

🇸🇵 API Mobile Burkina - Documentation Officielle

Version : ****1.0.0****

Auteur : ****mamadou-yeo****

Base URL : **`/api/v1/idcapture`**

Base de données : ****Microsoft SQL Server****

🧩 Vue d'ensemble

Cette API permet :

*** **La gestion des cartes de production** :**

* Recherche par critères

* Récupération avec pagination

* Mise à jour

* Distribution

*** **L'authentification des utilisateurs** :**

* Inscription

* Connexion (JWT)

* Récupération du profil utilisateur

👤 Authentification (Users)

Modèle User

```

User {

nom: string;

prenom: string;

login: string;

mdp: string;

idrole: number;

}

```

1. ➕ Inscription

****POST**** **`/user/register`**

Body (JSON)

```

{

```
"nom": "Yeo",
"prenom": "Mamadou",
"login": "mamayeo",
"mdp": "MonMotDePasse123!",
"idrole": 1
}
``
```

### ### ✓ Réponse 201 - Succès

```
``
{
 "message": "Utilisateur créé avec succès",
 "user": {
 "idutilisateur": 1,
 "nom": "Yeo",
 "prenom": "Mamadou",
 "login": "mamayeo",
 "uestatif": true,
 "idrole": 1
 }
}
``
```

### ### ✗ Erreurs

**\*\*500\*\*** : Erreur serveur ou login déjà existant

---

## ## 2. 🔑 Connexion

**\*\*POST\*\*** `/user/login`

### ### Body (JSON)

```
``
{
 "login": "mamayeo",
 "mdp": "MonMotDePasse123!"
}
``
```

### ### ✓ Réponse 200 - Succès

```
``
{
 "success": true,
 "message": "Connexion réussie",
 "token":
 "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwiYWV0IjYtYW1heWVvliwiaWRyb2xi"
}
```

```
lJoxLCJpYXQiOiE3NjQxNjkyMDIsImV4cCI6MTc2NDIwNTIwMn0.AQMHB0OnAKVhAhrwMGznpD
ZZ_Y6jEsKRS8hIKqtQBzY",
"data": {
 "idutilisateur": 2,
 "nom": "Yeo",
 "prenom": "Mamadou",
 "login": "mamayeo",
 "uestatif": true,
 "idrole": 1
}
}
``
```

### ### ✖ Erreurs

```
* **401** : Identifiants invalides ou utilisateur inactif
* **500** : Erreur serveur
```

```

```

### ## 3. 👤 Profil utilisateur (JWT)

```
GET `/user/profile`
```

### ### Headers

```
``
```

```
Authorization: Bearer
<eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwibG9naW4iOiJtYW1heWVvliwiaWRyb2
xIlJoxLCJpYXQiOiE3NjQxNDUzMDAsImV4cCI6MTc2NDE3NDEwMH0.f-
hOvo0hbozzHUO8TVa81bVuC3s1Q-n6U2pilvmABSg>
``
```

## NB : le token dure 10h

### ✔ Réponse 200 - Succès

```
``
```

```
{
 "idutilisateur": 1,
 "nom": "Yeo",
 "prenom": "Mamadou",
 "login": "mamayeo",
 "uestatif": true,
 "idrole": 1
}
``
```

### ### ✖ Erreurs

```
* **403** : Token manquant
* **401** : Token invalide ou expiré
```

---

## # 📄 CarteProduction - Endpoints

### ## Url base

http://localhost:2025/api/v1/idcapture

### ## 1. Récupérer toutes les cartes (pagination)

GET /carte/gestall?limit=2&page=3&is\_distributed=null&is\_produced=null

#### Réponses

200 OK → Liste des cartes avec pagination et champs calculés (status, message)

500 Internal Server Error → Erreur serveur

### ## 2. Rechercher des cartes

GET /carte/searchs?prenoms=SOUMAILA&nom=BAKO&datenaissance=1987-11-12

#### Réponses

200 OK → Résultats enrichis avec status (pending, produced, distributed)

404 Not Found → Aucune carte trouvée

500 Internal Server Error → Erreur serveur

### ## 3. Récupérer une carte par code unique

GET /carte/:unique\_code

#### Réponses

200 OK → Carte récupérée avec succès

409 Conflict → Carte trouvée mais pas encore produite

200 OK → Carte produite mais pas encore distribuée (status: produced)

200 OK → Carte produite et distribuée (status: distributed)

404 Not Found → Carte non trouvée

500 Internal Server Error → Erreur serveur

### ## 4. Distribuer une carte

PUT /carte/distribute/:unique\_code

#### Réponses

200 OK → Carte distribuée avec succès

400 Bad Request → Paramètre manquant

409 Conflict → Carte non produite ou déjà distribuée

404 Not Found → Carte non trouvée

500 Internal Server Error → Erreur serveur

### ## 5. Mettre à jour la photo

PUT /carte/photo/:unique\_code

Body

multipart/form-data avec champ photo (fichier)

ou JSON { "urlphoto": "http://..." }

Réponses

200 OK → Photo mise à jour avec succès (status: updated)

400 Bad Request → Aucune photo ou URL fournie (status: invalid)

404 Not Found → Carte non trouvée (status: not\_found)

500 Internal Server Error → Erreur serveur (status: error)

### ## 6. Mise à jour générique

PUT /carte/:unique\_code

Body

json

{ "production\_date": "2025-11-26", "distribution\_date": "2025-11-26", "is\_distributed": true, "is\_produced": true }

Réponses

200 OK → Carte mise à jour avec succès

400 Bad Request → Paramètre unique\_code manquant

404 Not Found → Carte non trouvée

500 Internal Server Error → Erreur serveur

### ## Codes de statut utilisés

200 OK → Succès (lecture, mise à jour)

201 Created → Ressource créée (utilisateur)

400 Bad Request → Paramètres manquants ou invalides

401 Unauthorized → Identifiants invalides ou token expiré

403 Forbidden → Token manquant

404 Not Found → Ressource non trouvée

409 Conflict → Ressource trouvée mais état métier incompatible (ex : carte non produite ou déjà distribuée)

500 Internal Server Error → Erreur serveur

### ### 📄 **Pagination complète**

- \* currentPage
- \* totalPages
- \* totalItems
- \* itemsPerPage

### ### ⚠ **Gestion des erreurs**

Codes utilisés : **\*\*200, 201, 202, 400, 401, 403, 404, 500\*\***

---

## # 🛠 **Installation & Utilisation**

### ### **Installer**

```
```\nnpm install\n```
```

Démarrer en développement

```
```\nnpm run dev\n```
```

### ### **Production**

```
```\nnpm run build\nnpm start\n```
```

Docker

Développement

```
sudo docker compose -f docker-compose.dev.yml --env-file .env.development up --build\n```\n
```

```
```\n
```

#### ## **Production**

```
sudo docker compose -f docker-compose-prod.yml --env-file .env.production up --build
```

---

## # 🖋 **Exemples d'utilisation**

### ### JavaScript (Fetch)

```
```\nconst response = await fetch('http://localhost:2025/api/v1/idcapture/carte/ABC123');\nconst data = await response.json();\n```\n
```

cURL

```
```\ncurl -X GET "http://localhost:2025/api/v1/idcapture/carte/ABC123"\n```\n
```

---

### # 📞 Contact

**\*\*Auteur :\*\*** mamadou-yeo

**\*\*Version :\*\*** 1.0.0

Licence : **\*\*ISC\*\***