Industrial Attachment – GizanTech

Embedded Systems

Daily logs

Team – 03

Team Members-

Shah Ahmed Raad_2003042

Zabed Iqbal Chowdhury_2003011

Daily Report Logs

Industrial Attachment – Daily Log (Day 1)
Date: 08.03.2025

Understanding Client Requirements

The day began with an introduction to **client requirement analysis**. We learned how **initial reports** are structured based on client needs and how formal documentation plays a key role in communication.

A structured workflow is followed:

- After preparing the initial report, a **three-day feature-based feasibility analysis** is conducted.
- The bidding process follows this analysis, ensuring that all requirements and technical aspects are thoroughly considered.
- The importance of **team collaboration and thorough analysis** was emphasized as key factors in ensuring client satisfaction.

Development Workflow & Task Management

We were introduced to the **Customer Relationship Management (CRM) and Human Resource (HR) systems**, which play a significant role in organizing tasks efficiently. Additionally, we observed how team discussions and queries are compiled, reviewed, and finalized within reports to maintain clarity and alignment.

Industry Exposure & Workplace Experience

An interactive discussion session provided valuable insights into different career paths. A key topic was the comparison between **Embedded Systems and Web Development**, highlighting why Embedded Systems may offer better opportunities in certain fields.

We also observed the **gap between academic knowledge and industry expectations**, emphasizing the importance of practical experience. The day concluded with an **office tour**, where we observed how teams collaborate and operate within a professional setting.

Hands-On Technical Work

We were introduced to **embedded systems development** and worked with **Arduino UNO and ESP32**. Under supervision, we performed **basic setup and testing**, gaining insight into the fundamental aspects of embedded programming.

Team Project Assignment

As part of the industrial attachment, we were assigned to a **group project on the Fingerprint Sensor System**. Our primary responsibility is to track **daily progress** and document findings, with a **final report** to be submitted at the end of the attachment period.

Challenges Faced

- Adapting to technical terminology and professional communication standards was initially challenging but became easier with guidance.
- Understanding the structured workflow of a professional setting required some adjustment, particularly in terms of documentation and reporting.

The first day provided a strong foundation for understanding client interactions, project workflows, and hands-on technical work.

Industrial Attachment – Daily Log (Day 2)
Date: 09.03.2025

Working with the R309 Fingerprint Sensor

The primary task for the day was to interface the **R309 fingerprint sensor** with the **ESP32 microcontroller**. We carefully followed the wiring connections and verified the setup before running the test program. However, despite multiple attempts, the sensor did not produce any output.

Debugging & Issue Diagnosis

To identify the root cause, we conducted a series of troubleshooting steps:

- **Checked hardware connections** to confirm correct pin alignment and wiring integrity.
- **Verified power supply levels** to ensure the sensor was receiving adequate voltage.
- **Tested different baud rates and serial communication settings** to eliminate configuration issues.
- **Reviewed the code** to verify correct implementation of serial communication.

Despite these efforts, the sensor remained unresponsive. After multiple failed attempts, we suspected that the issue was hardware-related rather than a software or wiring error.

Database Setup with MySQL

In addition to hardware debugging, we worked on setting up the **database** for the project. Using **MySQL**, we created the necessary database structure to store fingerprint data, which will later be used for authentication purposes. This involved:

- Designing the **database schema**, including tables for storing fingerprint templates and user details.
- Implementing **basic SQL queries** to insert, retrieve, and manage data.
- Ensuring that the database is structured efficiently for future integration with the fingerprint sensor system.

Task Adjustment – Storing Fingerprints as IDs

During discussions with the team, it was decided that **fingerprint data should not be stored as an image** but instead be stored as an **ID**. The R309 sensor processes fingerprint scans and generates a unique identifier for each fingerprint, which will be saved in the database rather than

raw fingerprint images. This approach improves **storage efficiency, security, and processing speed** when verifying fingerprints against the database.

Key Takeaways & Challenges Faced

- Debugging embedded systems requires a structured approach to eliminate potential failure points systematically.
- Hardware issues can sometimes be mistaken for software errors, making **alternative hardware testing essential** for accurate diagnosis.
- **Database management is a crucial part of the project**, as it will be responsible for storing and retrieving fingerprint data for authentication.
- **Fingerprint recognition systems store data as unique IDs**, rather than images, for efficiency and security.

Although we could not get the fingerprint sensor to work today, we successfully set up the database and finalized the **fingerprint storage method**, marking an important step toward system development. The next step will be to further investigate the ESP32 issue and integrate the fingerprint sensor with the database.

Industrial Attachment – Daily Log (Day 3)
Date: 10.03.2025

Troubleshooting the R309 Fingerprint Sensor with ESP32

Following the unsuccessful attempts to interface the **R309 fingerprint sensor** with the **ESP32** on the previous day, we shifted our focus to further troubleshooting. To determine whether the issue was with the sensor or the microcontroller, we decided to test the sensor using an **Arduino board** instead.

After making the necessary wiring connections and uploading the test code, we found that the **sensor functioned correctly with the Arduino**, confirming that the sensor itself was not faulty. This result indicated that the **problem was specific to the ESP32 setup**, prompting us to investigate further.

Identifying the Issues with ESP32

With the sensor working on Arduino but not on ESP32, we needed to pinpoint the exact reason for the failure. At this stage, **Rafiul Islam**, our project supervisor, assisted us in diagnosing the problem. Through debugging and analysis, we identified two critical issues:

1. **Incorrect Voltage Supply**

- We had initially powered the fingerprint sensor using the **5V output from the ESP32**. However, the sensor requires a **minimum of 3.3V** to operate properly. Later, we connected the fingerprint sensor with 3V power supply and it worked just fine.

2. **Software Bug in the Code**

- In addition to the voltage issue, there was a minor **bug in the code** that prevented the ESP32 from properly establishing serial communication with the fingerprint sensor.
- Our instructor helped us debug the code, identifying and fixing the error, allowing for a more stable communication setup.

Lessons Learned & Key Takeaways

- **Hardware compatibility matters:** Even if components are theoretically compatible, voltage and power requirements must always be verified before interfacing.
- **Debugging must follow a systematic approach:** Testing with alternative hardware (such as switching from ESP32 to Arduino) helped isolate whether the issue was hardware-related or software-related.

- **Small coding errors can lead to major issues:** A minor mistake in the program can prevent communication between components, reinforcing the importance of careful debugging and code reviews.
- **Seeking expert guidance accelerates problem-solving:** Instructor assistance helped us quickly identify the root cause, avoiding prolonged trial-and-error debugging.

Next Steps

With the correct voltage requirement identified and the code issue resolved, the next step will be to **retest the fingerprint sensor with the ESP32 using an appropriate power source**. This will allow us to verify whether the sensor can function correctly with the ESP32 under optimal conditions.

Industrial Attachment – Daily Log (Day 4)
Date: 11.03.2025

Fingerprint Recognition Bug – Debugging & Resolution

The primary focus of the day was debugging an issue with the **R309 fingerprint sensor** integration. The system was incorrectly recognizing a fingerprint **before any input was provided**, which prevented proper enrollment and verification.

Troubleshooting the Issue

To identify the root cause, we followed a systematic debugging approach:

- **Reviewed the sensor connection and wiring** to ensure there were no hardware faults.
- **Tested the code logic** to verify that the fingerprint enrollment function was being called at the correct time.
- **Analyzed serial output logs** to track unexpected sensor behavior and data processing errors.

Despite multiple attempts, the issue persisted, leading us to suspect a deeper compatibility issue between the software and hardware.

Identifying the Root Cause – ESP32 Board Manager Version

After an extensive debugging session, we discovered that the **problem was caused by the version of the ESP32 Board Manager**. We had been using the latest version, but the fingerprint sensor library required **ESP32 version 3.0.4** for stable operation. The newer version caused unexpected behavior in the fingerprint enrollment process.

Solution & Key Takeaways

- **Version compatibility is crucial** – Not all updates improve functionality; some may break compatibility with certain libraries.
- **Systematic debugging helps isolate issues** – Instead of making random changes, methodically testing different components (hardware, software, library versions) leads to a more effective solution.
- **Rolling back software versions** can sometimes resolve unexpected errors, especially when working with embedded systems.

Next Steps

With the correct **ESP32 version (3.0.4) installed**, the next step will be to reattempt fingerprint enrollment and proceed with database integration.

Industrial Attachment – Daily Log (Day 5)
Date: 12.03.2025

Database Connectivity & Web Integration

The main objective for the day was to establish communication between the **PHP backend and MySQL database** and ensure that the system could send and retrieve data as required.

Tasks Completed

1. PHP and MySQL Integration

- Successfully connected the **PHP backend** with the **MySQL database**, allowing the system to interact with stored fingerprint data.
- Tested the connection to ensure stability and responsiveness.

2. Implementing Wi-Fi Connectivity

- Configured the ESP32 to connect to a **Wi-Fi network**, enabling real-time communication with the server.
- Ensured that the network setup was reliable and that the device could consistently stay connected.

Key Takeaways & Challenges Faced

- The **database and Wi-Fi connection** worked well, but keeping the connection stable is important.

Next Steps

With **database and network integration complete**, the focus will now shift to refining system performance and testing real-world scenarios.

Industrial Attachment – Daily Log (Day 6)
Date: 13.03.2025

Final Code Integration & Testing

Until Day 5, we worked on the system in a modular approach, developing and testing each component separately. On Day 6, we focused on merging all the modules into a single, complete system.

Tasks Completed

1. Code Merging & Deployment

- Integrated all individual modules to create the final version of the system.
- Uploaded the merged code to the ESP32 and performed debugging to ensure stability.

2. Implementing OTA Updates

- Configured Over-The-Air (OTA) updates, allowing the ESP32 firmware to be updated remotely.
- Tested the OTA process to ensure smooth deployment of future updates.

3. Local Database Synchronization

- Sent locally stored attendance data to the central MySQL database.
- Verified whether the data was transmitted correctly and without loss.

4. Evaluation & Further Improvements

- The project was evaluated by the instructors, who reviewed the implementation and system performance.
- Based on their feedback, further refinements and improvements were recommended for the system.

Key Takeaways & Challenges Faced

- Merging modules required debugging and optimization to ensure smooth integration.
- OTA updates simplify future maintenance, making it easier to deploy new firmware versions.
- Local database synchronization was successfully tested, confirming the system's ability to store and forward attendance data when a network is available.

Next Steps

- Implement the suggested improvements based on instructor feedback.
- Conduct additional real-world testing to ensure system reliability.