Chapter 1 - Introduction

1.1 Background of the study:

Hospitals are essential part of our lives, providing us with medical facilities. It is necessary for the hospitals to keep track of its daily activities & records of its patients, doctors, nurses, and other staff so that there will be smooth running of the institution.

Everyone knows that keeping records using traditional methods can be very challenging, cumbersome, time consuming, prone to errors and inefficient. Especially, when considering the fact the population is increasing and you have so much to keep track of. For this reason, there has to be a better way to handle this problem.

The following challenges that are facing our current hospitals highlighted above prompted this research work. Although some the problems mentioned have been addressed by other researchers, I still want to address a few issues that I think haven’t been done much justice to. The end product of this research is to have a working system that will help solve of the limitations faced by some of our medical institutions. I then decided to title it as “Hospital Management System Using Bingham University as Case Study”.

1.2 Statement of problem:

In Bingham University, on several occasions I witnessed patients (students) not being attended to because they have either forgotten their ‘clinic cards’ or have forgotten the number on the card. The patient would be forced to manually go through the registration book until he/she finds their record. You can imagine an ailing person being subjected to this unimaginable stress before being given attention. It has happened to me sometimes back and sadly enough, has also claimed the life of a friend of mine. I asked a sick student going through the record for her details how he was feeling and he said that he was feeling like hell. On the other hand, the workers can’t help the situation at that moment because according to their ethics, they need the card to trace your folder before administering medication to you. Throwing all critical skills out of the window, it is very clear that this is posing a big threat to the students, placing their lives on the line and must be addressed to.

1.3 Motivation:

With the above problem stated above, the reason for this research work is highly justified and is worth being given due consideration. We are talking about the lives of the potential leaders of tomorrow being at stake and shouldn’t be toyed with. Going outside Bingham University where the population is high, this problem can be by definition, very dangerous to the society at large. For that reason, I am motivated to help contribute as much as I can to help reduce some of this problems or peradventure even get rid of it completely.

1.4 Aims and objectives

The aim of this research is to have a working system that will automate almost up to 80% of the manual record keeping techniques that is currently in use in the clinic. This is of course the bottom-line for the whole project, and will greatly reduce the stress the staff undergo while searching for folders and likewise the trauma patients face going through the registration to get their records.

1.5 Purpose of the study

The purpose of this research is summarized below:

1. Automate as many existing manual systems as possible
2. Having a paperless record keeping system
3. Have a reliable database backup in case of natural disasters
4. Ease the stress patients face when they forget their card numbers
5. Enhance communication among the staff for faster rendered services
6. I have a burden for the patients who happen to be victims of this hapless exploitation

1.6 Significance of the study:

This study is of highly significant to Bingham University and the entire society at large. Considering the problems it addresses, there is surely no doubt that all my time, resources, energy and effort committed to this work is worth it. The study will contribute to human knowledge in many aspects, one of them is harnessing the power of technology to solve the issue of misplaced clinic cards which end patients in distress and trauma.

The research is also an evidence that what I’ve been taught in class was not in vain, because I am using the knowledge to solve a real life problem. For anybody that values and cherishes humanity, admittedly, it’s undoubtedly true that human life comes first in everything. As the slogan says “health is wealth”. The possibilities of this software is limitless as long as time and space remain defined, because it’s going to be constantly improved over time and not be dumped upon completion.

With that, I believe that my effort is very much rewarding, and somebody will one day add more to what I’ve done. The Bingham community is going to be greatly imparted and of course the society at large.

1.7 Organization of the work:

1.8 Definition of terms:

1. CFD: – Context Flow Diagram
2. DFD: – Data Flow Diagram
3. IDE: – Integrated Development Environment
4. SRS: – Software Requirement Specification.
5. GUI: - Graphical User Interface
6. JavaScript – A web programming language
7. Node js - A JavaScript environment that allows JavaScript to be run on the server
8. Express js – A Node js web framework
9. MongoDB - A No-SQL database server
10. JSON – Javascript Object Notation
11. Sublime Text – A text editor

Chapter 2 – Literature Review

2.1 Overview

In order to understand the concepts associated with records management and or computer based records management systems, it is imperative to examine and analyze published material from experts regarding the field. The purpose of this review is to analyze and examine and obtain experience as regards the creation and archival processing of electronic records. The review is based on an exhaustive assessment of the literature on computerized electronic management and electronic records, and contains an overview of the main concepts associated with the creation of an electronic records management system from the perspective of published works by experts.

The use of computers in our medical centers today is very crucial. As highlighted earlier, computers greatly reduce stress and cost, at the same time improve performance. The following is a list of related works that have been done to improve the current existing system in the industry. Their valid reasons justify the fact that this research is worth my time.

2.2 Records

A record is recorded information produced or received in the initiation, conduct or completion of an institutional or individual activity and that comprises content, context and structure sufficient to provide evidence of the activity regardless of the form or medium.

According to the National Archives and Records Administration (NARA), records include “…all books, papers, maps, photographs, machine-readable materials, or other documentary materials, regardless of physical form or characteristics, made or received ... or in connection with the transaction of public business and preserved or appropriate for preservation by that agency or its legitimate successor as evidence of the organization, functions, policies, decisions, procedures, operations, or other activities of the Government or because of the informational value of the data in them.”

2.3 Existing System

Hospitals currently use a manual system for the management and maintenance of critical information. The current system requires numerous paper forms, with data stores spread throughout the hospital management infrastructure. Often information (on forms) is incomplete, or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. Multiple copies of the same information exist in the hospital and may lead to inconsistencies in data in various data stores.

Below we will study some of the existing systems and see some their strengths and weaknesses as well.

2.3.1 Computer use in the medical field

(Amber Weber and Morgan Swink 8 Sept. 2014) This research clearly states that a patient’s prescribed medications (past and present) can also be stored in a computer system in a hospital. This makes it much easier to transfer any prescriptions and data that a patient needs to local or national drug stores or hospitals. Past hospital visits and billing information can be stored and kept for further use or future hospital experience. For example, doctors keep a computer handy anytime they prescribe a medication. They can use the computer to find out what medicines the patient may be allergic to or what medicines may interfere with one another. Important information is at the tip of their fingers and this can be very helpful.

2.3.2 Improving Patient Records

(The Computer-Based Patient Record: An Essential Technology for Health Care National Academies Press, 1997 Don E. Detmer, Elaine B.) Computer-based patient records and the systems in which they function are becoming an essential technology for health care in part because the information management challenges faced by health care professionals are increasing daily.

Technological progress makes it possible for CPRs and CPR systems to provide total, cost-effective access to more complete, accurate patient care data and to offer improved performance and enhanced functions that can be used to meet those information management challenges. CPRs can play an important role in improving the quality of patient care and strengthening the scientific basis of clinical practice; they can also contribute to the management and moderation of health care costs.

2.3.3 The Computer-Based Patient Record System

(National Academy Press, 15 Jan 1997) Computer-based patient records, as defined by the committee, could positively affect the quality of patient care in at least four ways. First, they offer a means of improving both the quality of and access to patient care data. Second, they allow providers to integrate information about patients over time and between settings of care. Third, they make medical knowledge more accessible for use by practitioners when needed. Fourth, they provide decision support to practitioners.

Research efforts could also benefit from computer-based patient record keeping in two key ways. First, improved data and access to those data would be available to researchers. Second, research findings could be communicated to practitioners through computer-based patient record systems

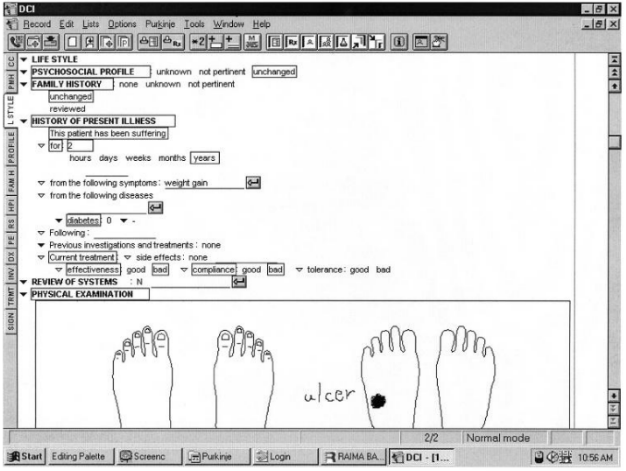
2.3.4 Impact of a Computer-based Patient Record System on Data Collection, Knowledge Organization, and Reasoning

(J Am Med Inform Assoc. 2000 Nov-Dec; 7(6): 569–585.) The CPR system is one example of such a system.10 Computer-based patient record systems are designed to allow physicians to directly enter patient data, findings, and notes into a computer system that may be linked to hospital-wide databases and decision support systems.

The objectives of implementing such systems include replacing hand-written paper-based records with CPRs to improve access to information and quality of health care decision making. However, use of these systems may also affect, in unanticipated ways, fundamental cognitive processes involved in health care.

The user can also search for a section of the system by entering key words. The data entry method is based on a point-and-click or pick- list technology, where the user clicks on categories on the screen for which values (e.g., blood pressure) can be entered, either by typing in the value, clicking to indicate the presence or absence of a finding (and the level of severity of symptoms), or using the pen to enter a value or note. In addition, the system incorporates software for handwriting recognition to allow for data entry by pen.

The system can also display sections of the relevant clinical items to the user; for example, it will adjust the display of information slightly depending on the patient's particular complaint (the system has a number of “filters,” which are basically variations of a general template for the organization of medical findings, based on general medical conditions). Screenshot of the CPR system below:



2.3.5 The Emergence of National Electronic Health Record Architectures in the United States and Australia: Models, Costs, and Questions

(J Med Internet Res. 2005 Jan-Mar; 7(1): e3.) The electronic record is at the center of the IOM's goal of eliminating most handwritten clinical data by the end of this decade. Electronic records are superior to paper records because they decrease error due to handwriting problems and ease physical storage requirements. Additionally, electronic records simultaneously leverage other error-reducing technologies and render them coherent.

EHR models present significant additional advantages because of their potential to deliver a longitudinal record that tracks all medical interactions by a particular patient and provide comprehensive data across populations. Thus, the IOM envisions a longitudinal collection of electronic health information for and about individuals and populations as feeding data into error-reducing “knowledge and decision support systems.”

2.3.6 Electronic health records not a panacea, researchers say

(By Diana Manos, December 14, 2009) In a study published Monday in the U.S. journal Milbank Quarterly, researchers at the University College of London (UCL) said they identified fundamental and often overlooked tensions in the design and implementation of EHRs. The study was based on findings from hundreds of previous studies from all over the world.

Researchers said their findings have implications for President Barack Obama’s election promise to establish electronic health records for every American by 2014, and for other large-scale EHR initiatives around the world.

Professor Trish Greenhalgh, lead author of UCL’s Department of Open Learning, said EHRs are often depicted as the cornerstone of a modern healthcare, capable of making care better, safer and cheaper. Yet, clinicians and managers the world over struggle to implement EHRs. "Depressingly, outside the world of the carefully-controlled trial, between 50 and 80 per cent of electronic health record projects fail – and the larger the project, the more likely it is to fail," Greenhalgh said.

"Our results provide no simple solutions to the problem of failed electronic patient records projects, nor do they support an anti-technology policy of returning to paper. Rather, they suggest it is time for researchers and policymakers to move beyond simplistic, technology-push models and consider how to capture the messiness and unpredictability of the real world,” according to Greenhalgh.

2.3.7 Designing a patient-centered personal health record to promote preventive care. Unpublished manuscript

(Krist AH, Rothemich SF, Loomis JF, et al) Greater provision of preventive services: A randomized controlled trial involving 2,250 patients in 8 primary care practices found that the proportion of IPHR users who were up-to-date with all recommended preventive services increased by 11.5 percent in the 16 months after implementation, well above the 2.2-percent increase among nonusers during the same time period.

The provision of many preventive services increased more for users than nonusers. For example, after 4 months, rates of colorectal, breast, and cervical cancer screening increased by 19.2, 15.0, and 11.7 percent, respectively, among IPHR users, compared to 3.9, 2.8, and 0.5 percent among nonusers. Similarly, rates of tetanus and pneumococcal immunization increased by 13.3 and 14.2 percent, respectively, among IPHR users, compared to increases of just 4.6 and 8 percent among nonusers. Both observational cohort analyses yielded similar findings; for example, one analysis found that, compared to nonusers, IPHR users were more likely to receive a tetanus vaccination (60 versus 38.9 percent) and screening for colon cancer (51.9 versus 33.6 percent), cervical cancer (58.1 versus 38.6 percent), elevated cholesterol (89 versus 64.8 percent), and diabetes (93.2 versus 80.8 percent).

2.3.8 Survey: Funding pressures are holding up the paperless NHS

(Claire 13 october, 2015) It explored the extent to which these leaders see digital healthcare as a priority – and revealed something of a mixed picture.

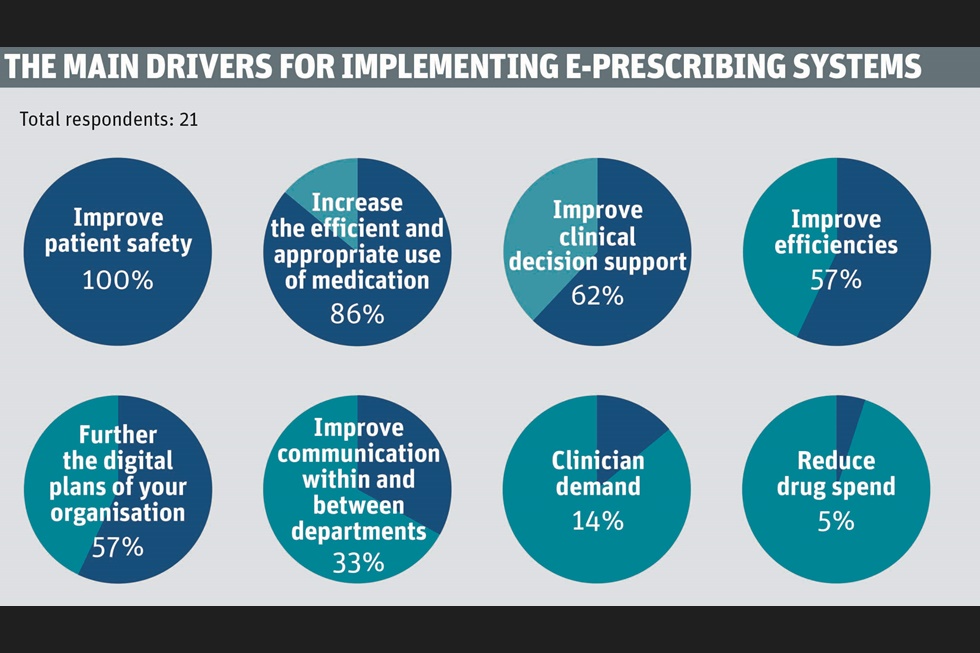
Take the creation of a paperless health service. While 87 per cent described it as high or very high priority, the remaining 13 per cent saw it as low or very low priority. Notably, even among those who did see it as important, there was concern about how realistic an aim it is.

“Not achievable by 2018,” said one acute trust chief executive in the North West. “It will be dependent on funding,” reported an acute trust information management and technology lead who nonetheless saw going paperless as a very high priority.

On electronic prescribing, the picture was even more varied. Mr. Kelsey argues: “The evidence is clear: electronic prescribing systems that support clinicians to ensure the right medicine is provided to the right person in the right quantity halve medication errors.”

Yet over half of those we surveyed said they had no such system in place at their organization, and that clinicians are still prescribing via pen and paper. Of those who did have such a system, only 29 per cent said it was fully implemented across their entire organization. Said one: “Lack of funds has prevented full rollout.”

This is the diagram for the survey:



2.3.9 How developments in technology and data in the NHS are improving outcomes for patients

(Sir Jeremy Heywood, Cabinet Secretary and Head of the Civil Service, 12 December 2014) In other parts of our lives we take technology and its benefits for granted. We are able to book our holidays and manage our finances online. But although 61% of UK adults have a smartphone and 76% of us access the internet every day, only 2% of the population have any kind of digital interaction with the NHS. This needs to change, not only for patient care, but also to improve the efficiency of the NHS, and in order to support the integration with social care services.

It is very important that DH and its partner organizations across health and social care have made further commitments in the area of data and technology, with the publication of Personalized Health and Care 2020. This sets out a framework for how the health and care sector will use data and technology to transform outcomes for patients and citizens.

2.3.10 Electronic Medical Record/Electronic Health Record Systems of Office-based Physicians

(Chun-Ju Hsiao, et al, 2014) The estimate of all or partial EMR/EHR systems was obtained from the question, “Does this practice use electronic medical records or electronic health records (not including billing records)?” In addition to the question asking about all or partial EMR/EHR systems, physicians also reported the computerized functionalities of their practices. EMR/EHR systems were classified as basic or fully functional.

There has been an increasing trend in EMR/EHR use among office-based physicians from 2001 through the preliminary 2010 estimates (Figure 1). Combined data from the 2009 surveys (mail survey and in-person survey) showed that 48.3% of physicians reported using all or partial EMR/EHR systems in their office-based practices. About 21.8% of physicians reported having systems that met the criteria of a basic system, and about 6.9% reported having systems that met the criteria of a fully functional system, a subset of a basic system. Preliminary 2010 estimates from the mail survey showed that 50.7% of physicians reported using all or partial EMR/EHR systems, similar to the 2009 estimate. About 24.9% reported having systems that met the criteria of a basic system, and 10.1% reported having systems that met the criteria of a fully functional system, a subset of a basic system. Between 2009 and 2010, the percentage of physicians reporting having systems that met the criteria of a basic or a fully functional system increased by 14.2% and 46.4%, respectively. Due to questionnaire modifications in 2010, survey items used to define basic and fully functional systems are slightly different from 2009. The diagram for their findings is shown below:



2.3.11 Federal Health Architecture

(FHA pub., 2013) The CONNECT open source software enables health IT systems to securely communicate via nationally recognized standards. With CONNECT, information can be shared securely via the Internet among doctor’s offices, federal agencies, state agencies, disability organizations, public health organizations, pharmacies and other health stakeholders. This enables health professionals to request, send and receive medical records so critical information can follow patients as they navigate through the health care system.

CONNECT was developed by more than 20 federal agencies working together through FHA. Rather than each federal agency independently building its own separate Nationwide Health Information Network compliant gateway, they decided to collaborate and create a single platform to enable the secure exchange of health information.

The collaboration that led to CONNECT’s development was unprecedented in the federal government. It has saved the government an estimated $200 million in development costs versus each agency building its own solution. Equally as compelling is the result of this collaboration. Once the solution was built, federal agencies decided to release CONNECT to the entire healthcare industry to be used by any organization without any cost for the software license.

2.3.12 Patient Engagement within the Patient-Centered Medical Home Model

(Jed C. Constantz, July 2017) This research focused on what best practices might exist among patient-centered primary care delivery to highly-activate their patients. The problem that was studied was what specific aspects of primary care delivery, such as physician leadership, staffing models, staff training and development, and electronic tools and resources, support or undermines the successful high-level activation of patients. The research method used was a qualitative study using the Delphi Method of collecting subject matter expert (SME) opinions from among four classifications of primary care practice team members as study participants.

Chapter 3 – Research Methodology

3.1 Introduction

The system being developed is going to be designed in such a way that it will accommodate changes along the way. Improvements are going to be made. Out of the numerous software development methodologies, I will discuss briefly on three and choose the one that best fit into my requirement. They are listed and explained below.

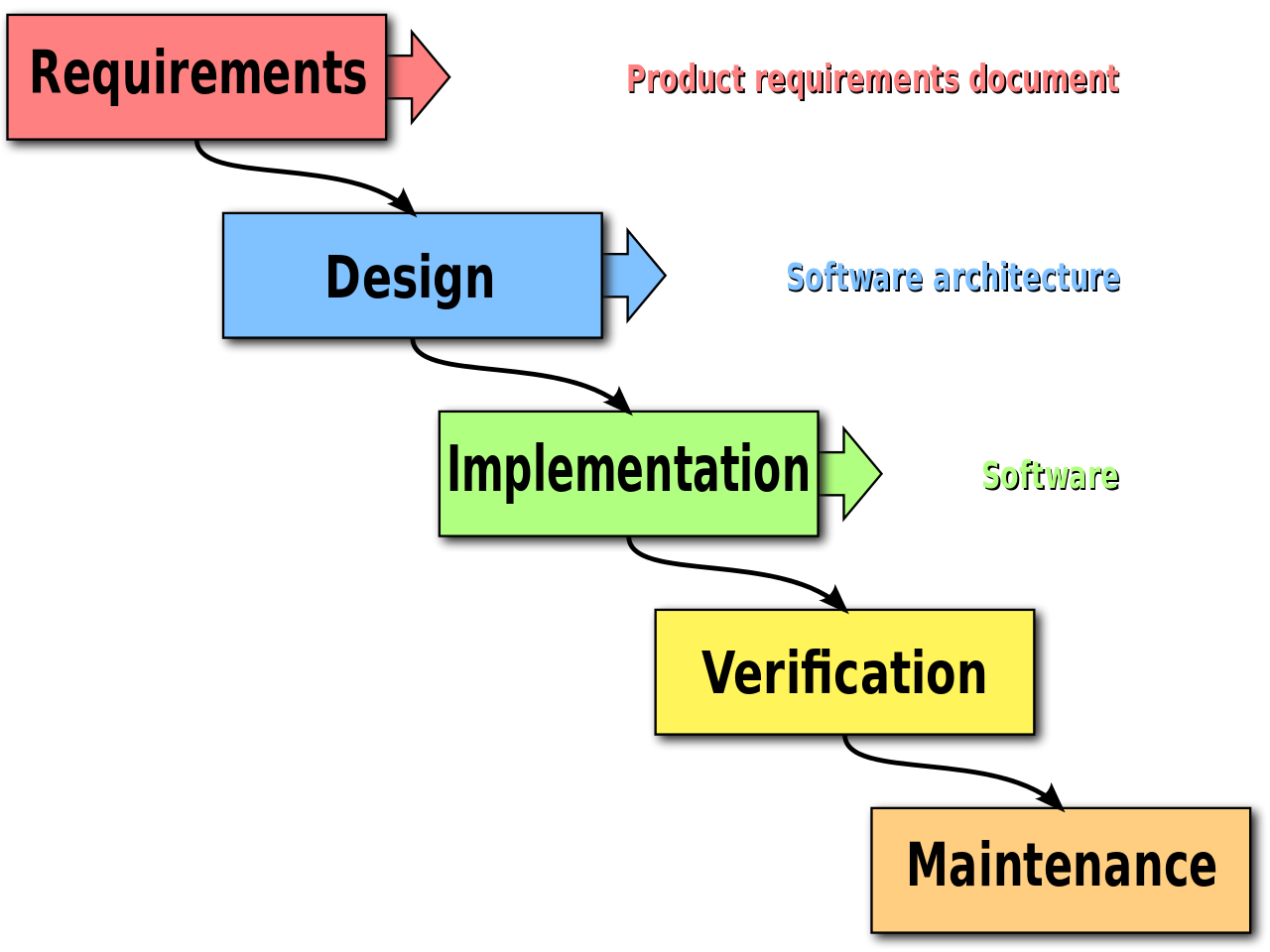
1. Waterfall model
2. Agile model
3. Iterative model

3.1.1 Software Development Cycle

What is Waterfall model?

The Waterfall Model was first model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. This type of software development model is basically used for the for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In this model, software testing starts only after the development is complete also, phases do not overlap. (ISTQBExamCertification, 2015)

Waterfall model stages



When to use the waterfall model:

1. This model is used only when the requirements are very well known, clear and fixed.
2. Product definition is stable.
3. Technology is understood.
4. There are no ambiguous requirements
5. Ample resources with required expertise are available freely
6. The project is short.

Advantages of waterfall model:

1. This model is simple and easy to understand and use.
2. It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
3. In this model phases are processed and completed one at a time. Phases do not overlap.
4. Waterfall model works well for smaller projects where requirements are very well understood.

Disadvantages of waterfall model:

1. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
2. No working software is produced until late during the life cycle.
3. High amounts of risk and uncertainty.
4. Not a good model for complex and object-oriented projects.
5. Poor model for long and ongoing projects.
6. Not suitable for the projects where requirements are at a moderate to high risk of changing.

. (ISTQBExamCertification, 2015)

What is agile model?

Agile software development methodology is a process for developing software (like other software development methodologies – Waterfall model, V-Model, Iterative model etc.). In English, Agile means ‘ability to move quickly and easily’ and responding swiftly to change – this is a key aspect of agile software development as well. (Wikipedia, 2017)

Brief overview of Agile Methodology

1. In traditional software development methodologies like Waterfall model, a project can take several months or years to complete and the customer may not get to see the end product until the completion of the project.
2. At a high level, non-Agile projects allocate extensive periods for Requirements gathering, design, development, testing and User Acceptance Testing, before finally deploying the project.
3. In contrast to this, agile projects have Sprints or iterations, which are shorter in duration (Sprints/iterations can vary from 2 weeks to 2 months) during which pre-determined features are developed and delivered.
4. Agile projects can have one or more iterations and deliver the complete product at the end of the final iteration. (Johnny Moore, 2014)

Characteristics of agile development

1. Modularity
2. Iterative
3. Time-bound
4. Incremental
5. Convergent
6. People-oriented
7. Collaborative

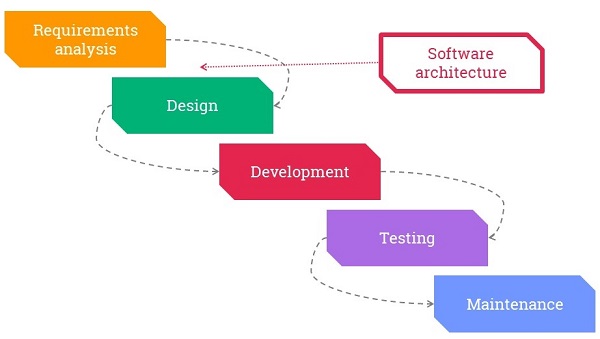
Advantages of agile methodology

1. .In agile methodology the delivery of software is unremitting.
2. The customers are satisfied because after every Sprint working feature of the software is delivered to them.
3. Customers can have a look of the working feature which fulfilled their expectations.
4. If the customers has any feedback or any change in the feature then it can be accommodated in the current release of the product.
5. In Agile methodology the daily interactions are required between the business people and the developers.
6. In this methodology attention is paid to the good design of the product.
7. Changes in the requirements are accepted even in the later stages of the development.

Disadvantages of agile methodology

1. In agile methodology, the documentation is less.
2. Sometimes in agile methodology, the requirement is not very clear hence it’s difficult to predict the expected result.
3. In few of the projects at the starting of the software development life cycle, it is difficult to estimate the actual effort required.
4. The projects following the agile methodology may have to face some unknown risks, which can affect the development of the project.

Agile development stages

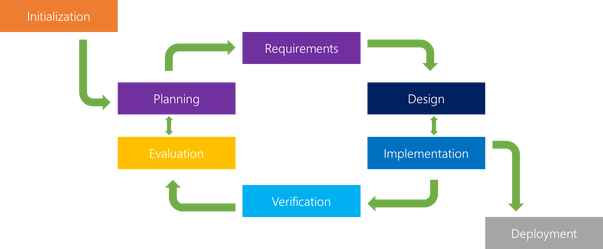


1. The Agile Process Flow
2. Concept - Projects are envisioned and prioritized
3. Inception - Team members are identified, funding is put in place, and initial environments and requirements are discussed
4. Iteration/Construction - The development team works to deliver working software based on iteration requirements and feedback
5. Release - QA (Quality Assurance) testing, internal and external training, documentation development, and final release of the iteration into production
6. Production - Ongoing support of the software
7. Retirement - End-of-life activities, including customer notification and migration

Iterative model

The iterative model is a particular implementation of a software development life cycle (SDLC) that focuses on an initial, simplified implementation, which then progressively gains more complexity and a broader feature set until the final system is complete. When discussing the iterative method, the concept of incremental development can also be used liberally and interchangeably, which describes the incremental alterations made during the design and implementation of each new iteration. (Powell-Morse, 2015)

Interactive model stages



Advantages of the Iterative Model

1. **Inherent Versioning**
2. **Rapid Turnaround**
3. **Suited for Agile Organizations**
4. **Easy Adaptability**

Disadvantages of Iterative Model

1. **Costly Late-Stage Issues**
2. **Increased Pressure on User Engagement**
3. **Feature Creep**

Why Agile?

Agile development methodology provides opportunities to assess the direction of a project throughout the development lifecycle. This is achieved through regular cadences of work, known as sprints or iterations, at the end of which teams must present a potentially shippable product increment. By focusing on the repetition of abbreviated work cycles as well as the functional product they yield, agile methodology is described as “iterative” and “incremental.” As earlier stated, this software is going to encounter a lot of iterations and changes and hence agile is best suited for our need. (The Agile Movement, 2008)

Diagram (ERD)?

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases. (Smartdraw, 2014)

In E-R diagram, there are 3 main Components:

|  |  |  |
| --- | --- | --- |
| Symbol | Name | Description |
|  | Entity | An entity can be any object, place, person or anything. |
|  | Attribute | An Attribute Describes a property or characteristics of an entity. |
|  | Relationship | A Relationship Describes relation between entities. |

E-R Diagram of Hospital Management System

Doctor/Nurse

Patient

Treatment

1

Data Flow Diagrams

The Data Flow Diagram (DFD) is a graphical representation of the flow of data through an information system. Data flow diagrams are used by systems analysts to design information-processing systems but also as a way to model whole organizations. The main merit of DFD is that it can provide an overview of what data a system would processes, what transformations of data are done, what data are stored and which stored data is used, and where the result is flow.

Standard Symbols used in DFD:

|  |  |  |
| --- | --- | --- |
| Symbol | Name | Function |
|  | Data Flow | Used to connect processes to each other. The arrowhead indicates direction of data flow. |
|  | Process | Performs some transformation to input data to output data. |
|  | Source or sink.(external entity) | A source of system inputs or sink of system outputs. |
| Data Store | A repository of data. Arrowheads indicate net inputs or net outputs to the store. |

Level 0 DFD

A context diagram is a top level (also known as Level 0) data flow diagram. It only contains one process node (process 0) that generalizes the function of the entire system in relationship to external entities. In level 0 DFD, system is shown as one process.

The Level 0 DFD shows how the system is divided into 'sub-systems' (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

Admin

Patient

Personal details

Level 1 DFD

Admin

Personal details of patient. Update Patient db

Update

Admin

Personal Details of doctor db Update Doctor

Non-Functional Requirement

Reliability

This application is a reliable product that produces fast and verified output of all its processes.

Availability

This application will be available to use and help them to carry out their operations conveniently.

Security

The application will be password protected. User will have to enter correct username, password and role in order to access the application.

Maintainability

The application is designed in such a way that it can easily be maintained. It will be easy to incorporate new requirements and featured in the individual modules.

Portability

The application will be easily portable on device that has a web browser installed on it. It is a cross-platform application

Functional Requirements

1. The admin can login to the system
2. The admin can register student
3. The admin can delete students
4. The admin can search for students using their names, age, level and id

CHAPTER 4

IMPLEMENTATION AND EVALUATION

4.1 Introduction

After observing this critical need of a patient record in Bingham university, I came up with the idea of a system that will stand in between this gap. This involves making the system user friendly, hence easy to use. The system is designed to make it possible for easy scaling as patients increase, it will not lag. It will also provide room for modifications as requirements may also change. This is achieved by the choice of programming platform and languages. I used different frameworks to achieve this.

4.2 Programming platform

The application is built using Node js, an event-driven, asynchronous, non-blocking server side runtime sitting on top of Google Chrome V8 runtime. It is known to be one of the fastest server side environment. I created a web server on to6p of the Node engine and it is blazing fast. I used Sublime Text, a text editor for coding. I also implemented a RESTful (Representational State Transfer) API (Application Programming Interface) service for communicating between the web server and the client. All the communication between the server and the client are done in form of JSON (JavaScript Object Notation). I also used MongoDB for the database. Then finally, used HTML, CSS and JavaScript for the rendering on the client side and other logic. This is obvious because they are the backbone of any web application. Mine is not an exception.

4.3 Choice of programming language and framework

JAVASCRIPT

Whenever you are talking about the web, it is not possible not mention JavaScript. This is the programming language that has eaten the web in all aspects ranging small, medium, and enterprise web applications. Every web browser has JavaScript language installed on it. Coupled to the fact that the language is easy to start working with and it’s ubiquity, I quickly jumped into it and started developing.

Back into history, the language was developed by Brendan Eich in 10 days during a hackathon at Netscape in 1994. This happens to be my birthday. Since then, the language had been rejected by many programmers because of its quirks.

In 2009, those programmers didn’t have option than to embrace it. The same language was made to be run on the server, serve as the server and handle many concurrencies. This has solved a lot of issues in the traditional web servers.

The most awesome reason is that I am using a single language across the stack for development. It is serving as my client side language, server side language and database as well. There’s no need of changing context when building the application

I am also using Express js, an MVC JavaScript server side framework and Angular js, a client side MVC framework. The combination of this stack is coined as “MEAN” stack (MongoDB, Express js Angular js, Node js). . Everything is JavaScript. There is a slogan for the language, “One language rule them all”

MONGODB

A NoSQL (originally referring to "non SQL" or "non relational") database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases (Wikipedia, 2017).

NoSQL databases are built to allow the insertion of data without a predefined schema. That makes it easy to make significant application changes in real-time, without worrying about service interruptions – which means development is faster, code integration is more reliable, and less database administrator time is needed. (MongoDB, 2017).

Considering the kind of database I am using, there is no predefined schema for this application; hence, the interactions between all the data entities are highly flexible.

The Benefits of NoSQL

When compared to relational databases, NoSQL databases are more scalable and provide superior performance, and their data model addresses several issues that the relational model is not designed to address:

1. Large volumes of rapidly changing structured, semi-structured, and unstructured data
2. Agile sprints, quick schema iteration, and frequent code pushes
3. Object-oriented programming that is easy to use and flexible
4. Geographically distributed scale-out architecture instead of expensive, monolithic architecture

Differences between SQL and NoQSL database

|  |  |  |
| --- | --- | --- |
| Database | SQL Databases | NOSQL Databases |
| Type | One type (SQL database) with minor variations | Many different types including key-value stores, document databases, wide-column stores, and graph databases |
| Development History | Developed in 1970s to deal with first wave of data storage applications | Developed in late 2000s to deal with limitations of SQL databases, especially scalability, multi-structured data, geo-distribution and agile development sprints |
| Examples | MySQL, Postgres, Microsoft SQL Server, Oracle Database | MongoDB, Cassandra, HBase, Neo4j |
| Schemas | Structure and data types are fixed in advance. To store information about a new data item, the entire database must be altered, during which time the database must be taken offline. | Typically dynamic, with some enforcing data validation rules. Applications can add new fields on the fly, and unlike SQL table rows, dissimilar data can be stored together as necessary. For some databases (e.g., wide-column stores), it is somewhat more challenging to add new fields dynamically. |
| Scaling | Vertically, meaning a single server must be made increasingly powerful in order to deal with increased demand. It is possible to spread SQL databases over many servers, but significant additional engineering is generally required, and core relational features such as JOINs, referential integrity and transactions are typically lost. | Horizontally, meaning that to add capacity, a database administrator can simply add more commodity servers or cloud instances. The database automatically spreads data across servers as necessary. |
| Development Model | Mix of open-source (e.g., Postgres, MySQL) and closed source (e.g., Oracle Database) | Open-source |
| Data Manipulation | Specific language using Select, Insert, and Update statements, e.g. SELECT fields FROM table WHERE… | Through object-oriented APIs |

Other features of the NoSQL databases include auto-sharding, load-balancing, replication, integrated caching and indexing. (MongoDB, 2017)

Haven listed all these features of NoSQL databases; also, being written in C/C++/JavaScript, it is obvious that it is not a bad choice.

4.4 Testing

Software testing is a process of executing a program or application with the intent of finding the software bugs. To the best of my ability, I carried out two types of tests namely development and user testing.

Development test

Development testing: this was carried out during the development of the software. It is done so as to make sure that there are no bugs before deployment into production environment. I tested for many things in the application. Below are the different things I tested during the development process.

1. I tested for the database connectivity before the application even starts running so that it can keep track of all “CRUD” operations in the system.
2. I tested for all RESTful API endpoints and made sure they are serving the required resources.
3. I tested for broken links
4. I tested for all environment variables
5. I tested for available ports on the machine

User test

User testing is the process of giving the application to the user of the software to check for any possible bugs. In my case, I didn’t give the clinic officials but instead gave it to some people around me to use the application and give me feedback, which they did. Below are the different things they tested.

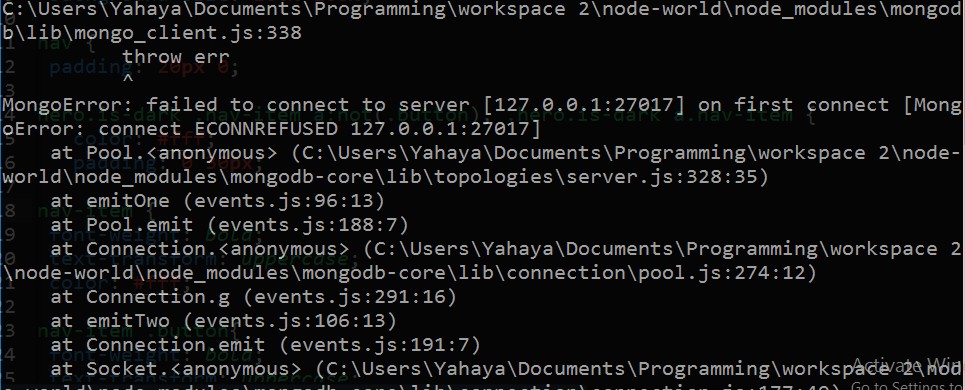
1. They tested for user UI/UX (User Interface/ User Experience).
2. Simple navigation
3. Speed of loading pages
4. Security
5. Hardware requirement

4.5 Test Results

I got some output after the test processes. Some parts of the application were working as intended but some were not. The results from these tests made me to improve some of the bad functionalities of the application. Though it took me couple times to fix those issues, I finally addressed them, to be honest, not all. The user interface needs more design

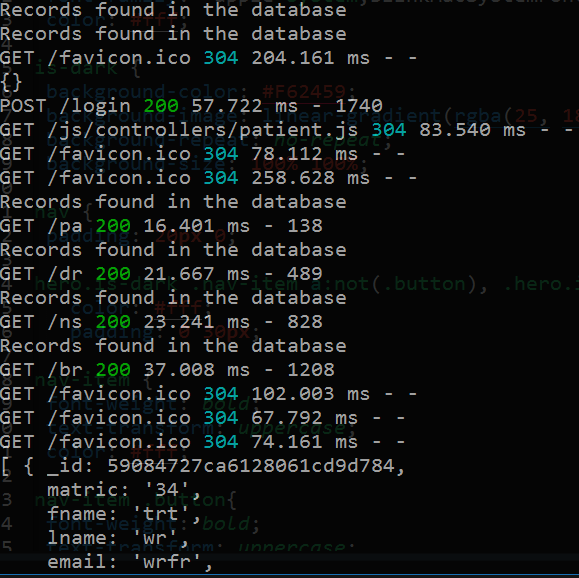
Development test result

* I had a persisting issue of not connecting to the database when the application starts.



Solution: I used an ORM (Object Relational Mapper) called MongoJS to handle the connection logic.

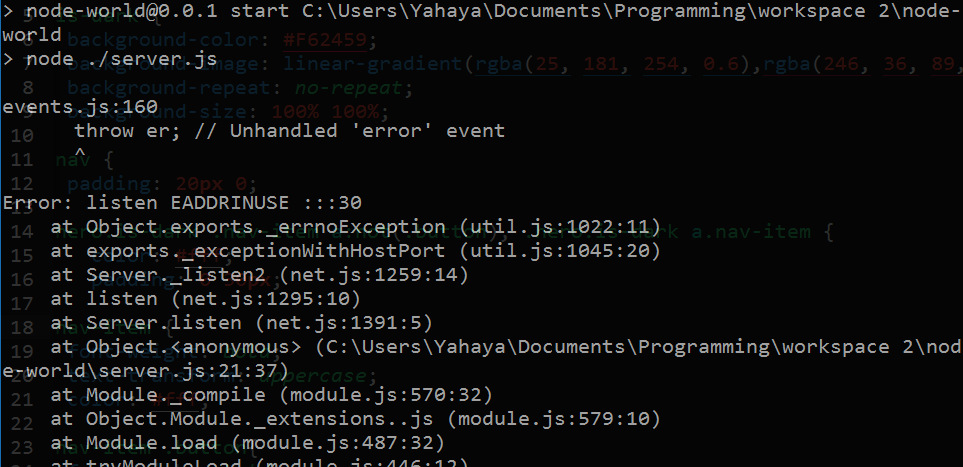
* There were no API endpoint errors because I was served with status code of 200, meaning all endpoints are working correctly.



* I didn’t encounter any broken links throughout the application development.
* I had an issue of error of not loading all the necessary environment variables, examples is the session secret. I didn’t have a way to load it to the system’s runtime.

Solution: I downloaded a module from npm (Node Package Manager) called ‘dotenv’ and it solved that issue.

* The last error I encountered in development test was using a port more than once to start the server.



Solution: I ensured that only one server terminal window was open and in use.

User test result

1. After testing for the UX/UI, they really complained that the interface was not looking nice and that I should improve it at least, a bit.

Solution: I used a CSS framework called Bulma to enhance the design, but there’s more work to be done to improve the design.

1. The users suggested that the navigation I used was too complex; they recommended making it simpler.

Solution: I simplified the navigation and made the application to be an SPA (Single Page Application). There are other parts that require the same feature, but I could not implemented that.

1. The speed of page loads and resource serving was really awesome according to them, they didn’t spot any bug with that.
2. Security features have not been fully implemented since login credentials are stored in session memory. But yet, the credentials entered must match with the ones stored on the server before anyone access the application.
3. The hardware requirement I specified was okay and didn’t show any sign of lag or slow performance, so it the application has passed the hardware test.

4.6 Implementation

This is the stage where the actual application comes into life. After using several platforms and technologies, I ended up with the workable version of the application. The key part players of this application are:

1. Jade: A HTML preprocessor that compiles to HTML. It is really easy to work with because of its clear syntax. It is a functional templating language that uses mixins, embedded JavaScript and so many things
2. CSS: As earlier mentioned, this is the heart of any web application design. It gives a visual sense of appeal to the user. It is responsible for beautifying the whole application and makes everything looks attractive.
3. JavaScript (Client side): The heartbeat of the whole web application. It is responsible for the application’s logic and makes everything happen.
4. Node js (Server side): An event-driven, asynchronous, non-blocking server side runtime built on top of Google Chrome V8 engine. This is the server itself. It is so powerful that you write the code to spin up your own webserver. It is really low level and flexible.
5. Express js: It is a web framework sitting on top of Node js to handle all the native Node API interactions on your webserver. It makes things such as routing and authentication a bit easier when compared to the native Node APIs.
6. MongoDB: The application has a sort of persistence to data storage. MongoDB handles that aspect. It stores data in a BSON format. It makes it easy to interact with all the API endpoints because you are basically not parsing any data when you receive it. You just store it directly in the database.

Login

The application has a single point of access. As stated earlier, there is no much security implementation to this system since the credentials are all stored in-memory storage. That is one of the limitations of the system that would be addressed in the next version because the project is open source and a work in progress. The admin enters a username and a password, the password is sent to the server in JSON format, the compared to the login credentials stored on the server and then grant access to the user if they match up and returns a 403 unauthorized status code if they do not. I used the simplest login implementation in the application.

Dashboard

On successful login, the admin is taken to the admin dashboard. The dashboard contains summary of the whole application data. Things such as the number of patients in the clinic, number of staff, number of death and birth recorded. It also has a graphical representation of all these information for easy understanding and to track all the metrics.

Logs

The logs is where all the entries regarding the admitted patients, birth rate and death rates are being entered. In provides all the input fields that are required to make these entries and sent to the server for data persistence.

Registration

This component is where the admin registers the patient into the system. Information such as the patient’s name, age, blood group, state, address, etc. are all collected in the provided input fields. Upon successful registration, the patient’s information can then be readily accessed in the patients page.

Patients

This section contains all the registered patients in the system. The admin can view all the registered patients and can also delete them. There is a search area where the admin can search for a patient. The search query can accept any input and then returns a result of the patient(s) matching that query. The search algorithm uses a filter technique to sort the result based on the input. The admin can then select the patient and view the patient’s information. The admin can also add treatment details administered to the patient and it keeps track of the admin that administered the treatment and the date and time of the treatment.

Admissions

This is the part of the application is responsible for keeping track of all admitted patients in the clinic. It displays all the list of admitted patients and gives the option to discharge the patient whenever their treatment is over.

4.7 Data Source

Primary data collection

1. Raw data (also known as primary data) is a term for data collected from a source. Raw data has not been subjected to processing or any other manipulation, and are also referred to as primary data.
2. Primary data is a type of information that is obtained directly from first-hand sources by means of surveys, observation or experimentation. It is data that has not been previously published and is derived from a new or original research study and collected at the source such as in marketing.
3. Primary data collection are observed and recorded directly from respondents. The information collected is directly related to the specific research problem identified. All the questions that one asks the respondents must be totally unbiased and formulated.

Secondary data collection.

Secondary data is data collected by someone other than the user. Common sources of secondary data for social science include censuses, organizational records and data collected through qualitative methodologies or qualitative research. The investigator conducting the research, by contrast, collects primary data.

In this project, I used only primary data collection method because I went to the school clinic (Bingham University Clinic) and asked obtained my data directly from the nurse. I obtained a form that contains the data that they always ask a patient to supply before getting registered in the clinic. I also verbally asked some students to also obtain my data for the research.

Chapter 4 - Implementation and evaluation

4.1 Implementation

This piece of code is responsible for rendering the homepage. It is written in jade. A html preprocessor.

Source code for homepgage

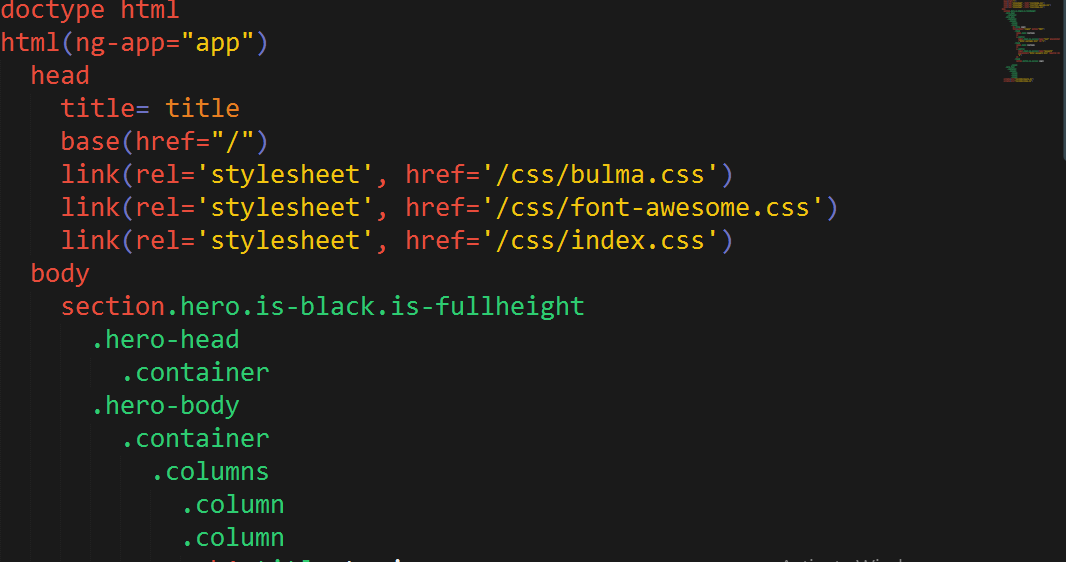
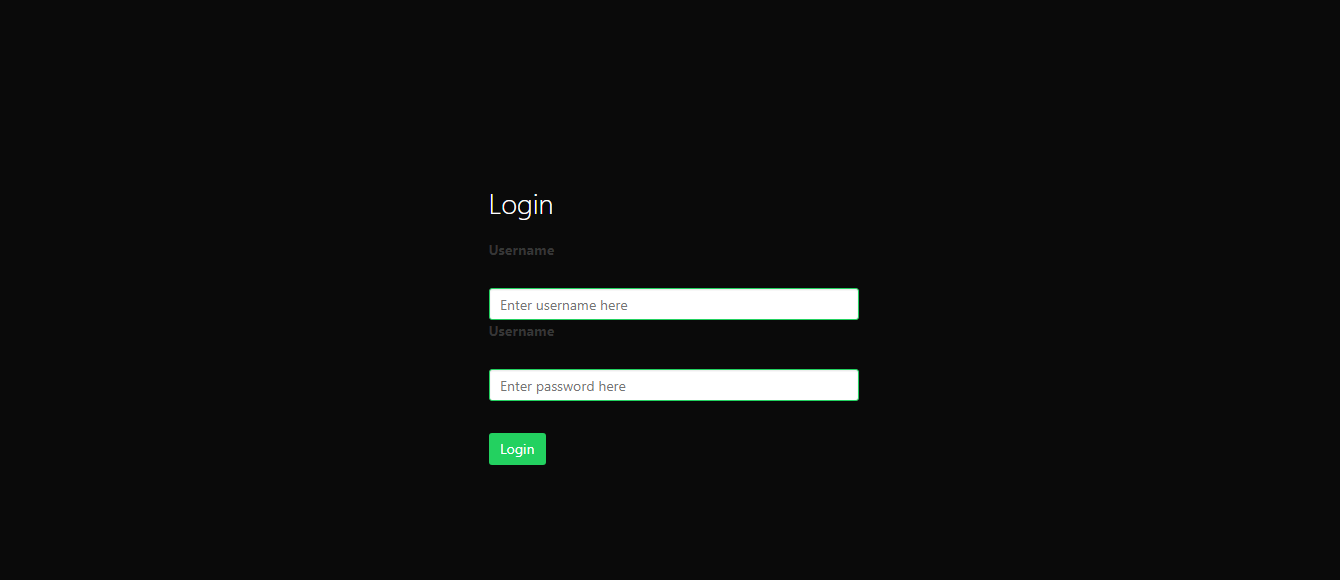
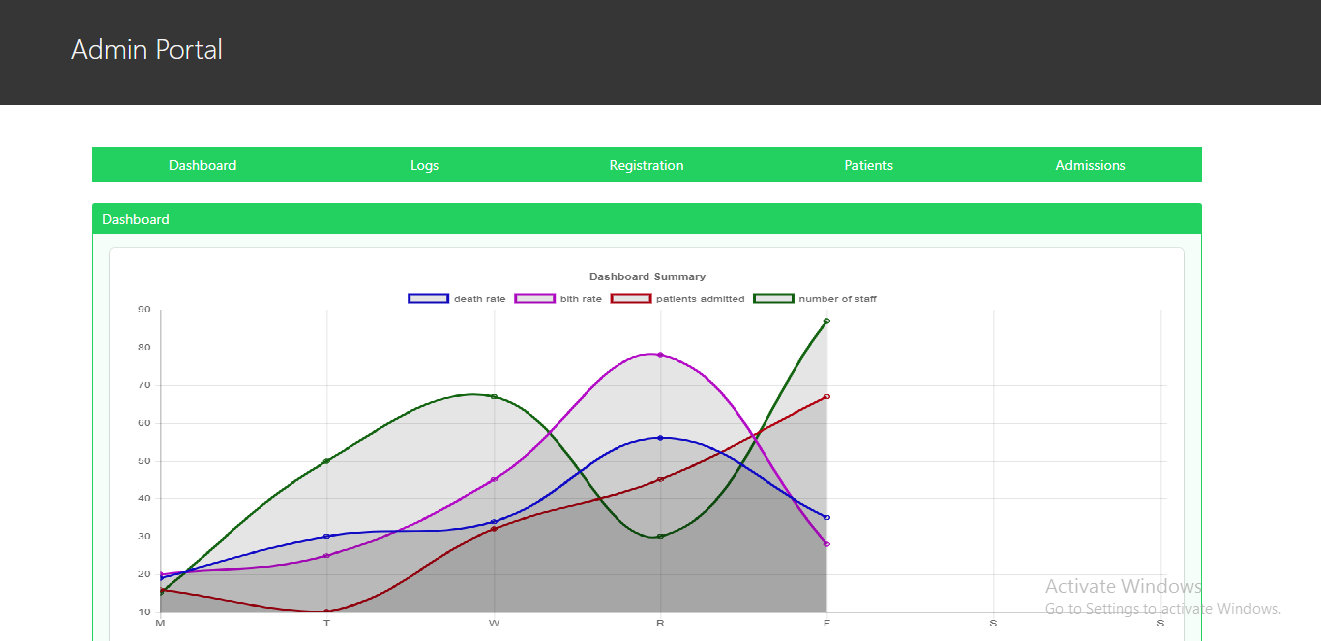


Image for homepage





Code for client router

This code snippet is responsible for making a single page application. It handles all the navigation functionalities in the application. It can be thought as the backbone of any SPA

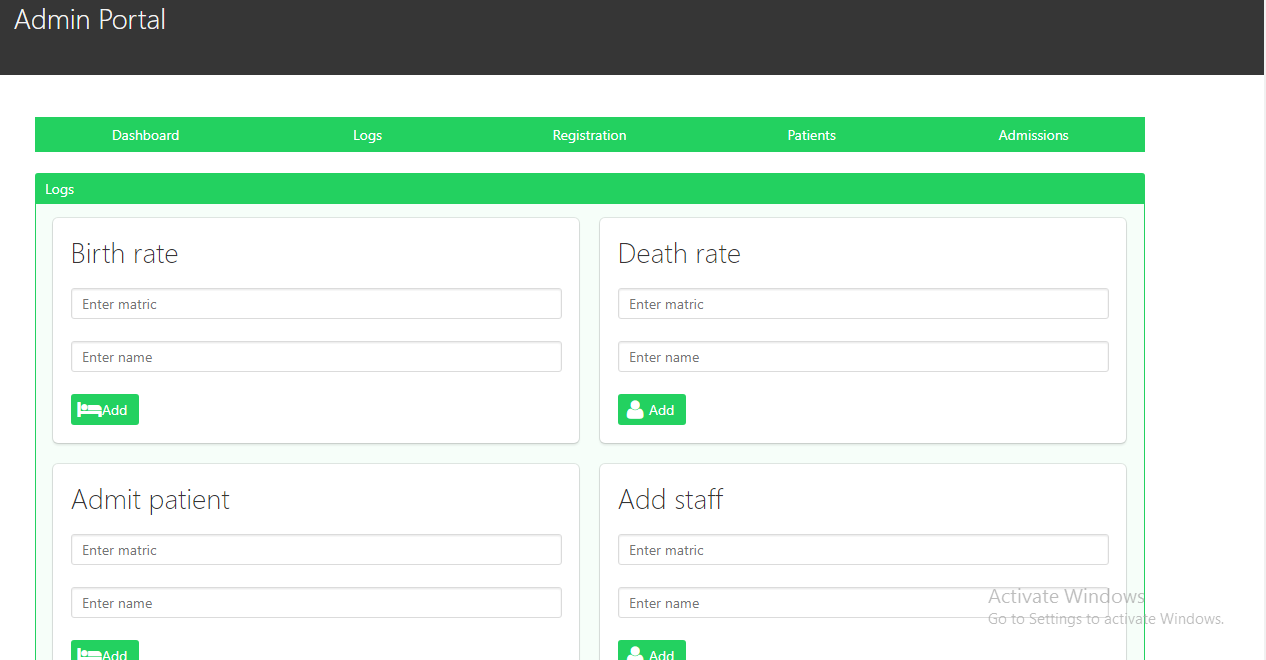


Code for config file

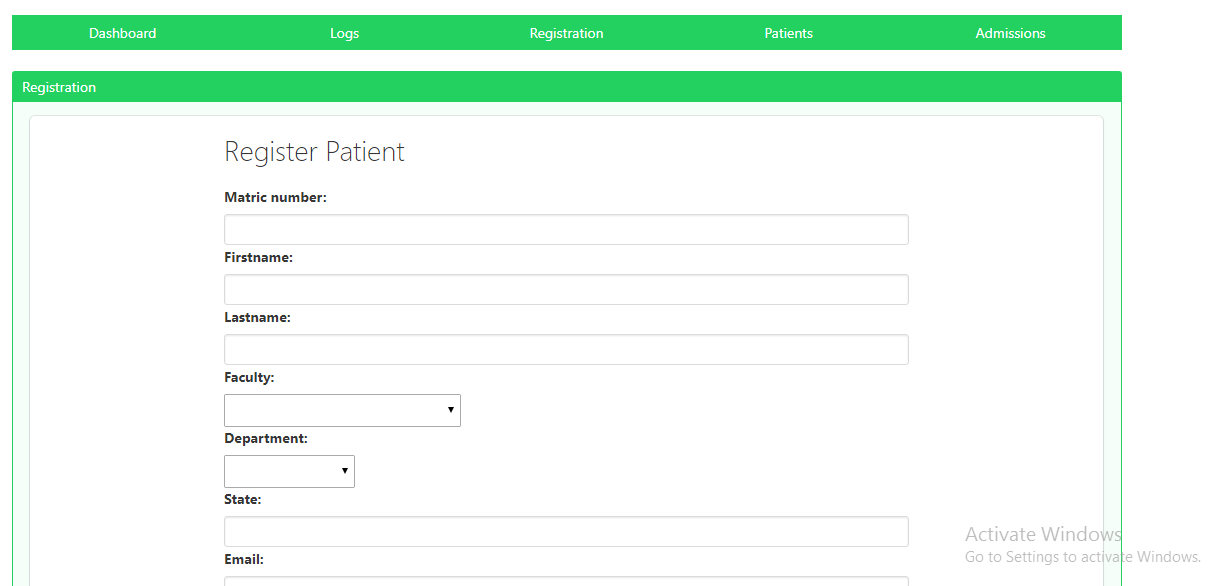
This code is for the configuration. It contains all the meta data responsible for the server config and the application start up.



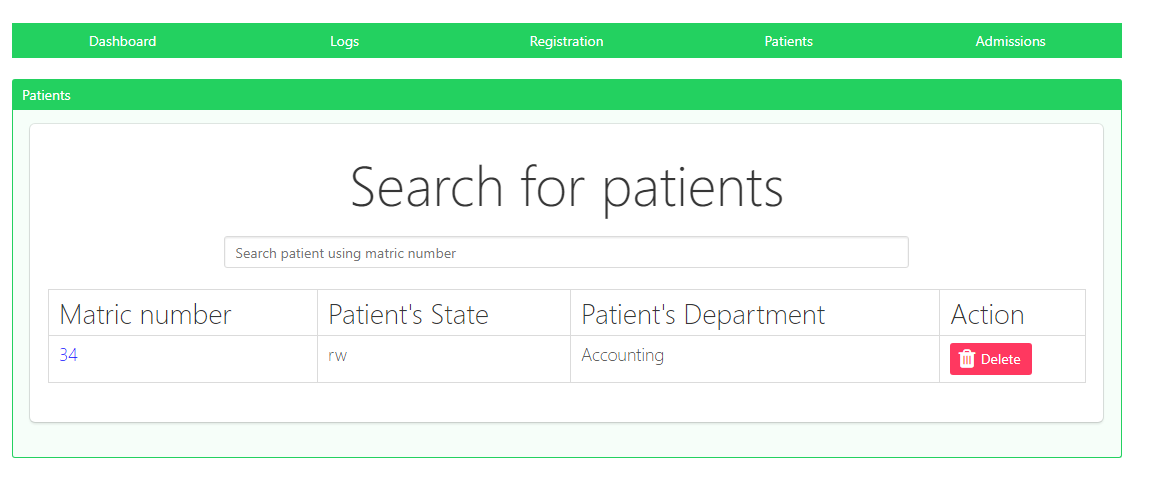
Interface for adding logs



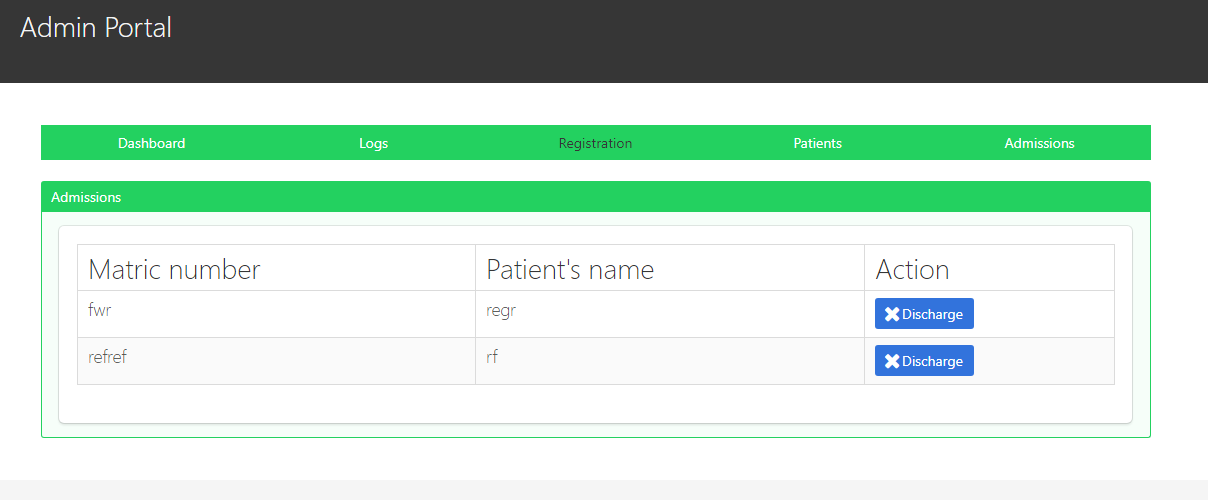
Interface for registration



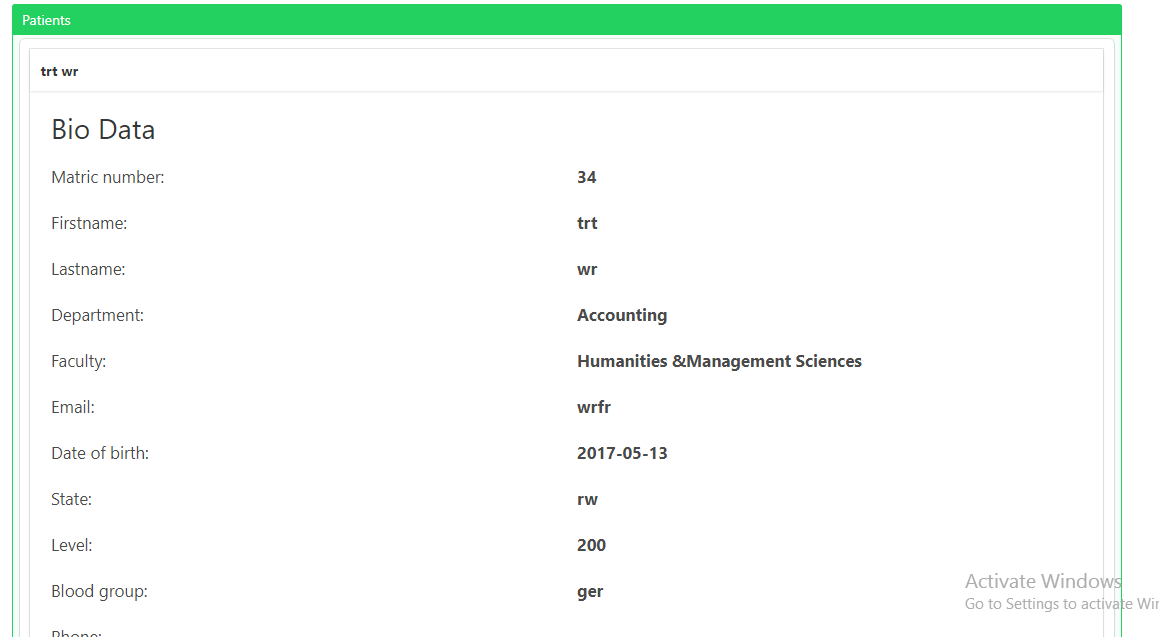
Interface for patients



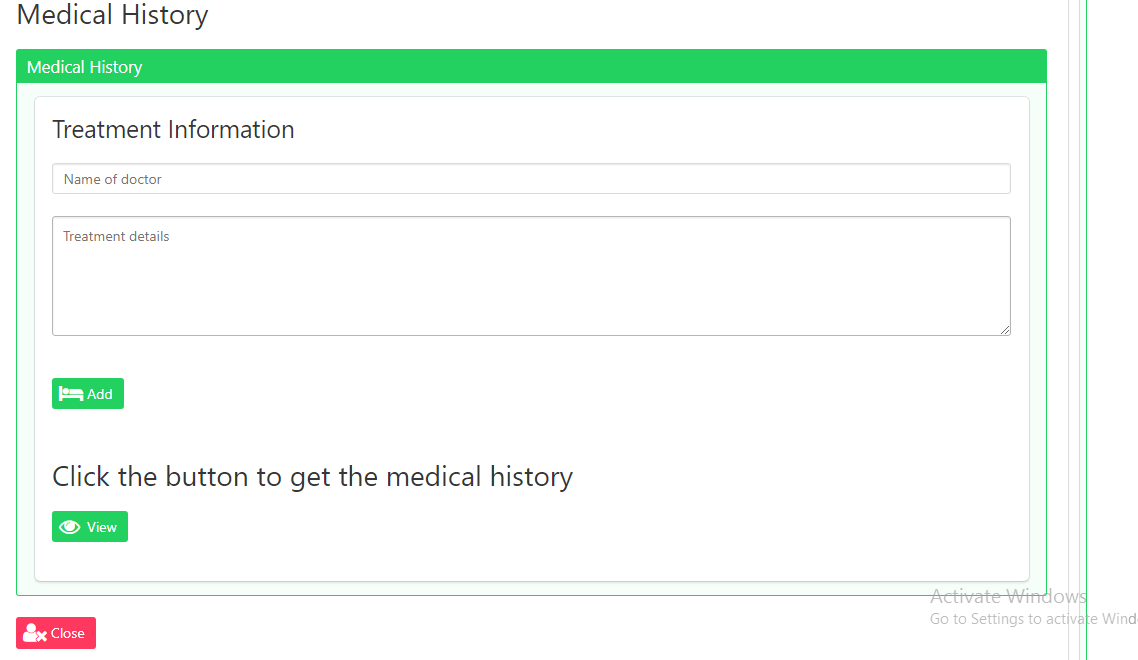
Interface for discharging a patient



Interface for patient bio data



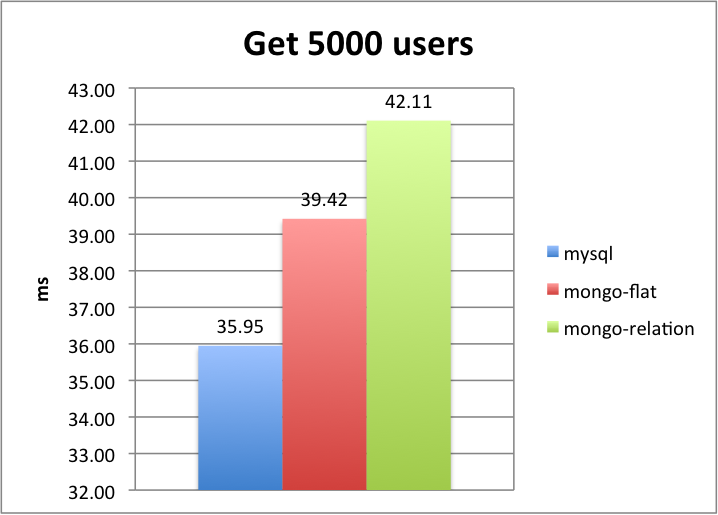
Interface for patient medical history



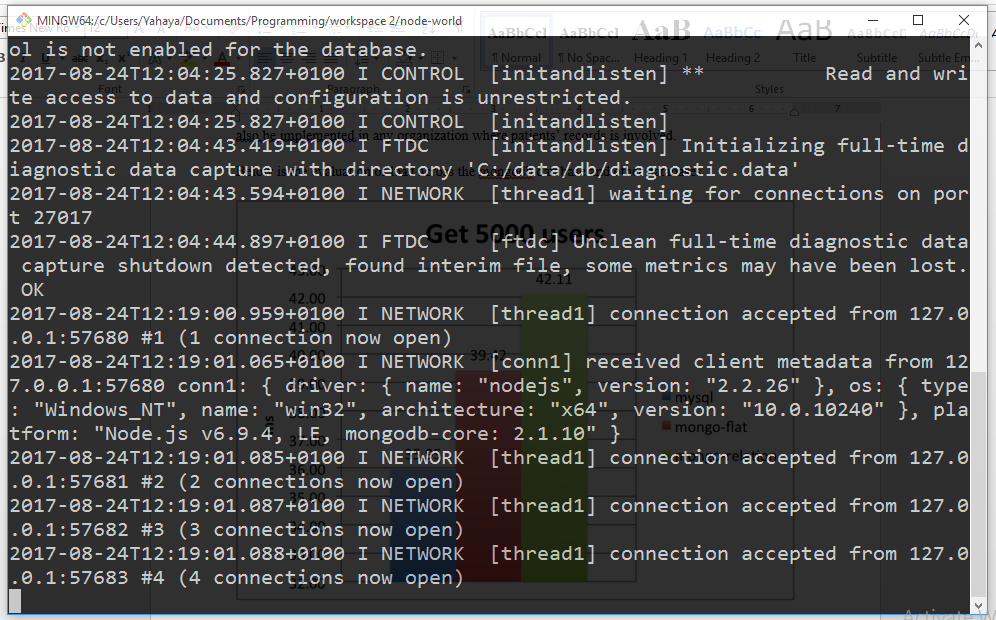
4. 2 Evaluation

Here, I evaluated my own work and then compared it with other similar stacks e.g the LAMP stack (Linux Apache MySQL PHP). The performance metrics on the Node server and the mongodb server is by far better than those ones. Below is the evaluation of the other stacks.

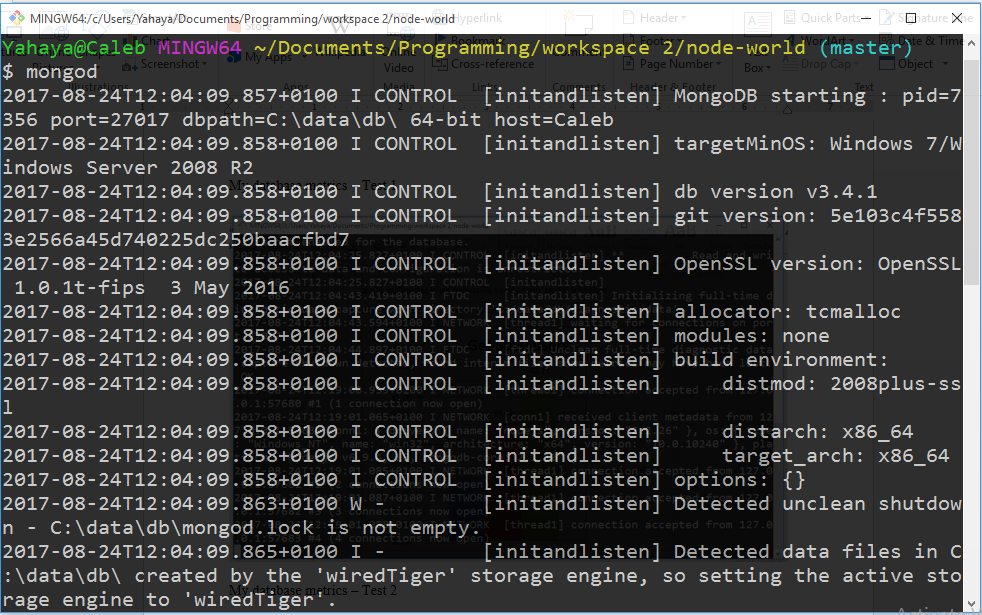
Industry Database and web server metrics



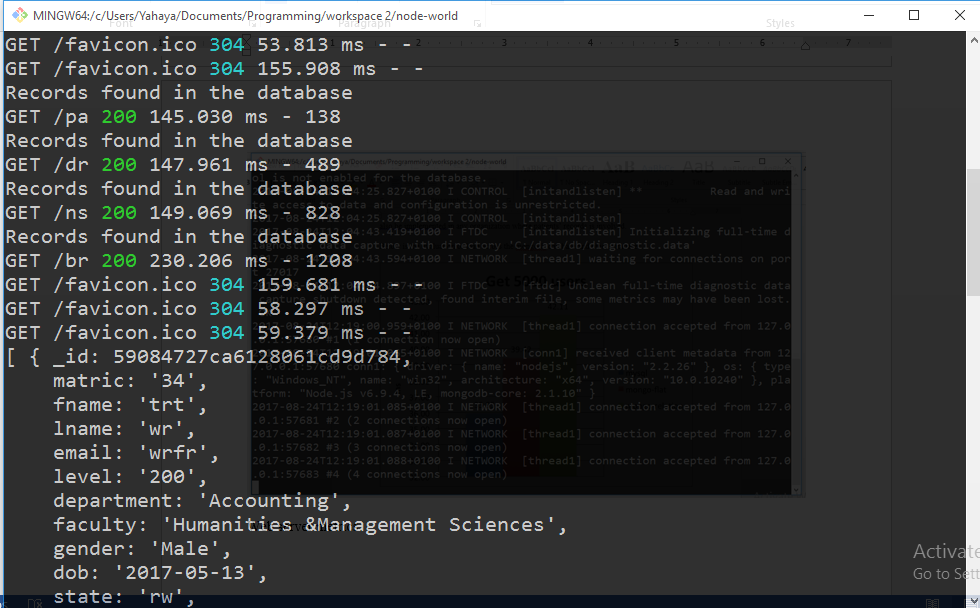
My database metrics – Test 1



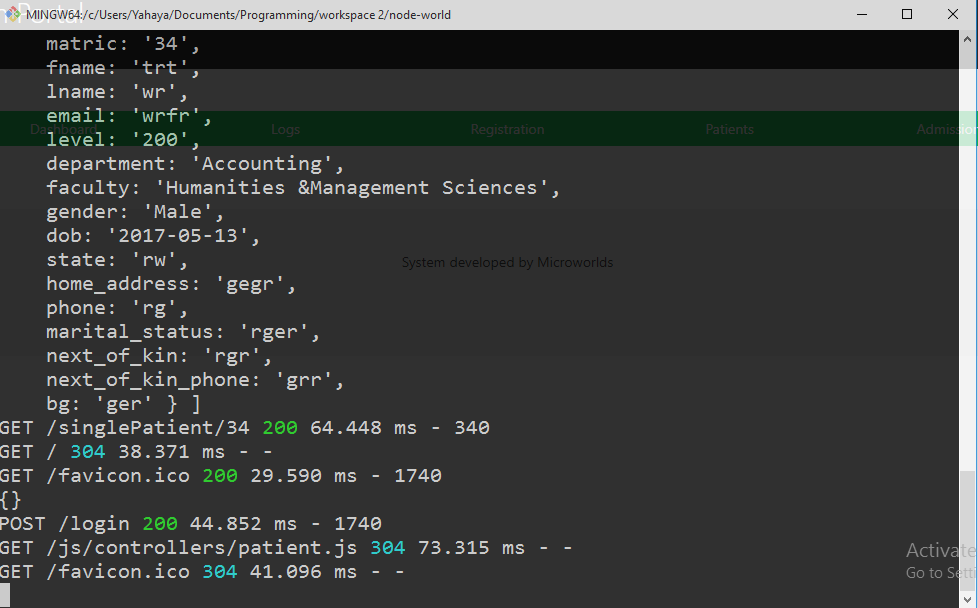
My database metrics – Test 2



My web server metrics – Test 1



My web server metrics – Test 2



It is obvious that the maximum time to serve any given resource is less than 60 millisecond and the database is still running on one thread. This is extremely fast. Although it can be improved over time.

Chapter 5 - Summary and conclusion

5.1 Summary

When it comes to the issue of scaling, this system can scale very well, in the sense that whenever there is an increase in the number of users in the application, it can accommodate such load. The system is designed to help people to reduce manual labor and help them automate the manual processes in the clinic. This will go a long way in improving our modern hospitals. It can also be implemented in any organization where patients’ records are involved.

5.2 Conclusion

After this system have been implemented, it really help automate daily manual tasks in the hospital and hence reducing the rate at which patients wait to search for their id number or clinic card numbers before being attended to. It will also provide a backup copy for all the data in the system in case of any natural disaster or hazard.

5.3 Recommendation

This system is still in its beta stage and it is also open source. I highly recommend going through the source code and improve this piece of software. Below is the repository of the project on github. Feel free to clone the software and make your own changes to it.

https://github.com/microworlds/node-world

The aspect of hospital management system is very broad; I solely focused on the patient aspect, which includes search indexing, logs recording, basic CRUD operations. My recommendation to anybody improving on this system is to focus on handling and fully automating of referring patients to wards, checking for space availability and booking an appointment with the doctor, and then each user should access his or her own profile on a remote server. When all these are completed and fully integrated into the system, then the application is going to be super functional