Sheet Metal Client Hub: Development Plan

Overview

The Sheet Metal Client Hub is a Python-based GUI application using Tkinter, designed to automate cost calculations for sheet metal parts, replacing manual quoting processes for UK fabricators. The system enables users to log in, input part details, calculate costs for 10 work centres (cutting, bending, welding, deburring, assembly, inspection, surface treatment, machining, forming, fastening) and additional processes (specialist finishing), adjust rates, and save results, with costs in GBP and dimensions in mm.

The project follows the Waterfall software development lifecycle (SDLC), spanning 1 April to 2 June 2025 (62 days, 65-80 hours), and meets SCQF Level 7 requirements for J5RH47 (project). This planning document outlines the scope, timeline, risks, and resources, with future deliverables (design document, source code, testing documentation) to follow.

Scope and Objectives

The project aims to deliver a desktop application that:

1) Provides secure user access.
2) Facilitates input and management of part details.
3) Calculates costs for sheet metal fabrication across 10 work centres and additional processes.
4) Saves results for record-keeping.
5) Ensures an accessible, intuitive interface.

The system will support future enhancements (e.g., STEP file integration), with planning focused on achieving these objectives within the academic timeline.

Waterfall Justification

The Waterfall methodology is selected for its structured approach, ideal for a fixed academic project, ensuring thorough planning before development. It follows SDLC phases (planning, design, development, testing, documentation), supporting the timeline and minimising scope creep.

Sources

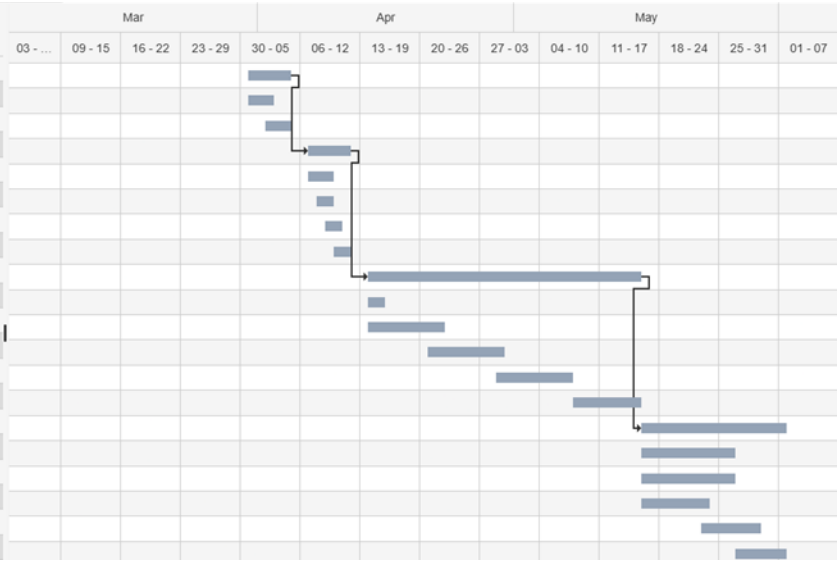
Fractory (www.fractory.com): Inspires the quotation engine's functionality.
Metals4u (metals4u.co.uk): Supplies material data (e.g., thickness ranges).
ETM Manufacturing (v3, www.etmmfg.com): "Estimating Sheet Metal Fabrication Costs".
Articulate Rise (Fife College learning platform): Guides structure chart design.

Timeline and Subtasks

The project spans 1 April to 2 June 2025, with SDLC phases for planning/design, development, and testing/documentation. The Gantt chart visualises task durations and concurrency, created in Visual Paradigm and inserted as a PNG.

Gantt Chart

ID	Title	Start Time	End Time
1	**Planning**	03/31/2025	04/05/2025
2	Define scope	03/31/2025	04/03/2025
3	Create development plan	04/02/2025	04/05/2025
4	**Design**	04/07/2025	04/12/2025
5	Wireframes	04/07/2025	04/10/2025
6	Data dictionary	04/08/2025	04/10/2025
8	Pseudocode	04/09/2025	04/11/2025
7	Structure chart	04/10/2025	04/12/2025
9	**Development**	04/14/2025	05/16/2025
10	Setup environment	04/14/2025	04/16/2025
11	Login module	04/14/2025	04/23/2025
12	Input module	04/21/2025	04/30/2025
13	Output module	04/29/2025	05/08/2025
14	Settings module	05/08/2025	05/16/2025
15	**Testing & Docs**	05/16/2025	06/02/2025
16	Unit testing	05/16/2025	05/27/2025
17	Integration testing	05/16/2025	05/27/2025
18	System testing	05/16/2025	05/24/2025
19	Debug and fix	05/23/2025	05/30/2025
20	Final documentation	05/27/2025	06/02/2025



Planning (1-7 April):
Define scope (3 days), create development plan (2 days).

Design (8-15 April):
Wireframes (3 days), data dictionary (2 days), structure chart (2 days), pseudocode (2 days).

Development (16 April – 15 May):
Setup environment (2 days), login module (5 days), input module (7 days), output module (7 days), settings module (5 days).

Testing & Documentation (16 May – 2 June):
Unit, integration, and system testing (7 days each), debug and fix (5 days), final documentation (6 days).

**** Note: concurrency in testing tasks to optimise the timeline. ****

Testing Strategy

Testing will occur during development and testing phases (16 April – 2 June 2025) to ensure functionality. The strategy includes:

1) **Unit Testing:** Validate individual components.
2) **Integration Testing:** Verify module interactions.
3) **System Testing:** Test end-to-end functionality and accessibility.

A detailed test plan will be submitted on 27 May 2025.

Risk Management

1) **Risk 1:** Incorrect cost rates may lead to inaccurate calculations.
   Mitigation: Plan for rate validation and prompts.
2) **Risk 2:** Timeline slippage due to coding challenges.
   Mitigation: Allocate extra development hours, seek support.

3) **Risk 3:** File handling errors.
   Mitigation: Plan robust error handling and logging.
4) **Risk 4:** Scope creep from additional features.
   Mitigation: Adhere to defined scope, defer enhancements.

Resource Requirements

1) **Hardware:** Windows PC with 4GB RAM, 500MB free disk space.
2) **Software:** Python 3.9, Tkinter, PyScripter, Git, Visual Paradigm.
3) **Training:** Basic Python tutorial (2-3 hours).
4) **Files:** users.txt, rates_<username>.txt, rates_global.txt, output.txt.

Assumptions and Constraints

Assumptions: Users have basic Python knowledge; single-user access initially.
Constraints: Limited to Python 3.9, Tkinter; no web/CAD integration; fixed timeline.

Tools

1) **Python 3.9:** Programming language.
2) **Tkinter:** GUI framework.
3) **PyScripter:** IDE**.**
4) **Git/GitHub:** Version control, hosted at https://github.com/LJMoffat81/Sheet-Metal-Client-Hub.
5) **Visual Paradigm:** Design diagrams.

Professional Conduct

The project will adhere to ethical standards, ensuring secure data handling and academic integrity, with detailed conduct documentation in the implementation phase.