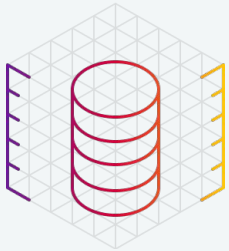
DATASTAX
DEVELOPERS

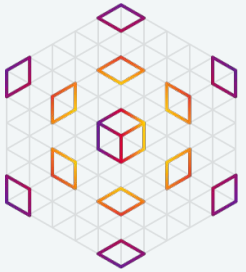
AI as an API

Machine Learning with some REST

Based on original content by **CodingEnterpreneurs**



Agenda



01

Housekeeping
Live and Hands-on

02

Our Goal
Use case and Technologies

03

Database Setup
Data model & Astra DB

04

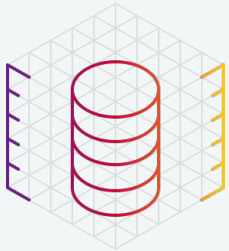
AI
Train a text classifier

05

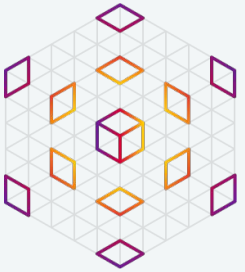
API
Bring to production

06

What's next?
Quiz, Homework, Agenda



Agenda



01

Housekeeping
Live and Hands-on

02

Our Goal
Use case and Technologies

03

Database Setup
Data model & Astra DB

04

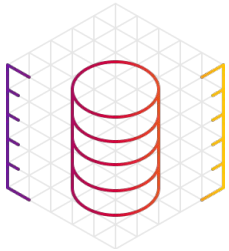
AI
Train a text classifier

05

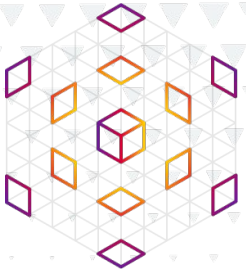
API
Bring to production

06

What's next?
Quiz, Homework, Agenda



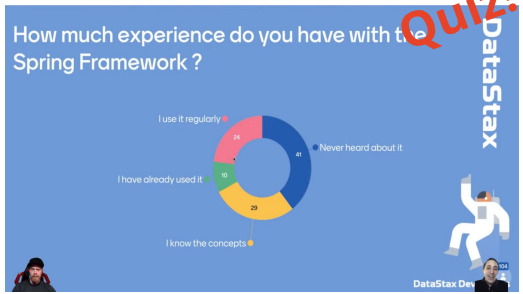
❯ Live & Interactive



youtube.com/DataStaxDevs

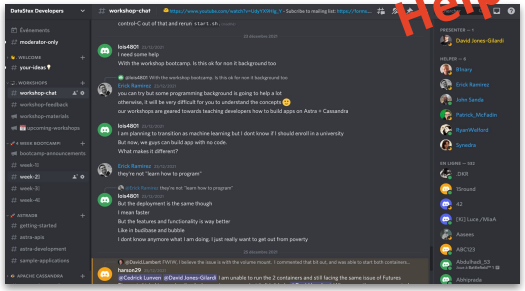
YouTube

⌨ !menti



Mentimeter

⌨ !discord



Discord (#workshop-chat)

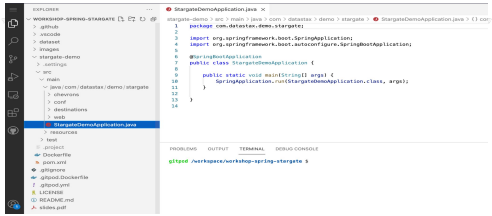
Hands-on housekeeping

Mini-course + slides + hands-on

DATASTAX ACADEMY

start from GitHub !github

IDE



Gitpod !gitpod

Database + Api + Streaming

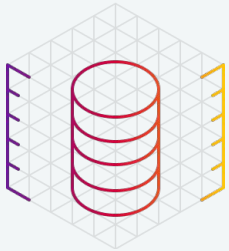
DATASTAX ASTRA DB !astra

CodingEnterpreneurs

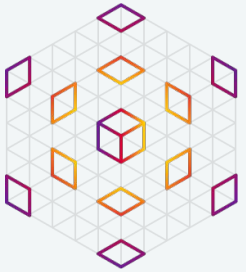
Youtube & blog for more

Nothing to install !

DATASTAX DEVELOPERS



Agenda



01

Housekeeping
Live and Hands-on

02

Our Goal
Use case and Technologies

03

Database Setup
Data model & Astra DB

04

AI
Train a text classifier

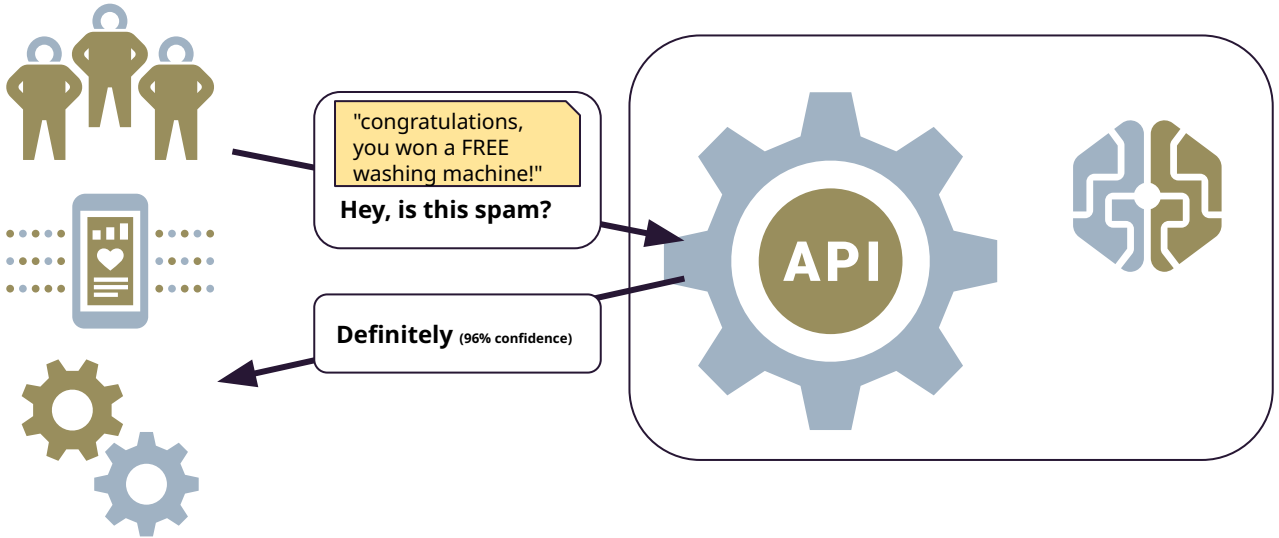
05

API
Bring to production

06

What's next?
Quiz, Homework, Agenda

What we want

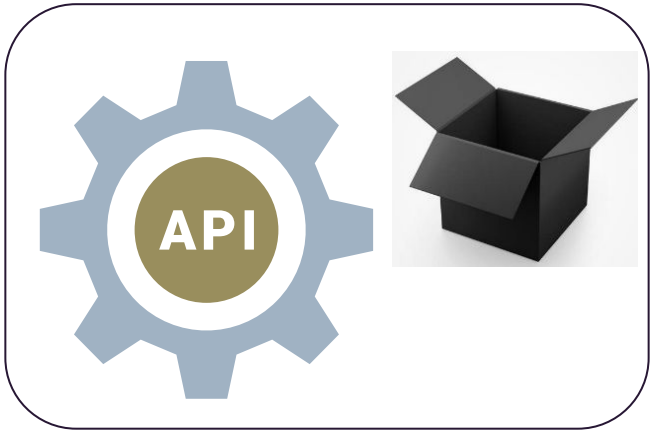

"congratulations, you won a FREE washing machine!"
Hey, is this spam?
Definitely (96% confidence)
API

Traditional viewpoint

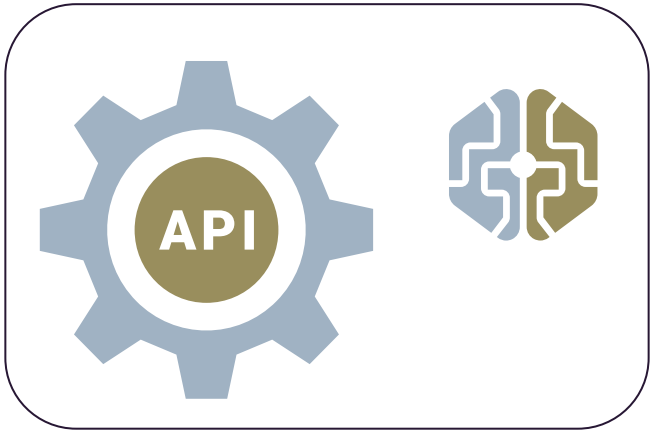
The API sees the AI part as a black box

API = concerned with stability & performance

Engineer Stuff™



Traditional viewpoint

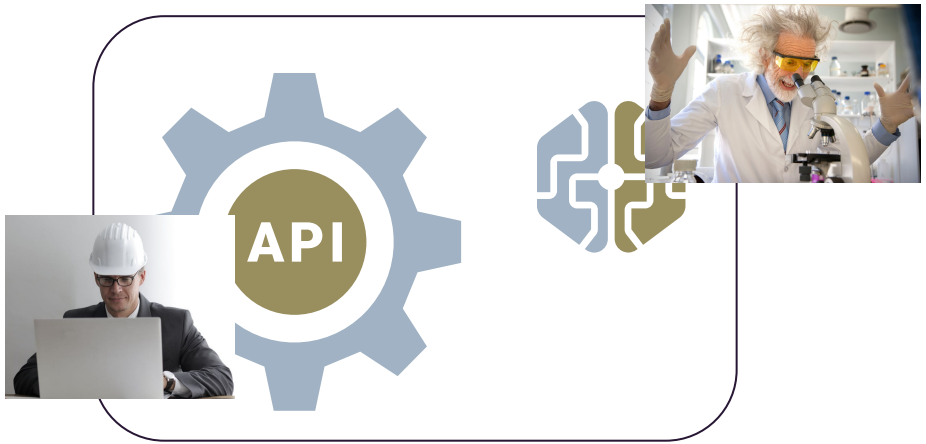


ML components, often messy (no offence meant)

Treasures hidden away (hardly reproducible)

A "science" proper (experimentation, prototyping...)

Two in one

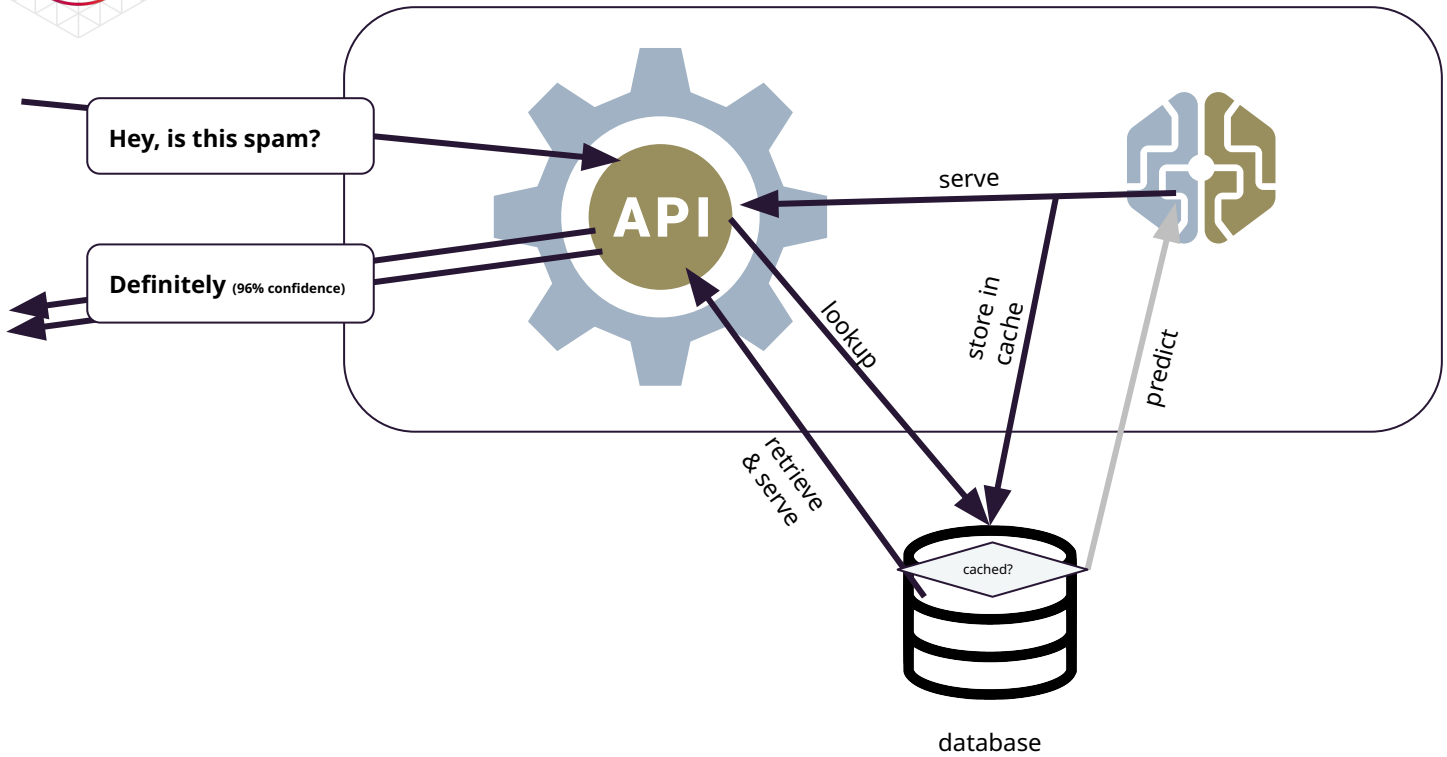


Properly exposing ML elements in production is a key skill

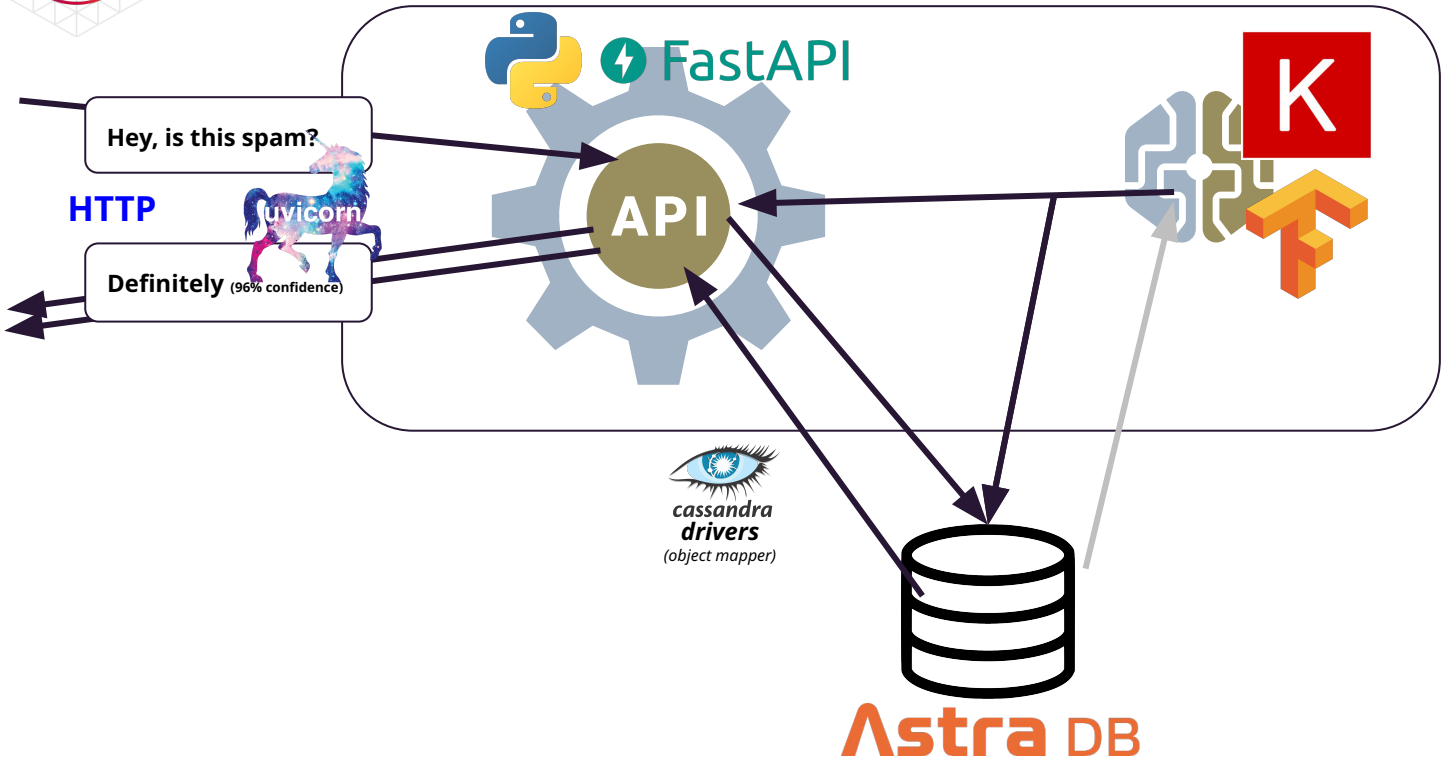
Still, two clearly separate phases:

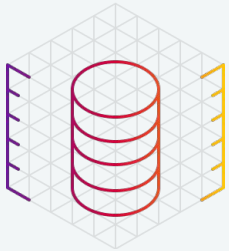
- create the classifier (design, train)
- bring it to production

Architecture sketch

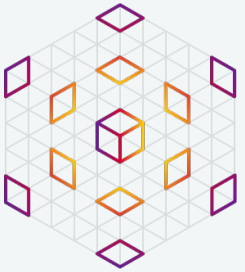


❯ Tech stack


FastAPI
Hey, is this spam?
HTTP
uvicorn
Definitely (96% confidence)
API
K
cassandra
drivers
(object mapper)
Astra DB



Agenda



01

Housekeeping
Live and Hands-on

02

Our Goal
Use case and Technologies

03

Database Setup
Data model & Astra DB

04

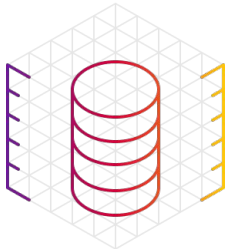
AI
Train a text classifier

05

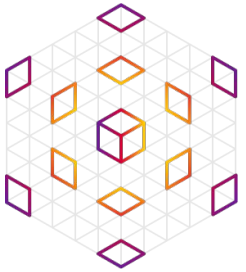
API
Bring to production

06

What's next?
Quiz, Homework, Agenda



Intro to Apache Cassandra

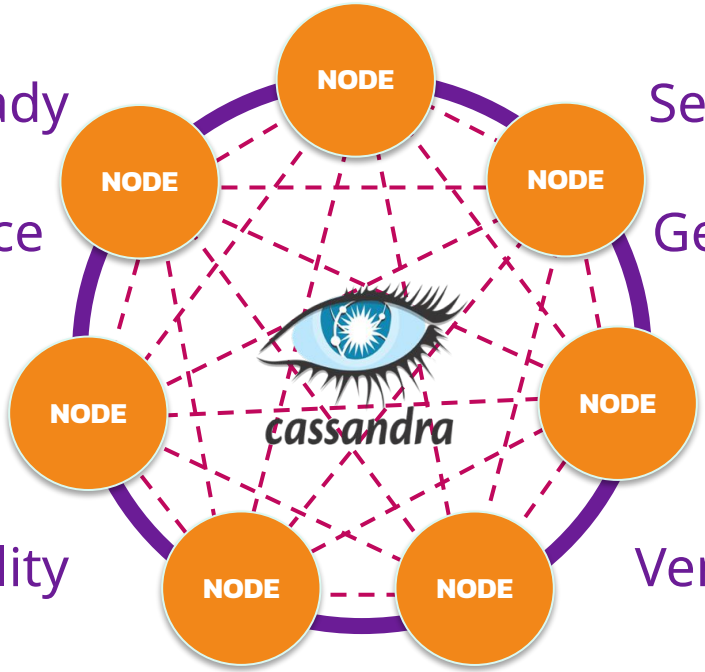


Big Data Ready

Read/Write Performance

Linear Scalability

Highest Availability

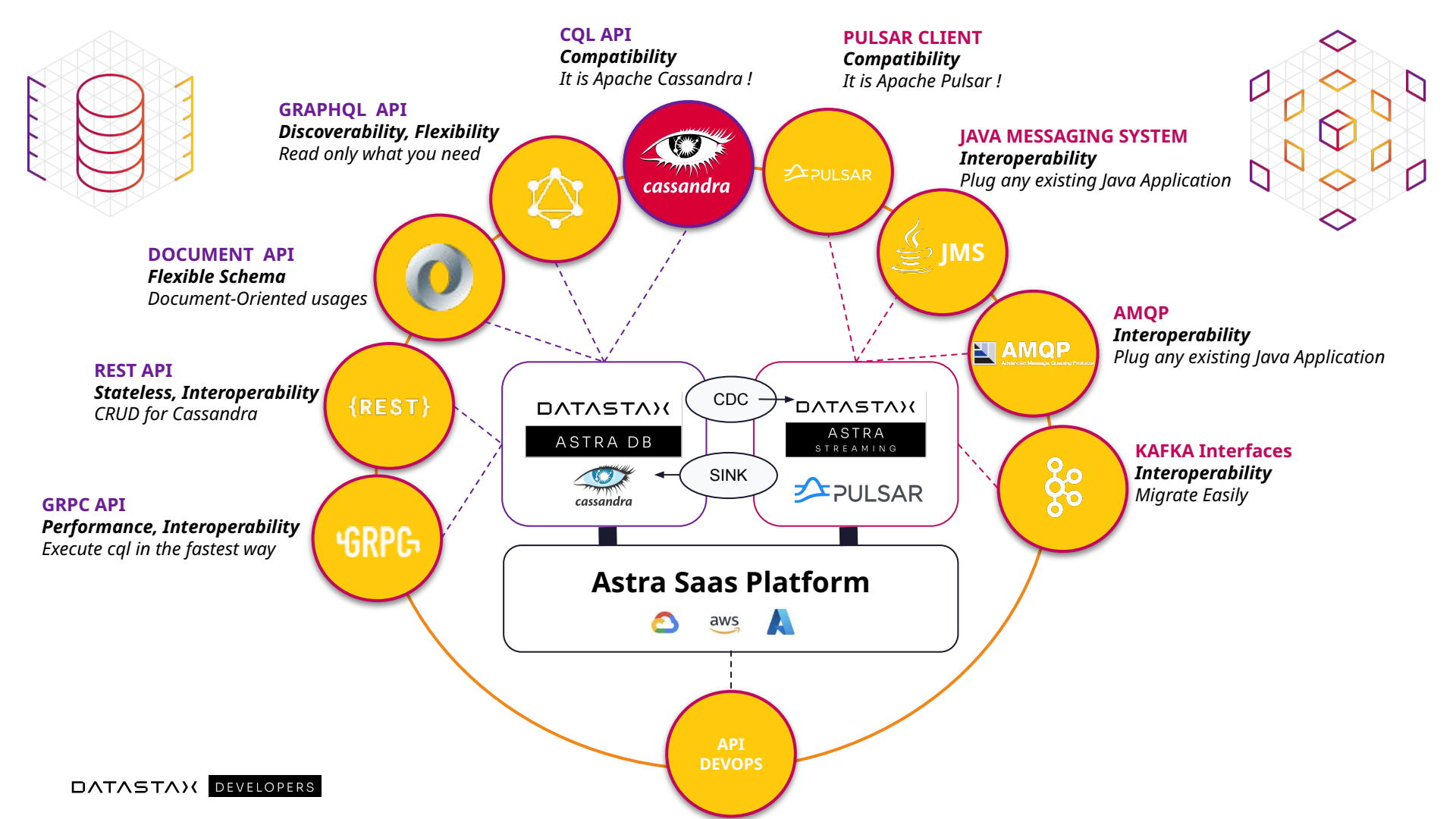


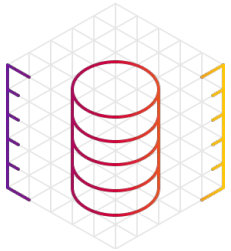
Self-Healing, Automation

Geographical Distribution

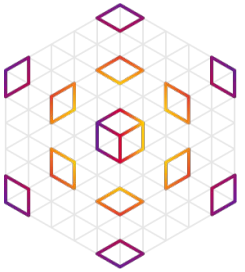
Platform-agnostic

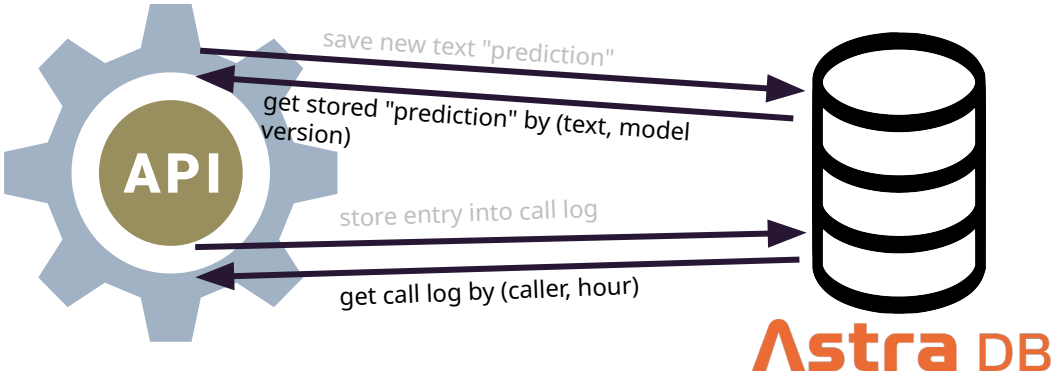
Vendor-independent


CQL API
Compatibility
It is Apache Cassandra !
PULSAR CLIENT
Compatibility
It is Apache Pulsar !
GRAPHQL API
Discoverability, Flexibility
Read only what you need
JAVA MESSAGING SYSTEM
Interoperability
Plug any existing Java Application
DOCUMENT API
Flexible Schema
Document-Oriented usages
AMQP
Interoperability
Plug any existing Java Application
REST API
Stateless, Interoperability
CRUD for Cassandra
KAFKA Interfaces
Interoperability
Migrate Easily
GRPC API
Performance, Interoperability
Execute cql in the fastest way
cassandra
PULSAR
JMS
AMQP
{REST}
GRPC
DATASTAX
ASTRA DB
cassandra
CDC
SINK
DATASTAX
ASTRA
STREAMING
PULSAR
Astra Saas Platform
aws
API
DEVOPS
DATASTAX DEVELOPERS



DB usage patterns



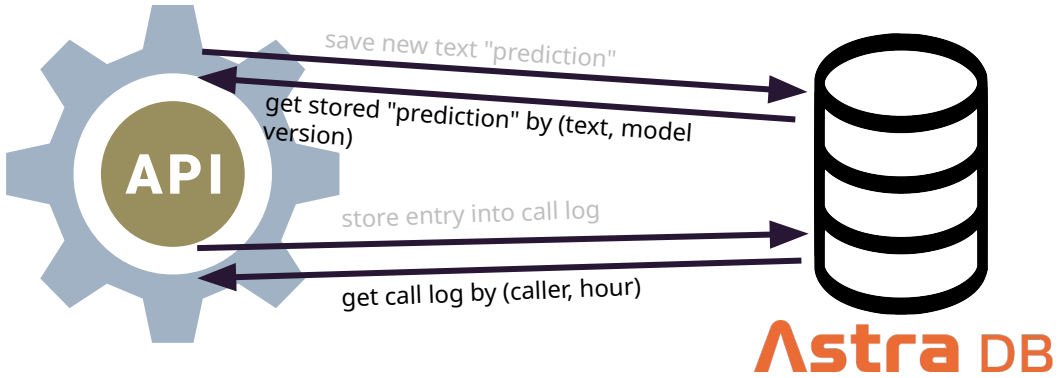


Data modeling, the Cassandra way:

"design the table after the query"

(also: tables are partitioned!)

Data modeling



spam_cache_items

model_version	**text**	**K**
input	**text**	**K**
confidence	float	
prediction_map	map<text,float>	
result	text	
stored_at	timeuuid	

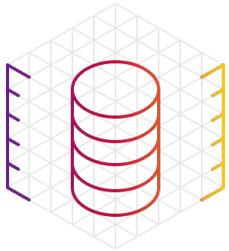
spam_calls_per_caller

caller_id	**text**	**K**
called_hour	**timestamp**	**K**
called_at	**timeuuid**	**C↑**
input_text	text	

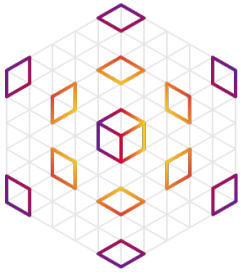
"partition key" (rows stored together)

"primary key" (row uniqueness)

"Chebotko diagrams"



❯ DB I/O in practice



cassandra drivers

(object mapper)

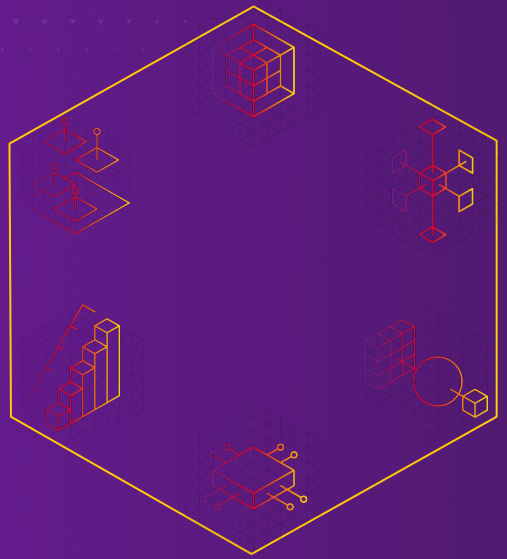
```
class SpamCallItem(Model):
    __table_name__ = 'spam_calls_per_caller'
    __keyspace__ = ASTRA_DB_KEYSPACE
    __connection__ = 'my-astra-session'
    caller_id = columns.Text(primary_key=True, par
    called_hour = columns.DateTime(primary_key=Tru
    called_at = columns.TimeUUID(primary_key=True,
    input = columns.Text()
```

Tables are created automatically…

```
query = SpamCallItem.objects().filter(
    caller_id=caller_id,
    called_hour=called_hour,
)
for item in query:
    yield item
```

```
#
# Database
logging.info('      DB initialization')
DBSession = initSession()
sync_table(SpamCacheItem)
sync_table(SpamCallItem)
logging.info('      API Startup completed.')
```

```
called_hour = getThisHour()
for input in inputs:
    SpamCallItem.create(
        caller_id=caller_id,
        called_hour=called_hour,
        input=input,
    )
```

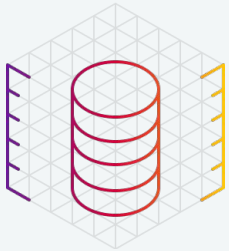


Lab 1

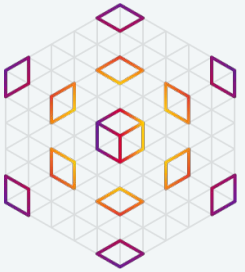
DB setup

Create your Astra DB instance

Create a "DB Admin" token



Agenda



01

Housekeeping
Live and Hands-on

02

Our Goal
Use case and Technologies

03

Database Setup
Data model & Astra DB

04

AI
Train a text classifier

05

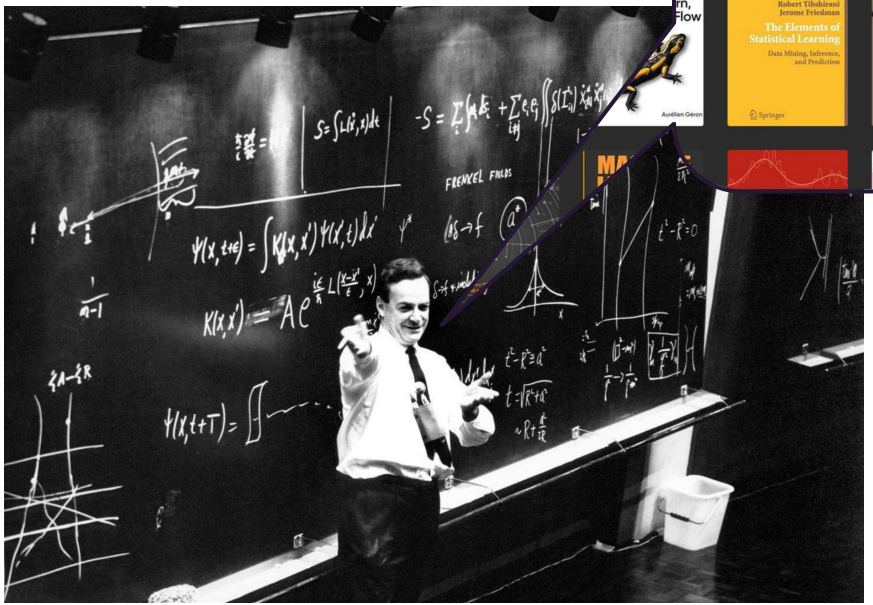
API
Bring to production

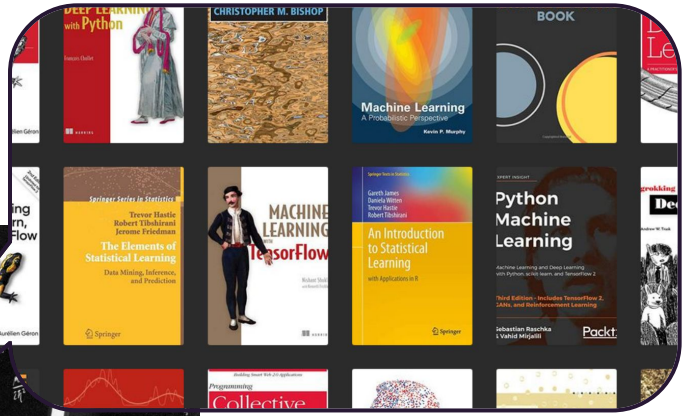
06

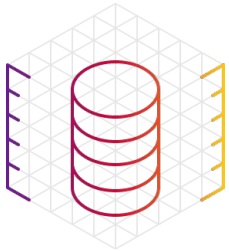
What's next?
Quiz, Homework, Agenda

❯ Disclaimer

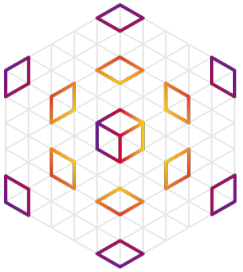
What this section is **not**:





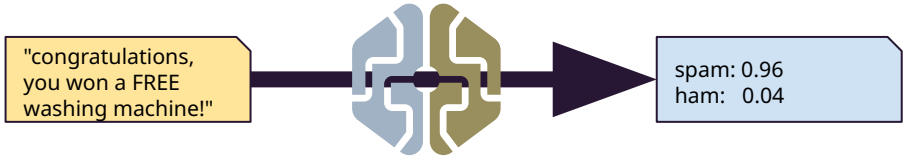


❯ AI and ML



"ML: LSTM RNN for NLP"

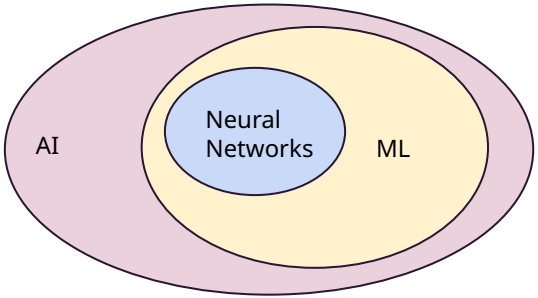
This is what we want:



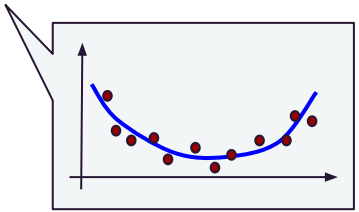
Machine Learning = *algorithms that improve by being fed data, without being explicitly instructed what to do.*

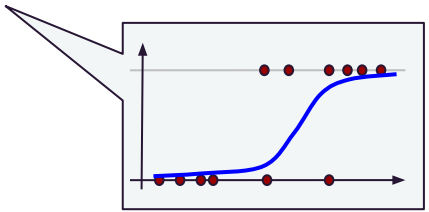
It's essentially statistical inference (with superpowers).

Lots of **math** involved (linear algebra, calculus, prob./statistics).
Nowadays accessible as neatly-packaged tools (good for us!)



Simple examples of ML: least-squares fits, logistic regressions.





❯ Supervised learning

"Blank slate" model

Training
(i.e. tune parameters)

Ready to predict !

(classify, recognize objects in images, do speech-to-text, ...)

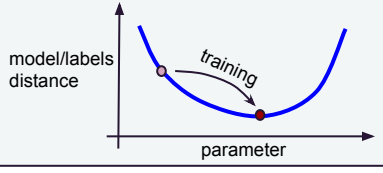
Labeled training

X	Y
Go until jurong poin	ham
Ok lar... Joking wif	ham
Free entry in 2 a wk	spam
U dun say so early h	ham
Nah I don't think he	ham
FreeMsg Hey there da	spam
Even my brother is n	ham
As per your request	ham
WINNER!! As a valued	spam
Had your mobile 11 m	spam

Choice of the model
(also "hyperparameters")

"Training":

tuning model parameters to make predictions as close to labeled data as possible *(more later)*



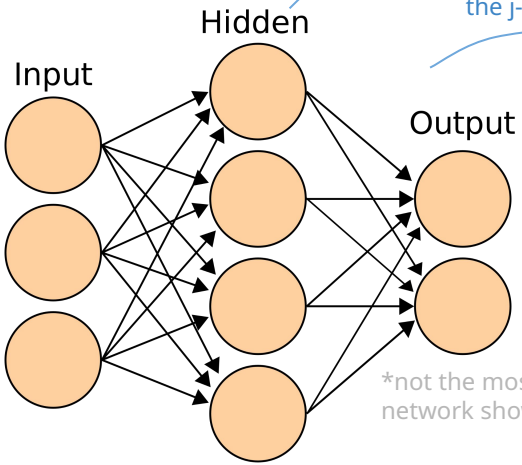
Neural Networks

Many algorithms and structures for ML:

linear/logistic regression, support-vector-machines, decision trees/random forests, gradient boosting, ...

Focus on **Neural Networks**

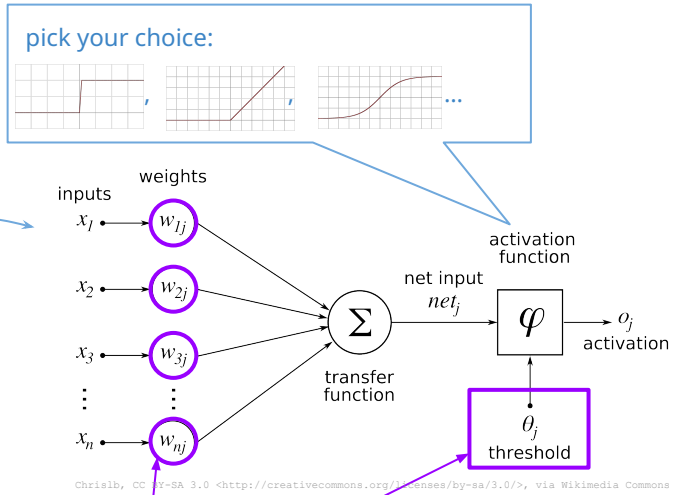
"deep" networks have several hidden layers



Input (text, image, ...) is encoded and fed here

*not the most general network shown here

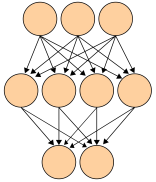
the j-th neuron looks like:



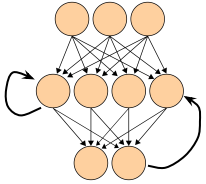
many parameters per neuron:
training gets CPU- and time-intensive

❯ RNN and LSTM

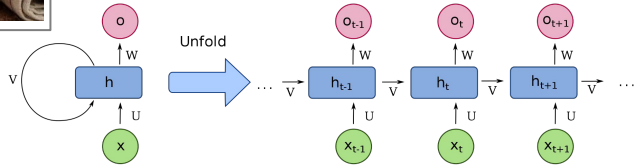
Architectures



"Feed-forward"



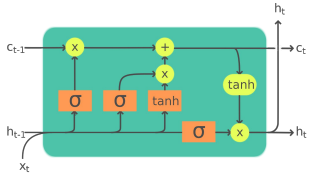
"Recurrent neural network"



By fdeloche - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=60109157

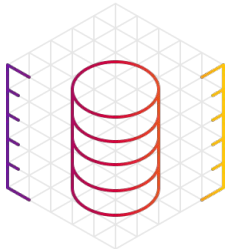
has *loops*: ~ "memory"
better with sequences (time series, speech, **texts**, ...)
loops "unfold" as time-steps: training gets harder

"Long-short-term-memory"

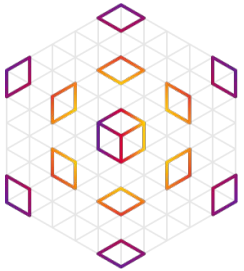


By Guillaume Chevalier - File:The_LSTM_Cell.svg, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=109362147

introduces explicit memory elements
overcomes the "vanishing gradient" training problem
a standard RNN to process **"input sequences"**



NLP

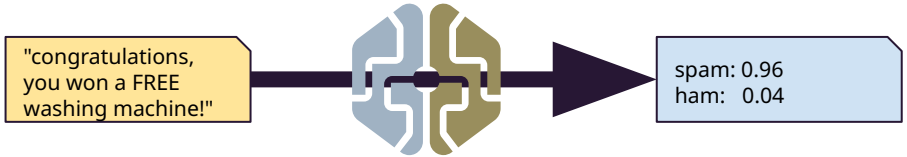


Natural language processing

A very wide field of application for ML (since ... forever)
translation, summarization, classification, clustering, autocompletion, generation, chatbots, sentiment analysis, ...

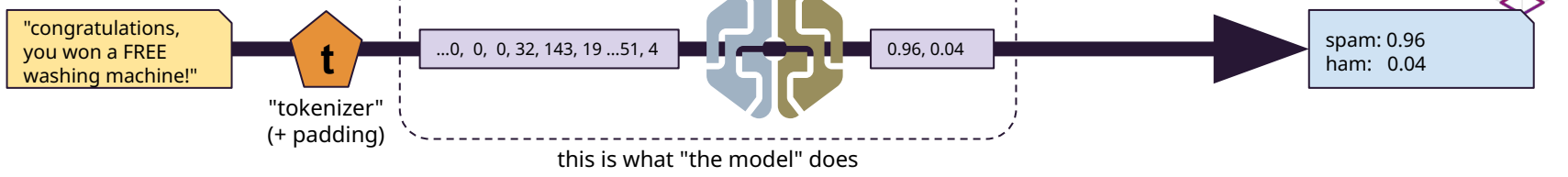
Most of the data "out there" is unstructured (e.g. **text**)

LSTM very apt at many of these tasks (memory along sequences)

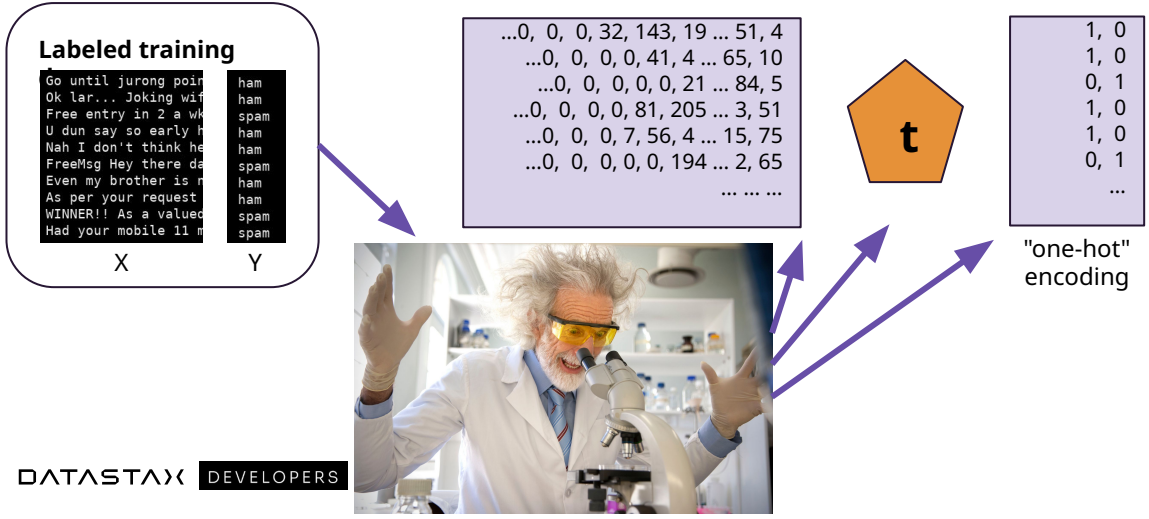


An evolving field (Word2Vec ~ 2013; transformers e.g. BERT ~ 2018, "foundation models" ~ 2020 , **and we all know where we're now***)*

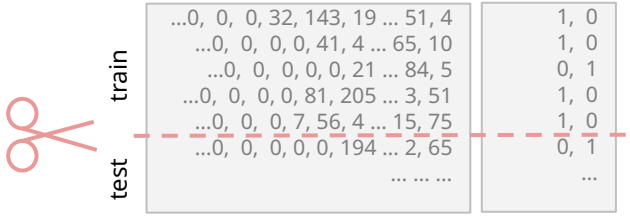
A closer look: numeric encoding

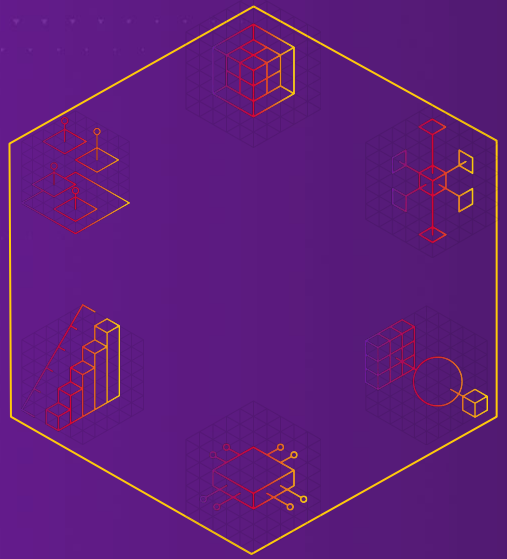


We must then *prepare the dataset* before training (including building the tokenizer):



Also split "train"/"test"





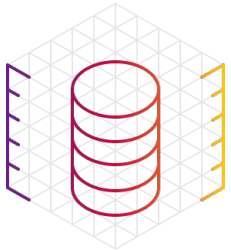
Lab 2

Train the model

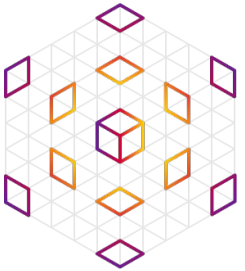
Prepare the dataset

Train the NLP model

Test the saved model



❯ The antispam model architecture




```
model = Sequential()
model.add(Embedding(maxNumWords, embedDim,
model.add(SpatialDropout1D(0.4))
model.add(LSTM(lstmOut, dropout=0.3, recurre
model.add(Dense(2, activation='softmax'))
model.compile(loss='categorical_crossentropy
```

A stack of layers:

one translates an input number into a vector

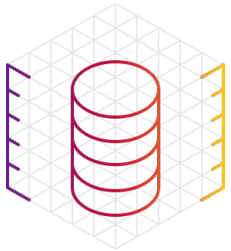
one randomly disables pieces of network to enhance training ("dropout")

then the LSTM (recurrent *within the layer*)

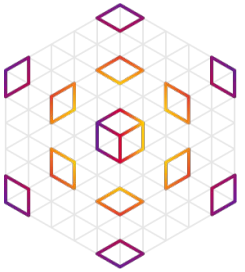
and a final ordinary layer reducing it all to a 2-component output (spam/ham)

"finalize" the model: ready to train!

inputs



❯ During training ...

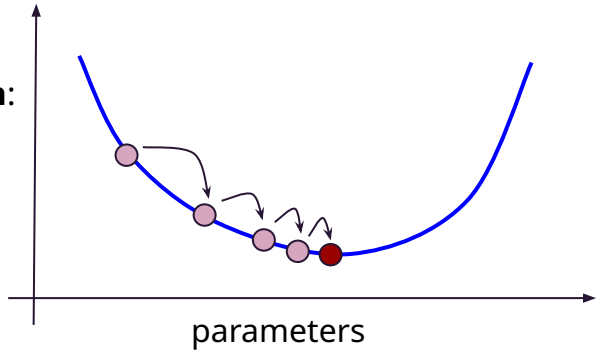


"Training":

(progressively) tuning model parameters to make predictions as close to labeled data as possible

loss function:
distance between provided labels and model output for training data

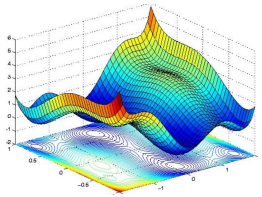
e.g. "categorical cross-entropy" (from information theory)



The reality:

lots of parameters, it's a mess!

```
=========================
Total params: 291,034
Trainable params: 291,034
Non-trainable params: 0
```

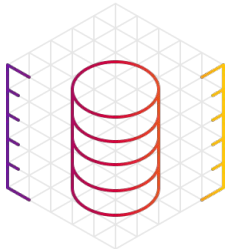


"Stochastic gradient descent" (SGD) 

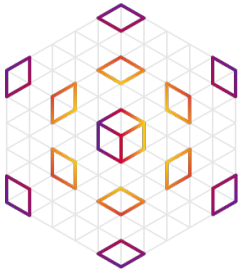
Backpropagation (glorified chain rule )

Drop-out 

...

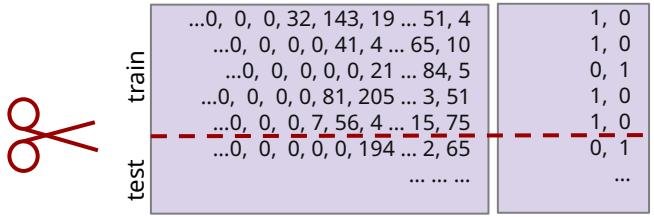


❯ Approaching the trained model



```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
```

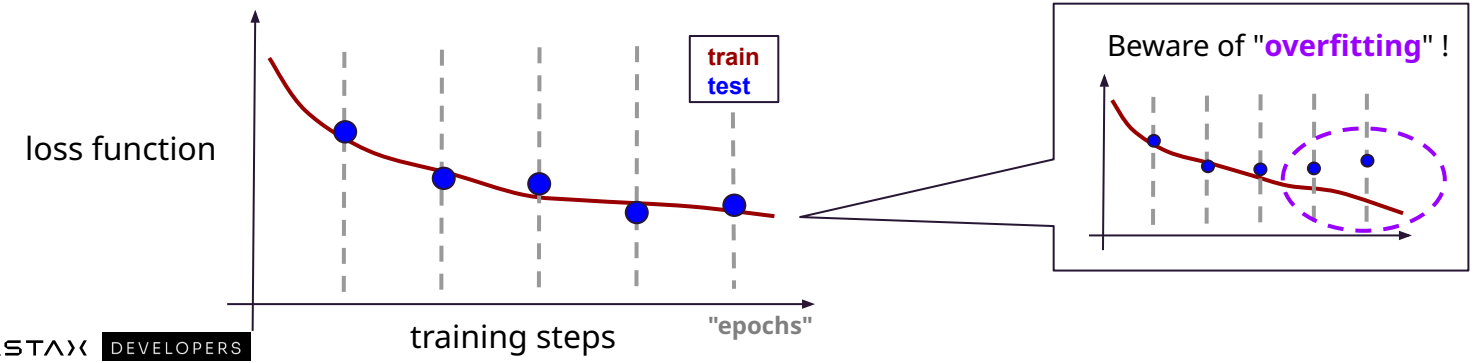
We split "train"/"test"

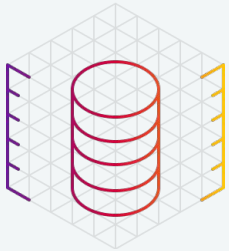


Train on part of input dataset,

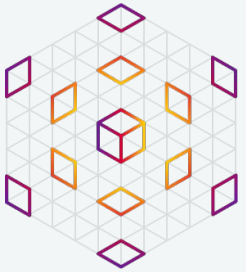
periodically validate on *another portion* of it

(independently verify we're actually learning the right problem)





Agenda



01

Housekeeping
Live and Hands-on

02

Our Goal
Use case and Technologies

03

Database Setup
Data model & Astra DB

04

AI
Train a text classifier

05

API
Bring to production

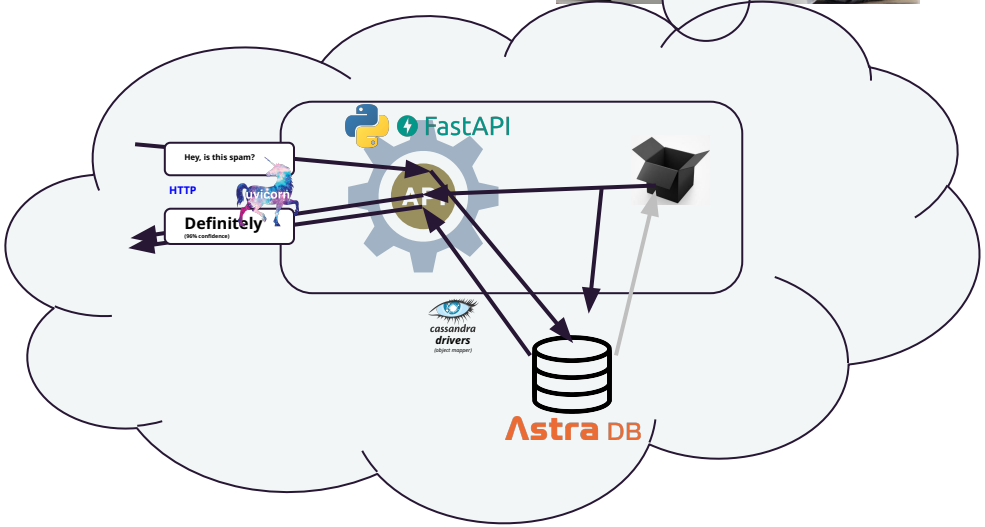
06

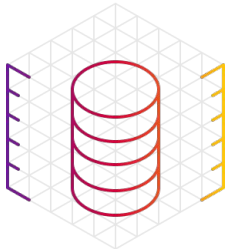
What's next?
Quiz, Homework, Agenda

What now?

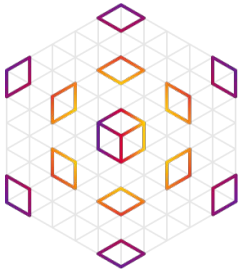
Here, take this

API

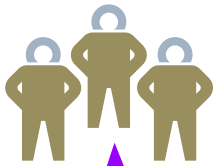


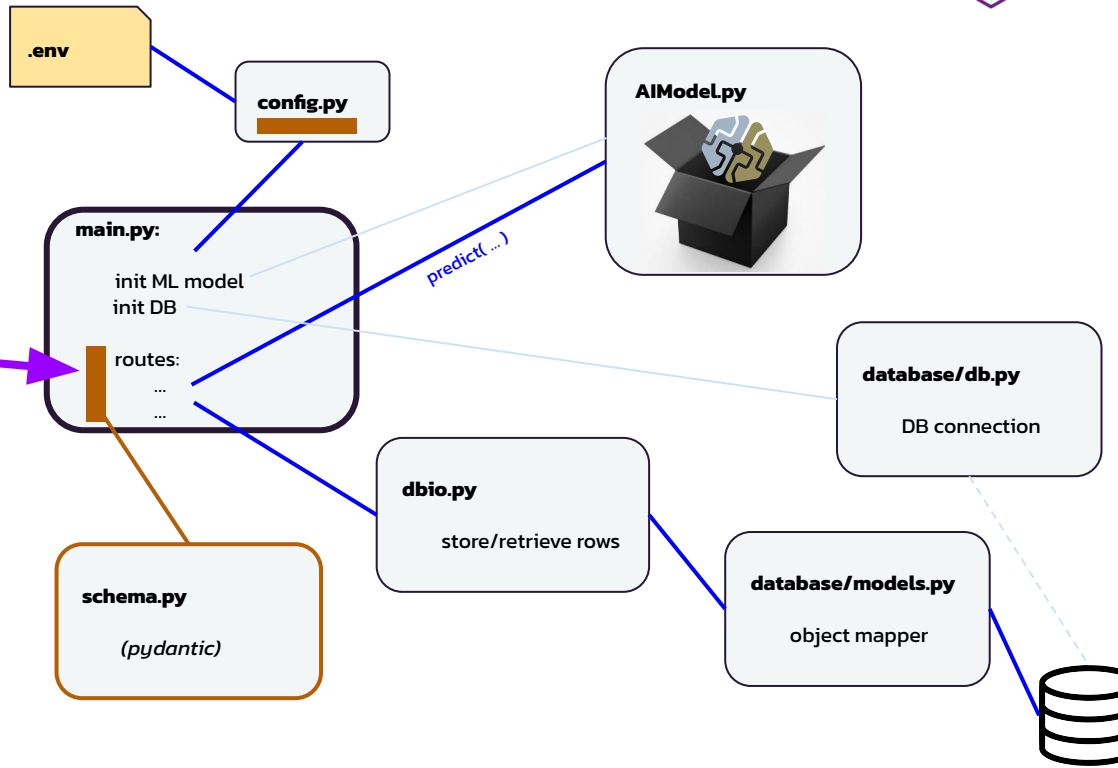


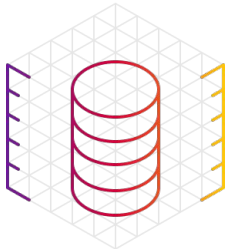
❯ API structure



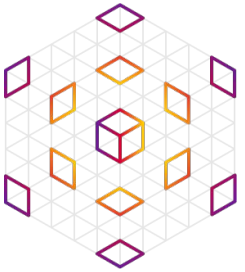
```
api
  ├── main.py
  ├── AIModel.py
  ├── schema.py
  ├── dbio.py
  ├── database
  │    ├── db.py
  │    └── models.py
  ├── config.py
  ├── minimain.py
  ├── MockSpamAIModel.py
  └── tests (...)
```







❯ Features & highlights



FastAPI fundamentals: routes, body validation

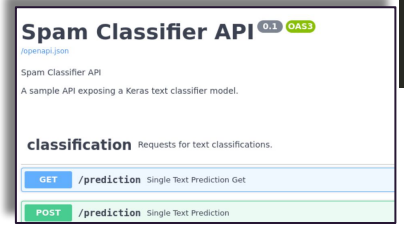
caching on DB to avoid re-processing the same input

API docs, Swagger(Open API) UI

call logs & **StreamingResponse**

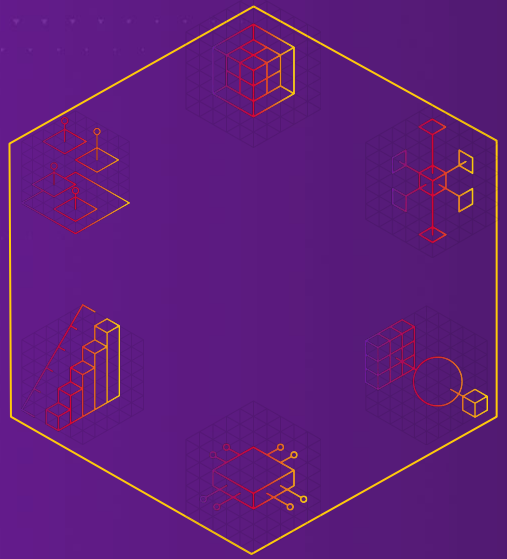
GET and query string

```
settings = getSettings()
cached = None if query.skip_cache else readCachedPrediction(settings.model_versio
storeCallsToLog([query.text], request.client[0])
if not cached:
    result = spamClassifier.predict([query.text], echoInput=query.echo_input)[0]
    cachePrediction(query.text, result)
    result['from_cache'] = False
    return PredictionResult(**result)
else:
    cached['from_cache'] = True
    return PredictionResult(**cached)
```




```
@app.get('/recent_log', response_model=List[CallerLogEntry], tags=['info'])
def get_recent_calls_log(request: Request):
    caller_id = request.client[0]
    called_hour = getThisHour()
    #
    return StreamingResponse(formatCallerLogJSON(caller_id, called_hour))
```

```
@app.get('/prediction', response_model=PredictionResult, tags=['classification'])
def single_text_prediction_get(request: Request, query: SingleTextQuery = Depends()):
    """
```



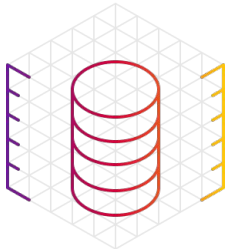
Lab 3

API

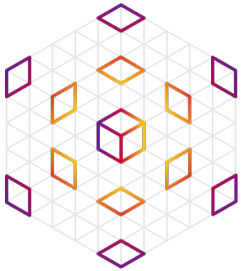
Serve with FastAPI

Explore Swagger (Open API) UI

Look at the DB



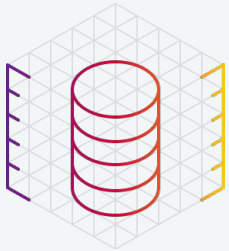
› More topics



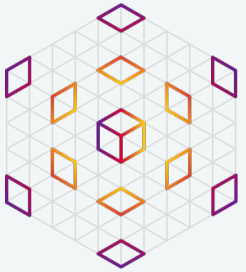
Want to know more on …

- feature stores
- architectural challenges of ML-based applications
- model versioning with FastAPI
- other MLOps-related issues

Check out **github.com/datastaxdevs/workshop-ai-as-api** (Appendix I, Appendix II)



Agenda



01

Housekeeping
Live and Hands-on

02

Our Goal
Use case and Technologies

03

Database Setup
Data model & Astra DB

04

AI
Train a text classifier

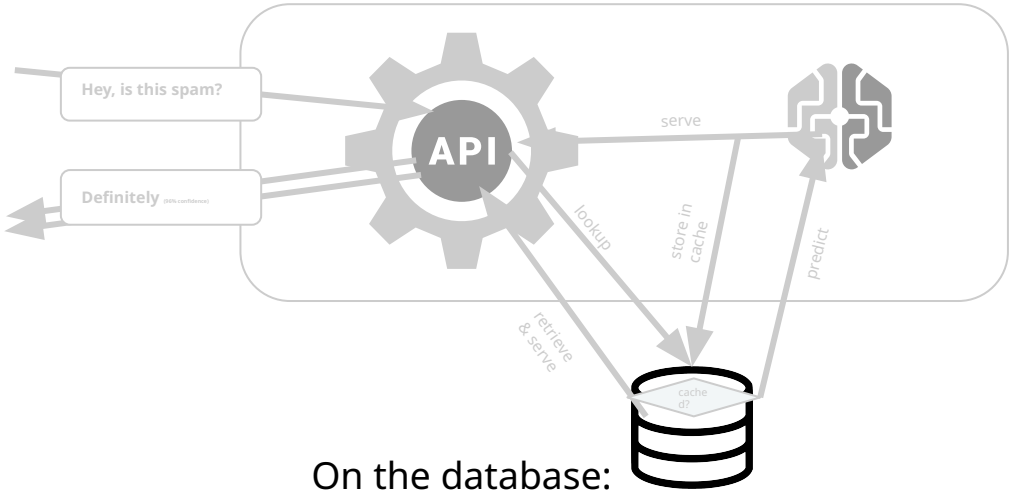
05

API
Bring to production

06

What's next?
Quiz, Homework, Agenda

❯ Going "real-time AI"



On the database:

- prediction store (caching, re-using)
- user history (calls, inputs)
- per-sender time-series (spam/ham events)

KASKADA

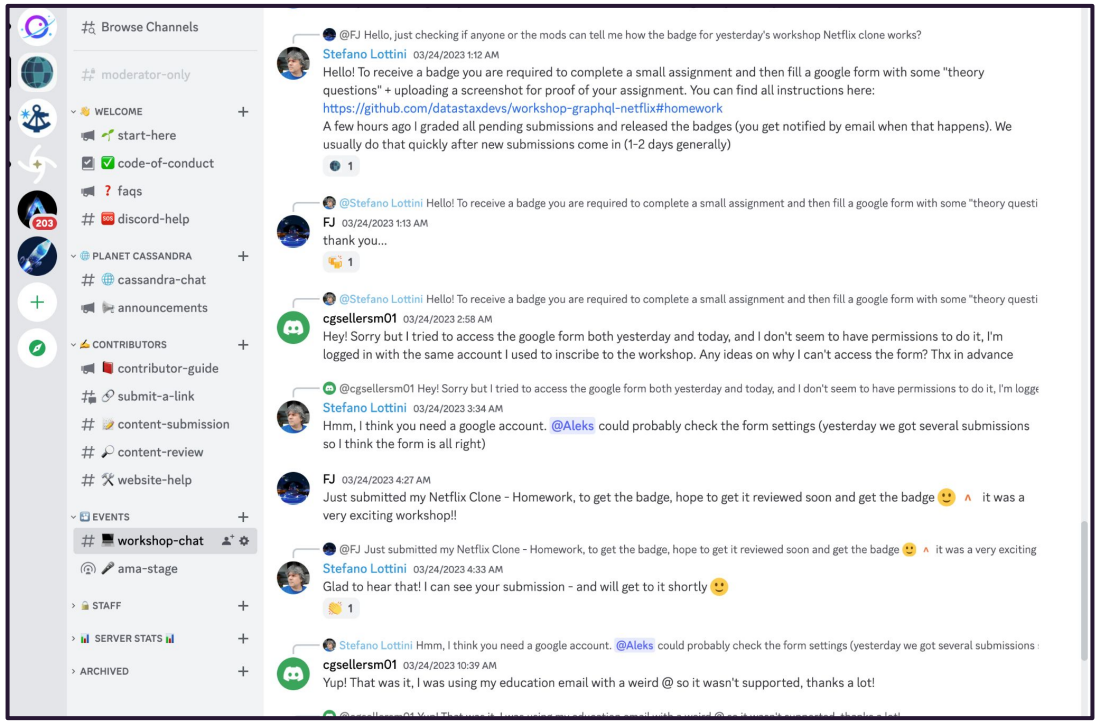
+

CDC (Change Data Capture)

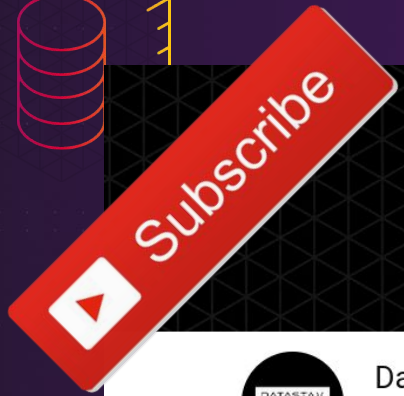
Preemptively block spammers
Optimize billing plans for users
Flag abuse of service
…

Planet Cassandra Discord server

!discord



dtsx.io/discord



DataStax Developers

@DataStaxDevs

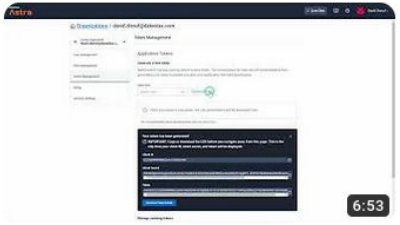
29.9K subscribers

HOME VIDEOS SHORTS LIVE PLAYLISTS COMMUNITY CHANNELS ABOUT

Recently uploaded Popular

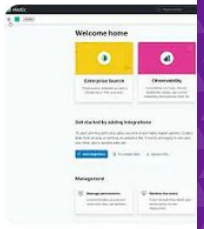
The Legend of DataStax - Cassandra Summit Training Day!

Cassandra Summit Training Day - Sunday March 12th



CDC for Astra DB Demo: Sink to Astra DB

92 views • 2 weeks ago



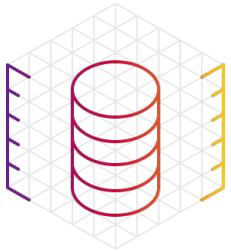
CDC for Astra DB Demo: ElasticSearch

Badges

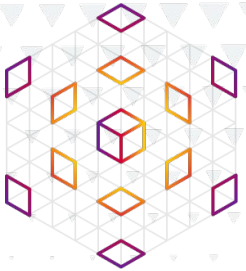

INTRODUCTION TO
cassandra
WORKSHOP
DataStax Developers
cassandra
Introduction to
NoSQL
WORKSHOP
DataStax Developers
cassandra
DATA MODELING
WORKSHOP
DataStax Developers
cassandra
SQL to NoSQL
WORKSHOP
DataStax Developers
K8SSANDRA
WORKSHOP
DataStax Developers
cassandra
Spring Data +
Cassandra
WORKSHOP
DataStax Developers
Vue.JS
WORKSHOP
DataStax Developers
ReactNative
WORKSHOP
DataStax Developers
GraphQL
WORKSHOP
DataStax Developers
Building a
ToDO App
WORKSHOP
DataStax Developers
TikTok Clone
WORKSHOP
DataStax Developers
Netflix Clone
WORKSHOP
DataStax Developers
Event
Streaming
WORKSHOP
DataStax Developers
Reactive
Spring
WORKSHOP
DataStax Developers
AI as an
API
WORKSHOP
DataStax Developers
Build APIs +
Microservices
WORKSHOP
DataStax Developers
Event Streaming
Learning Series
DataStax Developers
cassandra
Quarkus +
Cassandra
WORKSHOP
DataStax Developers

Badges


INTRODUCTION TO
cassandra
WORKSHOP
DataStax Developers
cassandra
Introduction to
NoSQL
WORKSHOP
DataStax Developers
cassandra
DATA MODELING
WORKSHOP
DataStax Developers
cassandra
SQL to NoSQL
WORKSHOP
DataStax Developers
K8SSANDRA
WORKSHOP
DataStax Developers
cassandra
Spring Data +
Cassandra
WORKSHOP
DataStax Developers
Vue.JS
WORKSHOP
DataStax Developers
ReactNative
WORKSHOP
DataStax Developers
GraphQL
WORKSHOP
DataStax Developers
Building a
ToDO App
WORKSHOP
DataStax Developers
TikTok Clone
WORKSHOP
DataStax Developers
Netflix Clone
WORKSHOP
DataStax Developers
Event
Streaming
WORKSHOP
DataStax Developers
Reactive
Spring
WORKSHOP
DataStax Developers
AI as an
API
WORKSHOP
DataStax Developers
Build APIs +
Microservices
WORKSHOP
DataStax Developers
Event Streaming
Learning Series
DataStax Developers
cassandra
Quarkus +
Cassandra
WORKSHOP
DataStax Developers



❯ Stay in touch!

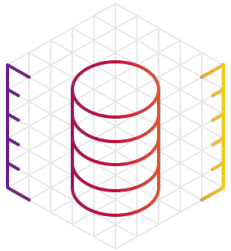


Discord: dtsx.io/discord

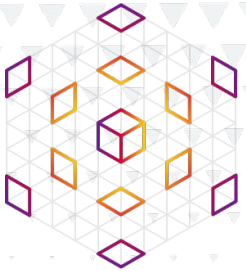
Academy: academy.datastax.com

Workshops: datastax.com/workshops

YouTube: @DataStax Developers



❯ You are not alone in your journey

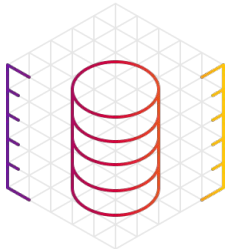


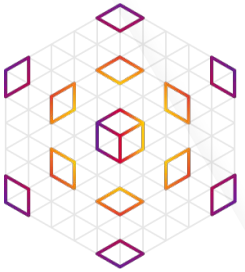
Discord: **dtsx.io/discord**

StackOverflow(*): **stackoverflow.com/questions/tagged/cassandra**

DBA Stack Exchange(*): **dba.stackexchange.com/questions/tagged/cassandra**

(*) For best results, follow the cassandra tag





Thank You