

# Bonus: Adding new endpoint and entity

## Environment Setup

Item	Details
Operating System	Linux
Programming Lang	Go
Git Repo	<a href="https://github.com/Muditha-Kumara/Go/tree/Adding_new_endpoint">https://github.com/Muditha-Kumara/Go/tree/Adding_new_endpoint</a>
Original Project Name	Vehicle Behavior Monitoring System
Git Repo	<a href="https://github.com/Muditha-Kumara/IntelligentDevices">https://github.com/Muditha-Kumara/IntelligentDevices</a>

### Project Structure:

```
API 0.1/  
  cmd/api/main.go  
  internal/api/handlers/  
  internal/api/middleware/  
  internal/api/repository/  
  internal/api/server/  
  internal/api/service/
```

## Introduction

This report documents the add new end point `/setting` to the Go backend codebase for intelligent devices in the context of a Vehicle Behavior Monitoring Project.

# Adding the /setting Endpoint

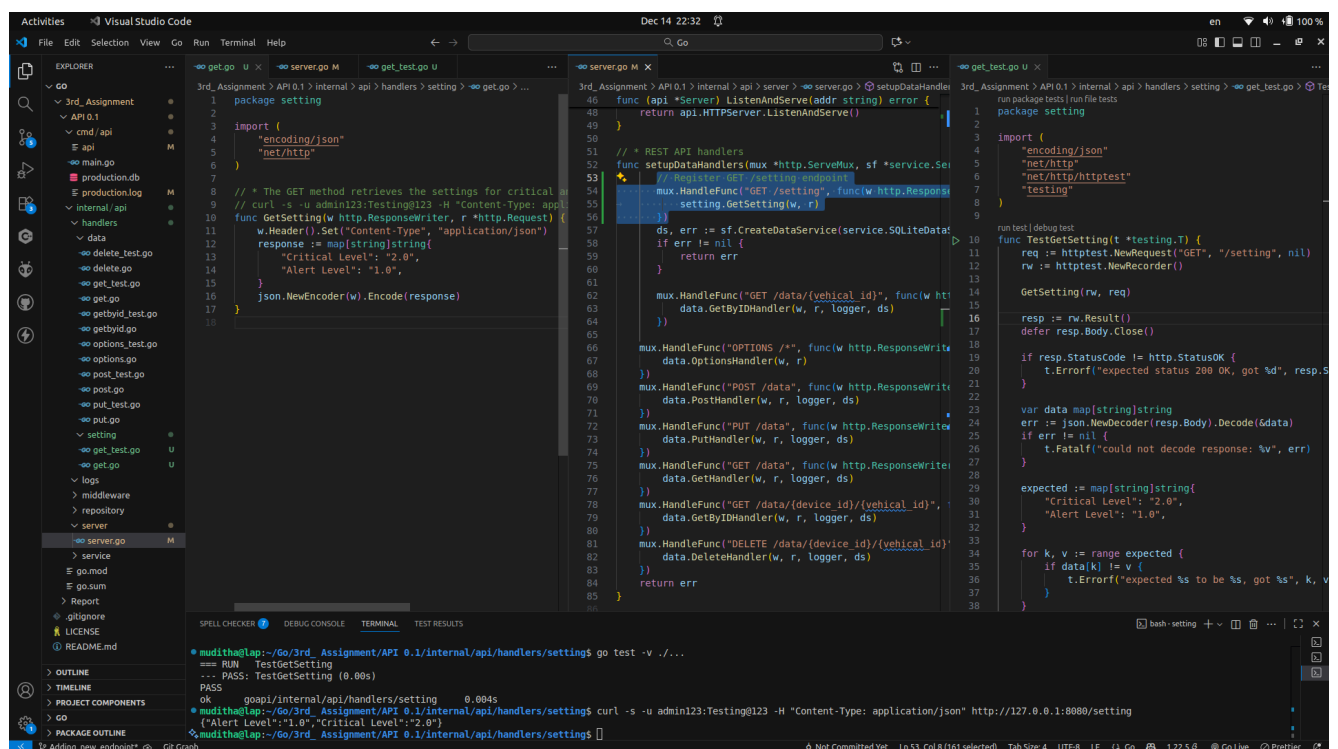
A new GET `/setting` endpoint was added to the API. This endpoint provides system configuration levels for clients and devices. It returns a JSON object with the current critical and alert levels:

```
{
  "Critical Level": "2.0",
  "Alert Level": "1.0"
}
```

This endpoint is protected by basic authentication, consistent with the rest of the API. Manual and automated tests were added to ensure the endpoint returns the correct values and enforces authentication.

## Testing and Debugging

- Unit tests were updated for the new data structure and logic.
- Manual API testing was performed using curl and Postman.
- The rapid POST logic was tested by sending two requests within 1 minute and confirming the correct response.
- The new `/setting` endpoint was tested to ensure it returns the expected configuration JSON.



The screenshot displays the Visual Studio Code interface with three main panels. The left panel shows the Explorer view with a project structure including `3rd_Assignment`, `API`, `cmd`, `api`, `main.go`, `production.db`, `internal`, `handlers`, `data`, `delete_test.go`, `delete.go`, `get_test.go`, `getbyid_test.go`, `getbyid.go`, `options_test.go`, `options.go`, `post_test.go`, `post.go`, `put_test.go`, `put.go`, `setting_test.go`, `setting.go`, `logs`, `middleware`, `repository`, `server`, `server.go`, `service`, `go.mod`, `go.sum`, `Report`, `gitignore`, `LICENSE`, and `README.md`. The middle panel shows the `setting.go` file with the following code:

```
package setting

import (
    "encoding/json"
    "net/http"
)

// The GET method retrieves the settings for critical and alert levels
func GetSetting(w http.ResponseWriter, r *http.Request) {
    w.Header().Set("Content-Type", "application/json")
    response := map[string]string{
        "Critical Level": "2.0",
        "Alert Level": "1.0",
    }
    json.NewEncoder(w).Encode(response)
}
```

The right panel shows the `setting_test.go` file with the following code:

```
package setting

import (
    "encoding/json"
    "net/http"
    "net/http/httptest"
    "testing"
)

func TestGetSetting(t *testing.T) {
    req := httptest.NewRequest("GET", "/setting", nil)
    rw := httptest.NewRecorder()
    GetSetting(rw, req)
    resp := rw.Result()
    defer resp.Body.Close()

    if resp.StatusCode != http.StatusOK {
        t.Errorf("expected status 200 OK, got %d", resp.StatusCode)
    }

    var data map[string]string
    err := json.NewDecoder(resp.Body).Decode(&data)
    if err != nil {
        t.Fatalf("could not decode response: %v", err)
    }

    expected := map[string]string{
        "Critical Level": "2.0",
        "Alert Level": "1.0",
    }

    for k, v := range expected {
        if data[k] != v {
            t.Errorf("expected %s to be %s, got %s", k, v, data[k])
        }
    }
}
```

The bottom panel shows the terminal output of the test command:

```
munitha@lap:~/Go/3rd_Assignment/API 0.1/internal/api/handlers/setting$ go test -v ./...
=== RUN TestGetSetting
--- PASS: TestGetSetting (0.00s)
PASS
ok      goapi/internal/api/handlers/setting  0.004s
munitha@lap:~/Go/3rd_Assignment/API 0.1/internal/api/handlers/setting$ curl -s -u admin123:Testing@123 -H "Content-Type: application/json" http://127.0.0.1:8080/setting
{"Alert Level":"1.0","Critical Level":"2.0"}
munitha@lap:~/Go/3rd_Assignment/API 0.1/internal/api/handlers/setting$
```

