|  |
| --- |
|  |

|  |
| --- |
| REVISION HISTORY |

|  |  |  |  |
| --- | --- | --- | --- |
| **VERSION** | **DATE** | **DESCRIPTION** | **AUTHOR** |
| V1 | 21.09.25 | The first document revision for the class on 21st. | **Patrik Tamm** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |
| --- |
| CONTENTS |

[1 SW System Overview 4](#_Toc208787546)

[1.1 Purpose 4](#_Toc208787547)

[1.2 Scope 4](#_Toc208787548)

[1.3 Use-Case Diagram 4](#_Toc208787549)

[1.4 General Constraints 4](#_Toc208787550)

[1.5 Assumptions and Dependencies 4](#_Toc208787551)

[1.6 Acronyms and Abbreviations 4](#_Toc208787552)

[2 SW Functional Requirements 5](#_Toc208787553)

[2.1 Features / Functions to be Implemented 5](#_Toc208787554)

[2.1 Acceptance Criteria 5](#_Toc208787555)

[2.2 Implementation Requirements 5](#_Toc208787556)

[3 SW Non-Functional Requirements 6](#_Toc208787557)

[3.1 Resource Consumption 6](#_Toc208787558)

[3.2 License Issues 6](#_Toc208787559)

[3.3 Coding Standard 6](#_Toc208787560)

[3.4 Modular Design 6](#_Toc208787561)

[3.5 Reliability 6](#_Toc208787562)

[3.6 Portability 6](#_Toc208787563)

[3.7 General Operational Guidelines 6](#_Toc208787564)

[4 SW Design Artifacts 7](#_Toc208787565)

[4.1 CRC Cards (Class–Responsibility–Collaboration) 7](#_Toc208787566)

[4.2 Conceptual UML Diagram (entities & relationships) 7](#_Toc208787567)

# SW System Overview

## Purpose

The purpose of the system is to create and enable the billing systems of a photo studio. The intended users are photo studios who need a system to manage orders and billing. It is being developed to reduce paperwork and reduce the time taken for administrative overhaul

## Scope

The system includes order management for photo printing and film developing, client information recording, automated pricing with express surcharges, consumable material tracking, and daily report generation for revenue and resource usage. The system excludes online customer portals, inventory procurement management, and complex scheduling features beyond basic order processing. Key benefits include a simple workflow between staff roles, accurate pricing calculations, elimination of paper-based processes, and automated reporting that reduces administrative overhead while improving operational transparency.

## Use-Case Diagram

Provided in the repo.

## General Constraints

The system is intended to be functional on both Windows and Ubuntu machines that are up to date and running hardware issued in the last 10 years. To fulfill said requirements, the program will be written in C++ 23 that can be run in most machines. Program will be CLI only.

## Assumptions and Dependencies

The shop has a stable source of internet, electricity and a computer fulfilling the constraints,

## Acronyms and Abbreviations

*List all acronyms and abbreviations used in the document along with their explanations.*

|  |  |
| --- | --- |
| **Terms Used** | **Description of terms** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# SW Functional Requirements

## 2.1 Features / Functions to be Implemented

User Story 1: As a client, I want to place an order for photo printing or film developing so that I can get my photos processed by the studio. The system should provide me with an order receipt showing my details and expected completion time.

User Story 2: As a receptionist, I want to record client details and completion times so that orders are properly tracked and scheduled. The system should generate both a client copy and a photographer copy of each order.

User Story 3: As a receptionist, I want to calculate pricing with express surcharges so that clients pay the correct amount for urgent orders. Express orders should automatically apply a 25% surcharge to the base price.

User Story 4: As a photographer, I want to receive order details and mark materials used so that I can process work efficiently and track consumables. The system should allow me to input quantities of paper, developer, and other materials consumed for each order.

User Story 5: As an administrator, I want to generate daily reports on completed orders and material usage so that I can monitor studio performance and inventory. Reports should include total revenue, number of orders completed, and a breakdown of materials consumed.

User Story 6: As a receptionist, I want to process payments and mark orders complete so that clients receive their finished photos and the transaction is finalized. Payment should only be accepted when the photographer has marked the order as ready for pickup.

Necessary components identified:

* An interface for receptionists to insert orders.
  + A form for the client
  + A for the photographer
* A method to identify express orders
* A way for the photographer to view orders
* An interface for the photographer to insert a report on the consumable
* A way for the administrator to view the used materials.

## Acceptance Criteria

Define how each requirement will be validated: test cases, acceptance tests, or quality metrics.

Requirements will be validated with test cases:

* A test to verify the validity of order forms (client name, contact info, order type, completion time),
* A test to confirm express orders automatically apply 25% surcharge to base pricing,
* A test to ensure standard orders use base pricing without surcharges,
* A test to verify photographer can record material consumption (paper, developer quantities),
* A test to validate daily reports generate correct totals for revenue and completed orders,
* A test to confirm orders cannot be marked complete until photographer processing is finished,
* A test to verify material usage tracking accurately reflects consumed quantities,
* A test to ensure both client and photographer copies of orders are generated,
* A test to validate payment processing only occurs for completed orders,
* A test to confirm client information is properly stored and retrievable,
* A test to verify completion time calculations are accurate for both standard and express orders,

Each test case will include specific input values, expected outputs, and pass/fail criteria to ensure the system meets functional requirements.

## Implementation Requirements

As the system is designed to work as a standalone application, there are no external dependencies that need to be considered.

# SW Non-Functional Requirements

## Resource Consumption

Considering the project, the program should be able to run on almost any kind of hardware as long they fulfill the system requirements

## License Issues

So far there are no license issues. No 3rd party libraries are used so far in the development process

## Coding Standard

Coding standards are used throughout the entire project, with the code being formatted in an unified style, types kept clean and modern architectural considerations

## Modular Design

Code is kept using OOP, object-oriented programming and properly defined header files allowing for fast code development and enabling joint effort development.

## Reliability

Smaller exceptions should be handled gracefully and with letting the user know, but without noticeable usage loss.

## Portability

The application should run on Windows machines with Visual Studio installations present.

## General Operational Guidelines

The code should be maintained regularly and following common standards for code.

# SW Design Artifacts

## CRC Cards (Class–Responsibility–Collaboration)

## The CRC cards can be found from the repository.

## Conceptual UML Diagram (entities & relationships)

*A diagram of a data flow

AI-generated content may be incorrect.*