

4-3 Contacts – Modifier ou ajouter

1 – Créer un nouveau contact

Je commence par créer la route qui va permettre l'ajout de nouveau contact

```
$routes->get('organisations/{:any}/contacts/{:any}', 'Contacts::getById/{1}/{2}');  
$routes->post('organisations/{:any}/contacts', 'Contacts::creerContact/{1}');  
$routes->get('organisations/{:any}/contacts/{:any}', 'Contacts::annulerId/{1}/{2}');
```

Dans le contrôleur Contacts.php, je crée la méthode creerContact()

```
public function creerContact($any) : void {  
    $model = model(ContactsModel::class);  
  
    // on récupère chaque donnée passée dans le corps de la requête  
    $id = $this->request->getVar("id");  
    $numOrga = $any;  
    $civilite = $this->request->getVar("civilite");  
    $nom = $this->request->getVar("nom");  
    $prenom = $this->request->getVar("prenom");  
    $email = $this->request->getVar("email");  
    $tel = $this->request->getVar("tel");  
    $fonction = $this->request->getVar("fonction");  
}
```

Elle récupère la valeur du numéro d'organisation entré dans l'URL de la requête et le reste des informations est récupéré dans la requête POST

On définit les règles de format à respecter pour chaque champ et on crée un tableau data, contenant les données de la requête, qui sera inspecté pour s'assurer que toutes les données respectent les expressions régulières

```
// On détermine les règles de validation à appliquer pour chaque champ  
$validation = \Config\Services::validation();  
  
$rules = [  
    "id" => "required|numeric",  
    "civilite" => "required|max_length[3]|regex_match[/[Me\. ]$/]",  
    "prenom" => "required|max_length[50]|regex_match[/[A-Za-z_\- ]$/]",  
    "nom" => "required|max_length[50]|regex_match[/[A-Za-z_\- ]$/]",  
    "email" => "permit_empty|valid_email",  
    "tel" => "permit_empty|regex_match[/^[0-9]{10}$/]",  
    "fonction" => "max_length[100]|regex_match[/[A-Za-z_\- ]/]"  
];  
$validation->setRules($rules);  
  
// Construction d'un tableau avec clés et valeurs des données à valider  
$data = [  
    "id" => $this->request->getVar("id"),  
    "civilite" => $this->request->getVar("civilite"),  
    "prenom" => $this->request->getVar("prenom"),  
    "nom" => $this->request->getVar("nom"),  
    "email" => $this->request->getVar("email"),  
    "tel" => $this->request->getVar("tel"),  
    "fonction" => $this->request->getVar("fonction")  
];
```

Si les données sont valides, le nouveau contact est créé et une réponse au code status 201 est renvoyée. Sinon, une réponse contenant une erreur 400 (requête invalide) est renvoyée

```
if($validation->run($data)) {
    // on appelle le modèle qui insérera le nouveau frais forfait auprès de la sour
    $newId = $model->createNew($id, $numOrga, $civilite, $nom, $prenom, $email, $te
    $codeStatus = 201;
    // on construit les données résultats, la réponse http et on l'envoie
    $result = ["message" => "Nouveau contact créé", "data" => $newId];
}

else {
    $codeStatus = 400;
    $result = ["message" => "Format de données invalide",
               "errors" => $validation->getErrors()];
}

$this->response
    ->setStatusCode($codeStatus)
    ->setHeader('Content-type', 'application/json')
    ->setJson($result);
$this->response->send();
```

Test de création d'un nouveau contact avec Talend API Tester (valide)

Creer-Contact

METHOD: POST

SCHEME :// HOST [":" PORT] [PATH ["?" QUERY]]

URL: \${"protocol"}://\${"host"}/\${"path"}/organisations/3/contacts

QUERY PARAMETERS

HEADERS

- Content-Type: application/json

Body:

```
1 {
2   "id": "8",
3   "civilite": "M. ",
4   "prenom": "Manoël",
5   "nom": "Hallu",
6   "email": "manoel.hallu@lycee-basch.fr",
7   "tel": "0299999999",
8   "fonction": "Professionnel"
9 }
```

201 Created

HEADERS

- Date: Tue, 13 Feb 2024 15:35:13 GMT -1s
- Server: Apache/2.4.56 (Win64) OpenSSL/1.1.1t PHP/8.2.4
- X-Powered-By: PHP/8.2.4
- Cache-Control: no-store, max-age=0, no-cache
- Cache-Control: no-store, max-age=0, no-cache

Body:

```
{
  message: "Nouveau contact créé",
  data: "8"
}
```

HISTORY ASSERTIONS 3/3 HTTP DESCRIPTION

<input checked="" type="checkbox"/>	Status code	Equals	201	✎	✕
<input checked="" type="checkbox"/>	JSON body	\$.data	Exists	✎	✕
<input checked="" type="checkbox"/>	JSON body	\$.message	Equals	Nouveau contact créé	✎ ✕

localhost/sio2/M_HALLU/ap32-stages-apirest/public/index.php/organisations/3/contacts

```
{2}
age : "OK"
[2]
{8}
{8}
id : 8
civilite : "M. "
nom : "Hallu"
prenom : "Manoël"
email : "manoel.hallu@lycee-basch.fr"
tel : "0299999999"
fonction : "Professionnel"
_selfLink : "http://localhost/sio2/M_HALLU/ap32-stages-apirest/public/index.php/organisations
```

Tests avec données invalides

Numéro de téléphone invalide

```
1 {
2   "id": "8",
3   "civilite": "M. ",
4   "prenom": "Manoël",
5   "nom": "Hallu",
6   "email": "manoel.hallu@lycee-basch.fr",
7   "tel": "0299ABCDEF",
8   "fonction": "Professionnel"
9 }
```

▶ BODY ⓘ

```
{
  message: "Format de données invalide",
  errors: {
    tel: "The tel field is not in the correct format."
  }
}
```

HISTORY	ASSERTIONS 3/3	HTTP	DESCRIPTION
<input checked="" type="checkbox"/>	Status code	Equals	400
<input checked="" type="checkbox"/>	JSON body	\$.data	Does not exist
<input checked="" type="checkbox"/>	JSON body	\$.message	Equals Format de données invalide

+ Add assertion Suggestions: Body content exists

Nom inexistant

```
{
  "id": "8",
  "civilite": "M. ",
  "prenom": "Manoël",
  "nom": "",
  "email": "manoel.hallu@lycee-basch.fr",
  "tel": "0299999999",
  "fonction": "Professionnel"
}
```



▶ BODY ⓘ

```
{
  message: "Format de données invalide",
  errors: {
    nom: "The nom field is required."
  }
}
```

<input checked="" type="checkbox"/>	Status code	Equals	400	✎	✕
<input checked="" type="checkbox"/>	JSON body	\$.data	Does not exist	✎	✕
<input checked="" type="checkbox"/>	JSON body	\$.message	Equals	Format de données invalide	✎ ✕

+ Add assertion Suggestions: Body content exists

Prénom invalide

BODY ⓘ

```
1 {
2   "id": "8",
3   "civilite": "M. ",
4   "prenom": "M4n0ë1",
5   "nom": "Hallu",
6   "email": "manoel.hallu@lycee-basch.fr",
7   "tel": "0299999999",
8   "fonction": "Professionnel"
9 }
```



▶ BODY ⓘ

```
{
  message: "Format de données invalide",
  errors: {
    prenom: "The prenom field is not in the correct format."
  }
}
```

	HISTORY	ASSERTIONS 3/3	HTTP	DESCRIPTION	
<input checked="" type="checkbox"/>	Status code	Equals	400	✎ ✕	
<input checked="" type="checkbox"/>	JSON body	\$.data	Does not exist	✎ ✕	
<input checked="" type="checkbox"/>	JSON body	\$.message	Equals	Format de données invalide	✎ ✕

+ Add assertion Suggestions: Body content exists

Dans les cas où les données sont invalides, le nouveau contact n'est pas créé et n'apparaît pas dans la base de données.

2 – Modifier un contact

Manque de temps pour compléter cette partie