CSS533: Program 4

Shreevatsa Ganapathy Hegde

University of Washington, Bothell

sghegde@uw.edu

Prof. Munehiro Fukuda

29th May, 2025

Table of Contents

Documentation:

This assignment implements two versions of an inverted indexing system using Hazelcast's distributed map. The local version retrieves files from the shared map and counts word occurrences on a single node. The remote version dispatches tasks to cluster nodes to perform local file indexing, demonstrating Hazelcast's support for distributed execution and performance measurement.

To implement the above program, the following classes were modified. Each class is explained in detail below.

InvertedIndexingLocal: This class implements the local version of the inverted indexing program using Hazelcast's distributed map. It connects to the cluster, retrieves all <filename, content> entries from the "files" map, and processes them sequentially on the local node. For each file, it counts how many times a given keyword appears in the text content and prints the result. The entire workload is executed on a single node without distribution, making this version useful as a baseline to measure and compare performance with the distributed approach. It uses simple string matching and standard iteration, and also records the execution time to highlight efficiency. This local version plays a key role in verifying indexing correctness before introducing distributed complexity, serving as a reference implementation for correctness and performance benchmarking.

InvertedIndexingEach: This class defines the task that is executed remotely on each cluster node in the distributed version of the indexing program. It implements Callable<Map<String, Integer>>, enabling Hazelcast to serialize and dispatch it across cluster members using its executor service. On execution, it accesses only the entries of the "files" map that are local to the executing node, counts the occurrences of the given search word in each file, and returns a map of <filename, count> results. The class is essential for enabling data-local computation, which reduces network overhead and improves scalability. It leverages Hazelcast's partition-aware execution to ensure each node processes only its own subset of data. This modular unit of computation supports parallel execution, and highlights the core mechanism by which Hazelcast enables distributed, scalable processing with low communication cost.

InvertedIndexingRemote: This class acts as the coordinator for executing the distributed indexing task across the Hazelcast cluster. It connects to the cluster, constructs instances of InvertedIndexingEach, and submits them to all remote nodes using Hazelcast's executor service. Once the remote tasks are dispatched, it waits for their completion, collects the partial results, merges them into a single combined result, and prints the final word counts per file. It measures and displays execution time, allowing for performance comparison with the local version. This class showcases Hazelcast's remote task submission and result aggregation features, orchestrating the distribution and collection of work efficiently. It plays a critical role in harnessing parallelism across the cluster and illustrates how distributed computation can improve performance by offloading work to multiple nodes while keeping the final aggregation centralized.

These are the classes that were needed as part of base functionality. Additional Features will be discussed in discussion section. To Conclude, the remote execution distributes the indexing task across cluster members, resulting in faster and more balanced processing, as seen from its quicker completion time (53 ms). In contrast, the local execution processes all files on a single node, taking longer overall (122 ms) despite producing similar output. Both of these outputs are attached in the Execution section.

Source Code:

InvertedIndexingLocal.java

```
import com.hazelcast.core.Hazelcast;
import com.hazelcast.core.HazelcastInstance;
import com.hazelcast.map.IMap;
import com.hazelcast.config.IndexConfig;
import com.hazelcast.config.IndexType;
import com.hazelcast.query.Predicates;
import com.hazelcast.query.Predicate;
import java.util.Collection;
import java.util.Set;
import java.util.Iterator;
import java.util.Map;
import java.util.HashMap;
import java.util.Hashtable;
import java.util.Map.Entry;


import com.hazelcast.core.HazelcastInstanceAware;
import java.util.concurrent.*;
import java.util.Date;
import java.io.Serializable;

public class InvertedIndexingLocal {
    public static void main( String[] args ) {
        // validate arguments
        if ( args.length != 1 ) {
            System.out.println( "usage: java InvertedIndexingLocal keyword " );
            return;
        }
        String keyword = args[0];

        // prepare a local map
        Hashtable<String, Integer> local = new Hashtable<String, Integer>( );

        // start hazelcast and retrieve a cached map
        HazelcastInstance hz = Hazelcast.newHazelcastInstance( );

        // start a timer
        Date startTimer = new Date( );

        IMap<String, String> map = hz.getMap( "files" );
        // examine each file
        Iterator<Entry<String, String>> iterator = map.entrySet().iterator();
```

```
        // Count occurrences of the keyword in each file
        while (iterator.hasNext()) {
            Entry<String, String> entry = iterator.next();
            String filename = entry.getKey();
            String contents = entry.getValue();

            int count = countOccurrences(contents, keyword);
            if (count > 0) {
                local.put(filename, count);
            }
        }
        Date endTimer = new Date( ); // before showing the result, stop the timer for the
perrformance measurement.

        // show the result
        for (Map.Entry<String, Integer> entry : local.entrySet()) {
            System.out.println("File[" + entry.getKey() + "] has " + entry.getValue());
        }

        System.out.println("Execution Time: " + (endTimer.getTime() - startTimer.getTime()) +
" ms");

        hz.shutdown();
    }
    // Count occurrences of a word in a given text
    private static int countOccurrences(String text, String word) {
        int count = 0;
        int index = 0;

        while ((index = text.indexOf(word, index)) != -1) {
            count++;
            index += word.length();
        }

        return count;
    }
}
```

InvertedIndexingEach.java

```
import com.hazelcast.core.Hazelcast;
import com.hazelcast.core.HazelcastInstance;
import com.hazelcast.map.IMap;
import com.hazelcast.config.IndexConfig;
import com.hazelcast.config.IndexType;
import com.hazelcast.query.Predicates;
import com.hazelcast.query.Predicate;
import java.util.Collection;
import java.util.Set;
import java.util.Iterator;
import java.util.Map;
import java.util.HashMap;
import java.util.Enumeration;

import com.hazelcast.core.HazelcastInstanceAware;
import java.util.concurrent.*;
import java.util.Date;
import java.io.Serializable;

public class InvertedIndexingEach
    implements Callable<String>, HazelcastInstanceAware, Serializable {
        // This class implements Callable to perform inverted indexing on each member of the
Hazelcast cluster
        String keyword;
        private transient HazelcastInstance hz;

        // default constructor
        public InvertedIndexingEach( ) {

        }

        // constructor used to receive a keyword for inverted indexing
        public InvertedIndexingEach( String keyword ) {
            this.keyword = keyword;
        }


        @Override
        public void setHazelcastInstance( HazelcastInstance hz ) {
            this.hz = hz;
        }

    @Override
    public String call( ) throws Exception {
```

```
        System.out.println( "started" );

        Map<String, Integer> local = new HashMap<>();
        // search for a collection of files
        IMap<String, String> map = hz.getMap( "files" );


        Set<String> localKeys = map.localKeySet();
        //Count occurrences of the keyword in each file
        for (String filename : localKeys) {
            String content = map.get(filename);
            int count = countOccurrences(content, keyword);
            if (count > 0) {
                local.put(filename, count);
            }
        }

        // Prepare the result as a string
        StringBuilder result = new StringBuilder();
        for (Map.Entry<String, Integer> entry : local.entrySet()) {
            result.append(entry.getKey()).append(" ").append(entry.getValue()).append(" ");
        }

        return result.toString().trim();

    }
    // Method to count occurrences of a word in a text
    private int countOccurrences(String text, String word) {
        int count = 0;
        int index = 0;
        while ((index = text.indexOf(word, index)) != -1) {
            count++;
            index += word.length();
        }
        return count;
    }
}
```

InvertedIndexingRemote.java

```
import com.hazelcast.core.*;
import com.hazelcast.map.IMap;
import com.hazelcast.executor.*;
import java.util.concurrent.*;
import java.util.*;
import com.hazelcast.cluster.Member;


public class InvertedIndexingRemote {
    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Usage: java InvertedIndexingRemote <keyword>");
            return;
        }

        // Get the keyword from command line arguments
        String keyword = args[0];
        // Create a local map to store results
        Map<String, Integer> local = new HashMap<>();

        System.out.println("---------Joining Hazelcast cluster----------");
        // Create a Hazelcast instance to join the cluster
        HazelcastInstance hz = Hazelcast.newHazelcastInstance();

        Date start = new Date();
        System.out.println("Start time recorded: " + start);

        System.out.println("Retrieving executor service");
        // Get the executor service from Hazelcast instance
        IExecutorService exec = hz.getExecutorService("exec");
        // Create a task for inverted indexing
        InvertedIndexingEach task = new InvertedIndexingEach(keyword);
        // Set the Hazelcast instance in the task
        Set<Member> members = hz.getCluster().getMembers();
        System.out.println("Dispatching task to " + members.size() + " cluster
members");

        // Submit the task to all members of the cluster
        exec.submitToMembers(task, members, new MultiExecutionCallback() {
            // This callback will be invoked when the task is executed on each
member
            @Override
            public void onResponse(Member member, Object value) {
```

```
                System.out.println("Received response from: " +
member.getAddress());
            }

            // This callback will be invoked when all responses are received
            @Override
            public void onComplete(Map<Member, Object> values) {
                System.out.println("All responses received. Processing results");
                Date end = new Date();

                for (Object resultObj : values.values()) {
                    String result = (String) resultObj;
                    String[] tokens = result.split(" ");
                    for (int i = 0; i < tokens.length - 1; i += 2) {
                        String filename = tokens[i];
                        int count = Integer.parseInt(tokens[i + 1]);
                        local.put(filename, count);
                    }
                }

                System.out.println("Inverted Indexing Results:");
                for (Map.Entry<String, Integer> entry : local.entrySet()) {
                    System.out.println("File[" + entry.getKey() + "] has " +
entry.getValue());
                }

                System.out.println("Remote Execution Time: " + (end.getTime() -
start.getTime()) + " ms");

                System.out.println("----------Shutting down Hazelcast instance---
-------");
                hz.shutdown();
            }
        });
    }
}
```

Additional Features:

Mobile Agent Development:

Place.java:

```
package Mobile;

import java.io.*;
import java.net.*;
import java.rmi.*;
import java.rmi.server.*;
import java.util.Queue;
import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.ConcurrentLinkedQueue;
import java.rmi.registry.*;
import com.hazelcast.core.*;
import com.hazelcast.map.IMap;
import com.hazelcast.config.Config;
import com.hazelcast.config.XmlConfigBuilder;

/**
 * Mobile.Place is the our mobile-agent execution platform that accepts an
 * agent transferred by Mobile.Agent.hop( ), deserializes it, and resumes it
 * as an independent thread.
 *
 * @author  Munehiro Fukuda
 * @version %I% %G$
 * @since   1.0
 */
public class Place extends UnicastRemoteObject implements PlaceInterface {
    private AgentLoader loader = null;  // a loader to define a new agent class
    private int agentSequencer = 0;     // a sequencer to give a unique agentId
    // Indirect communication
    private ConcurrentHashMap<Integer, Queue<String>> messageBox = new ConcurrentHashMap<>();
    // Agent directory to keep track of agent IDs
    private ConcurrentHashMap<String, Integer> agentDirectory = new ConcurrentHashMap<>();
    // HazelcastInstance declaration in Place
    HazelcastInstance hz = null;
    // HazelcastInstance initialization when Place is initialized
    {
        try {
            InputStream configInput = new
FileInputStream(System.getProperty("hazelcast.config"));
            Config config = new XmlConfigBuilder(configInput).build();
            hz = Hazelcast.newHazelcastInstance(config);
```

```
        } catch (FileNotFoundException e) {
            e.printStackTrace();
    }
    }

    /**
     * This constructor instantiates a Mobiel.AgentLoader object that
     * is used to define a new agen class coming from remotely.
     */
    public Place( ) throws RemoteException {
    super( );
    loader = new AgentLoader( );
    }

    /**
     * deserialize( ) deserializes a given byte array into a new agent.
     *
     * @param buf a byte array to be deserialized into a new Agent object.
     * @return a deserialized Agent object
     */
    private Agent deserialize( byte[] buf )
    throws IOException, ClassNotFoundException {
    // converts buf into an input stream
        ByteArrayInputStream in = new ByteArrayInputStream( buf );

    // AgentInputStream identify a new agent class and deserialize
    // a ByteArrayInputStream into a new object
        AgentInputStream input = new AgentInputStream( in, loader );
        return ( Agent )input.readObject();
    }

    /**
     * transfer( ) accepts an incoming agent and launches it as an independent
     * thread.
     *
     * @param classname The class name of an agent to be transferred.
     * @param bytecode  The byte code of  an agent to be transferred.
     * @param entity    The serialized object of an agent to be transferred.
     * @return true if an agent was accepted in success, otherwise false.
     */
    public boolean transfer( String classname, byte[] bytecode, byte[] entity )
    throws RemoteException {
        try {
            //  Load agent class
            loader.loadClass(classname, bytecode);
```

```
            System.out.println("Receiving agent " + classname + "...");


            // Deserialize agent
            Agent agent = deserialize(entity);
            System.out.println("Deserialized agent with ID: " + agent.getId());

            //  Assign unique ID if not yet set
            if (agent.getId() == -1) {
                int id = InetAddress.getLocalHost().hashCode() + agentSequencer++;
                agent.setId(id);
            }
            agent.setHazelcastInstance(hz); // inject Hazelcast instance


            //  Start agent thread
            Thread thread = new Thread(agent);
            thread.start();
            System.out.println("Started agent thread for: " + classname);

            return true;
        } catch (Exception e) {
            e.printStackTrace();
            return false;
        }

    }

    /**
     * main( ) starts an RMI registry in local, instantiates a Mobile.Place
     * agent execution platform, and registers it into the registry.
     *
     * @param args receives a port, (i.e., 5001-65535).
     */
    public static void main( String args[] ) {
        if (args.length != 1) {
        System.err.println("Usage: java Mobile.Place <port>");
        System.exit(1);
        }

        int port = 0;
        try {
            port = Integer.parseInt(args[0]);
            if (port < 5001 || port > 65535) {
                throw new IllegalArgumentException("Port must be between 5001 and 65535");
```

```
            }
        } catch (Exception e) {
            e.printStackTrace();
            System.exit(1);
        }

        try {
            startRegistry(port);
            Place place = new Place();
            Naming.rebind("rmi://localhost:" + port + "/place", place);
            System.out.println("Place is running at port " + port);
        } catch (Exception e) {
            e.printStackTrace();
        }

    }

    /**
     * startRegistry( ) starts an RMI registry process in local to this Place.
     *
     * @param port the port to which this RMI should listen.
     */
    private static void startRegistry( int port ) throws RemoteException {
        try {
            Registry registry =
                LocateRegistry.getRegistry( port );
            registry.list( );
        }
        catch ( RemoteException e ) {
            Registry registry =
                LocateRegistry.createRegistry( port );
        }
    }

    /**
     * deliverMessage( ) delivers a message to the specified agent.
     *
     * @param receiverId The ID of the agent to receive the message.
     * @param message The message to be delivered.
     */
    @Override
    public void deliverMessage(int receiverId, String message) throws RemoteException {
        messageBox.computeIfAbsent(receiverId, k -> new
ConcurrentLinkedQueue<>()).add(message);
        System.out.println("Delivered message to agent " + receiverId + ": " + message);
```

```
    }

    /**
     * retrieveMessage( ) retrieves a message for the specified agent.
     *
     * @param receiverId The ID of the agent to retrieve the message for.
     * @return The message for the agent, or null if no message is available.
     */

    @Override
    public String retrieveMessage(int receiverId) throws RemoteException {
        Queue<String> queue = messageBox.get(receiverId);
        if (queue == null || queue.isEmpty()) {
            return null;
        }
        return queue.poll();
    }
    /**
     * registerAgent( ) registers an agent with a name and ID.
     *
     * @param name The name of the agent.
     * @param agentId The ID of the agent.
     */
    @Override
    public void registerAgent(String name, int agentId) {
        agentDirectory.put(name, agentId);
    }

    /**
     * lookupAgentId( ) looks up the ID of an agent by its name.
     * @param name
     * @return The ID of the agent, or -1 if not found.
     */

    @Override
    public int lookupAgentId(String name) {
        return agentDirectory.getOrDefault(name, -1);
    }
}
```

InvertedIndexingWithAgent.java

```
import java.util.*;
import Mobile.*;
import java.io.*;
```

```
public class InvertedIndexingWithAgent {
    public static void main(String[] args) {
        if (args.length < 2) {
            System.out.println("Usage: java InvertedIndexingWithAgent <keyword> <host1>
<host2> ...");
            return;
        }

        String keyword = args[0];
        List<String> hopList = new ArrayList<>();

        for (int i = 1; i < args.length; i++) {
            hopList.add(args[i]);
        }

        int port = 5099; // fixing since this is not the primary focus of the task
        int agentId = new Random().nextInt(10000);
        // Creating indexing agent
        IndexingAgent agent = new IndexingAgent(keyword, hopList, port);
        agent.setId(agentId);

        System.out.println("Launching IndexingAgent with ID: " + agentId);
        agent.init();
    }
}
```

InvertedIndexAgentDeployer.java

```
import Mobile.*;
import java.rmi.Naming;
import java.util.*;
import java.io.*;

public class InvertedIndexAgentDeployer {
    public static void main(String[] args) {
        if (args.length < 2) {
            System.out.println("Usage: java InvertedIndexAgentDeployer <host> <keyword>");
            return;
        }

        String host = args[0];
```

```
        String keyword = args[1];
        int port = 5099;

        try {
            List<String> dummyList = new ArrayList<>();
            //Sending the umm dummy list to the agent becasue the focus is
            //on developing mobile agents rather than the hop.
            //hopping will be impleted in the next version.
            IndexingAgent agent = new IndexingAgent(keyword, dummyList, port);
            agent.setId(new Random().nextInt(10000));
            agent.setPort(port);
            agent.setHost(host);
            agent.setFunction("init"); // Set the function to invoke upon deployment

            // RMI lookup
            String url = "rmi://" + host + ":" + port + "/place";
            PlaceInterface remote = (PlaceInterface) Naming.lookup(url);
            boolean success = remote.transfer(agent.getClass().getName(), agent.getByteCode(),
serialize(agent));

            if (success) {
                System.out.println("Agent successfully deployed to " + host);
            } else {
                System.err.println("Agent deployment failed on " + host);
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static byte[] serialize(Agent agent) throws Exception {
        ByteArrayOutputStream out = new ByteArrayOutputStream();
        ObjectOutputStream os = new ObjectOutputStream(out);
        os.writeObject(agent);
        return out.toByteArray();
    }
}
```

IndexingAgent.java

```
package Mobile;
import com.hazelcast.core.*;
import com.hazelcast.map.IMap;
```

```java
import java.util.*;
import java.rmi.*;
import java.io.*;

public class IndexingAgent extends Agent {
    private String keyword;
    private List<String> hopList; // hostnames to hop
    private int currentHop;
    private Map<String, Integer> localResults;

    public IndexingAgent(String keyword, List<String> hopList, int port) {
        this.keyword = keyword;
        this.hopList = hopList;
        this.currentHop = 0;
        this.localResults = new HashMap<>();
        this.setPort(port);
    }

    // First method to invoke
    public void init() {
        System.out.println("IndexingAgent[" + getId() + "] starting on host: " + getHost());
        performIndexing(); // Call for  Moble Agent additional feature

    }

    // Hazelcast local indexing
    private void performIndexing() {
        HazelcastInstance hz = getHazelcastInstance();

        Date startTimer = new Date( );

        IMap<String, String> map = hz.getMap("files");

        Set<String> localKeys = map.localKeySet();

        for (String filename : localKeys) {
            String content = map.get(filename);
            int count = countOccurrences(content, keyword);
            if (count > 0) {
                localResults.put(filename, localResults.getOrDefault(filename, 0) + count);
            }
        }

        Date endTimer = new Date( );
```

```
        System.out.println("Indexing done at: " + getHost());
        System.out.println("=== Final Inverted Indexing Results ===");
        for (Map.Entry<String, Integer> entry : localResults.entrySet()) {
            System.out.println("File[" + entry.getKey() + "] has " + entry.getValue());
        }
        System.out.println("Execution Time in " + getHost() +" : " + (endTimer.getTime() -
startTimer.getTime()) + " ms");
        hz.shutdown();
    }

    private int countOccurrences(String text, String word) {
        int count = 0, idx = 0;
        while ((idx = text.indexOf(word, idx)) != -1) {
            count++;
            idx += word.length();
        }
        return count;
    }

}
```

Hierarchy Implementation:

This implementation uses mobile agents to develop a heirarcchy

HierarchicalIndexingAgent.java

```
package Mobile;

import com.hazelcast.core.*;
import com.hazelcast.map.IMap;

import java.io.Serializable;
import java.util.*;

public class HierarchicalIndexingAgent extends Agent implements Serializable {
    private String keyword;
    private List<String> childHosts;
    private int port;
    // Name of the shared map in Hazelcast where child results will be stored
    private String sharedMapName = "child-results";
```

```
    // Constructor to initialize the agent with keyword, child hosts, and port
    public HierarchicalIndexingAgent(String keyword, List<String> childHosts, int port) {
        this.keyword = keyword;
        this.childHosts = childHosts;
        this.port = port;
        this.setFunction("init");
    }

    public void init() {
        System.out.println("Hierarchical agent starting at: " + getHost());

        long start = System.currentTimeMillis();

        // Dispatch children
        for (String host : childHosts) {
            ChildIndexingAgent child = new ChildIndexingAgent(keyword,port);
            child.setId(new Random().nextInt(10000));
            child.setPort(port);
            child.setHost(host);
            child.setFunction("init");

            try {
                String url = "rmi://" + host + ":" + port + "/place";
                PlaceInterface remote = (PlaceInterface) java.rmi.Naming.lookup(url);
                boolean success = remote.transfer(child.getClass().getName(),
child.getByteCode(), serialize(child));
                if (success) {
                    System.out.println("Dispatched child to " + host);
                } else {
                    System.err.println("Failed to dispatch to " + host);
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        }

        // Wait for children to index
        try {
            Thread.sleep(5000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

        // Aggregate results from Hazelcast shared map
        HazelcastInstance hz = Hazelcast.newHazelcastInstance();
```

```
        collectResults(hz);
        hz.shutdown();
    }

    private byte[] serialize(Agent agent) throws Exception {
        java.io.ByteArrayOutputStream out = new java.io.ByteArrayOutputStream();
        java.io.ObjectOutputStream os = new java.io.ObjectOutputStream(out);
        os.writeObject(agent);
        return out.toByteArray();
    }

    // Collect results from all child agents stored in Hazelcast
    private void collectResults(HazelcastInstance hz) {
        IMap<String, Map<String, Integer>> resultMap = hz.getMap("results");
        System.out.println("=== Aggregated Results from All Children in Parent: "+ getHost() +
" ===");

        for (Map.Entry<String, Map<String, Integer>> entry : resultMap.entrySet()) {
            String childId = entry.getKey();
            Map<String, Integer> childResults = entry.getValue();



            if (childResults != null && !childResults.isEmpty()) {
                System.out.println("Results from: " + childId );
                for (Map.Entry<String, Integer> fileEntry : childResults.entrySet()) {
                    System.out.println("  File[" + fileEntry.getKey() + "] has " +
fileEntry.getValue());
                }
            } else {
                System.out.println("No files indexed or no matches found in "+ childId);
            }
        }
    }


}
```

ChildIndexingAgent.Java

```
package Mobile;

import com.hazelcast.core.*;
import com.hazelcast.map.IMap;
```

```java
import java.io.InputStream;
import java.io.FileInputStream;


import java.util.*;

public class ChildIndexingAgent extends Agent {
    private String keyword;
    // Map to store local results with filename as key and count of keyword occurrences as
value
    private Map<String, Integer> localResults;

    public ChildIndexingAgent(String keyword, int port) {
        this.keyword = keyword;
        this.localResults = new HashMap<>();
        this.setPort(port);
    }

    // Initializes the agent and starts the indexing process
    public void init() {
        System.out.println("ChildIndexingAgent[" + getId() + "] started on host: " +
getHost());
        performIndexing();
    }
    // Performs the indexing operation
    private void performIndexing() {
        HazelcastInstance hz;
        try {
            // Load Hazelcast configuration from the specified path or use default
configuration
            String configPath = System.getProperty("hazelcast.config");
            if (configPath != null) {
                InputStream configInput = new java.io.FileInputStream(configPath);
                com.hazelcast.config.Config config = new
com.hazelcast.config.XmlConfigBuilder(configInput).build();
                // Create a Hazelcast instance with the loaded configuration
                hz = Hazelcast.newHazelcastInstance(config);
            } else {
                hz = Hazelcast.newHazelcastInstance();
            }
        } catch (Exception e) {
            e.printStackTrace();
            return;
        }
        // Fetch the map containing files and their contents
```

```
        IMap<String, String> map = hz.getMap("files");
        Set<String> localKeys = map.localKeySet();
        long start = System.currentTimeMillis();
        for (String filename : localKeys) {
            String content = map.get(filename);
            int count = countOccurrences(content, keyword);
            if (count > 0) {
                localResults.put(filename, localResults.getOrDefault(filename, 0) + count);
            }
        }

        System.out.println("Indexing done at: " + getHost());
        for (Map.Entry<String, Integer> entry : localResults.entrySet()) {
            System.out.println("File[" + entry.getKey() + "] has " + entry.getValue());
        }

        long end = System.currentTimeMillis();
        System.out.println("Execution Time in " + getHost() + " : " + (end - start) + " ms");
        IMap<String, Map<String, Integer>> resultMap = hz.getMap("results");
        resultMap.put(getHost() + "-" + getId(), localResults);
        try{
            Thread.sleep(500);
        }catch (InterruptedException e) {
            e.printStackTrace();
        }

        hz.shutdown();
    }

    // Counts occurrences of the keyword in the given text
    private int countOccurrences(String text, String word) {
        int count = 0, idx = 0;
        while ((idx = text.indexOf(word, idx)) != -1) {
            count++;
            idx += word.length();
        }
        return count;
    }

}
```

Servlet Implementation:

InvertedIndexServlet.java

```java
import com.hazelcast.core.Hazelcast;
import com.hazelcast.core.HazelcastInstance;
import com.hazelcast.map.IMap;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import java.io.IOException;
import java.io.PrintWriter;


@WebServlet("/search")
public class InvertedIndexServlet extends HttpServlet {

    // Hazelcast instance to interact with the distributed map
    private HazelcastInstance hazelcastInstance;

    // Initializes the servlet and creates a Hazelcast instance
    @Override
    public void init() {
        hazelcastInstance = Hazelcast.newHazelcastInstance();
    }

    // Handles GET requests to search for a keyword in the indexed files
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
            throws ServletException, IOException {
        String keyword = req.getParameter("keyword");
        resp.setContentType("text/html");
        PrintWriter out = resp.getWriter();
        //fetch the map containing files and their contents
        IMap<String, String> map = hazelcastInstance.getMap("files");
        out.println("<html><body>");
        out.println("<h2>Results for keyword: " + keyword + "</h2>");

        for (String file : map.keySet()) {
            String content = map.get(file);
            long count = content.split("\\b" + keyword + "\\b", -1).length - 1;
            if (count > 0) {
                out.println("<p><strong>" + file + ":</strong> " + count + "
occurrences</p>");
            }
        }

        out.println("</body></html>");
```

```
    }

    @Override
    public void destroy() {
        hazelcastInstance.shutdown();
    }
}
```

Tomcat web.xml

```
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
         http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
         version="3.1">

    <servlet>
        <servlet-name>InvertedIndexServlet</servlet-name>
        <servlet-class>InvertedIndexServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>InvertedIndexServlet</servlet-name>
        <url-pattern>/search</url-pattern>
    </servlet-mapping>

</web-app>
```

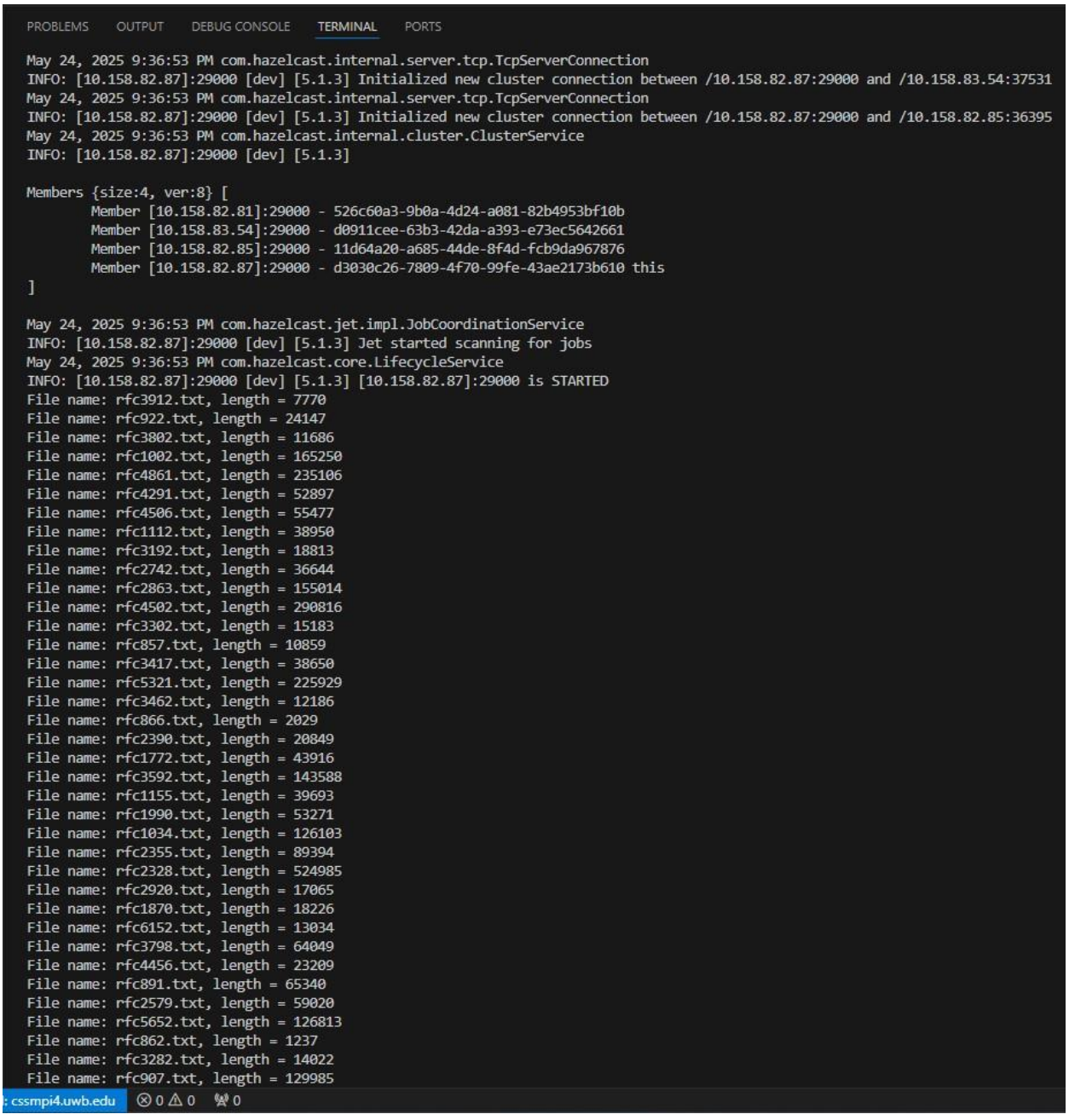
Index.html

```
<!DOCTYPE html>
<html>
<head>
    <title>Inverted Index Search</title>
</head>
<body style="margin: 0 auto;">
    <h2>Search Keyword in Files</h2>
    <form action="search" method="get">
        <label for="keyword">Enter keyword:</label>
        <input type="text" id="keyword" name="keyword" required>
        <button type="submit">Search</button>
    </form>
</body>
</html>
```

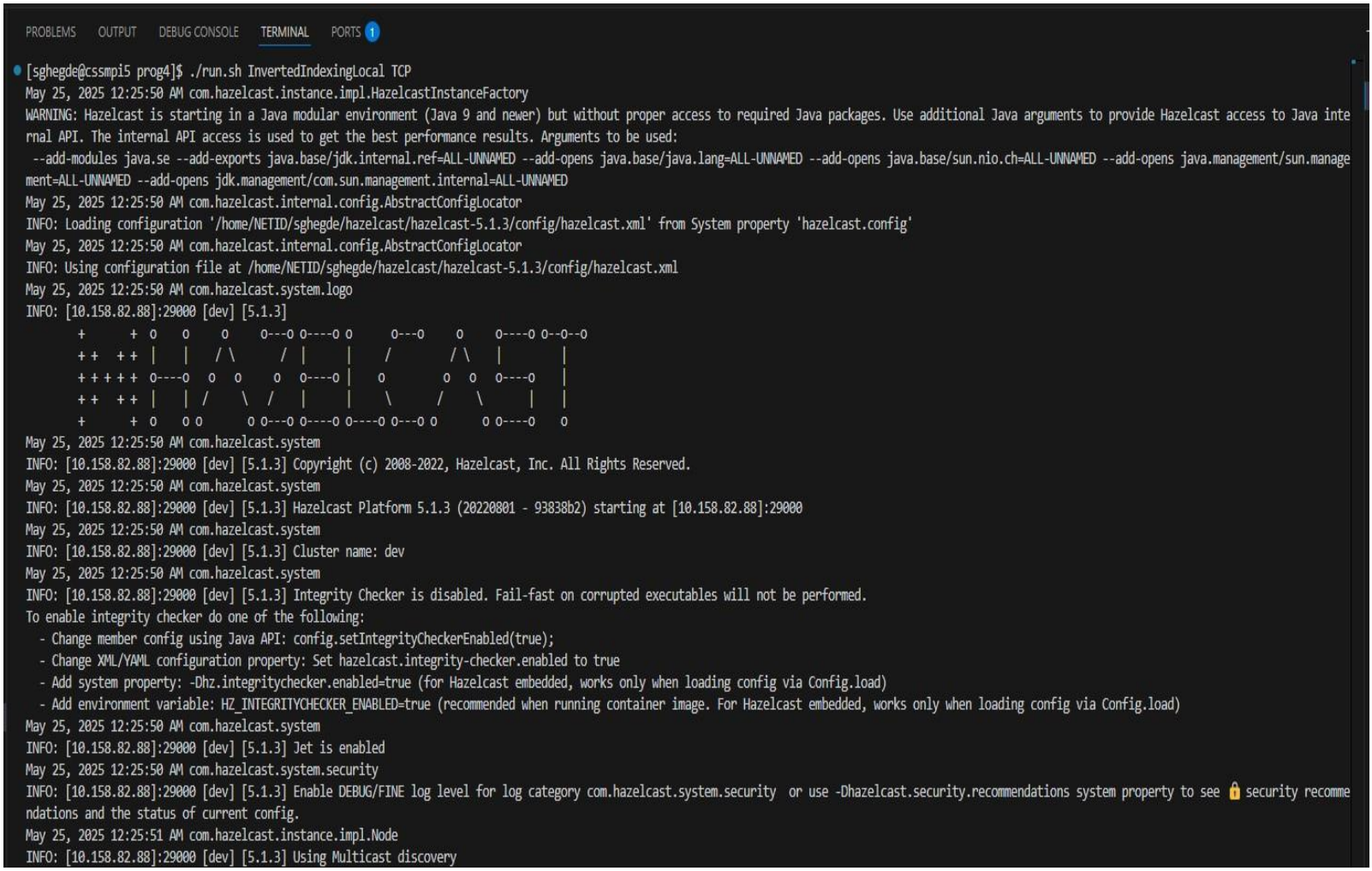
Execution Outputs:

Base Implementation:

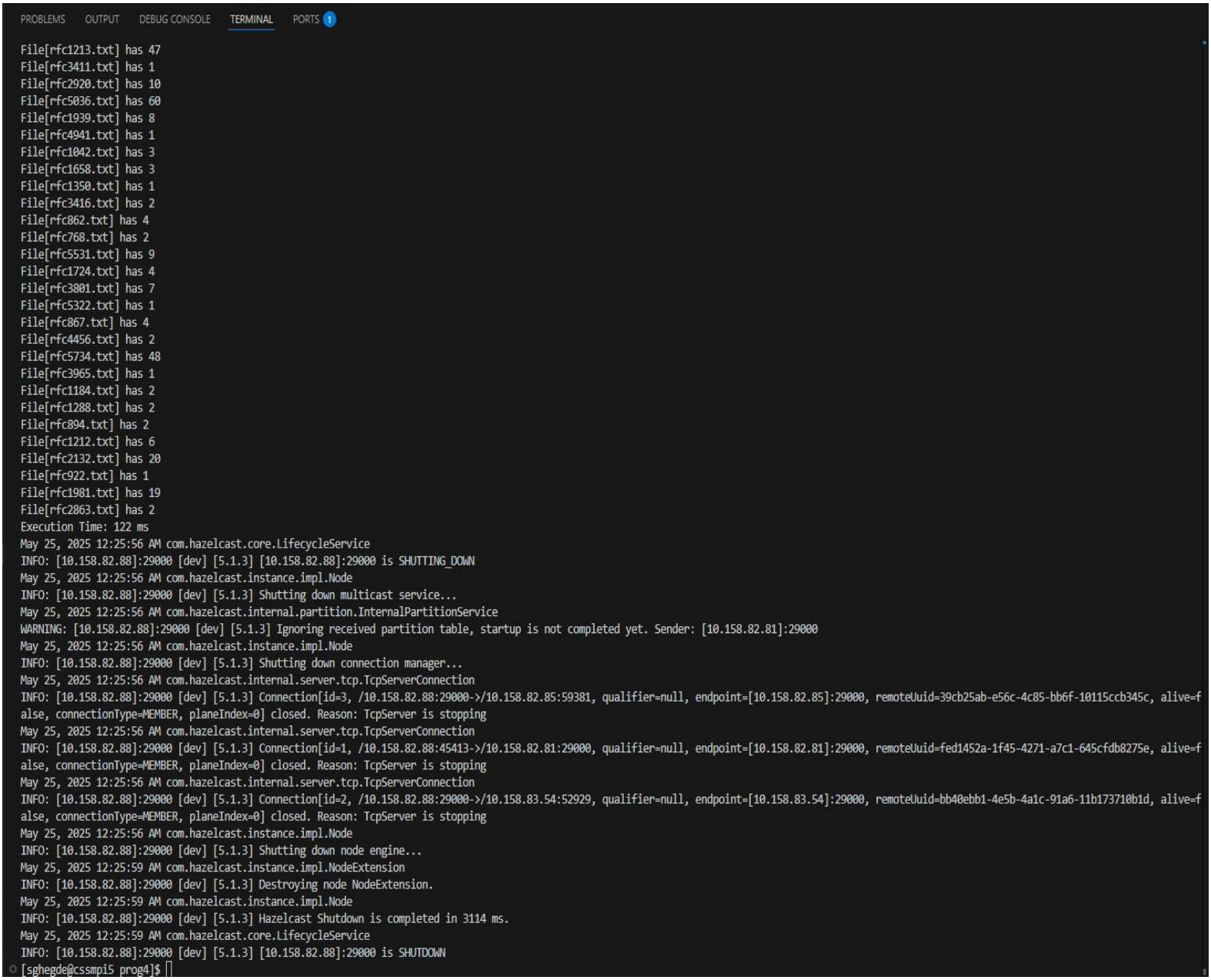
Two scenarios were used to test the base implementation. Below are the respective outputs for each. I apologize for the smaller text in the outputs—only the relevant sections have been included due to the large volume. However, the images are of high quality and can be zoomed in for clarity. The screenshot below shows the data loading process.



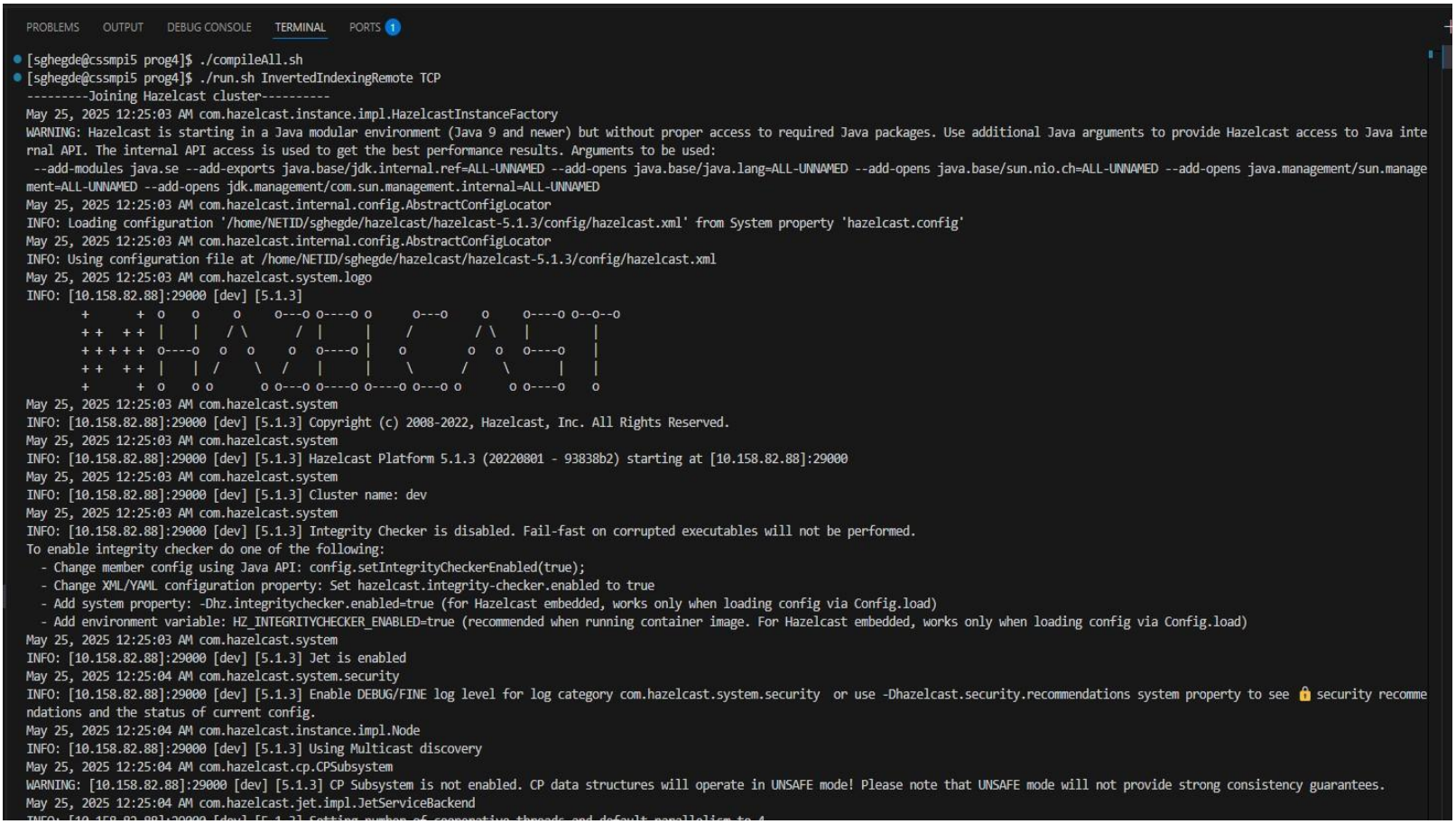
1. InvertedIndexingLocal.java’s execution
   Command screen:



Local Counting Output:



2. InvertedIndexingRemote.java’s execution:
   Command Screen:



Hazelcast server starting in all three machines (cssmpi1, cssmpi2, cssmpi3):
in next page

```
INFO: [10.158.82.81]:29000 [dev] [5.1.3]
        +      + o    o     o     o---o o----o o      o---o     o     o----o o--o--o
        + +   + +  |    |    / \       /  |      |     /        / \    |        |
        + + + + +  o----o   o   o     o   o----o |    o        o   o   o----o    |
        + +   + +  |    |  /     \   /    |      |     \      /     \       |    |
        +       +  o    o o       o o---o o----o o----o o---o o       o o----o    o
May 24, 2025 9:08:46 PM com.hazelcast.system
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Copyright (c) 2008-2022, Hazelcast, Inc. All Rights Reserved.
May 24, 2025 9:08:46 PM com.hazelcast.system
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Hazelcast Platform 5.1.3 (20220801 - 93838b2) starting at [10.158.82.81]:29000
May 24, 2025 9:08:46 PM com.hazelcast.system
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Cluster name: dev
May 24, 2025 9:08:46 PM com.hazelcast.system
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Integrity Checker is disabled. Fail-fast on corrupted executables will not be performed.
To enable integrity checker do one of the following:
  - Change member config using Java API: config.setIntegrityCheckerEnabled(true);
  - Change XML/YAML configuration property: Set hazelcast.integrity-checker.enabled to true
  - Add system property: -Dhz.integritychecker.enabled=true (for Hazelcast embedded, works only when loading config via Config.load)
  - Add environment variable: HZ_INTEGRITYCHECKER_ENABLED=true (recommended when running container image. For Hazelcast embedded, works only when loading config via Config.load)
May 24, 2025 9:08:46 PM com.hazelcast.system
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Jet is enabled
May 24, 2025 9:08:47 PM com.hazelcast.system.security
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Enable DEBUG/FINE log level for log category com.hazelcast.system.security  or use -Dhazelcast.security.recommendations system property to see 🔒 security reco
May 24, 2025 9:08:47 PM com.hazelcast.instance.impl.Node
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Using Multicast discovery
May 24, 2025 9:08:47 PM com.hazelcast.cp.CPSubsystem
WARNING: [10.158.82.81]:29000 [dev] [5.1.3] CP Subsystem is not enabled. CP data structures will operate in UNSAFE mode! Please note that UNSAFE mode will not provide strong consistency guarantees.
May 24, 2025 9:08:47 PM com.hazelcast.jet.impl.JetServiceBackend
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Setting number of cooperative threads and default parallelism to 4
May 24, 2025 9:08:47 PM com.hazelcast.internal.diagnostics.Diagnostics
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Diagnostics disabled. To enable add -Dhazelcast.diagnostics.enabled=true to the JVM arguments.
May 24, 2025 9:08:47 PM com.hazelcast.core.LifecycleService
INFO: [10.158.82.81]:29000 [dev] [5.1.3] [10.158.82.81]:29000 is STARTING
May 24, 2025 9:08:49 PM com.hazelcast.internal.cluster.ClusterService
INFO: [10.158.82.81]:29000 [dev] [5.1.3]

Members {size:1, ver:1} [
        Member [10.158.82.81]:29000 - 526c60a3-9b0a-4d24-a081-82b4953bf10b this
]

May 24, 2025 9:08:49 PM com.hazelcast.jet.impl.JobCoordinationService
INFO: [10.158.82.81]:29000 [dev] [5.1.3] Jet started scanning for jobs
May 24, 2025 9:08:49 PM com.hazelcast.core.LifecycleService
INFO: [10.158.82.81]:29000 [dev] [5.1.3] [10.158.82.81]:29000 is STARTED
May 24, 2025 9:08:49 PM com.hazelcast.internal.partition.impl.PartitionStateManager
```

H: cssmpi1.uwb.edu ⊗ 0 ⚠ 0 0

```
INFO: [10.158.82.85]:29000 [dev] [5.1.3]
        +      + o    o     o     o---o o----o o      o---o     o     o----o o--o--o
        + +   + +  |    |    / \       /  |      |     /        / \    |        |
        + + + + +  o----o   o   o     o   o----o |    o        o   o   o----o    |
        + +   + +  |    |  /     \   /    |      |     \      /     \       |    |
        +       +  o    o o       o o---o o----o o----o o---o o       o o----o    o
May 24, 2025 9:15:42 PM com.hazelcast.system
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Copyright (c) 2008-2022, Hazelcast, Inc. All Rights Reserved.
May 24, 2025 9:15:42 PM com.hazelcast.system
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Hazelcast Platform 5.1.3 (20220801 - 93838b2) starting at [10.158.82.85]:29000
May 24, 2025 9:15:42 PM com.hazelcast.system
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Cluster name: dev
May 24, 2025 9:15:42 PM com.hazelcast.system
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Integrity Checker is disabled. Fail-fast on corrupted executables will not be performed.
To enable integrity checker do one of the following:
  - Change member config using Java API: config.setIntegrityCheckerEnabled(true);
  - Change XML/YAML configuration property: Set hazelcast.integrity-checker.enabled to true
  - Add system property: -Dhz.integritychecker.enabled=true (for Hazelcast embedded, works only when loading config via Config.load)
  - Add environment variable: HZ_INTEGRITYCHECKER_ENABLED=true (recommended when running container image. For Hazelcast embedded, works only when loading config via Config.load)
May 24, 2025 9:15:42 PM com.hazelcast.system
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Jet is enabled
May 24, 2025 9:15:43 PM com.hazelcast.system.security
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Enable DEBUG/FINE log level for log category com.hazelcast.system.security  or use -Dhazelcast.security.recommendations system property to see 🔒 security recommend
May 24, 2025 9:15:43 PM com.hazelcast.instance.impl.Node
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Using Multicast discovery
May 24, 2025 9:15:43 PM com.hazelcast.cp.CPSubsystem
WARNING: [10.158.82.85]:29000 [dev] [5.1.3] CP Subsystem is not enabled. CP data structures will operate in UNSAFE mode! Please note that UNSAFE mode will not provide strong consistency guarantees.
May 24, 2025 9:15:44 PM com.hazelcast.jet.impl.JetServiceBackend
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Setting number of cooperative threads and default parallelism to 4
May 24, 2025 9:15:44 PM com.hazelcast.internal.diagnostics.Diagnostics
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Diagnostics disabled. To enable add -Dhazelcast.diagnostics.enabled=true to the JVM arguments.
May 24, 2025 9:15:44 PM com.hazelcast.core.LifecycleService
INFO: [10.158.82.85]:29000 [dev] [5.1.3] [10.158.82.85]:29000 is STARTING
May 24, 2025 9:15:44 PM com.hazelcast.internal.cluster.impl.MulticastJoiner
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Trying to join to discovered node: [10.158.82.81]:29000
May 24, 2025 9:15:44 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Initialized new cluster connection between /10.158.82.85:53421 and /10.158.82.81:29000
May 24, 2025 9:15:49 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Initialized new cluster connection between /10.158.82.85:29000 and /10.158.83.54:56347
May 24, 2025 9:15:49 PM com.hazelcast.internal.partition.operation.PartitionStateOperation
WARNING: [10.158.82.85]:29000 [dev] [5.1.3] com.hazelcast.spi.exception.CallerNotMemberException: Not Member! this: [10.158.82.85]:29000, caller: [10.158.82.81]:29000, partitionId: -1, operation: com.hazel
, service: hz:core:partitionService
May 24, 2025 9:15:49 PM com.hazelcast.internal.cluster.ClusterService
INFO: [10.158.82.85]:29000 [dev] [5.1.3]

Members {size:3, ver:7} [
        Member [10.158.82.81]:29000 - 526c60a3-9b0a-4d24-a081-82b4953bf10b
        Member [10.158.83.54]:29000 - d0911cee-63b3-42da-a393-e73ec5642661
        Member [10.158.82.85]:29000 - 11d64a20-a685-44de-8f4d-fcb9da967876 this
]

May 24, 2025 9:15:49 PM com.hazelcast.jet.impl.JobCoordinationService
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Jet started scanning for jobs
May 24, 2025 9:15:49 PM com.hazelcast.internal.partition.operation.MigrationOperation
INFO: [10.158.82.85]:29000 [dev] [5.1.3] Local partition stamp is not equal to master's stamp! Local: 0, Master: 13
```

4: cssmpi2.uwb.edu ⊗ 0 ⚠ 0 0

Output of the remote excecution:

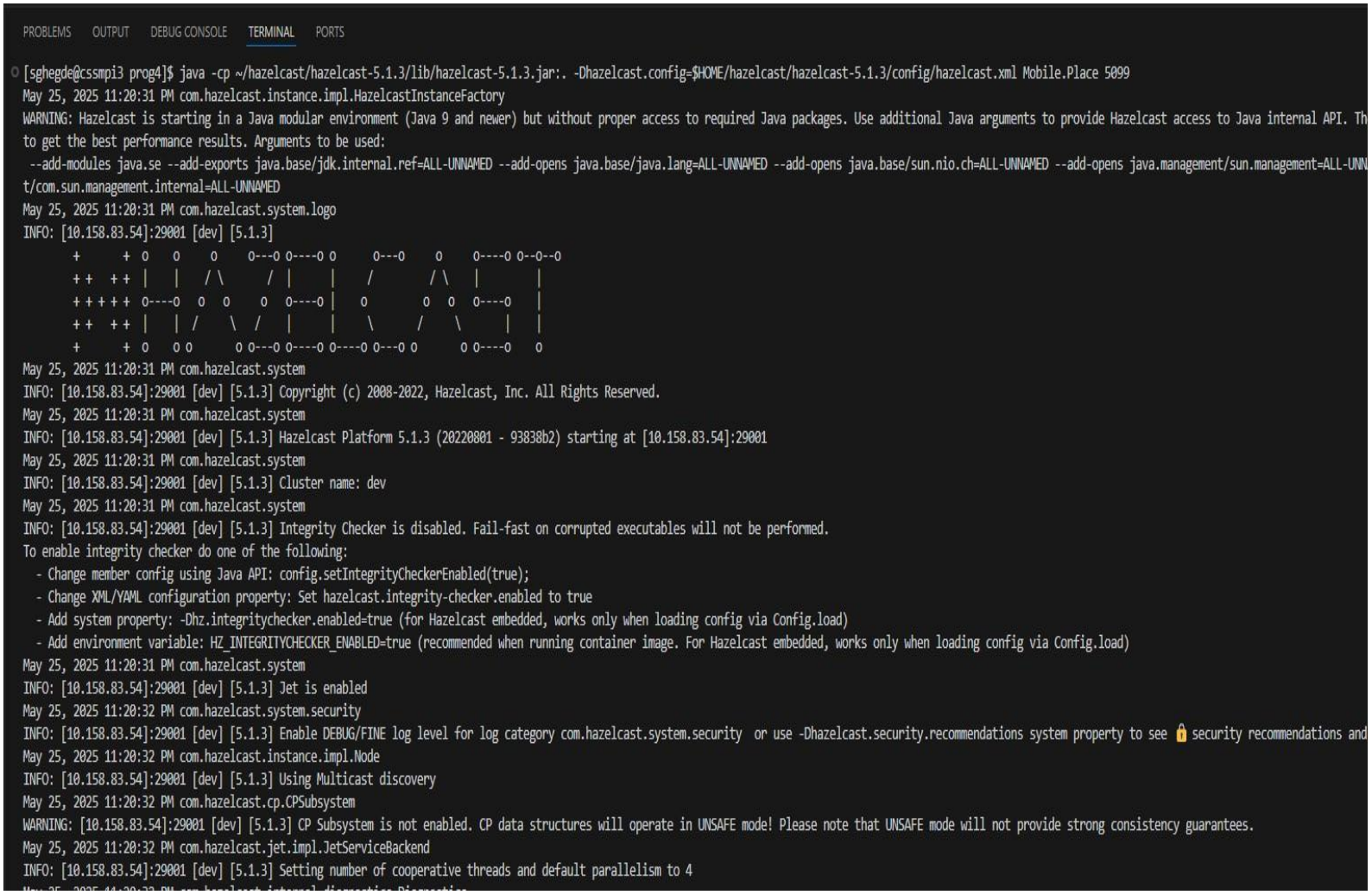
```
File[rfc4862.txt] has 9
File[rfc1659.txt] has 1
File[rfc1155.txt] has 8
File[rfc2131.txt] has 8
File[rfc4271.txt] has 129
File[rfc1772.txt] has 7
File[rfc1390.txt] has 3
File[rfc4941.txt] has 1
File[rfc1123.txt] has 52
File[rfc1042.txt] has 3
File[rfc895.txt] has 2
File[rfc959.txt] has 8
File[rfc1870.txt] has 1
File[rfc1055.txt] has 9
File[rfc1559.txt] has 6
File[rfc865.txt] has 4
File[rfc1356.txt] has 3
File[rfc3551.txt] has 31
Remote Execution Time: 53 ms
----------Shutting down Hazelcast instance----------
May 25, 2025 12:25:09 AM com.hazelcast.core.LifecycleService
INFO: [10.158.82.88]:29000 [dev] [5.1.3] [10.158.82.88]:29000 is SHUTTING_DOWN
May 25, 2025 12:25:09 AM com.hazelcast.instance.impl.Node
INFO: [10.158.82.88]:29000 [dev] [5.1.3] Shutting down multicast service...
May 25, 2025 12:25:09 AM com.hazelcast.internal.partition.InternalPartitionService
```

Additional features:

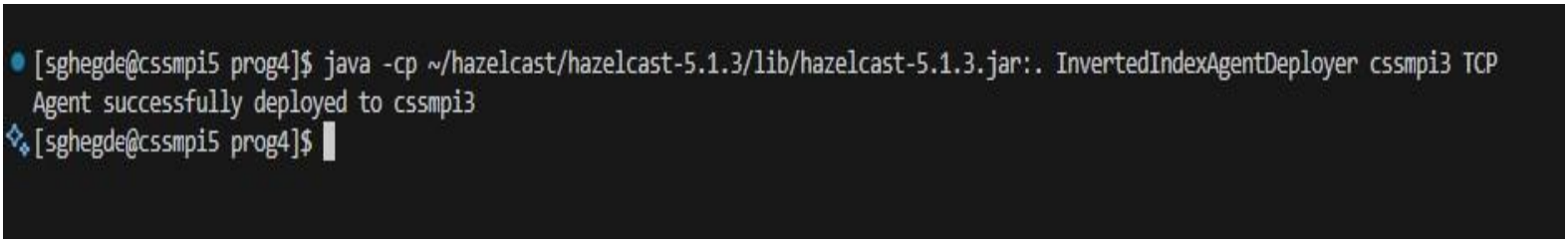
1. Mobile Agent addition:

   I have written the mobile agent required and the code is attached in the source code section. The below are the screenshots of the mobile agent feature.

   Place & Hazelcast Initialization:



Mobile agent deploy screen:



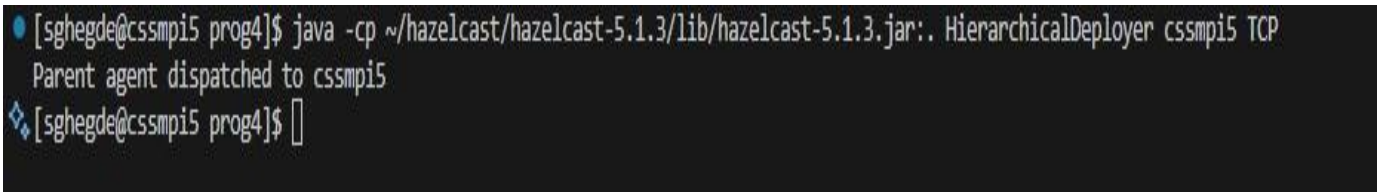
Mobile agent output:

```
File[rfc4456.txt] has 2
File[rfc6152.txt] has 1
File[rfc4941.txt] has 1
File[rfc1534.txt] has 1
File[rfc895.txt] has 2
File[rfc4502.txt] has 1
File[rfc2355.txt] has 2
File[rfc1184.txt] has 2
File[rfc919.txt] has 1
File[rfc3417.txt] has 3
File[rfc5734.txt] has 48
File[rfc1001.txt] has 151
File[rfc865.txt] has 4
File[rfc2460.txt] has 15
File[rfc1356.txt] has 3
File[rfc5036.txt] has 60
Execution Time in cssmpi3 : 27 ms
May 25, 2025 11:20:51 PM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29001 [dev] [5.1.3] [10.158.83.54]:29001 is SHUTTING_DOWN
May 25, 2025 11:20:52 PM com.hazelcast.instance.impl.Node
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Shutting down multicast service...
May 25, 2025 11:20:52 PM com.hazelcast.instance.impl.Node
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Shutting down connection manager...
May 25, 2025 11:20:52 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Connection[id=5, /10.158.83.54:29001->/10.158.82.88:45983, qualifier=null, endpoint=[10.158.82.88]:29000, remoteUuid=646a92b5-8aaa-45c5-9814-c21ab440d1f1, alive=false, connectionType=MEMBER, planeIndex=0]
 closed. Reason: TcpServer is stopping
May 25, 2025 11:20:52 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Connection[id=3, /10.158.83.54:29001->/10.158.82.87:55953, qualifier=null, endpoint=[10.158.82.87]:29000, remoteUuid=2b5fb5dc-59b3-4975-b00f-ad56bcce96ff, alive=false, connectionType=MEMBER, planeIndex=0]
 closed. Reason: TcpServer is stopping
May 25, 2025 11:20:52 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Connection[id=1, /10.158.83.54:57289->/10.158.83.54:29000, qualifier=null, endpoint=[10.158.83.54]:29000, remoteUuid=33ddcfdf-c34e-4c02-b3f7-63aa4146320c, alive=false, connectionType=MEMBER, planeIndex=0]
 closed. Reason: TcpServer is stopping
May 25, 2025 11:20:52 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Connection[id=4, /10.158.83.54:29001->/10.158.82.87:59877, qualifier=null, endpoint=[10.158.82.87]:29001, remoteUuid=28181b3a-10a8-44e2-9e00-ffee9c14d335, alive=false, connectionType=MEMBER, planeIndex=0]
 closed. Reason: TcpServer is stopping
May 25, 2025 11:20:52 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Connection[id=2, /10.158.83.54:29001->/10.158.82.81:35301, qualifier=null, endpoint=[10.158.82.81]:29000, remoteUuid=8f4638a2-3167-40ed-8742-df13d801d399, alive=false, connectionType=MEMBER, planeIndex=0]
 closed. Reason: TcpServer is stopping
May 25, 2025 11:20:52 PM com.hazelcast.instance.impl.Node
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Shutting down node engine...
May 25, 2025 11:20:52 PM com.hazelcast.instance.impl.NodeExtension
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Destroying node NodeExtension.
May 25, 2025 11:20:52 PM com.hazelcast.instance.impl.Node
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Hazelcast Shutdown is completed in 269 ms.
May 25, 2025 11:20:52 PM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29001 [dev] [5.1.3] [10.158.83.54]:29001 is SHUTDOWN
^C[sghegde@cssmpi3 prog4]$
```

2. Hierarchy development:

I have implemented a simplified hierarchy with just one parent node and 3 child node to demonstrate the functionality. The machine hostnames are hardcoded to maintain focus on core functionality and avoid expanding beyond the intended scope.

Command Screen:



Parent Start Screen:

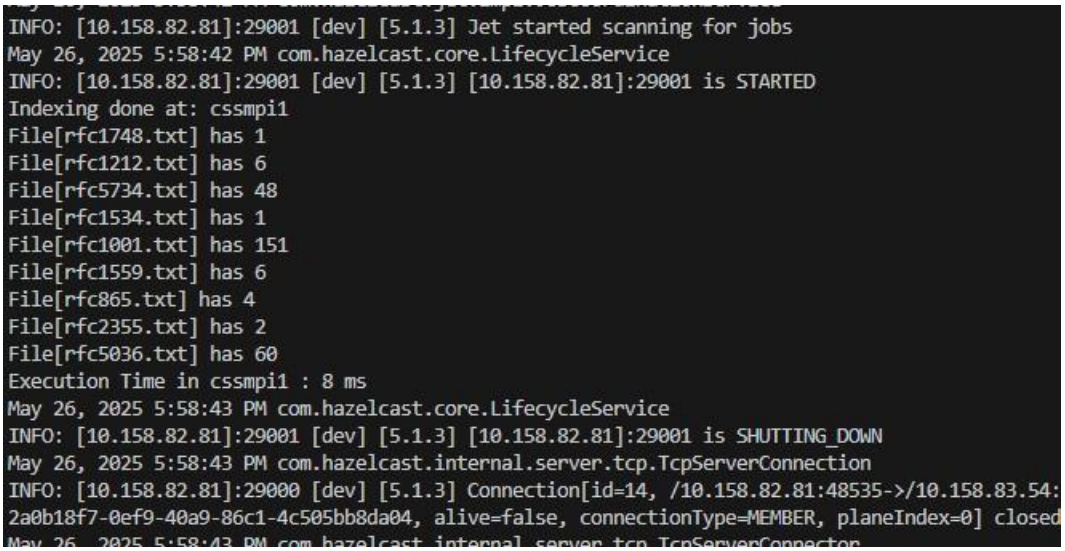
```
Receiving agent Mobile.HierarchicalIndexingAgent...
Deserialized agent with ID: 1673
Started agent thread for: Mobile.HierarchicalIndexingAgent
Agent 1673 running on cssmpi5, invoking: init
Hierarchical agent starting at: cssmpi5
Dispatched child to cssmpi1
Dispatched child to cssmpi2
Dispatched child to cssmpi3
May 26, 2025 5:43:05 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.82.88]:29000 [dev] [5.1.3] Initialized new cluster connection between /10.158.82.88:29000 and /10.158.82.85:42173
May 26, 2025 5:43:06 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.82.88]:29000 [dev] [5.1.3] Initialized new cluster connection between /10.158.82.88:29000 and /10.158.82.81:33621
May 26, 2025 5:43:06 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.82.88]:29000 [dev] [5.1.3] Initialized new cluster connection between /10.158.82.88:29000 and /10.158.83.54:34381
May 26, 2025 5:43:10 PM com.hazelcast.internal.config.AbstractConfigLocator
INFO: Loading configuration '/home/NETID/sghegde/hazelcast/hazelcast-5.1.3/config/hazelcast.xml' from System property 'hazelcast.config'
May 26, 2025 5:43:10 PM com.hazelcast.internal.config.AbstractConfigLocator
INFO: Using configuration file at /home/NETID/sghegde/hazelcast/hazelcast-5.1.3/config/hazelcast.xml
May 26, 2025 5:43:10 PM com.hazelcast.system.logo
INFO: [10.158.82.88]:29001 [dev] [5.1.3]
       +       + o    o      o     o---o o----o o      o---o     o     o----o o--o--o
       + +   + + |    |     / \      /  |      |     /         / \    |         |
       + + + + +  o----o   o   o     o   o----o |    o        o   o   o----o    |
       + +   + +  |    |  /     \   /    |      |     \       /     \       |    |
       +       +  o    o o       o o---o o----o o----o o---o o       o o----o    o
May 26, 2025 5:43:10 PM com.hazelcast.system
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Copyright (c) 2008-2022, Hazelcast, Inc. All Rights Reserved.
May 26, 2025 5:43:10 PM com.hazelcast.system
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Hazelcast Platform 5.1.3 (20220801 - 93838b2) starting at [10.158.82.88]:29001
May 26, 2025 5:43:10 PM com.hazelcast.system
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Cluster name: dev
May 26, 2025 5:43:10 PM com.hazelcast.system
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Integrity Checker is disabled. Fail-fast on corrupted executables will not be performed.
To enable integrity checker do one of the following:
  - Change member config using Java API: config.setIntegrityCheckerEnabled(true);
  - Change XML/YAML configuration property: Set hazelcast.integrity-checker.enabled to true
  - Add system property: -Dhz.integritychecker.enabled=true (for Hazelcast embedded, works only when loading config via Config.load)
  - Add environment variable: HZ_INTEGRITYCHECKER_ENABLED=true (recommended when running container image. For Hazelcast embedded, works only when loading config via Config.load)
May 26, 2025 5:43:10 PM com.hazelcast.system
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Jet is enabled
May 26, 2025 5:43:10 PM com.hazelcast.system.security
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Enable DEBUG/FINE log level for log category com.hazelcast.system.security  or use -Dhazelcast.security.recommendations system property to see 🔒 security recomme
ndations and the status of current config.
May 26, 2025 5:43:10 PM com.hazelcast.instance.impl.Node
```

Parent Output:

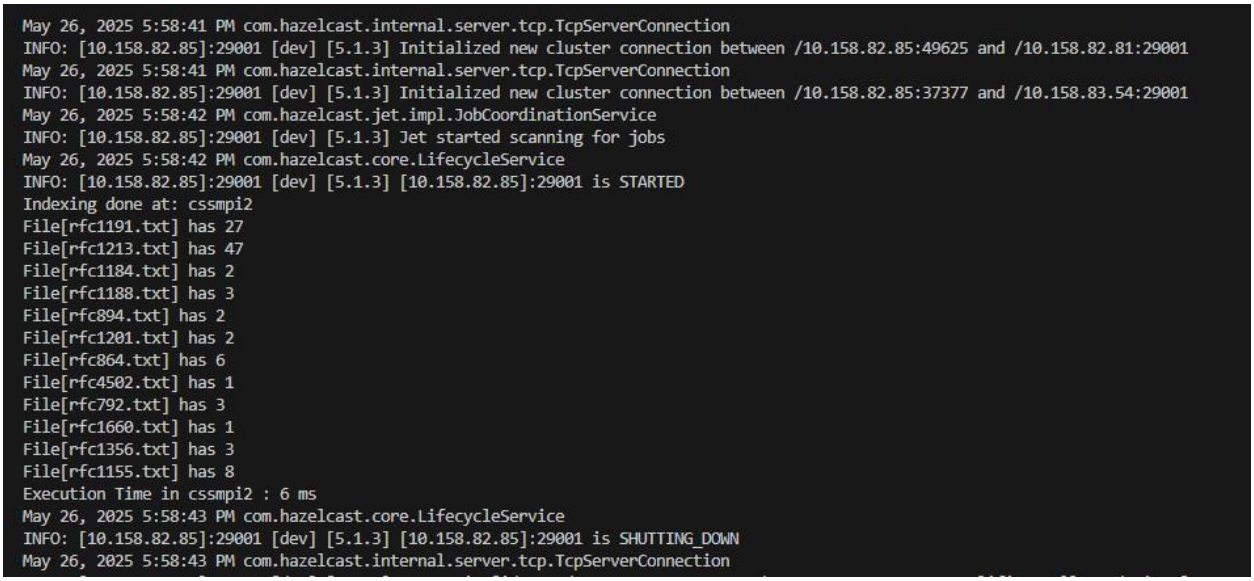
```
PROBLEMS 2   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS 1

May 26, 2025 5:58:46 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Initialized new cluster connection between /10.158.82.88:35881 and /10.158.82.88:29000
May 26, 2025 5:58:47 PM com.hazelcast.jet.impl.JobCoordinationService
INFO: [10.158.82.88]:29001 [dev] [5.1.3] Jet started scanning for jobs
May 26, 2025 5:58:47 PM com.hazelcast.core.LifecycleService
INFO: [10.158.82.88]:29001 [dev] [5.1.3] [10.158.82.88]:29001 is STARTED
=== Aggregated Results from All Children in Parent: cssmpi5 ===
No files indexed or no matches found in cssmpi1-9735
Results from: cssmpi2-547
  File[rfc4456.txt] has 2
  File[rfc3417.txt] has 3
  File[rfc5734.txt] has 48
  File[rfc1042.txt] has 3
  File[rfc2115.txt] has 2
  File[rfc3986.txt] has 3
  File[rfc1870.txt] has 1
Results from: cssmpi1-6578
  File[rfc6152.txt] has 1
  File[rfc826.txt] has 1
  File[rfc1188.txt] has 3
  File[rfc5065.txt] has 3
  File[rfc4502.txt] has 1
  File[rfc1659.txt] has 1
  File[rfc1989.txt] has 2
Results from: cssmpi3-413
  File[rfc2131.txt] has 8
  File[rfc1188.txt] has 3
  File[rfc1534.txt] has 1
  File[rfc791.txt] has 12
  File[rfc1044.txt] has 11
  File[rfc5730.txt] has 1
  File[rfc3986.txt] has 3
  File[rfc865.txt] has 4
  File[rfc1356.txt] has 3
  File[rfc1724.txt] has 4
  File[rfc3592.txt] has 2
```

Child outputs:

Child 1:



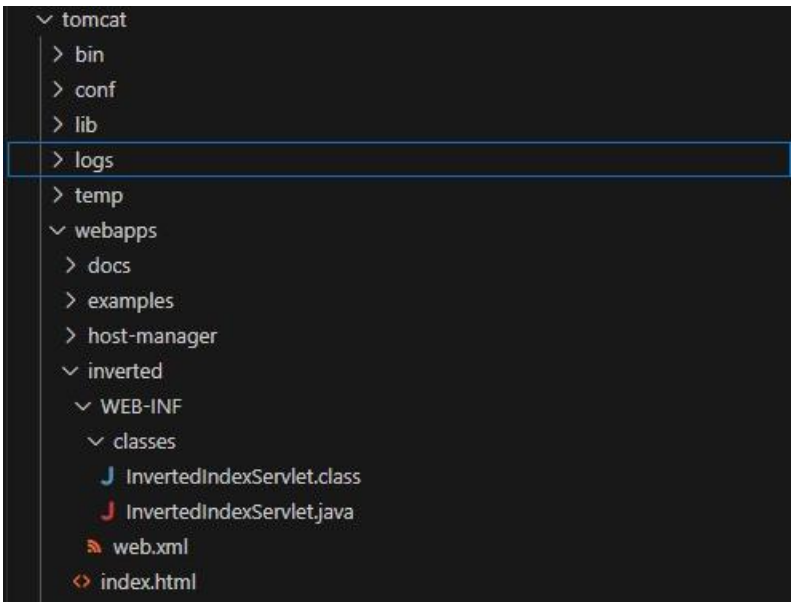
child 2:



Child 3:

```
May 26, 2025 5:58:41 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Initialized new cluster connection between /10.158.83.54:29001 and /10.158.82.85:37377
May 26, 2025 5:58:42 PM com.hazelcast.jet.impl.JobCoordinationService
INFO: [10.158.83.54]:29001 [dev] [5.1.3] Jet started scanning for jobs
May 26, 2025 5:58:42 PM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29001 [dev] [5.1.3] [10.158.83.54]:29001 is STARTED
Indexing done at: cssmpi3
File[rfc2863.txt] has 2
File[rfc2131.txt] has 8
File[rfc4456.txt] has 2
File[rfc1658.txt] has 3
File[rfc1694.txt] has 2
File[rfc3418.txt] has 2
File[rfc4941.txt] has 1
File[rfc1989.txt] has 2
File[rfc2790.txt] has 1
File[rfc919.txt] has 1
File[rfc5681.txt] has 112
File[rfc3417.txt] has 3
File[rfc1044.txt] has 11
File[rfc5322.txt] has 1
File[rfc3986.txt] has 3
File[rfc1350.txt] has 1
File[rfc1724.txt] has 4
Execution Time in cssmpi3 : 7 ms
May 26, 2025 5:58:43 PM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29001 [dev] [5.1.3] [10.158.83.54]:29001 is SHUTTING_DOWN
May 26, 2025 5:58:43 PM com.hazelcast.instance.impl.Node
```

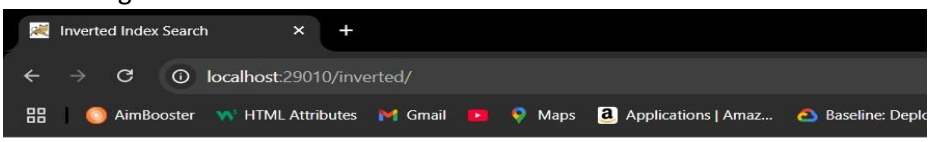
3. Servlet Implementation using Java, Tomcat and Hazelcast:
   folder structure:



Server start

```
[sghegde@cssmpi3 classes]$ ~/tomcat/bin/startup.sh
Using CATALINA_BASE:   /home/NETID/sghegde/tomcat
Using CATALINA_HOME:   /home/NETID/sghegde/tomcat
Using CATALINA_TMPDIR: /home/NETID/sghegde/tomcat/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/NETID/sghegde/tomcat/bin/bootstrap.jar:/home/NETID/sghegde/tomcat/bin/tomcat-juli.jar
Using CATALINA_OPTS:
Tomcat started.
[sghegde@cssmpi3 classes]$
```

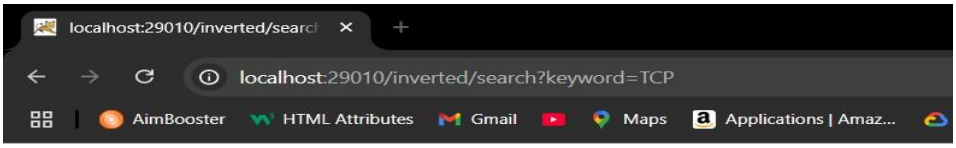
Search Page:



Search Keyword in Files

Enter keyword: Search

Results Page:



Results for keyword: TCP

rfc1534.txt: 1 occurrences

rfc4861.txt: 10 occurrences

rfc1658.txt: 3 occurrences

rfc826.txt: 1 occurrences

rfc867.txt: 4 occurrences

rfc5322.txt: 1 occurrences

rfc1724.txt: 4 occurrences

rfc3551.txt: 8 occurrences

rfc1035.txt: 16 occurrences

rfc2328.txt: 6 occurrences

rfc1034.txt: 10 occurrences

rfc959.txt: 8 occurrences

rfc5681.txt: 110 occurrences

rfc2580.txt: 3 occurrences

rfc793.txt: 326 occurrences

rfc862.txt: 4 occurrences

rfc4271.txt: 129 occurrences

rfc3411.txt: 1 occurrences

rfc5734.txt: 48 occurrences

Discussions:

I have implemented three additional features, which are as listed below. Each of these will be discussed in detail later in the page.

1. Integration of Mobile Agents and hazelcast.
2. Implementation of a simple Hierarchy.
3. Servlet implementation.

Mobile Agents: The mobile agent feature was implemented by creating the agent and a deployer class that can be dispatched to a remote node using Java RMI and the Place mechanism. The agent carries both code and data with it, migrates to a target machine (node), performs local inverted indexing by retrieving data from the distributed Hazelcast map, and computes word occurrence counts on that node. After completing execution, the agent returns the results back to the originator. The current implementation starts a hazelcast server when place command is ran enabling the data to be present and the server to be active when the mobile agent accesses the command. The execution speed was increased as compared to the local indexing program. Although the mobile agent is efficient, the current version only supports single-hop migration with fixed remote node as the focus of this feature was to implement the agent and compare the performance. This will be improved in later versions of the program.

Hierarchy implementation: The hierarchical dispatch feature was implemented by introducing a parent agent (HierarchicalIndexingAgent) that distributes indexing tasks to multiple child agents (ChildIndexingAgent) across selected nodes in the Hazelcast cluster. Each child agent is responsible for performing inverted indexing on the files stored locally at its assigned node. The parent agent tracks which nodes have been dispatched, waits for all child agents to finish their tasks, and collects the results from a shared Hazelcast map, where each child agent stores its output. This model mirrors a tree-based task delegation system, improving scalability by parallelizing work in a structured manner. Currently, the number of child agents and their target nodes are fixed. Additionally, child agents do not support fault tolerance or timeout handling if a node fails or becomes unresponsive. Enhancements could include dynamic partitioning of the file map based on data distribution, automatic recovery of failed child tasks, and deeper hierarchical layers (e.g., sub-children.

Servlet implementation: The servlet-based implementation adds a web interface to the inverted indexing system, allowing users to query the occurrence of a keyword via a simple HTML form. The InvertedIndexServlet is deployed on an Apache Tomcat server and accepts HTTP GET requests at the /search endpoint. When a user submits the keyword through the form, the servlet retrieves it as a request parameter, connects to the local Hazelcast instance, performs indexing over the file map, and returns the result as an HTML response. This enables non-technical users to access the distributed indexing functionality through a browser without needing to run command-line programs. The current servlet is limited to local execution and does not utilize the distributed or remote indexing features supported by Hazelcast. Future enhancements could include integration with the remote execution mechanism for full cluster utilization.

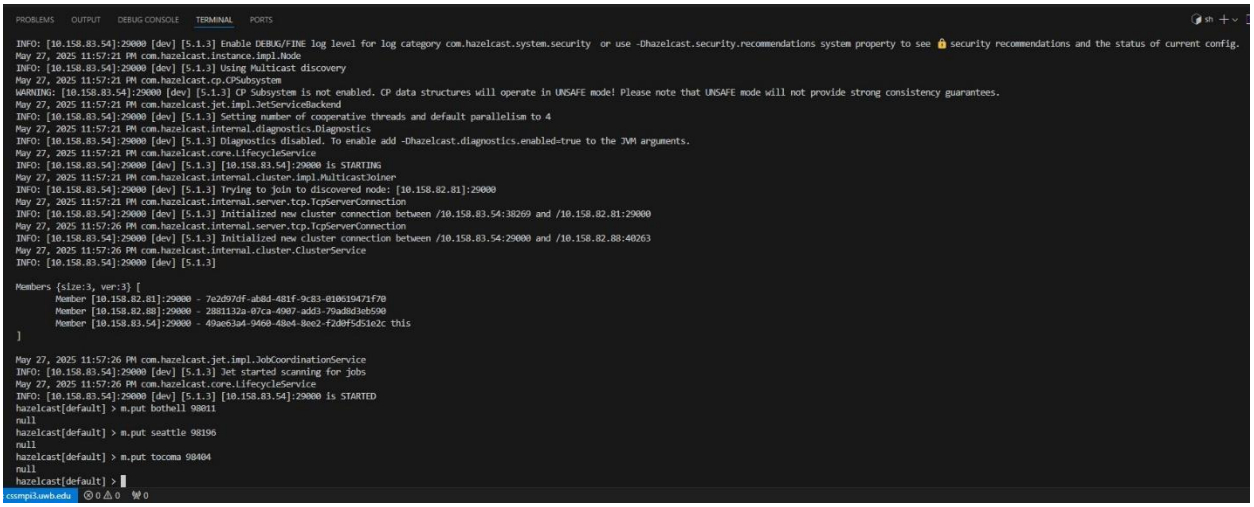
This completes the additional features explaination. All the Source code is in the Source Code section. The execution outputs are pasted in the Execution section.

Lab 8:

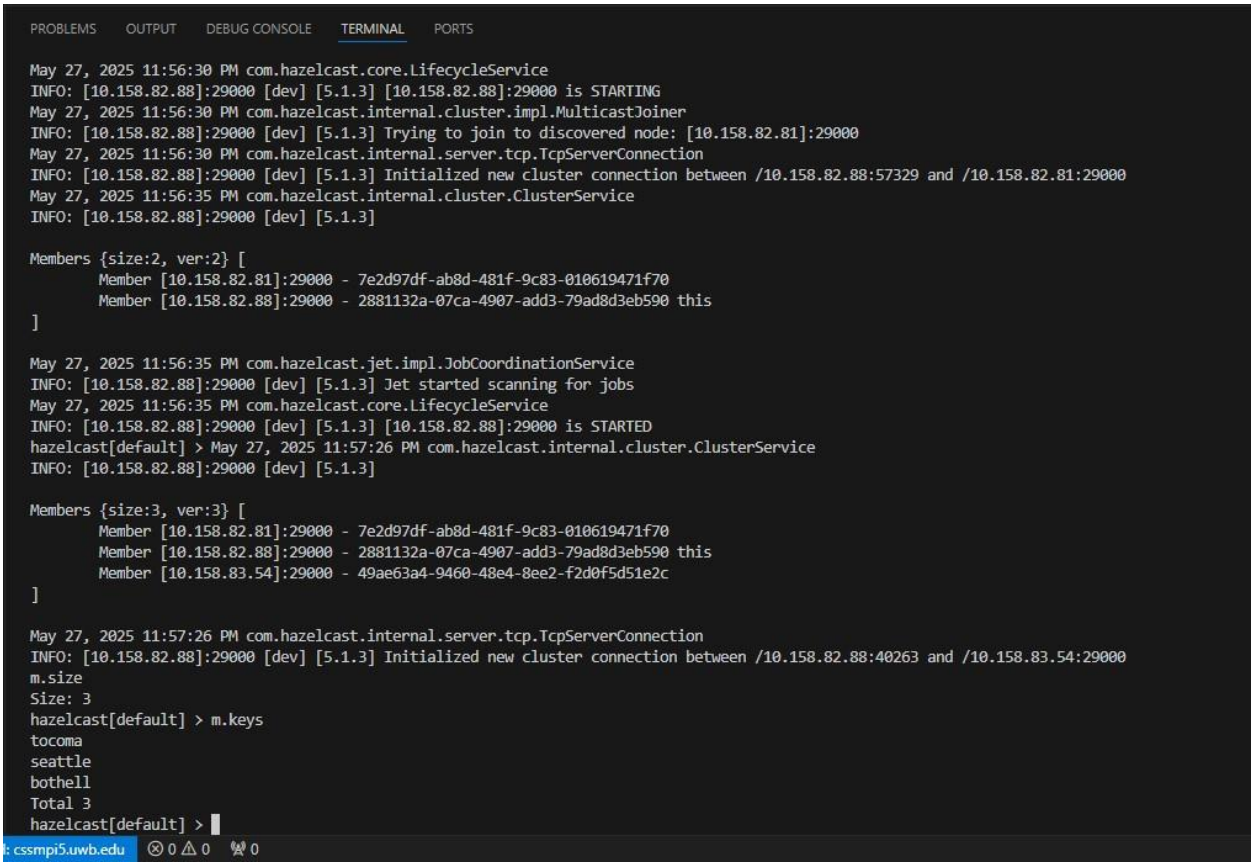
Outputs:

Hazelcast 1:

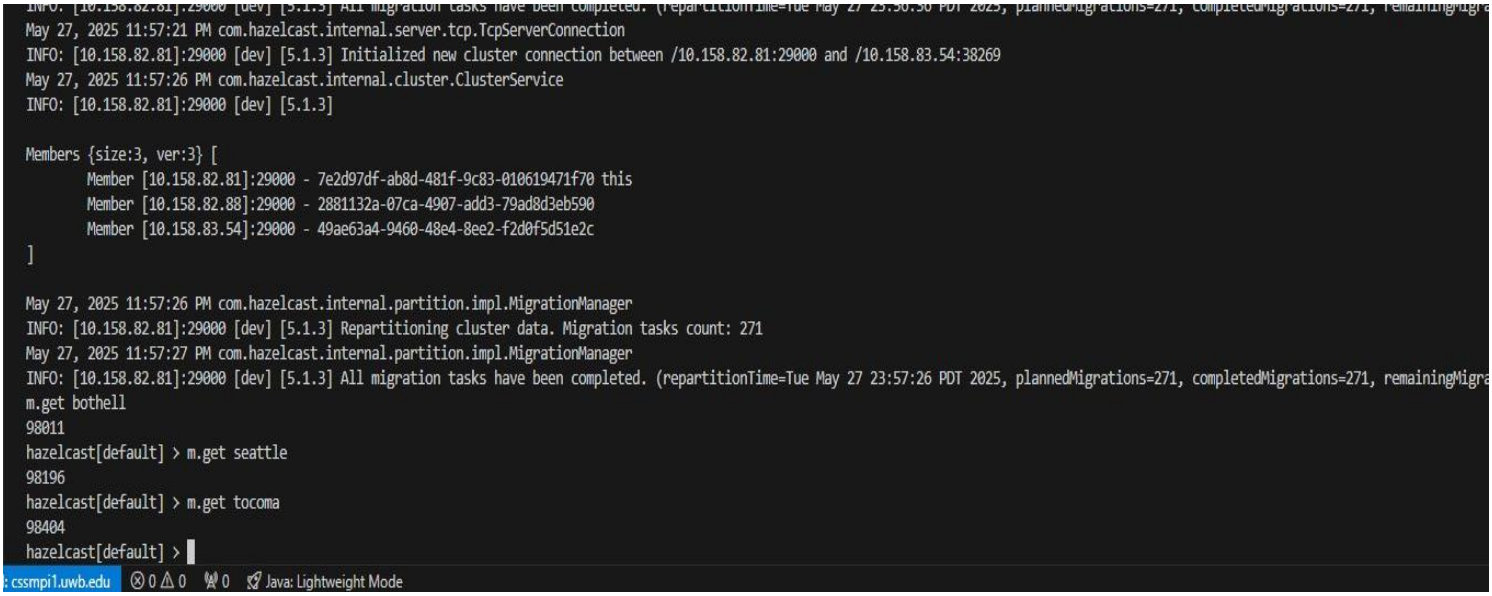
Map.Put



Key Fetch:

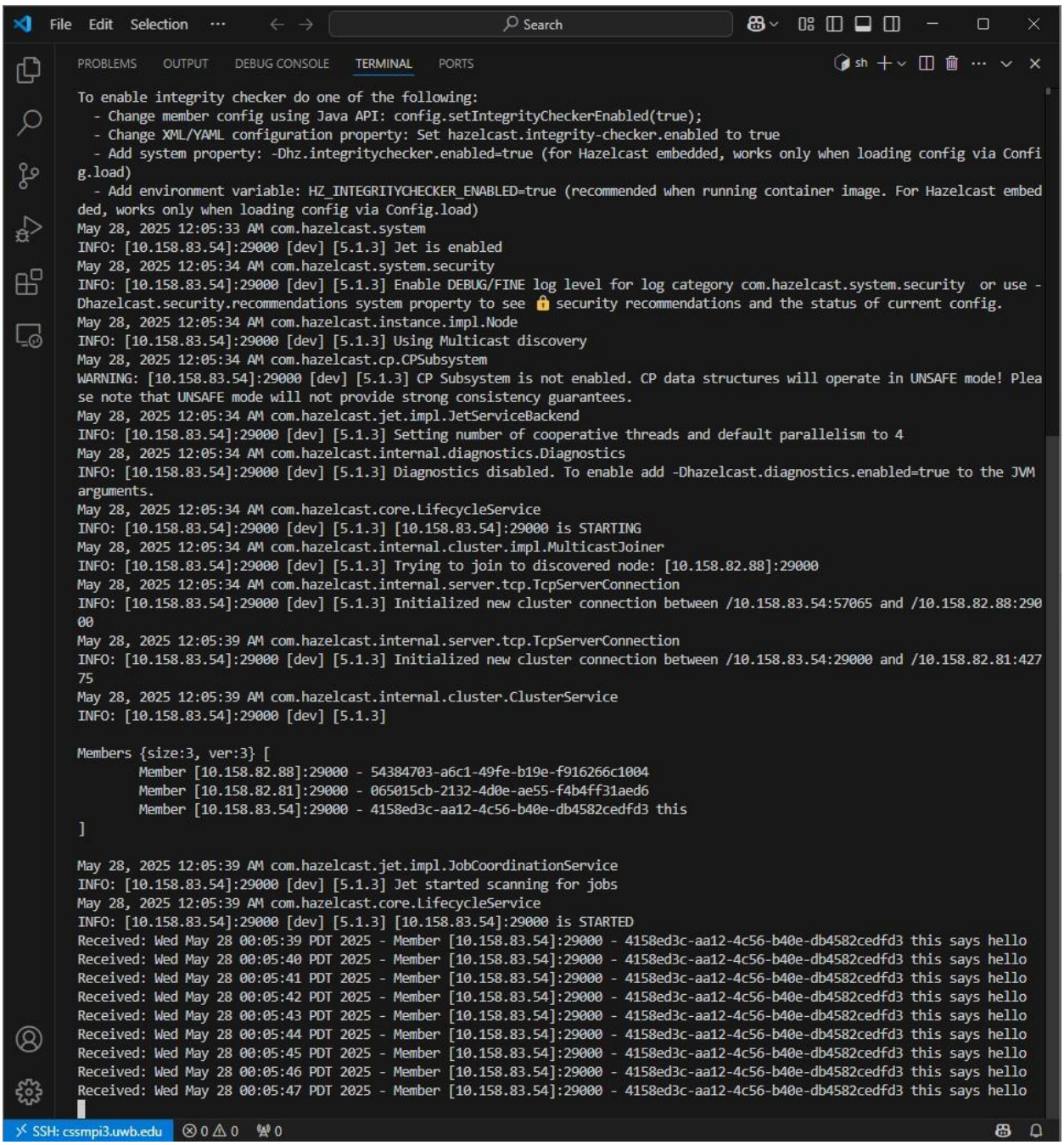


Value Fetch:

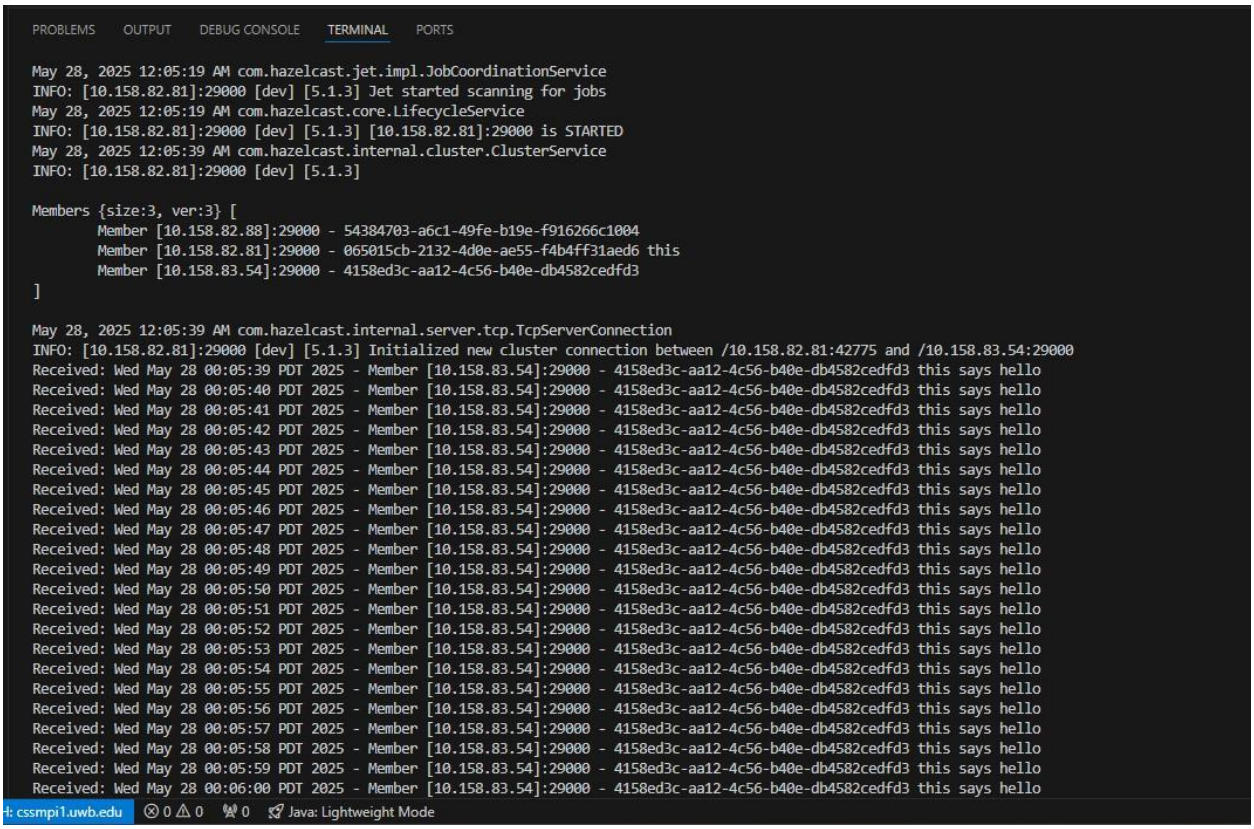


Hazelcast 2:

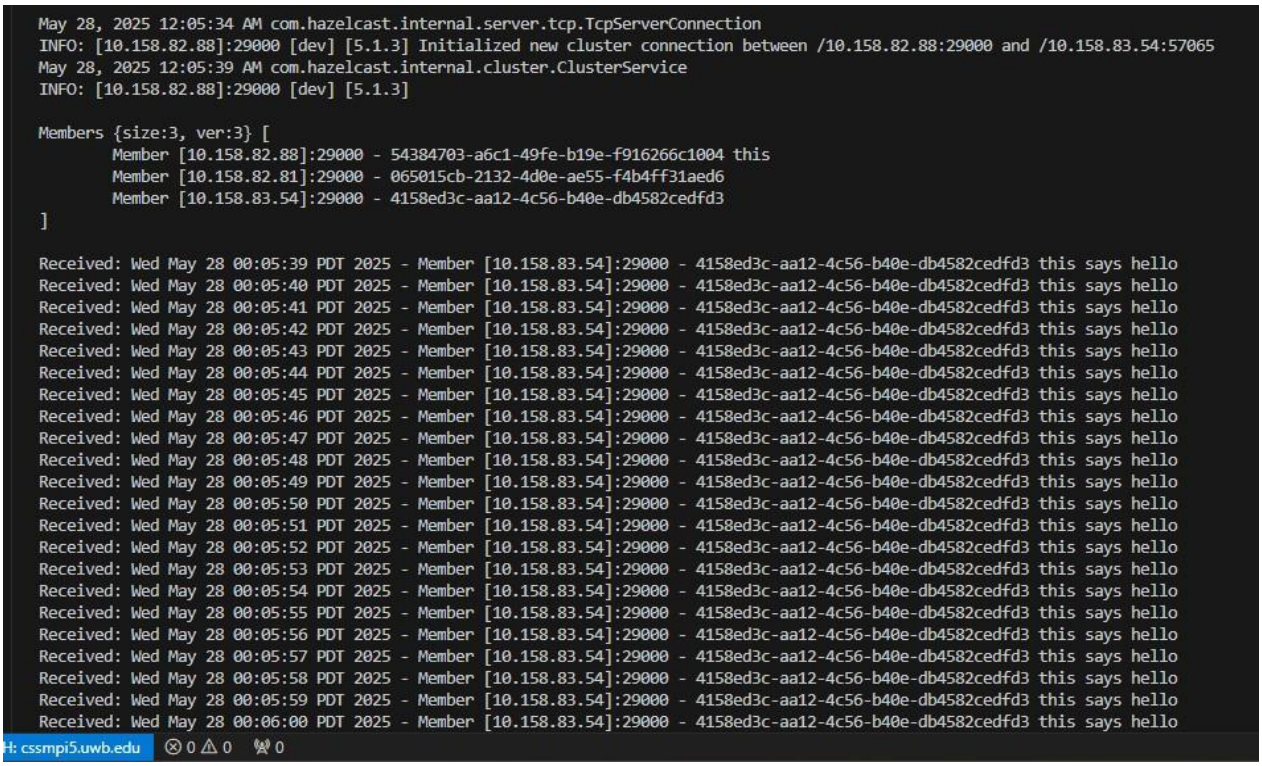
Message sender:


File Edit Selection ···
Search
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
sh
To enable integrity checker do one of the following:
- Change member config using Java API: config.setIntegrityCheckerEnabled(true);
- Change XML/YAML configuration property: Set hazelcast.integrity-checker.enabled to true
- Add system property: -Dhz.integritychecker.enabled=true (for Hazelcast embedded, works only when loading config via Config.load)
- Add environment variable: HZ_INTEGRITYCHECKER_ENABLED=true (recommended when running container image. For Hazelcast embedded, works only when loading config via Config.load)
May 28, 2025 12:05:33 AM com.hazelcast.system
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Jet is enabled
May 28, 2025 12:05:34 AM com.hazelcast.system.security
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Enable DEBUG/FINE log level for log category com.hazelcast.system.security or use -Dhazelcast.security.recommendations system property to see 🔒 security recommendations and the status of current config.
May 28, 2025 12:05:34 AM com.hazelcast.instance.impl.Node
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Using Multicast discovery
May 28, 2025 12:05:34 AM com.hazelcast.cp.CPSubsystem
WARNING: [10.158.83.54]:29000 [dev] [5.1.3] CP Subsystem is not enabled. CP data structures will operate in UNSAFE mode! Please note that UNSAFE mode will not provide strong consistency guarantees.
May 28, 2025 12:05:34 AM com.hazelcast.jet.impl.JetServiceBackend
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Setting number of cooperative threads and default parallelism to 4
May 28, 2025 12:05:34 AM com.hazelcast.internal.diagnostics.Diagnostics
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Diagnostics disabled. To enable add -Dhazelcast.diagnostics.enabled=true to the JVM arguments.
May 28, 2025 12:05:34 AM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29000 [dev] [5.1.3] [10.158.83.54]:29000 is STARTING
May 28, 2025 12:05:34 AM com.hazelcast.internal.cluster.impl.MulticastJoiner
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Trying to join to discovered node: [10.158.82.88]:29000
May 28, 2025 12:05:34 AM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Initialized new cluster connection between /10.158.83.54:57065 and /10.158.82.88:29000
May 28, 2025 12:05:39 AM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Initialized new cluster connection between /10.158.83.54:29000 and /10.158.82.81:42775
May 28, 2025 12:05:39 AM com.hazelcast.internal.cluster.ClusterService
INFO: [10.158.83.54]:29000 [dev] [5.1.3]
Members {size:3, ver:3} [
Member [10.158.82.88]:29000 - 54384703-a6c1-49fe-b19e-f916266c1004
Member [10.158.82.81]:29000 - 065015cb-2132-4d0e-ae55-f4b4ff31aed6
Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this
]
May 28, 2025 12:05:39 AM com.hazelcast.jet.impl.JobCoordinationService
INFO: [10.158.83.54]:29000 [dev] [5.1.3] Jet started scanning for jobs
May 28, 2025 12:05:39 AM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29000 [dev] [5.1.3] [10.158.83.54]:29000 is STARTED
Received: Wed May 28 00:05:39 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:40 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:41 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:42 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:43 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:44 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:45 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:46 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
Received: Wed May 28 00:05:47 PDT 2025 - Member [10.158.83.54]:29000 - 4158ed3c-aa12-4c56-b40e-db4582cedfd3 this says hello
SSH: cssmpi3.uwb.edu
0 0 0

Message Receiver 1:



Message Receiver 2:



Lab 9:

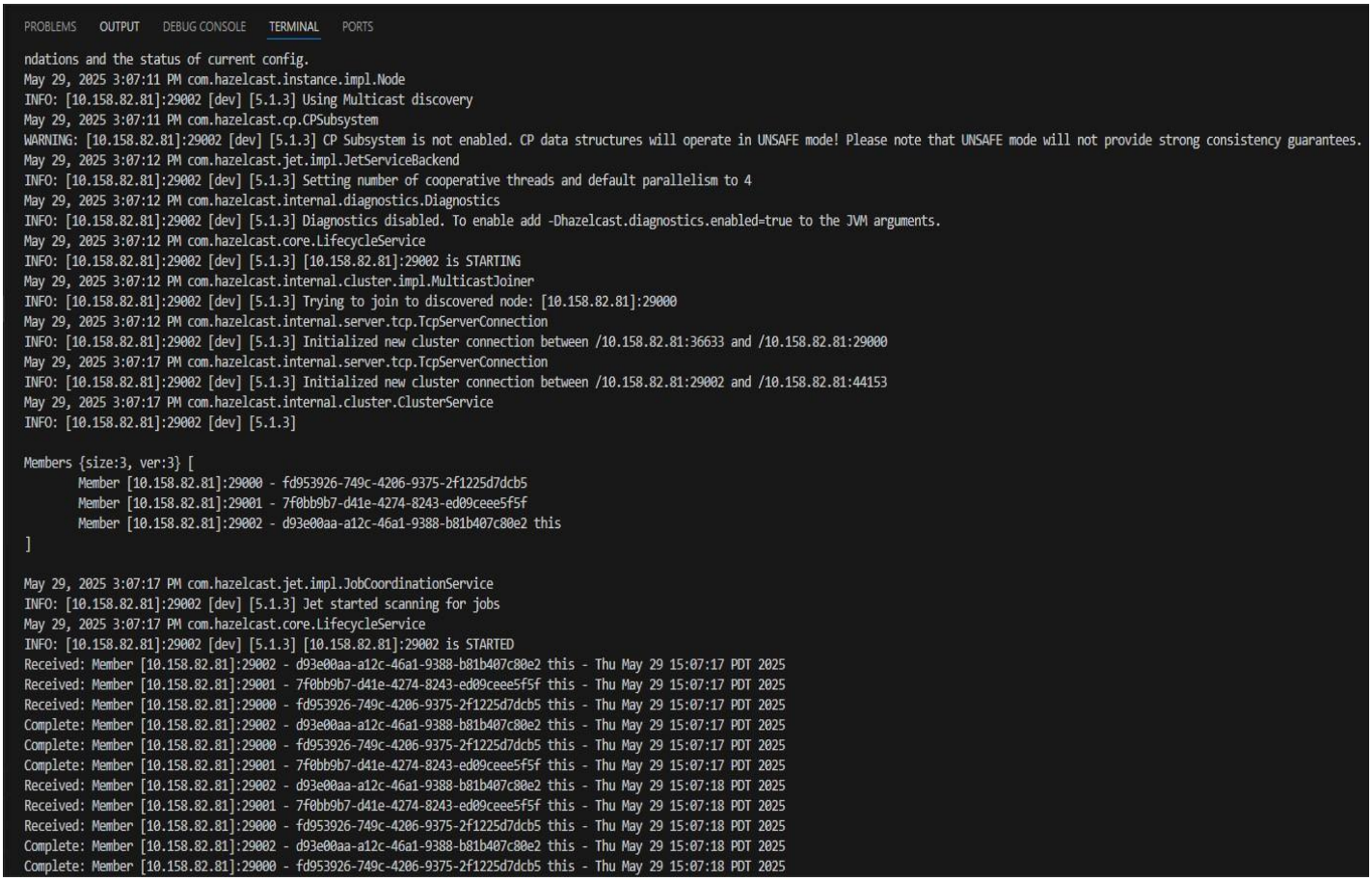
Source code:

```
import com.hazelcast.core.Hazelcast;
import com.hazelcast.core.HazelcastInstance;
import com.hazelcast.core.HazelcastInstanceAware;
import java.util.concurrent.*;
import java.util.Date;
import java.io.Serializable;
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class TimeInstanceAwareCallable
    implements Callable<String>, HazelcastInstanceAware, Serializable {

    private HazelcastInstance hz;

    @Override
    public void setHazelcastInstance( HazelcastInstance hz ) {
    this.hz = hz;
    }

    @Override
    public String call() {
        StringBuilder result = new StringBuilder();

        try {
            Process process = Runtime.getRuntime().exec("who");
            BufferedReader reader = new BufferedReader(
                    new InputStreamReader(process.getInputStream()));

            String line;
            while ((line = reader.readLine()) != null) {
                result.append(line).append("\n");
            }

            process.waitFor();
        } catch (Exception e) {
            result.append("Error: ").append(e.getMessage());
        }

        return result.toString();
```

```
    }
}
```

Output:

Scenario 1: Basic Master output



Scenario 2: Execution of “who”

```
May 30, 2025 6:29:31 PM com.hazelcast.instance.impl.Node
INFO: [10.158.83.54]:29002 [dev] [5.1.3] Using Multicast discovery
May 30, 2025 6:29:31 PM com.hazelcast.cp.CPSubsystem
WARNING: [10.158.83.54]:29002 [dev] [5.1.3] CP Subsystem is not enabled. CP data structures will operate in UNSAFE mode! Please note that UNSAFE mode will not provide strong consistency guarantees.
May 30, 2025 6:29:32 PM com.hazelcast.jet.impl.JetServiceBackend
INFO: [10.158.83.54]:29002 [dev] [5.1.3] Setting number of cooperative threads and default parallelism to 4
May 30, 2025 6:29:32 PM com.hazelcast.internal.diagnostics.Diagnostics
INFO: [10.158.83.54]:29002 [dev] [5.1.3] Diagnostics disabled. To enable add -Dhazelcast.diagnostics.enabled=true to the JVM arguments.
May 30, 2025 6:29:32 PM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29002 [dev] [5.1.3] [10.158.83.54]:29002 is STARTING
May 30, 2025 6:29:33 PM com.hazelcast.internal.cluster.impl.MulticastJoiner
INFO: [10.158.83.54]:29002 [dev] [5.1.3] Trying to join to discovered node: [10.158.83.54]:29000
May 30, 2025 6:29:33 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29002 [dev] [5.1.3] Initialized new cluster connection between /10.158.83.54:54135 and /10.158.83.54:29000
May 30, 2025 6:29:38 PM com.hazelcast.internal.server.tcp.TcpServerConnection
INFO: [10.158.83.54]:29002 [dev] [5.1.3] Initialized new cluster connection between /10.158.83.54:29002 and /10.158.83.54:55715
May 30, 2025 6:29:38 PM com.hazelcast.internal.cluster.ClusterService
INFO: [10.158.83.54]:29002 [dev] [5.1.3]

Members {size:3, ver:5} [
        Member [10.158.83.54]:29000 - fffd51d3-5541-4bda-9330-99109875a7e5
        Member [10.158.83.54]:29001 - 6b980ca9-7daf-412d-b700-4b261a539a63
        Member [10.158.83.54]:29002 - ef716c1a-1115-4c05-9971-f4bcf99ee7d4 this
]

May 30, 2025 6:29:38 PM com.hazelcast.jet.impl.JobCoordinationService
INFO: [10.158.83.54]:29002 [dev] [5.1.3] Jet started scanning for jobs
May 30, 2025 6:29:38 PM com.hazelcast.core.LifecycleService
INFO: [10.158.83.54]:29002 [dev] [5.1.3] [10.158.83.54]:29002 is STARTED
Received: Member [10.158.83.54]:29001 - 6b980ca9-7daf-412d-b700-4b261a539a63 this - Fri May 30 18:29:38 PDT 2025
Received: Member [10.158.83.54]:29000 - fffd51d3-5541-4bda-9330-99109875a7e5 this - Fri May 30 18:29:38 PDT 2025
Received: styata   pts/0        2025-05-30 13:16 (10.102.40.165)
knaswa   pts/1        2025-05-30 11:53 (10.102.58.41)
jsasid   pts/2        2025-05-30 14:21 (10.19.82.90)
styata   pts/4        2025-05-30 15:48 (10.102.40.165)
zeweng   pts/11       2025-05-30 18:22 (10.102.47.94)

Complete: Member [10.158.83.54]:29000 - fffd51d3-5541-4bda-9330-99109875a7e5 this - Fri May 30 18:29:38 PDT 2025
Complete: styata   pts/0        2025-05-30 13:16 (10.102.40.165)
knaswa   pts/1        2025-05-30 11:53 (10.102.58.41)
jsasid   pts/2        2025-05-30 14:21 (10.19.82.90)
styata   pts/4        2025-05-30 15:48 (10.102.40.165)
zeweng   pts/11       2025-05-30 18:22 (10.102.47.94)

Complete: Member [10.158.83.54]:29001 - 6b980ca9-7daf-412d-b700-4b261a539a63 this - Fri May 30 18:29:38 PDT 2025
Received: Member [10.158.83.54]:29001 - 6b980ca9-7daf-412d-b700-4b261a539a63 this - Fri May 30 18:29:39 PDT 2025
Received: Member [10.158.83.54]:29000 - fffd51d3-5541-4bda-9330-99109875a7e5 this - Fri May 30 18:29:39 PDT 2025
Received: styata   pts/0        2025-05-30 13:16 (10.102.40.165)
```