**Introduction**

Organizations and Institutions depend more and more on software solutions in today’s quickly changing technology environment to streamline their operations, increase efficiency, and satisfy the rising expectations of their stakeholders. To ensure these software systems are not only functional but also in line with the unique requirements and goals of the organization or institution, the creation of such systems calls for a systematic and organized approach.

The aim of employing a common Software Development Lifecycle (SDLC) model is the first of a series of questions that go into important facets of software development. The complete software development process is outlined by SDLC models, which also give a methodical approach to project management, risk reduction, and quality control. Successful software development within an organization or institution requires an understanding of the rationale for choosing a certain SDLC model.

We now investigate the software solution’s design components. We start by delving into the data model, a key component that establishes the rules for how data is organized, kept, and accessible within the system. For the purpose of assuring data integrity, security, and effectiveness, an effective data model must be designed. We’ll also quickly touch on the most effective ways to create data models.

We shall then talk about the system’s functional model. This calls for specifying how the software solution will work in practice, taking into account features, data gathering and handling, transitions, and information. A software solution must be created with a thorough functional model in order to satisfy the organization’s current needs while also being flexible enough to meet future ones. We will also look at the most effective methods for creating the system’s functional model.

In conclusion, these issues deal with important facets of software development inside a company or institution, starting with the selection of an appropriate SDLC model and continuing with the design of the data model and functional model. These components work together to create the framework for the effective execution of software solutions that support the organization’s objectives and successfully address the needs of its stakeholders.

**Introduction to Company**

In Sri Lanka, Durdans Hospital is a reputable and well-known tertiary healthcare facility that prioritizes patient care for both local and foreign patients. Durdans is the first healthcare facility to get Joint Commission International's Gold Seal of Approval accreditation.

**Company Statements Documents**

* **Vision**

Their Vision is to be acclaimed as the most trusted healthcare provider.

* **Mission**

The Mission is to integrate advanced technology with empowered healthcare teams to deliver exceptional patient-centric care.

**Organization Hierarchy**

Specialist Surgeons

Middle Level of Hospital Management

Lower Level of Hospital Management

Medical Students

Physician Assistant

Nursing

The Silent Doctors

Normal Specialists

Hospital Administrators

General Manager

Top Level of Hospital Management

**Current System Explanation**

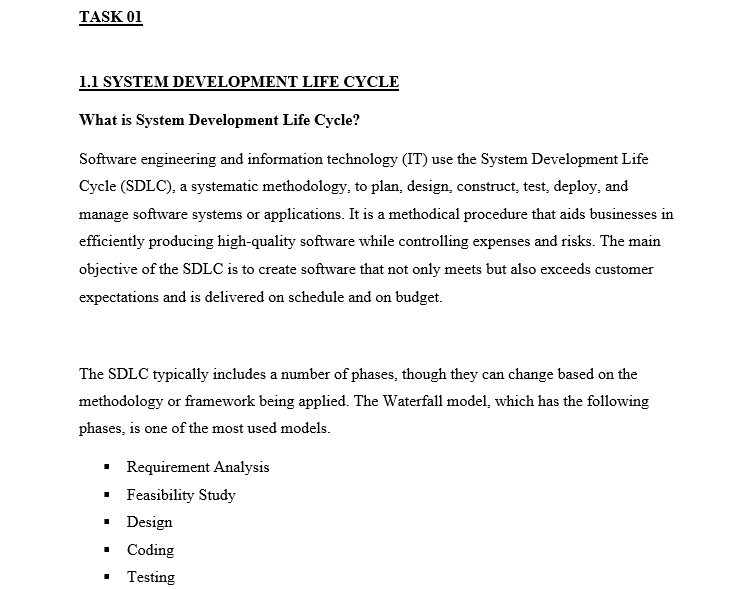
Durdans Hospital have few different systems. But mainly they use two systems to gather information from the patients. Among those two systems only one system is works as ann online system the other one works as a manual system and then they put those details to the system. The Durdans Pharmacy use the online system to reduce queue and they use the manual system to take down the patients details who comes to the hospital daily.

**Drawbacks at the Current System**

The system they use to take patient details who comes to the hospital is a major drawback which can be shown. It takes more time to input details to the system and it can be shown as a waste of time. The system which the pharmacy uses to reduce the queue has a face recognition system but it is still not in the online system they use because it is still on experimenting that system.

**Proposed System Solution**

According to the drawbacks and current system, here by proposed agile would be the best to use in a new system to use to take patient information. Because Agile development refers to carrying out the SDLC in a series of several iterations of full ’mini-SDLCs’ over a period of time. The goal of agile is early customer satisfaction, priority of allowing change, priority of communication over documentation. If we compare Agile with Waterfall Development, Agile Development can help saving more time than Waterfall Development because in Agile two phases can be done in one time but in Waterfall until one phase finishes another phase won’t start. Hospital System is changing quickly so we should use a model which can be useful and support that kind of changes. So, as I propose Agile Development would be the best module which we can use for the betterment of a new system.



The different phases of system development life cycle are shown in below.

Feasibility Study

System Design

Requiremnent

Analysis

Coding

Testing

Deployment

Maintenance

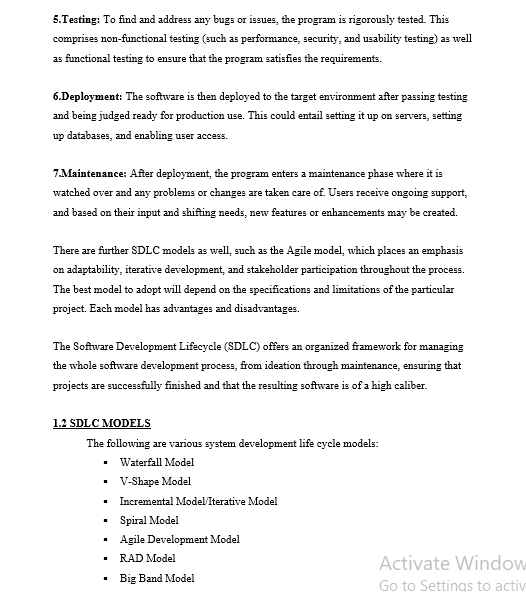
System Development Life Cycle

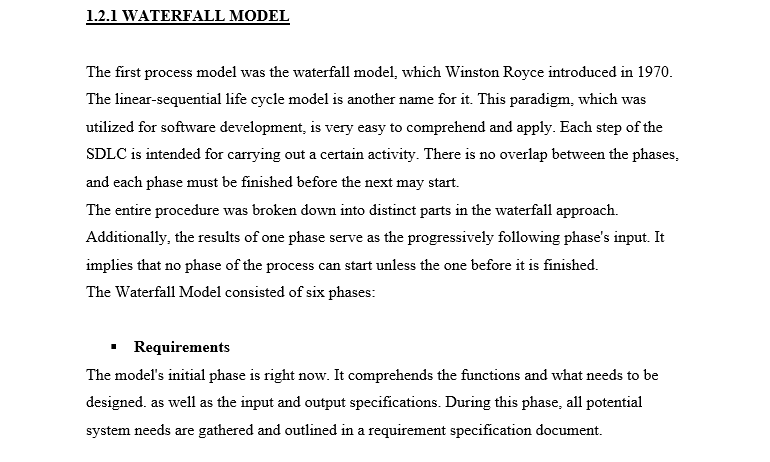
**1.Requirement Analysis:** Project objectives, scope, finances, and timetables are established at this phase. The project's feasibility is evaluated as the project's key stakeholders are identified. The essential resources, tasks, and schedules are outlined in a project plan.

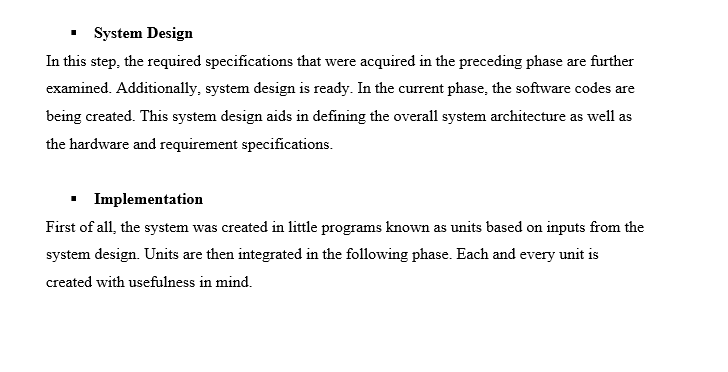
**2.Feasibility Study:** During this stage, stakeholders are surveyed regarding the software system's requirements. Both functional and non-functional requirements—such as those related to performance, security, etc.—are included in these specifications.

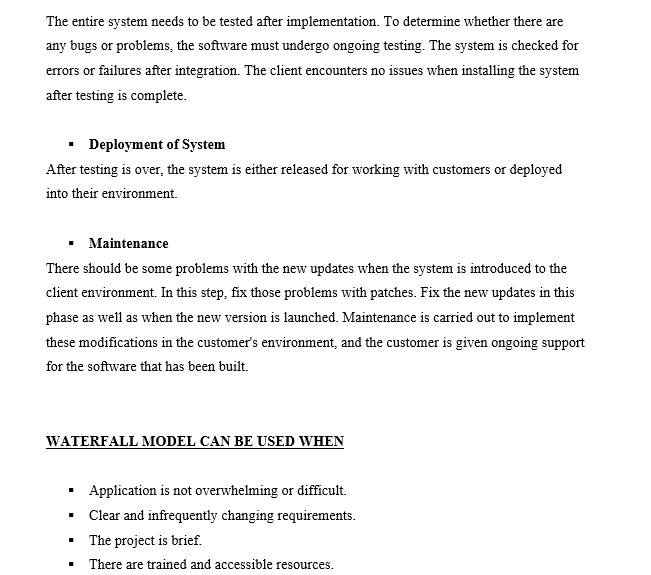
**3.Design:** The architecture of the system is created during this phase depending on the requirements. Planning for data storage and retrieval, establishing the software's structure, and creating high-level and technical design specifications are all included in this.

**4.Coding:** During this stage, the software is really coded. The source code is created by programmers in accordance with the design requirements. Unit testing is another step in this process that verifies the functionality of individual components.









**ADVANTAGES OF WATERFALL MODEL**

* Appropriate for smaller projects with clearly defined criteria.
* Simple and uncomplicated to comprehend.
* Each phase is processed and finished separately.
* Each stage of development must be finished before moving on to the next.
* Each step includes a review process and particular deliverables.
* Tasks are easily arranged and each level is well described.
* Process and outcomes are thoroughly documented.
* Any modifications to software are done during the development process.

**DISADVANTAGES OF WATERFALL MODEL**

* Errors can only be corrected during the phase.
* Until late in the life cycle, no functional software is developed.
* Poor model for ongoing, complicated, and object-oriented projects that last a long time.
* With this model, risk and uncertainty are significant.
* It is challenging to gauge development using phases.
* Is unable to adapt to changing needs.
* Each phase's timing and cost estimates are challenging.
* Developers and testers spend a lot of time on documentation.
* Customers' insightful input cannot be included into the continuing development stage.

**1.2.2 V-SHAPE MODEL**

This paradigm focuses on the sequential execution of processes and is very similar to the waterfall model, often known as the verification and validation model. In this methodology, testing is given more weight, and testing guidelines are created before any code is written. This implies that every stage of the development cycle has a direct counterpart in the testing stage. Additionally, there is a model that is quite rigorous. Only after the preceding phase is complete does the next one begins.

The V-Shaped Model consisted of eight phases:

* **Requirement Analysis**

The initial stage of the development cycle is this. In this phase, look for and comprehend the product requirements from the viewpoint of the client. Customers must be thoroughly communicated with in order to understand their expectations for the needs. This is a really essential truth, so you should be ready. The functional interface, performance, data, security, and other requirements are included in the document titled "User Requirements Document."

* **System Design**

System Design During this phase, system developers examine the requirements document to gain an understanding of the proposed system's operations. The system design includes knowledge of the specifics on the entire hardware and communication architecture for the product being developed. Here, the developer considers options and strategies that, depending on user needs, can be put into practice.

* **Architecture Design**

High level design, often known as this phase's concentration, is considered to be on system architecture and design. It gives the developer an overview of the available products, platforms, and solutions.

When the system design was further divided into models that covered various functionalities.

Usually, more than one technical solution is put out before a selection is made based on financial viability.

* **Module Design**

The actual software components are designed during the module design phase, which is also known as low level design. Here, the real logic for each system component is defined. The design's compatibility with the other modules in the system architecture and other external systems is crucial. When the designed system is divided into smaller pieces, the programmer can begin writing code right away.

* **Coding Phase**

The optimum programming language is chosen in this phase based on the system and architectural requirements. The actual coding for the system modules created during the design phase is done during this stage. The coding was done in accordance with norms and guidelines.

* **Unit Testing**

Although not all faults may be found by unit testing, it does assist to reduce errors early on. Unit testing is testing at the code level. This stage tested the accuracy of the smallest entity's operation when separated from the other codes.

* **Integration Testing**

This stage is connected to the phase of architectural design. This stage is used to evaluate how well the system's internal modules get along with one another and communicate.

* **System & Acceptance Testing**

System testing guarantees that application expectations are met and are directly related to the design process. The functionality, interdependence, and communication of the entire system are evaluated. Acceptance testing involves evaluating the product environment and is connected to the business requirement analysis step. Additionally, acceptance testing confirms that the provided system satisfies user requirements and is prepared for use immediately.

**V-SHAPE MODEL CAN BE USED WHEN,**

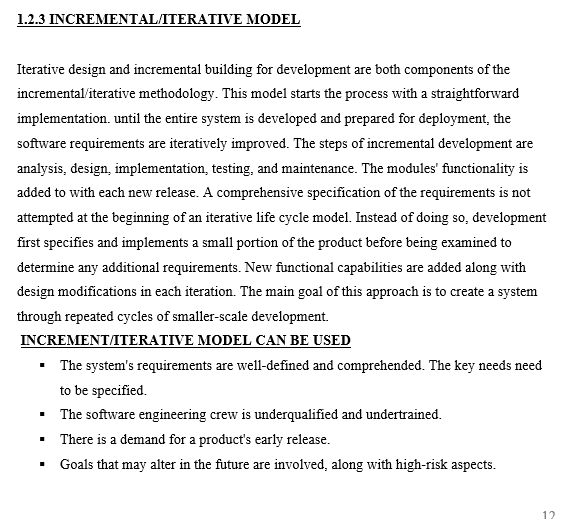
* Definition of the product is steady.
* The project's and its development team's understanding of technology is not dynamic.
* The requirements are fixed, well-defined, and properly documented.
* There aren't any ambiguous criteria.
* The project is not long.

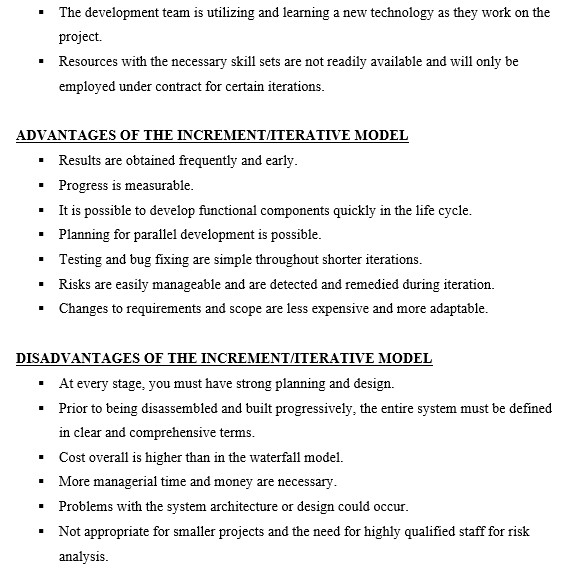
**ADVANTAGES OF A V-SHAPED MODEL**

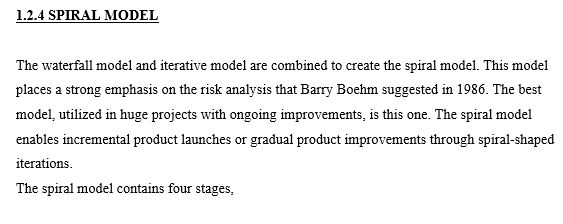
* Work effectively for smaller projects with clearly defined needs.
* Simple and uncomplicated to comprehend.
* Highly regimented model where each phase is finished separately.
* Each step includes a review process and particular deliverables.
* Saves a great deal of time on testing tasks like planning.
* Because of the rigidity of the model, it is simple to manage.

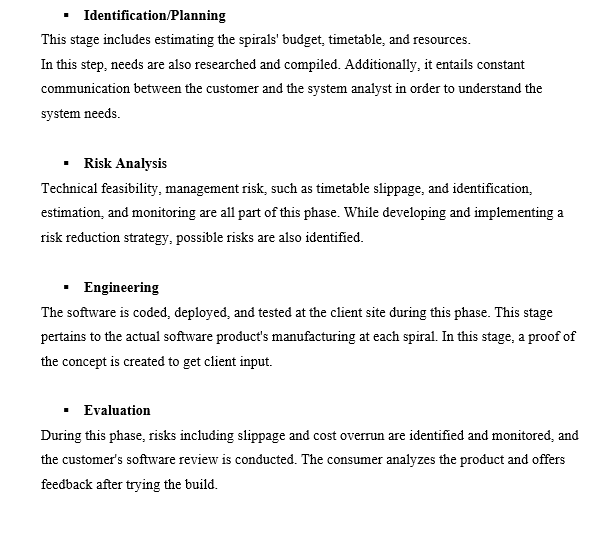
**DISADVANTAGES OF V SHAPED MODEL**

* Unsuitable as a model for intricate and object-oriented projects.
* Poor model for continuing, protracted projects.
* High danger, ambiguity, and expense.
* Once an application is being tested, it might be challenging to update functionality.
* Unsuitable for projects whose requirements are subject to a high to moderate risk of change.
* Until late in the life cycle, no functional software is developed.





.



**SPIRAL MODEL CAN BE USED**

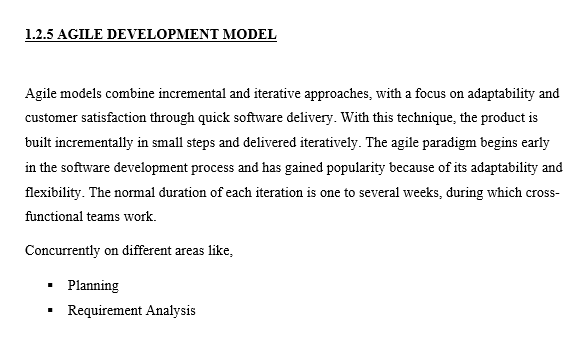
* For high-risk, medium-sized, and large projects.
* The requirements are difficult.
* The project requires frequent adjustments.
* The users are uncertain of their demands.
* The creation of a prototype is appropriate.

**ADVANTAGES OF THE SPIRAL MODEL**

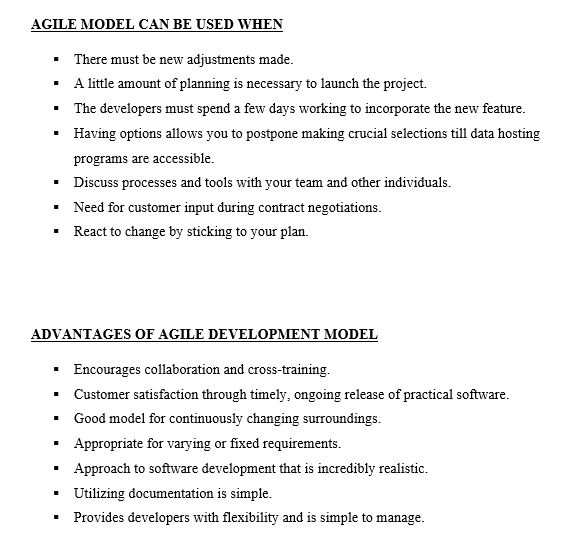
* It is possible to divide development into smaller, riskier parts.
* Development happens quickly, and features are introduced methodically.
* Cost estimation becomes simple.
* Prototypes may be used extensively.
* Modifications to the requirements can be made.
* There is always room for customer comments.
* Later on, it is possible to add functionality or make adjustments.

**DISADVANTAGES OF SPIRAL MODEL**

* Process and management are more complicated.
* The project's end may not be recognized right away.
* It might continue forever.
* Not appropriate for minor, low-risk initiatives.
* Documentation has extra stages because of this.
* Risk assessment calls for professionals.



* Coding
* Unit Testing
* Accepting Testing

.

**DISADVANTAGES OF AGILE DEVELOPMENT MODEL**

* Not appropriate for managing complicated dependencies.
* Greater risk of extensibility, maintenance, and sustainability.
* Depends mainly on interactions with customers.
* There is a significant amount of individual dependency.
* Due to a lack of documentation, transferring technology to new team members may be fairly difficult.
* Only senior programmers have the decision-making skills needed during the development process.

**1.2.6 RAD MODEL**

Rapid Application Development is RAD. Without any precise planning, this strategy is built on prototyping and iterative development. This model's procedure is a modification of the waterfall model. This model aims to create software in a brief amount of time.

In this paradigm, the parts or functions are created concurrently as if they were smaller projects or prototypes, then combined to create the whole product for quicker product delivery. The RAD methodology concentrates on acquiring client requirements through focus groups or workshops. This can be promptly sent to the client to get feedback on the delivery and their needs. The most crucial component of this paradigm is that reusable prototypes must be created in order for it to work.

The RAD model contains five phases.

* **Business Modeling**

To find the crucial data, a thorough business study is conducted. This product model has been created. The information flow between several corporate functions is discovered. Additionally, it was done in order to gain it.

* **Data Modeling**

In this phase, data is gathered, examined, and analyzed to create a set of data objects that are essential to the business. The relationship between the data that was established and defined, as well as its specifics with regard to the business model. The characteristics of data sets are recognized and described.

* **Process Modeling**

In this phase, any modifications or improvements to the data object sets are defined. To achieve a particular business aim, the defined data objects are translated to achieve the business information flow. The processes of adding, removing, retrieving, and altering are described by data objects.

* **Application Generation**

In this stage, the system itself is constructed. Automation tools are used to code process and data models into functional prototypes.

* **Testing and Turnover**

The data flow and interface between all the components must be thoroughly tested during this phase. As a result, overall testing time for the model is reduced because each iteration of the prototype includes independent testing. As a result, the majority of the programming components have previously been tested, lowering the possibility of any significant problems.

**RAD MODEL CAN BE USED WHEN**

* The system must be created quickly, and use will be required throughout the system's life cycle.
* There is less risk and known requirements.
* A system can be deployed incrementally over the course of two to three months if it is modularized.
* A budget is large enough to cover the cost of automated tools for code generation as well as designers for modeling.

**ADVANTAGES OF RAD MODEL**

* Flexible and change-resistant.
* Progress is measurable.
* Iteration cycles can be brief.
* Shorten the development period.
* Increase the component's capacity for reuse.
* Promote consumer feedback.
* Initial reviews are happening quickly.
* Productivity in a short period of time with fewer personnel.
* Lower the project's overall risk.

**DISADVANTAGES OF RAD MODEL**

* Requires developers and designers with advanced skills.
* High reliance on modeling abilities.
* More complexity exists in management.
* Appropriate for scalable, component-based systems.
* High reliance on modeling abilities.
* More management complexity
* Appropriate for projects with quicker development periods.

**1.2.7 BIG BANG MODEL**

The Big Bang Model doesn't adhere to any set procedure. The majority of the effort is devoted to coding and software development, and there is barely any structured development procedure. The development process starts with the necessary funds and resources, such as the input and output of the software development process. As a result, some forethought is necessary. This technique involves directing all available resources toward coding. The needs are recognized and carried out in accordance with the developer's wishes. Additionally, this strategy is ideal for small projects involving just one or two developers. Excellent for practice or academic tasks. Additionally, this model carries a very high chance of failure due to the possibility of poorly understood requirements.

**ADVANTAGES OF BIGBANG MODEL**

* Simple to control.
* There are not many resources needed.
* Very basic model.
* No preparation is necessary.
* Good tool for newcomers to learn from.
* Very accommodating.

**DISADVANTAGES OF BIGBANG MODEL**

* Not recommended for intricate and object-oriented designs.
* High danger and uncertainty.
* Poor model for continuing, protracted projects.
* Possibility of cost.
* If requirements are not understood, a developer may prove to be highly expensive.

**Entities:**

* Patient
* Doctor
* Appointment
* Reception
* Record
* Medicine
* Ward
* Nurse

**Attributes:**

* Patient:
  + Patient\_ID
  + Name – F\_Name

L\_Name

* + Address
  + Phone\_No
  + DOB
  + Date – Admitted Date

Discharged Date

* Doctor:
  + Doctor\_ID
  + Name – F\_Name

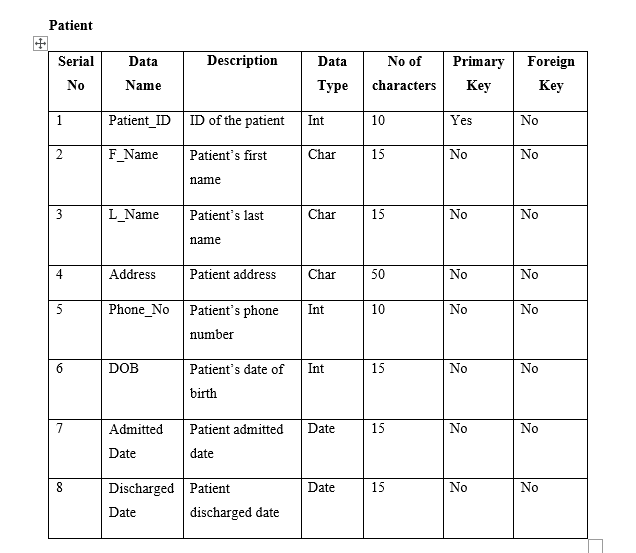
L\_Name

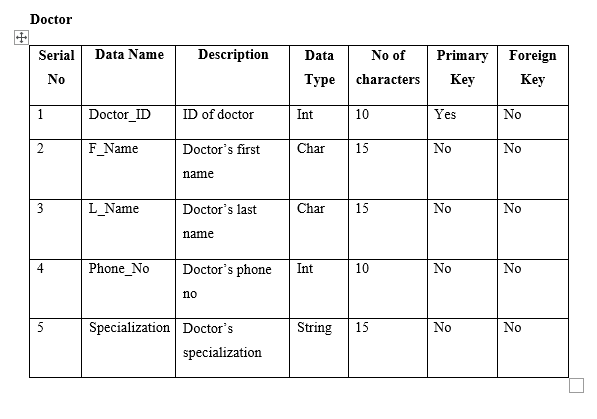
* + Phone\_No
  + Specialization
* Appointment:
  + Appointment\_ID
  + Date
  + Time
  + Patient\_ID
  + Doctor\_ID
  + Ward\_ID
  + Nurse\_ID
* Reception:
* Reception\_ID
* Record:
  + Record\_No
  + Patient\_ID
  + Appointment\_ID
* Medicine:
  + Code
  + Price
  + Quantity
* Ward:
  + Ward\_ID
  + Capacity
  + Ward Name
* Nurse:
  + Nurse\_ID
  + Name – F\_Name

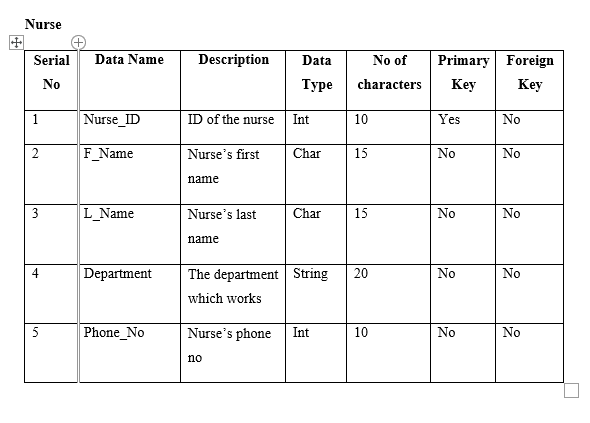
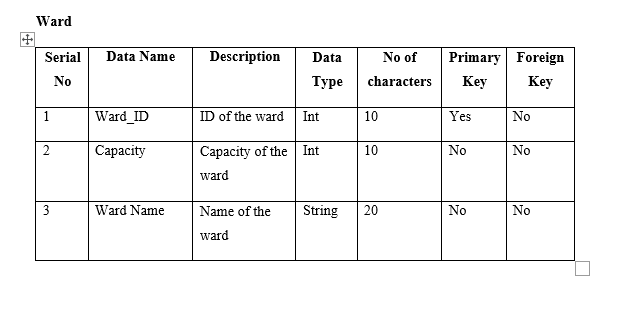
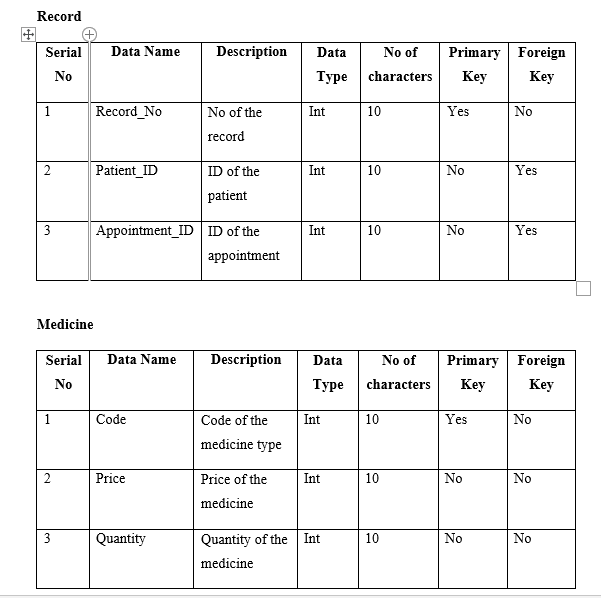
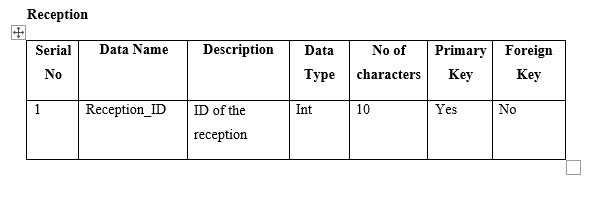
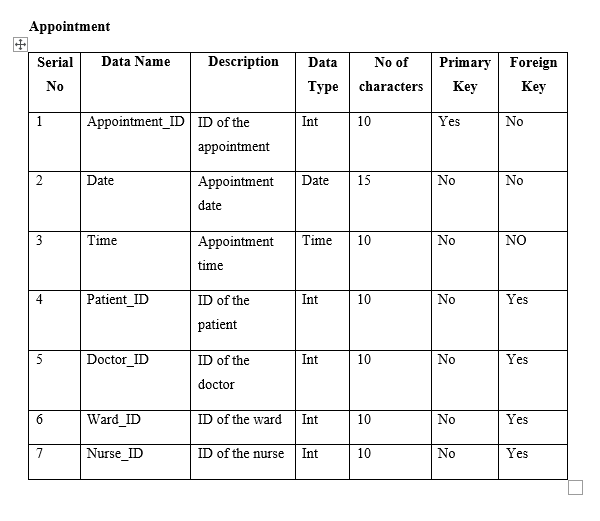
L\_Name

* + Department
  + Phone\_No

**Data Dictionary**







**Conclusion**

The System Analysis and Design assignment has shown the value of systematic techniques in addressing difficult technological and business problems. It emphasizes the switch from inefficient, manual processes to effective, well-designed ones for increased production and client satisfaction. Adopting these concepts can help firms adapt, innovate, and prosper in the digital era, which is essential in today's dynamic business climate.