**Logo

Description automatically generatedCairo University**

**Faculty of Graduate Studies for Statistical Research**

**Department of Computer Science**

**The Smart Battery Monitor**

**A Project Presented for Fulfillment**

**For Diploma Project in Computer Science**

**Submitted by**

**Mohamed Sayed Hemed El-Sayed**

**Sherif Mostafa Samy Ahmed**

**Supervised by**

**Dr. Ahmed Hamza**

**Cairo, Egypt**

**April 2021**

* **Abstract:**

In recent decades, smart devices have played the most significant role in human life, and power consumption has been one of the most trending issues. As a result, batteries and their lifespan have been one of the most sophisticated topics.

All people nowadays want to use smart devices for the longest time possible without recharge their batteries, that's because smart devices became one of the most important tools for all people regardless of their work positions or their business.

One of the solutions for dealing with this problem is trying to maintain battery lifespan as much as possible, so in this report, we introduce a smart solution to charge the battery with the standards that the major global companies advise, we support this hypothesis with many articles that specialist companies have been published.

This report discusses the design, implementation, and testing of a device that automates the charging process by protecting the human from the headache of monitoring the percentage of charging periodically.

* **Declaration:**

We here declare that the project entitled “The Smart Battery Charger” submitted for all training centers, and educational institutional, which serve the student affairs in smart and automated way.

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Name | Task | Signature |
| 1 | Mohamed Sayed Hemed | Hardware Implementation (Design Board & write low level codes) |  |
| 2 | Sherif Mostafa Samy | Software Implementation (Design interfaces & writing code) |  |

* **Acknowledgments:**

At the beginning and at the end we all thank Allah for helping us to achieve this work and ask Allah to benefit us with what he taught us and teach us what will benefit us.

On the behalf of the Faculty of Statistical Studies and Research, Cairo University, and on our own behalf, we would like to express our sincere gratitude to all those respectable Professors in capacity of Dr. Ahmed Hamza who guided us through the preparation of this project and for his continuous support, for his patience and motivation.

Our warm greetings for the computer science department management from the professors, doctors, teaching assistants and employees.

Finally, we thank our families for supporting us and helped us to reach so far and to have the tolerance to face difficulties and keep up.

* **Table of Contents**

|  |  |
| --- | --- |
| Abstract | 1 |
| Declaration | 2 |
| Acknowledgments | 2 |
| Table of Contents | 3 |
| Table of Figures | 5 |
| Table of Tables | 6 |
| Definitions, Acronyms, and abbreviations (Glossary) | 7 |
| **Chapter 1: Introduction** | **8** |
| 1.1 INTRODUCTION | 9 |
| 1.2 Motivation | 9 |
| 1.3 Problem Definition | 10 |
| 1.4 Background |  |
| 1.5 Related works |  |
| 1.6 Scope |  |
| 1.7 Aims and Objectives |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| 1.9 DEVELOPMENT PHASES & METHODS |  |
| 1.9.1 The System Development Life Cycle |  |
| 1.9.2 System Development Process Overview |  |
| 1.9.3 The system development life cycle is the Scrum Agile methodology |  |
| 1.9.4 Software development diagrams |  |
| 1.9.5 Development Tools |  |
| **Chapter 2: System Analysis** |  |
| 2.1 Requirements Analysis |  |
| 2.2 Business Requirements |  |
| 2.3 Non-Functional Requirements |  |
| 2.4 User story |  |
| 2.5 Actor Description |  |
| 2.6 Work Back Log |  |
| 2.7 Total work |  |
| **Chapter 3: System Design** |  |
| 3.1 UML DESIGN |  |
| 3.1.1 Use Case Diagram |  |
| 3.1.2 Use Case Description (Schema for Detailed Use Cases) |  |
| 3.1.3 Traceability Matrix |  |
| 3.2 System domain model |  |
| 3.2.1 Interaction diagrams |  |
| 3.3 Database Design |  |
| 3.3.1 Class Diagram |  |
| 3.3.2 ERD |  |
| 3.3.3 Mapping |  |
| **Chapter 4: System Implementation & Testing Plan** |  |
| 4.1 Interface Design |  |
| 4.1.1 Home page |  |
| 4.1.2 Sign Up page |  |
| 4.1.3 Login page |  |
| 4.1.4 Search page |  |
| 4.1.5 Adding Educational Institutions page |  |
| 4.1.6 Joining Request for School page |  |
| 4.1.7 Add Training Center page |  |
| 4.1.8 Add Course page |  |
| 4.1.9 Administrator interface |  |
| 4.2 Test Plan |  |
| 4.2.1 Test Case 1: |  |
| 4.2.2 Test Case 2: |  |
| 4.2.3 Test Case 3: |  |
| 4.2.5 Test Case 4: |  |
| 4.2.4 Test Case 5: |  |
| **Chapter 5: Conclusion and Future Work** |  |
| 5.1 CONCLUSION |  |
| 5.2 Future Work |  |
| **References** |  |

* **Table of Figures**

|  |  |
| --- | --- |
| Figure 1. 1: Iterative UAT [1] |  |
| Figure 1. 2: Agile scrum framework at a glance [2] |  |
| Figure 1. 3: Sprint cycles and weekly Scrum meetings [3] |  |
| Figure 3. 1: Entire System Use Case |  |
| Figure 3. 2: Unregistered (Viewer) user Use Case |  |
| Figure 3. 3: Training center & educational administrator Use Case |  |
| Figure 3. 4: Student & parent Use Case |  |
| Figure 3. 5: Administrator Use Case |  |
| Figure 3. 6: Schema for Use case 1 |  |
| Figure 3. 7: Schema for Use case 2 |  |
| Figure 3. 8: Schema for Use case 4 |  |
| Figure 3. 9: Schema for Use case 5 |  |
| Figure 3. 10: Schema for Use case 6 |  |
| Figure 3. 11: Schema for Use case 7 |  |
| Figure 3. 12: Schema for Use case 8 |  |
| Figure 3. 13: Schema for Use case 9 |  |
| Figure 3. 14: Schema for Use case 11 |  |
| Figure 3. 15: Schema for Use case 12 |  |
| Figure 3. 16: Schema for Use case 13 |  |
| Figure 3. 17: Schema for Use case 14 |  |
| Figure 3. 18: Schema for Use case 16 |  |
| Figure 3. 19: Schema for Use case 18 |  |
| Figure 3. 20: Schema for Use case 19 |  |
| Figure 3. 21: Schema for Use case 21 |  |
| Figure 3. 22: Schema for Use case 22 |  |

* **Table of Tables**

|  |  |
| --- | --- |
| Table 1. 1: agile advantages and disadvantages |  |
| Table 2. 1: Business Requirement |  |
| Table 2. 2 Non-Functional Requirements |  |
| Table 2. 3: User story |  |
| Table 2. 4: Actor Description |  |
| Table 2. 5: Work Back Log |  |
| Table 3. 1: Mapping: System requirements to Use cases |  |

* **Definitions, Acronyms, and abbreviations (Glossary)**

|  |  |
| --- | --- |
| UML | Unified Modeling Language |
| UAT | User Application Test |
| MCU | Microcontroller |
| IDE | Integrated Development Environment |
| PC | Personal Computer |
| UI | User Interface |
| CPU | Central Processing Unit |
| RAM | Random Access Memory |
| HD | Hard Disk |
| Li-ion | Lithium-Ion |
|  |  |

**Chapter 1: Introduction**

**1.1 INTRODUCTION:**

In recent decades, smart devices have played the most significant role in human life,

and power consumption has been one of the most trending issues. As a result,

batteries and their lifespan have been one of the most sophisticated topics.

All people suffer from smart devices batteries that must be replaced periodically, and these battery's performance decreases almost daily.

This report discusses one of the smart solutions that depend on using the global standards of charging batteries taking into consideration the advice of specialist companies in this field like Apple and Samsung.

The overall goal of the project is to design, implement and produce (as a sale product) a tiny hardware device (tiny to be portable for any user) that can enable or disable charger functionality in real life.

We start to design a software that co-operate with this hardware in windows platform, designing of software is built to be able to develop to be multiplatform in future and able to contact with smartphones or tablets regardless their brands.

We design the software (that connects with the hardware device) to be compatible with Windows platforms.

Designing the software takes the concept of functional programming, parallel programming, and object-oriented programming in (C#9.0 & dotnet 5.0) to be much more flexible to has the ability to develop to be multiplatform in the future and able to contact with smartphones or tablets regardless of their brands.

**1.2 Motivation:**

The motivation for designing this application came from some reasons:

Firstly: As students in the computer science department, the laptop is one of the most important tools for us, so we want to save its battery’s life as much as we can.

This is the first serious contribution to overcome this issue by building an intelligent device from scratch that saves batterie’s life, which based on Embedded Systems by combining software with hardware.

On other hand: for each student in any faculty or institution, a laptop can be one of the most precious especially for those who suffer from poverty.

Replace the battery with a little bit of money for these students can literally kill their studies progress, so we are seeking for saving money for these students to save their brains.

**1.3 Problem Definition:**

Batteries do not have an infinite lifespan. Most battery manufacturers claim that their products have a 300-500 cycle rating [ ].

After this, batteries would be unable to carry as much energy and will only be able to fuel the computer for limited periods of time [qur21]

The wrong way of charging Lithium-Ion batteries which included in laptops leads to a decrease in batterie’s lifespan by 25% which in turn lead to replacing them frequently,

as well as frequently and randomly power outages which lost user’s work.

Leaving the battery connected to the charger when the battery is completely charged while you are using it may lower battery lifespan if you do it repeatedly [sam21]

“So, a good range to aim for when charging a Li-ion battery is from about 40% to 80%

in one go. Try not to let the battery drop below 20%” [qur21]

As battery’s cost has risen, it is necessary to save money and enhancing batteries’ performance, so solving this problem now not later is the correct choice.

**1.4 Background:**

**1.5 Related works:**

**1.6 Scope:**

This system allows users to monitor their smart devices' batteries from the face of performance, user can furthermore expected time that this battery MUST be replaced without surprises in some critical times like exams or during emergency works, this became available thanks to corporation between the system and database that generates detailed and aggregated reports explains batteries performance (decrease and increase) in the charge for each 1% and its relationship with machine recourses usages such as processor, ram and hard disk.

**1.7 The system development life cycle is the Plan-driven methodology:**

* How Plan-driven model works:

“Plan-driven software development is a more formal specific approach to creating an application, Plan-driven methodologies all incorporate repeatability and predictability, a defined incremental process, extensive documentation, up-front system architecture, detailed plans, process monitoring, controlling and education, risk management, verification, and validation.” [wikiversity]

System and Software design

Implementation and unit testing

Requirement analysis

Operation and Maintenance

Integration and system Testing

* There are separate identified phases in the waterfall model:
  + - **Requirement analysis:** development team meets project stakeholders to identify needs and establish requirements to satisfy project goals. Requirement analysis initially is a feasibility study that identifies functional and non-functional requirements with its constraints. Documentation is generated.
    - **System and Software design:** development team converts requirements into representation of architectural model. Prototypes are created with functional algorithms and data structures. Creating accurate design in this stage is vital because after completing design stage, development team does not go back to the design previous stage. Design is finalized and overall, of the project outcome depends on it.
    - **Implementation and unit testing:** design is transformed into software.
    - **Integration and system Testing:** software domain is tested for bugs. Different sets of tools are available for testers and developers to detect and fix defects in the software.
    - **Operation and Maintenance:** stage where product is updated (patched), to meet changing needs and environments. Fixing undetected bugs during testing stage and defects that might arise during updates.

Plan-driven model Advantages and Disadvantages

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| The waterfall model is mostly used for sophisticated system engineering projects. | The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway. In principle, a phase must be complete before moving onto the next phase. |
| Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing customer requirements. |

**1.8 Software development diagrams:**

**1.9 Development tools:**

* + - Microsoft Visual Studio Enterprise 2019 Version 16.10.2

A picture containing text, electronics, loudspeaker

Description automatically generated

* + - Microsoft SQL Server Management Studio version 18.9.1

Graphical user interface, application

Description automatically generated

* + - JetBrains ReSharper 2021.1.3 built on 2021-5-25

A picture containing text, businesscard

Description automatically generated

* + - GitHub website

**Logo

Description automatically generated**

**2.0 Frameworks and programming languages used:**

* + - * Dot-Net core 5.0

A picture containing text, clipart

Description automatically generated

* + - * C-Sharp 9.0

A picture containing text, clipart

Description automatically generated

* + - * Microsoft SQL Server 2019

Logo, company name

Description automatically generated

* + - * git version 2.31.1. windows.1

Logo, icon

Description automatically generated

**Chapter 2: Technicality Overview Part One**

**A closer look at**

**The Hardware Engineering of The Project**

**Chapter 3: Technicality Overview Part Two**

**A closer look at**

**The Software Engineering of The Project**

**3.1 INTRODUCTION:**

The system as a software department classified into two main categories, the coding part using C-Sharp programming language and database part using Microsoft SQL Server engine.

For the coding part, there are many different technologies used together for the seeking of enhancement application performance such as LINQ, Multithreading, and functional programming concepts with some classes.

For the database part, indeed we mean to make the database so simple to make it lite at the run time, and to make the application's user does not feel of the sophisticated operations running in the background.

In this chapter, we will look deep into the C-Sharp code and architecture of the database in build and queries.

**Functional Requirements**

* Recording every transaction that battery take it without user intervention, this transaction recorded when battery percentage increase or decrease by 1%
* Ensuring that all transactions in the database, not more than a specific number to minimize resources needed by the application.
* Enable the application’s user to see every transaction as live streaming without needing of restarting the application.
* Enable the application’s user to filter transactions depend on date and time.
* Enable the application’s user to hide the application, this feature MUST freeze every non-functional operation and make the application work in the background with the usage of minimum resources.
* Enable the application’s user to generate report contain some analysis about his/her battery’s life such:
  + Average Battery Usage Up/Down 1%
  + Average Processor Usage.
  + Average RAM Usage.
  + Average Hard Disk Usage.
  + # Of Charger Online.
  + # Of Charger Offline.
  + # Of Records.

A picture containing text, electronics, computer

Description automatically generated

**Non - Functional Requirements**