

ELECTION MANAGEMENT SYSTEM

Mohamed Youssef Blidi

OVERVIEW

Inspiration

What is EMS?

technology used

Scalibility

Routes and Snippets

Conclusion

PRESENTATION TITLE

INSPIRATION



EMPOWERING ELECTIONS THROUGH TECHNOLOGY

- Leveraging technology in election management enhances security, transparency, and accessibility, addressing current challenges and fostering trust in the democratic process.



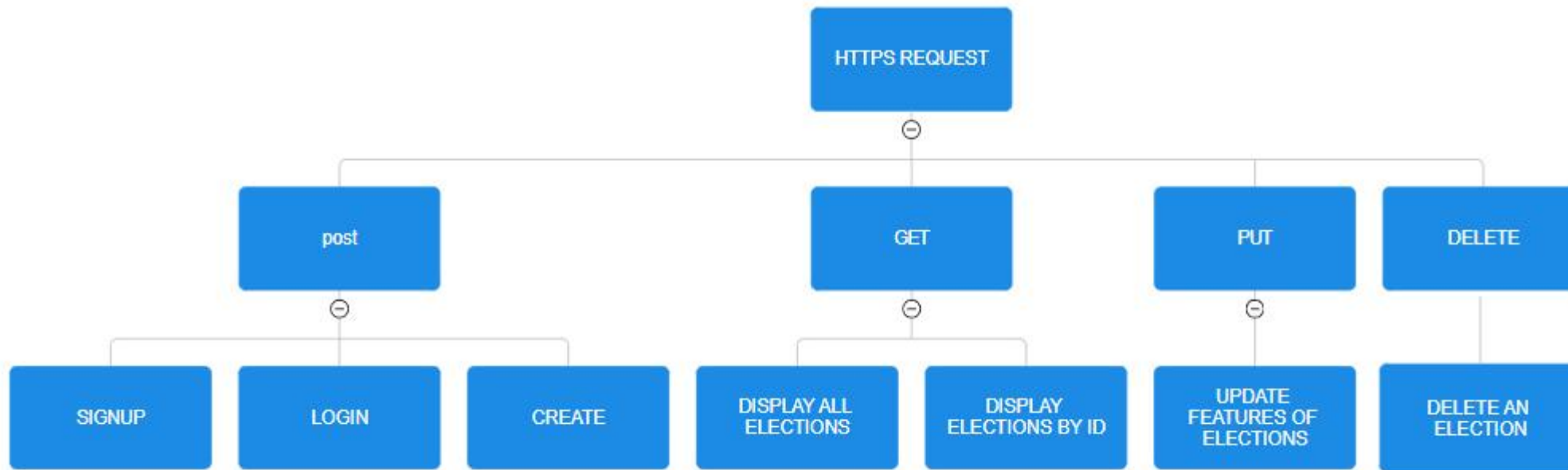
CHALLENGES IN ELECTION MANAGEMENT

- Common challenges in election management include security concerns, reliance on manual processes, and a lack of transparency.



WHAT IS EMS ?

- EMS is an election management rest API that enable users to do this manipulation:



TECHNOLOGY USED

BACKEND



EXPRESS 

EDET EMMANUEL ASUOJO

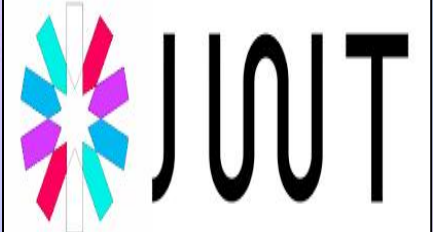
TESTING



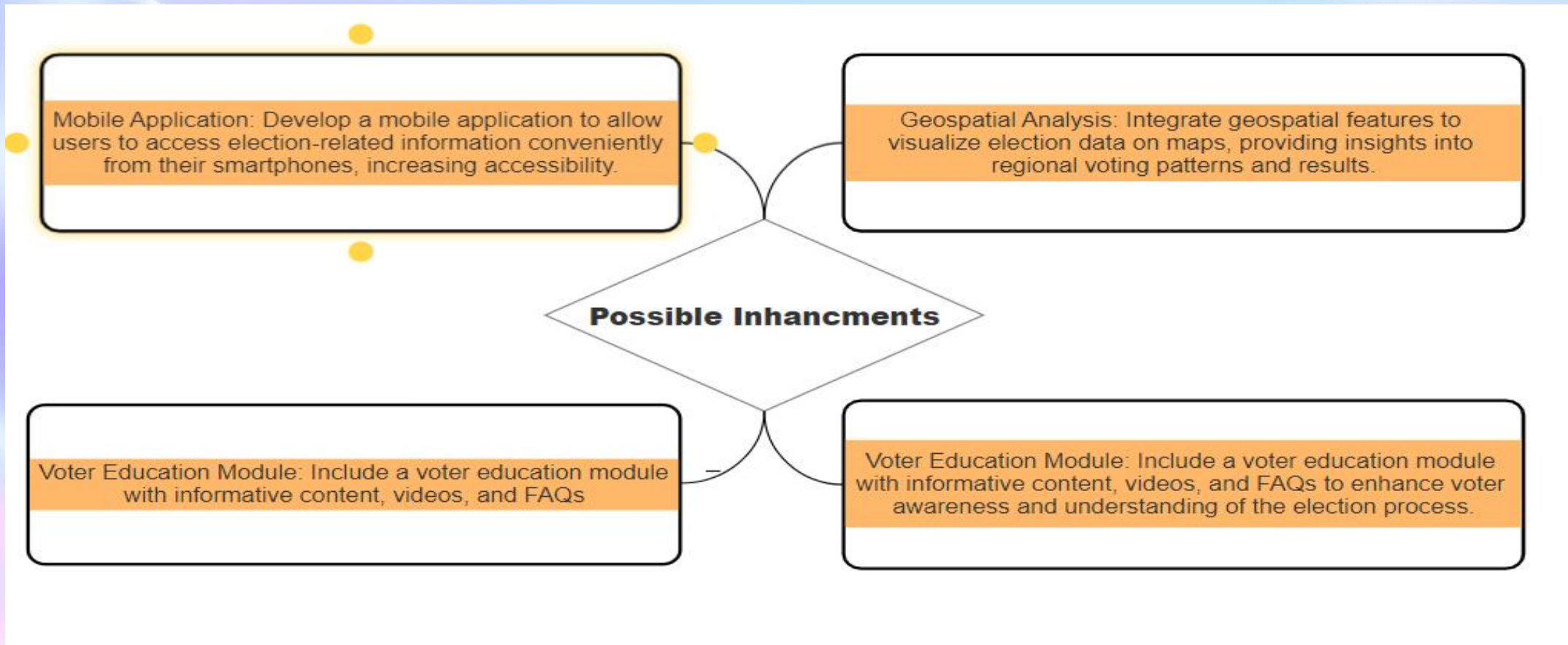
**DATABASE
MANAGEMENT
SYSTEM**



SECURITY



SCALIBILITY



ROUTES

AUTH ROUTE

```
const express = require('express');
const router = express.Router();
const electionController = require('../controllers/electionsController');

router.post("/create", electionController.createElection);
router.get('/display', electionController.getAllElections);
router.get('/display/:electionId', electionController.getElectionById);
router.put('/update/:electionId', electionController.updateElectionById);
router.delete('/delete/:electionId', electionController.deleteElectionById);

module.exports = router;
```

ELECTION ROUTE

```
routes > JS election.route.js > ...
You, 2 days ago | 2 authors (smarouen and others)
1  const express = require('express');
2  const router = express.Router();
3  const electionController = require('../controllers/electionsController');
4
5
6  router.post("/create", electionController.createElection);
7  router.get('/display', electionController.getAllElections);
8  router.get('/display/:electionId', electionController.getElectionById);
9  router.put('/update/:electionId', electionController.updateElectionById);
10 router.delete('/delete/:electionId', electionController.deleteElectionById);
11
12 module.exports = router;
13 |
```

sign up

The screenshot displays the Postman application interface. At the top, the navigation bar includes 'Home', 'Workspaces', and 'API Network'. A search bar labeled 'Search Postman' is present. The main workspace is titled 'My Workspace' and shows a list of API collections. The selected collection is 'REST API basics: CRUD, test & variable', which contains several endpoints. The 'POST singup' endpoint is highlighted. The request details for this endpoint are shown, including the URL 'http://localhost:3000/auth/signup' and the request body in JSON format. The response is displayed in the 'Body' tab, showing a successful status (200 OK) and a JSON object with user details. A notification at the bottom indicates that the screen is being shared via Google Meet.

Home Workspaces API Network

Search Postman

Invite Upgrade

My Workspace New Import

REST API basics: CRUD, test & variable

- GET Get data
- POST Post data
- PUT Update data
- DEL Delete data
- POST singup
- POST login
- POST create
- POST update
- GET displayall
- GET displaybyID
- GET delete

POST http://localhost:3000/auth/signup

Params Authorization Headers (10) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "name": "test1",
3   "email": "test1@gmail.com",
4   "password": "test123"
5 }
```

Body Cookies (1) Headers (9) Test Results

Status: 200 OK Time: 278 ms Size: 611 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "message": "user signed up successfully",
4   "user": {
5     "name": "test1",
6     "email": "test1@gmail.com",
7     "password": "$2b$10$Xmt8fqt3XytcTtRcbvms.jZcNbLuKzVZJp/v.mpwleTKjSqlJNsC",
8     "isConncted": false,
9     "_id": "65aaa8c3a3b79025bda0945d",
10    "createdAt": "2024-01-19T16:52:19.040Z",
11    "__v": 0
12  }
13 }
```

|| Votre écran est partagé par le biais de l'application meet.google.com. Arrêter le partage Masquer

Online Find and replace Console

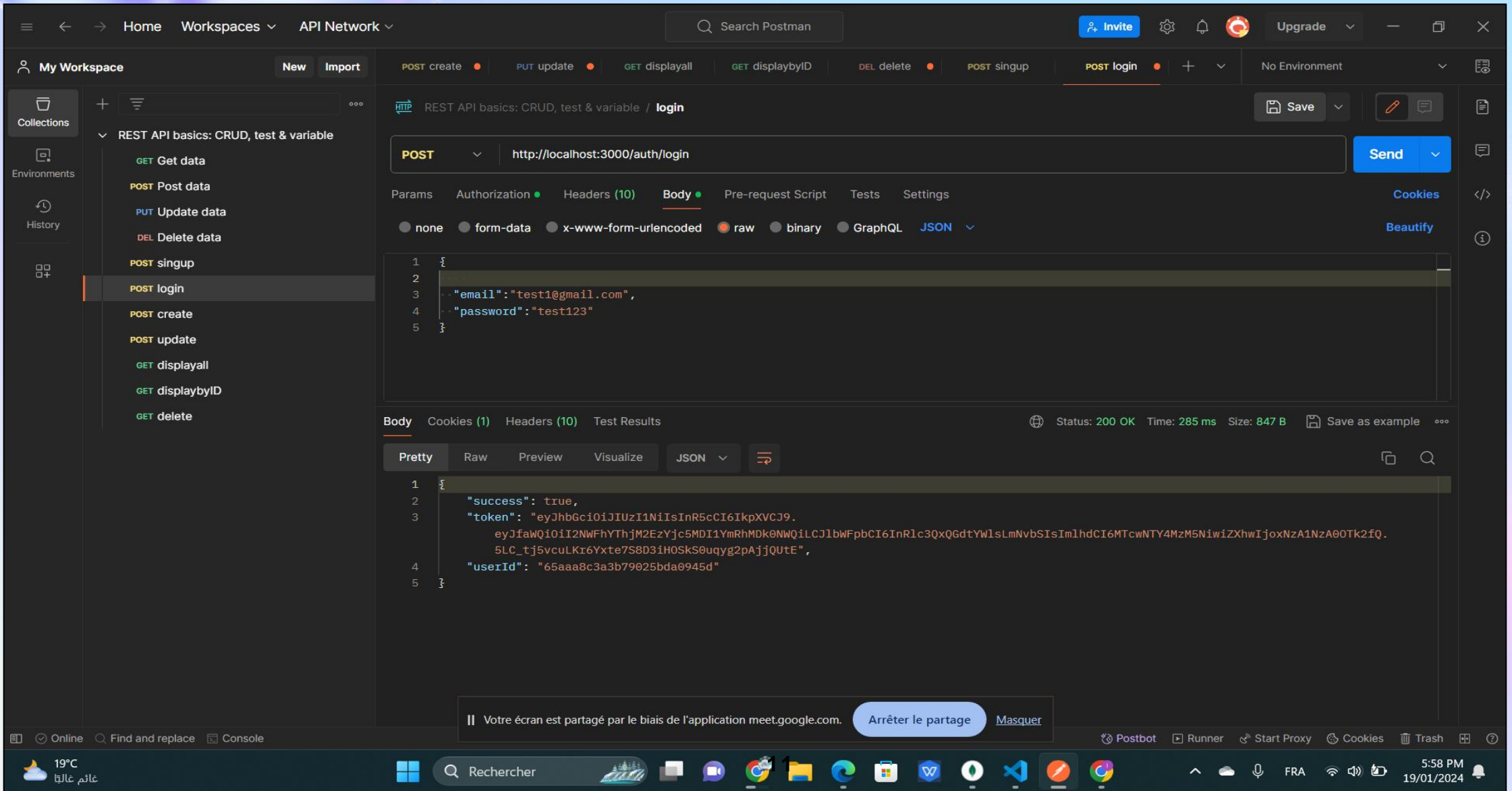
19°C غائم

Rechercher

Postbot Runner Start Proxy Cookies Trash

5:57 PM 19/01/2024

LOGIN



CREATE

The screenshot displays the Postman application interface. At the top, the navigation bar includes 'Home', 'Workspaces', and 'API Network'. A search bar labeled 'Search Postman' is present. On the left sidebar, 'My Workspace' is selected, showing a collection named 'REST API basics: CRUD, test & variable'. The 'POST create' endpoint is highlighted. The main panel shows the request configuration for a POST method to 'http://localhost:3000/elections/create'. The 'Body' tab is active, displaying a JSON payload:

```
{  "title": "district assembly elections",  "description": "A district assembly election happens when only voters or constituents who reside within an electoral district vote in an election held in there",  "date": "2025-03-10T10:00:00"}
```

. Below the request, the response is shown in the 'Pretty' format:

```
{  "success": true,  "message": "election Created Successfully",  "data": {    "title": "district assembly elections",    "description": "A district assembly election happens when only voters or constituents who reside within an electoral district vote in an election held in there",    "date": "2025-03-10T10:00:00",    "_id": "65aaa760a3b79025bda09458",    "createdAt": "2024-01-19T16:46:24.450Z",    "__v": 0  }}
```

. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 100 ms', and 'Size: 706 B'. A notification at the bottom states 'Votre écran est partagé par le biais de l'application meet.google.com.'.

UPDATE

The screenshot displays the Postman API client interface. The top navigation bar includes 'Home', 'Workspaces', and 'API Network'. The left sidebar shows 'My Workspace' with a collection named 'REST API basics: CRUD, test & variable'. The main panel shows a PUT request to 'http://localhost:3000/elections/update/65aa9e3ba3b79025bda09447'. The request body is a JSON object with 'title', 'description', and 'date' fields. The response is a JSON object indicating success and providing the updated data. The status bar at the bottom shows 'Status: 200 OK', 'Time: 77 ms', and 'Size: 665 B'.

Request:

```
PUT http://localhost:3000/elections/update/65aa9e3ba3b79025bda09447
```

Body (JSON):

```
{
  "title": "Primary elections",
  "description": "In a primary election, each political party selects its candidates to run for office during the general election",
  "date": "2024-09-15T09:00:00"
}
```

Response (JSON):

```
{
  "success": true,
  "message": "Election updated successfully",
  "data": {
    "_id": "65aa9e3ba3b79025bda09447",
    "title": "Primary elections",
    "description": "In a primary election, each political party selects its candidates to run for office during the general election",
    "date": "2024-09-15T09:00:00",
    "createdAt": "2024-01-19T16:07:23.017Z",
    "__v": 0
  }
}
```

DISPLAY ALL

The screenshot displays the Postman API client interface. The top navigation bar includes 'Home', 'Workspaces', and 'API Network'. The left sidebar shows 'My Workspace' with a list of collections, including 'REST API basics: CRUD, test & variable'. The main panel shows a GET request to 'http://localhost:3000/elections/display'. The response is a JSON array of two election objects, displayed in the 'Body' tab. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 61 ms', and 'Size: 1.63 KB'.

Request Details:

- Method: GET
- URL: http://localhost:3000/elections/display
- Body: This request does not have a body

Response Details:

Status: 200 OK, Time: 61 ms, Size: 1.63 KB

```
[{"date": "2024-07-23T10:00:00", "createdAt": "2024-01-19T16:39:56.003Z", "__v": 0}, {"_id": "65aaa760a3b79025bda09458", "title": "district assembly elections", "description": "A district assembly election happens when only voters or constituents who reside within an electoral district vote in an election held in there", "date": "2025-03-10T10:00:00", "createdAt": "2024-01-19T16:46:24.450Z", "__v": 0}]
```

DISPLAY BY ID

The screenshot displays the Postman application interface. The top navigation bar includes 'Home', 'Workspaces', and 'API Network'. The left sidebar shows 'My Workspace' with a collection named 'REST API basics: CRUD, test & variable'. The main panel shows a GET request to 'http://localhost:3000/elections/display/65aa9e3ba3b79025bda09447'. The response is a JSON object with the following structure:

```
1 {
2   "success": true,
3   "data": {
4     "_id": "65aa9e3ba3b79025bda09447",
5     "title": "government elections",
6     "description": "elections are considered as the most important event ever...",
7     "date": "2024-04-15T10:00:00",
8     "createdAt": "2024-01-19T16:07:23.017Z",
9     "__v": 0
10  }
11 }
```

The status bar at the bottom indicates 'Status: 200 OK', 'Time: 80 ms', and 'Size: 573 B'. A notification at the bottom of the Postman window states: 'Votre écran est partagé par le biais de l'application meet.google.com. Arrêter le partage Masquer'.

DELETE

The screenshot displays the Postman application interface. At the top, the navigation bar includes 'Home', 'Workspaces', and 'API Network'. A search bar labeled 'Search Postman' is present. The left sidebar shows 'My Workspace' with a list of API collections, including 'REST API basics: CRUD, test & variable'. The main panel shows a 'DELETE' request to the URL 'http://localhost:3000/elections/delete/65aac665a3b79025bda09471'. The request is configured with 'DELETE' as the method and the URL. The 'Body' tab is selected, showing a JSON response in 'Pretty' format. The response status is '200 OK' with a time of '73 ms' and a size of '686 B'. The response body is a JSON object indicating a successful deletion.

```
1 {
2   "success": true,
3   "message": "Election deleted successfully",
4   "data": {
5     "_id": "65aac665a3b79025bda09471",
6     "title": "test123",
7     "description": "A district assembly election happens when only voters or constituents who reside within an electoral district vote
8       in an election held in there",
9     "date": "2025-03-10T10:00:00",
10    "createdAt": "2024-01-19T18:58:45.900Z",
11    "__v": 0
12  }
```


CONCLUSION

In conclusion, our Election Management System leverages advanced tech for secure and efficient administration. Its structured design and ongoing enhancements ensure adaptability, promoting transparency in the democratic process.

The background features a soft, abstract illustration of a smiling face in shades of blue, purple, and pink. The face is positioned on the left side of the frame. On the right side, there are decorative elements: a white starburst icon near the top and a white wavy line near the bottom. Faint, concentric white circles are also visible on the right side of the background.

THANK YOU