**TP du 18 12**

Prévoir les routes de notre application qui seront utilisées :

|  |  |  |  |
| --- | --- | --- | --- |
| Document | Route |  |  |
| user | / user | GET | Get tous les users (avec détail) |
|  | / user /:id | GET | Get le user par l’id |
|  | / user /:pseudo | GET | Get le user par le pseudo |
|  | / user /:id/delete | POST | Supp le post id |
|  | / user /:id | PUT | Edit le post id |
|  | / user | POST | Add un post |
|  | / user / :peudo ?:password | GET | Savoir si user existe (pour log in) |
|  |  |  |  |
| post | /post | GET | Get tous les posts (avec détail) |
|  | / post / :pseudo | GET | Get tous les posts de ce user |
|  | / post /:id | GET | Get le post par l’id |
|  | / post /:id/delete | POST | Supp le post id |
|  | / post /:id | PUT | Edit le post id |
|  | / post | POST | Add un post |
| tutorial |  |  |  |
| event |  |  |  |

Format sous MongoDB :

**Cf. diagram ppt**

Dans le app.js

|  |
| --- |
| import routes from ‘./routes’ ;  app.use(‘/’, routes) // spécifier d’utiliser le middleware routes.js |

Dans le routes.js (cf p.22 DB) :

|  |
| --- |
| import default router ;  import routeCars from ‘./Cars ; // classe RouteCar du fichier Cars  let routes = express.router() ;  router.use(‘v1/cars’, routeCars) ; // tjr donner une version ici v1  // …spécifier les chemins |

Dans le dossier Cars/routes.js

|  |
| --- |
| let routes = express.router() ;  router.get(‘/’,get) ; // appel de la fonction get() |

MongoDB schema

|  |
| --- |
| var userSchema = mongoose.Schema({      firstName: String,      lastName: String  }); // recup d’un nom et prenom d’un user en dB  var authorSchema = mongoose.Schema({      \_id: mongoose.Schema.Types.ObjectId,      name: {              firstName: String,          lastName: String      },      biography: String,      twitter: String,      facebook: String,      linkedin: String,      profilePicture: Buffer,      created: {          type: Date,          default: Date.now      }  });// recup avec des schema qui contiennent des ref vers d’autres schema (ici book vers author)  Cf <https://code.tutsplus.com/articles/an-introduction-to-mongoose-for-mongodb-and-nodejs--cms-29527> |

Remarques :

|  |
| --- |
| Création d’une base de données MongoDB : <https://docs.mlab.com/>  {… port} // spread operator qui permet dans un tableau d’éclater notre objet avec copie attribut par attribut. C’est intéressant dans le cas où l’on veut un fonctionnement stateless et la modif d’une seule propriété.  Glitch.me // orienté nodejs et express  Mongoose // ORM js/ librairie pour mongodb  [https://expressjs.com/fr/guide/routing.html //](https://expressjs.com/fr/guide/routing.html%20//) API routes avec express |