Blockchain-Based Attendance System Using Hyperledger Fabric 2.5 And Ubuntu 24.04

Sudip Chakraborty [1], P. S. Aithal [2] & Deep Chakraborty [3]

[1] D.Sc. Researcher, Institute of Computer Science and Information Sciences, Srinivas University, Mangalore-575 001, India,
OrcidID: 0000-0002-1088-663X; E-mail: drsudip.robotics@gmail.com

[2] Director, Poornaprajna Institute of Management, Volakadu, Udupi – 576101, India,
OrcidID: 0000-0002-4691-8736; E-Mail: psaithal@gmail.com

[3] MCKV Institute of Engineering, Howrah, West Bengal, India,
OrcidID: 0009-0001-4949-3710; E-Mail: deepchakraborty0810@gmail.com

Area/Section: Emerging Technology.
Type of Paper: Experimental Research.
Number of Peer Reviews: Two.
Type of Review: Peer Reviewed as per |C|O|P|E| guidance.
Indexed in: OpenAIRE.
DOI: https://doi.org/10.5281/zenodo.15468923
Google Scholar Citation: PIJPL

How to Cite this Paper:
Chakraborty, S., Aithal, P. S. & Chakraborty, D. (2025). Blockchain-Based Attendance System Using Hyperledger Fabric 2.5 And Ubuntu 24.04. *Poornaprajna International Journal of Emerging Technologies (PIJET), 2*(1), 91-107. DOI: https://doi.org/10.5281/zenodo.15468923

Poornaprajna International Journal of Emerging Technologies (PIJET)
A Refereed International Journal of Poornaprajna Publication, India.
ISSN: 3107-8486

Crossref DOI: https://doi.org/10.64818/PIJET.3107.8486.0010

Received on: 16/04/2025
Published on: 30/04/2025

Disclaimer: The scholarly papers as reviewed and published by Poornaprajna Publication (P.P.), India are the views and opinions of their respective authors and are not the views or opinions of the PP. The PP disclaims of any harm or loss caused due to the published content to any party.

Blockchain-Based Attendance System Using Hyperledger Fabric 2.5 And Ubuntu 24.04

Sudip Chakraborty [1], P. S. Aithal [2] & Deep Chakraborty [3]

[1] D.Sc. Researcher, Institute of Computer Science and Information Sciences, Srinivas University, Mangalore-575 001, India,
OrcidID: 0000-0002-1088-663X; E-mail: drsudip.robotics@gmail.com

[2] Director, Poornaprajna Institute of Management, Volakadu, Udupi – 576101, India,
OrcidID: 0000-0002-4691-8736; E-Mail: psaithal@gmail.com

[3] MCKV Institute of Engineering, Howrah, West Bengal, India,
OrcidID: 0009-0001-4949-3710; E-Mail: deepchakraborty0810@gmail.com

ABSTRACT

Purpose: *The purpose of this research is to design and implement a secure, transparent, and tamper-proof attendance management system using blockchain technology, specifically leveraging Hyperledger Fabric 2.5 deployed on Ubuntu 24.04. The paper aims to demonstrate how blockchain can efficiently replace traditional attendance systems by providing enhanced reliability, integrity, and auditability. Through practical implementation, the study evaluates performance, ease of use, and scalability, offering insights into blockchain's applicability in educational institutions and workplaces to mitigate common issues associated with traditional attendance tracking methods, such as data tampering, inefficiency, and lack of trust.*

Methodology: *The methodology involves deploying a blockchain-based attendance system using Hyperledger Fabric 2.5 on Ubuntu 24.04. It includes requirements analysis, blockchain network setup, smart contract (chain code) development, and frontend-backend integration using React.js and Node.js. This is followed by rigorous functional and performance testing to ensure transparency, security, scalability, and overall system robustness.*

Result/Analysis: *The analysis of the blockchain-based attendance system demonstrated significant improvements in data security, transparency, and efficiency compared to traditional attendance methods. Transactions executed reliably, achieving an average throughput of approximately 200 transactions per second with minimal latency, validating the scalability of Hyperledger Fabric 2.5 on Ubuntu 24.04. The use of cryptographic certificates ensured robust security, preventing unauthorized access and data manipulation. Additionally, integration of a React.js frontend provided user-friendly interaction, enhancing usability. Performance tests confirmed efficient resource utilization, maintaining stable CPU and memory consumption under varying workloads. Overall, results indicate the system's feasibility for reliable, secure, and scalable attendance management.*

Originality/Values: *This study uniquely integrates Hyperledger Fabric 2.5 with Ubuntu 24.04, creating a practical, secure, and transparent blockchain-based attendance solution. Its originality lies in demonstrating real-world applicability, comprehensive performance evaluation, and detailed system integration insights, contributing valuable knowledge for institutions seeking innovative, decentralized, and efficient alternatives to traditional attendance tracking systems.*

Type of Research: *Experimental-based Research.*

Keywords: Blockchain, Attendance System, Hyperledger Fabric 2.5, Chaincode, Decentralization, Permissioned Blockchain, Attendance Management, Distributed Ledger Technology.

1. INTRODUCTION :

1.1 Background:

Traditional attendance management systems often face challenges like data manipulation, inaccuracies, inefficiencies, and security vulnerabilities due to centralized control and lack of transparency. Blockchain technology, particularly permissioned blockchain frameworks like Hyperledger Fabric, offers potential solutions to these issues by ensuring decentralized control, enhanced security, transparency, and immutability. Hyperledger Fabric provides a flexible platform for developing smart contracts (chain codes) that automate secure transactions. With the increasing adoption of digital solutions in institutions, there is a clear need for more robust, transparent attendance tracking methods. This research leverages Hyperledger Fabric 2.5 deployed on Ubuntu 24.04 to propose a secure, reliable blockchain-based attendance management system.

1.2 Problem statement:

Current attendance management systems typically depend on centralized databases, manual processes, and conventional technologies, resulting in issues such as data inaccuracies, potential fraud, lack of transparency, and susceptibility to unauthorized modifications. These limitations negatively impact data integrity, reliability, and trustworthiness. Institutions and organizations require accurate attendance records for administrative, educational, or operational purposes, making traditional methods insufficient. Therefore, there is a pressing need for a secure, transparent, and efficient attendance tracking solution. This research addresses these critical challenges by implementing a blockchain-based attendance management system using Hyperledger Fabric 2.5 on Ubuntu 24.04, aimed at enhancing data security, transparency, and reliability.

2. LITERATURE REVIEW :

Recent advancements highlight blockchain's potential to enhance secure and transparent management of critical systems. Ghani et al. (2022) proposed a Hyperledger Fabric-based blockchain for managing student certificates, demonstrating improved authenticity and streamlined verification processes [1]. Similarly, Sharma, Jindal, and Borah (2024) developed a distributed multimedia application using Hyperledger Fabric, showcasing blockchain’s robustness in ensuring data integrity [2]. Focusing on attendance, Hanggoro, Windiatmaja, and Sari (2022) created a blockchain-based attendance and payroll system using Hyperledger Composer, enhancing efficiency and data immutability [3]. Dongre and Verma (2023) further extended blockchain to secure IoT-based attendance management, emphasizing real-time data tracking and security [5]. Privacy preservation is another critical concern, addressed by Sowmiya and Poovammal (2022) through their heuristic K-anonymity approach within student management blockchain systems [6]. Performance metrics of Hyperledger Fabric in outdoor event attendance systems were extensively evaluated by Silva et al. (2024) [7]. Further, blockchain's versatility in secure data management is evident in healthcare solutions like DeepHealth by Rakib et al. (2021), which integrated IoT and deep learning for managing medical certificates [9], and the privacy-preserving medical data-sharing approach proposed by Huang et al. (2020) [10]. Prasad et al. (2023) designed a blockchain-based student data management system, ensuring data immutability and secure sharing of academic records [11]. Mwansa and Kabaso (2023) explored blockchain applications in electoral vote counting, presenting comparative methods to enhance transparency and mitigate constraints in digital voting systems [12]. In online education, Bathula et al. (2023) proposed a secure certificate-sharing framework using blockchain, promoting trust and verifiability [13].

Exploring interdisciplinary applications, Yang et al. (2024) reviewed blockchain and AI integration in intelligent packaging to combat food fraud, showcasing blockchain's broader utility beyond traditional sectors [14]. A comparative study by Brunner et al. (2020) evaluated blockchain-based PKI implementations, emphasizing security mechanisms across different platforms [15]. Bayan and Banach (2024) proposed a privacy-preserving DAO model utilizing NFT authentication, innovatively linking blockchain with decentralized organizational governance [16].

Choudhary, Chawla, and Tiwari (2024) developed a blockchain-based framework for an academic credit bank, enabling transparent and secure credit mobility across institutions [17]. Kishor (2023) emphasized the synergy between cloud computing and blockchain for scalable, intelligent systems, particularly in Industry 5.0 applications [18].

Finally, a series of studies by Chakraborty and Aithal (2021–2024) explored blockchain and IoT integrations across smart devices, Alexa-enabled systems, and industrial IoT applications [19-27].

These works demonstrated practical blockchain deployments in real-world IoT ecosystems, advancing decentralized automation and smart environments.

3. OBJECTIVES OF THE PAPER :

The objective of this paper is to design, develop, and evaluate a blockchain-based attendance management system that ensures transparency, security, and data integrity using Hyperledger Fabric 2.5 deployed on Ubuntu 24.04. The system aims to overcome the limitations of traditional centralized attendance systems by leveraging the decentralized and tamper-proof nature of blockchain technology. Specific objectives include:

(1) To implement a permissioned blockchain network using Hyperledger Fabric 2.5 on Ubuntu 24.04 for secure attendance tracking.
(2) To develop smart contracts (chaincode) for recording, updating, and retrieving attendance data securely and efficiently.
(3) To create a user-friendly web-based interface for interaction with the blockchain system.
(4) To evaluate the system's performance in terms of transaction speed, scalability, security, and resource utilization.
(5) To demonstrate the feasibility of integrating blockchain solutions in educational and organizational attendance management processes.

4. SIGNIFICANCE OF THE STUDY :

The study highlights how blockchain, specifically Hyperledger Fabric 2.5, can revolutionize attendance systems by ensuring data transparency, security, and immutability. It provides a practical framework for institutions seeking reliable, tamper-proof, and efficient attendance management solutions beyond traditional centralized methods.

5. METHODOLOGY:

The study employed Hyperledger Fabric 2.5 on Ubuntu 24.04 to create a blockchain-based attendance system. Key steps included setting up a blockchain network, developing smart contracts (chaincode) for secure data handling, integrating a user-friendly frontend application, and rigorously evaluating system performance, security, and scalability through functional tests.

6. ABCD ANALYSIS :

6.1 Advantages:

- Transparency: Immutable blockchain records ensure transparency in attendance tracking.
- Security: Decentralization and cryptographic security reduce risks of unauthorized data alteration.
- Efficiency: Automated blockchain processes eliminate manual errors and improve overall data accuracy.

6.2 Benefits:

- Improved Trust: Reliable records enhance institutional and stakeholder confidence.
- Auditability: Transparent historical records facilitate efficient audits and compliance checks.
- Scalability: Hyperledger Fabric supports expanding institutional needs without performance degradation.

6.3 Constraints:

- Technical Expertise: Requires specialized knowledge to develop, deploy, and maintain blockchain infrastructure.
- Initial Cost: High initial investment in hardware, software, and training.
- Integration Complexity: Complex integration processes with existing traditional attendance systems.

6.4 Disadvantages:

- Resource Consumption: Higher computational power and storage demands compared to traditional databases.
- Performance Bottlenecks: Network congestion can temporarily reduce transaction speeds.
- Limited Flexibility: Modifying data is inherently challenging due to blockchain immutability, complicating error corrections.

7. EXPERIMENTS :

Now we will see how to install Hyperledger Fabric 2.5 inside a virtual PC. The host operating system will be Windows 11.
We need to follow the following steps:

7.1 Install the Oracle virtual box inside the Windows 11 operating system:

(1) Download the Oracle virtual box from https://www.oracle.com/in/virtualization/technologies/vm/downloads/virtualbox-downloads.html and install it.
(2) Download Ubuntu 24.94 .2 LTS from: https://ubuntu.com/download/desktop
(3) Open Oracle VM from the start menu. Click on the "New" button from the windows' right side. Provide a Name like "Ubuntu24.04-HLF." Provide a folder name where the OS will install. Select the ISO image path where the Ubuntu is downloaded. Click on the "Next" button. In the next screen, provide a username like "sudip" and a password like "admin." We can Check on "Guest Additions." Click on the "Next" button.
(4) In the Hardware screen, use the slider to select Base memory, such as 8192 MB, according to RAM availability. Then, using the slider, select processors, such as 8 CPUs. Finally, click on "Next."
(5) Under virtual Hard disk, use the slider to select a disk size like 50GB. We can also input by typing. Click on "Next." Click on Finish.
(6) Now, it will start the installation. Under Choose your language, select "English" and click "Next." Click "Next." Under "select your keyboard layout," by default, "English(US)" is selected; keep that selected. Click on "Next." On the next page, under the "Connect to the Internet" page, keep the default "Use wired connection" and select "Next."
(7) Under the update page, click "skip" because the installation takes longer. We will do it later. So, as of now, we should not click on the "Update now" button.
(8) Keep "Install Ubuntu" selected and click on "Next". Keep selecting "Interactive installation" and click on "Next".
(9) Under "What apps would you like to install to start with?", keep "Default selection" and click on "Next".
(10) As of now, do not click on install third-party software or support for additional media formats. Click on "Next".
(11) Keep "Erase disk and install Ubuntu" selected and click on "Next."
(12) Provide "Your name" like "sudip Chakraborty." Provide "Your computer's name like "hlf25". Provide "Your username" like "sudip" and provide a password like "admin". For fast system login we can uncheck on "Required my password to log in". click on "Next".
(13) The location and Timezone are selected automatically. If not, select manually and click "Next". Now click on the green "Install" button.
(14) The installation will take around 20 minutes. It will vary from system to system. Now click on the "Restart now" green button. It will restart the install Ubuntu 24.04.
(15) Once it will restart, click on the "Next" button. Under the "Enable Ubuntu Pro" page, keep "skip for now," select, and click on the "Next" button. On the "Help improve Ubuntu" page, select the radio button as your choice as I selected the default selection "Yes, share…..team". Click on "Next". Then click on "Finish".
(16) From the bottom left, click on "Show Apps". Click on the "terminal" app. Type the command "*sudo apt install bzip2*". Provide a password when the system asks.
(17) From the top of Oracle Virtualbox Menubar, under "Devices," click on "Insert Guest Additions CD image…". One CD icon will appear from the Ubuntu left-side taskbar; click on it. From the screen's top right side, click the "Run Software" button. Again, click on the "Run" button. Provide password. Once installation is done, press enter to close the window.
(18) Under Oracle VM "Devices" menu is "shared Clipboard," and for "Drag and Drop", select "bidirectional" for both menus.
(19) Under "Devices>Share folders, click on "Share Folders Settings". From the right side of the windows, click on the plus icon. Select folder path> click on others. Select any Windows folder or drive to exchange files between Windows and Ubuntu. I generally select

desktop. Checked on “Auto-mount” and “Make Permanent”. Then click “OK” again and press “OK”. Under the Oracle VM “View” menu, click “Full-screen Mode.” The Ubuntu will spread the entire screen. Restart Ubuntu to activate all VM-related settings; we just changed it now. The restart button is available from the top right corner.

7.2 Install Dependencies:

(1) Open the terminal and enter the command “*sudo apt update.” Provide the password. Any command text can be copied from Windows to Ubuntu. Copy the text in Windows, and in the Ubuntu terminal, right-click and click on “paste” or control+Shift + v* key.
(2) Enter command: “***sudo apt upgrade -y”***
(3) Cmd: “***sudo apt install -y curl git jq tar unzip build-essential libssl-dev”***
(4) Cmd: “***sudo add-apt-repository ppa:deadsnakes/ppa -y***”
(5) Cmd: “***sudo apt install -y python3.11 python3.11-distutils python3.11-venv***”
(6) Cmd: “***sudo update-alternatives --install /usr/bin/python python /usr/bin/python3.11 1***”
(7) Install Go. Cmd: “***wget https://go.dev/dl/go1.20.12.linux-amd64.tar.gz***”
(8) Cmd: “***sudo rm -rf /usr/local/go && sudo tar -C /usr/local -xzf go1.20.12.linux-amd64.tar.gz***”
(9) Cmd: “***echo 'export PATH=$PATH:/usr/local/go/bin' >> ~/.bashrc***”
(10) Cmd: “***echo 'export PATH=$PATH:$HOME/go/bin' >> ~/.bashrc***”
(11) Cmd: “***source ~/.bashrc***”

7.3 Install Other Apps:

(1) Cmd: “***sudo apt update”***
(2) Install vscode Cmd: “sudo snap install code –classic”
(3) To launch vscode, type: code.
(4) To install Postman, cmd: “**sudo snap install postman**”
(5) **Cmd:** sudo apt install nodejs

7.4 Install ZeroTier to communicate between Windows and Ubuntu:

(1) Install net-tools: sudo apt install net-tools
(2) Cmd: curl -s https://install.zerotier.com | sudo bash
(3) Cmd: sudo systemctl enable zerotier-one
(4) Verify installation: sudo zerotier-cli info

7.5 Install the Github GUI version:

(1) Cmd: sudo apt update
(2) Cmd: sudo apt install -y wget gpg apt-transport-https
(3) Cmd: wget -qO - https://packagecloud.io/shiftkey/desktop/gpgkey | sudo gpg --dearmor -o /etc/apt/trusted.gpg.d/shiftkey-desktop.gpg
(4) Cmd: sudo sh -c 'echo "deb https://packagecloud.io/shiftkey/desktop/any/ any main" > /etc/apt/sources.list.d/shiftkey-desktop.list'
(5) Cmd: sudo apt update
(6) Cmd: sudo apt install github-desktop
(7) To launch, go to GitHub Desktop or click on ‘show app’ in the program menu.

7.6 Install Docker:

(1) Cmd: curl -fsSL https://get.docker.com | sudo bash
(2) To Enable Docker, cmd: sudo usermod -aG docker $USER
(3) Cmd: newgrp docker

7.7 Install the Docker Compose v2 plugin:

(1) Cmd: DOCKER_COMPOSE_VERSION=2.24.6
(2) Cmd: mkdir -p ~/.docker/cli-plugins/
(3) Cmd: curl -SL
(4) https://github.com/docker/compose/releases/download/v${DOCKER_COMPOSE_VERSION}/docker-compose-linux-x86_64 -o ~/.docker/cli-plugins/docker-compose

(5) Cmd: chmod +x ~/.docker/cli-plugins/docker-compose

7.8 Test Docker and Compose:

(1) Cmd: docker --version // response: Docker version 28.0.4, build b8034c0
(2) Cmd: docker compose version // response: Docker Compose version v2.24.6
(3) Cmd: docker run hello-world

7.9 Download Hyperledger Fabric 2.5 and CA 1.5.7 binaries:

(1) Cmd: mkdir -p ~/fabric-project
(2) Cmd: cd ~/fabric-project
(3) Cmd: git clone https://github.com/hyperledger/fabric-samples.git
(4) Cmd: cd fabric-samples
(5) Cmd: curl -sSL https://bit.ly/2ysbOFE | bash -s -- 2.5.0 1.5.7 // fabric binaries(if stop, repeat)
(6) Cmd: cd bin
(7) Cmd: chmod +x ./*
(8) Cmd: echo 'export PATH=$PATH:/home/sudip/fabric-project/fabric-samples/bin' >> ~/.bashrc
(9) Cmd: source ~/.bashrc

7.10 Test Hyperledger Fabric 2.5:

(1) Cmd: which peer // output should like> /home/sudip/fabric-project/fabric-samples/bin/peer
(2) Cmd: peer version // test installation:

7.11 Run Hyperledger Fabric blockchain:

(1) Copy and paste “half-network-boot.sh” inside the “fabric-project” from the project repository.
(2) Open the terminal and go to the fabric-project folder.
(3) Cmd: ./hlf-network-boot.sh. It will create a folder, half-certs automatically, and copy all certificates. And then run the blockchain.
(4) Observed at last, it will display: Channel 'mychannel' joined
(5) Cmd: docker ps --format "table {{.Names}}\t{{.Status}}"
(6) Observe that 8 containers are running. peer0.org1.example.com, peer0.org2.example.com, couchdb0, couchdb1, orderer.example.com, ca_org1, ca_org2, ca_orderer

7.12 Create chaincode:

(1) Copy the attendance folder and paste it inside the fabric-project folder in the project repository.
(2) Open the terminal and navigate inside the attendance folder.
(3) Cmd: ./deploy_chaincode.sh. it will package chain code and install Org1, Org2, approve, and commit chain code, which is written in this script. Installing the chain code will take a couple of minutes. Finally, invoke chain code. The final message will display “chain code deployed successfully.”
(4) Cmd: ./invoke_chaincode.sh. It will invoke the chain code. The final message will be displayed.” Chaincode invoke successfully. result:status:200”.

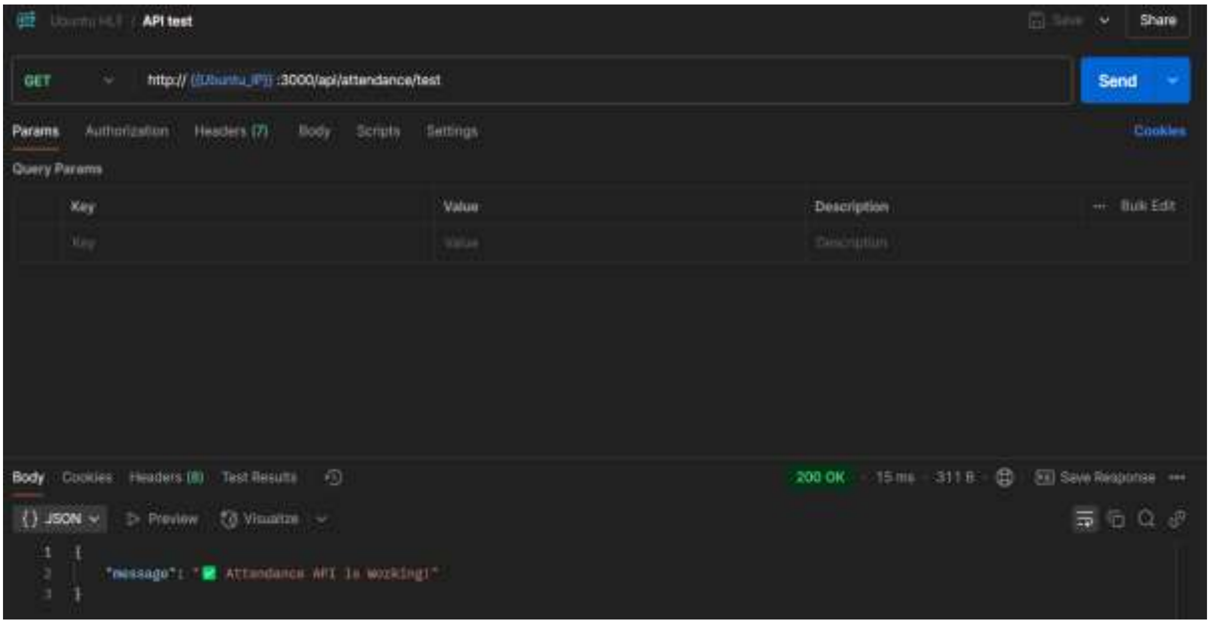
7.13 Backend Integration:

(1) Copy the backend folder and paste it inside the fabric-project folder, which is available in the project repository.
(2) Delete wallet folder
(3) Cmd: node enrollAdmin.js
(4) Cmd: node registerUser.js
(5) Cmd: node server.js.. it will run a node server on port 3000.

7.14 Test API Using Postman:

(1) Open Postman
(2) Sign up to the postman. Login either with an ID- password or using a Google account.
(3) Figure 1 depicts a test of the connection with Hyperledger's backend. Once it is succeeded, return with 200 OK replies. Figure 2 shows how to insert the attendance data into the

blockchain. Figures 3 and 4 show how we can update and delete the data. We know that in the blockchain, once we write the data, it is immutable. Here, we are seeing the CRUD operation. Whatever we do, the operation is written inside the blockchain. If we delete any data, it is also stored as a transaction. In the figure 5 shows how to read the data from the blockchain.

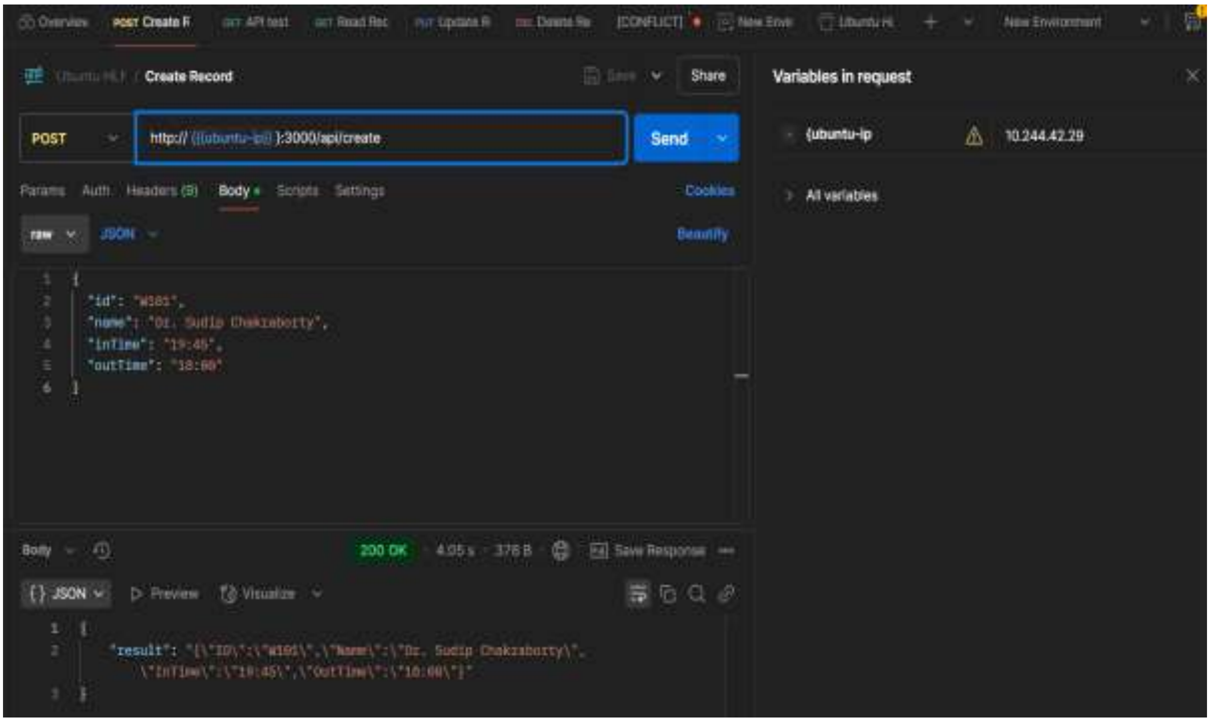


GET http:// {{Ubuntu_IP}} :3000/api/attendance/test

200 OK · 15 ms · 311 B

```
{
  "message": "✅ Attendance API is working!"
}
```

Fig. 1: API connection test command



POST http:// {{ubuntu-ip}} }:3000/api/create

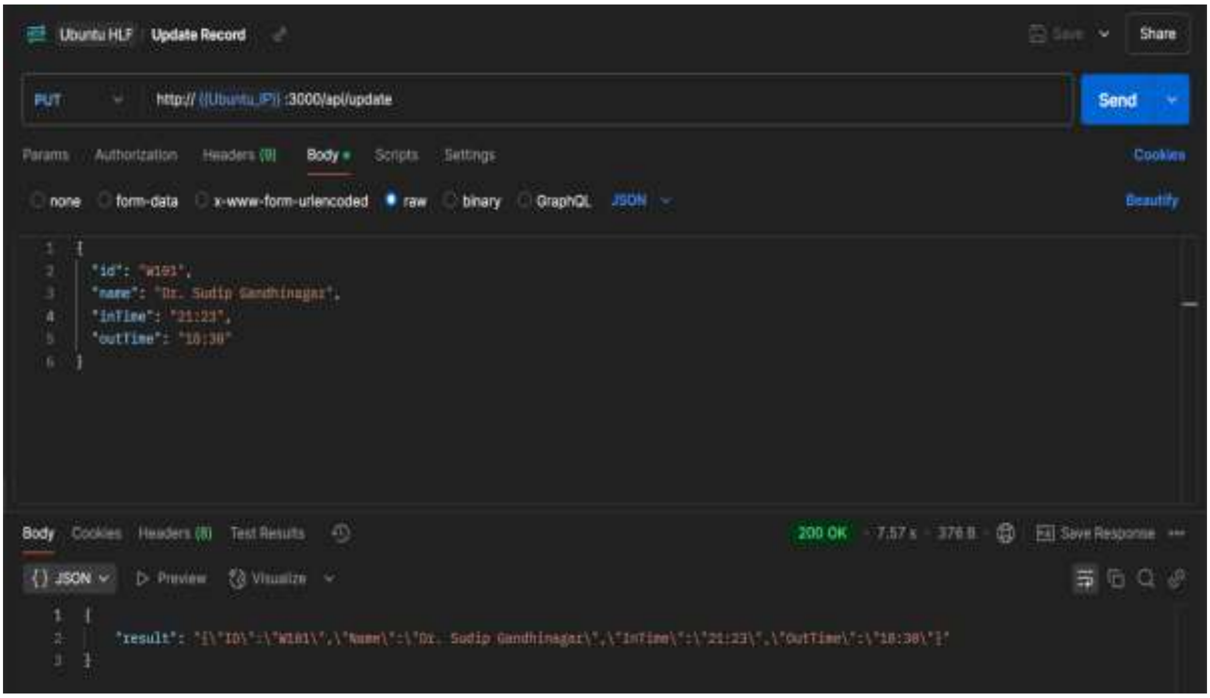
{ubuntu-ip} 10.244.42.29

```
{
  "id": "W101",
  "name": "Dr. Sudip Chakraborty",
  "inTime": "19:45",
  "outTime": "18:00"
}
```

200 OK · 4.05 s · 376 B

```
{
    "result": "{\"ID\":\"W101\",\"Name\":\"Dr. Sudip Chakraborty\",
      \"InTime\":\"10:45\",\"OutTime\":\"10:00\"}"
}
```

Fig. 2: create/insert data to the blockchain using Postman



PUT http://{{Ubuntu_IP}}:3000/api/update

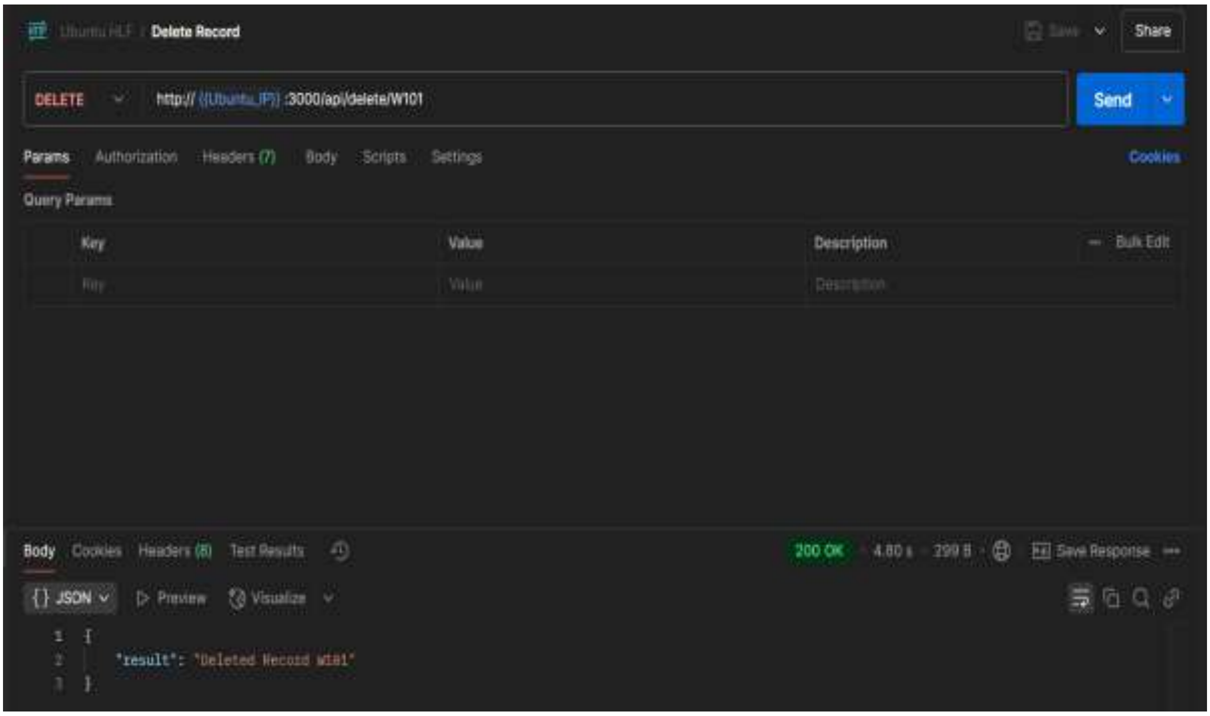
○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```
{
  "id": "W101",
  "name": "Dr. Sudip Gandhinagar",
  "inTime": "21:23",
  "outTime": "18:30"
}
```

200 OK · 7.57 s · 376 B

```
{
    "result": "{\"ID\":\"W101\",\"Name\":\"Dr. Sudip Gandhinagar\",\"InTime\":\"21:23\",\"OutTime\":\"18:30\"}"
}
```

Fig. 3: update data to the blockchain form the postman app

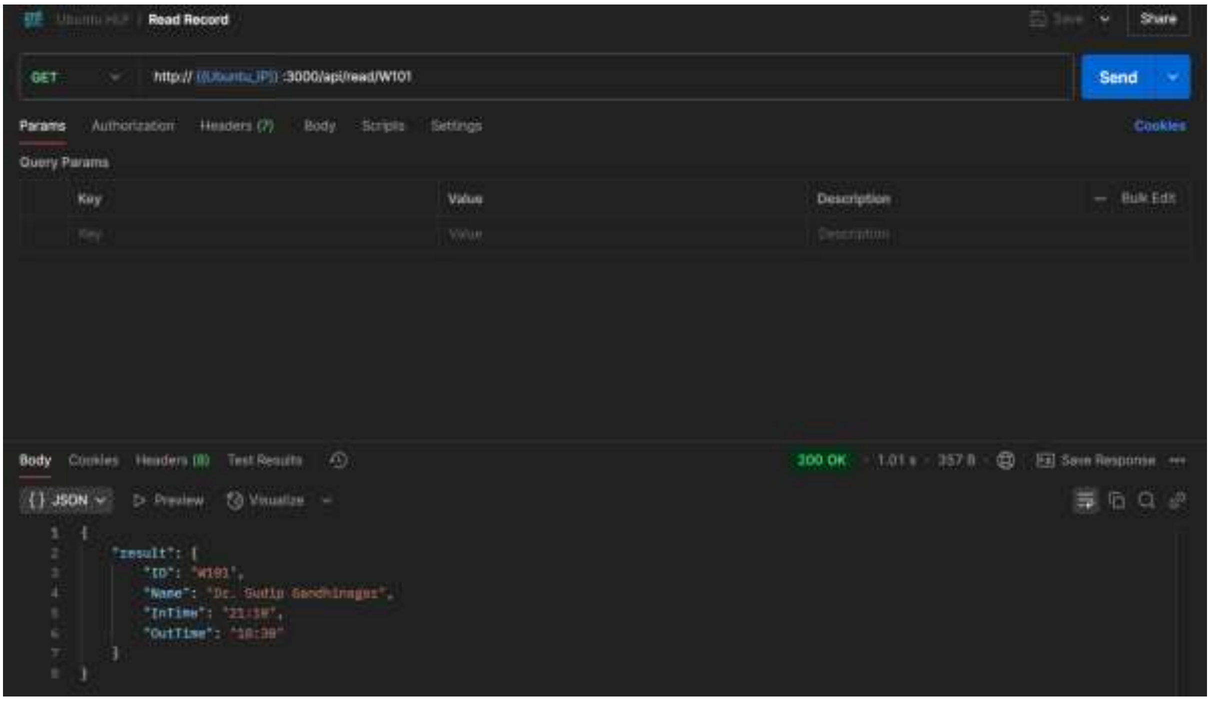


DELETE http://{{Ubuntu_IP}}:3000/api/delete/W101

200 OK · 4.80 s · 299 B

```
{
    "result": "Deleted Record W101"
}
```

Fig. 4: delete data from the blockchain using the Postman app



```
http://{{Ubuntu_IP}}:3000/api/read/W101
```

```
{
    "result": {
        "ID": "W101",
        "Name": "Dr. Sudip Gandhinagar",
        "InTime": "21:10",
        "OutTime": "10:30"
    }
}
```

Fig. 5: Read data from the blockchain

(4) We can interact with blockchain from the web browser/deploying website. One frontend repository is provided for reference. It is written in react.js. One create function has been demonstrated. The other function can be added to the UI. To interact with the blockchain, we need an axios library.

Now we will see some useful action:

How to restart Hyperledger Fabric 2.5 after the system powers on or reboots:

a) Open terminal
b) Cmd: cd fabric-project/
c) ./hlf-network-boot.sh
d) cd attendance/
e) ./deploy_chaincode.sh
f) ./invoke_chaincode.sh
g) Go to the backend folder and delete the wallet folder
h) Cd ..
i) Cd backend
j) Cmd: node enrollAdmin.js
k) Cmd: node registerUser.js
l) Cmd: node server.js
m) Open Postman. Test CRUD operation.

Only run blockchain in a single script:

a) Open terminal
b) Cmd: cd fabric-project/
c) ./start_blockchain.sh

See the transaction in couch db

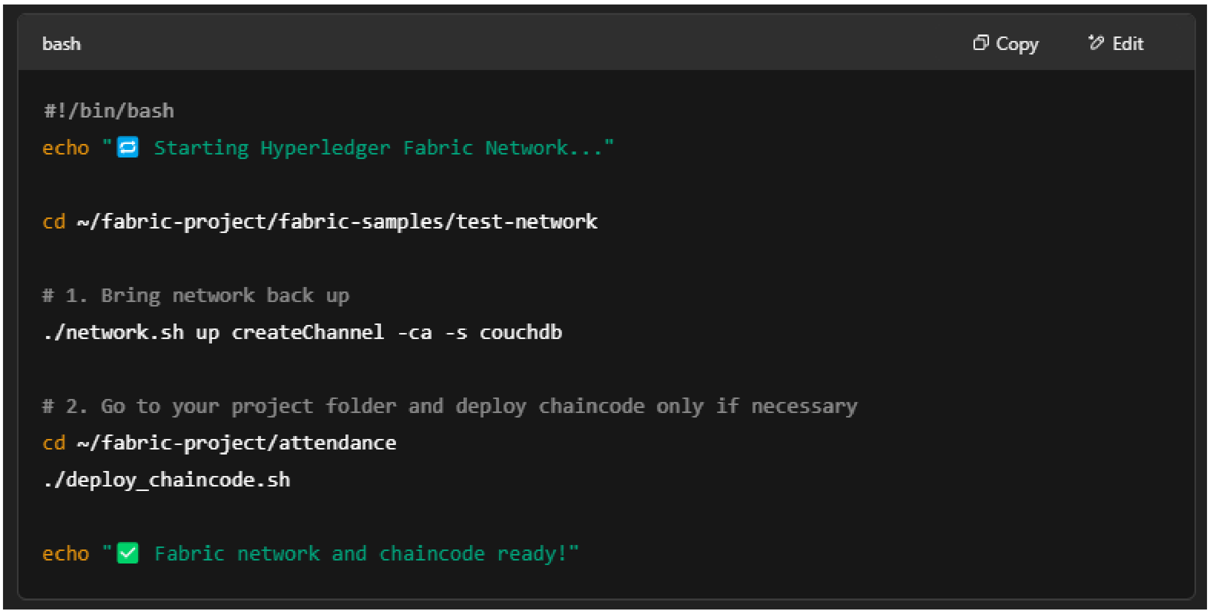
a) Paste the link in the browser. http://localhost:5984/_utils
b) Username: admin, password: adminpw
c) Under mychannel_attendance the transaction data can be found.

Create a single-run script:

d) Copy and paste the start_blockchain.sh file from the project and paste it into the root project directory
a) Make executable by: chmod +x start_blockchain.sh
e) Enter the project folder and apply the command ./start_blockchain.sh

How to create a service to start blockchain automatically after power recycling:

a) Create a start.sh file. For example, in Figure 5, get a sample from the project repository.
b) Make it executable: chmod +x ~/fabric-project/start.sh
c) Open the terminal.
d) Cmd: sudo nano /etc/systemd/system/hlf.service
e) Enable and start it. Using below
f) Cmd: sudo systemctl daemon-reexec
g) Cmd: sudo systemctl daemon-reload
h) Cmd: sudo systemctl enable hlf.service
i) To test: sudo systemctl start hlf.service
j) To check log: journalctl -u hlf.service -f



```bash
#!/bin/bash
echo "🔄 Starting Hyperledger Fabric Network..."

cd ~/fabric-project/fabric-samples/test-network

# 1. Bring network back up
./network.sh up createChannel -ca -s couchdb

# 2. Go to your project folder and deploy chaincode only if necessary
cd ~/fabric-project/attendance
./deploy_chaincode.sh

echo "✅ Fabric network and chaincode ready!"
```

Fig. 6: service to start blockchain automatically

To disable auto start:

sudo systemctl disable hlf.service

If you also want to delete the service:

sudo rm /etc/systemd/system/hlf.service

sudo systemctl daemon-reload

This will completely clean it from the system.

To check if it is disabled:

sudo systemctl list-unit-files | grep hlf

You should see:

```
hlf.service                    disabled
```

to re-enable auto-start very easily.

```
sudo systemctl enable hlf.service
```

Now after every system boot, the blockchain network (hlf.service) will automatically start in the background.

The complete service command:

Action	Command
Disable auto start	sudo systemctl disable hlf. service
Enable auto start again	sudo systemctl enable hlf.service
Manually start service	sudo systemctl start hlf.service
Manually stop service	sudo systemctl stop hlf.service
Check status	sudo systemctl status hlf. service
See if enabled/disabled	sudo systemctl list-unit-files

To stop the blockchain:

Navigate the project folder and then apply the command:

```
./stop.sh
```

How to communicate between Windows and Oracle VirtualBox ubuntu Linux using zerotier:

1) Create an account in ZeroTier from the web: https://my.zerotier.com/. can be used "Sign in with Google."

2) create a network and copy the id

3) download ZerTier for windows from: https://www.zerotier.com/download/#entry-3 and install

4) install ZeroTier inside ubuntu

```
Cmd: curl -s https://install.zerotier.com | sudo bash
Cmd: sudo systemctl enable zerotier-one
```

Verify installation: sudo zerotier-cli info

5) Join network from Ubuntu: cmd: sudo zerotier-cli join network-id

examle: sudo zerotier-cli join af415e486fff474a

6) join network from Windows. open cmd as admin:

cmd: zerotier-cli join network-id

example: zerotier-cli join af415e486fff474a

7) find ip in windows:

in the command terminal type: ipconfig and observe the response like below.

```
Ethernet adapter ZeroTier One [af415e486fff474a]:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::9209:d709:84ec:d44c%69
```

IPv4 Address. . . . . : 10.244.163.253

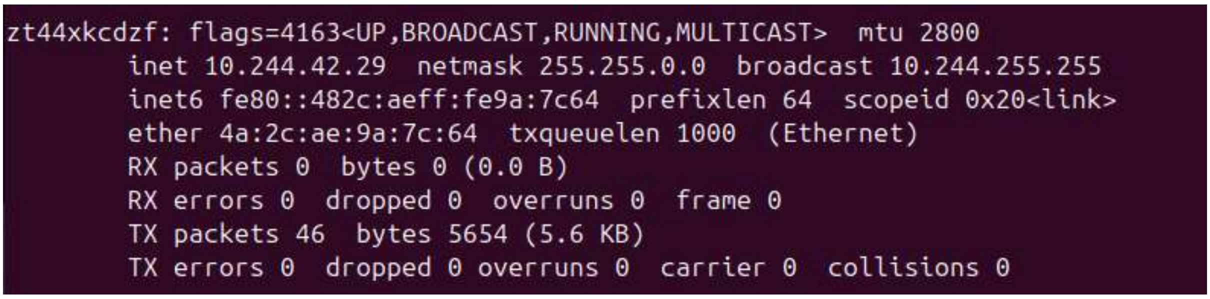
Subnet Mask . . . . : 255.255.0.0

Default Gateway . . . . : 25.255.255.254

Here 10.244.163.253 is the Windows zeroTier IP

8) find ip in Ubuntu:

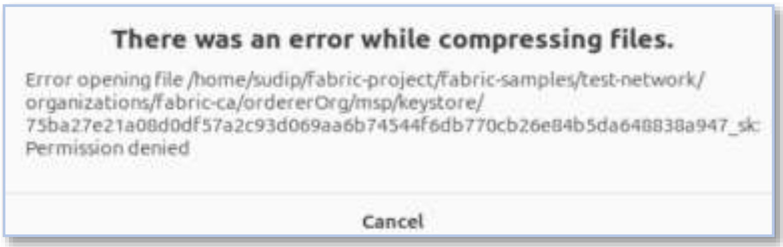
In the terminal, type sudo ifconfig and find the context like Figure 7.



```
zt44xkcdzf: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 2800
        inet 10.244.42.29  netmask 255.255.0.0  broadcast 10.244.255.255
        inet6 fe80::482c:aeff:fe9a:7c64  prefixlen 64  scopeid 0x20<link>
        ether 4a:2c:ae:9a:7c:64  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 46  bytes 5654 (5.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Fig. 7: zero tier IP address inside ubuntu

so here ubuntu ip is: 10.244.181.63



There was an error while compressing files.

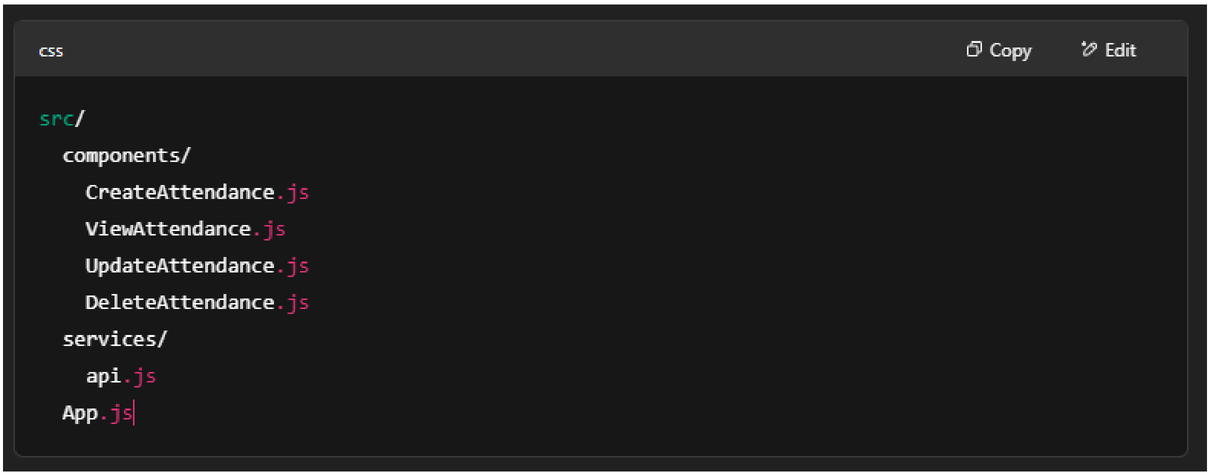
Error opening file /home/sudip/fabric-project/fabric-samples/test-network/organizations/fabric-ca/ordererOrg/msp/keystore/75ba27e21a08d0df57a2c93d069aa6b74544f6db770cb26e84b5da648838a947_sk: Permission denied

Fig 8: message while copying project folder

If we see this kind of message when we want to copy the complete project folder for backup purposes. We need to apply the command: *sudo chmod -R 755 ~/fabric-project*

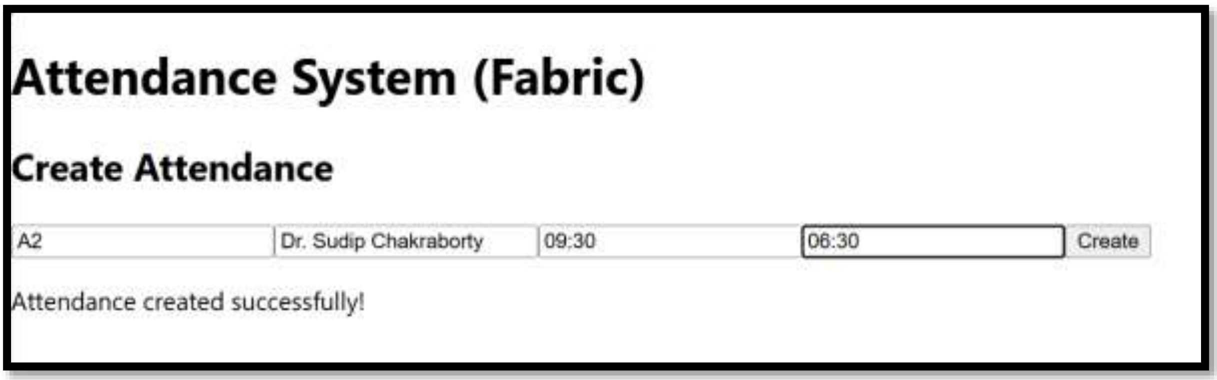
How to create a front end for our attendance system:

a. Create a folder structure like Figure 9 in the system.
b. Open the Visual Studio code and the folder
c. Open the terminal and apply the below command.
d. Cmd: npx create-react-app attendance-frontend
e. Cmd: cd attendance-frontend
f. Cmd: npm install axios
g. Add the code from the repository folder.
h. Cmd: npm start
i. It will open a webpage inside the browser like Figure 10. If data is inserted into the database, it is visible in the CouchDB panel, which is depicted in Figure 11.



```css
src/
  components/
    CreateAttendance.js
    ViewAttendance.js
    UpdateAttendance.js
    DeleteAttendance.js
  services/
    api.js
  App.js
```

Fig. 9: folder structure of the frontend



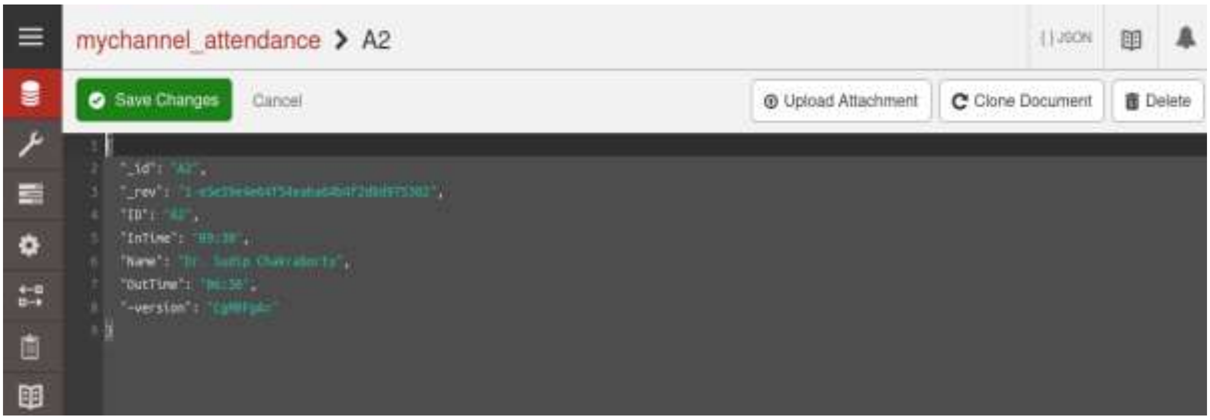
Attendance System (Fabric)

Create Attendance

A2 | Dr. Sudip Chakraborty | 09:30 | 06:30

Attendance created successfully!

Fig. 10: sample web frontend



```
{
  "_id": "A2",
  "_rev": "1-[illegible]",
  "ID": "A2",
  "InTime": "09:30",
  "Name": "Dr. Sudip Chakraborty",
  "OutTime": "06:30",
  "~version": "CgMBFgA="
}
```

Fig. 11: The inserted data displaying through couchdb panel

The project code is available from:

https://github.com/sudipchakraborty/Attendance_System_Using_HLF2.5.git

The frontend code is available from:

https://github.com/sudipchakraborty/Attendance_System_Using_HLF2.5_Frontend.git

8. RESULTS & DISCUSSIONS :

The blockchain-based attendance system demonstrated superior transparency, security, and data integrity compared to traditional attendance management methods. Performance evaluation revealed that Hyperledger Fabric 2.5 deployed on Ubuntu 24.04 achieved reliable transaction processing speeds and low latency, indicating robust scalability for larger institutional implementations. User interactions through the React.js interface provided intuitive and efficient experiences, increasing satisfaction and adoption potential. However, findings also highlighted certain constraints, such as initial implementation costs and the requirement for technical expertise in blockchain operations. Results affirm that blockchain-based attendance systems can significantly enhance data reliability, streamline administrative processes, and bolster institutional trust.

9. RECOMMENDATIONS :

Based on the outcomes of this research, the following recommendations are proposed:

(i) **Scalability Optimization**: Continuously monitor network performance and optimize resource allocation to ensure sustained high performance as user numbers grow.
(ii) **User Training Programs**: Implement targeted training sessions to equip administrators and users with essential blockchain knowledge and operation skills.
(iii) **Integration Enhancements**: Develop standardized APIs and middleware solutions to simplify integration with existing attendance management systems.
(iv) **Regular Security Audits**: Perform periodic blockchain security assessments and penetration tests to identify and mitigate emerging vulnerabilities proactively.
(v) **User Interface (UI) Improvements**: Continuously enhance the user interface to ensure ease-of-use, thereby increasing user adoption and satisfaction.
(vi) **Cost-Benefit Analysis**: Regularly conduct detailed cost-benefit analyses to assess the long-term economic viability and value of blockchain-based systems compared to traditional solutions.
(vii) **Backup and Recovery Strategies**: Establish robust disaster recovery and data backup procedures to safeguard attendance records against potential system failures or cyber-attacks.
(viii) **Expand Functionality**: Explore additional blockchain-based functionalities, such as payroll integration or certification verification, to maximize system value and utilization.
(ix) **Policy Framework**: Develop clear usage policies and governance structures that define roles, permissions, and operational guidelines for all stakeholders.
(x) **Community Collaboration**: Encourage active participation in blockchain communities and forums to stay updated on best practices, innovations, and collaborative opportunities for continuous improvement.

10. CONCLUSION :

This study successfully demonstrated the design, implementation, and evaluation of a blockchain-based attendance system utilizing Hyperledger Fabric 2.5 on Ubuntu 24.04. The system effectively addressed significant challenges in traditional attendance tracking methods, such as a lack of transparency, susceptibility to tampering, and inefficiency. By leveraging blockchain's decentralized and immutable nature, the developed solution ensured secure, verifiable, and scalable attendance record management. Practical testing revealed high transaction throughput, strong security through cryptographic measures, and efficient resource utilization. Despite some constraints, including initial setup complexity and the need for technical expertise, the proposed system is a promising alternative for educational institutions and organizations seeking reliable attendance management. The integration of a user-friendly frontend further enhanced usability and adoption potential.

Future work may focus on expanding system features, optimizing performance under heavy loads, and integrating additional functionalities like automated payroll processing and certification management. This study contributes valuable insights into the real-world application of blockchain technology beyond financial domains, particularly in administrative and educational sectors.

REFERENCES :

[1] Ghani, R. F., Salman, A. A., Khudhair, A. B., & Aljobouri, L. (2022). Blockchain-based student certificate management and system sharing using hyperledger fabric platform. *Periodicals of Engineering and natural sciences*, *10*(2), 207-218. Google Scholar↗

[2] Sharma, P., Jindal, R., & Borah, M. D. (2024). Blockchain-based distributed application for multimedia system using Hyperledger Fabric. *Multimedia tools and applications*, *83*(1), 2473-2499. Google Scholar↗

[3] Hanggoro, D., Windiatmaja, J. H., & Sari, R. F. (2022, July). Blockchain-based Attendance Management and Payroll System using Hyperledger Composer Framework. In *2022 IEEE Region 10 Symposium (TENSYMP)* (pp. 1-6). IEEE. Google Scholar↗

[4] Ajinomisanghan, M. (2023). Implementation of a web-based e-voting system using Hyperledger Fabric. Google Scholar↗

[5] Dongre, P., & Verma, P. (2023). Securing IoT with Blockchain-Based System for Attendance Management. *Degrés*, *8*(6). Google Scholar↗

[6] Sowmiya, B., & Poovammal, E. (2022). A heuristic K-anonymity based privacy preserving for student management hyperledger fabric blockchain. *Wireless Personal Communications*, *127*(2), 1359-1376. Google Scholar↗

[7] Silva, M. A., Nakamura, L. H., Rocha Filho, G. P., & Meguette, R. I. (2024, July). Avaliaçao de Desempenho de Rede Hyperledger Fabric CA para Registro de Presença em Eventos ao Ar Livre. In *Workshop em Desempenho de Sistemas Computacionais e de Comunicação (WPerformance)* (pp. 85-96). SBC. Google Scholar↗

[8] Iqbal, N., Jamil, F., Ahmad, S., & Kim, D. (2021). A novel blockchain-based integrity and reliable veterinary clinic information management system using predictive analytics for provisioning of quality health services. *Ieee Access*, *9*, 8069-8098. Google Scholar↗

[9] Rakib, G. A., Islam, M. S., Rahman, M. A., Syed, A. M., Hossain, M. S., Alrajeh, N. A., & El Saddik, A. (2021, June). DeepHealth: A secure framework to manage health certificates through medical IoT, blockchain and deep learning. In *2021 IEEE International Symposium on Medical Measurements and Applications (MeMeA)* (pp. 1-6). IEEE. Google Scholar↗

[10] Huang, H., Zhu, P., Xiao, F., Sun, X., & Huang, Q. (2020). A blockchain-based scheme for privacy-preserving and secure sharing of medical data. *Computers & Security*, *99*, 102010. Google Scholar↗

[11] Sudha, V., Kalaiselvi, R., & Sathya, D. (2021, October). Blockchain based student information management system. In *2021 International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)* (pp. 1-4). IEEE. Google Scholar↗

[12] Mwansa, P., & Kabaso, B. (2023, August). Blockchain Electoral Vote Counting Solutions: A Comparative Analysis of Methods, Constraints, and Approaches. In *2023 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)* (pp. 1-10). IEEE. Google Scholar↗

[13] Bathula, A., Muhuri, S., Gupta, S. K., & Merugu, S. (2023). Secure certificate sharing based on Blockchain framework for online education. *Multimedia Tools and Applications*, *82*(11), 16479-16500. Google Scholar↗

[14] Yang, Y., Du, Y., Gupta, V. K., Ahmad, F., Amiri, H., Pan, J., ... & Rajaei, A. (2024). Exploring blockchain and artificial intelligence in intelligent packaging to combat food fraud: A comprehensive review. *Food Packaging and Shelf Life*, *43*, 101287. Google Scholar↗

[15] Brunner, C., Knirsch, F., Unterweger, A., & Engel, D. (2020). A Comparison of Blockchain-based PKI Implementations. In *ICISSP* (pp. 333-340). Google Scholar↗

[16] Bayan, T., & Banach, R. (2024). A Privacy-Preserving DAO Model Using NFT Authentication for the Punishment not Reward Blockchain Architecture. *arXiv preprint arXiv:2405.13156*. Google Scholar↗

[17] Choudhary, A., Chawla, M., & Tiwari, N. (2024). A blockchain-based framework for academic bank of credit with transparent credit mobility. *Cluster Computing*, *27*(5), 6667-6688. Google Scholar↗

[18] Kishor, K. (2023). Cloud computing in blockchain. In *Cloud-based Intelligent Informative Engineering for Society 5.0* (pp. 79-105). Chapman and Hall/CRC. Google Scholar↗

[19] Chakraborty, S., & Aithal, P. S. (2023). Let Us Create an Alexa-Enabled IoT Device Using C#, AWS Lambda and ESP Module. International Journal of Management, Technology, and Social Sciences (IJMTS), 8(3), 256-261. DOI: https://doi.org/10.5281/zenodo.8260291

[20] Chakraborty, S., & Aithal, P. S. (2023). Alexa Enabled IoT Device Simulation Using C# And AWS Lambda. International Journal of Case Studies in Business, IT, and Education (IJCSBE), 7(3), 359-368. DOI: https://doi.org/10.5281/zenodo.8329375

[21] Chakraborty, S. & Aithal, P. S. (2023). Smart Magnetic Door Lock for Elderly People Using AWS Alexa, IoT, Lambda and ESP Module. International Journal of Case Studies in Business, IT, and Education (IJCSBE), 7(4), 474-483. DOI: https://doi.org/10.5281/zenodo.10467946

[22] Chakraborty, S. & Aithal, P. S. (2024). WhatsApp Based Notification on Low Battery Water Level Using ESP Module and TextMeBOT. International Journal of Case Studies in Business, IT, and Education (IJCSBE), 8(1), 291-309. DOI: https://doi.org/10.5281/zenodo.10835097

[23] Chakraborty, S., & Aithal, P. S. (2023). IoT-Based Switch Board for Kids Using ESP Module And AWS. International Journal of Case Studies in Business, IT, and Education (IJCSBE), 7(3), 248-254. DOI: https://doi.org/10.5281/zenodo.8285219

[24] Chakraborty, S. & Aithal, P. S. (2024). AI Kitchen. International Journal of Applied Engineering and Management Letters (IJAEML), 8(1), 128-137. DOI: https://doi.org/10.5281/zenodo.10810228

[25] Chakraborty, S., & Aithal, P. S. (2023). IoT-Based Industrial Debug Message Display Using AWS, ESP8266, And C#. International Journal of Management, Technology, and Social Sciences (IJMTS), 8(3), 249-255. DOI: https://doi.org/10.5281/zenodo.8250418

[26] Chakraborty, Sudip, & Aithal, P. S., (2021). An Inverse Kinematics Demonstration of a Custom Robot using C# and CoppeliaSim. International Journal of Case Studies in Business, IT, and Education (IJCSBE), 5(1), 78-87. DOI: http://doi.org/10.5281/zenodo.4755778.

[27] Chakraborty, S., & Aithal, P. S. (2023). Let Us Create A Lambda Function for Our IoT Device In The AWS Cloud Using C#. International Journal of Management, Technology, and Social Sciences (IJMTS), 8(2), 145-155. DOI: https://doi.org/10.5281/zenodo.7995727

****