**ETAPE 1 : la vue HTML**

<head>

<title>Todo List</title>

</head>

<body>

<div class="container">

<header>

<h1>Todo List Collaborative</h1>

{{> inscription}}

{{> submitTask}}

</header>

{{> tasksList}}

</div>

</body>

<!-- Formulaire d'inscription -->

<template name="inscription">

<h3>Identification</h3>

<form class="form-inline">

<div class="form-group">

<label for="name">Name</label>

<input type="text" id="name" class="form-control" placeholder="Name">

<select id='color' name="color" size="1" class="form-control">

<option value="red">red</option>

<option value="blue">blue</option>

<option value="green">green</option>

<option value="orange">orange</option>

<option value="purple">purple</option>

</select>

<label for="name">Email</label>

<input type="text" id="email" class="form-control" placeholder="Email">

</div>

<button class="btn btn-default">Join</button>

</form>

</template>

<!-- ajout d'une tache -->

<template name="submitTask">

<form id="formTask" class="form-inline">

<div class="form-group">

<label for="libelle">Tache</label>

<input type="text" name="libelle" class="form-control" placeholder="Nouvelle tache" />

<select id='priority' name="priority" size="1" class="form-control">

<option value="1">Critique</option>

<option value="2">Majeure</option>

<option value="3">Mineure</option>

</select>

<button type="submit" id="submit" class="btn btn-default">Submit</button>

</div>

</form>

</template>

<!-- liste des taches -->

<template name="tasksList">

<table class="table table-striped table-bordered">

<thead>

<tr>

<th>

A faire

</th>

<th>

Developpement

</th>

<th>

Validation

</th>

<th>

Terminée

</th>

</tr>

</thead>

<tbody>

<!-- TODO -->

</tbody>

</table>

</template>

**ETAPE 2 : Mise en place d’un CRUD**

* Déclaration des collections Mongo : collections.js

Tasks = new Mongo.Collection("tasks");

Users = new Mongo.Collection("users");

**Enregistement base Users**

if (Meteor.isClient) {

Template.inscription.events({

'click .btn': function(event) {

event.preventDefault();

var name = $('#name').val();

var color = $('#color').val();

var email = $('#email').val();

user = Users.insert({

name: name,

color: color,

email: email,

createdAt: Date.now()

}, function(error, result) {

if (result) {

console.log("New User : " + name);

}

});

}

});

}

Vérification de l’ajout via la console chrome : Users.find({}).fetch();

**Enregistement base Tasks**

Template.submitTask.events({

"submit #formTask": function(event) {

// Prevent default browser form submit

event.preventDefault();

// Get value from form element

var libelle = event.target.libelle.value;

var priority = event.target.priority.value;

var user = currentUserName;

var color = currentUserColor;

// Insert a task into the collection

Tasks.insert({

libelle: libelle,

priority: priority,

user: user,

color: color,

// afaire: true,

createdAt: new Date() // current time

});

console.log("New tache : " + libelle);

// Clear form

event.target.libelle.value = "";

},

});

}

**ETAPE 3 : Affichage de la liste des taches**

**Vue HTML : détailler les {{}}dans le contexte de la boucle**

….

<tbody>

{{#each tasks }}

<tr>

<td>

<div style="color:{{color}}">

<span class="text">{{priority}} - <b>{{libelle}}</b> - <i> {{user}}</i></span>

</div>

</td>

</tr>

{{/each}}

</tbody>…..

**Javascript : ajout d’un helpers**

Template.tasksList.helpers({

tasks: function() {

return Tasks.find({}, {

sort: {

priority: 1,

createdAt: -1

}

});

}

});

**ETAPE 4 : Gestion état des taches**

**Modification HTML : ajout des bouton**

<tbody>

{{#each tasks }}

<tr>

<td>

{{#if afaire}}

<div style="color:{{color}}">

<span class="text">{{priority}} - <b>{{libelle}}</b> - <i> {{user}}</i></span>

<button class="delete">&times;</button>

<button class="moveEnCours">&gt;</button>

</div>

{{/if}}

</td>

<td>

{{#if encours}}

<div style="color:{{color}}">

<span class="text">{{priority}} - <b>{{libelle}}</b> - <i> {{user}}</i></span>

<button class="moveAfaire">&lt;</button>

<button class="moveAvalider">&gt;</button>

</div>

{{/if}}

</td>

<td>

{{#if avalider}}

<div style="color:{{color}}">

<span class="text">{{priority}} - <b>{{libelle}}</b> - <i> {{user}}</i></span>

<button class="moveEnCours">&lt;</button>

<button class="moveTerminee">&gt;</button>

</div>

{{/if}}

</td>

<td>

{{#if terminee}}

<div style="color:{{color}}">

<span class="text">{{priority}} - <b>{{libelle}}</b> - <i> {{user}}</i></span>

<button class="moveAvalider">&lt;</button>

<button class="delete">&times;</button>

</div>

{{/if}}

</td>

</tr>

{{/each}}

</tbody>

**Modification JS : ajout etat de tache + events sur les boutons (penser a mettre l’etat sur l’event du submit task)**

Template.task.events({

"click .delete": function() {

Tasks.remove(this.\_id);

},

"click .moveAfaire": function() {

Tasks.update(this.\_id, {

$set: {

afaire: true,

encours: false,

avalider: false,

terminee: false

}

});

},

"click .moveEnCours": function() {

Tasks.update(this.\_id, {

$set: {

afaire: false,

encours: true,

avalider: false,

terminee: false

}

});

},

"click .moveAvalider": function() {

Tasks.update(this.\_id, {

$set: {

afaire: false,

encours: false,

avalider: true,

terminee: false

}

});

},

"click .moveTerminee": function() {

Tasks.update(this.\_id, {

$set: {

afaire: false,

encours: false,

avalider: false,

terminee: true

}

});

}

});

**ETAPE 5 : Notion routage**

Tout mettre dans un template ‘home’

<template name="home">

<head>

<title>tuto</title>

</head>

<body>

<div class="row">

<div class="col-md-12 text-center">

<h1>Waiting List</h1>

{{> form}}

</div>

</div>

<div class="row" style="margin-top:30px">

<div class="col-md-6 col-md-offset-3">

{{> waiters}}

</div>

</div>

</body>

</template>

Refresh -> Il ne se passe rien

meteor add iron:router

Refresh -> Infos pour definir une route

*router.js*

Router.route('/', function() {

this.render('home');

});

**ETAPE 5 : Affichage des formulaire - Session**

Par exemple, un systeme d’aunthentification. Une fois authentifier, le bandeau n’apparait plus.

Sol 1 : les Sessions. Prob = si je refresh mon navig, celle ci est re initialisée.

* Dans le code js, lors de l’inscription, ajouter :

Session.set("userId", result);

* Et ajouter un helpers sur le home :

Template.home.helpers({

isLogged: function() {

if (Session.get("userId")) {

var currentUser = Users.findOne({

\_id: Session.get("userId")

});

if (currentUser) {

console.log("user courant : " + currentUser);

return {

\_id: currentUser.\_id,

name: currentUser.name,

color: currentUser.color

}

}

} else {

return null

}

}

});

* Dans html, dans le template home :

<header>

<h1>Todo List Collaborative</h1>

{{#if isLogged}}

Bienvenue {{isLogged.name}}

{{> submitTask}}

{{else}}

{{> inscription}}

{{/if}}

</header>

Sol2 : pakage localstorage -> persistent session qui est en fait une extension (surcharge) a la session de base.

rechercher sur atmospherejs : persistent-session

* meteor add u2622:persistent-session

Puis changer par setPersitent

**ETAPE 6 : notion de context**

{{#with isLogged }}

Tu es deja inscrit {{name}}

{{/with}}

Plus besoin de préfixer name, puisque le context courant est celui passé dans le with

**ETAPE 7 : La sécurité**

TODO a partir de 55 min dans video meteor youtubehttps://www.youtube.com/watch?v=JRLP91sGq0I