# nk002734

Generated by Doxygen 1.8.17

# Chapter 1

# _FrontPage

Module Code: CS1PC20 Assignment report Title: Project Manager Student Number: 30002734 Date (when the work completed): Jan 10th 2022 Actual hrs spent for the assignment: $\sim$40

Chapter 2

# Coursework 2 - a project manager tool

Using the file system as the main datastore, this tool helps manage work breakdown methodology, milestones and time management.

The core elements will be strongly guided, and completion of them equates to approximately 40% of the marks. Beyond that, students should problem solve, in a graded hierarchy of difficulty and completion to achieve higher marks.

**It is unrealistic to expect to meet all the feature criteria listed**

Part of the task therefore involves deciding which elements to implement in order to maximise your mark.

Some of the features may be best implemented using shell (bash or zsh) scripting, or could be done directly in C.

Tests will be provided for 1...10, which will also define the command line arguments required (including the sub-command name for the features, where appropriate e.g. pm feature feature_name or pm feature move new_↩
feature_name ).

Coursework submission will include creating project documentation (using doxygen) so code ***must*** be well commented. Any shell scripts or make files included in the solution ***must*** be included in the documentation. The submission itself ***must*** be the URL of the CSGitLab repository used.

## 2.1 Core functionality:

### 2.1.1 Must (up to 10% for each top level feature)

1. Feature management
   (a) ***Should*** include setting up git branch as appropriate

### 2.1.2 Should (up to 10% for each top level feature)

1. Time/workload added to output diagram
   (a) ***Could*** also produce Gantt chart (using plantuml)

### 2.1.3 Could (up to 10% for each top level feature)

1. Output diagram includes links (when used in browser, for example)
   (a) ***Should*** use plantuml to do this
2. Dependencies information across tree branches
   (a) ***Must*** identify relevant other paths in tree to do this

### 2.1.4 Elite challenges ("Won't do" in MoSCoW terms) (up to 20% for each top level feature)

1. Guided work breakdown wizard (Slightly advanced, would require interactive questions/answer handling)
   (a) Needs a number of sub-features, such as minimum time allocation threshold, user input parsing
2. Multi-user (Advanced, would require some form of permissions model)
   (a) may be best done using a traditional SQL database, but can use flat files. Complex task.
3. Available as web application (Advanced, probably easiest creating a simple embedded server)
   (a) sample code for simple communications between applications will be covered
4. GOAP recommendation of suitable pathway (Advanced, can use existing GOAP library, however)
   (a) GOAP uses a 'heap' data structure