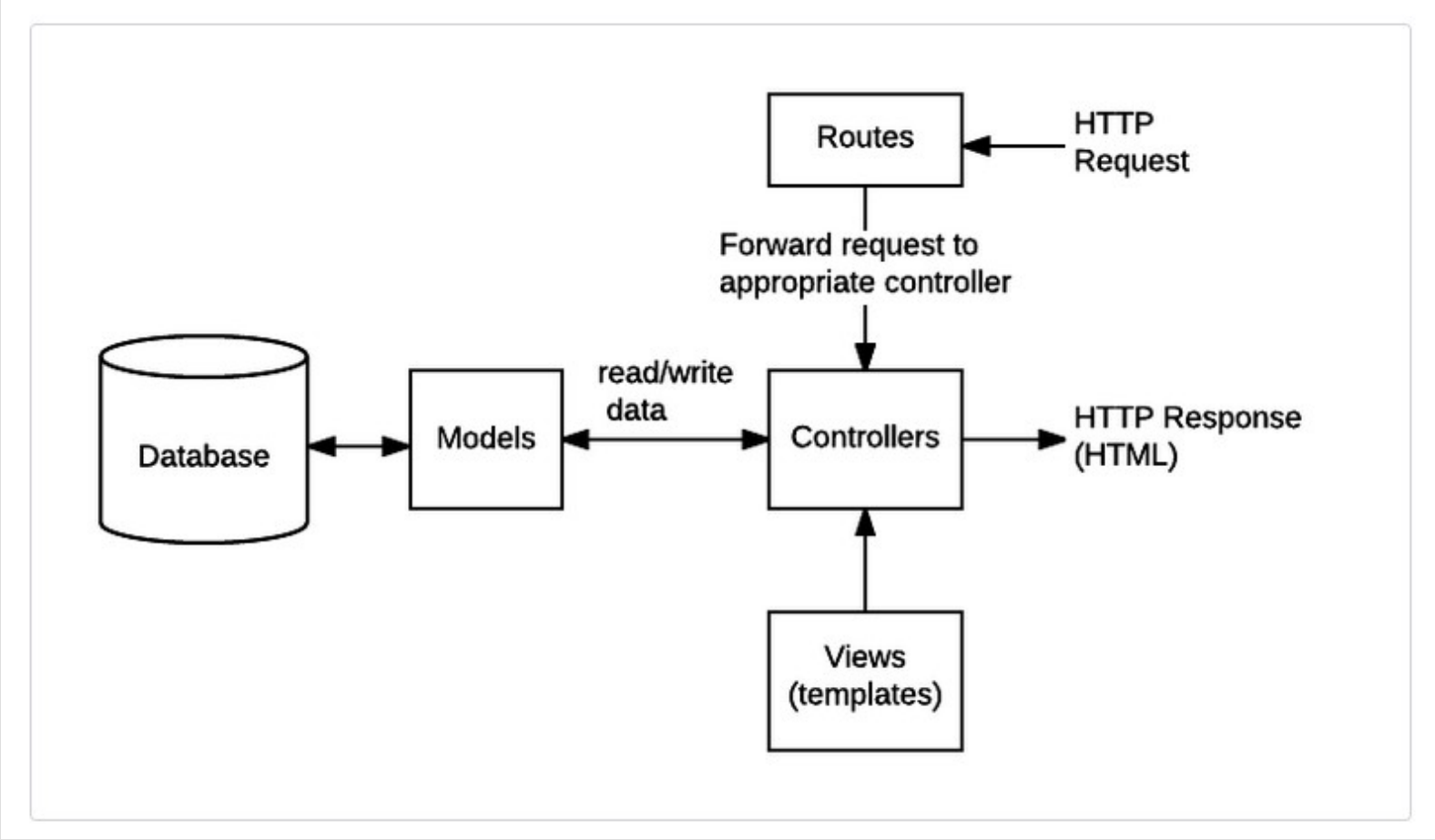
NodeJS : CRUD – Routes



Routeurs

- Les routeurs sont regroupés dans le dossier **/routes**
- Un premier routeur s'occupe des URL des pages à partir de la racine du site :
  fichier **index.js**
- Un second routeur s'occupe des URL des pages qui gèrent les données manipulées dans la base de données :
  fichier **messages.js**

Routeur : index.js

- Lien avec un premier contrôleur
  - Fichier : **root.controller.js**
- Page d’accueil
  - GET ‘/’
- Formulaire de contact – création de message
  - GET ‘/contact’
- Confirmation du message créé
  - POST ‘/traitement’

Router – index

```
var express = require('express');
var router = express.Router();
//var bodyParser = require('body-parser');    // avant Express v4.16

const root = require("../controllers/root.controller.js");

console.log("on passe dans routes/index.js");

// Définition des routes pour gérer les pages à la racine
// Toutes les routes définies dans ce router commencent par l'URL : localhost:8080/

/* GET home page. */
router.get('/', root.home);

/* Afficher le formulaire de contact qui va envoyer les données */
router.get('/contact', root.form);

/* Utiliser le middleware intégré body-parser pour analyser la requête POST */
//var urlencodedParser = bodyParser.urlencoded({ extended: false });
var urlencodedParser = express.urlencoded({extended:false});

/* Recevoir les données et afficher la page */
router.post('/traitement', urlencodedParser, root.traitement);

module.exports = router;
```

Routeur : messages.js

- Lien avec un second contrôleur
  - Fichier : **message.controller.js**
- Lire tous les messages
  - GET ‘/’ (sous forme de tableau, avec possibilité de sélection d’un seul message pour lire, modifier ou supprimer)
  - GET ‘/list’ (sous forme de liste)
- Créer un nouveau message
  - GET ‘/newmsg’ (afficher le formulaire)
  - POST ‘/create’ (écrire les données dans la DB)

Routeur : messages.js

- Lire un seul message à partir de son ID
  - GET ‘/read/:id’ (détails d’un message)
- Mettre à jour un message à partir de son ID
  - GET ‘/edit/:id’ (données existantes)
  - POST ‘/update/:id’ (nouvelles données)
- Supprimer un message à partir de son ID
  - GET ‘/confirm/:id (supprimer ceci ou pas ?)
  - POST ‘/delete/:id’ (effacer dans la DB)

Router – messages – 1ère partie

```
var express = require('express');
var router = express.Router();

const messages = require("../controllers/message.controller.js");

console.log("on passe dans routes/messages.js");

// Définition des routes pour gérer les messages
// Toutes les routes définies dans ce router commencent par l'URL : localhost:8080/messages

// Récupération de tous les messages : ReadAll
// URL => localhost:8080/messages/
router.get('/', messages.readAll);

// Création d'un message
// Enregistrement du nouveau message
// URL => localhost:8080/messages/create
router.post('/create', messages.create);

// Création d'un message
// Affichage du formulaire de création
// URL => localhost:8080/messages/newmsg
router.get('/newmsg', messages.newmsg);

// Lire tous les messages
// Affichage de la liste des messages
// URL => localhost:8080/messages/list
router.get('/list', messages.list);

// Lire un message en fonction de son id
// Affichage des détails d'un message
// URL => localhost:8080/messages/read/:id
router.get('/read/:id', messages.readById);
```

Router – messages – 2ème partie

```
// Mettre à jour un message en fonction de son id
// Récupérer les données actuelles du message
// URL => localhost:8080/messages/edit/:id
router.get('/edit/:id', messages.updateById);

// Mettre à jour un message en fonction de son id
// Sauvegarder les nouvelles données modifiées
// URL => localhost:8080/messages/update/:id
router.post('/update/:id', messages.update);

// Supprimer un message en fonction de son id
// Récupérer les données actuelles du message
// URL => localhost:8080/messages/confirm/:id
router.get('/confirm/:id', messages.deleteById);

// Supprimer un message en fonction de son id
// Effacer le message après confirmation
// URL => localhost:8080/messages/delete/:id
router.post('/delete/:id', messages.delete);

module.exports = router;
```