**CHAPTER 1**

**INTRODUCTION**

**DMW CNC SOLUTION** is a diesel machineries production company which is located in Perundurai, Erode road.

**They are specialized in machining of the following critical components**

* Cylinder Blocks
* Cylinder Heads
* Gear box Housings
* Axle Housings & Wheel Supports
* Air Compressor Crank Cases
* Air Compressor Cylinders
* Air Compressor & Engine Manifolds
* Control Panel Manifolds
* Braking Cylinders
* Brake box, Piston, Brake Flange
* Adaptor Plates
* Power Steering Pump Housings
* HT Transmission Components

The company uses excel sheet for entering all the production details and defect details. The employee attendance details are managed in paper records. There are in need of an application which is portable , to make data management and data retrieval easier.

We have suggested an idea to the company to meet their requirements by developing an android application. They agreed with us to develop this android application.

The following modules, clearly explains how **ANDROID APP FOR DMW CNC SOLUTIONS** developed to meet the various requirements of the company

**1.1 EXISTING SYSTEM**

In existing system,

**1.2 OBJECTIVE**

Medex Medical centre is a process of implementing all the activities of the hospital in a computerized automated way to fasten the performance. The main purpose of our system is to make hospital task easy and is to develop software that replaces the manual hospital system into automated hospital management system.

**1.4 SCOPE**

The proposed system consists of three major modules i.e.

* Patient
* Doctor
* Admin

1. The Scope of Patient module is to

* **Login:** Patient login
* **Register:** New Patient registration
* **Patient**: This will capture information about Patient register as well as Patient details
* **Book a Doctor’s Appointment:** Patient can book an appointment with the Doctor for consultation along with date and time preferred, and also we have option to select the specialty to consult

1. The Scope of Doctor module is to

* **Check Appointments:** Doctor can check the appointment scheduled
* **Add Diagnosis Info:** Doctor can add diagnosis details in form
* **View All Patient Info:** Doctor can view the past appointment history details

1. The Scope of the **Medex Admin** module is to

* **Reports:** The Admin will have the options to generate the vise kind of reports giving summary details of patients & doctors

**CHAPTER 2**

**GENERAL** **DESCRIPTION**

**2.1 PROJECT PERSPECTIVE**

The main perspective of the project is to reduce the paper work, data retrieval through automation. Our Project will replace all traditional and outdated means of tracking patient information and other data useful to the hospital. This system shall replace forms of databases using manual or outdated hardcopy databases. Accessing data can be better monitored, organized, and time conscientious.

**2.2 USER CHARACTERISTICS**

In this application, The admin can register the doctor details, the patient can register and login then they can book their appointment, view their appointment and cancel their appointment. The doctor can login and check their appointments, view patient details.

**2.3 DESIGN AND IMPLEMENTATION CONSTRAINTS**

The application is designed for DMW CNC solution to make the data entry and manipulation process simple, this application is portable and then the relation should satisfy the key constraints, domains, referential constraints and integrity constraints.

**2.4 PROPOSED SYSTEM**

The proposed system is a web application, which is completely related to automation. It consists of 3 modules such as Patient, Doctor and Admin module. The patient can register and login into the system, then they can book their appointment. The doctor can view their appointment and view the details about the patient. The admin only have the rights to register doctor details in database and they can generate report such as no of patient visited to the hospital and no of doctors who are working on the hospital.

**ADVANTAGES**

* Saves users time.
* portable
* Security - User id is provided with password or access only to the kind information.
* Data integrity

**CHAPTER 3**

**REQUIREMENTS SPECIFICATIONS**

**3.1 FUNCTIONAL REQUIREMENTS**

When the users enter to following link **http://localhost:8080/MedexMedicalCenter** in the browser, Home page appears which will have a short welcome message; a brief introduction to the purpose of the C-Medex, the user can choose appropriate options. Medex Medical Centers super-specialty services where specialist care is effectively administered under one roof. At Medex Medical Centers Hospital, there is a continuous drive for quality in order to meet patients’ standards through quality healthcare delivery.

**3.1.1 HOME PAGE INTERFACE REQUIREMENTS**

1. Home
2. Patient
3. Doctor
4. Medex Admin
5. About Us
6. Contact Us

**3.1.2 MODULE DESCRIPTION**

**3.1.2.1 PATIENT**

### 3.1.2.1.2 PATIENT - REGISTRATION PAGE

* When the patient/user clicked on the registration for free link, it should re-direct to registration for free form and the users/ patient need to fill some of the basic attributes/fields as mentioned below in requirement;
* Login Id, Password, First Name, Last Name, Age, Gender (Male or Female) ,Contact Number
* E-mail, Address, Zip code, City

**3.1.2.1.2 PATIENT - LOGIN PAGE**

When the users enter to following link **http://localhost:8080/MedexMedicalCenter** in the browser, the patient tab will provided the features like, Login id and Register for free.

Below are the fields:

* Login id
* Password

**3.1.2.1.3** **PATIENT - LOGGED IN PAGE**

When the patient logged in, they should able to view the following options in the page.

They are:

* Book Appointment
* View Appointment
* Cancel Appointment

**3.1.2.2 DOCTOR**

* In the doctor logged in page we have to give the upcoming weekend seminars details.
* The doctor can also check the appointment by choose the date pattern and should able to view patients details like, patients name, age, gender, address, appointment booked date etc.
* The doctor can update the diagnosis detail of the appropriate patient in the application.
* The doctor can also have a feature to view all the patient details by selecting the name.

**3.1.2.3 ADMIN**

* Admin only register the doctor and give their login id.
* The Medex admin should have a feature to view and to generate reports, like No of doctor in the hospital, No of patient visited to hospital and other varies kind of reports should be able to generated by Medex Admin.

**3.2 NON-FUNCTIONAL REQUIREMENTS**

**3.2.1 PERFORMANCE REQUIREMENTS**

* All front-end pages and reports should be served up in less than 5 seconds post click when up to 1000 users are on the application concurrently

**3.2.2 OPERATIONAL REQUIREMENTS**

* Application Java and J2EE for deployment with Mysql database for persistence
* This system must be up 24X7

**3.2.3 MAINTAINABILITY & PORTABILITY REQUIREMENTS**

* If the allocated memory space for the existing website needs to be increased, then it should be possible without any impact to performance

**3.2.4 SECURITY REQUIREEMENTS**

* No unauthorized users should be able to log on to the system
* User Id/Login Id should have combination of letter and numeric values, none

**3.3 USER INTERFACE**

**3.3.1 USER INTERFACES**

* Front-end software: Browser
* Scripting Languages: Front-end: JSP, HTML, Bootstrap
* Technologies: Servlet
* Back-end software: MySQL

**3.3.2 HARDWARE INTERFACES**

* Processor : Intel(R) Core(TM)i5
* RAM : 4 GB
* Memory : 1 TB

**3.3.3 SOFTWARE INTERFACES**

Following are the software used for the student information management online application.

|  |  |
| --- | --- |
| SOFTWARE USED | DESCRIPTION |
| Operating system | |  | | --- | | Windows operating system is chosen | | for its best support and user-friendliness. | |
| Server | Application needs Apache Tomcat server for deployment with MySQL 5.5 database for persistence |
| Database | |  | | --- | | SQLs database is chosen to save the customer | | records. | |

**3.3.4 SOFTWARE AND PROGRAMS USED**

A number of software and programs have been used in order to develop the sample online booking system. The types of programs and software used were programming languages such as Java, frameworks such as Servlets, development platform such as Eclipse, script language such as JavaScript and bootstrap, markup language such as HTML and database program such as SQLite, among others. Proper installation of all the programs used were required and basic understanding of the programs were utilized in the completion of the project.

**3.3.4.1 JAVA**

Java is a set of technologies (programming language and computing platform) for creating and running software. Java is a popular general-purpose programming language and computing platform. It is fast, reliable, and secure. According to Oracle, the company that owns Java, Java runs on 3 billion devices worldwide. Java was built with the philosophy of "write once, run anywhere" (WORA). The Java code (pure Java code and libraries) you write on one platform (operating system) will run on other platforms with no modification. To run Java, an abstract machine called Java Virtual Machine (JVM) is used. The JVM executes the Java bytecode. Then, the CPU executes the JVM. Since all JVMs works exactly the same, the same code works on other operating systems as well, making Java platform-independent. There are different styles of programming. Object-oriented approach is one of the popular programming styles. In object-oriented programming, a complex problem is divided into smaller sets by creating objects. This makes your code reusable, has design benefits, and makes code easier to maintain.

The earlier versions of Java were criticized for being slow. However, things are completely different now. The new JVMs are significantly faster. And, the CPU that executes JVM are also getting more and more powerful. The Java platform provides various features for security of Java applications. Some of the high-level features that Java handles are:  
  
- provides secure platform for developing and running applications  
- automatic memory management, reduces memory corruption and vulnerabilities  
- provides secure communication by protecting the integrity and privacy of data transmitted

**3.3.4.2 SERVLETS**

Today we all are aware of the need of creating dynamic web pages i.e the ones which have the capability to change the site contents according to the time or are able to generate the contents according to the request received by the client. If you like coding in Java, then you will be happy to know that using Java there also exists a way to generate dynamic web pages and that way is Java Servlet. But before we move forward with our topic let’s first understand the need for server-side extensions.

Servlets are the Java programs that runs on the Java-enabled web server or application server. They are used to handle the request obtained from the web server, process the request, produce the response, then send response back to the web server.

Servlets work on the server-side. Servlets capable of handling complex request obtained from web server.

1. The clients send the request to the web server.
2. The web server receives the request.
3. The web server passes the request to the corresponding servlet.
4. The servlet processes the request and generate the response in the form of output.
5. The servlet send the response back to the web server.
6. The web server sends the response back to the client and the client browser displays it on the screen.

**3.3.4.3 HYPERTEXT MARKUP LANGUAGE**

Hyper Text Markup Language (HTML) is a popular markup language used in web pages. HTML can be simply written in a text editor and tested through web browser. Writing in html is easy; with html it’s also possible to add media and images to the web page. HTML contains special markup tags like <title>, <h>, <p> etc. To declare the title of the page for example, the title has to be included in the title tags. Similarly, the paragraphs, headings and other different contents in website pages have to be included inside the respective HTML tags. It is easy to store HTML code; in a simple text file with filename followed by .html or .htm extension. HTML gives developers the possibility of creating sections in the document. As <title> tag gives the title for the web page, <H1> for example specifies the main content of the web page. Similarly, with H2, H3 and so on tags, HTML creates minor contents. There are tags for other features such as paragraphs (<p>), style of font (<b>bold</b>) and tables (<table>) etc.

**3.3.4.4 JAVASCRIPT**

JavaScript is one of the most popular script languages for webpages today. JavaScript (JS) is an object-oriented language that supports features such as imperative and functional programming. The syntax is similar to other object-oriented languages such as Java and C++, hence making it easy to learn for people who know these languages. JS is used to add interactive features such as buttons, animation, games etc. JS was invented by Brendon Eich. JavaScript is capable of creating many different features ranging from beginner to advanced such as 2D and 3D features on a website.

**3.3.4.5 BOOTSTRAP**

Build responsive, mobile-first projects on the web with the world’s most popular front-end component library. Bootstrap is an open source toolkit for developing with HTML, CSS, and JS. Quickly prototype your ideas or build your entire app with our Sass variables and mixins, responsive grid system, extensive prebuilt components, and powerful plugins built on jQuery. Bootstrap employs a handful of important global styles and settings that you’ll need to be aware of when using it, all of which are almost exclusively geared towards the normalization of cross browser styles.

**3.3.4.6 CASCADING STYLE SHEET (CSS)**

CSS is a language that was developed in 1992-1993. CSS helps design each and every element of the mark-up elements language such as HTML by giving complete control to the designer. While the HTML elements enables the web page designers to add what content they want, CSS makes it possible how to display the content to the user. CSS covers the areas such as colours, layout, advanced positions of elements, fonts and also allows the content to adapt the content to different devices such as phones, tablets, bigger screens and printers. CSS can operate independently as well as be used with any mark-up languages based in XML. CSS uses simple, everyday English words and has an easy syntax. CSS is crucial in advanced web designing as it gives control to the layout and offers numerous techniques to make the web page look sophisticated. Currently, the basic features of CSS are supported by all main browsers such as Internet Explorer, Safari, 10Opera, Chrome and Firefox. CSS has been used in the web pages for the development of sample online booking system for this thesis for positioning, layout, margins and colours for the HTML elements.

**3.3.4.7 SQLITE DATABASE**

Databases are collections of similar data. Databases are used for organized collection and storing of similar data, to be later used for specific purposes. A database contains tables with rows and columns populated with objects, which displays connection between them. A database acts as a shared resource for the programs which can use the information from the database. Many enterprises rely on databases today to store a wide range of information systematically. Databases are used almost everywhere: in small companies which can use database to save customers' information and also for more advanced scientific and military areas. Databases facilitate the task of searching hundreds and thousands of records much simpler by storing them in an organized manner. Among many database programs available today, SQLite Database has been used in the development work.

**CHAPTER 4**

**DETAILED DESIGN**

**4.1 ARCHITECTURAL DESIGN**

**login**

**4.2 DATA FLOW DIAGRAM**

ADMIN

PARTNER

MANAGER

INTIMATION

PROFIT AND LOSS

DATA

**4.3 DATABASE DESIGN**

Cloud Firestore is a NoSQL, document-oriented database. Unlike a SQL database, there are no tables or rows. Instead, you store data in documents, which are organized into collection.Each document contains a set of key-value pairs. Cloud Firestore is optimized for storing large collections of small documents.  
All documents must be stored in collections. Documents can contain sub collections and nested objects, both of which can include primitive fields like strings or complex objects like lists.Collections and documents are created implicitly in Cloud Firestore. Simply assign data to a document within a collection. If either the collection or document does not exist, Cloud Firestore creates it

Firebase provides a realtime database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud.[2 The company provides client libraries that enable integration with android, ios, Java and Node.js applications. The database is also accessible through a REST API and bindings for several  javascript such as  React, Ember.js and Backbone.js .The rest Api uses the Server-Sent Events protocol, which is an API for creating HTTP connections for receiving push notifications from a server. Developers using the realtime database can secure their data by using the company's server-side-enforced security rules. Cloud Firestore which is Firebase's next generation of the Realtime Database was released for beta use.

**4.3.1 ER Diagram**

An ER diagram is a diagram that helps to design database in efficient way. Attributes in ER diagrams are usually modeled as an oval with the name of the attribute, linked to entity or relationship that contains the attribute.

Within the relation model the final step can generally be broken down into two further steps, that of determining the grouping of information within the system, generally determining what are the basic objects about which information is being stored, and then determining the relationships between these groups of information, or objects.

An Entity Relationship Diagram (ERD) is a visual representation of different data using conventions that describe how these data are related to each other. ER diagrams are most often associated with complex databases that are used in software engineering and IT networks. In particular, ER diagrams are frequently used during the design stage of a development process in order to identify different system elements and their relationships with each other. For example, inventory software used in a retail shop will have a database that monitors elements such as purchases, item, item type, item source and item price.

**4.3.3 Database Manipulation**

Data Manipulation is retrieval of information from the database, insertion of new information into the database, deletion of information in the database modification of information in the database. The goal is to provide efficient human interaction with the system.

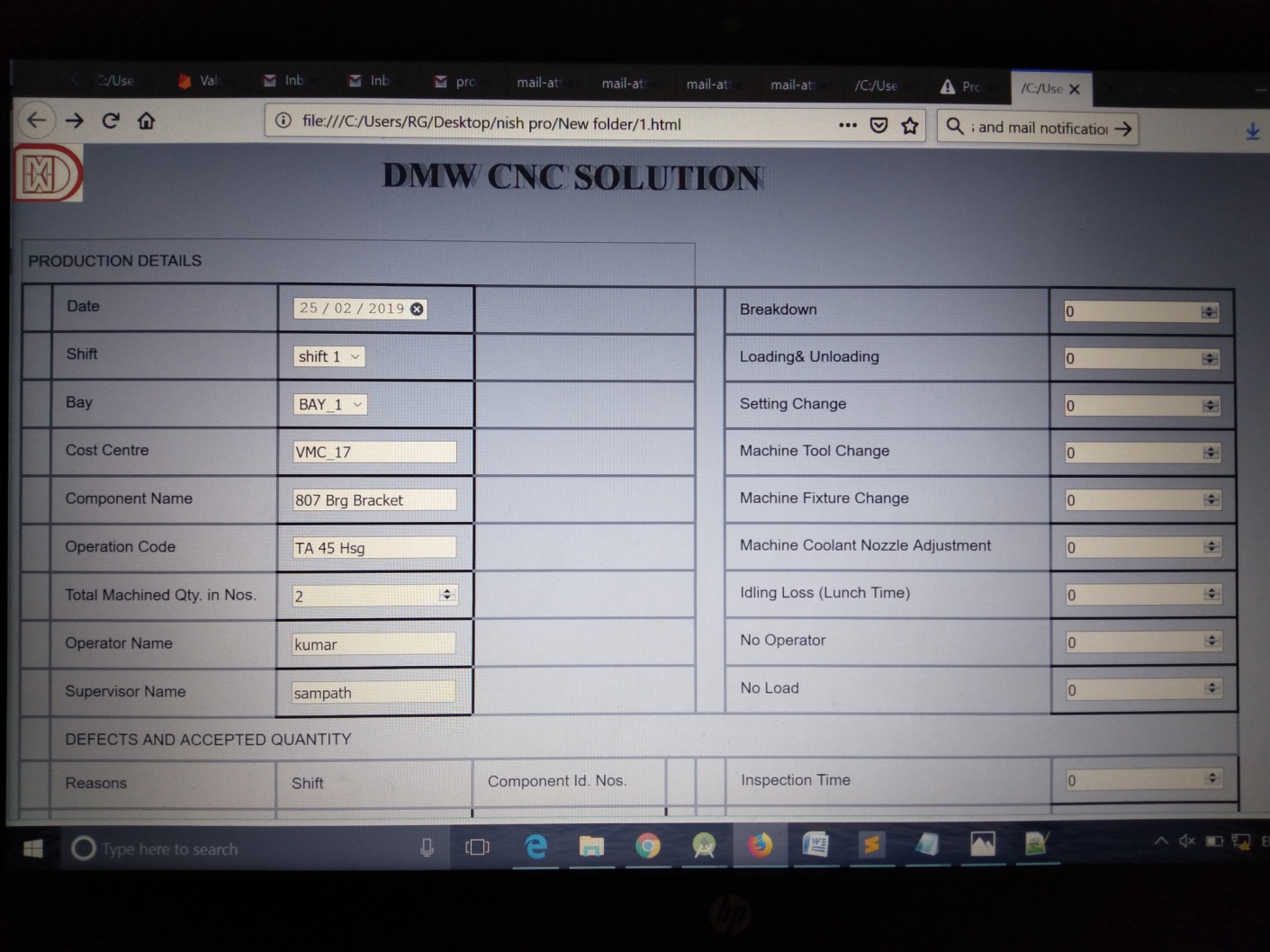
There are two types :

.

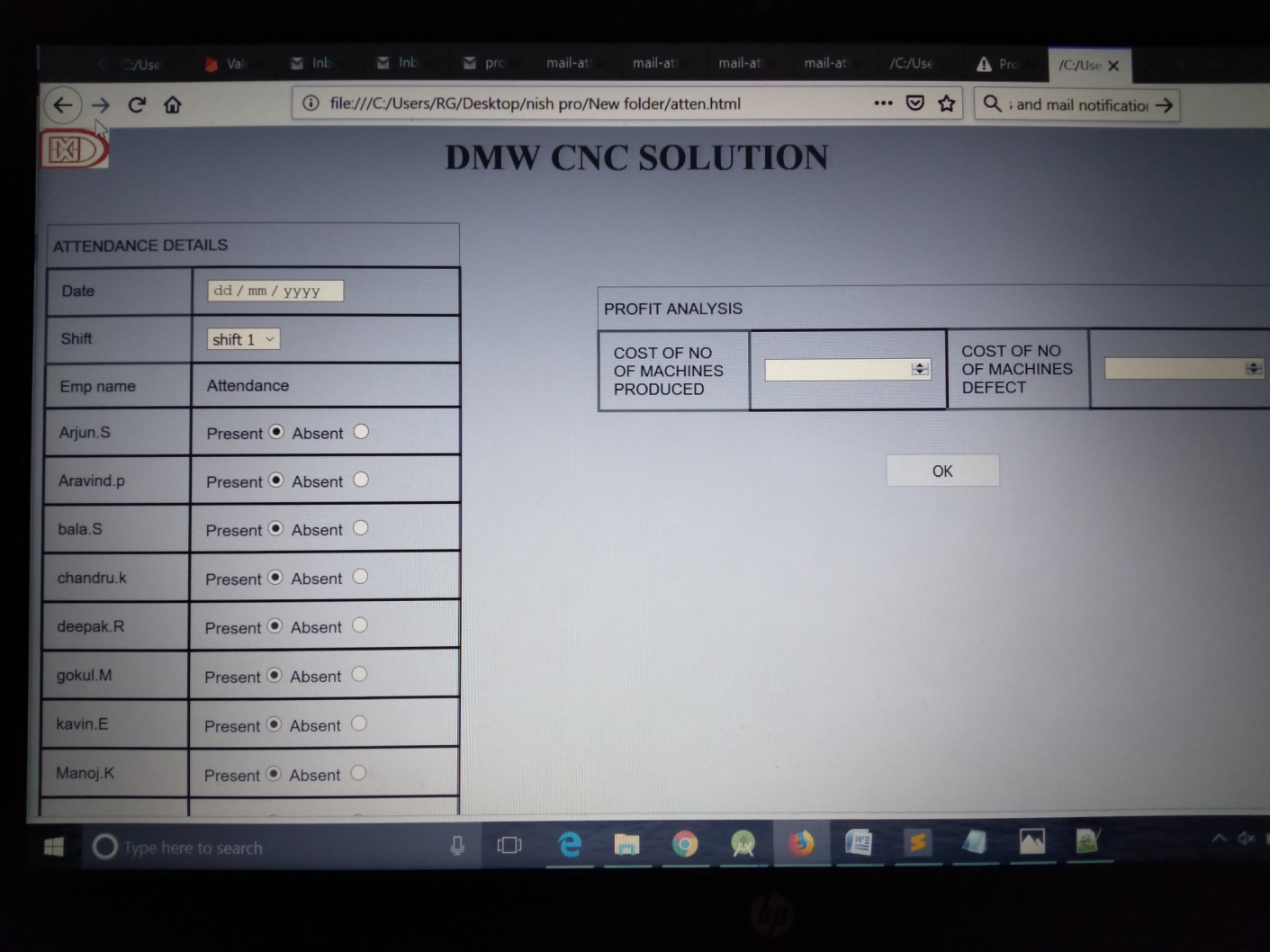
1. **Realtime Database** : Firebase's original database. It's an efficient, low-latency solution for mobile apps that require synced states across clients in realtime.
2. **Cloud Firestore** : Firebase's new flagship database for mobile app development. It improves on the successes of the Realtime Database with a new, more intuitive data model. Cloud Firestore also features richer, faster queries and scales better than the Realtime Database.

**4.4 OUTPUT DESIGN**

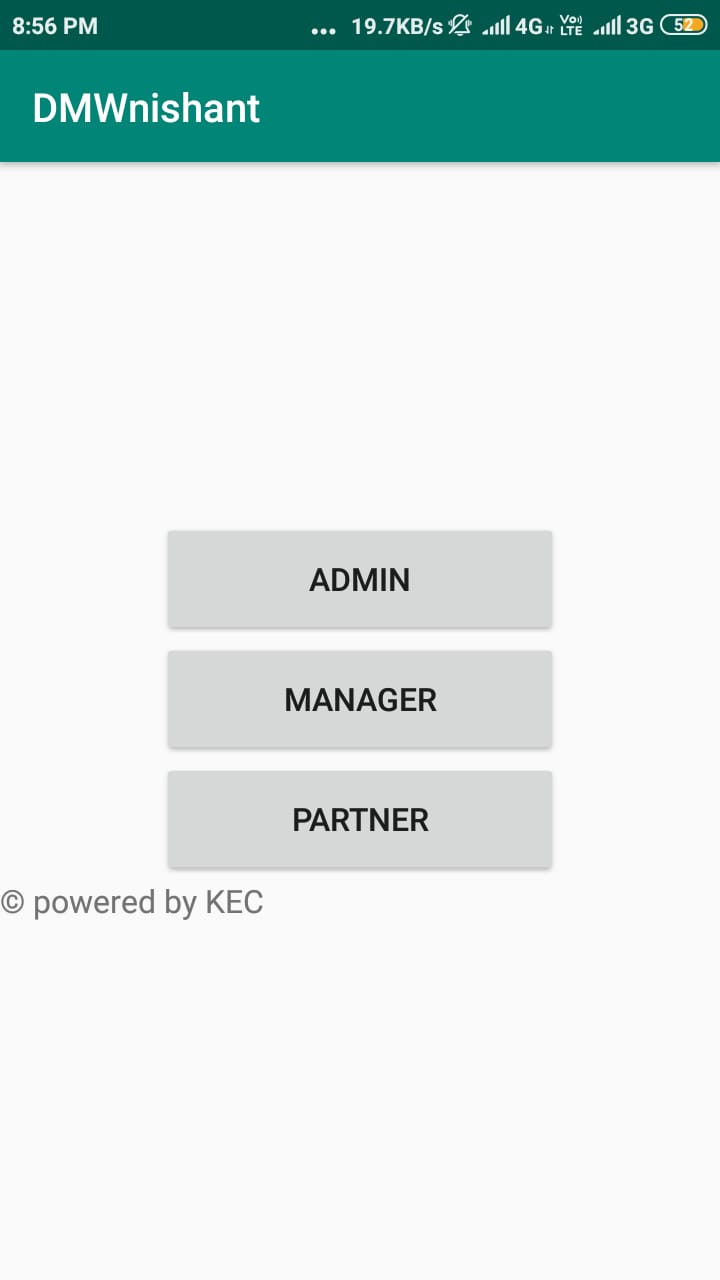
**4.4.1. production and defects entry page**



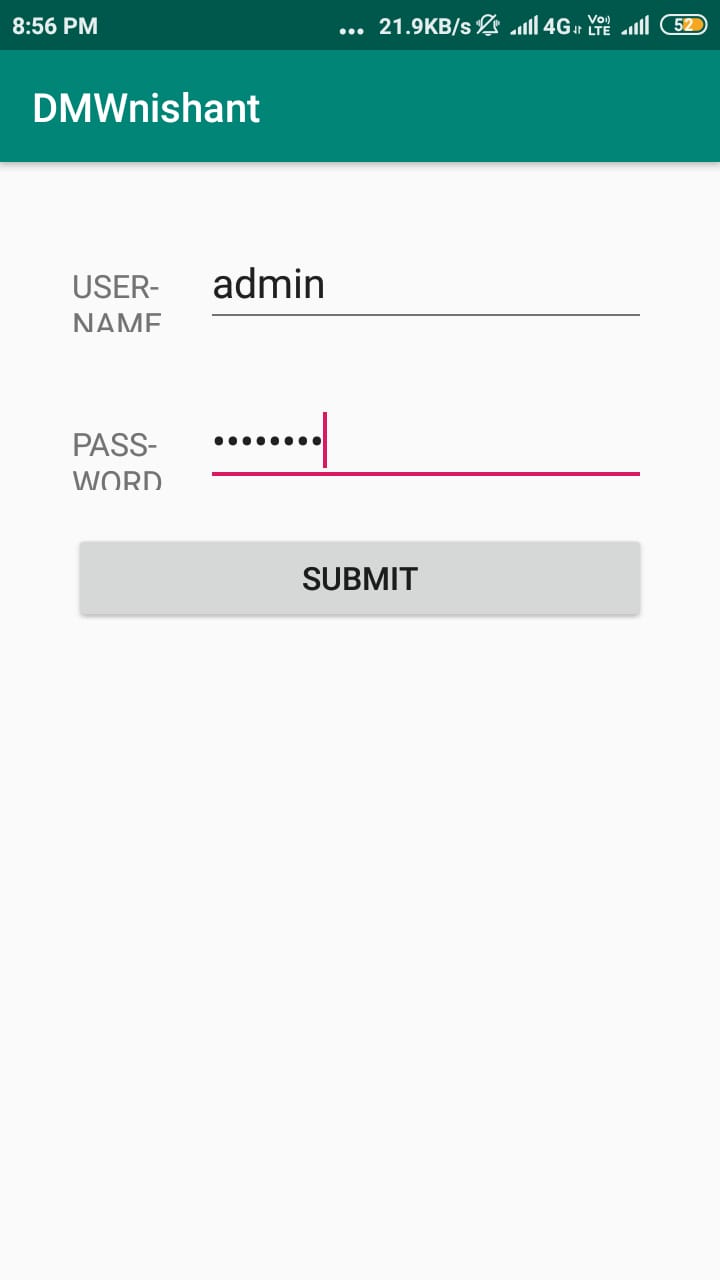
**4.4.2. ATTENDANCE AND PROFIT ANALYSIS PAGE**

****

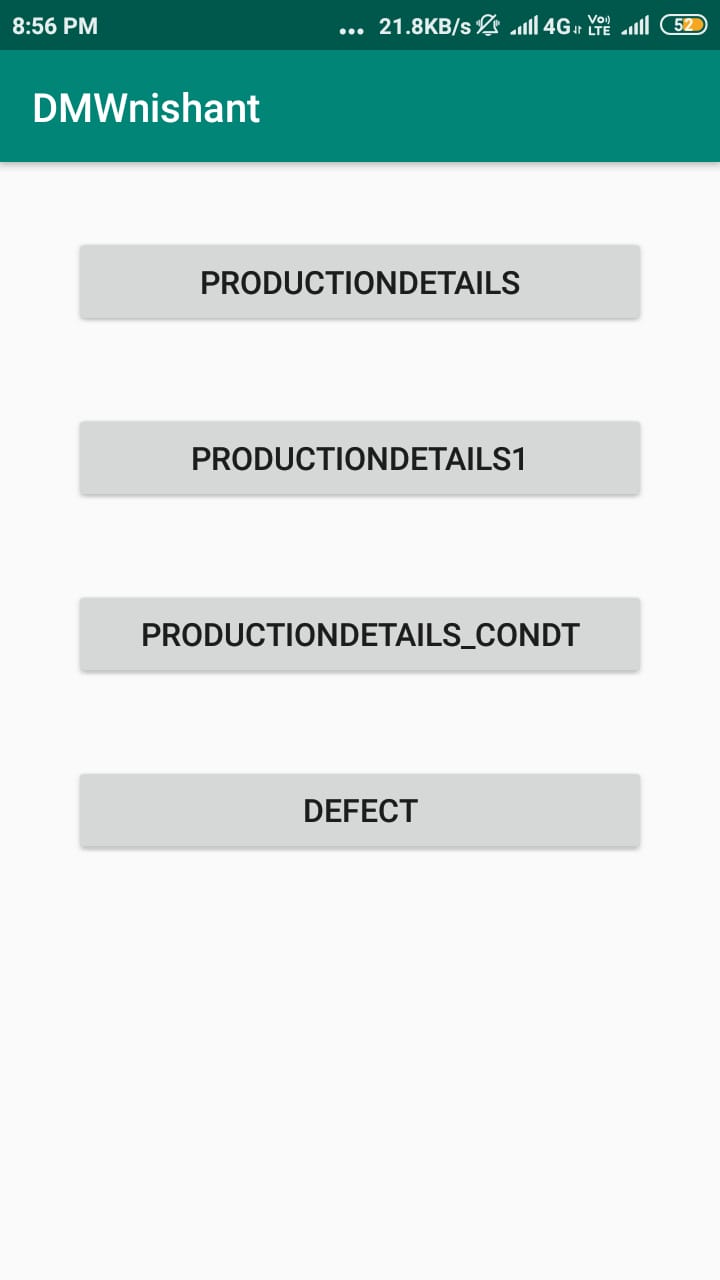
**4.4.3. MODULE PAGE**

****

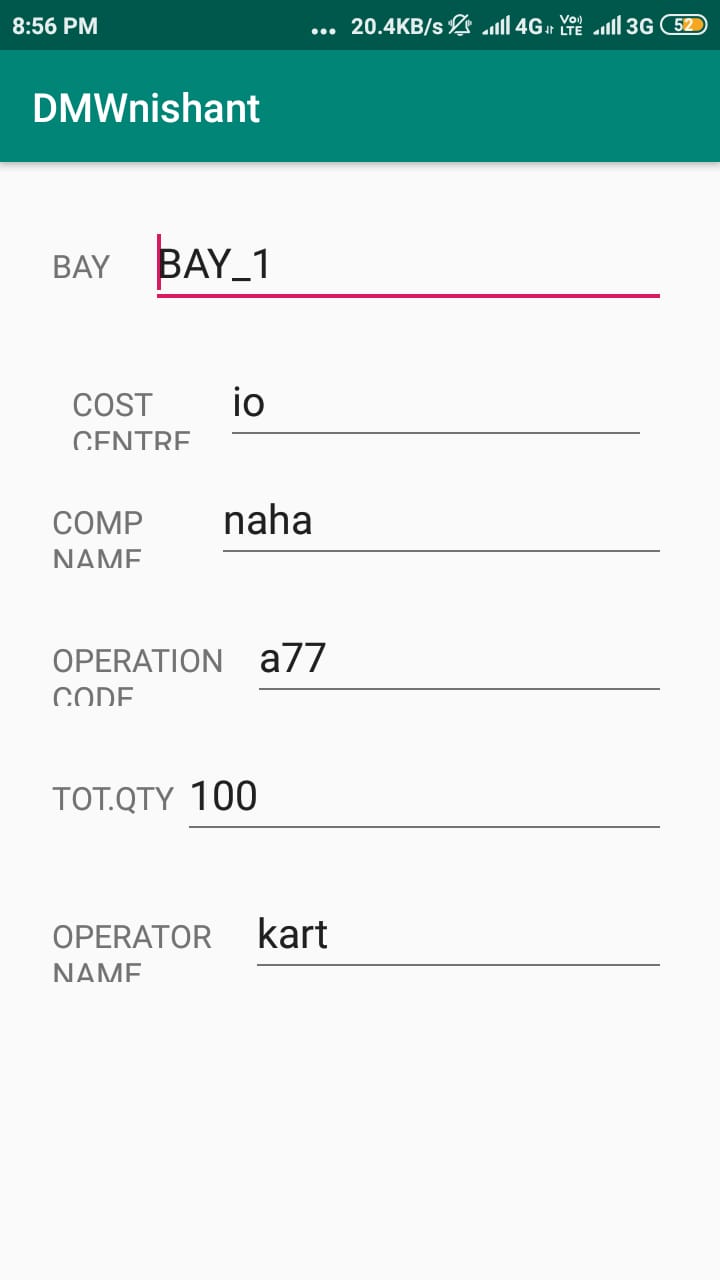
**4.4.4. ADMIN PAGE**

****

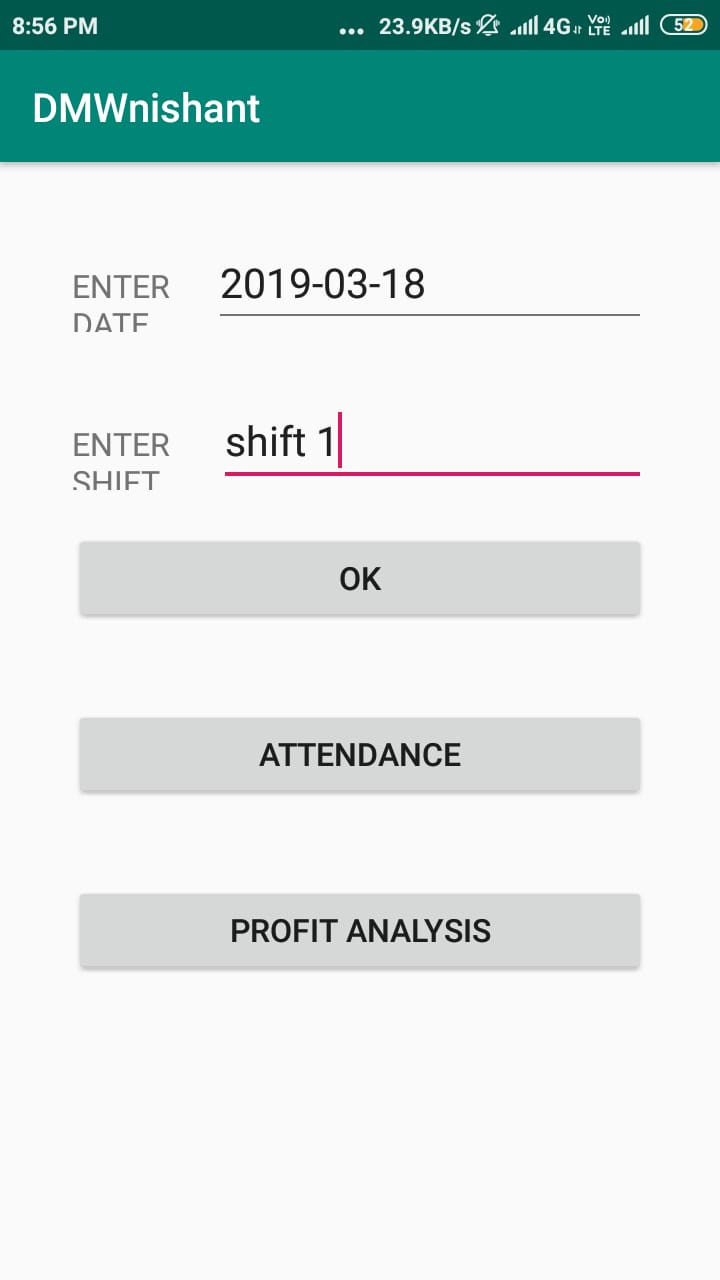
**4.4.5. PRODUCTION DETAILS PAGE**

****

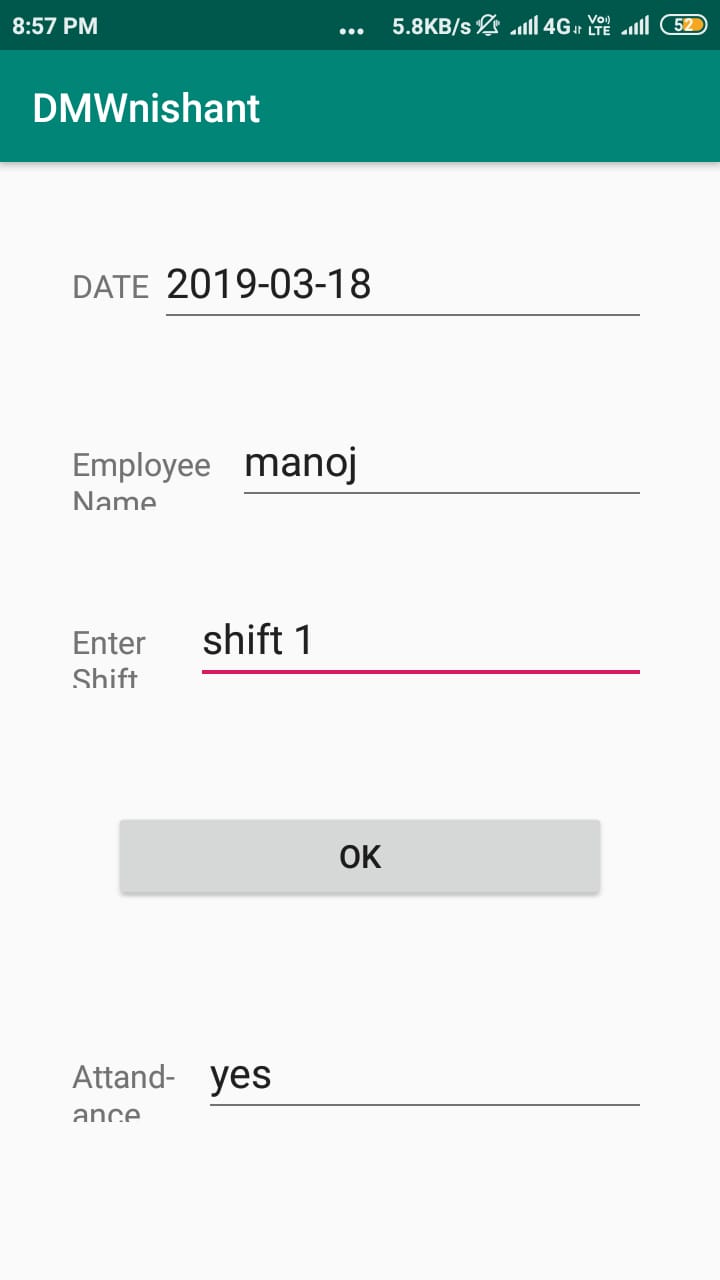
**4.4.6. ENTRY PAGE**

****

**4.4.7. DATE AND SHIFT PAGE**

****

**4.4.8. EMPLOYEE ATTENDANCE PAGE**

****

**CHAPTER 5**

**RESULTS AND DISCUSSION**

As a result, the application is highly secured and increased in performance, the system is user friendly, efficient, reliable and easily maintainable.

**TOOLS / TECHNIQUES**

**5.1 Firebase**

Firebase is a backend platform for building Web, Android and IOS applications. It offers real time database, different APIs, multiple authentication types and hosting platform. This is an introductory tutorial, which covers the basics of the Firebase platform and explains how to deal with its various components and sub-components.

**5.2 Node.js**

Node.js is a very powerful JavaScript-based platform built on Google Chrome's JavaScript V8 Engine. It is used to develop I/O intensive web applications like video streaming sites, single-page applications, and other web applications. Node.js is open source, completely free, and used by thousands of developers around the world.

**CHAPTER 6**

**TESTING**

**6.1 UNIT TESTING**

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation. Unit testing can be done manually but is often automated. In this process, each module (such as admin, user etc.) are considered and individual units are tested for proper operation. If each module meets up with the user’s requirement, then it is subjected to integration testing where more than one module are integrated and tested. In Servlet Framework Test Complete allows to run automated tests for desktop applications.

**6.2 REGRESSION TESTING**

This testing is carried out to ensure that the later developed code will not affect the features of the earlier developed module. For example, the booking invoice that is to be generated is carried out only after all the calculations are accurately processed. If any error may reflect in the calculations due to the generation, then the code must be changed accordingly. Each module must be tested to produce a satisfactory outcome.

**6.3 VERIFICATION**

Verification is a static practice of verifying documents, design, code and program. It includes all the activities associated with producing high quality software: inspection, design analysis and specification analysis. It is a relatively objective process. Verification will help to determine whether the software is of high quality, but it will not ensure that the system is useful. Verification is concerned with whether the system is well-engineered and error-free.

**6.4 VALIDATION**

The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements*.* Validation is the process of evaluating the final product to check whether the software meets the customer expectations and requirements. It is a dynamic mechanism of validating and testing the actual product.

**CHAPTER 7**

**CONCLUSION**

Android application DMW CNC PRIVATE LIMITED SOLUTION includes the features of managing the details of production ,details of employee attendance, profit and loss calculation made simple.It includes the facilities of knowing day-to-day updates of company from any where.It reduces the time constraints of the admin ,partner and manager. All the manual processes are done automatically by the application.

**FUTURE WORKS**

Research, as we know is a continuous process and this is not an exception. This has successfully been carried out and now in a better position to make necessary recommendation and suggestion. Below are some recommendation for the future research and addition to the project.

1. As the technology emerges, it is possible to upgrade the system.

2. Security can be improved using emerging technologies.

3. Additionally graphs can be added.

**APPENDICES**

**SOURCE CODE**

HTML CODE:

<html>

<head>

<style type="text/css">

.tg {border-collapse:collapse;border-spacing:0;}

.tg td{font-family:Arial, sans-serif;font-size:14px;padding:10px 12px;border-style:solid;border-width:3px;overflow:hidden;word-break:normal;border-color:black;}

.tg th{font-family:Arial, sans-serif;font-size:14px;font-weight:normal;padding:11px 5px;border-style:solid;border-width:1px;overflow:hidden;word-break:normal;border-color:black;}

.tg .tg-0pky{border-color:inherit;text-align:left;vertical-align:top}

body {

background-image: url("dmw logo.jpg");

background-position:center-left;

background-repeat: no-repeat, repeat;

background-color: #cccccc;

background-size: 5% 5%;

}

</style>

</head>

<body>

<h1><marquee>DMW CNC SOLUTION</marquee></h1>

<form method="post" action="enterData">

<table class="tg">

<tr>

<th class="tg-0pky" colspan="5"> PRODUCTION DETAILS</th>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class ="tg-0pky">Date</td><td><input type="date" name="date" required></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky" rowspan="9"></td>

<td class="tg-0pky">Breakdown</td><td><input type="number" name="" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Shift</td><td><label><select name="shift" required>

<option>shift 1</option>

<option>shift 2</option>

<option>shift 3</option>

</select></td><br>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">Loading&amp; Unloading</td><td><input type="number" name="loading\_unloading" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Bay</td><td><select name="bay">

<option>BAY\_1</option>

<option>BAY\_2</option>

<option>BAY\_3</option>

</select></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">Setting Change</td><td><input type="number" name="setting\_change" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Cost Centre</td><td><input type="text" name="cost\_centre" required></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">Machine Tool Change</td><td><input type="Number" name="machine\_tool\_change" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Component Name</td><td><input type="text" name="component\_name" required></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">Machine Fixture Change</td><td><input type="number" name="machine\_fixture\_change" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Operation Code</td><td><input type="text" name="operation\_code" required></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">Machine Coolant Nozzle Adjustment</td><td><input type="number" name="machine\_coolant\_nozzle\_adjustment" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Total Machined Qty. in Nos.</td><td><input type="number" name="total\_machined\_quantity\_in\_nos" required></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">Idling Loss (Lunch Time)</td><td><input type="number" name="idling\_loss" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Operator Name</td><td><input type="name" name="operator\_name" required></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">No Operator</td><td><input type="number" name="no\_of\_operator" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Supervisor Name</td><td><input type="text" name="supervisor\_name" required></td>

<td class="tg-0pky" colspan="2"></td>

<td class="tg-0pky">No Load</td><td><input type="number" name="no\_of\_load"></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky" colspan="7">DEFECTS AND ACCEPTED QUANTITY</td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Reasons</td>

<td class="tg-0pky">Shift</td>

<td class="tg-0pky">Component Id. Nos.</td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky">Inspection Time</td><td><input type="number" name="inspection\_time" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">M/D- Material Defect</td>

<td><input type="number" required></td>

<td><input type="number" required></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky">Power Shutdown</td><td><input type="number" name="power\_shutdown" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">M/C- Machining Defect</td>

<td> <input type="number" required></td>

<td><input type="number" required></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky">Others</td><td><input type="number" name="others" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">C/A- Conditionally Accepted</td>

<td><input type="number" name="enter" required></td>

<td><input type="number" name="enter\_shift\_for\_defect" required></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky">New Product Development- Mc.&nbsp;&nbsp;Run Time</td><td><input type="number" name="new\_product\_development" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Reworks</td>

<td><input type="number" name="enter"></td>

<td><input type="number" name="enter"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky">Remark</td><td><input type="text" name="remark" required></td>

</tr>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Accepted Qty. in Nos.</td>

<td class="tg-0pky">0</td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"colspan="1"></td>

<td class="tg-0pky">SHIFT TIME BALANCING REQUIRED</td><td><input type="number" name="shift\_timing\_balancing\_required" required></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Standard Cycle Time</td>

<td class="tg-0pky">0</td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"colspan="1"></td>

<td class="tg-0pky">Available capacity</td><td><input type="number" name="Available capacity"></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky">Machining Time in Min.</td>

<td class="tg-0pky">0</td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"colspan="1"></td>

<td class="tg-0pky">Machine Utilization</td><td><input type="number" name="Machine utilization"></td>

</tr>

<tr>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky"colspan="1"></td>

<td class="tg-0pky"></td>

<td class="tg-0pky">Quality Index</td><td><input type="number" name="Quality Index"></td>

</tr>

</table>

</td></tr></td></tr></td></tr></td></tr></td></tr></td></tr>

<br><br><br>

<center><input type="submit" value="next" style="width: 80px; height: 30px; color: "></center>

</form>

</body>

</html>

ATTENDANCE\_HTML

<html>

<head>

<style type="text/css">

.tg {border-collapse:collapse;border-spacing:0;}

.tg td{font-family:Arial, sans-serif;font-size:14px;padding:10px 12px;border-style:solid;border-width:3px;overflow:hidden;word-break:normal;border-color:black;}

.tg th{font-family:Arial, sans-serif;font-size:14px;font-weight:normal;padding:11px 5px;border-style:solid;border-width:1px;overflow:hidden;word-break:normal;border-color:black;}

.tg .tg-0pky{border-color:inherit;text-align:left;vertical-align:top}

body {

background-image: url("dmw logo.jpg");

background-position:center-left;

background-repeat: no-repeat, repeat;

background-color: #cccccc;

background-size: 5% 5%;

}

</style>

</head>

<body>

<h1><marquee>DMW CNC SOLUTION</marquee></h1>

<form method="post" action="/attendance">

<table class="tg" width="30%" align="LEFT" border="1px">

<tr>

<th class="tg-0pky" colspan="5"> ATTENDANCE DETAILS</th>

</tr>

<tr>

<td class ="tg-0pky">Date</td><td><input type="date" name="date" required></td>

</tr>

<tr>

<td class="tg-0pky">Shift</td><td><label><select name="shift" required>

<option>shift 1</option>

<option>shift 2</option>

<option>shift 3</option>

</select></td><br>

<tr>

<td class="tg-0lax">Emp name</td>

<td class="tg-0lax">Attendance</td>

</tr>

<tr>

<td class="tg-0lax">Arjun.S</td><td>Present<input type="radio" name="arjun" checked> Absent <input type="radio" name="arjun"></td>

</tr>

<tr>

<td class="tg-0lax">Aravind.p</td><td>Present<input type="radio" name="aravind" checked> Absent <input type="radio" name="aravind"></td>

</tr>

<tr>

<td class="tg-0lax">bala.S</td><td>Present<input type="radio" name="bala" checked> Absent <input type="radio" name="bala"></td>

</tr>

<tr>

<td class="tg-0lax">chandru.k</td><td>Present<input type="radio" name="chandru" checked> Absent <input type="radio" name="chandru"></td>

</tr>

<tr>

<td class="tg-0lax">deepak.R</td><td>Present<input type="radio" name="deepak" checked> Absent <input type="radio" name="deepak"></td>

</tr>

<tr>

<td class="tg-0lax">gokul.M</td><td>Present<input type="radio" name="gokul" checked> Absent <input type="radio" name="gokul"></td>

</tr>

<tr>

<td class="tg-0lax">kavin.E</td><td>Present<input type="radio" name="kavin" checked> Absent <input type="radio" name="kavin"></td>

</tr>

<tr>

<td class="tg-0lax">Manoj.K</td><td>Present<input type="radio" name="manoj" checked> Absent <input type="radio" name="manoj"></td>

</tr>

<tr>

<td class="tg-0lax">mani.E</td><td>Present<input type="radio" name="mani" checked> Absent <input type="radio" name="mani"></td>

</tr>

<tr>

<td class="tg-0lax">Nishan.E</td><td>Present<input type="radio" name="nishane" checked> Absent <input type="radio" name="nishane"></td>

</tr>

<tr>

<td class="tg-0lax">Nishan.A</td><td>Present<input type="radio" name="nishana" checked> Absent <input type="radio" name="nishana"></td>

</tr>

</table>

</tr></td></tr></td></tr></td></tr></td></tr></td></tr>

<br><br><br>

<table class="tg" width="50%" align="center" border="1px">

<tr>

<th class="tg-0pky" colspan="5">PROFIT ANALYSIS</th>

</tr>

<tr>

<td class ="tg-0pky">COST OF NO OF MACHINES PRODUCED</td><td><input type="number" name="profit" required></td>

<td class ="tg-0pky">COST OF NO OF MACHINES DEFECT</td><td><input type="number" name="loss" required></td>

</tr>

</table>

<br>

<br>

<center><input type="submit" value="OK" style="width: 100px; height: 30px; color: "></center>

</form>

</body>

</html>

**ANDROID CODE**

ADMIN

**package** com.example.dmwnishant;  
  
**import** android.animation.AnimatorListenerAdapter;  
**import** android.annotation.TargetApi;  
**import** android.content.Intent;  
**import** android.animation.Animator;  
**import** android.content.pm.PackageManager;  
**import** android.support.annotation.NonNull;  
**import** android.support.design.widget.Snackbar;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.app.LoaderManager.LoaderCallbacks;  
  
**import** android.content.CursorLoader;  
**import** android.content.Loader;  
**import** android.database.Cursor;  
**import** android.net.Uri;  
**import** android.os.AsyncTask;  
  
**import** android.os.Build;  
**import** android.os.Bundle;  
**import** android.provider.ContactsContract;  
**import** android.text.TextUtils;  
**import** android.view.KeyEvent;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.view.inputmethod.EditorInfo;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.AutoCompleteTextView;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**public class** adminn **extends** AppCompatActivity {  
 EditText **name**, **pass**;  
 Button **submit**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_adminn***);  
  
 **name**=(EditText)findViewById(R.id.***LabUserNameeditText***);  
 **pass**=(EditText)findViewById(R.id.***LabUserPassditText***);  
 **submit**=(Button)findViewById(R.id.***LabUserSubmit***);  
  
 **submit**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 **final** String n, p;  
 n = **name**.getText().toString();  
 p = **pass**.getText().toString();  
  
 **if**(n.equals(**"admin"**))  
 {  
 **if**(p.equals(**"iamadmin"**))  
 {  
 Intent intent=**new** Intent(getApplicationContext(),dmwdatass.**class**);  
 startActivity(intent);  
 }  
 **else** {  
 Intent intent=**new** Intent(getApplicationContext(),adminn.**class**);  
 Toast.*makeText*(getApplicationContext(),**"Password Wrong"**,Toast.***LENGTH\_LONG***).show();  
 startActivity(intent);  
 }  
 }  
 **else** {  
 Intent intent=**new** Intent(getApplicationContext(),adminn.**class**);  
 Toast.*makeText*(getApplicationContext(),**"Username Wrong"**,Toast.***LENGTH\_LONG***).show();  
 startActivity(intent);  
 }  
 }  
 });  
  
 }  
}

MAIN\_ACTIVITY

**package** com.example.dmwnishant;  
  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.support.annotation.NonNull;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.Button;  
**import** android.widget.Toast;  
  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.ValueEventListener;  
  
**public class** MainActivity **extends** AppCompatActivity {  
 Button **bt1**;  
 Button **bt2**;  
 Button **bt3**;  
 DatabaseReference **fref**=FirebaseDatabase.*getInstance*().getReference();  
 DatabaseReference **mfref**=**fref**.child(**"dmw-1-6ef84"**).child(**"condition"**);  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **bt1** = (Button) findViewById(R.id.***bt\_admin***);  
 **bt2** = (Button) findViewById(R.id.***bt\_manager***);  
 **bt3** = (Button) findViewById(R.id.***bt\_partner***);  
 **mfref**.addValueEventListener(**new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(@NonNull DataSnapshot dataSnapshot) {  
 String value = dataSnapshot.getValue(String.**class**);  
 **bt1**.setText(value);  
 Toast.*makeText*(getApplicationContext(),value,Toast.***LENGTH\_SHORT***).show();  
 }  
  
 @Override  
 **public void** onCancelled(@NonNull DatabaseError databaseError) {  
  
 }  
 });  
 *//DatabaseReference myRef = fref.getReference().child("condition");* Toast.*makeText*(getApplicationContext(),**"test"**,Toast.***LENGTH\_SHORT***).show();  
  
  
  
  
 *// Capture button clicks* **bt1**.setOnClickListener(**new** OnClickListener() {  
 **public void** onClick(View arg0) {  
  
 *// Start NewActivity.class* Intent myIntent = **new** Intent(MainActivity.**this**,  
 adminn.**class**);  
 startActivity(myIntent);  
 }  
 });  
 **bt2**.setOnClickListener(**new** OnClickListener() {  
 **public void** onClick(View arg0) {  
  
 *// Start NewActivity.class* Intent myIntent = **new** Intent(MainActivity.**this**,  
 managerr.**class**);  
 startActivity(myIntent);  
 }  
 });  
 **bt3**.setOnClickListener(**new** OnClickListener() {  
 **public void** onClick(View arg0) {  
  
 *// Start NewActivity.class* Intent myIntent = **new** Intent(MainActivity.**this**,  
 partnerr.**class**);  
 startActivity(myIntent);  
 }  
 });  
 }  
  
 @Override  
 **protected void** onStart() {  
 **super**.onStart();  
 **mfref**.addValueEventListener(**new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
 *// This method is called once with the initial value and again  
 // whenever data at this location is updated.* String value = dataSnapshot.getValue(String.**class**);  
 **bt1**.setText(value);  
 Toast.*makeText*(getApplicationContext(),value,Toast.***LENGTH\_SHORT***).show();  
 *// Log.d(TAG, "Value is: " + value);* }  
  
 @Override  
 **public void** onCancelled(DatabaseError error) {  
 *// Failed to read value  
 //Log.w(TAG, "Failed to read value.", error.toException());* Toast.*makeText*(getApplicationContext(),**"Error"**,Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 }  
}