EDUQAS

A Level Computer Science

Harris Samuel Bowman

Woldgate School

Candidate No.: 2024

Centre No.: 44219

Content

[1. Discussion 3](#_Toc498075874)

[1.1. Description of Project 3](#_Toc498075875)

[1.1.1. Introduction 3](#_Toc498075876)

[1.1.2. Aims 3](#_Toc498075877)

[1.1.3. Problems that could arise 3](#_Toc498075878)

[1.2. User requirements 4](#_Toc498075879)

[2. Investigation 4](#_Toc498075880)

[2.1. Investigation Plan 4](#_Toc498075881)

[2.2. Questionnaire for client 4](#_Toc498075882)

[2.2.1. Questionnaire Questions 5](#_Toc498075883)

[2.2.2. Results 6](#_Toc498075884)

[2.2.3. Summary of results 6](#_Toc498075885)

[2.3. Current system 6](#_Toc498075886)

[2.3.1. Description of the current system 6](#_Toc498075887)

[2.3.2. Advantages 6](#_Toc498075888)

[2.3.3. Disadvantages 6](#_Toc498075889)

[2.4. Investigation into other systems 7](#_Toc498075890)

[3. Design 8](#_Toc498075891)

[3.1. General Features Needed 8](#_Toc498075892)

[3.2. System Map 8](#_Toc498075893)

[3.2.1. Full Map 8](#_Toc498075894)

[3.2.2. Overall Summary 9](#_Toc498075895)

[3.2.3. Justification 9](#_Toc498075896)

[3.3. Forms Map 9](#_Toc498075897)

[3.3.1. Login 9](#_Toc498075898)

[3.3.2. Change Password 9](#_Toc498075899)

[3.3.3. Props and Costume 9](#_Toc498075900)

[3.3.4. Timetable/Room booker 9](#_Toc498075901)

[3.3.5. Resource Tracker 9](#_Toc498075902)

[3.3.6. Account Management/Preferences 9](#_Toc498075903)

[3.3.7. Admin Control Panel 9](#_Toc498075904)

[4. Prototyping 9](#_Toc498075905)

[5. Refinement of Design 9](#_Toc498075906)

[6. Software Development 9](#_Toc498075907)

[7. Testing 9](#_Toc498075908)

[8. Evaluation 9](#_Toc498075909)

# Discussion

## Description of Project

### Introduction

The client of this project is Hellen Bartlett who teaches the dramatic arts at Woldgate School and sixth form collage. She teaches over 1000 students between 11 years of age and 18 years of age, Drama at Key Stage 3 level, General Certificate of Secondary Education level and Advanced Level. To teach these lessons, Mrs Bartlett uses a mixture of teaching resources physical and digital, these include: PowerPoints, Music, Printouts (For herself and students), Props, and Online Videos. Computers are becoming an ever increasing major teaching resource in all departments in the school and Mrs Bartlett wants a system that can majorly benefit the quality of teaching in the drama department at Woldgate.

### Aims

The main three aims of this project are to:

Create a system where teachers can store lesson plans and resources in a database, can access them on any computer, add / edit / delete them from any computer and export the lesson resources to a local storage location like the PC currently in use, or a network location like a local user area. The lessons include homework where students can log in and see / download / print the resources needed to do the homework.

Create a booking system where students and teachers can book rooms throughout the school for use in drama. Students can view a timetable of when certain rooms are available and when there’s a lesson in the room.

A Prop tracking system so teachers can track and view what props they have, mark props as broken or lost so that if applicable new ones can be bought in future. Knowing what props a classroom has can be helpful when a certain performance or lesson may need them.

There will be a permissions system where teachers of the drama department and extended arts department can log in with a username and password so they can have extended features unavailable to students such as edit / delete / replace / and set homework.

When browsing the database, the user can search for the title of the resource, tags of the resource (such as what subject it is related to,) date of the video resource, date when the resource was added, type of resource (Eg. clothing or physical prop) and teaching level of resource (Eg. KS3 year 7, GCSE Year 11.)

### Problems that could arise

One of the problems that could arise in this system is if an item / room is taken without updating the system. This creates un- synchronisation between real like and the database and could create clashes so it is essential that students and teachers can easily understand the interface and how to use it.

Another problem that could arise is that students could use or move props and costume without entering the changes into the system which is also an issue of de-synchronisation.

Unstable code could stop a teacher from gathering resources for their lessons if the program keeps crashing and losing their work when an action is performed. This problem could also arise when the room booking system is being used and the program fails to do its intended task and breaks.

## User requirements

The layman user need an interface which is easy to understand and a strong and stable system that always works within reason. When I say within reason I mean when all computers in the school are down the layman user would not expect the system to still work.

The staff in the Drama department need to be able to access and edit the database efficiently and reliably to ensure all information in the database is accurate and up to date. This is to ensure no other lower permission users are effected by a human error in the database. The staff will also need to be able to easily add classes of students from the schools SIMS database, to make it easier to set homework for a whole class of students at a time. Another requirement is to quickly export and import lesson plans and lesson resources, including online video links, offline video links, pictures on and offline, PowerPoints and written lesson plans. A way to do this is to allow staff to pre compile resources in the program and when ready to do a lesson, export a folder of resources to a memory stick or a location on the school’s network where it can be easily accessed. One thing that will need to happen is resources should be in a format both Microsoft Windows 7 PCs can read and Apple Mac OSX PCs can read, this is because in one of the drama rooms (E1 specifically) is a Macintosh computer that is used frequently in lessons.

The students using the program will require an easy to understand interface for fast and easy use of the program, they also need printing facilities so they can print off homework with the default Microsoft Windows 7 interface. In regards to the booking system, lower school students should be able to request a room, held in a queue for teacher approval.

# Investigation

## Investigation Plan

The first thing to do is to create a questionnaire for the teachers I the drama department. This is to confirm what aspects and features of the program are most important to them and what would be most helpful to the department and what features they think could also help. This ensures I create a feature rich program and I meet all their needs. Next, I will look at other systems, analyse what they are, how they work, and how I can incorporate some of their ideas into my system. This is so my program can be as effective in its main job as it can and to create a coherent successful program while looking at other popular successful programs.

## Questionnaire for client

To increase the effectiveness of the project in meeting the client’s needs and to add / remove any points of the system that they may find useful / not necessary. Also, the questionnaire can be used to find what parts of the system need the most attention to during the project, for example, if the teachers really wanted the resource tracking, that will be the section of the system that would gain most attention. All parts of the system will be developed to a high standard though, it just means that a lot of coding and feature detail will go into the part that the clients choose.

### Questionnaire Questions

The main points I am trying to discover with this project are what part of the system is most important to the subject, what type of resource is the highest priority that needs to be stored in a database (props, online videos, offline videos, lesson plans or music etc.) Another question am interested in hearing is are they satisfied with the current room situation? As most rooms are already booked for lessons and if they want a system so that any student can log in and see if the room is available for a lunch rehearsal / six formers for a free period rehearsal.

The standard for questionnaires is to have a multiple choice questions. So for each of these questions the options will be “Strongly agree,” “Agree,” “Neither / don’t know,” “Disagree” and “Strongly disagree.”

1. The most important part of the system should be online (Video Links, Music Links) resource tracking.
2. The most important part of the system should be offline (Lesson plans, PowerPoints) resource tracking.
3. The most important part of the system is physical (Props, Costume) resource tracking.
4. Searching for online resources is annoying and slows down lessons.
5. The current management of music / videos is unorganised.
6. Allowing students to book E1 / J17 / J3 and view when those rooms are being used by a lesson or a lunch rehearsal is a helpful feature.
7. Tracking what props and costume the department owns is a helpful feature.
8. Storing homework such as quizzes and/or online links to homework/homework related sites is a helpful feature.

### Results

Raw Data:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Question # | St. Agree | Agree | DN | Disagree | St. Disagree |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |
| 8 |  |  |  |  |  |

### Summary of results

blah

## Current system

### Description of the current system

The current system is a non-recorded free for all, where students and teachers can freely take and use props, costumes and rooms throughout the school. No records are taken on what props/costumes are taken, used, where they are stored or used, when they were taken, who they were taken by. Rooms are taken sometimes without adult consent and supervision, any student in the rooms can take any prop they want.

In regards to lessons and homework. Currently teachers create a lesson plan and print it out, before the lesson they load up videos on the computer in the classroom in multiple tabs in a web browser. Specifically in the room E1 with the Macintosh computer, music is loaded up in Apple ITunes in an assorted file system where songs and albums are stored in multiple places on the PC, and the school network. Students get asked to research or do homework in their books. Sometimes a certain web resource needs to be given to students where sometimes they struggle to find it online.

### Advantages

The advantages of this system is its fast and extremely flexible. There is no need to track any props as most students can be trusted with them and they rarely get stolen, just lost. Most of the time online videos can be easily remembered and found if only one computer is used. Also, finding a prop or costume that is needed for a certain type of performance is not necessarily time consuming or hard.

### Disadvantages

One of the biggest struggles of the drama department is the storage of online resources used in drama lessons. This is the initial problem that led me to decide to develop this system. Staff in the drama department rely on remembering websites and video names for the lesson. Most of the time this works, but videos can be deleted online, lost, renamed or staff / students can just forget what they’re called. Downloaded music and videos are scattered throughout the computer on E1 (The main drama room for lessons.) It’s unorganised and is a reason why it is one of the three major parts of the system.

Another disadvantage of the system, room booking in particular is that it is “free for all” as in students can come in and use rooms at any time they want. This creates a problem because as one drama student doing their GCSEs told me, it’s annoying when they make a rehearsal schedule on paper and then too many groups turn up to rehearsal in one room, so they have to spend 5 minutes walking across school looking for a new room to rehearse in and this wastes time. Another example is if an A Level students’ needs some emergency rehearsal space during free periods, they need to know beforehand which rooms are free and which are taken up by a lesson, easily resolved with a system that can display when lessons are on and when rooms are booked for rehearsal.

## Investigation into other systems

One system that I heavily am considering using in my larger system is a barcode tracking system where props have an assigned ID or code which can be either entered by the user or scanned with a barcode scanner. This ID can be stored in a table that is related to other tables so multiple parts of the program can access this id and manipulate it / run procedures on it / link to other tables.

Woldgate School’s current room booking system is custom made for the school by a previous A-Level student. Teachers can log in, book out rooms, see what rooms are available and delete bookings of rooms. This system is adequate for most teacher’s needs, however not for my client as this system has no links to the resources available in the room, only available for computer rooms, and the program used is slow and unstable. To implement this in my system it will have to be its own separate table linked to other tables in the database and it will need to run faster than the program the school uses right now to book computer rooms, and of course, use Drama rooms instead of computer rooms when creating the database.

For the database of online resources like music and videos, I researched a popular media collection management and streaming application called Plex. Plex has two parts, the server and the client, the server can be pointed to a collection of videos and music files and organise them with metadata (Like name, artist, album, composer, album artist, genre, year/date, ratings, reviews, cast, crew, studio) and cover images pulled from the files themselves (Like the year of the movie in the file name or the artist from the ID3 tag of an mp3 file.) This is stored in a multitude of xml tables. The client can then connect to the server on the local network or through the internet and watch/listen to these videos and music presented in a clean interface with inspiration from Netflix and Amazon Prime Video. This is an over simplification of the program but it explains what parts I am going to use for my system. The analysing of video and music files and adding metadata into a table so the information about the files can be presented in a clean way, and can add to the simplicity of search query’s and filters for locating specific resources.

# Design

The design section is where I will be designing the general and slightly in depth parts of the program. These include the structure of forms in the program, structure of the modules in the program and how they connect with each other and the structure of the subroutines within the modules of the program.

The reason why the design section of the program is so important is because to create a program with this scale and complexity it is humanly impossible to code a program that big in one go without planning. Repeating and redundant code will be apparent, bugs will take longer to figure out and fix and the stability of the program will be compromised.

Splitting the code of the program into modules, and modules into subroutines which can be each individually explained will help when a developer comes to code this program later and needs to upgrade/edit the code if needed.

## General Features Needed

This is a general overview of the objects the databases of the program:

Props and costume:

* A list of all props and costume available
* The room each prop or costume is in
* The unique ID of the prop or costume
* A link to the subject the prop or costume relates to

Drama rooms:

* Rooms available to book
* When the rooms are available
* The unique ID of each room
* How many students the room can hold
* Lessons going on in those rooms and when they are

Resources for the Drama department:

* A big table of the available resources
* Unique ids for all resources
* Identifier for the type of resource
* Tags that the user can input that relate to the resource
* The profanity level of the resource
* A link to the subject the resource belongs to.

The Subjects:

* Every subject covered in the drama curriculum from Year 7 to A-Level
* A unique id that links to the other parts of the system.

The login identities of the Staff:

* Username
* Encrypted Password

## System Map

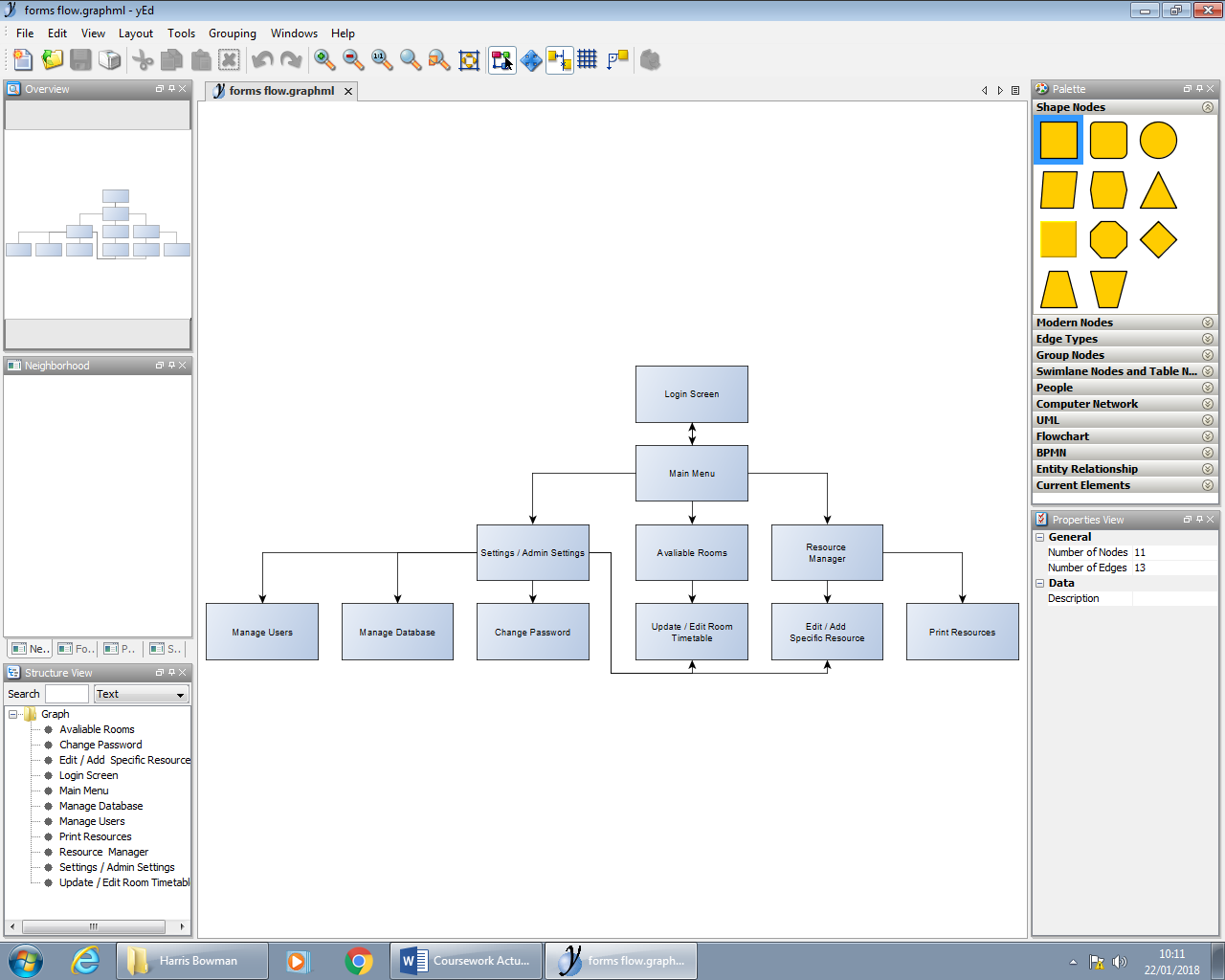
### Full Map

### Overall Summary

### Justification

## Forms Map

The forms map shows the flow of forms in the developing program. The program opens to the login screen, then the user continues onto the main menu.



This is the most basic representation of the forms in the system. This could change in future as sometimes the developer discovers that a module needs its own form to improve he user interface.

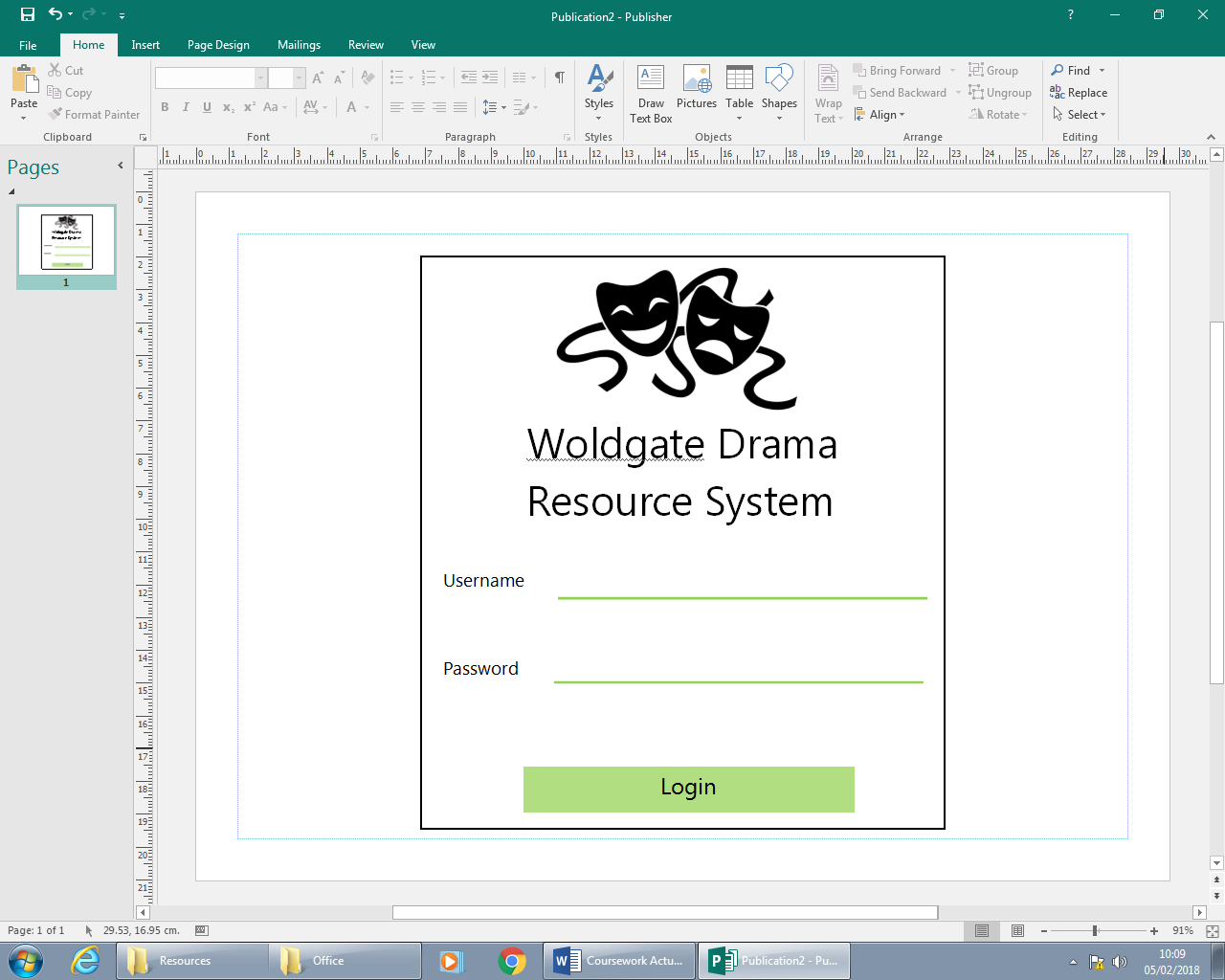
## Forms Design / UI

### Login

The login form is the page that will open when the user first enters the program. Its purpose is to let the user login with their credentials. The form will need two text boxes, two labels to represent what the text boxes mean and a login button. The login button will check if ther username and password is correct with the entry in the database and if the user has that permission to login (they might have been banned from the system for example.)

To create a clean and simple interface, I wanted to use as little “boxes” as possible to make the program look modern and sophisticated. This is why the text box is invisible (only the curcor shows) and theres a green line underneth it to show the user where abouts to type.

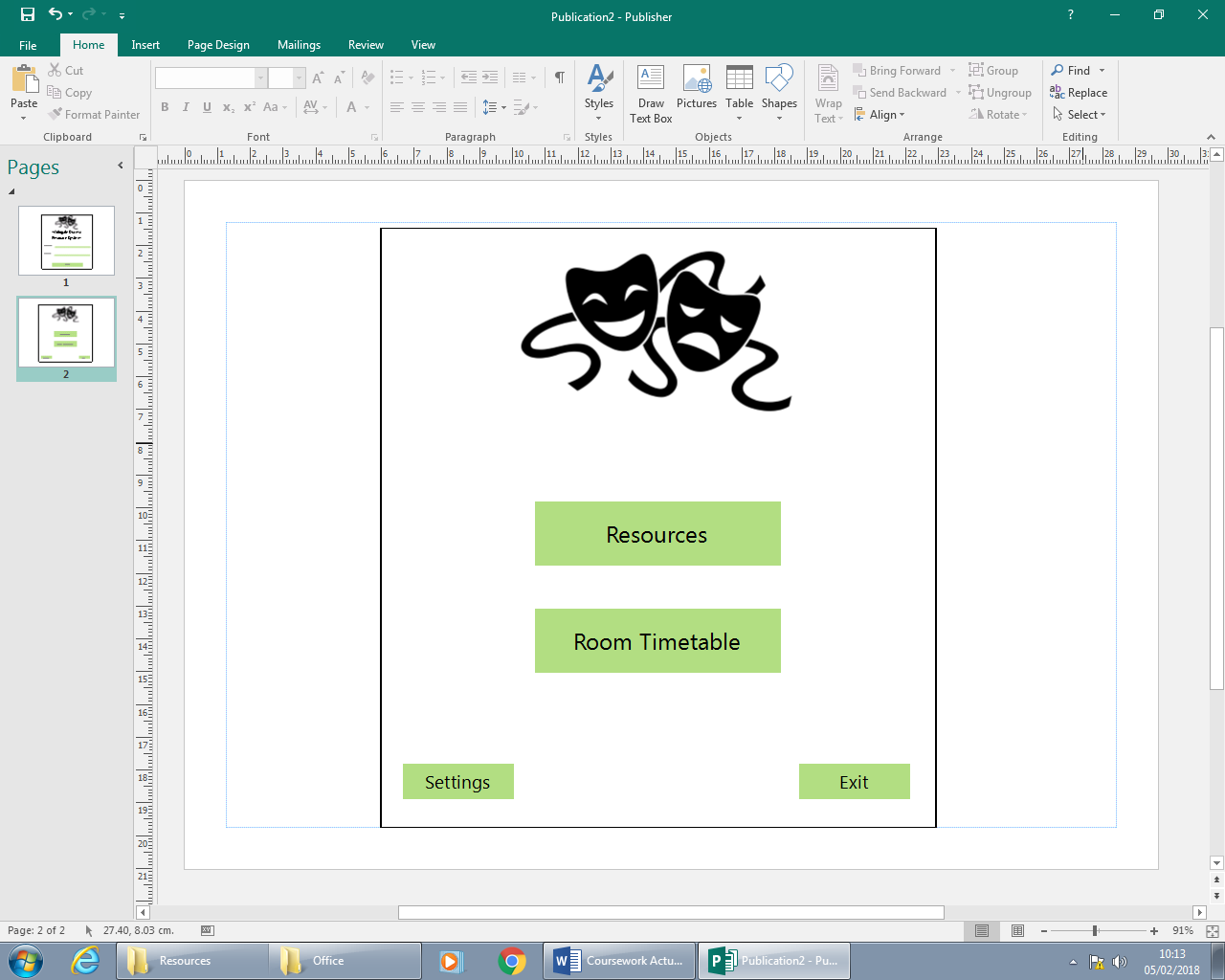
The login button does not have an outline and is a light colour which adds to the simplicity.



### Main Menu

The main menu is where the user will select what they would like to do in the program. It will have 4 options, Resources, Timetable, Settings and Exit. This acts as a hub between the differeence major modules of the program and conects them together so the user can easilly access all of them.

To continue the design set in the Login form, the same colour was used for the buttons and the same picture was used (although this might be changed at a later date due to copyright issues or the choice to use more diverse resources, to look more professional.)



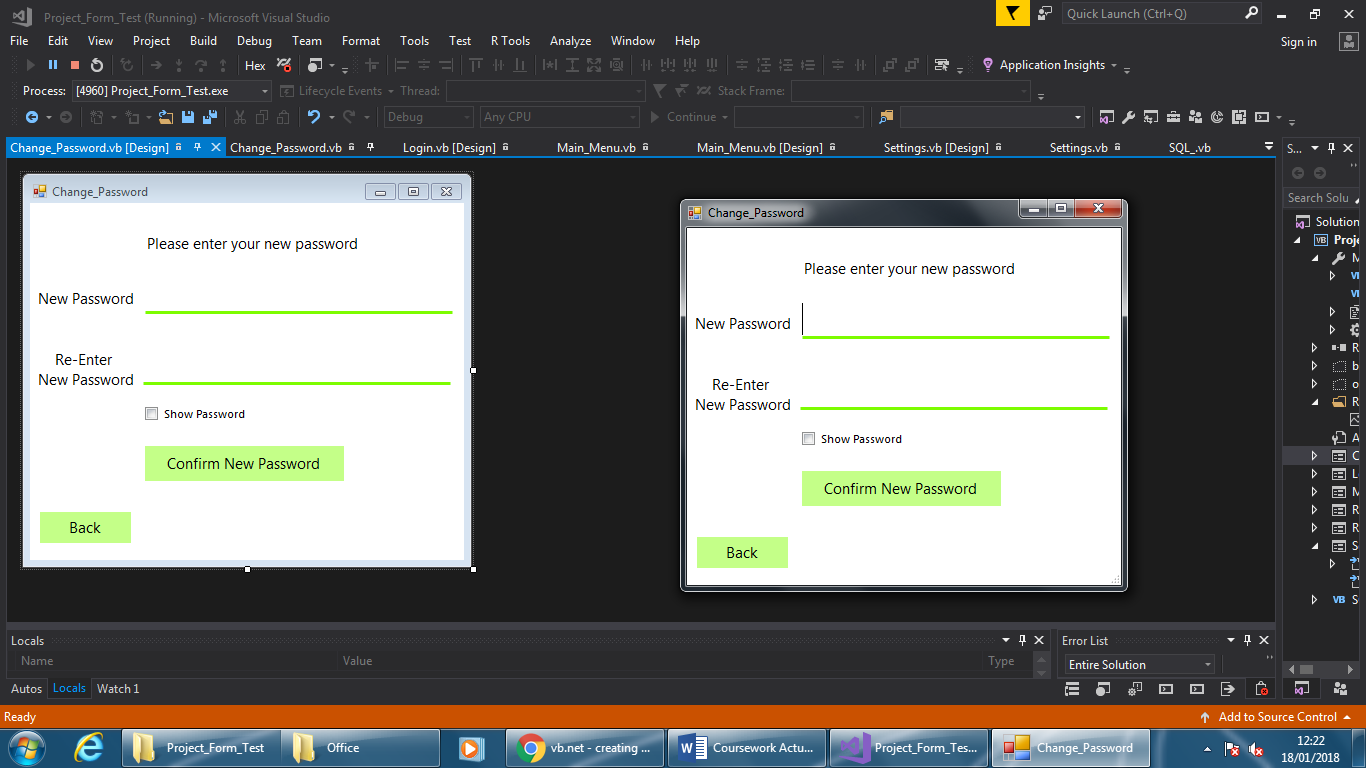
### Change Password

The change password form is one of the least commonly used forms however it is still important. The form needs two text boxes to enter the new desired password. This is common practice in most websites or systems where a personal password is needed. It is done incase the user accidentally presses a wrong key when typing their new password, then that wrong password is set in the system and the user does not know what he/she did wrong when typing. This is one type of vallidation which will make the program less stressful and more professional.

The form will also include a “Show Password” checkbox to give the user an option to be 100% sure about their password change.

The confirm new password and back boxes use the same design as previous forms for continuity.

When the user presses the “Confirm new password” box, a Message Box will appear with a Yes Or No selection to confirm that the user is absolutely shure they want to change their passwod.



### Timetable/Room booker

### Resource Tracker

### Account Management

### Settings

The aim of the settings is to allow the admins and users of the system to access settings of the program. These settings include the adding and removal of user accounts, changing the user’s password, adding lesson times to the room database, adjusting variables like the number of props a user is allowed to book at a certain time, setting permissions on which years are allowed to book a room, ban certain users from booking rooms / props. Only some of these actions can be performed by certain users eg. Admins can do everything whereas a student can only change their password.

The design will have to show all these options without feeling crowded. To do this the form will use multiple buttons grouped into categories, these will link to new forms

## Global Algorithms / Functions

### Validation

Validation is an essential part of any program in any language. It makes sure a user’s input doesn’t have any characters or values that the coding language or database can’t handle. To reduce redundant and repeated code, creating a global validation algorithm that checks whether any illegal characters have been used and stop it from returning to the code is the most efficient method. If it is an acceptable string input then it will return the string to the code previously running.

The multiple functions of validation will have a few parameters that can be set when calling the function, such as the maximum length of the string you want to validate.

There will be 3 types of validation in the globally referable module: Text Only, Number only and Text + Numbers. If during the code the developer wants to use one of these functions to check, for example, if a user entered password is longer than 32, then they would call the function with the parameters of the string itself and the number 32. This ensures the functions can be used multiple times throughout the program and reduce repeated code.

VALLIDATION\_TEXT\_ONLY(user\_input, max\_len, is\_allowed)

len\_ok = False #crates two local parametres that are used to compare if the string is the right length and is made of all text

txt\_ok = False

I = 0

FOR I to len(user\_input) #loops from 0 to the length of the user's input string

I += 1

Next

IF I <= max\_len THEN # I is then compared to the parameter max\_len to see if it is valid.

len\_ok = True # the len\_ok parameter is changed accordingly

ELSEIF I > max\_len THEN

len\_ok = False

ENDIF

IF IsTxt(user\_input) = True THEN #The IsTxt subroutine checks whether the string / doubble includes only text

txt\_ok = True

Else

txt\_ok = False #the txt\_ok parameter is changed accordingly

ENDIF

IF len\_ok AND txt\_ok = True THEN # the subroutine then compares both len\_ok and txt\_ok to see if they both equal true

is\_allowed = True # if both equal true, then the string is valid and the subroutine returns the is\_allowed paramter as True.

ELSE

is\_allowed = False

ENDIF

RETURN is\_allowed

END SUBROUTINE

VALLIDATION\_NUMB\_ONLY(user\_input, max\_len, is\_allowed)

len\_ok = False #crates two local parametres that are used to compare if the string is the right length and is made of all numbers.

num\_ok = False

I = 0

FOR I to len(user\_input) #loops from 0 to the length of the user's input string

I += 1

Next

IF I <= max\_len THEN # I is then compared to the parameter max\_len to see if it is valid.

len\_ok = True # the len\_ok parameter is changed accordingly

ELSEIF I > max\_len THEN

len\_ok = False

ENDIF

IF IsNumeric(user\_input) = True THEN # The IsNumeric subroutine checks whether the data is made of all numbers

num\_ok = True

Else

num\_ok = False # the num\_ok parameter is changed accordingly

ENDIF

IF len\_ok AND num\_ok = True THEN # the subroutine then compares both len\_ok and nub\_ok to see if they both equal true

is\_allowed = True # if both equal true, then the string is valid and the subroutine returns the is\_allowed paramter as True.

ELSE

is\_allowed = False

ENDIF

RETURN is\_allowed

END SUBROUTINE

VALLIDATION\_TEXT\_AND\_NUMB(user\_input, max\_len, is\_allowed)

len\_ok = False #crates two local parametres that are used to compare if the string is the right length and is made of all ascii valid characters

asc\_ok = False

I = 0

FOR I to len(user\_input) #loops from 0 to the length of the user's input string

I += 1

Next

IF I <= max\_len THEN # I is then compared to the parameter max\_len to see if it is valid.

len\_ok = True # the len\_ok parameter is changed accordingly

ELSEIF I > max\_len THEN

len\_ok = False

ENDIF

IF IsAscii(user\_input) = True THEN # The IsAscii subroutine checks whether the data is made of all valid ascii charactrs

asc\_ok = True

Else

asc\_ok = False # the num\_ok parameter is changed accordingly

ENDIF

IF len\_ok AND asc\_ok = True THEN # the subroutine then compares both len\_ok and nub\_ok to see if they both equal true

is\_allowed = True # if both equal true, then the string is valid and the subroutine returns the is\_allowed paramter as True.

ELSE

is\_allowed = False

ENDIF

RETURN is\_allowed

END SUBROUTINE

### SQL Statements

To communicate with the database the program must perform a bunch of SQL statements that get sent to the database and then the program receives the data it requested. In virtual Basic, the functionality to communicate with a database is engraved into the .NET framework and pulling information from a database is very easily done. However the SQL statements have to be 100% perfect for it to work. Because the SQL statement is not enclosed and read by the Virtual Basic debugger, it is the biggest failure point of the entire program, as it relies so heavily on communicating with the database. Virtual Basic doesn’t recognise an SQL statement, in fact, the statement is sent as just a sting of text to the database. So if no information is returned during the designing and prototyping of the program, then there is no way the developer will know what they’re doing wrong. This is why it is important to plan out the dummy database and create prototypes with working SQL statements before the program is created.

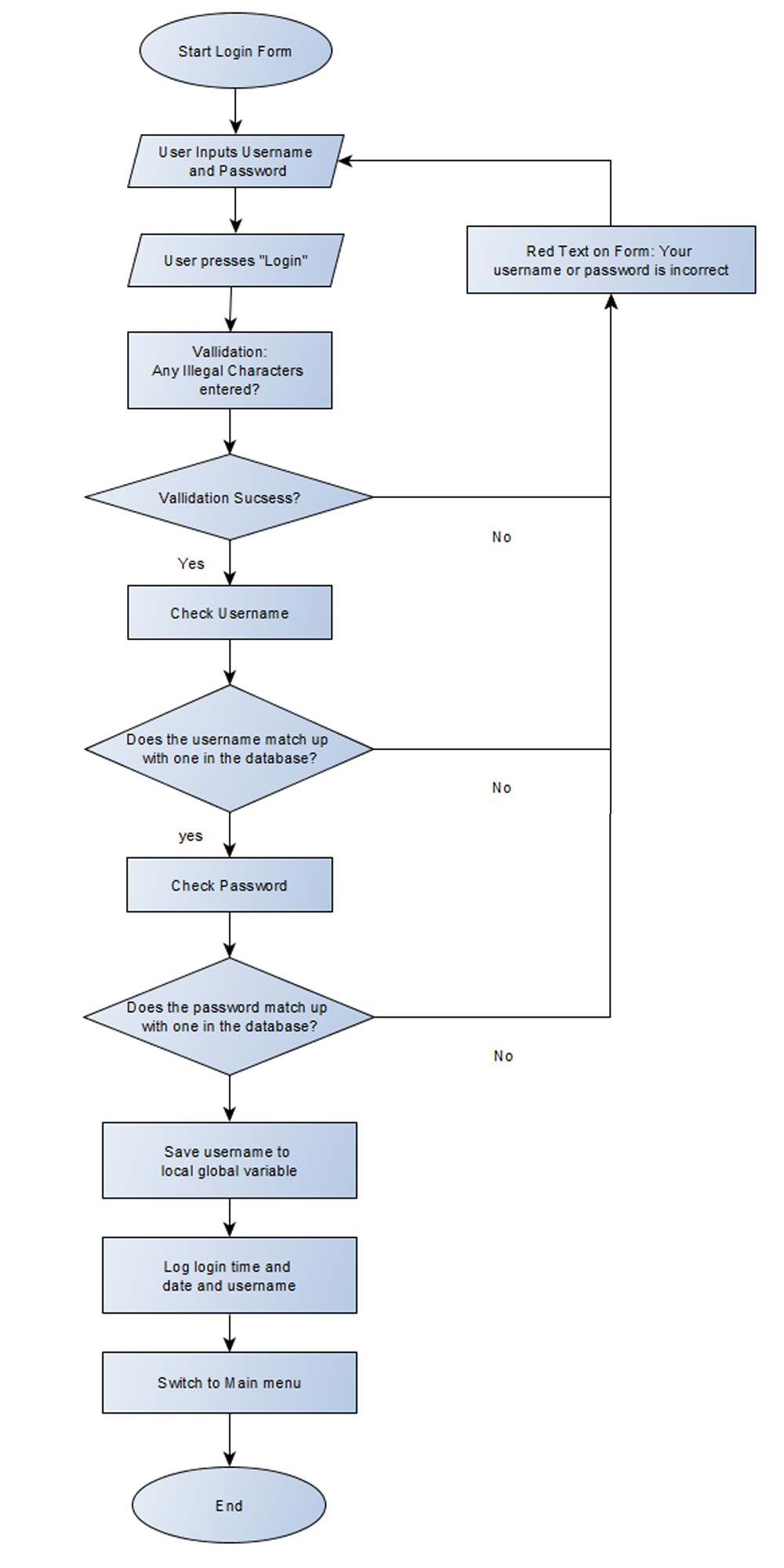
## Form Specific Algorithms / Functions

### Creating an account

This is one of the most complex parts of the program. To create a user account we need to create multiple unique SQL statements that add the information to the database and check through the database for repeated values. For example, if a user has the same name as an already registered user, their username record will have a 1 added to the end of it.

### Login Form

To represent to module flow of the login form the following flowchart may be consulted for reference. The algorithms involved in this flowchart are Validation, Check username, Check password, Save Username to global variable and switch to Main Menu form.



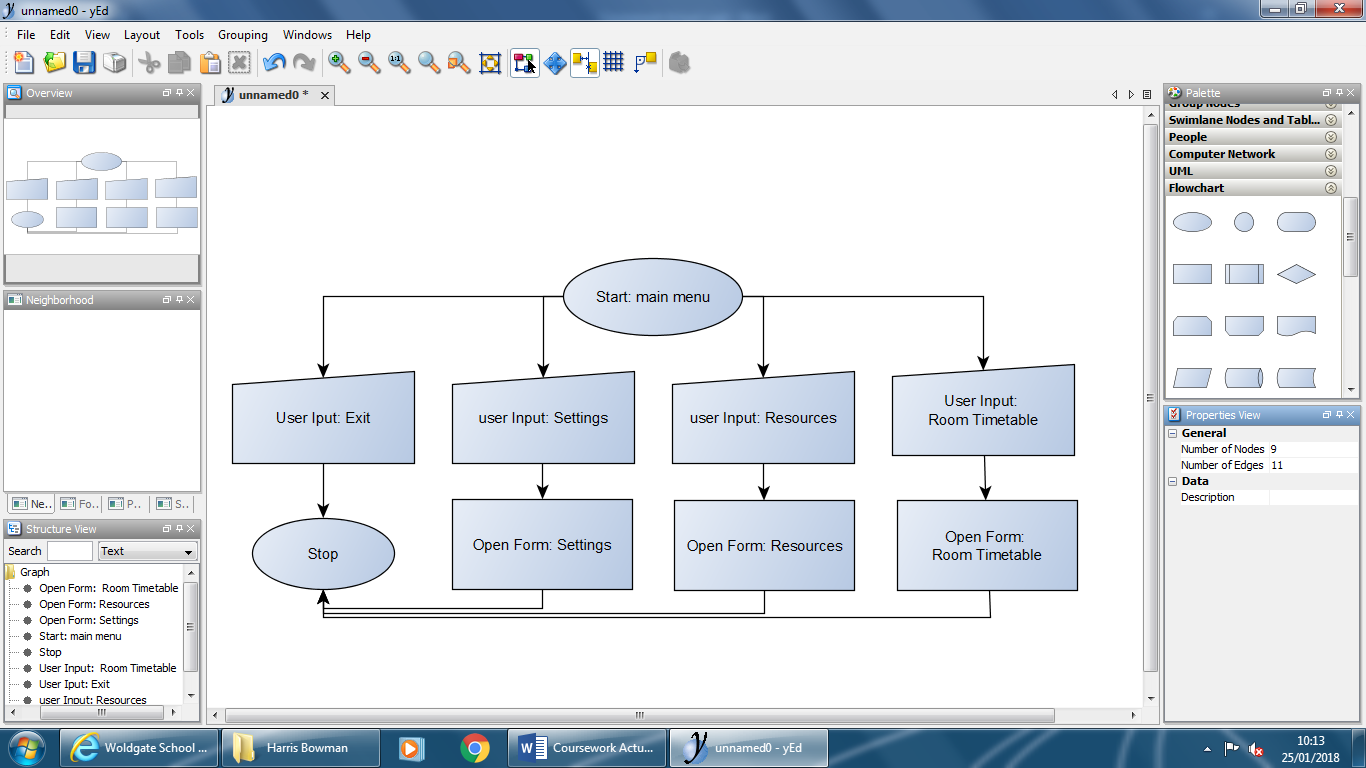
The red warning message is a label on the form so that the user doesn’t have to click “Okay” on a message box that appears every time they enter wrong information.

The text shown can also be changed if needed depending on where the error came from.

The check username and check password functions will link to an external module which contains the algorithms and furthermore, the SQL statements needed for reviewing if the username exists, and if it does exist, if the password is correct.

However if none of these are correct, the previously mentioned red label appears on the form.

### Main Menu



The only function of the main menu is to present to the user a clean interface for selecting the function they wish to perform on this program. The form uses buttons to switch the user to another form or open a message box.

### Room Bookings

### Resources

### Settings

### Another Thing that’s probably important

# Prototyping

## Prototype Plan

Prototyping is the process of creating a working “barebones” program that performs the basic functions and can be expanded upon. There can be multipole versions of prototypes throughout a programs development.

The easiest way to prototype is to create lots of little programs and algorithms that do not relate to the main program apart from dummy data and algorithm. For example create a separate program that handles room booking or a part of the program that displays the resources. These will all be listed in a later section and the developer could work through them one by one.

The reason why prototyping is important is because if the developer takes small steps towards the final program, the bugs can be ironed out in a safe environment separate from the main program, algorithms can be manipulated easily with dummy data, user interface design features can be implemented / coded / arranged and removes if necessary.

## Parts of the program that will be prototyped

### Main Menu / Dummy Forms

This is the easiest part to prototype, the main menus only job is allow the user to move between forms effectively and easily. The only algorithms needed are switching forms. The flow of forms is dictated by the flowchart in the [design section of this document.](#_Forms_Map)

In this initial prototype I will not design every form in accordance to my design section, but will use dummy forms with the names they well be in the final program and with only the buttons needed to switch forms so the code can be implemented as close to the final product as possible at this stage.

Doing this is good because it creates the backbone of which the program will be built around, each form needed is created with Public classes being set and form switching. This will help the development in future as it shows the developer where global variables are needed, local variables are needed, and if the initial designing of the forms in the design section will need to be changed at all before more subroutines are added to the forms. It also gives early prototype testers a feel of the program and if the flow of forms feels natural and easy to use, then if needed, can be changed at this point rather than far into development.

### Login Form

The prototype of the login form will have 3 aspects to it: validation of the users input, checking the users input with the data in the external database and if the user inputs a correct username and password the program then opens the main menu.

### Database creation

To test the algorithms and forms of the program, the most important piece of the program needs to be made, the database. The database will be referenced almost constantly during main development in code and notation. Therefore it is essential to get a coherent database file created and filled with dummy data to test the program out on.

The database will be created inside Microsoft Access 2016 as .acid file, which can be natively read by virtual basic with some code and SQL statements.

# Refinement of Design

# Software Development

# Testing

# Evaluation