CRUD

Types of operations - Create

- Create a new entry
- Must not already exist
- Check within database to avoid conflicts
- Mention mandatory vs optional fields (name, address, mobile number….)

Types of operations - Read

- Get a list of students
- Summarize number of students, age distribution, geographic locations
- Plot histograms of marks

Types of operations - Update

- Change of address
- Update marks
- Change start date of course

Types of operations - Delete

- Remove graduated students
- Delete mistaken entries
- Unenroll student from course

CRUD

Create - Read - Update - Delete

- Originally in context of database operations: nothing to do with the Web
- Reflects cycle of data models
- Databases optimized for various combinations of operations
    - Read-heavy: lots of reading, very little writing or creating
    - Write-heavy: security archive logs

API - Application Programming Interface

- Standardized way to communicate with a server
- Client only needs to know API - not how the server implements the database for example
- CRUD is a good set of functionality for a basic API
  - Usually considered the first level API to implement a web application
- Deals only with the data model life cycle - other control aspects possible

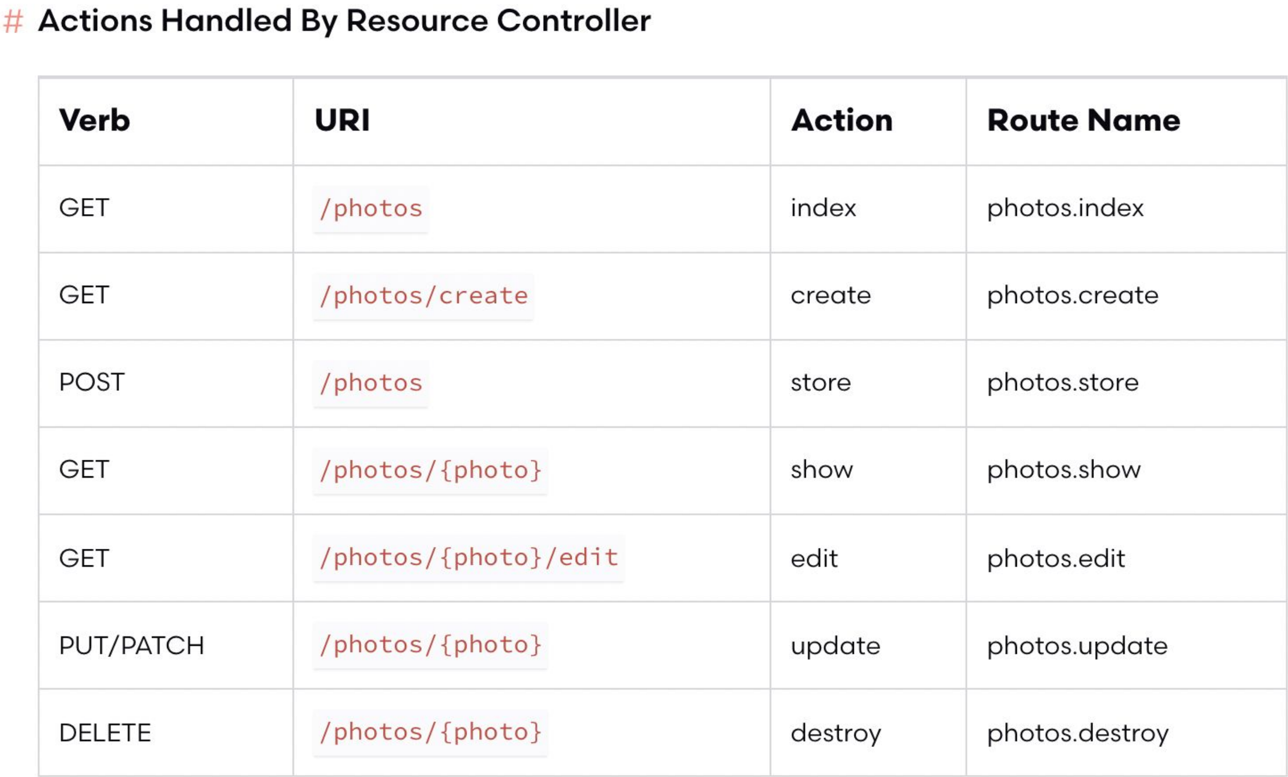
Controllers?

Actions vs Controllers?

- CRUD etc are a set of actions
- Other actions:
    - send email
    - update logs
    - send alert on WhatsApp / Telegram
- Can actions be groups together logically?

Controller!

Example: Laravel PHP framework



Actions Handled By Resource Controller

Verb	URI	Action	Route Name
GET	`/photos`	index	photos.index
GET	`/photos/create`	create	photos.create
POST	`/photos`	store	photos.store
GET	`/photos/{photo}`	show	photos.show
GET	`/photos/{photo}/edit`	edit	photos.edit
PUT/PATCH	`/photos/{photo}`	update	photos.update
DELETE	`/photos/{photo}`	destroy	photos.destroy

Summary

- Actions: interaction between view and model
- Controller: group actions together logically
- API: complex set of capabilities of server
- Interaction through HTTP requests
- HTTP Verbs used to convey meanings

Rules of Thumb

- Should be possible to change views without the model ever knowing
- Should be possible to change underlying storage of model without views every knowing
- Controllers / Actions should generally NEVER talk to a database directly

In practice:

- Views and Controllers tend to be more closely interlinked than with Models
    - More about a way of thinking than a specific rule of design