Implement a ReactNative **Mobile App**

Learn how to implement a React Native mobile application using NodeJS, JamStack, and Astra DB

LEVEL UP with the DataStax Developers

DataStaxDevs: Developer Advocates Team

Cedrick Lunven

Aleksandr Volochnev

Jack Fryer

David Gilardi

Stefano Lottini

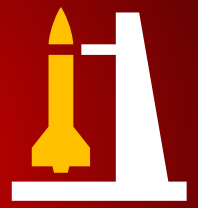
Ryan Welford

Rags Srinivas

🇫🇷 U B S T S

Agenda

01 HouseKeeping



02 Overview

03 React to React Native

"Front-end"

04 DataStax Astra React to React Native

"Config & Back-end"

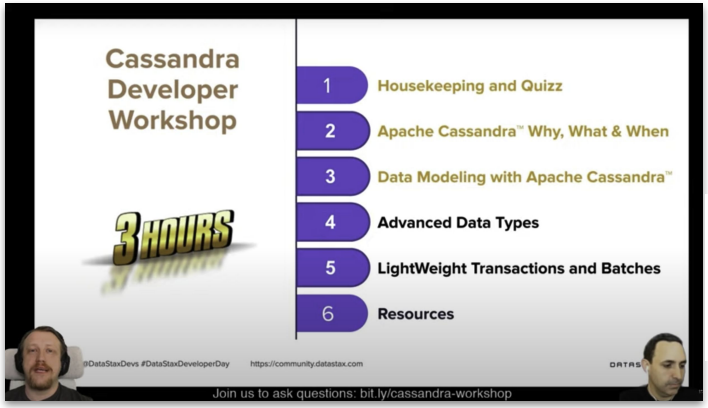
05 Native Features

06 Resources

Housekeeping #1: Attending the Workshop

Live and Interactive!

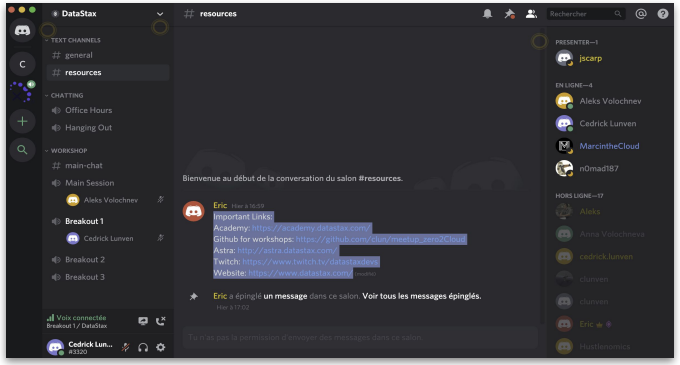
Livestream: youtube.com/DataStaxDevs



YouTube

Twitch

Questions: https://dtsx.io/discord

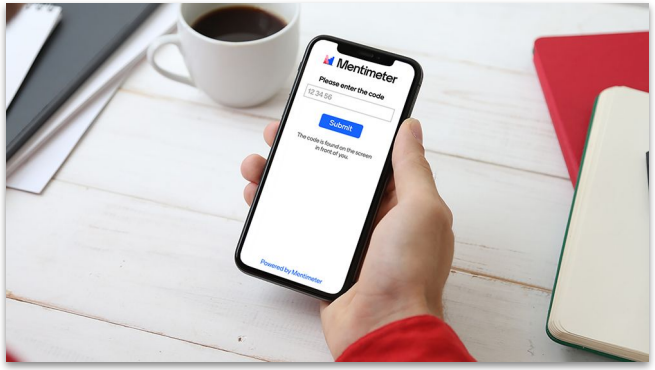


Discord

YouTube

Available on the iPhone App Store

Games menti.com



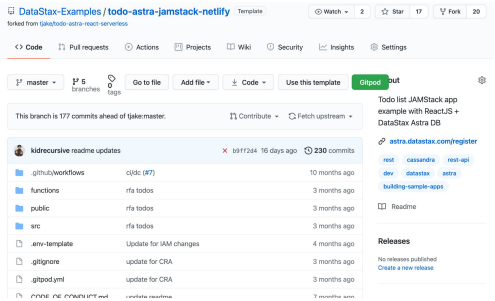
Mentimeter

GET IT ON Google play

Housekeeping #2: Doing Hands-On

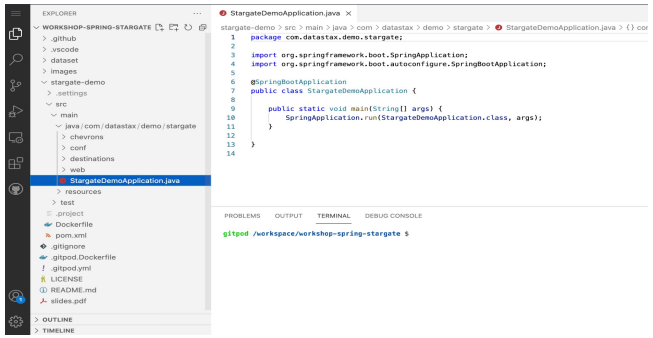
Only install Expo Go!

Source code + exercises + slides



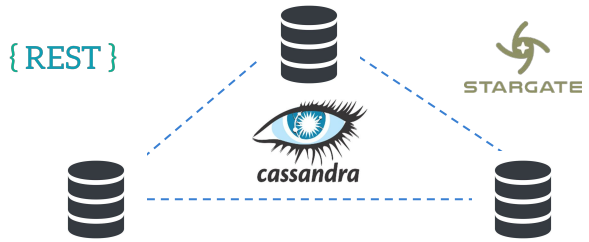
GitHub

IDE



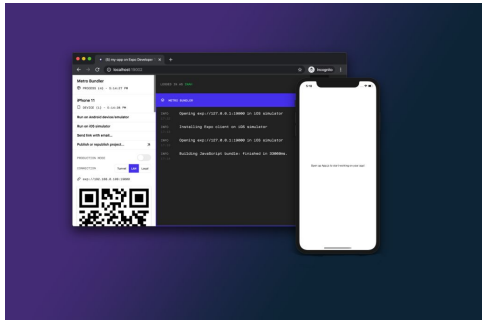
Gitpod

Database + GraphQL + PlayGround



DataStax Astra DB

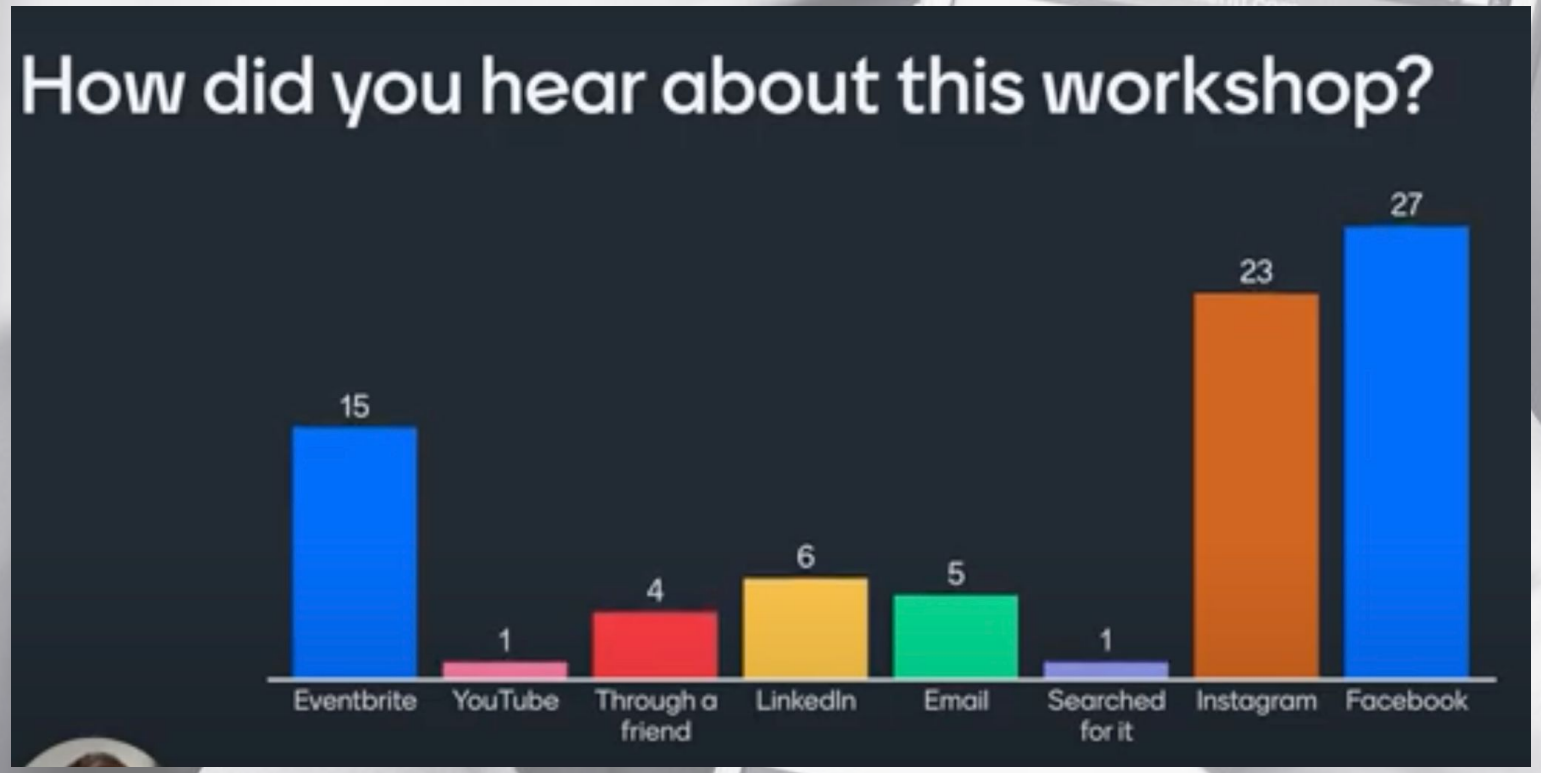
React Native CLI



Expo

DataStax Developers

menti.com First Questions!



Mentimeter

Demo #1:

- View mobile native Todo application

GitHub

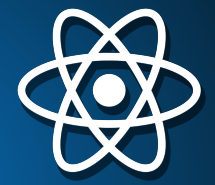
https://github.com/datastaxdevs/workshop-todo-native-mobile

Agenda

01 HouseKeeping

02 Overview

03 React to React Native "Front-end"



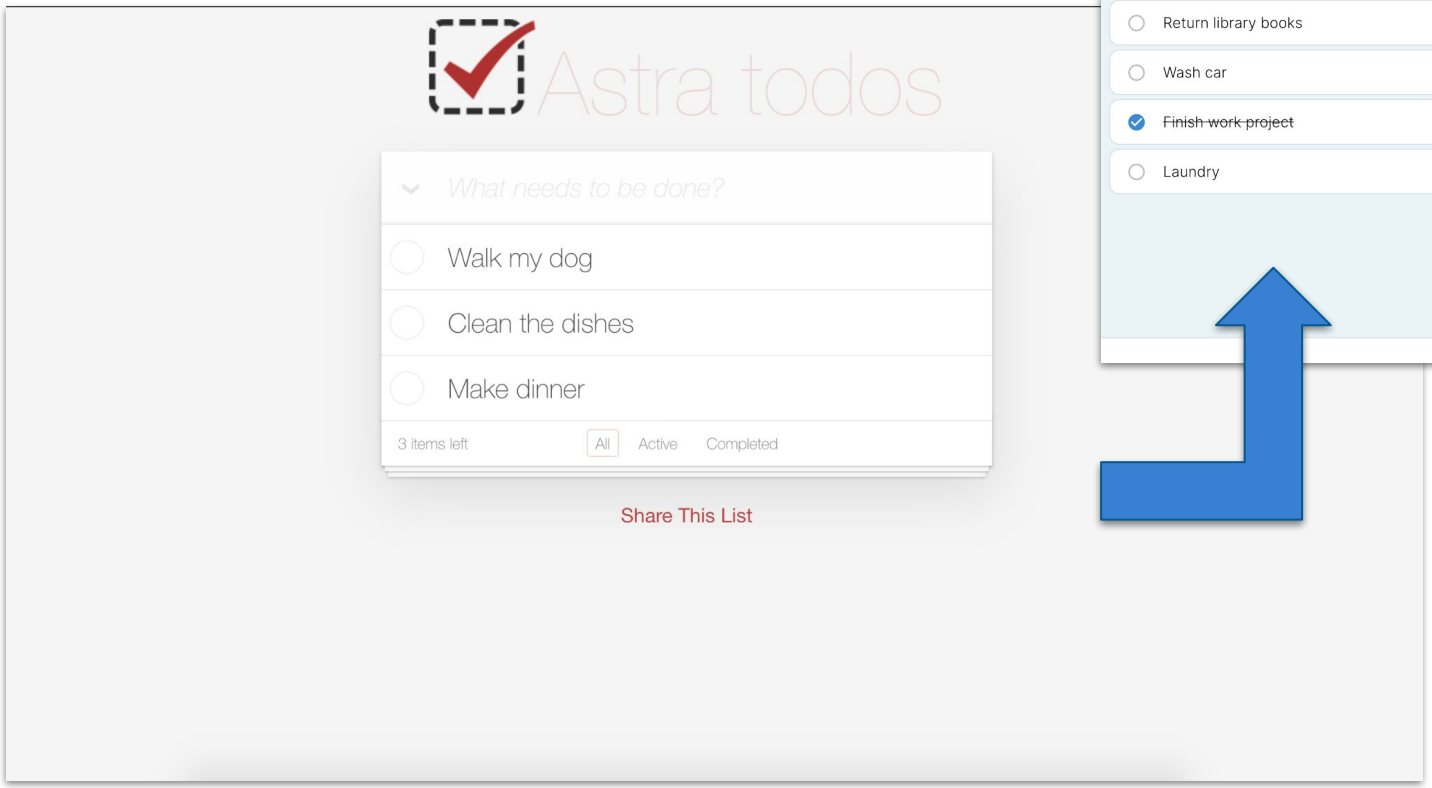
04 DataStax Astra React to React Native "Config & Back-end"

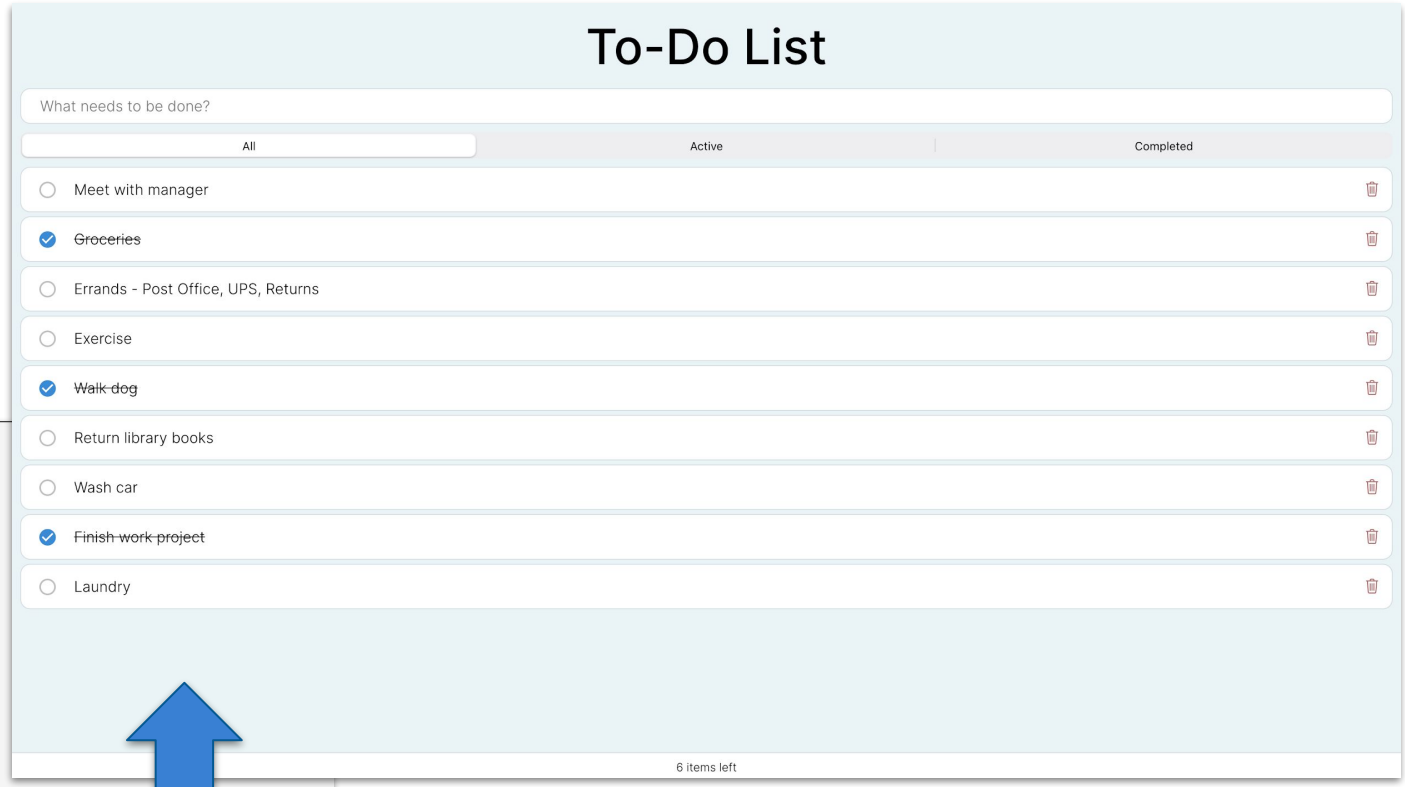
05 Native Features

06 Resources

What You Will Build: React Native Web & Mobile Applications

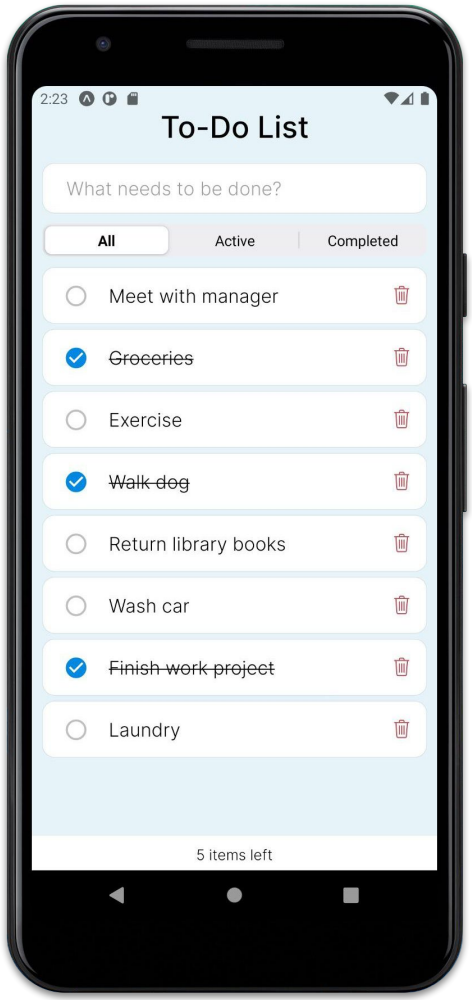
BEFORE: React Web Application



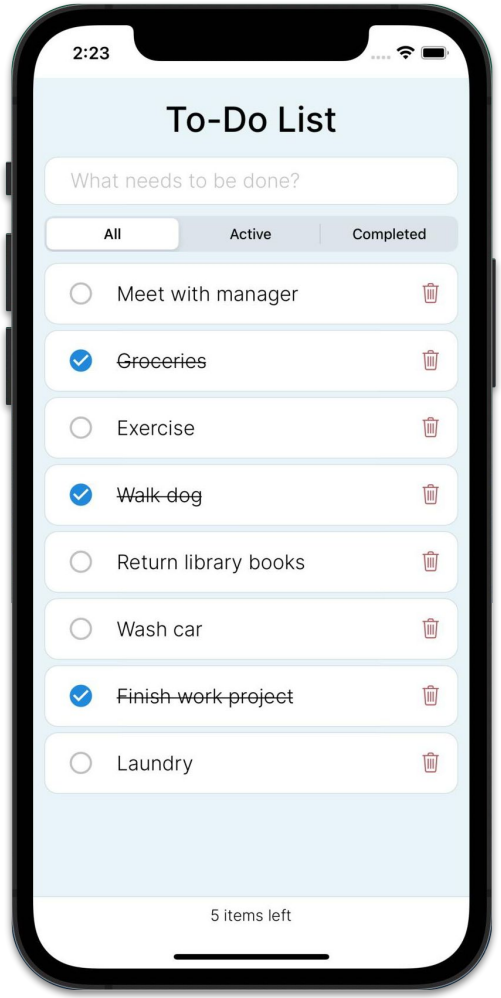


AFTER: React Native Web Application

What You Will Build: React Native Web & Mobile Applications



React Native Android Application



React Native iOS Application

Native Application Development: Rationale

- **Without React Native and Flutter:** if you wanted to build a responsive, native application for mobile and web, you need to build the same app, in three different languages, on three different platforms.
- **Example**: Swift for iOS in Xcode, in Java or Kotlin for Android in Android Studio, and in JavaScript for Web in a range of IDEs.
- **With React Native**: no need to build the same application three times!

Native Application Development: Flutter vs. React Native

Flutter

Expo

- **Starting Point:** React-based web application in JavaScript, HTML, and CSS
- Chose to use **React Native** and **Expo** for converting the application to native
- **Flutter** would require converting the entire infrastructure to **Dart**

Expo: Toolkit for React Native Application Development

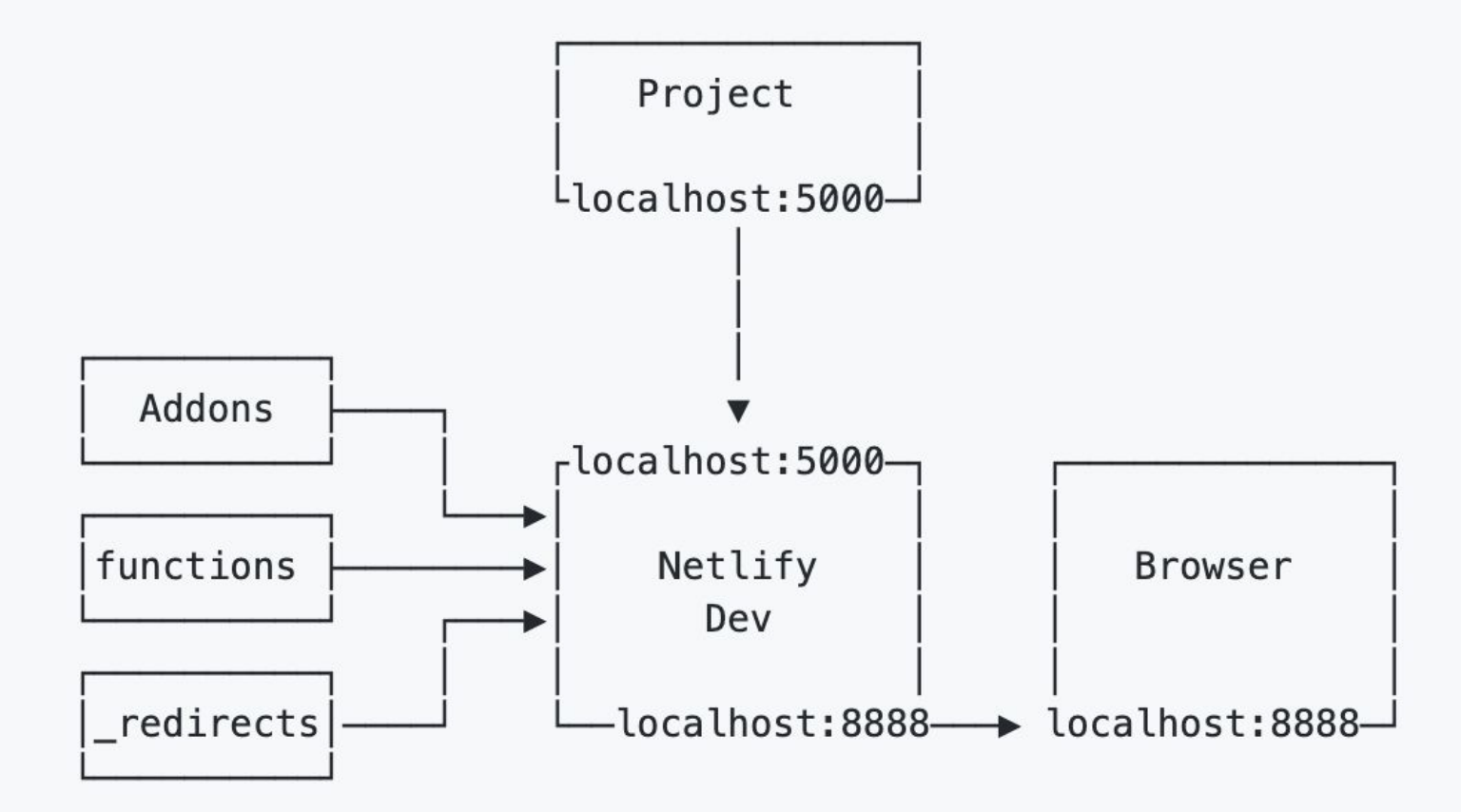
- Develop, build, deploy, and quickly iterate on iOS, Android, and web apps from the **same JavaScript/TypeScript codebase**, all using React and React Native components.
- **Mobile Expo Go App** acts as your simulator so you don't need to download both Xcode and Android Studio
- Also offers **Snack**, a playground in your browser, on which you can enter code snippets and view the output in a browser-based emulator for web, Android, and iOS

BackEnd: Networking with Expo

Learn more here.

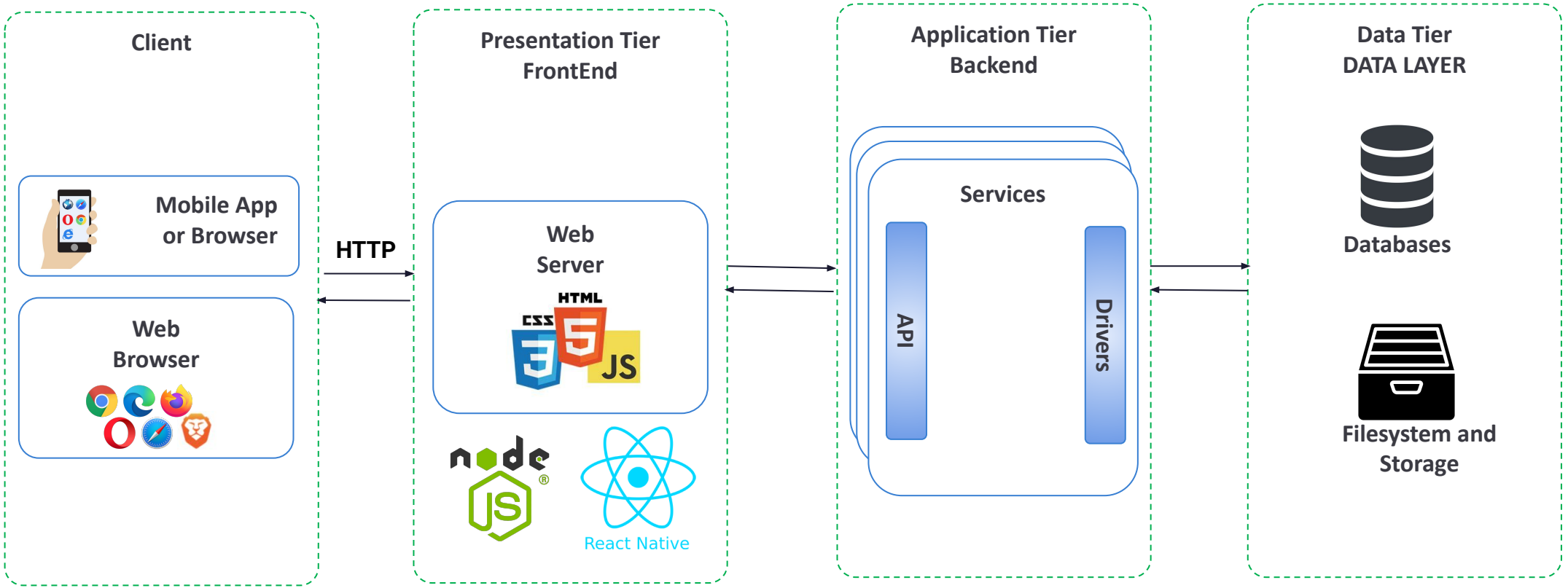
- When you start an app with Expo CLI, you're running the Expo Development Server and Metro bundler.
- **Dev Server:** endpoint that you hit first when you type the URL into the Expo app. Its purpose is to provide a communication layer between Expo CLI and the Expo app on your phone or simulator.
- **Metro Bundler:** serves your app JavaScript compiled into a single file and translates any JavaScript code that you wrote which isn't compatible with your phone's JavaScript engine. (react-native)
- Expo CLI also spawns a t**unnel process**, which allows devices outside of your LAN to access the above servers without you needing to change your firewall settings. (expo start --tunnel)

BackEnd: Networking with Netlify



- The command we use to run Netlify, netlify dev, runs Netlify's production routing engine in a local dev server.
- The local dev server makes all redirects, proxy rules, function routes or add-on routes, available locally.
- It also injects the correct environment variables from your site environment, installed add-ons or your netlify.toml file into your build and function environment.

Overview: Architecture



Hands-On #1:

- Create Astra Instance
- Create a token
- Deploy to Netlify

GitHub

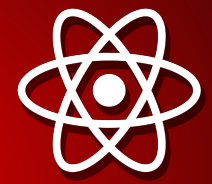
https://github.com/datastaxdevs/workshop-todo-native-mobile

Agenda

01 HouseKeeping

02 Overview

03 React to React Native
"Front-end"

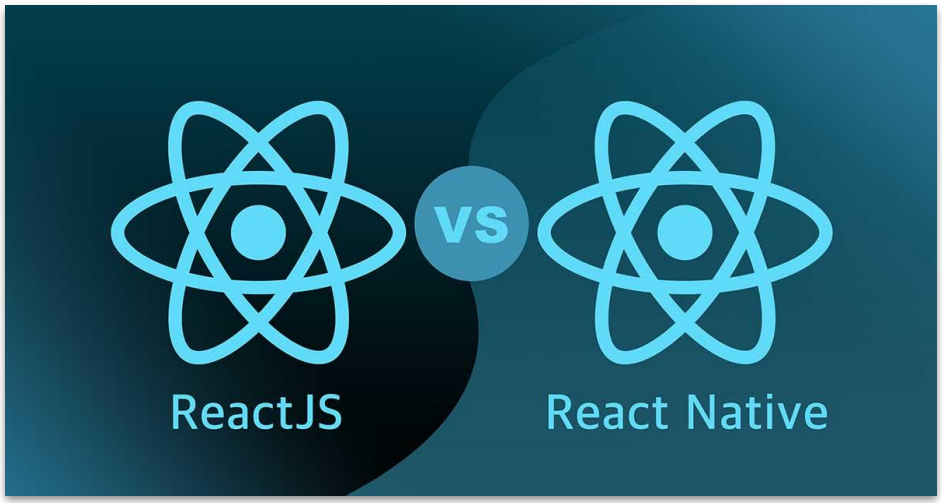


04 DataStax Astra
React to React Native
"Config & Back-end"

05 {REST}
Native Features

06 Resources

React vs. React Native: What's the Difference?

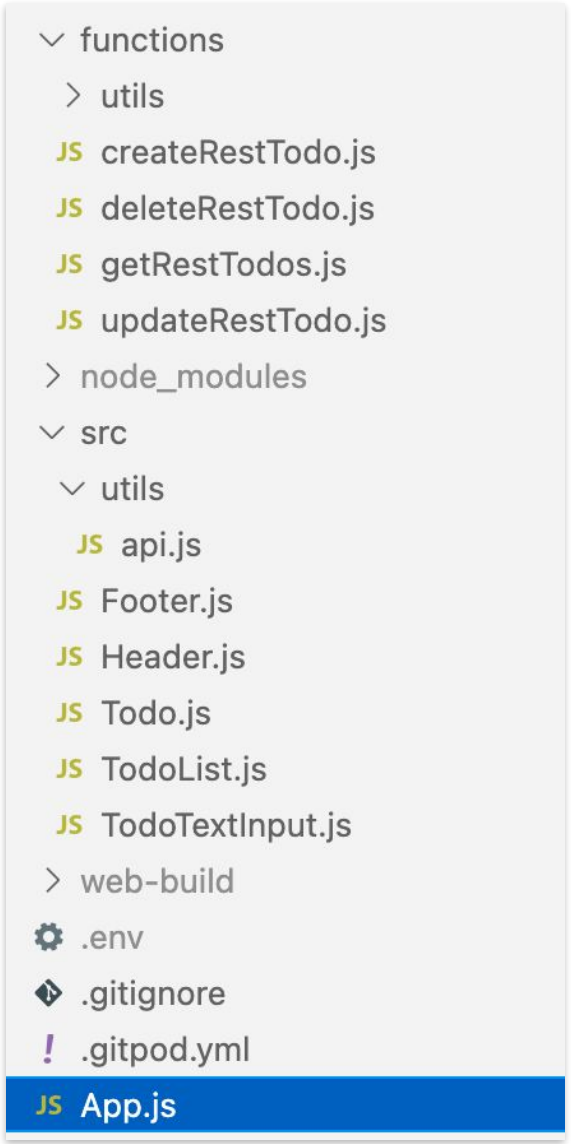


- Both open-sourced by Facebook, and are utilized in applications like Facebook, Instagram, Discord, AirBnB, Pinterest, UberEats, Skype, and SalesForce.

- **React Native** is a framework that allows you to build native, cross-platform mobile apps
- **React/React.js** is a JavaScript library you use for constructing a high performing UI layer.

- React uses HTML and CSS, whereas React Native uses its own components and libraries.

React to React Native: Importing Files



- Move App.js (the root component of your application) into the root folder of the Expo project

- Copy over the following:
  - Folders: src, functions, node modules,
  - Files: package.json, , netlify.toml, .env (if you want to use the same database as before)

- Run npm install to install all packages, and make sure node and nvm are at least versions 15 and 7

React to React Native: Key Code Differences

- HTML Tags vs. React Native Components

```
<div> vs <View>
<input> vs <TextInput>
<li> vs <FlatList>
```

- CSS vs. StyleSheets (see next slide)
- Import Statements

```
import { SafeAreaView, StyleSheet, View, TextInput, Button } from 'react-native';
```

- Layouts, Navigation, Animation and more!

Grid vs. FlexBox
React Router vs. Navigator
CSS Animations vs Animated Library

React to React Native: Props & State

```
const { todo, completeRestTodo, deleteRestTodo } = props;

const [isChecked, setIsChecked] = React.useState(todo.completed);

const [isVisible, setIsVisible] = React.useState(true);
```

- **Props is short for properties**, and they are used to pass data between React components (For example, between the App.js api calls and the TodoList component) React's data flow between components is unidirectional (from parent to child only). You can think of them are like parameters passed into a constructor.

- **State** allows components to create and manage their own data. So unlike props, components cannot pass data with state, but they can create and manage it internally. For example, we use state to manage the state of the checkbox.

React to React Native: CSS vs. StyleSheets

CSS in React:

```
<div className="complete"> </div>

complete: {
    text-decoration: line-through;
    font-size: 18;
    font-family: Inter_300Light;
}
```

StyleSheet in ReactNative:

```
<View style={styles.complete}> </View>

const styles = StyleSheet.create({
    complete: {
        textDecorationLine: 'line-through',
        fontSize: 18,
        fontFamily: 'Inter_300Light',
        flexWrap: 'wrap'
    }

});
```

- React Native uses camel case instead of hyphens, and requires commas (since it uses JSON) after lines instead of semicolons.
- Values need to be in ‘quotes’ in React native, except for numeric values
- Certain attributes are the same, others are not
  (ie. CSS box-shadow: inset 0 -1px 5px 0 rgba(0, 0, 0, 0.2))
- No className in React Native, but rather the style attribute
- Frequently use {curly braces} instead of quotes

React to React Native: Import Statements

```
import React from "react";
import { StyleSheet, View, FlatList } from 'react-native';
import Todo from "./Todo.js";
import Footer from "./Footer.js";
import SegmentedControl from '@react-native-segmented-control/segmented-control';
```

- Retain the React import statement, as well as the custom components in your project, but you also need to import each component from React Native explicitly
- If you install additional libraries, these need to be imported separately as well. As you can see in the above example, using single or double quotes is allowed.

Hands-On #2:

- Launch GitPod IDE
- Install libraries

GitHub

https://github.com/datastaxdevs/workshop-todo-native-mobile

DataStax Developers

Agenda

01 HouseKeeping

02 Overview

03 React to React Native

"Front-end"

04 DataStax Astra

React to React Native

"Config & Back-end"

05 Native Features

06 Resources

React to React Native: netlify.toml & .env

netlify.toml:

Before:

```
[build]
command = "npm run build"
functions = "functions"
publish = "build"
```

After:

```
[build]
command = "expo build:web"
functions = "functions"
publish = "web-build"
targetPort = 8888
```

- The netlify.toml file is a configuration file that specifies how Netlify builds and deploys your site — including redirects, branch and context-specific settings, and more.
- Expo has a different build command for each platform, but since Netlify is just building web, we use that build command.
- Expo also published the build to a different folder name, which can cause hiccups when working with Netlify if this step is skipped over

```
HOST="192.168.86.95" // Add your local IP here or GitPod url
PORT="8888"
IS_PROD="false"
GITPOD="false" // Change to true if on GitPod
```

React to React Native: GitPod.yml

```
tasks:
  - name: todonativemobileapp
    before: |
      cd /workspace/todonativemobileapp
      nvm install node
      npm install
      npm install -g expo-cli (Command line interface for Expo)
      npm install -g netlify-cli (Command line interface for Netlify)
      npm install astra-setup (To create the .env file during the workshop)
      npm install whatwg-fetch
      npm install -g @expo/ngrok (For tunnels on GitPod)
      npm install @expo/ngrok@4.1.0
      npm install react-native-gesture-handler (For swipe to delete/complete gesture)
      npm install @react-native-segmented-control/segmented-control  (For filter based on completeness)
      npm install @expo-google-fonts/inter --legacy-peer-deps (For custom fonts)
      npm install babel-plugin-inline-dotenv --legacy-peer-deps (For using inline environment variables)
```

- Sets up cloud-based IDE, GitPod for workshop deployment
- Installs necessary, packages, libraries, and CLIs so the app can work with Netlify, Expo, and React Native

React to React Native: Configuring Endpoints

Before:

```
const response = await fetch(`/.netlify/functions/getRestTodos`);
```

After:

```
// GENERATE
const generateEndpoint = () => {
  const ipAddress = process.env.HOST;
  const port = process.env.PORT;

  // Netlify deploy
  if (process.env.IS_PROD === "true") {
    return ``;
  }
  // Running on GitPod
  else if (process.env.GITPOD === "true") {
    return ipAddress;
  }
  // Local configuration
  else {
    return `http://${ipAddress}:${port}`;
  }
}

// Add to Create, Read, Update, and Delete API calls
const endpoint = generateEndpoint();
 ...
const response = await fetch(`${endpoint}/.netlify/functions/...RestTodo`...
```

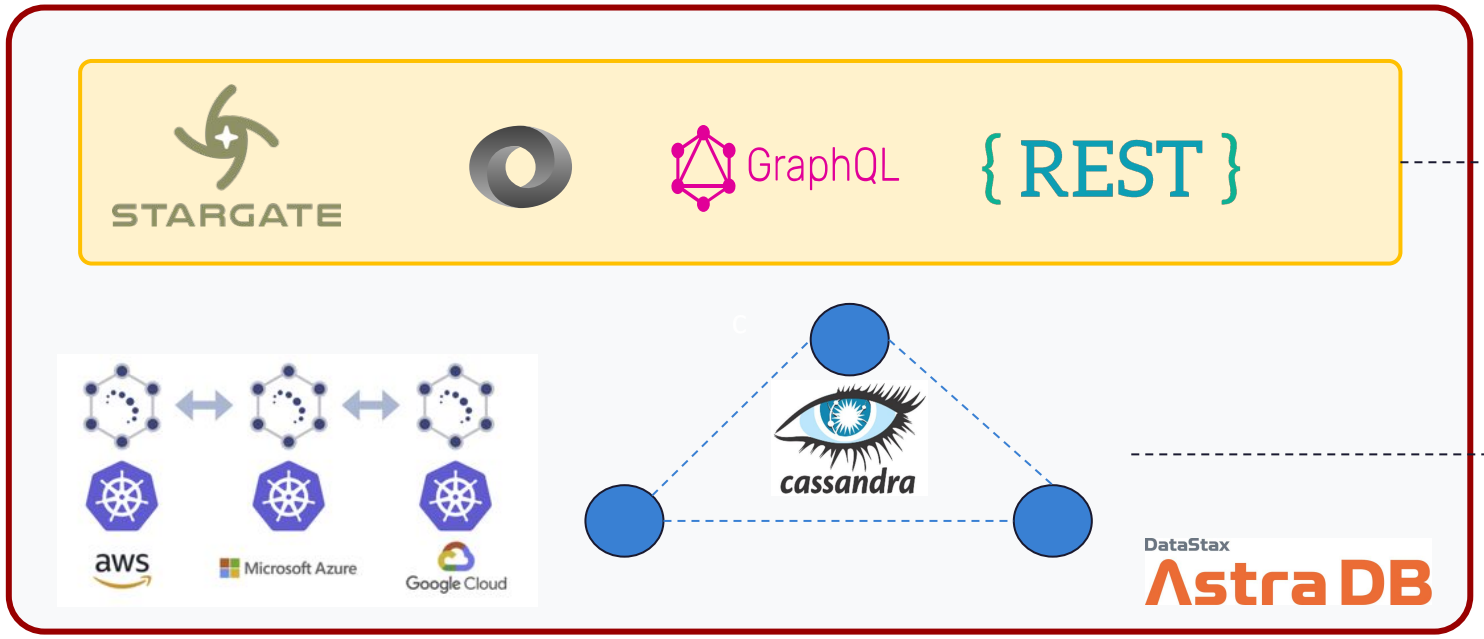
- Add additional configuration for three development environments - locally, on Netlify, and in GitPod
- Add configuration to be able to access environment variables directly inline in api.js file

babel.config.js

babel.config.js > ...

```
1  module.exports = function(api) {
2    api.cache(true);
3    return {
4      presets: ['babel-preset-expo'],
5      plugins: [
6        "inline-dotenv"
7      ]
8    };
9  };
```

React to React Native: Astra Provides APIs



OSS Stargate.io
A data gateway to allow multiple usages

STARGATE

OSS Apache Cassandra
A Column oriented NoSQL Database

cassandra

React to React Native: API

Application Programming Interface

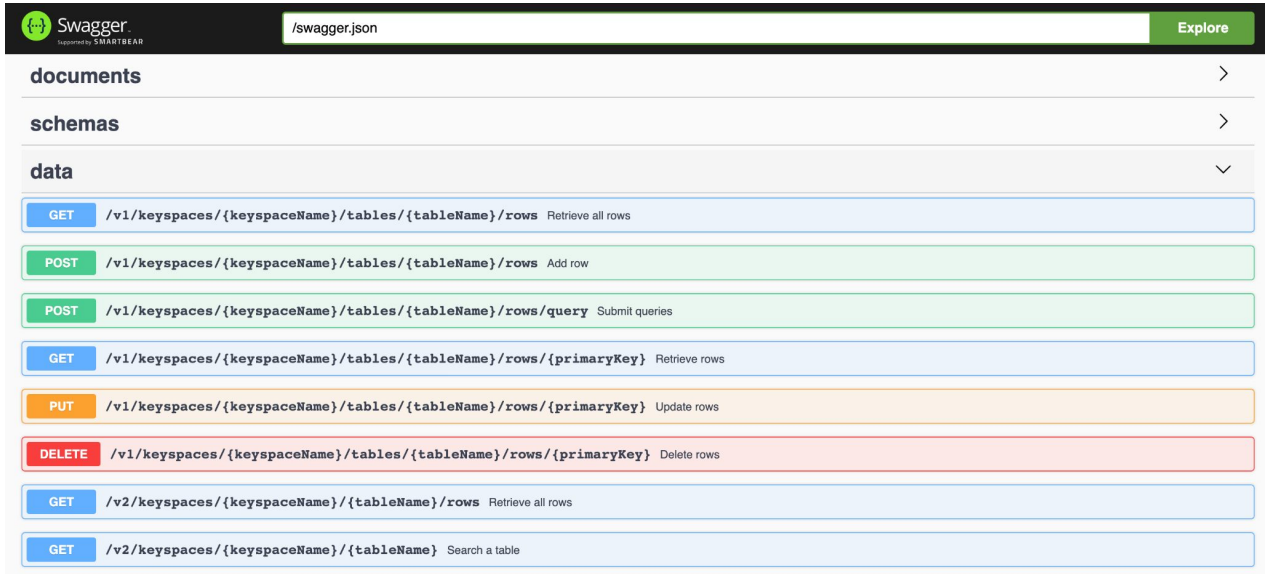
- API lists operations that developers can use, along with a description of what they do.
- Make Life Easier for Developers
- APIs Are Used For Communication Between Services

REST API

- Use HTTP format, methods and verbs (GET,POST,PUT,DELETE) to expose functionality

- Multiple specifications
  - Open API (Swagger)
  - GraphQL

No Change Needed!!!



Agenda

01 HouseKeeping

02 Overview

03 React to React Native

"Front-end"

04 DataStax Astra

React to React Native

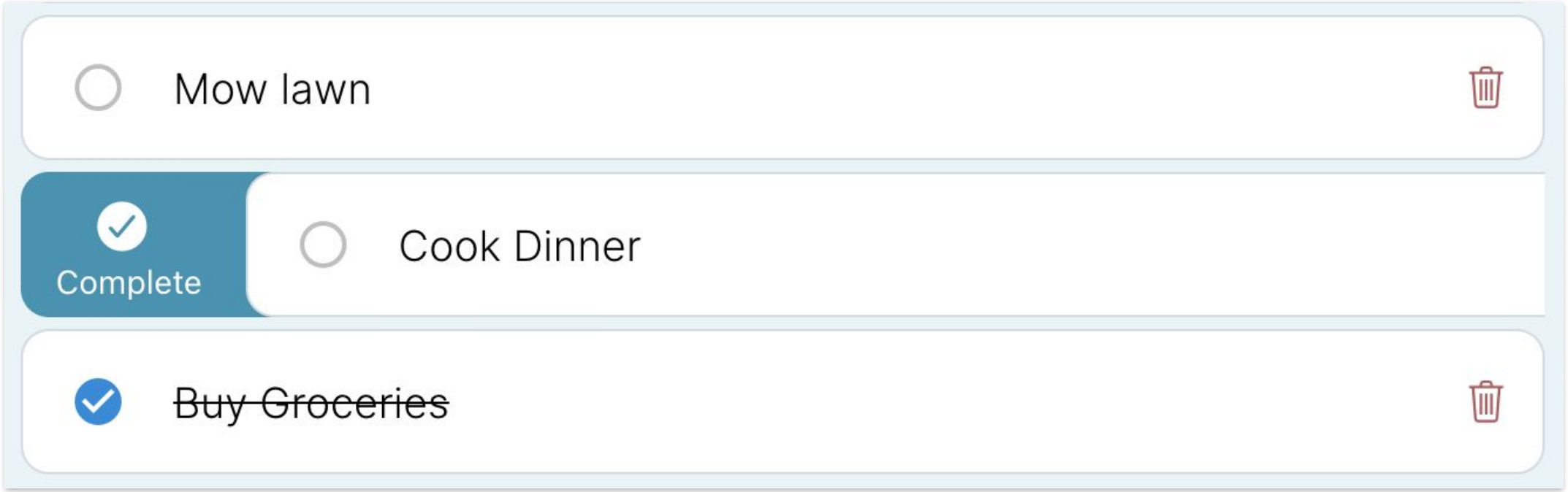
"Config & Back-end"

05 {REST}

Native Features

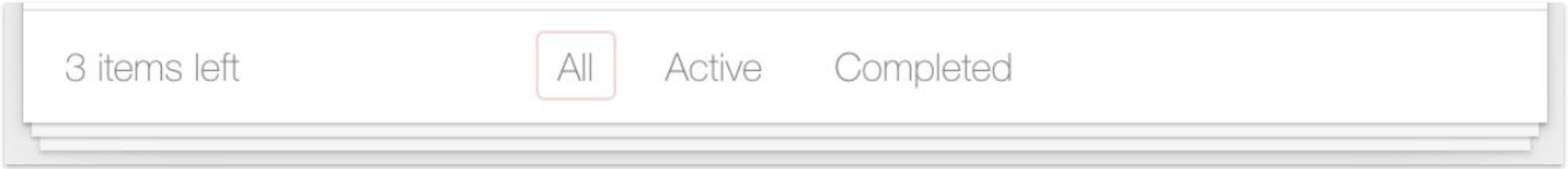
06 Resources

Native Features: Swipeable Gestures



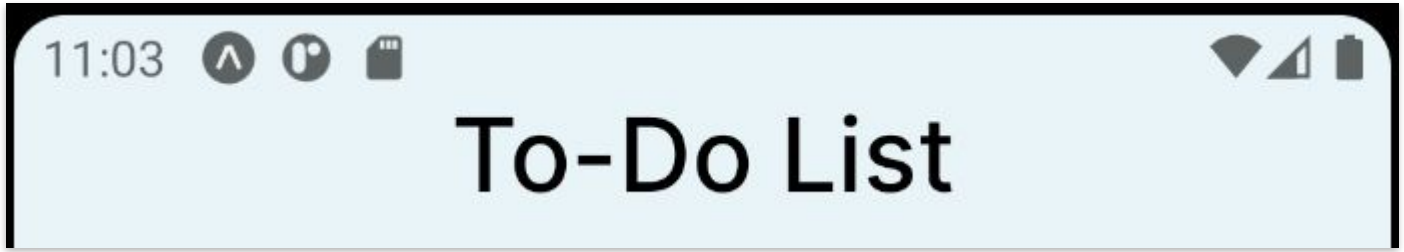
- Swipe left to delete, and swipe right to mark the todo complete or incomplete
- Use refs to make the todo item close on its own once the gesture is finished or released prematurely

Native Features: Segmented Control



- **Before:** filter in the footer appended to the bottom of the list
- **After:** Reconfigured footer filters to be added near the top in case many todos are added (Imported library because it is not available in the base set of components

Native Features: UI Libraries



- **flexWrap**: Property needed to prevent horizontal and vertical overflow from a long Todo item -- flexWrap: 'wrap'.
- **Removing border when TextInput is selected on the web**: Perfect example of a platform specific bug -- on web when the input box is selected, it is highlighted in blue, so you can import Platform to specify platform-related properties.
- **Custom Fonts:** Add custom fonts from Google Fonts to allow for the same font across platforms
- **StatusBar** - possible for Android but not iOS. You can change the color behind the StatusBar on Android, but not on iOS (shown above)

Tips & Tricks: More to Explore

- **Platform-Specific Bugs:** Sometimes native behavior is different between platforms so keep all emulators open when building

- **Peer Dependency Errors**: In case you are getting faulty peer dependency errors, first look at your package.json to see if you can resolve these manually, otherwise try re-running the npm command with the legacy peer dependency flag. These appear to occur because NPM 7 is more picky about peer dependencies than NPM 6. The legacy peer dependencies flag reverts to NPM 6 standards for peer dependencies. (Ex: npm install @expo-google-fonts/inter --legacy-peer-deps)

- **Finding Additional Features**: Expo and React Native may not include all the components and extended functionality you may need. As a result, it is frequently necessary to look for libraries that contain that feature in the React Native Directory. For example, there isn't a in-built React Native Checkbox component that works on all platforms, so this directory is helpful to lookup all the options available and assess their community support, as well when they were last updated.

- **Search for Solutions**: Google, StackOverflow, blogs, and forums are the best teachers; if you're stumped on an issue, it's very probable that another developer has faced the same issue. Search for a solution using keywords and code snippets, and you should be able to find many workarounds for the problem. If all else fails, find the forum for the library you're using and post a question there.

- **Create a PWA, Progressive Web App:** Expo is automatically set up to build a PWS should you want to have one. You can follow the steps here to make one in under five minutes!

Hands-On #3:

- Register Expo Go
- Launch Todo native app

GitHub

https://github.com/datastaxdevs/workshop-todo-native-mobile

DataStax Developers

Agenda

01 HouseKeeping

02 Overview

03 React to React Native

"Front-end"

04 DataStax Astra

React to React Native

"Config & Back-end"

05 {REST}

Native Features

06 Resources

menti.com Quiz Time !

Mentimeter

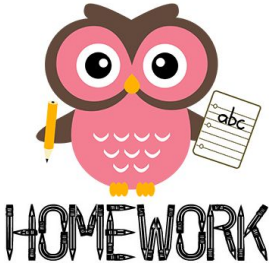
Swag Winners 🎉

Congratulations to 1st, 2nd, and 3rd place on the menti quiz!

To claim your prize, please send an email to:

jack.fryer@datastax.com

**** Include a screenshot of your Menti screen**



Get your badge!
(and brag on Linkedin)

#1 - [Complete hands-on]

Run the todo Application in Gitpod connected to AstraDB and launch the app on your phone with Expo Go THROUGH STEP 9

#2 - [(Optional) production deployment in Netlify]

Finish off the rest of the instructions to deploy the mobile native Todo app in Netlify and launch the app from your Netlify site using steps 10 and 11

All needed info in github repo

https://github.com/datastaxdevs/workshop-todo-native-mobile

Claim your FREE Certification Voucher 🎉

Vouchers (normally 145$ each)

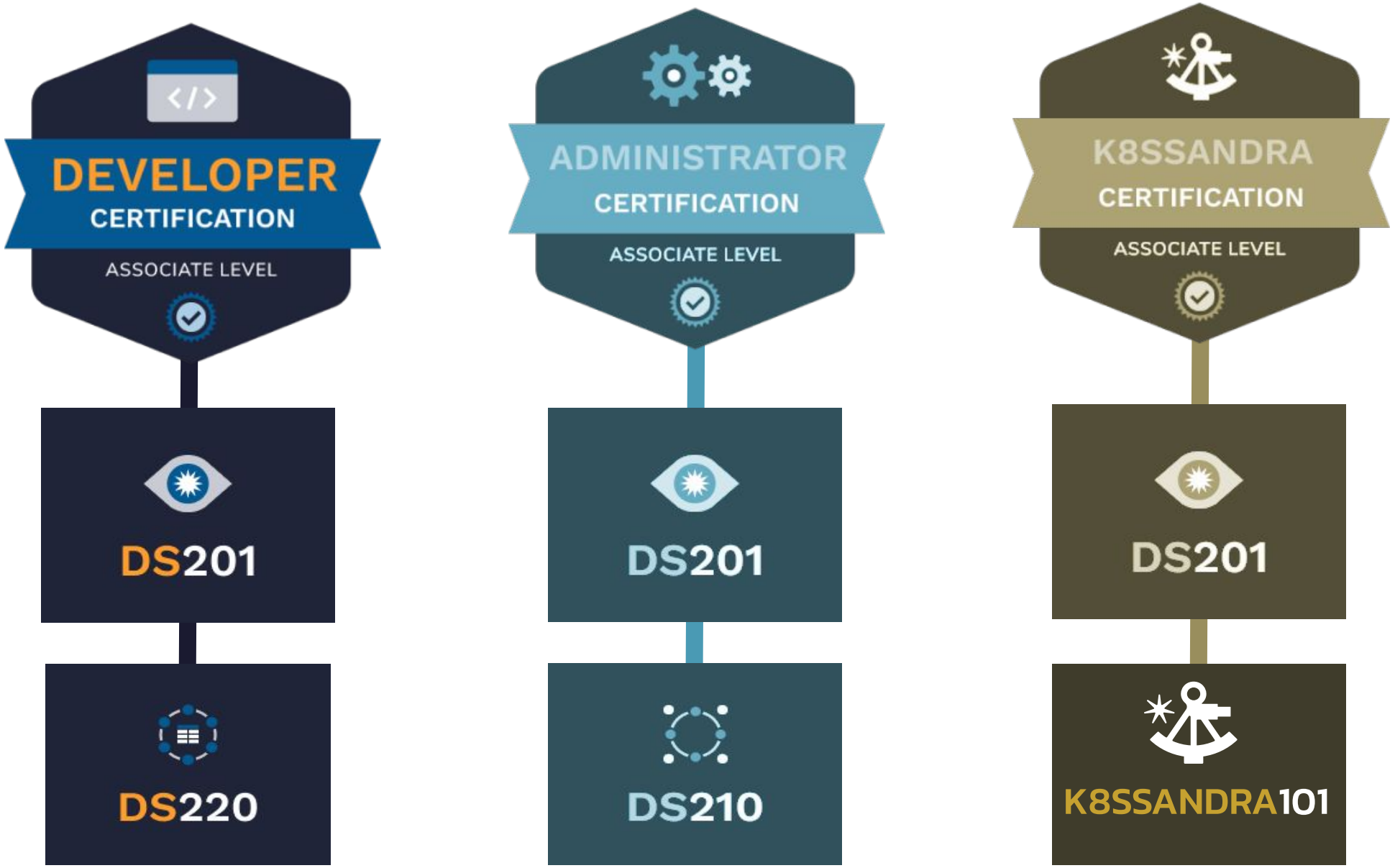
- valid for 3 months,
- valid for 2 attempts

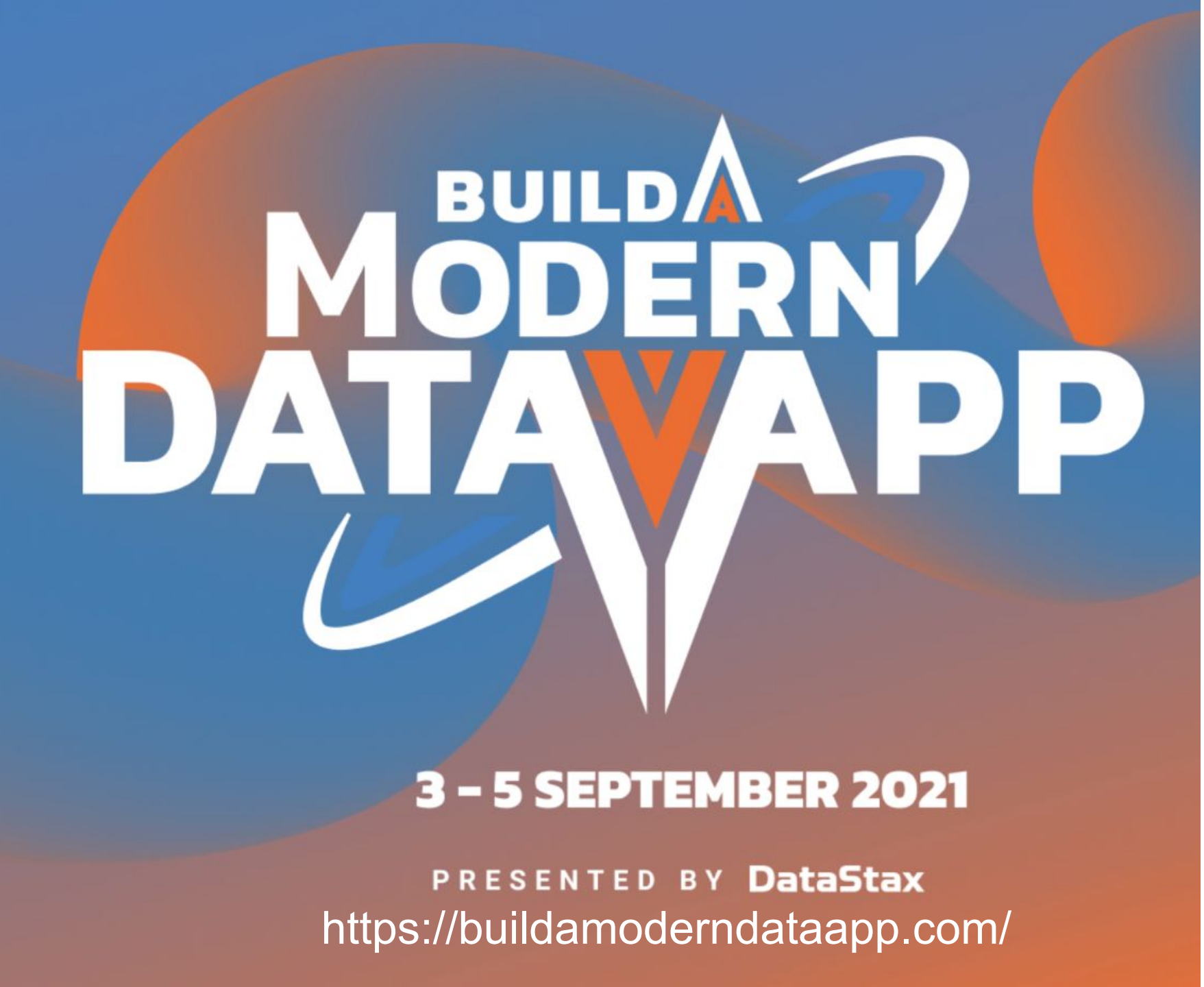
https://www.datastax.com/dev/certifications

Claim using the link:

http://dtsx.io/workshop-voucher

LEARNING PATHS at the academy.datastax.com

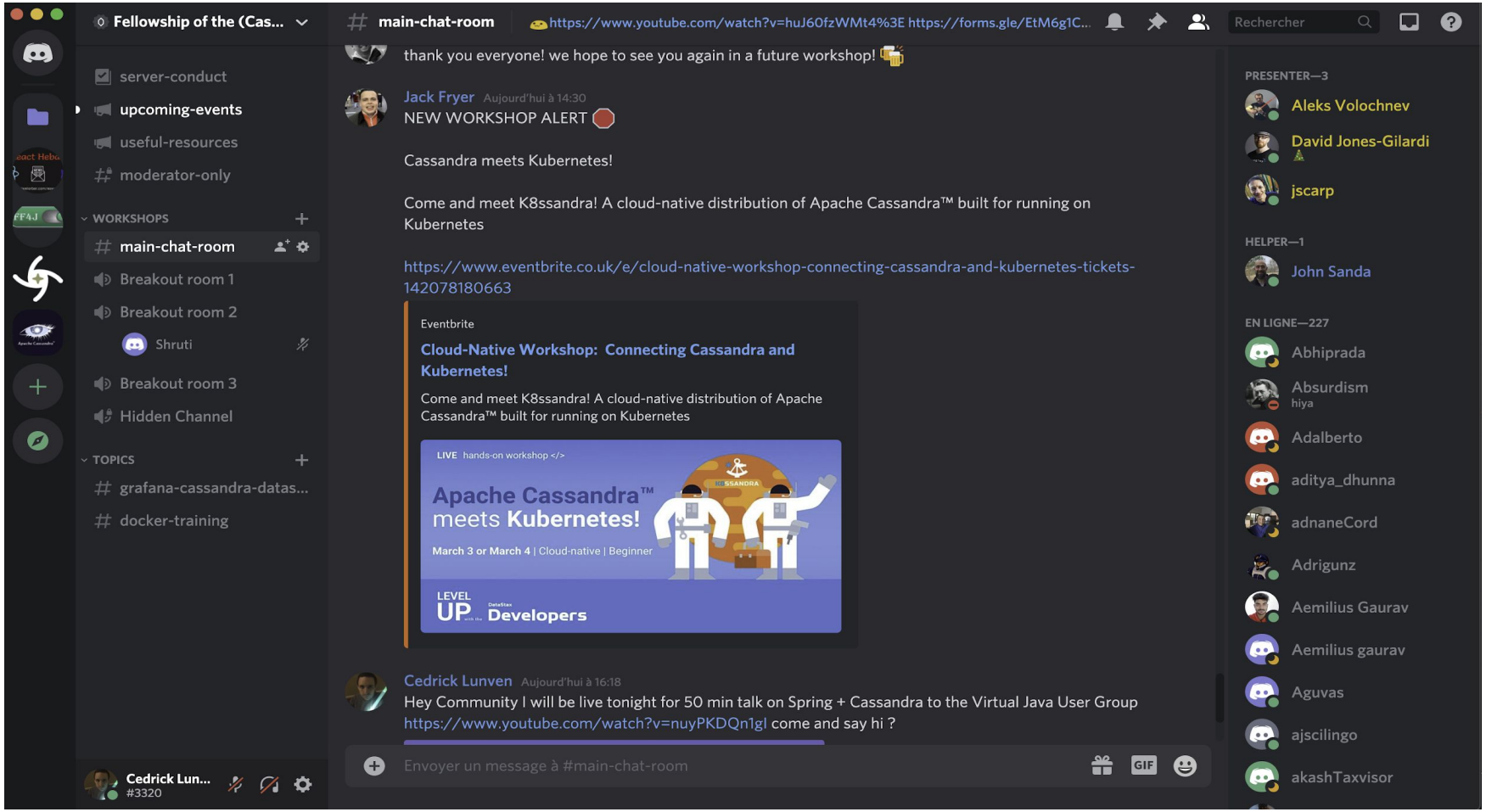




$26,000
in prizes

REGISTER:

Join our 10k Discord Community
The Fellowship of the RINGS
dtsx.io/discord



DataStax Developers

Subscribe

Thank you!

Thank you!

Subscribe