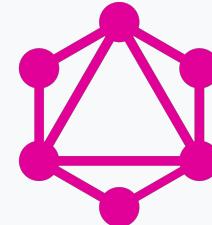


Developers

Introduction to GraphQL

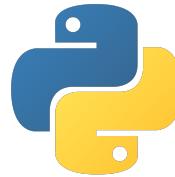
Query only what you need and when you need it ...



GraphQL



Developer Advocate



dtsx.io/stefano



@stefano-lottini



@hemidactylus



@rsprrs

- Developer/Architect
- Apache Cassandra™ certified
- Background in computational physics
- Distributed systems
- Love to teach and communicate



Stefano Lottini

Director of Developer Relations



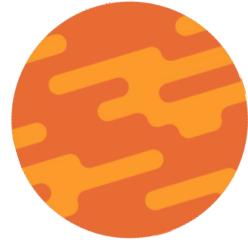
- Trainer
- Public Speaker
- Developers Support
- Developer Applications
- Developer Tooling

- Creator of ff4j (ff4j.org)
- Maintainer for 8 years+

- Happy developer for 14 years
- Spring Petclinic Reactive & Starters
- Implementing APIs for 8 years



Cédrick Lunven



S



Cedrick
Lunven



David
Dieruf



Rags
Srinivas



Artem
Chebotko



Stefano
Lottini



Aleksandr
Volochnev



Aaron
Ploetz



S



Jack
Fryer



Kirsten
Hunter



Gary
Harvey



Mary
Grygleski



Ryan
Welford



David
Gilardi



DataStax Developers Crew

01



Housekeeping

Live and Hands-on



02

Database Setup

Data model & Astra DB

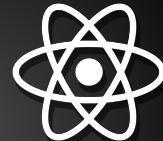


03

GraphQL

GraphQL

Design & Toolings



Backend

Expose GQL endpoint



04

05

Frontend

Consume GQL Endpoint



06

What's next?

Quiz, Homework, ...



Agenda





Housekeeping

Live and Hands-on



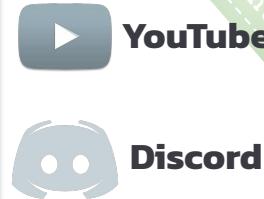
Livestream: youtube.com/DataStaxDevs

Questions: <https://dtsx.io/discord>

Agenda



A screenshot of a Discord channel titled "workshop-chat". The channel has over 100 members. A message from user "isole001" is highlighted, discussing the workshop's background and its relevance to learning DataStax Astra.



Games and quizzes: menti.com

A Menti poll titled "How much experience do you have with the Spring Framework ?" with the following results:

| Experience Level | Percentage |
|------------------------|------------|
| Never heard about it | 41% |
| I know the concepts | 29% |
| I have already used it | 10% |
| I use it regularly | 20% |

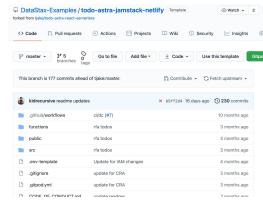
A small video player at the bottom left shows a person speaking.



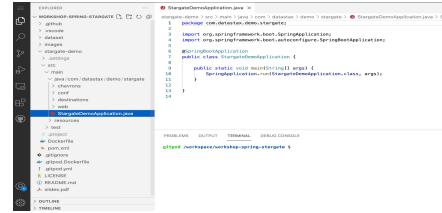
Live Sessions

Nothing to install !

Source code, exercises, slides



IDE

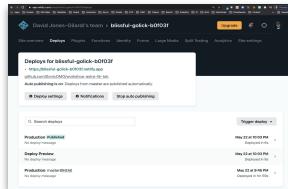


Database + GraphQL + PlayGround



DataStax
Astra DB

Deploy + Run + Host



Hands-On Housekeeping



Get your badge for today

- **BEGINNERS**

- Relax, no complex code to write
- Get the most of if by following what presenters do and ask Questions !
- Try to improve and develop the application later!

- **INTERMEDIATE**

- Scroll and do exercises on your own PACE
- Sample code, ready to go to copy-paste

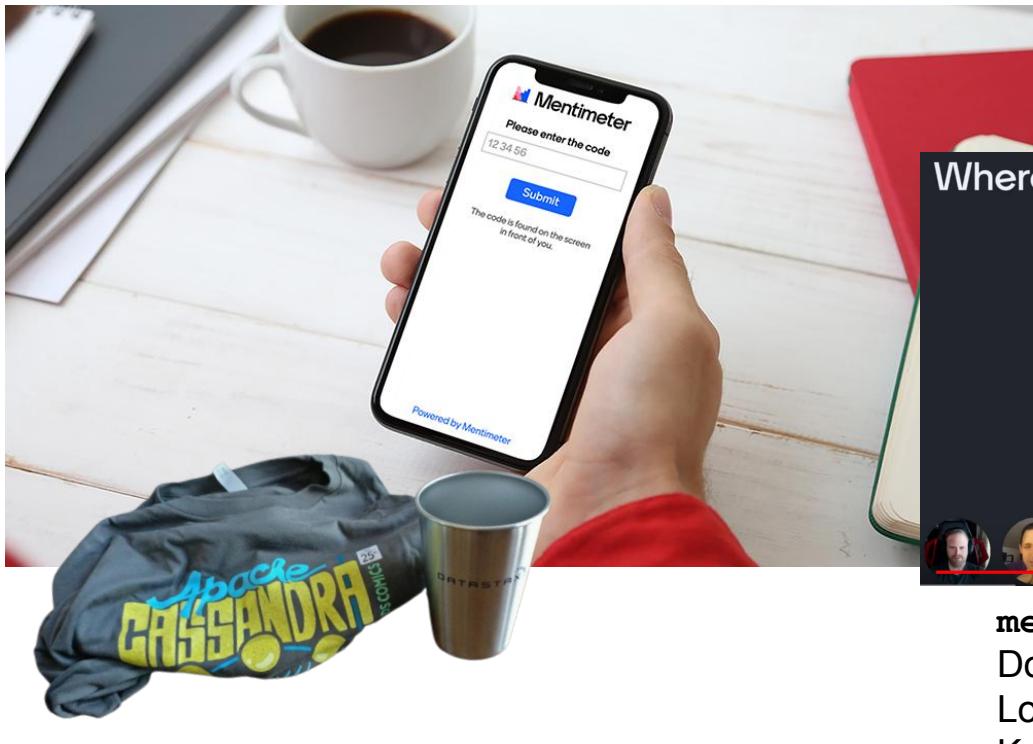
- **EXPERTS**

- Run stuff locally
- Challenge



Choose your path





Where are you from?



menti.com ⇒ enter code
Don't answer in YT chat
Look at phone (not at YT)
Keep it open for later

"Menti" for survey and quiz !

01



Housekeeping Live and Hands-on

02



Database Setup Data model & Astra DB

03



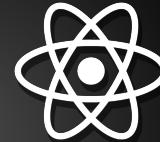
GraphQL Design & Toolings

04



Backend Expose GQL endpoint

05



Frontend Consume GQL Endpoint

06

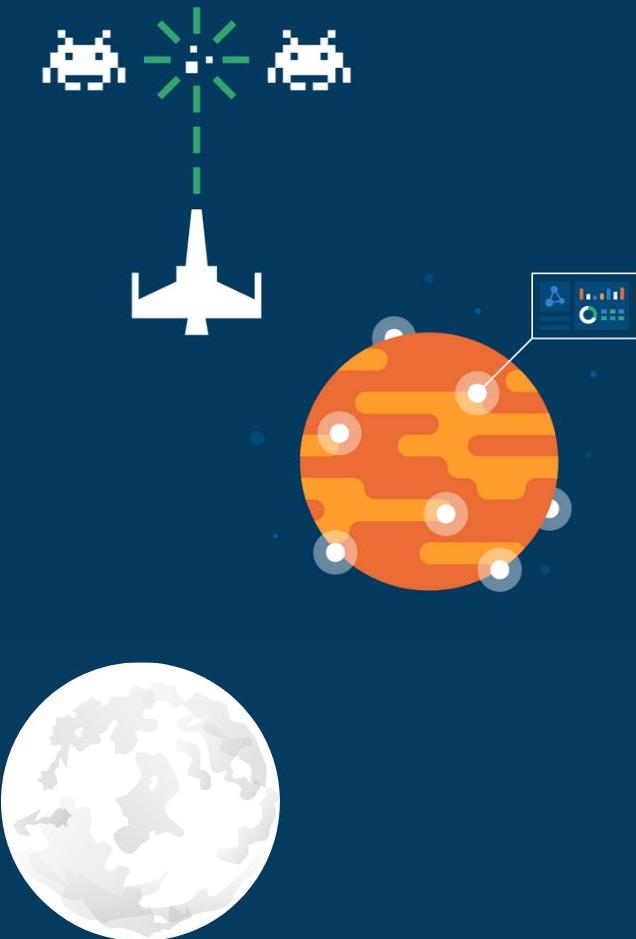


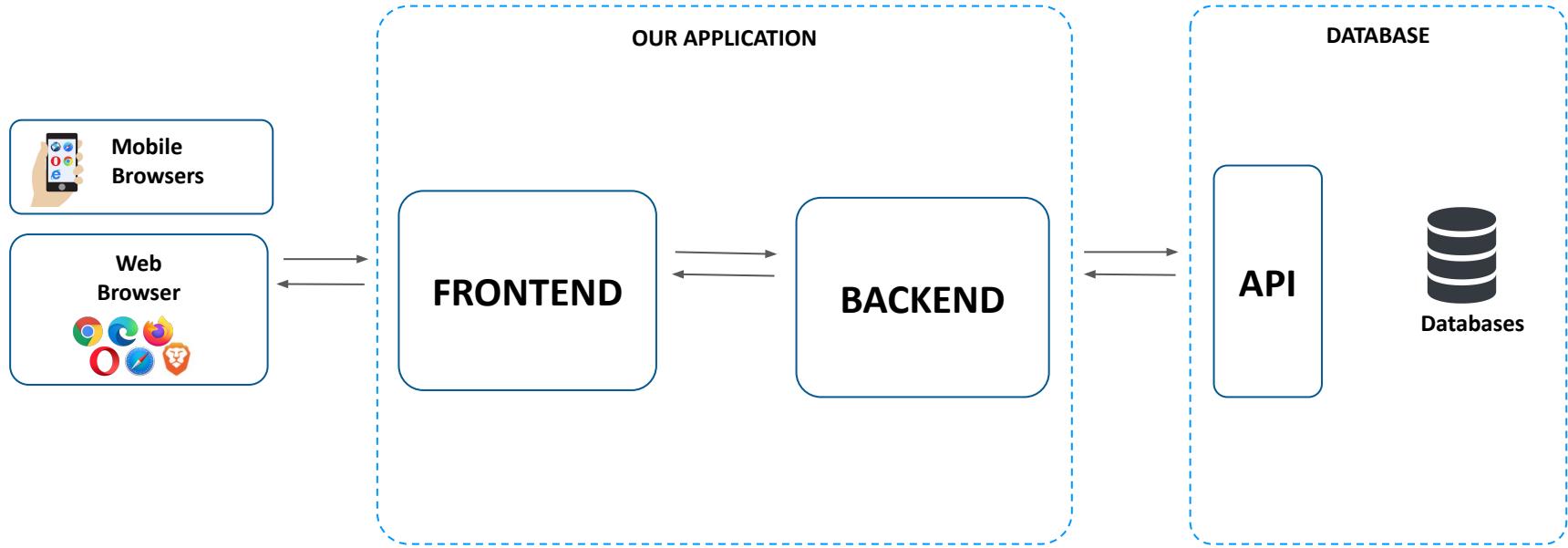
What's next? Quiz, Homework, ...



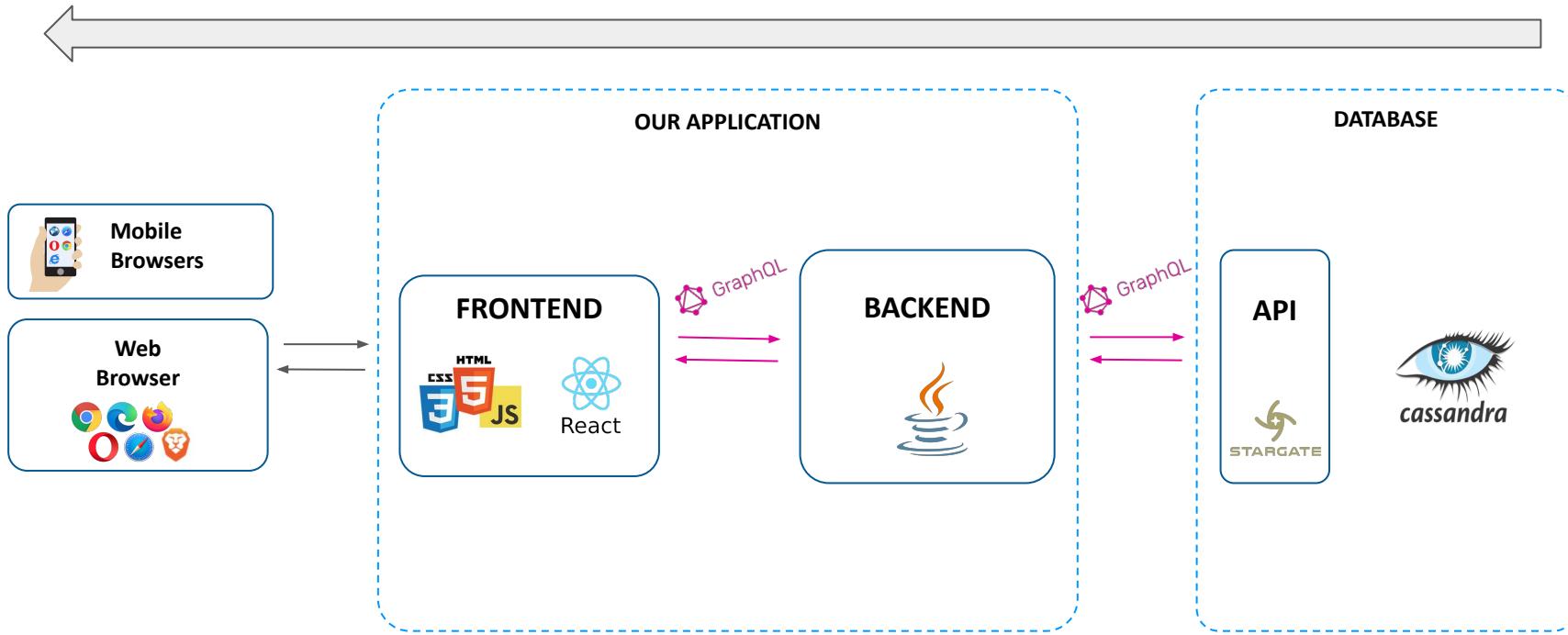
Agenda







Building a Full-stack application



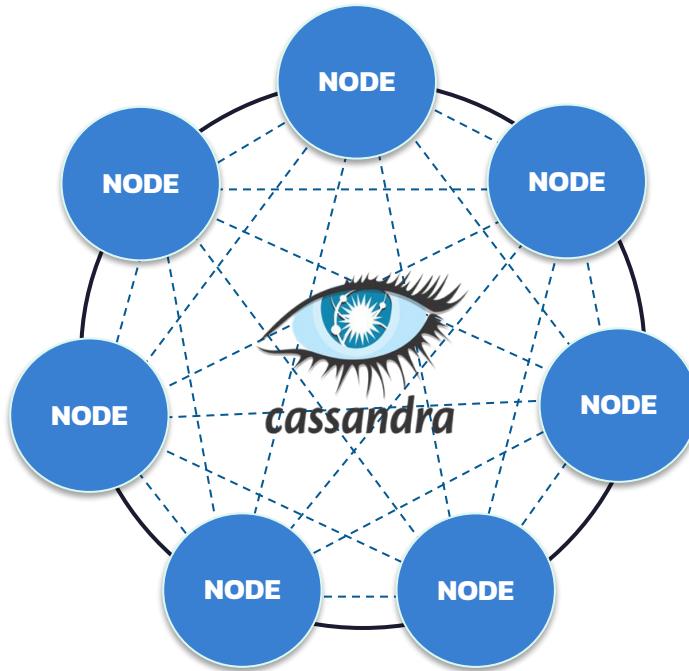
Building a Full-stack application

Database setup



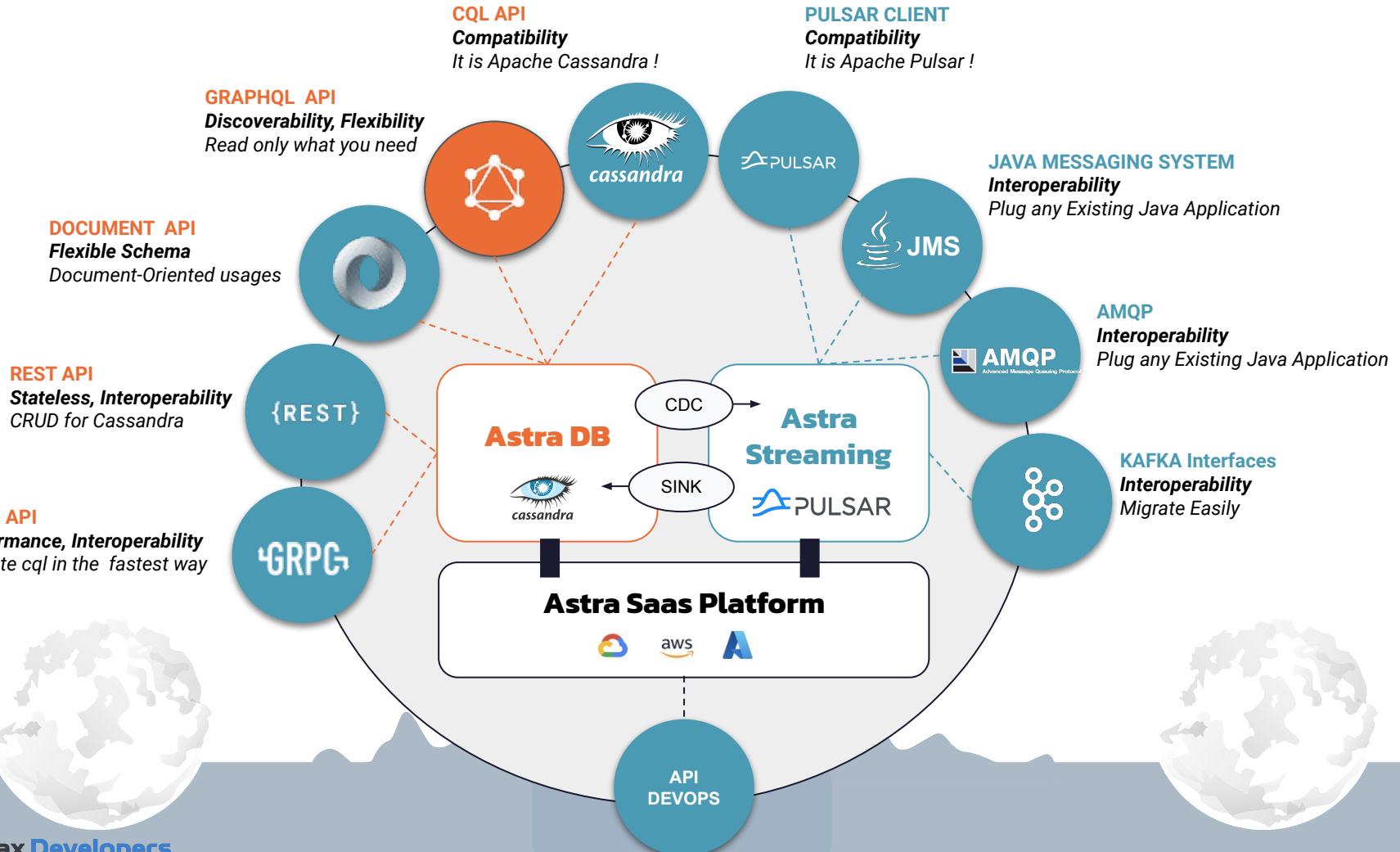
Data model & Astra DB

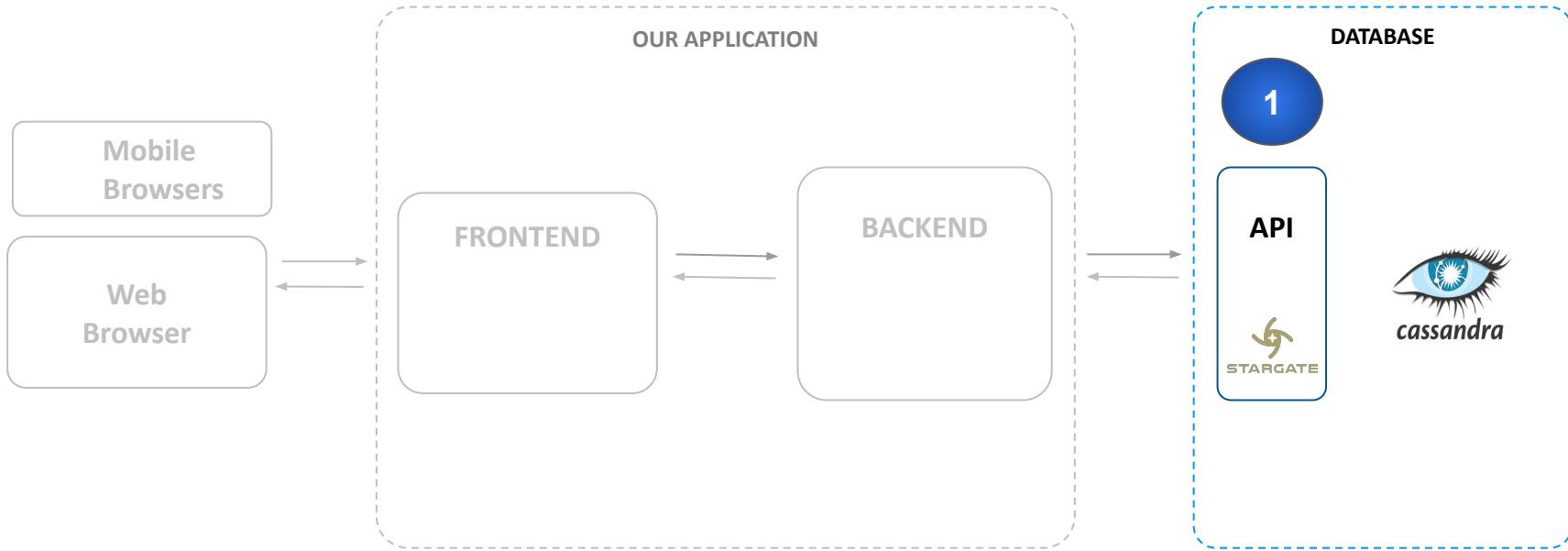




- Big Data Ready
- Read / Write Performance
- Linear Scalability
- Highest Availability
- Self-Healing and Automation
- Geographical Distribution
- Platform Agnostic
- Vendor Independent

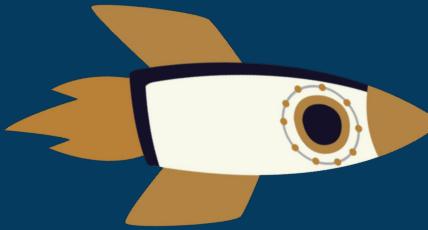
Introduction to Apache Cassandra™





Setup Database





Hands-on (!github)

#1 DB SETUP

- ✓ Register & Create your Database
- ✓ Create a security token
- ✓ Launch Gitpod



01



Housekeeping Live and Hands-on

02



Database Setup Data model & Astra DB

03



Backend Expose GQL endpoint

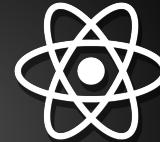
04



GraphQL

GraphQL Design & Toolings

05



Frontend Consume GQL Endpoint

06



What's next? Quiz, Homework, ...



Agenda



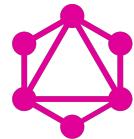


 GraphQL

GraphQL

Learn & Design

- Created by Facebook
Used internally for mobile apps
- 2012
- Facebook give talk at ReactJs Conf
- 2015
- Facebook open sourced GraphQL
- 2015
- Github announced move to GQL
- 2016



GraphQL

Definition

GraphQL is an application programming interface (API) query language and server-side runtime that prioritises giving customers precisely the data they request.



Meet GraphQL



It is a web protocol (HTTP) that specifies the way we build and query remote APIs using a Tree/JSON Syntax.

```
incomingWorkshops {  
    title  
    abstract  
    speakers  
}
```

Based on a **strongly typed** language named GraphQL SDL
(Schema Definition Language)

```
type Workshop {  
    title: String!  
    abstract: String  
    speakers: [Speaker]  
    releaseYear: int  
}
```



<https://graphql.org/learn/schema/#type-language>



What is graphQL ?

A protocol that allows the client to specify exactly what data it needs from a model

```
incomingWorkshops {  
    title  
}
```

It allows one to aggregate data from **multiple relations** in a single query

```
incomingWorkshops {  
    title: String!  
    abstract: String  
    speakers: {  
        name  
    }  
    releaseYear: int  
}
```



<https://graphql.org/learn/schema/#type-language>



What is graphQL ?

- **Int:** An integer type (example: 10)
 - **Float:** Floating point number, example 3.43
 - **String:** A sequence of characters ("Hello World")
 - **Boolean:** A boolean example true
 - **ID:** An object identifier
-
- **User Defined types**
 - Create your structure with constraints

```
type Workshop {  
    title: String!  
    abstract: String  
    speakers: [Speaker]  
    releaseYear: int  
}  
  
type Speaker {  
    name: String!  
    email: String  
}
```

```
{  
  hero {  
    name  
    friends {  
      name  
      homeWorld {  
        name  
        climate  
      }  
      species {  
        name  
        lifespan  
        origin {  
          name  
        }  
      }  
    }  
  }  
}
```

```
type Query {  
  hero: Character  
}  
  
type Character {  
  name: String  
  friends: [Character]  
  homeWorld: Planet  
  species: Species  
}  
  
type Planet {  
  name: String  
  climate: String  
}  
  
type Species {  
  name: String  
  lifespan: Int  
  origin: Planet  
}
```

Describe what's possible with a type system

GraphQL APIs are organized in terms of types and fields, not endpoints. Access the full capabilities of your data from a single endpoint. GraphQL uses types to ensure Apps only ask for what's possible and provide clear and helpful errors. Apps can use types to avoid writing manual parsing code.

<https://graphql.org>

Schema Definition

- Used for READ ONLY requests.
- Used to fetch data from the server using the graphQL SDL syntax
- Describe what data the requester wishes to fetch from whoever is fulfilling the GraphQL Query



```
query listWorkshops {  
  workshops(sortedBy: "date") {  
    name,  
    abstract,  
    speakers {  
      name  
    }  
  }  
}
```

What is a GraphQL Query ?

```
{  
  hero {  
    name  
    height  
    mass  
  }  
}  
  
{  
  "hero": {  
    "name": "Luke Skywalker",  
    "height": 1.72,  
    "mass": 77  
  }  
}
```

Ask for what you need,
get exactly that

Send a GraphQL query to your API and get exactly what you need, nothing more and nothing less. GraphQL queries always return predictable results. Apps using GraphQL are fast and stable because they control the data they get, not the server.

<https://graphql.org>

Declarative Query

- Used to change resources data or execute actions on the server
- Client specifies the arguments and the action to be executed and then receives a response or a resource updated

```
mutation createWorkshop {  
  createWorkshops({  
    name: "Intro to GraphQL"  
    abstract: "TODO"  
    speakers: [  
      {name: "Gilardi"},  
      {name: "Hunter"}  
    ]  
  }  
  {  
    name  
  }  
}
```



- ❖ Discoverability, introspection
- ❖ Declarative Data Fetching
- ❖ Schema stitching
- ❖ Customize responses
- ❖ Flexible versioning
- ❖ Match standards (Json | Http)



- ❖ Single endpoint (*versioning, monitoring, security*)
- ❖ Complex implementation (*tooling, still young*)
- ❖ Nice for customers, nasty for DB (*N+1 select*)
- ❖ Rate limiting is required
- ❖ Caching is hard (Apollo is a good solution here
<https://www.apollographql.com>) <- this is what Netflix uses



- ❖ BFF : Backend for frontend
- ❖ Service aggregation | composition, "federation" (*joins*)
- ❖ When bandwidth matters (*mobile phones*)



GraphQL S.W.O.T. analysis



 **GraphQL**
GraphQL
Tooling

GraphiQL

ENDPOINT URL

EXECUTE

QUERY

RESULTS

The diagram illustrates the workflow for executing a GraphQL query. It starts with the **ENDPOINT URL**, which points to the **GraphiQL** interface. In the **GraphiQL** interface, the **EXECUTE** button is used to run the **QUERY**. The results of the query are displayed in the **RESULTS** pane. A circled section of the results is labeled **Schema**, indicating the type information for the returned data.

graph LR; A[ENDPOINT URL] --> B[GraphiQL]; B -- EXECUTE --> C[RESULTS]; D[Schema] --- E[RESULTS];

```
# Welcome to GraphQL
#
# GraphQL is an in-browser IDE for
# testing GraphQL queries.
#
# Type queries into this side of the
# see intelligent typeheads aware of
# live syntax and validation errors
#
# To bring up the auto-complete at any
# Press the run button above, or Cmd-Shift-Ctrl-Enter
# will appear in the pane to the right
{
  data: {
    allFilms: {
      films: [
        {
          title: "A New Hope",
          episodeID: 4
        },
        {
          title: "The Empire Strikes Back",
          episodeID: 5
        },
        {
          title: "Return of the Jedi",
          episodeID: 6
        },
        {
          title: "The Phantom Menace",
          episodeID: 1
        },
        {
          title: "Attack of the Clones",
          episodeID: 2
        },
        {
          title: "Revenge of the Sith"
          episodeID: 3
        }
      ]
    }
  }
}

# A single film.
# IMPLEMENTS Node
# FIELDS
# title: String
# episodeID: Int
# openingCrawl: String
# director: String
# producers: [String]
# releaseDate: String
# speciesConnection(after: String, first: Int, before: String, last: Int): FilmSpeciesConnection
# starshipConnection(after: String, first: Int, before: String, last: Int): FilmStarshipsConnection
# vehicleConnection(after: String, first: Int, before: String, last: Int): FilmVehiclesConnection
# characterConnection(after: String, first: Int, before: String, last: Int): FilmCharactersConnection
# planetConnection(after: String, first: Int, before: String, last: Int): FilmPlanetsConnection
# created: String
# edited: String
```

GraphQL Playground

DDL: DATA DEFINITION

Create Table

Create Schema

DML: DATA MANAGEMENT

INSERT DATA

ENDPOINT URL

EXECUTE

HEADER

The screenshot shows the GraphQL Playground interface. At the top, there are tabs for "graphql-schema" and "graphql". Below the tabs, there are buttons for "PRETTYIFY" and "HISTORY". The URL is displayed as <https://105.77.121.853a-4dd7-adb3-4245-1a81191-1.eu-central-1.apps.astra.datastax.com/api/graphql-schema>. The main area contains a code editor with the following query:

```
1 { keyspaces { name } }
```

Below the code editor is a "QUERY VARIABLES" section with a single variable:

```
1 iuZspoYB:98936cfa48c7eacc14cf1b0fb4169b394a15b52cb7e2463ece31995bdf10feb4
```

To the right of the code editor is a results viewer showing a JSON response:

```
{ "data": { "keyspaces": [ { "name": "system_traces" } ] } }
```

At the bottom left is an "HTTP HEADERS" section with one header entry:

```
1 iuZspoYB:98936cfa48c7eacc14cf1b0fb4169b394a15b52cb7e2463ece31995bdf10feb4
```

On the far right, there is a "Schema" browser with sections for "QUERIES" and "MUTATIONS". The "QUERIES" section lists:

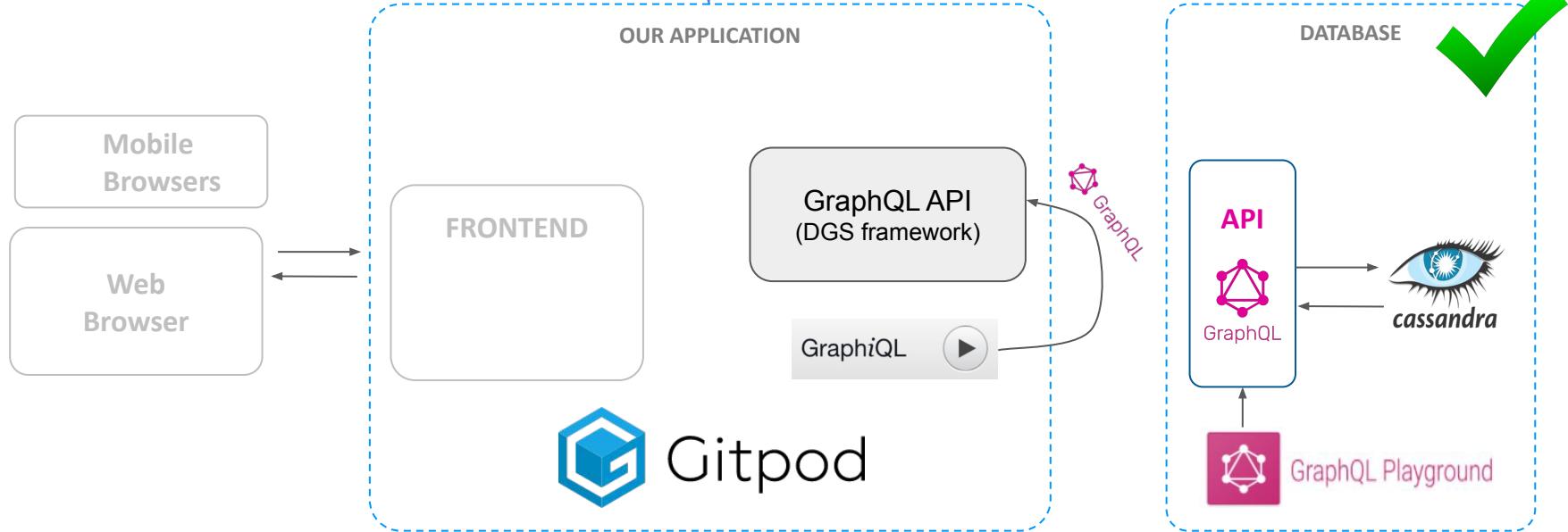
- keyspace(...): Keyspace
- keyspaces: [Keyspace]

The "MUTATIONS" section lists:

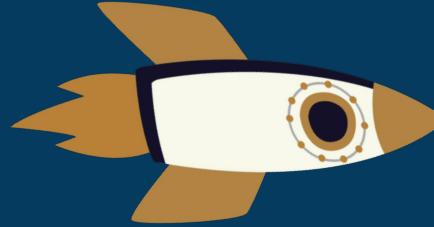
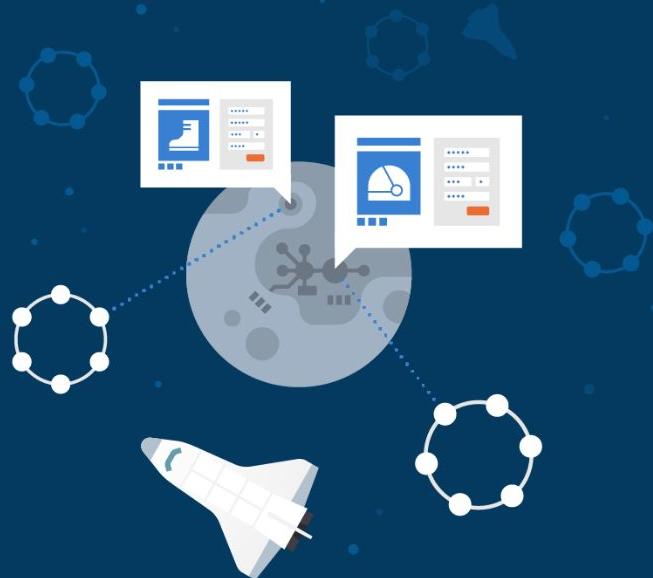
- createTable(...): Boolean
- alterTableAdd(...): Boolean
- alterTableDrop(...): Boolean
- dropTable(...): Boolean
- createType(...): Boolean
- dropType(...): Boolean
- createIndex(...): Boolean
- dropIndex(...): Boolean
- createKeyspace(...): Boolean
- dropKeyspace(...): Boolean

Schema

evelopers



What's coming in next Hands-on ?



Hands-on (!github)

#2 First Touch

- ✓ Experiment with Graph*QL*
- ✓ Demo with GraphQL Playground



01



Housekeeping Live and Hands-on

02



Database Setup Data model & Astra DB

03



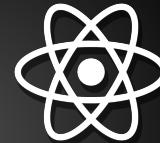
GraphQL Design & Toolings

04



Backend Expose GQL endpoint

05



Frontend Consume GQL Endpoint

06

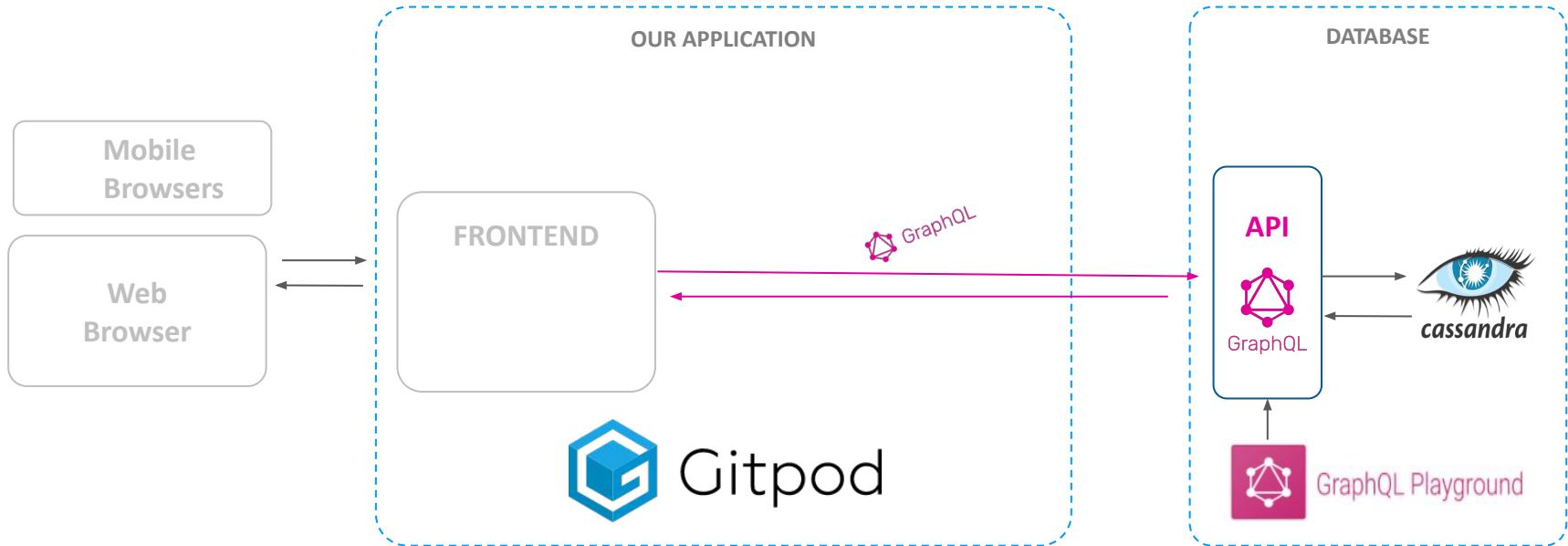


What's next? Quiz, Homework, ...

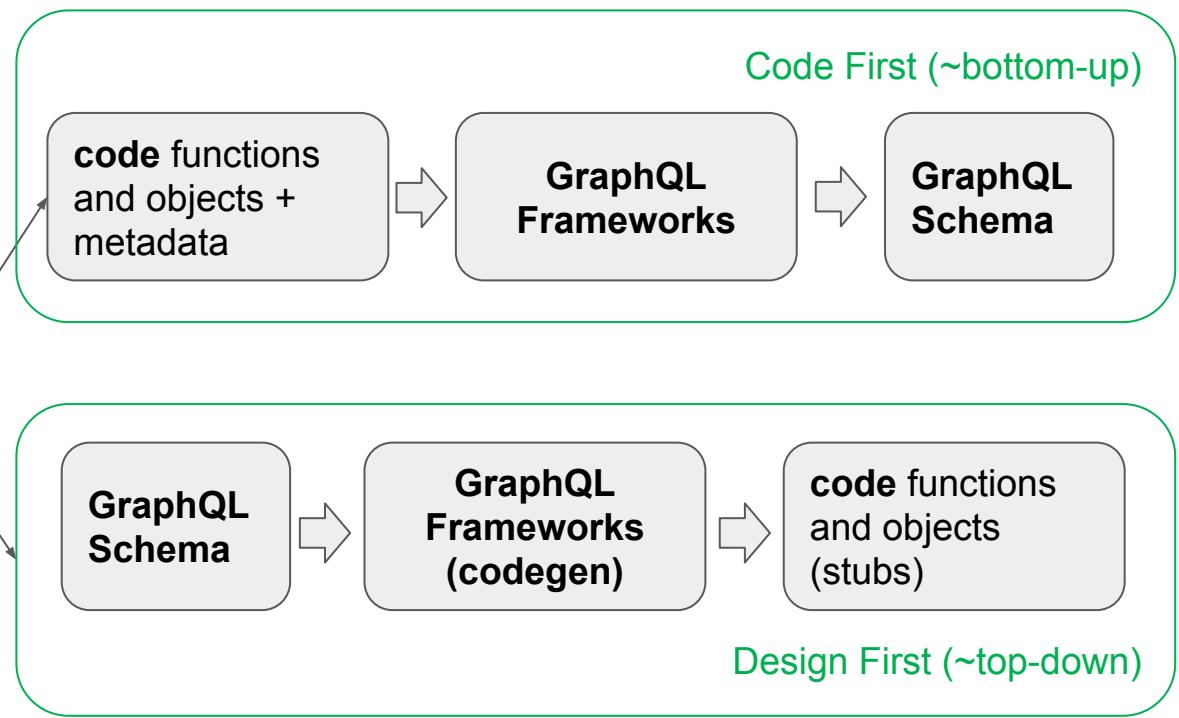
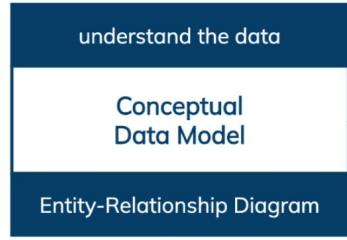


Agenda

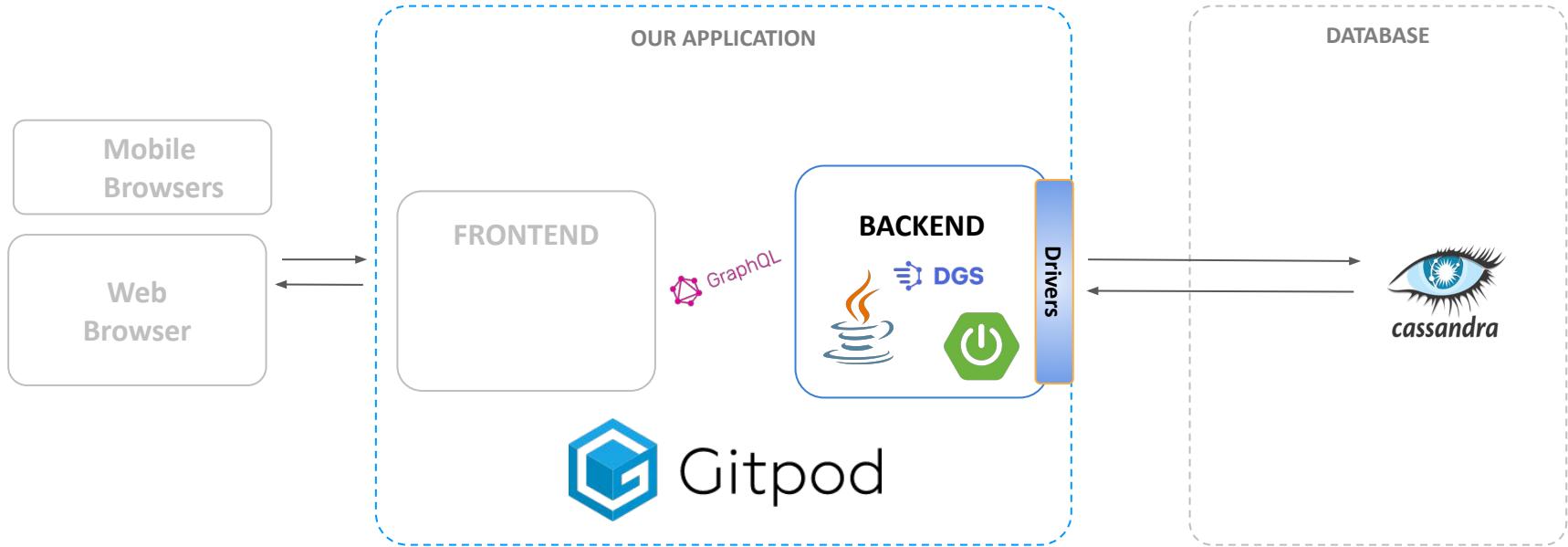




Sooo Connect the Front end to Graphql



Build a GraphQL API



Introduction to DGS by Netflix

```
public class Genre {  
    private final String value;  
  
    public Genre(String value) {  
        this.value = value;  
    }  
  
    public String getValue() {  
        return value;  
    }  
}
```

DataFetcher as @DgsComponent

```
@DgsComponent  
public class GenresDatafetcher {  
    Logger logger = Logger.getLogger(GenresDatafetcher.class.getName());  
  
    private final List<Genre> genres = List.of(  
        new Genre("Action"),  
        new Genre("Anime"),  
        new Genre("Award-Winning"),  
        new Genre("Children & Family"),  
        new Genre("Comedies")  
    );
```

Expose query with @DgsQuery

```
@DgsQuery  
public List<Genre> genres(@InputArgument String labelFilter) {  
    logger.info("QUERY executed - Genres value is: " + genres.get(0).getValue());  
  
    if(labelFilter == null) {  
        return genres;  
    }  
  
    return genres.stream().filter(s -> s.getValue().contains(labelFilter)).collect(Collectors.toList());  
}
```



Exposing a GQL Endpoint with

01



Housekeeping Live and Hands-on

02



Database Setup Data model & Astra DB

03



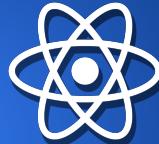
GraphQL Design & Toolings

04



Backend Expose GQL endpoint

05



Frontend Consume GQL Endpoint

06

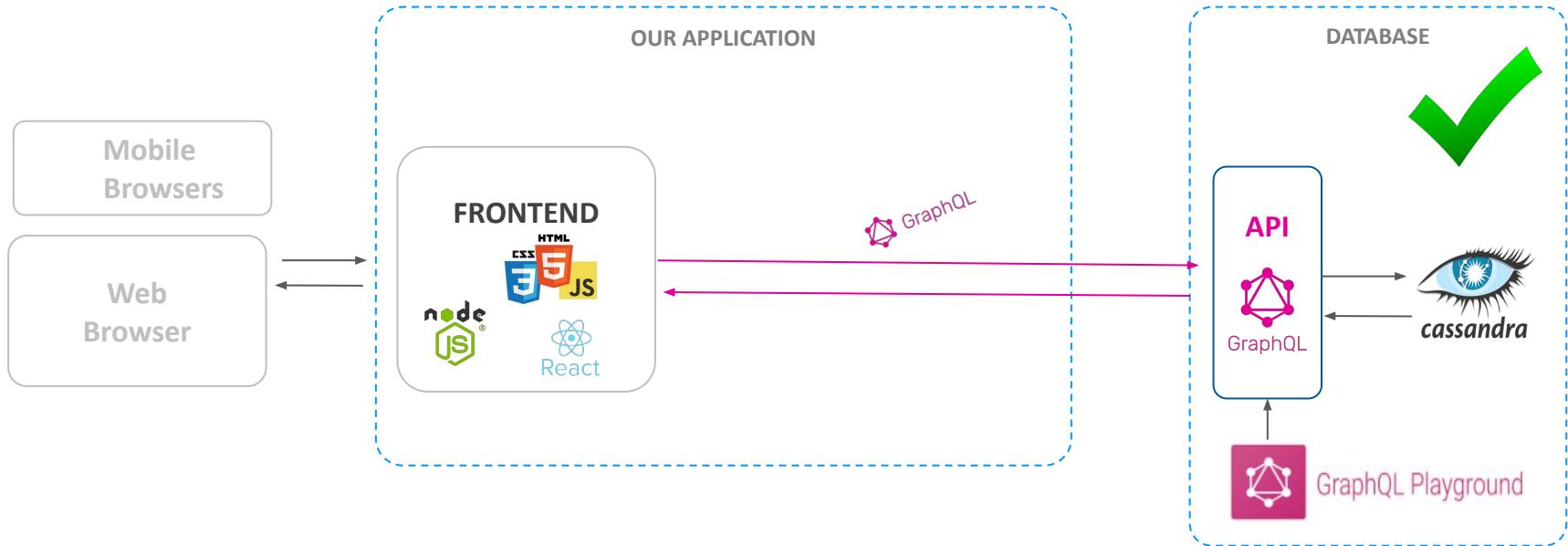


What's next? Quiz, Homework, ...

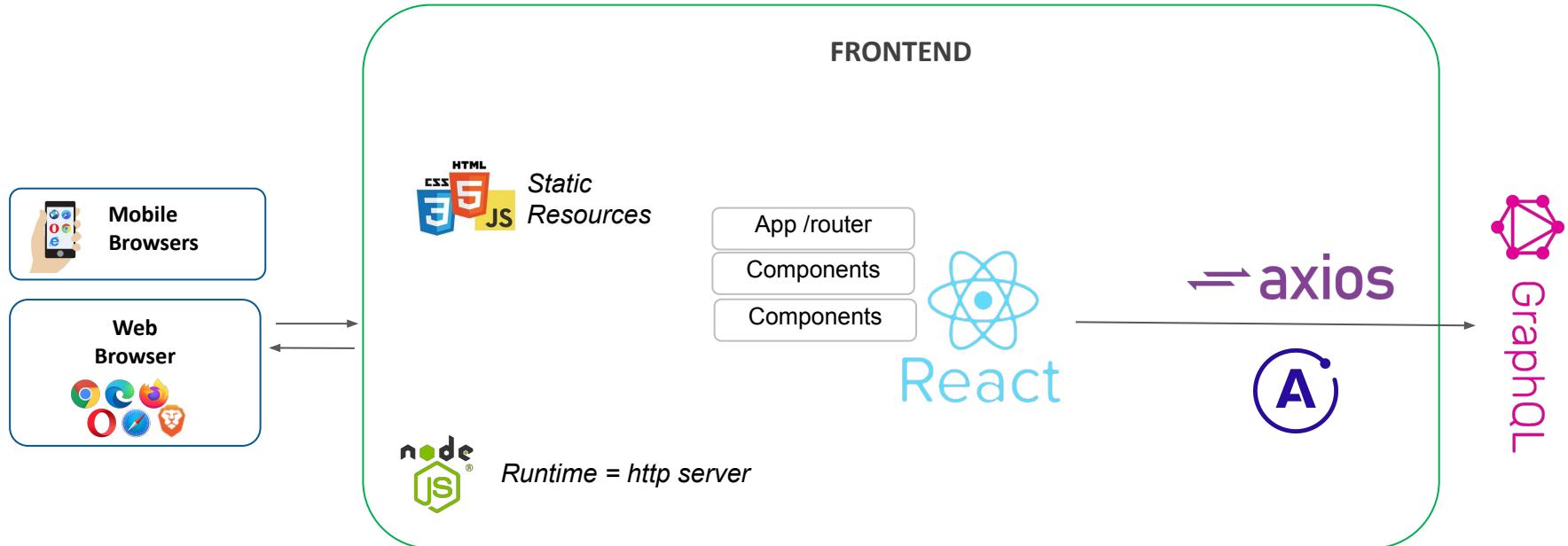


Agenda





Sooo Connect the Front end to Graphql

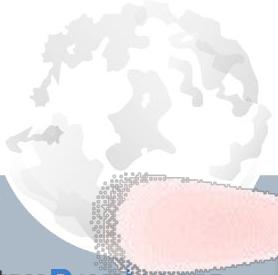


(*) Check out Apollo's `@apollo/client` library as well ...



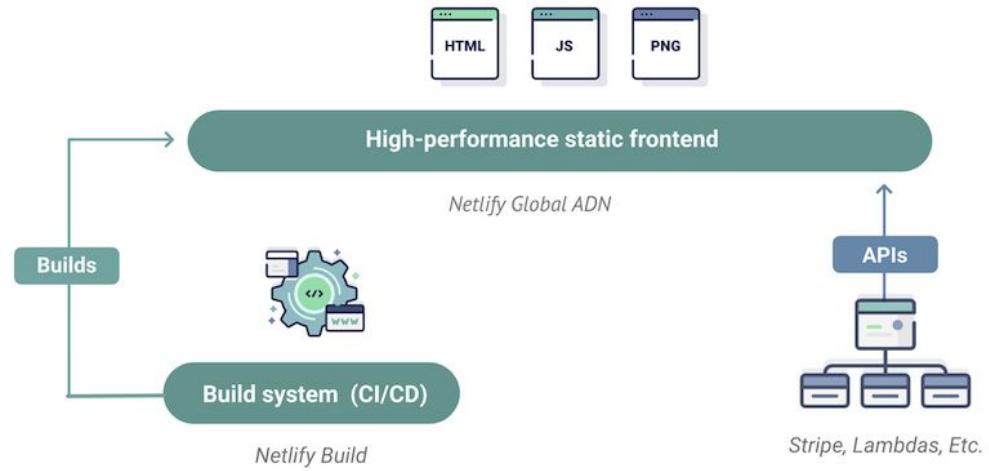
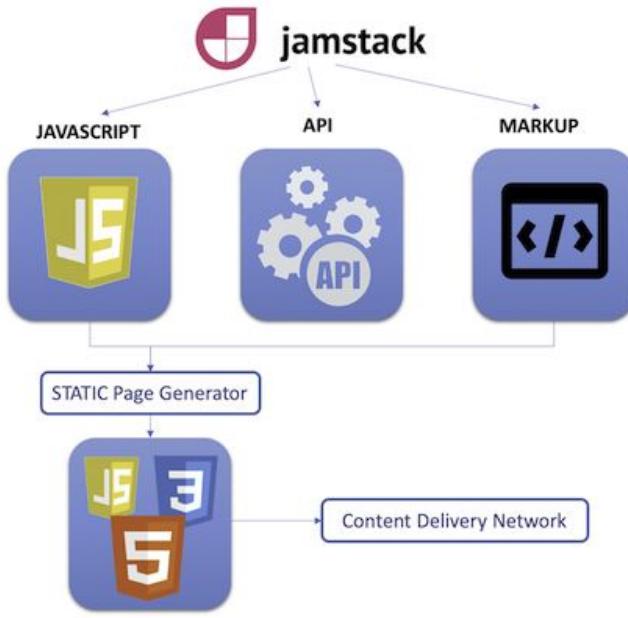
Our Front End



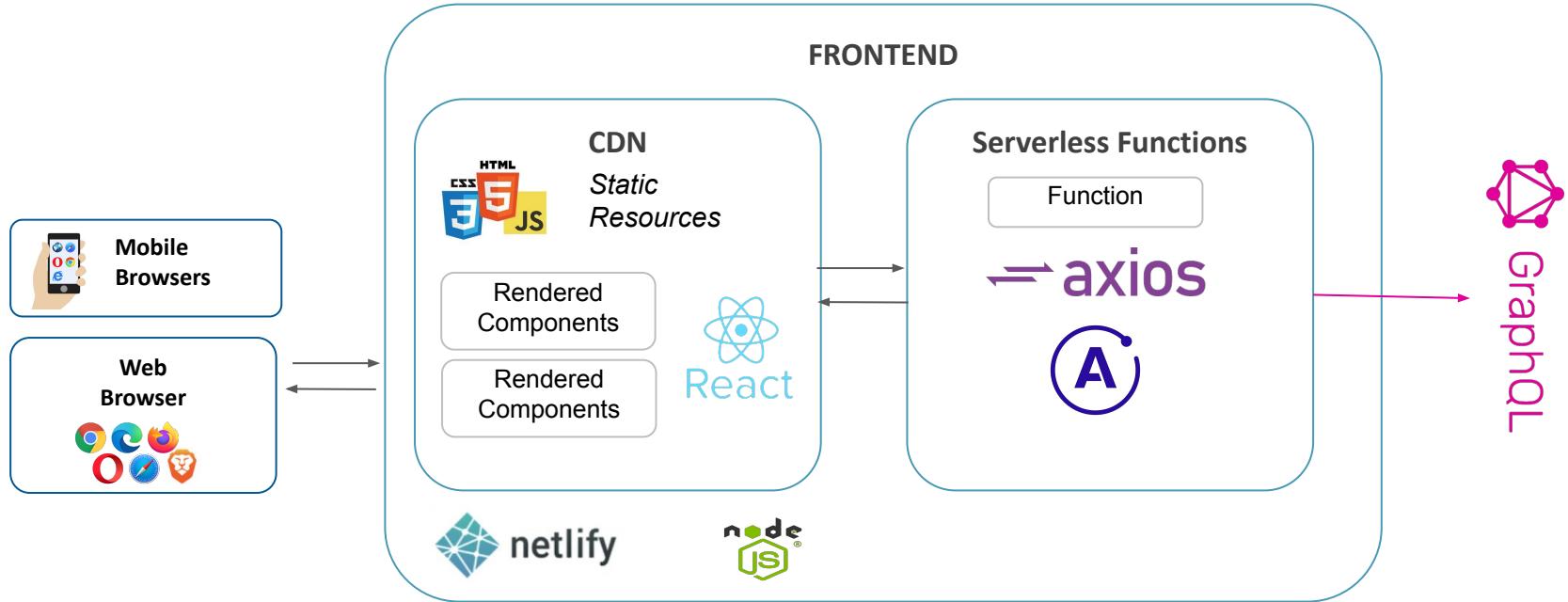


Cops are coming for you

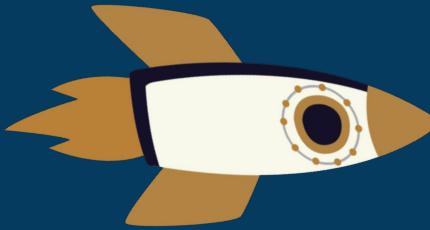




Introduction to Netlify



Our Front End



Hands-on (!github)

#3 Run the app

- ✓ React (start client, examine code)
- ✓ DB Connectivity (link app and DB)
- ✓ Run the final app (connected to DB, last touch)

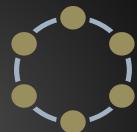


01



Housekeeping Live and Hands-on

02



Database Setup Data model & Astra DB

03



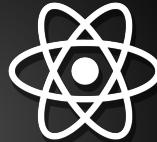
GraphQL Design & Toolings

04



Backend Expose GQL endpoint

05



Frontend Consume GQL Endpoint

06



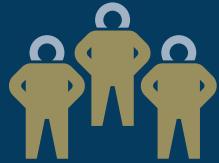
What's next? Quiz, Homework, ...



Agenda

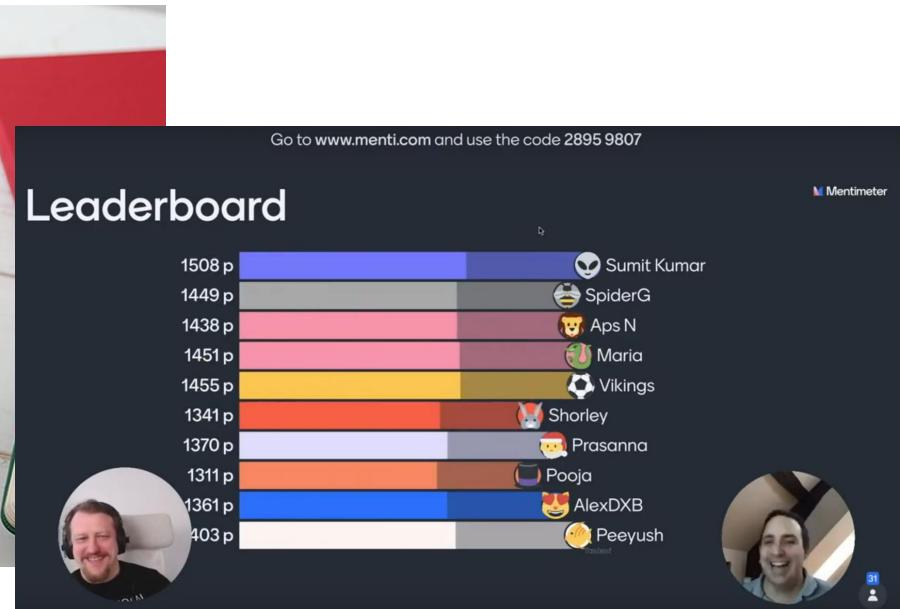
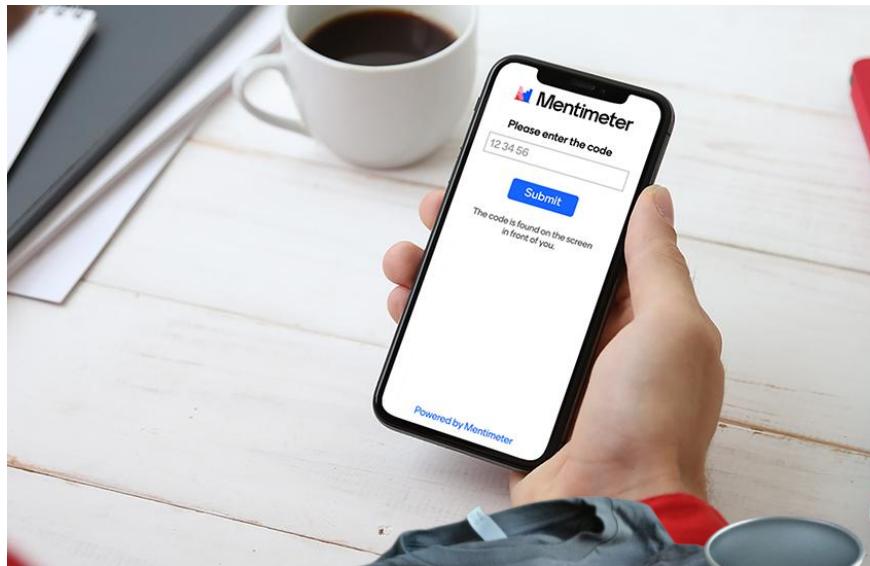


What's next?



Quiz, Homework, ...





**menti . com ⇒ enter code
Don't answer in YT chat
Look at phone (not at YT)**

Quiz on "Menti" !

SWAG WINNERS



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

To claim your prize, please send an email to:

gary.harvey@datastax.com

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



Swag Winners!



- **Homework**

- Complete the practice steps after step #9
- Insert (mutate) a show or genre of your choice in the database.
- Submit screenshot to us ... and wait for your **badge!**



- Optionally

- Watch "Netflix Clone GraphQL" workshop recording

- Do reach us for anything: dtsx.io/discord !



All info on Github – !homework



Assignment and Badge



DataStax Developers

workshop-chat

<https://www.youtube.com/watch?v=MuwT5xxFVVI> - Subscribe to mailing list: [http...](http://)

Rechercher

PRESENTER — 1
David Jones-Gilardi

HELPER — 7
012345
AaronP
B1nary
Chelsea Navo
Jeremy Hanna
John Sanda
Patrick_McFadin

EN LIGNE — 560
-samu-
6304-42JB
Aahlya
Abdurahim
abhi3pathi
Abhiis.s
Abhineet
Abirsh

Événements
moderator-only
. WELCOME
start-here
code-of-conduct
introductions
upcoming-events
useful-resources
memes
your-ideas
@the-stage

WORKSHOPS
workshop-chat
workshop-feedback
workshop-materials
upcoming-workshops

ASTRA DB
getting-started
astra-apis
astra-development
sample-applications

APACHE CASSANDRA
Cedrick Lun...

RIGGITYREKT Hier à 21:14
I have a 5 node datacenter, 4 nodes are on dse version 5.1.20, one is on dse5.0.15. I am doing some mixed version testing for a class and the one node that is 5.0.15 is coming up as an analytics workload. I dont have /etc/default/dse, instead I am using /etc/init.d/dse-cassandra.
how do I make that node start in cassandra workload, not in analytics?

RIGGITYREKT Hier à 23:39
Okay I found out my issue, when i started DSE 5.0.15 it had endpointsnitch set to DseSimpleSnitch, the rest of my cluster is using PropertyFileSnitch, when i change it to PropertyFileSnitch, it still uses the simple snitch config. looking at the docs i see there is a way to go to GossipingPropertyFileSnitch, but i need the property file one. I can wipe this dbs, do anything with this node to get this done. how do i fix this?
@here

19 novembre 2021
Erick Ramirez Aujourd'hui à 02:19
mixed versions isn't supported and you're guaranteed to run into weird issues that will cause further problems down the track

@RIGGITYREKT I have a 5 node datacenter, 4 nodes are on dse version 5.1.20, one is on dse5.0.15. I am doing some mixed v...
Cedrick Lunen Aujourd'hui à 09:01
When you start a node you have parameters -k for analytics, -g for graph and -s for search. To remove analytics check and remove -k

Envoyer un message dans #workshop-chat

!discord

dtsx.io/discord



DataStax Developers Discord (19k+)





**This summer..
Get Certified**

7/20

Getting started with
your first NoSQL
Database: Cassandra
Fundamentals

7/27

Scale like crazy with
proper data
modelling for
Apache Cassandra

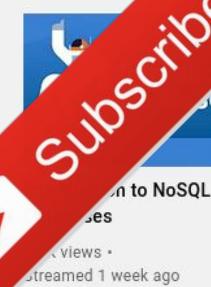
8/3

Build efficient
applications for
Apache Cassandra.



Upcoming live events (!workshops)

Subscribe



Subscribe



How to create an Authentication Token in...

37 views • 4 weeks ago

How to use the Data Loader in Astra DB

62 views • 4 weeks ago

Astra DB Sample App Gallery

36 views • 4 weeks ago

How to use Secure Connect in Astra DB

42 views • 4 weeks ago

Cassandra Day India: CL Room (Workshops)

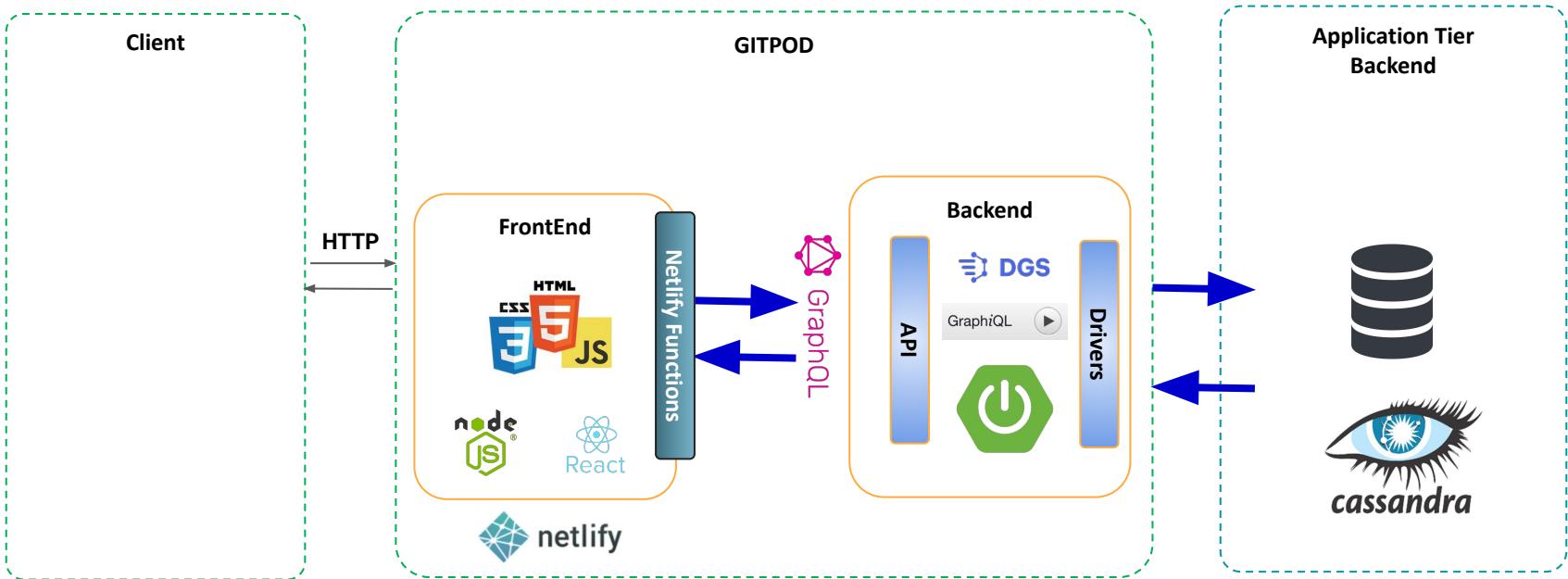
2.4K views • Streamed 4 weeks ago

Cassandra Day India: RF Room (Talks)

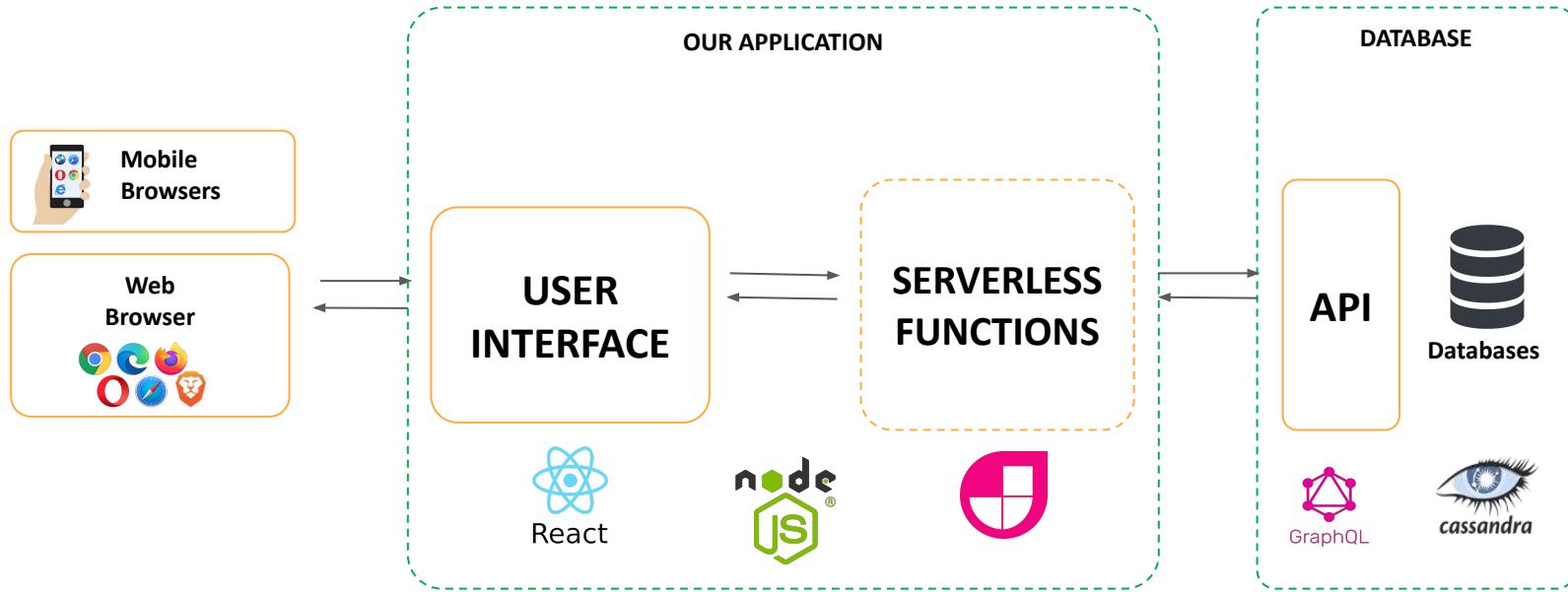
1.3K views • Streamed 1 month ago

Thank You!





Our Application Today



Building a Full-stack application

```
[query] [$label] {
```

```
[
```

```
$function_name[(arguments)] {
```

```
values_to_retrieve_for_each_type 1...n
```

```
}
```

```
1...n
```

```
}
```



Anatomy of query

