Hosting a Secure Web Application Server

**Project description/goals:**

For our project we will be making an online server that hosts our website. To make our server secure and efficient we will have to enable new user creation, configure SSH for key-based login, set file permissions for the web server, and secure the server using a firewall. We will also be doing a cron job to restart the service automatically. We will write a backup script that compresses the web directory, schedule it with cron for daily backups. The server is configured for automated deployment using cron jobs: whenever changes are pushed to the GitHub repository, the website automatically updates. This setup will allow us to host simple static sites and emphasizes basic security practices, reliability, and ease of maintenance.

**Platform of choice:** WebServer.

**Demonstration Plan:** using an SSD

**Requirements:** We need a Linux machine and install nmap, open-ssh, nginx packages, and a domain name. We will need to setup basic system security, schedule cron jobs for backup and website update from GitHub, create services with systemd, use SSH for authentication and configure a firewall (if we can using UFW).

**Major technical solutions compared** 1) Another solution to the scenario is that we use Ubuntu to host a secure online web server instead of Debian. 2) Nginx has better and easier performance in static site hosting scenarios, but we can also consider using Apache instead of nginx for its similarity.3) We can also consider using Firewalld instead of UFW. It’s a dynamic firewall manager used in many Red Hat–based systems and is more flexible than UFW, but UFW seems more beginner-friendly.

4) Also, we thought about making a morse code decoder with Arduino and an online server. However, after some research, we found that making an internet connection with an Arduino is too much to handle in 3 weeks, since we do not have any experience or knowledge about that.

**Timeline**

**Week 1:** Start with updating/upgrading the system and installing the web server (nginx) we will need to get our domain name server, create the website to be uploaded in the server.

**Week 2:** Work and implement the project as much as possible. We will need to configure our DNS to point to the public IP address. We need to set up the system security by disabling password authentication and enabling login through SSH keys

**Week 3:** Finalize, make sure that we schedule regular backups through cron jobs or systemd and present the project.

**Team composition:**

Talia Muro (2050123), Mariam Salim (6229528), Selihom Efrem Ogbe (2057195)