



Crudit

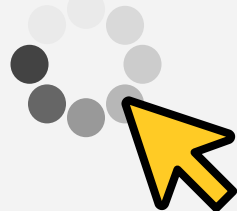
Micro-backend
for fast
shipping
applications





Hi! We are
Daniele^2!

We are software developers,
working in Sintra Consulting,
making e-commerce solutions
and integrations



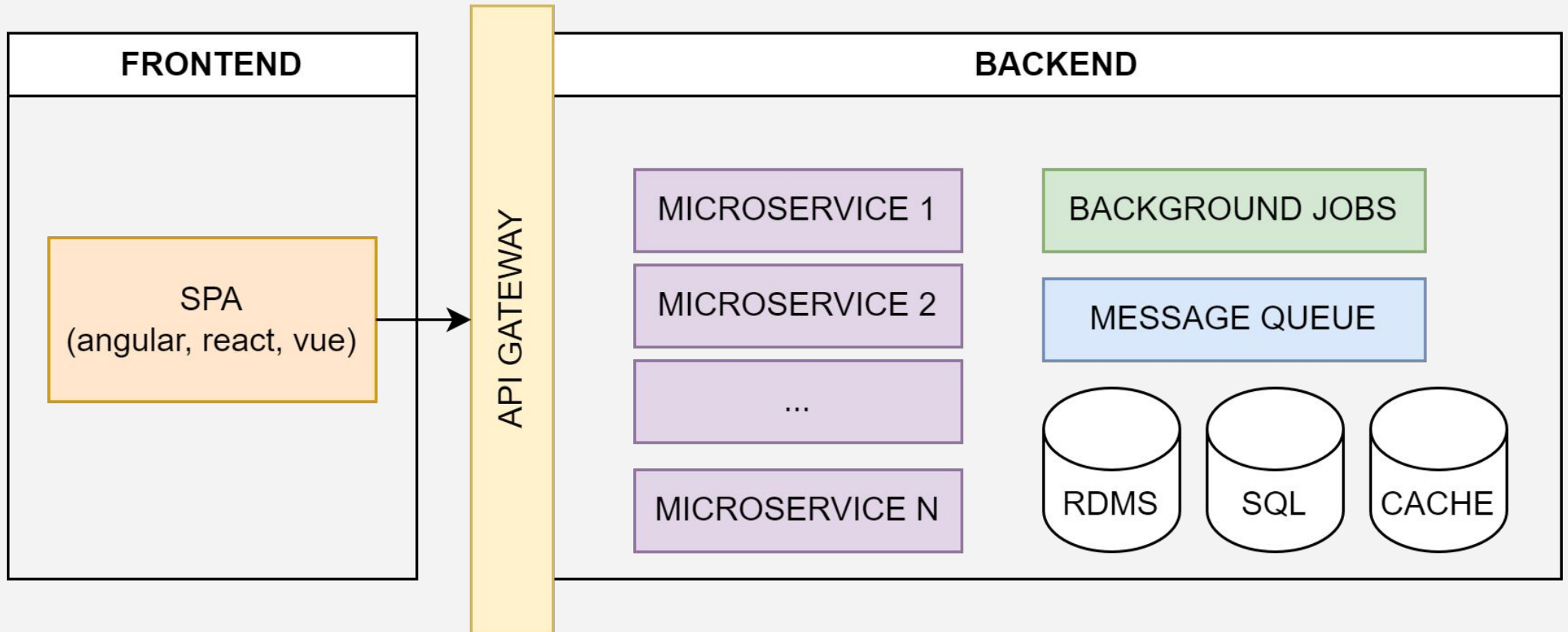


Developing an application in 2023





Modern app architecture...





That's good because

- BEST OF BREED
- WORK PARALLELIZATION ACROSS
MULTIPLE TEAMS
- BETTER PERFORMANCE
- SEPARATION OF CONCERNS



... but brings some issues

- YOU NEED MULTIPLE TEAMS OR COMPETENCES
- A SINGLE PERSON CANNOT DO THE JOB
- IN CASE OF PROBLEMS, WHERE TO LOOK?



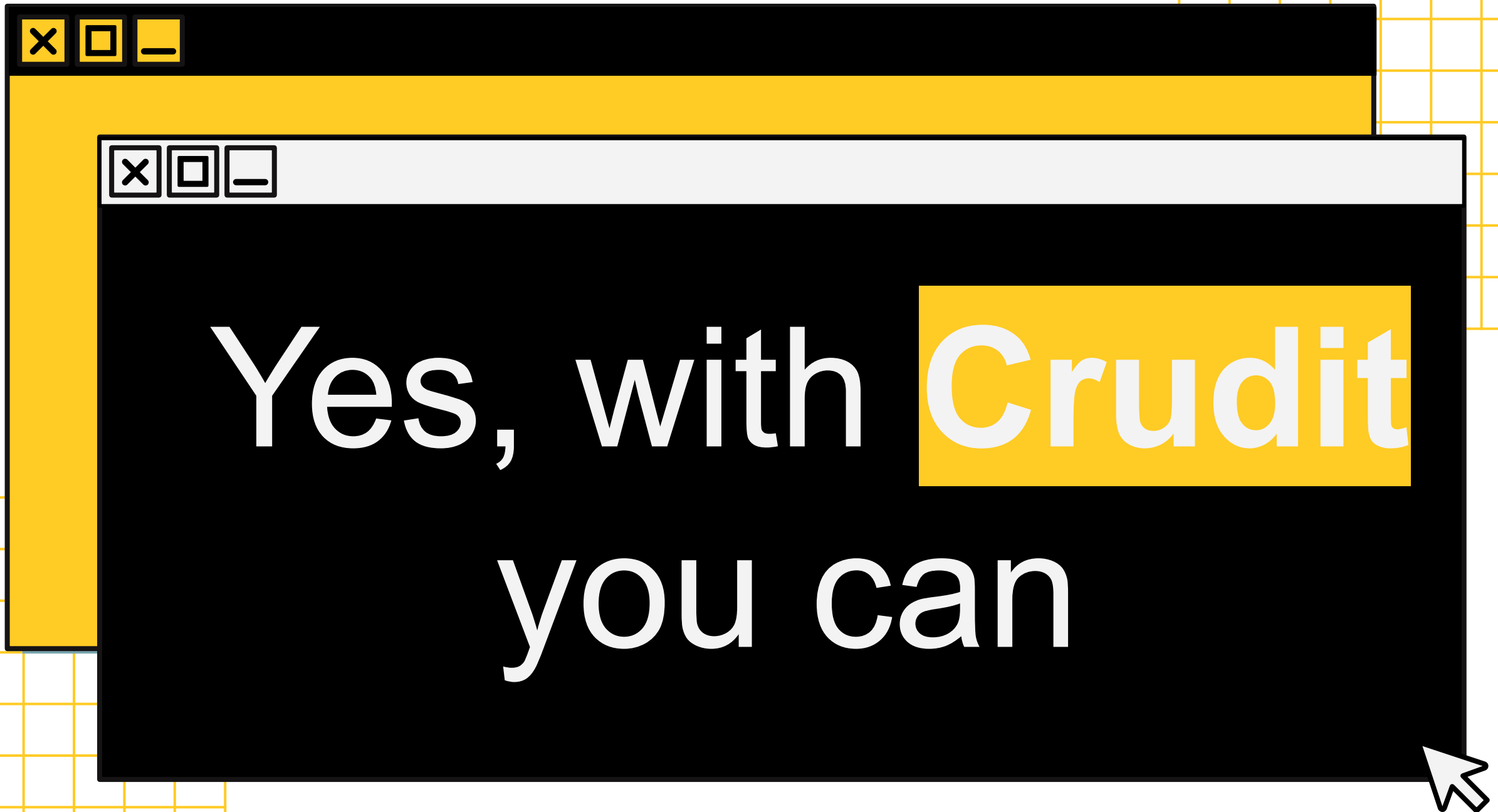
Who is impacted more?

- LOW-BUDGET PROJECTS
- SMALL COMPANIES
- STARTUP
- NO MORE QUICK AND DIRTY APPLICATIONS

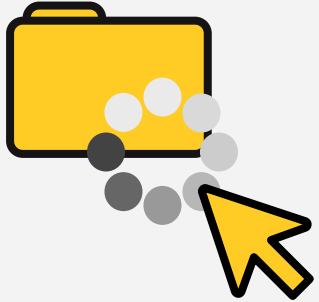


**It's still possible to
develop a web
application with ease?**





Yes, with **Crudit**
you can

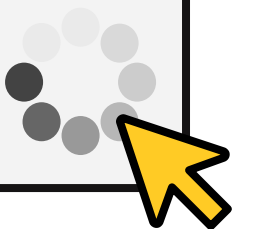


WHAT IS **CRUDIT** ?

CRUDIT IS THE **LOW-CODE**
LIGHTWEIGHT JAVASCRIPT LIBRARY THAT
ENABLES DEVELOPERS TO CREATE
APPLICATION WITHOUT ANY BACKEND OR
SYSADMIN KNOWLEDGE .

What is Crudit for?

- Fast shipping Multi-tenant Node.js applications
- Mutate existing MongoDB data fast and easily
- Form retrieval with server-side validations

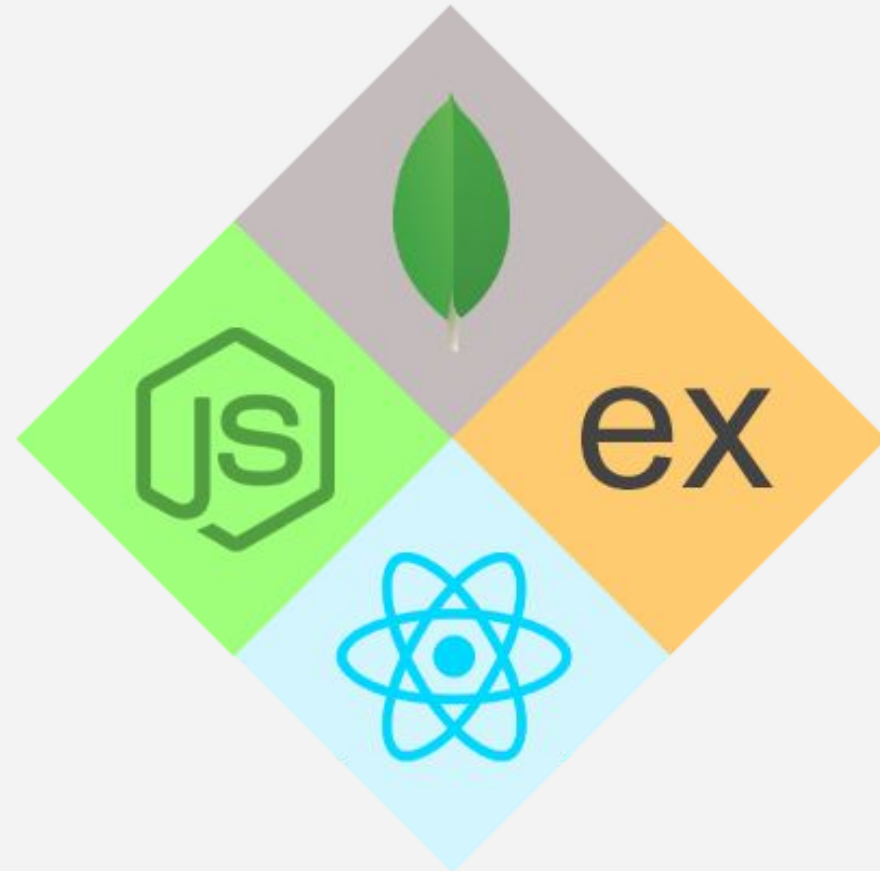




But How?

Crudit can be deployed in a Edge Function, connected to a Mongo Atlas Database, and **forget about it**

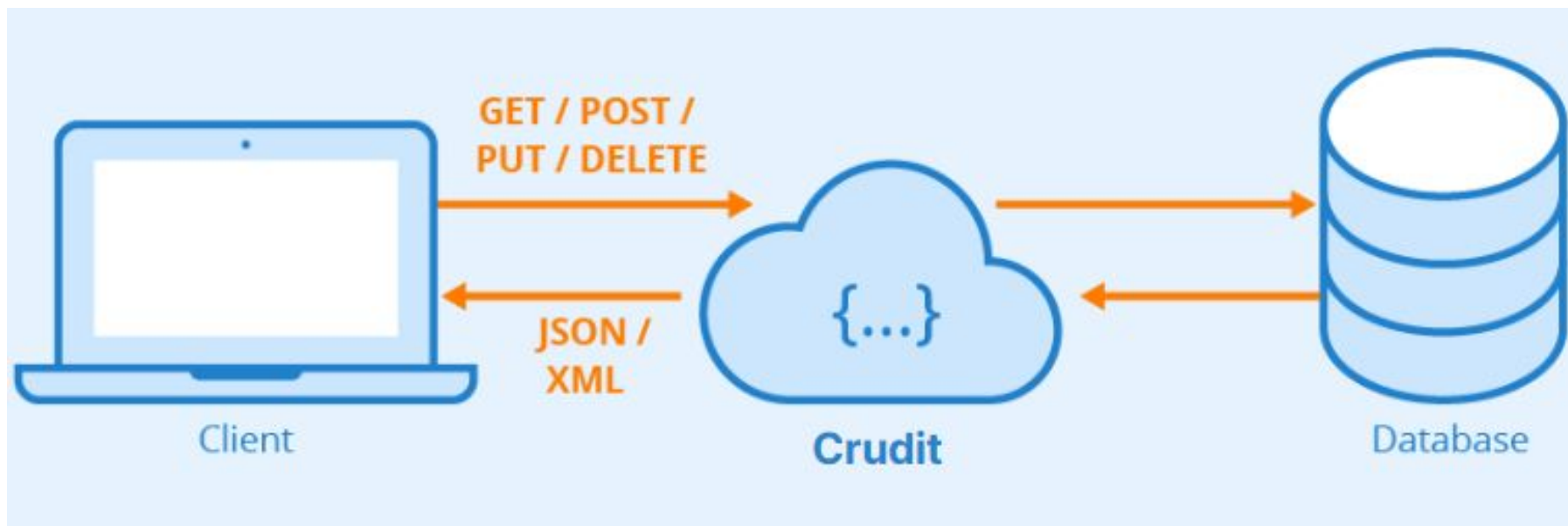
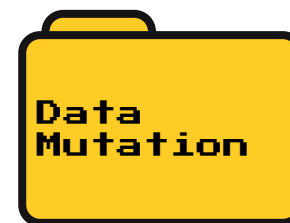
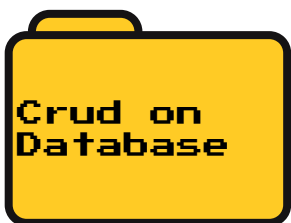
You get a complete **REST Compliant API** in a few Javascript Lines!





What can Crudit do?

Crudit is a Javascript Library, that can perform many operations and cover many use cases





Features!

Multi Tenant RESTful API For CRUD Operations

Integrate with a
MongoDB Atlas almost
without configurations

```
const app = express();
app.use(express.json());

// configure settings
crudy.config(
  function(config){
    config.database.url=process.env.CRUDIT_DBURL;
  }
);

app.all('/api/handler',
  async (request, response) => {
    return await crudy.run(request,response);
  }
);
```



The calls follow the REST protocol:

Create with POST

Read with GET

Uppdate with PUT

Delete with DELETE

I

Try with `npm i crudit`



A Database Interface

Crudit expose a multi tenant RESTful API.

Each user writes on its own MongoDB database, and the path decides the collection the document are written on

Custom Endpoint for Dedicated API Routes

Create routes with
custom handlers.

Use this method to
create a login and
register method, and
let users authenticate

```
crudy.request(  
  "publish",  
  "post",  
  true,  
  async function({loggedRequest}){  
    return handleRequest(loggedRequest);  
  }  
);
```

Server Side Validation

When creating an entity, you can define a name or a Regex for a Database and/or Collection.

Then create a validation object using `validator.js` syntax, and your entity will be validated before been written on database

```
crudy.configEntity('validateBoth', {
  db: 'testValidationBothDB',
  collection: 'testCollectionBoth',
  validation: {
    age: 'min:18|required',
    email: 'email|required'
  }
});
```



Hooks on events

Create custom function on events.

The function can be triggered always, but also only connected to a database or collection name

The event handler can modify the data and has access to the user, database and collection.



```
crudy.hook('log',
  events.beforeSave,
  'logs',
  'before_save',
  async function({params}){
    let log = {
      db: params.db,
      collection: params.collection,
      user: params.user,
      date: new Date().toISOString()
    }

    database.insert('logs', 'before_save', log);
  }
);
```

Mutations

Register a mutation than execute it.

A mutation can be registered with a REGEX for Collection name and, and can be executed on a single database or in every database

```
crudy.mutation(
  'mutationOnEverything',
  '*',
  async (databaseName, prevExec)=>{
    return await mutateData(databaseName);
  }
);

mutations.applySingle('thisDatabase', 'thisMutation');

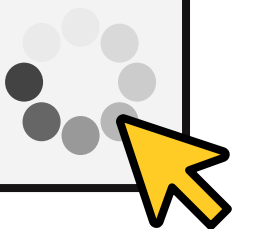
mutations.applyOne('thisDatabase', true);

mutations.applyAll();
```



**And What About DevOps?
Don't Need it Either!**

Crudit can be used in every Node.js compatible runtime, from Lambda To Cloud Functions, and even Edge Functions!





Who is For?

- Modern FullStack Javascript Projects that needs a Database connection (Frontend Driven App)
- Node.js & MongoDB Applications
- Any Projects that don't need complex infrastructures and are written mostly in Javascript
- Explorative e/or Amateur Projects, reducing costs to zero thanks to the free tiers




Who needs something else

- Multimedia-Focused Applications
- Strong Relational Data
- High Business Logic Complexity
- Separation of Business Logic from Data
- Needs of Graphical Interface for Backend





What we want to add?

- Configuration Import from YAML and JSON
- Support for media assets, without creating a custom endpoint
- Suggestions for future implementations
- . . .
- Github Stars 

Contributions and Pull requests are open!



**Thank
you!**

