# Plastering Management Application

Kyle Kirkby

October 3, 2014

# Contents

# Chapter 1

# Analysis

## 1.1 Introduction

### 1.1.1 Client Identification

My client is 30 year old plasterer Dan Austin who runs his own plastering business known as DnA Plastering. Dan mainly uses his Toshiba laptop (Dual Core Intel with 6 GB Ram and running Windows 8 64 bit) to do basic tasks such as social networking and receiving/sending emails.

The current system is a paper based method where he records the prices and measurements of the plastering/screening/rendering jobs he undertakes. Dan works in an around the Suffolk/Essex area but occasionally takes on larger jobs further afield in places such as London or Epping. All the recording and calculations are done by Dan himself and does not require additional assistance in completing these tasks but is looking for a digital solution to the organisation problems faced with the current manual paper method.

Dan is looking to introduce a computer based system to replace the current one in order to make keeping track of jobs and pricing up new jobs easier and more efficient. Alongside this he would like to be able to keep information on all of his customers so he can simply search for clients' details and contact information all in one location. He will also be able to look up the jobs that he has done for them to make sending invoices easier and manageable.

### 1.1.2 Define the current system

The current system in place is a paper/notebook based system where details of clients are stored along with prices of jobs and cost of materials needed etc.

The details of the clients include their address, phone number, email, first name and surname. The infromation about the job usually includes the measurements of what needs to be plastered along with how long it will take to complete and if he is taking any labourers to too. Calculations are often also made to work out how much to charge depending on the price he is charging per square meter. This rate often changes depending on the current economy.

Once all the calculations are made, he works out how much the materials are going to cost and also how long it will take him to complete the job. Once all these calculations and prices have been evaluated he notifies the client of the price; when the price is confirmed the job is undertaken.

Finally, Dan writes out an invoice using a standard invoice book purchased from a stationary store to inform the client of the costs and charges of the job. The current folder containing the invoices for his clients is not organised and offers another problem whereby finding information for jobs is difficult due to the inability to search quickly for any given customer.

### 1.1.3   Describe the problems

Problems are plentiful in the current system. One of the main problems is keeping valuable client data from being lost or damaged as there is only one hard copy made in a notebook. Another problem with the notebook is not being able to easily search through the details of all the clients to find specific phone numbers or contact details. Using a computer based system would allow Dan to search through his clients efficiently and allow him to make backups of the valuable client and job data.

### 1.1.4   Section appendix

**Interview with client**

1. **What system are you currently using?**

   I've got a Toshiba laptop with Windows 8, 6GB RAM and a dual core processor

2. **Are there any issues with the system currently being employed?**

   Just organisational issues when it comes to finding clients information etc. Would be good to have a database which I could search for info with.

3. **What data do you record at the moment?**

   The clients name, address, phone number, job information like measurements of the rooms etc.

4. **How often do you record data in the current system and how much data is stored each time?**

A few times a week and not much, a few lines in the workbook I use.

5. **What happens to older client information?**

Normally gets lost as and when I replace any of my books I write the details down in.

6. **Are there any storage issues when it comes to storing data manually?**

No, nothing really as I only need room for a book or two.

7. **Are changes often made to existing records of client data or job info?**

Occasionally when a client gives me a new contact number but nothing major.

8. **What is the typical routine when it comes to gaining a new client or job?**

Normally I get a call from a colleague who gives me the details about a job which I can take on or not depending on the time I have available. Sometimes clients will find my number through my website dnaplastering.co.uk or through word of mouth and will ask me to give them a quote on a job. Then I will go to the job and work out any costs involved and the time it would take to complete. Then I can arrange a time to do the job and after the job is completed I will usually write out an invoice for the client telling them about the cost for materials,labouring and profit etc.

9. **What does the client recieve in terms of invoices?**

I write out an invoice and give it to them in person after the job is completed.

10. **What format should the outputs be in within the new system?**

It would be good to be able to print the clients invoices out with minimal effort involved and possibly email the clients the invoice.

11. **Will you need to print reports and invoices or will it be sent entirely digitally to the client via email?**

I definitely need to print out some of the invoices as most of my work is done face to face so it is easier to give it to them in person. It might be

useful to email it to them as well.

12. **Is there a security issue in regards to the data you store in the current system?**

   Not much of an issues as it is just client information but added security would be a bonus.

13. **Are there any foreseeable constraints required in the proposed system?**

   None that I am aware of.

14. **Do you have a picture in mind of what the new system could look like?**

   Not really bothered about the look of it too much anything that works as expected would be brilliant.

**Signed:**



## 1.2  Investigation

### 1.2.1  The current system

**Data sources and destinations**

There are four main data sources within the current system - The plasterer, the client, the builders merchant and visting the clients job. A client contacts Dan through a phone call placed to Dan's mobile. Sometimes a client may leave Dan a voicemail message if he is too busy to answer the call at that given moment. If this is the case then Dan will get back to the client as soon as possible. Most of the data in the current system will come from the client or the clients job - this data will be the job measurements and the clients contact information. The main data destinations are the forms given to the client i.e the quote and the invoice document.

| Source | Data | Example Data | Destination |
|---|---|---|---|
| Client | Client Contact information First-name - Lastname - phoneNumber - AddrLine1 - Addr-Line2 - AddrLine3 - AddrLine4 - PostCode - Email - JobType | John - Smith - 07809726812 - 15 - Crowley Road - Haverhill - Suf-folk - CB90DJ - john@gmail.com - Plastering Bed-room | Appointment and Client Book. |
| Plasterer | Appointment Time and Place | 16:00 at 15 Crowley Road, Haverhill | Client Calendar or Diary |
| Visiting Job Site | Measurements of Job Size and Mate-rials that need to be purchased | 4m x 5m x 3m = $60m^2$ 10 Bags of Plaster | Work Notebook |
| Plasterers Calcula-tions | Quote for the work that needs doing and agree a date it can be done. | £600, 1 Day, 15th October | Quote written out on paper or agree in person. |
| Plasterers calcula-tions for the mate-rials needed for the job | Quantity of materi-als needed for the job | 25 bags of plaster and 12m of angle beading | Builders Merchant |
| Builders Merchant | A price for the ma-terials needed | £350 for the bags of plaster and angle beading | Plasterer |
| Plasterer | Total cost of the job broken down - cost of parts,labouring and vat. Date of Job | £600 - £350 mate-rials - £50 VAT - 14/08/14 | Client. |

**Algorithms**

There are three main algorithms utilised in the current system. The first is an algorithm to agree the price of the job with the client.

---
**Algorithm 1** Agreeing a price Algorithm
---
1: **SET** *agreed* **TO** *false*
2: **WHILE** agreed = False **DO**
3:     **IF** "Client does not agree with quoted price" **THEN**
4: Discuss price and change quote if new price is agreed upon.
5:     **ELSE**
6:        **SET** *agreed* **TO** *true*
7: Arrange a date for the work to be started on.
8:     **END IF**
9: **END WHILE**
---

The second algorithm currently being used in the system is an algorithm used to see whether the work is completed.

---
**Algorithm 2** Checking whether work is complete or not.
---
1: **SET** *Complete* **TO** *False*
2: **WHILE** Complete = False **DO**
3:     **IF** "Issue/problem not fixed." **THEN**
4: Check the current problem and fix issue.
5:     **ELSE**
6:        **SET** *Complete* **TO** *True*
7:     **END IF**
8: **END WHILE**
9: Create and send invoice
---

The third algorithm being used in the system is an algorithm used to see whether the work has been paid for completely.

---
**Algorithm 3** Checking whether work has been paid for yet.
---
1: **SET** *Paid* **TO** *False*
2: **WHILE** Paid = False **DO**
3:     **IF** "Money has not been given." **THEN**
4: Send invoice and contact client
5:     **ELSE**
6:        **SET** *Paid* **TO** *True*
7:     **END IF**
8: **END WHILE**
9: Update job to paid for in book.
---

**Data flow diagrams**

**Key**

Data Store

Interface

Process
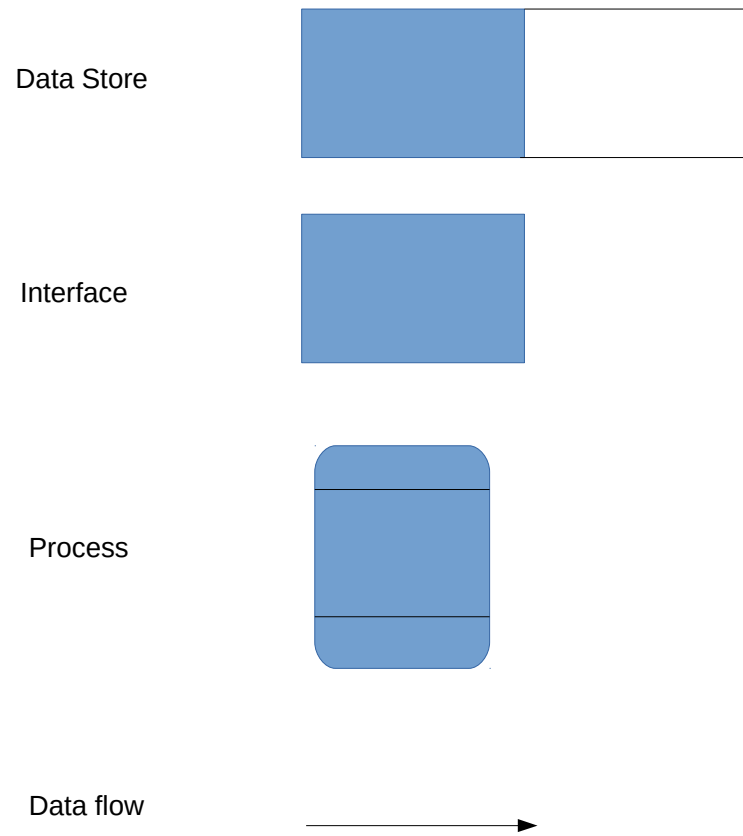
Data flow

Figure 1.1: This is the Key to be used for the following data flow diagrams.
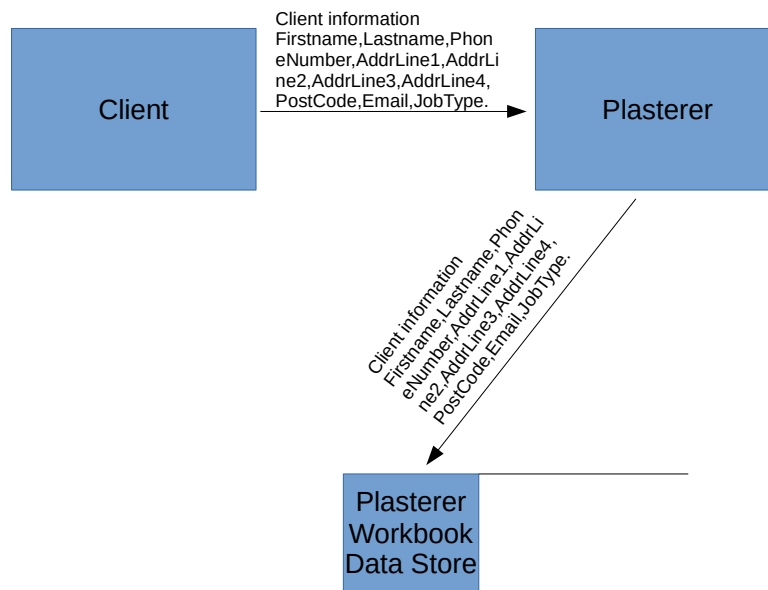
Figure 1.2: This diagram shows the flow of data when gaining a new clients information.
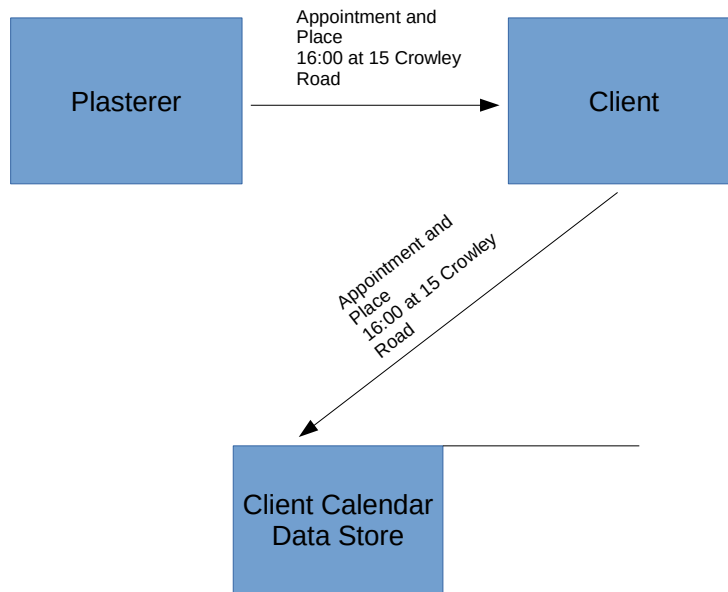
Figure 1.3: The signifies the flow of data when making an appoint for a job.
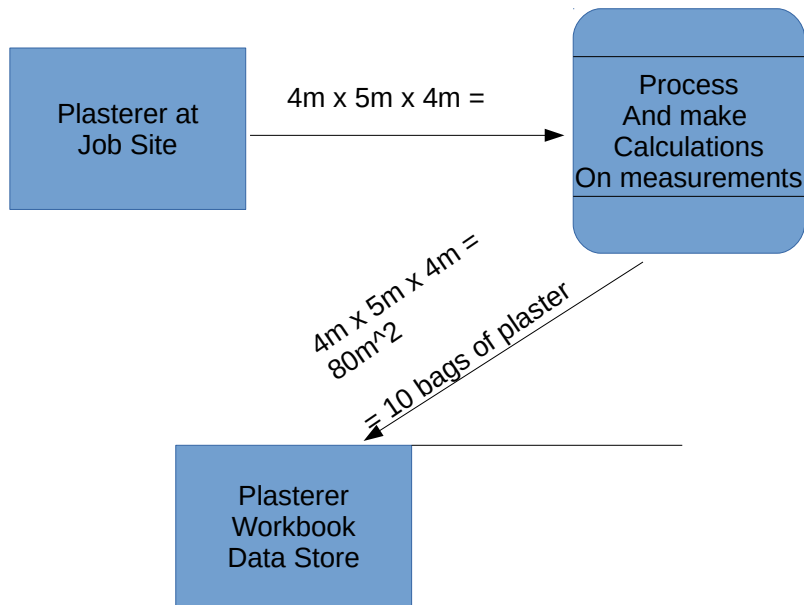
Figure 1.4: This diagram shows the flow of data when collecting the measurements for a job.
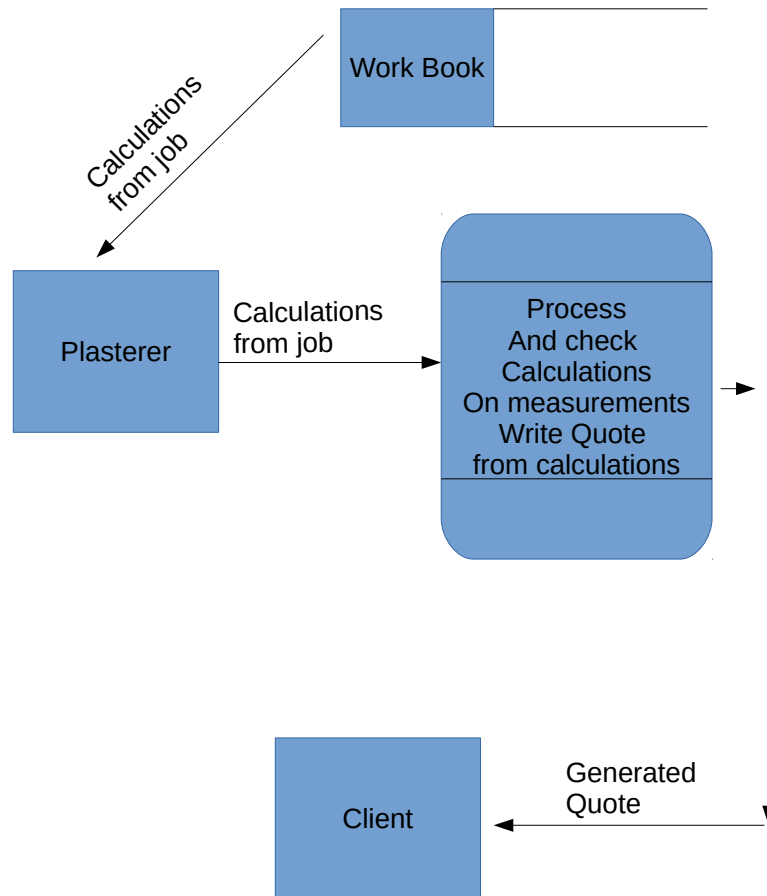
Figure 1.5: The flow of data when generating a quote for the client.
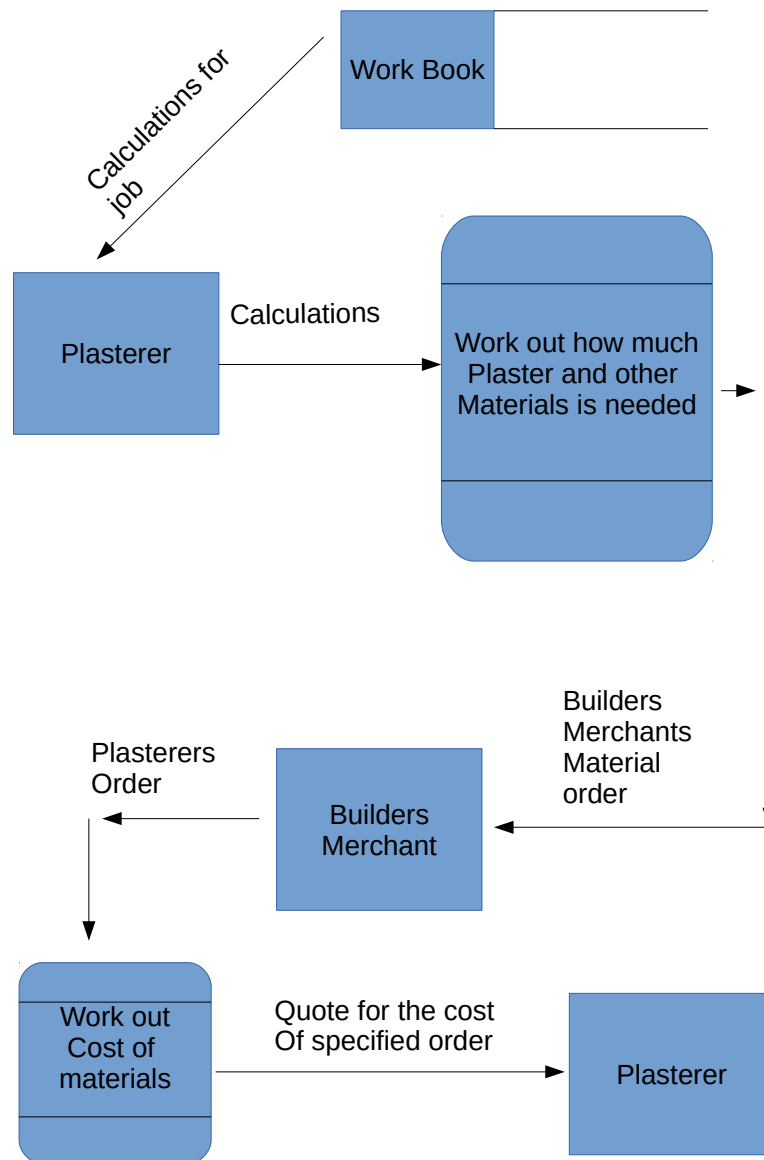
Figure 1.6: This shows the data flow when getting a quote for the materials.
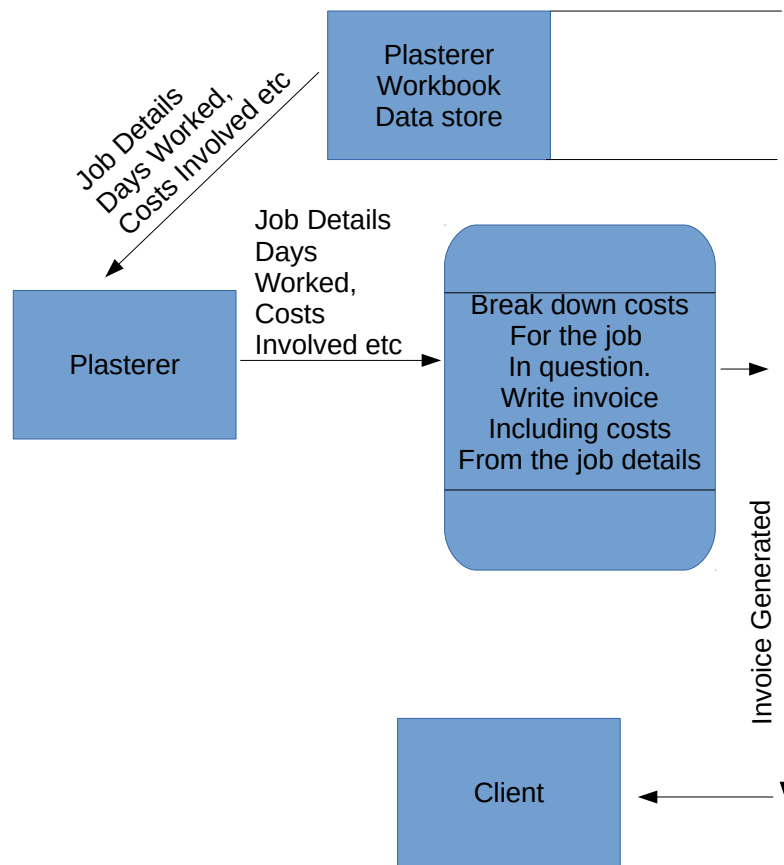
Figure 1.7: The Data flow when an invoice is given to the client.

**Input Forms, Output Forms, Report Formats**

In the system currently being used there are two main forms - the work book input form and the invoice output form. Below are a few examples of these forms as they are currently being used in the system:

Figure 1.8: This is an example of an input form where data is put into the system and is a page from a work book used to store details about jobs.

Figure 1.9: This is an example of an invoice output form which is given to clients.
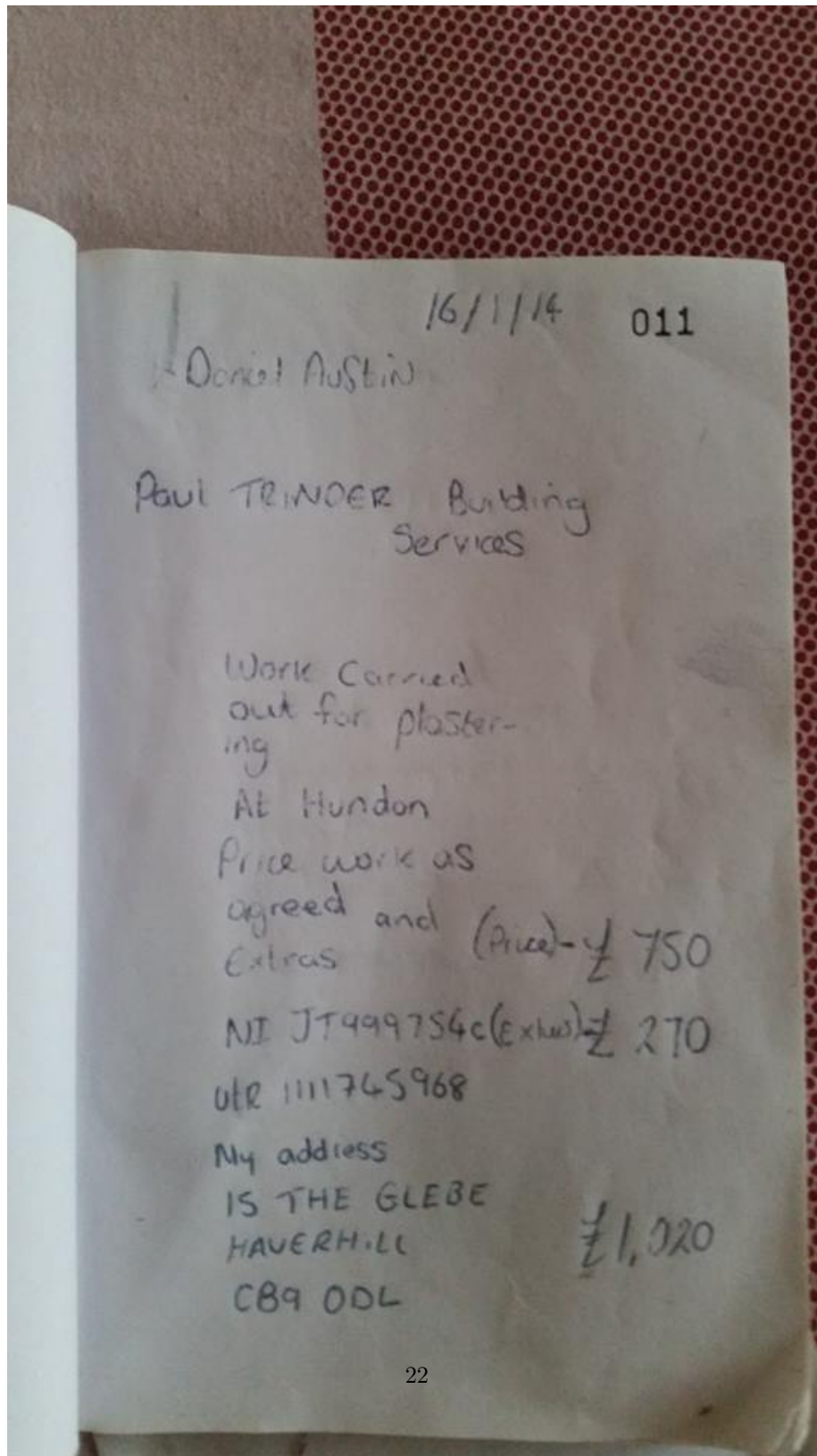
Figure 1.10: Here is an illustration of the type of information which goes on an invoice to the client.

16/7/2013 001

Daniel Austin

D. Mason Developments

Price work as
agreed invoice
up to date

work Carried
Out at
Toppersfield
on the Barn
total                    £ 2,860

1 day at
Chelmsford.
for Plastering  £ 140.00

£ 3,000,00

23
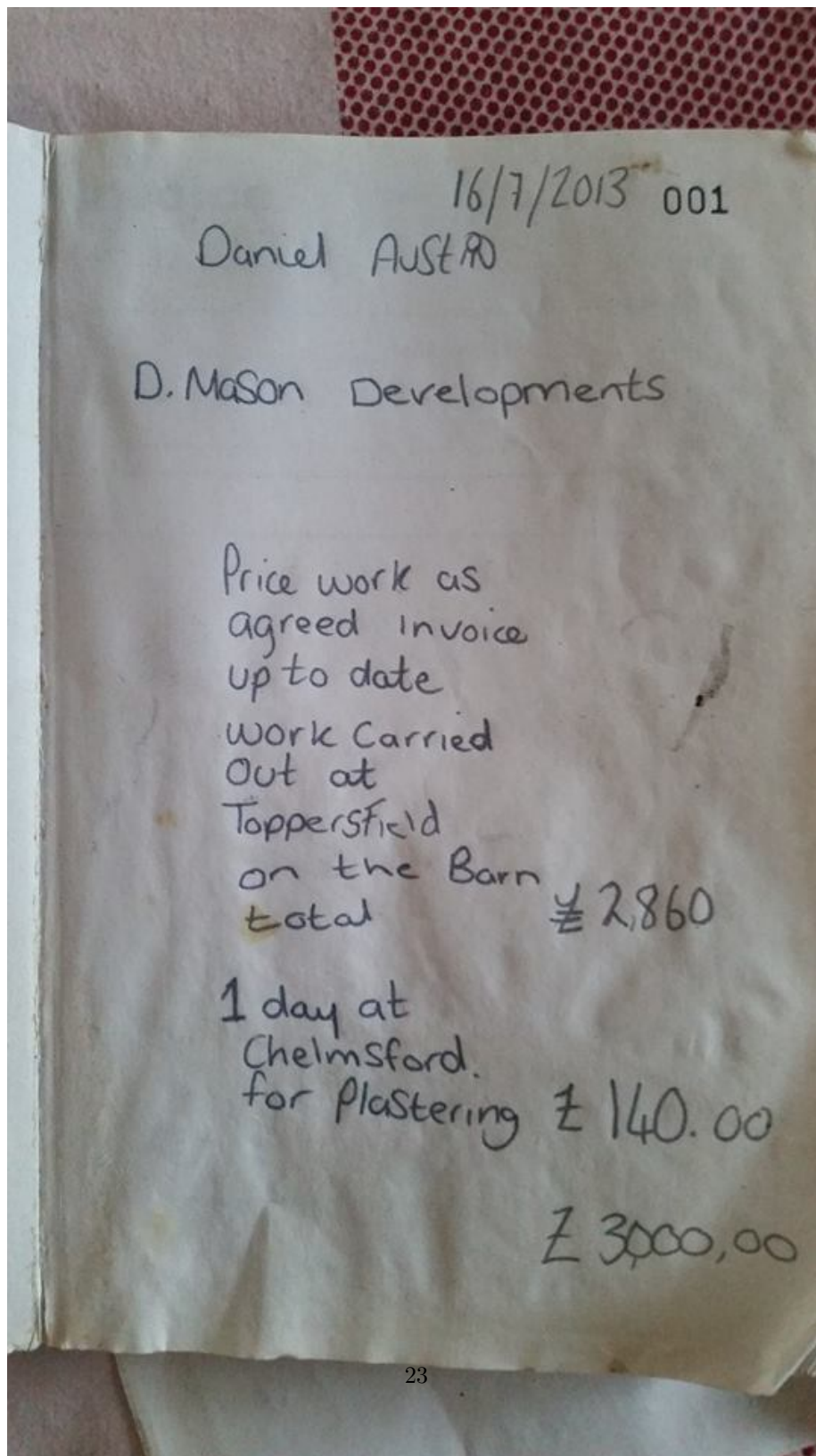
Figure 1.11: This is also another example of an invoice which is given to a client.

## 1.2.2   The proposed system

**Data sources and destinations**

| Source | Data | Data Type | Destination |
|--------|------|-----------|-------------|
| Client | Firstname | String/text | Plasterer |
| Client | Surname | String/text | Plasterer |
| Client | AddrLine1 | String/text | Plasterer |
| Client | AddrLine2 | String/text | Plasterer |
| Client | AddrLine3 | String/text | Plasterer |
| Client | AddrLine4 | String/text | Plasterer |
| Client | PostCode | String/text | Plasterer |
| Client | Email | String/text | Plasterer |
| Client | MobNumber | String/text | Plasterer |
| Plasterer | ClientID | Integer | Client Records |
| Plasterer | Firstname | String/text | Client Records |
| Plasterer | Surname | String/text | Client Records |
| Plasterer | AddrLine1 | String/text | Client Records |
| Plasterer | AddrLine2 | String/text | Client Records |
| Plasterer | AddrLine3 | String/text | Client Records |
| Plasterer | AddrLine4 | String/text | Client Records |
| Plasterer | PostCode | String/text | Client Records |
| Plasterer | Email | String/text | Client Records |
| Plasterer | MobNumber | String/text | Client Records |
| Job site | JobID | Integer | Job Records |
| Job site | ClientID | Integer | Job Records |
| Job site | Job Desc | Text | Job Records |
| Job site | AddrLine1 | String/text | Job Records |
| Job site | AddrLine2 | String/text | Job Records |
| Job site | AddrLine3 | String/text | Job Records |
| Job site | AddrLine4 | String/text | Job Records |
| Job site | PostCode | String/text | Job Records |
| Job site | Job Total Price | Currency | Job Records |
| Job site | JobPaid | Boolean | Job Records |
| Job site | JobDaysWorked | Integer | Job Records |
| Job site | JobComplete | Boolean | Job Records |
| Plasterer | AppointmentID | Integer | Appointment Records |
| Plasterer | ClientID | String/text | Appointment Records |
| Plasterer | PlastererID | String/text | Appointment Records |
| Plasterer | AppointmentDate | String/text | Appointment Records |

| Plasterer | AppointmentTime | String/text | Appointment Records |
|-----------|-----------------|-------------|---------------------|
| Plasterer | AppointmentAddr1 | String/text | Appointment Records |
| Plasterer | AppointmentAddr2 | String/text | Appointment Records |
| Plasterer | AppointmentAddr3 | String/text | Appointment Records |
| Plasterer | AppointmentAddr4 | String/text | Appointment Records |
| Plasterer | InvoiceID | Integer | Invoice Records |
| Plasterer | ClientID | Integer | Invoice Records |
| Plasterer | JobID | Integer | Invoice Records |
| Plasterer | PlastererID | Integer | Invoice Records |
| Plasterer | InvoiceAmountPreTax | Currency | Invoice Records |
| Plasterer | InvoiceAmountAfterTax | Currency | Invoice Records |
| Plasterer | InvoiceReceived | Boolean | Invoice Records |
| Plasterer | InvoiceDate | DateTime | Invoice Records |
| Plasterer | InvoiceText | Text | Invoice Records |
| Plasterer | JobMaterialsID | Integer | JobMaterials Records |
| Plasterer | JobID | Integer | JobMaterials Records |
| Plasterer | MaterialsID | Integer | JobMaterials Records |
| Plasterer | JobMaterialsQuantity | Integer | JobMaterials Records |

**Data flow diagrams**
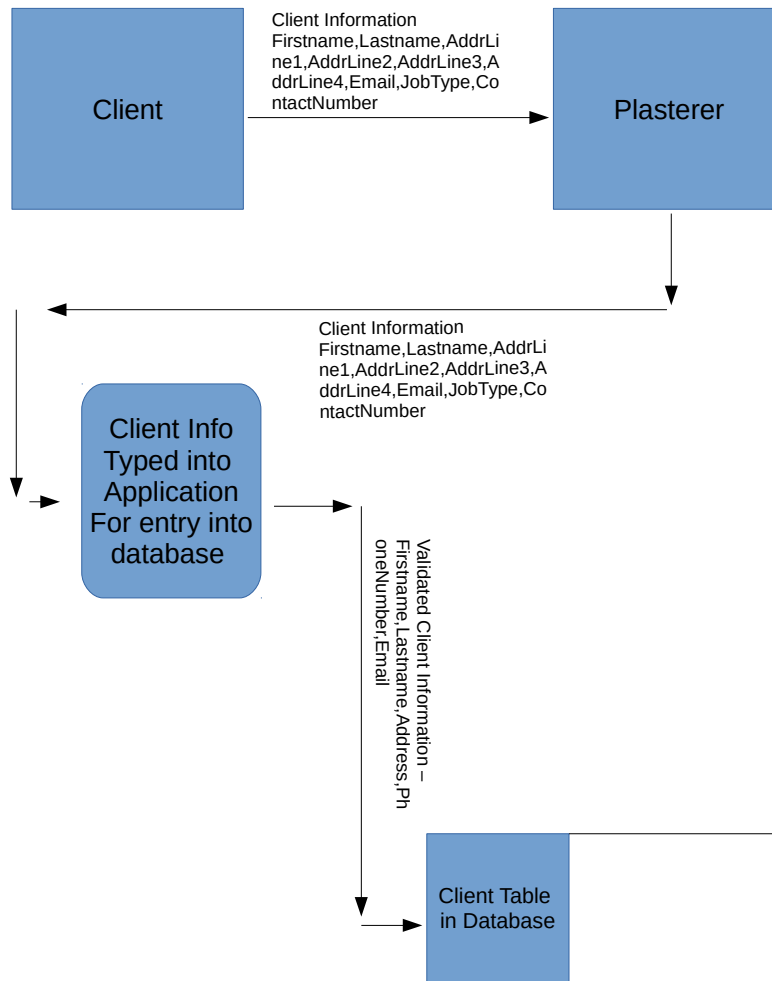


Figure 1.12: This data flow diagram signifies the flow of data in the proposed system when gaining a new clients info.
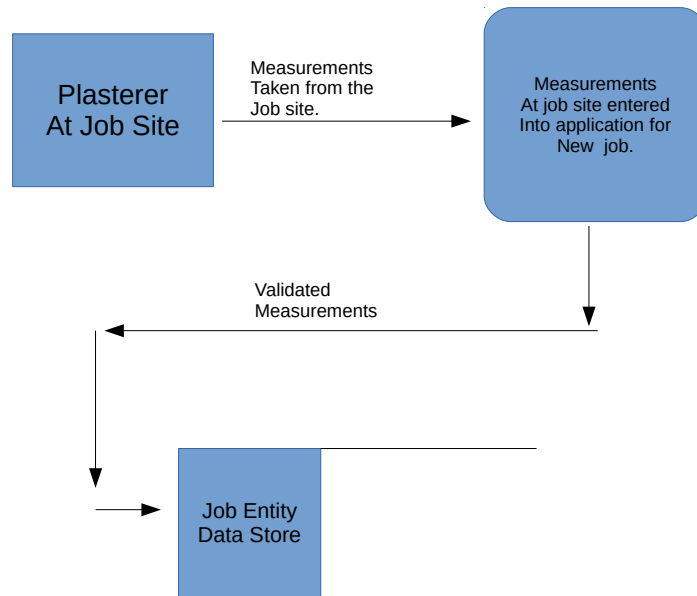
Figure 1.13: This data flow diagram signifies the flow of data in the proposed system when collecting the measurements from the job site.
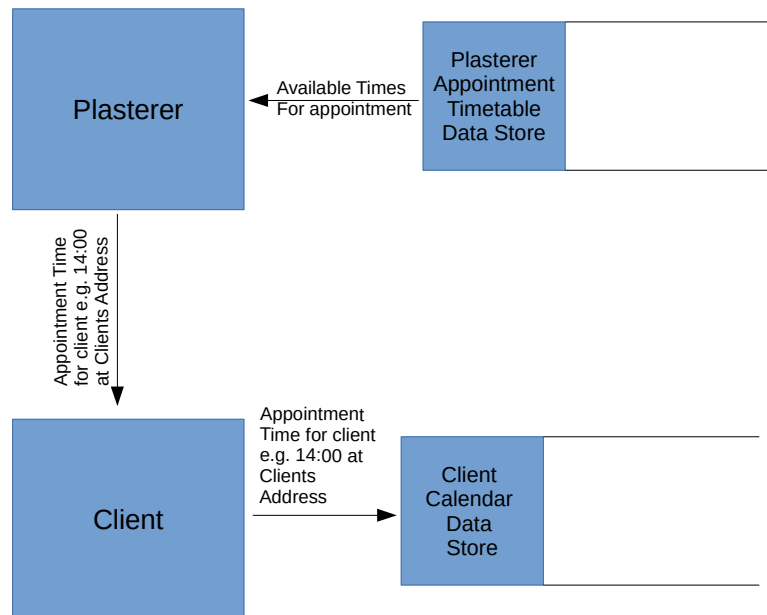
Figure 1.14: This diagram shows the flow of the data in the proposed system when a client is given an appointment time.

Job Price and details from
the Job Records in the
proposed system.

Job
Records

Job Price and
details from the
Job Records in
the proposed
system.

Plasterer

Work out how much
Plaster is needed
From the description and
Details from the Job
Records.

Details about the amount of
plaster and materials needed.

Details about the
amount of plaster and
materials needed.

Builders merchants
Process the materials
Data and work out their
Best quote for those
Materials.

Builders
Merchant

A Quote for the price
Of the materials
needed for a job.

Plasterer

Figure 1.15: This diagram shows the flow of the data in the proposed system
when the plasterer gets a quote for the materials from the builders merchants.

**Job Price and details from the Job Records in the proposed system.**

**Job Records**

**Plasterer**

Job Price and details from the Job Records in the proposed system.

Job prices and Details are Worked out to Form a quote For the client.

**Client**

Quote generated from the job details by the plasterer. Plasterer calls or meets client to tell them quote.
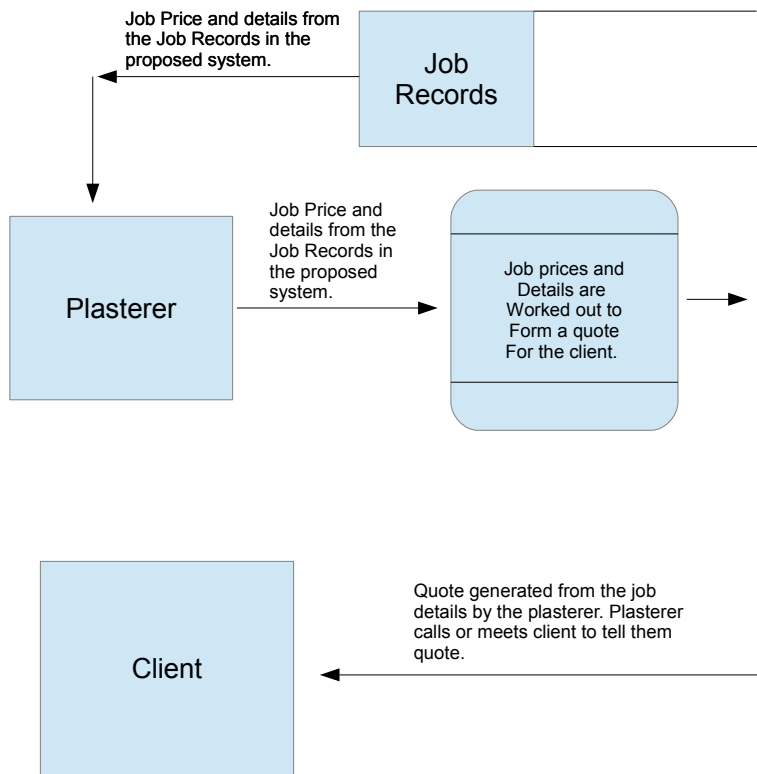
Figure 1.16: This diagram shows the flow of the data in the proposed system when a quote is generated for the client and given to them.

**Data dictionary**

| Name | Data Type | Length | Validation | Example Data | Comment |
|---|---|---|---|---|---|
| ClientID | Integer | 1 - 1000 | Range | 1, 200, 45 | Primary Key |
| ClientTitle | String | 2 - 10 Chars | Length | Mr,Mrs,Sir | |
| ClientFirstName | String | 3 - 25 Chars | Length | Dan, kyle | |
| ClientSurname | String | 3 - 35 Chars | Length | Austin, Kirkby | |
| ClientAddrLine1 | String | 5 - 30 Chars | Length | 15 Crowley Road | Street name |
| ClientAddrLine2 | String | 5 - 30 Chars | Length | Haverhill | Town / City |
| ClientAddrLine3 | String | 5 - 30 Chars | Length | Suffolk | County |
| ClientAddrLine4 | String | 6 - 7 Chars | Length and Format Check | CB9 0DJ | Post Code |
| ClientEmail | String | 7 - 50 Chars | Length and format check | dan@ gmail.com | |
| ClientPhoneNumber | String | 11 Chars | Length | 07809 726 812 | |
| PlastererID | Integer | 1 - 50 | Range | 1, 2, 45 | Primary Key |
| PlastererFirstName | String | 3 - 30 Chars | Length | Dan, Kyle | |
| PlastererSurname | String | 3 - 30 Chars | Length | Austin, Kirkby | |
| PlastererAddrLine1 | String | 5 - 30 Chars | Length | 15 Long Road | Street Name |
| PlastererAddrLine2 | String | 5 - 30 Chars | Length | Cambridge | Town / City |
| PlastererAddrLine3 | String | 5 - 30 Chars | Length | Essex | County |
| PlastererAddrLine4 | String | 6 - 7 Chars | Length and Format Check | CB2 5HD | Post Code |
| PlastererEmail | String | 7 - 50 Chars | Length and Format Check | dan@ gmail.com | |
| PlastererPhoneNumber | String | 11 Chars | Length | 07710 300 678 | |
| PlastererDailyRate | Currency | 40 - 250 | Range Check | 70, 150, 200 | |
| JobID | Integer | 1 - 2000 | Range Check | 1, 3, 5 | Primary Key |
| JobDescription | Text | 10 - 1000 Chars | Range Check | 5m x 4m x 3m Living Room to be plastered. | Description of job. |
| JobAddrLine1 | String | 5 - 30 Chars | Length | 15 Long Road | Street Name |

| JobAddrLine2 | String | 5 - 30 Chars | Length | Cambridge | Town / City |
|---|---|---|---|---|---|
| JobAddrLine3 | String | 5 - 30 Chars | Length | Essex | County |
| JobAddrLine4 | String | 6 - 7 Chars | Length and Format Check | CB2 5HD | Post Code |
| JobDaysWorked | Integer | 0 - 30 | Range Check | 1, 7 14 | |
| JobComplete | Boolean | | Presence Check | TRUE, FALSE | |
| JobPaid | Boolean | | Presence Check | TRUE, FALSE | |
| MaterialID | Integer | 1 - 50 | Range Check | 3, 4, 29 | Primary Key |
| MaterialName | String | 3 - 50 | Length | Angle Beading | |
| MaterialPrice | Currency | 1 - 1000 | Range Check | 1, 3, 20 | |
| JobMaterialsID | Integer | 1 - 4000 | Range Check | 1, 300, 3563 | Primary Key |
| JobMaterialsQuantity | Integer | 1 - 500 | Range Check | 59, 245, 309 | |
| InvoiceID | Integer | 1 - 2000 | Range Check | 1, 2, 45 | Primary Key |
| InvoiceAmountPreTax | Currency | 50 - 7000 | Range Check | 140, 890, 1050 | |
| InvoiceAmountAfterTax | Currency | 50 - 7000 | Range Check | 100, 800, 900 | |
| InvoiceReceived | Boolean | | Presence Check | TRUE, FALSE | |
| InvoiceDate | DateTime | | Presence Check | 14/12/2013 14:20 | |
| InvoiceText | Text | 1 - 1000 | Length | 5 Days Worked | |
| AppointmentID | Integer | 1 - 2500 | Range Check | 1, 2, 45 | Primary Key |
| AppointmentDate | Date | | Presence Check | 07/12/2013 | |
| AppointmentTime | Time | | Presence Check | 16:10 | |
| AppointmentAddrLine1 | String | 5 - 30 Chars | Length | 15 Long Road | Street Name |
| AppointmentAddrLine2 | String | 5 - 30 Chars | Length | Cambridge | Town / City |
| AppointmentAddrLine3 | String | 5 - 30 Chars | Length | Essex | County |
| AppointmentAddrLine4 | String | 6 - 7 Chars | Length and Format Check | CB2 5HD | Post Code |

**Volumetrics**

In the current system Dan only has a few regular clients but it is possible to receive up to 4-5 additional clients each week (mentioned in interviequestion 8). Therefore using the guidline of around 4 clients a week and a trial period for the application of 3 months, there could be upto 48 new clients (4 clients a week x 4 weeks in month x 3 months = 48) added into the database. It would also be useful to add the existing clients from paper so another 50 clients could be added from what is already stored manually. So around 100 clients may be added to the new system. This size can be increased later if necessary.

In the job details table each client may have up to 10 or more jobs. Some clients will only have one and some will be recurring customers so will have a few more. In the calculations for minimum storage space I will say that each field takes up 1KB of hard disk space and all together there will be around 50 related fields for each client so:

- 100 x 50KB = 5000 KB
- 5000 KB / 1024 = 4.8MB

s If the application took up around 5MB itself then the total space required for the proposed system would be around 9.8MB (4.8MB + 5MB = 9.8MB). Dans computer has plenty of hard disk space that could be used when installing the application. This proposed system would therefore have enough storage space for 100 clients to be added to the client database.

## 1.3   Objectives

### 1.3.1   General Objectives

- Clean and easy to use GUI.
- Use a database for storing data.
- Make it as easy as possible to find data.
- Add clients to database.
- Make it easy to calculate costs involved.
- Sort client data and add search functionality.
- Ability to add multiple jobs per client.

### 1.3.2   Specific Objectives

**Client Data Store Objectives**

- Ability for Dan to add a client to a database.

- Ability for Dan to be able to delete a client from the database if needed.

- Dan should be able to modify and append exisitng clients details with and easy to use system.

- A search feature that will let Dan filter the database of clients to dfind vital information.

**Jobs Objectives**

- Dan will be able to add a job which will relate to a client stored in the client entity.

- Multiple jobs can be added.

- The jobs will contain the job details (description) and address etc (see entity descriptions for more details).

- Each job will be able to generate invoices from the data.

- The invoice will be able to be sent to the client digitally(by email) or manually (print invoice).

- The user will have the ability to edit the details if needed for each job.

**Management features objectives.**

- Dan will be able to generate reports between x and y time periods in order to see the amount earned within that period.

- A feature will be implemented whereby the user can deduct tax and other specified costs from the amount earned within that period.

- Ability to print this pay report and generate a digital versatile copy of the report.

### 1.3.3   Core Objectives

- The application must store the client details in a database.

- The application must be able to add jobs for each client.

- The application must be able modify client and job details.

- The application must be able to send an invoice to the client via email.

- The application must be able to generate a report for the amount earned within a time period.

### 1.3.4   Other Objectives

- The application may be able to print an invoice for the client.

- The application may be able to send quotes for jobs via email.

- The application may be able to print a report for the amount earned within a time period.

## 1.4  ER Diagrams and Descriptions

### 1.4.1  ER Diagram
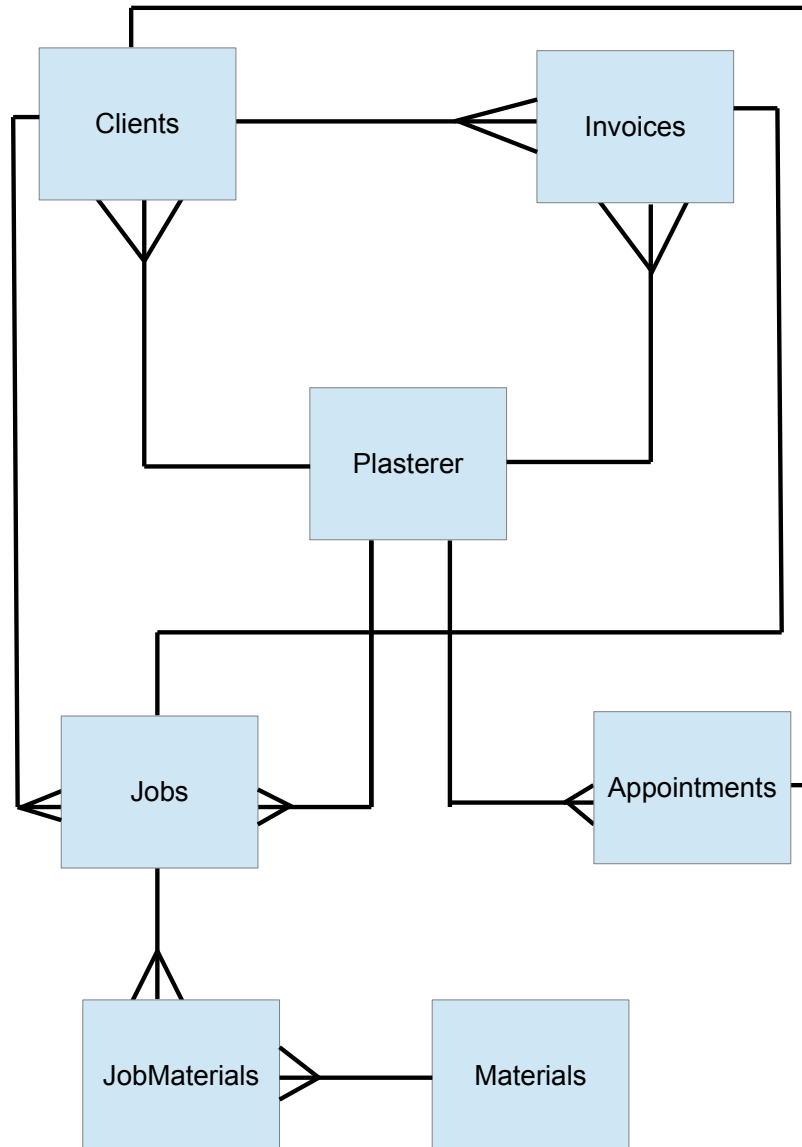


Figure 1.17: This is the entity relationship diagram for the sqlite3 database.

### 1.4.2   Entity Descriptions

Below are the entity descriptions for the various entites in the proposed system. An <u>underlined</u> attribute denotes a primary key and a *emphasised* attribute signifies a foreign key in the entity.

**Client**(<u>ClientID</u>, ClientTitle, ClientFirstName, ClientSurname, ClientAddrLine1, ClientAddrLine2, ClientAddrLine3, ClientAddrLine4, ClientEmail, ClientPhoneNumber, *PlastererID*)

**Plasterer**(<u>PlastererID</u>, PlastererFirstName, PlastererSurname, PlastererAddrLine1, PlastererAddrLine2, PlastererAddrLine3, PlastererAddrLine4, PlastererEmail,PlastererPhoneNumber, PlastererDailyRate)

**Job**(<u>JobID</u>, *ClientID*, *PlastererID*, JobDescription, JobAddrLine1, JobAddrLine2, JobAddrLine3, JobAddrLine4, JobDaysWorked, JobComplete, JobPaid, *InvoiceID*)

**Material**(<u>MaterialID</u>,MaterialName,MaterialPrice)

**JobMaterials**(<u>JobMaterialsID</u>, *JobID*, *MaterialsID*, JobMaterialsQuantity)

**Invoice**(<u>InvoiceID</u>, *ClientID*, *JobID*, *PlastererID* InvoiceAmountPreTax, InvoiceAmountAfterTax, InvoiceReceived, InvoiceDate, InvoiceText)

**Appointment**(<u>AppointmentID</u>, *ClientID*, *PlastererID*, AppointmentDate, AppointmentTime, AppointmentAddrLine1, AppointmentAddrLine2, AppointmentAddrLine3, AppointmentAddrLine4)

## 1.5   Object Analysis

### 1.5.1   Object Listing

**Objects**

- Client
- Jobs
- Materials
- JobsMaterials
- Invoices
- Appointment
- Plasterer

### 1.5.2 Relationship diagrams
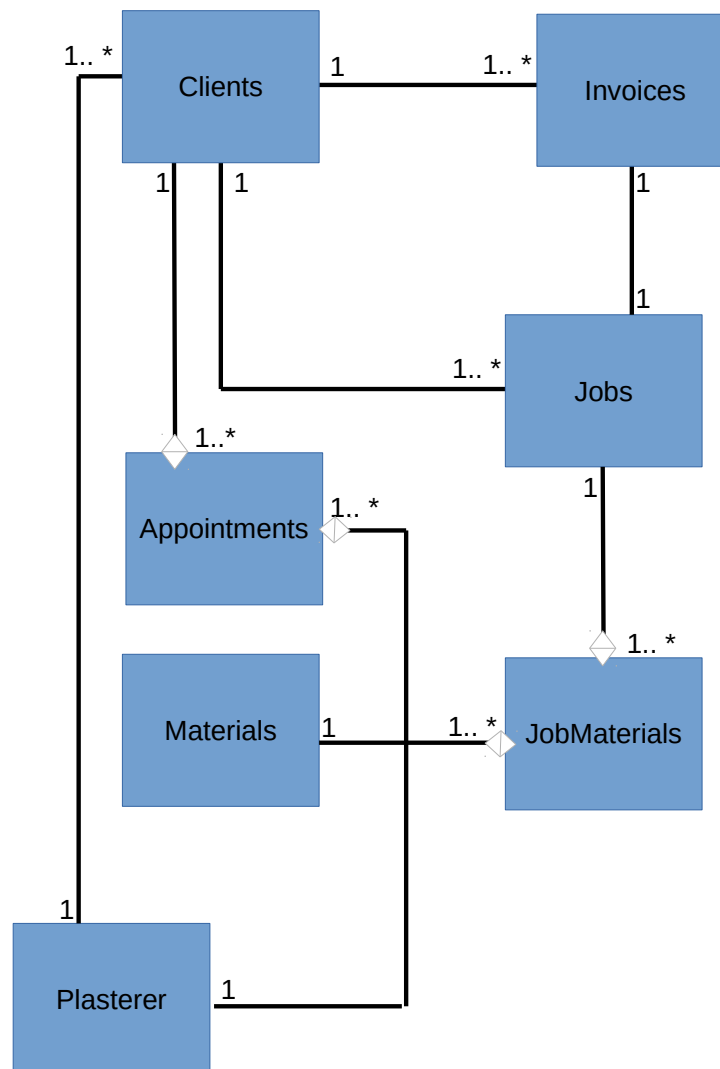


Figure 1.18: Relationship Diagram for the entities in the proposed system.

### 1.5.3　Class definitions

**Key:**

| Label |
| --- |
| Attributes |
| Behavious |

| Client: |
| --- |
| ClientID |
| ClientTitle |
| ClientFirstName |
| ClientSurname |
| ClientAddrLine1 |
| ClientAddrLine2 |
| ClientAddrLine3 |
| ClientAddrLine4 |
| ClientEmail |
| ClientPhoneNumber |
| AddClientTitle |
| AddClientFirstName |
| AddClientSurname |
| AddClientAddrLine1 |
| AddClientAddrLine2 |
| AddClientAddrLine3 |
| AddClientAddrLine4 |
| AddClientEmail |
| AddClientPhoneNumber |
| EditClientTitle |
| EditClientFirstName |
| EditClientSurname |
| EditClientAddrLine1 |
| EditClientAddrLine2 |
| EditClientAddrLine3 |
| EditClientAddrLine4 |
| EditClientEmail |
| EditClientPhoneNumber |

| **Plasterer:** |
| --- |
| PlastererID |
| PlastererFirstName |
| PlastererSurname |
| PlastererAddrLine1 |
| PlastererAddrLine2 |
| PlastererAddrLine3 |
| PlastererAddrLine4 |
| PlastererEmail |
| PlastererPhoneNumber |
| PlastererDailyRate |
| AddPlastererFirstName |
| AddPlastererSurname |
| AddPlastererAddrLine1 |
| AddPlastererAddrLine2 |
| AddPlastererAddrLine3 |
| AddPlastererAddrLine4 |
| AddPlastererEmail |
| AddPlastererPhoneNumber |
| AddPlastererDailyRate |
| EditPlastererFirstName |
| EditPlastererSurname |
| EditPlastererAddrLine1 |
| EditPlastererAddrLine2 |
| EditPlastererAddrLine3 |
| EditPlastererAddrLine4 |
| EditPlastererEmail |
| EditPlastererPhoneNumber |
| EditPlastererDailyRate |

| Job: |
| --- |
| JobID |
| ClientID |
| PlastererID |
| JobDescription |
| JobAddrLine1 |
| JobAddrLine2 |
| JobAddrLine3 |
| JobAddrLine4 |
| JobDaysWorked |
| JobComplete |
| JobPaid |
| InvoiceID |
| AddJobDescription |
| AddJobAddrLine1 |
| AddJobAddrLine2 |
| AddJobAddrLine3 |
| AddJobAddrLine4 |
| AddJobDaysWorked |
| EditJobComplete |
| EditJobPaid |
| EditJobDescription |
| EditJobAddrLine1 |
| EditJobAddrLine2 |
| EditJobAddrLine3 |
| EditJobAddrLine4 |
| EditJobDaysWorked |

| Material: |
| --- |
| MaterialID |
| MaterialName |
| MaterialPrice |
| AddMaterialName |
| AddMaterialPrice |
| EditMaterialName |
| EditMaterialPrice |

| JobMaterials: |
| --- |
| JobMaterialsID |
| JobID |
| MaterialsID |
| JobMaterialsQuantity |
| AddJobMaterialsQuantity |
| EditJobMaterialsQuantity |

| **Invoice:** |
| --- |
| InvoiceID |
| ClientID |
| JobID |
| PlastererID |
| InvoiceAmountPreTax |
| InvoiceAmountAfterTax |
| InvoiceReceived |
| InvoiceDate |
| InvoiceText |
| AddInvoiceAmountPreTax |
| AddInvoiceAmountAfterTax |
| AddInvoiceReceived |
| AddInvoiceDate |
| AddInvoiceText |
| EditInvoiceAmountPreTax |
| EditInvoiceAmountAfterTax |
| EditInvoiceReceived |
| EditInvoiceDate |
| EditInvoiceText |

| **Appointment:** |
| --- |
| AppointmentID |
| ClientID |
| PlastererID |
| AppointmentDate |
| AppointmentTime |
| AppointmentAddrLine1 |
| AppointmentAddrLine2 |
| AppointmentAddrLine3 |
| AppointmentAddrLine4 |
| AddAppointmentDate |
| AddAppointmentTime |
| AddAppointmentAddrLine1 |
| AddAppointmentAddrLine2 |
| AddAppointmentAddrLine3 |
| AddAppointmentAddrLine4 |
| EditAppointmentDate |
| EditAppointmentTime |
| EditAppointmentAddrLine1 |
| EditAppointmentAddrLine2 |
| EditAppointmentAddrLine3 |
| EditAppointmentAddrLine4 |

## 1.6   Other Abstractions and Graphs

## 1.7   Constraints

### 1.7.1   Hardware

Dan currently has a Toshiba Laptop with the following hardware components:

- 6GB DDR3 RAM

- Intel 2.0ghz Dual Core Processor

- High Resolution 1920 x 1080 Display

- 1 TB HDD

- Intel Onboard Integrated Graphics

The specification of Dan's laptop is more than powerful enough and has
enough RAM to run the proposed python application alongside multiple other
programs that he regularly uses (such as a Web Browser and Media Player).
The hard drive has enough storage space to install the application (which will
only be around 10MB) so storage is not a problem. The only possible
complication when it comes to hardware may be the screen resolution as the
application will have to keep within the resolution and be optimized for this
screen size and many more to keep the software versatile.

### 1.7.2   Software

Dan is running the Windows 8 operating system on his laptop. The proposed
system will need to be able to run on this operating system which is not a
problem as the proposed system can easily be packaged into a installation
executable which he will be able to run and install the proposed system. This
will remove the constraint of not having python natively installed on the
system.

### 1.7.3   Time

Dan is very flexible when it comes to time as he is a freelancer and is in no
rush for a replacement to his old system. He does not mind how long the new
system takes to be implemented just that it works and functions as expected.
Therefore the only time constraint for the proposed system would be the
coursework deadline which is Friday 27th March 2015.

### 1.7.4   User Knowledge

Dan has no formal qualifications in IT and has never performed complex computerised tasks. Browsing the internet, watching films and playing music is the extent to which Dan currently uses his laptop. Therefore user knowledge is a possible constraint and the proposed system will need to be easy to navigate and use. This may be achieved by providing a simple non-complex GUI and detailed tutorials/documentation.

### 1.7.5   Access restrictions

The proposed system will not need additional security measures as no one else will be using the system and Dan is the only person that will use the application so different user accounts will not be needed.

## 1.8   Limitations

### 1.8.1   Areas which will not be included in computerisation

- The collection of the measurements from the job site will still have to be done manually so will not be included in computerisation.

- Getting the quote for the price of the materials will still be done manually as it will be difficult to computerise as you have to visit the Builders Merchants in person.

### 1.8.2   Areas considered for future computerisation

- The collection of the measurements, which is currently being done manually, may be implemented using a device which has kinect like sensors, you could scan a room and pick up the measurements of it using just a phone for example. This might be possible.

## 1.9   Solutions

### 1.9.1   Alternative solutions

| Solution | Advantages | Disadvantages |
|---|---|---|
| Web Application | Can be accessed on multiple devices where theres is an internet connection. Only a web browser required (already installed in most cases). Can be easier to make it look "pretty" as there are many graphical libraries for the web. Can use different server side languages to program the backend of the website. (Python (Django),Ruby(RoR) ,PHP(Zend,Magneto)). | Need an internet connection to visit site if not hosted locally. A web host may need to be purchased. Need to optimize for different browsers (Internet Explorer, Chrome, Firefox, Safari). Problems may happen with host which means the client may not be able to access their application (site may be subject to DDOS (may need to purchase Cloudflare or similiar service)). |
| Mobile Phone Application | Can mean the application is extremely mobile so the client will be able to take the app wherever he goes which may useful to take to jobs. Client is familiar with using a mobile phone so may be easier for them to use. | Will take longer to complete. It is more complex and can be solved easier with a different method. Client may lose phone which would then result in loss of application. |
| Command-Line Application | No long waiting time for GUI to load. System resources are not used up as much. | Can be more complicated, so it seems to the client. Interface would be less effective and harder to use for the non-technical client. Errors may occur and look unfamiliar to client. If laptop is lost/stolen the the program may be lost too so backups need to be kept online (not so much a problem with a web application). |

| Spreadsheet | Can be faster than a application with a GUI to load. Data can be exported easily. Queries are easy to run and execute fast. | No abstraction from the data. Client would have to familiarise themselves with the syntax for queries. You can add a lot more features in PyQt and Python Applications. No GUI to make it easier for the client to control the functions and run queries. |
|---|---|---|
| PyQt4 GUI and Python3 Application | Python is easy to write and can therefore get the project done quicker leaving more time for debugging and testing. The GUI will make it a lot easier for the client to use the application. With this form of application it can be packaged for Mac,Windows and Linux Systems. GUI gives a clear visual representation of the applications features. | Uses python which needs to be packaged with the program as windows does not have Python natively installed but Mac and Linux do. Writing a GUI is a lot more work than other solutions. |

### 1.9.2   Justification of chosen solution

I have chosen to use Python 3 alongside the Qt Framework (PyQt4) to create a standalone application. This is due to the versatility which Python and PyQt4 offers. Using this method to program a solution to my clients problem I will be able to get it done effeciently due to my existing knowledge of Python and I will be able to generate different executables for different operating systems which may be beneficial to the client as he is looking to buy a Apple System. It will also offer a visual representation through the use of GUI as opposed to hard to use Console application approach; which would not be suitable for the client as he does not have enough technical knowledge with computers to understand and use it easily - therefore going for something with a GUI would be a good choice. A web application may have been good but

sometimes Dan takes his laptop where there is no internet connection and hosting the site locally would not be feasible. Also there are hidden costs involved with purhcasing a domain and hosting if a web application solution was chosen. A mobile application may have also been good but would have taken longer to accomplish due to the time it takes to develop complex mobile phone applications. A spreadsheet would also not have been the perfect solution as it requires knowledge of writing queries which, due to Dan's lack of technical computer knowledge, he does not have. Therefore I believe that a Python and PyQt4 application would be the best solution to go for.

# Chapter 2

# Design

## 2.1   Overall System Design

### 2.1.1   Short description of the main parts of the system

### 2.1.2   System flowcharts showing an overview of the complete system

## 2.2   User Interface Designs

## 2.3   Program Structure

### 2.3.1   Top-down design structure charts

### 2.3.2   Algorithms in pseudo-code for each data transformation process

### 2.3.3   Object Diagrams

### 2.3.4   Class Definitions

## 2.4   Prototyping

## 2.5   Definition of Data Requirements

### 2.5.1   Identification of all data input items

### 2.5.2   Identification of all data output items

### 2.5.3   Explanation of how data output items are generated

### 2.5.4   Data Dictionary

### 2.5.5   Identification of appropriate storage media

### 2.9.1 Outline Plan

| Test Series | Purpose of Test Series | Testing Strategy | Strategy Rationale |
|---|---|---|---|
| Example | Example | Example | Example |

### 2.9.2 Detailed Plan

| Test Series | Purpose of Test | Test Description | Test Data | Test Data Type (Normal/ Erroneous/ Boundary) | Expected Result | Actual Result | Evidence |
|---|---|---|---|---|---|---|---|
| Example | Example | Example | Example | Example | Example | Example | Example |

# Chapter 3

# Testing

## 3.1   Test Plan

### 3.1.1 Original Outline Plan

| Test Series | Purpose of Test Series | Testing Strategy | Strategy Rationale |
|---|---|---|---|
| Example | Example | Example | Example |

### 3.1.2 Changes to Outline Plan

| Test Series | Purpose of Test Series | Testing Strategy | Strategy Rationale |
|---|---|---|---|
| Example | Example | Example | Example |

### 3.1.3 Original Detailed Plan

| Test Series | Purpose of Test | Test Description | Test Data | Test Data Type (Normal/ Erroneous/ Boundary) | Expected Result | Actual Result | Evidence |
|---|---|---|---|---|---|---|---|
| Example | Example | Example | Example | Example | Example | Example | Example |

### 3.1.4 Changes to Detailed Plan

| Test Series | Purpose of Test | Test Description | Test Data | Test Data Type (Normal/ Erroneous/ Boundary) | Expected Result | Actual Result | Evidence |
|---|---|---|---|---|---|---|---|
| Example | Example | Example | Example | Example | Example | Example | Example |

## 3.2  Test Data

### 3.2.1  Original Test Data

### 3.2.2  Changes to Test Data

## 3.3  Annotated Samples

### 3.3.1  Actual Results

### 3.3.2  Evidence

## 3.4   Evaluation

### 3.4.1   Approach to Testing

### 3.4.2   Problems Encountered

### 3.4.3   Strengths of Testing

### 3.4.4   Weaknesses of Testing

### 3.4.5   Reliability of Application

### 3.4.6   Robustness of Application

# Chapter 4

# System Maintenance

## 4.1 Environment

### 4.1.1 Software

### 4.1.2 Usage Explanation

### 4.1.3 Features Used

## 4.2 System Overview

### 4.2.1 System Component

## 4.3 Code Structure

### 4.3.1 Particular Code Section

## 4.4 Variable Listing

## 4.5 System Evidence

### 4.5.1 User Interface

### 4.5.2 ER Diagram

### 4.5.3 Database Table Views

### 4.5.4 Database SQL

### 4.5.5 SQL Queries

## 4.6 Testing

### 4.10.1 Module 1

# Chapter 5

# User Manual

## 5.1 Introduction

## 5.2 Installation

### 5.2.1 Prerequisite Installation

**Installing Python**

**Installing PyQt**

**Etc.**

### 5.2.2 System Installation

### 5.2.3 Running the System

## 5.3 Tutorial

### 5.3.1 Introduction

### 5.3.2 Assumptions

### 5.3.3 Tutorial Questions

**Question 1**

**Question 2**

### 5.3.4 Saving

### 5.3.5 Limitations

## 5.4 Error Recovery

# Chapter 6

# Evaluation

## 6.1  Customer Requirements

### 6.1.1  Objective Evaluation

## 6.2  Effectiveness

### 6.2.1  Objective Evaluation

## 6.3  Learnability

## 6.4  Usability

## 6.5  Maintainability

## 6.6  Suggestions for Improvement

## 6.7  End User Evidence

### 6.7.1  Questionnaires

### 6.7.2  Graphs

### 6.7.3  Written Statements