WORKER'S ACTIVITIES

1.REQUIREMENTS:

Description

Describe about my project

Requirements

High Level Requirments

Features of my project

Low Level Requirments
Commands or Functions
Linkage of High Level to Low Level
SWOT
4W's & 1H

State Of Research:

Abstract:

In this world of growing technologies everything has been computerized, large number of work opportunities the Human workforce has increased.Thus there is a need of a system which can handle the data of such a large number of Employees in an organization. This Project simplifies the task of Worker's maintain Records of its user Friendly nature.

Features of This Project:

This project will allow admin to add New Employees after proper authentication. Admin can also add new departments and posts.

It can allocate all personal details of employees such as:

* Date Of Birth
* Full Name
* Educational Background
* Skill Sets
* Work Experience
* Current and Past Projects

2.Architecture:

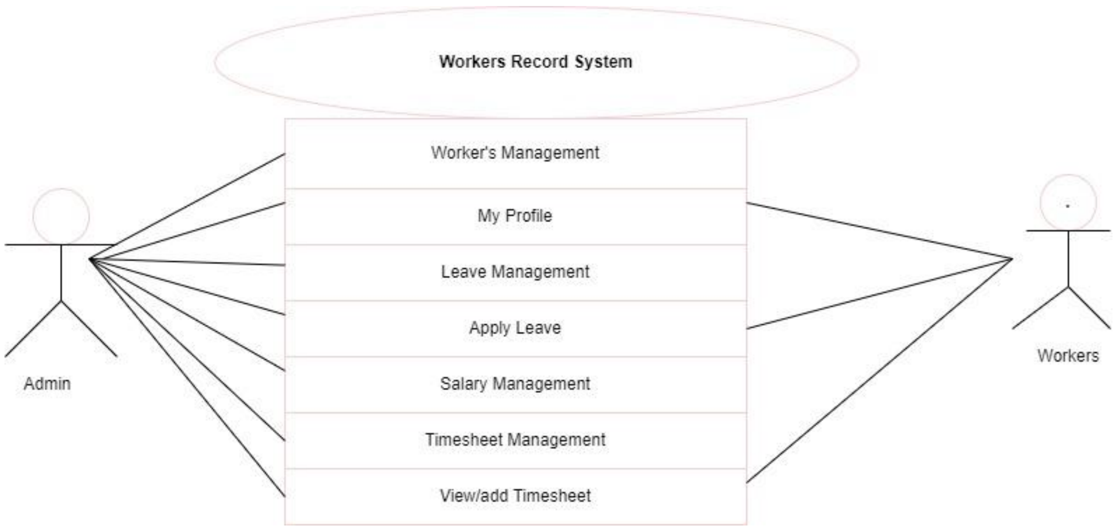
Design

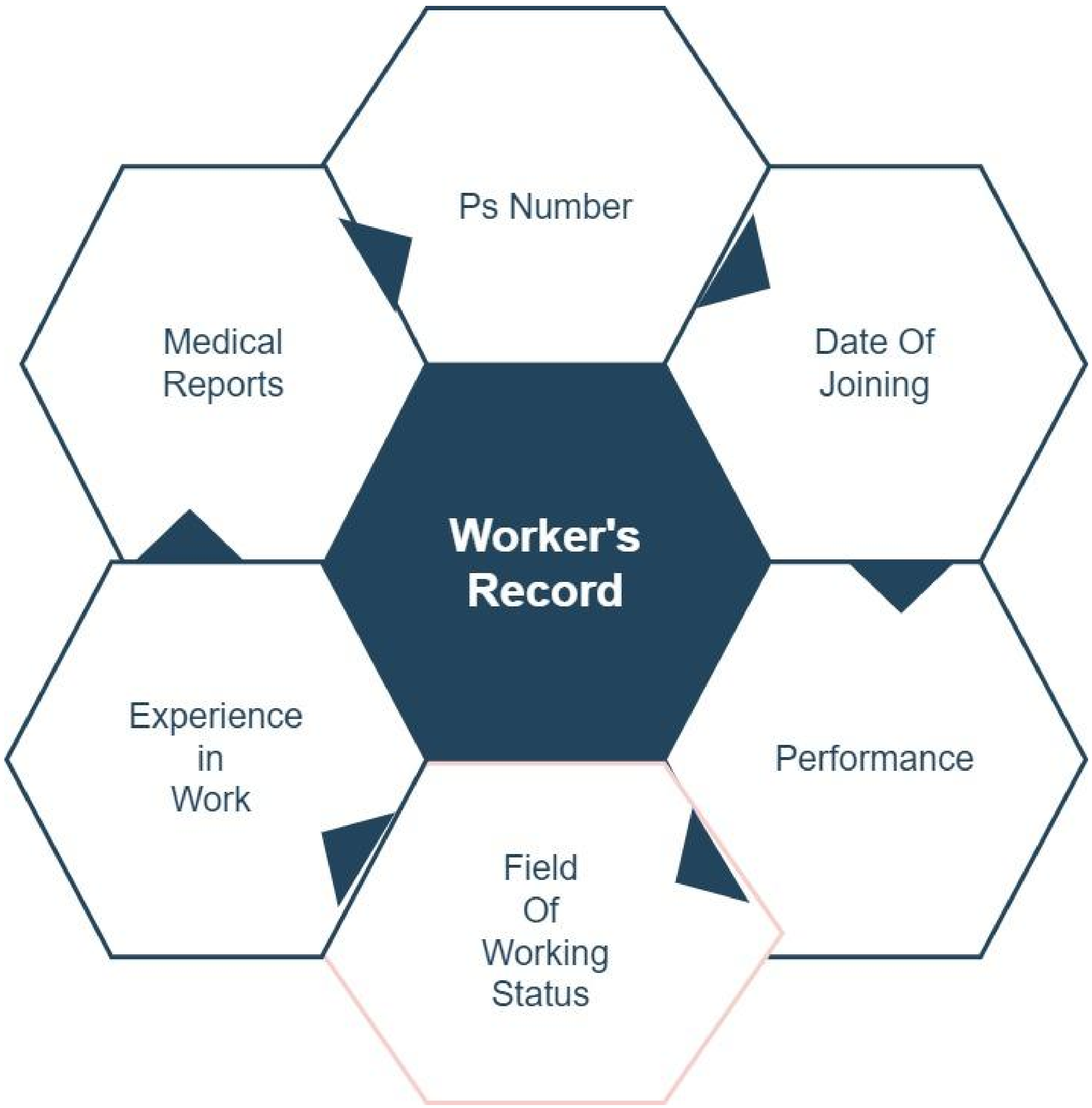
* Structural
* Behavioural
* Flowcharts
* Use Case diagrams
* Retirement Plans

TOOLS:

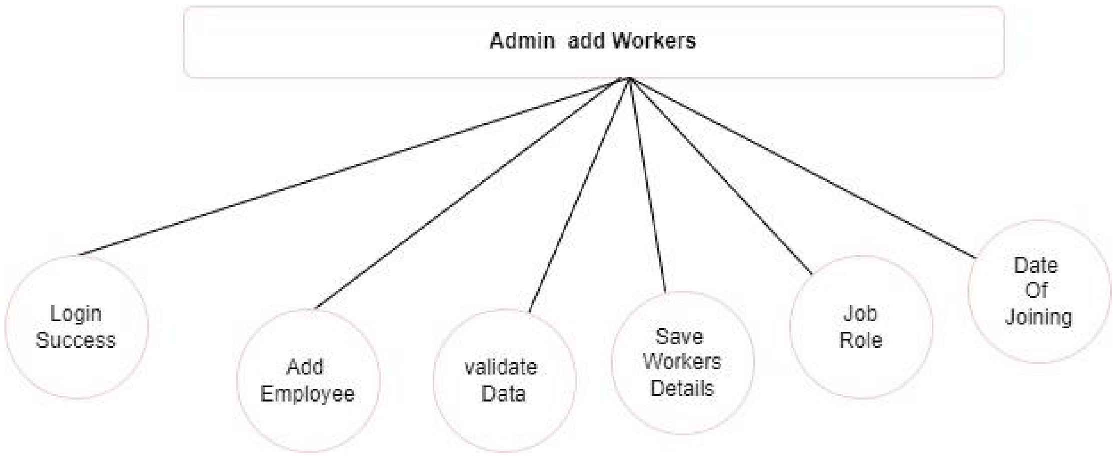
* Draw.io
* UML Diagram

Block Diagram:


Workers Record System
Worker's Management
My Profile
Leave Management
Apply Leave
Salary Management
Timesheet Management
View/add Timesheet
Admin
Workers


Ps Number
Medical Reports
Date Of Joining
Worker's Record
Experience in Work
Performance
Field Of Working Status

Structural Diagram:



Implementation:

Sample_Code:

```
#include <stdio.h> ///for input output functions like printf, scanf
#include <stdlib.h>
#include <conio.h>
#include <windows.h> ///for windows related functions (not important)
#include <string.h>  ///string operations

/** List of Global Variable */
COORD coord = {0,0}; /// top-left corner of window

/**
    function : gotoxy
    @param input: x and y coordinates
    @param output: moves the cursor in specified position of console
*/
void gotoxy(int x,int y)
{
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),coord);
}

/** Main function started */

int main()
{
    FILE *fp, *ft; /// file pointers
    char another, choice;

    /** structure that represent a employee */
    struct emp
    {
        char name[40]; ///name of employee
        int age; /// age of employee
        float bs; /// basic salary of employee
    };
```

```
struct emp e; /// structure variable creation

char empname[40]; /// string to store name of the employee

long int recsize; /// size of each record of employee

/** open the file in binary read and write mode
* if the file EMP.DAT already exists then it open that file in read write mode
* if the file doesn't exit it simply create a new copy
*/
fp = fopen("EMP.DAT","rb+");
if(fp == NULL)
{
   fp = fopen("EMP.DAT","wb+");
   if(fp == NULL)
   {
      printf("Connot open file");
      exit(1);
   }
}

/// sizeo of each record i.e. size of structure variable e
recsize = sizeof(e);

/// infinite loop continues untile the break statement encounter
while(1)
{
   system("cls"); ///clear the console window
   gotoxy(30,10); /// move the cursor to postion 30, 10 from top-left corner
   printf("1. Add Record"); /// option for add record
   gotoxy(30,12);
   printf("2. List Records"); /// option for showing existing record
   gotoxy(30,14);
   printf("3. Modify Records"); /// option for editing record
   gotoxy(30,16);
   printf("4. Delete Records"); /// option for deleting record
   gotoxy(30,18);
   printf("5. Exit"); /// exit from the program
   gotoxy(30,20);
   printf("Your Choice: "); /// enter the choice 1, 2, 3, 4, 5
   fflush(stdin); /// flush the input buffer
   choice  = getche(); /// get the input from keyboard
   switch(choice)
   {
   case '1':  /// if user press 1
      system("cls");
      fseek(fp,0,SEEK_END); /// search the file and move cursor to end of the file
      /// here 0 indicates moving 0 distance from the end of the file

      another = 'y';
      while(another == 'y')  /// if user want to add another record
      {
         printf("\nEnter name: ");
         scanf("%s",e.name);
         printf("\nEnter age: ");
         scanf("%d", &e.age);
```

```
                printf("\nEnter basic salary: ");
                scanf("%f", &e.bs);

                fwrite(&e,recsize,1,fp); /// write the record in the file

                printf("\nAdd another record(y/n) ");
                fflush(stdin);
                another = getche();
            }
            break;
        case '2':
            system("cls");
            rewind(fp); ///this moves file cursor to start of the file
            while(fread(&e,recsize,1,fp)==1)  /// read the file and fetch the record one record per fetch
            {
                printf("\n%s %d %.2f",e.name,e.age,e.bs); /// print the name, age and basic salary
            }
            getch();
            break;

        case '3':  /// if user press 3 then do editing existing record
            system("cls");
            another = 'y';
            while(another == 'y')
            {
                printf("Enter the employee name to modify: ");
                scanf("%s", empname);
                rewind(fp);
                while(fread(&e,recsize,1,fp)==1)  /// fetch all record from file
                {
                    if(strcmp(e.name,empname) == 0)  ///if entered name matches with that in file
                    {
                        printf("\nEnter new name,age and bs: ");
                        scanf("%s%d%f",e.name,&e.age,&e.bs);
                        fseek(fp,-recsize,SEEK_CUR); /// move the cursor 1 step back from current
position
                        fwrite(&e,recsize,1,fp); /// override the record
                        break;
                    }
                }
                printf("\nModify another record(y/n)");
                fflush(stdin);
                another = getche();
            }
            break;
        case '4':
            system("cls");
            another = 'y';
            while(another == 'y')
            {
                printf("\nEnter name of employee to delete: ");
                scanf("%s",empname);
                ft = fopen("Temp.dat","wb");  /// create a intermediate file for temporary storage
                rewind(fp); /// move record to starting of file
                while(fread(&e,recsize,1,fp) == 1)  /// read all records from file
                {
                    if(strcmp(e.name,empname) != 0)  /// if the entered record match
                    {
                        fwrite(&e,recsize,1,ft); /// move all records except the one that is to be deleted to
temp file
                    }
                }
                fclose(fp);
                fclose(ft);
                remove("EMP.DAT"); /// remove the orginal file
                rename("Temp.dat","EMP.DAT"); /// rename the temp file to original file name
                fp = fopen("EMP.DAT", "rb+");
                printf("Delete another record(y/n)");
                fflush(stdin);
                another = getche();
            }
            break;
        case '5':
            fclose(fp);  /// close the file
            exit(0); /// exit from the program
        }
      }
      return 0;
}
```

TestPlan

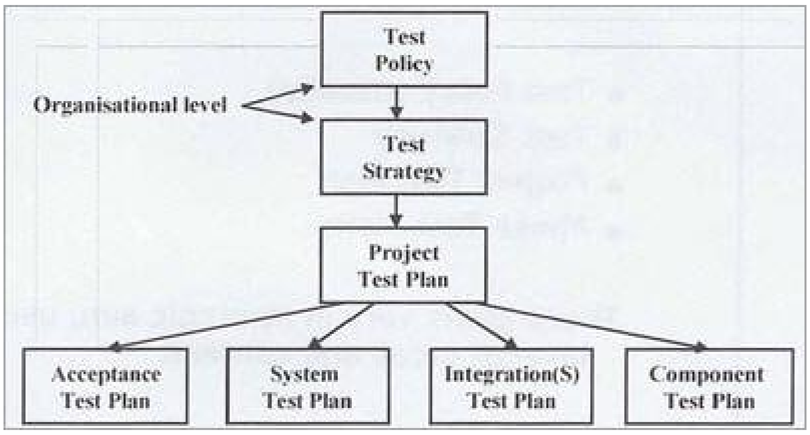


Table1:

High Level Test Plan

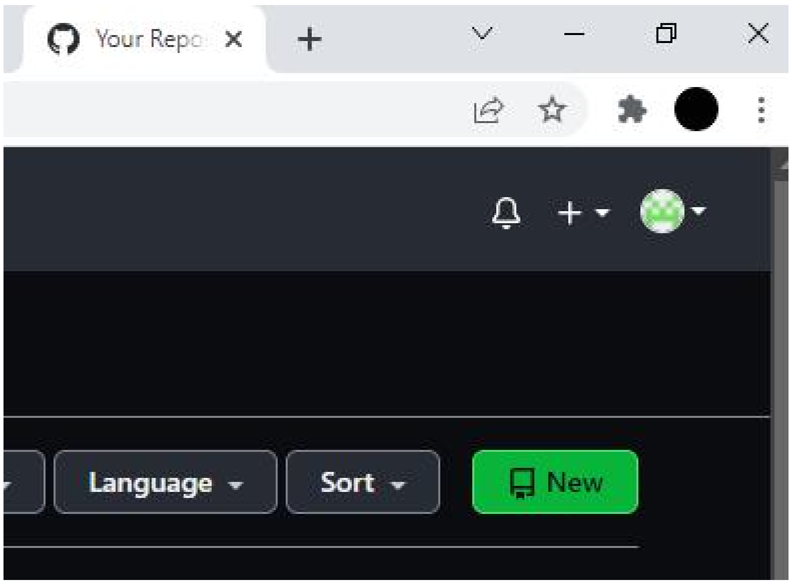
User ID	Description	Category	Status
HLTP1	Admin able to add new Worker's Record	Technical	Implemented
HLTP2	Admin able to Collect Worker'S Data	Technical	Implemented
HLTP3	Admin able to Store a Worker's data	Technical	Implemented

Low Level Test Plan

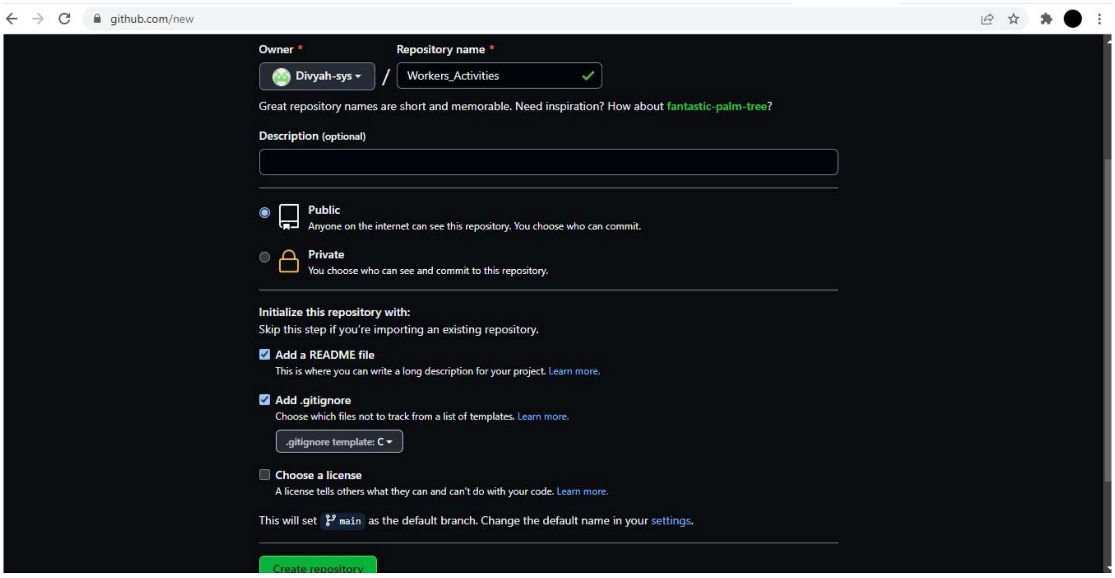
User ID	Description	Category	Status
LLTP1	New Record ID Unique shall be added	HRTP1	Future
LLTP2	ID are possible to visible Through Online Mode	HRTP1	Future

Images:

NEW REPOSITIORY:



CREATE NEW REPOSITIORY:



Owner *

Divyah-sys /

Repository name *

Workers_Activities

Great repository names are short and memorable. Need inspiration? How about fantastic-palm-tree?

Description (optional)

(●) Public
Anyone on the internet can see this repository. You choose who can commit.

( ) Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

[✓] Add a README file
This is where you can write a long description for your project. Learn more.

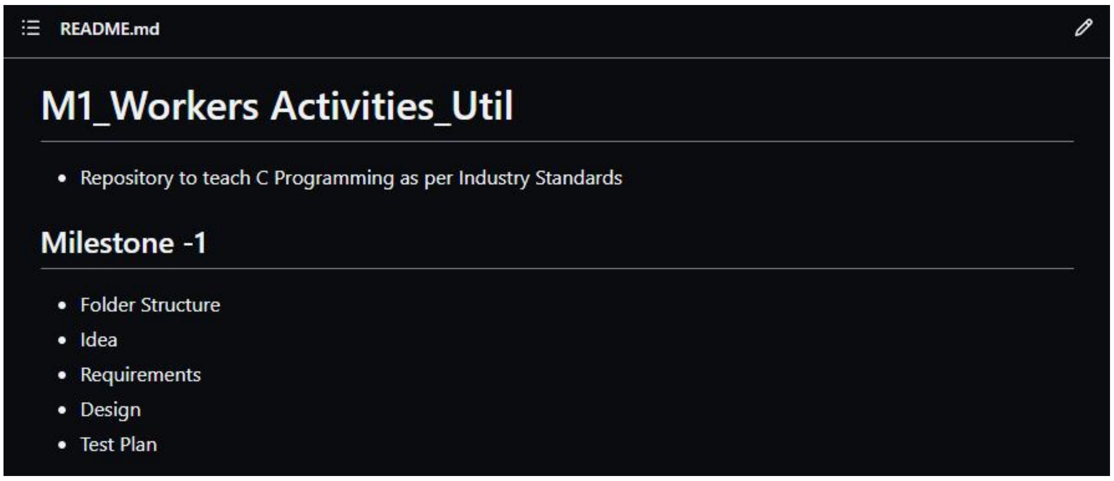
[✓] Add .gitignore
Choose which files not to track from a list of templates. Learn more.

.gitignore template: C

[ ] Choose a license
A license tells others what they can and can't do with your code. Learn more.

This will set `main` as the default branch. Change the default name in your settings.

Create repository



README.md

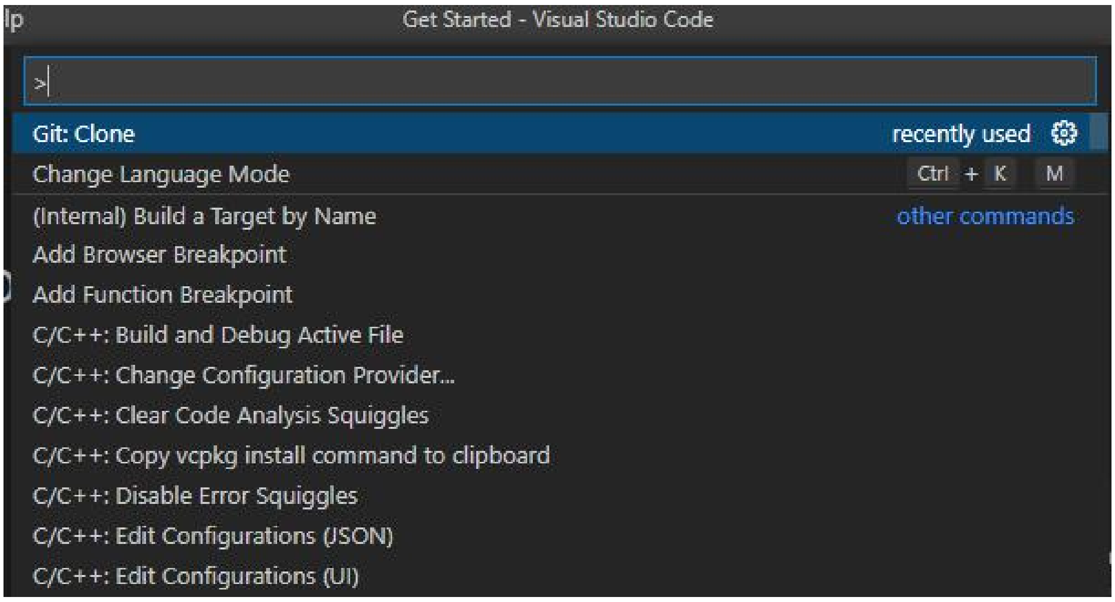
M1_Workers Activities_Util

- Repository to teach C Programming as per Industry Standards

Milestone -1

- Folder Structure
- Idea
- Requirements
- Design
- Test Plan

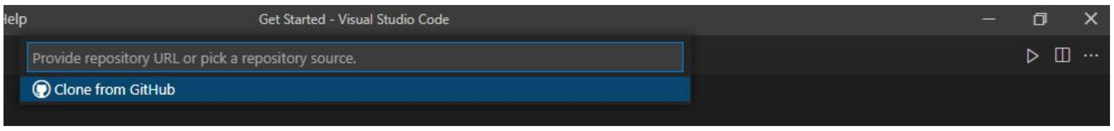
CLONE:



lp   Get Started - Visual Studio Code

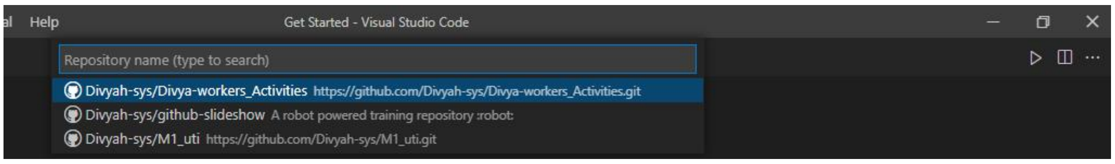
```
>|
Git: Clone                                              recently used
Change Language Mode                                    Ctrl + K  M
(Internal) Build a Target by Name                       other commands
Add Browser Breakpoint
Add Function Breakpoint
C/C++: Build and Debug Active File
C/C++: Change Configuration Provider...
C/C++: Clear Code Analysis Squiggles
C/C++: Copy vcpkg install command to clipboard
C/C++: Disable Error Squiggles
C/C++: Edit Configurations (JSON)
C/C++: Edit Configurations (UI)
```

CLONE WITH GITHUB:



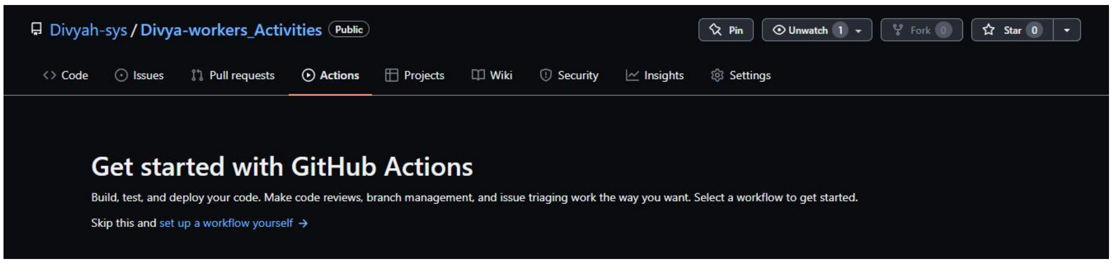
lelp   Get Started - Visual Studio Code

```
Provide repository URL or pick a repository source.
Clone from GitHub
```



al   Help   Get Started - Visual Studio Code

```
Repository name (type to search)
Divyah-sys/Divya-workers_Activities  https://github.com/Divyah-sys/Divya-workers_Activities.git
Divyah-sys/github-slideshow  A robot powered training repository :robot:
Divyah-sys/M1_uti  https://github.com/Divyah-sys/M1_uti.git
```

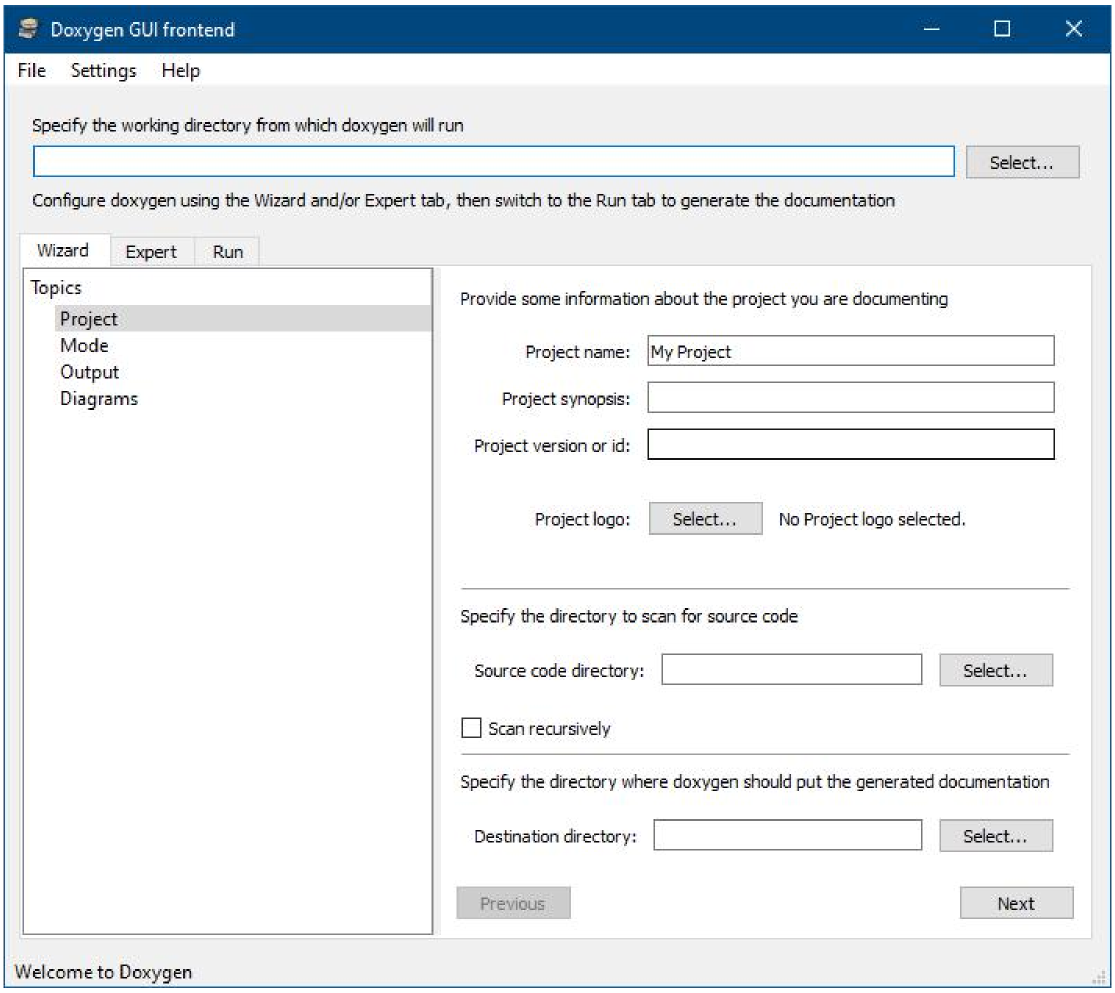

Divyah-sys / Divya-workers_Activities Public

Get started with GitHub Actions

Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow to get started.

Skip this and set up a workflow yourself →

DOXYEN:


Doxygen GUI frontend

Specify the working directory from which doxygen will run

Configure doxygen using the Wizard and/or Expert tab, then switch to the Run tab to generate the documentation

Provide some information about the project you are documenting

Project name: My Project

Project synopsis:

Project version or id:

Project logo: Select... No Project logo selected.

Specify the directory to scan for source code

Source code directory:

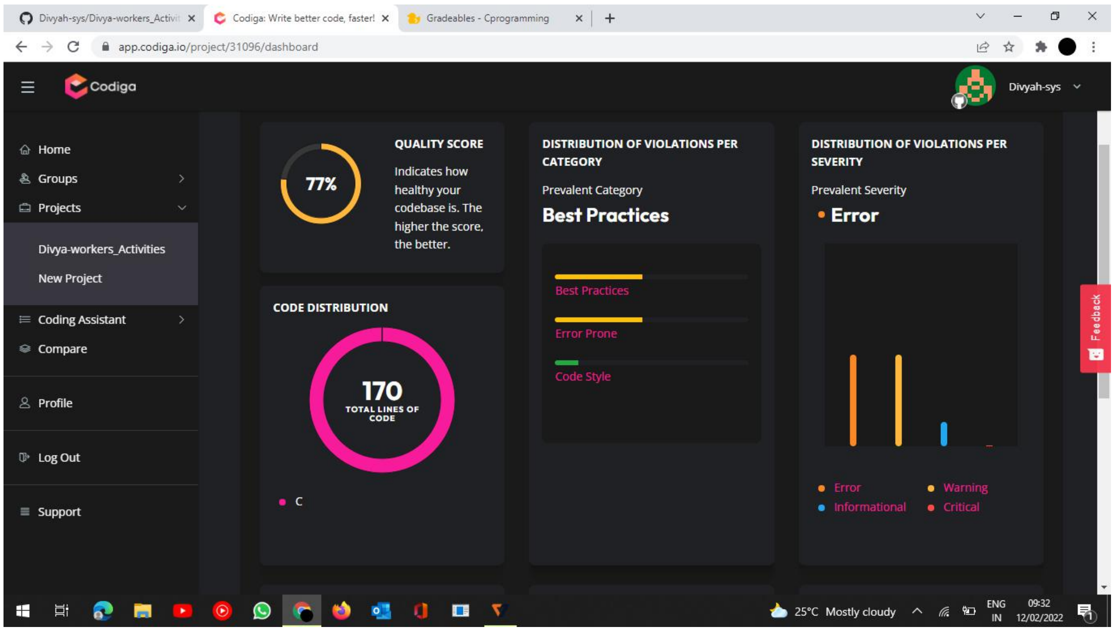
Scan recursively

Specify the directory where doxygen should put the generated documentation

Destination directory:

Welcome to Doxygen

Badges:



QUALITY SCORE

77%

Indicates how healthy your codebase is. The higher the score, the better.

DISTRIBUTION OF VIOLATIONS PER CATEGORY

Prevalent Category

Best Practices

Best Practices

Error Prone

Code Style

DISTRIBUTION OF VIOLATIONS PER SEVERITY

Prevalent Severity

Error

● Error ● Warning

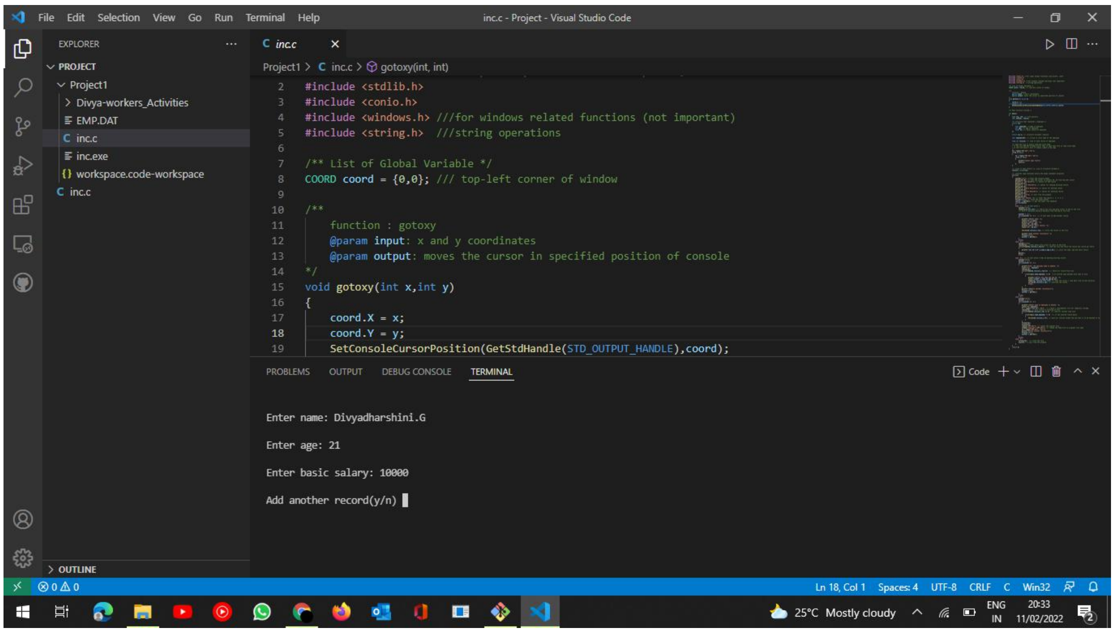
● Informational ● Critical

CODE DISTRIBUTION

170 TOTAL LINES OF CODE

● C

OUTPUT:


File Edit Selection View Go Run Terminal Help
inc.c - Project - Visual Studio Code
EXPLORER
PROJECT
Project1
Divya-workers_Activities
EMP.DAT
inc.c
inc.exe
workspace.code-workspace
inc.c
OUTLINE
Project1 > inc.c > gotoxy(int, int)
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Code
Ln 18, Col 1 Spaces: 4 UTF-8 CRLF C Win32

```c
#include <stdlib.h>
#include <conio.h>
#include <windows.h> ///for windows related functions (not important)
#include <string.h>  ///string operations

/** List of Global Variable */
COORD coord = {0,0}; /// top-left corner of window

/**
    function : gotoxy
    @param input: x and y coordinates
    @param output: moves the cursor in specified position of console
*/
void gotoxy(int x,int y)
{
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),coord);
```

```
Enter name: Divyadharshini.G

Enter age: 21

Enter basic salary: 10000

Add another record(y/n)
```