

TEST du site web, réalisés par Bastien BALMES

Tous les tests ont été effectués en utilisant l'outil Cypress, un framework de test automatisé populaire pour les applications web, qui permet de vérifier le bon fonctionnement des applications en simulant l'interaction de l'utilisateur avec le navigateur..

1. Connexion	2
Connexion avec compte client :	2
Connexion avec mot de passe incorrect :	3
Modifier les informations du compte :	4
2. CRUD vacataires	7
Création d'un vacataires :	7
Suppression d'un vacataire :	9
Lire les informations d'un vacataire :	10
Modifier les informations d'un vacataire :	11
3. CRUD modules	12
Création d'un module :	13
Suppression d'un module :	14
Lire les informations d'un module :	15
Modifier les informations d'un module:	15
4. Filtre vacataires	17
Filtre vacataire affecté :	18
Filtre vacataire non affecté :	19
Filtre vacataire en attente :	20
Filtre par matière :	21
Recherche d'un vacataire en fonction de son nom :	22
5. Filtre modules	23
Filtre par matière :	23
Filtre par département:	24

1. Connexion

Dès que l'utilisateur arrive sur le site, il est accueilli par un formulaire de connexion, où il peut élégamment insérer son identifiant et son mot de passe pour accéder à son compte.

Test	Attendu	Résultat
Connexion avec compte client	Redirection vers la page vacataires	Ok
Connexion avec mdp incorrect	Redirection vers formulaire de connexion avec message d'erreur explicite	Ok
Modifier les informations du compte	Change les informations du compte	Ok

Connexion avec compte client :


```
it('Connexion', () => {  
  cy.visit('/')  
  cy.get('#pseudo').type('admin');  
  cy.get('#password').type('testps');  
  
  // Vérifiez que les champs contiennent les données saisies  
  cy.get('#pseudo').should('have.value', 'admin');  
  cy.get('#password').should('have.value', 'testps');  
  
  cy.contains('Connexion').click();  
})
```

Dans ce test, nous allons saisir les véritables informations de connexion d'un utilisateur "admin", avec "testps" comme mot de passe. Ensuite, nous vérifierons que les champs d'identification et de mot de passe contiennent effectivement les valeurs "admin" et "testps". Enfin, nous cliquerons sur l'élément contenant le libellé "Connexion". Si les informations fournies sont correctes, nous serons dirigés vers la page de l'espace vacataire.

Connexion avec mot de passe incorrect :

```
it('Erreur dans les informations', () => {  
  cy.visit('/')  
  cy.get('#pseudo').type('admin');  
  cy.get('#password').type('testpp');  
  
  cy.contains('Connexion').click();  
  cy.wait(1000);  
  cy.contains('Identifiants invalides').click();  
});
```

Dans ce scénario de test, nous allons délibérément entrer un mot de passe incorrect. Par conséquent, nous anticipons la présence d'un message sur la page indiquant "Identifiants invalides" et aucune connexion réussie.



Identifiant

Mot de passe

Connexion

Identifiant

admin

Mot de passe

.....

Identifiants invalides

Connexion

Modifier les informations du compte :

Dans ce cas de test, nous allons procéder à la modification du mot de passe du compte, puis vérifier que la connexion avec l'ancien mot de passe "testps" n'est plus possible, tandis qu'avec le nouveau mot de passe "testpp", la connexion s'effectue avec succès.

On modifie les informations du compte :

```
it('Modifier information compte', () => {
  cy.visit('/')
  cy.get('#pseudo').type('admin');
  cy.get('#password').type('testps');

  cy.contains('Connexion').click();
  cy.wait(1000);
  cy.contains('Profil').click();
  cy.get('#password').type('testps');
  cy.get('#newPassword').type('testpp');
  cy.get('.btn-primary').click();
});
it('Vérification mot de passe changer', () => {
  cy.visit('/')
  cy.get('#pseudo').type('admin');
  cy.get('#password').type('testps');

  cy.contains('Connexion').click();
  cy.wait(1000);
  cy.contains('Identifiants invalides').click();
});
```

Mot de passe actuel

Nouveau mot de passe

Changer le mot de passe

Déconnexion

Puis on essaye de se connecter avec l'ancien mot de passe :

```
it('Vérification mot de passe changer', () => {  
  cy.visit('/')  
  cy.get('#pseudo').type('admin');  
  cy.get('#password').type('testps');  
  
  cy.contains('Connexion').click();  
  cy.wait(1000);  
  cy.contains('Identifiants invalides').click();  
});
```

Identifiant

Mot de passe

Identifiants invalides

Connexion

Enfin nous allons tenter la connexion en utilisant le nouveau mot de passe. L'objectif est d'être redirigé vers la page des vacataires et de visualiser la chaîne de caractères "Bastien", qui correspond à un vacataire existant :

```
it('Connexion nouveau mot de passe', () => {  
  cy.visit('/')  
  cy.get('#pseudo').type('admin');  
  cy.get('#password').type('testps');  
  
  cy.get('#pseudo').should('have.value', 'admin');  
  cy.get('#password').should('have.value', 'testps');  
  cy.contains('Connexion').click();  
  cy.wait(1000);  
  cy.contains('Bastien');  
});
```



Bastien Balmes
Vacataire n°1
bastien@gmail.com
Skills : Communication
Modules :



2. CRUD vacataires

Test	Attendu	Résultat
Création d'un vacataires	Un nouveau vacataire apparaît dans la liste des vacataires	Ok
Suppression d'un vacataires	Le vacataire choisie pour être supprimer n'apparaît plus dans la liste des vacataires	Ok
Lire les informations d'un vacataires	Ouvre un modal avec les informations du vacataires	Ok
Modifier les informations d'un vacataires	Change les informations du vacataire choisi	Ok

Création d'un vacataires :

Dans cette procédure de test, nous allons créer un vacataire. Pour ce faire, nous allons cliquer sur le bouton "Ajouter un vacataire", ce qui ouvrira un formulaire dans une fenêtre modale permettant de créer un nouveau vacataire. Ensuite, nous allons saisir les informations requises dans le formulaire, le valider, puis vérifier la présence de la chaîne de caractères "Bastien" sur notre page, correspondant ainsi au prénom du nouveau vacataire créé.

```

it('Ajouter vacataire', () => {

  // Cliquez sur le bouton "Ajouter un vacataire"
  cy.get('[data-bs-toggle="modal"]').click();

  // Saisissez les informations dans le formulaire
  cy.get('#name').type('Bastien');
  cy.wait(100);
  cy.get('#lastName').type('Balmes');
  cy.wait(100);
  cy.get('#phone').type('060606060606');
  cy.wait(100);
  cy.get('#email').type('bastien@gmail.com');
  cy.wait(100);
  cy.get('#github').type('Noix2zekoko');

  // Sélectionnez une compétence dans la liste déroulante
  cy.get('#skills').select(6);

  // Cliquez sur le bouton "Ajouter Compétence"
  cy.get('.btn-danger').contains('Ajouter Compétence').click();

  // Vérifiez que la compétence a été ajoutée
  cy.get('.skills-container').contains('Communication').should('be.visible');

  // Cliquez sur le bouton correspondant à la classe .btn-primary qui contient le texte "Ajouter"
  cy.get('.btn-primary').contains('Ajouter').click();

  cy.reload();

  // Vérifiez que le formulaire a été soumis avec succès
  cy.contains('Bastien');
});

```

UNIVERSITÉ TOULOUSE
Jean Jaurès

Filtres ▼ Rechercher par nom :

Pas de vacataires trouvés.

Ajouter un vacataire :

Prénom :
Bastien

Nom de famille :
Balmes

Numéro de téléphone :
060606060606

Adresse email :
bastien@gmail.com

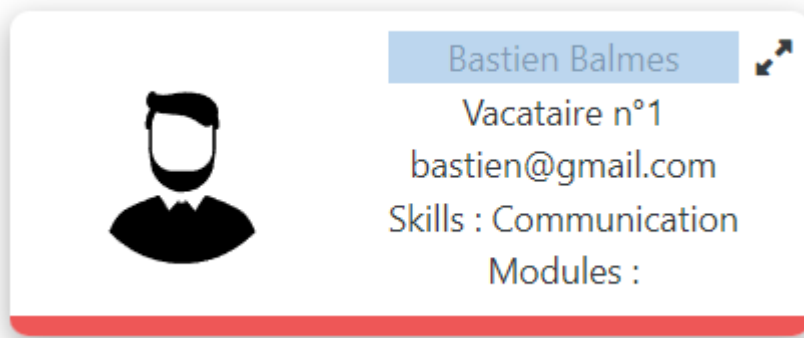
Profil GitHub :
Noix2zekoko

Sélectionnez une compétence ▼

Ajouter Compétence

Communication ✕

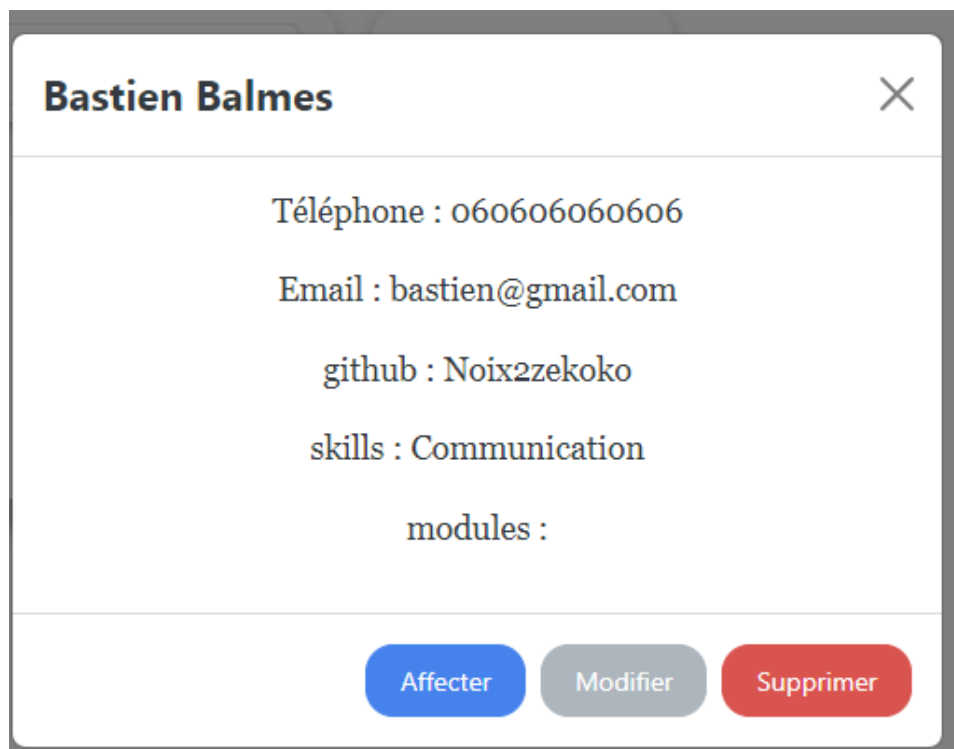
Annuler Ajouter



Suppression d'un vacataire :

Dans ce scénario de test, notre objectif est de supprimer un vacataire. Nous commencerons par cliquer sur le bouton pour ouvrir une fenêtre modale affichant les informations du vacataire. Ensuite, nous procéderons en cliquant sur le bouton "Supprimer". Pour conclure, nous vérifierons que le vacataire supprimé n'est plus présent.

```
it('Supprimer vacataire', () => {  
  cy.get('#openModal').click();  
  cy.contains('Supprimer').click();  
  cy.contains('Bastien').should('not.exist');  
});
```



Filtres ▾

Rechercher par nom :

Ajouter un vacataire

Pas de vacataires trouvés.

Lire les informations d'un vacataire :

Dans ce test, notre but est de lire les informations d'un vacataire. Nous allons confirmer que son adresse e-mail correspond bien à "bastien@gmail.com".



Bastien Balmes
Vacataire n°1
bastien@gmail.com
Skills : Communication
Modules :



```
it('Lire vacataire', () => {  
  cy.get('#openModal').click();  
  cy.contains('bastien@gmail.com');  
});
```



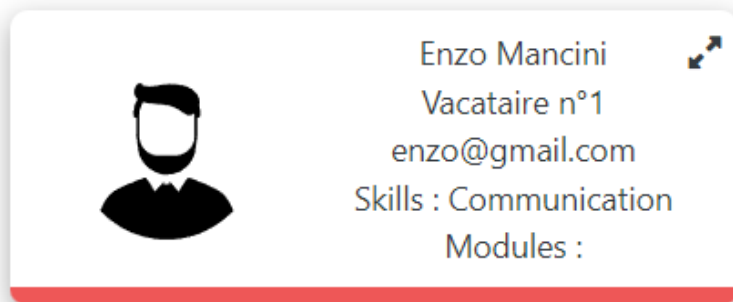
Bastien Balmes
Vacataire n°1
bastien@gmail.com
Skills : Communication
Modules :



Modifier les informations d'un vacataire :

Dans cette séquence de test, notre démarche consiste à d'abord créer un nouveau vacataire sous le nom "Enzo Mancini". Ensuite, nous allons procéder à la modification de ses informations pour le transformer en "Balmes Bastien". Enfin, nous vérifierons que "Bastien" est désormais présent dans la liste des vacataires, tandis qu'"Enzo" ne sera plus répertorié.

```
it('Ajouter vacataire', () => {  
  // Cliquez sur le bouton "Ajouter un vacataire"  
  cy.get('[data-bs-toggle="modal"]').click();  
  // Saisissez les informations dans le formulaire  
  cy.get('#name').type('Enzo');  
  cy.wait(100);  
  cy.get('#lastName').type('Mancini');  
  cy.wait(100);  
  cy.get('#phone').type('01010101010');  
  cy.wait(100);  
  cy.get('#email').type('enzo@gmail.com');  
  cy.wait(100);  
  cy.get('#github').type('enzozozo');  
  // Sélectionnez une compétence dans la liste déroulante  
  cy.get('#skills').select(6);  
  // Cliquez sur le bouton "Ajouter Compétence"  
  cy.get('.btn-danger').contains('Ajouter Compétence').click();  
  // Vérifiez que la compétence a été ajoutée  
  cy.get('.skills-container').contains('Communication').should('be.visible');  
  // Cliquez sur le deuxième bouton correspondant à la classe .btn-primary qui contient le texte "Ajouter"  
  cy.get('.btn-primary').contains('Ajouter').click();  
  cy.reload();  
  // Vérifiez que le formulaire a été soumis avec succès  
});
```





```
it('Modifier vacataire', () => {  
  cy.wait(1000);  
  cy.get('#openModal').click();  
  cy.wait(100);  
  cy.get('.name').should('contain', 'Enzo');  
  cy.contains('Modifier').click();  
  cy.wait(1000);  
  cy.get('#name').clear().type('Bastien');  
  cy.wait(100);  
  cy.get('#lastName').clear().type('Balmes');  
  cy.get('#phone').clear().type('0202020202020');  
  cy.get('#email').clear().type('bastien@gmail.com');  
  cy.get('#github').clear().type('GitHubTest');  
  // Cliquez sur le deuxième bouton correspondant à la classe .btn-primary qui contient le texte "Ajouter"  
  cy.get('.btn-primary').contains('Modifier le vacataire').click();  
  cy.reload();  
  cy.get('.name').should('contain', 'Bastien Balmes');  
  cy.contains('Enzo').should('not.exist');  
});
```

Filtres ▾

Rechercher par nom :

Ajouter un vacataire



Bastien Balmes 

Vacataire n°1

bastien@gmail.com

Skills : Communication

Modules :

3. CRUD modules

Test	Attendu	Résultat
Création d'un module	Un nouveau module apparaît dans la liste des module	Ok
Suppression d'un module	Le module choisie pour être supprimer n'apparaît plus dans la liste des modules	Ok
Lire les informations d'un module	Ouvre un modal avec les informations du module	Ok
Modifier les informations d'un module	Change les informations du module choisi	Ok

Création d'un module :

Dans ce scénario de test, nous allons créer un module. Pour ce faire, nous commencerons par cliquer sur le bouton "Ajouter un module", ce qui ouvrira une fenêtre modale contenant un formulaire pour la création du module. Ensuite, nous saisissons les informations requises dans le formulaire, le validerons, et vérifierons que la chaîne de caractères "Base de données relationnelle" est bien présente sur la page. Cette chaîne correspond au nom du nouveau module que nous venons de créer.

```
it('Ajouter module', () => {  
  cy.wait(1000);  
  cy.contains('Cours').click();  
  cy.contains('Ajouter un module').click();  
  cy.wait(1000);  
  cy.get('#name').type('Base de données relationnelle');  
  cy.wait(100);  
  cy.get('#name_reduit').type('BD');  
  cy.get('#departement').select(['INFO', 'RT']); // Sélectionnez les départements de votre choix  
  cy.get('#matiere').type('Base de données');  
  
  // Vous pouvez également tester la sélection de la couleur  
  cy.get('#color_hexa').invoke('val', '#00FF00').trigger('input');  
  
  // Cliquez sur le bouton "Ajouter" pour soumettre le formulaire  
  cy.get('.btn-primary').contains('Ajouter').click();  
  cy.wait(1000);  
  cy.contains('Base de données relationnelle');  
});
```

Ajouter un module : X

Nom :
Base de données relationnelle

Nom réduit :
BD

Départements :
INFO
RT
CS
...

Matière :
Base de données

Couleur du module :

Annuler

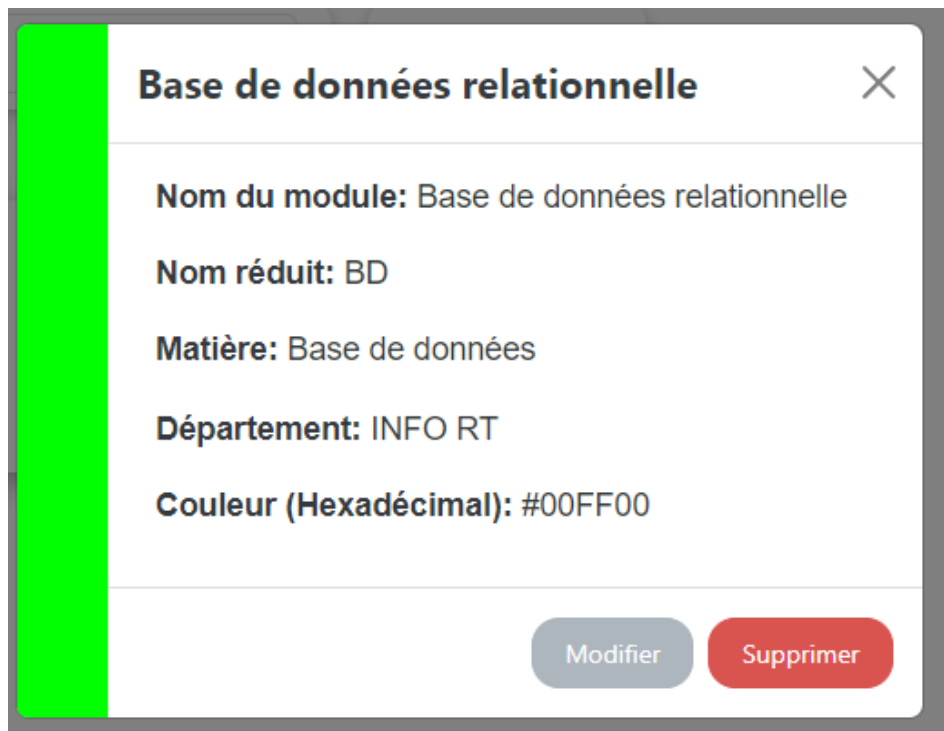
Ajouter



Suppression d'un module :

Dans ce scénario de test, notre objectif est de supprimer un module. Nous allons commencer par cliquer sur le bouton permettant d'ouvrir une fenêtre modale qui affiche les informations du module. Ensuite, nous cliquerons sur le bouton "Supprimer". Pour conclure, nous vérifierons que le module supprimé n'est plus présent.

```
it('Supprimer module', () => {  
  cy.wait(1000);  
  cy.contains('Cours').click();  
  cy.get('#openModal').click();  
  cy.contains('Supprimer').click();  
  cy.contains('Base de données relationnelle').should('not.exist');
```



Filtres ▾

Rechercher par nom :

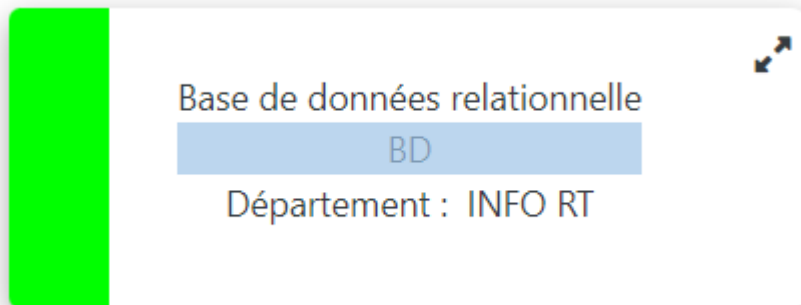
Ajouter un module

Pas de cours trouvés.

Lire les informations d'un module :

Dans ce scénario de test, nous allons lire les informations d'un module et nous assurer que son nom abrégé est correct "BD".

```
it('Lire vacataire', () => {  
  cy.wait(1000);  
  cy.contains('Cours').click();  
  cy.contains('BD');  
});
```



Modifier les informations d'un module:

Dans cette procédure de test, nous allons débuter en créant un nouveau module sous l'intitulé "Base de données relationnelle". Par la suite, nous procéderons à la modification de ses informations pour le transformer en "Dev Java". Enfin, nous confirmerons que "Dev Java" est à présent répertorié dans la liste des modules, tandis que "Base de données relationnelle" ne sera plus présent.

```

it('Ajouter module', () => {
  cy.wait(1000);
  cy.contains('Cours').click();
  cy.contains('Ajouter un module').click();
  cy.wait(1000);
  cy.get('#name').type('Base de données relationnelle');
  cy.wait(100);
  cy.get('#name_reduit').type('BD');
  cy.get('#departement').select(['INFO', 'RT']); // Sélectionnez les départements de votre choix
  cy.get('#matiere').type('Base de données');
  // Vous pouvez également tester la sélection de la couleur
  cy.get('#color_hexa').invoke('val', '#00FF00').trigger('input');

  // Cliquez sur le bouton "Ajouter" pour soumettre le formulaire
  cy.get('.btn-primary').contains('Ajouter').click();
  cy.wait(1000);
});

```

Base de données relationnelle
BD
Département : INFO RT

```

it('Modifier module', () => {
  cy.wait(1000);
  cy.contains('Cours').click();
  cy.contains('Base de données relationnelle');
  cy.get('#openModal').click();
  cy.contains('Modifier').click();
  cy.wait(1000);
  cy.get('#name').clear().type('Dev java');
  cy.wait(100);
  cy.get('#name_reduit').clear().type('DEV');
  cy.wait(100);
  cy.get('#matiere').clear().type('DEV');

  // Vous pouvez également tester la sélection de la couleur
  cy.get('#color_hexa').invoke('val', '#00FF00').trigger('input');

  // Cliquez sur le bouton "Ajouter" pour soumettre le formulaire
  cy.get('.btn-primary').contains('Valider').click();
  cy.wait(1000);
  cy.contains('Dev java');
  cy.contains('Base de données relationnelle').should('not.exist');
});

```




4. Filtre vacataires

Test	Attendu	Résultat
Filtre vacataire affecté	On voit à l'écran seulement les vacataires affectés à un module	Ok
Filtre vacataire non affecté	On voit à l'écran seulement les vacataires qui ne sont pas affectés à un module	Ok
Filtre vacataire en attente	On voit à l'écran seulement les vacataires en attente (ceux qui ont déjà eue une affectation de module, mais n'en ont plus actuellement)	Ok
Filtre des affectation par matière	On voit à l'écran seulement les vacataires qui sont affectés à une matière précise	Ok
Recherche d'un vacataire en fonction du nom	On voit à l'écran seulement le vacataire recherché s'il existe	Ok

Filtre vacataire affecté :

Dans ce scénario de test, nous allons utiliser le filtre pour afficher les vacataires affectés. Parmi les trois vacataires disponibles, Bastien Balmes, Enzo Mancini et Victor Thompson, seul Enzo Mancini est affecté à un module. Par conséquent, sur l'écran, nous ne verrons que les informations d'Enzo Mancini.

```
it('Test du filtre Affecté', () => {  
  // Ouvrir le menu "Filtres"  
  cy.contains('Filtres').click();  
  // Attendre que le menu "Filtres" soit visible  
  cy.get('.dropdown-menu').should('be.visible');  
  // Survoler l'élément "Affectation" pour déclencher le dropdown  
  cy.get('#affectation').trigger('mouseover');  
  // Attendre un certain temps pour que le dropdown apparaisse complètement  
  cy.wait(1000);  
  // Cliquez sur l'élément "Affecté" dans le dropdown en utilisant {force: true}  
  cy.contains('Affecté').click({ force: true });  
  cy.get('.name').should('not.contain', 'Bastien Balmes');  
  cy.get('.name').should('contain', 'Enzo Mancini');  
  cy.get('.name').should('not.contain', 'Victor Thompson');  
});
```

Filtres ▼

Rechercher par nom :

Ajouter un vacataire



Enzo Mancini



Vacataire n°2

Skills :

Modules : Dev java

Filtre vacataire non affecté :


Dans ce scénario de test, nous allons utiliser le filtre pour afficher les vacataires non affectés. Parmi les trois vacataires disponibles, Bastien Balmes, Enzo Mancini et Victor Thompson, seul Enzo Mancini est affecté à un module. Par conséquent, sur l'écran, nous verrons les informations de Bastien Balmes et Victor Thompson.

```
it('Test du filtre non affecté', () => {  
  // Ouvrir le menu "Filtres"  
  cy.contains('Filtres').click();  
  // Attendre que le menu "Filtres" soit visible  
  cy.get('.dropdown-menu').should('be.visible');  
  // Survoler l'élément "Affectation" pour déclencher le dropdown  
  cy.get('#affectation').trigger('mouseover');  
  // Attendre un certain temps pour que le dropdown apparaisse complètement  
  cy.wait(1000);  
  // Cliquez sur l'élément "Affecté" dans le dropdown en utilisant {force: true}  
  cy.contains('Non Affecté').click({ force: true });  
  cy.get('.name').should('contain', 'Bastien Balmes');  
  cy.get('.name').should('not.contain', 'Enzo Mancini');  
  cy.get('.name').should('contain', 'Victor Thompson');  
});
```

Filtres ▾

Rechercher par nom :

Ajouter un vacataire




Victor Thompson

Vacataire n°1

Skills :

Modules :



Bastien Balmes

Vacataire n°3

Skills :

Modules :

Filtre vacataire en attente :

Dans ce test, nous allons utiliser le filtre pour afficher les vacataires en attente. Parmi les trois vacataires disponibles, Bastien Balmes, Enzo Mancini et Victor Thompson, seul Enzo Mancini est affecté à un module. Par conséquent, sur l'écran, nous ne devrions voir aucun vacataire, car aucun d'entre eux n'est actuellement en attente.

```
it('Test du filtre en Attente', () => {  
  // Ouvrir le menu "Filtres"  
  cy.contains('Filtres').click();  
  // Attendre que le menu "Filtres" soit visible  
  cy.get('.dropdown-menu').should('be.visible');  
  // Survoler l'élément "Affectation" pour déclencher le dropdown  
  cy.get('#affectation').trigger('mouseover');  
  // Attendre un certain temps pour que le dropdown apparaisse complètement  
  cy.wait(1000);  
  // Cliquez sur l'élément "Affecté" dans le dropdown en utilisant {force: true}  
  cy.contains('En Attente').click({ force: true });  
  cy.contains('Bastien Balmes').should('not.exist');  
  cy.contains('Enzo Mancini').should('not.exist');  
  cy.contains('Victor Thompson').should('not.exist');  
});
```

Filtres ▾

Rechercher par nom :


Ajouter un vacataire

Pas de vacataires trouvés.

Filtre par matière :

Dans ce scénario de test, nous allons utiliser le filtre pour afficher les vacataires non affectés. Parmi les trois vacataires disponibles, Bastien Balmes, Enzo Mancini et Victor Thompson, seul Enzo Mancini est affecté à un module. Par conséquent, sur l'écran, nous ne verrons que les informations d'Enzo Mancini.

```
it('Test du filtre des matières', () => {  
  // Ouvrir le menu "Filtres"  
  cy.contains('Filtres').click();  
  // Attendre que le menu "Filtres" soit visible  
  cy.get('.dropdown-menu').should('be.visible');  
  // Survoler l'élément "Affectation" pour déclencher le dropdown  
  cy.get('#affectation').trigger('mouseover');  
  // Attendre un certain temps pour que le dropdown apparaisse complètement  
  cy.wait(1000);  
  // Cliquez sur l'élément "Affecté" dans le dropdown en utilisant {force: true}  
  cy.contains('Dev java').click({ force: true });  
  cy.contains('Bastien Balmes').should('not.exist');  
  cy.contains('Enzo Mancini');  
  cy.contains('Victor Thompson').should('not.exist');  
});
```



Enzo Mancini

Vacataire n°2

Skills :

Modules : Dev java

Recherche d'un vacataire en fonction de son nom :

Dans ce scénario de test, nous allons utiliser une barre de recherche pour trouver un vacataire spécifique, en l'occurrence "Bastien". Si ce vacataire a été créé, nous verrons seulement les informations de Bastien Balmes sur l'écran.

```
it('Test de la barre de recherche', () => {  
  // Ouvrir le menu "Filtres"  
  cy.get('#search').type('Bastien').type('{enter}');  
  
  cy.get('.name').should('contain', 'Bastien Balmes');  
  cy.get('.name').should('not.contain', 'Enzo Mancini');  
  cy.get('.name').should('not.contain', 'Victor Thompson');  
});
```

Filtres ▾

Rechercher par nom :

Bastien

Ajouter un vacataire



Bastien Balmes



Vacataire n°3

Skills :

Modules :

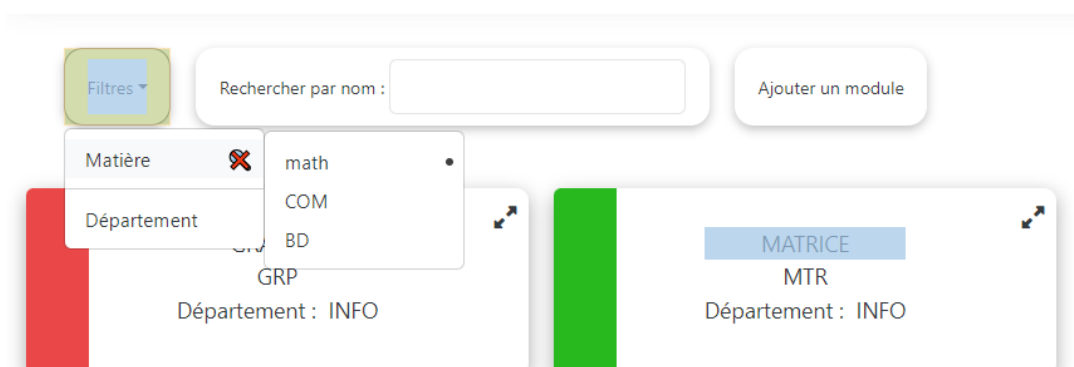
5. Filtre modules

Test	Attendu	Résultat
Filtre par matière	On voit à l'écran seulement les modules qui correspondent à la matière sélectionner	Ok
Filtre par département	On voit à l'écran seulement les modules qui appartiennent à un certain département	Ok

Filtre par matière :

Dans ce scénario de test, nous allons utiliser le filtre pour afficher les modules par matière. Nous avons créé 4 modules "Graphes" qui appartiennent à la matière "math", un module "Matrice" également associé à la matière "math", ainsi que les modules "Com" et "BD". Pour ce test, nous chercherons les modules ayant la matière "math", et sur l'écran, nous verrons les modules "Graphes" et "Matrice".

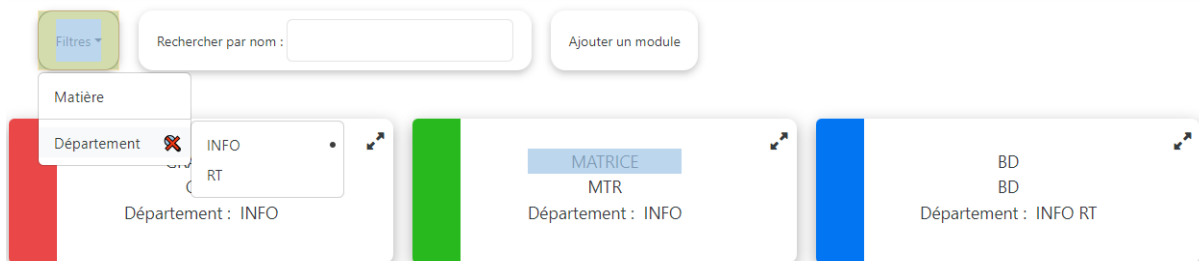
```
it('Test filtre par matière', () => {
  cy.wait(1000);
  cy.contains('Cours').click();
  cy.contains('Filtres').click();
  cy.get('.dropdown-menu').should('be.visible');
  // Attendre que le sous-menu "Matière" soit visible
  cy.get('#matieres').trigger('mouseover');
  // Cliquez sur le sous-menu "math"
  cy.contains('math').click({ force: true });
  // Vérification des modules
  cy.contains('GRAPHES');
  cy.contains('MATRICE');
  cy.contains('COM').should('not.contain'); //à modifier
  cy.contains('BD').should('not.contain'); //à modifier
});
```



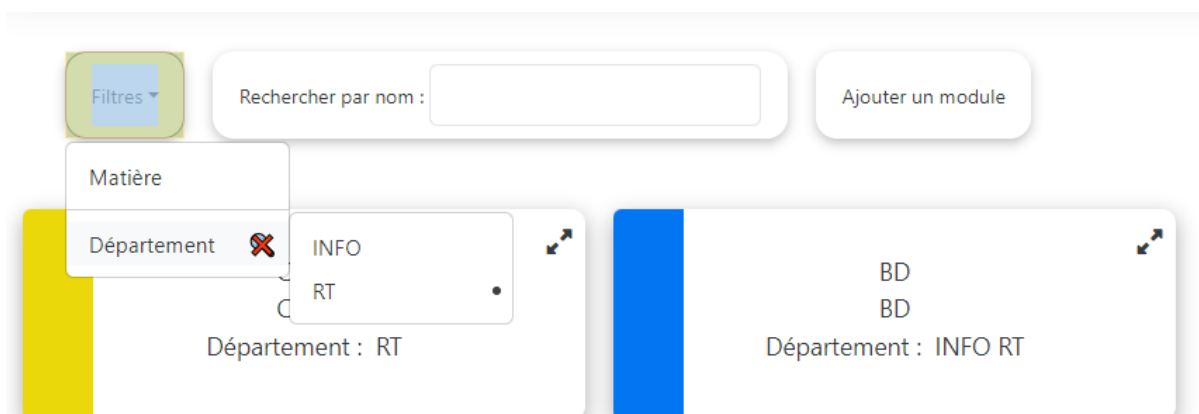
Filtre par département:

Dans ce cas de test, nous allons utiliser le filtre pour afficher les modules par département. Nous avons créé 4 modules "Graphes" qui appartiennent à la matière math, un module "Matrice" également associé à la matière math, ainsi que les modules "Com" et "BD". Pour ce test, nous chercherons les modules ayant pour département "INFO", et sur l'écran, nous verrons les modules "Graphes", "Matrice" et "BD".

```
it('Test filtre par départements (INFO)', () => {  
  cy.wait(1000);  
  cy.contains('Cours').click();  
  cy.contains('Filtres').click();  
  cy.get('.dropdown-menu').should('be.visible');  
  // Attendre que le sous-menu "Matière" soit visible  
  cy.get('#matieres').trigger('mouseover');  
  // Cliquez sur le sous-menu "math"  
  cy.contains('INFO').click({ force: true });  
  // Vérification des modules  
  cy.contains('GRAPHES');  
  cy.contains('MATRICE');  
  cy.contains('BD');  
  cy.contains('COM').should('not.contain'); //à modif  
});
```



Même test mais avec le département RT :



REGEX Ajouter vacataire

Test regex du nom :

Dans ce scénario de test, nous testons que dans le nom on ne puisse pas entrer des caractères spéciaux comme une * ou /, et aussi qu'on ne puisse pas mettre de chiffre. Si le nom contient un caractère spécial ou un chiffre, alors le bouton "Ajouter" sera désactivé et une phrase avec un message d'erreur indiquant que le nom ne doit contenir que des lettres apparaît.

Test avec chiffre :

```
it('Test regex du nom dans le formulaire (chiffre)', () => {  
  // Cliquez sur le bouton "Ajouter un vacataire" pour ouvrir le modal  
  cy.get('[data-bs-toggle="modal"]').click({ waitForAnimations: false });  
  
  cy.wait(100);  
  cy.get('[data-cy=name-input]').type('John06');  
  cy.wait(100);  
  cy.get('[data-cy=last-name-input]').type('Balmes');  
  cy.wait(100);  
  cy.get('[data-cy=phone-input]').type('0606060606');  
  cy.wait(100);  
  cy.get('[data-cy=email-input]').type('bastien@gmail.com');  
  
  cy.contains('Le prénom ne doit avoir que des lettres !');  
  // Vérifie que le bouton est désactivé  
  cy.get('[data-cy=ajouter-input]').should('be.disabled');  
});
```

Ajouter un vacataire :

Prénom :
John06
Le prénom ne doit avoir que des lettres !

Nom de famille :
Balmes

Numéro de téléphone :
0606060606

Adresse email :
bastien@gmail.com

URI du profil GitHub :

Sélectionnez une compétence

Ajouter Compétence

Annuler Ajouter

Test avec caractère spécial :

```
it('Test regex du nom dans le formulaire (caractère spéciaux)', () => {  
  // Cliquez sur le bouton "Ajouter un vacataire" pour ouvrir le modal  
  cy.get('[data-bs-toggle="modal"]').click({ waitForAnimations: false });  
  
  cy.wait(100);  
  cy.get('[data-cy=name-input]').type('John*/°=$');  
  cy.wait(100);  
  cy.get('[data-cy=last-name-input]').type('Balmes');  
  cy.wait(100);  
  cy.get('[data-cy=phone-input]').type('0606060606');  
  cy.wait(100);  
  cy.get('[data-cy=email-input]').type('bastien@gmail.com');  
  
  cy.contains('Le prénom ne doit avoir que des lettres !');  
  // Vérifie que le bouton est désactiver  
  cy.get('[data-cy=ajouter-input]').should('be.disabled');  
});
```

Ajouter un vacataire :



Prénom :

John*/°=\$

Le prénom ne doit avoir que des lettres !

Nom de famille :

Balmes

Numéro de téléphone :

0606060606

Adresse email :

bastien@gmail.com

URI du profil GitHub :

Sélectionnez une compétence



Ajouter Compétence

Annuler

Ajouter

Test regex du prénom :

Dans ce scénario de test, nous testons que dans le prénom on ne puisse pas entrer des caractères spéciaux comme une * ou /, et aussi qu'on ne puisse pas mettre de chiffre. Si le prénom contient un caractère spécial ou un chiffre, alors le bouton "Ajouter" sera désactivé et une phrase avec un message d'erreur indiquant que le prénom ne doit contenir que des lettres apparaît.

Test avec chiffre :

```
it('Test regex du prénom dans le formulaire(chiffre)', () => {  
  // Cliquez sur le bouton "Ajouter un vacataire" pour ouvrir le modal  
  cy.get('[data-bs-toggle="modal"]').click({ waitForAnimations: false });  
  
  cy.wait(100);  
  cy.get('[data-cy=name-input]').type('John');  
  cy.wait(100);  
  cy.get('[data-cy=last-name-input]').type('Balmes06');  
  cy.wait(100);  
  cy.get('[data-cy=phone-input]').type('0606060606');  
  cy.wait(100);  
  cy.get('[data-cy=email-input]').type('bastien@gmail.com');  
  
  cy.contains('Le nom de famille ne doit avoir que des lettres !');  
  // Vérifie que le bouton est désactiver  
  cy.get('[data-cy=ajouter-input]').should('be.disabled');  
});
```

Ajouter un vacataire :

×

Prénom :
John

Nom de famille :
Balmes06
Le nom de famille ne doit avoir que des lettres !

Numéro de téléphone :
0606060606

Adresse email :
bastien@gmail.com

URI du profil GitHub :

Sélectionnez une compétence
▼

Ajouter Compétence

AnnulerAjouter

Test avec caractère spécial :

```
it('Test regex du prénom dans le formulaire(caractère spéciaux)', () => {  
  // Cliquez sur le bouton "Ajouter un vacataire" pour ouvrir le modal  
  cy.get('[data-bs-toggle="modal"]').click({ waitForAnimations: false });  
  
  cy.wait(100);  
  cy.get('[data-cy=name-input']).type('John');  
  cy.wait(100);  
  cy.get('[data-cy=last-name-input']).type('Balmes*/*/$*ù');  
  cy.wait(100);  
  cy.get('[data-cy=phone-input']).type('0606060606');  
  cy.wait(100);  
  cy.get('[data-cy=email-input']).type('bastien@gmail.com');  
  
  cy.contains('Le nom de famille ne doit avoir que des lettres !');  
  // Vérifie que le bouton est désactiver  
  cy.get('[data-cy=ajouter-input']).should('be.disabled');  
});
```

Ajouter un vacataire :



Prénom :
John

Nom de famille :
Balmes*/*/\$*ù

Le nom de famille ne doit avoir que des lettres !

Numéro de téléphone :
0606060606

Adresse email :
bastien@gmail.com

URI du profil GitHub :

Sélectionnez une compétence ▼

Ajouter Compétence

Annuler Ajouter

Test regex du téléphone:

Dans ce scénario de test, nous testons que dans le téléphone on ne puisse pas entrer des caractères spéciaux comme une * ou /, et aussi qu'on ne puisse pas mettre de lettre. Si le téléphone contient un caractère spécial ou une lettre, alors le bouton "Ajouter" sera désactivé et une phrase avec un message d'erreur indiquant que le téléphone ne doit contenir que des lettres apparaît.

Ajouter un vacataire :



Prénom :

John

Nom de famille :

Balmes

Numéro de téléphone :

fghfgh

Le numéro doit être au format FR (sans espace)!

Adresse email :

bastien@gmail.com

URI du profil GitHub :

Sélectionnez une compétence



Ajouter Compétence

Annuler

Ajouter

Test regex du mail:

Dans ce scénario de test, nous testons que le mail soit au bon format c'est-à-dire avec un @ et un .quelque chose. Si le mail n'est pas valide, alors le bouton "Ajouter" sera désactivé et une phrase avec un message d'erreur indiquant que le téléphone ne doit contenir que des lettres apparaît.

Ajouter un vacataire :



Prénom :

John

Nom de famille :

Balmes

Numéro de téléphone :

0606060606

Adresse email :

bastiengmailcom

L'adresse email n'est pas valide !

URI du profil GitHub :

Sélectionnez une compétence



Ajouter Compétence

Annuler

Ajouter

Test regex valide:

Dans ce scénario de test, nous testons que tous les champs entrés sont valide, si c'est le cas alors le bouton Ajouter sera activé donc cliquable pour valider l'ajout. Et donc le vacataire John Balmes sera ajouté à la liste des vacataires.

Ajouter un vacataire :



Prénom :

John

Nom de famille :

Balmes

Numéro de téléphone :

0606060606

Adresse email :

bastien@gmail.com

URI du profil GitHub :

Sélectionnez une compétence



Ajouter Compétence

Annuler

Ajouter



John Balmes

Vacataire n°1

