

;Handlers

```
(defn echo-handler [request]
  (ok (without-str (pp/pprint request))))

(defn weapons-query-handler [{:strs [name]} :params :keys [conn] :as request]
  (ok
    (if name
      (w/weapons @conn [name])
      (w/everybods-weapons @conn))))

(defn add-weapons-handler [{:keys [params conn] :as request}]
  (let [data (map (fn [[k v]] {:name k :weapon v}) params)]
    (do
      (d/transact! conn data)
      (ok (w/weapons @conn (keys params))))))

(defn files-handler [{:keys [sql-conn]}]
  (ok (f/all-processed-files sql-conn)))
```

;Routes - All data

```
(def basic-routes
  ["/echo" {:get echo-handler}]
  ["/files" {:get files-handler}])

(def weapons-routes
  ["/weapons" weapons-query-handler]
  ["/add_weapon" add-weapons-handler])
```

;Router

```
(def router
  (ring/router
```

;Handlers

```
(defn echo-handler [request]
  (ok (without-str (pp/pprint request))))

(defn weapons-query-handler [{:strs [name]} :params :keys [conn] :as request]
  (ok
    (if name
      (w/weapons @conn [name])
      (w/everybods-weapons @conn))))

(defn add-weapons-handler [{:keys [params conn] :as request}]
  (let [data (map (fn [[k v]] {:name k :weapon v}) params)]
    (do
      (d/transact! conn data)
      (ok (w/weapons @conn (keys params))))))

(defn files-handler [{:keys [sql-conn]}]
  (ok (f/all-processed-files sql-conn)))
```

;Routes - All data

```
(def basic-routes
  ["/echo" {:get echo-handler}
   "/files" {:get files-handler}])

(def weapons-routes
  ["/weapons" weapons-query-handler
   "/add_weapon" add-weapons-handler])
```

;Router

```
(def router
  (ring/router
```