**PROJECT REPORT**

***Submitted by***

**RITAM PAL (RA2211033010076)**

**ARYAN KUMAWAT (RA2211033010080)**

**AAYUSH GAUTAM (RA2211033010099)**

**ARAV RAJ JANAIK (RA2211033010107)**

***Under the Guidance of***

**Dr. R. Siva**

**Assistant Professor, Department of CINTEL, SRMIST**

***In partial satisfaction of the requirements for the degree of***

## **BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE ENGINEERING**

**with specialization in Software Engineering**

## Logo, company name Description automatically generated

**SCHOOL OF COMPUTING**

# **COLLEGE OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR - 603203**

**MAY 2023**

Logo, company name

Description automatically generated

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603203

**BONAFIDE CERTIFICATE**

Certified that this Project Report titled **“**ATTENDANCE MANAGEMENT SYSTEM**”** is the bonafide work done by RITAM PAL (RA2211033010076), ARYAN KUMAWAT (RA2211033010080), AAYUSH GAUTAM (RA2211033010099), ARAV RAJ JANAIK (RA2211033010107) who completed the project under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

| **SIGNATURE**  Dr. R. Siva  **OODP – Course Faculty**  Assistant Professor  Department of CINTEL  SRMIST | **SIGNATURE**  Dr. R. Annie Uthra  **Professor and Head of the Department**  Department of CINTEL  SRMIST |
| --- | --- |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **CONTENT** | **PAGE NO.** |
| 1 | Problem Statement | 1 |
| 2 | Modules of Project | 2 |
| 3 | UML Diagrams | 6-14 |
|  | 1. Use Case Diagram | 6 |
|  | 1. Class Diagram | 7 |
|  | 1. Sequence Diagram | 8 |
|  | 1. Collaboration Diagram | 9 |
|  | 1. State Chart Diagram | 10 |
|  | 1. Activity Diagram | 11 |
|  | 1. Package Diagram | 12 |
|  | 1. Component Diagram | 13 |
|  | 1. Deployment Diagram | 14 |
| 4 | Code/Output | 15 |
| 5 | Conclusion | 18 |
| 6 | References | 19 |

**PROBLEM STATEMENT**

Attendance Management keeps track of your employee hours. It is the system you use to document the time your employees work and the [time they take off](https://www.peoplehum.com/glossary/paid-time-off-pto). Attendance Management can be done by recording employee hours on paper, using spreadsheets, punching time cards, or using online attendance software for your company.

Every organization should have an Attendance Management System for employees. Your system needs to comply with the [Fair Labor Standards Act](https://www.peoplehum.com/glossary/fair-labor-standards-act) timekeeping requirements, regardless of how you implement it.

So, there is a need for an Attendance Management System in Workplaces, Schools, Offices, etc.

**MODULES OF PROJECT**

**What is the importance of Attendance Management?**

Every organization should have an Attendance Management System for employees. Your system needs to comply with the [Fair Labor Standards Act](https://www.peoplehum.com/glossary/fair-labor-standards-act) timekeeping requirements, regardless of how you implement it.

Attendance Management Systems allow you to calculate the hours for which employees work accurately. This is especially beneficial if you have employees working on an hourly basis. You need to be able to [calculate the exact wages](https://www.peoplehum.com/glossary/net-pay) you owe to your employees. And, you need to know if you owe overtime wages to any employees.

If you have [salaried employees](https://www.peoplehum.com/glossary/salaried-employee), you can always see how often employees are working. Employees can record and document the time both at and away from your business. You can keep a track of how much they are working, even when you don’t see them.

[Tracking employee time](https://www.peoplehum.com/glossary/time-tracking) will also let you know if employees are punctual. This only works if you require employees to document the exact time they enter or leave. You can see if an employee tends to show up late or skip out early from work.

Attendance management also lets you keep track of how many days off employees use. This is crucial if your business has a policy that lets employees use a certain number of sick or [vacation days](https://www.peoplehum.com/glossary/floating-holiday).

**What are the key elements of Attendance Management System?**

Most of the employer Attendance Management Programs share these five key elements.

‍

**1. Definition of an ‘absence’ for the usage of the AMP**

The term “[absenteeism](https://www.peoplehum.com/glossary/absenteeism-policy)” refers to many types of absences and can include those caused by chilly weather, statutory holidays, vacation, illness, family-related demands and stresses in the workplace. The AMP should focus on absences that distort the workplace and clearly articulate which ones will be “absences” for the purpose of AMP.

Scheduled absences like vacation and statutory holidays are generally considered useful and are usually easily absorbed by the employer. But unscheduled, habitual absences, late arrivals, and early departures can be troublesome to the workplace, an irritant to other employees and carry notable costs and [effects on overall productivity](https://www.peoplehum.com/blog/productivity-hacks).

‍

**2. A distinction between “culpable” and “innocent” absence**

‍One of the most critical distinctions an employer must make when dealing with attendance issues generally, and in an AMP specifically, is between “culpable” and “innocent” [absenteeism](https://www.peoplehum.com/glossary/absence-management).

The AMP should clearly refer to this difference and classification; it’s also helpful if it sets out what can be considered a “non-culpable” absence and the employer’s expectations around it. The distinction is crucial to how the employer handles the issue: while in either of the cases, excessive absenteeism may lead to [termination](https://www.peoplehum.com/glossary/employee-termination), the employer must take a different approach to each. It’s therefore important to classify the absence upfront and clearly, describe how the employer will deal with each.

‍‍**Culpable**

“Culpable” absenteeism is the absence within the employee’s power to address and correct – and is thus “blameworthy” absenteeism for which the employer can hold the employee responsible and typically leads to a progressive disciplinary response.

‍‍**Innocent**

‍“Innocent” (or “non-culpable”) absence is not fault-based (for example, the employer doesn’t suggest an employee abuses [sick leave](https://www.peoplehum.com/glossary/medical-leave-of-absence) entitlements or otherwise) or blameworthy and typically mandates a non-disciplinary progressive coaching process – precisely what an AMP is supposed to handle. But this doesn’t mean an employer is precluded from setting reasonable expectations and taking corrective (though non-disciplinary) measures, or can never terminate an employment.

It does mean the steps leading up to and including any termination should not be “disciplinary”. For the employer to terminate an employee on the basis of innocent absenteeism, it must establish that the employee’s absenteeism is extreme when measured against a reasonable standard; and that there’s little likelihood of a change in the degree of absence in the future.

‍‍**Verification of the absence**

This is key to the employer’s ability to correctly classify the absence. And since employees are required to provide regular ongoing attendance, the employer has the right to be fully informed of the basis of an employee’s absence and any work limitations or modifications she needs to perform her [job duties](https://www.peoplehum.com/glossary/job-description).

It’s vital that the employer consistently enforce its right to proper information and notification from employees: if an employer allows absenteeism to go unchecked, it is very difficult for it to later assert its rights. It’s useful to set out in the AMP the employee’s obligation to provide medical information, what information is required, and from whom.[‍](https://www.peoplehum.com/glossary/medical-leave-of-absence)

[Illness or any medical condition](https://www.peoplehum.com/glossary/medical-leave-of-absence) is probably the most frequent reason for absenteeism, and verification might require the disclosure of employee medical information. Employers should have a limited right to access an employee’s medical information, and a corresponding obligation to protect the confidentiality of that information. Where the absence isn’t for a medical / illness reason, the employer is entitled to know the specific reason for the absence, subject to any limitations imposed by privacy or human rights laws, and to verification of that cause.

‍

**3. A mechanism for absence reporting**

‍The AMP should address, in some detail, both employees’ obligations to report absences and how they should do so, and provide for the employer’s regular [assessment of employees’ attendance records](https://www.peoplehum.com/blog/attendance-management-software). It’s essential to the evaluation and monitoring of any AMP that the employer should implement a system to report and track individual employee absences. Most AMPs provide for employer tracking of paid sick leave, unpaid sick leave, illness during a shift, illness in the family, and medical appointments where absence from work exceeds one hour.

‍**4. Thresholds for entry into and progression through the AMP**

‍The AMP should clearly set out a threshold for entry into and progression through a coaching procedure and detail what each will cover. It’s legally acceptable and practically essential that the employer set an absenteeism “threshold” for entry into the AMP and for moving through each [stage of the coaching process](https://www.peoplehum.com/blog/expanding-the-value-of-coaching) up to and including any [non-disciplinary termination](https://www.peoplehum.com/glossary/employee-termination).

These thresholds are typically based on a specified number of hours of non-culpable absences with a defined period of time (for example, entry at 67.5 hours of non-culpable absences within any 12 months period).  ‍

**5. Preservation of employer discretion**

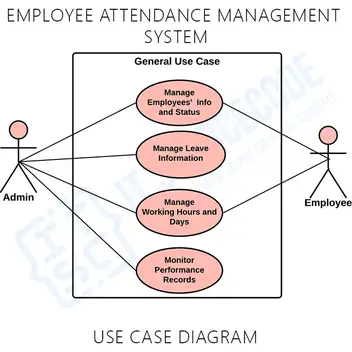
It’s critical that the employer bear in mind its legal obligation not to discriminate and its duty to accommodate under human rights legislation (and any applicable collective agreement) when creating, implementing and enforcing any AMP. If not, it runs the risk that a court, tribunal or arbitrator will find it discriminatory and strike it down.

The AMP should restore the employer’s ability to exercise discretion and deviate from the AMP so it can adjust based on the specific circumstances in each case – and in particular, to make sure it can always meet its duty to accommodate as and when it might arise. A common AMP deviation is not to count the hours of absence towards the AMP thresholds during a period of disability.

Employers must be alive to “flags” raising the possibility of an accommodation need when dealing with absenteeism so they can meet their legal obligations. Innocent absenteeism could be related to an employee’s membership in a group protected by human rights legislation. The employer’s duty to accommodate begins when it has been made aware of the need for accommodation, or the circumstances are such that it ought reasonably to have known of the need for accommodation. Frequently, it’s during the coaching sessions in the context of an AMP where the employee will disclose the need for an accommodation, or there will be “flags” that raise that possibility and trigger the employer’s duty to accommodate.

**UML DIAGRAMS**

1. **USE CASE DIAGRAM**

****

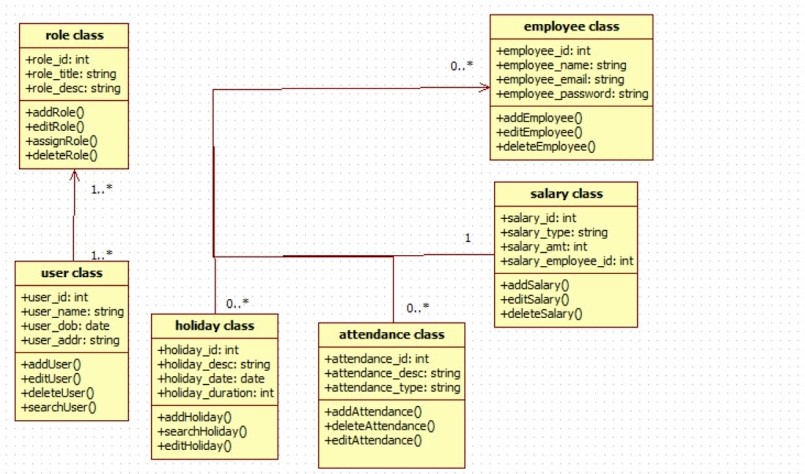
The use case diagram are usually referred to as behavior diagram used to describe the actions of all user in a system. All user describe in use case are actors and the functionality as action of system.  
The Use case diagram is a collection of diagram and text together that make action on goal of a process.

**Use case diagram elements:**

The use case diagram consist of six graphics elements that represent whole system :

* Systems
* Actors
* Use cases
* Association
* Dependencies
* Generalization

1. **CLASS DIAGRAM**

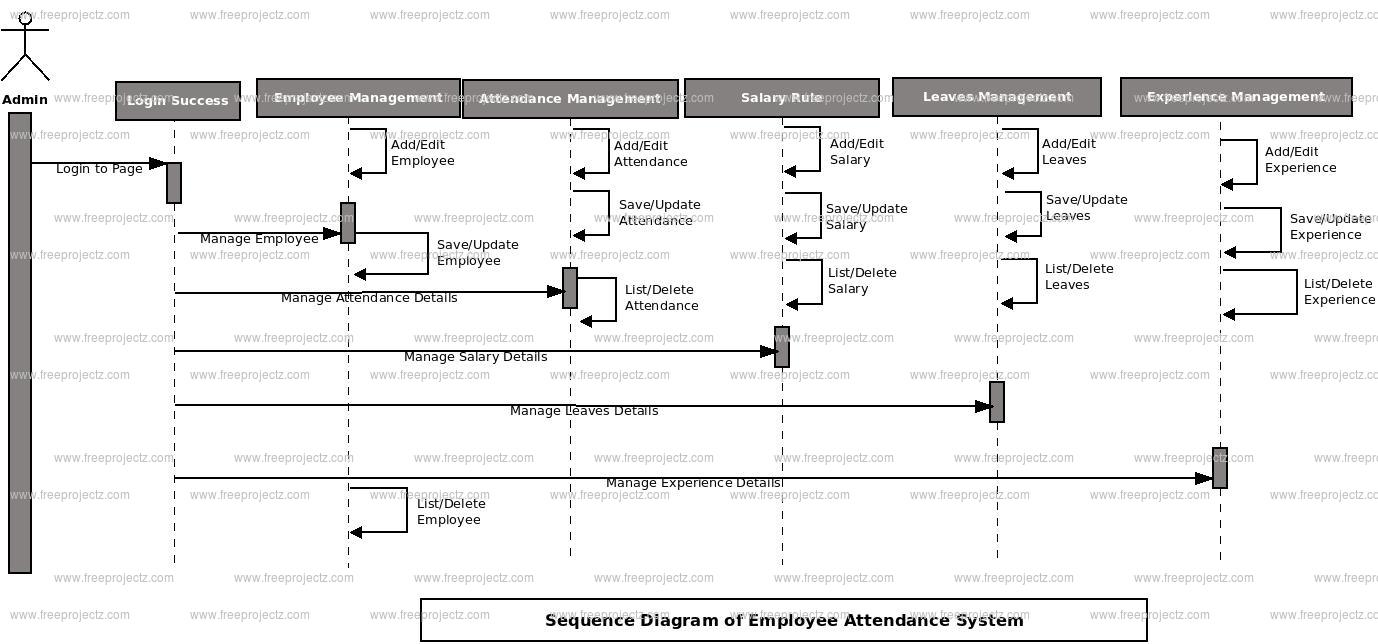
****

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of objectoriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

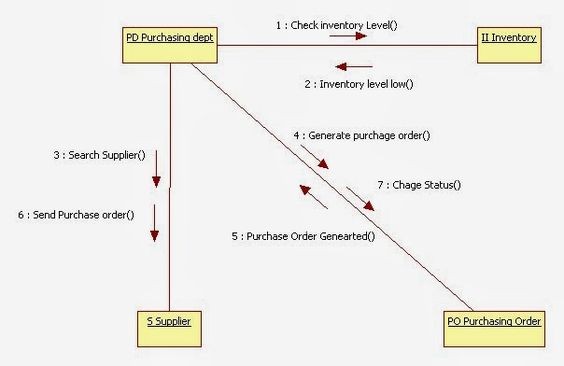
Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

1. **SEQUENCE DIAGRAM**

****

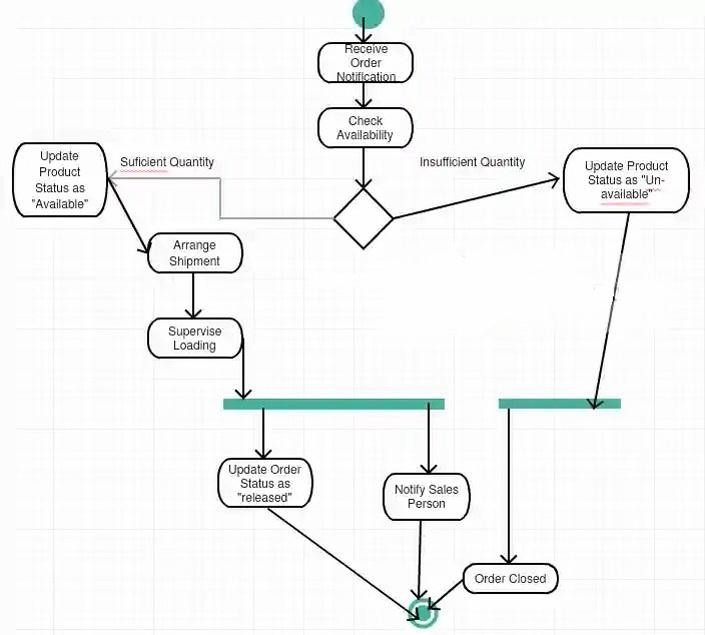
A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process.

1. **COLLABORATION DIAGRAM**



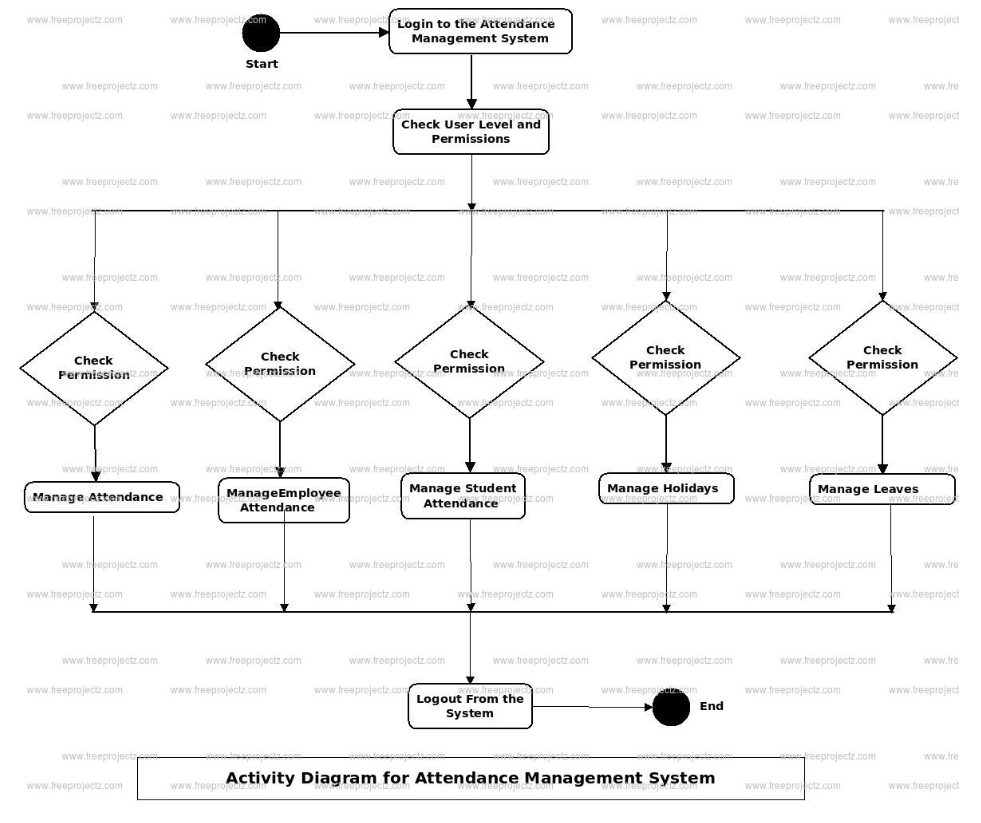
A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object. The above shown diagram represents the relation between the orders placed and the checking of the availability in the inventory system.

1. **STATE CHART DIAGRAM**



State diagrams mainly depict states and transitions. States are represented with rectangles with rounded corners that are labelled with the name of the state. Transitions are marked with arrows that flow from one state to another, showing how the states change. We have shown how the inventory will be checked after placing the order with updating product status and supervise loading process being the decision-making steps.

1. **ACTIVITY DIAGRAM**

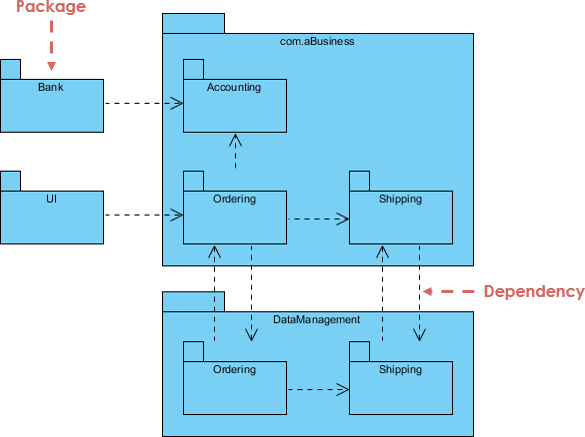
****

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent.

The purpose of an activity diagram can be described as −

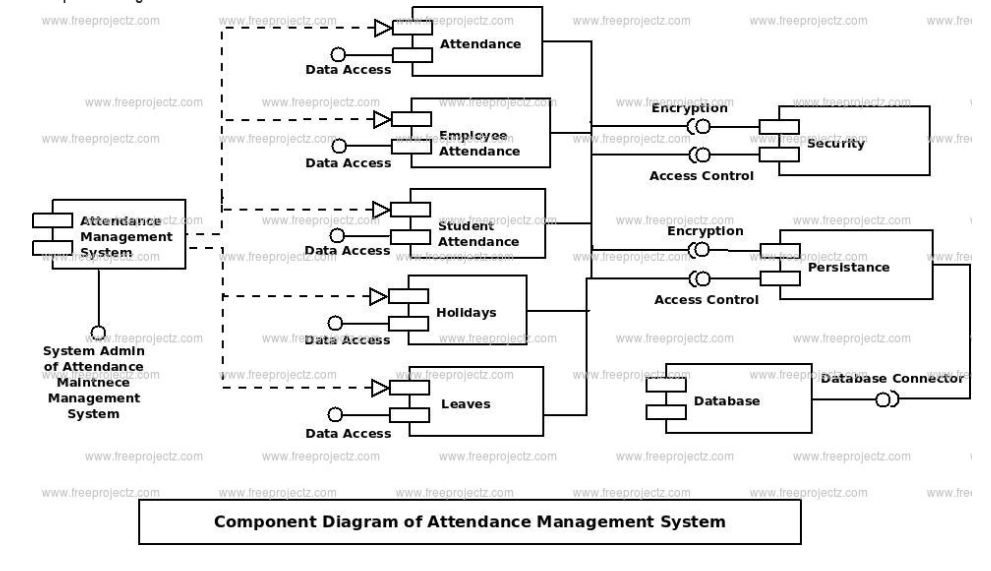
* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

1. **PACKAGE DIAGRAM**



Package diagrams are structural diagrams used to show the organization and arrangement of various model elements in the form of packages. A package is a grouping of related [UML](https://www.lucidchart.com/pages/what-is-UML-unified-modeling-language) [elements](https://www.lucidchart.com/pages/what-is-UML-unified-modeling-language), such as diagrams, documents, classes, or even other packages. Each element is nested within the package, which is depicted as a file folder within the diagram, then arranged hierarchically within the diagram. Package diagrams are most commonly used to provide a visual organization of the layered architecture within any UML classifier, such as a software system.

1. **COMPONENT DIAGRAM**

****

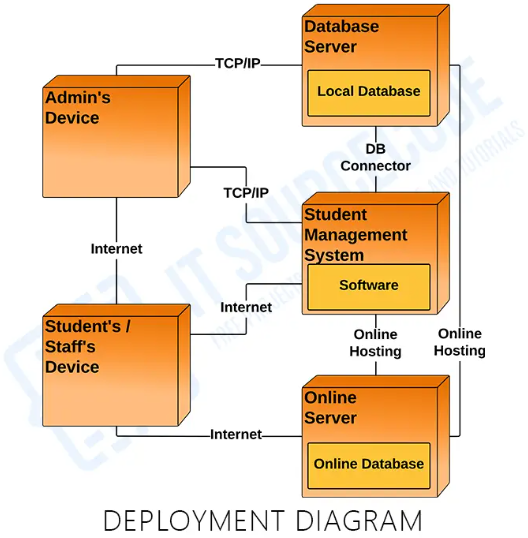
Component diagrams are different in terms of nature and behavior. Component diagrams are used to model the physical aspects of a system. Now the question is, what are these physical aspects? Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

The purpose of the component diagram can be summarized as −

* Visualize the components of a system.
* Construct executables by using forward and reverse engineering.
* Describe the organization and relationships of the components.

1. **DEPLOYMENT DIAGRAM**

****

Deployment diagrams are used to visualize the topology of the physical components of a system, where the software components are deployed.

Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

The purpose of deployment diagrams can be described as −

* Visualize the hardware topology of a system.
* Describe the hardware components used to deploy software components.
* Describe the runtime processing nodes.

**CODE/OUPUT**

#include <iostream>

#include <string>

#include <vector>

using namespace std;

class Employee

{

private:

    string name;

    int id;

    bool present;

public:

    Employee(string name, int id)

    {

        this->name = name;

        this->id = id;

        present = false;

    }

    string getName()

    {

        return name;

    }

    int getId()

    {

        return id;

    }

    bool isPresent()

    {

        return present;

    }

    void markPresent()

    {

        present = true;

    }

    void markAbsent()

    {

        present = false;

    }

};

class AttendanceManager

{

private:

    vector<Employee> employees;

public:

    void addEmployee(Employee employee)

    {

        employees.push\_back(employee);

    }

    void markAttendance(int id, bool present)

    {

        for (int i = 0; i < employees.size(); i++)

        {

            if (employees[i].getId() == id)

            {

                if (present)

                {

                    employees[i].markPresent();

                }

                else

                {

                    employees[i].markAbsent();

                }

                return;

            }

        }

        cout << "Employee with ID " << id << " not found." << endl;

    }

    void printAttendance()

    {

        cout << "Attendance Report:" << endl;

        for (int i = 0; i < employees.size(); i++)

        {

            cout << employees[i].getName() << " (ID: " << employees[i].getId() << ") - ";

            if (employees[i].isPresent())

            {

                cout << "Present" << endl;

            }

            else

            {

                cout << "Absent" << endl;

            }

        }

    }

};

int main()

{

    AttendanceManager manager;

    Employee emp1("John Doe", 1001);

    Employee emp2("Jane Smith", 1002);

    manager.addEmployee(emp1);

    manager.addEmployee(emp2);

    manager.markAttendance(1001, true);

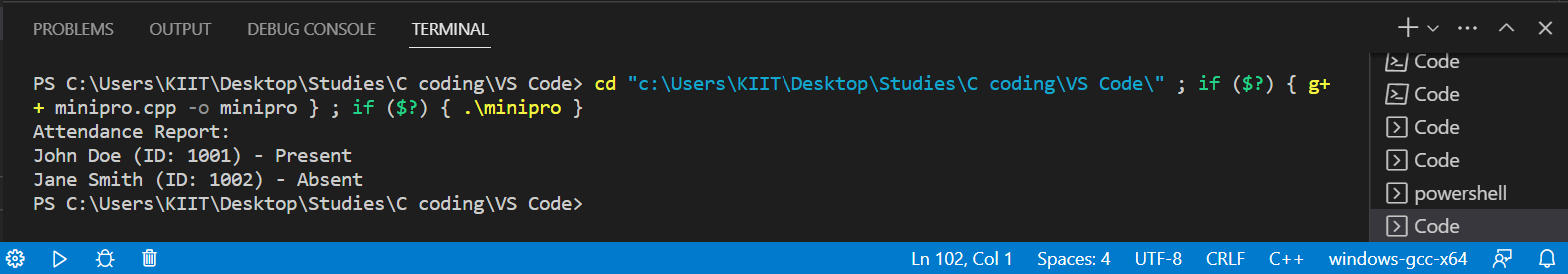
    manager.markAttendance(1002, false);

    manager.printAttendance();

    return 0;

}

**OUTPUT:**

****

**CONCLUSION**

The given problem was solved using C++ programming language. The program was tested and the desired result was computed.

**REFERENCES**

* [www.google.com](http://www.google.com)
* [www.tutorialspoint.com](http://www.tutorialspoint.com)
* [www.geeksforgeeks.com](http://www.geeksforgeeks.com)