Web Application Architectures

Module 3: Database Interactions
Lecture 5: The Blog App – Iteration 3 (Validations)

THE UNIVERSITY *of* NEW MEXICO

Data Validations in Web Apps

THE UNIVERSITY *of* NEW MEXICO

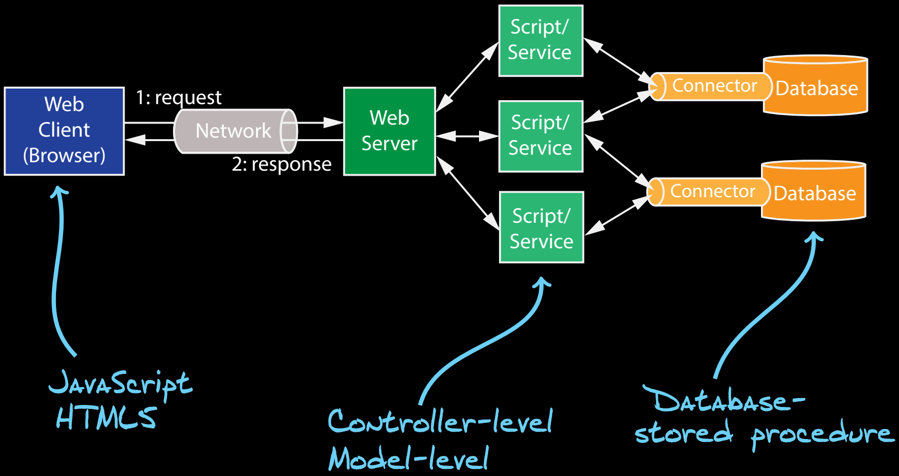
- Data validation is the process of ensuring your web application operates on clean, correct and useful data.

  Ex.

  - Ensure a user provides a valid email address, phone number, etc.
  - Ensure that inputs (505) 255-1234, 505-255-1234 and 5052551234 are all treated the same.
  - Ensure that "business rules" are not being violated, e.g., a comment cannot be stored without a post ID.

- The most common web application security weakness is failure to validate client-side input – SQL injection, cross-site scripting and buffer overflow attacks are enabled.

Data Validations in Web Apps



Data Validations in Web Apps

Client-side – Involves checking that HTML forms are filled out correctly.

- JavaScript, running in the browser, has traditionally been used.
- HTML5 now has more specific "input types" that can be checked, along with a "required" attribute.
- Works best when combined with server-side validations.

Server-side – Checks made after an HTML form has been submitted.

- Database (stored procedures) – Database-dependent, so not portable. Useful if many applications are using the database.
- Controller-level – We'll see later that you don't want to put too much logic in the controller (keep them skinny).
- Model-level – A good way to ensure that only valid data is stored in your database, in a database agnostic way.

`ActiveRecord` Callbacks

- We can think of the objects in an OO system as having a lifecycle – they are first created, can later be updated and also destroyed.
- `ActiveRecord` objects have methods that can be called in order to ensure their integrity at the various stages of their lifecycle.
  Ex.
  - Don't create a new user object if the user already exists in the database.
  - Ensure that all of an object's attributes are valid before allowing it to be saved to the database.
  - When destroying an object, destroy all of the objects that depend on it.
- Callbacks are methods that get called at certain points in an `ActiveRecord` object's lifecycle – they are "hooks" into the lifecycle, allowing you to trigger logic before or after the state of an object changes.

`ActiveRecord` Validations

- Validations are a type of `ActiveRecord` callback that can be used to ensure only valid data is stored in your Rails databases.

- The create, save and update methods trigger validations, and will only allow a valid `ActiveRecord` object to be saved to the database.

- Validations are defined in your models.

  Ex.

```
class Person < ActiveRecord::Base
  validates_presence_of :name
  validates_numericality_of :age, :only_integer => true
  validates_ confirmation_of :email
  validates_length_of :password, :in => 8..20
end
```