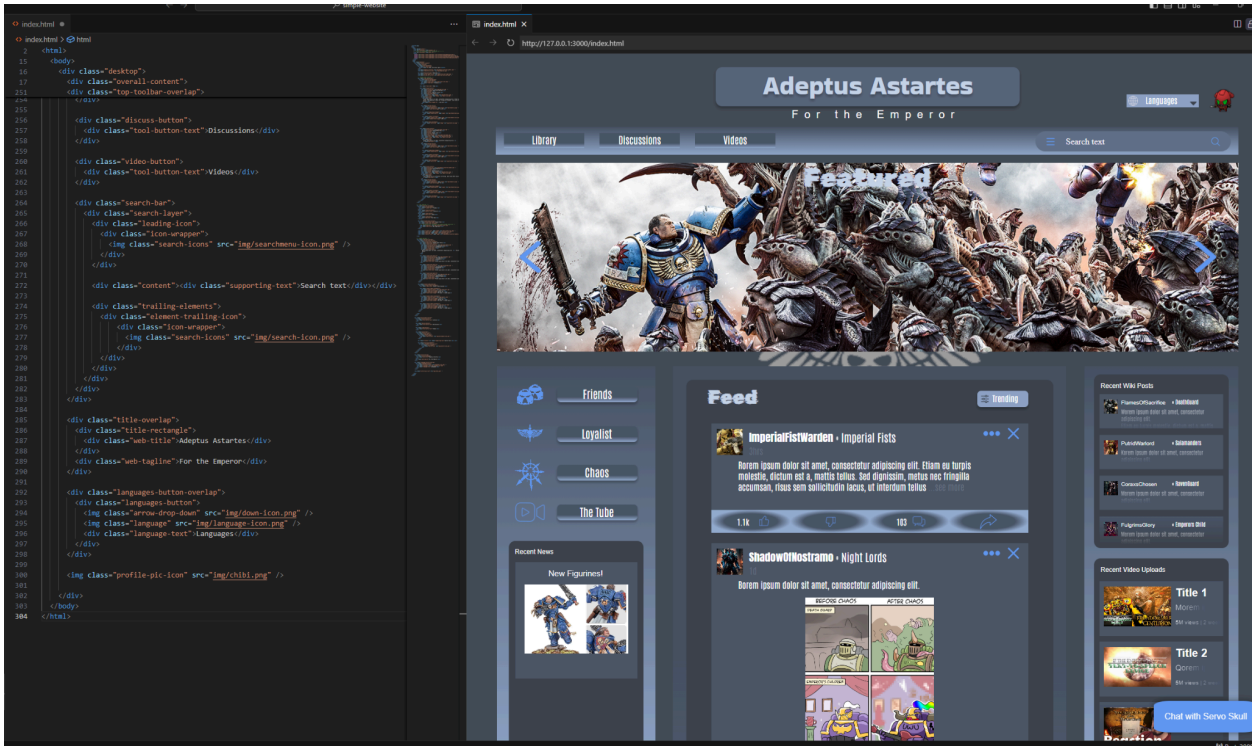

Figure 1: Example of Live Preview Plugin via VSCode

VSCode has a helpful plugin called Live Preview, which allows developers to view their web application in a side panel on the IDE (Marketplace. 2024.). The application updates as the developer manipulates the HTML or CSS code. The structure for the website, in the beginning, will follow a basic index.html and main.css structure. HTML is a language that can handle the element structures and layout of web design, while CSS handles the appearance of these elements. Another difficult decision for front-end consideration is the handling of icons and images for the web application. When developing, having a local instance of an image to utilize for the website is easy in the initial stages, but consideration for how the website will display images publicly has to be under consideration. Temporarily, for prototyping, the group is going to place icons and images inside of the web application project itself under an image folder so locally the website can access images, as well as anyone who clones down the repository.

Processing Implementation

The processing tool that will primarily be utilized is JS (JavaScript) and Node.js. JS is technically a language and can directly communicate with the index.html that provides functionality to elements. Node.js is a runtime environment that allows developers to create applications to send to a cloud, as well as connect to a MongoDB Database.

Utilizing JS and Node.js will be useful for basic processing. For example, it will assist in controlling what happens when a user interacts with a button on the home page, and send the user to a link attached to the button. JS is best used for asynchronous reaction, efficient speed for web applications, and provides scalability for a project (Matellio. 2023.). Since the scale of this project is small, JS provides a lot of open-source support and references, as well.

Node.js is going to be useful because it provides a scalable and flexible architecture that can be utilized in a cloud (Soares, Juan. 2024.). It provides the capability for setting up an event-driven JS runtime that can handle connections, as well as prevent lock-ups (Node.Js. 2024.). It has been decided that this tool will be utilized to create server-side applications that will enable the website to become a downloadable web application for desktop use. This tool will also enable this web application to connect with MongoDB via a JavaScript Driver, which MongoDB will be the main database source for the overall web application (Varshney, Harsh. 2024.).

In conjunction with Node.js, MongoDB was selected as our database of choice due to the built-in functionality it has with NoSQL databases. These seem to be more well suited and robust for unstructured data and dynamic web apps. MongoDB due to its robust handling of

storing unstructured and robust formats like JSON data; would help us with handling web based applications easier than SQL or Relational based databases.

Node.js would be used for our server side that would integrate toward our backend with MongoDB for data handling. This would allow us to have a seamless and easier transition using technologies and software that are integrated with one another; rather than having to use an API and another interface for us to handle. This also allows us to have an easier time maintaining and growing our web app. Rather than spending more time handling maintenance and scalability issues.

Deployment and the Cloud

After extensive research, for deployment purposes, the web application and website deployment tool will be AWS (Amazon Web Services). Since the web application will likely be utilized more as a social media platform with a bit of a database implementation for storing Warhammer facts, AWS was the cloud service that was decided upon; after extensive research across Google Cloud and AZURE (Amazon. 2024.).

AWS allows our project to have enhanced scalability, reliability and security by using this cloud service in particular. Other cloud services like AZURE or Google Cloud seem more geared toward commercial or business applications; while AWS can handle those, it also handles our small web app with ease of use. AWS has integrated instances called EC2 instances that would allow use to host our Node.js applications for our web app. Integrating AWS into our project would enhance the scalability, reliability, and security of the web application. This would allow for flexibility and scalability as we intend to build off our web app as we progress through development and not

have to backtrack as we make continued progress. For databases we intend and plan on using MongoDB for our robust use of backend data storage. However AWS also has a database that uses MongoDB called Amazon DocumentDB (MongoDB. 2024.), this is a service that is compatible for scaling, database operations, and security purposes using MongoDB. For our static assets and CSS files we could also use S3 which is a simple storage service used for handling files, images, videos, backups and web content. This would allow us to build up our web app flawlessly without having to worry about access to our files.

A newer service that seems to have popped up would be AWS Cognito, this would allow for a secure and more scalable solution to user authentication and profiles (Amazon. 2024b.). This would give more security and another layer of protection of our assets and infrastructure. Lastly for our cloud services we have AWS Lambda, this is used for running event-driven backend services, reducing the need for managing dedicated servers (Amazon. 2024a.). AWS seems to be the overall and final pick for our team that would allow for an easier time with robust, scalable and high performance of a cloud service for our web app.

The Prototype Approach

The prototype for the web application has been completed by following the steps of Figure 2. This figure is a basic outline of the prototype steps, including the use of Figma, CSS, HTML and VSCode. JavaScript hasn't been implemented just yet on the actual prototype, but it's something that will be worked on in the coming weeks to make it fully functional with each button. Also, the prototype only outlines the landing home page. Many other pages such as the library and user profile page are in development on wireframes/mock-ups, currently.

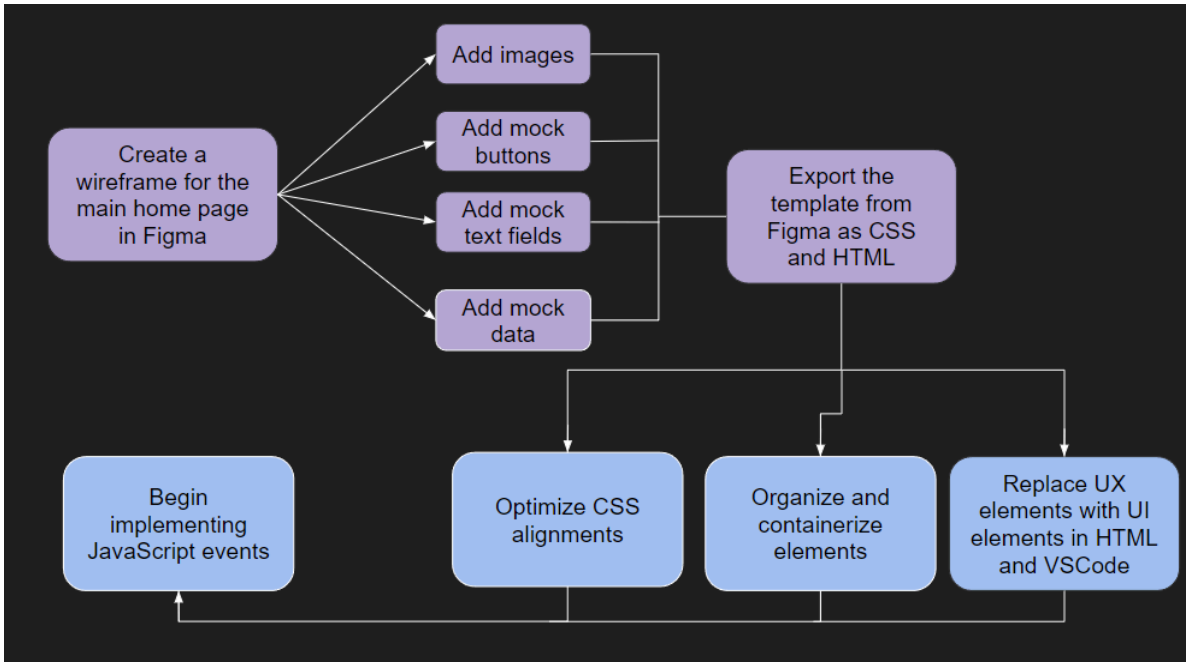

Figure 2: Prototype Approach Diagram

Firstly, the design of the home page had to be planned out by prototyping a wireframe. Wireframes are just shapes with no dimension, instead posing as a basic layout for a concept of where elements will live.

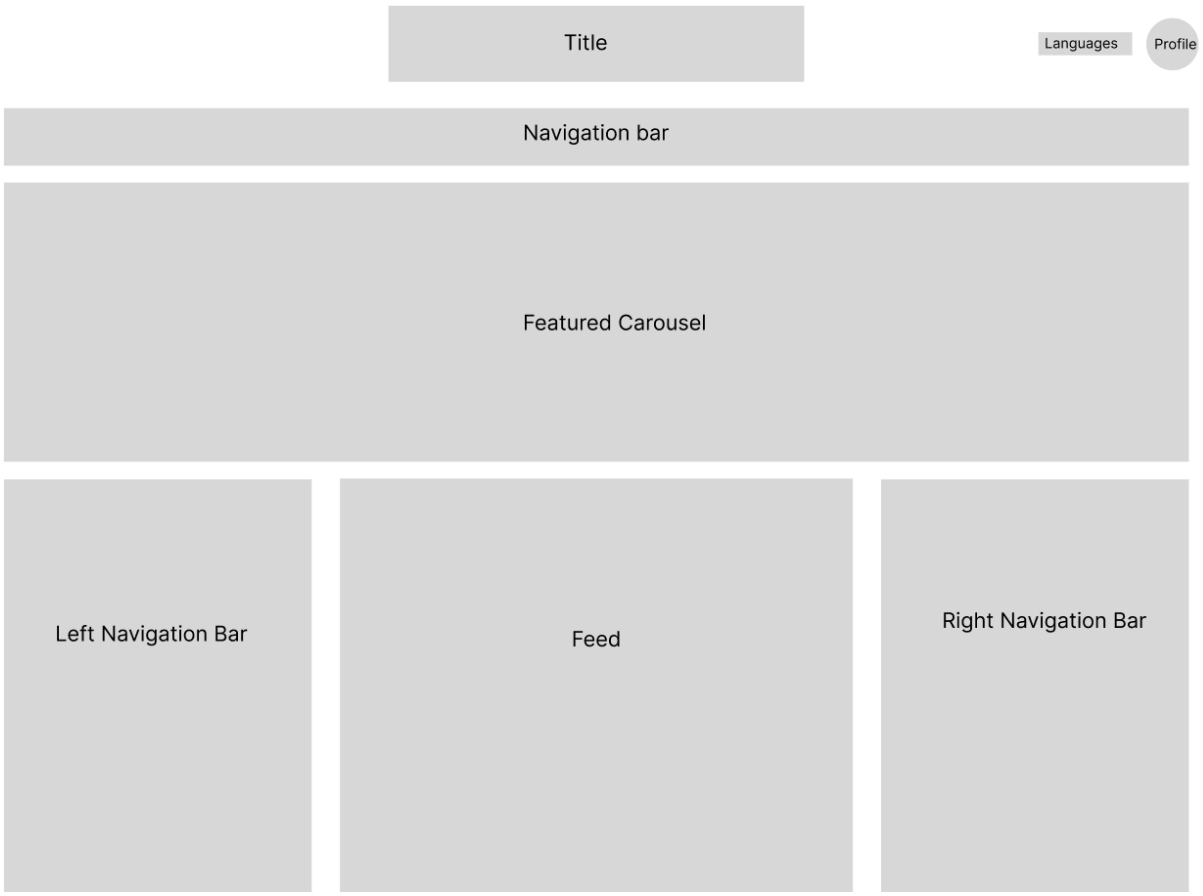

Figure 3: Wireframe Layout

Figure 3 is a screenshot for what the wireframe is in Figma. In this case, many elements needed to be planned for an intuitive layout for users: a top navigation bar, left and right navigation bar, a feed for social posts, featured carousel for featured content on the website and Warhammer updates, and a profile button at the top right, where most designed social media sites have it in the top right.

Then, a more detailed UX mock-up is implemented, where images, text, icons, and fake buttons/text-fields are implemented.

Figure 4: Detailed Mock-Up

Figure 4 is a screenshot of the mock-up designed in Figma, but all elements seen here are actually just basic shapes with no functionality. This is primarily used to mock-up how the web application and website want to be set-up for users to view, with examples of expected

content in the relative navigation bars. Though it poses as a useful template, Figure 4 has no actual buttons, text fields, links, or back-end connection to anything.

This can be changed by exporting the HTML and CSS from Figma, and refined in the IDE, VSCode. Many shape elements in the HTML needed to be replaced by actual HTML elements, such as buttons for the filter and the text field for the search widget. Although, some of the shapes exported from Figma are legitimately there just for visual enhancement, so certain elements were renamed with the simple shape name and what element it is describing. Also, containerizing some of the elements is essential to allow reuse of CSS methods. For example, the social media cards could be containerized and only utilize one CSS element instead of three separate elements for each example card. This will also make the actual real-time implementation of a social media feed easier, which would only entail filling out the fields of the element instead of creating a new one each time a user posts something. Since not every shape in Figma is perfectly aligned, the CSS is also refined by improving margin alignments more specifically with actual matching values.

References

Amazon. 2024. “Amazon Documentation Overview.” *Amazon*. Accessed October 18. https://aws.amazon.com/documentation-overview/

Amazon. 2024a. “What Is Amazon S3? - Amazon Simple Storage Service.” *Amazon*. Accessed October 18. https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html.

Amazon. 2024b. “Amazon Cognito User Pools.” 2024. *Amazon*. Accessed October 18. https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-identity-pools.html.

Figma. 2024b. “Figma.” *Figma*. Accessed October 18. https://www.figma.com/.

Marketplace. 2024. “Live Preview - Visual Studio Marketplace.” Marketplace. Accessed October 18. https://marketplace.visualstudio.com/items?itemName=ms-vscode.live-server.

Matellio. 2023. “Spring Boot vs Node JS: Choosing the Right Framework for Your Business Project!” *Matellio Inc*. Matellio Inc. December 8. https://www.matellio.com/blog/spring-boot-vs-node-js/#:~:text=js%20is%20a%20JavaScript%20runtime,applications%20of%20an%20enterprise%2Dcaliber.

MongoDB. 2024. “MongoDB Atlas Database: Multi-Cloud Database Service.” *MongoDB*. Accessed October 18. https://www.mongodb.com/cloud/atlas.

Node.Js. 2024. “Node.Js - about Node.Js®.” *Node.js*. Accessed October 18. https://nodejs.org/en/about.

Soares, Juan. 2024. “Node.Js in the Cloud: Unleashing the Power of Modern Web Development.” *LinkedIn*. August 12. https://www.linkedin.com/pulse/nodejs-cloud-unleashing-power-modern-web-development-juan-soares-ewmde/.

Varshney, Harsh. 2024. “MongoDB Javascript (Nodejs) Connector Simplified 101.” *Hevo*. October 1. https://hevodata.com/learn/mongodb-javascript/.