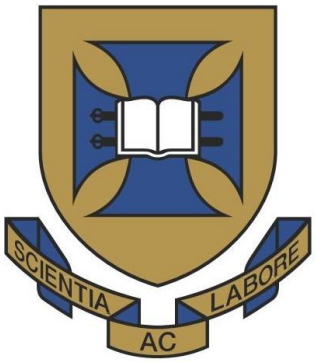


THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

Final Report Team 03

Prepared by: Vigneshwar Parikumar (43952991)

Members:

Vigneshwar Parikumar

Matthew Tap

Tri Nyguen

Charles Gore

Team Design Overview

- The design process was based on meeting the demo requirements
    - Design started with GUI functionality and the e-paper display
    - Partial update was also implemented at this stage
    - The next stage involved the UV sensor, sleep mode and display functioning
    - Additional features missing were implemented after, including the SD card interfacing
- Tasks were allocated in week 4
- Weekly meetings were held at 2pm in the Axon lab
    - These were used to update the team on individual progress and work on immediate deliverables as a team
    - Encouraged peer feedback
    - Was effective until week 9 when individual commitments became heavier
- Leadership was a shared role
    - All team members were responsible for individual parts of the project
    - Created accountability
    - Required members to be flexible
- Slack was initially used for project communication, however by week 10, Messenger was a more responsive means for team communication
- Matthew Tap completed the E-paper, partial update, sleep mode and hardware construction and design
- Tri Nyugen completed the SD card interfacing, UV sensor, E-paper updating and hardware construction
- Charles Gore was involved with the hardware design
- Vigneshwar Parikumar was involved with the software design, serial functionality, ensuring the software works correctly on the lab computers and firmware debugging

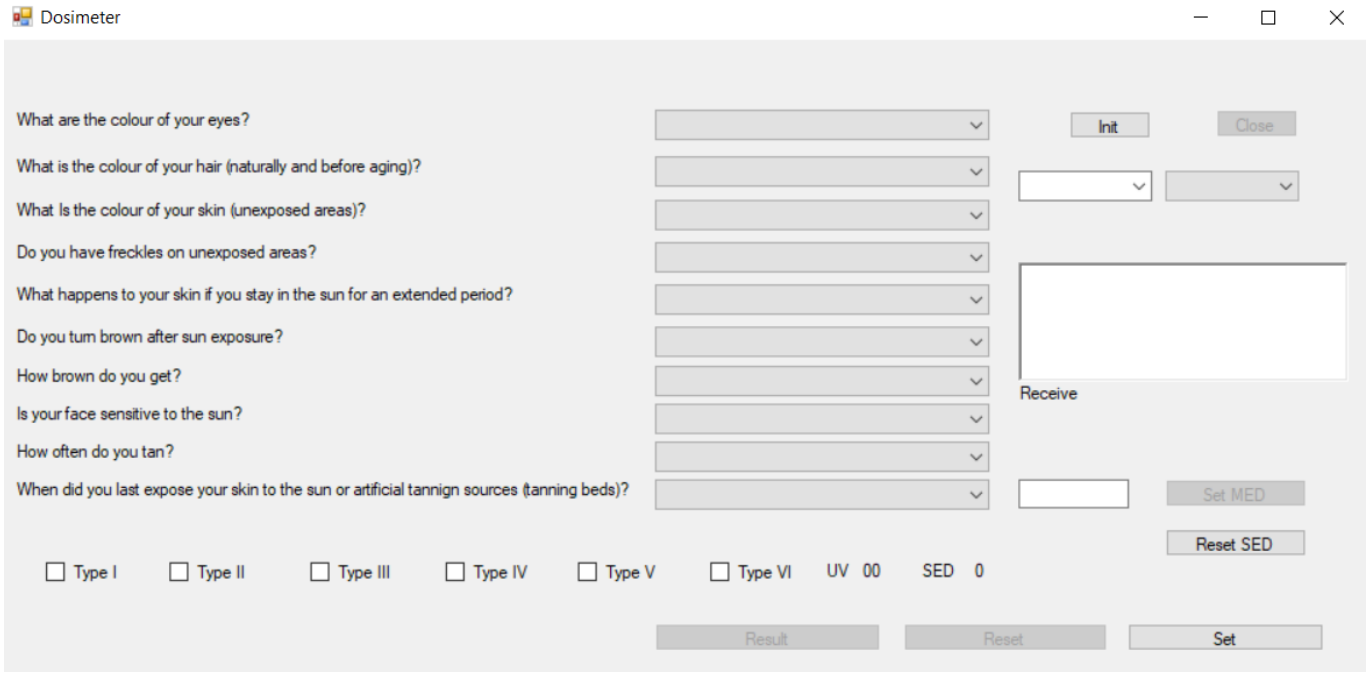
Individual Contribution

The design of the GUI started with consideration of requirements of the client. These are summarised below:

- All cost of licences for additional libraries need to be costed into the BOM: To reduce additional cost of licencing, it was decided to use a language that had close integration with the Windows framework, hence the decision to develop using the .NET Framework. The main tool used to develop for the .NET Framework is the Visual Studio suite of IDEs. Given the options Visual Studio Community was chosen for development for the following reasons:
    - Visual studio community can be used for free to develop software for commercial use if the software is open source
    - Gives access to the Winforms UI Framework which allows for rapid development of the interface

    Using the Winforms UI Framework gave options of two languages to develop with; VB.NET and C#. VB.NET was chosen over C# as they both have the same feature set and, however VB.NET is also easier to implement.

- The quiz interface was the first component designed and was designed to be a single page dashboard style layout to allow intuitive navigation by the end user.



To make this possible, the design incorporated drop down boxes which store the answers. Exception handling for the quiz was done by disabling the result button, only enabling the it when all answers are input by the user.

```
0 references
Private Sub SelectedIndexChanged(sender As Object, e As EventArgs) Handles cboQ1.SelectedIndexChanged, cboQ2.SelectedIndexChanged,
cboQ3.SelectedIndexChanged, cboQ4.SelectedIndexChanged, cboQ5.SelectedIndexChanged, cboQ6.SelectedIndexChanged, cboQ7.SelectedIndexChanged,
cboQ8.SelectedIndexChanged, cboQ9.SelectedIndexChanged, cboQ10.SelectedIndexChanged
    'Re-enables result button
    If (Not String.IsNullOrEmpty(cboQ1.Text)) And (Not String.IsNullOrEmpty(cboQ2.Text)) And (Not String.IsNullOrEmpty(cboQ3.Text)) _
    And (Not String.IsNullOrEmpty(cboQ4.Text)) And (Not String.IsNullOrEmpty(cboQ5.Text)) And (Not String.IsNullOrEmpty(cboQ6.Text)) _
    And (Not String.IsNullOrEmpty(cboQ7.Text)) And (Not String.IsNullOrEmpty(cboQ8.Text)) And (Not String.IsNullOrEmpty(cboQ9.Text)) _
    And (Not String.IsNullOrEmpty(cboQ10.Text)) Then
        btnResult.Enabled = True
    End If
End Sub
```

The manual override of skin type was implemented through checkboxes to allow for easy user interaction. Additionally, the manual override of the MED was implemented through a text box. When the Set MED button is pressed, only values between 0 and 441 are accepted and this button changes the MED.

```
0 references
Private Sub BtnMEDOvr_Click(sender As Object, e As EventArgs) Handles btnMEDOvr.Click
    'Manual override of MED
    If rtbMED.Text <> String.Empty Then
        Dim MED As Integer
        MED = rtbMED.Text
        If (Convert.ToInt32(MED) > 0 And Convert.ToInt32(MED) < 441) Then
            m_IntMED = MED
            m_IntMEDFlag = 1
            lblMED.Text = Format(m_IntMED, "00#") & m_IntMEDFlag
        End If
    End If

End Sub
```

It was required that the result stored by the quiz variable was accessible to other functions, hence requiring the use of variables with module level scope.

- The serial functionality was the next feature implemented. The .NET Framework provides a native serial port class, System.IO.Ports, which includes a rich set of functions for serial implementation. The serial port is opened through native class functions and can only be opened when a COM port and baud rate is selected.

```
'Handles serial port connection
0 references
Private Sub BtnInit_Click(sender As Object, e As EventArgs) Handles btnInit.Click
    If (Not String.IsNullOrEmpty(cboCOM.Text)) And (Not String.IsNullOrEmpty(cboBaud.Text)) Then
        SerialPort1.PortName = cboCOM.Text
        SerialPort1.BaudRate = cboBaud.Text
        SerialPort1.Open()
        btnInit.Enabled = False
        btnWrite.Enabled = True
        btnClose.Enabled = True
    End If
End Sub
```

It was decided to simplify the communication between the hardware and software as much as possible to allow for more robust implementation. To simplify the communication between the GUI and the hardware and to reduce the complexity of decoding the information for use by the hardware, it was decided to send one string to the hardware, which the hardware can decode easily. The string sent from the GUI consists of four parts:

- A flag to indicate that the GUI has sent data
- A time stamp formatted as ddMMyyyyHHmmss – the hardware uses this to synchronise the RTC with the system time
- The skin type as an integer
- A flag 0 or 1 – If the flag is 0, the hardware uses the default MED corresponding to the skin type. If the flag is 1, the user set MED is appended to the string before being sent to the hardware, and the hardware uses this internally.

Further exception handling is implemented by enabling the send button only when a skin type and MED is specified, and the serial port is open.

```
Private Sub BtnSet_Click(sender As Object, e As EventArgs) Handles btnSet.Click
    'Sends formatted data to the hardware
    If ((m_StrResult <> "") And (m_IntMED > 0) And (m_IntMED < 441) And (SerialPort1.IsOpen = True)) Then
        If m_IntMEDFlag = 1 Then
            m_strTimeDate = m_strSED & Format(Now, "ddMMyyyyHHmmss") & m_StrResult & m_IntMEDFlag & Format(m_IntMED, "00#") & Chr(62)
        ElseIf m_IntMEDFlag = 0 Then
            m_strTimeDate = m_strSED & Format(Now, "ddMMyyyyHHmmss") & m_StrResult & m_IntMEDFlag & Chr(62)
        End If
        lblDateTime.Text = m_strTimeDate

        For ctn As Integer = 0 To (Len(m_strTimeDate) - 1)
            SerialPort1.Write(m_strTimeDate.Chars(ctn))
            SerialPort1.DiscardOutBuffer()
        Next
        m_strSED = "0"
    End If
End Sub
```

Once the communication is opened, the GUI needed a way to confirm that it was still communicating with the hardware. To enable this, the GUI was designed to send the same string that it sent originally, and the device responds back with a string formatted as UV,SED,MED. The UV and SED data are displayed on the GUI. This also has an additional advantage of synchronising the RTC of the hardware.

```
Private Sub ReceivedText(ByVal [text] As String)
    If Me.rtbRec.InvokeRequired Then
        Dim x As New SetTextCallBack(AddressOf ReceivedText)
        Me.Invoke(x, New Object() {(text)})
    Else
        Me.lblUV.Text = ""
        Me.rtbRec.Text &= [text]
        Me.lblUV.Text &= [text].Split(",")(0) & Chr(10)
        Me.lblSED.Text = ""
        Me.lblSED.Text &= [text].Split(",")(1) & Chr(10)
        Me.rtbRec.Text &= [text]

        'Send acknowledgement to hardware
        If m_IntMEDFlag = 1 Then
            m_strTimeDate = m_strSED & Format(Now, "ddMMyyyHHmmss") & m_StrResult & m_IntMEDFlag & Format(m_IntMED, "00#") & Chr(62)
        ElseIf m_IntMEDFlag = 0 Then
            m_strTimeDate = m_strSED & Format(Now, "ddMMyyyHHmmss") & m_StrResult & m_IntMEDFlag & Chr(62)
        End If

        For ctn As Integer = 0 To (Len(m_strTimeDate) - 1)
            SerialPort1.Write(m_strTimeDate.Chars(ctn))
            For ctn2 As Integer = 0 To 20

            Next
            SerialPort1.DiscardOutBuffer()
        Next
        If (m_strSED = "1") Then
            m_strSED = "0"
        End If
    End If
End Sub
```

The SED reset was the final feature to be implemented. This was implemented by simply setting a flag to 1 when the “SED Reset” button is before or during communication. This flag is then reset by the code above.

Sustainability Considerations

Software sustainability pertains to the ability of the software to endure changes in hardware and software over time. As such, important considerations were made regarding:

- The evolution of the code to newer hardware revisions
- The migration of code to different operating systems
- The ability to migrate source code to different languages

This was a key consideration when selecting the platform to develop the GUI on. A migration approach was considered from the beginning of the design process, which narrowed the choice of IDE to Visual Studio Community, which provided a portable development platform and a rich set of development tools. The .NET Framework has been designed to have backwards compatibility from .NET Framework 4.5 and later. This backwards compatibility allows for easier migration of older code to the new framework version. Additionally, the Winforms UI Framework used to design the GUI has a choice of two languages; VB.NET and C#, both of which have the same functionality. This allows the code to be migrated to a different language with relative ease. Cross platform needs can also be accommodated by porting the code written in the .NET Framework to the .NET Core, which supports cross platform deployment.

Migration can come in two forms: partial migration or full migration. Partial migration involves tweaking the source code to be compatible with a new version of an operating system or programming language. A full migration may involve rewriting the source code from the beginning, however the programmer is constrained by the old architecture. The potential resource heaviness of this approach is its biggest drawback.

Alternatively, the development could have taken a cultivation approach to software sustainability by making the source code open source. Cultivation allows more contributors to engage in the software

lifecycle, allowing the sharing of maintenance and providing redundancies if a developer decides to drop out. The major disadvantage to this approach is the fact that it is a long-term process that involves planning effort and community building to ensure the longevity of the software.

Both approaches to software sustainability have benefits and pitfalls. Being able to migrate the code may be resource intensive if a full migration is required, however poses advantages with partial migration. A cultivation approach allows for sustained support by contributors, however is time consuming. A hybrid approach with these two approaches can ensure that the software can be ported to other platforms and has continuous and sustained support.

Course Reflection

Going into this project, I expected this project to be challenging while also being engaging. The challenge I was expecting was integrating smoothly with my team, as I have had distasteful team experiences in the past. I was pleasantly surprised upon meeting my team, who were eager to start the project, and had analysed the project specifications before we first met. I believe that this enthusiasm from my team helped our team integrate and work cohesively throughout the semester.

After receiving the specification, I started working on the GUI from an early stage as I felt at the time that this was something I can begin before finalising our team strategy. This was quite challenging as I have only every worked with GUI design in CSSE1001 before this. Despite this handicap, this experience has helped me learn a new language (VB.NET), learn how to use different development tools (Visual Studio) and experience GUI design firsthand. I also had some experience debugging the firmware, using tools such as the STM32 IDE for the first time.

During much of the project, we met once a week to discuss individual project developments over the week and plan for the next week. This worked well as it helped us maintain a balance between our other commitments and this project while ensuring that we did not become complacent regarding deadlines for this project. These meetings were mostly informal design meetings with little documentation. After completing the project, I now feel as though these meetings could have been enhanced with more meeting artefacts to refer to later as needed. Unfortunately, despite all our planning throughout the semester, we came across a problem late in the project where we lost our components for the final product due to miscommunication. This experience, although disheartening, did serve as a lesson as to the importance of having a contingency plan, which I would now implement at the beginning of the project given the chance to again. Additionally, our GUI refused to initially work properly on the lab computers days before the project was due. This issue was resolved; however, a contingency plan would have helped reduce pressure that weekend.

The Code of Ethics was mainly inherent in our design principles during the semester. This was mostly followed through the semester, and I feel as though the code of ethics we created at the beginning was indicative of industry Code of Ethics. Our Code of Conduct was mostly adhered to; however, I now feel as though tighter enforcement of the Code of Conduct could have prevented some tight situations. For example, we initially stated on our Code of Conduct that we would give notice advance notice if we are unable to attend meetings and check on members if they are missing meetings. As the semester became busier, we started overlooking this code, and members would occasionally give late notice of absence before the meeting or no notice at all.

Our team's performance was at a high level through most of the semester, with deadlines being met early and regular communication between team members. Since we were all electrical engineering students, our strengths naturally lied in hardware, so I believe that the delegation of the GUI component would not have made any difference with our overall performance. Despite our strengths in hardware, our team did not have major difficulty with the GUI aspect, with the greatest difficulty being learning VB.NET and the final hurdle of getting the software to function properly on the lab computers. From the beginning, we delegated tasks based on the demo requirements, and if these were met before the deadline we continued

with additional features. This seemed to work well and helped us stay ahead of our schedule for the most part. I felt as though we could have done more to stay on track, particularly with regards to demo 2. In this demo, we lost marks for not having the sleep a measurable sleep current through the voltage regulator. Upon reflection, this demo would have gone more smoothly if the focus was on implementing the sleep mode rather than the bonus mark requirement. After this experience however, our team's focus shifted to the important deliverables, with secondary features implemented given time.

As for the Engineering Australia Stage 1 Competencies that I feel I developed, the following list ranks what I believe I developed most with the completion of this project and why I believe so:

1. 2.3 Application of systematic engineering synthesis and design processes – This project required me to apply the knowledge I have gained over my degree to enable us to meet the requirements of specification. I believe that this is a significant area of personal improvement thorough the semester as the project required several individual design components to integrate coherently. This integration process was the component I feel I developed most through the completion of this project, which required debugging to seamlessly integrate the systems.
2. 2.4 Application of systematic approaches to the conduct and management of engineering projects – Completion of this project required me to be flexible and adaptive with my management style. From the beginning we assessed the scope, however reassessment was required when the scope changed. I believe that this project exposed me to the real challenges of project management in industry.
3. 3.2 Effective oral and written communication in professional and lay domain – The aspect of this project that really helped me develop my communication skills is the project seminar, particularly the seminar recording beforehand. I found that the recording process helped us reflect on our own presentation style, and I found the guidance my team members extremely valuable, as they pointed out potential aspects of improvement in my presentation. I feel as though my seminar presentation was optimised thanks to this exercise beforehand. This is an exercise I will take with me into the professional domain.
4. 3.6 Effective team membership and team leadership – This project really required me to be flexible with my teamwork, as I needed to be both a member and take on leadership roles within the team. The leadership role involved me taking executive decisions for the team and accepting the consequences for these decisions. An example of this was when I made the executive decision to start work and take control of the GUI development before our roles were finalised. Since I took on this role, I was responsible for meeting the deadlines regarding the GUI with a high level of execution.

I feel as though taking risks by taking responsibility for the software early on was a great experience which allowed me to expand my knowledge and skill base. Upon completion, I do however wish I had more input on the hardware design. However, my design approach over the semester was still scattered, however I feel as though this was due to the learning curve required to learn a new language. I enjoyed the course, however I felt little need for some small requirements like the specific box size needed for the hand in requirement. I feel that the greatest benefit from this course was the seminar in week 11, and particularly the self-feedback process requirement. Through this, I received valuable feedback from my team as to how to improve my overall presentation, and I gave a more focused presentation as a result.