Developer's Hub (Cyber Security Internship)
Submitted by Hasan Raza

Week 3: Advanced Security and Final Reporting

1. Basic Penetration Testing

```
┌──(hasanraza㉿kali)-[~/nodegoat]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d4:26:2f brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
       valid_lft 61976sec preferred_lft 61976sec
```

<u>Command Used:</u>

```
┌──(hasanraza㉿kali)-[~/nodegoat]
└─$ nmap -sV -p- 10.0.2.15

Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-07-31 02:18 IST
Nmap scan report for 10.0.2.15
Host is up (0.0000040s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE    SERVICE VERSION
3306/tcp  filtered mysql
4000/tcp  open     http    Node.js Express framework
27017/tcp filtered mongod

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.66 seconds
```

This revealed that port 4000 is open and running a Node.js Express service. MongoDB (27017) and MySQL (3306) were filtered, indicating a firewall or restricted access is in place. No unnecessary services were exposed.

2. Set Up Basic Logging

Installed using:

npm install winston

Purpose:

Winston is added to log important application and security events. It helps monitor server behavior, track issues, and maintain logs both in the console and in a file (security.log).

```
// Added: Winston logging setup [TASK 2]
const winston = require("winston");
const logger = winston.createLogger({
    transports: [
        new winston.transports.Console(),
        new winston.transports.File({ filename: "security.log" })
    ]
});
logger.info(" Application started"); // Log server startup

MongoClient.connect(db, (err, db) => {
    if (err) {
        logger.error("❌ DB connection error"); // Winston logging
        console.log("Error: DB: connect");
        console.log(err);
        process.exit(1);
    }
    logger.info("Connected to the database"); //  Winston logging
    console.log(`Connected to the database`);
```

```
// Insecure HTTP connection
http.createServer(app).listen(port, () => {
    logger.info(`🌐  Express server listening on port ${port}`); // Winston logging
    console.log(`Express http server listening on port ${port}`);
});
```

A Winston logger is initialized to capture logs in both the terminal and a file named security.log.
logger.info("Application started") confirms the application is running.
This logging system helps with monitoring, debugging, and future auditing.

Winston is used to log database connection status:

- logger.error(...) captures DB connection failures.
- logger.info(...) confirms successful DB connection and server startup.

These logs help in diagnosing issues quickly and keeping a record of application lifecycle events.

Confirmed by:

Winston was configured to log important security and operational events such as application startup, database connection, and server status. Logs are recorded in both the terminal output and a persistent file (security.log).

Purpose: Confirms Winston is logging to the console.

Verification: Terminal logs confirm that Winston is actively tracking app events.

```
{"level":"info","message":" Application started"}
{"level":"info","message":"Connected to the database"}
Connected to the database
{"level":"info","message":"🌐  Express server listening on port 4000"}
Express http server listening on port 4000
```

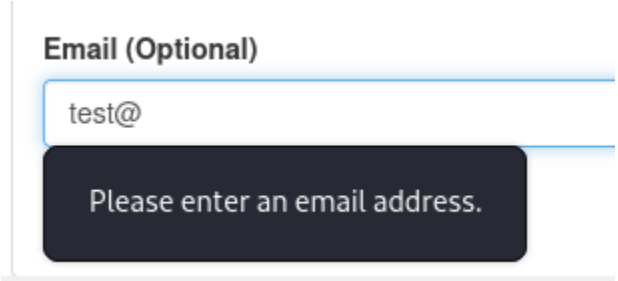
Shows presence of `security.log` file in project directory, confirming Winston is saving logs persistently:

```
┌──(hasanraza㉿kali)-[~/nodegoat]
└─$ ls
CODE_OF_CONDUCT.md  Dockerfile    LICENSE   README.md  app.json   config        docker-compose.yml  nodemon.json       package.json  server.js
CONTRIBUTING.md     Gruntfile.js  Procfile  app        artifacts  cypress.json  node_modules        package-lock.json  security.log  test

┌──(hasanraza㉿kali)-[~/nodegoat]
└─$ cat security.log
{"level":"info","message":" Application started"}
{"level":"info","message":"Connected to the database"}
{"level":"info","message":"🌐  Express server listening on port 4000"}
```

3. Create a Simple Checklist

Input Validation

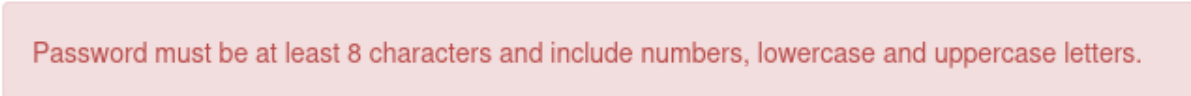
- All user inputs validated using regex and `validator` library
- Emails verified with `validator.isEmail`
- 

HTTPS and Secure Headers

- Helmet.js enabled for secure HTTP headers
- In production, HTTPS must be enforced

Password Security

- Passwords hashed with `bcrypt` before storage
-
```
"_id" : 4,
"userName" : "hasanraza2049",
"firstName" : "hasan",
"lastName" : "raza",
"benefitStartDate" : "2045-11-07",
"password" : "$2b$10$WIcY8EtDsd499G2vOYXJ0egglPCTGJXX3Xdj2as8zci7hoUXEeDiy"
```

- Minimum 8-character complexity enforced
- 

Authentication

- JWT token-based authentication with 1h expiry
- Session-based access control

Logging

- Winston logger captures login attempts and server events

- ```
  $ cat security.log
  level":"info","message":" Application started"}
  level":"info","message":"Connected to the database"}
  level":"info","message":"🌐  Express server listening on port 4000"}
  ```

Other

- MongoDB not exposed publicly
- JWT sent via httpOnly cookie to prevent XSS token theft
  ```