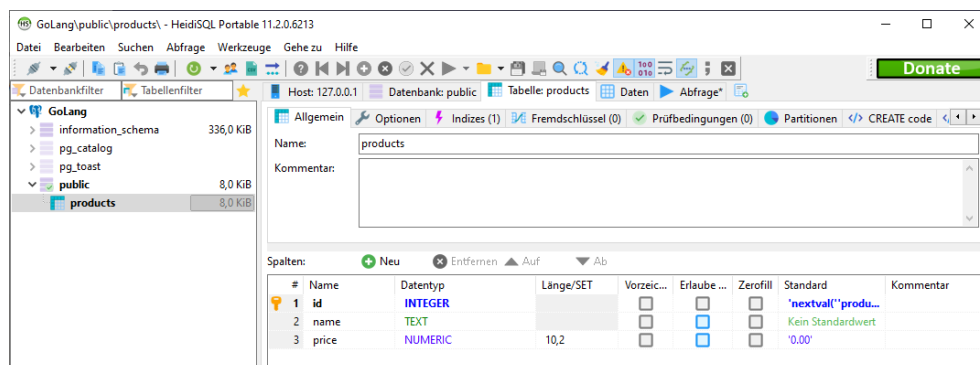


Exercise 02 - Programming in Go

<https://github.com/mleister97/cicd-golang>

Setup

- Installed Go <https://golang.org/doc/install>
- Installed Postgres <https://www.postgresql.org/download>
- Heidi SQL <https://www.heidisql.com/download.php>

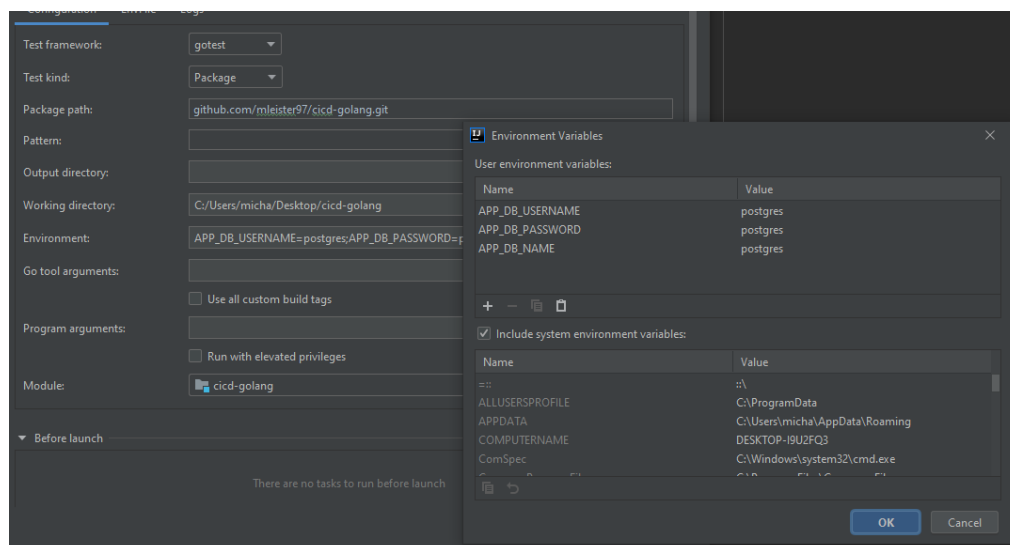


```
go mod init github.com/mleister97/cicd-golang
go get -u github.com/gorilla/mux
go get -u github.com/lib/pq
```

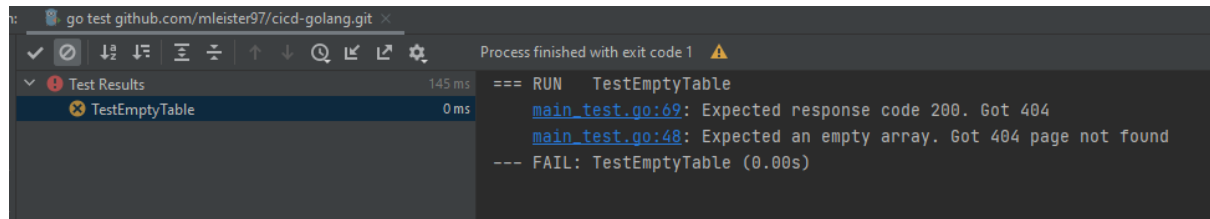
I was facing the following issue:

2021/03/25 14:52:56 pq: SSL is not enabled on the server

Issue was: Environment variables were not set.



Testing:

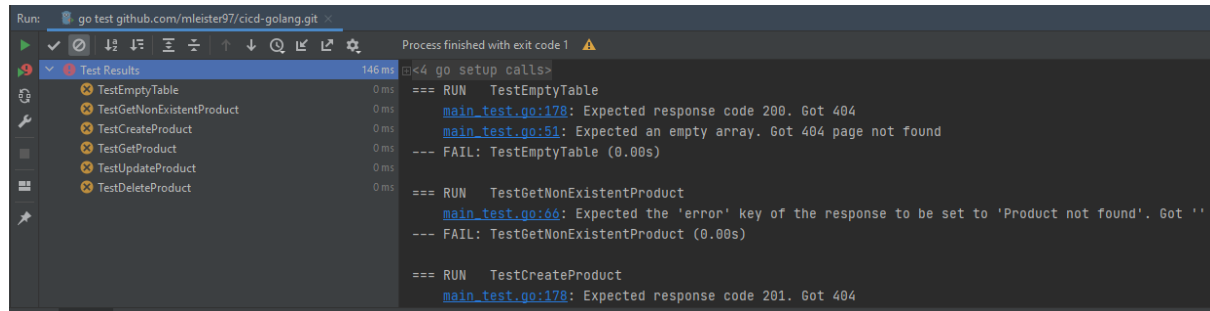


The screenshot shows a VS Code terminal window with the command `go test github.com/mleister97/cicd-golang.git`. The output indicates that the test `TestEmptyTable` failed. The failure is due to two assertions: the first expected a response code of 200 but got 404, and the second expected an empty array but got a "404 page not found" message. The test execution time was 0.00s.

```
Process finished with exit code 1

Test Results 145 ms
  TestEmptyTable 0 ms

=== RUN    TestEmptyTable
main_test.go:69: Expected response code 200. Got 404
main_test.go:48: Expected an empty array. Got 404 page not found
--- FAIL: TestEmptyTable (0.00s)
```



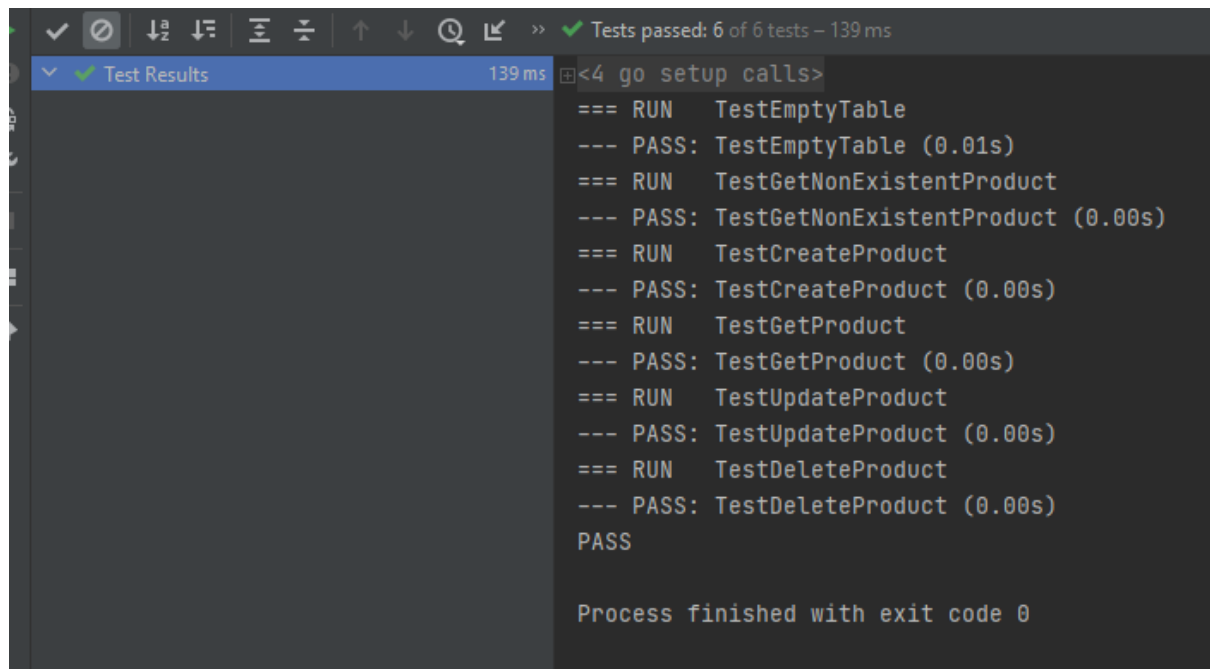
This screenshot shows the results of running multiple Go tests. The tests `TestEmptyTable`, `TestGetNonExistentProduct`, and `TestCreateProduct` all failed. The failure for `TestGetNonExistentProduct` is because the 'error' key in the response was not set to 'Product not found'. The test `TestCreateProduct` failed because it expected a response code of 201 but got 404. The test `TestEmptyTable` failed for the same reasons as in the first screenshot. The test `TestGetProduct` and `TestUpdateProduct` passed. The test `TestDeleteProduct` also passed. The total execution time for the tests was 146 ms.

```
Test Results 146 ms
  TestEmptyTable 0 ms
  TestGetNonExistentProduct 0 ms
  TestCreateProduct 0 ms
  TestGetProduct 0 ms
  TestUpdateProduct 0 ms
  TestDeleteProduct 0 ms

=== RUN    TestEmptyTable
main_test.go:178: Expected response code 200. Got 404
main_test.go:51: Expected an empty array. Got 404 page not found
--- FAIL: TestEmptyTable (0.00s)

=== RUN    TestGetNonExistentProduct
main_test.go:66: Expected the 'error' key of the response to be set to 'Product not found'. Got ''
--- FAIL: TestGetNonExistentProduct (0.00s)

=== RUN    TestCreateProduct
main_test.go:178: Expected response code 201. Got 404
```



This screenshot shows the results of running the same Go tests as in the previous screenshots, but all of them have now passed. The test `TestEmptyTable` passed with a response code of 200 and an empty array. The test `TestGetNonExistentProduct` passed because the 'error' key in the response was correctly set to 'Product not found'. The test `TestCreateProduct` passed because it received a response code of 201. The test `TestGetProduct` and `TestUpdateProduct` also passed. The test `TestDeleteProduct` passed. The total execution time for the tests was 139 ms.

```
Tests passed: 6 of 6 tests - 139 ms

Test Results 139 ms
  TestEmptyTable 0 ms
  TestGetNonExistentProduct 0 ms
  TestCreateProduct 0 ms
  TestGetProduct 0 ms
  TestUpdateProduct 0 ms
  TestDeleteProduct 0 ms

=== RUN    TestEmptyTable
--- PASS: TestEmptyTable (0.01s)

=== RUN    TestGetNonExistentProduct
--- PASS: TestGetNonExistentProduct (0.00s)

=== RUN    TestCreateProduct
--- PASS: TestCreateProduct (0.00s)

=== RUN    TestGetProduct
--- PASS: TestGetProduct (0.00s)

=== RUN    TestUpdateProduct
--- PASS: TestUpdateProduct (0.00s)

=== RUN    TestDeleteProduct
--- PASS: TestDeleteProduct (0.00s)

PASS

Process finished with exit code 0
```

Custom Feature #1

Add a “available” attribute to the database. Modify the existing code. Only return products which are available in the GET routes. Write test-cases for the availability.

Changes can be found in a separate commit called “*Add availability feature*”.

Custom Feature #2

I will add another endpoint function that will allow me to ONLY update the price. If you pass more details to the request, they will be ignored. Write test-cases for the new implemented route. Changes can be found in a separate commit called “*Add update price only feature*”.