Cyber Recycling Inventory Management

Lane Crouch

Introduction

- The Cyber Recycling project at Enactus serves both the Union community and the Jackson community
- Donation → wiping → Linux Lite installation → System configuration → Distribution
- A personal computer is essential to student success in college
- We have also distributed computers to local businesses

Introduction

- A useful tool for Cyber Recycling is an inventory management system; something we currently lack. With it, it would be easier to see what the project does and does not have.
- My goal for this project was to produce a cloud-based web application for this purpose that could be easily used and maintained.
- Amazon Web Services (AWS) was chosen on advice from Dr. Li, my project advisor, and due to his familiarity with it.

Amazon Web Services

- Amplify: “An opinionated, category-based client framework for building scalable mobile and web apps.”
- AppSync: “AWS AppSync [lets] you create a flexible API to securely access, manipulate, and combine data from one or more data sources.”
- DynamoDB: NoSQL database
- S3: Simple Storage Service
- And more

React & GraphQL

- “React: A JavaScript library for building user interfaces”
- Create React App: A program for abstracting away build tools and dependencies
- Material-UI: A React framework for building UI components
- GraphQL: “A query language for APIs.”

Beginning the Project

- Difficulty figuring out where to start
- Create React App tutorial (https://facebook.github.io/create-react-app/docs/getting-started)
- AWS Amplify React tutorial (https://aws-amplify.github.io/docs/js/react)

Pulling The App Together

Some more tutorials which helped me quickly get the app off the ground:

- https://medium.freecodecamp.org/going-serverless-with-react-and-aws-amplify-development-environment-set-up-9b15c3363bd
- https://medium.com/@jameshamann/serverless-graphql-react-app-using-aws-amplify-part-one-8ff92d3705e7
- https://medium.com/@jameshamann/serverless-graphql-react-app-using-aws-amplify-part-two-1a2ea2b0439

The Small Problem

Edit dialog not allowing editing

- Controlled vs. uncontrolled React components
- Setting “value” property overrides natural form field behavior
- Solution: set defaultValue instead to avoid the complexity of controlled components

The Big Problem

Front-end (React) and back-end (AppSync/GraphQL) apparently not on speaking terms

- Add, Edit, and Delete dialogs working earlier in development, but no longer
- No errors detected, just silent failure
- It’s unclear exactly how debugging works in this system
  - AWS CloudWatch

Attempted Solutions

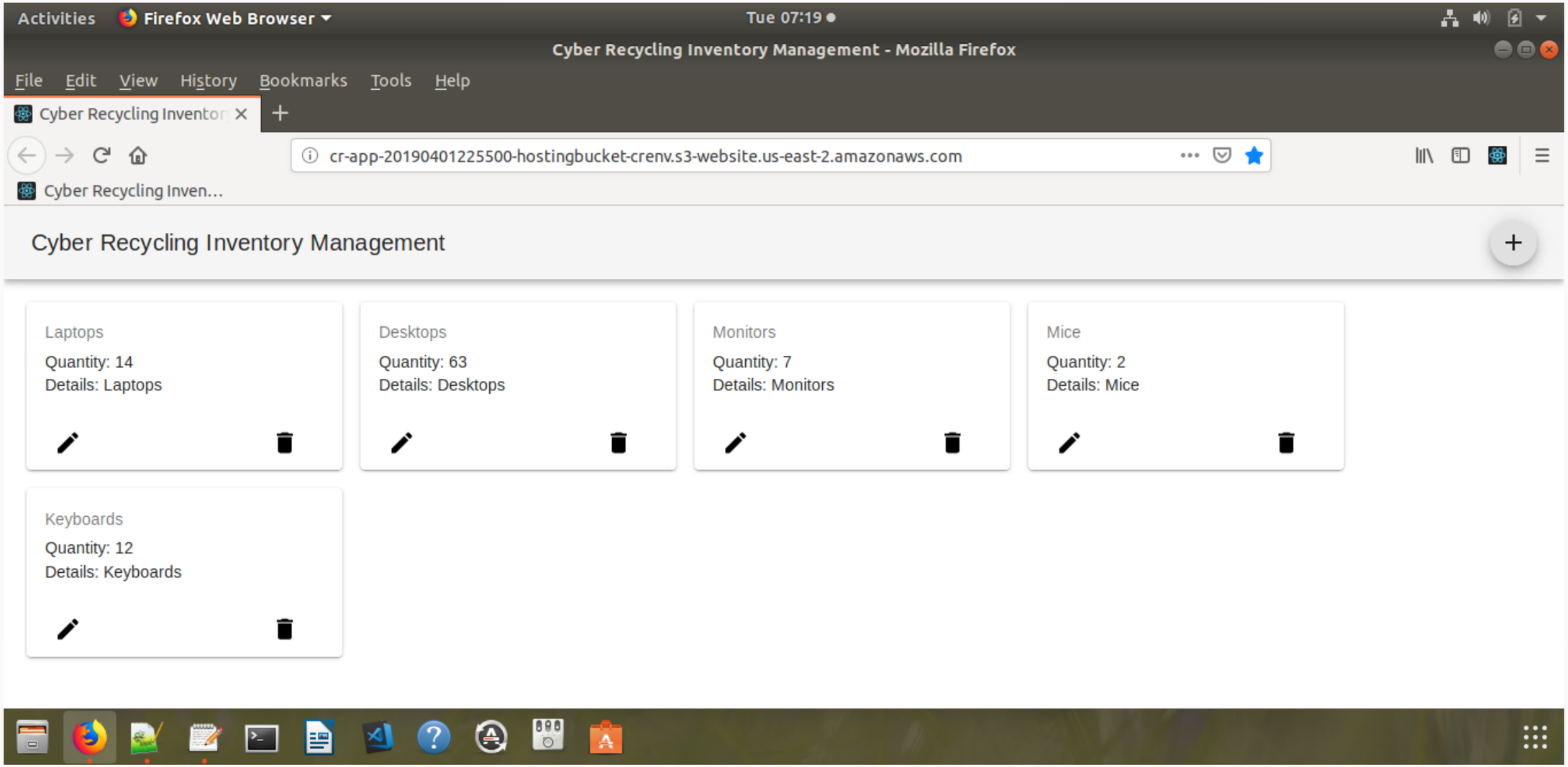
- Is the form data being properly passed to the AppSync API?
  - Yes; printing to console just before API call shows this.
  - API.graphql(graphqlOperation(mutations.createCategory, {input: submission));
  - This is an Amplify method- no documentation from Amazon.
  - Code snippets from other people seem to match mine.
- Other attempts

Conclusion

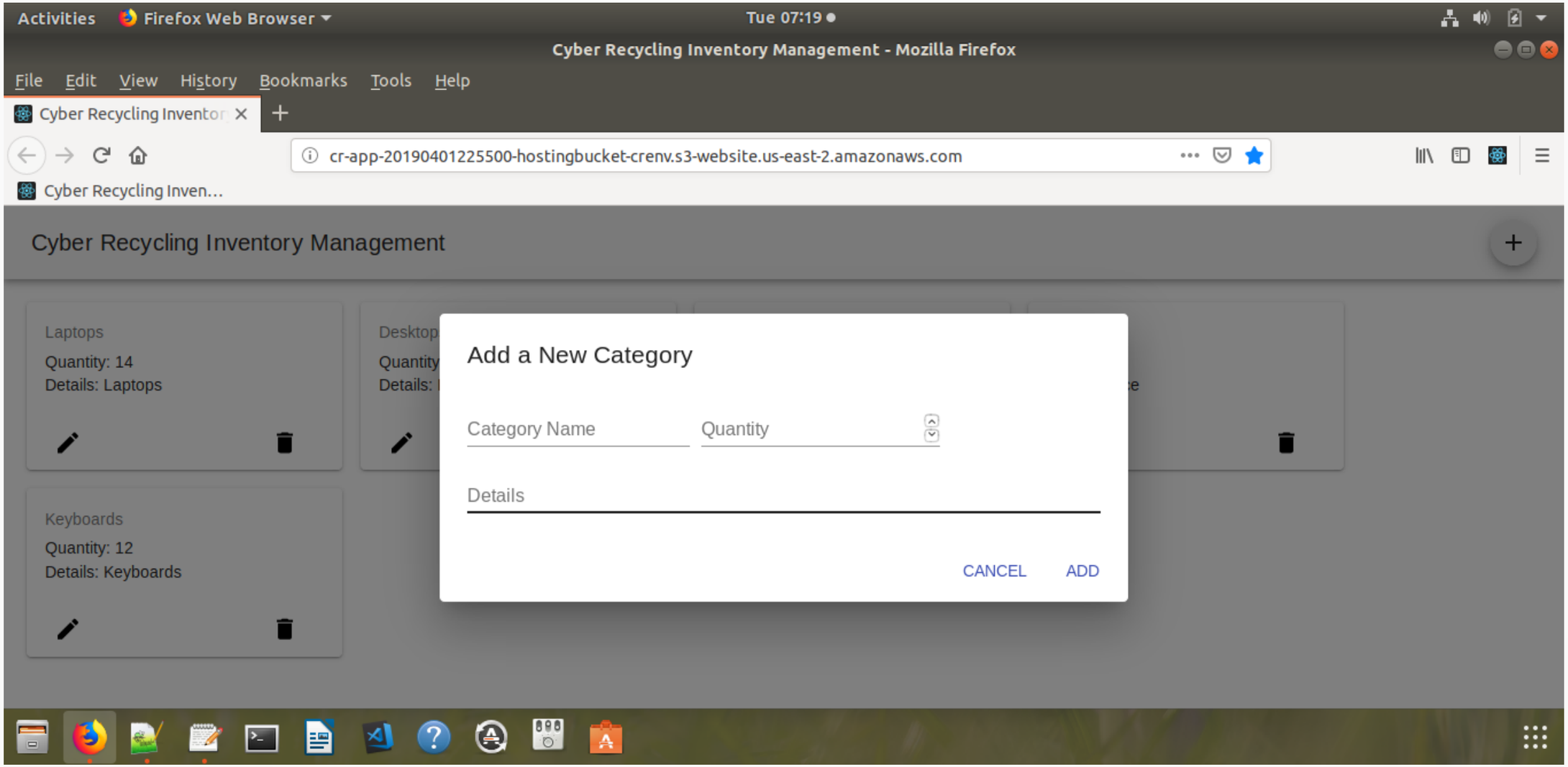
- AWS seems very powerful but also very complicated.
- Using one service is simple and straightforward, but integrating them is difficult.
- The documentation that I found is lacking.
- However, documentation for React and GraphQL was abundant, and learning them is rather simple.

Project Demonstration

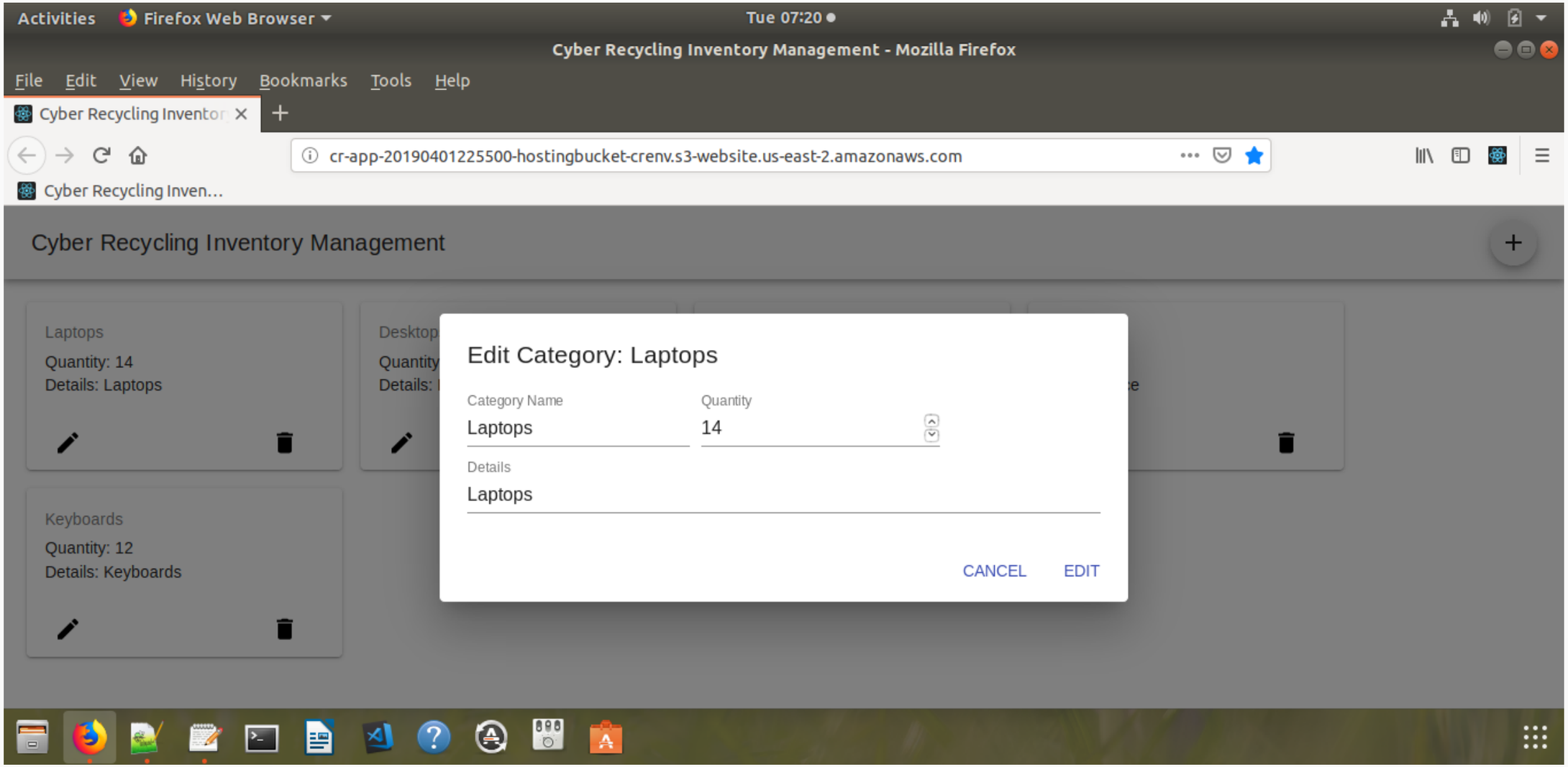
ListCategories



AddCategory



EditCategory



DeleteCategory