A diagram of a cat

Description automatically generatedA diagram of a diagram

Description automatically generated

1. **"Starting with the User Experience":**

Imagine you open the website to adopt a cat. From the moment you load the page, everything runs smoothly. The front-end is built with **React**, designed to be lightweight and responsive, ensuring that whether you're browsing through dozens of cats or narrowing down your favorites, everything works seamlessly.

**React Router** manages navigation between pages, enabling users to switch between cat listings, their profile, or the cart without reloading the page. The goal here was to ensure the user has a fast, enjoyable experience while interacting with the site.

1. **"Building a Secure and Scalable Backend":**

Every click you make on the site sends a request to the **Node.js** and **Express** backend. One of the most challenging parts of the backend was ensuring it could handle multiple users simultaneously while maintaining security.

For example, when a user registers or logs in, the **Auth API** handles the request. To secure user sessions, we used **JWT (JSON Web Token)**, which acts like a digital passport. This decision was made because JWT allows for stateless authentication, meaning the server doesn't have to store user session data, which improves scalability and security.

By generating a **JWT token** upon login, we ensured that each subsequent action, whether it's adding a cat to the cart or making a purchase, is securely linked to that specific user. This added a layer of protection against unauthorized access.

1. **"Challenges with Managing State and Data":**

One challenge we encountered was ensuring that data, like cat listings or a user's cart, was consistently updated across the app. Initially, we considered using the **Context API** but soon realized that **Redux** was better suited for handling complex state management, especially with multiple components interacting with shared data.

Redux allowed us to maintain a global state, ensuring that any updates (like adding a cat to the cart) were instantly reflected across the site, regardless of where the user was.

1. **"Handling Payments with Security and Ease":**

When users decide to adopt a cat, they go through the payment process. This was a critical part of the app because it needed to be both secure and intuitive. To handle payments, we integrated **Stripe** via the **Payment API**.

Stripe was chosen because of its robust security features and ease of integration with the Node.js backend. The biggest challenge here was ensuring that all payment data was handled securely without adding too much complexity to the user experience.

By securely processing payments via Stripe, we ensured that users could adopt cats with confidence, knowing their payment details were protected. The **Order Service** then takes over, confirming the order and updating the database.

1. **"Keeping User Data and Transactions Safe":**

Security was a top priority throughout this project. Every sensitive action, from login to payment, goes through a **verification process** using JWT tokens. The **VerifyToken middleware** acts as a gatekeeper, ensuring that only authenticated users can make changes to their account or place orders.

For example, when a user wants to view their cart, the **Cart API** checks the JWT token to verify their identity before retrieving their saved items. This protects users' data and ensures no unauthorized access occurs.

1. **"A Robust and Scalable Database with MongoDB":**

Behind the scenes, all data—whether it's user profiles, cats up for adoption, or shopping cart contents—is stored in **MongoDB**. We carefully designed the database schema to allow for flexibility, ensuring that as the app grows, it can accommodate new features or data.

For example, each user's cart is stored in a **Cart Model** that tracks the cats they’ve selected for adoption, and the **Order Model** keeps track of purchases. **MongoDB**’s scalability ensures that even as more users adopt cats, the database can handle increasing loads without performance issues.

One of the major challenges was implementing secure authentication with JWT while keeping the experience seamless for users. Ensuring state consistency across multiple components with Redux also required careful planning.

Another significant challenge was integrating **Stripe** for payments. It required meticulous attention to detail to ensure payments were processed securely while maintaining a smooth user experience.

Throughout the process, collaboration with team members using **JIRA** for task management helped maintain a clear roadmap, and **CI/CD pipelines** ensured that the code was always deployed and tested efficiently.