.Net

Capstone report on:

Captivate

Date: 29th March,2023

|  |
| --- |
| 1. Captivate Commerce: Introduction |

Captivate Commerce is an e-commerce platform designed to deliver a captivating and immersive online shopping experience to customers. Our primary goal was to incorporate innovative technologies, user-centered design, and exceptional functionality to create a platform that not only meets the ever-evolving technological advancements but also offers a unique and engaging user experience.

Our team of skilled and passionate developers have come together to craft a platform that embodies the essence of e-commerce excellence. With a strong focus on collaboration and teamwork, we have successfully developed a platform that seamlessly integrates a diverse range of technologies, including .NET, C#, Angular, SQL Server, and more. The foundation of our platform lies in the microservices architecture, which enables us to independently develop, deploy, and scale each microservice, ultimately leading to enhanced maintainability and flexibility of the platform as it evolves over time.

One of the key driving factors behind Captivate Commerce's development was our commitment to incorporating industry best practices and patterns throughout the development process. By adhering to these principles, we ensure that our platform maintains the highest level of quality, performance, and reliability.

To further enhance our website's capabilities, we have made use of advanced tools and techniques that facilitate the creation of dynamic and responsive web applications. Angular, a leading frontend framework, plays a pivotal role in our frontend development, allowing us to build modular and reusable components that result in efficient and maintainable user interfaces. In addition, we leverage the power of TypeScript, a statically typed superset of JavaScript, to develop scalable and maintainable code that can be easily understood and modified by developers.

When it comes to data management, we utilize a combination of SQL Server and MongoDB to cater to our platform's diverse data storage requirements. SQL Server serves as our primary database, providing a robust and secure solution for managing complex relational data, while MongoDB, a NoSQL database, is employed for storing order receipts in a flexible and scalable manner.

Security is a top priority at Captivate Commerce, and we have implemented role-based authorization using JSON Web Tokens (JWT) to ensure that users can only access resources and functionalities that correspond to their assigned roles. This security mechanism helps protect sensitive data from unauthorized access and maintains the integrity of our platform.

Our dedication to delivering an exceptional user experience extends beyond the core functionality of our platform. We believe that a captivating and visually appealing design is crucial for capturing the attention of our users and keeping them engaged throughout their shopping journey. We created user interfaces that are not only functional but also adhere to modern design principles, resulting in a shopping experience that is truly captivating.

In summary, Captivate Commerce is a groundbreaking e-commerce platform that combines advanced technologies, industry best practices, and a strong focus on user experience to create an unparalleled online shopping experience.

|  |
| --- |
| 1. Tech Stack used |

Each technology incorporated in this project complements and enhanced the others, ensuring a powerful and cohesive platform that delivers the best possible user experience. A well-planned technology stack is crucial for the success of any software project, as it defines the building blocks and tools that shape the application's architecture, performance, and scalability. By combining the strengths of these diverse technologies, we have created a platform that is both robust and flexible.

* 1. .Net and C#

Our backend leverages the power and flexibility of the .NET framework and C# programming language. This combination provides a solid foundation for building scalable and high-performance web applications. C# offers strong typing, object-oriented programming, and modern language features that enable our developers to write clean and maintainable code.

* 1. SQL Server:

As our primary database, SQL Server offers a robust and secure solution for managing data within our platform. This relational database management system provides high-performance querying, efficient indexing, and advanced features for managing transactions and data integrity, ensuring the best possible performance and reliability.

* 1. MongoDB:

As our primary database, SQL Server offers a robust and secure solution for managing data within our platform. This relational database management system provides high-performance querying, efficient indexing, and advanced features for managing transactions and data integrity, ensuring the best possible performance and reliability.

* 1. Angular:

Angular is a powerful and modern frontend framework that enables us to create dynamic and responsive web applications. With its component-based architecture, Angular promotes modularity and reusability, making it easier to develop and maintain complex user interfaces.

* 1. Typescript:

TypeScript is a statically typed superset of JavaScript, adding optional static typing and modern language features to the widely used language. This allows our developers to write more maintainable and scalable code by catching errors early and providing better tooling and code navigation.

.

* 1. HTML and CSS:

We utilize HTML and CSS to create visually appealing, responsive, and accessible user interfaces that adhere to modern design principles. Our design team collaborates closely with developers to create user experiences that are both functional and captivating.

* 1. Git:

Git is our chosen version control system, enabling efficient collaboration among team members and tracking changes to our codebase. This ensures that our development process is smooth, and we can easily revert to previous versions if needed.

* 1. Visual Studio & Visual Studio Code:

Our development team utilizes both Visual Studio and Visual Studio Code as Integrated Development Environments (IDEs) to write, debug, and deploy code efficiently. These powerful tools offer features like IntelliSense, code refactoring, and integrated debugging, which streamline our development process.

|  |
| --- |
| 1. Practices Used |

Adhering to industry-standard best practices is essential for the success and long-term maintainability of any software project. Implementing these practices ensures that a project is built on a solid foundation and can adapt to changing requirements, scale with user growth, and remain maintainable throughout its lifecycle.

* 1. CQRS Pattern:

The Command Query Responsibility Segregation (CQRS) pattern is a key architectural principle in our microservices. This pattern separates read and write operations, optimizing performance by allowing us to scale each operation independently and enabling better caching strategies for read-heavy workloads.

* 1. Microservices:

Our platform is built around a microservices architecture, which breaks down the application into smaller, independently deployable services. This approach provides better scalability, maintainability, and flexibility, allowing us to evolve and adapt the platform more easily.

* 1. API Gateway:

The API Gateway serves as the entry point for all our microservices, streamlining communication between services and clients. It provides a single access point, simplifying the client-side code and enabling features like load balancing, authentication, and request routing. By acting as an intermediary between clients and services, the API Gateway helps to maintain the security and stability of our system.

* 1. Clean Code:

Our development team adheres to clean code principles, which emphasize readability, maintainability, and modularity. Writing clean code involves using meaningful names for variables and functions, adhering to the Single Responsibility Principle, and keeping functions small and focused. This approach reduces technical debt and makes it easier for team members to understand and modify the codebase.

* 1. Angular Services:

In our Angular frontend, we employ a services-based approach to separate business logic from presentation logic. This involves creating services that are responsible for interacting with APIs, handling data manipulation, and performing other tasks that do not directly involve user interface components. By adopting this approach, we maintain a clear separation of concerns, allowing for easier testing, maintainability, and modularity.

* 1. Role Based Authorization:

Our platform uses JSON Web Tokens (JWT) to implement role-based authorization. This security mechanism ensures that users can only access resources and functionalities appropriate for their role (admin, user, or visitor). By implementing role-based access control, we maintain a high level of security and protect sensitive data from unauthorized access.

* 1. MediatR:

We use the MediatR library to implement the mediator pattern in our microservices, which helps to decouple components and reduce dependencies between them. With MediatR, we can define commands, queries, and event handlers, simplifying the communication between components and making it easier to modify and extend the system.

* 1. Unit Testing with Moq:

To ensure the reliability and maintainability of our platform, we have implemented comprehensive unit tests using Moq and xUnit frameworks. Moq is a popular mocking library for .NET that allows us to create mock objects for testing purposes, while xUnit is a widely used testing framework. By incorporating these tools, we can validate the behavior of individual components and catch potential issues before they become critical.

|  |
| --- |
| 1. Flow of the Project |

Captivate Commerce provides a seamless and user-friendly flow for visitors, customers, and administrators alike. Here's a detailed walkthrough of the platform's functionalities, showcasing the integration of the technology stack and best practices used throughout the development process.

* 1. Home Page:

When a visitor first arrives at Captivate Commerce, they are greeted with a visually appealing homepage showcasing the available products. The intuitive navigation bar provides options for logging in or signing up. If the visitor attempts to add a product to their cart, they are prompted to log in or sign up to continue.

A picture containing text, screen, different

Description automatically generated

Graphical user interface, website

Description automatically generated

Graphical user interface, application

Description automatically generated

* 1. Login Page:

The login page allows users to enter their username and password to access their accounts. The system employs role-based authorization using JWT tokens to ensure secure and personalized experiences for different users.

Graphical user interface, application, website

Description automatically generated

* 1. Signup page:

If the user doesn't have an account yet, they can sign up by filling out a registration form with their basic details. This process is facilitated by the .NET backend, which securely stores user information in the SQL Server database.

Graphical user interface, application

Description automatically generated

* 1. User Home Page:

Once logged in, users return to the homepage, now with updated navigation options to view their cart and log out. They can browse through the products, with Angular-powered components providing a smooth and responsive browsing experience.

A screenshot of a cell phone

Description automatically generated with low confidence

Graphical user interface, website

Description automatically generated

* 1. View Product:

By selecting a product, users are directed to the view product page, which displays detailed product information. Here, they have the option to add the item to their cart.

Graphical user interface

Description automatically generated

* 1. Cart Details:

The cart page provides an overview of the user's selected products, along with their quantities and total price. A checkout button leads to the shipping details page, where the user will provide their delivery information.

Graphical user interface, website

Description automatically generated

* 1. Checkout Page:

The checkout page prompts users to fill out their shipping details, captured and securely stored using the .NET backend. Upon confirmation, a "Place Order" button appears.

Graphical user interface, text, application

Description automatically generated

* 1. Order Summary Page:

After placing the order, users receive an order summary displaying the details of their purchase. This receipt is stored in the MongoDB database, taking advantage of its flexibility and scalability for handling semi-structured data.

Graphical user interface, text, application

Description automatically generated

* 1. Admin Dashboard:

If an admin user logs in, they gain access to a specialized dashboard that offers CRUD (Create, Read, Update, Delete) functionality for both user and product management. The admin dashboard also provides search capabilities, allowing admins to easily locate specific users or products by name, category, or other attributes.

A picture containing graphical user interface

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

|  |
| --- |
| 1. Conclusion |

In conclusion, Captivate Commerce represents the culmination of hard work, collaboration, and dedication to excellence by our team. Throughout the development process, we faced numerous challenges and overcame obstacles that have not only strengthened our technical skills but also our ability to work effectively as a team.

We would like to express our gratitude to our training manager, Moin, for his guidance and support throughout the project. We are also grateful to our HR representative, Shrishti, and the entire Learning and Development team for providing us with the resources and opportunities necessary for our growth. A special thanks goes to our skilled trainer, Kishore, who played an instrumental role in helping us understand and master the various technologies used in this project.

Our batchmates have also been a constant source of inspiration, encouragement, and camaraderie, creating an environment that fostered learning and innovation. The challenges and experiences we shared together have undoubtedly contributed to our success as a team and enhanced our confidence in working with the chosen technology stack.

This project has significantly improved our overall development skills and has prepared us to tackle more complex projects in the future. As we continue to advance in our careers, we will carry the valuable lessons and insights gained from this experience, using them to shape our approach to problem-solving and collaboration.

Thank You!