

Lab 8

ICS423 - Internet of Things

Jayant Kolapkar - 2021BCS0132

Question

Task 1: Assuming that the IoT data is available in csv format or json format (atleast 100 rows of values), develop a frontend using reactjs to display the output (refresh the data periodically).
Task 2: Design a CI/CD pipeline using Jenkins to continuously update the codebase of Arduino sketches.

Task 1

Algorithm and Code

1. Create a node server which generates the IOT (temperature data) periodically and stores the data in a CSV file.

Code: server.js

```
const express = require("express");
const fs = require("fs");
const cors = require("cors");
const fastCsv = require("fast-csv");
const app = express();
const PORT = 5000;
const FILE_PATH = "iot_data.csv";
```

```
// Enable CORS
app.use(cors());

// Function to generate random IoT data
const generateData = () => {
    const timestamp = new Date().toISOString();
```

```

const temperature = (Math.random() * 15 + 20).toFixed(2); // Random
temp between 20-35°C
return `${timestamp}, ${temperature}\n`;
};

// Function to write data to CSV file every 2 seconds
const appendData = () => {
    const header = "timestamp,temperature\n";
    if (!fs.existsSync(FILE_PATH)) {
        fs.writeFileSync(FILE_PATH, header);
    }
    setInterval(() => {
        fs.appendFileSync(FILE_PATH, generateData());
    }, 2000);
};

// API endpoint to get the last 100 rows
app.get("/data", (req, res) => {
    const rows = [];
    fs.createReadStream(FILE_PATH)
        .pipe(fastCsv.parse({ headers: true }))
        .on("data", (row) => rows.push(row))
        .on("end", () => {
            res.json(rows.slice(-100)); // Send last 100 rows
        });
});

// Start the server and begin data generation
app.listen(PORT, () => {
    console.log(`Server running on http://localhost:${PORT}`);
    appendData();
});

```

2. Start the server. You will observe that the data will get stored in the CSV file continuously.
3. Create React dashboard by creating the react app and installing the necessary dependencies.

4. Fetch the data from the CSV file on the server and fetch the data using the GET api by integrating the react frontend with node backend.

Code: App.js

```
import React, { useState, useEffect } from "react";
import axios from "axios";
import "./App.css"; // Import external CSS

const API_URL = "http://localhost:5000/data";

const App = () => {
  const [data, setData] = useState([]);
  const [lastUpdated, setLastUpdated] = useState(null);
  const [rowCount, setRowCount] = useState(0);

  useEffect(() => {
    const fetchData = () => {
      axios.get(API_URL)
        .then((response) => {
          setData(response.data);
          setLastUpdated(new Date().toLocaleTimeString()); // Set last
update time
          setRowCount(response.data.length); // Set row count
        })
        .catch((error) => console.error("Error fetching data:", error));
    };

    fetchData(); // Initial fetch
    const interval = setInterval(fetchData, 5000); // Fetch every 5
seconds

    return () => clearInterval(interval); // Cleanup on unmount
  }, []);

  return (
    <div className="container">
```

```

<h2 className="title">IoT Sensor Data</h2>
<p className="info">
    <strong>Last Updated:</strong> {lastUpdated || "Fetching..."} |
    <strong> Rows Fetched:</strong> {rowCount}
</p>
<div className="table-container">
    <table className="styled-table">
        <thead>
            <tr>
                <th>Timestamp</th>
                <th>Temperature (°C)</th>
            </tr>
        </thead>
        <tbody>
            {data.map((row, index) => (
                <tr key={index}>
                    <td>{row.timestamp}</td>
                    <td>{row.temperature}</td>
                </tr>
            )));
        </tbody>
    </table>
</div>
</div>
);

};

export default App;

```

5. Open the localhost and view the updated data on the dashboard.

IoT Sensor Data

Last Updated: 10:10:20 pm | Rows Fetched: 100

Timestamp	Temperature (°C)
2025-03-03T16:37:00.111Z	25.83
2025-03-03T16:37:02.115Z	32.79
2025-03-03T16:37:04.121Z	23.91
2025-03-03T16:37:06.125Z	28.60
2025-03-03T16:37:08.130Z	27.01
2025-03-03T16:37:10.138Z	34.09
2025-03-03T16:37:12.146Z	20.39
2025-03-03T16:37:14.161Z	26.70
2025-03-03T16:37:16.162Z	21.89
2025-03-03T16:37:18.169Z	21.13
2025-03-03T16:37:20.173Z	26.54
2025-03-03T16:37:22.181Z	28.86
2025-03-03T16:37:24.189Z	31.66

IoT Sensor Data

Last Updated: 10:10:35 pm | Rows Fetched: 100

Timestamp	Temperature (°C)
2025-03-03T16:37:16.162Z	21.89
2025-03-03T16:37:18.169Z	21.13
2025-03-03T16:37:20.173Z	26.54
2025-03-03T16:37:22.181Z	28.86
2025-03-03T16:37:24.189Z	31.66
2025-03-03T16:37:26.189Z	28.12
2025-03-03T16:37:28.197Z	20.26
2025-03-03T16:37:30.201Z	23.31
2025-03-03T16:37:32.209Z	33.66
2025-03-03T16:37:34.218Z	34.76
2025-03-03T16:37:36.222Z	20.85
2025-03-03T16:37:38.229Z	32.67
2025-03-03T16:37:40.239Z	20.56

IoT Sensor Data	
Last Updated: 10:10:56 pm Rows Fetched: 100	
Timestamp	Temperature (°C)
2025-03-03T16:37:36.222Z	20.85
2025-03-03T16:37:38.229Z	32.67
2025-03-03T16:37:40.239Z	20.56
2025-03-03T16:37:42.240Z	30.03
2025-03-03T16:37:44.241Z	21.59
2025-03-03T16:37:46.244Z	32.51
2025-03-03T16:37:48.251Z	21.99
2025-03-03T16:37:50.251Z	28.78
2025-03-03T16:37:52.254Z	23.69
2025-03-03T16:37:54.269Z	30.00
2025-03-03T16:37:56.274Z	33.65
2025-03-03T16:37:58.283Z	29.83
2025-03-03T16:38:00.288Z	24.92

6. View and verify the Data getting stored in CSV file.

Name	Date modified	Type	Size
node_modules	03-03-2025 21:42	File folder	
package	03-03-2025 21:42	JSON Source File	1 KB
package-lock	03-03-2025 21:42	JSON Source File	32 KB
server	03-03-2025 21:43	JavaScript Source ...	2 KB
iot_data	03-03-2025 22:11	CSV File	5 KB

A screenshot of a spreadsheet application interface. The top bar shows the cell A1 and a dropdown arrow. To the right of the cells are icons for search, formula (fx), and timestamp. The table has four columns: timestamp, temperature, C, and D. The timestamp column contains dates from March 3, 2025, to March 3, 2025. The temperature column contains values ranging from 20.99 to 34.76. The bottom navigation bar includes arrows for navigating between sheets and a tab labeled "iot_data".

	A1	timestamp	temperature	C	D
1	A	2025-03-03	30.38		
2		2025-03-03	25.83		
3		2025-03-03	32.79		
4		2025-03-03	23.91		
5		2025-03-03	28.6		
6		2025-03-03	27.01		
7		2025-03-03	34.09		
8		2025-03-03	20.39		
9		2025-03-03	26.7		
10		2025-03-03	21.89		
11		2025-03-03	21.13		
12		2025-03-03	26.54		
13		2025-03-03	28.86		
14		2025-03-03	31.66		
15		2025-03-03	28.12		
16		2025-03-03	20.26		
17		2025-03-03	23.31		
18		2025-03-03	33.66		
19		2025-03-03	34.76		
20		2025-03-03	20.85		
21		2025-03-03	32.67		
22		2025-03-03	20.56		
23		2025-03-03	30.03		
24		2025-03-03	21.59		
25		2025-03-03	32.51		
26		2025-03-03	21.99		
27		2025-03-03	28.78		
28		2025-03-03	23.69		
29		2025-03-03			

	A	B	C	D
120	2025-03-03	22.53		
121	2025-03-03	28.86		
122	2025-03-03	32.57		
123	2025-03-03	23.9		
124	2025-03-03	23.58		
125	2025-03-03	35		
126	2025-03-03	21.02		
127	2025-03-03	32.49		
128	2025-03-03	24.45		
129	2025-03-03	26.8		
130	2025-03-03	33.56		
131	2025-03-03	27.57		
132	2025-03-03	21.69		
133	2025-03-03	21.51		
134	2025-03-03	34.26		
135	2025-03-03	26.45		
136	2025-03-03	20.76		
137	2025-03-03	24.94		
138	2025-03-03	27.45		
139	2025-03-03	22.09		
140	2025-03-03	21.28		
141	2025-03-03	27.2		
142	2025-03-03	21.65		
143	2025-03-03	30.92		
144	2025-03-03	27.35		
145	2025-03-03	28.51		
146	2025-03-03	27.01		
147	2025-03-03	20.19		
148	2025-03-03	27.33		

|< < > >|

iot_data

+



Task 2

Algorithm:

1. Install Java version 17 or 21. (For our case we will proceed with Java version 17)

```
C:\Users\kanak>java -version
java version "17.0.12" 2024-07-16 LTS
Java(TM) SE Runtime Environment (build 17.0.12+8-LTS-286)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.12+8-LTS-286, mixed mode, sharing)
```

2. Install Jenkins.

3. Start jenkins using CMD (as administrator)

```
C:\Windows\System32>C:\Users\kanak>net start jenkins
```

```
C:\Windows\System32>net start jenkins
The requested service has already been started.
```

4. Start your localhost (port 8080 which was added during installation).

<http://localhost:8080>

5. Install necessary plugins and setup admin.

Login to jenkins. Go to dashboard

The screenshot shows the Jenkins dashboard at localhost:8080. The top navigation bar includes links for 'Dashboard', 'Build History', 'Manage Jenkins', and 'My Views'. On the left, there are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (0/2). The main content area features a 'Welcome to Jenkins!' message, a 'Start building your software project' button, and a 'Set up a distributed build' section with options for 'Set up an agent', 'Configure a cloud', and 'Learn more about distributed builds'. A status bar at the bottom indicates the URL as localhost:8080/view/all/newjob and the Jenkins version as 2.492.1.

7. Go to manage jenkins from the left panel.

The screenshot shows the 'Manage Jenkins' page at localhost:8080/manage. The left sidebar has a 'Manage Jenkins' link selected. The main content area is titled 'Manage Jenkins' and includes a note about security. It features several configuration sections: 'System Configuration' (with 'Tools', 'Nodes', 'Clouds', and 'Plugins' sub-sections), 'Security' (with 'Security' and 'Credentials' sub-sections), and 'Credential Providers'. A search bar at the top right is labeled 'Search settings'.

8. Go to plugins

Dashboard > Manage Jenkins > Plugins

Plugins

Updates

Available plugins

Installed plugins

Advanced settings

Download progress

No updates available

Disabled rows are already upgraded, awaiting restart. Shaded but selectable rows are in progress or failed.

REST API Jenkins 2.492.1

9. Install and enable Git Plugin and Pipeline Plugin.

Git plugin 5.7.0

This plugin integrates Git with Jenkins.

Report an issue with this plugin

Enabled

Pipeline 600.vb_57cdd26fdd7

A suite of plugins that lets you orchestrate automation, simple or complex. See [Pipeline as Code with Jenkins](#) for more details.

Report an issue with this plugin

10. Create a new jenkins job, enter desired name and select pipeline.

New Item

Enter an item name

Arduino-Cl-CD

Select an item type

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder

OK

11. Create git repository for arduino.

Kanak0202 / Arduino-Cl-CD

Type ⌘ to search

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Arduino-Cl-CD Public

Set up GitHub Copilot

Add collaborators to this repository

Quick setup — if you've done this kind of thing before

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

```
echo "# Arduino-Cl-CD" >> README.md  
git init  
git add README.md  
git commit -m "first commit"  
git branch -M main  
git remote add origin https://github.com/Kanak0202/Arduino-Cl-CD.git  
git push -u origin main
```

12. Setup project on local.

```
● PS D:\KANAK\SEM-VIII\IOT\Arduino-CI-CD> git init
    Initialized empty Git repository in D:/KANAK/SEM-VIII/IOT/Arduino-CI-CD/.git/
● PS D:\KANAK\SEM-VIII\IOT\Arduino-CI-CD> git remote add origin https://github.com/Kanak0202/Arduino-CI-CD
○ PS D:\KANAK\SEM-VIII\IOT\Arduino-CI-CD> [ ]
```

```
Administrator: C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.22631.4974]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>mkdir C:\arduino-cli-data
A subdirectory or file C:\arduino-cli-data already exists.

C:\Windows\System32>arduino-cli config init
Config file written to: C:\Users\kanak\AppData\Local\Arduino15\arduino-cli.yaml

C:\Windows\System32>arduino-cli config set directories.data C:\arduino-cli-data

C:\Windows\System32>arduino-cli config dump
board_manager:
  additional_urls: []
directories:
  data: C:\arduino-cli-data

C:\Windows\System32>[ ]
```

Push the code to github onto your repository.

The screenshot shows a GitHub repository page for 'Arduino-CI-CD'. The repository has 1 branch and 0 tags. It contains several files: 'Jenkinsfile update' (by Kanak0202, 16 commits, 47 minutes ago), 'Arduino-CI-CD.ino' (sketch file, 1 hour ago), 'Jenkinsfile' (Jenkinsfile update, 47 minutes ago), and 'sketch.yaml' (sketch file, 1 hour ago). On the right side, there are sections for 'About' (no description, website, or topics provided), 'Activity' (16 commits), 'Star 0', 'Fork 0', 'Watching 1', and 'Forks 0'. Below that is a 'Releases' section (no releases published) and a 'Packages' section (no packages published). The 'Languages' section shows C++ at 100.0%.

Run your build.

The screenshot shows the Jenkins console output for a build. The left sidebar includes links for Status, Changes, Console Output (which is selected), Edit Build Information, Timings, Git Build Data, Pipeline Overview, Pipeline Console, Thread Dump, Pause/resume, Replay, Pipeline Steps, Workspaces, and Previous builds. The main area shows the console output:

```

Started by user Kanak Khandelwal
Obtained Jenkinsfile from git https://github.com/Kanak0202/Arduino-CI-CD.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\Arduino-CI-CD-V1
[Pipeline] {
[Pipeline] stage
[Pipeline] ( Declarative: Checkout SCM )
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\Arduino-CI-CD-V1\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/Kanak0202/Arduino-CI-CD.git # timeout=10
Fetching upstream changes from https://github.com/Kanak0202/Arduino-CI-CD.git
> git.exe --version # timeout=10
> git --version # 'git version 2.38.1.windows.1'
> git.exe fetch -tags --force --progress -- https://github.com/Kanak0202/Arduino-CI-CD.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision 163a5bd3bc79bef4f12a433027eff7e097f03f1c (refs/remotes/origin/main)

```

```

Dashboard > Arduino-CI-CD-V1 > #21

No credentials specified
> git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\workspace\Arduino-CI-CD-V1\.git # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/Kanak0202/Arduino-CI-CD.git # timeout=10
Fetching upstream changes from https://github.com/Kanak0202/Arduino-CI-CD.git
> git.exe --version # timeout=10
> git --version # 'git' version 2.38.1.windows.1'
> git.exe fetch -tags --force --progress -- https://github.com/Kanak0202/Arduino-CI-CD.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision 163a5bd3bc79bef4f12a433027eff7e097f03f1c (refs/remotes/origin/main)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f 163a5bd3bc79bef4f12a433027eff7e097f03f1c # timeout=10
> git.exe branch -a -v --no-abbrev # timeout=10
> git.exe branch -D main # timeout=10
> git.exe checkout -b main 163a5bd3bc79bef4f12a433027eff7e097f03f1c # timeout=10
Commit message: "jenkinsfile update"
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Setup Environment)
[Pipeline] bat

C:\ProgramData\Jenkins\workspace\Arduino-CI-CD-V1>set ARDUINO_DATA_PATH=C:\arduino-cli-data
[Pipeline] bat

C:\ProgramData\Jenkins\workspace\Arduino-CI-CD-V1>"C:\Users\kanak\AppData\Local\Arduino15\arduino-cli.exe" core update-index
"arduino-cli update-index" is not recognized as an internal or external command

```



```

Dashboard > Arduino-CI-CD-V1 > #21

operable program or batch file.
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Compile Sketch)
Stage "Compile Sketch" completed
[Pipeline] getContext
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Successful)
Stage "Build Successful" completed
[Pipeline] getContext
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```

REST API Jenkins 2.492.1

Code

Arduino-CI-CD.ino

```
// Simple Blink Sketch

#define LED_PIN 13
void setup() {
    pinMode(LED_PIN, OUTPUT);
}
```

```

void loop() {
    digitalWrite(LED_PIN, HIGH); // Turn LED on
    delay(1000);
    digitalWrite(LED_PIN, LOW); // Turn LED off
    delay(1000);
}

```

Jenkinsfile

```

pipeline {
    agent any

    environment {
        ARDUINO_CLI =
"C:\\\\Users\\\\kanak\\\\AppData\\\\Local\\\\Arduino15\\\\arduino-cli.exe"
        ARDUINO_DATA_PATH = "C:\\\\arduino-cli-data"
    }

    stages {
        stage('Checkout Code') {
            steps {
                git branch: 'main', url:
'https://github.com/Kanak0202/Arduino-CI-CD.git'
            }
        }

        stage('Setup Environment') {
            steps {
                bat 'set ARDUINO_DATA_PATH=C:\\\\arduino-cli-data'
                bat '"%ARDUINO_CLI%" core update-index'
                bat '"%ARDUINO_CLI%" core install arduino:avr'
            }
        }

        stage('Compile Sketch') {
            steps {
                bat '"%ARDUINO_CLI%" compile --fqbn arduino:avr:uno .'
            }
        }
    }
}

```

```
}

stage('Build Successful') {
    steps {
        echo 'Build completed successfully!'
    }
}
}
```

