MVC

MODEL - VIEW - CONTROLLER

... YOU WALK INTO A CAFE

...

You request a
large **Cappuccino**

Barista is the Controller
(Drinks Controller)

… BARISTA KNOWS IT'S A COFFEE BASED DRINK

Model Drink

THERE ARE MANY
INSTANCES OR
KINDS OF "DRINK" ON THE MENU

EACH DRINK...

- HAS A NAME
- HAS A RECIPE
- HAS A INGREDIENTS, ETC.

id	name	recipe
1	Cappucino	1/3 steamed milk, espresso, coccoa powder
2	Late	espresso, 2/3 steamed milk
3	Machiatto	espresso, 1/3 steamed milk
4	Americano	water, coffee

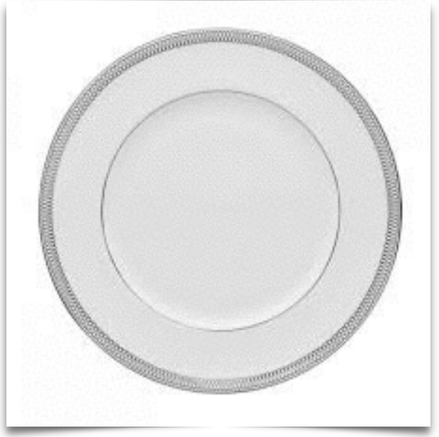
(drinks database) ...
some drinks have same ingredients

BARISTA PRESENTS ...

... THIS VIEW

NOTICE … CUP IS THE “VIEW”

… NOT A BUCKET OR PLATE



makes things happen

THE BARISTA (CONTROLLER)....

INTERACTS WITH

THE "VIEW"

THE "MODEL" - DRINK

VIEW IS ....

THE “VIEW”

… only responsible for presentation,
little logic, no computation

… only displays/renders the Model “drink”

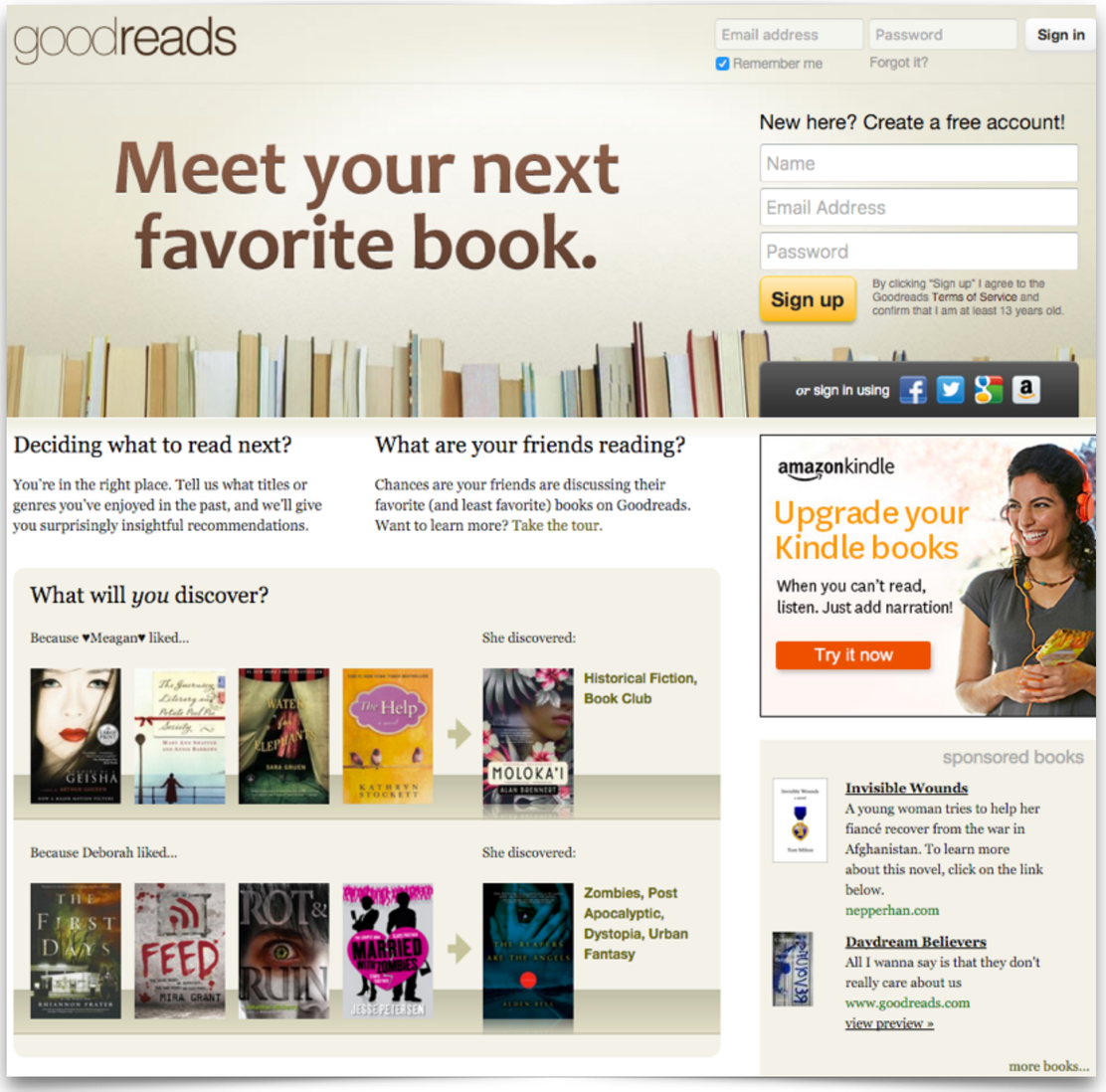
MODEL IS WHERE ....

THE “MODEL” - DRINK

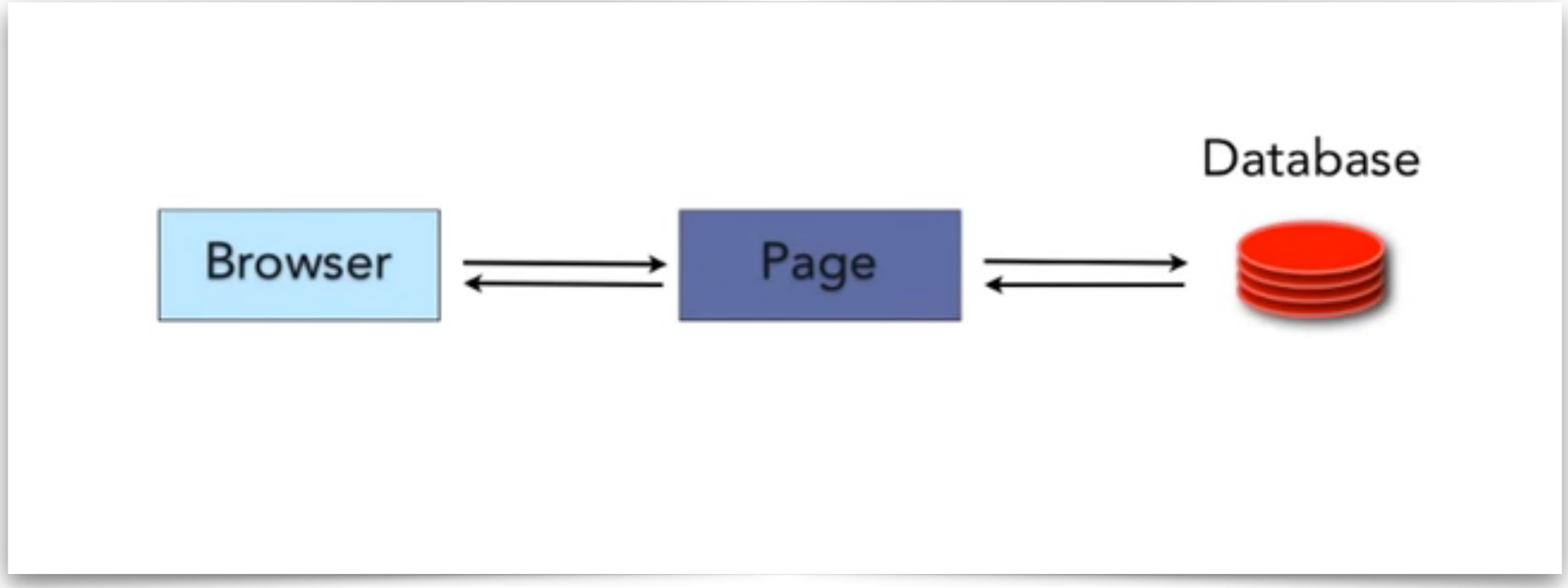
… most of the computation/logic, i.e figuring what ingredients should go with what.

… only displays/renders the Model “drink”

... CONSIDER A WEB APP LIKE GOODREADS

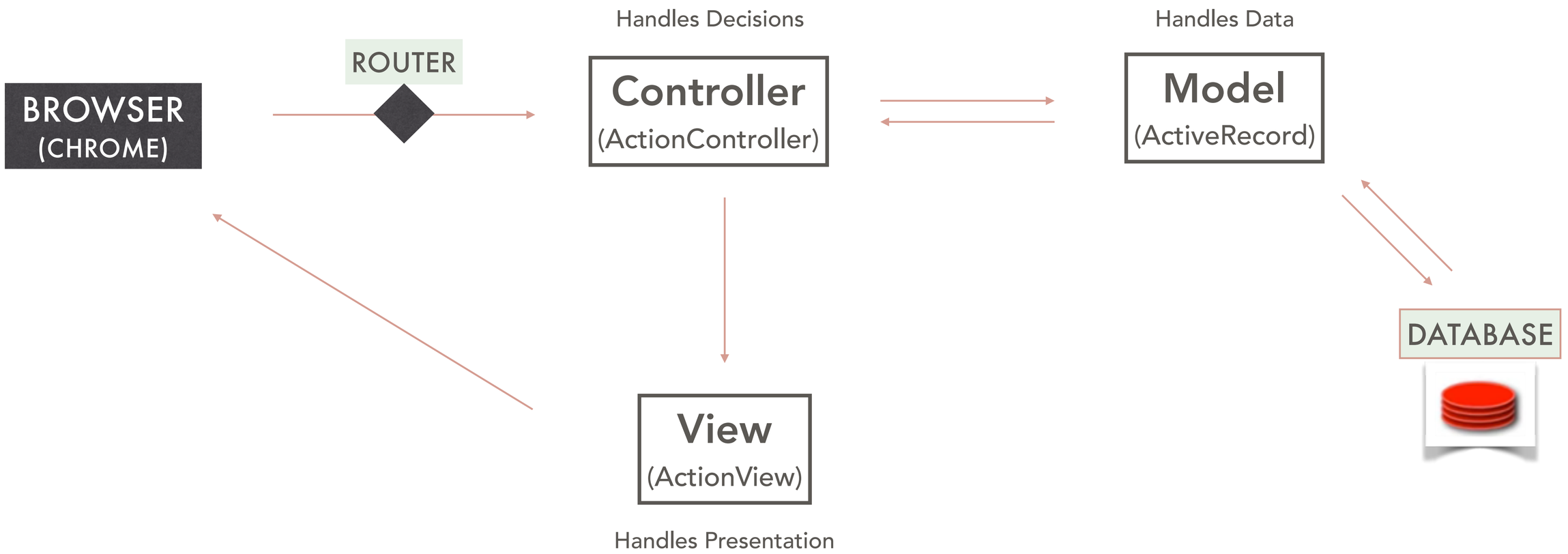


Basic Web Architecture

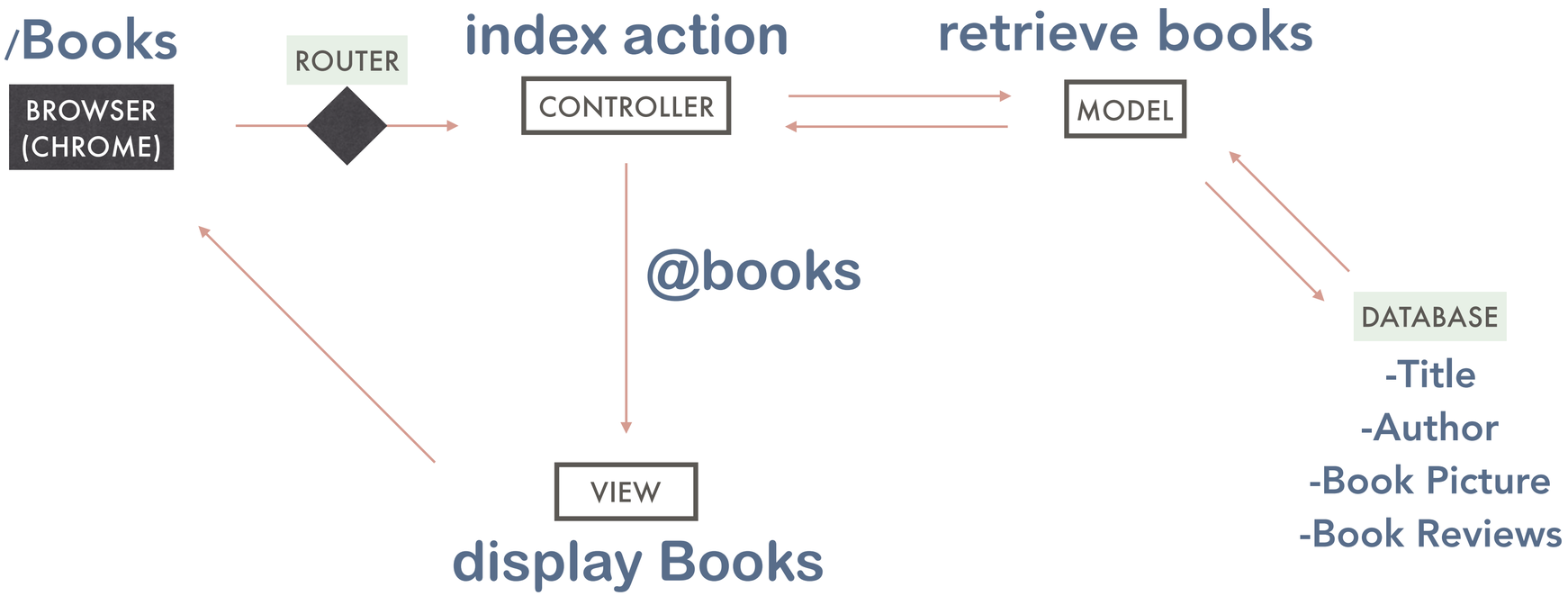


MVC

Beautiful Rails Architecture



MVC


/Books
BROWSER (CHROME)
ROUTER
index action
CONTROLLER
retrieve books
MODEL
@books
DATABASE
-Title
-Author
-Book Picture
-Book Reviews
VIEW
display Books

Quick Summary

- Benefits: Clear separation of interests.
- Model (ActiveRecord) — handles relationships, validations, and business logic.
- View (ActionView) — renders view, uses ERb (embedded Ruby) for templating (the cup is a template for all drinks)
- Controller (ActionController) — Controls which view to use, makes data available as instance variables @books/@drinks. Contains several actions/defs. Controls rendering and redirection.