EPITA Hub – Final Project Documentation

Author: Paul Norais
Student ID: 35608891
EC2 Public IP: 13.54.212.81
Domain Name (DNS): epitahub.fun
GitHub Repo: https://github.com/Luapsiaron/epita-hub

---

1. Overview

EPITA Hub is a collaborative web platform designed for EPITA students to manage their team projects efficiently. It combines secure user authentication, project tracking, and real-time communication in a unified and intuitive interface.

To ensure privacy and personalized usage, the platform includes a secure login system: users must authenticate themselves before accessing any collaborative features such as project boards, chat, or calendars. This guarantees that only authorized students can participate in team activities.

Key Features:

This project simplifies the complexities of managing user interactions and project data. The core features include:

- **User Authentication** Secure sign-up and login with persistent user data
- **User Profiles** with bio, photo, and campus selection
- **Project Management** with task tracking, member assignment, and deadlines
- **Team Chat System** with public and private channels
- **Shared Calendar** to synchronize team activities
- **Lightweight Storage** using structured JSON files (no SQL database)
- **Easy Deployment** on a single EC2 instance with built-in setup scripts

---

Technologies used:

- `Node.js`, `Express.js` – for building the backend and REST endpoints
- `Socket.IO` – for real-time messaging and updates
- `Apache2` – acting as a reverse proxy for domain and SSL support
- `PM2` – to ensure continuous availability of the Node.js server
- JSON-based file storage – for simplicity and portability

2. Server Setup and Deployment

Full Manual Setup

1. Prepare a EC2 instance
   - Create a new Ubuntu 22.04 EC2 instance on AWS
2. Open the following **ports in the security group**:
   - `22` (SSH)
   - `80` (HTTP)
   - `443` (HTTPS, if DNS)
3. Connect via SSH:

```
❯ ssh -i your-key.pem ubuntu@<your-ec2-ip>
```

4. Update the system and install dependencies

```
❯ sudo apt update && sudo apt upgrade -y
❯ sudo apt install -y apache2 nodejs npm git
```

5. Install PM2

```
❯ sudo npm install -g pm2
```

6. Clone the project repository

```
❯ git clone https://github.com/Luapsiaron/epita-hub.git ~/epita-hub-auth
❯ cd ~/epita-hub-auth
```

7. Install Node.js dependencies

```
❯ npm install socket.io
❯ npm install
```

8. Start the server with PM2 Enable required Apache modules:

```
❯ sudo a2enmod proxy proxy_http
```

Create a new virtual host config:

```
❯ sudo nano /etc/apache2/sites-available/epita-hub.conf
```

Paste the following content:

```
<VirtualHost *:80>
ProxyPreserveHost On
ProxyPass / http://localhost:3000/
ProxyPassReverse / http://localhost:3000/
</VirtualHost>
```

Activate the config:

```
❯ sudo a2dissite 000-default.conf
❯ sudo a2ensite epita-hub.conf
❯ sudo systemctl reload apache2
```

You can now visit http://< your-ip >.

Fast Setup (Automated Script)

If you want to deploy the server quickly on a clean Ubuntu machine, run the following:

1. Prepare a EC2 instance

   - Create a new Ubuntu 22.04 EC2 instance on AWS

2. Open the following **ports in the security group**:

   - `22` (SSH)
   - `80` (HTTP)
   - `443` (HTTPS, if DNS)

3. Connect via SSH:

   ```
   ❯ ssh -i your-key.pem ubuntu@<your-ec2-ip>
   ```

4. Download the Setup script: Download setup_server.sh

   ```
   ❯ curl -O https://raw.githubusercontent.com/luapsiaron/epita-hub/main/scripts/setup_server.sh
   ```

5. Make the script executable and run it:

   ```
   ❯ chmod +x setup_server.sh
   ❯ ./setup_server.sh
   ```

This will install dependencies, clone the repo, install PM2, start the Node server, and set up Apache2 as a reverse proxy — all automatically.

3. Project Structure & Key Files

File / Directory	Description
`server.js`	Main backend server using Express and Socket.IO
`projects.json`	Stores all project-related data
`users.json`	Contains user credentials, bios, and profiles
`messages.json`	Chat message history
`channels.json`	Channel configurations (public & private)
`public/`	Frontend HTML/JS/CSS files
`scripts/`	Server management scripts
`setup_server.sh`	Auto-installs and configures the whole stack

4. Script Documentation

`setup_server.sh`

```bash
#!/bin/bash

# Update system packages
sudo apt update && sudo apt upgrade -y

# Install required tools
sudo apt install -y apache2 nodejs npm git

# Install PM2 globally
sudo npm install -g pm2

# Clone the project
git clone https://github.com/Luapsiaron/epita-hub.git ~/epita-hub-auth

# Install Node dependencies
cd ~/epita-hub-auth
npm install socket.io
npm install

# Start the app with PM2
pm2 start server.js --name epita-hub
pm2 save

# Configure Apache
sudo a2enmod proxy proxy_http
sudo tee /etc/apache2/sites-available/epita-hub.conf > /dev/null <<EOF
<VirtualHost *:80>
    ProxyPreserveHost On
    ProxyPass / http://localhost:3000/
    ProxyPassReverse / http://localhost:3000/
</VirtualHost>
EOF

sudo a2dissite 000-default.conf
sudo a2ensite epita-hub.conf
sudo systemctl reload apache2
```

This script automates the full setup of the EPITA Hub platform on a new EC2 Ubuntu instance. It installs system packages, clones the project, installs Node dependencies, configures PM2 to run the server continuously, and sets up Apache as a reverse proxy. It eliminates manual configuration and ensures consistent, repeatable deployment.

You can use the `redeploy.sh` script to restore a previous backup of the server.

```
./scripts/redeploy.sh [archive-name].tar.gz
```

This will:

- Remove the current project directory
- Extract the specified backup
- Restart the server via PM2

5. Linking the Server with a DNS Domain Name

To make the platform accessible via a custom domain name, we connected the public IP of the EC2 instance to a DNS entry and configured HTTPS encryption. Here's how:

DNS Configuration (Hostinger)

1. Log in to your Hostinger account.
2. Go to **DNS Zone Editor** for your domain (e.g., `epitahub.fun`).
3. Add an **A record** pointing your domain to the **public IPv4** of your EC2 instance:
   - **Name**: `@`
   - **Type**: `A`
   - **Points to**: `your-public-ip` -**TTL**: `300`

HTTPS Configuration with Let's Encrypt

To secure the server with HTTPS:

1. Install **Certbot**:

```
❯ sudo apt install certbot python3-certbot-apache -y
```

2. Run Certbot to generate the SSL certificate and configure Apache:

```
❯ sudo certbot --apache
```

3. Follow the instructions to select your domain and enable automatic HTTPS redirection.

4. Once completed, Certbot:

   - Creates SSL certificates using **Let's Encrypt**
   - Updates the Apache virtual host config (`epita-hub-le-ssl.conf`)

6. Server Scripts

Script	Purpose
`restart.sh`	Restarts the server and clears PM2 logs
`backup.sh`	Stops the server, creates a timestamped backup, then restarts it
`redeploy.sh`	Replaces the project directory with a backup and restarts
`setup_server.sh`	Full deployment script

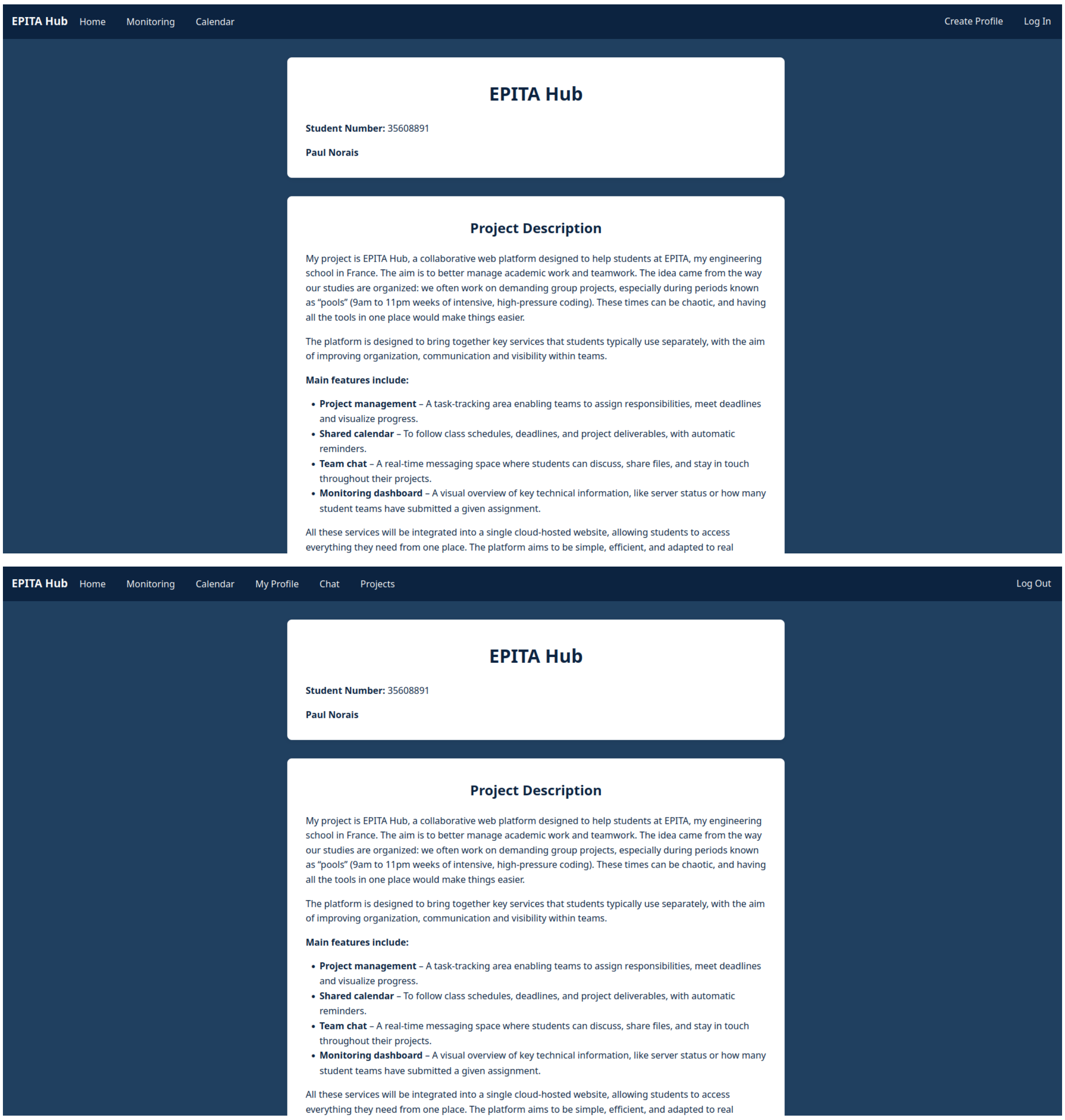
Each script includes comments to explain what it does and when to use it.

---

7. Testing & Verification

To test your deployment:

- Open your browser at: `http://[YOUR_PUBLIC_IP]`
- Try to:
  - Sign up with a user
  - Log in and edit your profile
  - Send a message in a public channel
  - Create a project and assign members



EPITA Hub

Student Number: 35608891

Paul Norais

Project Description

My project is EPITA Hub, a collaborative web platform designed to help students at EPITA, my engineering school in France. The aim is to better manage academic work and teamwork. The idea came from the way our studies are organized: we often work on demanding group projects, especially during periods known as "pools" (9am to 11pm weeks of intensive, high-pressure coding). These times can be chaotic, and having all the tools in one place would make things easier.

The platform is designed to bring together key services that students typically use separately, with the aim of improving organization, communication and visibility within teams.

Main features include:

- **Project management** – A task-tracking area enabling teams to assign responsibilities, meet deadlines and visualize progress.
- **Shared calendar** – To follow class schedules, deadlines, and project deliverables, with automatic reminders.
- **Team chat** – A real-time messaging space where students can discuss, share files, and stay in touch throughout their projects.
- **Monitoring dashboard** – A visual overview of key technical information, like server status or how many student teams have submitted a given assignment.

All these services will be integrated into a single cloud-hosted website, allowing students to access everything they need from one place. The platform aims to be simple, efficient, and adapted to real

EPITA Hub

Student Number: 35608891

Paul Norais

Project Description

My project is EPITA Hub, a collaborative web platform designed to help students at EPITA, my engineering school in France. The aim is to better manage academic work and teamwork. The idea came from the way our studies are organized: we often work on demanding group projects, especially during periods known as "pools" (9am to 11pm weeks of intensive, high-pressure coding). These times can be chaotic, and having all the tools in one place would make things easier.

The platform is designed to bring together key services that students typically use separately, with the aim of improving organization, communication and visibility within teams.

Main features include:

- **Project management** – A task-tracking area enabling teams to assign responsibilities, meet deadlines and visualize progress.
- **Shared calendar** – To follow class schedules, deadlines, and project deliverables, with automatic reminders.
- **Team chat** – A real-time messaging space where students can discuss, share files, and stay in touch throughout their projects.
- **Monitoring dashboard** – A visual overview of key technical information, like server status or how many student teams have submitted a given assignment.

All these services will be integrated into a single cloud-hosted website, allowing students to access everything they need from one place. The platform aims to be simple, efficient, and adapted to real

Top: user not logged in — Bottom: user logged in and accessing features

Video Demonstration

A short video demonstration is available to show the full setup process using the `setup_server.sh` script (Fast setup method).
It walks through the EC2 instance configuration, script execution, and launching the platform.

To test real-time:

- Open the chat in two browsers and check live updates

To test scripts:

```
./backup.sh
./restart.sh
./redeploy.sh
```

8. Common Issues & Fixes

Problem	**Solution**
Apache default page is shown	Ensure 000-default is disabled and `epita-hub.conf` enabled
Socket.IO error	Run `npm install socket.io` explicitly
PM2 not found	Run `source ~/.bashrc` or restart shell after install

---

9. References

- https://expressjs.com/
- https://socket.io/
- https://pm2.keymetrics.io/
- StackOverflow/GitHub discussions
- AWS EC2 user guide