ListView

Application Web

Paul Arzul
Steven Jeanneret
INF3DLM-B

Sommaire

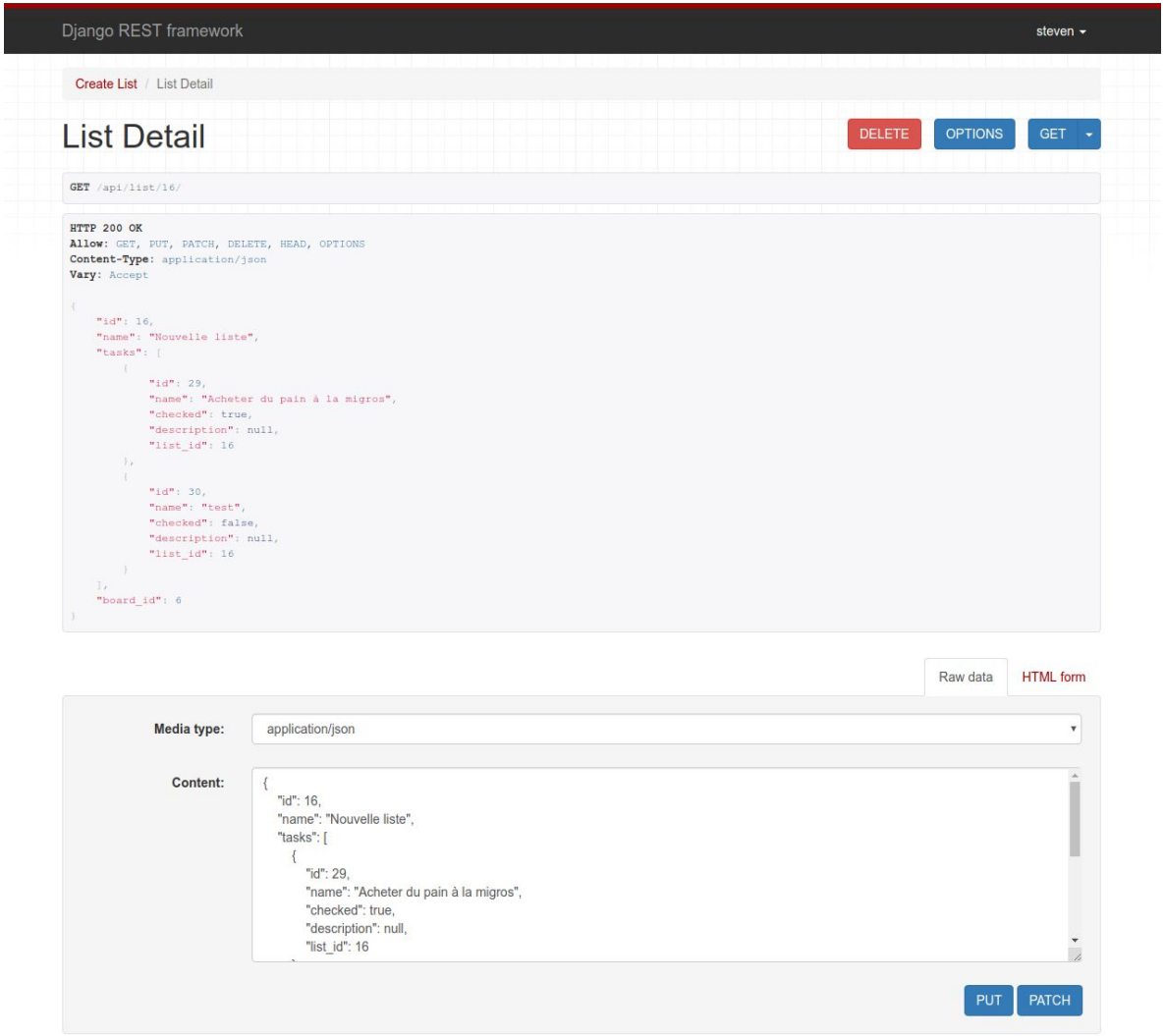
- Introduction
- Technologies employées
- Authentification
- Démonstration
- Explications
- Conclusion

Introduction

- Kanban / checklist
- Inspiration : Trello
- Gestion de teams
- Créer des boards
- Créer des listes
- Créer des tâches

Technologies utilisées

- API REST : Django et Django REST framework
  - Des classes pour les requêtes courantes
- Front end : Vue.js et Nuxt.js
  - Structure des dossiers pour Vue.js
  - Des modules inclus par défaut


Django REST framework
steven
Create List / List Detail
List Detail
DELETE
OPTIONS
GET
GET /api/list/16/
HTTP 200 OK
Allow: GET, PUT, PATCH, DELETE, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept
{
"id": 16,
"name": "Nouvelle liste",
"tasks": [
{
"id": 29,
"name": "Acheter du pain à la migros",
"checked": true,
"description": null,
"list_id": 16
},
{
"id": 30,
"name": "test",
"checked": false,
"description": null,
"list_id": 16
}
],
"board_id": 6
}
Raw data
HTML form
Media type:
application/json
Content:
{
"id": 16,
"name": "Nouvelle liste",
"tasks": [
{
"id": 29,
"name": "Acheter du pain à la migros",
"checked": true,
"description": null,
"list_id": 16
PUT
PATCH

Authentification

- Token JWT
- Système maison
  - Implémenter la sécurité
- Système tierce, Auth0
  - Intégration sur notre site
  - Login sur leur site

Démonstration

Board - view

- ListCreateAPIView, get et post
- RetrieveUpdateDestroyAPIView, get et put/patch et delete
- queryset, les données
- serializer_class
- [permission_classes], isAuthenticated, custom

Board - view

```python
class BoardList(generics.ListCreateAPIView):
    queryset = Board.objects.all()
    serializer_class = BoardSerializer
    permission_classes = (permissions.IsAuthenticated,)


class BoardDetail(generics.RetrieveUpdateDestroyAPIView):
    queryset = Board.objects.all()
    serializer_class = BoardDetailSerializer
    permission_classes = (permissions.IsAuthenticated, BoardAccessPermission,)
```

Board - serializer

- ModelSerializer
- model
- fields

Board - serializer

```python
class BoardDetailSerializer(serializers.ModelSerializer):
    lists = ListSerializer(many=True, read_only=True)


    class Meta:
        model = Board
        fields = ('id', 'name', 'lists')
```

Board - requête

```
loadLists () {
  this.$axios.get(`/api/boards/${this.$route.params.boardId}`).then(result => {
    this.lists = result.data.lists
    this.name = result.data.name
  })
```

Board - affichage

```
<h1 class="display-3 text-center">{{name}}</h1>
<div class="row text-center">
  <div class="col-auto" v-for="l in lists">
    <list :name="l.name" :key="l.id" :tasks="l.tasks" :id="l.id"></list>
  </div>
```

Point à améliorer

- Drag and drop
- Gestion d’invitation d’équipe
- Ajout de rôle dans une équipe (admin, utilisateur)
- Date d'échéance des tâches

CONCLUSION

- Projet fonctionnel et sécurisé
- Nouvelle technologies découvertes

Merci de votre attention
Des questions?