**Analysis Tools - Solution Requirements Specifications**

1. Introduction
   1. **Purpose:**

To record work hours and calculate the pay in the format chosen by the user (day, week or month)

* 1. **Users:**

Christy MORAD - Main user

Other employees - who sign up

* 1. **Platform / OS:**

Windows 10 - Visual Studio 2017

* 1. **Project Management (Gantt Chart):**

<https://drive.google.com/file/d/1t9tzUZT3xQ8_KqwBHiaVZyHt39XddN8v/view?usp=sharing>

1. **Solution Requirements**
   1. **Functional:**

Input:

* Signup details for the first time using the software (username + password)
* Login details - every time using the software
* Start time and finish time (could also be entered automatically)
* Pay per hour

Output:

* Pay per day
* Pay per week
* Pay per month
* List of the user’s achievements over the week listed from the highest to the lowest pay.

Functions:

* Data manipulation:

The **signup data** will be recorded/stored in an XML file, and later when **login** the second time, the data will be searched for in the XML file, if the data does not exist, the solution will ask for either re-entering data or signing up.

After entering the **start time** and the **finish time** (that will be stored in the XML file under the username), the start time will be subtracted from the finish time to find out the number of hours worked and finally multiplied by the **pay per hour** that will be entered later in the program. Also, further calculations will be done on the hours worked and the pay to find pay per week and month.

* Validation:

**Signup and login data**, all characters will be accepted, however, no blank spaces are accepted

**Start time and finish time**, will be chosen from time picker and not entered by the keyboard, therefore no validation required.

**Pay per hour** is only allowed to be a double variable - no letters, blank, negative values or zero (at the start) are accepted and no specific range for this number.

* 1. **Non Functional – useability, reliability, portability, robustness, maintainability**

Useability: :

My solution has clear instructions and multiple forms specifically 7 - to make it easier to use.

Reliability:

By using the most efficient algorithm, accurate results will be guaranteed as less mistakes are likely to occur.

Portability:

This solution is made only for visual studio windows application, however, the XML file that will store the users’ data can be operated/edited using notepad - not only visual studio.

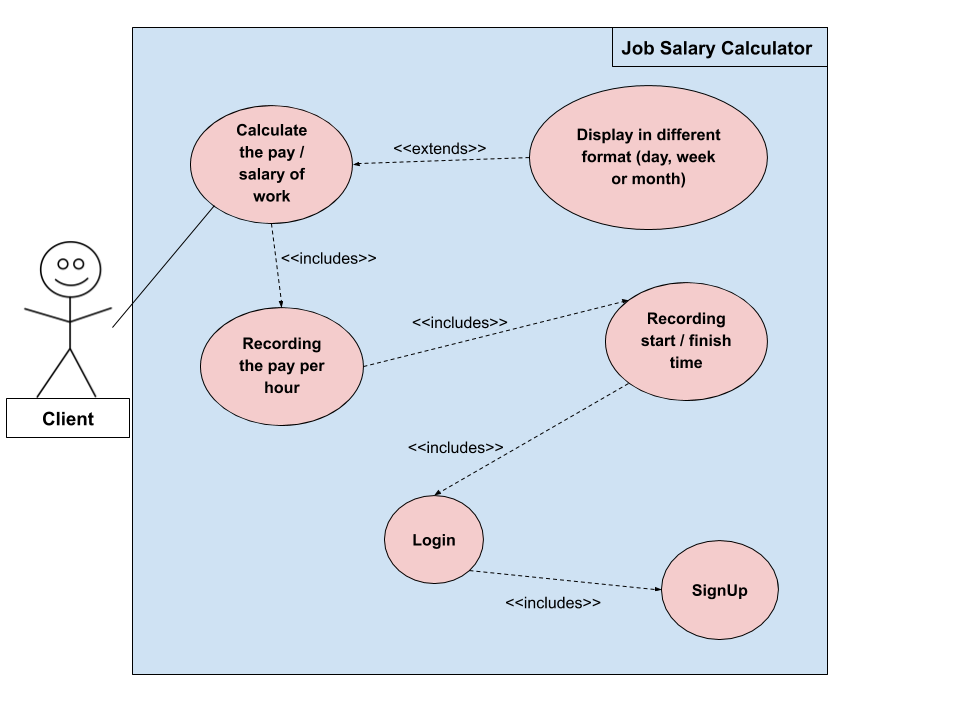
Robustness:

The solution will have validation for every single input to insure that whatever the user entre/press the the solution will generate something (ie. a message box in case something wrong is entered)

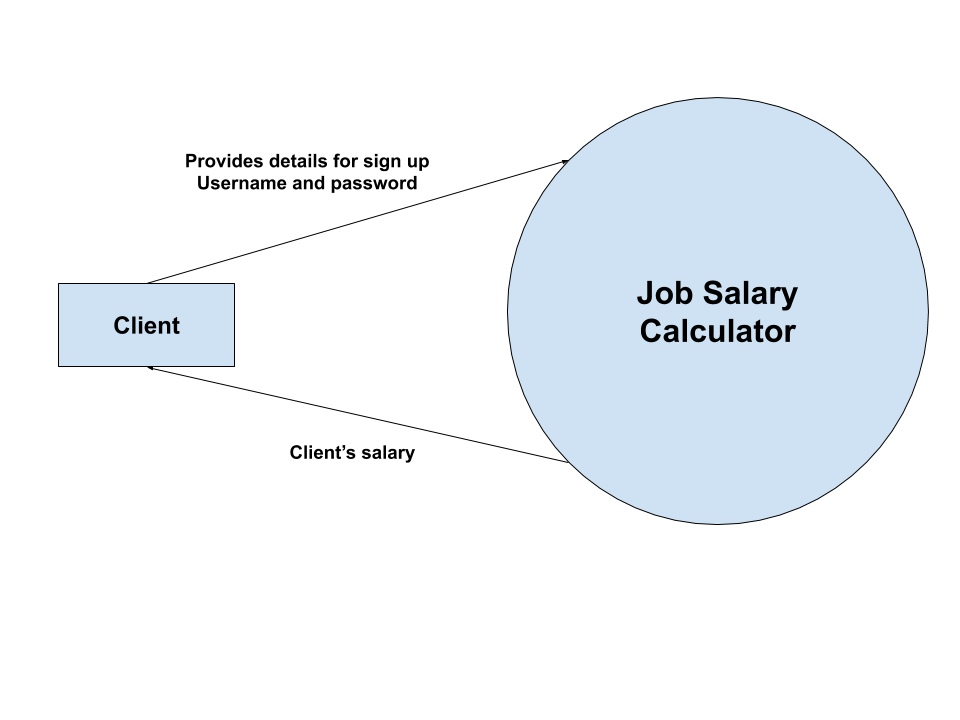
Maintainability:

My solution will be easy to maintain as I am going to comment on the code/algorithm so that if somethi- ng needs to be fixed, I will not have to read the whole code to find out where to fix. Also, in regards to maintaining data in the XML file, it will be easy to open it in notepad and directly edit it.

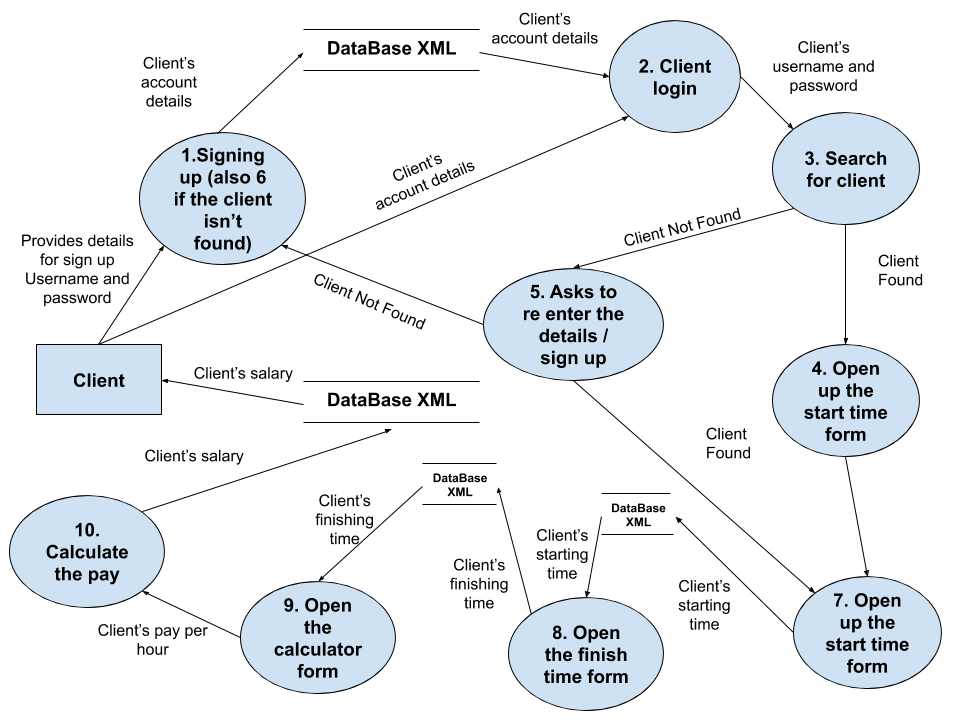
1. **Analysis Tools**
   * 1. **UCD**



* + 1. **CD**



* + 1. **DFD**

****

1. **Solutions Constraints**
   1. **Economic**

* Cost:

There’s no cost to the development of this solution as the materials used are school owned (computer + visual studio).

* Time:

Enough time is given to design and develop it, all the stages are recorded in the project plan.

* 1. **Technical**
* Speed of processing:

This solution is designed to work in a relatively fast

manner using efficient coding (ie. using methods that will allow some part of the code to run every single time without running the whole thing.

* Capacity:

The solution is designed to work under a relatively small capacity - no more than 50 MB free space in the user’s HDD as no images/videos/music are being used

* Availability of equipment:

No equipment constraints for this solution as it only requires a small number of equipment that are:

* Window operating computer, mouse, keyboard and visual studio software.
* Compatibility:

It is compatible with Windows OS, however, not compatible with IOS, Android or any other operating system.

* Security:

The user’s data/details will be secured with login as the solution will ask for login details every time the solution operates.

* 1. **Social**
* Levels of expertise of users:

The users only need to have simple/basic computer knowledge as they will only be required to enter some data, and press few buttons. The interface is designed to have clear instructions so that the user does not have to think a lot/ or get confused while using it which will be done using labels - with different font format, multiple forms and few colours.

* 1. **Legal**
* Copyright:

No images/symbols/videos or music are being used, therefore, no break to the copyright law.

* Privacy:

The user owns their own data and have the right to share or not (when designing the solution an idea of displaying names/work credits form different users was suggested, but the client refused to use it unless a consent is taken from every single user, therefore, this idea was rejected. Also, the user’s details will be hidden as they enter them and also give the user the choice to show or hide them.

* 1. **Useability**
* Usefulness:

The solution is designed to make it easier for the user to calculate their pay in a few simple steps in addition to record it in an XML file which are the requirements that the client asked for.

Also, in addition to the multiple forms, clear instructions and colours discussed previously, the solution is designed to effectively use the **Tab** button on the keyboard - to move from a control to another as well as the **Enter** button to confirm in case the user prefers to use the keyboard instead of the mouse

1. **Scope**
   1. **What the solution can do**

The solution will be able to:

* Record the sign up details for the client/users
* Search for the details to check when logging in (manually or automatically)
* Allow entering the start and finish time of work
* Allow entering the pay per hour
* Record it in an XML file along with the user’s details
* Do calculations on the data (subtract the start from finish time and then multiply the result by the pay per hour)
* Display the results depending on the user’s selection (weather per day, week or month)
* Sort the user’s pay per day from the highest to the lowest in the **Achievement board** after more than one day has been recorded
  1. **What the solution will not do**
* One of the things that this solution will not be able to do is that it will not be portable - only used on a computer whilst it could have been designed to work on Android phones for example.
* Another thing that the solution won’t be able to do is to let the user choose what date to enter time for - for example, if the user forgot to add one day, they will have to enter on the next day - perhaps added to the currnet day’s work as the solution is designed to take current date only