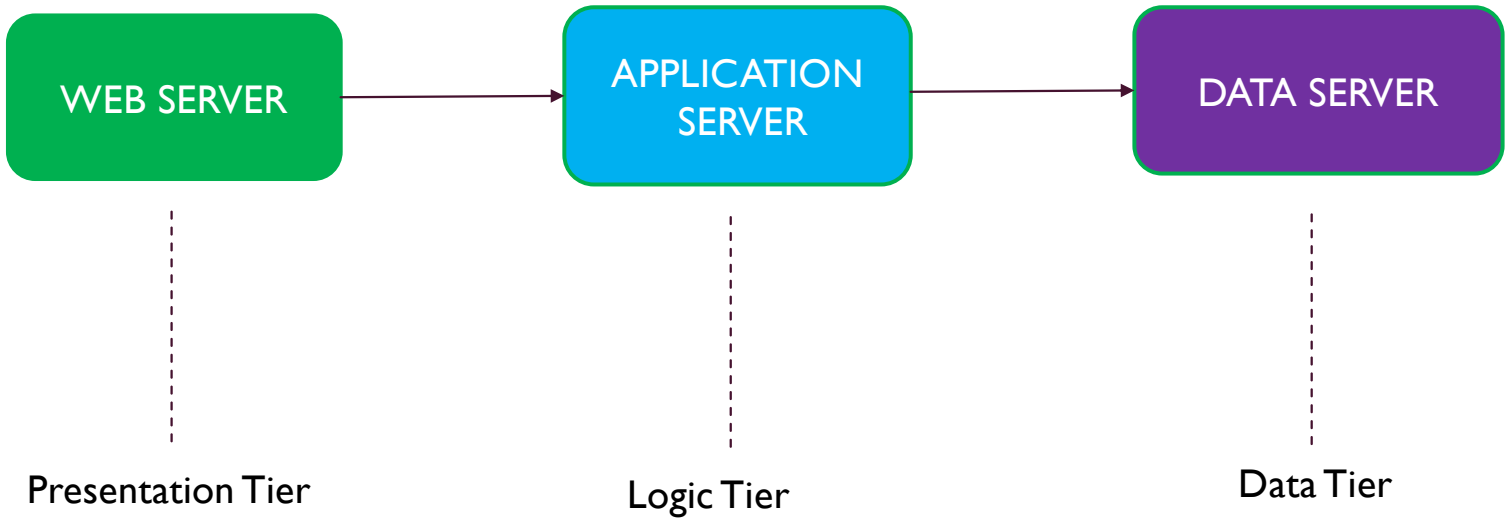
LABORATORIO APPLICAZIONE SW E SICUREZZA INFORMATICA

ROBERTO BERALDI

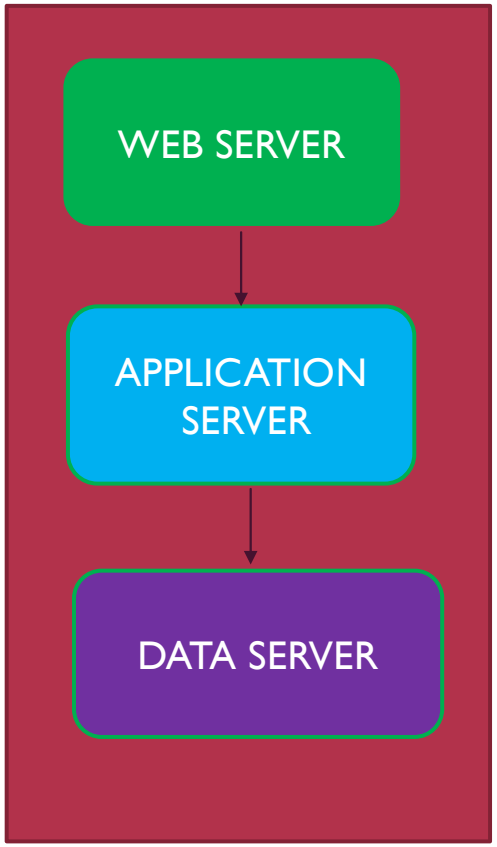
ARCHITETTURA 3-TIER
SUDDIVISIONE IN LIVELLI



ARCHITETTURA 3-TIER
REALIZZAZIONE SILO

ASP (Active Server Page) → C#,VB.NET
Django → Python
Flask → Python
Enterprise Java Beans (EJB) →Java
Java Server Page (JSP) –Java
Rails (RoR)→ Ruby
Spring → Java
Zend → php

HTTP/HTML



CLIENT GRAFICO (**BROWSER**),

REALIZZAZIONE 'SILO'

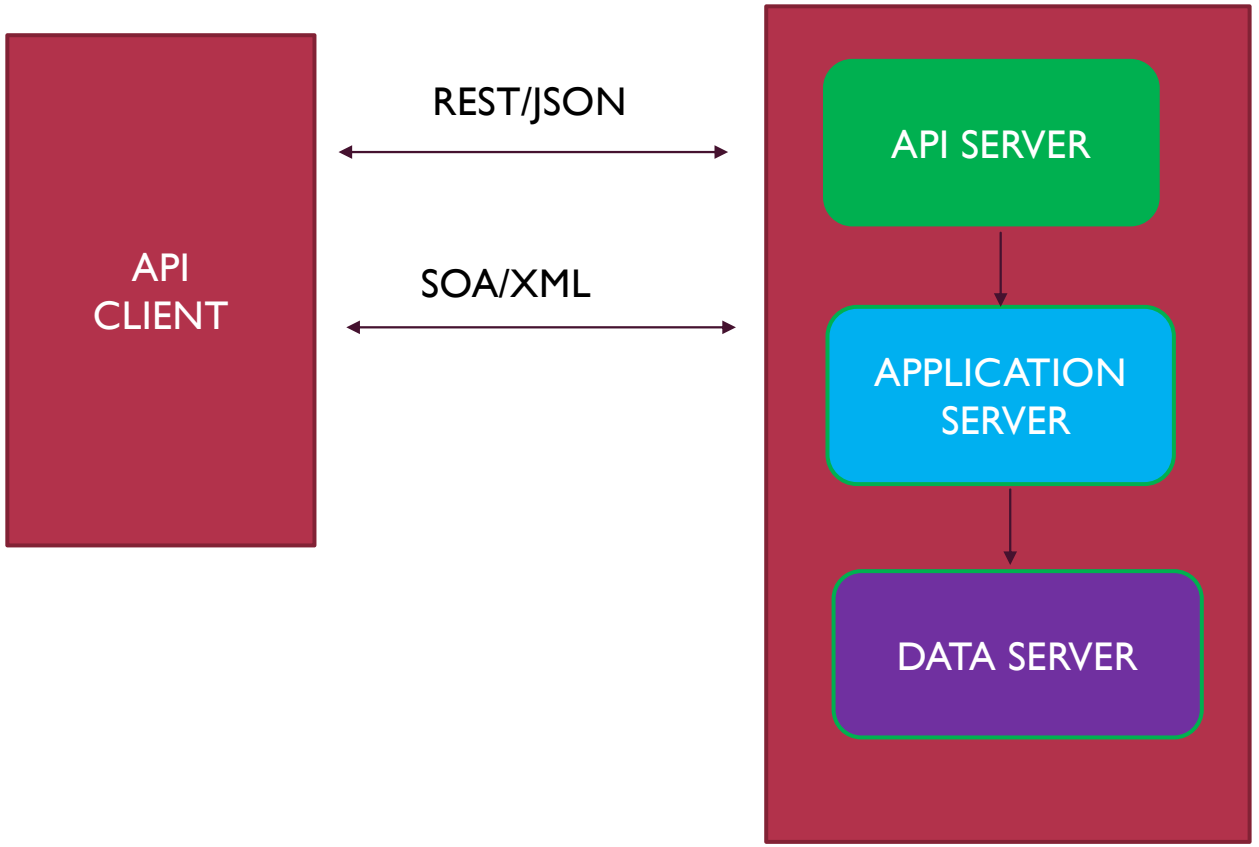
- I sistemi SILO sono SW 'auto-contenuti' e 'chiusi'
- Gestiscono direttamente i dati e o non comunicano con altri sorgenti dati o lo fanno con difficoltà
- In altri termini **non si integrano** con altri sistemi
- .. O non sono progettati per essere integrati
- .. Potremmo definirli sistemi chiusi

REALIZZAZIONE A SERVIZIO
SOFTWARE AS-A-SERVICE, SAAS

CLIENT NON
NECESSARIAMENTE
USATO DA UN UTENTE FINALE

OFFRONO ACCESSO A DATI
TRAMITE UNA INTERFACCIA SW

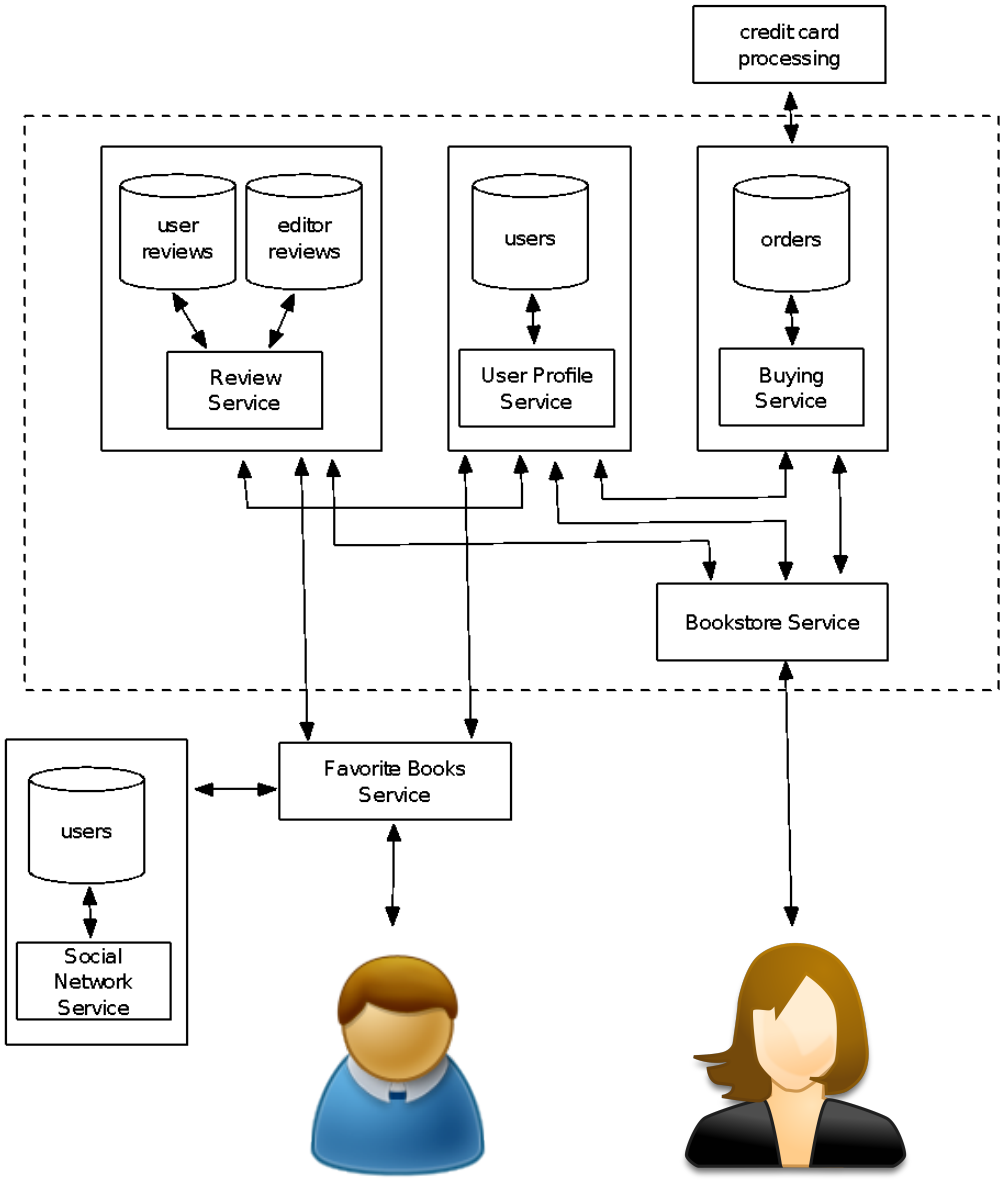
SONO **APERTI**



REST VS SOA

- SOAP è un protocollo, REST è uno stile architetturale
- SOAP espone espone la logica applicative mediante una interfaccia (definita caso per caso), REST ha un'interfaccia fissa (CRUD)
- SOAP più verboso (XML)

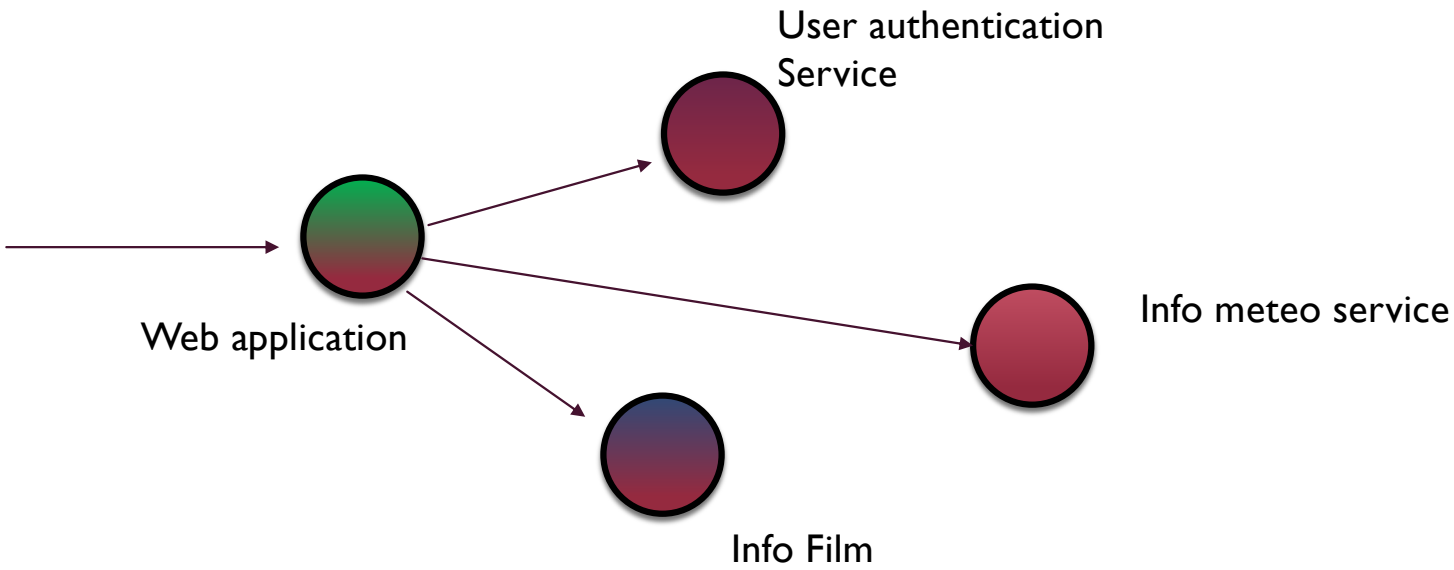
Esempio Sistema SoA
(Fig. 1.7 del libro di testo)



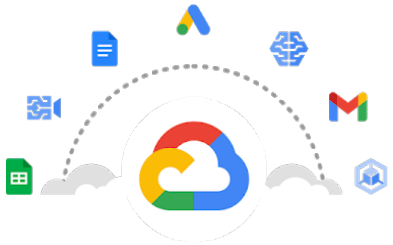
ESEMPIO DI SERVIZIO (AUTENTICATION SERVICE)

- E’ un servizio che consente l’autenticazione di un utente

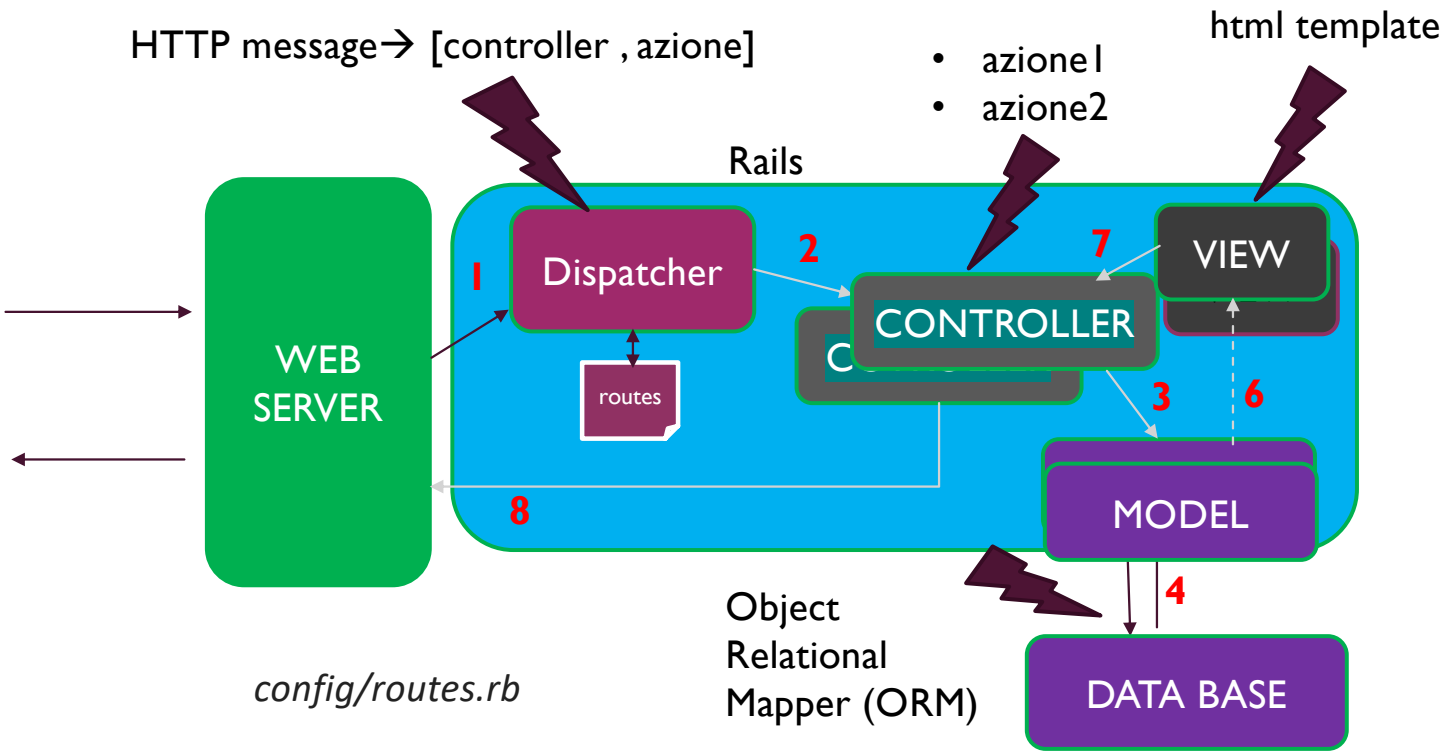
SAAS E CLOUD COMPUTING

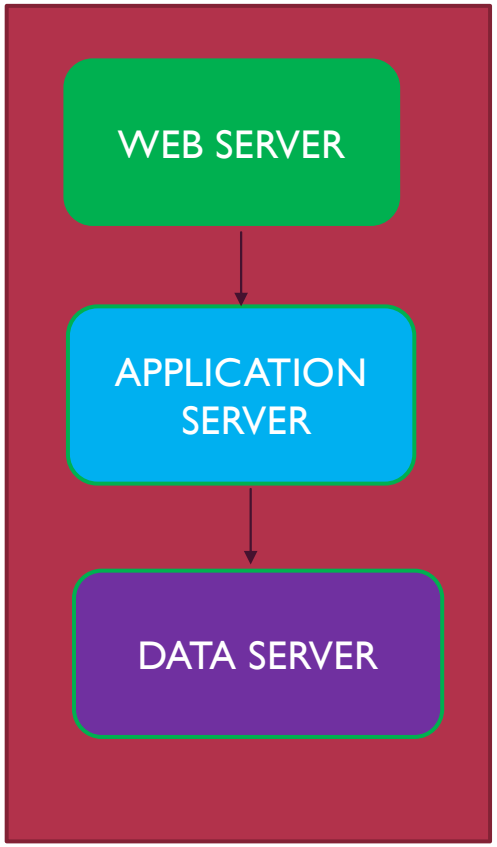


ESEMPI DI SAAS



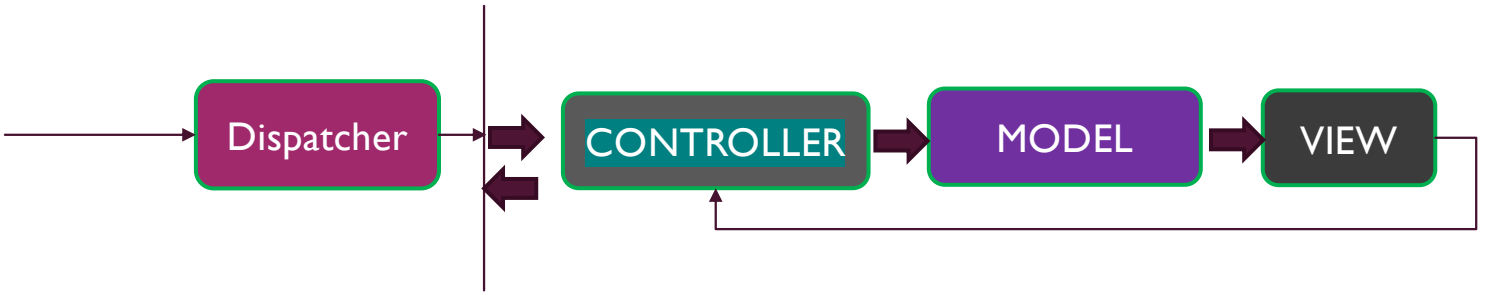
RAILS (VISTA D'INSIEME)

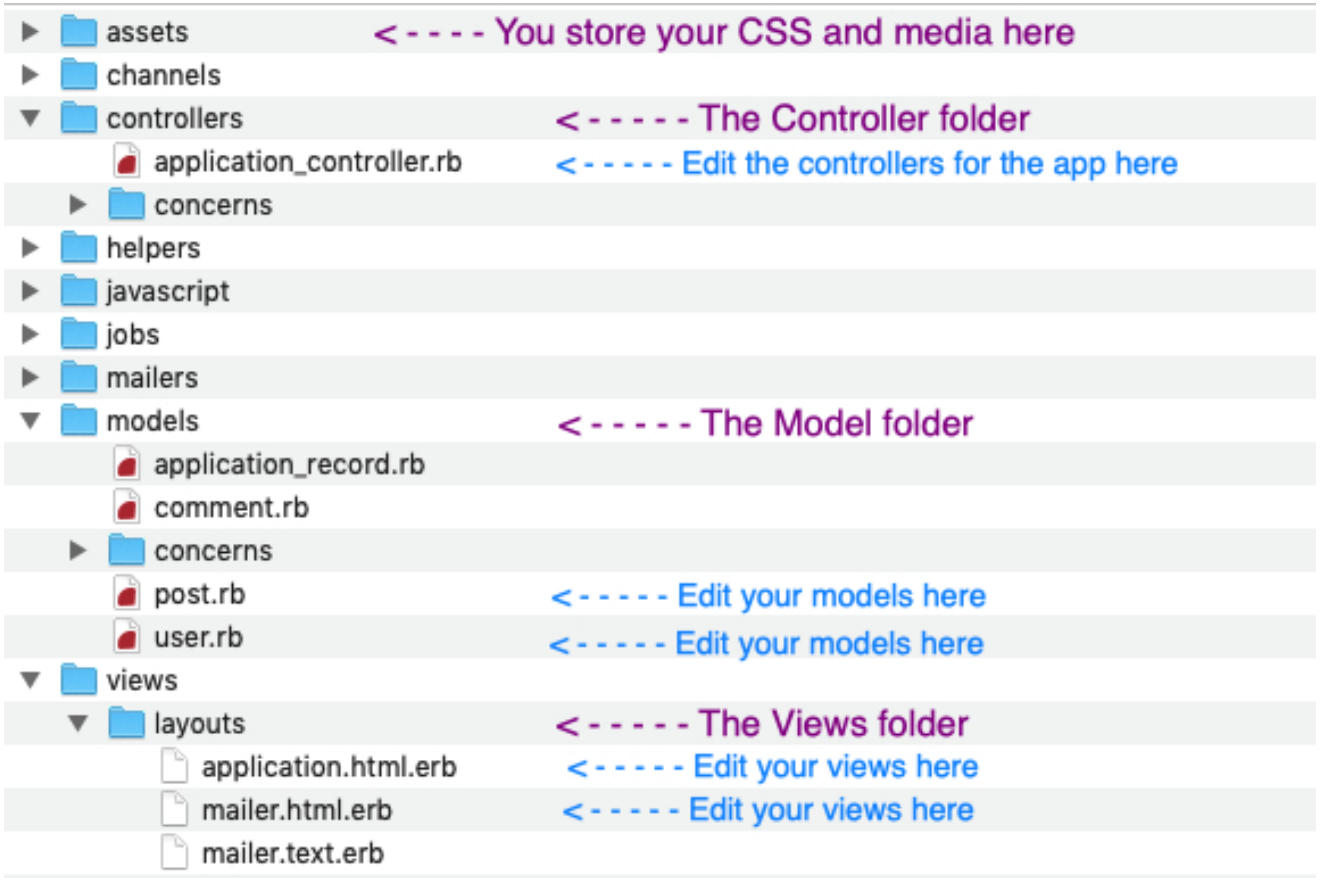




RAILS REQUEST PROCESSING

Action Controller → Action Model –> Action View




assets
<- - - - You store your CSS and media here
channels
controllers
<- - - - - The Controller folder
application_controller.rb
<- - - - - Edit the controllers for the app here
concerns
helpers
javascript
jobs
mailers
models
<- - - - - The Model folder
application_record.rb
comment.rb
concerns
post.rb
<- - - - - Edit your models here
user.rb
<- - - - - Edit your models here
views
layouts
<- - - - - The Views folder
application.html.erb
<- - - - - Edit your views here
mailer.html.erb
<- - - - - Edit your views here
mailer.text.erb

STRUMENTI DI AMMINISTRAZIONE: RAKE

- Rake è una utility Ruby autonoma che sostituisce il 'make' di Unix

- Rake viene utilizzato per attività di amministrazione comuni
- **Esempi**
- ***rake assets:clean*** # Remove old compiled assets
- ***rake db:create*** # Create the database from config/database.yml for the current Rails.env
- ...
- ***rake routes***

COMANDI RAILS

- Rails è il nome di di un programma di gestione dell'intero framework
- Esempi

- **rails server**
- **rails generate …**
- **rails new *app_name***

ESEMPIO

rake routes

```
GET /movies           {:action=>"index", :controller=>"movies"}
POST /movies           {:action=>"create", :controller=>"movies"}
GET /movies/new       {:action=>"new", :controller=>"movies"}
GET /movies/:id/edit {:action=>"edit", :controller=>"movies"}
GET /movies/:id       {:action=>"show", :controller=>"movies"}
PUT /movies/:id       {:action=>"update", :controller=>"movies"}
DELETE /movies/:id      {:action=>"destroy", :controller=>"movies"}
```

INTERFACCIA CRUD (CON RIFERIMENTO A RAILS)
READ (ALL)

GET /movies

HTML con lista dei movies


```
GET /movies          {:action=>"index", :controller=>"movies"}
```

INTERFACCIA CRUD (CON RIFERIMENTO A RAILS)
READ (ONE)

GET /movies/3

HTML con un solo movie


```
GET /movies/:id      {:action=>"show", :controller=>"movies"}
```

INTERFACCIA CRUD (CON RIFERIMENTO A RAILS)
CREATE

1. GET /movies/new

2. HTML con form da riempire

3. POST /movies/

name=..,..

4. HTML con risultato


```
GET /movies/new      {:action=>"new", :controller=>"movies"}
POST /movies          {:action=>"create", :controller=>"movies"}
```

INTERFACCIA CRUD (CON RIFERIMENTO A RAILS)
UPDATE

1. GET /movies/3/edit

2. HTML con form da modificare

3. PUT /movies/3

name=..,..

4. HTML con risultato


```
GET /movies/:id/edit {:action=>"edit", :controller=>"movies"}
PUT /movies/:id      {:action=>"update", :controller=>"movies"}
```

INTERFACCIA CRUD (CON RIFERIMENTO A RAILS)
DELETE

1.DELETE /movies/3/

2. HTML con risultato


```
DELETE /movies/:id      {:action=>"destroy", :controller=>"movies"}
```