Figure 1

Foundation Certificate for Higher Education

Module: DOC 333 Introduction to Programming Principles
Module Leader: Mr. Sudarshan Welihinda
Assignment Type: Individual
Submission Date: 2023.12.11

Student ID: 20231264
Student Name: Ranuga Disansa Belpa Gamage
Student Email: ranuga.20231264@iit.ac.lk

Acknowledgement

I want to express my appreciation to those who have helped complete this report.

I have been able to complete this report because of the Academic advisor Mr. Sudarshan Welihinda for their invaluable support, mentorship, and feedback and the faculty members of Doc 333 (Introduction to Programming Principles).

Additionally, I would like to thank my family members for their unwilling encouragement and help throughout the report.

Thank you.

Table of Contents

List of Figures

Introduction

THE PROBLEM

The company “XYZ” a company that undertakes large housing construction projects needs an information system to maintain details of the projects they undertake. This system should keep details of all ongoing projects, the details and the number of workers available to assign to a new project. Before undertaking a project, the company makes sure they have enough workers if not the company doesn’t undertake the project. Once the project is finished it is taken out of the ongoing projects and assigned to completed projects and the workers are released when the project is completed.

THE SOLUTION

The solution that is created is for the problem to maintain the details of projects the construction company named “XYZ” undertakes. The solution is an information system built using Python programming language.

Algorithm

The solution which is implemented by Python Programming language is stated below in the form of algorithm steps and with an explanation of how each aspect of the program functions.

PESUDOCODE

1. Start
2. # importing packages
   - IMPORT datetime
3. # initialization of variables
   - SET company_name TO “XYZ Company.”
   - SET workers TO 0
   - SET choice TO 0
   - SET all_projects TO []
   - SET completed_projects TO []
   - SET execute TO True
   - SET project_names TO []
   - SET possible_inputs TO [“ongoing”, “completed”, “on hold”]
   - SET statistics_list TO [0] * len(possible_inputs)
   - SET redirect_choice TO False
   - SET redirect_to TO None
4. define function menu(redirect,to,company_name,msg):
   - SET main_menu TO company_name + ”””
     Main Menu
     1. Add a new project to existing projects.
     2. Remove a completed project from existing projects
     3. Add new workers to available workers group
     4. Update details on ongoing projects
     5. Project Statistics
     6. Exit
     “””
   - OUTPUT “Redirecting…” if redirect is True else main_menu
   - RETURN to if redirect is True else INPUT user choice

5. define function remove_completed_projects(code_of_project,every_project,workers_tot,stats_list,complete_projects,possible_stats):
    - TRY
    - SET index_of_project TO project_names.index(code_of_project)
    - SET date_time TO datetime.datetime.now()
    - SET actual_end_date TO date_time.strftime("%m/%d/%Y")
    - SET code_of_project,clients_name,start_date,expected_end_date,number_of_workers,_,index TO every_project[index_of_project]
    - SET completed_project_details TO [code_of_project,clients_name,start_date,expected_end_date,number_of_workers,actual_end_date]
    - SET workers_tot TO workers_tot + num_of_workers
    - SET stats_list[index] TO stats_list[index] - 1
    - SET stats_list[possible_stats.index("completed")] TO stats_list[possible_stats.index("completed")] + 1
    - APPEND completed_project_details TO completed_projects
    - DELETE every_project[index_of_project]
    - DELETE project_names[index_of_project]
    - RETURN (True, "Successful removed completed projects.",workers_tot,status_list,completed_projects,every_project,project_names)
    - EXCEPT Exception as e
    - RETURN (False,e,workers_tot,status_list,completed_projects,every_project,project_names)
6. define function create_project(status_list,index,code_of_project,clients_name,start_date,expected_end_date,number_of_workers,project_status,workers_tot,every_project,stats_list,possible_stats)
    - TRY
    - SET status_list[index] TO status_list[index] + 1
    - SET project_date TO [code_of_project,clients_name,start_date,expected_end_date,number_of_workers,project_status,index]
    - IF project_status EQUALS "ongoing" and number_of_workers > workers_tot
    - RETURN (False,"There is not enough workers", workers_tot)
    - IF project_status EQUALS "ongoing"
    - SET workers_tot TO workers_tot – number_of_workers
    - APPEND code_of_project TO project_names
    - APPEND project_data to all_projects
    - RETURN (True, "Successfully created a new project", workers_tot)
    - EXCEPT Exception as e
    - RETURN (False, e, workers_tot)
7. define function update_project_details(status_list,index,previous_index,code_of_project,clients_name,sta

rt_date,expected_end_date,number_of_workers,project_status,current_workers,workers_tot,previous_project_status)

- TRY
- IF project_status EQUALS “ongoing”
- IF number_of_workers > workers_tot + (current_workers if previous_project_stautus EQUALS “ongoing” else 0)
- RETURN (False, “Workers chosen are too much”, workers_tot)
- IF project_status EQUALS “ongoing”
- SET workers_tot TO workers_tot – number_of_workers
- IF previous_project_status EQUALS “ongoing”
- SET workers_tot TO workers_tot + current_workers
- SET status_list[index] TO status_list[index] + 1
- SET status_list[previous_index] TO status_list[previous_index] – 1
- SET project_data TO [code_of_project,clients_name,start_date,expected_end_date,number_of_workers,project_status,index]
- SET index TO project_names.index(code_of_project)
- SET all_projects[index] TO project_data
- RETURN (True, “Project details updated successfully”, workers_tot)
- EXCEPT Exception as e:
- RETURN (False,e,workers_tot)

8. define function date_vertification(msg)
   - SET date TO INPUT(msg)
   - SET splitted_date TO date.split(date[2] if len(date) > 3 else “ “)
   - IF len(splitted_date) != 3
   - OUTPUT “Enter a valid format of the date…!”
   - SET month TO splitted_date[0]
   - SET date TO splitted_date[1]
   - IF month > 12
   - OUTPUT “Enter a valid month! “
   - RETURN date_vertification(msg)
   - IF date > 31
   - OUTPUT “Enter a valid date !”
   - RETURN date_vertification(msg)
   - RETURN date
9. define function project_status_vertification(msg,update_status)
   - SET project_state TO INPUT(msg).replace(“ “,””).lower()
   - IF project_state NOT IN possible_inputs:
   - OUTPUT “The entered project status is incorrect”
   - IF update_status IS True
   - SET statistics_list[possible_inputs.index(project_state)] TO statistics_list[possible_inputs.index(project_state)] + 1
   - RETURN (project_state,statistics_list,possible_inputs.index(project_state))
10. define function project_code_vertification(msg,project_codes)
    - SET project_code TO INPUT(msg)

- IF project_code IN project_codes
- OUTPUT “Project code already exists”
- RETURN project_code_vertification(msg,project_codes)
- RETURN project_code

11. define function check_if_int(msg)
    - TRY
    - RETURN int(INPUT(msg)
    - EXCEPT
    - OUTPUT “The msg entered was not an integer”
    - RETURN check_if_int(msg)
12. while execute
    - SET choice TO menu(redirect_choice,redirect_to)
    - SET redirect_choice TO False
    - SET redirect_to TO None
    - IF choice EQUALS “1”
    - OUTPUT company_name + “Add a new project”
    - SET code_of_project TO project_code_vertification(“Project Code : “, project_names)
    - IF code_of_project EQUALS “0”
    - CONTINUE
    - SET clients_name TO INPUT(“Clients Name : “)
    - SET start_date TO date_vertification(“Start Date (MM/DD/YYYY) : ”)
    - SET expected_end_date TO date_vertification(“Expected end date (MM/DD/YYYY)”)
    - SET number_of_workers TO check_if_int(“Numbers of Workers : “)
    - SET project_status, status_list, index TO project_status_vertification()
    - SET save to INPUT(“Do you want to save the project (Yes/No)”)
    - IF save.upper() EQUALS “YES”
    - SET execution_status, response_msg, workers TO create_project(status_list,index,code_of_project,clients_name,start_date,expected_end_date,number_of_workers,project_status,workers,all_projects,statistics_list,possible-Inputs)
    - OUTPUT response_msg + execution_status
    - ELSE:
    - OUTPUT “The project was *not* saved !”
    - ELSEIF choice EQUALS “2”
    - OUTPUT company_name + “Remove completed project”
    - SET code_of_project TO INPUT(“Project Code : “)
    - IF code_of_project NOT IN project_names
    - OUTPUT “The project does not exist”
    - CONTINUE
    - SET save TO INPUT(“Do you want to save the project (Yes/No) ? “)
    - IF save.upper() EQUALS “YES”
    - SET Execution_status,response_msg,workers,status_list,completed_projects,every_project

```
, project_names TO
remove_completed_projects(code_of_project,all_projects,workers,statistics_list,com
pleted_projects,possible_inputs)
-           OUTPUT response_msg + execution_status
-       ELSE
-           OUTPUT “The project was not removed”
-   ELSEIF choice EQUALS “3”
-       OUTPUT company_name + “Add new workers”
-       SET new_no_of_workers TO check_if_int(“Number Workers to Add : “)
-       IF new_no_of_workers < 0
-           OUTPUT “Workers must be more than 0”
-       IF save.upper() EQUALS “YES”
-           SET workers TO workers + new_no_of_workers
-       ELSE
-           OUTPUT “Workers were not added”
-   ELSEIF choice EQUALS “4”
-       OUTPUT company_name + “Update Project Details”
-       SET code_of_project TO INPUT(“Project Code : “)
-       IF code_of_project NOT IN project_names
-           OUTPUT “There isn’t a project with the mentioned project code…!”
-       IF code_of_project.replace(“ “,””) EQUALS “0”
-           CONTINUE
-       SET clients_name TO INPUT(“Clients Name : “)
-       SET start_date TO date_vertification(“Start Date (MM/DD/YYYY) : “)
-       SET expected_end_date TO date_vertification(“Excepected End Date
(MM/DD/YYYY) : “)
-       SET number_of_workers TO check_if_int(“Numbers of Workers : “)
-       SET project_status,status_list,index TO project_status_vertification()
-       SET save TO INPUT(“Do you want to update the project details (Yes/No)”)
-       IF save.upper() EQUALS “YES”
-           SET current_workers,previous_project_status,previous_index TO
all_projects[project_names.index(code_of_project)][4:]
-           SET execution_status,response_msg,workers TO
update_project_details(status_list,index,previous_index,code_of_project,client_name
,start_date,expected_end_date,number_of_workers,project_status,current_workers,w
orkers,previous_project_status)
-           OUTPUT response_msg + execution_status
-       ELSE
-           OUTPUT “The project was not updated”
-   ELSEIF choice EQUALS “5”
-       OUTPUT company_name + “Project Statistics”
-       FOR idx, item IN enumerate(possible_inputs)
-           OUTPUT “Number of “ + item + “projects : “ + statistics_list[idx]
-       OUTPUT “Number of Available Workers: “ + workers
-       SET add_project TO INPUT(“Do you want to add the project”)
```

- IF add_project.upper() EQUALS “YES”
- SET redirect_choice, redirect_to TO True, “1”
- ELSEIF choice EQUALS “6”
- SET execute TO False
- ELSE
- OUTPUT “Please enter a valid choice!”

VERIFICATION FUNCTIONS

date_vertification()

```
def date_verification(msg: str) -> str:
    """A function that uses recursion to make sure that the entered date is in a correct format...
    Keyword arguments:
    msg (str) -- the message that should be displayed...
    Return: A string which contains a correct date format...
    """
    date = input(msg)
    splitted_date = date.split(date[2] if len(date) > 3 else " ")
    if len(splitted_date) != 3:
        print("Enter a valid format of the date..!")
        return date_verification(msg)
    month, date, _ = splitted_date
    if int(month) > 12:
        print("Enter a valid month..!")
        return date_verification(msg)
    if int(date) > 31:
        print("Enter a valid date..!")
        return date_verification(msg)
    return date
```

Figure 2

`date_vertification()` is a function that uses recursion to make sure the entered date is in the correct format. It has 1 argument which is `msg` which is the message that is displayed to the user. A string containing the correct date format is returned. The function works by first asking the user for a date, and then the data is split by the second character (3’rd letter), for example, “12/21/2008” The second element which is “/” will be used to split and it is taken in consideration that the string may be smaller than 3 letters so if it is then an empty string will be used. Then the length of the spliced list is checked and if it is not 3 then the `date_vertification()` function calls itself (recursion). Then after that, the month and date are extracted from the list. Finally, the month and date are checked if they are higher than 12 and higher than 31 respectively, and if all the arguments are passed then the date is returned.

project_status_vertification()

```python
def project_status_verification(
    msg: str = f"Project Status ({"/".join(possible_inputs)}) : ",
    update_status: bool = False,
) -> (str, list, int):
    """An function that uses recursion to make sure that an input is enter as required
    Keyword arguments:
    msg -- The message that should be displayed to the user to get the project status input
    update_status -- Whether to update the status count
    Return: Tuple[The state enter by the user,
                  the statistic list used to track the project status count,
                  the index of the enter state
                ]
    """
    project_state = str(input(msg)).replace(" ", "").lower()
    if project_state not in possible_inputs:
        print("The entered project status is incorrect...")
        return project_status_verification()
    if update_status:
        statistics_list[possible_inputs.index(project_state)] += 1
    return (
        project_state,
        statistics_list,
        possible_inputs.index(project_state),
    )
```

Figure 3

The `project_status_vertification()` function checks whether the project status that is entered is allowed and if not it uses recursion to make sure that the user enters the allowed status. It has 2 arguments which is `msg` which is the message that should be displayed to the user and `update_status` which is a Boolean argument that if True the statistics_list is updated with the status that was entered. First, the user has displayed a message which they need to respond to then the " " (blank spaces) are replaced with "and the entire message is lowered, then it is checked if the project_state entered is not in the possible_inputs list, if it is not then the function is calling itself (recursion) and if the project_state is in the possible_input the `statistics_list` is updated and then the following is returned : (project_state, statistics_list, possible_inputs.index(project_state)) => (The project state, the statistics list that is used for the choice `5`, the index of the project state in the list possible_inputs), an example would be: ["ongoing",[2,1,5],1]

project_code_vertification()

```
def project_code_verification(msg: str, project_codes: list) -> str:
    """Project Code Verification function with the use of recursion...
    Keyword arguments:
    msg (str) -- The message that is displayed and ask the user to enter the project code
    project_codes (list) -- The list of project codes that already exists
    Return: (str) of a project code that doesnt already exist...
    """
    project_code = str(input(msg))
    if project_code in project_codes:
        print("Project Code already exists..!")
        return project_code_verification(msg, project_codes)
    return project_code
```

Figure 4

`project_code_vertification()` is a function which uses recursion to make sure the project_code entered does not exist already. The parameters are `msg` which is the message that should be displayed to the user and `project_codes` which is the list of project_codes where the function checks if the entered project code exists or not, and finally if the project code does not exist it is returned.

check_if_int()

```
def check_if_int(msg) -> int:
    try:
        return int(input(msg))
    except:
        print("The msg entered was not an integer")
        return check_if_int(msg)
```

Figure 5

`check_if_int()` function uses recursion with having 1 parameter `msg` which is what is displayed to the user then the message is displayed and the function tries to return the message by trying to convert the inputted data into an integer and if an error is caused then a message saying “The msg entered was not an integer” is displayed and then the function calls itself (recursion).

EXPLANATION

Add a new project to existing projects.

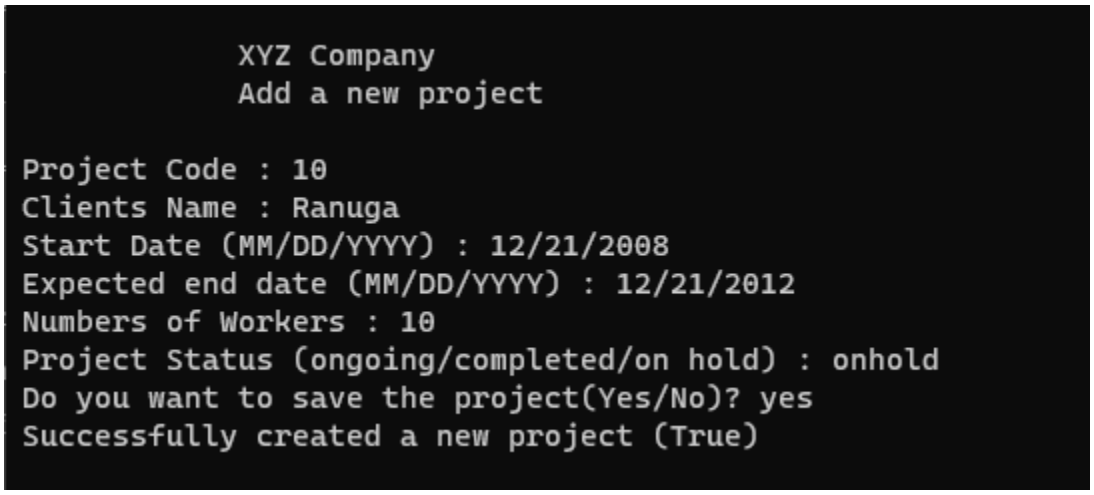


Figure 6

When the user enters the choice `1` the program asks the user for the Project Code which is directed through a function `project_code_vertification()` which makes sure that the project code is unique, for the start date and expected end date, function `date_vertification()` is used which makes sure the date entered is in the correct format of MM/DD/YYYY. When asking the user for the project status it is passed through a function `project_status_vertification()` which checks whether the status entered was either “ongoing”, “completed” or “on hold”.

Removing a Completed project from existing projects

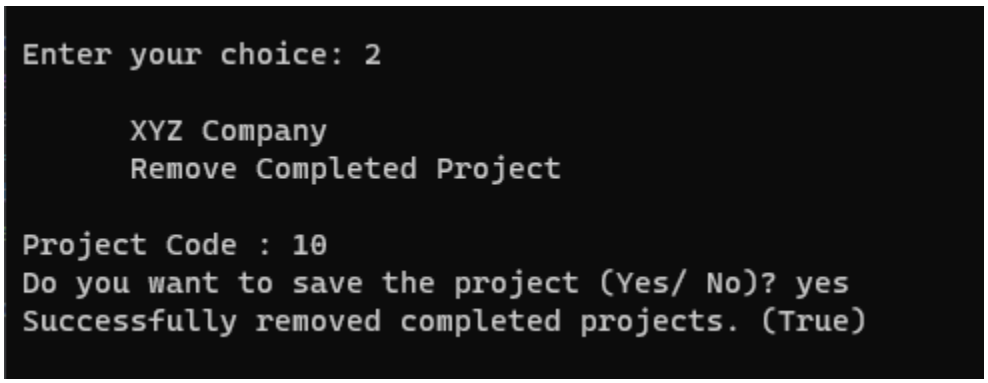


Figure 7

When the user enters the choice `2` the program asks for the project code which needs to be removed, and it makes sure that the project code exists in the list `project_names`.

Adding new workers to the available worker's group

```
Enter your choice: 3

      XYZ Company
      Add new Workers

Number Workers to Add : 100
Do you want to add ? (Yes / No) yes
Workers added successfully..!
```

Figure 8

When the user chooses the choice `3` the user is asked for the number of workers that they want to add and the function `check_if_it()` is used to make sure the user enters an integer value.

Updating Details to an ongoing project

```
Enter your choice: 4

      XYZ Company
      Update Project Details

Project Code : 10
Clients Name : Devin
Start Date (MM/DD/YYYY) : 06/16/2017
Expected end date (MM/DD/YYYY) : 06/16/2020
Numbers of Workers : 25
Project Status (ongoing/completed/on hold) : ongoing
Do you want to update the project details (Yes/No)?yes
Project details updated successfully (True)
```

Figure 9

When the user chooses the choice `4` the user is asked for the project code and other details such as Clients Name, Start Date, Expected end Date, Number of workers, and Project Status which uses functions such as `date_vertification()`, `check_if_int()` and `project_status_vertification()`, and this updates the data in the main list `all_projects`

Project Statistics

```
Enter your choice: 5

      XYZ Company
      Project Statistics

Number of ongoing projects : 1
Number of completed projects : 0
Number of onhold projects : 0
Number of available workers : 75
Do you want to add the project (Yes/No)?yes
Redirecting...

            XYZ Company
            Add a new project

Project Code : 0
```

Figure 10

When the user chooses the choice `5` it displays the project statistics such as ongoing, completed, hold projects and the available workers. It also allows the user to be redirected to the 1st choice.

Exiting the Program

```
elif choice == "6":
    print('Exiting Program...')
    execute = False
```

```
while execute:
    choice = menu(
```

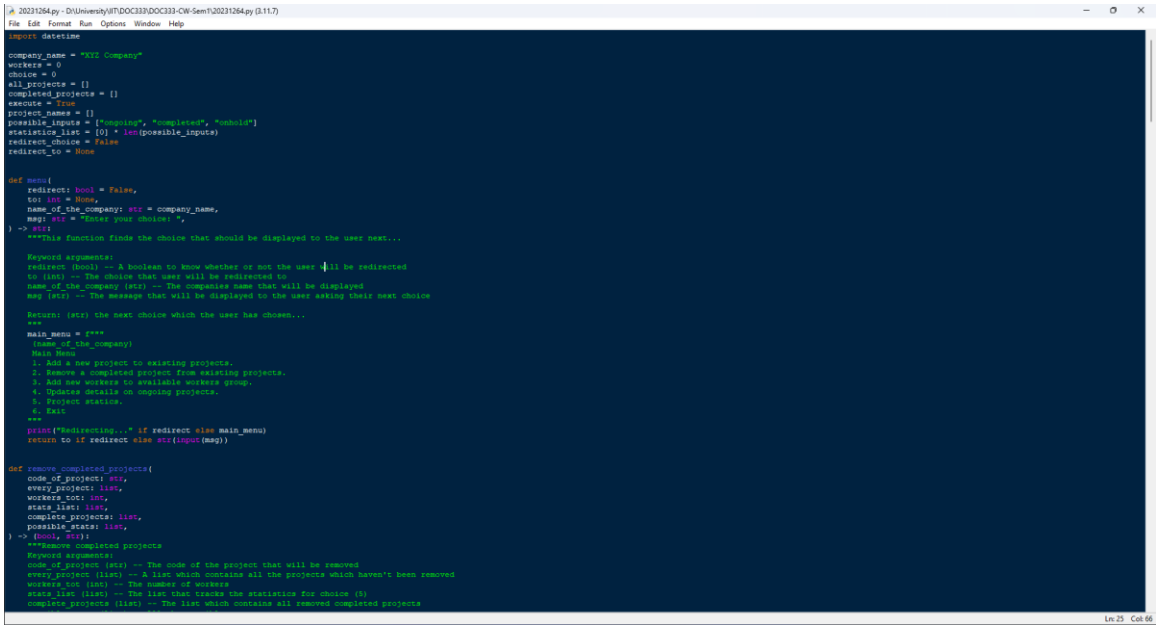
When the user chooses the choice `6` then the program exits by changing the variable `execute` which is the condition that is given to the while loop in turn looping until `execute` is False.

Assumptions

The listed below are assumptions that were made about the solution, are as listed below.

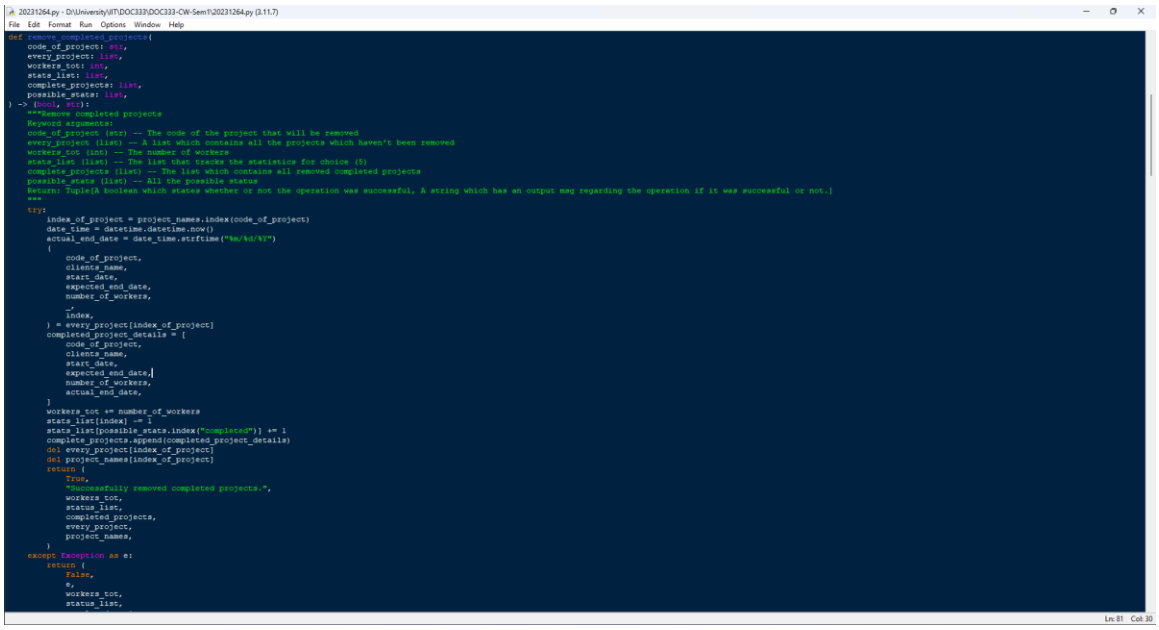
1. Unique Project Code
   It was assumed that the project code must be unique throughout all other projects with either ongoing or on-hold status.
2. When removing a project, the current date should be stored.
   It was assumed that the date on which the user removes a project needs to be stored as stated in the “DOC 333 CW Specification”, due to that reason the package “datetime” was imported to get the date on which the project is removed.
3. Only workers are assigned to ongoing projects.
   It was assumed that only ongoing projects need to be assigned workers, the other statuses do not get assigned workers when they are created, but if they are updated to ongoing then workers are assigned to those projects.
4. The date format was assumed to be (MM/DD/YYYY)
   It was assumed that the Date format that should be used is MM/DD/YYYY
5. There were 0 workers initially.
   It is assumed that there are no workers initially when the program starts so it is required for the user to add workers before adding an `ongoing` project.

Python Code



```
import datetime

company_name = "XYZ Company"
workers = 0
choice = 0
all_projects = []
completed_projects = []
execute = True
project_names = []
possible_inputs = ["ongoing", "completed", "onhold"]
statistics_list = [0] * len(possible_inputs)
redirect_choice = False
redirect_to = None


def menu(
    redirect: bool = False,
    to: int = None,
    name_of_the_company: str = company_name,
    msg: str = "Enter your choice: ",
) -> str:
    """This function finds the choice that should be displayed to the user next...

    Keyword arguments:
    redirect (bool) -- A boolean to know whether or not the user will be redirected
    to (int) -- The choice that user will be redirected to
    name_of_the_company (str) -- The companies name that will be displayed
    msg (str) -- The message that will be displayed to the user asking their next choice

    Return: (str) the next choice which the user has chosen...
    """
    main_menu = f"""
     {name_of_the_company}
     Main Menu
     1. Add a new project to existing projects.
     2. Remove a completed project from existing projects.
     3. Add new workers to available workers group.
     4. Updates details on ongoing projects.
     5. Project status.
     6. Exit
    """
    print("Redirecting..." if redirect else main_menu)
    return to if redirect else str(input(msg))


def remove_completed_projects(
    code_of_project: str,
    every_project: list,
    workers_tot: int,
    stats_list: list,
    complete_projects: list,
    possible_stats: list,
) -> (bool, str):
    """Remove completed projects
    Keyword arguments:
    code_of_project (str) -- The code of the project that will be removed
    every_project (list) -- A list which contains all the projects which haven't been removed
    workers_tot (int) -- The number of workers
    stats_list (list) -- The list that tracks the statistics for choice (5)
    complete_projects (list) -- The list which contains all removed completed projects

Ln: 25  Col: 66
```

Figure 11



```
def remove_completed_projects(
    code_of_project: str,
    every_project: list,
    workers_tot: int,
    stats_list: list,
    complete_projects: list,
    possible_stats: list,
) -> (bool, str):
    """Remove completed projects
    Keyword arguments:
    code_of_project (str) -- The code of the project that will be removed
    every_project (list) -- A list which contains all the projects which haven't been removed
    workers_tot (int) -- The number of workers
    stats_list (list) -- The list that tracks the statistics for choice (5)
    complete_projects (list) -- The list which contains all removed completed projects
    possible_stats (list) -- All the possible status
    Return: Tuple[A boolean which states whether or not the operation was successful, A string which has an output msg regarding the operation if it was successful or not.]
    """
    try:
        index_of_project = project_names.index(code_of_project)
        date_time = datetime.datetime.now()
        actual_end_date = date_time.strftime("%m/%d/%Y")
        (
            code_of_project,
            clients_name,
            start_date,
            expected_end_date,
            number_of_workers,
            _,
            index,
        ) = every_project[index_of_project]
        completed_project_details = [
            code_of_project,
            clients_name,
            start_date,
            expected_end_date,
            number_of_workers,
            actual_end_date,
        ]
        workers_tot += number_of_workers
        stats_list[index] -= 1
        stats_list[possible_stats.index("completed")] += 1
        complete_projects.append(completed_project_details)
        del every_project[index_of_project]
        del project_names[index_of_project]
        return (
            True,
            "Successfully removed completed projects.",
            workers_tot,
            status_list,
            completed_projects,
            every_project,
            project_names,
        )
    except Exception as e:
        return (
            False,
            e,
            workers_tot,
            status_list,

Ln: 81  Col: 30
```

Figure 12

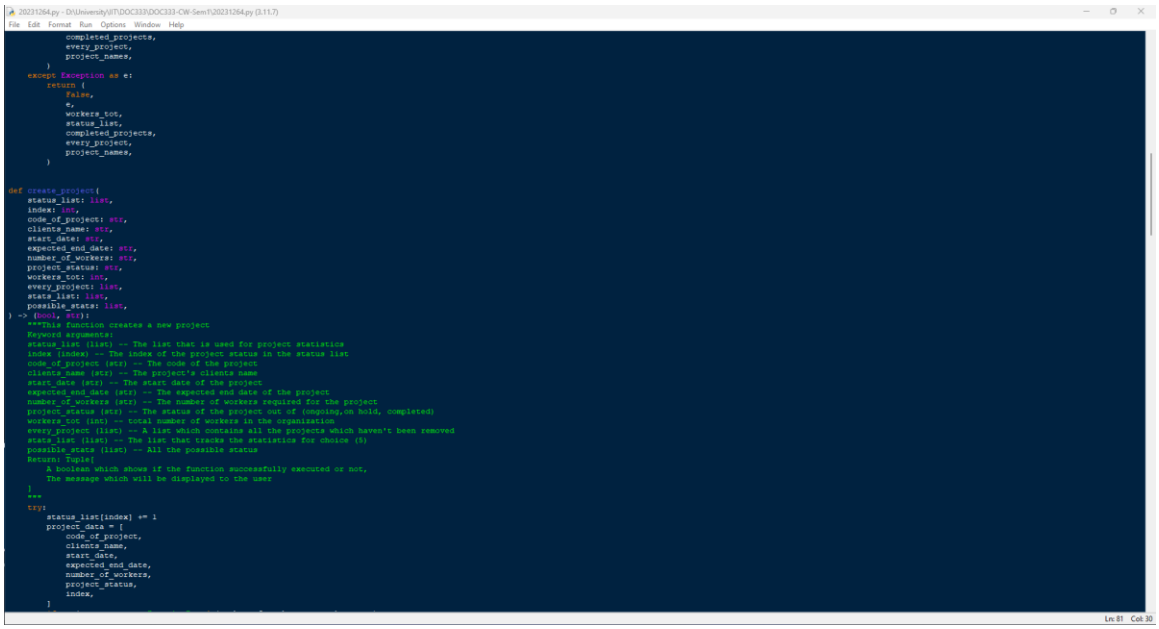


Figure 13

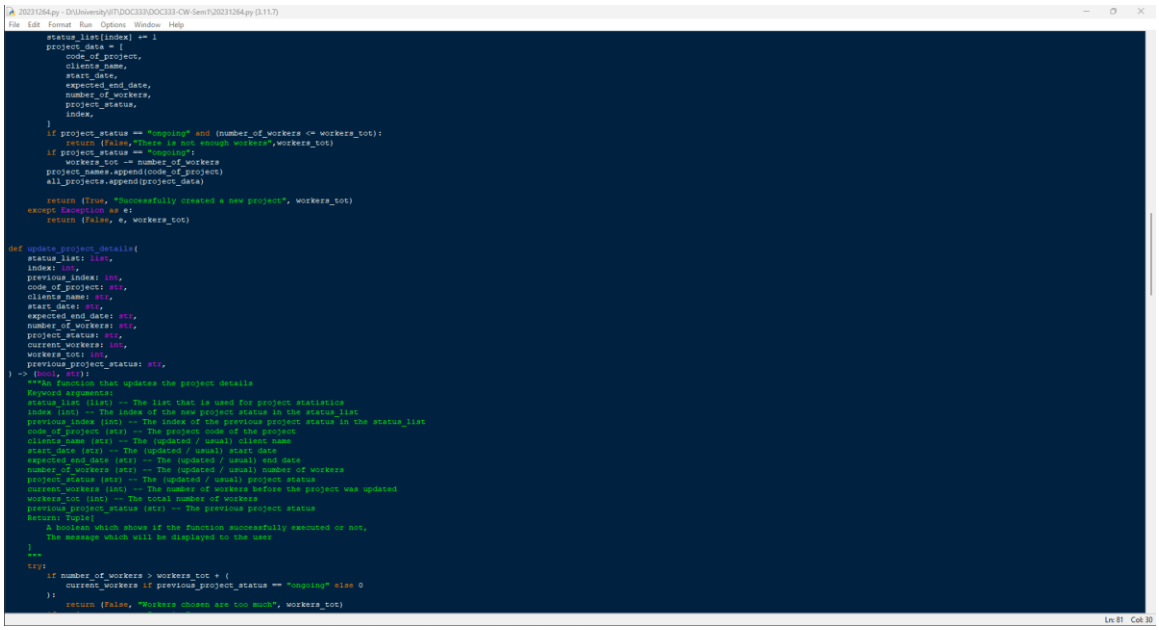


Figure 14

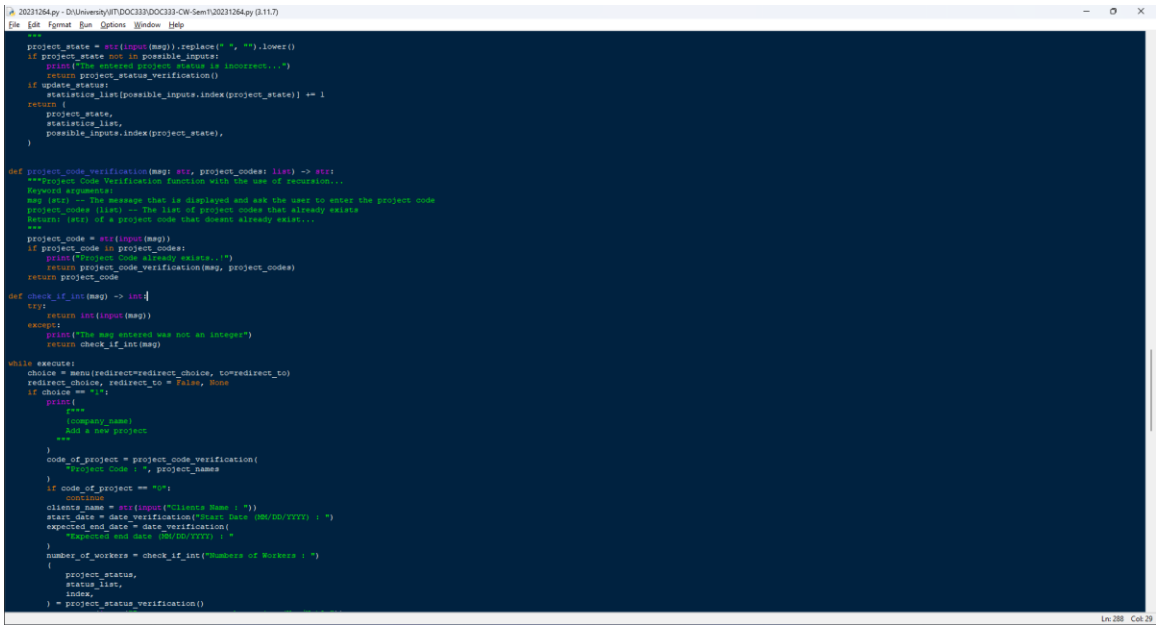

20231264.py - D:\University\IIT\DOC333\DOC333-CW-Sem1\20231264.py (3.11.7)
File Edit Format Run Options Window Help
Ln: 288 Col: 29

Figure 15

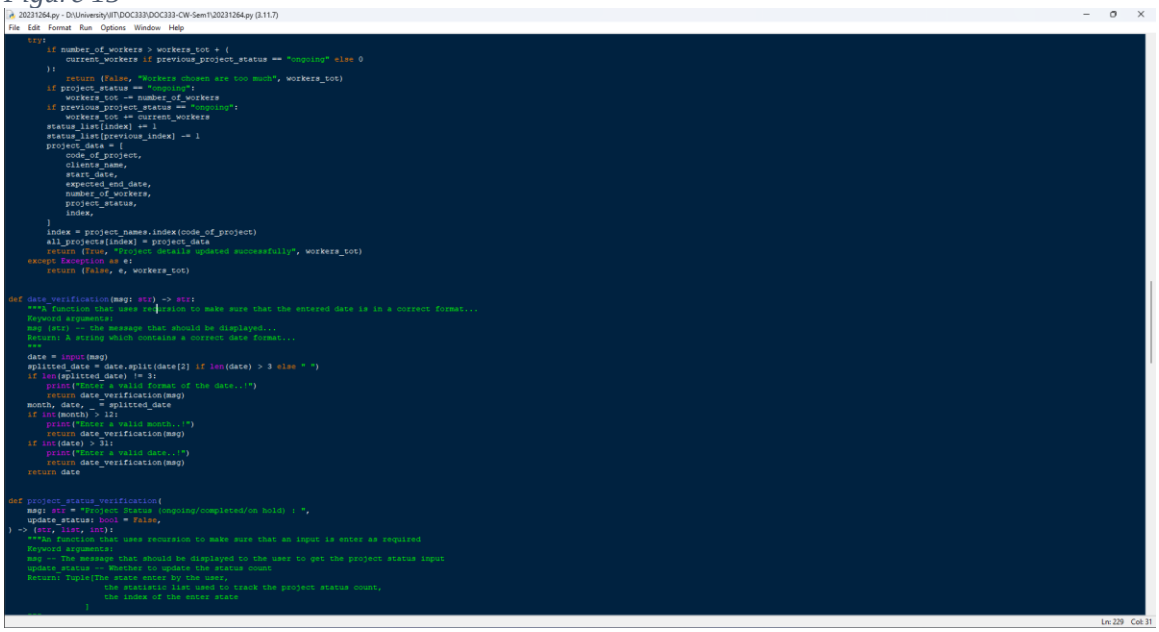
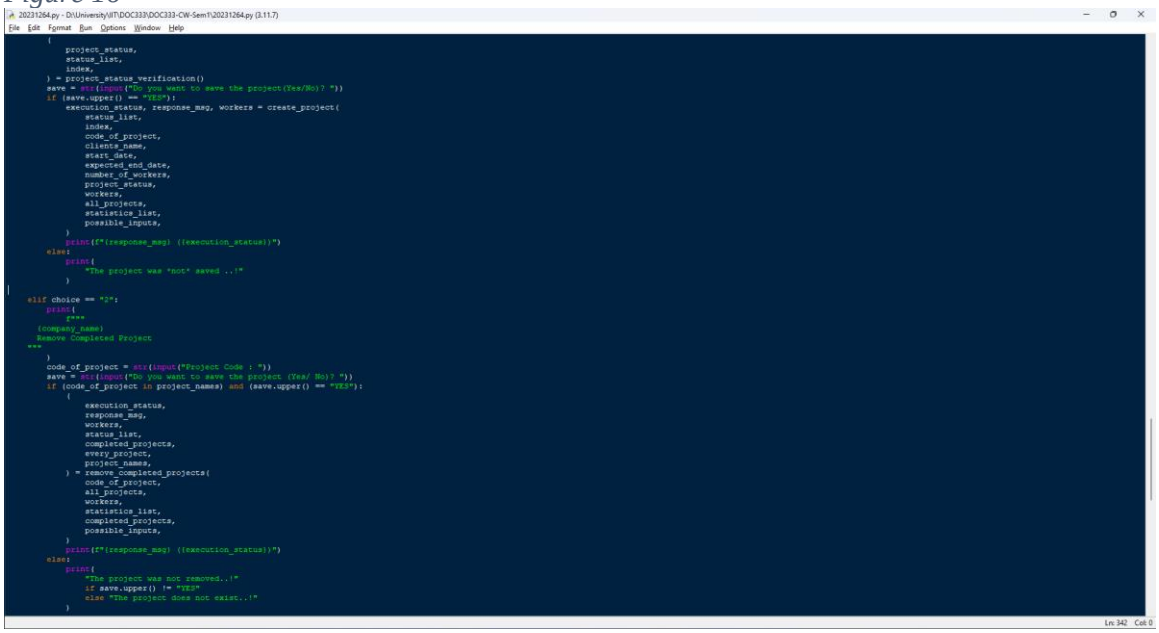
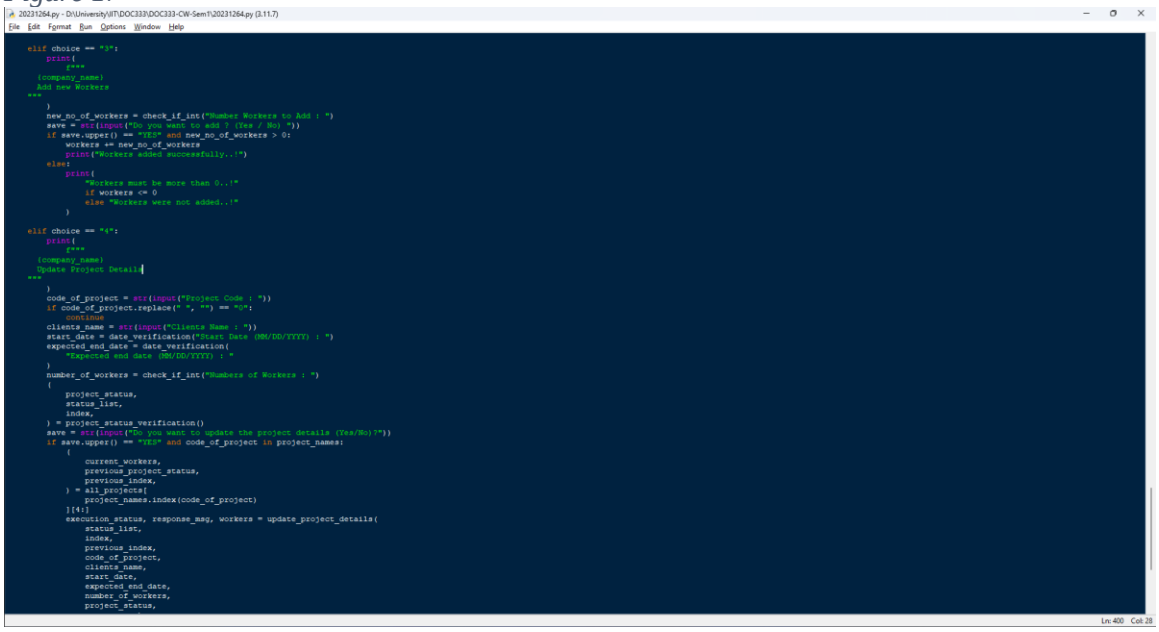

20231264.py - D:\University\IIT\DOC333\DOC333-CW-Sem1\20231264.py (3.11.7)
File Edit Format Run Options Window Help
Ln: 229 Col: 31

Figure 16



```
20231264.py - D:\University\IIT\DOC333\DOC333-CW-Sem1\20231264.py (3.11.7)
File  Edit  Format  Run  Options  Window  Help
        (
            project_status,
            status_list,
            index,
        ) = project_status_verification()
        save = str(input("Do you want to save the project(Yes/No)? "))
        if (save.upper() == "YES"):
            execution_status, response_msg, workers = create_project(
                status_list,
                index,
                code_of_project,
                clients_name,
                start_date,
                expected_end_date,
                number_of_workers,
                project_status,
                workers,
                all_projects,
                statistics_list,
                possible_inputs,
            )
            print(f"{response_msg} ({execution_status})")
        else:
            print(
                "The project was *not* saved ..!"
            )

    elif choice == "2":
        print(
            f"""
      {company_name}
      Remove Completed Project
    """
        )
        code_of_project = str(input("Project Code : "))
        save = str(input("Do you want to save the project (Yes/ No)? "))
        if (code_of_project in project_names) and (save.upper() == "YES"):
            (
                execution_status,
                response_msg,
                workers,
                status_list,
                completed_projects,
                every_project,
                project_names,
            ) = remove_completed_projects(
                code_of_project,
                all_projects,
                workers,
                statistics_list,
                completed_projects,
                possible_inputs,
            )
            print(f"{response_msg} ({execution_status})")
        else:
            print(
                "The project was not removed..!"
                if save.upper() != "YES"
                else "The project does not exist..!"
            )

Ln: 342  Col: 0
```

Figure 17



```
20231264.py - D:\University\IIT\DOC333\DOC333-CW-Sem1\20231264.py (3.11.7)
File  Edit  Format  Run  Options  Window  Help
    elif choice == "3":
        print(
            f"""
      {company_name}
      Add new Workers
    """
        )
        new_no_of_workers = check_if_int("Number Workers to Add : ")
        save = str(input("Do you want to add ? (Yes / No) "))
        if save.upper() == "YES" and new_no_of_workers > 0:
            workers += new_no_of_workers
            print("Workers added successfully..!")
        else:
            print(
                "Workers must be more than 0..!"
                if workers <= 0
                else "Workers were not added..!"
            )

    elif choice == "4":
        print(
            f"""
      {company_name}
      Update Project Details
    """
        )
        code_of_project = str(input("Project Code : "))
        if code_of_project.replace(" ", "") == "0":
            continue
        clients_name = str(input("Clients Name : "))
        start_date = date_verification("Start Date (MM/DD/YYYY) : ")
        expected_end_date = date_verification(
            "Expected end date (MM/DD/YYYY) : "
        )
        number_of_workers = check_if_int("Numbers of Workers : ")
        (
            project_status,
            status_list,
            index,
        ) = project_status_verification()
        save = str(input("Do you want to update the project details (Yes/No)?"))
        if save.upper() == "YES" and code_of_project in project_names:
            (
                current_workers,
                previous_project_status,
                previous_index,
            ) = all_projects[
                project_names.index(code_of_project)
            ][4:]
            execution_status, response_msg, workers = update_project_details(
                status_list,
                index,
                previous_index,
                code_of_project,
                clients_name,
                start_date,
                expected_end_date,
                number_of_workers,
                project_status,

Ln: 400  Col: 28
```

Figure 18

Test Cases

ADD A NEW PROJECT

Test 1

The following test cases are for the 1st choice ("Add a new project to existing projects.")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"0"	Exit and Go to the Main Menu	Exit and Go to the Main Menu	Pass

Figure 19

Test 2

The following test cases are for the 1st choice ("Add a new project to existing projects.")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"2"	Error Raised, asking the user to enter the date again…	Error Raised, asking the user to enter the date again…	Pass

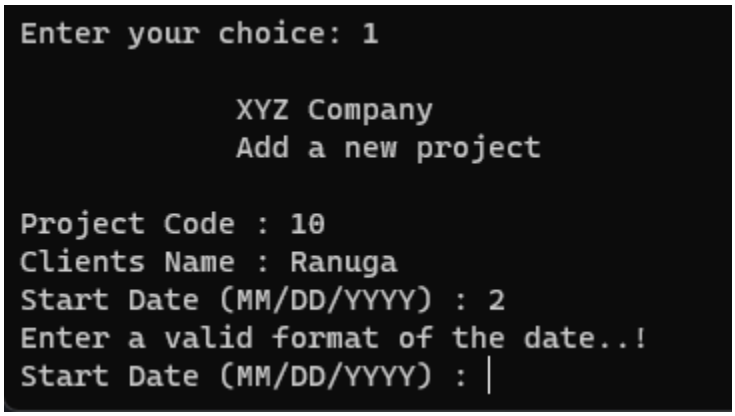


Figure 20

Test 3

The following test cases are for the 1st choice ("Add a new project to existing projects.")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"12/12/2000"	Continue	Continue	Pass
Expected End Date	"01/01/2001"	Continue	Continue	Pass
Number of Workers	"!@"	"The msg entered was not an integer"	"The msg entered was not an integer"	Pass

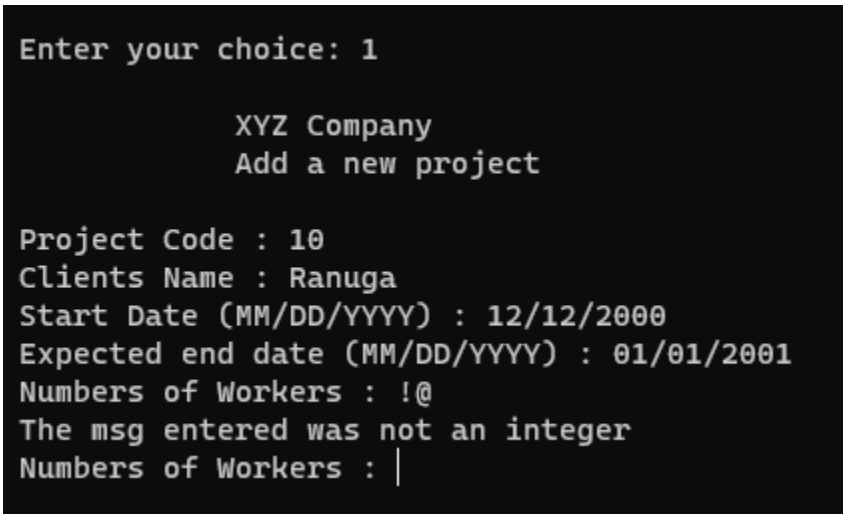


Figure 21

Test 4

The following test cases are for the 1st choice ("Add a new project to existing projects.")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"12/12/2000"	Continue	Continue	Pass
Expected End Date	"01/01/2001"	Continue	Continue	Pass
Number of Workers	10	Continue	Continue	Pass
Project Status	"ongone"	"The entered project status is incorrect..."	"The entered project status is incorrect..."	Pass

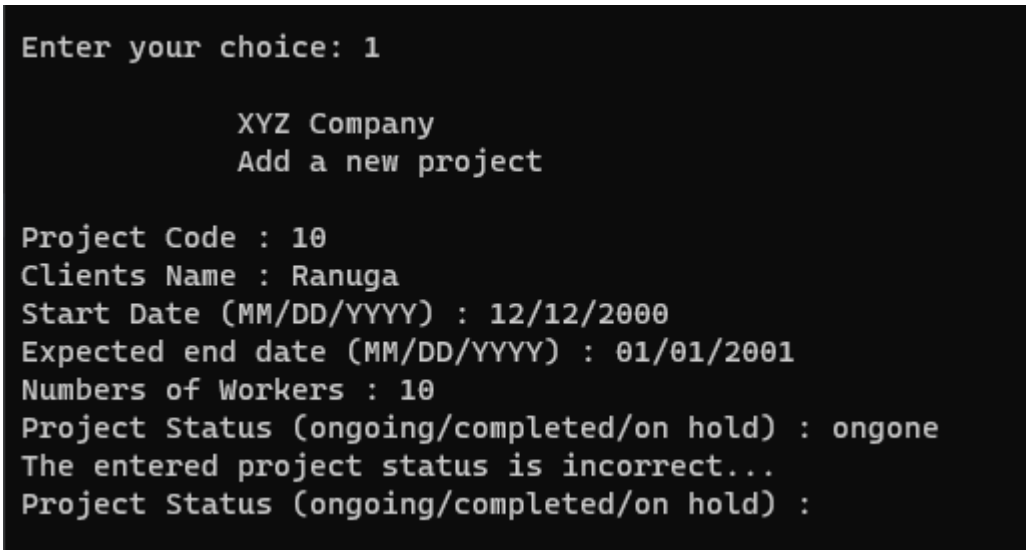


Figure 22

Test 5

The following test cases are for the 1st choice ("Add a new project to existing projects.")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"12/12/2000"	Continue	Continue	Pass
Expected End Date	"01/01/2001"	Continue	Continue	Pass
Number of Workers	10	Continue	Continue	Pass
Project Status	"on hold"	Continue	Continue	Pass
Save	"No"	"The project was *not* saved ..!"	"The project was *not* saved ..!"	Pass

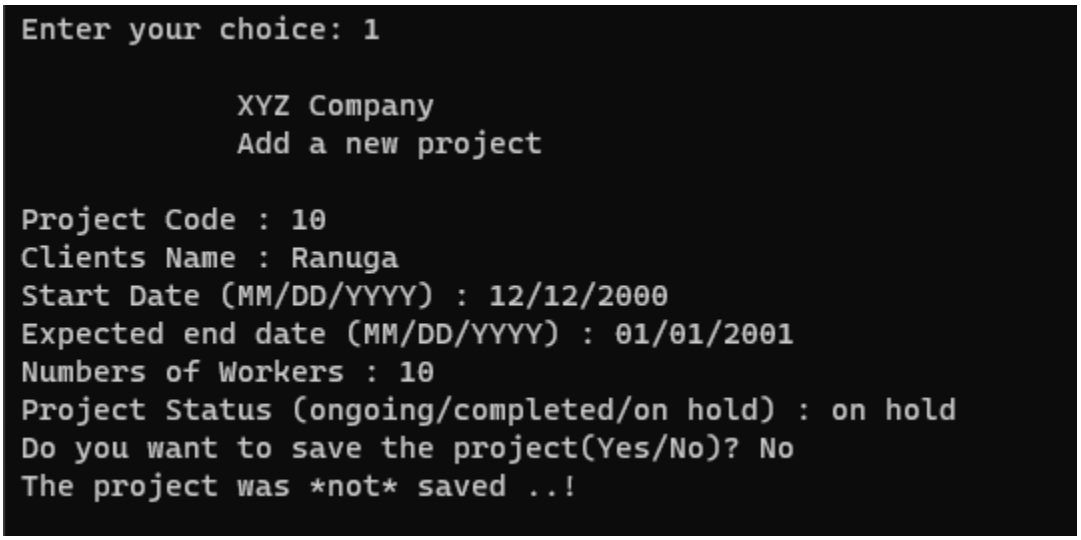


Figure 23

Test 6

The following test cases are for the 1st choice ("Add a new project to existing projects.")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"12/12/2000"	Continue	Continue	Pass
Expected End Date	"01/01/2001"	Continue	Continue	Pass
Number of Workers	10	Continue	Continue	Pass
Project Status	"on hold"	Continue	Continue	Pass
Save	"Yes"	"Successfully created a new project (True)"	"Successfully created a new project (True)"	Pass

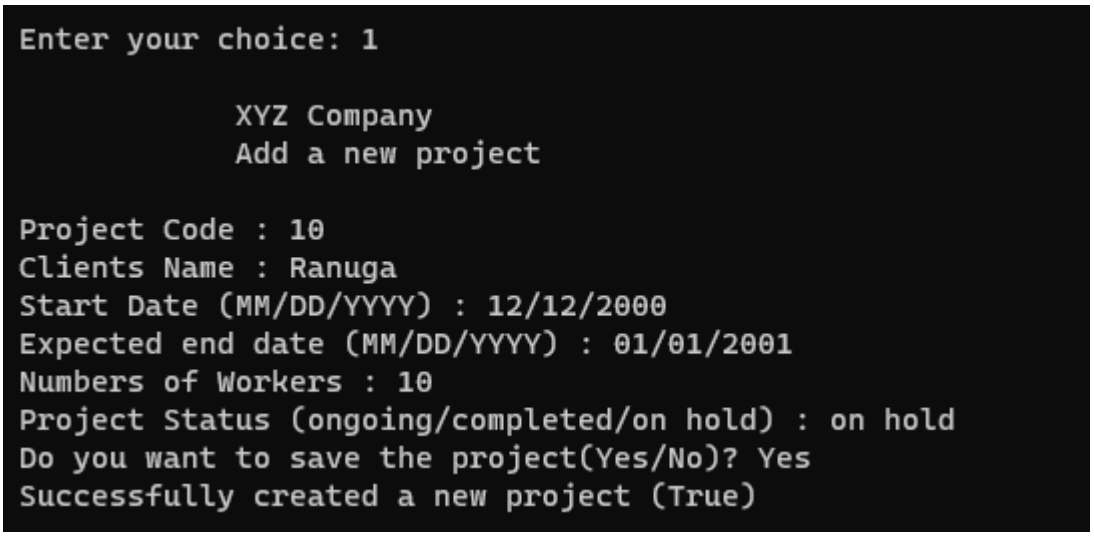


Figure 24

REMOVE A PROJECT

Assuming the project below has been added

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"12/12/2000"	Continue	Continue	Pass
Expected End Date	"01/01/2001"	Continue	Continue	Pass
Number of Workers	10	Continue	Continue	Pass
Project Status	"on hold"	Continue	Continue	Pass
Save	"Yes"	"Successfully created a new project (True)"	"Successfully created a new project (True)"	Pass

Test 1

The following test cases are for the 2nd choice ("Remove Completed Project")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"90"	"The project does not exist..!"	"The project does not exist..!"	Pass

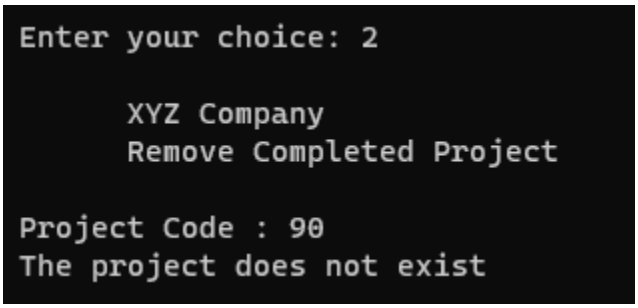


Figure 25

Test 2

The following test cases are for the 2nd choice ("Remove Completed Project")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Save	"No"	"The project was not removed…!"	"The v was not removed…!"	Pass

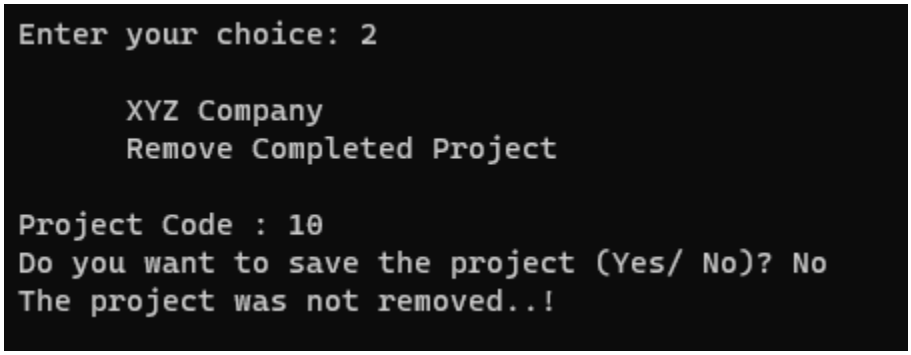


Figure 26

Test 3

The following test cases are for the 2nd choice ("Remove Completed Project")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Save	"Yes"	"Successfully removed completed projects."	"Successfully removed completed projects."	Pass

```
Enter your choice: 2

      XYZ Company
      Remove Completed Project

Project Code : 10
Do you want to save the project (Yes/ No)? yes
Successfully removed completed projects. (True)
```

Figure 27

ADD NEW WORKERS

Test 1

The following test cases are for the 3rd choice ("Add new Workers")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Number of Workers	-10	"Workers must be more than 0..!"	"Workers must be more than 0..!"	Pass

```
Enter your choice: 3

      XYZ Company
      Add new Workers

Number Workers to Add : -10
Workers must be more than 0..!
```

Figure 28

Test 2

The following test cases are for the 3rd choice ("Add new Workers")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Number of Workers	"test"	"The msg entered was not an integer"	"The msg entered was not an integer"	Pass

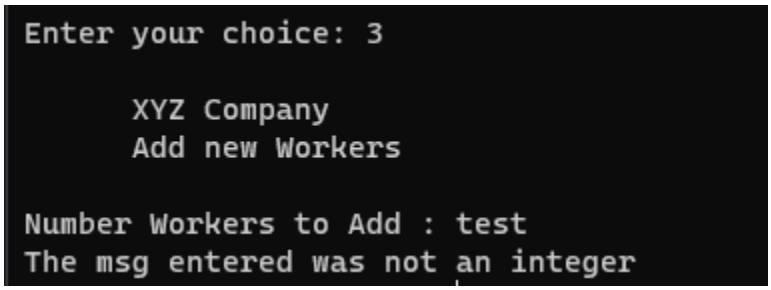


Figure 29

Test 3

The following test cases are for the 3rd choice ("Add new Workers")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Number of Workers	10	Continue	Continue	Pass
Save	No	" Workers were not added...!"	" Workers were not added...!"	Pass

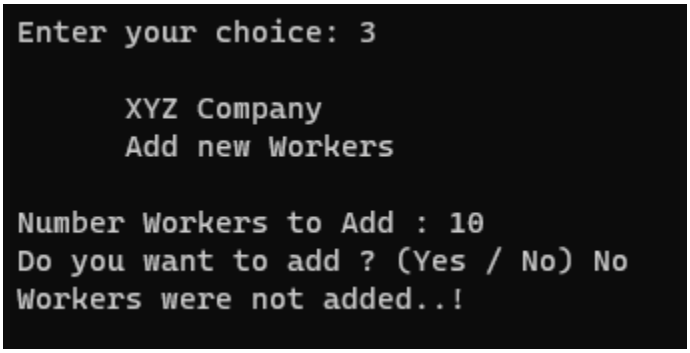


Figure 30

Test 4

The following test cases are for the 3rd choice ("Add new Workers")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Number of Workers	10	Continue	Continue	Pass
Save	Yes	"Workers added successfully...!"	"Workers added successfully...!"	Pass

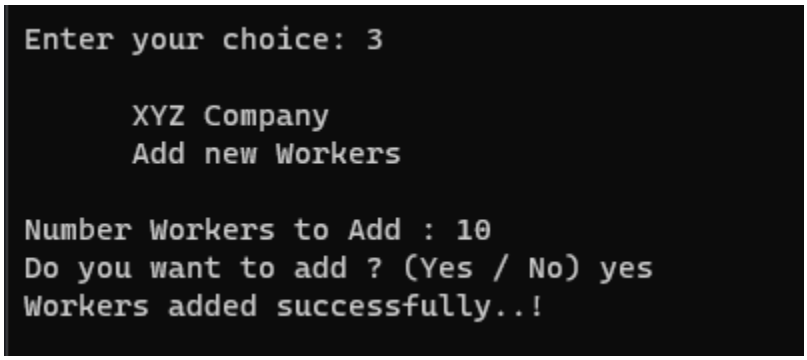


Figure 31

UPDATE PROJECT DETAILS

Assuming the project below has been added

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"12/12/2000"	Continue	Continue	Pass
Expected End Date	"01/01/2001"	Continue	Continue	Pass
Number of Workers	10	Continue	Continue	Pass
Project Status	"on hold"	Continue	Continue	Pass
Save	"Yes"	"Successfully created a new project (True)"	"Successfully created a new project (True)"	Pass

Test 1

The following test cases are for the 4th choice ("Update Project Details")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	0	Exit and Go to the Main Menu	Exit and Go to the Main Menu	Pass

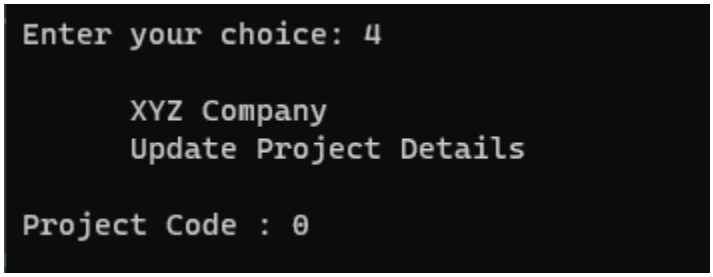


Figure 32

Test 2

The following test cases are for the 4th choice (“Update Project Details”)

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	“10”	Continue	Continue	Pass
Clients Name	“Devin”	Continue	Continue	Pass
Start Date	“0000”	“Enter a valid format of the date…!”	“Enter a valid format of the date…!”	Pass

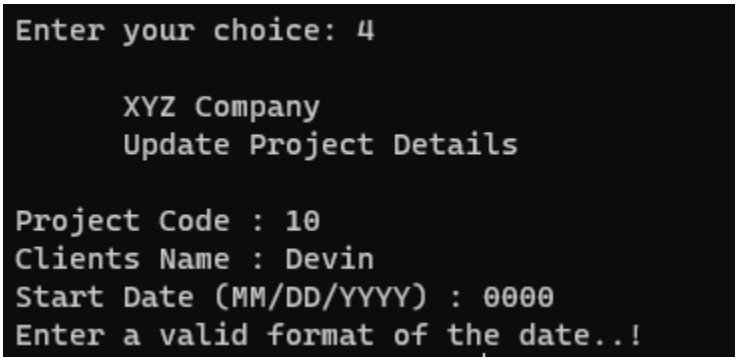


Figure 33

Test 3

The following test cases are for the 4th choice (“Update Project Details”)

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	“10”	Continue	Continue	Pass
Clients Name	“Devin”	Continue	Continue	Pass
Start Date	“01/01/2020”	Continue	Continue	Pass
Expected End Date	“007”	“Enter a valid format of the date…!”	“Enter a valid format of the date…!”	Pass

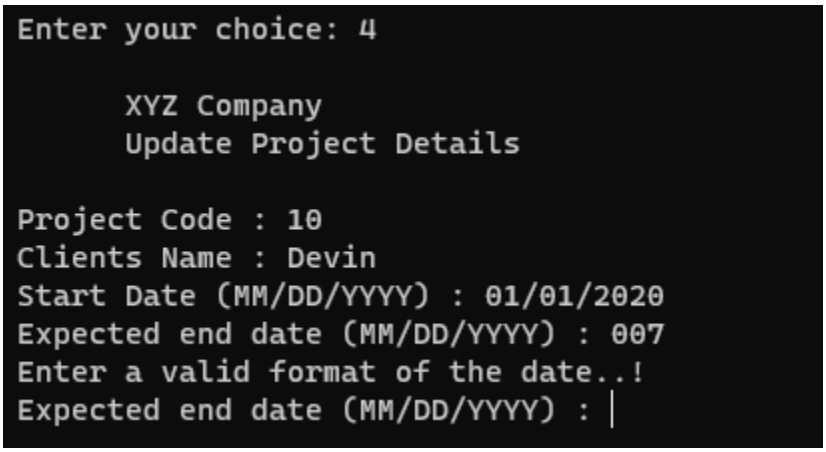


Figure 34

Test 4

The following test cases are for the 4th choice ("Update Project Details")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Devin"	Continue	Continue	Pass
Start Date	"01/01/2020"	Continue	Continue	Pass
Expected End Date	"12/12/2022"	Continue	Continue	Pass
Number of Workers	"test"	"The msg entered was not an integer"	"The msg entered was not an integer"	Pass

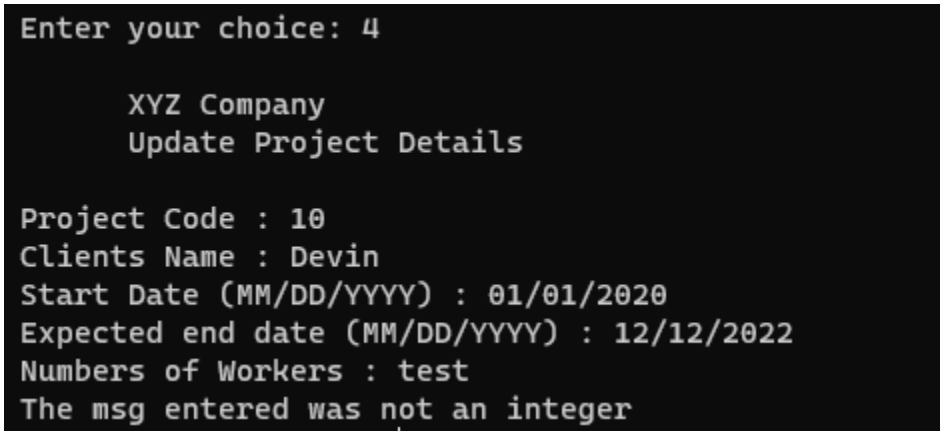


Figure 35

Test 5

The following test cases are for the 4th choice (“Update Project Details”)

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	“10”	Continue	Continue	Pass
Clients Name	“Devin”	Continue	Continue	Pass
Start Date	“01/01/2020”	Continue	Continue	Pass
Expected End Date	“12/12/2022”	Continue	Continue	Pass
Number of Workers	25	Continue	Continue	Pass
Project Status	“on done”	“The entered project status is incorrect…”	“The entered project status is incorrect…”	Pass

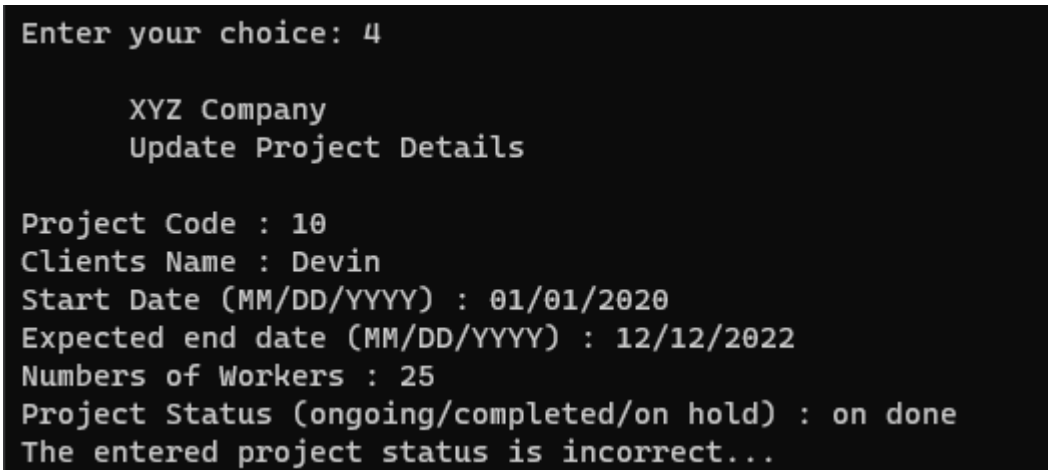


Figure 36

Test 6

The following test cases are for the 4th choice (“Update Project Details”)

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	“10”	Continue	Continue	Pass
Clients Name	“Devin”	Continue	Continue	Pass
Start Date	“01/01/2020”	Continue	Continue	Pass
Expected End Date	“12/12/2022”	Continue	Continue	Pass
Number of Workers	25	Continue	Continue	Pass
Project Status	“on hold”	Continue	Continue	Pass
Save	“no”	“The project was not updated”	“The project was not updated”	Pass

```
Enter your choice: 4

      XYZ Company
      Update Project Details

Project Code : 10
Clients Name : Devin
Start Date (MM/DD/YYYY) : 01/01/2020
Expected end date (MM/DD/YYYY) : 12/12/2022
Numbers of Workers : 25
Project Status (ongoing/completed/on hold) : on hold
Do you want to update the project details (Yes/No)?no
There isn't a project with the mentioned project code..!
```

Figure 37

Test 7

The following test cases are for the 4th choice ("Update Project Details")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Devin"	Continue	Continue	Pass
Start Date	"01/01/2020"	Continue	Continue	Pass
Expected End Date	"12/12/2022"	Continue	Continue	Pass
Number of Workers	25	Continue	Continue	Pass
Project Status	"on hold"	Continue	Continue	Pass
Save	"yes"	"The project was not updated"	"The project was not updated"	Pass

```
Enter your choice: 4

      XYZ Company
      Update Project Details

Project Code : 10
Clients Name : Devin
Start Date (MM/DD/YYYY) : 01/01/2020
Expected end date (MM/DD/YYYY) : 12/12/2022
Numbers of Workers : 25
Project Status (ongoing/completed/on hold) : on hold
Do you want to update the project details (Yes/No)?yes
Project details updated successfully (True)
```

Figure 38

Test 8

The following test cases are for the 4th choice ("Update Project Details")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"58"	"There isn't a project with the mentioned project code…!"	"There isn't a project with the mentioned project code…!"	Pass

```
Enter your choice: 4

      XYZ Company
      Update Project Details

Project Code : 58
There isn't a project with the mentioned project code..!
```

Figure 39

PROJECT STATISTICS

Assuming the project below has been added from Choice 3

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Number of Workers	100	Continue	Continue	Pass
Save	"Yes"	"Workers added successfully"	"Workers added successfully"	Pass

Assuming the project below has been added from Choice 1

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Project Code	"10"	Continue	Continue	Pass
Clients Name	"Ranuga"	Continue	Continue	Pass
Start Date	"12/12/2000"	Continue	Continue	Pass
Expected End Date	"01/01/2001"	Continue	Continue	Pass
Number of Workers	10	Continue	Continue	Pass
Project Status	"on going"	Continue	Continue	Pass
Save	"Yes"	"Successfully created a new project (True)"	"Successfully created a new project (True)"	Pass

Test 1

The following test cases are for the 5th choice ("Project Statistics")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Add Project	"No"	Exit and Go to the Main Menu	Exit and Go to the Main Menu	Pass

```
Enter your choice: 5

      XYZ Company
      Project Statistics

Number of ongoing projects : 1
Number of completed projects : 0
Number of onhold projects : 0
Number of available workers : 90
Do you want to add the project (Yes/No)?no
```

Figure 40

Test 2

The following test cases are for the 5th choice ("Project Statistics")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Add Project	"Yes"	Redirect to Choice 1	Redirect to Choice 1	Pass

```
Enter your choice: 5

      XYZ Company
      Project Statistics

Number of ongoing projects : 1
Number of completed projects : 0
Number of onhold projects : 0
Number of available workers : 90
Do you want to add the project (Yes/No)?yes
Redirecting...

            XYZ Company
            Add a new project

Project Code :
```

Figure 41

EXIT

Test 1

The following test cases are for the 6th choice ("Exit")

Field	Input Entered	Expected Outcome	Actual Outcome	Results
Choice	6	"Exiting Program…"	"Exiting Program…"	Pass

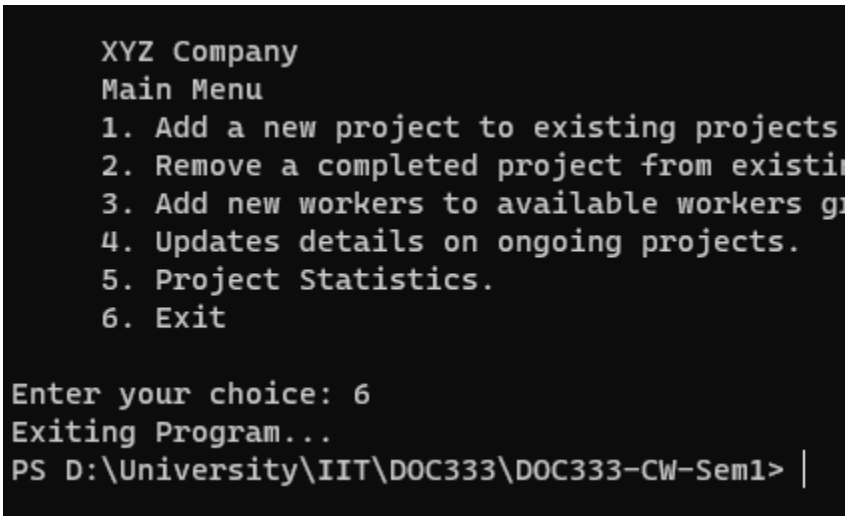


Figure 42