

# HARDWARE MAINTENANCE SYTEM

K. Bhanu Phani Venkat, K. Meghana, V. Bhaskar

Mr. M. Samba Siva Rao,

Department of Computer Science and Engineering,

Usharama College of Engineering and Technology, Telaprolu,

Andhra Pradesh, India.

## Abstract

In this project, we intended to create a centralized data of systems and its peripherals of Usharama. All this data will be stored in a database (MongoDB) along with a backup. This project consists of six shareholders namely College Heads, Head of Departments, Lab Technician, Lab Incharge, Guest (Students & Teachers) and Admin. It consists details of every laboratory, computers, printers and all hardware peripherals in Usharama. Whenever a student or a faculty member faces an issue regarding system performance, he/she can login into the HMS by selecting guest login and rise a token. This token should be saved to know the status of the issue. This issue gets stored in the database and ne notified to technician dashboard. Each technician will be assigned the work to be done automatically and after completing the work they would change the status in the dashboard; which has to be justified by the incharge. This entire process can be monitored by administrators who can also get status reports.

## Key Words

Digitalized records, Token generation, Dashboard, Lazy Loading, Status reports, System tracking.

## Introduction

Now-a-days, computers and its peripherals have become a part of our daily life automating several tasks and producing outputs time efficiently. Yet, even their face with several hardware or software problems such as ram issues, power fault, storage disk etc. It is the duty of Lab Technician to take care of such problems. In general scenario, tech support maintains a ledger containing a record of all problems reported. As it is a non-digital record it is susceptible to several issues like getting lost, paper damage etc.

Currently all hardware peripherals don't have a digitalized record making things difficult for the lab technicians to solve them whenever a problem arises. All complaints regarding the systems or its peripherals has to be manual forwarded to technician using phone call or ledger which often results in delay of solution. Even after rectifying no one knows the result except the technician. Administrates are left blind regarding the peripherals used and status reports.

The aim of this project is to have a centralized digital record of all computers and its peripherals of each lab in Usharama. This project helps to track down each problem easily and update its status.

## **Problem Statement**

All complaints regarding the systems or its peripherals has to be manual forwarded to technician using phone call or meeting him directly and reporting the issue. This process is the time consuming one which often leads to delay of the problem rectification.

Also, we cannot guarantee the safety of the ledger. As it is just a book without locks anyone can enter any misleading entries into it. Also, it can be stolen or the paper can be damaged due to water etc. Now, we have to do the same process again because we don't have any backup for such fault conditions.

## **Motivation**

One of the main motives for using this application is, it saves a lot of time in reporting and taking necessary actions. This project also helps in tracking down required tasks easily. A ledger cannot be properly secured as it is susceptible to certain issues. In case of being damaged or thief, the entire work has to be done from scratch again.

## **Proposed System**

By adapting to the new system, we can have a centralized record of all peripherals available in every lab and department. The user(Guest) who have reported the user can track the problem to get its status to any given time by using its token ID. College Administrators and Head of Departments can get the status reports of any labs at any given time stating no.of working systems, no.of non-working, systems count etc.

A login portal assured a secured data entry. Users can track down their problems using their token ID. As this project also does a period backup in

mongoDB, our data would be saved in case of any disasters.

## **Related work**

- Manual reporting

All complaints regarding the systems or its peripherals has to be manual forwarded to technician using phone call or meeting him directly and reporting the issue. Only then he will be aware of the problem and record it in ledger. Now, after being aware of the problem; the technician starts working on it. After completing the issue, he again has to make a new entry in the ledger stating its status. Here only the technician who solved the problem is been aware of its status.

- Ledger entry

This is the old fashioned way where all issues have to be entered in a ledger. According to the entries in it, technician start working on it and later updates its status as a new entry. Upon which incharge verifies it.

## **Implementation**

Whenever a student or a faculty member faces an issue regarding system performance, he/she can login into the HMS by selecting guest login and rise a token. This token should be saved to know the status of the issue. This issue gets stored in the database and be notified to technician dashboard.

Each technician will be assigned the work to be done automatically and after completing the work they would change the status in the dashboard; which has to be justified by the incharge. HMS has various status levels to easily understand the status of the problem. They are working, reported, in\_progress, awaiting, solved, need\_care.

Lab Technician can change the status to any of the above levels determining its condition.

The user who have reported the user can track the problem to get its status to any given time by using its token ID. College Administrators and Head of Departments can get the status reports of any labs at any given time stating no.of working systems, no.of non-working, systems count etc.

### Status Levels

**Working:** Indicates that a system is working perfectly without any faults.

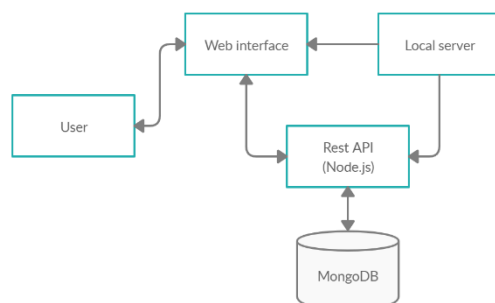
**Reported:** Indicates that a user has reported a system as not working and needs technician attention.

**In\_progress:** Indicates that a technician is working on the problem reported.

**Awaiting:** Indicates that technician has solved the issue and lab incharge must verify its condition.

**Solved:** Indicates that the issue has been rectified, user can work on it again.

**Need\_care:** This level is used to indicate that technician has encountered some serious hardware/software fault and he needs extra variables such as buying new component to solve this problem.



## Mechanisms

### Angular

Angular is a platform and framework for

building single-page client applications using HTML and TypeScript. Angular is written in TypeScript. It implements core and optional functionality as a set of TypeScript libraries that you import into your apps.

AngularJS uses the Model-View-Controller (MVC) architecture, which is used in web app development. Angular apps are modular and Angular has its own modularity system called NgModules.

NgModules are containers for a cohesive block of code dedicated to an application domain, a workflow, or a closely related set of capabilities. A component controls a patch of screen called a *view*. Angular creates, updates, and destroys components as the user moves through the application. Your app can take action at each moment in this lifecycle through optional lifecycle hooks, like `ngOnInit()`.

### Node.js

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

It is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

### Express.js

Express.js is a web application framework that provides you with a simple API to build

websites, web apps and back ends. With ExpressJS, you need not worry about low level protocols, processes, etc.

Express provides a minimal interface to build our applications. It provides us the tools that are required to build our app. It is flexible as there are numerous modules available on npm, which can be directly plugged into Express.

### **MongoDB**

MongoDB is a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in MongoDB. Collections contain sets of documents and function which is the equivalent of relational database tables. MongoDB is a database which came into light around.

With the rise in data all around the world, there has been an observable and increasing interest surrounding the wave of the non-relational database, also known as 'NoSQL'. Businesses and organizations are seeking new methods to manage the flood of data and are drawn toward the alternate database management tools and systems that are different from the traditional relational database systems. Here comes MongoDB into the picture. The data model available within MongoDB allows you to represent hierarchical relationships, to store arrays, and other more complex structures more easily.

### **Guest Module:**

Guest can be anyone whether a student or a faculty member. When a user encounters an issue regarding system performance, he/she

can login into the HMS by selecting guest login and rise a token. User will be prompted with a token ID which can be used to track his issue and check its status.

### **Technician Module:**

Every lab has a technician assigned to rectify computer components issues. When a user reported a new problem, it raises a token and will be added to technician dashboard. Technician can next the reported issues and start working on them. After solving the user, he can update its status to awaiting or need\_care.

### **Incharge Module:**

Incharge is responsible for taking care of each and every system in his lab. After technician completes his work, this token will be notified to incharge dashboard. Incharge now check the system and again update its status to reported if not working or solved is the task is completed.

### **Head of Department Module:**

Head of Department can raise a token by selecting any non working system in his department and allocate the task to any technician in the college. He can also get the status reports of all systems in his department.

### **College Administrators Module:**

Chairman, Director, Principal comes under this category. They can get the overall status reports of all systems, labs, departments in the college.

### **Admin Module:**

Admin is the person responsible to care of the proper execution of web application. He can either create, update and delete any lab, department, members in the college. His main responsible is to create labs and assign

systems and lab incharge and lab technician to it. He has write permission to the database. He can also update status of lost issues.

## FEASIBILITY ANALYSIS

An important outcome of preliminary investigation is the determination that the system request is feasible. This is possible only if it is feasible within limited resource and time. The different feasibilities that have to be analysed are

## OPERATIONAL FEASIBILITY

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally eliminates the wastage of resources and effectively tracking the issues status. This kind of automation will surely reduce the time and energy, which previously consumed in manual work. Based on the study, the system is proved to be operationally feasible.

## ECONOMIC FEASIBILITY

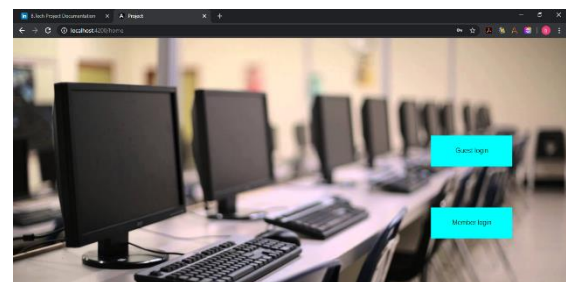
Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer-based project. As hardware is a pre-existing component, thus the cost on project of hardware is low. Since the system is an angular based, so the project supports lazy loading mechanism which can be stated as economically feasible.

## TECHNICAL FEASIBILITY

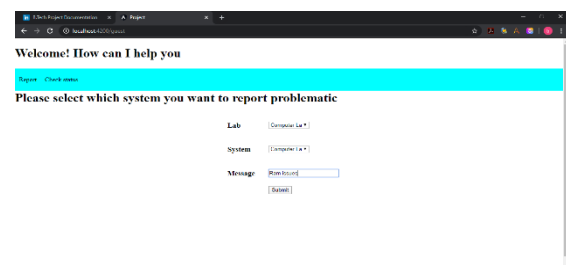
Technical Feasibility is the assessment of the technical resources of the organization. The organization needs a local server fixed in the laboratory with mongoDB installed in it. The technical feasibility has been carried out. The system is technically feasible for development and can be developed with the existing facility.

## Application

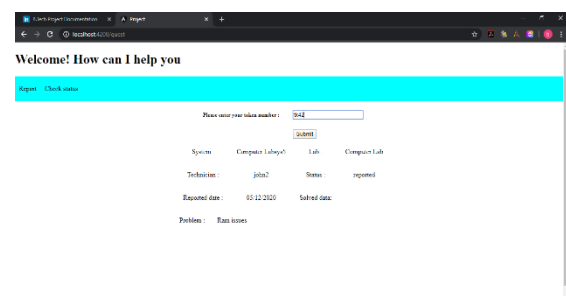
### Home page



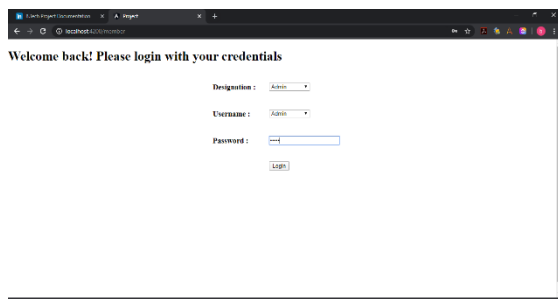
### Report page



### Status page



## Login page



Welcome back! Please login with your credentials

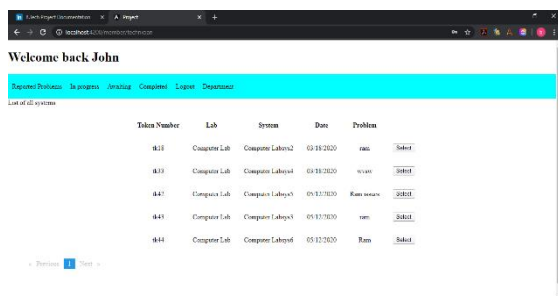
Designation : Admin

Username : Admin

Password : [input field]

Login

## Dashboard page



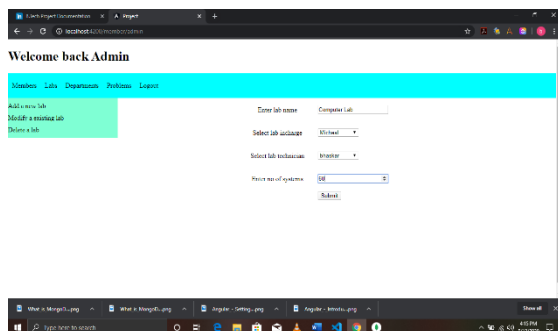
Welcome back John

Dashboard | Devices | In progress | Settings | Computers | Logout | Departments

List of all systems

Taken Number	Lab	System	Date	Problem
012	Computer Lab	Computer Lab001	03-10-2020	ram
013	Computer Lab	Computer Lab004	03-10-2020	screen
043	Computer Lab	Computer Lab003	05-12-2020	Ram issue
041	Computer Lab	Computer Lab005	05-12-2020	ram
044	Computer Lab	Computer Lab001	05-12-2020	Ram

## Admin page



Welcome back Admin

Dashboard | Labs | Departments | Problems | Logout

Add a new lab  
Modify a existing lab  
Delete a lab

Enter lab name: Computer Lab

Select lab category: Micro

Select lab technician: [input field]

Price per system: 100

Submit

## TESTING

Testing is that the method of attempting to get each conceivable fault or weakness in an exceedingly work product

## ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements. Test Results: All the test

cases mentioned above passed successfully. No defects encountered.

## TESTING STRATEGY

A strategy for system testing integrates system test cases and design techniques into a well-planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements. Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

## CONCLUSION

By adapting to this project, we would be able to have a centralized record of all peripherals available in every lab and department. The issue which are reported can be stored in the database and be used to effectively measure and monitor the future investments in computer components.

All the systems will be properly utilized and accounted for. It would be so easy to track down any system issues or the management can also easily measure the performance of the lab technicians and incharges. In case of any damage to the record can also be revived due to periodic backup of collections.

## FUTURE SCOPE

In the near future, we would like to implement speech recognition and facial unlocking features as an optional feature to the product. These implementations would be very helpful to the user for providing easily navigation of the application and also hands-free navigation.

In addition to these features we would also like to make it completely automated software which would auto detect any problems related to the computer and its peripherals such as keyboard, mouse etc. Basic concept of this idea is to build a windows service file which would be running in the background of the computer and be triggered by hardware related events. Then the service would be periodically checking any error codes in the machine. If any error code is encountered then it would send response to the central server using rest api.

In this way, we can auto detect the issues without any one checking and passing over to the technicians. After solving the issue, technicians can also check the compatibility themselves using this service without any third party justification.

## REFERENCES

- [1] "[Angular Docs](#)". angular.io.
- [2] "[What's the difference between AngularJS and Angular?](#)". gorrior.io. September 19, 2017. Retrieved 2018-01-28.
- [3] "[Angular: Branding Guidelines for AngularJS](#)". Retrieved 2017-03-04.
- [4] Coman Hamilton. "[A sneak peek at the radically new Angular 2.0](#)". Retrieved 2015-10-21.
- [5] "[Ok... let me explain: it's going to be Angular 4.0](#)". angularjs.blogspot.kr. Retrieved 2016-12-14.
- [6] "[Angular 4.0.0 Now Available](#)". angularjs.blogspot.ca. Retrieved 2017-03-23.
- [7] "[Angular 4 coming in 2017, to be backwards compatible with Angular 2](#)". react-etc.net. Retrieved 2016-12-14.
- [8] Fluin, Stephen. "[Version 5.0.0 of Angular Now Available](#)". Retrieved 2 November 2017.
- [9] "[Angular 5 JavaScript framework delayed](#)".
- [10] "[Version 6.0.0 of Angular Now Available](#)". Retrieved 4 May 2018.
- [11] Fluin, Stephen (2018-10-18). "[Version 7 of Angular — CLI Prompts, Virtual Scroll, Drag and Drop and more](#)". Angular Blog. Retrieved 2019-06-07.
- [12] "[node-v0.x-archive on GitHub](#)". Retrieved 2 August 2014.
- [13] "[Node.js 14 ChangeLog](#)". Retrieved 6 May 2020 – via [GitHub](#).
- [14] "[The MIT License](#)". Open Source Initiative. 17 September 2018. Retrieved 17 September 2018.
- [15] "[Express4.x changelog](#)". expressjs.com
- [16] "[https://github.com/expressjs/express/releases/latest](#)"
- [17] "[Express.js home page](#)".
- [18] "[Mean.io: The Friendly & Fun Javascript Fullstack for your next web application](#)". Archived from the original on 13 June 2019. Retrieved 15 July 2019.