galvanize

Module 2

Week 7 | Day 1 | **Intro to Software Design**

Major Themes

Week 1

1. Modeling a business problem as object-oriented software.
2. Breaking monolithic apps into **microservices**.
3. Sending data across HTTP in the form of a JSON object.

HACK REACTOR
BY GALVANIZE

Goals for the Week

Monday

* Modeling a hotel management system

* **Domain-Driven Design**: Entities & Value Objects

* JSON

* **Lab:** Intro to Conference GO

Tuesday

* Extending our hotel model

* **Domain-Driven Design**: Aggregates

* Build a JSON Library

* **Lab:** Build your own JSON library

Wednesday

* RESTful API's

* **Domain-Driven Design**: Bounded Contexts

* **Lab:** RESTfulize your app!

Thursday

* HTTP

* More REST

* **Domain-Driven Design**: Anti-Corruption Layers

* **Lab:** Integrate 3rd Party data

Friday: Docker!

GALVANIZE

GALVANIZE | HACK REACTOR BY GALVANIZE

Today's Agenda

Intro to Software Design & Architecture

Domain-Driven Design

1. **Activity:** Designing a Hotel Reservation System
2. **Theory:** Entities & Value objects
3. **Code-Along:** JSON

AFTERNOON
Technical Practice:

JSON Responses/Requests with Django

slido

Join at slido.com
#056358

GALVANIZE

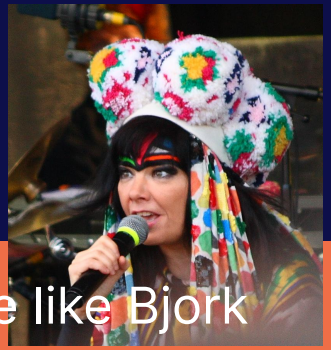
slido

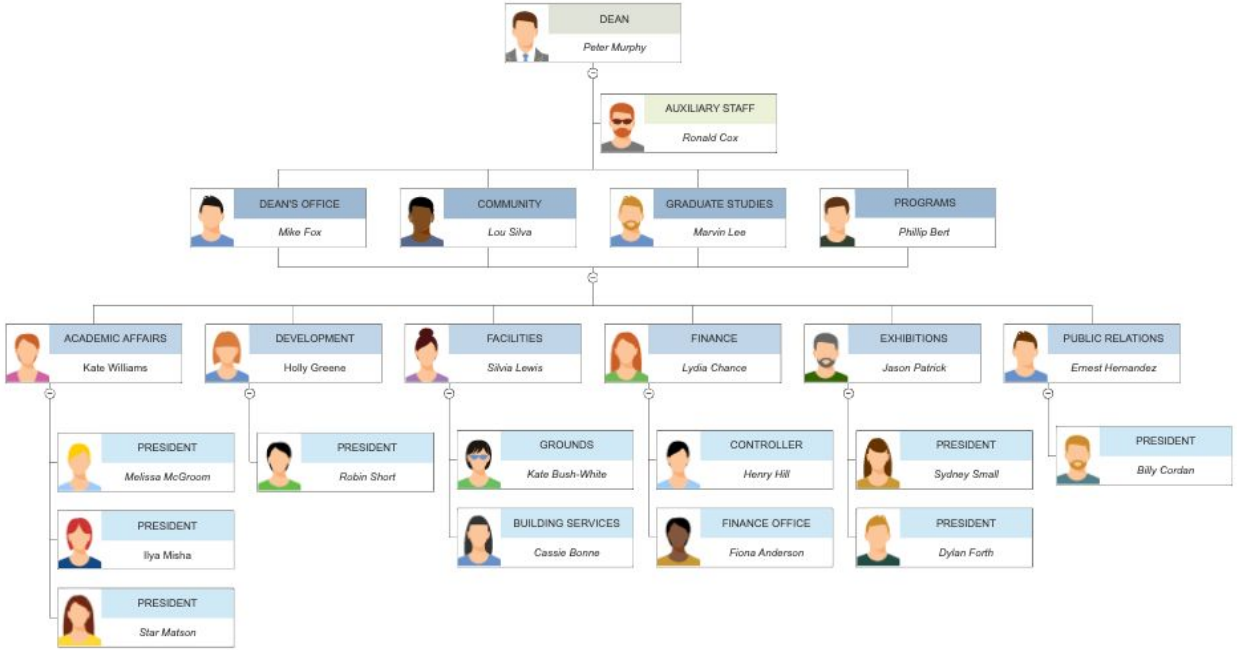
Tell me what you know already about software design.

What are some important considerations in designing good software for humans?

GALVANIZE

Software →
HUMAN BEHAVIOR



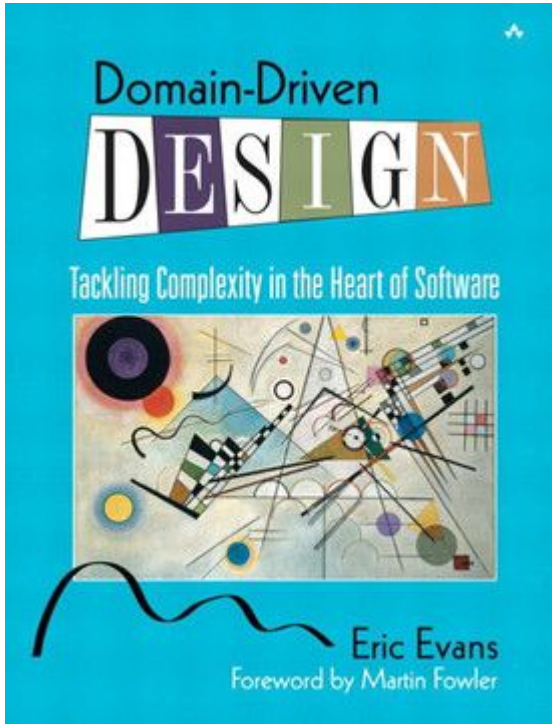


1. Domain-Driven Design

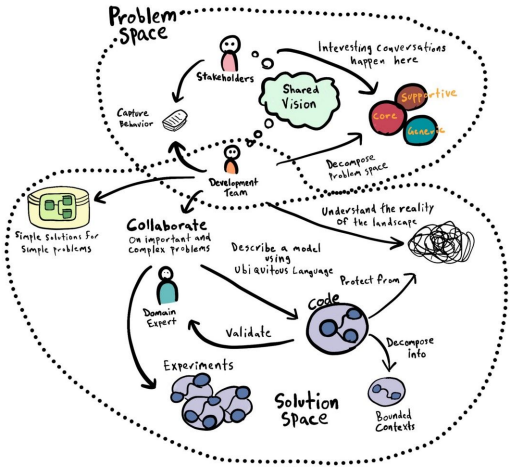
Applying object-oriented programming to modelling human behaviors and business problems.

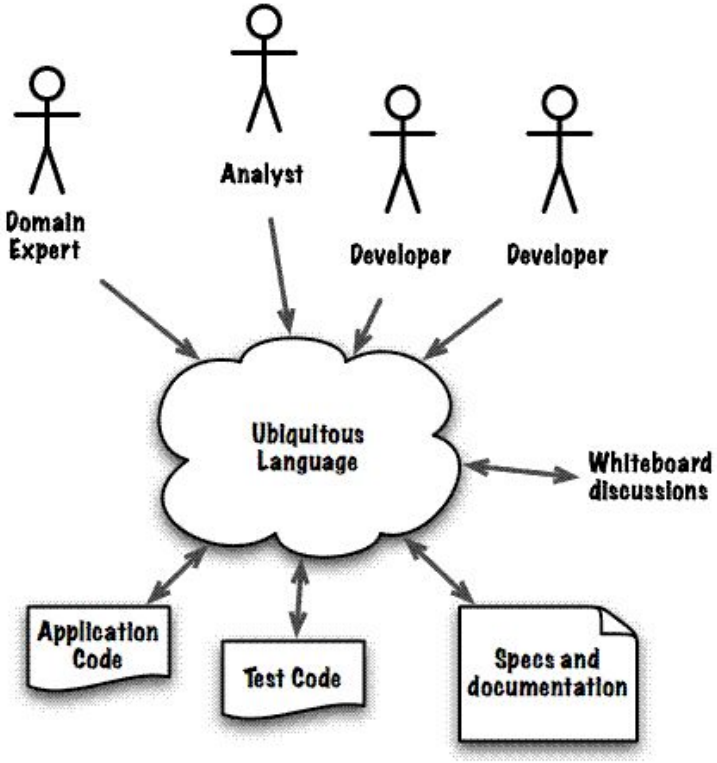
Domain-Driven Design (DDD)

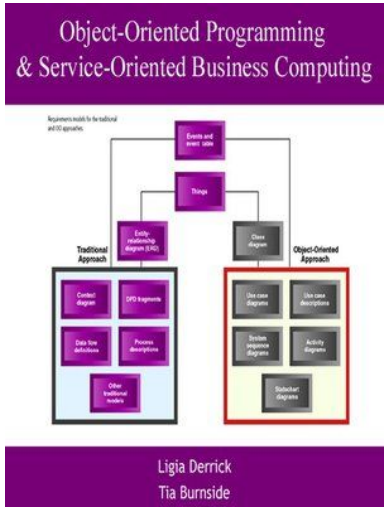
- A software design approach that focuses on modeling software to match a particular business domain's object names and methods.
  - **Domain** - the subject area to which the user models a software program
  - **Ubiquitous language** - the common language shared by domain experts, users, and developers
    - Eric Evans' book in 2003:



Object-Oriented ALL THE THINGS!







HACK REACTOR
BY GALVANIZE

Example: Hotel Reservation Software

Departments / People:

- Finance department
- Cleaning staff
- Concierge
- Management

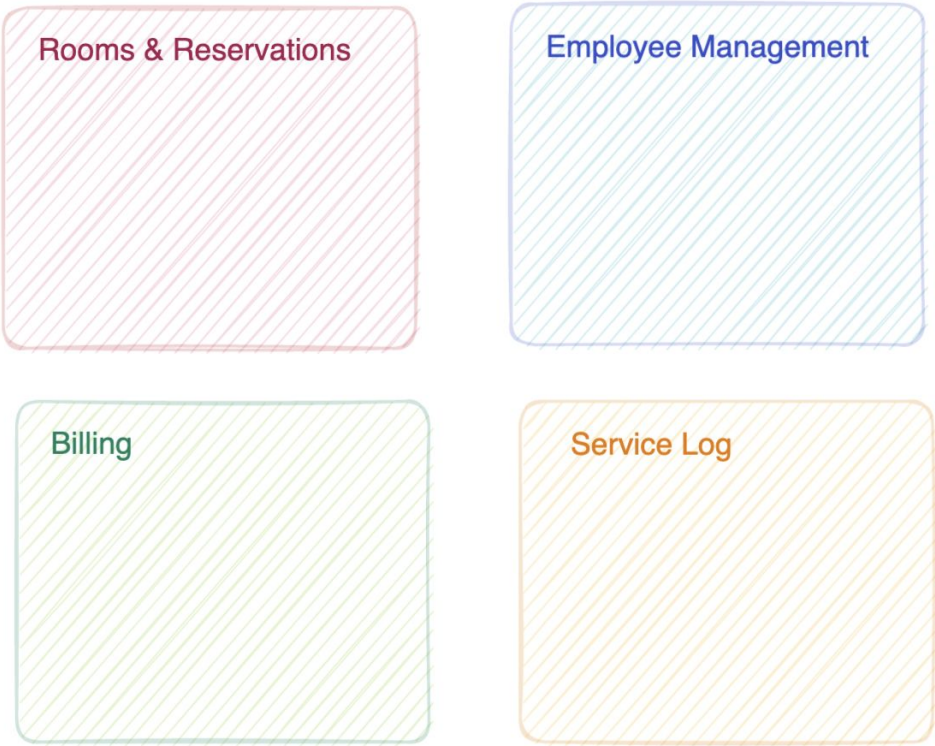
Example: Hotel Reservation Software

Business Problems:

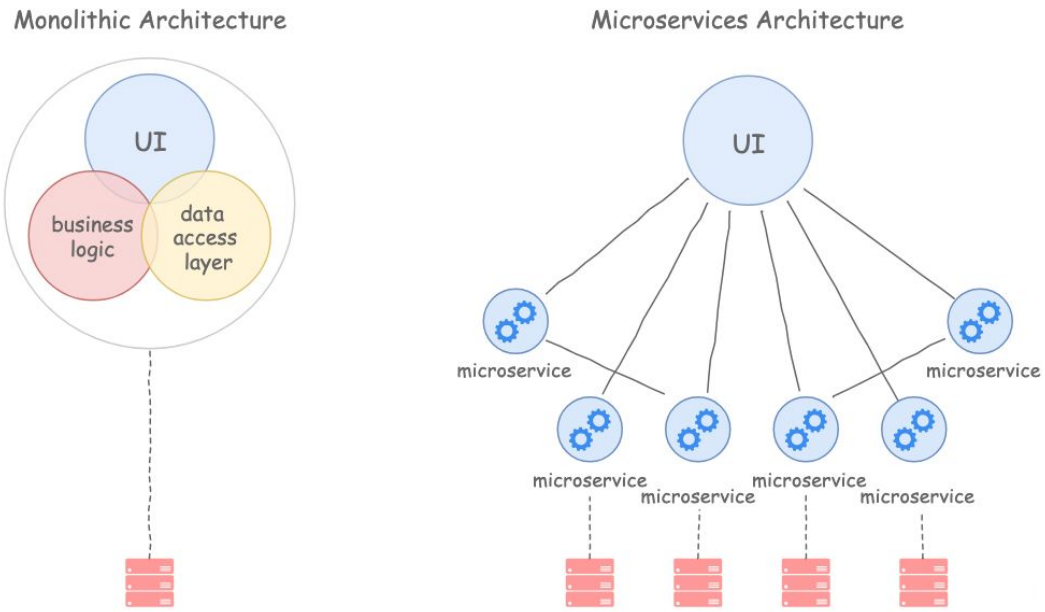
1. Billing – Add up all the charges for services with taxes.

2. Reservations - Assign rooms, schedule cleaning services.

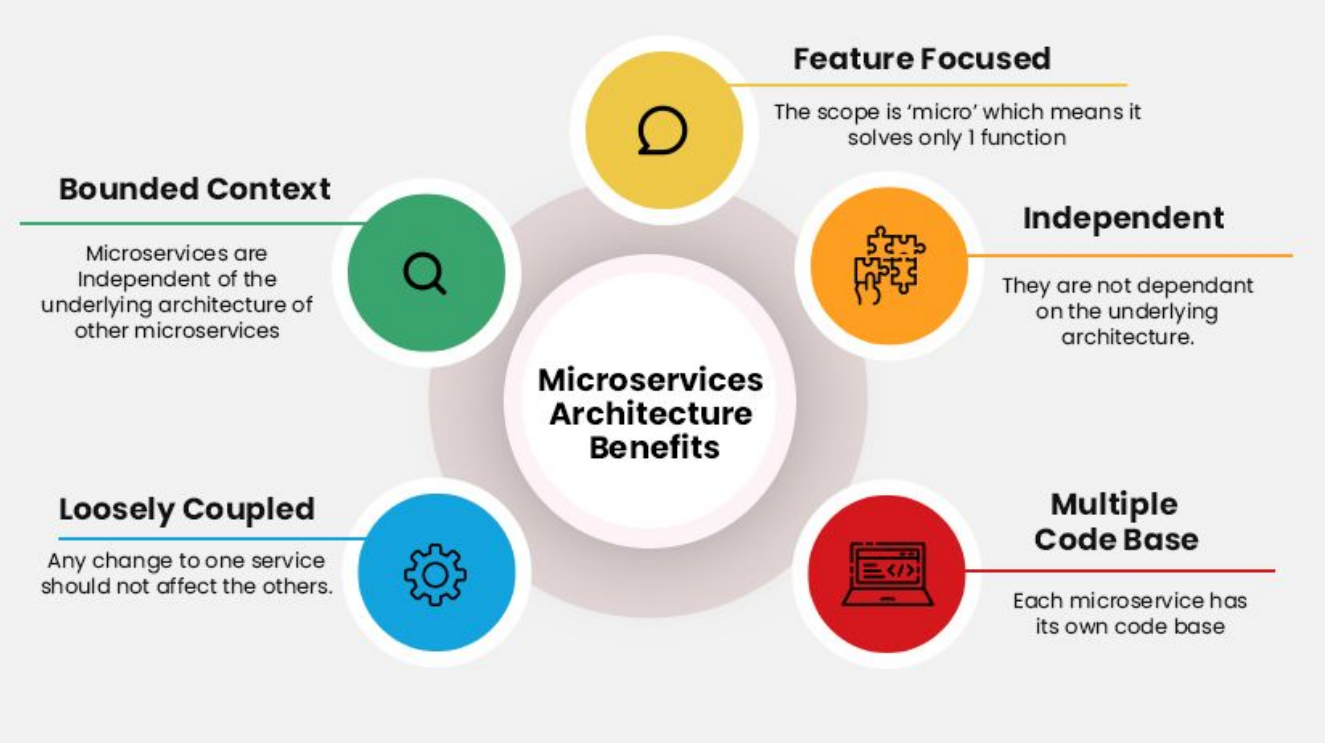
3. Payroll - Pay employees for the services and hours they work.



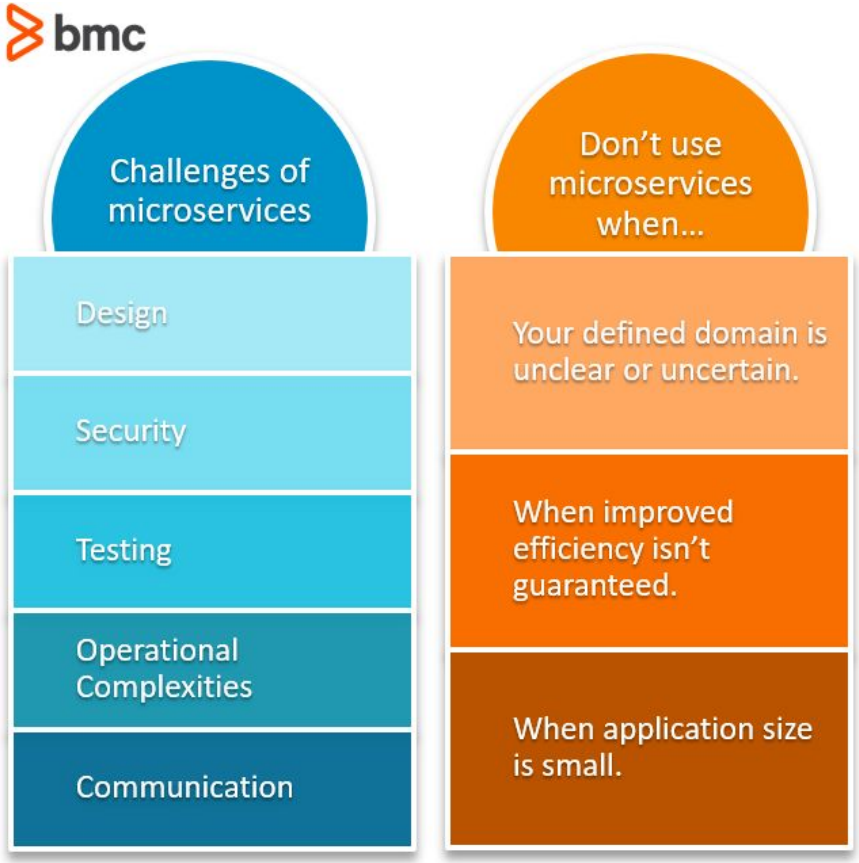
Make them into Microservices!



2. Advantages of Microservice Architecture:



3. Disadvantages of Microservice Architecture:



- Increased complexity
- More expensive
- Greater security risk

What You've Done So Far:

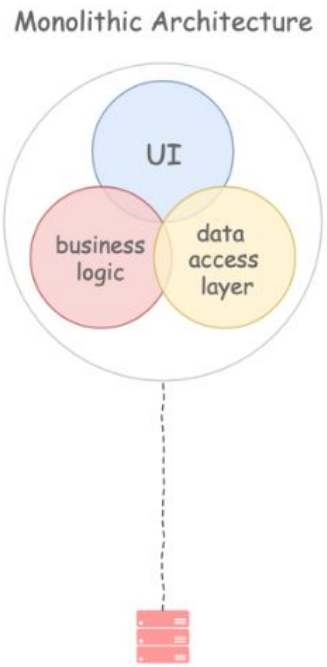
MONOLITH

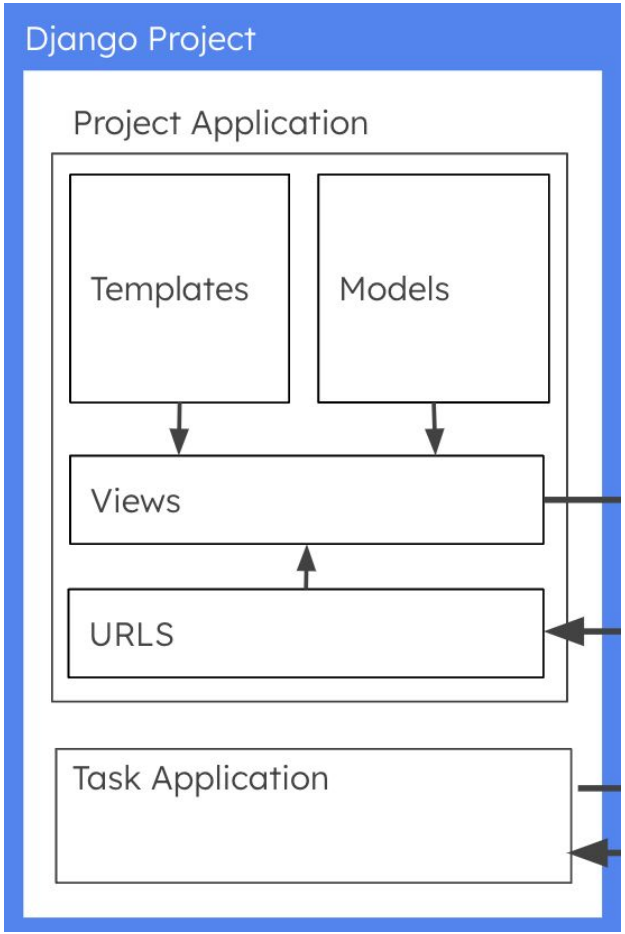
Pros:

- Easier to get up and running
- Everything in one language
- Clear process for each feature

Cons:

- Doesn't scale as well!
- Locked into one language /platform across all teams
- Hard to add big new features with multiple teams



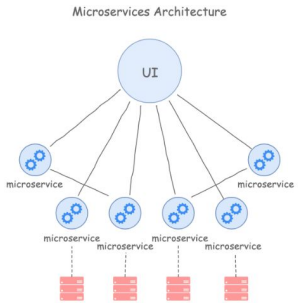


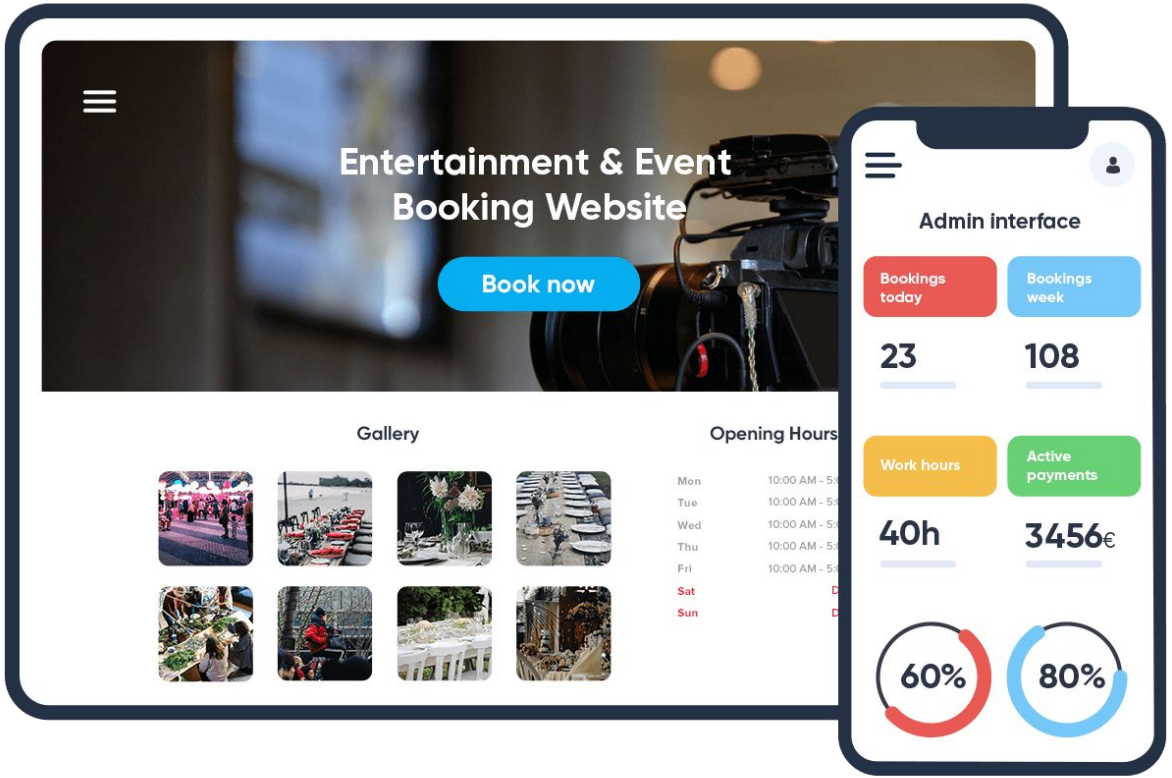
HACK REACTOR
BY GALVANIZE

Your Project This Module:

CONFERENCE GO

- Microservices Architecture
- Django as API



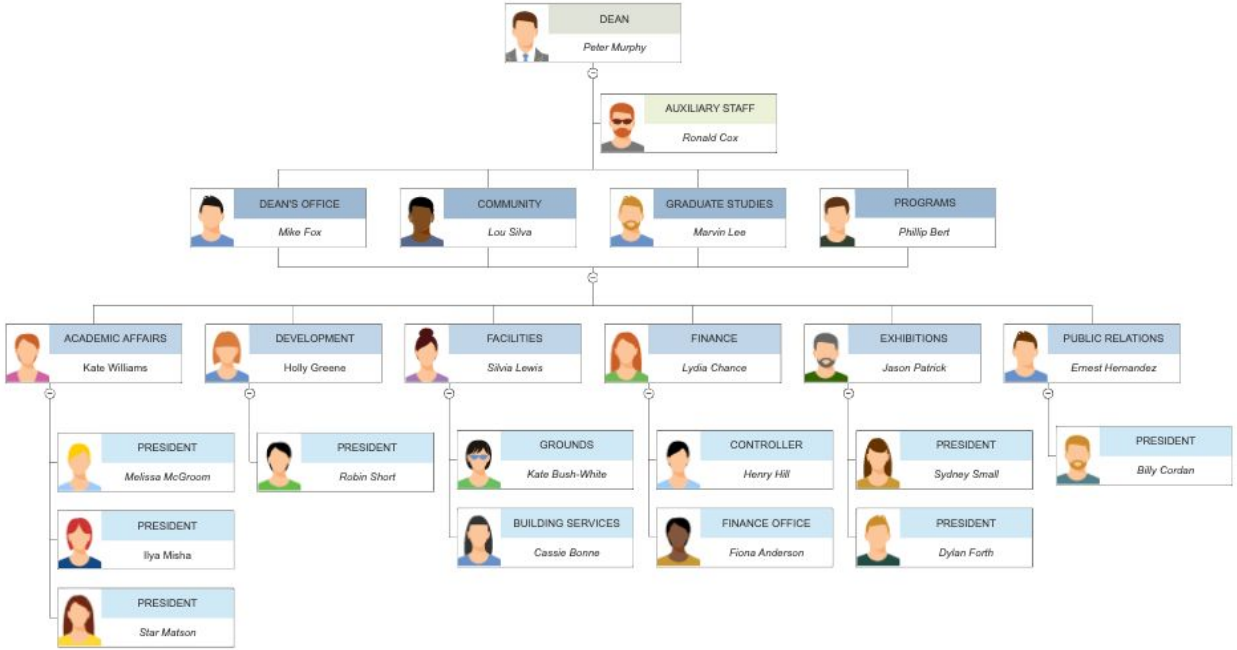


HACK REACTOR
BY GALVANIZE

BREAK

5 min

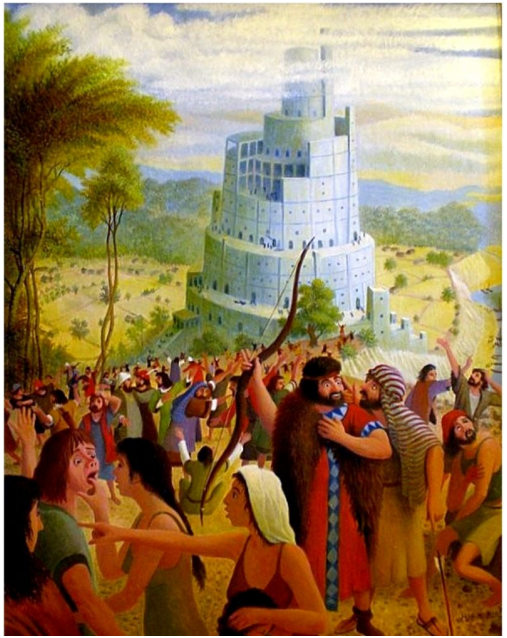
Be like Bjork



Domain-Driven Design

Applying object-oriented programming to modelling human behaviors and business problems.

4. Ubiquitous Language



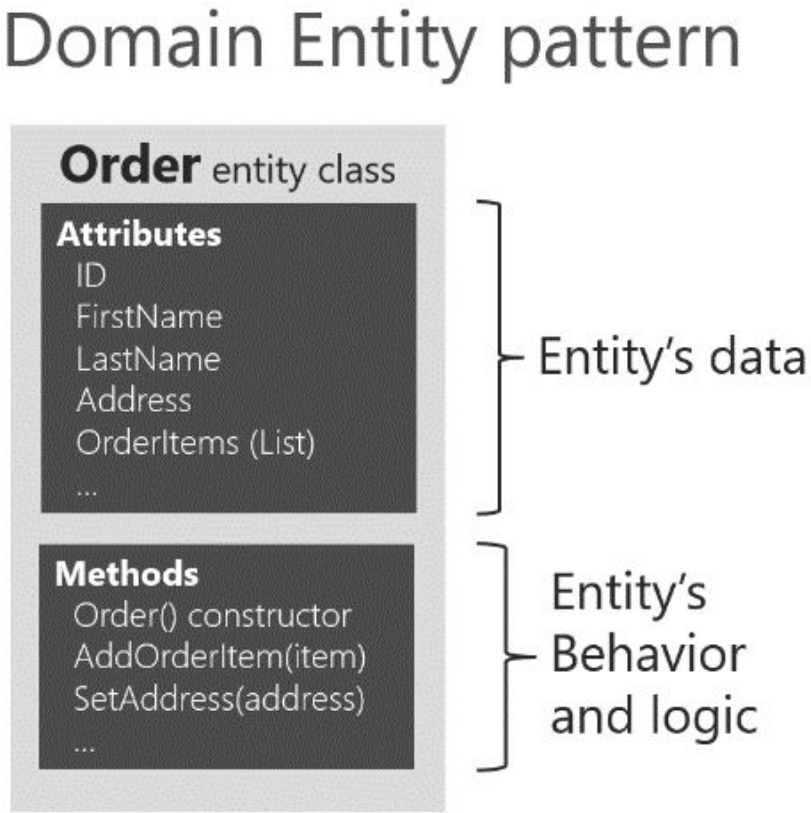
The common language shared by domain experts, users, and developers:

- Databases
  - Data associations / relationships
- Methods
  - Business processes
- Variables
  - Entities

HACK REACTOR
BY GALVANIZE

5. Entities As Objects:

- In Domain-Driven Design, an entity is a **representation of an object in the domain.**
- *"An object primarily defined by its identity is called an Entity." (Eric Evans)*
- An entity has attributes that are likely to change over time.

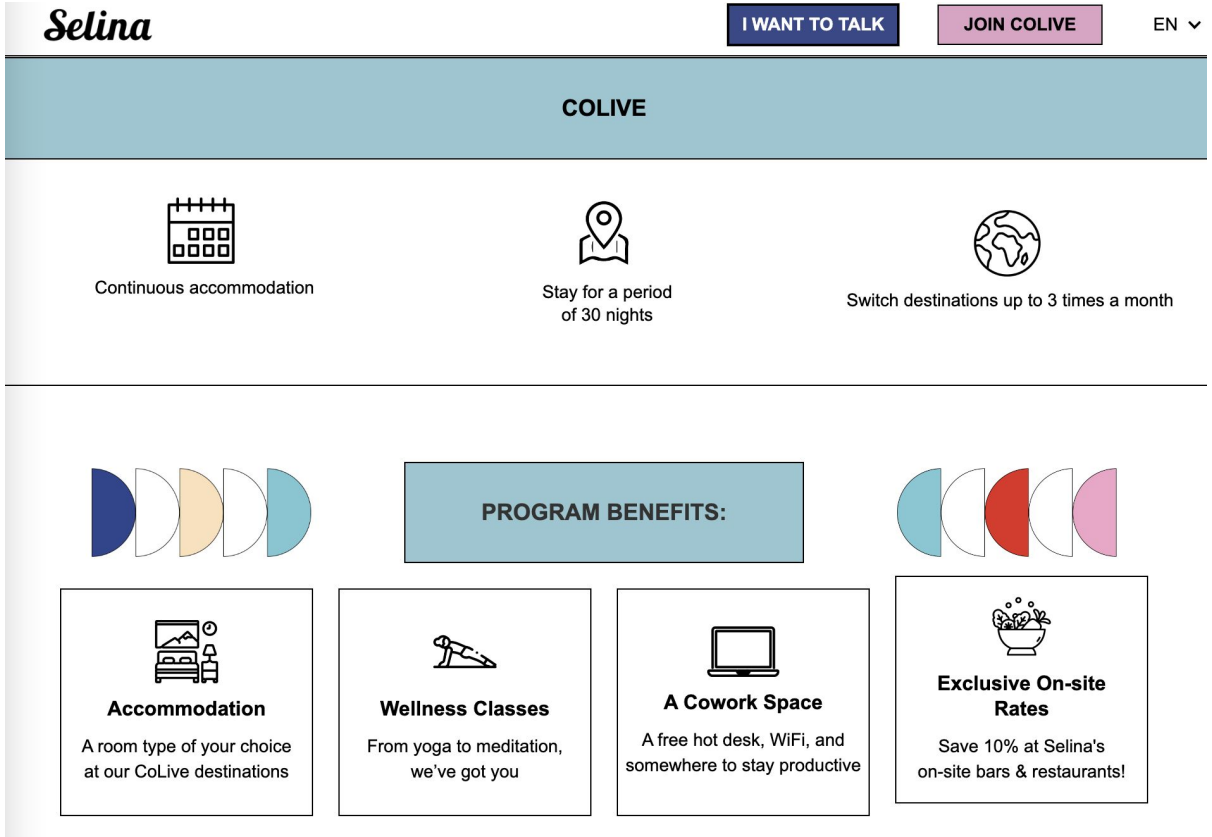


Example: Making a Reservation at Selina

Business Problems:

1. **Booking** - Assign rooms by room type, location and availability.

2. **Billing** – Add up all the charges for the reservation, with services and taxes.



slido

Name some attributes for a room reservation at Selina CoLive:

GALVANIZE

slido

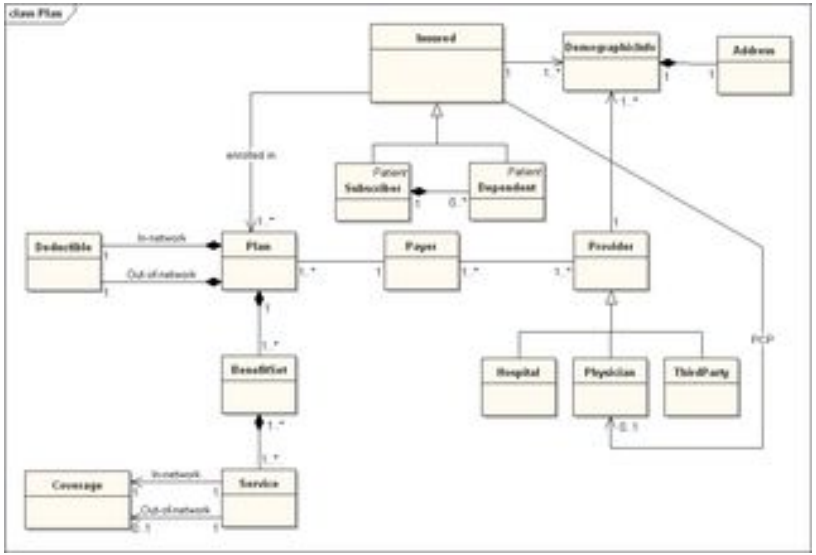
Name some methods for generating a bill at Selina CoLive:

GALVANIZE

Data Modeling: Hotel Reservation App

1. **Booking** - Assign rooms by room type, location and availability.

2. **Billing** – Add up all the charges for the reservation, with services and taxes.



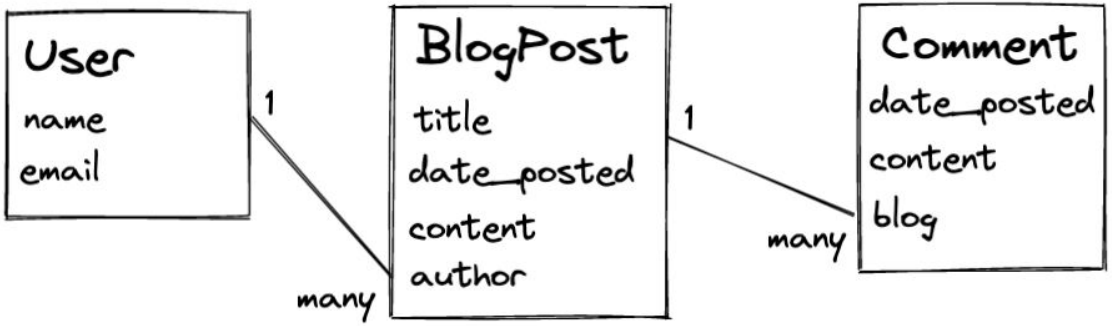
HACK REACTOR
BY GALVANIZE

Your Turn: Model a Hotel Reservation

Business Problems:

1. **Booking** - Assign rooms by room type, location and availability.

2. **Billing** – Add up all the charges for the reservation, with services and taxes.



- Object Names
- Attributes & Methods
- Relationships between objects:
  - One-to-one
  - One-to-many

HACK REACTOR
BY GALVANIZE

GALVANIZE

6. Entities vs. Value Objects

Entities vs. Value Objects

Entities

- Persistent Identity
- Has a lifecycle
- Attributes may change, but the identity of the object does not change

Entities vs. Value Objects

Entities

- Persistent Identity
- Has a lifecycle
- Attributes may change, but the identity of the object does not change

EXAMPLES:

- Automobile
- Customer
- Invoice

Entities vs. Value Objects

Entities

- Persistent Identity
- Has a lifecycle
- Attributes may change, but the identity of the object does not change

EXAMPLES:

- Automobile
- Customer
- Invoice

Value Objects

- Indicates static value – it IS its value
- Probably does not have an ID
- Used to define entities

RED

Entities vs. Value Objects

Entities

- Persistent Identity
- Has a lifecycle
- Attributes may change, but the identity of the object does not change

EXAMPLES:

- Automobile
- Customer
- Invoice

Value Objects

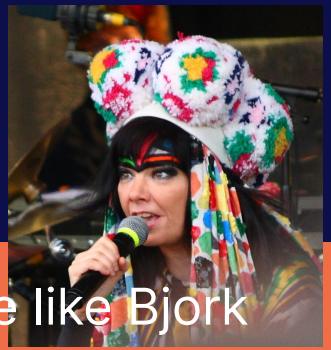
- Indicates static value – it IS its value
- Probably does not have an ID
- Used to define entities

EXAMPLES:

- Paint color RED
- Currency
- Vehicle type

HACK REACTOR
BY GALVANIZE

Quiz Game:
ENTITY or VALUE OBJECT?



slido

Hotel Room

GALVANIZE

slido

Room type:
Standard, Suite, or Community

GALVANIZE

slido

Guest

GALVANIZE

slido

Bill

GALVANIZE

slido

Currency

GALVANIZE

GALVANIZE

{JSON}

JavaScript Object Notation

7. JSON

BREAK

5 min

Make it JSON!

What is JSON?

- **A lightweight text-based format for storing and transporting data.**

What is JSON?

- A lightweight text-based format for storing and transporting data.

- Often used to transmit data as a string of characters in web applications, across HTTP from a web server to the browser.

What is JSON?

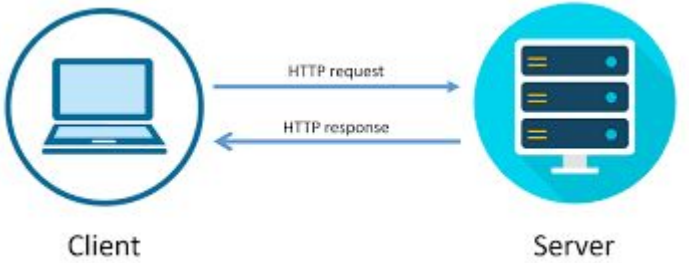
- **A lightweight text-based format for storing and transporting data.**

- **Often used to transmit data as a string of characters in web applications, across HTTP from a web server to the browser.**

JSON Example

This example is a JSON string:

```
'{"name":"John", "age":30, "car":null}'
```



HACK REACTOR
BY GALVANIZE

Rules

1. **Data is in name/value pairs**

JSON Example

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

Rules

1. **Data is in name/value pairs**
2. **Data is separated by commas**

JSON Example

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

Rules

1. Data is in name/value pairs
2. Data is separated by commas
3. Curly braces hold objects

JSON Example

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

Rules

1. Data is in name/value pairs
2. Data is separated by commas
3. Curly braces hold objects
4. Square brackets hold arrays

JSON Example

```
{"employees":[
  { "firstName":"John", "lastName":"Doe" },
  { "firstName":"Anna", "lastName":"Smith" },
  { "firstName":"Peter", "lastName":"Jones" }
]}
```

Rules

- **You can access a property in a JSON object using dot notation:**

person.name

person.age

peson.car

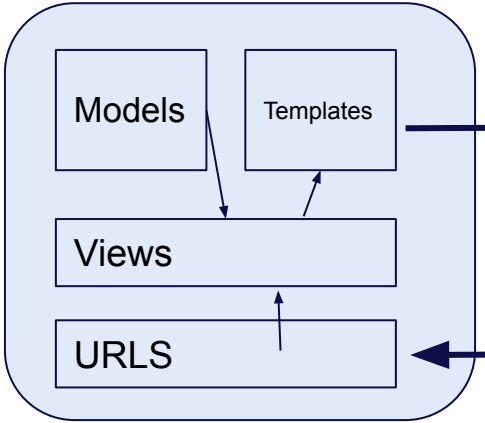
JSON Example

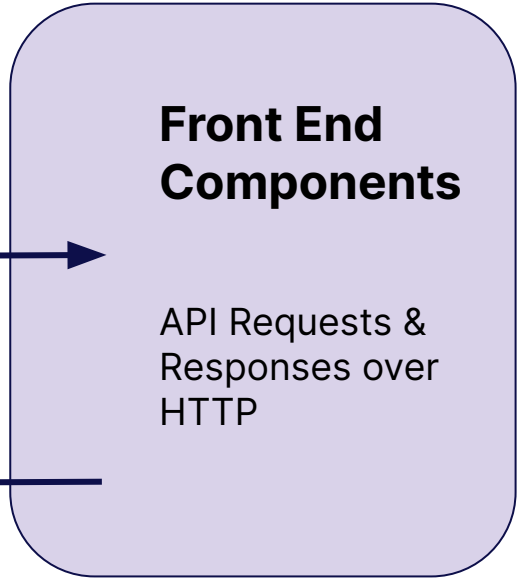
This example is a JSON string:

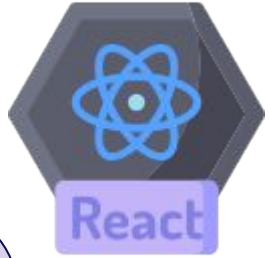
```
'{"name":"John", "age":30, "car":null}'
```

What are we using it for?

- **Pass data between back end and front end:**

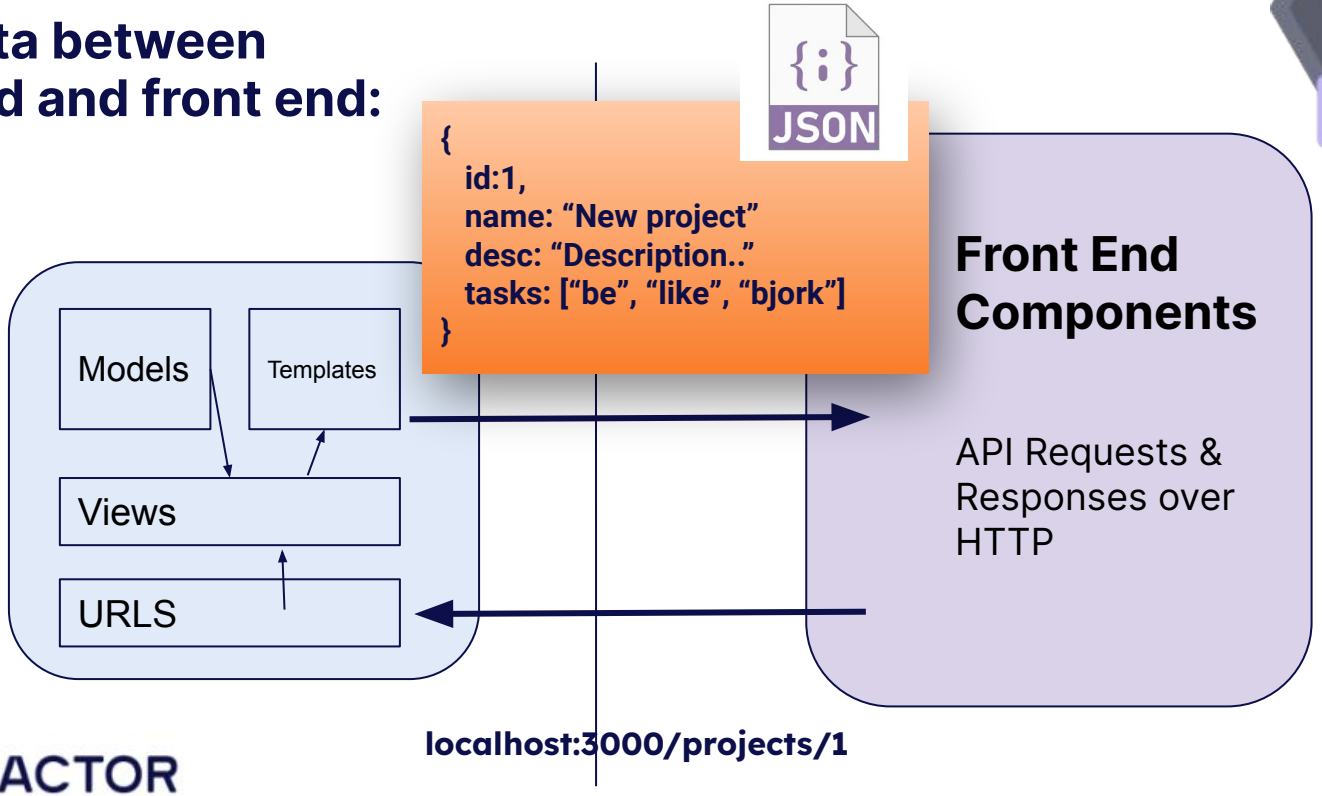






What are we using it for?

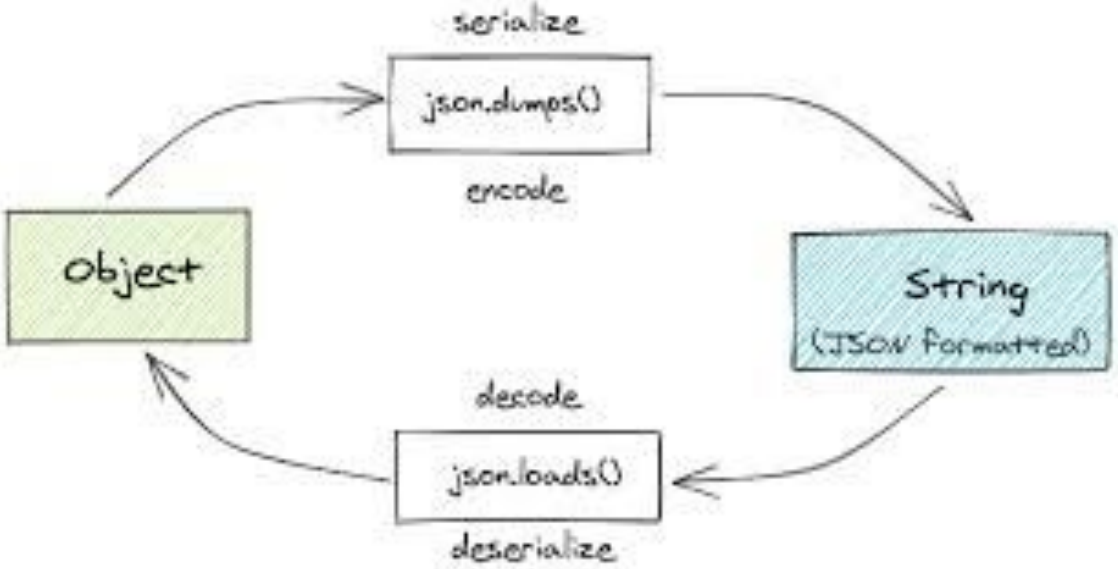
- **Pass data between back end and front end:**



Rules

- **You can transform Python dictionaries into JSON strings, using:**

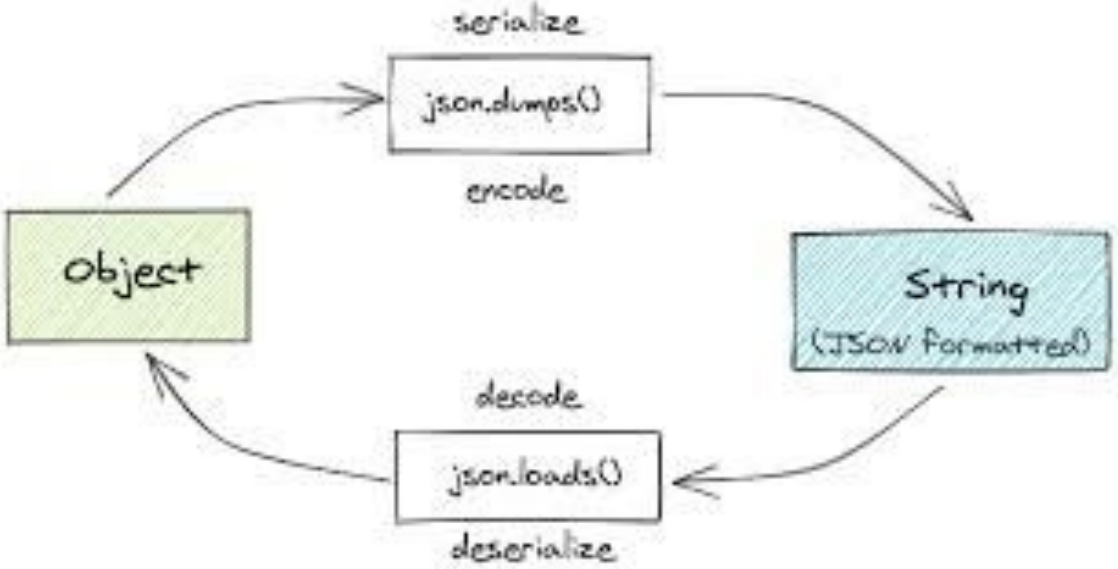
JSON.dumps( )



Rules

- **You can transform JSON strings into Python dictionaries, using:**

JSON.loads( )



Code-Along

You can load the following JSON data types:

- String
- Number
- Boolean
- Array
- Object
- Null

```python
import json

json.loads("3")
json.loads("6.28")
json.loads("3") + json.loads("6.28")

json.loads('"Hello"')
json.loads("true")
json.loads("false")

json.loads("null")
```

slido

Name some of the rules of JSON:

slido

What is Domain-Driven Design?

GALVANIZE

GALVANIZE | HACK REACTOR BY GALVANIZE

Review – Day 1

Intro to Software Design & Architecture

Domain-Driven Design

1. **Activity:** Designing a Hotel Reservation System
2. **Theory:** Entities & Value objects
3. **Code-Along:** JSON

AFTERNOON
Technical Practice:

JSON Responses/Requests with Django

Goals for the Week

THEORY: Domain-Driven Design

APPLICATION: Building an API back end

Monday

THEORY:

* **Entities & Models** - What are our models?

APPLICATION:

* Django Models and **JSON**

Tuesday

THEORY:

* **Aggregates** - How do we group our collections?

APPLICATION:

* Building a **JSON** Library

Wednesday

THEORY:

* **Bounded Context** - What are our domain systems?

APPLICATION:

* Making **API Interfaces**

Thursday

THEORY:

* **Anti-Corruption Layer** - How do we structure and protect data?

APPLICATION:

* Working with **3rd party API's**

GALVANIZE

Friday: Containerization with **Docker**

Major Themes

Week 1

1. Modeling a business problem as object-oriented software.
2. Breaking monolithic apps into **microservices**.
3. Sending data across HTTP in the form of a JSON object.

HACK REACTOR
BY GALVANIZE