IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

IMPLEMENTATION OF SMART GENETICS FOR SMARTER HEALTH

[1]Khan Shoeb Nasir, [2]Dr. V.M. Deshmukh

[1]Final year M.E, [2]Department of Computer Science and Engineering, PRMIT&R
Computer Science and Engineering,
[1]Student, Amravati, India

Abstract: The most important part of a city is the people who live in it. Therefore, wellness and health are indispensable for the Smart Cities concept. In this paper, we introduce an approach that combines genetic research with environmental monitoring in real time by using an integrated system of IT and IOT that helps to understand better and prevent diseases, improving the quality of life of the citizens. We are working to develop a virtual platform that creates and establishes correlations between the genetic information of the user and their environment. The principal advantages of this project are to allow health care providers, governments and citizens to make smarter decisions oriented to health care, implementing Health Information Technologies (HIT) and improving the concept of Mobile Health (mHealth), all of these based on the Quadruple Helix Innovation model.

Keywords—genetics, HIT, smart health, predictive modelling, smart cities

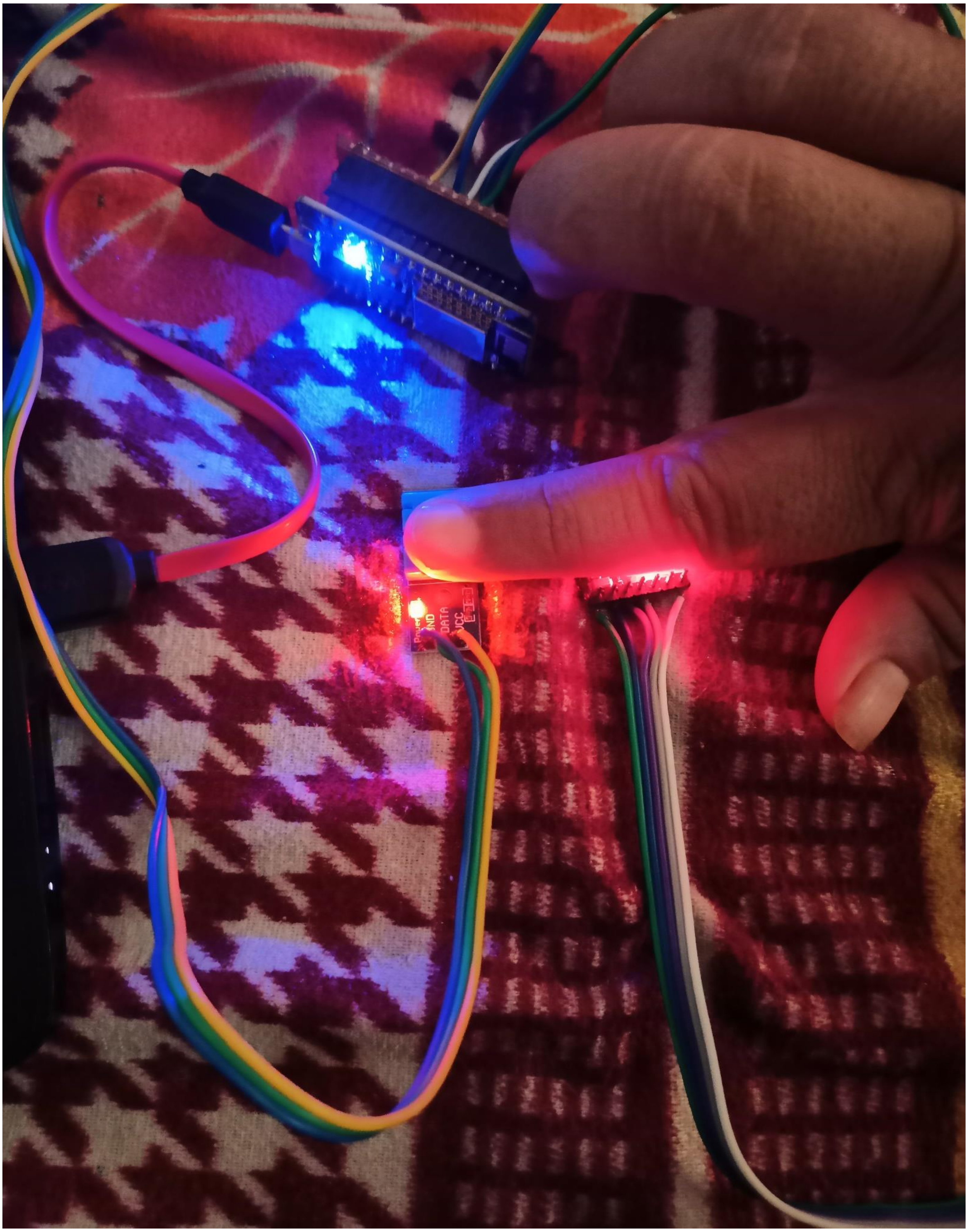
1. Introduction

There are several activities and approaches being applied to help reduce the reproduction rate of COVID-19. These include self-isolation methods such as working from home, improved basic hygiene such as increased hand washing and the deployment of personal protective equipment (PPE) to reduce the prospect of infection.

Smart and connected health care is of specific significance in the spectrum of applications enabled the Internet of Things (IoT). Networked sensors, either embedded inside our living system or worn on the body, enable to gather rich information regarding our physical and mental health. In specific, the accessibility of information at previously unimagined scales and spatial longitudes combined with the new generation of smart processing algorithms can expedite an advancement in the medical field, from the current post-facto diagnosis and treatment of reactive framework, to an early-stage proactive paradigm for disease prognosis combined with prevention and cure as well as overall administration of well-being rather than ailment. This paper sheds some light on the current methods accessible in the Internet of Things (IoT) domain for healthcare applications. The proposed objective is to design and create a healthcare system centred on Mobile-IoT by collecting patient information from different sensors and alerting both the guardian and the doctor by sending emails and SMS in a timely manner. It remotely monitors the physiological parameters of the patient and diagnoses the illnesses swiftly.

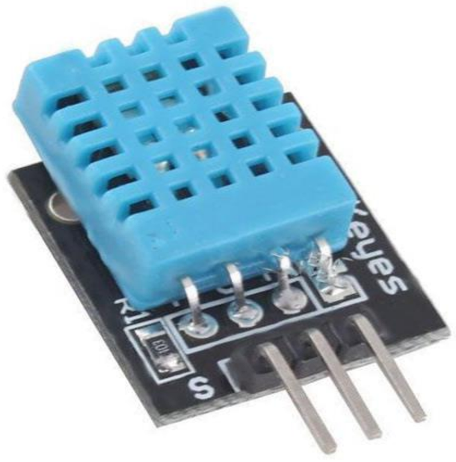
As there are the frequent contact comes with patient in corona ward there might be chances to be corona to health worker so that were proposed a system which will work on the evaluation of such system which automatically whole data from sensor connected to body of patient and monitor if any up comes or risk occur it will show the alert to doctor.

2. Implementation



Hardware requirements

1. DHT11



Pin Identification and Configuration:

No:	**Pin Name**	**Description**
For DHT11 Sensor		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	NC	No Connection and hence not used
4	Ground	Connected to the ground of the circuit
For DHT11 Sensor module		
1	Vcc	Power supply 3.5V to 5.5V
2	Data	Outputs both Temperature and Humidity through serial Data
3	Ground	Connected to the ground of the circuit

DHT11 Specifications:

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%

The **DHT11** is a commonly used **Temperature and humidity sensor.** The sensor comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data. The sensor is also factory calibrated and hence easy to interface with other microcontrollers.The sensor can measure temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of ±1°C and ±1%. So if you are looking to measure in this range then this sensor might be the right choice for you.

2. Pulse Oximeter

The sensor is integrated **pulse oximetry** and **heart-rate monitor** sensor solution. It combines two **LED's, a photodetector, optimized optics**, and low-noise analog signal processing to detect pulse and heart-rate signals. It operates from **1.8V** and **3.3V** power supplies and can be powered down through software with negligible standby current, permitting the power supply to remain connected at all times.

Features of MAX30100 Pulse Oximeter

1. Consumes very low power (operates from 1.8V and 3.3V)
2. Ultra-Low Shutdown Current (0.7µA, typ)
3. Fast Data Output Capability

Working of MAX30100 Pulse Oximeter and Heart-Rate Sensor

The device has two **LEDs, one emitting red light, another emitting infrared light**. For pulse rate, only the **infrared light** is needed. Both the **red light** and **infrared light** is used to measure oxygen levels in the blood.

When the heart pumps blood, there is an increase in **oxygenated blood** as a result of having more blood. As the heart relaxes, the volume of oxygenated blood also decreases. By knowing the time between the increase and decrease of oxygenated blood, the **pulse rate** is determined.

It turns out, **oxygenated blood** absorbs more infrared light and passes more red light while **deoxygenated blood** absorbs red light and passes more infrared light. This is the main function of the **MAX30100**: it reads the absorption levels for both light sources and stored them in a buffer that can be read via I2C.

3. NodeMCU

NodeMCU is an open source Lua based firmware for the ESP8266 WiFi SOC from Espressif and uses an on-module flash-based SPIFFS file system. NodeMCU is implemented in C and is layered on the Espressif NON-OS SDK.The firmware was initially developed as is a companion project to the popular ESP8266-based NodeMCU development modules, but the project is now community-supported, and the firmware can now be run on *any* ESP module. NodeMCU is an open source firmware for which open source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit).[8] The term "NodeMCU" strictly speaking refers to the firmware rather than the associated development kits.

4. RAM of 8 GB

5. i3 processor

6. 500 GB of hard disk

Software requirements

1. Android Studio

1Android Studio is the official Integrated Development Environment (IDE) for android application development. Android Studio provides more features that enhance our productivity while building Android apps.

Android Studio was announced on 16th May 2013 at the Google I/O conference as an official IDE for Android app development. It started its early access preview from version 0.1 in May 2013. The first stable built version was released in December 2014, starts from version 1.0.

Since 7th May 2019, Kotlin is Google's preferred language for Android application development. Besides this, other programming languages are supported by Android Studio.

Features of Android Studio

- It has a flexible Gradle-based build system.
- It has a fast and feature-rich emulator for app testing.
- Android Studio has a consolidated environment where we can develop for all Android devices.
- Apply changes to the resource code of our running app without restarting the app.
- Android Studio provides extensive testing tools and frameworks.
- It supports C++ and NDK.
- It provides build-in supports for Google Cloud Platform. It makes it easy to integrate Google Cloud Messaging and App Engine.

2. JDK 8.0

The **Java Development Kit** (**JDK**) is an implementation of either one of the Java Platform, Standard Edition, Java Platform, Enterprise Edition, or Java Platform, Micro Edition platforms released by Oracle Corporation in the form of a binary product aimed at Java developers
on Solaris, Linux, macOS or Windows. The JDK includes a private JVM and a few other resources to finish the development of a Java application. Since the introduction of the Java platform, it has been by far the most widely used Software Development Kit (SDK).The JDK is available for 64-bit x64 macOS (and that version also works with Rosetta 2), while an early access build (developer preview) from Microsoft is also available to support recent Apple M1 Macs.The JDK has as its primary components a collection of programming tools, including:

- appletviewer – this tool can be used to run and debug Java applets without a web browser
- apt – the annotation-processing tool[6]
- extcheck – a utility that detects JAR file conflicts
- idlj – the IDL-to-Java compiler. This utility generates Java bindings from a given Java IDL file.
- jabswitch – the Java Access Bridge. Exposes assistive technologies on Microsoft Windows systems.
- java – the loader for Java applications. This tool is an interpreter and can interpret the class files generated by the javac compiler. Now a single launcher is used for both development and deployment. The old deployment launcher, jre, no longer comes with Sun JDK, and instead it has been replaced by this new java loader.
- javac – the Java compiler, which converts source code into Java bytecode
- javadoc – the documentation generator, which automatically generates documentation from source code comments
- jar – the archiver, which packages related class libraries into a single JAR file. This tool also helps manage JAR files.
- javafxpackager – tool to package and sign JavaFX applications
- jarsigner – the jar signing and verification tool

- javah – the C header and stub generator, used to write native methods
- javap – the class file disassembler
- javaws – the Java Web Start launcher for JNLP applications
- JConsole – Java Monitoring and Management Console
- jdb – the debugger
- jhat – Java Heap Analysis Tool (experimental)
- jinfo – This utility gets configuration information from a running Java process or crash dump. (experimental)
- jmap Oracle jmap - Memory Map– This utility outputs the memory map for Java and can print shared object memory maps or heap memory details of a given process or core dump. (experimental)
- jmc – Java Mission Control
- jpackage – a tool for generating self-contained application bundles. (experimental)
- jps – Java Virtual Machine Process Status Tool lists the instrumented HotSpot Java Virtual Machines (JVMs) on the target system. (experimental)
- jrunscript – Java command-line script shell.
- jshell - The new jshell introduced in java 9.
- jstack – utility that prints Java stack traces of Java threads (experimental)
- jstat – Java Virtual Machine statistics monitoring tool (experimental)
- jstatd – jstat daemon (experimental)
- keytool – tool for manipulating the keystore
- pack200 – JAR compression tool
- policytool – the policy creation and management tool, which can determine policy for a Java runtime, specifying which permissions are available for code from various sources.
- VisualVM – visual tool integrating several command-line JDK tools and lightweight[*clarification needed*] performance and memory profiling capabilities
- wsimport – generates portable JAX-WS artifacts for invoking a web service.
- xjc – Part of the Java API for XML Binding (JAXB) API. It accepts an XML schema and generates Java classes.

Advantages

- Proposed system will help to monitor the patients from anywhere anytime
- Intelligence system will help to monitor the complete health of the patient using single device
- Doctors can able to control and maintain the track of every patient on single click
- Serious patient can be identified easily and send the alert easily
- Help to track to track the patient in corona pandemic

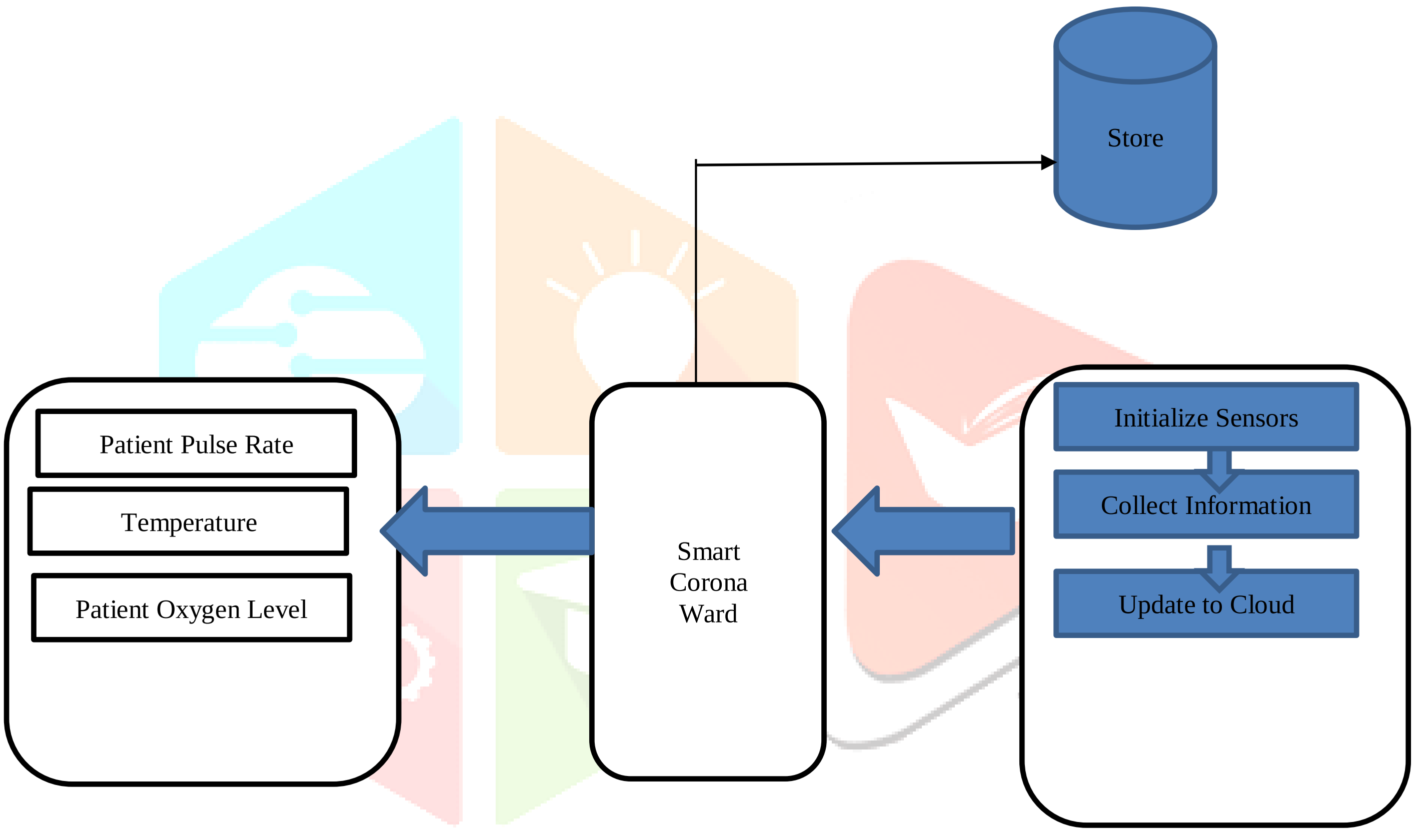
Disadvantages

- The proposed system always needs internet connectivity if the connectivity loss, then system will be unable to send the data

4. System Design

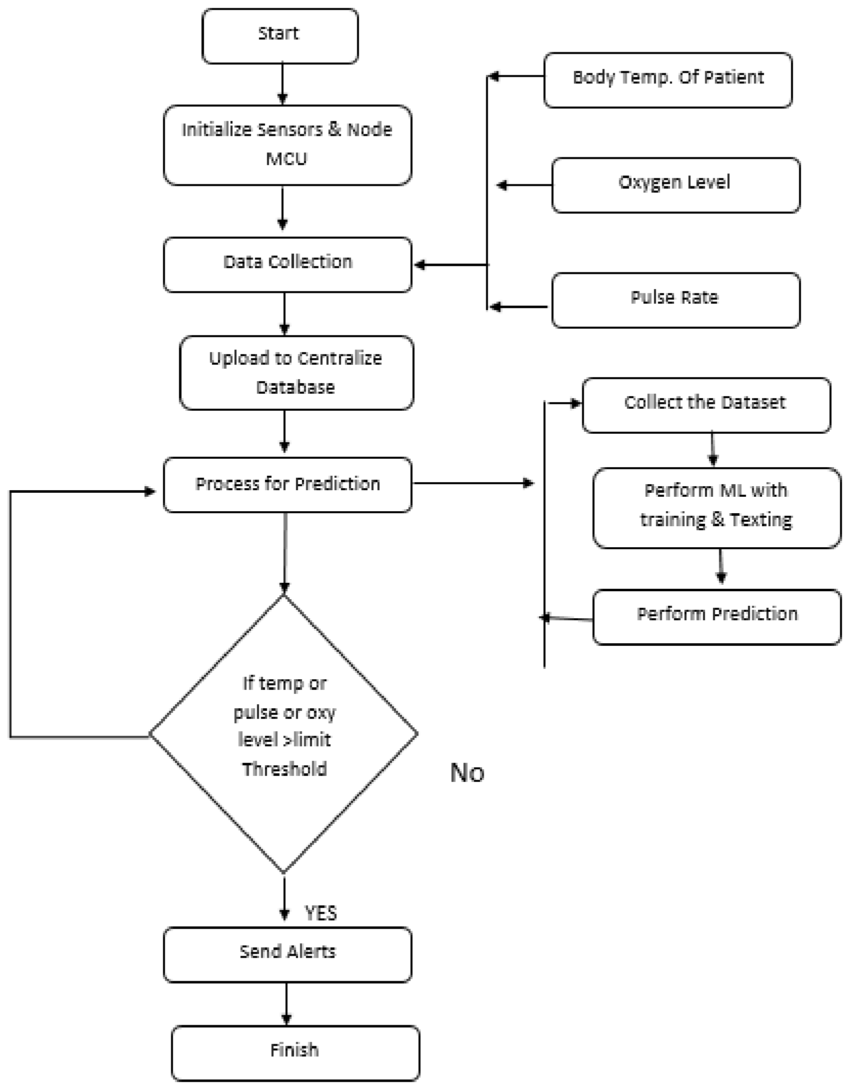
4.1 Proposed system architecture

Block diagram



4.2 Proposed system design

Flowchart



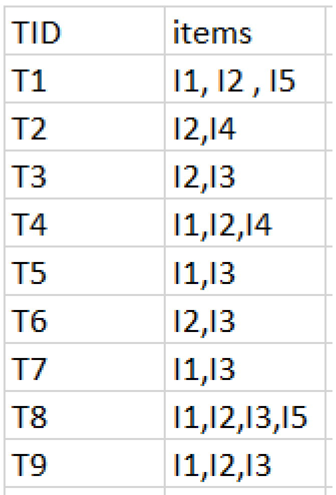
4.3 Proposed Algorithm

Apriori algorithm is given by R. Agrawal and R. Srikant in 1994 for finding frequent itemsets in a dataset for Boolean association rule. Name of the algorithm is Apriori because it uses prior knowledge of frequent itemset properties. We apply an iterative approach or level-wise search where k-frequent itemsets are used to find k+1 itemsets.

To improve the efficiency of level-wise generation of frequent itemsets, an important property is used called *Apriori property* which helps by reducing the search space.

Apriori Property –

All non-empty subset of frequent itemset must be frequent. The key concept of Apriori algorithm is its anti-monotonicity of support measure.

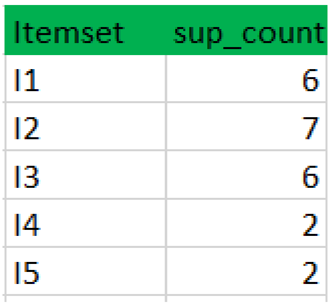


TID	items
T1	I1, I2 , I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

minimum support count is 2
minimum confidence is 60%

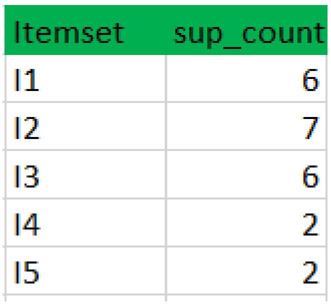
Step-1: K=1

(I) Create a table containing support count of each item present in dataset – Called **C1(candidate set)**



Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

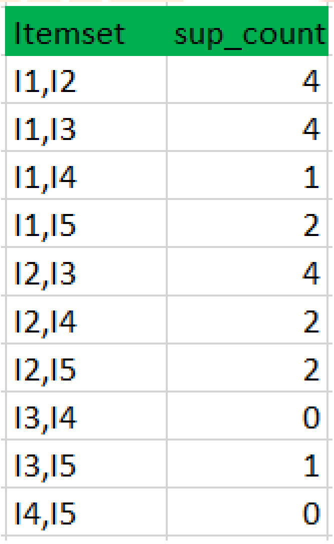
(II) compare candidate set item's support count with minimum support count(here min_support=2 if support_count of candidate set items is less than min_support then remove those items). This gives us itemset L1.



Itemset	sup_count
I1	6
I2	7
I3	6
I4	2
I5	2

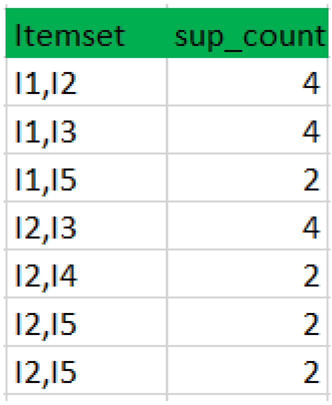
Step-2: K=2

- Generate candidate set C2 using L1 (this is called join step). Condition of joining L_{k-1} and L_{k-1} is that it should have (K-2) elements in common.
- Check all subsets of an itemset are frequent or not and if not frequent remove that itemset. (Example subset of {I1, I2} are {I1}, {I2} they are frequently checked for each itemset)
- Now find support count of these item sets by searching in dataset.



Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

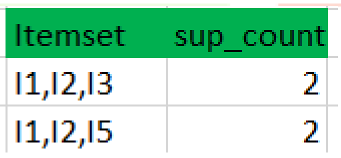
(II) compare candidate (C2) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L2.



Itemset	sup_count
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I2,I5	2

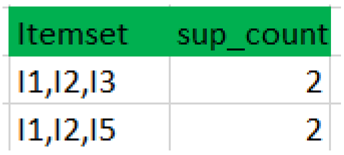
Step-3:

- Generate candidate set C3 using L2 (join step). Condition of joining L_{k-1} and L_{k-1} is that it should have (K-2) elements in common. So here, for L2, first element should match. So itemset generated by joining L2 is {I1, I2, I3}{I1, I2, I5}{I1, I3, i5}{I2, I3, I4}{I2, I4, I5}{I2, I3, I5}
- Check if all subsets of these itemsets are frequent or not and if not, then remove that itemset.(Here subset of {I1, I2, I3} are {I1, I2},{I2, I3},{I1, I3} which are frequent. For {I2, I3, I4}, subset {I3, I4} is not frequent so remove it. Similarly check for every itemset)
- find support count of these remaining itemset by searching in dataset.



Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

(II) Compare candidate (C3) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.



Itemset	sup_count
I1,I2,I3	2
I1,I2,I5	2

Step-4:

- Generate candidate set C4 using L3 (join step). Condition of joining L_{k-1} and L_{k-1} (K=4) is that, they should have (K-2) elements in common. So here, for L3, first 2 elements (items) should match.

- Check all subsets of these itemsets are frequent or not (Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contains {I1, I3, I5}, which is not frequent). So no itemset in C4
- We stop here because no frequent itemsets are found further

Thus, we have discovered all the frequent item-sets. Now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.

Confidence –
A confidence of 60% means that 60% of the customers, who purchased milk and bread also bought butter.

Confidence(A->B)=Support_count(A∪B)/Support_count(A)

So here, by taking an example of any frequent itemset, we will show the rule generation.
Itemset {I1, I2, I3} //from L3
SO rules can be
[I1^I2]=>[I3] //confidence = sup(I1^I2^I3)/sup(I1^I2) = 2/4*100=50%
[I1^I3]=>[I2] //confidence = sup(I1^I2^I3)/sup(I1^I3) = 2/4*100=50%
[I2^I3]=>[I1] //confidence = sup(I1^I2^I3)/sup(I2^I3) = 2/4*100=50%
[I1]=>[I2^I3] //confidence = sup(I1^I2^I3)/sup(I1) = 2/6*100=33%
[I2]=>[I1^I3] //confidence = sup(I1^I2^I3)/sup(I2) = 2/7*100=28%
[I3]=>[I1^I2] //confidence = sup(I1^I2^I3)/sup(I3) = 2/6*100=33%

So if minimum confidence is 50%, then first 3 rules can be considered as strong association rules.

5. Conclusion

5.1 References

1. Marcos-Pablos, S.; García-Peñalvo, F.J. Technological ecosystems in care and assistance: A systematic
literature review. Sensors 2019, 19, 708. [CrossRef]

2. Mutlag, A.A.; Ghani, M.K.A.; Arunkumar, N.A.; Mohamed, M.A.; Mohd, O. Enabling technologies for fog
computing in healthcare IoT systems. Future Gener. Comput. Syst. 2019, 90, 62–78. [CrossRef]

3. Skarlatidou, A.; Hamilton, A.; Vitos, M.; Haklay, M. What do volunteers want from citizen science technologies?
A systematic literature review and best practice guidelines. JCOM J. Sci. Commun. 2019, 18, A02. [CrossRef]

4. Grossi, G.; Lanzarotti, R.; Napoletano, P.; Noceti, N.; Odone, F. Positive technology for elderly well-being:
A review. Pattern Recognit. Lett. 2019, in press. [CrossRef]

5. Queirós, A.; Alvarelhão, J.; Cerqueira, M.; Silva, A.; Santos, M.; Rocha, N.P. Remote care technology:
A systematic review of reviews and meta-analyses. Technologies 2018, 6, 22. [CrossRef]

6. Chauhan, S.; Agarwal, N.; Kar, A.K. Addressing big data challenges in smart cities: A systematic literature
review. Info 2016, 18, 73–90. [CrossRef]

7. Irshad, M. A Systematic Review of Information Security Frameworks in the Internet of Things (iot). In Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on
Data Science and Systems (HPCC/SmartCity/DSS), Sydney, Australia, 12–14 December 2016.

8. Purnomo, F.; Prabowo, H. Smart city indicators: A systematic literature review. J. Telecommun. Electron.
Comput. Eng. (JTEC) 2016, 8, 161–164.

9. Moher, D.; Liberati, A.; Tetzlaff, J.; Altman, D.G. Preferred reporting items for systematic reviews and
meta-analyses: The PRISMA statement. Ann. Intern. Med. 2009, 151, 264–269. [CrossRef]

10. Ghapanchi, A.; Aurum, A. Antecedents to IT personnel's intentions to leave: A systematic literature review.
J. Syst. Softw. 2011, 84, 238–249. [CrossRef]

11. Sánchez Bernabeu, J.M.; Berna-Martinez, J.V.; Maciá Pérez, F. Smart sentinel: Monitoring and prevention

system in the smart cities. Int. Rev. Comput. Softw. (IRECOS) 2014, 9, 1554–1559. [CrossRef]

12. O'Neill, D.; Peoples, C. A Web-Based Portal for Assessing Citizen Well-Being. IT Prof. 2017, 19, 24–30.
[CrossRef]

13. Zschippig, C.; Kluss, T. Gardening in ambient assisted living. Urban For. Urban Green. 2016, 15, 186–189.
[CrossRef]

14. Casino, F.; Patsakis, C.; Batista, E.; Borràs, F.; Martínez-Ballesté, A. Healthy routes in the smart city:
A context-aware mobile recommender. IEEE Softw. 2017, 34, 42–47. [CrossRef]

15. Trencher, G.; Karvonen, A. Stretching "smart": Advancing health and well-being through the smart city
agenda. Local Environ. 2017, 24, 610–627. [CrossRef]

16. Gilart-Iglesias, V.; Mora, H.; Pérez-delHoyo, R.; García-Mayor, C. A computational method based on radio
frequency technologies for the analysis of accessibility of disabled people in sustainable cities. Sustainability
2015, 7, 14935–14963. [CrossRef]

17. Samani, H.; Zhu, R. Robotic automated external defibrillator ambulance for emergency medical service in
smart cities. IEEE Access 2016, 4, 268–283. [CrossRef]

18. Azimi, S.; Delavar, M.R.; Rajabifard, A. Multi-Agent Simulation of Allocating and Routing Ambulances
under Condition of Street Blockage after Natural Disaster. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.
2017, 42, 325–332. [CrossRef]

19. Palmieri, F.; Ficco, M.; Pardi, S.; Castiglione, A. A cloud-based architecture for emergency management and
first responders localization in smart city environments. Comput. Electr. Eng. 2016, 56, 810–830. [CrossRef]

20. Wray, A.; Olstad, D.L.; Minaker, L.M. Smart prevention: A new approach to primary and secondary cancer
prevention in smart and connected communities. Cities 2018, 79, 53–69. [CrossRef]

21. Thacker, S.B.; Berkelman, R.L. Public health surveillance in the United States. Epidemiol. Rev. 1988, 10, 164–190.
[CrossRef]

22. Patsakis, C.; Clear, M.; Laird, P.; Zigomitros, A.; Bouroche, M. Privacy-aware Large-Scale Virologic and

Epidemiological Data Monitoring. In Proceedings of the 2014 IEEE 27th International Symposium on Computer-Based Medical Systems, New York, NY, USA, 27–29 May 2014.

23. Shikhar, A.; Naveen, J.S.; Sowmya, B.J.; Srinivas, K.G. Data Analytics on Accident Data for Smarter Cities

and Safer Lives. In Proceedings of the 2016 IEEE International Conference on Computation.