A Model of the Operation of

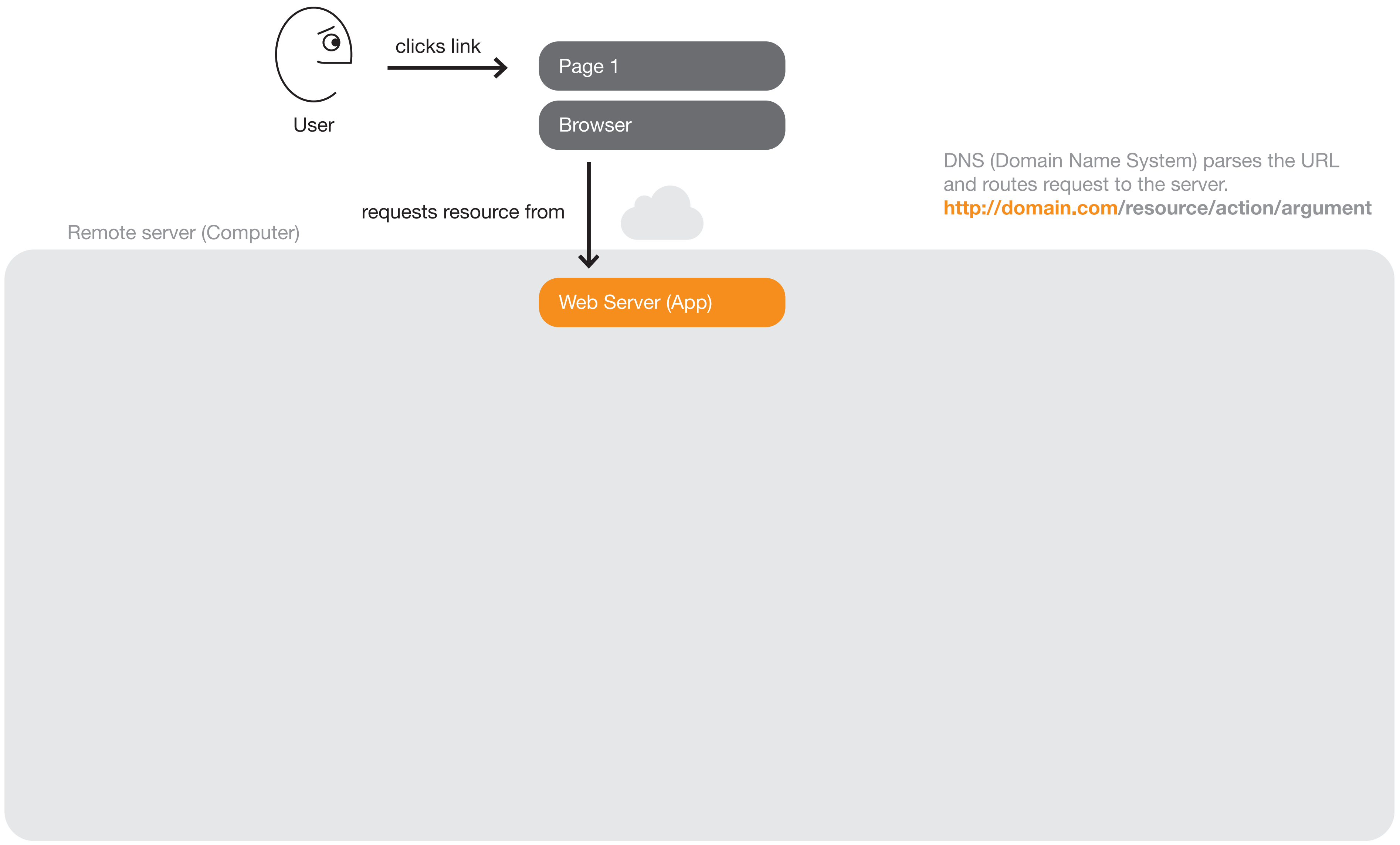
The Model-View-Controller Pattern

in a Rails-Based Web Server

Dubberly Design Office
January 10, 2011
v 0.4

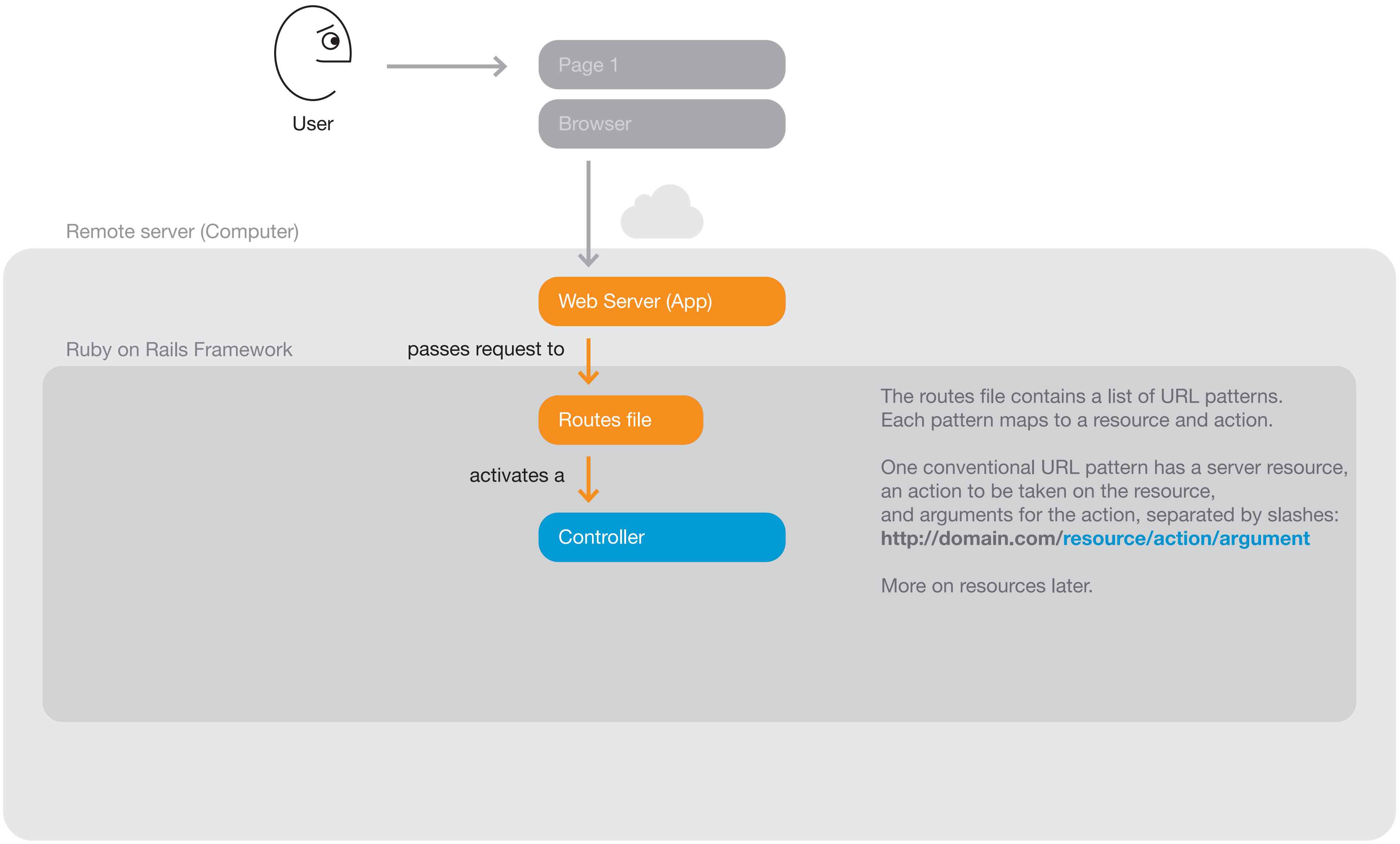
Responding to a page request

A user clicks a link to a page in a web application.



DNS (Domain Name System) parses the URL and routes request to the server.
http://domain.com/resource/action/argument

The web server receives the request URL (Universal Resource Location). Rails uses a routes file to match the URL with a controller action.

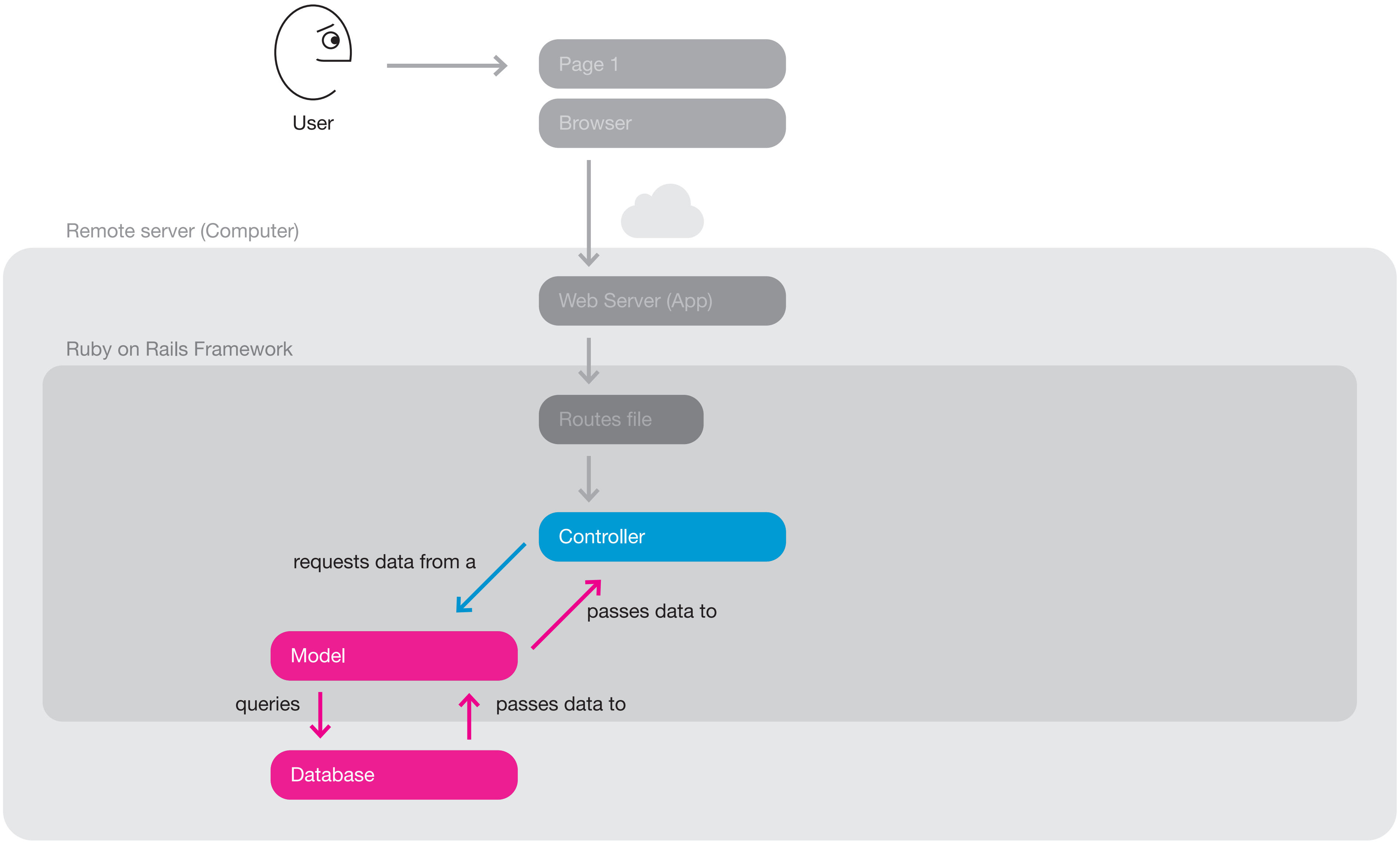


The routes file contains a list of URL patterns.
Each pattern maps to a resource and action.

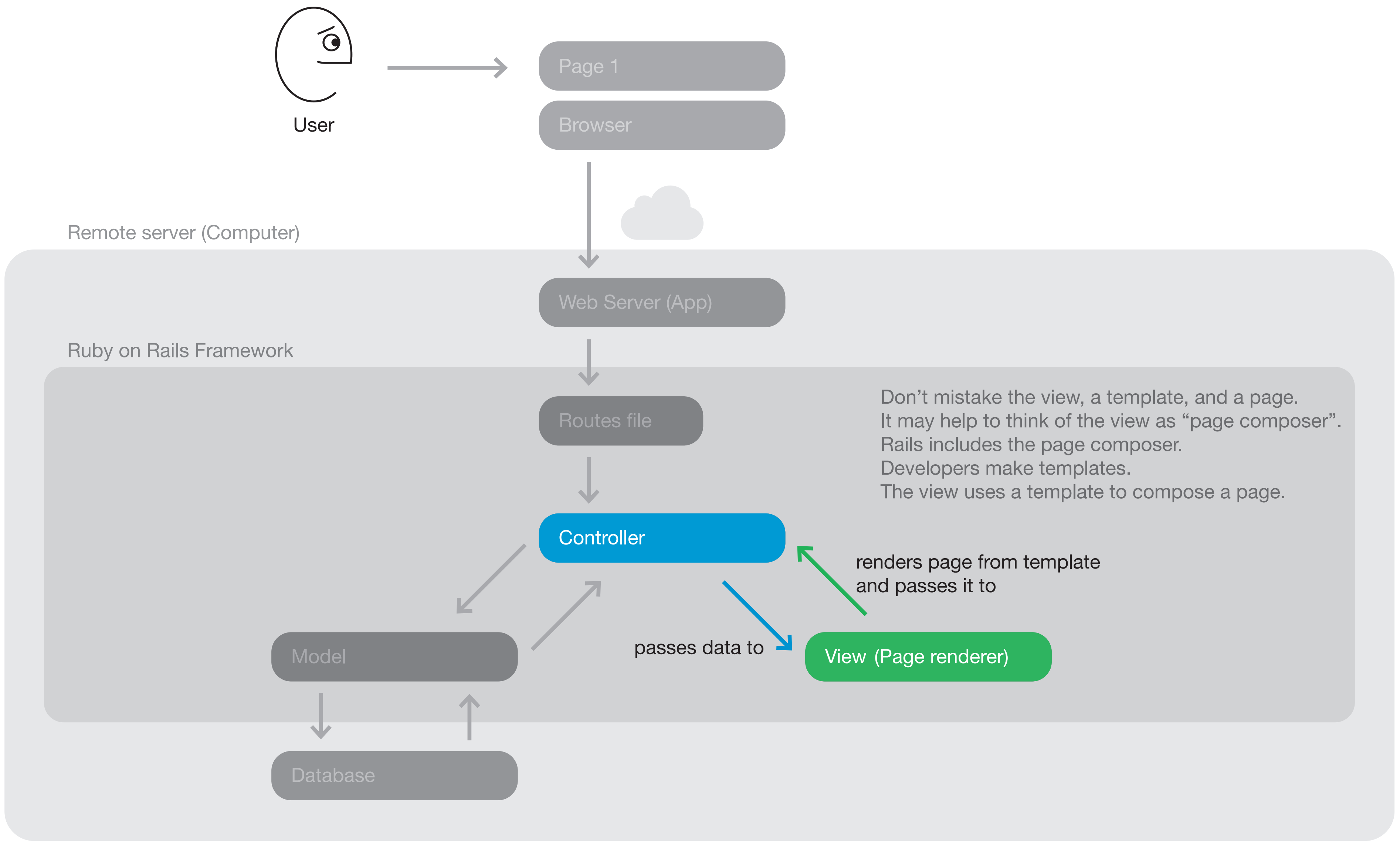
One conventional URL pattern has a server resource, an action to be taken on the resource, and arguments for the action, separated by slashes:
http://domain.com/resource/action/argument

More on resources later.

The invoked controller action requests data from a model. The model queries the database and hands data back to the controller.

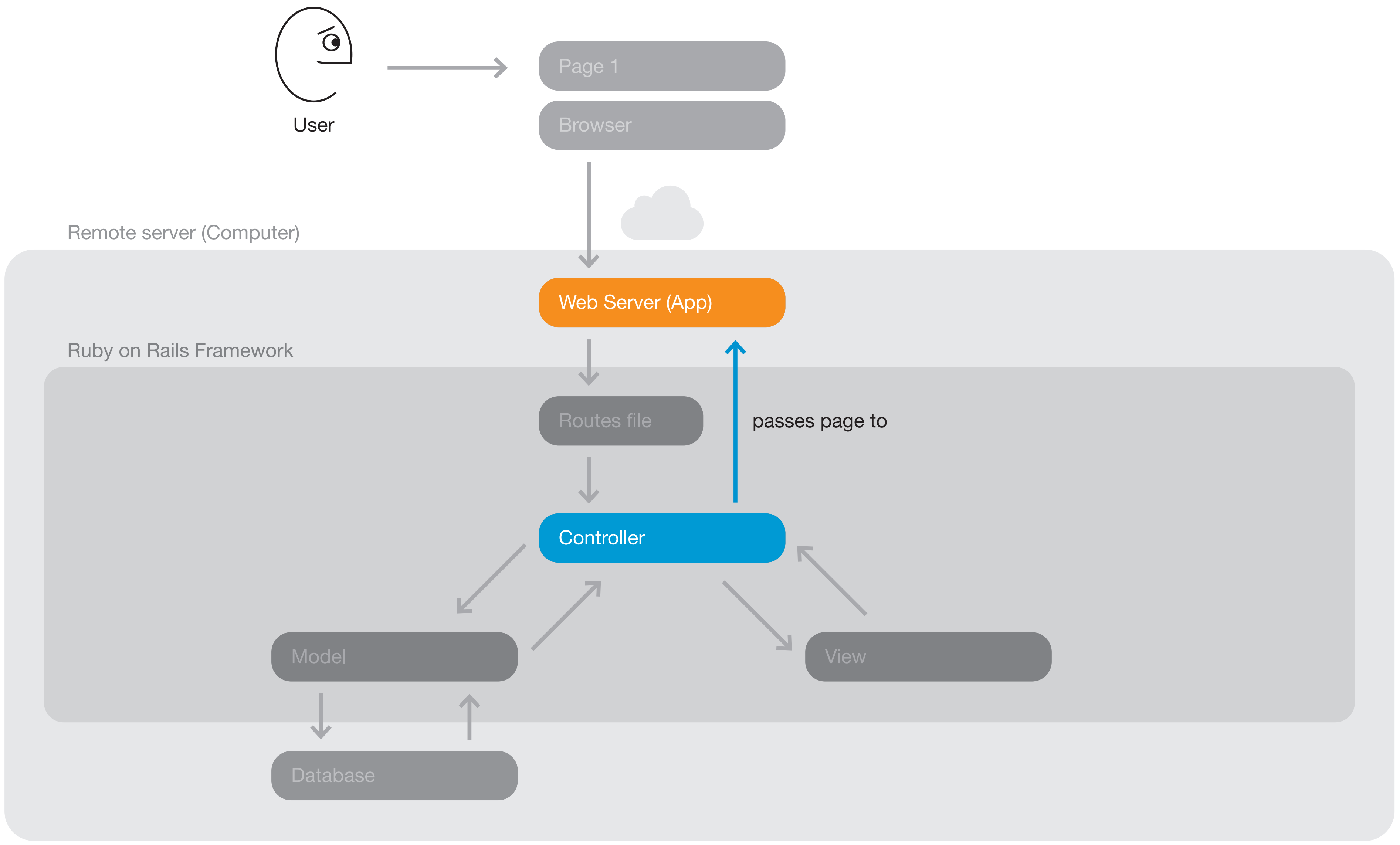


The controller action then passes data to a corresponding view. The view uses the data and a template to compose a page.

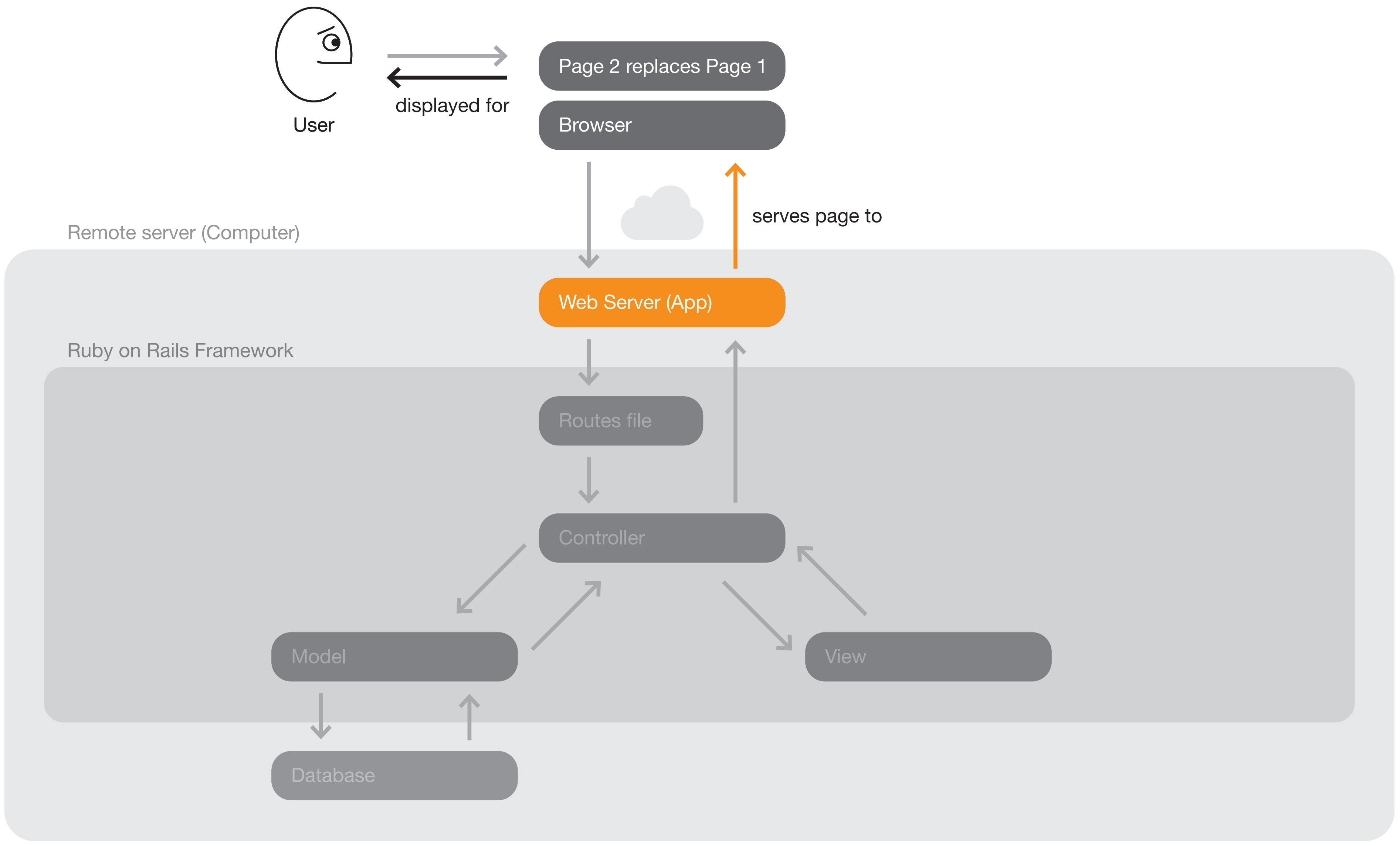


Don’t mistake the view, a template, and a page.
It may help to think of the view as “page composer”.
Rails includes the page composer.
Developers make templates.
The view uses a template to compose a page.

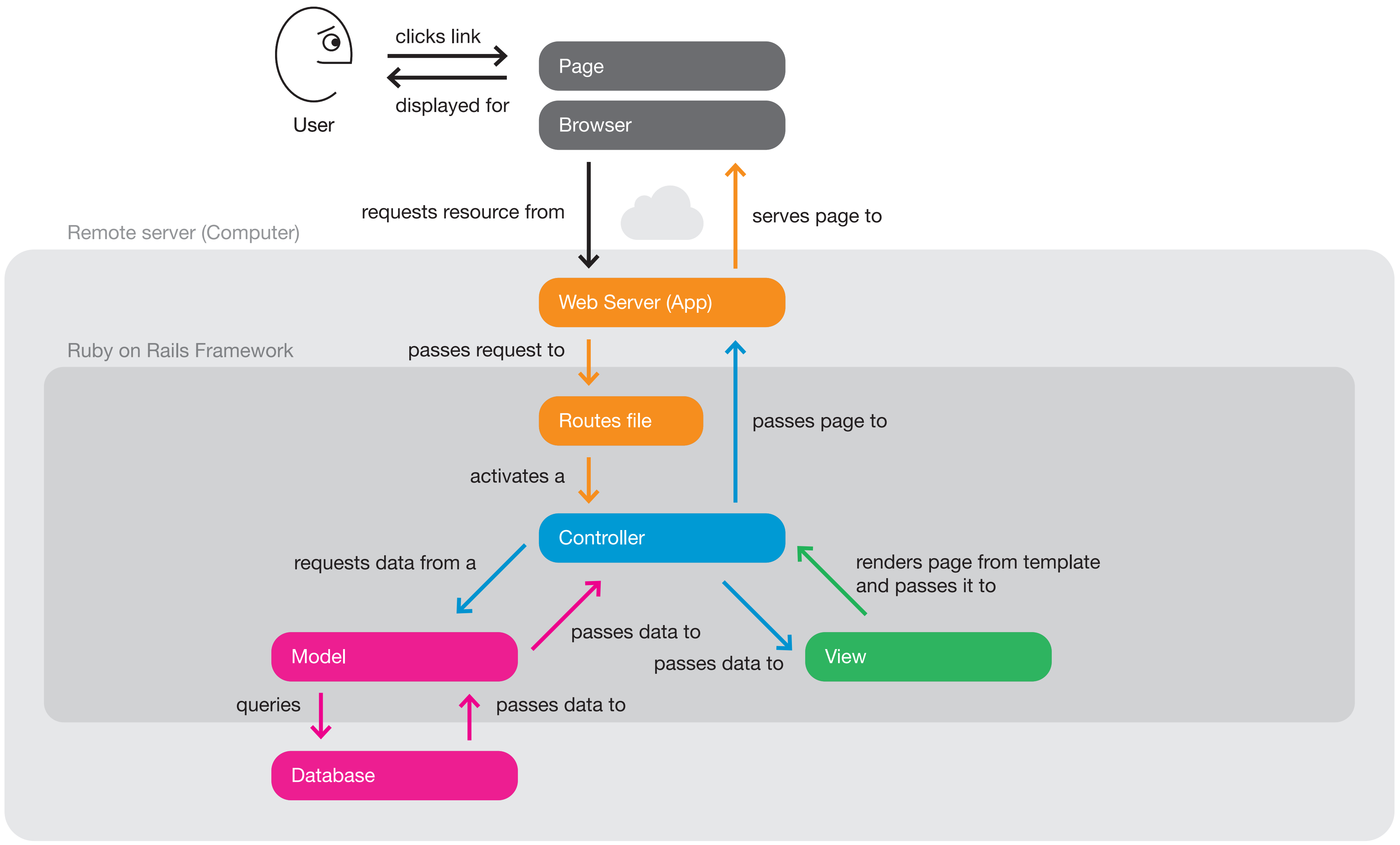
The controller passes the complete page to the web server.



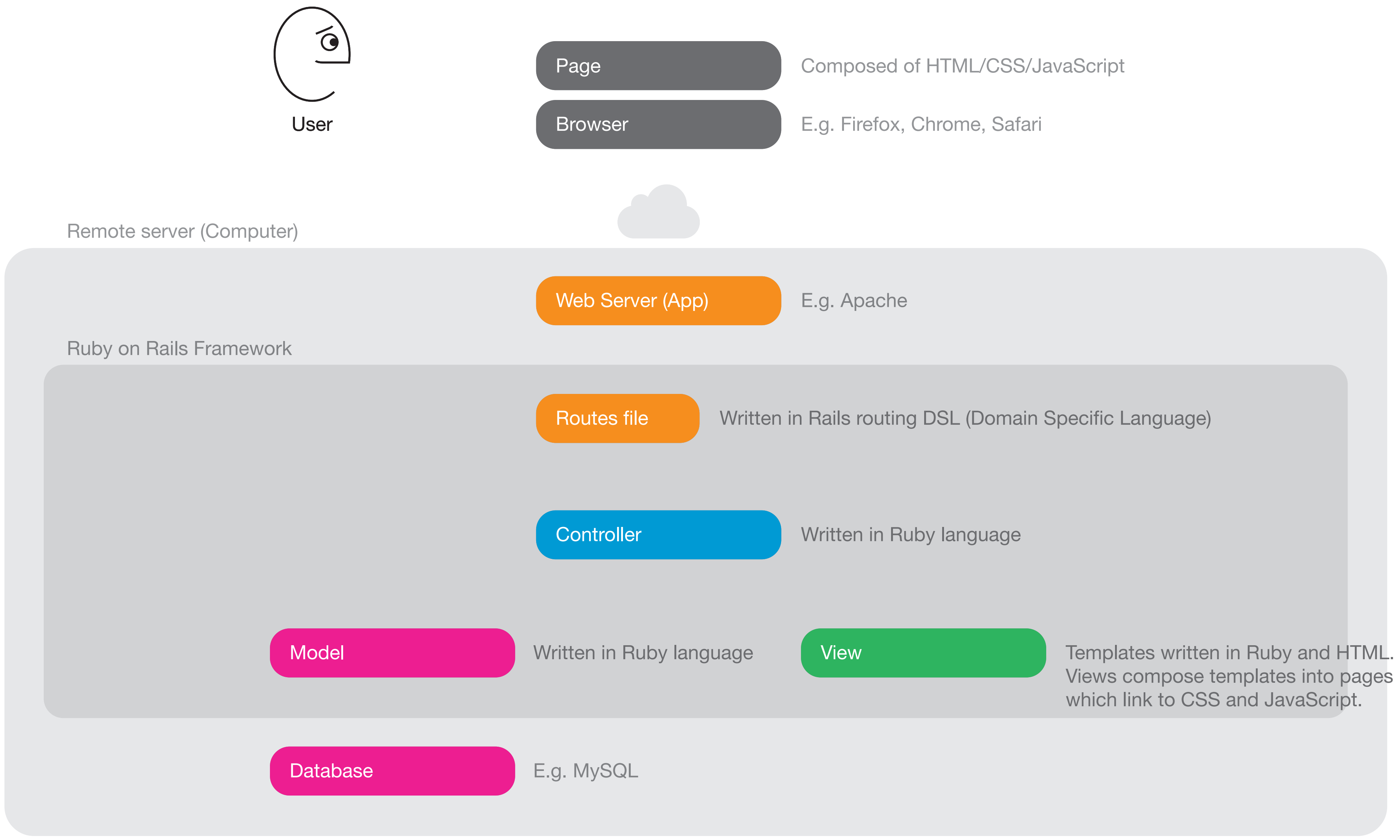
The web server serves the page to the browser.
The browser renders the new page in place of the first one.



Ruby on Rails provides a framework for this MVC flow. It enables developers to work on what makes their apps unique rather than spend time re-implementing conventions.

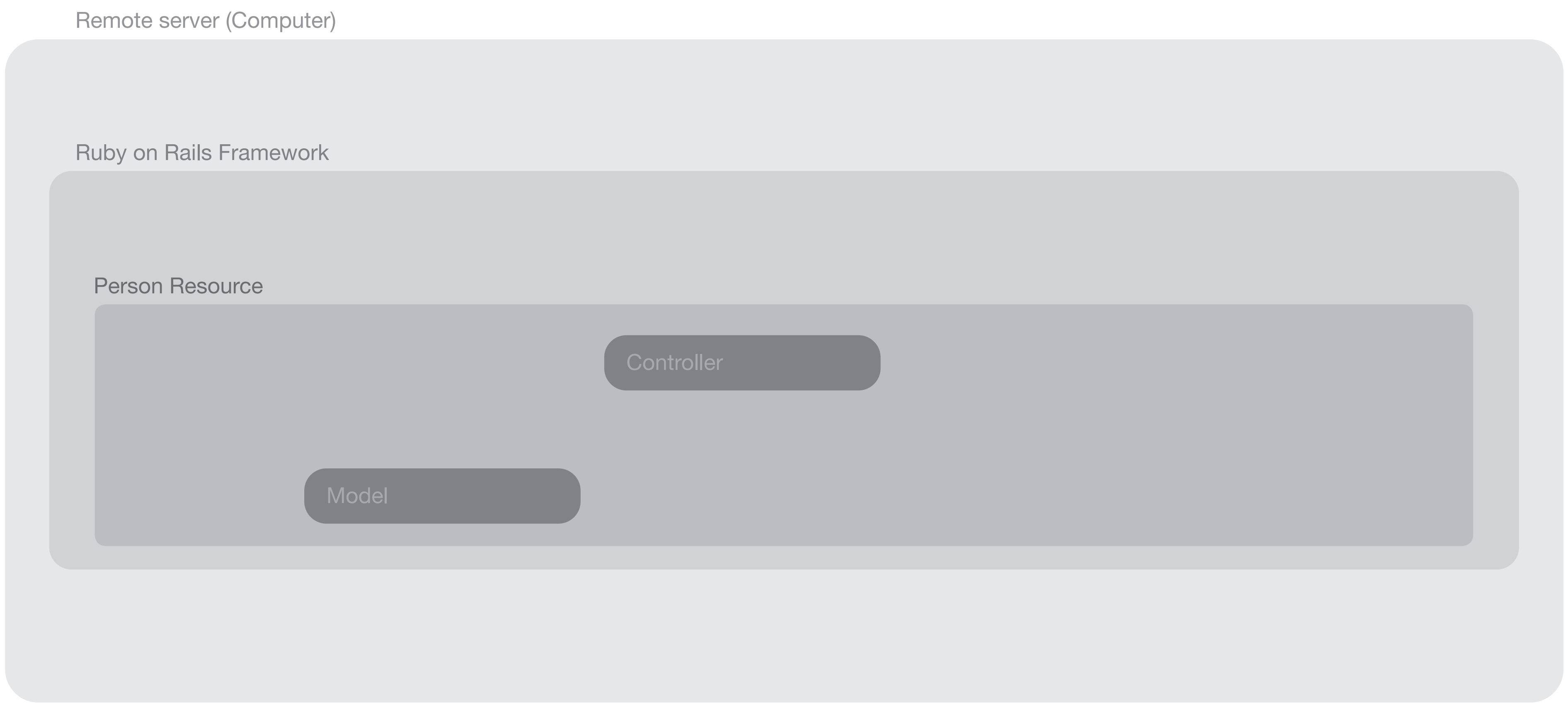


Developers write components in a variety of languages.


User
Page
Composed of HTML/CSS/JavaScript
Browser
E.g. Firefox, Chrome, Safari
Remote server (Computer)
Web Server (App)
E.g. Apache
Ruby on Rails Framework
Routes file
Written in Rails routing DSL (Domain Specific Language)
Controller
Written in Ruby language
Model
Written in Ruby language
View
Templates written in Ruby and HTML. Views compose templates into pages which link to CSS and JavaScript.
Database
E.g. MySQL

A model with a controller and a route is called a resource.

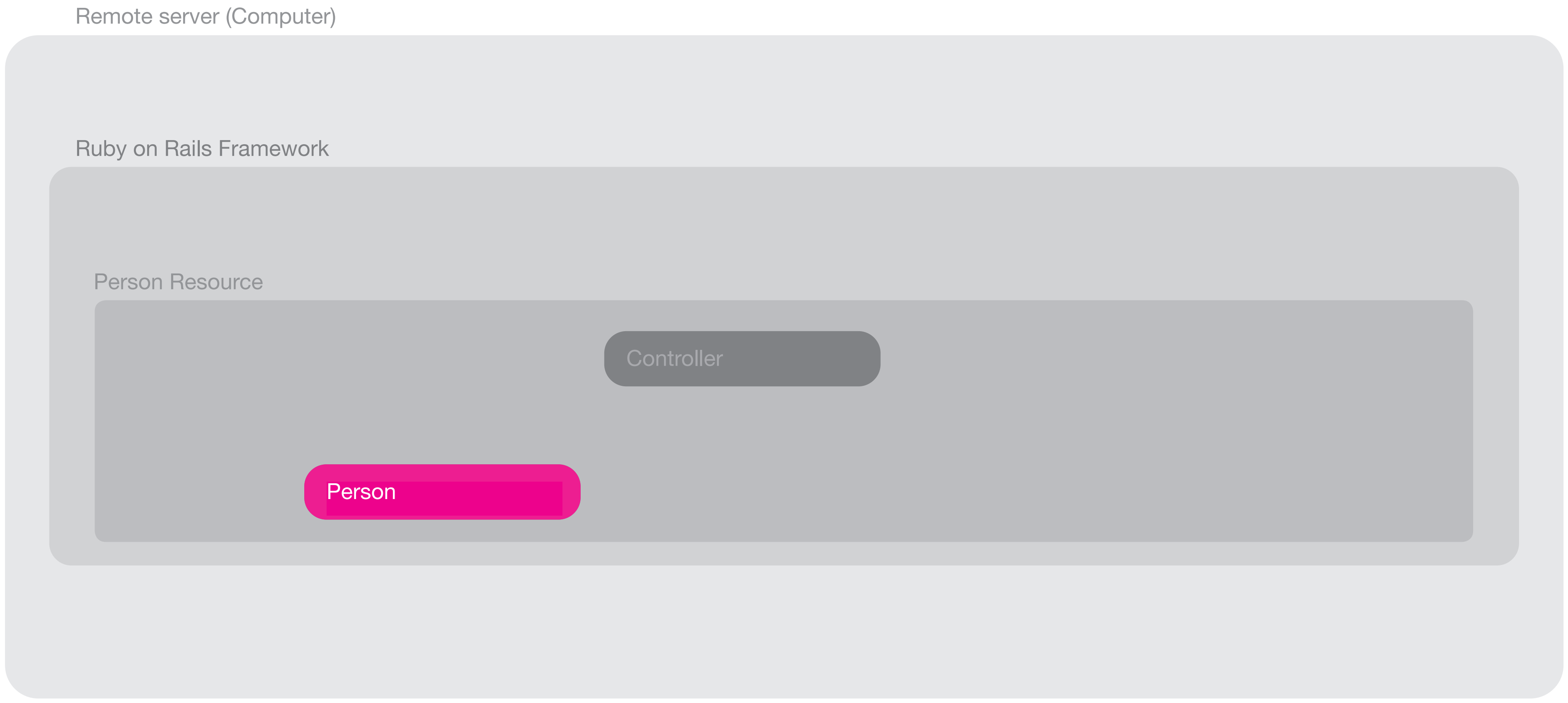
Resources are named with nouns.
In a health-related application you may find resources such as:
Person
Vital
Goal
Prescription



**Attributes (or data) of a resource are accessed through the model.
Attributes (model methods) are also named with nouns.**

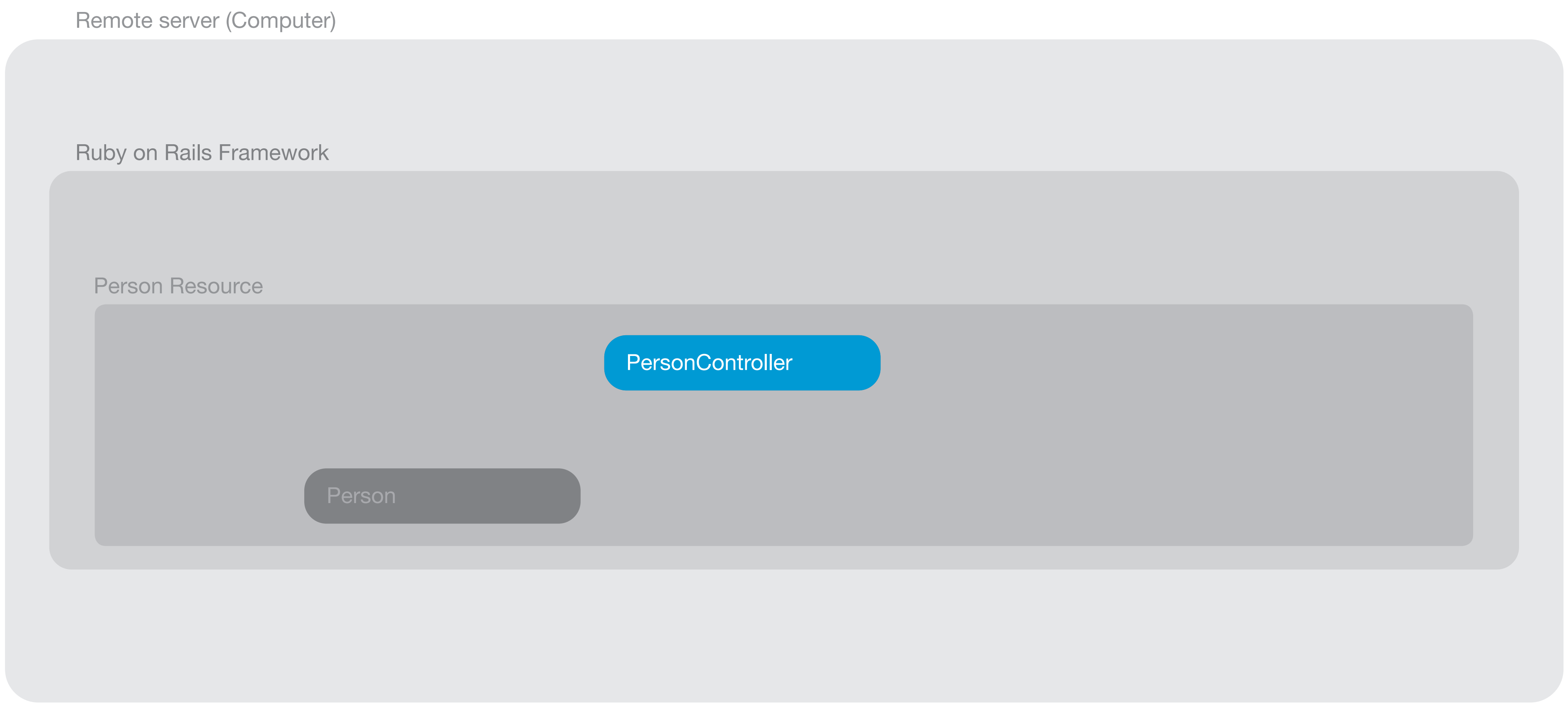
A person model may have methods such as:

Person.date_of_birth
Person.height
Person.gender
Person.allergies



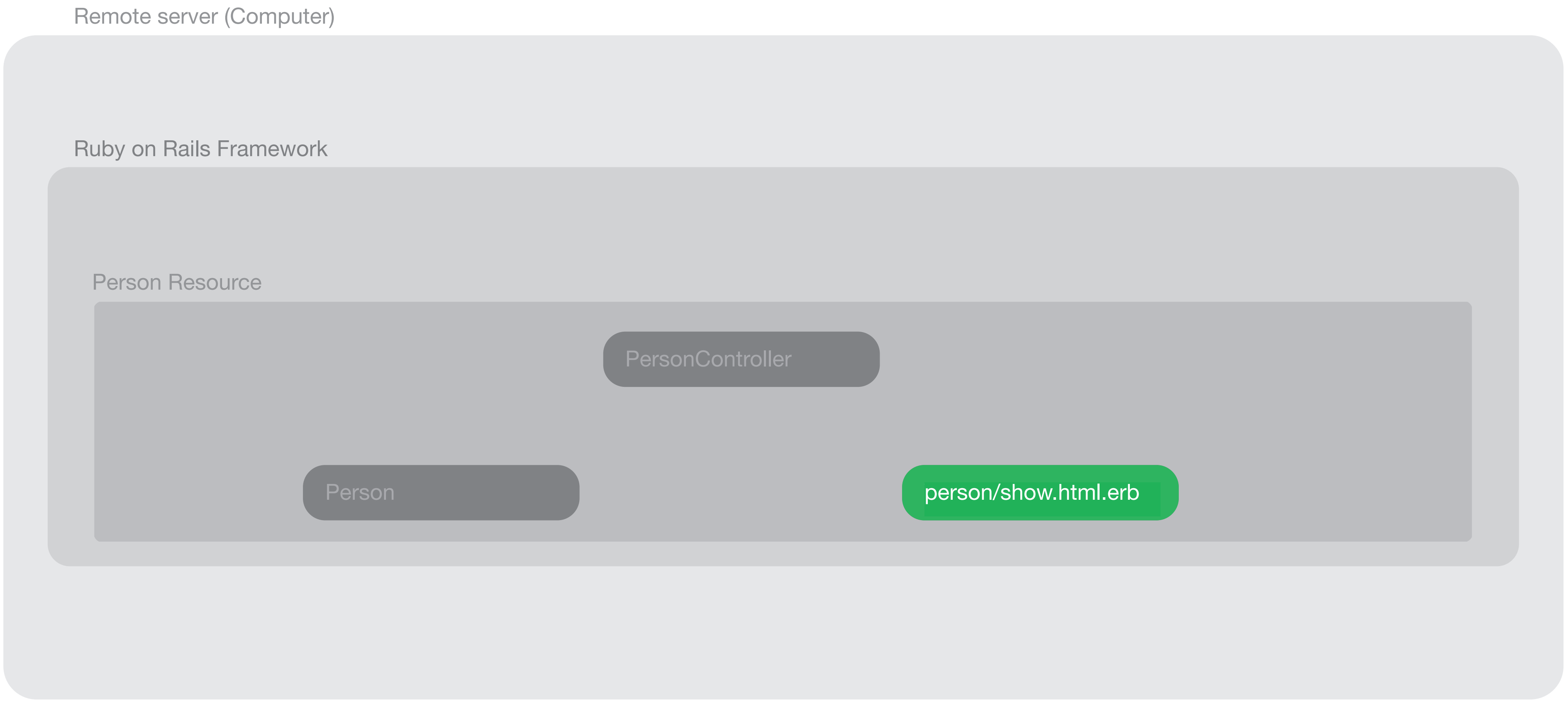
Actions available to a resource are accessed through the controller. Actions (controller methods) are named with verbs.

Controllers typically have one or more standard methods such as:
PersonController.create
PersonController.show
PersonController.update
PersonController.destroy



Resource actions may be visualized in a page, from a view template. Names of view templates correspond to controller methods.

Templates corresponding to standard controller methods are:
person/create.html.erb
person/show.html.erb
person/update.html.erb
person/destroy.html.erb



Responding to an AJAX request

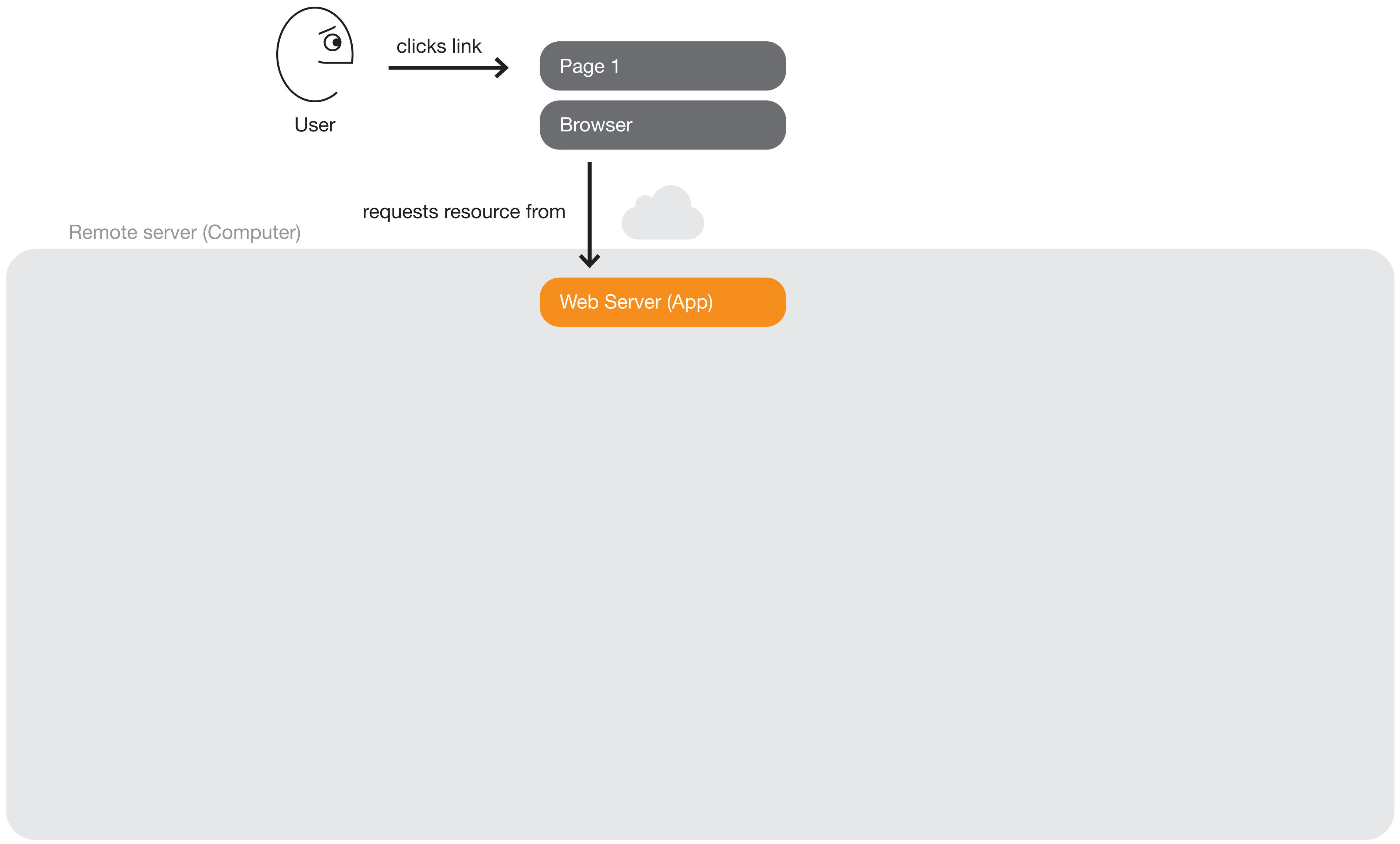
Early web applications sent a page from the server
after every user interaction.

Now, using AJAX (Asynchronous JavaScript and XML) techniques,
it is possible to send and receive data and fragments of pages,
increasing responsiveness.

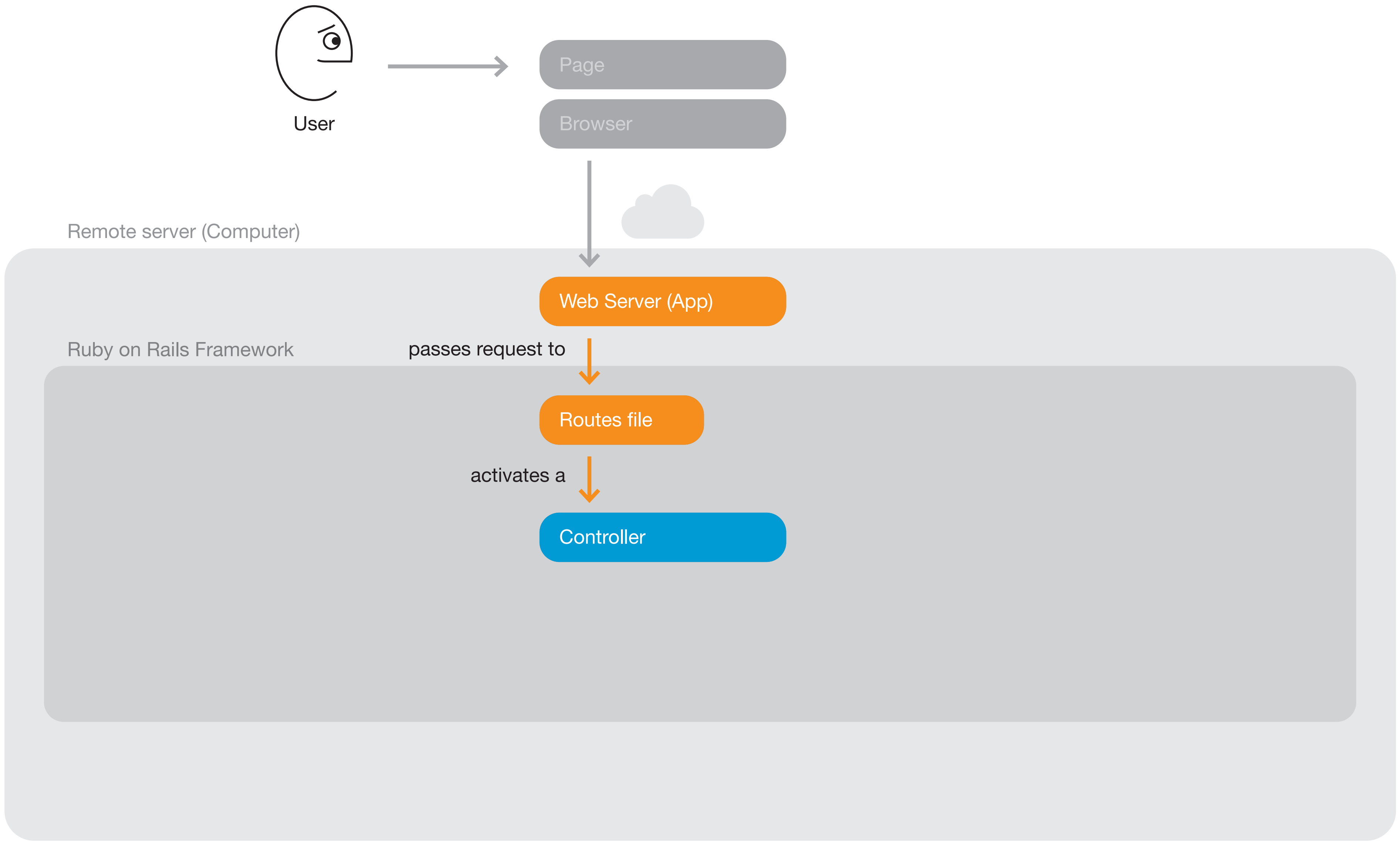
AJAX involves sending or requesting data from a server
“asynchronously”—in the background—
while the user may continue to interact with the page.

The page may be altered when the server responds.
(The “page” also caches some data locally
anticipating possible user interactions.)

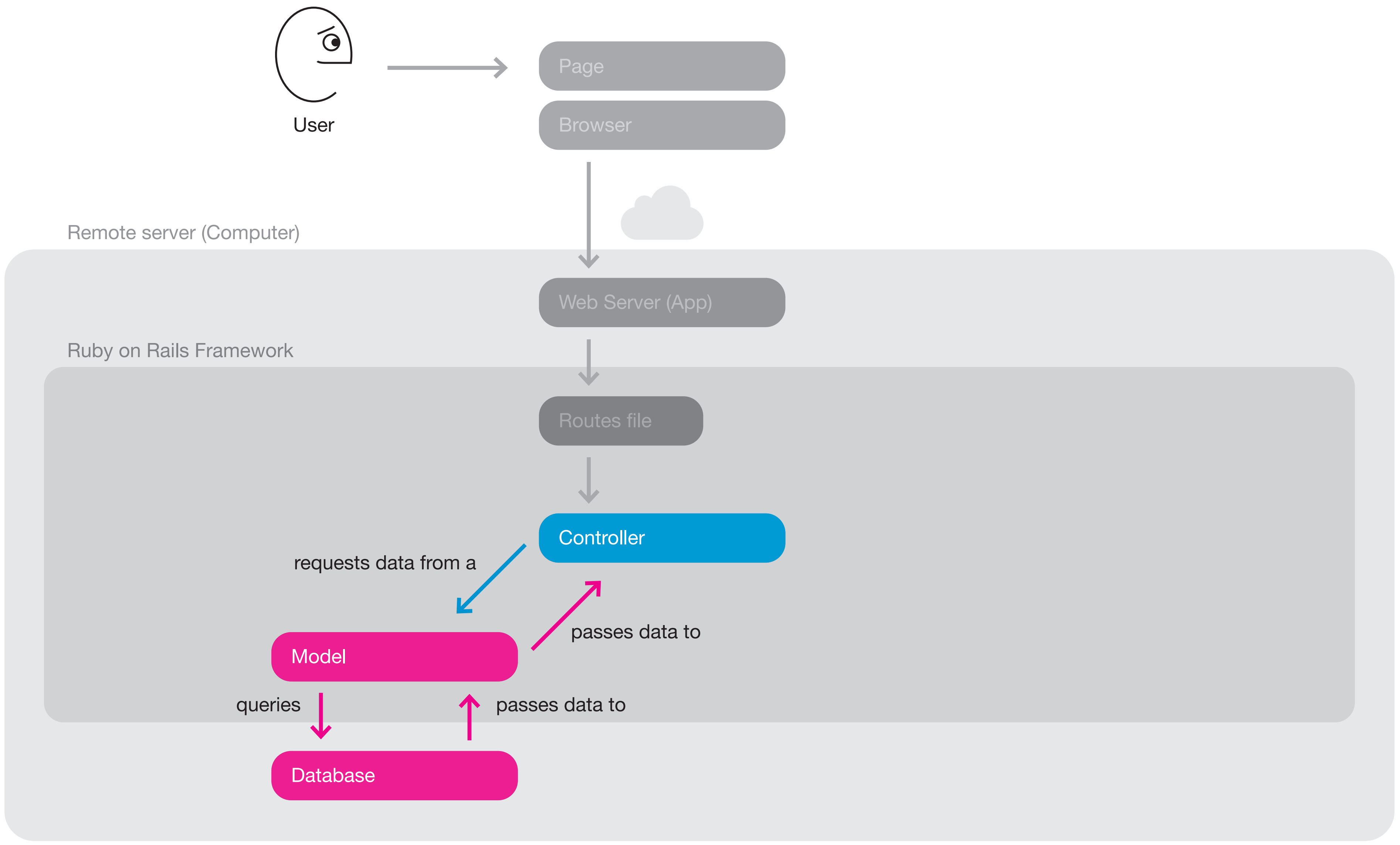
A user action triggers a request for additional data from a server.



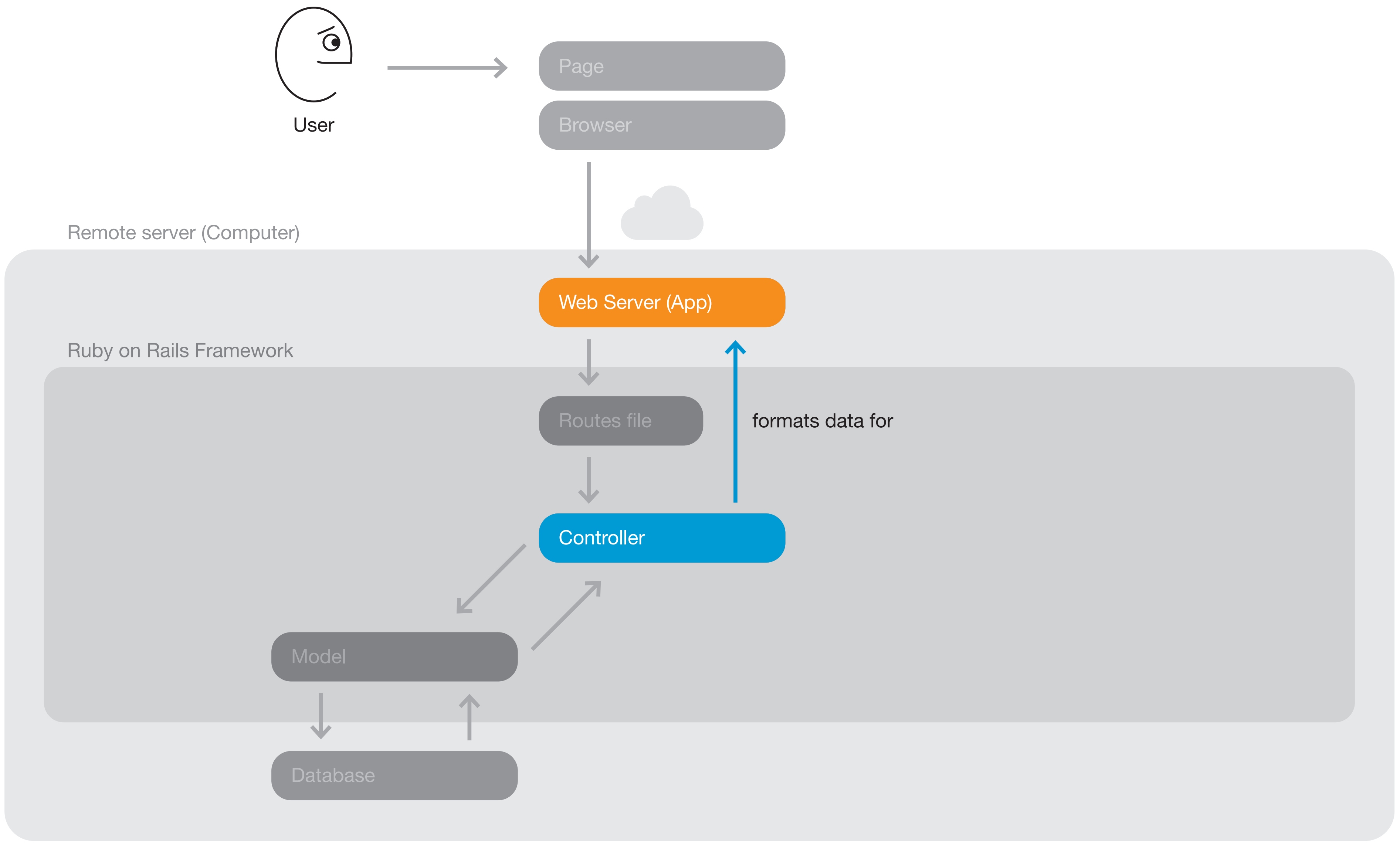
In the same way a page request is processed, the web server uses route definitions to pass the data request to the appropriate controller action.



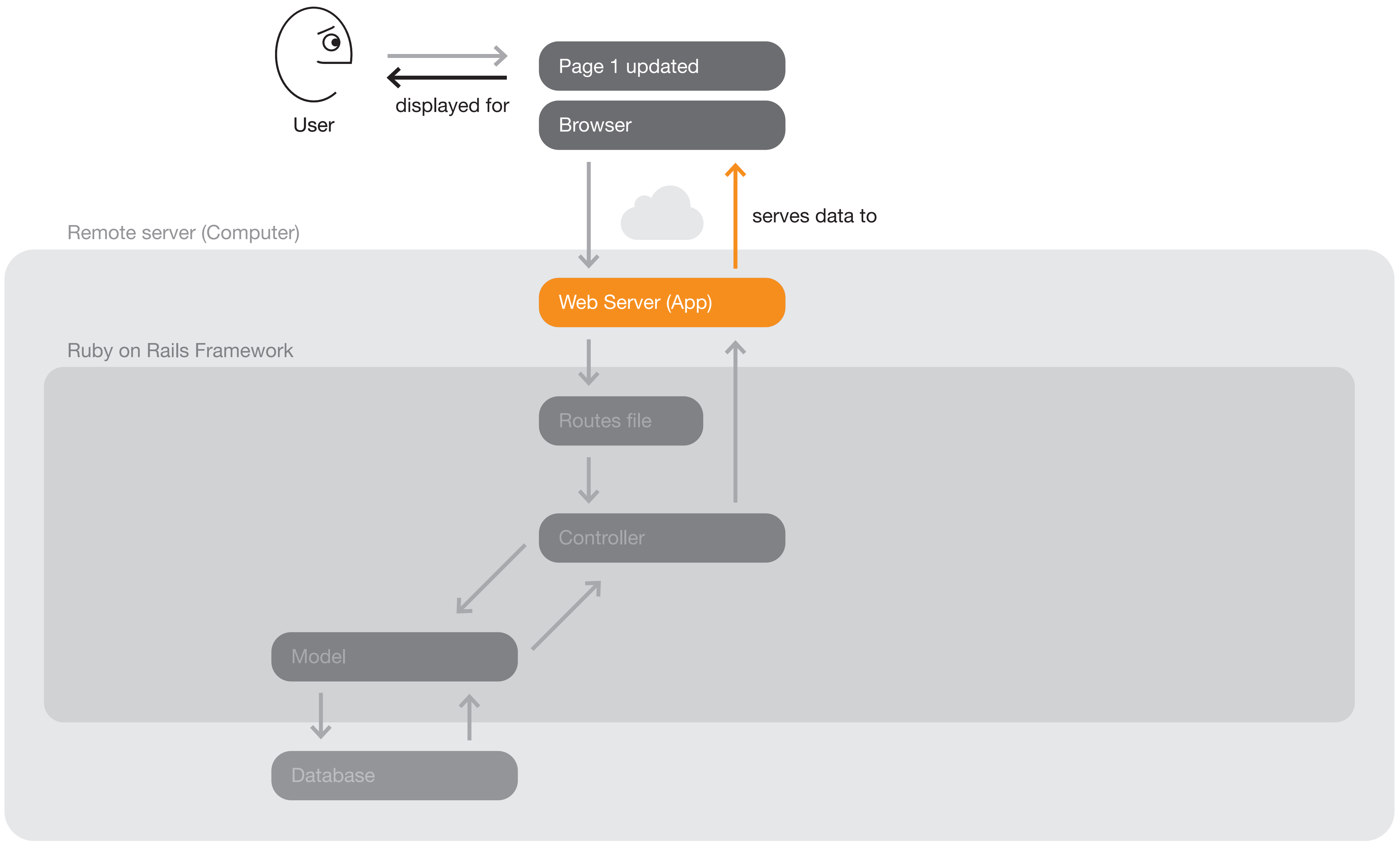
The controller requests data from a model.
The model queries the database and hands data back to the controller.



Rather than rendering a page via a view, the controller formats the data as XML or JSON (JavaScript Object Notation) and hands it to the web server.



The web server’s response is rendered in the current page.


User
displayed for
Page 1 updated
Browser
serves data to
Remote server (Computer)
Web Server (App)
Ruby on Rails Framework
Routes file
Controller
Model
Database