Node & Loopback Services

Hybrid Integration Enablement

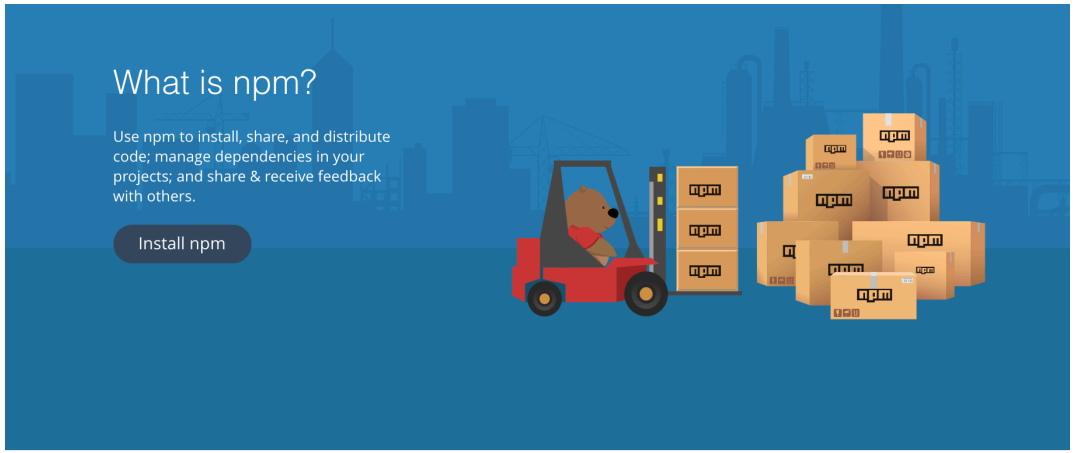
October 2017

LOOPBACK Microservices

Use Case: What are the benefits loopback will provide to build node.js microservices.

Why Node.JS ?

- Node is Fast
  - Non-blocking I/O
  - Built on V8 Compiler
- One language for server and client side
- Solid Standard (ECMA Script)
- Vibrant Ecosystem (NPM)
  - Over 475,000 packages



What can you make with 475,000 building blocks?

The npm registry hosts almost half a million packages of free, reusable code — the largest software registry in the world.

Find

Libraries like JQuery, Bootstrap, React, and Angular, and components from frameworks such as Ember.

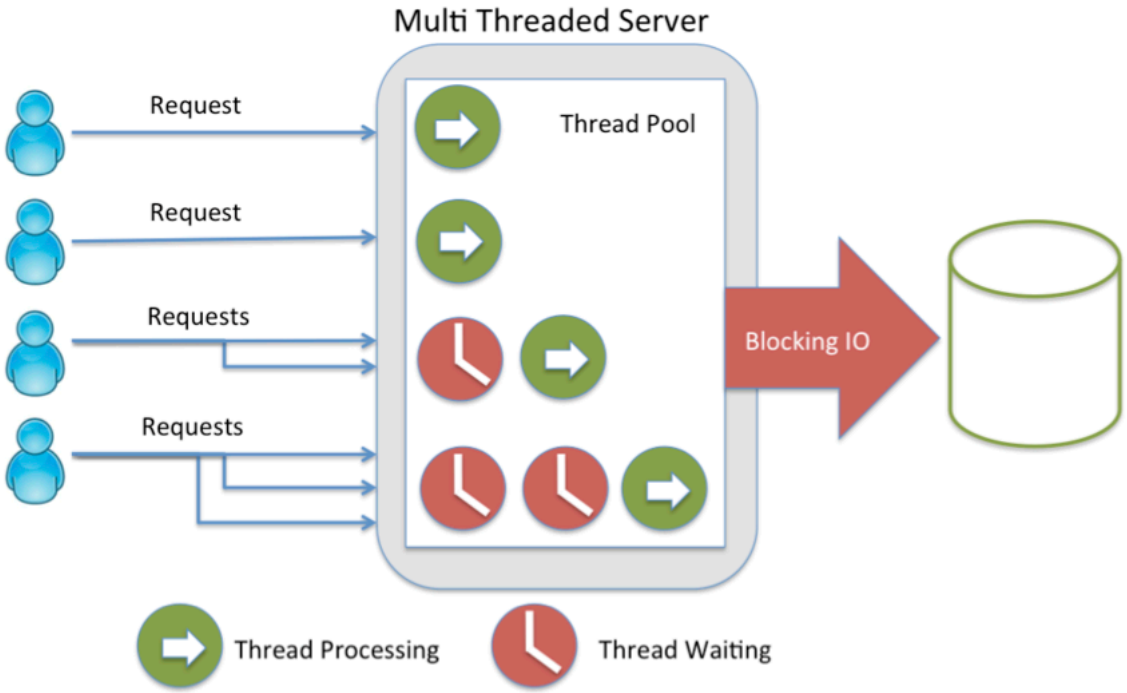
Discover

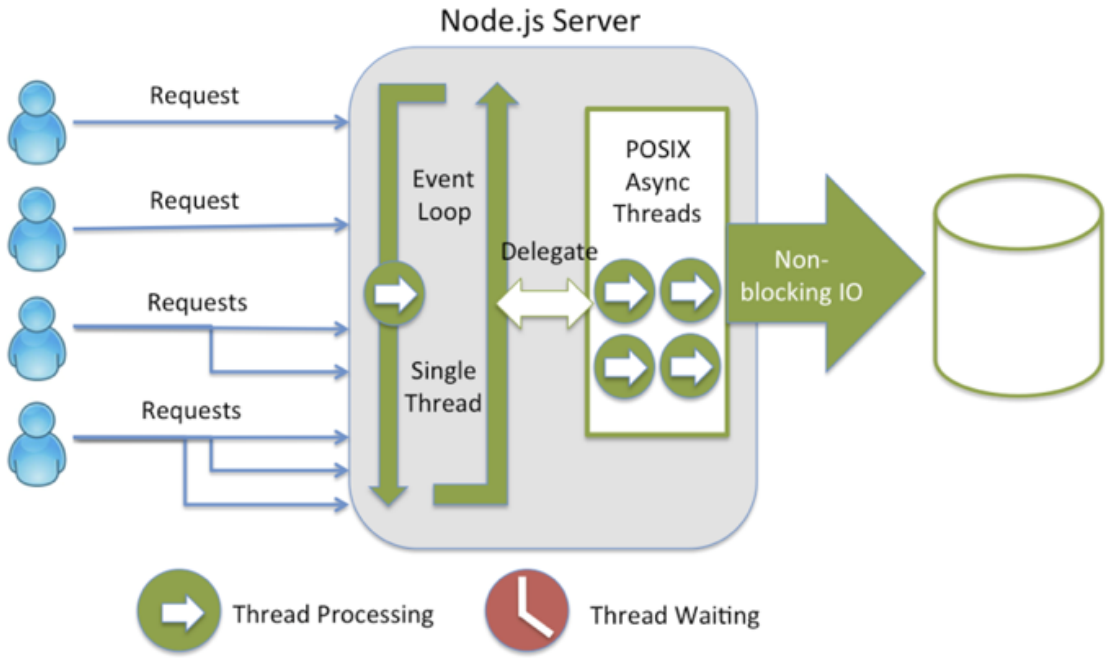
Packages for mobile, IoT, front end, back end, robotics... everything you need to start building amazing things.

Build

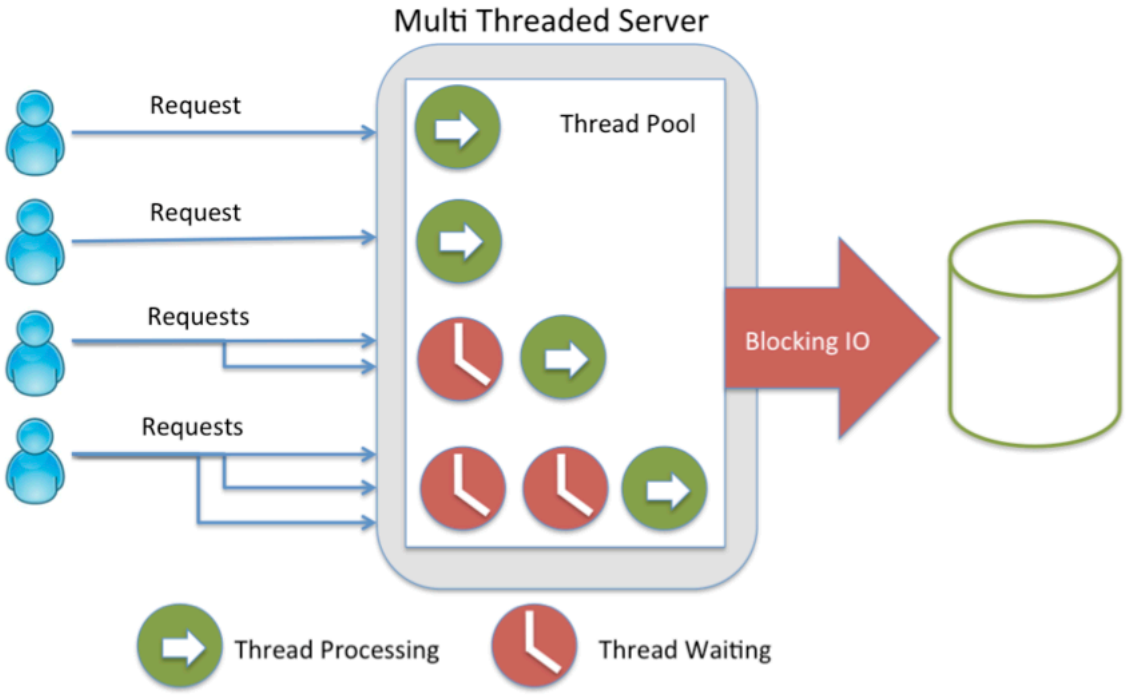
Assemble packages like building blocks to quickly develop awesome new projects.

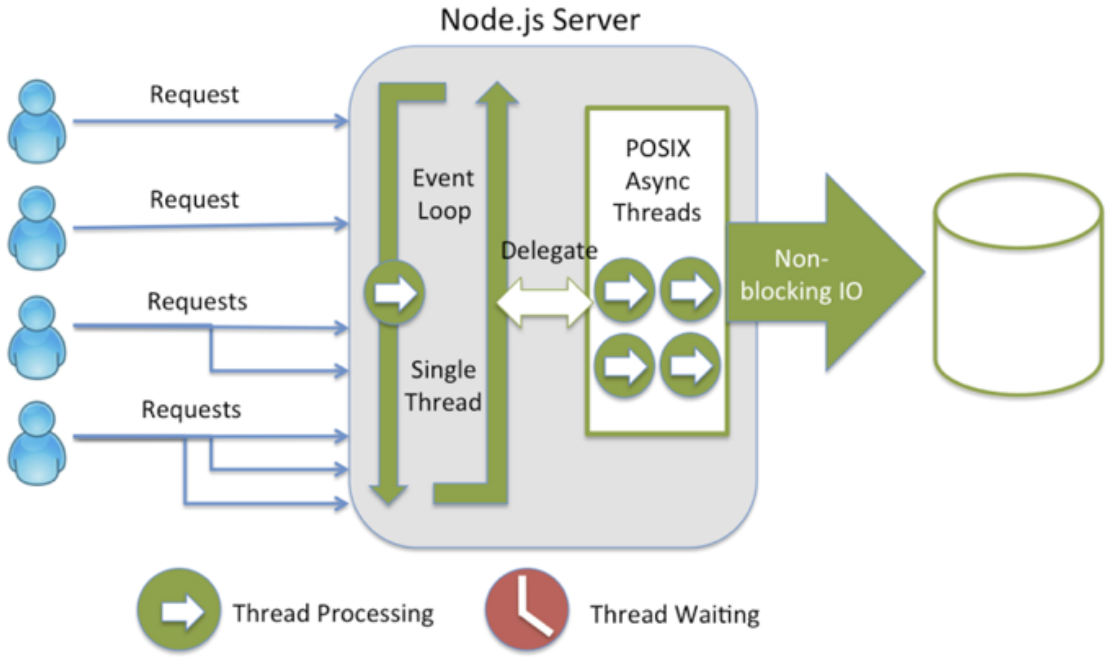
What do you mean Non Blocking?





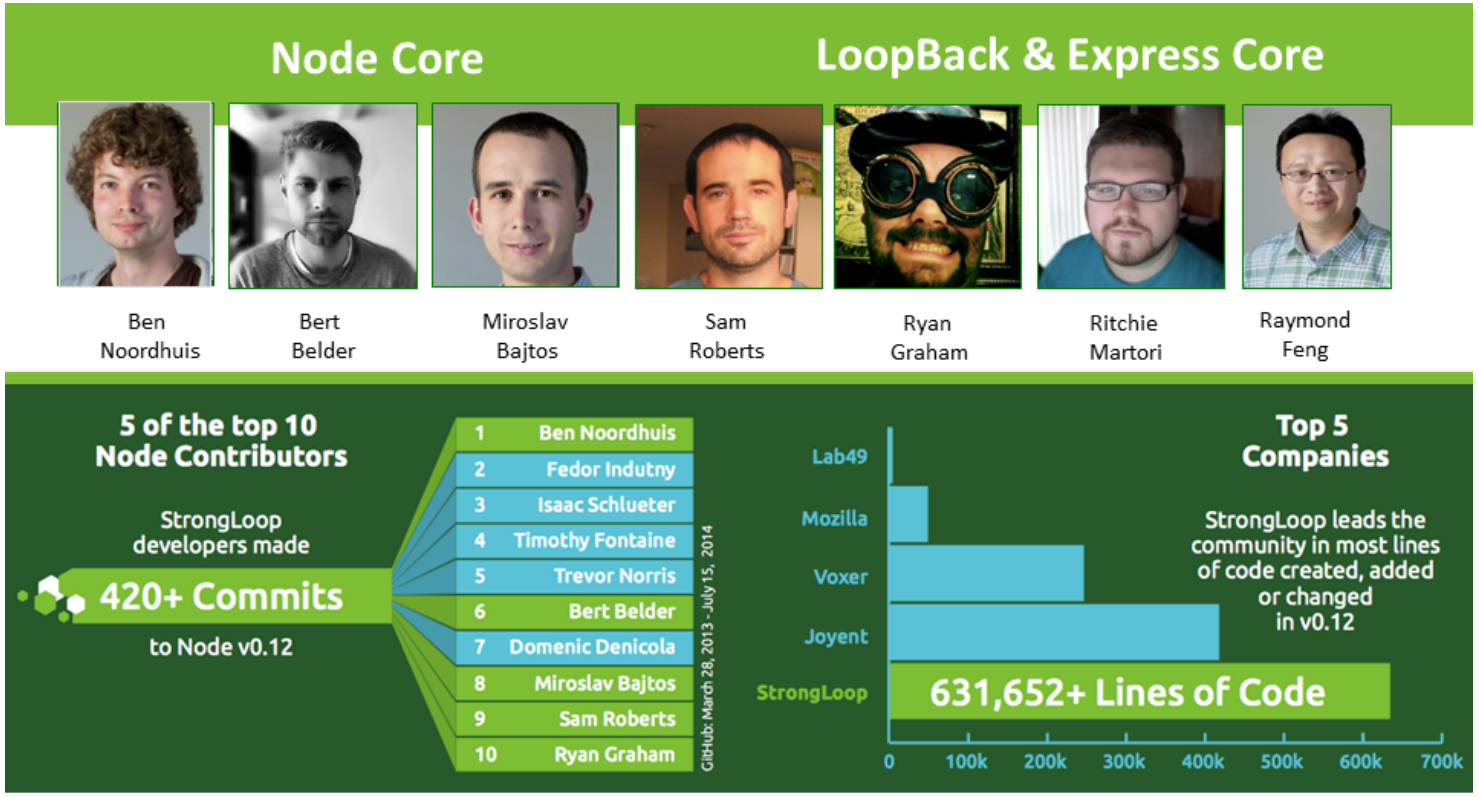
What do you mean Non Blocking?

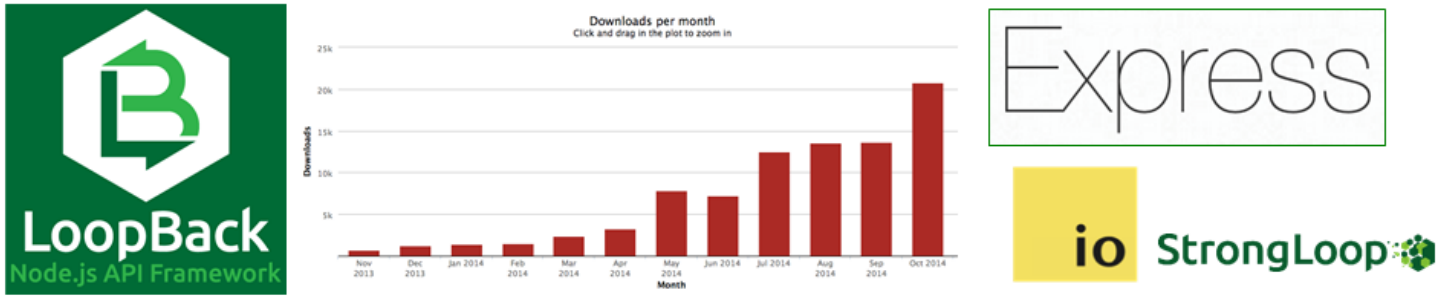




What is StrongLoop?

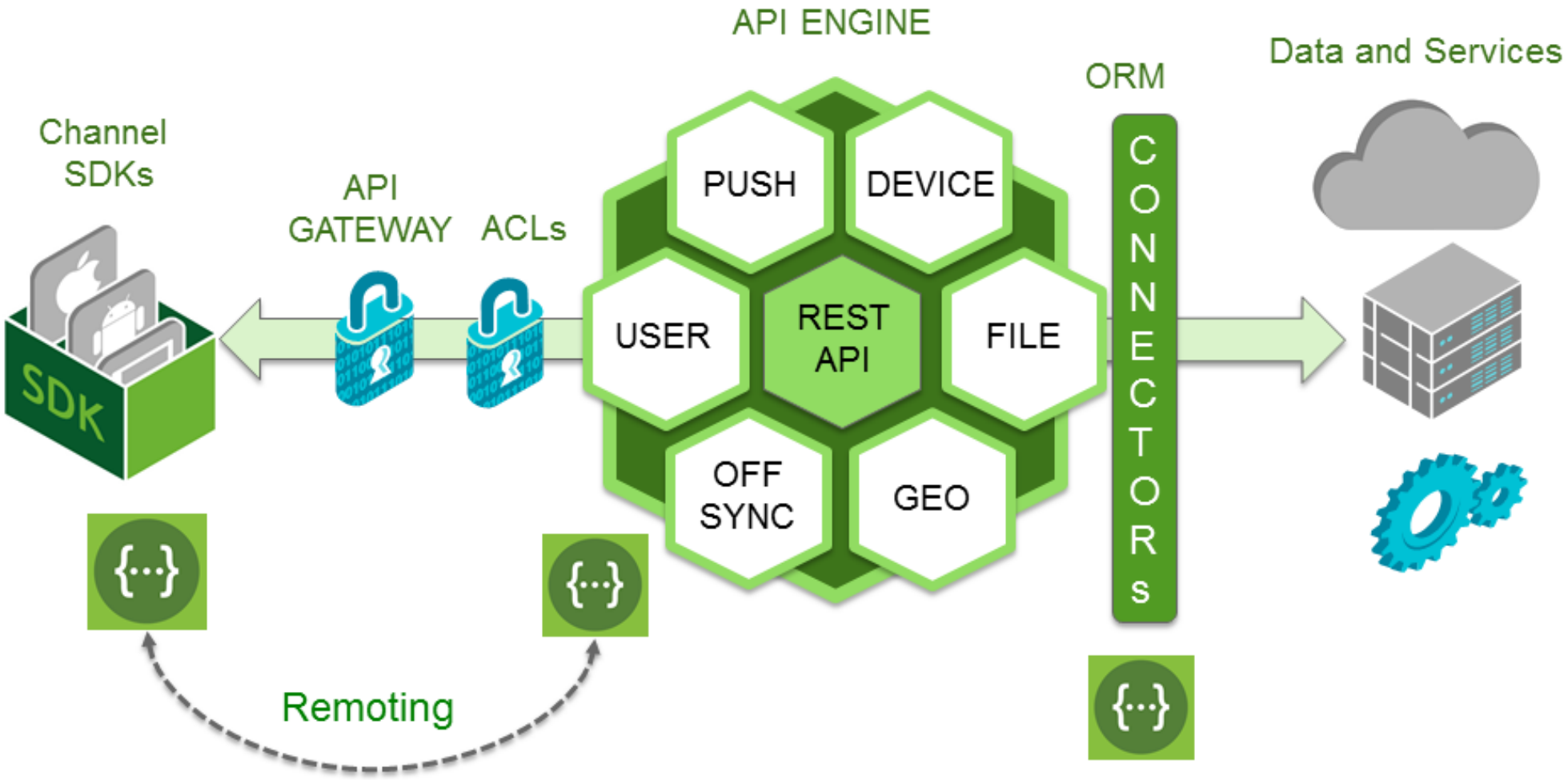
- IBM Acquired Strongloop in 2015
  - Worked hard to bring Node.JS to the enterprise.
  - Hired a few people involved with core contributers to Node.JS
- Completed the API management Product offering to provide Create Functionality.
- IBM Continues to contribute to Node.JS, now with the Linux Foundation and community.





What is Loopback?

- Loopback **is a highly-extensible, open-source Node.js framework**



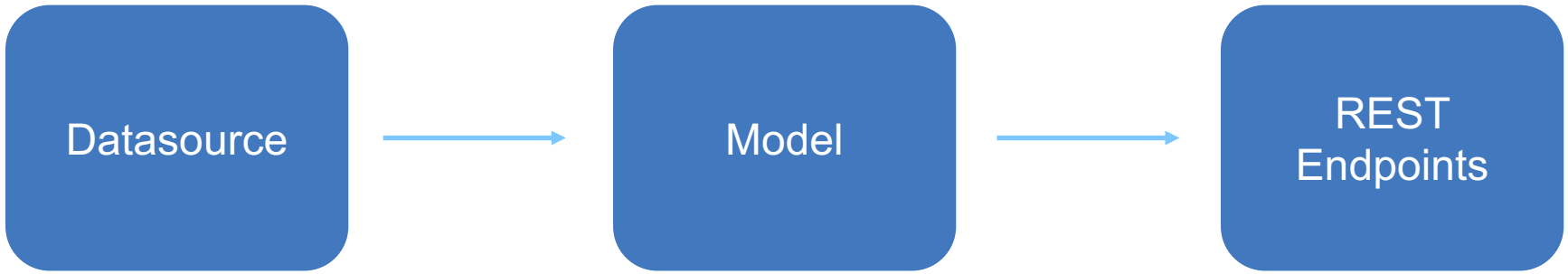
Top Features of Loopback

- Model Driven API Development
- Dynamic REST API endpoint Generation
- Connect to any datasource (SQL, NoSQL, REST, SOAP)
- Rich Model Relations
- Access Controls (built in token authentication)

Data Sources Matrix

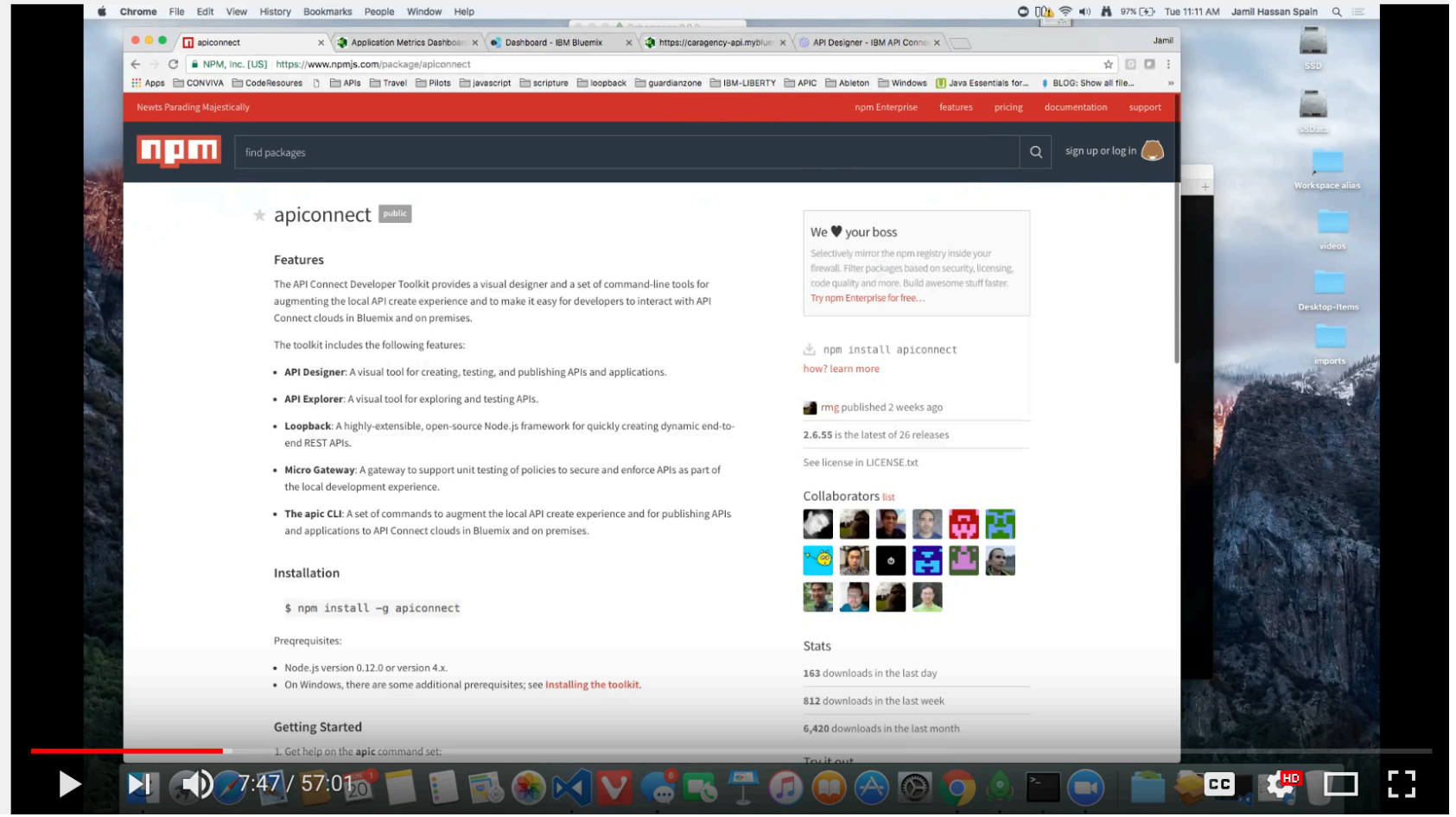
Persisted Models = System APIs

- Models generated against these datasources produce System APIs
- These endpoints are auto generated to create CRUD
- Any activity on these endpoints directly communicates to Systems of Record



Webinar Resource

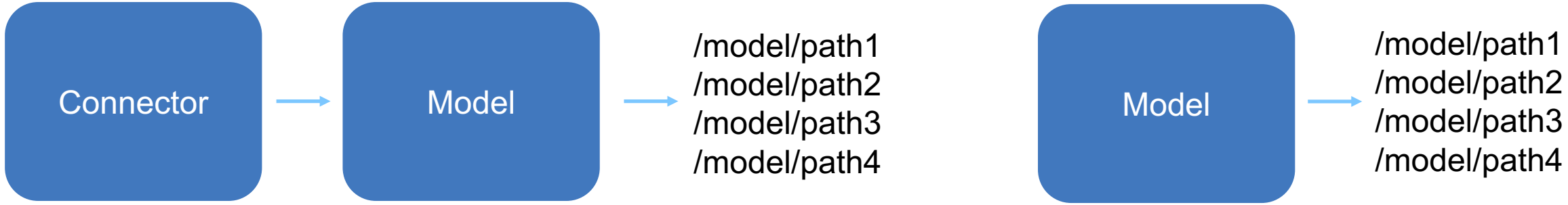
- Webinar with Digital Team overview of UI



https://www.youtube.com/watch?v=RGuG8_l4gtE

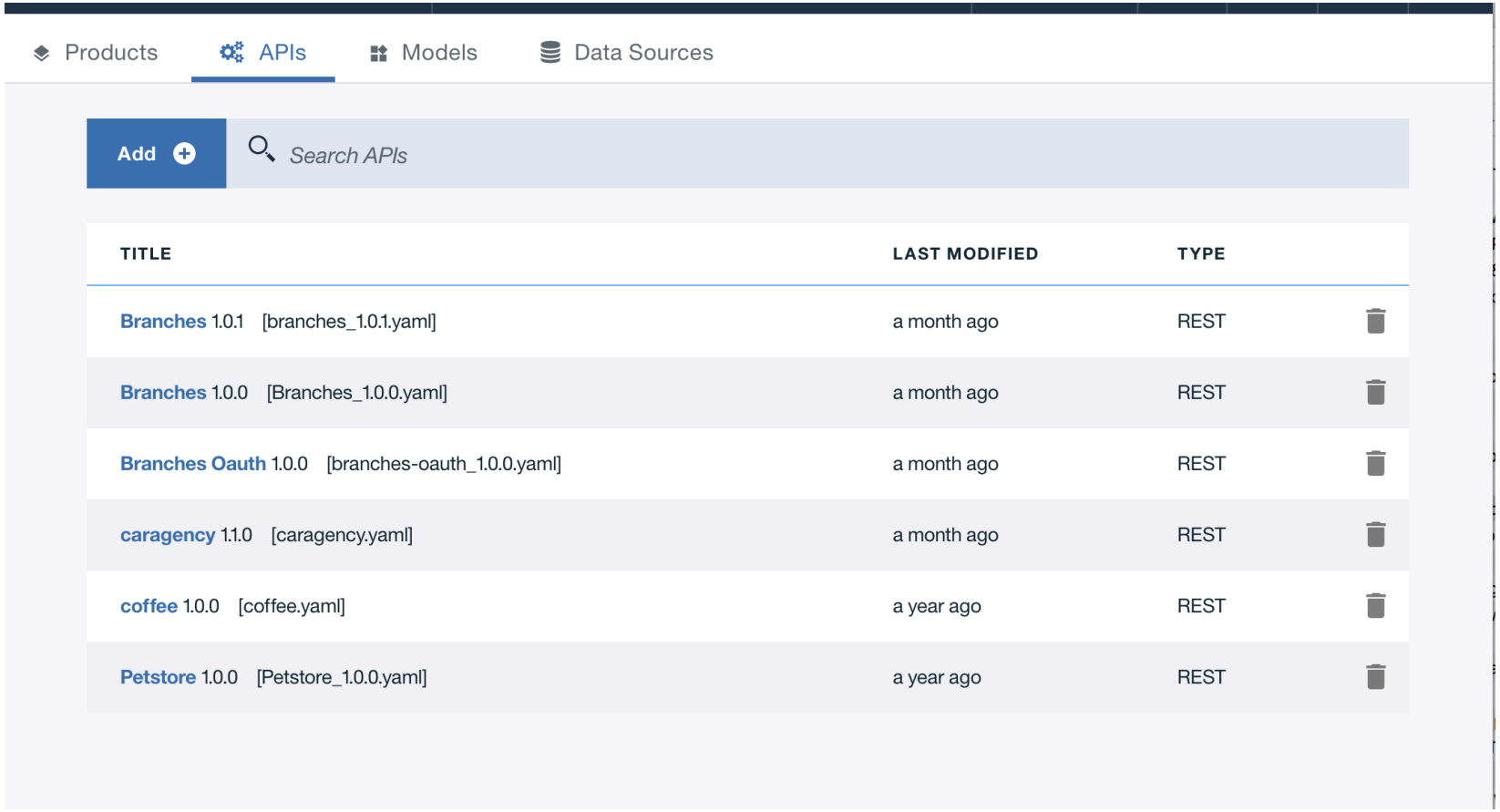
Models = Interaction API

- Models **without** persistence can interact with REST (Other REST APIs), 3rd Party (Twilio), or Built In (Email)
- Models **without** persistence can generate custom API paths through ”Remote Methods”



API Designer enables Loopback

- Loopback Project creation is available via CLI and API Designer
- API Designer can create System API models
- Interaction API Models must be implemented via coding however, but can be created via API Designer



Syncing API Designer with Project

- Any Changes you make inside the API Designer will write code to the Loopback Project.
- However, any direct code changes may not reflect in the API Designer.
- Use the command below to sync changes (and get things documented in Swagger).

```
Jamils-MacBook-Pro:caragency jamilspain$ apic loopback:refresh

Updating swagger and product definitions
Created /Users/jamilspain/Z-IBM-Workspace/Demo - API Connect/caragency/definitions/caragency.yaml swagger description
```

Models are connected

- In code, all models have a logic and definition file.
  - Model.json (definition file)
  - Model.js (business logic / interaction)
- Any Model can communicate with any other model for incredible mashups

```
module.exports = function(Car) {

        Car.app.models.Inventory.find({}, function(err, cars){
            cb( null, cars );
        });

};
```

Models can connect to REST services

- Google Maps Datasource to their REST API

```
"geoRest": {
  "name": "geoRest",
  "connector": "rest",
  "operations": [{
    "template": {
      "method": "GET",
      "url": "http://maps.googleapis.com/maps/api/geocode/{format=jso
      "headers": {
        "accepts": "application/json",
        "content-type": "application/json"
      },
      "query": {
        "address": "{street},{city},{zipcode}",
        "sensor": "{sensor=false}"
      },
      "responsePath": "$.results[0].geometry.location"
    },
    "functions": {
      "geocode": ["street", "city", "zipcode"]
    }
```

Models can implement Datasource REST Connections

```
"Widget": {
    "dataSource": "geoRest",
    "public": true
}
```

- Exposing all functions as REST Endpoints

- Other models can use this model via code

```
Widget.geocode('107 S B St', 'San Mateo', '94401', function(res) {
    // ... handle the response
});
```

Loopback Summary

- All the following can exist in one project
  - Persisted Models to different Databases
  - REST Connectors to 3rd Party Services
  - Consuming other services (other APIs)

- Models can allow Programmatic Mashups between all Models

- Together, these features make a very powerful webservice creation tool.

Loopback API Questions

- Open Conversation / Questions?

Up Next: Publishing Loopback Services

Complete Lab 4