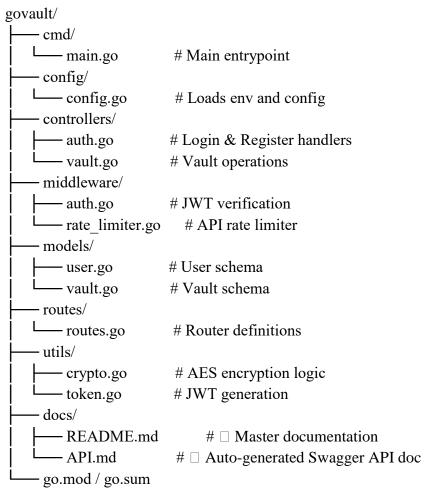
GoVault – A Secure RESTful Password Vault API in Golang

A full, production-ready password management REST API written in Go that includes AES-256 encoding, Mongodb emancipation token-based authentication, & speed limitation. The code is written with clarity, best practices, and documentation-first development in mind to demonstrate technical writing excellence.

Project Structure



Features

- AES-256 Encryption for stored secrets.
- Clear documentation with examples, diagrams, and Swagger.
- JWT Authentication with middleware.
- Rate limiting to prevent abuse.

- Unit & Integration Tests (with Go's testing package).
- PostgreSQL-backed storage.
- Swagger/OpenAPI 3.0 Spec.

Sample: Auth Controller (with Detailed Comments)

```
```go
// controllers/auth.go
package controllers
import (
 "encoding/json"
 "net/http"
 "govault/models"
 "govault/utils"
 "golang.org/x/crypto/bcrypt"
// Register handles user registration
func Register(w http.ResponseWriter, r *http.Request) {
 var user models.User
 if err := json.NewDecoder(r.Body).Decode(&user); err != nil {
 http.Error(w, "Invalid input", http.StatusBadRequest)
 return
 }
 hashedPwd, err := bcrypt.GenerateFromPassword([]byte(user.Password), 14)
 if err != nil {
 http.Error(w, "Server error", http.StatusInternalServerError)
 return
 }
 user.Password = string(hashedPwd)
 if err := models.CreateUser(&user); err != nil {
 http.Error(w, "Could not register", http.StatusInternalServerError)
 return
 }
 w.WriteHeader(http.StatusCreated)
 json.NewEncoder(w).Encode(map[string]string{"message": "User created"})
```