Case Study: BugTrack – Bug Tracking System

1.1 INSTRUCTIONS

- Create a directory by your name in drive "**D:**" Under that create a subdirectory called **MiniProject** and Store your Project there.
- Use C or C++. You can refer to your course material.
- The code modules in the mini project should follow all the coding standards.
- You may also look up the help provided in the MSDN Library.

Number of participants for this project = 3 to 4

PROBLEM STATEMENT

To create a bug tracking system BugTrack for the software production. The application is to be develope as Desktop Application. There is one entity Employee who acts like meeting organizer or participant.

1.3 ABSTRACT OF THE PROJECT

1. Employee should be able to login through application.
2. System should have details about the project.
3. Employees should be able to raise ticket for the bug.
4. Provide priority for bugs [(L)ow, (M)edium, (H)igh, (S)evere).
5. Provide status for bugs [(O)pen, (A)ssigned, (F)ixed, (S)ubmitted, (Q)uality checked, (C)lose].
6. Employee should be able to add note to the ticket.
7. Employee should be able to paste code snippets.
8. Once bug is solved employee should be able to close the ticket.
9. All projects can be viewed by all employees. Employee can modify only projects created by him.
10. All tickets can be viewed by all employees if he/she knows Project ID.
11. Tickets can be modified by employee who either created it or is assigned to him/her.
12. System should maintain trail of changes in "Assigned To" and "Status" fields of tickets.
13. Handle data and errors properly. Show appropriate messages to user.
14. Display good input, output messages and reports in proper form at.
15. Security features should be implemented whereever possible. For example user passwords can be stored in encrytped format.

1.4 FUNCTIONAL COMPONENTS OF THE PROJECT

Following is a list of functionalities of the system. Wherever, the description of functionality is not adequate; you can make appropriate assumptions and proceed.

1. When MyCalendar starts it displays Following Screen -

----------Login Screen----------
Enter Employee ID <EmpID>
Enter password <password>

Information about employees is available in "employees.txt" file where one line correspond to one employee. It is comma separated file containing following information about employee -

Employee ID	number(4)
Employee Password	string(10)
Employee Name	string(50)
Department	string(10)
Mobile Number	number(10)
email	string(30)
Skype ID	string(20)

Create "employees.txt" in notepad having information about at least 20 employees from different departments.

When Employee ID and password is entered it is checked against entry in "employees.txt".

- If match is found "BugTrack Menu" is displayed.
- If match is not found then message "Invalid Employee ID or password" is displayed and system exits.

2. BugTrack Menu

---------- BugTrack Menu----------

1. Manage Projects
2. Manage Tickets
3. Reports
0. Quit

Enter your option : <option>

option = 1 (Manage Projects)

Information about projects is stored in "projects.txt" file where one line correspond to one project. It is comma separated file containing following information about project -

Project ID	number(4) auto increment
Created By	number(4) = Employee ID of login Employee
Project Description	string(50)
Create Date	date (DD/MM/YYYY format) = today's date
Project Duration	int(2) = number of months
Department	string(50)
Group	string(50)

Following sub-menu will be displayed -

---------- Projects Menu----------

1. Add Project

2. Modify Project
3. Delete Project
0. Quit

Only projects created by login employee can be modified.
Only projects which do not have any tickets raised can be deleted.

Option = 2 (Manage Tickets)

Information about tickets is stored in "tickets.txt" file where one line correspond to one bug. It is comma separated file containing following information about ticket -

Project ID	number(4)
Ticket ID	number(4) auto increment
Priority	char(1) = (L)ow, (M)edium, (H)igh, (S)evere
Created By	number(4) = Employee ID of login Employee
Bug Description	string(50)
Create Date	date (DD/MM/YYYY format) = today's date (initially)
Assigned To	number(4) = Employee ID of login Employee (initally)
Closed on	date (DD/MM/YYYY format)
Status	char(1) = (O)pen, (A)ssigned, (F)ixed, (S)ubmitted, (Q)uality checked, (C)lose
Estimated Fix Time	int(2) = number of hours required to fix bug
Note	string(540)
Code Snippet	string(540)

Following sub-menu will be displayed -

---------- Tickets Menu----------
1. Add Ticket
2. Modify Ticket
3. Close Ticket
4. Add note to Ticket
5. Add code snippet to Ticket
0. Quit

Updation to tickets should be made according to option selected in above menu.

The system should maintain Ticket Trail in "trails.txt" file where one line correspond to one change. It is comma separated file containing following information about changes in ticket status -

Ticket ID	number(4)
ChangeDate	date (DD/MM/YYYY format)
OldEmployeeID	number(4) Employee who made the change
NewEmployeeID	number(4) Employee to whom ticket is assigned
OldStatus	char(1)
NewStatus	char(1)

Option = 3 (Reports)

Following sub-menu will be displayed -

---------- Reports Menu----------

1. Prioritywise list of tickets which are not Closed for given project
2. Prioritywise list of tickets which assigned by logged in employee
3. Prioritywise list of tickets created by logged in employee
....
0. Quit

Provide at least 10 reports based on data present in projects.txt, tickets.txt and trails.txt by taking appropriate inputs.

Assumptions:

Note:

If you are using C

1. Use Linked Lists to read data from corresponding text files at the beginning of program.
2. Updates on data during program execution should be done in these Linked Lists.
3. When user quits the program all Linked Lists data to be writting in corresponding text files so that updated data is available in next program execution.

If you are using C++ Arrays of objects can be used instead of Linked List.

Set Up Checklist for Mini Project

Software Requirement:

Visual studio 6.0 and above (To write code using C or C++).

Minimum System / Hardware Requirements:

Intel Pentium 90 or higher (P166 recommended)
Microsoft Windows 95, 98, or NT 4.0, 2k, XP, Access to UNIX server through Telnet
Memory: 32MB of RAM (64MB or more recommended)