1.2 System Requirements

**The main system requirements needed to handle the following software application are:**

1. A PC or laptop running at least Windows 10 os (or macOS/Linux equivalent) to ensure compatibility and performance.
2. A **keyboard** for data entry, report generation, and dashboard interaction;
3. A **mouse or trackpad** to navigate and interact with the graphical user interface.
4. A **stable internet connection** for real-time data upload, cloud storage access, and user–clinician communication.
5. A **modern web browser** (such as Google Chrome orFireFox) to access the Sensore dashboard and data visualisation tools;
6. **Sufficient storage capacity** (minimum 15 GB free space) for local caching and handling pressure map CSV datasets;
7. **Minimum hardware specifications:**
   * Processor: Dual-core 2.0 GHz or higher
   * RAM: 8 GB or more for smooth data analysis and visualisation
   * Display: at least 1080p HD resolution for clear heat map and graph representation.

1.3 Project Plan

This project was predominantly completed using The Agile methodology with some aspects of the Software Lifecycle aswell , The reason we largely relied on the Agile methodology was due to its reliance on feedback based decision making and real time changes to the project architecture.

To demonstrate the specifics, we’ve listed the cons of both systems below and explained how we made our decision.

|  |  |
| --- | --- |
| **Agile methodology** | **Software Lifecycle** |
| Requires cultural change as it is not the norm | Inflexibility to Change & High Cost of Errors |
| Can be hard to scale for large organizations. | Overemphasis on Documentation |
| Needs consistent involvement from relevant parties. | Poor Customer Feedback During Development |

Our team came to the conclusion that the consequences of using the traditional waterfall system were far greater than those of using the Agile system for this project, Since this project is already largely based on patient feedback due to the practical testing and analysis of patients using the pressure distribution heat maps. The research team as well as the database creation team are small-mid scale in terms of man power based on the required skills and precise nature of the venture and due to this same reason We believe they can withstand the cultural shift for the efficiency of the Agile system. However we still chose to rely on extensive documentation as a research tool for future projects of this venture as well as future ventures in the field for the sake of readability and understanding.

The project referenced in this report was largely completed using the Agile Methodology(Figure 1)

A diagram of a process

AI-generated content may be incorrect.

*Figure 1. Agile Methodology*

The project referenced in this report was also partially completed using the Software Lifecycle (Figure 2)

A diagram of software development cycle

AI-generated content may be incorrect.

*Figure 2. Software Development Cycle*

In the proposed *Graphene Trace Sensore* Case Study, the main functionality is expressed in 28 User Stories, covering core requirements. Accordingly, the application is designed using three UML Diagrams.

* **Use Case Diagram** – a behavioural-based diagram summarising the actions performed by each user (patient, clinician, and admin) and how they interact with the system’s core modules.
* **Class Diagram** – a structural-based diagram outlining the system’s main classes, attributes, and relationships, such as User, Clinician, Pressure Data, Alert, and Database Manager.
* **Sequence Diagram** – an interaction-based diagram illustrating the sequence of operations between user interfaces, analysis modules, and the database .

To accommodate user requirements, a database was implemented to store all persistent data, including user profiles, sensor readings, alerts, and feedback threads. An Enhanced Entity Relationship Diagram (EERD) was developed to visually represent the database schema and the relationships among entities.

The application was developed in C# using the Visual Studio 2022 IDE, with .NET MAUI chosen for building the cross-platform user interface. GitHub was used for version control, enabling pair programming and collaborative iteration across sprints.

1.4 Resources Used

* **Visual Studio 2022**, Integrated Development Environment (IDE) for developing the C# application.
* **.NET Framework 6.0**, used for building and running the backend logic and user interface.
* **Microsoft SQL Server Management Studio (SSMS)**, version 19.1, for database creation and management.
* **Microsoft Word**, version 2021, for preparing the project report and documentation.
* **Microsoft Excel**, version 2021, for organizing pressure data samples and calculating preliminary metrics.
* **GitHub Desktop**, version 3.3.6, for version control, team collaboration, and code repository management.
* **Google Drive**, version 2024.10, for file sharing and collaborative access to project materials.
* **Lucidchart**, online tool for workflow and UI flowchart design
* **Mozilla Firefox**, version 132.0, for testing the web-based dashboard interface and ensuring cross-browser compatibility;
* **Figma**, version 123.5, for creating and prototyping the user interface and interactive dashboard elements;