# CALICO PRESENTATION

MY EXPERIENCE

CLEON MULLINGS

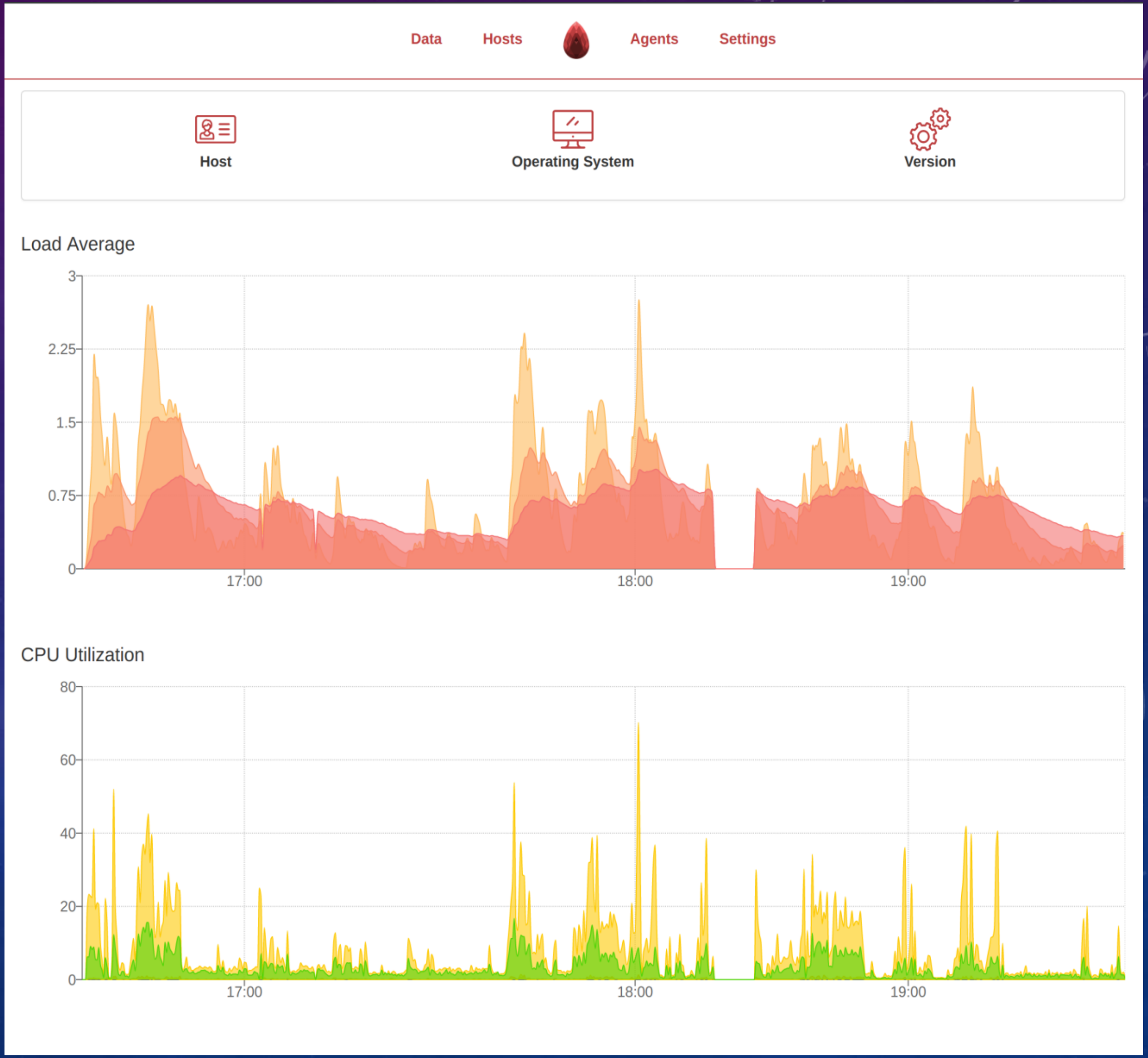
## WHO AM I?

- My name is Cleon J. Mullings, a first year student at the University of the West Indies who is pursuing a double major in Software Engineering and Medical Physics. I enjoy math, and tinkering with electronics.

## WHAT I WORKED ON

- garnet-ui - Garnet is React based a dashboard which visualizes time series data stored by infosetDB.

React



## WHY I CHOOSE GARNET

- Wanted to foray into front end development, previously I didn’t have much experience with Frontend web development(esp. JavaScript).

## THE PROBLEM

- Previously with React, components managed their own state but as new features /components are added state management can become tedious.

## THE SOLUTION

- *Redux is a predictable state container for JavaScript apps.*

## COMPONENT/CONTAINER

- Before implementing Redux, garnet was organized into Container/Component structure
- Common approach for writing React apps is to structure components using the "container / presentational" pattern:
- **"Container“(smart) components**: responsible for fetching some kind of data, and passing it as props to presentational children
- **"Presentational“(dumb) components**: render some portion of UI solely based on props from parent. Usually no meaningful logic.

http://krasimirtsonev.com/blog/article/react-js-presentational-container-components

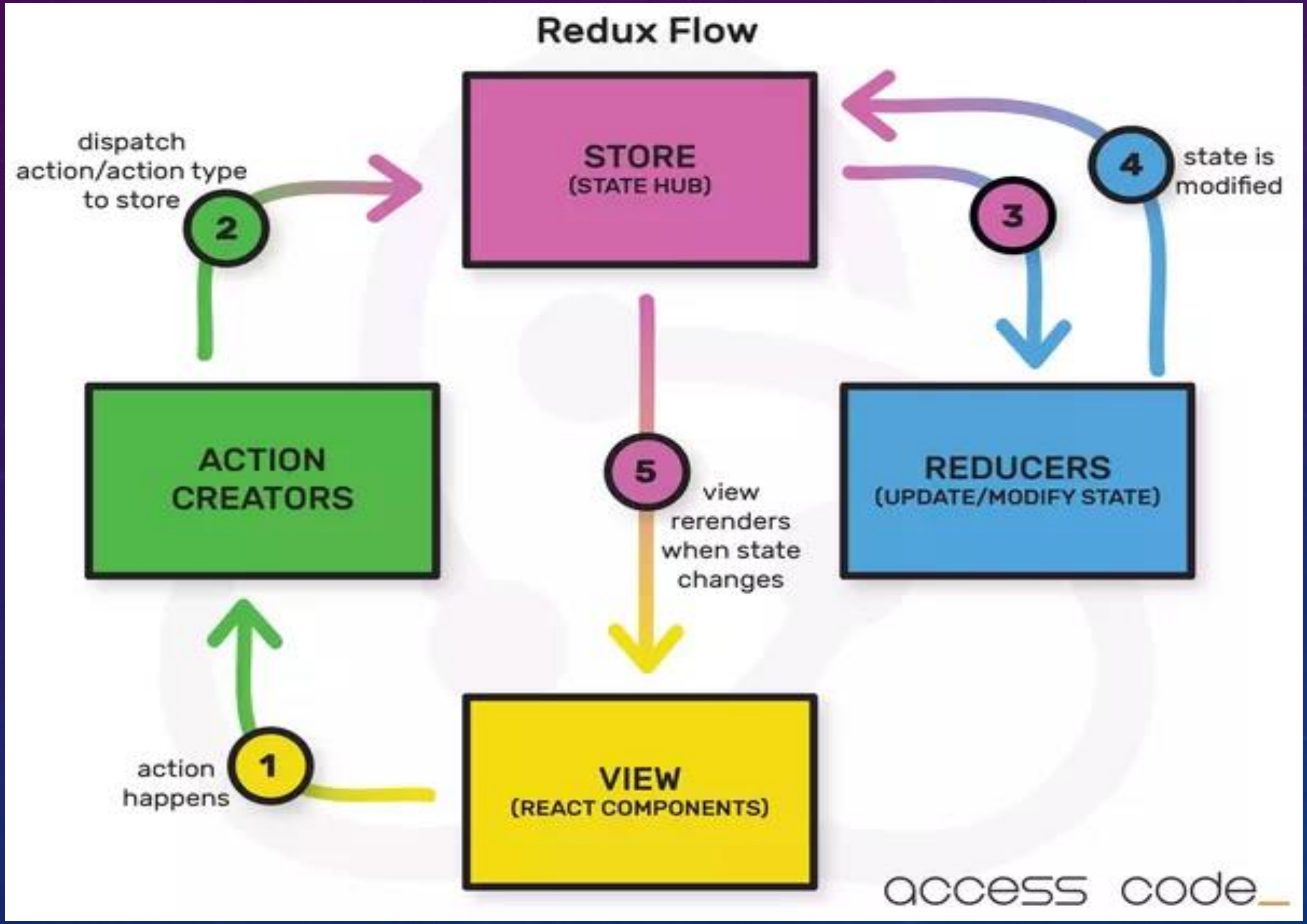
## WHY WAS THE COMPONENT/CONTAINER STRUCTURE NEEDED?

- Redux

Redux keeps the **state** of the app in a single **store**. So when data  is needed it is extracted from the state and passed to the components as props. This lets us keep data in one global place (the store) and feeds it directly to *any* component in the app, without the gymnastics of passing props down multiple levels.

The loose coupling of container allows for reusability and  the ease of replacing components. i.e., If we decided to make changes to the UI.

## HOW DOES REDUX.JS WORK?



## REDUX CORE CONCEPT

- Redux maintains the state of an entire application in a single immutable state tree (object), which can’t be changed directly. When something changes, a new object is created (using actions and reducers).

Actions were created for created for the fetching of chart data from infosetDB,

- FETCH_CHART_DATA_SUCCESS,
- FETCH_CHART_DATA_REQUEST,
- FETCH_CHART_DATA_ERROR

An async function(fetchChartData()) was created to dispatch the above actions to the store.

A reducer was also implemented to make changes to the store.

- First a request is made to infosetDB  using  fetchChartData() and upon success the JSON data will be dispatched to state using the reducer otherwise an error will be dispatched. In either case the presentational component (chart) will  not be concerned with fetching  of data it simply accesses the props.

## SKILLS LEARNT

- Got a better understanding of Version Control with GitHub/Git., React , Redux ,CSS styling.

## IN CONCLUSION

- My experience with Calico was enjoyable, and informative, I worked with my mentor Jordan Jones who was instrumental in my understanding key concepts I needed to complete my assigned tasks.
- It gave me an greatly appreciated insight into the realm of professional software development and open source collaboration.